MobyDick: An Interactive Multi-swimmer Exergame

Woohyeok Choi, Jeungmin Oh, Taiwoo Park*, Seongjun Kang
Miri Moon†, Uichin Lee, Inseok Hwang‡, Junehwa Song§
Dept. of Knowledge Service Engineering, KAIST, *Dept. of Media and Information, Michigan State University
†Div. of Web Science and Technology, KAIST, ‡IBM Research Austin, §Dept. of Computer Science, KAIST
{woohyeok.choi, jminoh, sjkang89, uclee, †miri.moon, §junesong}@kaist.ac.kr,
*twp@msu.edu, ‡ihwang@us.ibm.com

Abstract

The unique aquatic nature of swimming makes it very difficult to use social or technical strategies to mitigate the tediousness of monotonous exercises. In this study, we propose MobyDick, a smartphone-based multi-player exergame designed to be used while swimming, in which a team of swimmers collaborate to hunt down a virtual monster. In this paper, we present a novel, holistic game design that takes into account both human factors and technical challenges. Firstly, we perform a comparative analysis of a variety of wireless networking technologies in the aquatic environment and identify various technical constraints on wireless networking. Secondly, we develop a single phone-based inertial and barometric stroke activity recognition system to enable precise, real-time game inputs. Thirdly, we carefully devise a multi-player interaction mode viable in the underwater environment highly limiting the abilities of human communication. Finally, we prototype MobyDick on waterproof off-the-shelf Android phones, and deploy it to real swimming pool environments (n = 8). Our qualitative analysis of user interview data reveals certain unique aspects of multi-player swimming games.

Categories and Subject Descriptors

K.8.0 [**Personal Computing**]: General—*Games*; C.3 [**Special-purpose and Application-based Systems**]: Real-time and embedded systems; C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*

General Terms

Measurement, Human Factors, Experimentation

Keywords

Exertion Games, Swimming, Activity Recognition, Wireless Networking Performance

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.

SenSys'14, November 3–5, 2014, Memphis, TN, USA.
Copyright 2014 ACM 978-1-4503-3143-2/14/11 ...$15.00
http://dx.doi.org/10.1145/2668332.2668352

1 Introduction

Owing to the high dropout rates of fitness activities, Human-Computer Interaction (HCI)-related research on gamifying exercise activities has received significant attention in both academia and industry in recent years. Wii Fit and Kinect Sports are the leading gaming consoles that support various exercise activities, for example, boxing, tennis, and running. Moreover, recent advances in sensor and device technology have spurred the creation of new exergames, which range from transforming existing physical activities into virtual games [3, 16, 24, 40, 51] to enabling remote social interaction among exercisers [33, 34, 35, 40].

While most previous work has focused on ground-based exercise activities, there has recently been an immense interest in examining various water activities. For instance, in the Games4Health workshop at ACM CHI 2013 [9], various gamification scenarios were explored. Pell and Mueller [44] have prototyped Gravity well, an interactive underwater system that enables exertion play under altered gravity conditions. Dungeons and Swimmers [27] is a single-player game where swimmers strokes are used as input for interacting with virtual objects.

The goal of this study is to transform normal swimming activity into an interactive multi-player game. Swimming is a widely enjoyed sport for all, and offers a number of benefits, such as increased cardiovascular endurance and reduced risk of joint injuries. In the past, related swimming studies have mainly been focused on assistive technologies, such as record-keeping and computer-assisted training. For example, SwimMaster [4] analyzes swimming motions by means of several body-attached sensors, in order to provide feedback for swimming posture adjustment. Swimmoid [52] is a mobile underwater robot that follows a swimmer to visually analyzing the swimmer's posture and providing real-time feedback. Davey et al. presented a system [11] that visualizes motion data of swimmers for performance analysis. Several ongoing research projects have investigated how smartphones can be used to enable water activity sensing and user interaction [27, 31]; however, these studies have not provided any detailed information about the sensing mechanisms or the evaluation results obtained.

In this paper, we propose an interactive multi-player game in which a group of swimmers coordinate themselves in order to achieve the common goal of hunting a monster. However, interconnecting the swimmers in an inherently isolated environment need to overcome several key technical chal-

lenges, as well as human factor constraints. For interactive game play, wireless data exchange among users and real-time swimming stroke detection need to be supported. Assuming that the smartphone is attached to the swimmer's arm with an armband (as it is generally the case for outdoor exercises), the smartphone will be periodically immersed in water. However, it has not been established which networking technologies would work, or how well they would perform, under such circumstances. There is a lack of literature focusing on comparative performance analyses of various wireless networking technologies in aquatic environments. Furthermore, previous studies on sensor-based swimming analysis were limited since sensing techniques have mainly been designed for the training of elite or semi-elite athletes. In such cases, motion data, recorded using multiple body-worn motion sensors, are analyzed offline so that feedback can be provided [4, 11]. None of the earlier work examined real-time stroke classification or the impact of skill differences on classification accuracy.

The challenges to transform swimming into a socially interactive game also involve those with human factors and design constraints. Importantly, the in-game multi-player interaction should be carefully designed in a viable form with highly limited visual communication, inability of spoken communication, and reduced sensitivity of auditory communication. Through such careful considerations, we designed *MobyDick*, a team-based game in which a huge underwater monster, *Leviathan*, is hunted. As in popular multiplayer online games, multiple swimmers form a team and play MobyDick collaboratively, attacking Leviathan, evading the attack of Leviathan, and healing the wounded. To this end, MobyDick features the unique mode of interaction by employing team-wide audio-based broadcast of social awareness cues, where each player is encouraged to individually devise autonomous but highly strategic and collaborative game play.

In the design and evaluation of MobyDick, the key research elements include 1) networking performance analysis to understand the non-functional requirements of exergames; 2) real-time swimming activity detection to enable user interactions with the game content (using swimming action as a game controller); and 3) system implementation and a preliminary user study under the autonomous multi-player interaction mode. The key contributions of this paper can be summarized as follows.

- We performed a comparative analysis of networking performance (delay, packet loss, and reconnection) in aquatic environments. We found that periodic blackouts and high packet loss occur when WiFi is used. Swimming strokes have a great impact on networking performance on the whole. Overall, LTE provides fairly consistent performance with a reasonable latency of 50ms (suitable for an interactive game design), despite the fact that it may experience bulk loss and disconnection.
- We built a real-time stroke detection module called *StrokeSense* by leveraging multiple sensors in a smartphone, namely, an accelerometer, a gyroscope, and a barometer. Unlike earlier sensing systems designed for athlete training [4, 11], StrokeSense enables real-time classification of four popular strokes. We concluded that considering differences in individual swimming capability is critical for achieving high classification accuracy, and demonstrated that personalized model training is preferable for our gaming scenario.
- We designed MobyDick and implemented a working prototype by carefully considering earlier findings on networking and stroke sensing, as well as the design constraints of underwater human communication. In particular, we demonstrated that certain technical challenges, such as temporary disconnection, can be overcome with interaction design (for example, by changing a player to a "stunned" state). We performed a preliminary user study (n = 8). Our participants reported that the proposed game was enjoyable, stroke sensing was timely, mappings between stroke types and game actions were intuitive, and the broadcast of social awareness cues provided a uniquely thrilling swimming experience.

2 Networking Performance in the Pool

We analyzed the networking characteristics in the swimming pool and drew implications for designing interactive systems. The environment is unique in that the smartphone is periodically submerged in the water, at which times network performance can degrade, and even worse, network connection can be lost. In this section, we aim to answer the following questions: 1) How do the different networking technologies (WiFi, 3G, and LTE) perform when the smartphone is under water? 2) If the network is disconnected, how long does it take to recover its connection? 3) Will there be significant differences in networking performance for different swimming strokes?

2.1 Experimental Setup

We measured the round trip time (RTT), packet loss rate, and received signal strength (RSS) of WiFi, 3G, and LTE. We used constant bit rate traffic in which a packet was scheduled to be sent in 50 ms. The size of a packet was set to 128 bytes, which is a common packet size observed in online interactive games [14]. After sending a packet, a submerged node waited for an ACK packet of the same size, i.e., simulating periodic game state exchanges between a client and a server. A server was connected via the Internet, and a smartphone client sent/received packets via WiFi or cellular links. In the experiment, we used two waterproof smartphones: the Casio G'zOne Commando LTE and the Samsung Galaxy S4 Active. Both smartphones are advertised as being able to sustain water immersion at 1 meter for 30 minutes.

A server was located at a laptop that was connected to a wired network, with a CPU of 2.5 GHz and memory of 8 GB. For WiFi, we used ipTIME's A2004Plus access point, which supports 802.11b/g/n/ac and has two antennas for each bandwidth, i.e., 2.4 Ghz and 5 Ghz. Furthermore, we considered four protocols for WiFi: 802.11g (2.4 GHz), 802.11n (2.4 GHz), 802.11a/n (5 GHz), and 802.11ac (5 GHz), with two different AP's transmission power settings (100% and 50%). Since the Casio G'zOne Commando did not support 5 GHz operations, we used Galaxy S4 Active for WiFi measurements. For cellular network experiments (3G and LTE), we used two major cellular operators in Korea, namely LG U+, and KT Olleh. Casio G'zOne Commando was used for LG U+, and Galaxy S4 Active was used for KT Olleh.

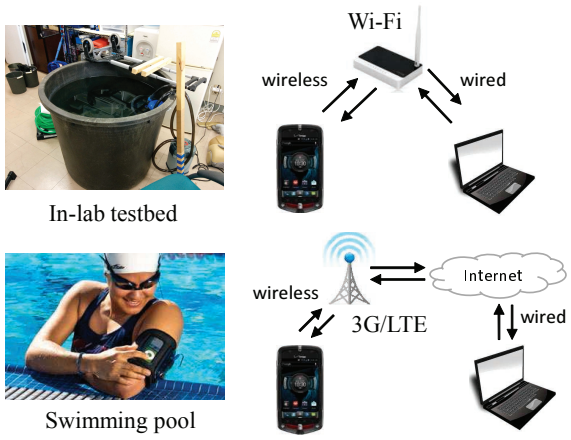


Figure 1: Measurement configuration

We measured the RSS values at the smartphones in dBm. RSS reports are dependent on the characteristics of wireless communication methods. In WiFi, Android periodically reports RSS values. In Wideband Code Division Multiple Access (WCDMA), Android reports Common Pilot Channel (CPICH)'s Received Signal Code Power (RSCP) (dBm), measuring the power level the pilot channel of a cell. In LTE, Android reports Reference Signal Received Power (RSRP) (dBm), measuring the average power of cell-specific reference signals that exist in all downlink sub-frames. For ease of illustration, we simply call these metrics RSS.[1]

To understand performance changes according to different water depths, we varied the depth from 5cm to 20cm in 5cm increments. This reflects the fact that a swimmer's upper arm is typically immersed approximately 10-30 cm into the water. We built an in-lab test bed to simulate a swimming pool, using a large water container with a diameter of 100 cm and depth of 100 cm (see Figure 1). We used water with a chlorine level similar to that of a real swimming pool, as ionized particles of chlorine can affect wireless signal attenuation [55].

