Location-Based Augmented Reality with Pervasive Smartphone Sensors: Inside and Beyond Pokemon Go!

Ryan Shea*, *Member, IEEE,* Silvery Fu*, *Student Member, IEEE,* Andy Sun*, *Student Member, IEEE,* Chao Cai†, *Student Member, IEEE,* Xiaoqiang Ma†, *Member, IEEE,* Xiaoyi Fan*, *Student Member, IEEE,* Wei Gong*, *Member, IEEE,* and Jiangchuan Liu*, *Fellow, IEEE,*

***Abstract*—Combining powerful sensors and near ubiquitous distribution the smartphone has become an irreplaceable part of modern day life. Using its pervasive sensing capabilities, the smart-phone guides us to our destination with precise step-by-step directions, advises us on what to have for lunch, and improves our photography skills through stabilizing our camera. Using the popular Augmented Reality (AR) smartphone app Pokemon Go as a case study, we explore the world of pervasive sensing. In this paper we show both the current state of the art that enable applications such as Pokemon GO to thrive, as well as the limitations and opportunities inherent in current pervasive sensing applications.**

***Index Terms*—Pervasive Sensors, Cloud computing, Augmented Reality, Video Streaming, Quality of Experience**

I. Introduction

A new phenomenon is being noticed in public places around the world. The sight of someone following step-by-step directions to their destination, checking restaurant reviews, or posting live videos of them themselves to social media has become a nearly universal experience. All of these things were a near impossibility for the average person without expensive and specialized equipment less than a decade ago, but are now as simple as downloading an app. The smartphone has become a nearly indispensable part of life for many around the world, and recent studies have placed smartphone ownership rates of over 68% in the world's "advanced economies" and 37% in the "emerging world"[1].

The meteoric rise of this once luxury device to a ubiquitous computing and communication platform has been astounding and has opened up a plethora of new applications and services. Modern smartphone devices are being built not only with powerful processing abilities such as multi-core CPUs and dedicated GPUs but also a vast array of sensors for detecting their environments. Sensors including global positioning system (GPS), compasses, accelerometers, and gyro-scopes have pushed the functionality of the devices in many new and innovative directions. In this paper we explore both the current success of these pervasive sensing devices as well as many hot topics being explored in this domain. Through this paper we will explore the real world deployment of an application that relies heavily on these advanced sensors, namely the popular app Pokemon GO.

Combining augmented reality, edge computing, ubiquitous smartphone usage, and location based massively multi-player features, Pokemon GO (PKG) exploded onto smartphones in the summer of 2016. It is estimated that at the peak of the craze PKG was installed by over 10% of smart phone users in the USA[2]. While one can not disregard the marketing and popularity of the Pokemon franchise when discussing its near meteoric rise to popularity, under the hood, it is many technological advances, including the pervasive sensors on mobile devices, that are brought together to make PKG a success.

In this article, we consider PKG as a representative AR application and discuss how to enhance their gaming experience through novel services enabled by pervasive sensors. However, adding these sensor-based services is coupled with other issues such as increased processing workloads and power consumption in the mobile devices. We jointly consider these issues in a proposed cloud based offloading framework.

II. Exploring Pokemon Go

A. Pokemon Go Overview

Pokemon Go is one of few games that are truly mobile - it forces the player to physically roam an area by utilizing a mobile device's capabilities. This new genre of gaming creates architectural challenges and design choices that must be carefully considered. As the player's avatar is now attached to a physical location, strong game server scalability becomes a necessity due to the high clustering seen in human population density. Game client optimization also becomes a high priority, as the mobile device's sensors are constantly on and causing drain on the battery. With these in mind, we begin our analysis by examining the architectural model of PKG. Pokemon Go can trace its origin back to the Google Maps Pokemon Challenge. In 2014, Google announced a new job position of "Pokemon Master" and required applicants to capture all 721 Pokemon before being offered the role[3]. The challenge drew in an enormous positive response, and set in motion a series

Simon Fraser University, Canada.* Huazhong University of Science and Technology, China.†

[1]http://www.pewglobal.org/2016/02/22/smartphone-ownership-and-internet-usage-continues-to-climb-in-emerging-economies/

[2]https://www.similarweb.com/blog/pokemon-go-update

[3]https://blog.google/products/maps/become-pokemon-master-with-google-maps/

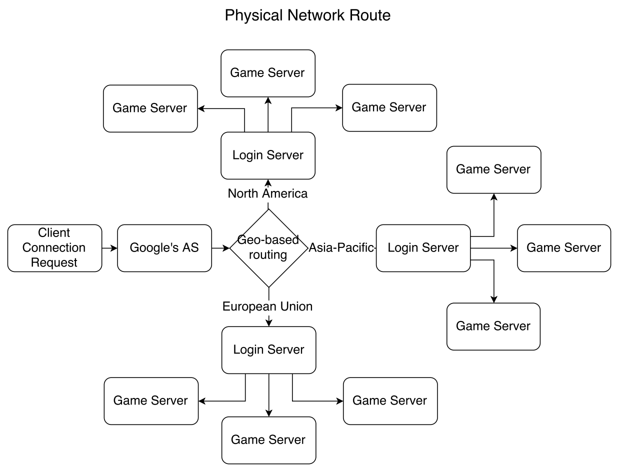


Fig. 1: PGO Server Selection

of events which culminated in the creation of PKG by Niantic, an internal Google start-up[4].

B. Networking Architecture of Pokemon Go

We observed PKG to have a logical networking topology resembling a star topology. Namely, a central URL, pgorelease.nianticlabs.com/plfe/rpc, directs a client to a regional server which then proceeds to serve all future requests until the connection is terminated. To reconnect back to a game server, the client must contact the central server and wait for a response. The client is not guaranteed to connect to the same edge server. It is important to note that while exploring the networking topology we found that although the connection all appear to route to a single Google IP located in Mountain View California the connection is actually being serviced by a server closer to the client. We performed an investigation using trace-route and RTT analysis, and by inspecting the autonomous systems our packets traversed. Our discoveries are presented in Figure 1. In our measurements from servers located in six geo-distributed locations we discovered that Google handles authentication requests in at least three distinct locations. Further, regardless of which region we resolve the URL pgorelease.nianticlabs.com/plfe/rpc in our experiments, we are always given the same IP address corresponding to a location in Mountain View California. Based on our network analysis we find that it is likely Google uses agreements with major Internet exchanges in order to service authentication requests closer to the clients.

All communications between the client and the server are handled over HTTPS and all data is exchanged in the protobuf format. The bulk of network transactions are the retrieval and updating of map objects from the server, based on the player's location.

Specific API calls are wrapped and sent in the repeated requests field, and the authentication ticket received from the central server is added verbatim into a auth_ticket. The request hash signature is generated using xxHash followed by an in-house encryption algorithm.

[4]http://www.cbc.ca/news/technology/pokemon-google-origins-1.3690769

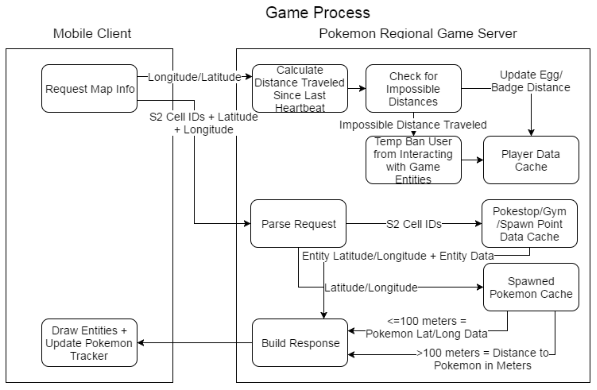
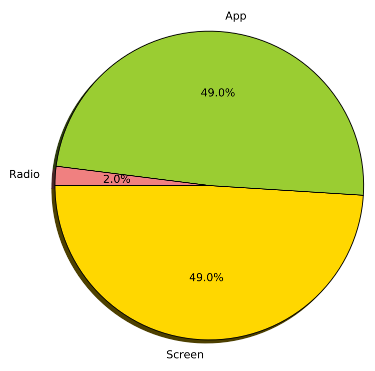

