Designing Interactive Multiswimmer Exergames: A Case Study

WOOHYEOK CHOI and JEUNGMIN OH, Korea Advanced Institute of Science and Technology
TAIWOO PARK, Michigan State University
SEONGJUN KANG, Korea Advanced Institute of Science and Technology
MIRI MOON, SK Telecom
UICHIN LEE, Korea Advanced Institute of Science and Technology
INSEOK HWANG, IBM Research Austin
DARREN EDGE, Microsoft Research Asia
JUNEHWA SONG, Korea Advanced Institute of Science and Technology

The unique aquatic nature of swimming makes it difficult to use social or technical strategies to mitigate the tediousness of monotonous exercises. In this study, we propose the use of a smartphone-based multiplayer exergame named *MobyDick*. MobyDick is designed to be played while swimming, where a team of swimmers collaborate to hunt down a virtual monster. To this end, we take into account both human factors and technical challenges under swimming contexts. First, we perform a comparative analysis of a variety of wireless networking technologies in the aquatic environment and identify various technical constraints on wireless networking. Second, we develop a swimming activity recognition system to enable precise and real-time game inputs. Third, we devise a multiplayer game design by employing the unique interaction mode viable in an underwater environment, where the abilities of human communication are highly limited. Finally, we prototype MobyDick on waterproof off-the-shelf Android phones, and we deploy it in real swimming pool environments ($n = 8$). Our qualitative analysis of user interview data reveals certain unique aspects of multiplayer 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

Additional Key Words and Phrases: Activity recognition, exertion games, swimming, underwater wireless networking performance

This work was partly supported by Institute for Information & Communications Technology Promotion (IITP) and National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No. B0101-15-1272, Development of Device Collaborative Giga-Level Smart Cloudlet Technology; No. R22121500140001002, Designing Interactive Technologies for Active Workstations; No. 2011-0018120, Next Generation Smartphone Platform Supporting User-centered Smart Spaces and Services).
The corresponding author of this work is Uichin Lee.
This work extends on our earlier conference paper presented at ACM SenSys '14 [Choi et al. 2014].
Authors' addresses: W. Choi, J. Oh, S. Kang, and U. Lee, Department of Knowledge Service Engineering, Korea Advanced Institute of Science and Technology, Daejeon, South Korea; emails: woohyeok.choi@kaist.ac.kr, jminoh@kaist.ac.kr, sjkang@kaist.ac.kr, uclee@kaist.ac.kr; T. Park, Department of Media and Information, Michigan State University, East Lansing, MI; email: twp@msu.edu; M. Moon, Product Planning Division, SK Telecom, Seoul, South Korea; email: miri.moon@sk.com; I. Hwang, IBM Research Austin, Austin, TX; email: ihwang@us.ibm.com; D. Edge, Microsoft Research Asia, Beijing, China; email: darren.edge@microsoft.com; J. Song, School of Computing, Korea Advanced Institute of Science and Technology, Daejeon, South Korea; email: junesong@kaist.ac.kr.

DOI: http://dx.doi.org/10.1145/2888399

ACM Reference Format:
Woohyeok Choi, Jeungmin Oh, Taiwoo Park, Seongjun Kang, Miri Moon, Uichin Lee, Inseok Hwang, Darren Edge, and Junehwa Song. 2016. Designing interactive multiswimmer exergames: A case study. ACM Trans. Sen. Netw. 12, 3, Article 17 (July 2016), 40 pages.
DOI: http://dx.doi.org/10.1145/2888399

1. INTRODUCTION

There are various forms of water-based exercises, ranging from fitness swimming and aquarobics to rehabilitation programs. Compared to ground-based exercises, water-based exercises generally have a lower risk of exercise-related injuries (i.e., lower impacts on joints due to buoyancy) and higher exercise intensity (i.e., burning up more calories due to water resistance). In this work, we will consider fitness swimming, which ranks seventh for popular aerobic exercises in the United States [SFIA 2013].

Recent advances in sensor and device technology have spurred the creation of a variety of swimming-related research prototypes and commercial products. For example, Garmin Swim^{TM} tracks a swimmer's stroke type, workout distance, and time. SwimMaster [Bächlin et al. 2009] analyzes swimming motions with several body-attached sensors and provides feedback for swimming posture adjustment. Swimmoid [Ukai and Rekimoto 2013] is a mobile underwater robot companion that visually analyzes the swimmer's posture and provides real-time feedback while following a swimmer. Davey et al. [2008] presented a system that visualizes the motion data of swimmers for performance analysis. Lee et al. [2013] proposed an exertion game for swimming, Dungeons and Swimmers, in which a swimmer interacts with a virtual object using swimming strokes.

In this article, we explore the design considerations for transforming swimming activities into social exergames. Generally, a swimmer swims the length of a pool alone and repetitively, which is similar to repetitive, individual, aerobic (RIA) exercises, such as jogging or stationary cycling [Park et al. 2012]. A social exergame can mitigate monotonous nature of RIA exercises and provide more engaging experiences [Park et al. 2012; Ahn et al. 2009].

However, we should consider several constraints to design a socially interactive game for swimming. According to social exergame design guidelines proposed by Park et al. [2012], we should carefully select target activities that are intuitive to users and do not disturb the core mechanics of the exercise, as players interact with the game using their physical movements. Moreover, those selected target activities are recognized accurately for interactive game play. For this, we choose intrinsic swimming activities, such as swimming styles, stroke timing, and a turn, and we build the recognition module around these activities.

Another consideration is how to support interplayer interaction. To interconnect multiple players while swimming, wireless data exchange needs to be supported. However, networking technologies (specifically electromagnetic-based networking technologies) suffer from severe performance degradation in the aquatic environment of a swimming pool [Lanbo et al. 2008]. To understand networking characteristics in underwater environments and draw game design implications that are applicable to swimming contexts, we perform a comparative analysis on a variety of wireless technologies.

In addition, we explore human factors with respect to swimming. For example, people would suffer from highly limited visual communication, lack of spoken communication, and reduced sensitivity of auditory communication while swimming. Through careful consideration, we design an interactive multiswimmer exergame, named *MobyDick*, as a case study. In this game, multiple swimmers form a team and hunt down a virtual white whale, MobyDick. Players attack the monster, defend themselves, and heal the wounded by using swimming styles. MobyDick supports socially enriched swimming with a unique mode of interaction by employing a 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 this work, the key research elements include (1) networking performance analysis to understand the nonfunctional requirements of exergames, (2) real-time swimming activity detection to enable players to interact with the game (using swimming actions as a game controller), and (3) system implementation and a preliminary study of user experiences on our game. We expect that our findings complement earlier swimming-related work and may be applied to a variety of water-based exercises.

The key contributions of this article can be summarized as follows:

—We perform a comparative analysis of networking performance (e.g., packet exchange, packet loss, and reconnection latency) on popular electromagnetic-based networking technologies, such as 3G, LTE, and WiFi, in aquatic environments. Overall, LTE provides fairly consistent performance despite the fact that it may experience bulk loss and disconnection. WiFi shows periodic blackouts and high packet loss, and 3G does not work at all underwater.

—We build a real-time swimming activity detection module, named *StrokeSense*, by leveraging multiple sensors in a smartphone, namely an accelerometer, a gyroscope, a magnetometer, and a barometer. Unlike earlier sensing systems designed for athlete training [Bächlin et al. 2009; Davey et al. 2008], StrokeSense enables real-time classification of four popular strokes, as well as turn event and stroke timing detection. We conclude that considering differences in individual swimming capability is critical for achieving high classification accuracy, and we demonstrate that personalized model training is preferable for our gaming scenario.

—We design MobyDick and implement 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 demonstrate that certain technical challenges, such as temporary network disconnection, can be overcome with game interaction design. We perform a preliminary user study with eight participants. Our interview results show that the proposed game is enjoyable, stroke sensing is timely, mappings between stroke types and game commands are intuitive, and the broadcast of social awareness cues provide a uniquely thrilling swimming experience.

This work significantly extends the prior work [Choi et al. 2014], and the rest of this article is organized as follows. In Section 2, we briefly illustrate the characteristics of swimming and design considerations for social exergames. In Section 3, we perform comparative analyses among wireless technologies, such as 3G, LTE, and WiFi, and draw design implications to interconnect multiple swimmers. We then explain our swimming activity recognition system, StrokeSense, in Section 4. In Section 5, we introduce MobyDick and its design consideration on technical challenges and human factors. In Section 6, we describe a detailed implementation of the MobyDick working prototype and findings from a preliminary user study. In Section 7, we present related work, and finally, we conclude this work with design implications and limitations in Section 8.

2. TRANSFORMING SWIMMING TO A MULTIPLAYER EXERGAME

To transform swimming into a multiplayer exergame, it is important to study the behavioral and environmental nature of swimming. Therefore, in this section, we illustrate the kinematics and characteristics of swimming and propose a set of game design requirements and constraints.

2.1. Background: Understanding Swimming

Swimming is an exercise of moving through water using special movements of the entire body, called *swimming styles*. There are a variety of swimming styles, but popular ones are freestyle (front crawl), backstroke (back crawl), butterfly, and breaststroke. Each of these styles can be divided into a sequence of phases, which generally include arm pulling, which is dragging an arm underwater, and recovery, which is taking an arm out of the water [Colwin 2002].

With the freestyle and backstroke, one arm stroke is followed by the other arm stroke. Each stroke consists of one arm pulling while the other arm is recovering, and a swimmer can move forward by performing these alternately. Butterfly strokes are different from those of freestyle and backstroke. With the butterfly, each stroke starts with arm pulling and then pushing with leg kicking. This is followed by arm recovery and a flying phase. With the breaststroke, the arm pulling phase is followed by an elbow gathering phase. Then, arm recovery and a leg flexion phase are performed, and a stroke ends with a leg kicking and gliding phase. In addition to swimming styles, another movement, turn, can be added when considering pool swimming activities. There are corresponding types of turns to each swimming style, but they all typically involve the following sequence of phases: turning/rotation, pushing, gliding, underwater kicking, and pull-out [Colwin 2002].

For efficient and fast swimming, a swimmer should perform all movements with a high level of body balance and coordination, compared to land-based exercises such as jogging or cycling. The high density of water, which is about 800 times that of air, places high resistance on swimmers' bodies, and they consume most of their energy overcoming water resistance while swimming [Hines 2008]. To minimize such resistance, swimmers need to balance their bodies horizontally with the water surface and synchronize the movements of their body parts [Hines 2008; Bächlin et al. 2009].

2.2. Gamification of Swimming

Park et al. [2012] suggested several design guidelines to gamify existing exercises. We extended those guidelines by considering swimming contexts and derived a set of design requirements for a multiswimmer exergame.

2.2.1. Adapting Swimming Activities into the Game. To gamify an exercise, it is necessary to decide how to map exercise activities into game activities (i.e., game inputs). For example, in the case of swimming, we can use swimming speed, swimming styles, stroke timing, elapsed time, total distance, or the number of turns as game inputs. Based on a particular mapping, game design and interaction could be different, even if the same exercise was considered. For instance, in the case of running activities, Kukini [Campbell et al. 2008] considered movement distance in running as a game input where a player explores a virtual world by performing quests. In Jogging over a Distance [Mueller et al. 2007, 2010], a player jogs with a distant jogger, and they can balance relative exertion using their heart rates via spatialized audio.

In addition to exercise activities that can be directly extracted from a target exercise, additional activities can be incorporated into the game. For example, SwanBoat [Ahn et al. 2009] is mainly based on treadmill running, but it also includes additional gestures such as punching and flapping. Those additional activities can provide rich interactions and fun to the game. However, employing such additional activities should not disrupt the originally intended exercise. Swimming especially requires the coordination of all body parts, so it has a small degree of freedom in which to add other activities.

When transforming activities of a certain exercise into virtual game actions, designers have to take intuitiveness of interaction into account so that players can engage in game play quickly. For example, SwanBoat [Ahn et al. 2009] maps the difference between two runners' speeds into a virtual swan boat's direction (i.e., left or right). In addition, when players punch, the swan boat strikes the other team's boat using a wing.