We mounted a smartphone onto a wooden rod, which was used to place the phone at a specific depth under water. Since touch sensing does not work under water, we programmed the logging software to automatically start its transmission when the smartphone's pressure level was greater than a given threshold level for 2 seconds. A stream of 2000 consistent bit rate (CBR) packets was sent for each trial, and we repeated this measurement for 10 times for each experimental condition. We also performed a field trial in a real swimming pool with four 25 m lanes. A participant wore a smartphone armband on the upper arm (as shown in Figure 1) and performed four stroke types, namely freestyle, backstroke, breaststroke, and butterfly. The participant swam 100 meters of each stroke.

2.2 Measurement Results

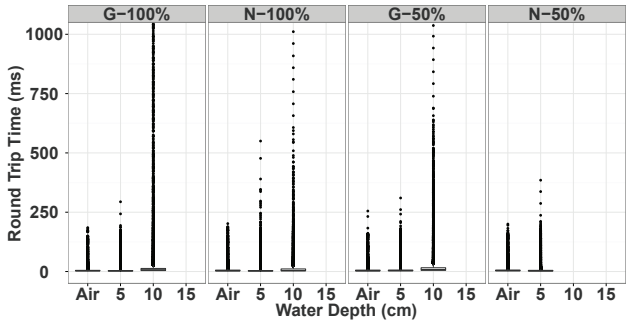
2.2.1 RTT/RSS/Packet Loss Rate Results

We observed that the performance of wireless communication was severely degraded by the water depth. Wireless channel conditions became worse as a device was immersed deeper. Accordingly, the packet loss rate and the variance of RTT values increased. WiFi had the lowest RTT (less than 10 ms), whereas cellular networks had much larger RTT values than WiFi (LTE: 50 ms, 3G: 160 ms), which is in line with the previous measurement studies [19]. The RTT of LTE was much shorter than that of 3G, as it uses millisecond-level scheduling. However, LTE had larger latency values than WiFi, which is possibly due to the multi-hop nature of IP data delivery in cellular network architecture [26].

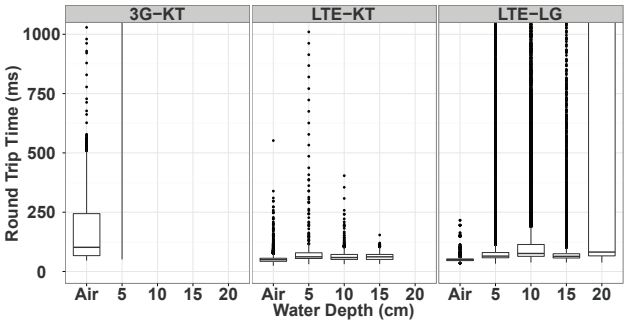
In WiFi measurements, the packet loss rate and the RTT values were not affected when a smartphone was slightly submerged under water (5 cm). There was no packet loss except the case with 802.11g (50% TX power, a depth of 5 cm) that had 1% packet loss. The mean RTT values were ranged from 5 ms to 8 ms (in both Air and 5 cm under water). 802.11g and 802.11n lowered the data rates to cope with degraded channel conditions (median rates of 11 Mbps and 39 Mbps, respectively). However, we observed notable performance degradation at a depth of 10 cm. Furthermore, connectivity was completely lost at a depth of 15 cm. The loss rates went up to 10%, although the data rates were adjusted to 1 Mbps (802.11g) and 13 Mbps (802.11n)—802.11n occasionally set the rate to 1 Mbps, but the median rate was 13 Mbps. Also, lowering TX power in 802.11n had significant impact on connectivity since 802.11n with 50% TX power lost its connectivity at a depth of 10 cm. We hypothesized that these performance differences were mainly due to rate adaption algorithms that vary widely across different protocols and vendors [43]. While the results are not shown due to space limit, *802.11n/ac at 5GHz did not work at all underwater*. In the air measurement, the observed data rates at 5GHz were significantly higher than those at 2.4 Ghz (802.11ac: 433 Mbps, 802.11n: 150 Mbps), and yet, they immediately lost connectivity when they were submerged under water. As explained later in the paper, this is because higher frequency signals are more susceptible to absorption.

In LTE measurements, no significant difference was observed in the air; i.e., the mean RTT values of KT and LG U+ were 49.9 ms and 50.2 ms, respectively. However, we found significant RTT differences under water. KT did not experience any packet loss till a depth of 10 cm, but at a depth of 15 cm, it had 63% packet loss. Its connection was completely lost at a depth of 20 cm. In contrast, LG U+ had around 5% packet loss at a depth of both 5 cm and 10 cm, but the loss rate increased to 50% and 66% at a depth of 15 cm and 20 cm, respectively. KT had consistently lower RTT values than LG U+; KT maintained the mean RTT values around 60 ms, whereas the mean RTT values of LG U+ substantially increased, ranging from 50.2 ms (air) and 170.1 ms (5 cm) to 329.3 ms (15 cm) and 2026.0 ms (20 cm). Also, KT had consistently higher RSS values than LG U+. Such differences may be due to the locations of base stations, and we performed additional in-air measurements in four different sites, all located within one kilo-meter from the original experiment site. Under the sites considered, we found that KT had consistently higher RSS values (with the RSS differences ranging from 2 dBm to 9 dBm) and lower RTT values (with the RTT differences ranging from 4 ms to 7 ms). When we measured the RSS values at a parking lot of the building where we had conducted our experiment, the mean RSS

[1] In WCDMA and LTE, actual RSS calculation should consider the carrier bandwidth; e.g., in LTE, RSRP should be multiplied by the number of sub-carriers.

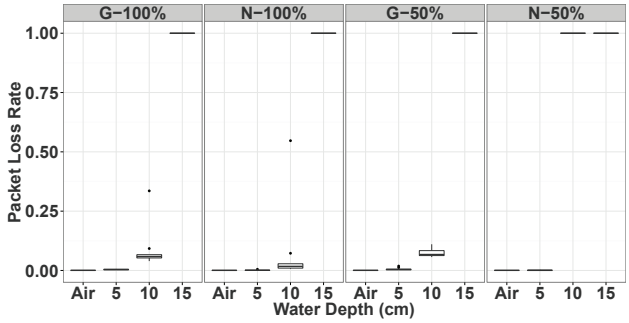


(a) RTT: 802.11g/n at 2.4Ghz; 100%/50% AP TX power

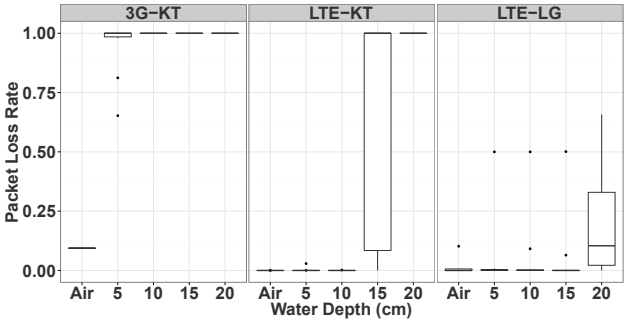


(b) RTT: 3G and LTE (KT and LG U+)

Figure 2: Round trip time (RTT) measurement results

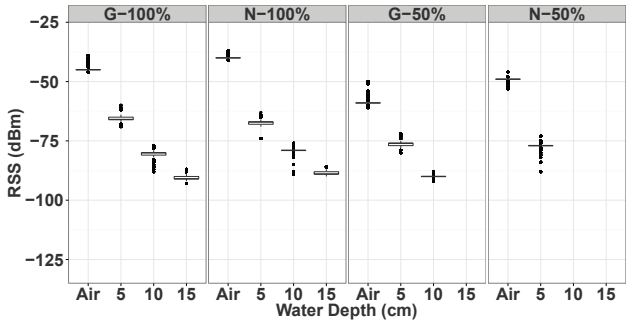


(a) PLR: 802.11g/n at 2.4Ghz; 100%/50% AP TX power

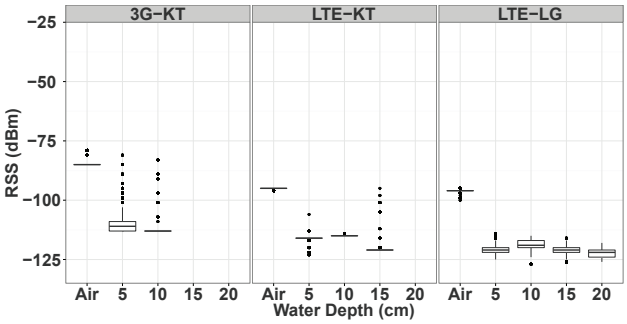


(b) PLR: 3G and LTE (KT and LG U+)

Figure 3: Packet loss rate (PLR) measurement results



(a) RSS: 802.11g/n at 2.4Ghz; 100%/50% AP TX power

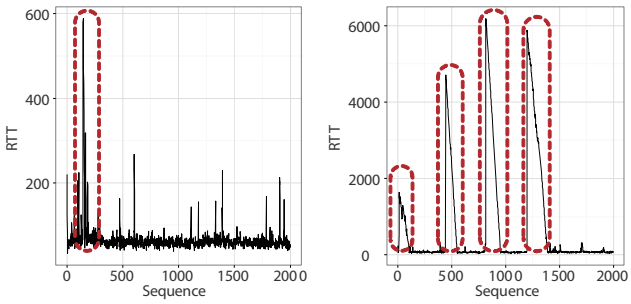


(b) RSS: 3G and LTE (KT and LG U+)

Figure 4: Received signal strength (RSS) measurement results

values of KT and LG U+ were increased to -73.2 dBm and -78.3 dBm, respectively. Those RSS values were substantially greater than those inside the building (approximately 20 dBm). Although significant differences in in-air RTT values were not observed, we expect that network performance under water will be improved outdoors due to better RSS values.