Game Process
Mobile Client
Pokemon Regional Game Server
Request Map Info
Longitude/Latitude
S2 Cell IDs + Latitude + Longitude
Calculate Distance Traveled Since Last Heartbeat
Check for Impossible Distances
Update Egg/ Badge Distance
Impossible Distance Traveled
Temp Ban User from Interacting with Game Entities
Player Data Cache
Parse Request
S2 Cell IDs
Pokestop/Gym /Spawn Point Data Cache
Entity Latitude/Longitude + Entity Data
Latitude/Longitude
Spawned Pokemon Cache
Draw Entities + Update Pokemon Tracker
Build Response
<=100 meters = Pokemon Lat/Long Data
>100 meters = Distance to Pokemon in Meters

Fig. 2: Pokemon Go Architecture

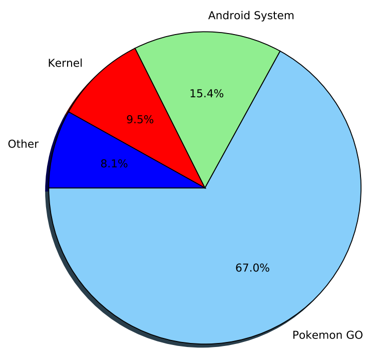
C. Client Architecture and Dataflow of Pokemon Go

In Figure 2, we provide an abstracted datapath of the augmented reality update process in Pokemon Go. The heart of the application is the update process to retrieve new Pokemon spawns and other map entities. We have discovered two processes to update map entities, a major update and a minor update. The major update occurs immediately post-login and retrieves all map entities, i.e. Pokemon, Pokestops, gyms, and spawn points, over a large area centered on the player. The minor update is identical to the major update, the difference being that the minor update receives only Pokemon, Pokestop, and gym data in a local area, and that it occurs more frequently than the major process - about once every 6 seconds as opposed to once every 60 seconds. Figure 2 describes the major update, where the client requests all map entities around a specified latitude and longitude. The request is composed of the player's current latitude and longitude coordinates, and a list of S2 cell IDs[5] to retrieve Pokestop and gym data for. Once the request is received by the server, it performs a displacement check between the current coordinates sent and the last coordinates received. If this displacement exceeds a threshold of what is considered physically possible, the player is silently temporarily banned from interacting with game entities for an unknown duration. If the displacement does not exceed this threshold, a secondary check occurs to determine if the displacement should be counted as valid for the game mechanics of egg incubation and badge credit. To pass this check, the displacement must not exceed 300 meters per minute. Since this distance check is not pertinent to producing a server response, we conjecture that this happens asynchronously while the server builds a response. To construct the response, the server takes the latitude and longitude of the request and constructs a list of all Pokemon currently active within a 200 meter radius of the player. If a Pokemon is less than 100 meters away from the player, the exact coordinates and time-to-live in milliseconds of the Pokemon are included as

[5]docs.google.com/presentation/d/1Hl4KapfAENAOf4gv-pSngKwvS_jwNVHRPZTTDzXXn6Q/



(a) Power total



(b) Power app

Fig. 3: MotoG: Power Consumption (Total: 3544 mW)

attributes. Otherwise, only the distance in meters is included as an attribute. Simultaneously, the server also takes the S2 cell IDs sent and constructs a list of all Pokestops, gyms, and spawn points. Pokestop entities contain an attribute for active modifiers (currently the only modifier is a lure module), gyms contain attributes for the current prestige, team owner, and the current highest combat power (CP) pokemon in the gym. Once both lists are generated, the response is sent to the client.

D. Energy Consumption

During the initial distribution of Pokemon Go there were many reports of the app having a deleterious effect on the battery life of mobile devices. Motivated by these reports, we investigated and quantified the power usage of a smartphone running Pokemon GO, in order to determine what improvements might be introduced. We devised a measurement strategy involving a real world Android device, namely the Moto G 3rd Generation. Our test platform specifications include Quad-core 1.4 GHz Cortex-A53 CPU, a Adreno 306 GPU, 2 GB of RAM and 16 GB of internal flash memory. The devices operating system was updated to latest available Android version 6.0 (Marshmallow). We used the phone's built in battery discharge sensor and the measurement application *GSam Battery Monitor* to profile the Pokemon Go application. To make a stable testing environment we ran the Pokemon Go app for 30 minutes and collected the average battery discharge rate. The adaptive brightness setting of the screen was disabled to ensure that changes in the testing environments ambient environment would not affect the measurements.

We find that under our testing conditions Pokemon Go uses a system wide power consumption of 3544 mW. Figure 3 illustrates the percent of energy and which subsystem is consuming it as well as a breakdown of which running apps are consuming power. As can be seen we have an even split at 49% for both the screen and apps, and only 2% being consumed by the radio. We find that the device's screen consumes exactly 1736.71 mW, Pokemon Go app 1169.62 mW, other apps 567.10 mW, and finally the radio only 70.90 mW. Our results make it clear that the augmented reality app itself has a very high energy consumption cost.

It is well established in the literature that the screen can be a large drain on the battery of a smart phone. Why the augmented reality app Pokemon Go consumes so much power required further exploration. To that end we further profiled the application using the development platform *Android Studio*. By profiling the app we find that over 80% of the CPU cycles are being used by the function call UnityPlayer.nativeRender, which is responsible for processing 3D objects for display. We conjecture that this function call is likely where the in-game and AR objects are composed for viewing by the user, and that this task is extremely computationally expensive. Of the approximatively 20% remaining CPU time, the largest contributor is a function call to the Android system's "ContextService". The context service is responsible for gathering data from the sensors such as the GPS, accelerometer, and gyroscope. Pokemon Go makes heavy use of this service to feed data from the phones sensors to the game engine to update the game world. In Section V, we further explore these findings and propose power saving techniques.

E. Mobility and Scalability Pokemon Go

Next, we wanted to investigate the network scalability of PKG during its initial deployment. It was widely reported that many users had difficulties connecting to the server in order to play, and we wanted to investigate which components of networking architecture was being overloaded. To that end we collected data of the failure events experienced by our user during gameplay. Presented in Figure 4, these anomalies were grouped by type, and the number of events experienced on each day measured. When logging in failed, this was due to a login server overload or a timeout event. A Remote Procedure Call (RPC) fail occurred when too many users were requesting entities at the edge server, resulting in a error message. The Misc fails were from unknown or invalid responses, essentially when the response received was unintelligible to our user. These results give us insight into the load experienced by the servers on different days, and beyond that, they tell us when there were not enough resources allocated. For example, if failure events of all types were relatively high on a day, such as occurred on July 22, we observe that servers were overloaded but resources were fairly distributed. However, if login failures were low but RPC and Misc events were relatively high such as on July 26, it likely indicates that server resources could have been better allocated to cover these services. The extremely