Technically, designing interaction devices and recognizing physical activities should be considered as well to let users play the games using their own bodies. For example, Wii Fit provides a controller that measures weight and senses body balance. Kinect recognizes a player's movement using a special camera. Moreover, recent advances of sensor technologies have prompted the development of a variety of interactive devices, such as an arm ergometer [Guo et al. 2006], an interactive treadmill [Ahn et al. 2009], a spirometer [Lange et al. 2009], a tangible ball [Kern et al. 2006], and playful gadgets [Chang et al. 2008; Chiu et al. 2009; Lo et al. 2007]. A swimming exergame should consider how to recognize swimming activities using specific sensors or devices according to key design primitives.

2.2.2. Feedback for Swimmers. Another important issue in exergame design is how to provide game feedback to a user. This should be decided according to sensory and cognitive capabilities. Many current exergames use visual, auditory, or tactile feedback; however, in swimming, a swimmer can be influenced by auditory noise from splashing water and tactile stimuli from water resistance. Designers need to carefully consider the performance characteristics of each output mechanism in such limited conditions. According to a recent study [Bächlin et al. 2009], visual feedback is considered to be more effective than auditory and tactile feedback, but it requires special hardware (e.g., interactive goggles). In our work, we consider off-the-shelf, waterproof earphones to deliver auditory feedback.

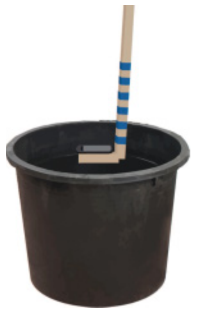
2.2.3. Social Interaction Between Swimmers. Social interaction can motivate an exerciser to continue an exercise [Wilson and Brookfield 2009]. Therefore, prior exergame studies have incorporated interplayer interaction into their games, such as providing social awareness to remote exercisers [Mueller et al. 2007, 2010] and enabling competition or cooperation among players [Park et al. 2012; Mueller et al. 2009; Mueller and Agamanolis 2005]. For swimming, there are a variety of candidate social interactions. For example, designers can transform a solitary swimming workout into social swimming or enrich user experience in group swimming activities, such as swimming lessons.

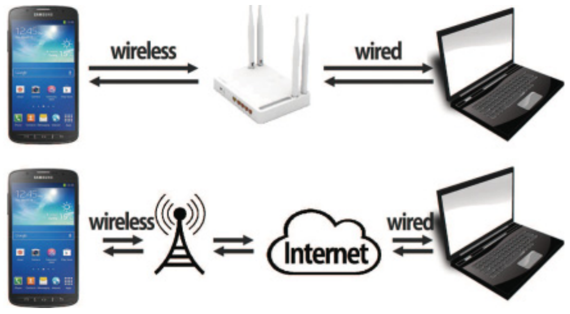
Many current social exergames use wireless technologies to enable remote interactions. However, for water-based exercises, such as swimming or skin diving, it is necessary to take thorough account of network communication due to the loss of signal propagation underwater. Such an environmental characteristic can influence not only technical constraints, such as device design or network protocols, but also game contents.

3. NETWORK PERFORMANCE DURING SWIMMING

To allow swimmers to play exergames with each other, wireless communication is necessary. However, the aquatic environment of swimming makes wireless communication difficult. For example, acoustic waves suffer from multipath propagation, and electromagnetic waves suffer from signal attenuation in shallow and conductive water conditions, such as a swimming pool [Lanbo et al. 2008]. Designing multiplayer swimming exergames requires an understanding of networking characteristics in the context of swimming.

In this section, we provide a comparative analysis of the performance of different wireless networking technologies in the swimming pool environment and draw implications for designing interactive systems. Specifically, we aim to answer the





(a) In-lab testbed (b) Network connection flow: WiFi (top) and 3G/LTE (bottom)

Fig. 1. Network performance measurement configuration.

following questions: (1) How do the different networking technologies (WiFi, 3G, and LTE) perform when the device is underwater? (2) If the network is disconnected, how long does it take to restore its connection? (3) Will there be significant differences in networking performance for different swimming strokes?

3.1. Network Performances at Different Depths

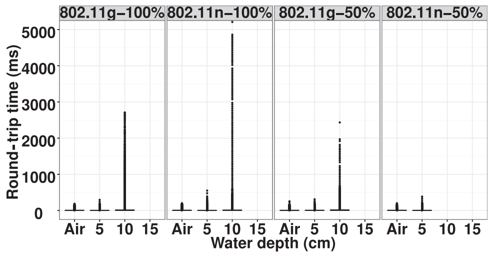
We first observed the changes in the overall performance of networking technologies when underwater. To do this, we measured the round-trip time (RTT), packet loss rate (PLR), and received signal strength (RSS) of WiFi, 3G, and LTE at varying water depths. In addition, we compared the changes in the link speed at different WiFi settings and water depths.

3.1.1. Experimental Setup. We built an in-lab testbed to simulate a swimming pool using a large water container with a diameter of 1m and a depth of 1m (Figure 1(a)). Because the ionized particles of chlorine can affect wireless signal attenuation, we used fresh water with added chlorine to simulate a real swimming pool.

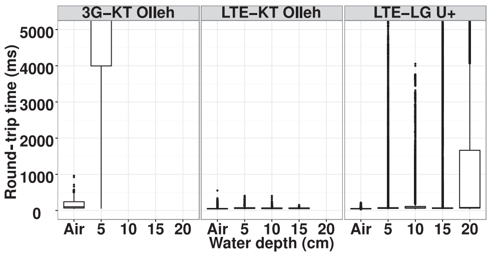
We mounted a smartphone onto a wooden rod, which was used to place the phone at specific depths underwater. The rod and smartphone were lowered into the water in 5cm increments. While submerged, the smartphone sent a User Datagram Protocol (UDP) packet every 50ms to a remote server (2,000 packets in total) located by a laptop and connected to a wired network (see Figure 1(b)). Each packet sent was 128 bytes, a common packet size observed in online interactive games [Färber 2002]. After sending a packet, a submerged node waited for an ACK packet of the same size; thus, we simulated periodic state exchanges between a client and a server. We repeated these measurements 10 times for each experimental condition.

We considered a variety of experimental conditions. For the WiFi measurements, we varied the protocols to include 802.11g (2.4GHz), 802.11n (2.4GHz), 802.11a/n (5GHz), and 802.11ac (5GHz), with two different transmission power settings for the access point (AP): 100% and 50%. For LTE measurements, we used two major cellular operators in Korea, LG U+ and KT Olleh. For 3G measurements, we used KT Olleh.

In the experiment, we used two waterproof Android smartphones: the Casio G'zOne Commando 4G LTE and the Samsung Galaxy S4 Active, which are advertised as being able to sustain water immersion at 1m for 30 minutes. Since the Casio G'zOne Commando 4G LTE did not support 5GHz WiFi operations or the KT Olleh operator, we used the Galaxy S4 Active for WiFi and KT Olleh's LTE measurements; the Casio G'zOne Commando 4G LTE was used for LG U+'s LTE measurements only. The laptop for the server had a 2.5GHz CPU and 8GB memory. To measure the WiFi connection,

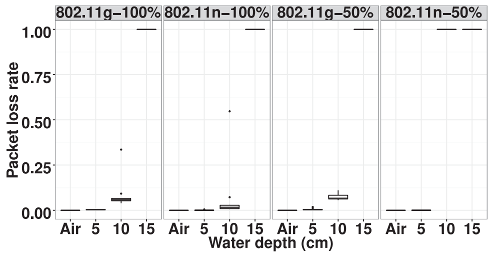


(a) RTT: 802.11g/n at 2.4GHz; 100%/50% AP Tx power

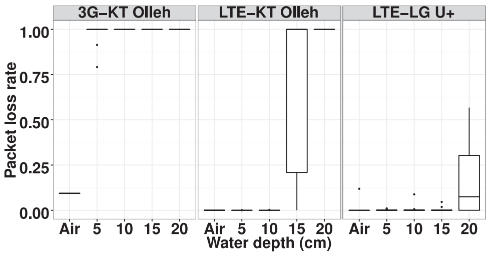


(b) RTT: 3G and LTE (KT Olleh and LG U+)

Fig. 2. RTT measurement results.



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



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

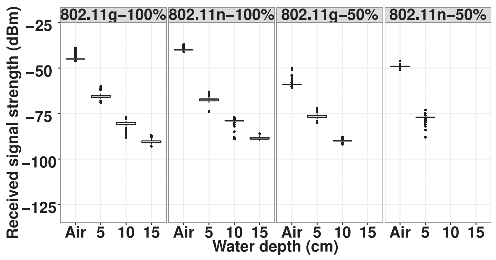
Fig. 3. PLR measurement results.

we used ipTIME's A2004Plus AP, which supports 802.11b/g/n/ac and has two antennas for each bandwidth, 2.4GHz and 5GHz.

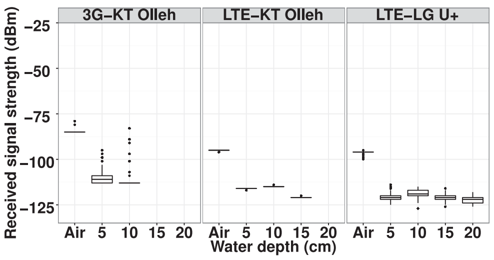
We note that RSS values are dependent on the characteristics of wireless communication methods. Android periodically reports RSS values for WiFi communication. For Wideband Code Division Multiple Access (WCDMA), Android reports a Common Pilot Channel (CPICH)'s Received Signal Code Power (RSCP) in decibel-milliwatt,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

3.1.2. Measurement Results. Measurement results are plotted in Figures 2 through 5. In addition, we provide descriptive statistics of measurement results in Online Appendix A. We observed that the performance of wireless communication deteriorated as depth increased. Accordingly, PLR and the variance of RTT values increased. In air, WiFi showed the lowest RTT (less than 10ms), whereas cellular networks had much larger RTT values (3G-KT Olleh: 165.00ms; LTE-KT Olleh: 50.04ms; LTE-LG U+: 50.21ms), agreeing with previous measurement studies [Huang et al. 2012]. We theorize that the reason LTE has greater latency than WiFi is due to the multihop nature of IP data delivery in cellular network architecture [Larmo et al. 2009]. However, LTE uses millisecond-level scheduling and has a smaller latency than 3G.

[1]In WCDMA and LTE, actual RSS calculation should consider the carrier bandwidth. For example, LTE's RSRP should be multiplied by the number of subcarriers.



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



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

Fig. 4. RSS measurement results.

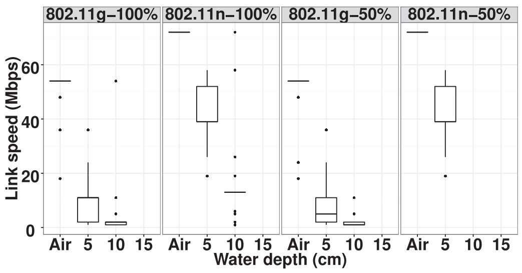


Fig. 5. Link speed measurements results: 802.11g/n at 2.4GHz; 100%/50% AP Tx power.

Of the three, LTE performed the best at the greatest water depth. WiFi 2.4GHz showed an increase in the PLR at a depth of 10cm, and even worse, network disconnection for the case of 802.11n with 50% Tx power. The 802.11n/ac at 5GHz did not work at all underwater, although the results are not shown due to space limitations. 3G also nearly failed to sustain connectivity when the device was immersed at a depth of 5cm. The PLR of LTE started increasing notably at greater depths compared to WiFi and 3G: 15cm (KT Olleh) and 20cm (LG U+); this shows that LTE is more robust than the other networking technologies. There are several possible explanations for this observation. First, LTE typically uses lower frequency bands than 3G's Evolved High-Speed Packet Access (HSPA+) and WiFi (LTE: 800MHz vs. 3G: 2GHz vs. WiFi: 2.4GHz/5GHz); thus, LTE's water absorption coefficient is smaller than the other technologies. According to the Beer-Lambert 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 [Wozniak and Dera 2007]. Second, LTE uses multiple antennas, and its physical modulation employs the Orthogonal Frequency-Division Multiple Access (OFDMA) scheme. 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 communication more reliable and robust. Third, 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) [Larmo et al. 2009].