Compared to LTE, 3G Evolved High-Speed Packet Access (HSPA+) did not work well under water, as it failed to sustain connectivity when a device was immersed deeper than 5 cm. There are several possible explanations for this observation. Firstly, LTE typically uses lower frequency bands than 3G HSPA+ (LTE: 800 Mhz vs. 3G: 2 Ghz), and thus, LTE's water absorption coefficient is smaller. According to Beer-Lambert's law, the intensity of a radio wave is exponentially attenuated along its traversing distance, and the absorption coefficient is dependent on the frequency of the radio wave; that is, the higher the frequency, the higher the absorption coefficient [55]. Secondly, LTE uses multiple antennas and its physical modulation is based on Orthogonal Frequency-division Multiple Access (OFDMA). Whereas 3G's WCDMA spreads a signal into a wide band, LTE's uplink uses narrower frequency bands for transmission, and the use of multiple antennas makes communications more reliable and robust. Thirdly, LTE has more robust automatic repeat-request (ARQ) schemes for reliable data delivery (one at the Medium Access Control (MAC) layer, and another at the Radio Link Control (RLC) layer) [26]. While the me-



(a) Water depth of 5 cm (b) Water depth of 15 cm

Figure 5: RTT traces of LTE (LG U+) at two different water levels

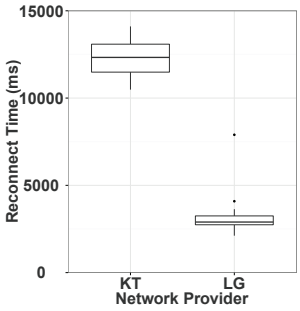


Figure 6: Reconnection time between LTE network providers (KT, LG)

dian values of LTE were not very large, we observed RTTs with values greater than 500 ms in certain trials, particularly when a phone was immersed into water more than 10 cm deep. We manually investigated the RTT and link state traces to find the root cause of this large latency. In Figure 5, we present two plots: 5 cm and 15 cm cases. In both cases, we found that there were sudden spikes in RTT, and then RTT gradually decreased over time. These spikes were immediately followed by a bulk packet loss, ranging from only a few packets to hundreds of packets.

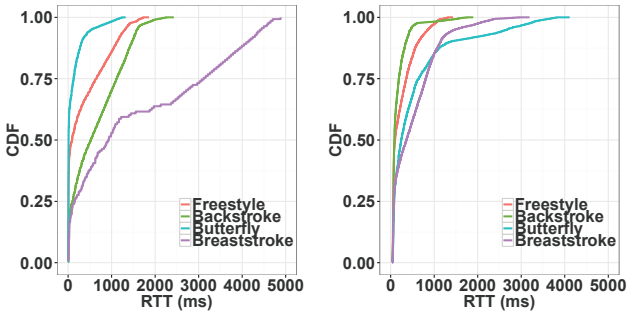
In our experiments, we used UDP packet streams that will be buffered at the UDP transport layer and the LTE link layer. When the channel conditions are excellent, scheduled packets are delivered immediately. However, when the channel conditions worsen (due to signal attenuation and destructive interferences), LTE's link layer will take significantly longer for packet (re-)transmissions. As illustrated earlier, LTE has the dual layers of ARQ [26]. LTE can have eight outstanding link layer packets, each of which is scheduled in a 1 ms subframe. The size of a subframe can be up to tens of kilobytes, depending on the channel bandwidth. When the channel condition becomes very poor, there are many retransmissions, and in the meanwhile, a large number of packets will be backlogged. These packets will then reach their maximum transmission limits and eventually be dropped, thus leading to bulk packet loss. Moreover, since an LTE base station typically uses subframe-level link scheduling, under the poor channel conditions, it may allocate subframes to those nodes with good channel conditions, which makes more packets to be backlogged. In practice, each operator may use different scheduling policies, although proportional fairness algorithms are common.

2.2.2 *Network Reconnection Results*

When a connection is lost, a wireless client attempts to re-establish its wireless connectivity. In the case of swimming games, a node suffers from poor channel conditions, which may lead to network disconnection. Analyzing network reconnection patterns can greatly assist game designers and developers with solving network disconnection problems. Here, we investigate the latency of network connectivity being re-established in WiFi and LTE links.

WiFi reconnection. Reconnection was found to take a long time. The current implementation of 802.11 states that WiFi reconnection follows a series of steps: i.e., AP scan, association, and DHCP. Our analysis of the Android source code revealed that AP scan is scheduled for every 15 seconds. This means that when disconnection takes place, a scanning event may happen after up to 15 seconds. If the scan results contain a preferred WiFi AP, a client automatically attempts to reconnect to that AP. Our measurement results showed that scan, association, and DHCP took 1080.0 ms, 141.1 ms, and 1892 ms, respectively, on average. Here, the major sources of delay were AP scan interval (15 s), scanning time (1.0 s) and DHCP (1.8 s). For fast reconnection, we can handle these two delay sources as follows: whenever disconnection takes place while swimming, we can immediately reconnect to the previously associated AP, by reusing a previously assigned IP address. This method can be implemented by modifying a dynamically loadable module (called a wpa_supplicant), which handles WiFi operations in Android. In our implementation, the modified WiFi client performed scanning immediately after disconnection (instead of waiting for up to 15 s), and DHCP was omitted by reusing a previously assigned IP address. This patch enabled us to reduce reconnection latency to below 1 s in Android.

LTE reconnection. We reported the measurement results for the LTE reconnection latency. We immersed a smartphone into the water (20 cm deep) and waited until it lost network connectivity. The time taken for disconnection to occur varied widely; on average, it was 10–30 s. Upon disconnection, we took the phone out of the water (mimicking swimming strokes). In our measurements, we logged the sensor readings from a barometer to detect when the smartphone client emerged from the water, as there were significant pressure level changes at this point. In our experiment, the average amount of time spent taking the phone out of the water was 1.76 s (SD = 0.25). We found that a new IP address was always assigned after reconnection; in our measurement software, we re-initialized the underlying transport sockets whenever a connection was re-established. In Figure 6, we have plotted reconnection time, which is the interval between the time of lifting and the time of successfully exchanging a packet. There were significant differences between network providers in terms of the reconnection time. The mean reconnection latency of LG U+ was 3.2 s (SD = 1.2), whereas that of KT was 12.3 s (SD 1.0). The maximum reconnection latencies of LG U+ and KT were given as 7.9 s and 14.1 s, respectively. Our data revealed that LG U+ immediately recovered LTE connectivity, whereas KT always re-connected to 3G networks (first Universal Mobile



(a) RTT CDF (WiFi) (b) RTT CDF (LTE)

Figure 7: RTT CDF of WiFi and LTE (LG U+)

Telecommunication System (UMTS), then HSPA+) and continued to maintain 3G connectivity throughout the current session (without switching to LTE). If there were no packet exchanges in KT (after a device was back in the air), however, it then switched back to LTE networks.

2.2.3 Field Trial Results

We measured the RTT values in a real swimming scenario. Our goal was to determine whether the different swimming strokes showed significant performance variations, as they have different levels of water immersion. Figure 7 shows the cumulative distribution of RTT values (LG U+ LTE with Casio G'zOne Commando and 802.11g with Galaxy S4 Active). We found that RTT values were largely influenced by stroke types. In the breaststroke and butterfly, the upper arms are typically submerged more deeply than the other strokes. The results showed that those strokes had higher RTT values and lower signal strength. As in the labaratory experiment, LTE did not have any notable packet loss (typically less than 1% or none) due to its robust retransmission mechanisms. In contrast, 802.11g had much higher loss rates: freestyle (.30), backstroke (.34), butterfly (.51), and breaststroke (.73). Furthermore, 802.11g's RTT values were significantly increased: freestyle (M: 353.3 ms, SD: 458.8), backstroke (M: 640.9 ms, SD: 568.3), butterfly (M: 110.1 ms, SD: 212.2), and breaststroke (M: 1600.0 ms, SD: 1596.3). When compared to RTT distribution of LTE, the high packet loss rate of WiFi made considerable influence on the overall RTT distribution. In our measurements, we did not explore personal characteristics on networking performance (e.g., skill level and stroke habits), but we expect that such characteristics may have significant impact on overall performance.

2.3 Discussion: Game Design Implications

We studied the networking performance of popular wireless technologies, namely WiFi, 3G, and LTE in the laboratory, as well as in actual swimming scenarios. Our results confirm that water immersion has a significant effect on end-to-end latency. We found that LTE was more robust than the other wireless networking technologies, such as WiFi and 3G, under the conditions we considered. Although WiFi had much shorter RTT values (typically less than 10 ms), our field trials showed that it suffered from periodic blackouts and high packet loss rates. Water immersion patterns differed widely across different stroke types while swimming, which contributed to the RTT variations. Despite the robustness of LTE links, when poor channel conditions were prolonged, there could be a bulk packet loss; however, the likelihood of such incidents occurring was much less than with WiFi, as confirmed in the field trials. When network disconnections happen, we found that it took considerable amount of time for re-establishing connectivity.

In the game design, we should carefully consider end-to-end latency, reliable transport, and network connectivity issues when designing game content and implementing inter-device communications. For example, the RTT constraints will help the designers to understand the degree of interactivity that a given networking technology can provide and will guide them to design game content accordingly. While typical data exchanges are done in UDP for game development, some important game states (e.g. health states) should be reliably delivered to the participants. The data transfer layer of the game client/server should consider packet loss patterns. Furthermore, in swimming scenarios, connectivity loss may happen occasionally, which is quite rare in in-air networks. Likewise, the designer should consider how to handle such a long period of connectivity loss within a game.

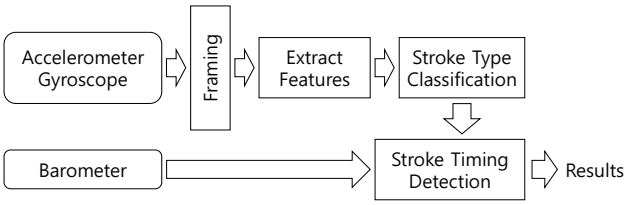


Figure 8: StrokeSense's data processing sequence

3 Swimming Stroke Recognition

In this section, we begin with an overview and the requirements of our swimming action recognition module. We then explain the kinematics of swimming strokes to illustrate our real-time stroke recognition algorithms. Furthermore, we outline the architecture of *StrokeSense*, the proposed swimming style and timing recognition system, and we elaborate on the detailed algorithms, i.e., stroke type classification and stroke timing detection.