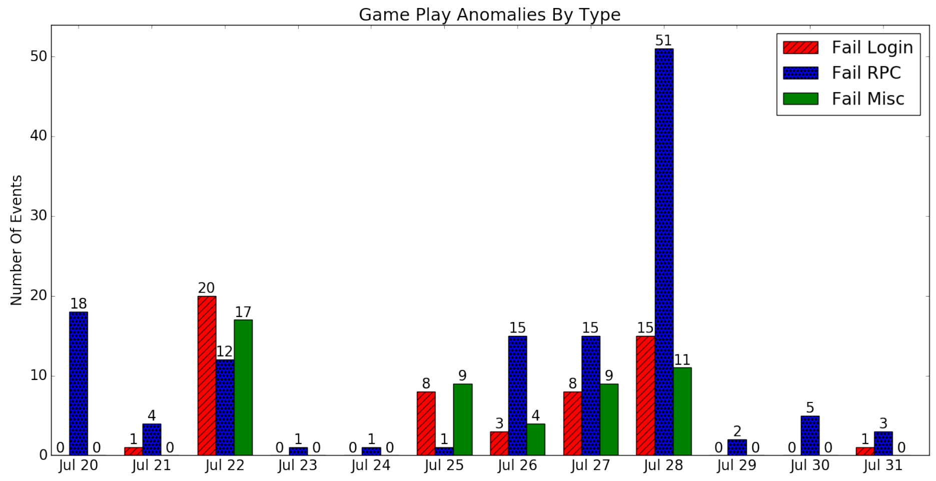


Fig. 4: Pokemon Go Login Failures

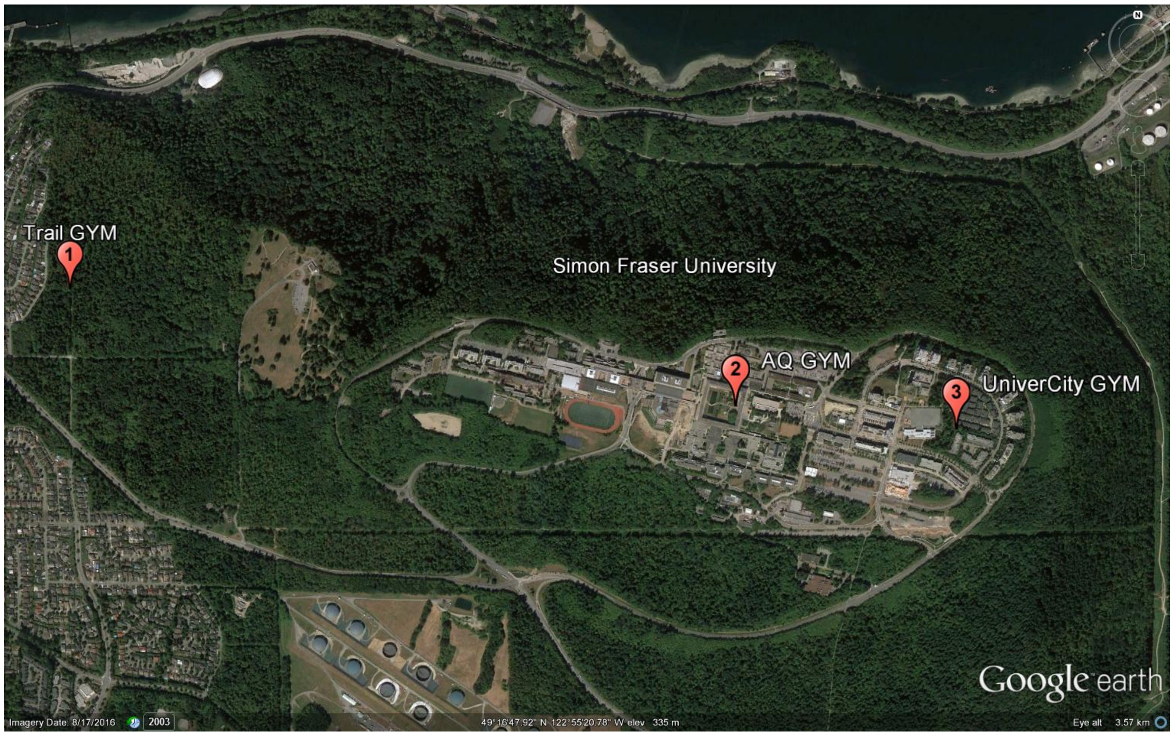
high incidence of RPC failures on July 28 stands out clearly from this pattern. On this day, service disruptions due to massive increase in players were reported throughout both east and west coasts of North America, as well as Japan, related to issues with Google Login. [6]

In an effort to study the distribution of players on our University campus we created a simulated Pokemon Go user which we directed to interact with Gym locations in the game every 5 mins. By tracking the Experience level of the Gym as our user observed it, we were able to measure the number of interactions occurring at each location. These interactions are the sum of all the unique users who interact with that location and multiple interactions from the same users, giving an overall picture of where and when users are playing. Figure 5 shows graphs of this interaction activity at 3 locations at and near Simon Fraser University on a weekday and on a weekend day, with some very interesting patterns emerging. Location 1 is a gym located on a relatively remote hiking trail, so it is not surprising that it has the fewest interactions. The users observed at this location on the weekday are likely people using the trail to walk to and from work, and for leisure in the evening. There are more interactions on the weekend, when more people have time to hike for leisure. Even at this remote location in a forest, at least one user was playing at 2 am. The gym at location 2 is at the centre of the university campus with the interactions representing the traffic that this busy academic centre hosts, so it is not surprising that this location is the busiest, with consistently high activity throughout the weekday. This academic centre is also within easy walking distance to the residential area of campus and the nearby community, which explains why the pattern of high activity is maintained throughout the evening and into the night, and why it is nearly as high on the weekend. Location 3 is in the residential development called UniverCity, inhabited by students and staff of the university with their families but also members of the community not affiliated with the university. Because the number of interactions here is so much lower than in the academic centre, we can tell that students who do not live at the university must be the biggest majority of users at location 2. These activity graphs paint a picture of smartphone use for gaming by telling us where, when, and how interactions there are, and even allows us insight into which demographics these users might be (students, residents, etc). We have provided an interactive heat map of many more locations and times around Simon Fraser University at https://goo.gl/uXbX4V.

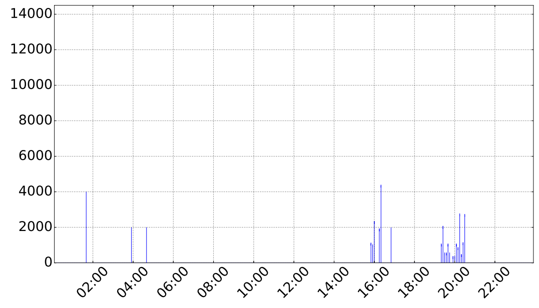
III. QoE Enhancement with Accelerometer and WiFi Signals

It is common for the game's players to use the Pokemon Go app in indoor environments such as cafes, plazas, libraries, and offices etc. The satellite based GPS suffers from significant signal attenuation indoors. Since the game relies on GPS to locate the player, the mobility of the game is severely degraded indoors. The rapid development of smartphones with rich built-in sensors has brought great prosperity to various indoor location-based services (LBS) such as indoor positioning, personal tracking, emergency management and navigation [1]. The demand for indoor LBS has driven innovative research on indoor localization and creates the opportunity to improve and incorporate such a service for AR applications. Given the pervasiveness of WiFi infrastructures, many WiFi fingerprint-based indoor localization approaches have been developed, e.g., [2], [3], [4], [5], [6]. Such approaches normally involve a cumbersome offline site survey (also known as calibration) process that collects the WiFi fingerprints in the *whole indoor area*, and since the WiFi signal is very sensitive to environmental change, this process is often repeated from time to time to update the WiFi fingerprints [7]. Hence, such approaches are not efficient for practical deployment such as AR gaming where players are hopping between place to place (e.g. "Arenas" in PKG). Furthermore, the accuracy of fingerprint-based localization systems is not satisfactory. Depending on

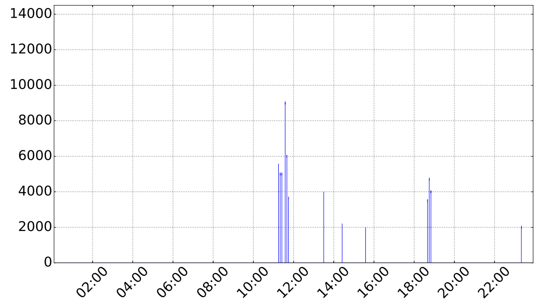
[6]https://www.slashgear.com/today-pokemon-go-server-status-is-down-what-to-do-28449884/



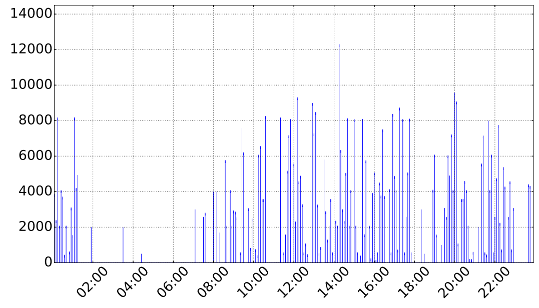
(a) GYM locations



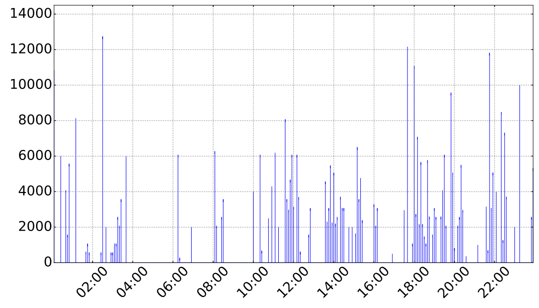
(b) Trail Gym Weekday



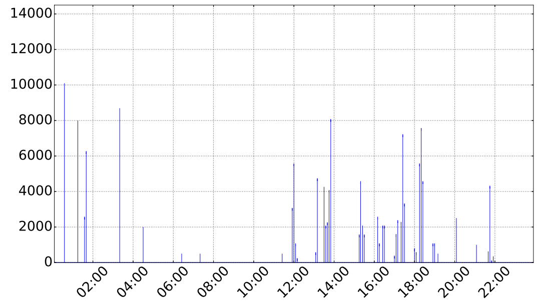
(c) Trail Gym Weekend



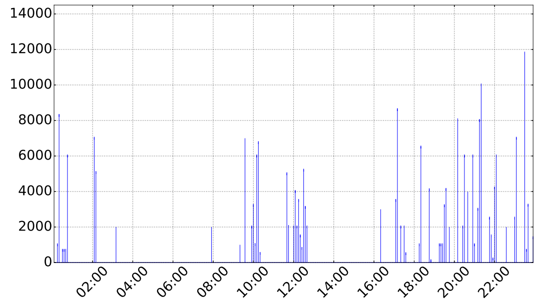
(d) Academic Quadrangle Weekday



(e) Academic Quadrangle Weekend



(f) UniverCity Weekday



(g) UniverCity Weekend

Fig. 5: Pokemon Go activity at SFU (July 21st and July 23rd)