We compared the performances of different WiFi 2.4GHz settings; in no case were the RTT and PLR values affected when the smartphone was slightly submerged (5cm). The mean RTT values ranged from 4ms to 6ms in air and 5cm underwater. There was no packet loss except for the case of 802.11g with 50% Tx power, which showed only 1% packet loss. However, the signal strength were significantly reduced by more than

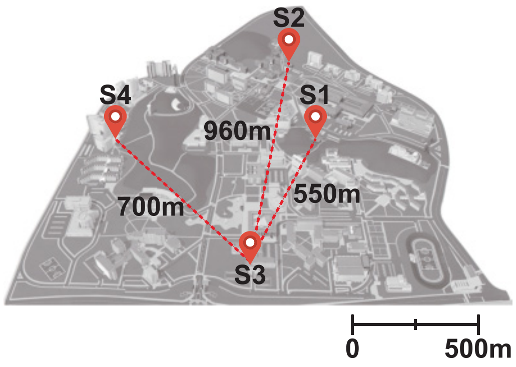


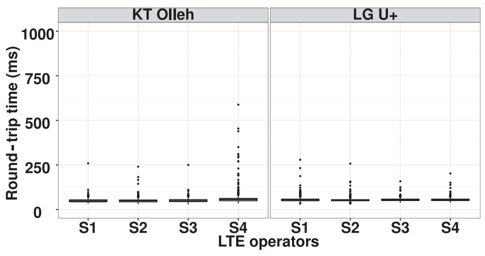
Fig. 6. Locations of four sites.

17dBm when the smartphone was submerged. In addition, both 802.11g and 802.11n lowered the data rates to cope with degraded channel conditions; the median link speed were dropped from 54Mbps to 11Mbps (802.11g) and from 72Mbps to 39Mbps (802.11n) for 100% Tx power. When Tx power was reduced to 50%, the link speed for 802.11g was adjusted to 5Mbps at a depth of 5cm, whereas 802.11n did not show the different link speed.

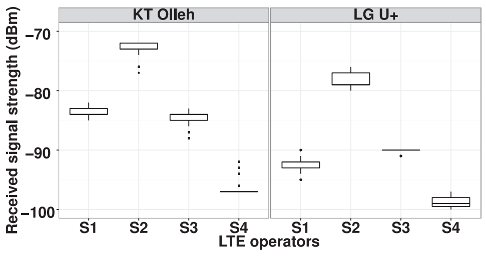
At a depth of 10cm, notable performance degradation was observed in both 802.11g and 802.11n; the mean RTT values significantly increased, ranging from 22ms to 37ms, and 7% to 9% of packets were lost on average for 100% Tx power. The signal strength was reduced by more than 11dBm for all settings. The 802.11g adjusted the median link speed to 2Mbps and 1Mbps for 100% and 50% Tx power. The 802.11n showed the faster link speed than 802.11g; the median link speed of 802.11n for 100% Tx power was 13Mbps. In addition, halving the Tx power made 802.11n lose connectivity. All WiFi 2.4GHz settings completely failed to sustain their connectivity at a depth of 15ms. We hypothesized that these performance differences were mainly due to rate adaptation algorithms that vary widely across different protocols and vendors [Lacage et al. 2004; Pefkianakis et al. 2013]. In an over-the-air measurement, we observed that WiFi 5GHz showed significantly higher data rates than WiFi 2.4GHz (802.11ac: 433Mbps, 802.11n: 150Mbps) and yet immediately lost connectivity when the smartphone was submerged.

When we compared the performances of two LTE operators, the over-the-air measurements were not significantly different. The mean RTT values of KT Olleh and LG U+ were 50.04ms and 50.21ms, respectively. However, we observed significant performance differences between the two operators underwater. KT Olleh did not experience any packet loss to a depth of 10cm, but at a depth of 15cm, it had a packet loss of 68%. Connection was completely lost at a depth of 20cm. In contrast, LG U+ had 1% packet loss at a depth of 15cm, and the loss rate increased to just 16% at a depth of 20cm. We also compared RTT; KT Olleh had consistently lower RTT values than LG U+ underwater. KT Olleh maintained mean RTT values of around 60ms (although its connection was disrupted after 10cm), whereas those of LG U+ substantially increased, ranging from 50.21ms (air) to 2,026.00ms (20cm depth).

Despite the higher KT Olleh RSS values, LG U+ had better performance underwater. We explain some possibilities for this here. LG U+ uses 800MHz bands, whereas KT Olleh uses 900MHz bands. As shown earlier, lower-frequency bands are more robust underwater. Furthermore, LG U+ may implement more robust link scheduling policies, resulting in lower PLR values but higher RTT values. We measured RSS and RTT values at four sites to investigate how RSS and RTT values vary, as shown in Figure 6. Measurement results are plotted in Figure 7. Of the sites considered, we found that

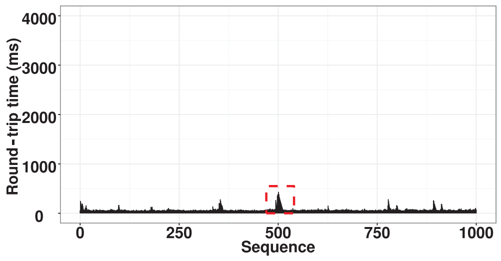


(a) RTT: KT Olleh and LG U+ at different sites

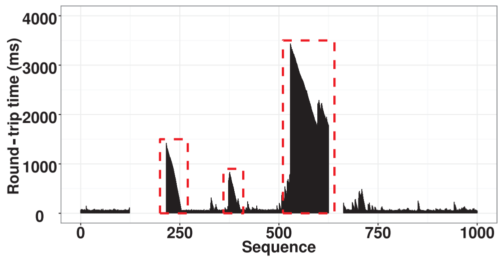


(b) RSS: KT Olleh and LG U+ at different sites

Fig. 7. Performance comparison of two LTE providers at different sites.



(a) Water depth of 5cm



(b) Water depth of 15cm

Fig. 8. RTT traces of LTE (LG U+) at two different water depths.

KT Olleh had consistently higher RSS values (with the RSS differences ranging from 2dBm to 9dBm) and lower RTT values (with the RTT differences ranging from 4ms to 7ms), similar to the results of our testbed experiments.[2] Given that LG U+ had better performance in terms of PLR, even with higher RSS values, we can hypothesize that there could be RSS bias due to carrier differences (e.g., cell planning and power/channel allocation) and/or frequency bands, and link scheduling policies may have a more critical impact on the overall performance than RSS values.

Although the median values of LTE were not very large, we observed RTTs with values greater than 1,000ms in certain trials, specifically when a phone was immersed in water more than 10cm deep. We manually investigated the RTT traces to find the root cause of this large latency (Figure 8). In depths of 5cm and 15cm, 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 are buffered at the UDP transport layer and the LTE link layer. When the channel conditions are good, scheduled packets are delivered immediately. However, when the channel conditions worsen (due to signal attenuation and destructive interferences), the LTE link layer takes a significantly longer time for packet (re)transmissions. As illustrated earlier, LTE has dual ARQ layers [Larmo et al. 2009]. LTE can have eight outstanding link-layer packets, each of which is scheduled in a 1ms subframe. The size of a subframe can be up to tens of kilobytes, depending on the channel bandwidth. When channel conditions become very poor, a large number of packets are backlogged and retransmitted. These packets will then reach their maximum transmission limits and

[2]RTT outliers are due to sporadic RTT spikes that may be due to other factors than RSS variation, such as flow control latency and queueing delays at intermediate network nodes.

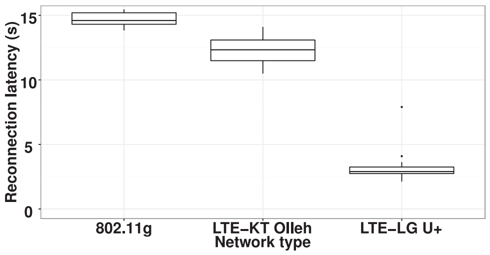


Fig. 9. Reconnection time between WiFi and two LTE providers (KT Olleh and LG U+).

eventually be dropped, thus leading to bulk packet loss. Moreover, since an LTE base station typically uses subframe-level link scheduling under poor channel conditions, it may allocate subframes to those nodes with good channel conditions. This just creates more backlogged packets. In practice, each operator may use different scheduling policies, although proportional fairness algorithms are commonly used.

3.2. Network Reconnection Latency

Due to the frequent submergence of the communication device while swimming and the resulting severe channel conditions, the devices suffer from frequent connectivity loss. To support seamless game play, a wireless client should re-establish a connection in a reasonable time; otherwise, a special game design considering network reconnection patterns would be required. Therefore, understanding reconnection latency can help exergame designers and developers deal with network connectivity problems. In this study, we investigate the latency of re-establishing network connectivity in WiFi 2.4GHz and LTE protocols; 3G and WiFi 5GHz were excluded due to their poor performance underwater.

3.2.1. Experimental Setup. Using the aforementioned testbed, we immersed a smartphone to a depth of 20cm and waited until it completely lost network connectivity. The time taken for disconnection to occur varied widely (5 to 30 seconds). On disconnection, we took the phone out of the water to mimic swimming strokes, which take 1.2 seconds on average (SD: 0.67). We performed this measurement 20 times and recorded the reconnection time as the interval between the time of lifting the phone out of the water and the time of successfully exchanging a packet. To detect when the smartphone client emerged from the water, the sensor readings from a barometer were logged, using the significant pressure level changes at this point. In addition, we found that LTE always assigned a new IP address after reconnection. Therefore, our measurement software reinitialized the underlying transport sockets whenever a connection was re-established with LTE.

3.2.2. Measurement Results. In Figure 9, reconnection times for WiFi 2.4GHz, LTE KT Olleh, and LTE LG U+ are plotted. WiFi was found to have the longest reconnection time (mean: 14.7 seconds, SD: 0.65). To find the major sources of delay for WiFi, we analyzed the Android WiFi source code (i.e., Java framework and native process) and the Android WiFi control module, *wpa_supplicant*, which runs in the background and handles WiFi operations. The current implementation of 802.11 states that WiFi reconnection follows a series of steps (i.e., AP scan, association, and DHCP), and those are triggered by a periodic AP scan request. If the scan results, which are obtained by an AP scan event, contain a preferred WiFi AP, the client automatically attempts to reconnect to that AP. We found that the AP scan, association, and DHCP took 1,080.00ms, 141.1ms, and 1,892ms, respectively. An AP scan request is scheduled for every 15 seconds—that is, the reconnection process will happen up to 15 seconds after disconnection takes place.

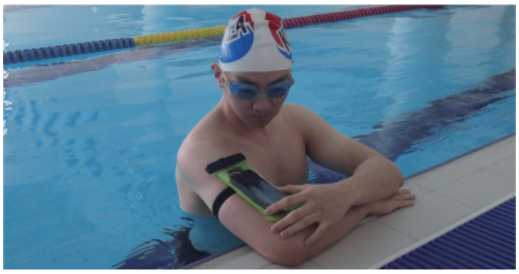


Fig. 10. Mounting the smartphone on the arm using the waterproof armband.

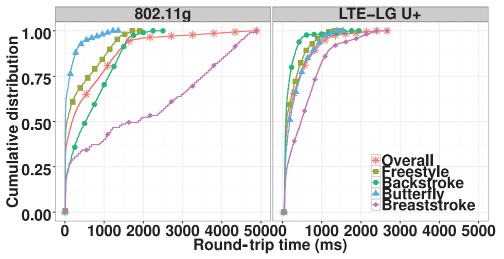
Here, the major sources of delay were AP scan interval (up to 15 seconds) and DHCP (1.8 seconds). To decrease reconnection time, we can handle these two delay sources as follows: whenever disconnection takes place while swimming, we can immediately reconnect to the previously associated AP by reusing a previously assigned IP address. This method can be implemented by modifying *wpa_supplicant*. In our implementation, the modified WiFi client performed scanning immediately after disconnection instead of waiting for up to 15 seconds, and DHCP was omitted by reusing a previously assigned IP address. This patch enabled us to reduce reconnection latency to below a second while using Android. We note that our scenario is different from WiFi seamless roaming [Shin et al. 2004; Eriksson et al. 2008]. With seamless roaming, disconnection takes place when the distance from the currently associated AP becomes too great, and thus the client tries to associate with better candidate APs nearby. In contrast, in our scenario, a currently associated AP is still the best AP even if disconnection occurs, because we assume that a swimming pool can be covered by a single AP, and disconnection results from the severe channel conditions underwater, not due to distance from the AP.