3.1 Overview: StrokeSense

StrokeSense recognizes two types of swimming action information, namely swimming style and stroke timing. When a game begins, sensor data from the onboard accelerometer, gyroscope, and barometer are segmented into a window of 2 seconds. Taking existing work on activity classification and transport mode detection [5, 46] into account, we used a small window size for the timely recognition of stroke events, so that the classification results could readily be used for user interaction during game play. StrokeSense then classifies the current stroke type every 0.5 seconds, using calculated features from the segmented three-axis accelerometer and gyroscope data. At the same time, a swimmer's arm pull action is detected by means of peak detection technique, using barometer signals. When an arm pull action is detected, StrokeSense reports the recently classified swimming style and the timing of the arm pull occurrence to the game logic. The overall processing sequence is shown in Figure 8.

Taking multi-party communication and real-time game interaction, StrokeSense is designed to satisfy the following two requirements.

- A user's smartphone is secured on the upper arm using a waterproof armband, in order to enable multi-user collaboration via wireless network. There could be other positions such as the back, abdomen, and wrist. The back/abdomen is unsuitable since it is completely under water in certain stroke types: the back is under water during backstroke, while the abdomen is under water during the other stroke types. The wrist are repeatedly emerge from the water with any strokes; however, headset wires should be tightened along the arm, in order not to interfere with swimming strokes. Therefore, in our prototype, we chose the upper arm for placement of the smartphone on the candidates.
- StrokeSense should provide highly accurate recognition of the four stroke types and real-time stroke timing detection, since stroke types and timing are used as the game inputs. Prior studies limited to off-line sensor data processing for stroke type recognition [22, 31, 48]. Furthermore, none of these studies considered the upper arm as the smartphone location. StrokeSense complements the prior studies in that we detail real-time algorithms, report comprehensive evaluation results, and demonstrate real implementation in a multi-player game [22, 27, 31, 48]. The key challenge would be 1) exploring various smartphone sensors, and finding a set of appropriate features and classification models which best discriminate stroke types, and 2) designing a robust algorithm for stroke-timing detection to provide sufficient interactivity.

3.2 Swimming Kinematics

We considered four popular strokes, each of which defines a particular swimming style, namely, freestyle (front crawl), breaststroke, butterfly, and backstroke (back crawl). These strokes involve rhythmic movements of major body parts, i.e., the torso, arms, legs, hands, feet, and head. Casual swimmers typically complete laps using various stroke types to meet their exercise needs. They perform a series of strokes followed by turning at the end of the poll, thus repeatedly alternating between the *stroke period* and the *turning period* by changing directions.

We now briefly explain the kinematics of stroke types to illustrate our stroke recognition algorithms. Basically, each stroke can be divided into a sequence of phases, and generally begins with an arm pulling activity (see Figure 9). In freestyle and backstroke, a left arm stroke is followed by a right arm stroke. A left arm stroke, for example, consists of two phases: left pull and right recovery, and left push and right recovery. A butterfly stroke starts with arm pulling and then pushing with leg kicking. This is followed by the arm recovery and flying phases. Similarly, in a breaststroke, an arm pulling phase is followed by an elbow gathering phase. After the arm recovery and leg flexion phases, the leg kicking and gliding phases are repeated. In the case of turns, there are several styles, such as open and flip turns, but they all typically involve the following phase sequence: turning/rotation, pushing, gliding, underwater kicking, and pull-out.

3.3 Sensor Selection and Data Collection

We first explored which sensors are used for data collection. As in the previous studies, we used motion sensors such as an accelerometer and a gyroscope [22, 27, 31, 48, 50].

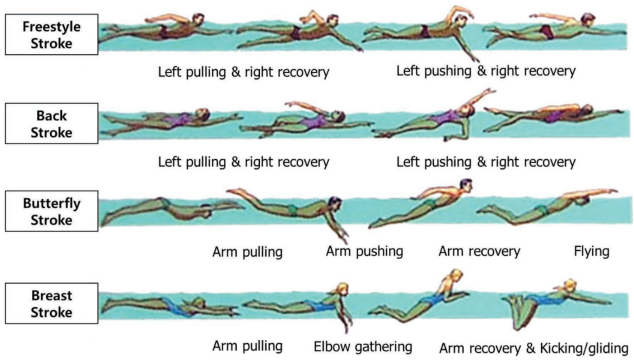


Figure 9: Illustration of stroke phase sequences. Arm pulling is common in all four strokes.

Furthermore, we considered a barometer to measure ambient pressure, which consists of the hydrostatic pressure with the weight of the water column and the atmospheric pressure on the free surface. The ambient pressure significantly changes according to the depth of the water, compared to the height in the air. For example, the change in the ambient pressure of 1 hPa requires the change in depth of only 0.01 m under water; this change can be achieved by altering the height of 7.9 m in the air. Because sensing error of the barometer are small (less than 0.2 hPa) [36], it can be used as a valuable information source for activity recognition underwater. However, we found that the ambient pressure is not appropriate for stroke classification due to its dynamics; i.e., the pressure value changes significantly under different contexts such as water density, body motion, weather/atmospheric conditions and time-of-day [53]. Even if we were to use the relative differences of pressure between above and under the water, the dynamics must be accurately captured in the pre-processing step, which may be error-prone. Nonetheless, we later show that the barometer can provide useful information for stroke-timing detection due to the key properties of ambient pressure, i.e., highly periodic signals (synchronized with each stroke) with consistent patterns across different stroke types and swimmers.

For data collection, we employed the CASIO G'zOne CA-201L, a rugged Android smartphone with OS version 4.0.3. The smartphone supported LTE mobile network and included IPX5/X8 water resistance as well as various sensors such as an accelerometer, a gyroscope, a compass and a barometer.

We recruited 11 participants (1 female and 10 males), who were between 19 and 26 years old from a campus swimming club. The swimming pool used for the experiments had four lanes of 25 m in length, and a depth ranging from 1.3 m to 1.7 m. Participants were asked to place the smartphone on their upper arm using an armband and to swim two round trips for each stroke type, i.e., eight round trips in total. While they swam, we collected 3-axis accelerometer, 3-axis gyroscope, and barometer data, as well as recording video clips for ground truth. After data collection, we used the recorded video clips to carefully annotate activity tags for the following movements: *no movement, turn, freestyle, butterfly, backstroke*, and *breaststroke*.

Table 1: Features used in swimming style classification

Measurement	Features
AccelX, AccelY, AccelZ, AccelMag, GyroX, GyroY, GyroZ, GyroMag	Min, Max, Mean, Variance

Table 2: Selected features from CFS and their information gain (IG) scores

Feature	IG score	Feature	IG score
accely_mean	1.508	gyroy_min	0.699
accely_max	0.922	accely_variance	0.634
gyrox_mean	0.886	accely_min	0.632
gyrox_min	0.857	accelmag_variance	0.571
gyroy_max	0.778	gyromag_variance	0.543
accelz_min	0.739	gyroy_mean	0.482
gyrox_max	0.722	accelz_mean	0.321
accelx_min	0.721	accelx_mean	0.159

3.4 Stroke Type Classification

3.4.1 Preprocessing and Feature Calculation

StrokeSense classified the four stroke types by using 3-axis accelerometer and gyroscope data. The sampling frequency of the accelerometer and gyroscope data was 100 Hz, obtained by selecting SENSOR_DELAY_FASTEST setting in the Android SensorManager class. To suppress high-frequency noises, sensor data was smoothed by averaging the five most recent samples. They were then divided into 2-second windows, with a slide of 0.5 seconds, so that two consecutive windows had an overlap of 1.5 seconds. We decided the length of a window by considering the normal period of stroke repetitions in swimming. In most cases, the time window contains one full repetition of a stroke for all swimming styles.

In addition to the 3-axial sensor data in each case, we used the magnitude by calculating the root-mean-square of the values from all three axes. Each of these four types of data, i.e., X, Y, Z, and magnitude, was used to calculate features such as minimum, maximum, mean, and variance from 2-second window. In total, we extracted 32 features from the two sensors (see Table 1).

3.4.2 Feature Selection

In order to remove irrelevant or redundant features, we ran a Correlation Feature Selection (CFS) algorithm for feature selection by using the Weka Machine Learning Toolkit 3.7 [17]. CFS determines a subset of features with the highest predictive power, while reducing redundancy among the features themselves [15, 56]. We extracted a set of 16 features from the initial 32s by using the CFS algorithm, and calculated information gain for each, which is a measurement for estimating the feature's discriminative quality [21] (see Table 2. Furthermore, in order to minimize potential over-fitting issues, we attempted to reduce the number of features carefully, while maintaining classification accuracy, i.e., the rate of correct predictions made by the model over a data set. We evaluated the accuracy of reduced feature sets with the top 4, top 8, and top 12 features in terms of the information gain values.

3.4.3 Classification Models and Evaluation

We considered the following well-known classification algorithms for stroke type classification: the Decision Tree (DT), Naive Bayes (NB), and Support Vector Machine (SVM). Readers can find more details about these algorithms in Witten et al. [54]. For DT, the C4.5 algorithm was used. For the SVM, we used linear and radial basis function (RBF) kernels for building models.

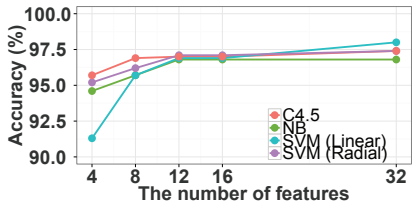


Figure 10: Accuracies of stroke type classification against the number of features

We first trained each classification model using the top 4, top 8, top 12, top 16, and all 32 features from all of the users' data, and validated the performance using 10-fold cross-validation. As shown in Figure 10, in the case of the top 4 features, all classifiers except for the SVM with the linear kernel showed accuracies of over 94%. The cases with the top 12, top 16 and all 32 features showed accuracies of over 96%. The SVM with the linear kernel showed the greatest increase in accuracy, as the number of features increased. Its accuracy was under 92% for the top 4 features, but it achieved the best performance, which was approximately 98%, among the classifiers with all 32 features. We conjectured that the this was due to over-fitting, so there may be a potential challenge in actual applications, which necessitates further studies.