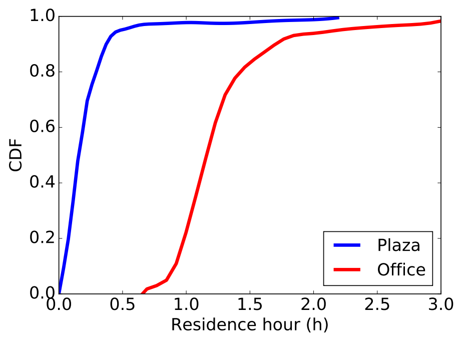


Fig. 6: CDF of people's residence time in a large plaza and office

the environment settings and calibration effort, the median localization accuracy varies from 2 to 10 meters [2], [6], [7], [8] which may affect the QoE of the game. For example, a player may not be able to reach an Pokemon Go Arena indoors even if he/she is in the range of the Arena.

In this section, we discuss a promising indoor localization solution which greatly alleviates the difficulties of fingerprint collection by smartly leveraging the relatively stationary people *stationary peers (SPs)* that are largely available in common indoor environments to assist localization. The key motivation behind this solution is that 1. areas where the app is used are often near locations where people tend to be stay in the same position for a significant amount of time, e.g., cafe and office; and 2. we observe that many Pokemon Go players stay in a fixed location and do not change their position while playing. We can therefore leverage the sensors on their devices to perform indoor localization and improve the QoE of the AR application.

A. Are There Sufficient Stationary Peers?

We have conducted two-hour site surveys in a large five-story plaza and an indoor office building, respectively. Here, we define the time during which a person does not change his/her position as *residence time*. Figure 6 depicts the cumulative distribution function (CDF) of people's residence time in these two environments. We can see that people in the office all have a residence time of over half an hour; while in the plaza, the majority of people's residence times are within half an hour. Considering that a localization process only needs around one second, the people with a residence time greater than a few minutes can be employed to assist localization, and Figure 6 clearly demonstrates the availability of these people. In our paper, we define that a person can be employed as an SP when his/her residence time is over ten minutes and his/her position is within a cubic area with a side length of 2 meters. It is worth noting that this condition can be adapted for different indoor scenarios.

Another issue is obtaining sufficient numbers and distribution of SPs, as the trilateration localization method needs at least three beacons surrounding a target. We conduct another measurement study to explore the distribution of candidate SPs in an indoor office and a shopping mall. Figure 7 depicts the layout of candidate SPs in these two real world environments. Given that the typical acoustic range is around 10 meters (the maximum is around 40 meters) [9], [10], [11], [12], any point in the area can be covered by at least three SPs in these two environments.

In summary, our measurements demonstrate both the availability and sufficiency of SPs in typical indoor environments and thus validate the feasibility of utilizing SPs for indoor localization. The availability of these SPs in the indoor environment combined with the rich sensors present in smartphones allow a natural way to bring AR applications such as Pokemon GO indoors.

B. Overview of the SP Based Localization

We propose an SP-based localization system. The system includes three main components: SP identification, SP localization, and target localization.

SP Identification. The goal of SP Identification is to determine whether a person is *relatively stationary* or not. Using the accelerometer to identify SPs is a natural solution given that accelerometers have been widely used to track people's activities [13], [14], [15], [16].

Through analyzing SPs' activities and the corresponding acceleration signals, we propose a practical and efficient method to identify SPs. We find that SP's acceleration signals either have low magnitude or low root mean square (RMS). Furthermore, the unbiased autocorrelation of the acceleration signals exhibits no repetitive pattern. These distinctive features effectively help us identify SPs.

We have also considered using RSSI, as RSSI signatures are supposed to remain unchanged at a fixed location. Yet it has been shown that RSSI is subject to temporal fluctuation [7]. Therefore, utilizing RSSI to identify SP is not viable.

SP Localization: To make the system lightweight and easy-to-deploy, we do not rely on any customized hardware or fine-grained physical layer signatures; instead, we still use RSSI. Due to multipath fading, medium contention and other electromagnetic noise, RSSI is inherently time-variant [17] and normally incurs coarse localization accuracy [7]. To this end, for each WiFi AP, we utilize *a set of RSSI measurements* as the fingerprints for each reference location. Compared with traditional RSSI fingerprint localization methods [2] that depend only on one-shot measurements and do not account for temporal variation.....

Target Localization: To locate a target, we need a reliable ranging scheme. Traditional RF (Radio Frequency) based TOA or Time-Difference-of-Arrival (TDOA) [18] methods normally require high precision synchronization, and signal strength-based approaches [19] not only suffer from coarse ranging accuracy but are also easily impaired by human obstruction. Acoustic ranging schemes [9], [10], which have proven they can be accurate to the range of centimeters, can largely overcome these shortcomings.

Initially, when a target requests the localization service, it broadcasts a chirp signal [9], [10]. The surrounding SPs who receive this signal send a confirmation message to the remote server. Then, under the direction of the remote server,

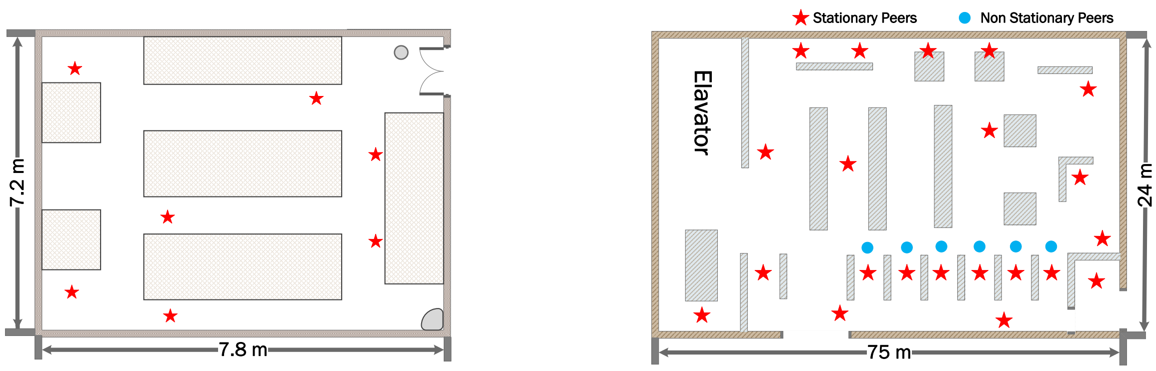


Fig. 7: The stars represent SPs. The circles are non SPs, e.g., the customers who just stay at the locations for minutes.

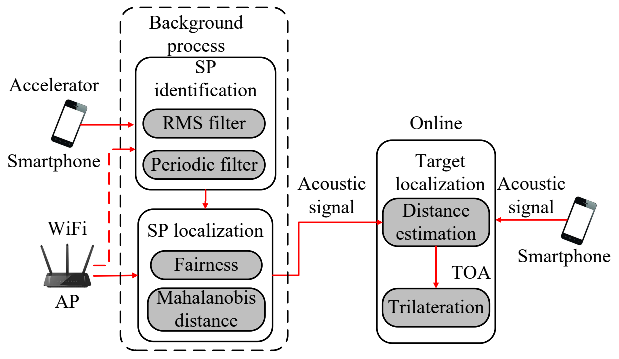


Fig. 8: System overview

an acoustic ranging method [10] is applied to estimate the relative distance between SPs and the target. Finally, the trilateration method is applied to obtain the target's location. Taken together, these localization strategies could help solve the issue of indoor localization for augmented reality applications, potentially expanding the range of locations available and improving QoE.