In the LTE measurements, reconnection times between network providers were significantly different. The mean reconnection latency of KT Olleh was 12.3 seconds (SD: 1.0), whereas that of LG U+ was 3.2 seconds (SD: 1.2). The maximum reconnection latencies of KT Olleh and LG U+ were given as 14.1 seconds and 7.9 seconds, respectively. Our data revealed that LG U+ immediately recovered LTE connectivity, whereas KT Olleh always reconnected to 3G networks (first Universal Mobile 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 for a device using KT Olleh (after the device was back above water), it then switched back to LTE networks.

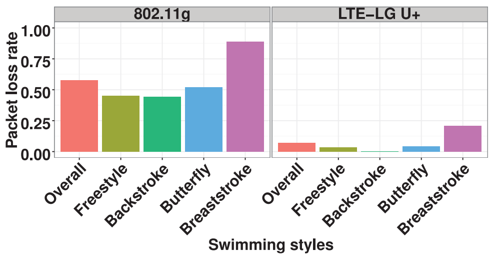
3.3. Network Performance in a Real Swimming Scenario

Thus far, we have presented the general performance of wireless networking technologies underwater. In this section, we present networking performance in a real swimming scenario to determine whether wireless communication is possible underwater and the influence of different swimming styles on network performance.

3.3.1. Experimental Setup. We measured RTT and PLR while performing four swimming styles, namely freestyle, backstroke, breaststroke, and butterfly, using the same logging software as in Section 3.1.1. We considered only 802.11g (with a Galaxy S4 Active) and LTE LG U+ (with a Casio G'zOne Command 4G LTE); we excluded KT Olleh because it had the longest reconnection latency. We asked a participant who had more than a year of swimming experience and was capable of performing four swimming styles to swim each style in 25m lanes four times, for a total of 100m. We had the participant place the smartphone on the upper arm (Figure 10). We did this because measurement



(a) Cumulative distribution of RTT with different swimming styles



(b) PLR with different swimming styles

Fig. 11. Network performance measurements while swimming with WiFi (802.11g) and LTE (LG U+).

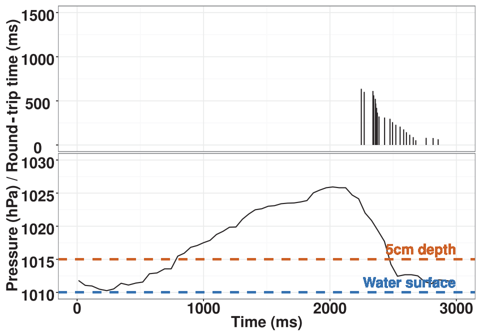
results showed that network connectivity can be sustained only up to a depth of 20cm. To enable consistent wireless communication, a device should not be submerged too deeply for a long time. We carefully reviewed the characteristics of swimming motions and submersion patterns of each body part while swimming. We found that the back and the abdomen are consistently immersed. For example, the back is immersed while swimming backstrokes, and the abdomen is immersed while swimming the other styles. In contrast, the arm and the wrist emerge (or nearly emerge) from the water during the arm recovery phase for each stroke, providing opportunities for packet exchanges. Therefore, we placed the smartphone on the participant's upper arms.

3.3.2. Measurement Results. Figure 11 shows the cumulative distribution of RTT and PLR values with respect to the different swimming styles. Overall, LTE showed better performance than 802.11g. The mean RTT of LTE was shorter than that of 802.11g (LTE: 320.63ms; 802.11g: 543.60ms). In addition, LTE did not have any notable packet loss compared with 802.11g (LTE: 0.07; 802.11g: 0.58) due to its robust retransmission mechanism.

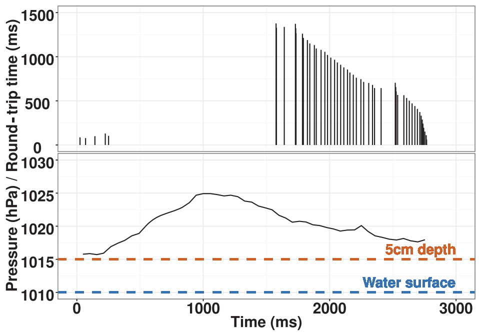
We found that swimming styles greatly affected network performance. Swimming breaststroke produced significantly longer RTT and higher PLR values for both technologies than the other styles. In 802.11g, the mean RTT and PLR for swimming breaststroke were 1,941.48ms and 0.89; in LTE, the mean RTT and PLR for swimming breaststroke were 583.50ms and 0.21. On the other hand, swimming other styles produced relatively higher network performance: in 802.11g, the mean RTT and PLR were lower than 650ms and 0.52; in LTE, the mean RTT and PLR were lower than 350ms and 0.21. These differences resulted from the varying water immersion patterns for each swimming style. As shown in Figure 12, packets were transmitted more rapidly as the depth of the upper arm was decreased. From a form standpoint, a swimmer's upper arm was consistently submerged when doing the breaststroke, whereas it periodically broke the water surface when doing the freestyle. Thus, swimming the breaststroke makes the channel conditions poorer than the other styles and brings about a longer RTT value and notable packet loss. In our measurements, we did not explore effects of 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, and thus further investigation is needed.

4. SWIMMING ACTIVITY RECOGNITION

We devised a real-time swimming activity recognition module called *StrokeSense* to adapt swimming activities into a social exergame. Among a variety of swimming activities, StrokeSense detects stroke timing, swimming styles, and turns using multiple sensors, including an accelerometer, a gyroscope, a magnetometer, and a barometer. In



(a) RTT and pressure traces of LTE (LG U+) while performing a freestyle stroke



(b) RTT and pressure traces of LTE (LG U+) while performing a breaststroke stroke

Fig. 12. RTT and pressure traces of LTE (LG U+) while performing a single stroke.

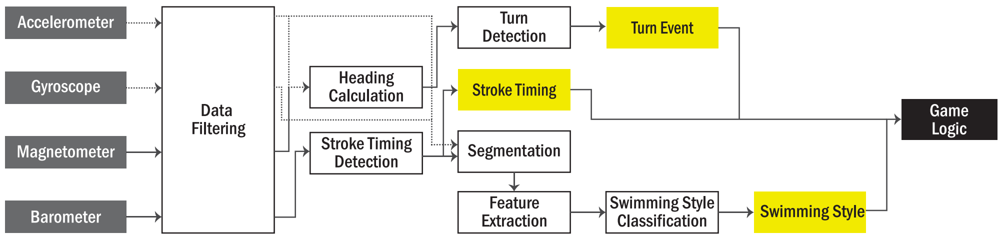


Fig. 13. Data processing flow of StrokeSense.

this section, we give an overview and list the requirements of StrokeSense. We then elaborate on our algorithms for turn detection, swimming style recognition, and stroke timing detection.

4.1. StrokeSense Overview

As shown in Figure 13, StrokeSense is composed of three submodules for turn detection, stroke timing detection, and swimming style classification. When a game begins, sensor data from the onboard magnetometer, accelerometer, gyroscope, and barometer are smoothed using a digital filtering scheme. Filtered magnetic field values are used for calculating the current heading and detecting turn events. A swimmer's arm pull action (i.e., stroke timing) is detected by means of the peak detection technique, using barometer signals. Whenever stroke timing is detected, StrokeSense classifies the current swimming style using three-axis accelerometer and gyroscope data between two consecutive stroke timing points, and it reports the classified swimming style and the timing of the arm pull occurrence to the game logic.

Using multiparty communication and real-time game interaction, StrokeSense is designed to satisfy the following two requirements:

—A user's device must be secured on the upper arm to enable multiuser communication via a wireless network. Other positions, such as the back, abdomen, and wrist, were also considered. The back/abdomen is unsuitable, as it is completely immersed during certain strokes. For example, the back is submerged with backstroke, and the abdomen is submerged with most other stroke types. The arm/wrist, however, emerges repeatedly from the water during any type of stroke. In our prototype, we chose the upper arm for placement of the device on the candidates.

—StrokeSense must provide highly accurate real-time recognition of turn events, stroke timing, and swimming styles, as they are used as the game inputs. Prior studies were limited to offline sensor data processing for the recognition of swimming styles [James et al. 2004; Marshall 2013; Siirtola et al. 2011]. Furthermore, none of those studies considered the upper arm for the smartphone location. StrokeSense complements prior studies in that we detail real-time algorithms, report comprehensive evaluation results, and demonstrate actual implementation in a multiplayer game. The key challenges were (1) devising an effective way to detect different types of turns (e.g., open turn or flip turn), (2) designing a robust algorithm for stroke timing detection to provide sufficient interactivity, and (3) building a classifier that best discriminates between different swimming styles.

4.2. Sensor Selection and Data Collection

We first explored what sensors should be used for data collection. As in the previous studies, we used motion sensors, such as an accelerometer, a gyroscope [James et al. 2004; Lee et al. 2013; Marshall 2013; Siirtola et al. 2011; Slawson et al. 2008], and a magnetometer [Lester et al. 2006; Junker et al. 2008]. Furthermore, we considered a barometer to measure ambient pressure, which includes atmospheric pressure on the free surface and hydrostatic pressure due to the weight of the water column (when the device is underwater). The ambient pressure changes significantly with the the depth of the water and very little with the height in the air. For example, a change in the ambient pressure of 1 hectopascal (hPa) requires a change in depth of 0.01m underwater but a change in height of 7.9m in the air. The sensing error of the barometer is small (less than 0.2hPa) [Muralidharan et al. 2014], and thus it can be used as a valuable information source for activity recognition underwater.

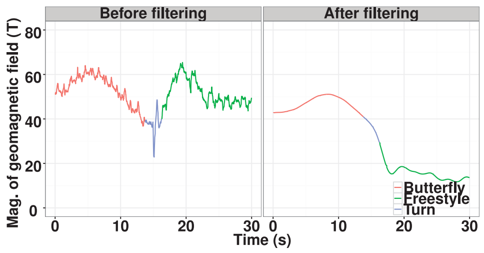
We collected data using the Casio G'zOne Command 4G LTE, a rugged Android smartphone (OS version 4.0.3). The smartphone supports use of an LTE mobile network and included IPX7 water resistance (covering water immersion up to 1m), as well as various sensors, such as an accelerometer, gyroscope, magnetometer, and barometer.

We recruited 11 participants (1 female and 10 males), who were between ages 19 and 26 years from a campus swimming club. The swimming pool used for the experiments had four 25m lanes and a depth ranging from 1.3 to 1.7m. Each participant was asked to place the smartphone on the upper left arm using an armband and to swim four lengths for each stroke type, eight round-trips in total. While the participants swam, we collected data from the three-axis accelerometer, three-axis gyroscope, three-axis magnetometer, and barometer, and recorded video clips to annotate activity tags for the following swimming movements: *turn, freestyle, butterfly, backstroke*, and *breaststroke*. We further refined the boundary data between tags as we investigated data traces visually.

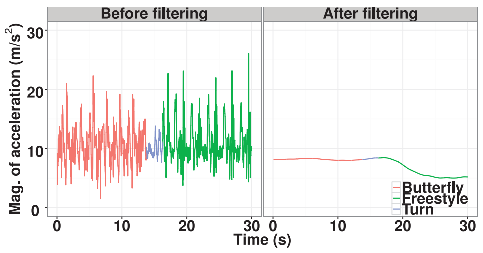
4.3. Turn Event Detection

When swimming in a pool, a swimmer makes a turn at the end of the lane to continue swimming. The number of turns is generally used to track the distance that an individual swims. Therefore, detecting turning events help swimmers measure the amount of exercise they perform. In this section, we introduce the core idea for detecting a turn event and then illustrate our turn detection algorithm based on a heading calculation technique using magnetometer values.

4.3.1. Turn Event: Change of Heading. As mentioned in Section 2.1, there are various types of turns in swimming, which vary depending on swimming styles. Less-skilled swimmers may just turn their back at the end of the lane. In general, a turn often involves a complex series of movements, such as rotation, pushing, gliding, underwater



(a) Geomagnetic field data traces before and after applying a low-pass filter



(b) Acceleration data traces before and after applying a low-pass filter

Fig. 14. Sensor data traces before and after applying a low-pass filter.

kicking, and pull-out. Therefore, it is too technically challenging to recognize a turn event itself. We instead simplified a turn event as a change in consecutive headings, because a swimmer's body is rotated to the opposite directions after a turn has completed.