The results show that our system is as effective as, and in some cases outperforms, those of previous methods of swimming style recognition. In [48], a quadratic classifier using a wrist-worn accelerometer achieved 89.8% accuracy, and one using an upper back-worn accelerometer showed an accuracy of 95.3%. We conjecture that the reasons for the higher accuracy of our results is the use of a gyroscope sensor. It is known that a gyroscope is more effective than an accelerometer in monitoring body movements, as most body movements involve joint rotations [41]. For the following evaluations, we used the SVM with the linear kernel as the classification model, based on its code portability and computation efficiency.

3.4.4 User Variance

Since user differences have been taken into account in previous studies on activity classification, it is logical to examine the issue in the context of swimming. As with other human physical activities, swimming involves various movements of almost all of the muscles and joints in the human body, which means that there is great potential for user differences in actual situations. Therefore, we investigated the differences in swimming motions among users by using a similar approach to that of [46]. We built two model categories: *user-specific models* and *leave-one-user-out models*. For the user-specific models, we used one user's data at a time for training and testing. Each model built was tested by 10-fold cross-validation. For the leave-one-user-out models, the classifier was trained with the data of all participants except for one, and tested with the data of the participant who was not in the training dataset.

As shown in Table 3, user-specific models showed an

Table 3: Accuracy comparison between user-specific and leave-one-user-out models. (Values are in %)

Participant	User-specific	Leave-one-user-out	Diff.
P1	98.51	94.67	-3.84
P2	98.36	97.41	-0.95
P3	98.65	96.82	-1.83
P4	99.1	98.08	-1.02
P5	97.53	90.89	-6.64
P6	98.5	86.93	-11.57
P7	97.75	96.87	-0.88
P8	90.22	87.31	-2.91
P9	99.61	91.7	-7.91
P10	96.09	88.54	-7.55
P11	97.97	95.27	-2.7
Avg.	97.48	93.14	-4.35

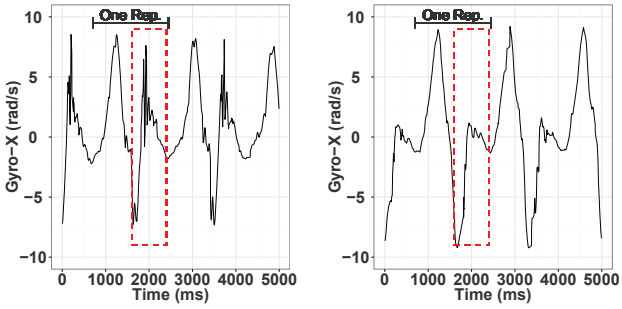
average classification accuracy of 97.48%. For the leave-one-user-out models, the average accuracy was 93.14%, and the minimum per-participant accuracy was 86.93%. We furthermore analyzed the confusion matrices of the participants whose data showed relatively low accuracies, namely, P5, P6, P9, and P10. We observed two classes of the confusion patterns: confusion between butterfly and freestyle (P5, P6) and between backstroke and breaststroke (P9, P10). We conjectured that the reason for the confusions was the similarity of arm movements in the swimming styles; for example, butterfly and freestyle sometimes showed similar crawling arm movements.

3.4.5 *Differences between Skilled Swimmers and Novices*

We conducted a pilot test with four additional participants in order to verify the performance of the best classification model. The four participants (4 male) were university students who were capable of performing all four stroke types, and their age ranged from 26 to 36. The participants occasionally swam, but none of them regularly swam on a weekly basis. This means their swimming skill was expected to be lower when compared to the casual swimmers with regular training. We collected their swimming motion data in a similar manner as shown in Section 3.3, and we classified stroke types using the best classification model. Surprisingly, the accuracy of the models was very low, at times even under 50%.

We investigated the potential causes of the accuracy degradation. We first examined whether our models were over-fitted to the training dataset. Since feature selection can reduce the possibility of over-fitting, we tested our models with fewer features, but this did not improve the accuracy.

We then examined the confusion matrices. Our model classified freestyle and backstroke well; however, breaststroke and butterfly showed two different confusion patterns: failure to distinguish freestyle from butterfly and backstroke from breaststroke in one group, and failure to distinguish backstroke from both butterfly and breaststroke in another. At this point, we realized that the classification accuracy may be dependent on the difficulty of swimming styles, as freestyle and backstroke are generally easier to perform well than breaststroke and butterfly. We collected the training data from a campus swimming club, and our interviews with the club members informed us that they usually participated in swimming drills several times a week. While club members' skill levels were rather diverse (particularly among our participants), their swimming skills were relatively high, and motion patterns tended to be fairly consistent to one another, due to the regular swimming drills. However, the additional participants swam as a hobby and thus can be regarded as less skilled swimmers than the club members.



(a) Less trained casual swimmer (b) Highly trained amateur

Figure 11: Differences of motion sensor signal between less trained casual swimmers and highly trained amateurs during butterfly

Table 4: Accuracy of pilot test using a highly trained amateur model and a user-specific model

Participant	Highly trained amateur	User-specific
A.P1	69%	100%
A.P2	50%	99%
A.P3	48%	95%
A.P4	53%	98%

To confirm our intuition, which is the classification accuracy may be dependent on the difficulty of swimming styles, we manually examined the motion dataset (i.e., the acceleration and gyroscope samples) in order to determine why such classification errors occurred. After inspecting the dataset, we found that the patterns of the acceleration and rotation curves appeared somewhat different to those of the training dataset (see Figure 11). In the figure, the red-dotted circles represent the difference in butterfly motion, specifically in dragging the arms. While swimmers are dragging up their arms, the arms of highly trained amateurs rotate more; that is approximately 90 degrees, to the front, than those of less skilled swimmers. While the original training data appeared to be highly consistent across different users, the dataset from the pilot study did not clearly exhibit such patterns.

Table 4 shows the classification results of the less trained casual swimmers using the model based on 11 highly trained amateurs, as well as the user-specific models, which use the same swimmer's data for training and testing. From the results, it appears to be possible to apply a user-independent classification model for highly trained amateurs, but not for less trained casual swimmers. Therefore, we surmised that user-specific models can cover larger swimmer groups of swimmers, including less trained casual swimmers. Assuming that a user's skill does not significantly change in a short time, collecting a dataset and building models can be occasionally done (say once in a few months). To lower swimmers' burden for data collection, an semi-automatic collection method using an audio guide would be useful; e.g. a

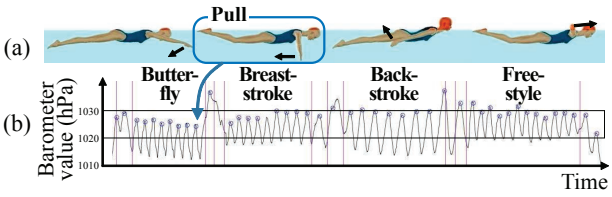


Figure 12: Description of stroke timing detection, (a) pull action of butterfly (b) example of barometer signals

mobile application tells a user to collect sensor data by performing one lap for each stroke—this level of manual data collection will not be burdensome for casual swimmers. An alternative would be to employ a hybrid technique, where a user-independent model is initially used, and individual user data is combined via adaptive learning.

3.5 Stroke Timing Detection

3.5.1 Robustness of the Barometer Signals

StrokeSense detects stroke timings in real time by analyzing peaks in barometer signals as well as their values and temporal sequences, and reports recently classified strokes, with stroke timing, to the game logic. We employed barometer signals because they are: 1) highly periodic along with every stroke, and 2) highly robust against stroke type and swimmer-specific differences.

Figure 12(b) clearly shows the periodic barometer signal during swimming. For every single stroke, the swimmer's arm completes a circle, drawing a semicircle in the air and the other semicircle under water. Such periodic surficial transitions create clear periodic cycles in barometer signals. As mentioned in Section 3.3, diving by every centimeter depth increases the barometer output by 1 hPa. We found that in most cases the local maxima of the barometer signal corresponded to the moments when the swimmer was pulling the water strongly at the middle point of the underwater semicircle. Interestingly, this is when the swimmers exert a strong muscular force on their arm, so that we can leverage this as a highly intuitive timing to provide auditory feedback for each stroke.

Also, barometer signals yield very little differences between stroke types when compared to motion sensors such as the accelerometer and the gyroscope. As shown in Figure 12(b), the major reason for this robustness is that the signals are closely related to the water depth, and detailed motions do not lead to significant changes in the signals. Thus, regardless of the current stroke type of a swimmer, it is possible to apply the same stroke-timing detection method. This method is also robust against personal differences in swimming motion; for example, the angle of the arms and acceleration rates. It is worth noting that stroke timing detection methods based on motion sensors [48] require specific heuristics for each stroke type, thereby requiring prior knowledge of the current stroke types. In this case, if the stroke type classification fails, the motion-based timing detection also fails.

3.5.2 Timing Detection Algorithm Design

Barometer data were first smoothed by averaging the most recent five samples to suppress high-frequency noises. In our experimental device, CA-201L, with the SENSOR_DELAY_FASTEST setting in the Android SensorManager class, barometer signals were sampled at 33 Hz, which is fast enough to provide rapid action detection. From consecutive smoothed signals, local maximum and minimum peaks were found, and every significant local maximum peak was reported as a stroke timing, as shown in Figure 12(b). The significance was determined by establishing whether the value difference between the recent local maximum and minimum peaks was above a certain threshold (i.e., the pressure threshold). Furthermore, we ignored other significant peaks for a certain amount of time (i.e., the time threshold) if previously significant peaks were detected. This method is based on the intuition that it takes a certain amount of time to complete each stroke. We empirically set the pressure threshold to 2.5 hPa and the time threshold to 1 second, which yielded a sufficient detection performance for playing the game. In particular, the pressure threshold of 2.5 hPa was large enough that we could ignore the 0.2 hPa error mentioned in the prior work [36]. To the best of our knowledge, this was one of the earliest attempts that employed a barometer for motion detection.