IV. QoE Enhancement with RFID Sensors

Aside from the location information, AR applications often involve rich user activities such as sitting, standing, and walking; correctly identifying these activities could potentially enhance the QoE of these applications. In its current form, users in PokemonGo interact with the AR world by GPS geo-located movements and tactile interfaces such as the touchscreen. These represent only a small fraction of the movements humans perform on a daily basis. Pervasive sensors in modern smartphones could help accurately measure our fine-scale movements such as sitting, eating or talking, which could be integrated into increasingly rich AR worlds. For example, Disney has built a RFID gaming system that can sense when the player is moving or touching objects attached with tags in near real time [20]. The past few years have seen booming interest in human activity identification that provides a range of Internet-of-Things applications, such as health care and smart homes [21]. Traditional solutions mainly use radars [22], cameras [23], and various inertial sensors [24]. Yet, sensor or device based radar solutions require targets carrying sensors/wireless devices that are often not negligible in both size and weight. While camera-based and device-free radar-based systems have freed this limitation, they suffer poor performance in accurately identifying multiple objects, especially under Non-Line-of-Sight (NLoS) scenarios. Radio Frequency Identification (RFID) is a promising technology that can overcome those difficulties due to its low cost, small form size, and not needing a power source, making it widely used in a range of mobile applications.

The mobility of targets is an essential and important metric to differentiate various human activities [25][26], e.g., sitting and walking. Yet, the granularity of mobility quantified in existing solutions is not adequate. For example, [25][27] can only distinguish static and mobile objects, while [26][28] deal with targets moving at similar speed.

Therefore, quantifying the intensity of mobility that is closely related to typical indoor activities is not well addressed yet. One may think of making use of the RFID localization techniques that have successfully achieved centimeter or even millimeter accuracy for mobility detection. Unfortunately, while such advanced solutions as RF-IDraw [28] and Tagoram [29] achieve high accuracy through exploring antenna arrays, their performance degrades heavily for indoor environments with multipath. Intuitively, their phase measurement, a core operation, can be remarkably affected by multipath, invalidating the key assumption [30] that the Angle-of-Arrival of the direct path is related to the measurement phase difference between antennas, especially in Non-Line-of-Sight (NLoS) cases. Other localization solutions relying on predeployed reference tags [31][32] generally require the tagged objects to be static or with limited moving velocities (i.e., 0.17-0.3 m/s), which is not even applicable for walking (1$\sim$1.4 m/s) and running (5$\sim$7 m/s). Mobility may also be estimated through the doppler effect [26]. Yet it works with only static communication environments and will again become unstable in fast-changing indoor environments with dynamic multipath, random signal/thermal noise, and varying antenna orientations. Empirically, we show that prior schemes suffer from serious performance degradation for detecting realworld moving tags in typical indoor environments, since using a sole parameter for mobility detection is ineffective in multipath scenarios.

In this section, we present a mobility-aware activity identification system for RFID tags through intelligent pro-

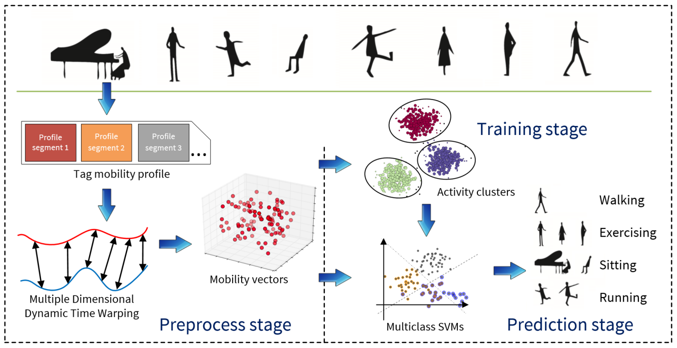


Fig. 9: Our supervised learning framework for mobility detection and activity identification

filing, which works robustly in multipath-rich indoor environments called i^{2}tag. i^{2}tag constantly generalizes a huge amount of fine-grained mobility, which further enables us to utilize a supervised learning framework for activity identification as shown in Fig. 9.

At a high level, it goes through the following major steps:

- **Preprocessing stage.** we employ a novel fine-grained *mobility profile* to quantify different levels of mobility, which seamlessly integrates RSSI variance and packet loss rate, as well as a relative-phase-based fingerprint. The latter is highly effective in distinguishing tag mobility in complicated indoor environments with random signal noise and multipath.
  By comparing the measured mobility profile against known reference mobility profiles, we detect the tag velocity through a Multiple Dimensional Dynamic Time Warping (MDDTW) [33] algorithm. We classify tag mobility into multiple categories based on the estimated velocity; for instance, *stationary*, *micro-mobility*, and *macro-mobility*.[7] In this stage, we split tag mobility profile $\mathcal{P}_i = \{p_i^1, p_i^2, ...\}$ into segments in equal window size τ as $\{\mathbf{p}_i^1, \mathbf{p}_i^2, ...\}$, which will be transferred into a mobility vector as $\mathbf{v}_i = \{\nu_i^1, \nu_i^2, ...\}$, where mobility vector as an underpinning unit is applied in a multiclass support vector machine (SVM) [35].
- **Training stage.** Each tag mobility profile $\mathcal{P}_i$ is represented by a corresponding mobility vector $\mathbf{v}_i$, then we can distinguish different kinds of activities, e.g., sitting, exercising, walking, and running. To be specific, V_{train} in training samples with corresponding labels will be trained to build the mapping σ from the feature $\mathbf{x}_i$ of mobility vector $\mathbf{v}_i$ to activity label y_i.
- **Prediction stage.** We perform activity recognition in a supervised learning way. For each mobility vector $\mathbf{v}_i \in V_{test}$, we determine whether the feature $\mathbf{x}_i$ of mobility vector is concentrated in certain activities, then label it via σ to achieve corresponding y_i.

i^{2}tag is readily deployable using off-the-shelf RFID readers[8] (a single UHF reader with limited number of antennas) and allows reusing existing RFID readers for indoor activity identification. We have implemented a prototype of i^{2}tag using a `Thingmagic` reader and `Impinj` tags, and have conduct extensive experiments in indoor environments. The results demonstrate that, with i^{2}tag, a single RFID reader with two connected antennas can accurately distinguish the tag velocity, classify the fine-grained mobility and four categories of activities, with an average detection rate up to 96%. More details of this work can be found in our paper [36]. Techniques to measure and identify fine scale movements using pervasive sensors such as the ones we have described, allow for richer interactions in AR worlds such as Pokemon Go.

V. Cloud based AR content delivery: System Overview

As shown in previous sections, AR applications with pervasive sensing capabilities such as Pokemon Go can consume significant battery power of mobile devices. Since the Pokemon Go infrastructure is largely built atop Google's cloud, there are great opportunities if we can offload more heavy lifting workloads to the cloud end. In this section, we outline a video streaming-based, cloud offloading framework to take charge of the AR rendering. The framework is motivated by the following observations:

- While AR content rendering consumes the largest amount of power, due to the pervasiveness of video applications, mobile devices nowadays contain power-efficient hardware chips for video processing (e.g., decoding and encoding); processing the AR content video stream can be much more power efficient than rendering the AR content locally.
- The cloud-based framework could significantly reduce the hardware requirements of the mobile devices. From the App developers' perspective, they do not have to deal with the vast heterogeneity of mobile devices, adapting and testing the game against different OS platforms.
- Hosting AR content generation in the cloud could substantially reduce the time-to-market of the App. It also reduces the complexity of applying patches and updates to the app for the game makers.

A. Cloud Server Setup

In Figure 10, we depict the architecture of our cloud based AR streaming platform, Rhizome-AR.

At the server side, the first two modules are the *MetaData processor* and the *Client Interaction*. The *Application Logic* is essentially the game instance. It processes the sensor data and client actions from the previous two modules and compute the updates to the game world, based on which the rendering is then performed by the *Screen Rendering* module. The rendered scenes are passed to the *Video Encoder* module that contains a video encoder and a discreet framer. The video encoder, which we will highlight later, is selectable, consisting of either

[7]Zhou *et al.* [34] proposed a random mobility model for the different mobile situations, e.g., the user may slowly move the tag although he/she is stationary or his/her movement is confined within a small area.