In a swimming scenario, swimmers mount a device on their upper arms, which means that a magnetometer is not only tilted but also dynamically moving. Such conditions make the magnetometer signals fluctuate rapidly; thus, estimating headings is an error-prone task. We resolved this by (1) removing the high-frequency noises caused by arm strokes and (2) estimating the heading of a device by transforming device coordinates to global coordinates.

The first error reducing technique was easily achieved with a low-pass filter. Because each turn event can occur every 10 seconds at best (the 50m freestyle swimming world record is 20.26 seconds for a 2×25 m lane [Fina 2015a] and 20.91 seconds for a 50m lane [Fina 2015b]), our filter passes the signals over a period of 10 seconds (cutoff frequency: 0.1Hz). As shown in Figure 14, filtered sensor data does not fluctuate rapidly, so we assume that a device is stable.

To estimate the heading, we apply the TRIAD algorithm, which determines the orientation (i.e., heading, elevation, and bank) by finding a rotation matrix with two pairs of vector observations [Shuster and Oh 1981]. Let $\mathbf{G_D}$ and $\mathbf{B_D}$ be three-axis acceleration and geomagnetic values measured by a device, respectively. We can then find a 3×3 rotation matrix $\mathbf{R_{D\to I}}$, which transforms a device's coordinate system to an intermediate coordinate system as follows:

$$\mathbf{R_{D\to I}} = \left[\mathbf{R_{D\to I,1}}\ \ \mathbf{R_{D\to I,2}}\ \ \mathbf{R_{D\to I,3}}\right], \tag{1}$$

where

$$\mathbf{R_{D\to I,1}} = \frac{\mathbf{G_D}}{|\mathbf{G_D}|},\quad \mathbf{R_{D\to I,2}} = \frac{\mathbf{G_D} \times \mathbf{B_D}}{|\mathbf{G_D} \times \mathbf{B_D}|},\quad \mathbf{R_{D\to I,3}} = \mathbf{R_{D\to I,1}} \times \mathbf{R_{D\to I,2}}. \tag{2}$$

Let $\mathbf{G_E}$ and $\mathbf{B_E}$ be three-axis acceleration and geomagnetic values in the geographic coordinate system, respectively:

$$\mathbf{G_E} = \left[0\ \ 0\ \ g\right],\quad \mathbf{B_E} = \left[0\ \ m\ \ 0\right], \tag{3}$$

where g is the magnitude of the gravitational force and m is the magnitude of the geomagnetic field. Then we can find another 3×3 rotation matrix $\mathbf{R_{E\to I}}$, which transforms this coordinate system to the intermediate coordinate system as follows:

$$\mathbf{R_{E\to I}} = \left[\mathbf{R_{E\to I,1}}\ \ \mathbf{R_{E\to I,2}}\ \ \mathbf{R_{E\to I,3}}\right], \tag{4}$$

where

$$\mathbf{R_{E\to I,1}} = \frac{\mathbf{G_E}}{|\mathbf{G_E}|},\quad \mathbf{R_{E\to I,2}} = \frac{\mathbf{G_E} \times \mathbf{B_E}}{|\mathbf{G_E} \times \mathbf{B_E}|},\quad \mathbf{R_{E\to I,3}} = \mathbf{R_{E\to I,1}} \times \mathbf{R_{E\to I,2}}. \tag{5}$$

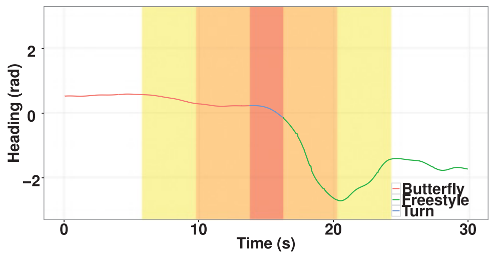


Fig. 15. Illustration about three cases of predicted turn events (red: close; orange: tight; yellow: loose).

From those two matrices, $\mathbf{R_{D\to I}}$ and $\mathbf{R_{E\to I}}$, we get the final rotation matrix, which transforms the device coordinate system to the Earth coordinate system as follows:

$$\mathbf{R_{D\to E}} = \mathbf{R_{E\to I}}\mathbf{R_{I\to E}} = \mathbf{R_{E\to I}}\mathbf{R_{E\to I}}^T = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}. \tag{6}$$

We can convert the rotation matrix $\mathbf{R_{D\to E}}$ to the Euler angles, which depend on the $Y_1X_2Z_3$ convention. The heading ψ is given as follows:

$$\psi = \text{atan2}(r_{21}, r_{22}). \tag{7}$$

We can then detect a turn by continuously monitoring significant heading changes.

4.3.2. Algorithm Design and Evaluation. Continuous three-axis geomagnetic field and three-axis acceleration values were sampled at 50Hz and 100Hz, respectively. The measurements were obtained by selecting SENSOR_DELAY_FASTEST setting in the Android SensorManager class. Sampled values were filtered using a Butterworth low-pass filter, which passes the signals over a period of 10 seconds (cutoff frequency: 0.1Hz). We then calculated heading values using equations described earlier. Our turn detection algorithm found local maxima and local minima in the consecutive headings for the most recent 7 seconds. If the differences between the local maximum and local minimum were above a certain threshold, a turn event was reported. We used a time lag threshold, during which turning events were ignored, to prevent erroneous and duplicated detections. In our implementation, heading and time lag thresholds were empirically set as 1.0rad and 12 seconds, respectively.

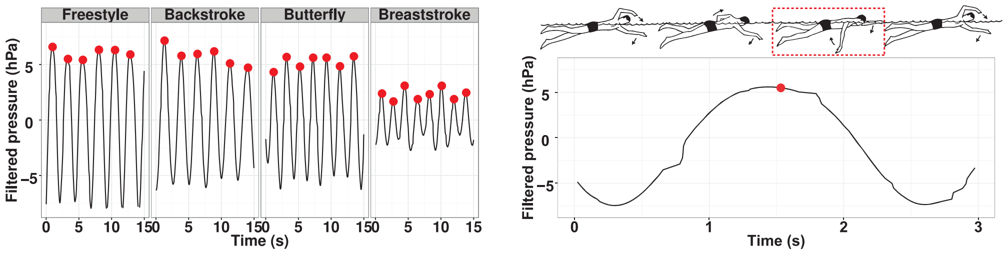
To evaluate our turn detection algorithm, we divided four cases of predicted turn events: *overall*, *loose*, *tight*, and *close* (Figure 15):

—The *overall* case compares the number of predicted turns to the number of observed turns.
—The *loose* case requires that a predicted turn falls within 8 seconds of the corresponding observed turn.
—The *tight* case requires that a predicted turn falls within 4 seconds of the corresponding observed turn.
—The *close* case requires that a predicted turn falls within the exact boundaries of the corresponding observed turn.

In each case, we calculated precision value, the proportion of correctly predicted turns against all predicted turns, and recall value, the proportion of correctly predicted turns against all actual turns, using the precollected dataset mentioned in Section 4.2. Table I shows the precision and recall values for each case. Overall, our algorithm performed well when detecting the occurrence of turn events; we achieved over 85% for precision and recall values in the overall case and loose cases. However, the method

Table I. Evaluation Results of Turn Detection Algorithm

Case	Precision	Recall
Overall	96.2%	100.0%
Loose	85.7%	90.0%
Tight	70.5%	74.0%
Close	27.6%	29.0%



(a) Filtered pressure data traces while swimming

(b) Pull action and pressure data traces while performing a freestyle stroke (red rectangle: pull action)

Fig. 16. Description of a stroke timing.

was not suitable for reporting the exact timing of a turn; the close case showed much lower performance. Our algorithm cannot support on-time turn detection because heading values do not significantly change while making a turn; the differences between consecutive heading values only differ greatly after a turn is completed.

4.4. Stroke Timing Detection

4.4.1. 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 it 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 16 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 with another semicircle underwater. Such periodic surficial transitions create clear periodic cycles in barometer signals. As mentioned in Section 4.2, diving by every centimeter depth increases the barometer output by 1hPa. 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 swimmers exert a strong muscular force on their arms; therefore, we can leverage this as a highly intuitive timing to provide auditory feedback for each stroke.

In addition, 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 16, 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

Table II. Evaluation Results of Stroke Timing Detection for Each Swimming Style

Swimming Style	Precision	Recall
Freestyle	98.8%	100.0%
Backstroke	99.7%	99.7%
Breaststroke	87.3%	99.5%
Butterfly	98.3%	100%

differences in swimming motion, such as the angle of the arms and acceleration rates. It is worth noting that stroke timing detection methods based on motion sensors [Siirtola et al. 2011] 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.

4.4.2. Algorithm Design and Evaluation. In our experimental device, the CA-201L, with the SENSOR_DELAY_FASTEST setting in the Android SensorManager class, barometer signals were sampled at 33Hz, which is fast enough to provide rapid action detection. To use the data that were only influenced by strokes and to ignore the atmospheric pressure, we passed those signals that had a similar period of a single stroke using a Butterworth band-pass filter—that is, a period of 1.5 seconds to that of 4 seconds, or 0.25Hz to 0.67Hz. From consecutive smoothed signals, local maximum and minimum peaks were found, and every significant local maximum peak was reported as a stroke timing. 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 observation that it takes a certain amount of time to complete each stroke. We empirically set the pressure threshold to 2.5hPa and the time threshold to 1.5 seconds, which yielded a sufficient detection performance for playing the game. In particular, the pressure threshold of 2.5hPa was large enough that we could ignore the 0.2hPa error mentioned in the prior work [Muralidharan et al. 2014]. To the best of our knowledge, this was one of the earliest attempts to employ a barometer for motion detection.

We evaluated the performance of the proposed stroke timing detection method by using the precollected dataset described in Section 4.2. We used the manually ground-tagged stroke timing data obtained from videos. As shown in Table II, the detection results are comparable to the state-of-the-art stroke counter described in Siirtola et al. [2011]. Again, it should be noted that StrokeSense uses one common stroke timing detection algorithm, whereas Siirtola et al. [2011] 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.

4.5. Swimming Style Classification

To classify four types of swimming styles (i.e., freestyle or front crawl, backstroke or back crawl, butterfly, and breaststroke), we designed different swimming style classification algorithms by using two models, a hidden Markov model (HMM) and a support vector machine (SVM). Here, an HMM is a stochastic model that is comprised of sequences of observations and hidden states. HMM has been implemented in a variety of areas, including protein structure prediction [Karplus 2009], handwriting recognition [Hu et al. 1996], speech recognition [Huang et al. 1990], and gesture classification [Park et al. 2011]. We considered the HMM to test whether it is possible to exploit the temporal correlation of swimming activities, such as rotation angle changes for different swimming styles. An SVM is a nonprobabilistic classification model that finds a separating hyperplane that best discriminates a training dataset. The SVM has been used in a wide range of domains, such as activity recognition [Sadanand and Corso 2012; Morris et al. 2014], speech turn detection [Jayagopi et al. 2009], image classification [Chapelle et al. 1999; Lin et al. 2011], and document categorization [Joachims 1998]. We elaborate on each swimming classification algorithm, including preprocessing, feature extraction, and implementation. In addition, we show the classification performances of these two algorithms, as well as the performance change due to differences in swimming skill.

4.5.1. HMM-Based Swimming Style Classification. To perform each swimming style, swimmers must rotate their arm in a particular sequence. Thus, we can represent each swimming style as a specific sequence of an arm's rotation angles. Based on this idea, our algorithm is composed of the following processes: (1) segmenting angular speed data obtained from a gyroscope into the lengths of single stroke windows, (2) estimating rotations for every timestep, (3) generating an observation sequence through transforming rotations into a finite number of observations, and (4) building a left-to-right discrete HMM classifier for each swimming style using generated observation sequences.