Table 5: Stroke-timing detection result (TP: True Positive, FP: False Positive, TN: True Negative) for each swimming styles

Swimming Style	Ground Truth	TP	FP	TN
Freestyle	328	324	4	-
Backstroke	346	341	5	1
Breaststroke	435	378	55	11
Butterfly	359	353	6	-

3.5.3 Timing Detection Algorithm Evaluation

We evaluated the performance of the proposed stroke-timing detection method by using the pre-collected dataset described in Section 3.3. We used the manually ground-tagged stroke-timing data obtained from videos. As shown in Table 5, the detection results are comparable with the state-of-the-art stroke counter described in [48]. Again, it should be noted that StrokeSense uses one common stroke-timing detection algorithm, whereas [48] relies on multiple swimming-style-specific timing detection algorithms. In the game experience study, we also used the module for extracting stroke timing to give participants immediate feedback and to utilize the timing information as a meaningful game input.

4 MobyDick: Game Design

We designed *MobyDick* taking into account key human factors and technical challenges when transforming swimming activities into game play. To put it briefly, MobyDick is a team-based hunting game against a huge underwater monster, *Leviathan*. As with popular multi-player online games, multiple swimmers form a team and play MobyDick collaboratively, attacking Leviathan, evading the attack of Leviathan, and healing the wounded. Below, we outline the game modalities and design.

4.1 Swimmer-to-Game Interaction Modalities

Game inputs. Swimming is a full-body exercise requiring highly coordinated and continuous motions of all four limbs. Therefore, swimming allows for only very small degrees of freedom when creating game inputs within the existing activity, and common strategies adopted in ground-

based exergames are inapplicable. Dynamically changing paces [35, 40] is not as responsive as on the ground and may unbalance the swimmer. There are few free body parts to use during swimming, as with punching while running [3]. As a result, we focused on the intrinsic stroke activities of swimming. We utilized two types of swimming action information as game inputs: stroke timing and swimming style (*freestyle, breaststroke, backstroke, and butterfly*). MobyDick senses stroke activities by means of the smartphone mounted on the swimmer's upper arm with an anti-friction armband. Other body locations (e.g. the back and wrist) were ruled out due to technical challenges (wireless connectivity) and swimmers' discomfort (wired earphones).

Game outputs. We identified auditory output to be an appropriate choice for a smartphone-based swimming game, as we could simply use waterproof wired earphones. Note that alternative modalities, such as visual ones, would require special devices like display-embedded goggles. We built a type of an audio-only game [57], which are unlike typical video games that use audio merely to provide additional information [39]. MobyDick continuously updates the game progress, team members' status, and interaction events through background music, narration, and sound icons.

4.2 Game Design

Currently, MobyDick is designed to support four swimmers fighting together against Leviathan. Each stroke type corresponds to a distinct action category in the game: freestyle is for attack, breaststroke is for evasion, and backstroke is for healing, while butterfly is for critical attack due to the higher level of mastery required for this stroke. Each stroke constitutes executing a single action of the category corresponding to the particular stroke type. To increase user interactivity, we map each stroke with its own sound icon (or "earcon"); i.e., two attack earcons for freestyle and butterfly, one evasion earcon for breaststroke, and one healing earcon for backstroke. A player's stroke events are broadcast in the background so that players have a general awareness of other players' presence (the volume of the sound from other players is lower than a user's own sound). Furthermore, there is an in-game narrator who continually describes game progress, team members' statuses, and interaction events, for example, health point (HP) status, as well as who is currently attacked by Leviathan. Unlike existing exergames that typically used verbal communication to facilitate social interaction, such as Jogging over a distance [35] and ExerLink [40], MobyDick employs various social awareness cues, such as the narrative of the virtual avatar interactions and the stroke earcons of players.

Game flow. One round of the game lasts for three minutes, a length of time that was chosen considering the average casual swimmers' stamina; that is, how long they are able to keep swimming for at one time. For each 30-second turn, Leviathan randomly chooses a swimmer and breathes fire at the swimmer in a constant cycle. While the victim is swimming breaststroke to evade the attack of the monster, the other swimmers are free to attack Leviathan by using freestyle or butterfly. To evade the fire breaths, the victim should synchronize breaststroke cycles with the firing cycles in order to be under the surface at the very moment that the fire is breathed out; otherwise, the victim's health points were deducted. When the victims have "died", that is their health points have reached zero or lower, they can revive themselves by swimming backstroke for a certain number of strokes. The other non-victim players may participate in helping the dead player to revive him/her more quickly, by swimming backstroke together. The team wins if all of Leviathan's health points are depleted within 3 minutes. They lose if either the time is up or everyone is "dead".

Multi-player collaboration with social awareness cues. MobyDick features the unique mode of asymmetric multiplayer communication for collaborative game play, which is different from those in two-way verbal communications between players in collaborative exercise games [40] and nonverbal communications in 3D virtual environments [30]. In MobyDick, each swimmer is given no means of *explicit outbound* communication, either verbal or nonverbal. The only information available to the game participants is the team-wide audio broadcast, delivering the actions and status of Leviathan and each swimmer (designated by a unique call sign, namely, Alpha, Bravo, Charlie, or Delta)—this audio broadcast works as social awareness cues [13]. When listening to the events and progress along the game play, each swimmer is expected to perform strategic actions based on their own decision. For example, a swimmer, say Delta, is under the concentrated attack from Leviathan, having her life at stake. Having heard of this status, the swimmer, Alpha, may make a strategic decision on his own. He can continue freestyle to deplete the slightly remaining Leviathan's health points, or alternatively switch to backstroke to keep Delta alive as she is the most proficient butterfly swimmer and thus, is the most important attacker of the entire team. Due to lack of explicit outbound communication, we expected that such *silent and autonomous* teamwork can alternatively provide social fun, and players will be able to devise better strategies as they figure out each other's play style and swimming competence.

Latency-aware game design. As illustrated in Section 3, users may face network connectivity loss and be unable to exchange any status update messages, which are critical for interactive game play. We therefore devised several design choices to yield user experiences that are less sensitive to connectivity loss and long latency issues.

Firstly, we deliberately hid the accurate number of remaining health points of Leviathan and other swimmers, and represented them only in quartiles, for example: "*Leviathan has less than 75% H.P.!*" This technique decreases the frequency of received status update messages from others and increases players' tension and immersion as in some video games [1]. Secondly, a MobyDick client *locally* computes one's own status change, e.g., own death or revival, and immediately notifies the swimmer of it, and remotely synchronizes the server (as is the case in most interactive games). It would be annoying for the swimmer to notice latency in their own obvious status changes. Thirdly, a swimmer might temporarily lose connection that is not restored immediately. We observed that such an incident is not highly likely, but may occur during breaststroke or butterfly. In order to keep players in the game in spite of disconnection, their MobyDick clients tells them that they have been "stunned" and recommends that they use freestyle or backstroke for rapid recovery.

5 Preliminary Evaluation

We conducted the experiments on MobyDick with a group of swimmers in an actual swimming pool. In the user study, we investigated the efficacy of our game design mechanics and socially enriched swimming experiences.

5.1 MobyDick Implementation

The MobyDick implementation includes the following modules: StrokeSense, game logic, and communication. We implemented the StrokeSense module in Android (v4.0.3 API 15) using an SVM library called liblinear-1.94. For personalized model building, a new user initially follows an audio guide whereby they are asked to swim a certain stroke for a given lap.

The game logic module manages the overall game flow through the game handler, and user interactions through the interaction manager. The game handler receives user input from StrokeSense and tracks the current game status, i.e., attacking, evading, Leviathan's status. It also implements the details in overall game flows. Whenever the interaction manager receives status changes, it delivers aural information, namely game instruction, one's own input, and others' input, to the user. It also provides the user interfaces for game manipulation (i.e., start/stop games). This includes disabling screen touching events while users are swimming, as water is sensed by the capacitive screen. A client's current status message is reported to the server every 100 ms; and the server broadcasts state changes to all the group members.

The communication module manages message exchanges. Since MobyDick requires the timely delivery of small control packets, it uses the UDP transport protocol, which is typically used in online game design. For each client-server pair, the communication module implements reliable UDP packet transmission schemes. Depending on the importance of the control packets, we divided the state messages into low and high priority ones. High-priority messages are used to report critical status changes such as the death of Leviathan/player or the end of a game, whereas low-priority ones are used to report other player activities, such as attack and healing. To achieve reliable transmission, if a node fails to receive an ACK in 50ms, a message will be retransmitted. Low-priority messages expire after 2 seconds, while high-priority messages do not have a time limit at which they expire. It monitors wireless network connectivity: when previously lost connectivity is detected, it informs the game logic of such an event. Also, it keeps track of a client's IP address to handle IP address change after reconnection.

5.2 User Study Design

We recruited eight participants from online communities, with the major criteria being: 1) the participants must have had more than one year of swimming experience, 2) they must swim more than three times a week, and 3) they must be capable of performing all four stroke types. Table 6 lists the participants' demographic information. We conducted the study in a 25-meter, five-lane swimming pool at the local university gym. The participants were divided into two four-member teams. Each team played two rounds of games, with 10 minutes of intermission in between. Each lane was exclusively used by a single swimmer at a time. For exit interviews, we conducted focus group interviews followed by 1.5-hour one-on-one interviews directly after the games. The video and audio of all the interviews were recorded. Two researchers transcribed and coded the interviews. We now discuss the major themes and findings of the study.

Table 6: List of participants

Team	ID	Age	Gender	Experience
1	P1	23	F	2 years
	P2	20	F	4 years
	P3	22	M	3 years
	P4	23	M	5 years
2	P5	20	M	1 years
	P6	20	M	1 years
	P7	24	M	5 years
	P8	22	M	2 years

5.3 Findings

Dissociation from intrinsic swimming activity. All of the participants reported that the swimming game was enjoyable. Specifically, we observed that most of the participants experienced certain degrees of dissociation from the activity of swimming. P1 stated the following: *"(When swimming without the game) usually it's kind of boring. (...) My mind wanders away but it often gets me out of balance. (...) [Playing the game] definitely took such feelings away from me."*

Literature reports that one's dissociative state effectively reduces one's perceived exertion [45] and leads to a mental disconnection from painful sensory inputs [32]. However, one participant (P6) commented that paying excessive attention to the game resulted in a higher level of exertion than usual experiences: *"I used to pace myself to keep it mild, but this time it was just way faster."*

Perceiving timely game responses. We found that six of the participants enjoyed the immediate auditory feedback received from the game. In particular, P2 liked the exact timing: *"I enjoyed the sound of clashing swords; I could hear them right at the moment I pull the water. (...) I really feel like my stroke is wielding the sword."* The participants said that they could hear the sounds loudly and clearly. This may be attributed to our design choice of setting the sound timing not to be at the moment that swimmers' arms hit the surface, but rather when they pull the water, thus preventing a water splash noise.