[8]It is worth noting that the limited programming interface posed by commercial tag readers provides only RSSI and phase values. As such, advanced algorithms for powerful wireless device are not necessarily applicable here.

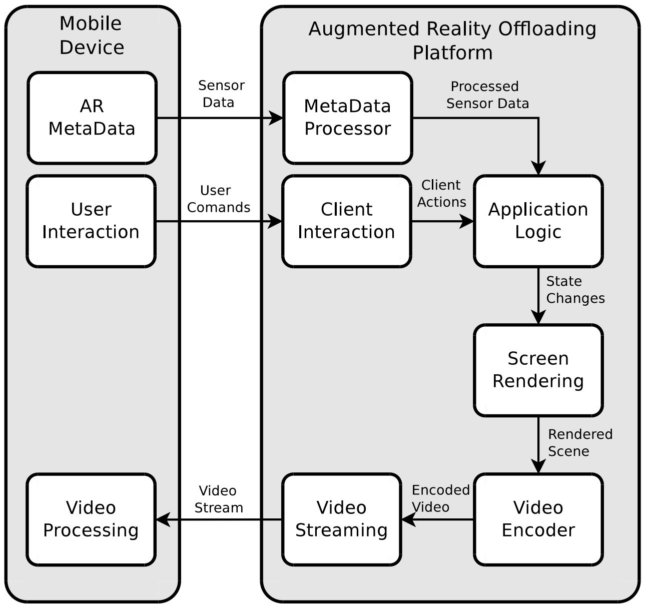


Fig. 10: Cloud based AR Streaming Framework

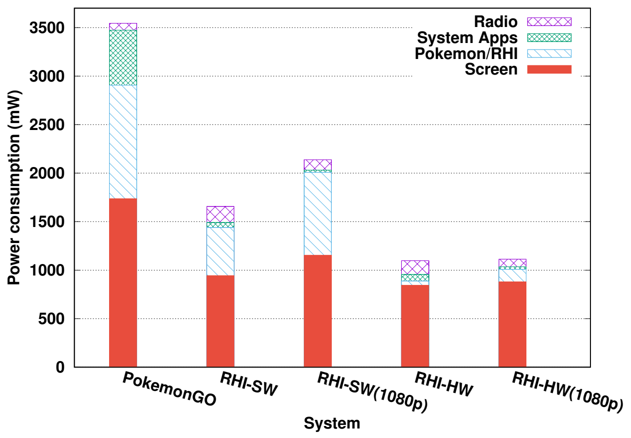


Fig. 11: Comparison of the original PKG and video offloading based PKG

a software or hardware H.264 encoder. In either case, the encoder needs additional support to be adapted for real-time streaming, namely, a discreet framer, which allows the Live555 streaming library to request live frames from the encoders at a desired video stream frame rate. The encoded video stream is then encapsulated and transported in either UDP, TCP, or HTTP.

The module we want to highlight is the encoding module. Rhizome not only supports software encoding using the highly optimized x264 encoder for H.264 video but also NVIDIA GRID and its hardware H.264 encoder. The choice of streaming protocol can also be customized, e.g., RTSP over UDP or TCP, as well as HTTP streaming, using the the widely deployed open-source streaming library Live555 as the streaming engine.

Our design and implementation are platform-independent, although a GRID GPU is required for our hardware encoding implementation. We have deployed and experimented the system on Amazon EC2 GPU Instances (G2), a of cloud instances backed by the Intel Xeon E5-2670 (Sandy Bridge) processors and the NVIDIA GRID-K520 board that contains a GK104 GPU with 1536 CUDA cores and 4GB of video memory. The GRID's on-board hardware video encoder supports up to 8 live HD video streams (720p at 30 fps) or up to 4 live FHD video streams (1080p at 30 fps), as well as low-latency frame capture for either the entire screen or selected rendering objects, enabling a G2 instance to offer high-quality interactive streaming such as 3D game streaming.

B. Mobile Client Setup

One of key motivations of migrating AR games to the cloud is to enable resource-constrained (in terms of computation, memory, battery, etc.) mobile clients to play advanced games. The *AR MetaData* module is in charge of casting the various sensor data collected at the mobile device to the cloud server, interfaced with MetaData Processor module. Hence, when the video is received by a mobile client, our client-side *Video Processing* module is configured to utilize the hardware decoder with real-time optimizations to decode the video and display it on the client device. The *User Interaction* module supports input devices including game-pad, keyboard and mouse.

The selected test system mobile device is a Moto G 3rd Generation smartphone which includes Quad-core 1.4 GHz Cortex-A53 CPU, a Adreno 306 GPU, 2 GB of RAM and 16 GB of internal flash memory. We updated the device's operating system to the latest available Android version 6.0 (Marshmallow).

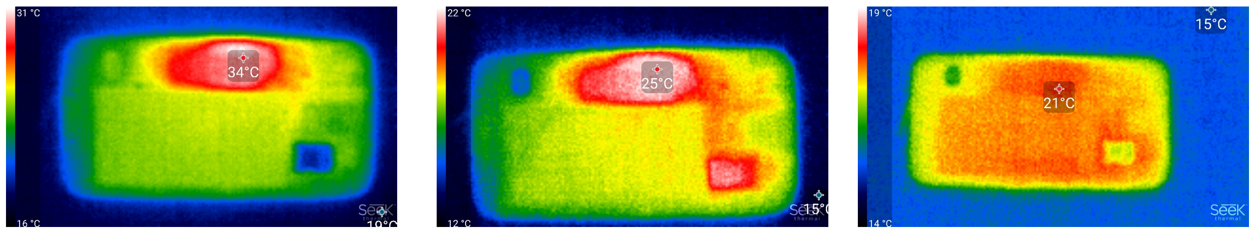
C. Mobile Device Power Profiling

In Figure 11, we depict the power profiling results on the tested mobile device. Regardless of which video decoder being used, Rhizome-AR is able to substantially reduce the power consumption of Pokemon Go. We see slight increases on the radio, which is due to the increased usage of radio for video streaming. Another part of the power saving comes from the screen. Rhizome-AR is able to adjust the Frame Rate per Second (FPS) based on players' preference, helping reduce the power consumption on the screen when players do not require high frame rate.

Further, we compare our video based offloading approach with the locally-rendered Pokemon Go through thermal imaging. As shown in Figure 12, with the use of hardware decoder, video based offloading greatly reduces the heat generated at the CPU (block in red), which confirms with our findings in the previous power profiling.

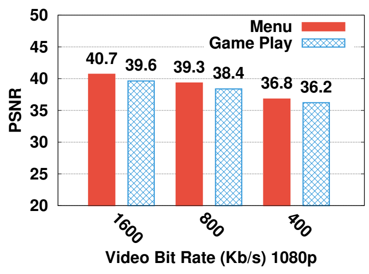
D. Streaming Quality

We describe the streaming quality of Rhizome-AR in Figure 13 We analyze the video using two classical metrics, namely *Peak Signal-to-Noise Ratio* (PSNR) and *Structural Similarity Index Method* (SSIM). The results for PSNR are given in Figure 13a and SSIM are given in Figure 13b, respectively. The PSNR method quantifies the amount of

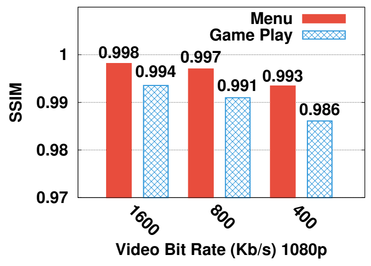


(a) Pokemon GO (b) Offloading SW Decoder (c) Offloading HW Decoder

Fig. 12: Effect of video based offloading on Temperature



(a) PSNR 1080p



(b) SSIM 1080p

Fig. 13: Image Quality

error (noise) in the reconstructed video, which has been added during compression. The SSIM method calculates the structural similarity between the two video frames. In terms of both metrics, Rhizome-AR is able to attain high streaming quality even at the low bit rate, which allows our system to be used without excessive bandwidth requirements. More details of the Rhizome platform can be found in our papers [37][38]. Offloading processing tasks to the cloud helps alleviate a key limitation of the PokemonGO app, namely its reputation for rapidly draining batteries. This offloading also free up resources and battery life for more other tasks such as running advanced sensors.