Preprocessing and feature calculation. Our experimental device, CA-201L, reports three-axis angular speed data obtained from a gyroscope every 10ms. We smoothed this data using low-pass digital filters to suppress high-frequency noise. In our dataset, the duration of a stroke ranged from 1.5 to 4 seconds. Thus, we used a low-pass filter to remove the signals whose period was shorter than 1.5 seconds. We then segmented consecutive data into the lengths of single stroke windows using the proposed stroke timing detection algorithm. To estimate the rotation angles of every data point, we employed a quaternion, which is composed of a four-dimensional vector (i.e., one real and three imaginary parts):

$$\begin{aligned}\mathbf{q} &= (w, x, y, z)\\ &= w + xi + yj + zk,\end{aligned} \tag{8}$$

where

$$w, x, y, z \in \mathbb{R}, \quad i^2 = j^2 = k^2 = ijk = -1. \tag{9}$$

The quaternion representation has frequently been used to calculate rotations in three-dimensional space, because it does not suffer from gimbal lock (the degeneration of the degree of freedom) and can be utilized with faster computing speed and more efficient memory usage than a rotation matrix representation. At every timestep, we integrated one sample of three-axis angular speed data over time to estimate the magnitude of the rotation angle, and converted each sample and its magnitude (i.e., axis-angle representation) into a quaternion. Let $\boldsymbol{\omega}_t$ be a three-dimensional angular speed vector at time t, and let $\mathbf{u}_t$ be a unit vector of $\boldsymbol{\omega}_t$ at time t. Then a quaternion at time t, $\mathbf{q}_t$, is given by

$$\mathbf{q}_t = \left(\cos\frac{|\boldsymbol{\omega}_t|}{2}, \sin\frac{|\boldsymbol{\omega}_t|}{2}u_{t,x}, \sin\frac{|\boldsymbol{\omega}_t|}{2}u_{t,y}, \sin\frac{|\boldsymbol{\omega}_t|}{2}u_{t,z}\right), \tag{10}$$

where

$$\boldsymbol{\omega}_t = (\omega_{t,x}, \omega_{t,y}, \omega_{t,z}), \quad \mathbf{u}_t = (u_{t,x}, u_{t,y}, u_{t,z}) = \left(\frac{\omega_{t,x}}{|\boldsymbol{\omega}_t|}, \frac{\omega_{t,y}}{|\boldsymbol{\omega}_t|}, \frac{\omega_{t,z}}{|\boldsymbol{\omega}_t|}\right). \tag{11}$$

For constitute quaternions over time, we represent the aggregated quaternion $\mathbf{Q}_t$, a total rotation until time t, as follows:

$$\begin{aligned}
\mathbf{Q}_0 &= \mathbf{q}_0 = (1, 0, 0, 0)\\
\mathbf{Q}_1 &= \mathbf{q}_1\mathbf{Q}_0\\
\mathbf{Q}_2 &= \mathbf{q}_2\mathbf{Q}_1\\
&\vdots\\
\mathbf{Q}_n &= \mathbf{q}_n\mathbf{Q}_{n-1}.
\end{aligned} \quad . \tag{12}$$

Generating observation sequences for HMM. In the next step, we transform aggregated quaternions into a finite number of observations. To do this, we converted the equally divided Euler angles into quaternions. For example, each Euler angle (i.e., yaw, pitch, and roll) can be equally divided into two parts: $[-\pi, 0)$, $[0, \pi]$ for yaw and roll, and $[-\pi/2, 0)$, $[0, \pi/2]$ for pitch. We then calculated a centroid for each interval (e.g., $-\pi/2$ for $[-\pi, 0)$). Those centroids were converted into quaternions, which depend on the $Y_1Z_2X_3$ convention, as follows:

$$\begin{aligned}
Q(\mathbf{c}_i) =& Q(\text{yaw, pitch, roll})\\
=& Q(\theta, \phi, \psi)\\
=& \Big(\cos\frac{\theta}{2}\cos\frac{\phi}{2}\cos\frac{\psi}{2} - \sin\frac{\theta}{2}\sin\frac{\phi}{2}\sin\frac{\psi}{2},\\
& \sin\frac{\theta}{2}\sin\frac{\phi}{2}\cos\frac{\psi}{2} + \cos\frac{\theta}{2}\cos\frac{\phi}{2}\sin\frac{\psi}{2},\\
& \sin\frac{\theta}{2}\cos\frac{\phi}{2}\cos\frac{\psi}{2} + \cos\frac{\theta}{2}\sin\frac{\phi}{2}\sin\frac{\psi}{2},\\
& \cos\frac{\theta}{2}\sin\frac{\phi}{2}\cos\frac{\psi}{2} - \sin\frac{\theta}{2}\cos\frac{\phi}{2}\sin\frac{\psi}{2}\Big),
\end{aligned} \tag{13}$$

where $\mathbf{c_i}$ is the i-th centroid and $Q(\mathbf{c_i})$ is a quaternion conversion of $\mathbf{c_i}$. For every aggregated quaternion, we calculated the index of the nearest quaternion centroid, which was obtained from consecutive gyroscope data. The distance measure between two quaternions, $\mathbf{q_1}$ and $\mathbf{q_2}$, is given by

$$\begin{aligned}
d(\mathbf{q}_1, \mathbf{q}_2) &= 1 - \mathbf{q}_1 \cdot \mathbf{q}_2\\
&= 1 - (w_1, x_1, y_1, z_1) \cdot (w_2, x_2, y_2, z_2)\\
&= 1 - (w_1w_2 + x_1x_2 + y_1y_2 + z_1z_2).
\end{aligned} \tag{14}$$

Finally, we generated an observation sequence that is composed of the indices of the nearest centroids to each aggregated quaternion and then used to classify swimming styles.

Building HMM-based classifiers. From the collected data described in Section 4.2, we first generated a sequence of aggregated quaternions $\mathbf{q} = \{q_t\}_{t=1}^{T}$ in the T-length of a single stroke. We then chose k quaternion centroids $Q(\mathbf{C}) = \{Q(\mathbf{c}_i)\}_{i=1}^{k}$, which was obtained from three equally divided Euler angles. We transformed the sequence of aggregated quaternions into a sequence of observation symbols $\mathbf{o} = \{o_t\}_{t=1}^{T}$. Using these observation sequences, we defined an N-state n-th order left-to-right HMM $\lambda_{N,n} = (\mathbf{A}, \mathbf{B}, \Pi)$, where $\mathbf{A} = \{a_{ij}\}_{i,j=1}^{N}$ is the state transition probability distribution, $\mathbf{B} = \{b_i(\mathbf{o})\}_{i=1}^{N}$ is the output probability distribution, and $\Pi = \{\pi_i\}_{i=1}^{N}$ is the initial state probability distribution.

Table III. Features Extracted for SVM-Based Swimming Style Classification

Types of Raw Data	Features
Accel-X, Accel-Y, Accel-Z, Accel-Mag, Gyro-X, Gyro-Y, Gyro-Z, Gyro-Mag	Min, Max, Mean, Std. Dev.

We trained each HMM with the Baum-Welch algorithm, which estimates the optimal HMM parameters using iterative expectation-naximization (EM) steps [Welch 2003]. Our classifier included four HMMs (an HMM for each swimming style) and reported the most likely swimming style based on a given observation sequence. We denote the proposed classifier as follows:

$$M(\mathbf{C}, N, n) = (\mathbf{C}, \lambda_{N,n,freestyle}, \lambda_{N,n,backstroke}, \lambda_{N,n,butterfly}, \lambda_{N,n,breaststroke}). \tag{15}$$

4.5.2. SVM-Based Swimming Style Classification. Our SVM-based swimming style classification algorithm was built through three steps: (1) segmenting sensor data obtained from an accelerometer and a gyroscope into the lengths of single stroke windows, (2) calculating and selecting features that best discriminate between swimming styles, and (3) building an SVM-based swimming style classifier using a set of selected features.

Preprocessing and feature calculation. We smoothed three-dimensional accelerometer and gyroscope data using the same filter used by the HMM-based classifier algorithm—that is, a Butterworth low-pass filter (cutoff frequency: 0.67Hz). For each type of data, we added an additional dimension, magnitude, by calculating the root mean square of the values from the three axes. The data were divided into windows of single strokes using our stroke detection algorithm. From the four-dimensional data (X, Y, Z, and magnitude for the accelerometer and the gyroscope) in each window, we calculated features including the minimum, maximum, mean, and standard deviation. In total, we extracted 32 features from the two sensors (Table III).

Feature selection. To remove irrelevant or redundant features, we ran a wrapper subset evaluation algorithm for feature selection using the Weka Machine Learning Toolkit 3.7 [Hall et al. 2009]. This algorithm selects a subset of features by estimating the performance (e.g., accuracy, F-measure, or root mean squared error) of a target classification model using repetitive cross validation [Kohavi and John 1997]. In general, it is the best method to evaluate features, although it has a relatively high computational cost compared to other feature selection schemes [Hall and Holmes 2003]. We set the F-measure as the target performance measure and selected an SVM with the linear kernel function as the target learning algorithm, which has been frequently used in the literature for activity recognition [Sadanand and Corso 2012; Morris et al. 2014]. Using the wrapper subset evaluation algorithm with the linear SVM, we selected a set of 7 features from the initial 32 features. Table IV lists the selected features and their information gains, a measurement for estimating each feature's discriminative quality [Quinlan 1993]. We used the selected features to classify a swimmer's current swimming style.

4.5.3. Implementation and Performance Comparison. We implemented two different swimming style classifiers using open source libraries. An HMM-based classifier is based on *JaHMM 0.6.2*, which is a Java-based HMM library [Francois 2010], and an SVM-based classifier is based on *liblinear 1.9.6*, which supports fast classification of large datasets with a linear SVM [Fan et al. 2008]. We then trained those two classifiers using collected data, described in Section 4.2. For the HMM-based classification model $M(\mathbf{C}, N, n)$, we tested a variety of settings to find parameters with the best performance

Table IV. Selected Features from Wrapper Subset Evaluation and Their Information Gain (IG) Scores

Feature	IG score
Gyro-Y_Mean	1.7808
Accel-X_Min	1.1577
Accel-Y_Min	1.0910
Gyro-Mag_Max	1.0318
Gyro-Mag_Mean	0.8343
Accel-Y_Max	0.7204
Gyro-X_Max	0.3541

Table V. Accuracy Comparison of 10-Fold Cross Validation for an HMM- and an SVM-Based Classifier

Swimming Style	HMM	SVM
Freestyle	96.6%	100.0%
Backstroke	98.7%	100.0%
Butterfly	93.7%	99.5%
Breaststroke	93.1%	100.0%
Average	95.5%	99.9%

and empirically set those parameters as follows:

$$\begin{aligned} N &= 13 \\ n &= 2 \\ \mathbf{C} &= \{\mathbf{c_i}\}_{i=1}^{k} \\ &= \{(\theta, \phi, \psi)\}_{i=1}^{k} \quad \text{for } \theta \in \left(-\frac{2\pi}{3}, 0, \frac{2\pi}{3}\right) \\ &\qquad\qquad\qquad \phi \in \left(-\frac{5\pi}{12}, -\frac{3\pi}{12}, -\frac{1\pi}{12}, \frac{1\pi}{12}, \frac{3\pi}{12}, \frac{5\pi}{12}\right) \\ &\qquad\qquad\qquad \psi \in \left(-\frac{\pi}{2}, \frac{\pi}{2}\right) \\ &\qquad\qquad\qquad \mathbf{c_i} = \mathbf{c_j} \iff i = j. \end{aligned}$$

Table V shows the results of 10-fold cross validation for each classifier. Both classifiers had greater than 95% overall accuracy, with the SVM classifier reporting slightly better performance (SVM: 99.9% vs. HMM: 95.5%). The accuracy of our classifiers outperforms previous methods used for swimming style recognition; in research by Siirtola et al. [2011], a quadratic classifier using a wrist-worn accelerometer achieved 89.8% accuracy, and a method using an upper back–worn accelerometer had an accuracy of 95.3%. We conjecture that the reasons for the higher accuracy of our results is the use of a gyroscope sensor; a gyroscope is more effective than an accelerometer in monitoring body movements, as most body movements involve joint rotations [Park et al. 2011].