Keeping track of game status updates. MobyDick delivers extensive game-related information through aural messages. We found that, when beginning a game, the swimmers could keep track of most information, such as game progress, the remaining health points of themselves and Leviathan, other swimmers' statuses, etc. However, towards the end of a game, the participants tended to be exhausted and not have a sufficient cognitive span to keep track of others' activities. P5 recalled: *"(At the last part) I couldn't listen to what others are doing. Just I keep hitting [Leviathan]."*

Intuitive behavioral metaphor. The participants liked the mappings between stroke types and in-game action. P7 stated: *"It makes good sense to me. (...) When I freestyle or butterfly, it seems like hitting the water. (...) Backstroke seems like lying on the water and taking some rest."*

Socially enriched swimming experience. Even though MobyDick does not change anything in the sense that the swimmers cannot speak or express themselves at all to the

other team members, we found that the unique mode of collaboration in MobyDick may create new feelings of bond and team spirit among swimmers, e.g., real-time interaction and sympathy with each other. P7 commented: *"It cheers me up to hear that [Alpha and Bravo] are attacking when I am attacking too. I feel we are the same team indeed"*. Also, playing the game often changed the participants' usual swimming patterns. *"Leviathan was nearly dead, but everyone but me was gone and healing themselves. So I did a lot of butterfly. (...) It was really unusual for me. It's so hard and I don't do that much while swimming alone."* Note that a previous work reported that exercise-based games may result in higher level of exercise intensity [42], and MobyDick newly showed a potential to promote exercisers to do a certain style of workout (e.g., butterfly).

Autonomous and dynamic team-play with social awareness cues. As expected earlier in the game design, we found that the players dynamically change their tactics by listening to the team-wide audio broadcast and making an individual decision on what would be best for each other. One participant commented: *"[Bravo] isn't good at backstroke. When he's dead, I did backstroke together to get him back ASAP."* Note that previous on-line game studies reported that there are simple forms of collaboration between strangers or friends without explicit inter-player communication [37]. However, we found that the current design of MobyDick has a potential to facilitate *highly strategic* collaboration between players, even without explicit communication channel between players.

6 Related Work

Our work expands on earlier studies on exergame design, wireless networking performance evaluation, and sensor-based swimming analysis. In the following sections, we review each of these themes in detail.

6.1 Exergame Review

Wii Fit and Kinect Sports are the leading gaming consoles for stimulating exercise activities, for example, boxing, tennis, and running. While these gaming consoles provide an excellent opportunity for enjoying such activities indoors, this does not usually provide enough exercise intensity. In recent years, as sensor and device technology have advanced, creating new kinds of exergames has been an active area of research. Moreover, strong scientific evidence on the exercise benefits of exergames has been provided [18]. Typical exergame design involves augmenting or visualizing existing sports activities. One way of doing this is to revise existing exercise equipment or develop new devices for physical activities that can be used as game input, for example, an arm ergometer [16], an interactive treadmill [3], a spirometer [25], a tangible ball [24], and playful gadgets [8, 10, 29]. Exergame Fitness Co. provides several types of exergame controllers, including game cycles and interactive floor and wall systems [2]. Allowing for the remote participation of exercise activities that leverage social support and influence has been another widely used mechanism for exergame design. Mueller et al. recently developed a set of exergames known as "sports over distance", which use players' physical actions directly for remote sport play, for example, "Jogging Over a Distance," "Remote Impact", and "Table Tennis for Three" [33, 34, 35]. Sharing heart rates allows remote exercise participants to exchange their physical intensity levels [12, 35, 38, 51]. Park et al. [40] proposed an exergaming platform called ExerLink that allows remotely connected participants to use multiple heterogeneous exercise devices for game play. Our main contribution is to gamify swimming activities and to design/evaluate an interactive game that considers multi-user coordination. Readers can find more information about recent advances in exergames, and their design principles, in [7, 49].

6.2 Wireless Networking in the Pool

LTE performance measurement has become a topic that is of great interest to the research community. Huang et al. performed a comprehensive measurement study on LTE networks and showed that LTE's downlink/uplink throughput is much higher than that of 3G. Their LTE network modeling revealed that LTE-related parameter selection, e.g., tail power, had a significant effect on energy consumption [19]. LTE displays significantly lower RTTs than those of 3G networks [20], but researchers found that there are various inefficiencies in TCP compared to LTE, such as undesired slow start, and called for further research into developing LTE-friendly transport protocols. Our work supplements these studies as we considered rather extreme networking environments, i.e., that of a phone being immersed under water. We performed a comparative study on WiFi, 3G, and LTE networks in laboratory and swimming pool scenarios. Given that recent smartphones have begun to include waterproofing features (for example, the Galaxy S5 Active), and waterproof smartphone cases are commonly used in the water theme parks (for example, for taking photos). Our measurement results provide valuable insights into novel application design for water activities, such as photo uploading, audio/video streaming, and interactive games.

Previous studies on underwater sensor networks have examined the use of radio communications under water [28, 47], although the majority of studies have focused on using acoustic data transmission. Lloret et al. experimented with wireless communication between two submerged wireless nodes and showed that significant packet loss (more than 30%) occurred when the nodes were approximately 15cm apart. Our experimental results regarding communication between a submerged node and a node in the air are consistent with these results. Jiang et al. studied the effect of the frequency band on underwater RF signal propagation. They found that propagation loss increases significantly in high-frequency bands (above 100 MHz) [23]. Sandra et al. analyzed the performance differences between different WiFi channels [28]. Our work differs from these studies in that our experimental condition involves wireless communications between a node in the air and a node in the water. We conducted a comparative study on three popular wireless networking technologies, namely WiFi, 3G, and LTE, by systematically analyzing delay, packet loss, and network reconnection, as well as conducting a supplementary measurement study in a real swimming scenario.

6.3 Sensor-based Swimming Analysis

Performance analysis in swimming has mostly been based on offline, manual processing of recorded images in order to derive quantitative and qualitative measures of performance. Researchers in sports science and human kinematics have recently examined how on-body motion sensors can be used

as an alternative to image analysis, providing valuable insights into automatic recognition. James et al. analyzed the acceleration data of elite swimmers using a back-mounted accelerometer [22]. Similarly, Slawson et al. showed that acceleration data can provide useful information for performance analysis [50]. As the first system of its kind, the aim of SwimMaster was to provide a fine-grained assessment of swimming by extracting swimming parameters such as velocity, arm strokes, body balance, and body rotation [4]. To enable fine-grained monitoring, SwimMaster uses three motion sensors: one wrist sensor and two lower/upper back sensors. The major limitation of this system is that it only supports offline data analysis and does not recognize different swimming styles other than freestyle. Along the same line of research, Siirtolar et al. used two body-worn sensors, one on the wrist and one on the back, for the automatic classification of strokes, and showed that accurate classification is possible with a back-mounted motion sensor [48]. While this study considered three swimming styles, it was largely based on offline machine learning and did not provide any insights into real-time detection and inter-personal skill differences. Recently, Marshall [31] and Lee [27] demonstrated that smartphones can be used as a platform for supporting stroke sensing and real-time feedback, but they did not provide any detailed algorithms in their study.

Our work significantly extends earlier studies in that: 1) we used off-the-shelf smartphones mounted on the upper arm, where motion data is much noisier than in other locations, 2) we proposed real-time activity detection algorithms that can accurately detect four popular stroke, and 3) we showed that individual skill differences have a significant effect on recognition performance, particularly among recreational swimmers, as well as the fact that personalized machine learning can significantly improve recognition accuracy.

7 Discussion

We present the practical implications of our main findings and discuss the limitations of this work.

7.1 Implications

Our preliminary experiences of MobyDick resulted in several practical design implications and also provided insights into practical mobile software design in aquatic environments.

As mentioned by our participants, MobyDick's system of sharing user information has a major limitation in that users tend to disregard others' information when they become tired. One way of mitigating this problem is to support pair-wise collaboration and competition in the game's content. If a user is paired with another user and asked to collaborate, it is expected that users may feel less burdened as they do not need to track all the other users.

Exercise intensity should be carefully coordinated in the game logic. In certain cases, our participants had to perform a series of strokes that require significant physical effort, which could lead to physical injury or burnout. Since it is possible to monitor a swimmer's heartbeat using a special headset device, it is possible to incorporate heart rate information into the game design; in turn, heart rate information can easily be translated into exercise intensity [6]. This would allow the game logic to automatically adapt to a swimmer's changing conditions.

Our results on networking performance in an aquatic environment provide a valuable insight into various system designs. We identified certain unique patterns in different wireless network technologies, for example, periodic blackouts with WiFi and bulk loss or long latency in LTE. Although we did not thoroughly evaluate TCP performance, we empirically observed that transmission often fails, which means a very low throughput. Since the depth of the water is related to networking performance, it is possible to use barometric sensing to estimate channel conditions and schedule packet transmissions.

We mainly used Casio's rugged smartphones in our experiments. Two smartphones were damaged during the experiments, due to several reasons. The rubber isolators sometimes become loose, possibly as a result of large bodily movements. Moreover, the wires of waterproof headsets may interfere with swimming strokes, which then pull out the headset. Given that such events may occur during various aquatic activities, the rugged phone design should take this fact into account.

7.2 Limitations

Our networking performance measurement was limited to two handsets and two operators. Given that the network parameter configurations and policy settings of cellular operators vary widely, the generalizability of this work is limited. Nonetheless, the results obtained in the air are, to an extent, consistent with the measurement results for the US operators, and we expect that similar behaviors may be observed in other operators' networks.

For activity recognition, we tested the models with the dataset obtained from only 11 participants, due to the difficulty of hiring individuals who were capable of completing the experiments. Moreover, it was possible that our model training could exhibit gender bias. To evaluate whether this was the case, in one experiment we left the one female participant out and trained/tested the model. However, we found that there was still a very high level of accuracy, indicating that gender bias may not exist. While we need to test our systems with a greater number of female participants, we hypothesize that skill is more important than gender in obtaining the desired results. In our dataset, skilled users tended to show similar motion patterns, whereas less skilled users could show less patterned motion behavior. Our personalized models also need to be tested with more people to validate their accuracy across different individuals.