VI. Related Work

A. Location Based AR

Prior to Pokemon Go, Ingress is among the first location based mobile games that paved the way for the later popularity of the location based AR games [39]. Cloud computing is the driving force behind these successes. Both games are powered by the Google Cloud and the Google edge network to achieve global, high-quality coverage [40][41]. Other than gaming, AR has been used in areas such as healthcare and commerce [42]. These services share similar server architecture and client device usage. The future of cloud gaming was discussed in a recent article by Cai *et al*. The article forecasted changes in the programing paradigm of gaming applications to facilitate better integration between games and cloud offloading [43].

B. Indoor Localization

RSSI-based Localization Previous work on RF-based positioning primarily relied on RSSI information [44][45][46][47][48]. The RF fingerprinting, pioneered by Radar [44], employs RSSI based fingerprinting matching against a database to determine the indoor location. LANDMARC [45] introduces the RF fingerprinting technique to RFID localization. Vire [46] used imaginary reference tags, referred to as "virtual tags" to achieve higher accuracy. EZ [47] requires site surveys at only a few user locations. Later on several other improvements over RSSI fingerprinting have been proposed, such as incorporating inertial sensor hints [48]. They typically deployed reference tags on a monitoring region and then use RSSI values to locate a specific tag. The major limitation of RSSI-based approaches is unreliable, since RSSI measured values are highly sensitive to multipath effects, and thus difficult to achieve high-precision localization. Other works on device-free localization rely on RSSI fingerprints [49][50], which are generated in the training phase by requiring a person to stand in different locations throughout the area of interest. In the testing phase, they localize a person by mapping the resulting RSSI to the closest fingerprint

Distance Ranging One of the simplest approaches is to calculate the distance between the transmitter and receiver based on received phase measurements. Here, we discuss only some recent and relevant works. Li *et al.* [51] propose a multi-frequency based ranging method for passive RFID tag localization. Using phase measurement for distance ranging, theoretically, could achieve high localization accuracy. Due to the multipath effects, the phase measurement is not corresponding to the dominated path, and leads to high ranging error. Liu *et al.* [52] presents an RFID localization scheme by using multiple antennas to receive phase measurements from tags, where the hyperbolic positioning method is employed to correlate phase measurements.

The above methods are somewhat not applicable in mobile cases, and we focus on leveraging the changing mobility profile for mobility detection. The intuition behind our design is that by analyzing the spatial-temporal dynamics in the mobility profiles, we can accurately estimate the mobility of tags. Previous work may rely on the reference tags or the dedicated hardware with many antennas to capture the mobility profile. Moreover, SAR-style techniques require constantly moving either the RFID reader or tags. In contrast, our scheme works on COTS devices with only two antennas in multipath-rich indoor environments.

C. Activity Recognition

Activity recognition solutions exploit the change of wireless signals incurred by the human's actions. RF-compass [53] presents a state-of-the-art WiFi-based interface, yet it only supports the detection and classification of a predefined set of nine gestures. WiVi [54] utilizes WiFi signals to detect users through walls and identify their gestures, which focuses on tracking through dense walls such as concrete by using MIMO interference nulling to eliminate reflections off static objects. RistQ [55] leverages the accelerations from a wrist strap to detect and recognize smoking gestures. RF-IDraw [28] can track human writing by tracking a passive RFID tag attached to his/her pen. E-eyes [56] is a location-oriented activity identification system, which leverages WiFi signals to recognize in-home human activities.

Ding *et al.* [26] developed FEMO that uses the frequency shifts of the movements to determine what exercise a user is performing.

VII. Conclusion

In this article, we explored pervasive sensing applications through the deployment of the popular augmented reality application Pokemon GO. We discussed real world limitations of the current systems, from scalability to energy consumption. Further, we discussed how to improve these sensing applications by integrating advances such as indoor localization, and enriching the AR environment by improving activity identification. Finally, because these pervasive sensing application can have high power consumption, we investigate how innovative cloud offloading platforms could enhance these applications.

Acknowledgment

The work is supported by the Natural Sciences and Engineering Research Natural Sciences and Engineering Research Council (NSERC) of Canada. This publication was also made possible by NPRP grant #[8-519-1-108] from the Qatar National Research Fund (a member of Qatar Foundation). The findings achieved herein are solely the responsibility of the authors.

References

[1] Y. Gu, A. Lo, and I. Niemegeers, "A survey of indoor positioning systems for wireless personal networks," *IEEE Communications Surveys Tutorials*, vol. 11, no. 1, pp. 13–32, January 2009.

[2] P. Bahl and V. Padmanabhan, "Radar: an in-building rf-based user location and tracking system," in *Proceedings of the 19th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2000.

[3] M. Youssef and A. Agrawala, "The horus wlan location determination system," in *Proceedings of the 3rd International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2005.

[4] Y. Ji, S. Biaz, S. Pandey, and P. Agrawal, "Ariadne: a dynamic indoor signal map construction and localization system," in *Proceedings of the 4th International Conference on Mobile Systems, Applications and Services (MobiSys)*, 2006.

[5] C. Wu, Z. Yang, Y. Liu, and W. Xi, "Will: wireless indoor localization without site survey," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 4, pp. 839–848, April 2013.

[6] L. Li, G. Shen, C. Zhao, T. Moscibroda, J.-H. Lin, and F. Zhao, "Experiencing and handling the diversity in data density and environmental locality in an indoor positioning service," in *Proceedings of the 20th Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2014.

[7] Z. Yang, Z. Zhou, and Y. Liu, "From rssi to csi: indoor localization via channel response," *ACM Computing Survey*, vol. 46, no. 2, pp. 25:1–25:32, December 2013.

[8] S. He, T. Hu, and S.-H. G. Chan, "Contour-based trilateration for indoor fingerprinting localization," in *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2015.

[9] H. Liu, J. Yang, S. Sidhom, Y. Wang, Y. Chen, and F. Ye, "Accurate wifi based localization for smartphones using peer assistance," *IEEE Transactions on Mobile Computing*, vol. 13, no. 10, pp. 2199–2214, Oct 2014.

[10] C. Peng, G. Shen, Y. Zhang, Y. Li, and K. Tan, "Beepbeep: a high accuracy acoustic ranging system using cots mobile devices," in *Proceedings of the 5th International Conference on Embedded Networked Sensor Systems (SenSys)*, 2007.

[11] R. Nandakumar, K. K. Chintalapudi, and V. N. Padmanabhan, "Centaur: locating devices in an office environment," in *Proceedings of the 18th Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2012.

[12] W. Huang, Y. Xiong, X. Y. Li, H. Lin, X. Mao, P. Yang, Y. Liu, and X. Wang, "Swadloon: direction finding and indoor localization using acoustic signal by shaking smartphones," vol. 14, no. 10, Oct 2015, pp. 2145–2157.

[13] P. H. Veltink, H. J. Bussmann, W. de Vries, W. J. Martens, and R. C. V. Lummel, "Detection of static and dynamic activities using uniaxial accelerometers," *IEEE Transactions on Rehabilitation Engineering*, vol. 4, no. 4, pp. 375–385, December 1996.

[14] S. J. Preece*, J. Y. Goulermas, L. P. J. Kenney, and D. Howard, "A comparison of feature extraction methods for the classification of dynamic activities from accelerometer data," *IEEE Transactions on Biomedical Engineering*, vol. 56, no. 3, pp. 871–879, March 2009.

[15] J. A. Ward, P. Lukowicz, G. Troster, and T. E. Starner, "Activity recognition of assembly tasks using body-worn microphones and accelerometers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 10, pp. 1553–1567, October 2006.

[16] Z. He, "Activity recognition from accelerometer signals based on wavelet-ar model," in *Proceedings of International Conference on Progress in Informatics and Computing (PIC),*, 2010.

[17] H. Abdelnasser, M. Youssef, and K. A. Harras, "Wigest: A ubiquitous wifi-based gesture recognition system," in *Proceedings of the 34th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2015.

[18] F. Gustafsson and F. Gunnarsson, "Mobile positioning using wireless networks: possibilities and fundamental limitations based on available wireless network measurements," *IEEE Signal Processing Magazine*, vol. 22, no. 4, pp. 41–53, July 2005.

[19] M. Hazas, C. Kray, H. Gellersen, H. Agbota, G. Kortuem, and A. Krohn, "A relative positioning system for co-located mobile devices," in *Proceedings of the 3rd International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2005.