We also investigated changes in the classification performance depending on the swimmers. As with other physical activities, swimming involves various movements and uses almost all of the muscles and joints in the human body, meaning that there is great potential for differences between swimmers in practice. Therefore, we examine the differences in swimming motions among swimmers by using a similar approach to that of Reddy et al. [2010]. We built two model categories: swimmer-specific models and leave-one-swimmer-out models. For the swimmer-specific models, we used only one swimmer's data for training and testing at a time. Each model built was tested

Table VI. Accuracy Comparison Between Swimmer-Specific and Leave-One-Swimmer-Out Models for an HMM- and an SVM-Based Classifier

Participant	Swimmer-Specific Model	Leave-One-Swimmer-Out Model	Differences
P1	88.8%	93.0%	–4.2%
P2	68.9%	92.3%	–23.4%
P3	83.4%	66.4%	17.0%
P4	87.2%	92.9%	–5.7%
P5	92.3%	97.1%	–4.8%
P6	96.9%	76.1%	20.8%
P7	68.9%	84.7%	–15.8%
P8	94.8%	97.2%	–2.4%
P9	91.1%	83.9%	7.2%
P10	95.3%	95.6%	–0.3%
P11	85.7%	89.7%	–4.0%
Average	86.7%	88.1%	–1.4%

(a) Accuracy comparison between swimmer-specific and leave-one-swimmer-out models for an HMM-based classifier

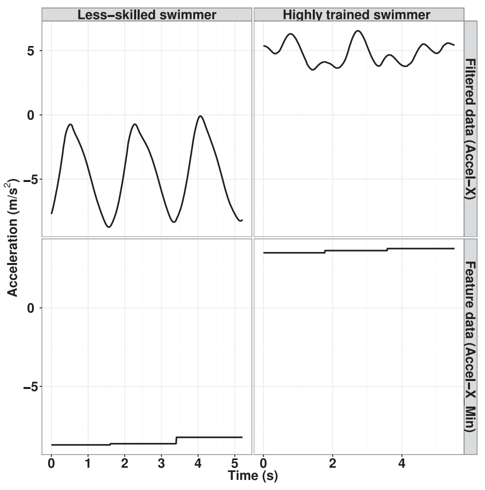
Participant	Swimmer-Specific Model	Leave-One-Swimmer-Out Model	Differences
P1	100.0%	100.0%	0.0%
P2	100.0%	100.0%	0.0%
P3	97.8%	96.9%	0.9%
P4	100.0%	100.0%	0.0%
P5	100.0%	100.0%	0.0%
P6	100.0%	100.0%	0.0%
P7	100.0%	100.0%	0.0%
P8	100.0%	100.0%	0.0%
P9	100.0%	100.0%	0.0%
P10	100.0%	100.0%	0.0%
P11	100.0%	99.4%	0.6%
Average	99.8%	99.7%	0.1%

(b) Accuracy comparison between swimmer-specific and leave-one-swimmer-out models for an SVM-based classifier

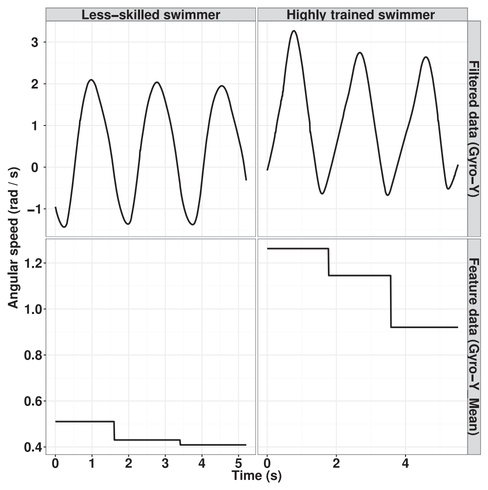
via 10-fold cross validation. For the leave-one-swimmer-out models, the classifier was trained with the data of all participants except for one, but it was tested with the data from the swimmer who was not in the training dataset.

As shown in Table VI, swimmer-specific models showed slightly lower performance with an HMM (swimmer specific: 86.7% vs. leave one swimmer out: 88.1%) and similar performance with an SVM (swimmer specific: 99.8% vs. leave one swimmer out: 99.9%). We further analyzed confusion matrices and data distributions to investigate the lower performance of the HMM for swimmer-specific models. We observed that our HMM classifier suffered from an underflow problem in swimmer-specific models. It appears that the probability distribution was not sufficiently populated with limited training data (collected per person), and a stochastic classification model does not perform well under such conditions.

4.5.4. Differences Between Less-Skilled and Highly Trained Swimmers. We conducted a pilot test with seven additional participants to verify the performance of our classification model. The seven participants (seven males; L.P1 to L.P7) were university students who were capable of performing all four stroke types, with ages ranging from 26 to 30 years. The participants occasionally swam, but none of them swam on a regular weekly basis. We expected this group to have a lower swimming skill level when compared to the swimmers with regular training. We collected their swimming motion data similarly to



(a) Acceleration data traces between less-skilled and highly trained swimmers



(b) Angular speed data traces between less-skilled and highly trained swimmers

Fig. 17. Differences of motion sensor signal between less-skilled and highly trained swimmers while performing butterfly.

the method outlined in Section 4.2 and classified stroke types using each classifier in Section 4.5.3. Surprisingly, the accuracy of the models decreased significantly, to about 50% at times, for certain participants.

To investigate the potential causes of the accuracy degradation, we examined the confusion matrices. Our HMM-based classifier failed to discriminate freestyle and butterfly from the other stroke types (butterfly: 31.0%, freestyle: 76.7% vs. backstroke: 91.5%, breaststroke: 90.9%). The SVM did not classify butterfly and breaststroke well compared to the results from freestyle and backstroke tests (butterfly: 52.9%, breaststroke: 60.7% vs. freestyle: 100.0%, backstroke: 99.2%). For both models, we found that there are two types of confusion patterns: failure to distinguish butterfly from freestyle and backstroke, and failure to distinguish breaststroke from backstroke. From this, we realized that classification accuracy may be dependent on the difficulty of swimming styles, based on the observation that freestyle and backstroke are generally easier to perform with good form 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. Although club members' skill levels were rather diverse (particularly among our participants), their swimming skills were relatively high, and their motion patterns tended to be fairly consistent with one another, due to the regular swimming drills. However, the additional participants swam as a hobby and thus were less-skilled swimmers compared to the club members.

To confirm our idea that classification accuracy may be dependent on the difficulty of the swimming style, we manually examined the motion dataset (i.e., the acceleration and gyroscope samples) to determine why such classification errors occurred. After inspecting the dataset, we found that the the acceleration patterns and rotation curves of the less-skilled swimmers were somewhat different from those in the training dataset, as shown in Figure 17. While swimmers are dragging up their arms, the arms of highly trained amateurs rotate at a higher frequency and more quickly. In addition, highly trained amateurs rotate their arm approximately 90 degrees to the front, a much more defined arm angle than that of less-skilled swimmers. Although the original training

Table VII. Accuracy Comparison of the Pilot Test Using a Swimmer-Specific Model and a Highly Trained Swimmer Model

Less-Trained Swimmer	Swimmer-Specific Model	Highly Trained Swimmer Model
L.P1	59.2%	76.9%
L.P2	51.9%	72.6%
L.P3	82.7%	81.4%
L.P4	73.9%	76.0%
L.P5	82.9%	52.9%
L.P6	92.5%	67.7%
L.P7	85.8%	80.4%
Average	72.6%	75.6%

(a) Accuracy of the pilot test using a swimmer-specific model and a highly trained swimmer model for an HMM.

Less-Trained Swimmer	Swimmer-Specific Model	Highly Trained Swimmer Model
L.P1	100.0%	51.4%
L.P2	100.0%	83.3%
L.P3	100.0%	71.4%
L.P4	100.0%	91.2%
L.P5	100.0%	68.5%
L.P6	100.0%	91.7%
L.P7	100.0%	90.0%
Average	100.0%	78.2%

(b) Accuracy of the pilot test using a swimmer-specific model and a highly trained swimmer model for an SVM

data appeared to be highly consistent across different users, the dataset from the pilot study did not clearly exhibit such patterns.

Table VII shows the classification results of each individual less-trained swimmer (L.P1 to L.P7) with the model built using (1) a same less-trained swimmer's data (swimmer-specific model) and (2) 11 highly trained swimmers' data (P1 to P11; highly trained swimmer model). Our HMM-based classifier did not show significant differences between the swimmer-specific and highly trained models, whereas the SVM-based classifier showed a significant performance increase in the swimmer-specific model. In case of limited training data, such as the swimmer-specific models, the HMM-based classifier showed lower classification performance, as stated earlier. On the other hand, the SVM-based classifier can distinguish different swimming styles, even with limited training data, as mentioned previously. The performance difference with the highly trained swimmers' models between two classifiers was not significant (3.2%). Therefore, we conclude that the SVM-based user-specific models can cover more various 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 done occasionally (say, once in a few months). To decrease the data collection burden on the swimmer, a semiautomatic collection method using an audio guide would be useful, such as a mobile application that tells a user to collect sensor data by performing one lap for each stroke, which is a level of manual data collection that would not be burdensome for casual swimmers. An alternative would be to employ a hybrid technique, where a user-independent model (with SVM or HMM) is used initially and then individual user data is collected and used to build the SVM-based classifier.

5. A CASE STUDY: MOBYDICK

We designed MobyDick as a case study, 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 action exergame pitting team members against a huge underwater monster, *Leviathan*. As with popular multiplayer online games, multiple swimmers form a team and play MobyDick collaboratively, attacking Leviathan, dodging the attack of Leviathan, and healing the wounded. Next, we outline the game modalities and design.

5.1. 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 For example, dynamically changing paces [Mueller et al. 2010; Park et al. 2012] or making additional gestures [Ahn et al. 2009] during swimming may unbalance the swimmer. As a result, we focused on the intrinsic swimming activities. We utilized two types of stroke action information as game inputs: stroke timing and swimming style (freestyle, breaststroke, backstroke, and butterfly). In addition, turn events were recorded to track swimming histories. As described earlier, MobyDick senses those activities by means of the smartphone mounted on the swimmer's upper arm with an antifriction armband.

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 audio-only game [Yuan et al. 2011], which is unlike typical video games that use audio merely to provide additional information [Ng and Nesbitt 2013]. MobyDick continuously communicates the game progress, team members' status, and interaction events through background music, narration, and sound icons.

5.2. 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 dodging, and backstroke is for healing, whereas 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")—that is, two attack earcons for freestyle and butterfly, one dodging 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, such as health point status, as well as who is currently being attacked by Leviathan. Unlike existing exergames that typically use verbal communication to facilitate social interaction, such as Jogging over a Distance [Mueller et al. 2010] and ExerLink [Park et al. 2012], 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 3 minutes, a length of time that was chosen by considering the average casual swimmers' stamina—that is, how long they are able to swim continuously. 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 the breaststroke to dodge the attack of the monster, the other swimmers are free to attack Leviathan by using freestyle or butterfly strokes. To evade Leviathan's fire attack, a victim should synchronize his or her breaststroke cycles with Leviathan's firing cycles to be under the water surface at the very moment that the fire is breathed out; otherwise, the victim's health points would be deducted. When the victims have "died"—that is, when 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 to revive the dead player more quickly by swimming the 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."

Multiplayer collaboration with social awareness cues. MobyDick features a unique mode of asymmetric multiplayer communication for collaborative game play, which is different from two-way verbal communications between players in other collaborative exercise games [Park et al. 2012] and nonverbal communications in three-dimensional virtual environments [Manninen and Kujanpää 2002]. 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 [Erickson and Kellogg 2000]. When listening to the events and progress along the game play, each swimmer is expected to perform strategic actions based on his or her own decisions. For example, a swimmer, say Delta, is under a concentrated attack from Leviathan, and her life is at stake. Having heard of this status, the swimmer, Alpha, may make a strategic decision on his own. He can continue freestyle strokes to deplete the remaining Leviathan's health points, or he can switch to the backstroke to keep Delta alive, as she is the most proficient butterfly swimmer and thus is the most important attacker. Due to the lack of explicit outbound communication, we expected that such silent and autonomous teamwork could provide social fun, where players would be able to devise better strategies as they figure out each other's playing 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. First, 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.!" As described earlier, wireless communications suffer from delayed transmission underwater. This technique allows the network packets to be updated infrequently and players not to be aware of delayed updates. Second, a MobyDick client *locally* computes one's own status change (e.g., death or revival), immediately notifies the swimmer, and remotely synchronizes the change with the server (as is the case in most interactive games). It would be annoying for the swimmer to notice latency in his or her own obvious status changes. Third, a swimmer might temporarily suffer a loss of connection that is not restored immediately. We observed that such an incident is not highly likely, but it may occur during breaststroke or butterfly. To keep players in the game during disconnection, their MobyDick clients tell them that they have been "stunned" and recommend that they use freestyle or backstroke for rapid recovery.