Our preliminary user study included only a small number of participants (n=8). After several iterations of the game design, we will conduct an experiment with five to ten groups of three to four users each, and draw more certain conclusions from both qualitative and quantitative data analysis.

8 Acknowledgements

This work was partly supported by the National Research Foundation of Korea Grant funded by the Korean Government (MSIP) (NRF-2012R1A1A1008858, NRF-2011-0018120) and the ICT R&D program of MSIP/IITP [10041313, UX-oriented Mobile SW Platform]. We would like to thank the anonymous reviewers and shepherd for their helpful comments. The corresponding author of this work is U. Lee.

9 References

[1] Monster hunter. http://www.capcom.com/us/. Accessed: 2014-04-01.

[2] Exergame Fitness, http://www.exergamefitness.com, 2012.

[3] M. Ahn, S. Kwon, B. Park, K. Cho, S. Choe, I. Hwang, H. Jang, J. Park, Y. Rhee, and J. Song. Running or gaming. In *ACM ACE 2009*.

[4] M. Bächlin, K. Förster, and G. Tröster. Swimmaster: a wearable assistant for swimmer. In *ACM Ubicomp 2009*.

[5] L. Bao and S. S. Intille. Activity recognition from user-annotated acceleration data. In *Pervasive computing*, pages 1–17. Springer, 2004.

[6] G. A. v. Borg. Psychophysical bases of perceived exertion. *Med sci sports exerc*, 14(5):377–381, 1982.

[7] T. Campbell, B. Ngo, and J. Fogarty. Game design principles in everyday fitness applications. In *ACM CSCW 2008*.

[8] Y. Chang, J. Lo, C. Huang, N. Hsu, H. Chu, H. Wang, P. Chi, and Y. Hsieh. Playful toothbrush: ubicomp technology for teaching tooth brushing to kindergarten children. In *ACM CHI 2008*.

[9] A. Chatham, B. A. Schouten, C. Toprak, F. Mueller, M. Deen, R. Bernhaupt, R. Khot, and S. Pijnappel. Game jam. In *ACM CHI EA 2013*.

[10] M. Chiu, S. Chang, Y. Chang, H. Chu, C. Chen, F. Hsiao, and J. Ko. Playful bottle: a mobile social persuasion system to motivate healthy water intake. In *ACM Ubicomp 2009*.

[11] N. Davey, M. Anderson, and D. A. James. Validation trial of an accelerometer-based sensor platform for swimming. *Sports Technology*, 1(4-5):202–207, 2008.

[12] R. de Oliveira and N. Oliver. Triplebeat: enhancing exercise performance with persuasion. In *ACM MobileHCI 2008*.

[13] T. Erickson and W. A. Kellogg. Social translucence: An approach to designing systems that support social processes. *ACM Trans. Comput.-Hum. Interact.*, 7(1):59–83, 2000.

[14] J. Färber. Network game traffic modelling. In *ACM WNSSG 2002*.

[15] E. E. Ghiselli. *Theory of psychological measurement*, volume 13. McGraw-Hill New York, 1964.

[16] S. Guo, G. Grindle, E. Authier, R. Cooper, S. Fitzgerald, A. Kelleher, and R. Cooper. Development and qualitative assessment of the gamecycle exercise system. *IEEE TNSRE 2006*.

[17] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: An update. *SIGKDD Explor. Newsl.*, 11(1):10–18, 2009.

[18] H. A. Hernandez, Z. Ye, T. Graham, D. Fehlings, and L. Switzer. Designing action-based exergames for children with cerebral palsy. In *ACM CHI 2013*.

[19] J. Huang, F. Qian, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck. A close examination of performance and power characteristics of 4g lte networks. In *ACM MobiSys 2012*.

[20] J. Huang, F. Qian, Y. Guo, Y. Zhou, Q. Xu, Z. M. Mao, S. Sen, and O. Spatscheck. An in-depth study of lte: Effect of network protocol and application behavior on performance. In *ACM SIGCOMM 2013*.

[21] E. B. Hunt, J. Marin, and P. J. Stone. Experiments in induction. 1966.

[22] D. A. James, N. Davey, and T. Rice. An accelerometer based sensor platform for insitu elite athlete performance analysis. In *IEEE Sensors 2004*.

[23] S. Jiang and S. Georgakopoulos. Electromagnetic wave propagation into fresh water. *Journal of Electromagnetic Analysis & Applications*, 3(7), 2011.

[24] D. Kern, M. Stringer, G. Fitzpatrick, and A. Schmidt. Curball–a prototype tangible game for inter-generational play. In *IEEE WETICE 2006*.

[25] B. Lange, S. Flynn, A. Rizzo, M. Bolas, M. Silverman, and A. Huerta. Breath: a game to motivate the compliance of postoperative breathing exercises. In *IEEE ICVR 2009*.

[26] A. Larmo, M. Lindstrom, M. Meyer, G. Pelletier, J. Torsner, and H. Wiemann. The lte link-layer design. *Communications Magazine, IEEE*, 47(4):52–59, 2009.

[27] H. Lee, M. Moon, T. Park, I. Hwang, U. Lee, and J. Song. Dungeons & swimmers: designing an interactive exergame for swimming. In *ACM Ubicomp 2013*.

[28] J. Lloret, S. Sendra, M. Ardid, and J. J. Rodrigues. Underwater wireless sensor communications in the 2.4 ghz ism frequency band. *Sensors*, 12(4):4237–4264, 2012.

[29] J. Lo, T. Lin, H. Chu, H. Chou, J. Chen, J. Hsu, and P. Huang. Playful tray: adopting ubicomp and persuasive techniques into play-based occupational therapy for reducing poor eating behavior in young children. *Springer UbiComp 2007*.

[30] T. Manninen and T. Kujanp. Non-verbal communication forms in multi-player game session. In *Springer HCI 2002*.

[31] J. Marshall. Smartphone sensing for distributed swim stroke coaching and research. In *ACM Ubicomp 2013*.

[32] W. P. Morgan and M. L. Pollock. Psychologic characterization of the elite distance runner. *Annals of the New York Academy of Sciences*, 301(1):382–403, 1977.

[33] F. Mueller and S. Agamanolis. Sports over a distance. *Computers in Entertainment (CIE)*, 3(3):4–4, 2005.

[34] F. Mueller, M. Gibbs, and F. Vetere. Design influence on social play in distributed exertion games. In *ACM CHI 2009*.

[35] F. Mueller, F. Vetere, M. Gibbs, D. Edge, S. Agamanolis, and J. Sheridan. Jogging over a distance between europe and australia. In *ACM UIST 2010*.

[36] K. Muralidharan, A. J. Khan, A. Misra, R. K. Balan, and S. Agarwal. Barometric phone sensors: More hype than hope! In *ACM HotMobile 2014*.

[37] B. Nardi and J. Harris. Strangers and friends: Collaborative play in world of warcraft. In *ACM CSCW 2006*.

[38] V. Nenonen, A. Lindblad, V. Häkkinen, T. Laitinen, M. Jouhtio, and P. Hämäläinen. Using heart rate to control an interactive game. In *ACM CHI 2007*.

[39] P. Ng and K. Nesbitt. Informative sound design in video games. In *ACM IE 2013*.

[40] T. Park, I. Hwang, U. Lee, S. I. Lee, C. Yoo, Y. Lee, H. Jang, S. P. Choe, S. Park, and J. Song. Exerlink: enabling pervasive social exergames with heterogeneous exercise devices. In *ACM MobiSys 2012*.

[41] T. Park, J. Lee, I. Hwang, C. Yoo, L. Nachman, and J. Song. E-gesture: a collaborative architecture for energy-efficient gesture recognition with hand-worn sensor and mobile devices. In *ACM SenSys 2011*.

[42] T. Park, C. Yoo, S. P. Choe, B. Park, and J. Song. Transforming solitary exercises into social exergames. In *ACM CSCW 2012*.

[43] I. Pefkianakis, S.-B. Lee, and S. Lu. Towards mimo-aware 802.11n rate adaptation. *IEEE/ACM Trans. Netw.*, 21(3):692–705, 2013.

[44] S. J. Pell and F. Mueller. Gravity well: underwater play. In *ACM CHI EA 2013*.

[45] J. W. Pennebaker and J. M. Lightner. Competition of internal and external information in an exercise setting. *Journal of personality and social psychology*, 39(1):165, 1980.

[46] S. Reddy, M. Mun, J. Burke, D. Estrin, M. Hansen, and M. Srivastava. Using mobile phones to determine transportation modes. *ACM Transactions on Sensor Networks (TOSN)*, 6(2):13, 2010.

[47] S. Sendra, J. Lloret, J. Rodrigues, and J. Aguiar. Underwater wireless communications in freshwater at 2.4 ghz. 2013.

[48] P. Siirtola, P. Laurinen, J. Roning, and H. Kinnunen. Efficient accelerometer-based swimming exercise tracking. In *IEEE CIDM 2011*.

[49] J. Sinclair, P. Hingston, and M. Masek. Considerations for the design of exergames. In *ACM GRAPHITE 2007*.

[50] S. Slawson, L. Justham, A. West, P. Conway, M. Caine, and R. Harrison. Accelerometer profile recognition of swimming strokes (p17). In *The Engineering of Sport 7*, pages 81–87. Springer, 2008.

[51] T. Stach, T. Graham, J. Yim, and R. Rhodes. Heart rate control of exercise video games. In *GI 2009*.

[52] Y. Ukai and J. Rekimoto. Swimoid: interacting with an underwater buddy robot. In *IEEE HRI 2013*.

[53] A. Willumsen, O. Hagen, and P. Boge. Filtering depth measurements in underwater vehicles for improved seabed imaging. In *IEEE OCEANS 2007*.

[54] I. H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2005.

[55] B. Wozniak and J. Dera. *Light absorption in sea water*, volume 33. Springer, 2007.

[56] L. Yu and H. Liu. Efficient feature selection via analysis of relevance and redundancy. *The Journal of Machine Learning Research*, 5:1205–1224, 2004.

[57] B. Yuan, E. Folmer, and F. C. Harris, Jr. Game accessibility: A survey. *Univers. Access Inf. Soc.*, 10(1):81–100, 2011.