[20] A. Spielberg, A. Sample, S. E. Hudson, J. Mankoff, and J. McCann, "RapID: A Framework for Fabricating Low-Latency Interactive Objects with RFID Tags," in *Proceedings of ACM CHI 2016 CHI Conference on Human Factors in Computing Systems*, 2016.

[21] M. R. Alam, M. B. I. Reaz, and M. A. M. Ali, "A review of smart homespast, present, and future," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 6, pp. 1190–1203, 2012.

[22] J. Xiao, Z. Zhou, Y. Yi, and L. M. Ni, "A survey on wireless indoor localization from the device perspective," *ACM Computing Surveys (CSUR)*, vol. 49, no. 2, p. 25, 2016.

[23] J. M. Chaquet, E. J. Carmona, and A. Fernández-Caballero, "A survey of video datasets for human action and activity recognition," *Computer Vision and Image Understanding*, vol. 117, no. 6, pp. 633–659, 2013.

[24] A. Bulling, U. Blanke, and B. Schiele, "A tutorial on human activity recognition using body-worn inertial sensors," *ACM Computing Surveys (CSUR)*, vol. 46, no. 3, p. 33, 2014.

[25] P. Zhang, J. Gummeson, and D. Ganesan, "Blink: A high throughput link layer for backscatter communication," in *Proceedings of the 10th international conference on Mobile systems, applications, and services*. ACM, 2012, pp. 99–112.

[26] H. Ding, L. Shangguan, Z. Yang, J. Han, Z. Zhou, P. Yang, W. Xi, and J. Zhao, "Femo: A platform for free-weight exercise monitoring with rfids," in *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*. ACM, 2015, pp. 141–154.

[27] C. Wang, L. Xie, W. Wang, T. Xue, and S. Lu, "Moving tag detection via physical layer analysis for large-scale rfid systems," in *IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*. IEEE, 2016, pp. 1–9.

[28] J. Wang, D. Vasisht, and D. Katabi, "Rf-idraw: virtual touch screen in the air using rf signals," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 4, pp. 235–246, 2015.

[29] L. Yang, Y. Chen, X.-Y. Li, C. Xiao, M. Li, and Y. Liu, "Tagoram: Real-time tracking of mobile rfid tags to high precision using cots devices," in *Proceedings of the 20th annual international conference on Mobile computing and networking*. ACM, 2014, pp. 237–248.

[30] J. Ryoo and S. R. Das, "Phase-based ranging of rfid tags with applications to shopping cart localization," in *Proceedings of the 18th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*. ACM, 2015, pp. 245–249.

[31] J. Wang and D. Katabi, "Dude, where's my card?: Rfid positioning that works with multipath and non-line of sight," *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4, pp. 51–62, 2013.

[32] L. Shangguan, Z. Yang, A. X. Liu, Z. Zhou, and Y. Liu, "Relative localization of rfid tags using spatial-temporal phase profiling," in *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)*, 2015, pp. 251–263.

[33] S. Salvador and P. Chan, "Toward accurate dynamic time warping in linear time and space," *Intelligent Data Analysis*, vol. 11, no. 5, pp. 561–580, 2007.

[34] L. Zhou, "Mobile device-to-device video distribution: Theory and application," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 12, no. 3, p. 38, 2016.

[35] S. Knerr, L. Personnaz, and G. Dreyfus, "Single-layer learning revisited: a stepwise procedure for building and training a neural network," in *Neurocomputing*. Springer, 1990, pp. 41–50.

[36] X. Fan, G. Wei, and J. Liu, "i2tag: Rfid mobility and activity identification through intelligent profiling," *IEEE Transactions on Intelligent Systems and Technology*, vol. 28, no. 10, pp. 1553–1567, October 2006.

[37] R. Shea, S. Member, D. Fu, S. Member, J. Liu, and S. Member, "Cloud Gaming : Understanding the Support From Advanced Virtualization and Hardware," vol. 25, no. 12, pp. 2026–2037, 2015.

[38] R. Shea, D. Fu, and J. Liu, "Rhizome: utilizing the public cloud to provide 3d gaming infrastructure," in *Proceedings of the 6th ACM Multimedia Systems Conference*. ACM, 2015, pp. 97–100.

[39] H. Hodson, "Google's ingress game is a gold mine for augmented reality," *New Scientist*, vol. 216, no. 2893, p. 19, 2012.

[40] (2016, 09) Bringing pokémon go to life on google cloud. Google Cloud Platform Blog. [Online]. Available: https://cloudplatform.googleblog.com/2016/09/bringing-Pokemon-GO-to-life-on-Google-Cloud.html

[41] Google edge network. [Online]. Available: https://peering.google.com/\#/infrastructure

[42] Tech. Rep., 2016. [Online]. Available: http://www.goldmansachs.com/our-thinking/pages/technology-driving-innovation-folder/virtual-and-augmented-reality/report.pdf

[43] W. Cai, R. Shea, C.-Y. Huang, K.-T. Chen, J. Liu, V. C. Leung, and C.-H. Hsu, "The future of cloud gaming [point of view]," *Proceedings of the IEEE*, vol. 104, no. 4, pp. 687–691, 2016.

[44] P. Bahl and V. N. Padmanabhan, "Radar: An in-building rf-based user location and tracking system," in *Proceedings of IEEE INFOCOM 2000*, vol. 2. IEEE, 2000, pp. 775–784.

[45] L. M. Ni, Y. Liu, Y. C. Lau, and A. P. Patil, "Landmarc: indoor location sensing using active rfid," *Wireless networks*, vol. 10, no. 6, pp. 701–710, 2004.

[46] Y. Zhao, Y. Liu, and L. M. Ni, "Vire: Active rfid-based localization using virtual reference elimination," in *Proceedings of 2007 International Conference on Parallel Processing (ICPP 2007)*. IEEE, 2007, pp. 56–56.

[47] K. Chintalapudi, A. Padmanabha Iyer, and V. N. Padmanabhan, "Indoor localization without the pain," in *Proceedings of the sixteenth annual international conference on Mobile computing and networking*. ACM, 2010, pp. 173–184.

[48] A. Rai, K. K. Chintalapudi, V. N. Padmanabhan, and R. Sen, "Zee: zero-effort crowdsourcing for indoor localization," in *Proceedings of the 18th annual international conference on Mobile computing and networking*. ACM, 2012, pp. 293–304.

[49] M. Youssef, M. Mah, and A. Agrawala, "Challenges: device-free passive localization for wireless environments," in *Proceedings of the 13th annual ACM international conference on Mobile computing and networking*. ACM, 2007, pp. 222–229.

[50] M. Seifeldin, A. Saeed, A. E. Kosba, A. El-Keyi, and M. Youssef, "Nuzzer: A large-scale device-free passive localization system for wireless environments," *IEEE Transactions on Mobile Computing*, vol. 12, no. 7, pp. 1321–1334, 2013.

[51] X. Li, Y. Zhang, and M. G. Amin, "Multifrequency-based range estimation of rfid tags," in *2009 IEEE International Conference on RFID*. IEEE, 2009, pp. 147–154.

[52] T. Liu, L. Yang, Q. Lin, Y. Guo, and Y. Liu, "Anchor-free backscatter positioning for rfid tags with high accuracy," in *IEEE INFOCOM 2014-IEEE Conference on Computer Communications*. IEEE, 2014, pp. 379–387.

[53] J. Wang, F. Adib, R. Knepper, D. Katabi, and D. Rus, "Rf-compass: robot object manipulation using rfids," in *Proceedings of the 19th annual international conference on Mobile computing & networking*. ACM, 2013, pp. 3–14.

[54] F. Adib and D. Katabi, "See through walls with wifi!" in *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM-SIGCOMM*, vol. 13. Association for Computing Machinery, 2013, pp. 12–16.

[55] A. Parate, M.-C. Chiu, C. Chadowitz, D. Ganesan, and E. Kalogerakis, "Risq: Recognizing smoking gestures with inertial sensors on a wristband," in *Proceedings of the 12th annual international conference on Mobile systems, applications, and services*. ACM, 2014, pp. 149–161.

[56] Y. Wang, J. Liu, Y. Chen, M. Gruteser, J. Yang, and H. Liu, "E-eyes: device-free location-oriented activity identification using fine-grained wifi signatures," in *Proceedings of the 20th annual international conference on Mobile computing and networking*. ACM, 2014, pp. 617–628.