6. IMPLEMENTATION AND USER STUDY

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

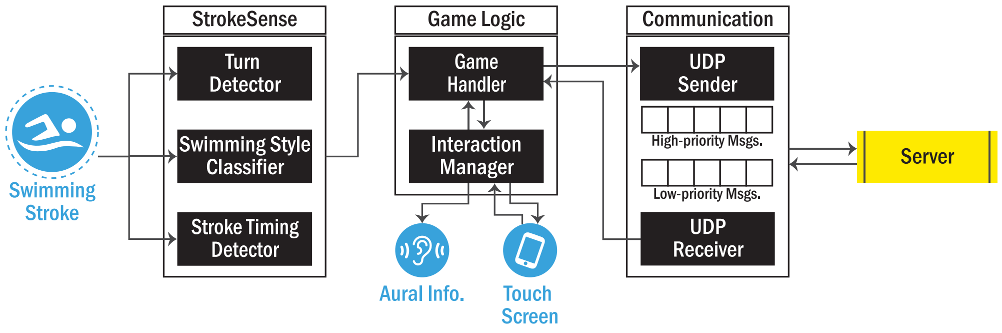


Fig. 18. Overall architecture of MobyDick.

6.1. MobyDick Implementation

As shown in Figure 18, 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* [Fan et al. 2008]. For personalized model building, a new user initially follows an audio guide whereby he or she is 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, dodging, and 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 instructions, one's own input, and other players' input, to the user. It also provides 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 100ms, and the server broadcasts state changes to all 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 classified the state messages into low and high priority. High-priority messages are used to report critical status changes, such as the death of Leviathan or a player, or the end of a game, whereas low-priority messages 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, whereas high-priority messages do not have an expiration time. The communication module also monitors wireless network connectivity to inform the game logic of a connectivity loss event and to keep track of a client's IP address to handle any IP address change after reconnection.

6.2. User Study Design

We recruited eight participants from our campus online community, with the major criteria being as follows: (1) the participants must have had more than 1 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 VIII lists the participants' demographic information. We conducted the study in a 25m, four-lane swimming pool at a local university gym. The participants were divided into two four-member teams. Each team played two rounds of games, with 10 minutes of intermission. Each lane was

Table VIII. 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

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 interviews were recorded. Two researchers transcribed and coded the interviews. We now discuss the major themes and findings of the study.

6.3. Findings

Dissociation from intrinsic swimming activity. All 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."*

Previous literature has reported that one's dissociative state effectively reduces one's perceived exertion [Pennebaker and Lightner 1980] and leads to a mental disconnection from painful sensory inputs [Morgan and Pollock 1977]. 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 felt like my stroke was 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 for themselves and Leviathan, and other swimmers' statuses. However, toward the end of a game, the participants tended to be exhausted and could not maintain a sufficient cognitive span to monitor other players' activities. P5 recalled: *"[At the last part] I couldn't listen to what others were doing. [I just] kept hitting Leviathan."*

Intuitive behavioral metaphor. The participants liked the mappings between stroke types and in-game actions. P7 stated: *"It makes good sense to me. . . . Swimming the freestyle or butterfly seems like hitting the water. . . . Backstroke seems like lying on the water and taking a 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 bonding and team spirit among swimmers (e.g., real-time interaction and sympathy with each other). P7 commented: *"It cheered me up to hear that Alpha and Bravo were attacking when I was attacking too. I felt we were the same team indeed."* In addition, 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 [Park et al. 2012], 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 online game studies reported that there are simple forms of collaboration between strangers or friends without explicit interplayer communication [Nardi and Harris 2006]. However, we found that the current design of MobyDick has the potential to facilitate highly strategic collaboration between players, even without an explicit communication channel between players.

7. 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 subjects in detail.

7.1. Exergame Review

As sensor and device technologies have advanced in recent years, creating new kinds of exergames has been an active area of research. Typical exergame design involves augmenting or visualizing existing sports activities. One way of doing this is to repurpose existing exercise equipment or develop new devices for physical activities that can be used as game inputs, such as an arm ergometer [Guo et al. 2006], an interactive treadmill [Ahn et al. 2009], a spirometer [Lange et al. 2009], a tangible ball [Kern et al. 2006], and playful gadgets [Chang et al. 2008; Chiu et al. 2009; Lo et al. 2007]. Exergame Fitness Co.[3] provides several types of exergame controllers, including game cycles and interactive floor and wall systems. Allowing for remote participation in exercise activities to 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 a Distance, which use players' physical actions directly for remote sport play, for example, Jogging over a Distance, Remote Impact, and Table Tennis for Three [Mueller and Agamanolis 2005; Mueller et al. 2009, 2010]. Sharing heart rates allows remote exercise participants to exchange their workout intensity levels [de Oliveira and Oliver 2008; Mueller et al. 2010; Nenonen et al. 2007; Stach et al. 2009]. Park et al. proposed an exergaming platform called *ExerLink* that allows remotely connected participants to use multiple heterogeneous exercise devices for game play [Park et al. 2012]. Our main contribution is to gamify swimming activities and to design and evaluate an interactive game that considers multiuser coordination.

[3]http://exergamefitness.com.

Readers can find more information about recent advances in exergames and their design principles in Campbell et al. [2008] and Sinclair et al. [2007].

7.2. Wireless Networking in the Pool

LTE performance measurement has become a topic that is of great interest to the research community. Huang et al. [2012] 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 selecting LTE-related parameters, such as tail power, had a significant effect on energy consumption. LTE displays significantly lower RTTs than those of 3G networks [Huang et al. 2013], but researchers found that the TCP has various inefficiencies, 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, such as 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. Recent smartphones, such as the Galaxy S4 Active or the Casio G'zOne, have begun to include waterproofing features, and waterproof smartphone cases are commonly used in water theme parks (e.g., for taking photos). Our measurement results provide valuable insights for novel application design with respect to 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 [Lloret et al. 2012; Sendra et al. 2013]; however, the majority of these studies have focused on using acoustic data transmission. Lloret et al. [2012] 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 and Georgakopoulos [2011] studied the effect of the frequency band on underwater Radio Frequency (RF) signal propagation. They found that propagation loss increases significantly in high-frequency bands (above 100MHz). Lloret et al. [2012] analyzed the performance differences between different WiFi channels. Our work differs from these studies in that our experimental condition involved 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 an actual swimming scenario.

7.3. Sensor-Based Swimming Analysis

Performance analysis in swimming has mostly been based on offline, manual processing of recorded images 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. [2004] analyzed the acceleration data of elite swimmers using a back-mounted accelerometer. Similarly, Slawson et al. [2008] showed that acceleration data can provide useful information for performance analysis. As the first system of its kind, the aim of SwimMaster was to provide a fine-grain assessment of swimming by extracting swimming parameters, such as velocity, arm strokes, body balance, and body rotation [Bächlin et al. 2009]. To enable fine-grain monitoring, SwimMaster uses three motion sensors: a wrist sensor, a lower back sensor, and an upper back sensor. 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, Siirtola et al. [2011] used two body-worn sensors, one on the wrist and one on the back, for the automatic classification of strokes, and they showed that accurate classification is possible with a back-mounted motion sensor. Although 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 interpersonal skill differences. Recently, Marshall [2013] and Lee et al. [2013] 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 the following ways: (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 swimming strokes; 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.

8. DISCUSSION

We now present the practical implications of our main findings and discuss the limitations of the research. The discussion is organized into the following sections: activity recognition (Section 8.1), network research (Section 8.2), game design implications (Section 8.3), application design opportunities (Section 8.4), and limitations (Section 8.5).

8.1. Swimming Activity Recognition

From our experiments, we showed that the smartphone barometer provided an accurate estimation of water depth. Previously, it was shown that atmospheric pressure (measured in the air) could be used to predict floor levels [Muralidharan et al. 2014] or to detect user activities such as idling, walking, and driving [Sankaran et al. 2014]. In our work, we systematically showed that ambient pressure significantly changes with water depth, especially compared to change with height in only the air. For example, a change in the ambient pressure of 1hPa requires a change in depth of only 0.01m underwater or a change in height of 7.9m in the air. Another potential application is to detect water emersion events for sensor data analysis (or to segment sensor data based on the events). There are significant pressure changes at the time of water emersion, and we can easily find emersion events when visually examining the dataset. Likewise, we can automatically detect the event by tracking the ratio of pressure changes over time, as there will be significant pressure increment at the time of water emersion.

This work provides highly accurate online recognition of turn events, swimming styles, and stroke timing using off-the-shelf smartphones. Our recognition procedure can be extended to be used for recognition of other water-based activities, such as aquarobics. Recognizing activities that involve water emersion can potentially benefit from detecting water emersion events. Unlike land-based exercises, water resistance makes physical activities more challenging. We showed that a user's skill level in the water must be carefully taken into account. It is difficult to accurately classify less-skilled users' activities even when using sophisticated models based on temporal sequences. Instead, we showed that a user-specific model can be an excellent alternative, resulting in highly accurate classification results.

Recently it has become increasingly popular to track personal data using smart devices, such as the Fitbit Flex and Basis B1. This interest has been extended to encompass water-based activities. Our algorithm suite can be easily extended for users to track their own swimming activities. One interesting direction for future work would be to release a self-tracking swimming application in the application store and to conduct a large-scale, in-the-wild test on swimming activity recognition. The key challenge would be developing a full-service suite and addressing practical concerns such as device heterogeneity and usability issues, which can be addressed by leveraging user-specific modeling (or device pattern mining) and a user-centered, iterative design process.

8.2. Network Research

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 considered wireless networking technologies under our experimental conditions. Despite the robustness of LTE links, when poor channel conditions were prolonged, bulk packet loss was observed; however, the likelihood of such incidents occurring was much lower than for WiFi. When network disconnection occurred, we found that it took a considerable amount of time to re-establish connectivity. In addition, water immersion patterns differed widely across different stroke types while swimming, which contributed to the RTT variations and PLR. Since water depth is related to networking performance, it is possible to use barometric sensing to estimate channel conditions and schedule packet transmissions. For example, one interesting network research application is to enable real-time audio streaming for swimming environments (e.g., delivering an instructor's voice messages in real time or listening to real-time audio/video broadcasting).

We mainly used Casio's rugged smartphones in our experiments. Two smartphones were damaged during the experiments, and we propose a few possible reasons. The rubber isolators sometimes become loose, possibly as a result of forceful body movements. Moreover, waterproof headset wires sometimes interfere with swimming strokes, pulling out the headset. Given that such events may occur during various aquatic activities, any rugged phone design should take these problems into account.

8.3. Game Design Implications

When designing a social exergame for swimming, one should carefully consider end-to-end latency, reliable transport, and network connectivity issues when designing game content and implementing interdevice communications. For example, the RTT constraints will help designers understand the degree of interactivity that a given networking technology can provide and will guide them to design game content accordingly. Although typical data exchanges are done in UDP for game development, some important game messages should be reliably delivered to the players. 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 with over-the-air networks. Likewise, a designer should consider how to handle long periods of connectivity loss during a game.

8.4. Application Design Opportunities

8.4.1. Interactive Coaching. Our work adds to the body of work on interactive coaching. It is possible to design intelligent workout programs; a simple approach is to ask users to follow existing exercise regimens [Marshall 2013] or provide customized feedback for posture correction [Bächlin et al. 2009]. Beyond such intelligent coaching, we can explore the possibilities of enabling real-time interactions between coaches and swimmers via wireless networking technologies. For example, sensor data can be processed in real time, and the results can be immediately sent to the coaches, who can give specific feedback to the swimmers (e.g., voice and visual feedback).

8.4.2. Remote Cheering. In recent years, HCI communities have designed various applications that facilitate interactions between exercisers and remote supporters. For example, HeartLink allows runners to broadcast their physical and psychological states (e.g., speed, heart rates) and online supporters to send cheers to the runners via vibration [Curmi et al. 2013]. With RUFUS, supporters can send multiple signals such as "Go Go Go" and "I'm thinking of you." These signals are delivered to the runner's wristwatch (encoded in different colors for intuitive recognition while running) [Woźniak et al. 2015]. Similarly, we consider the possibility of remote cheering applications for swimming activities (e.g., triathlon, masters' swimming).

8.4.3. Exercise Game Design. We demonstrated that swimming activities can be used as game input and designed an application called *MobyDick* that allows multiple swimmers to collaborate and hunt down a virtual monster. Our preliminary experiences with MobyDick resulted in several game design implications and provided insights into practical mobile software design for aquatic environments. As mentioned by our participants, MobyDick's system of sharing user information has a major limitation in that users tend to disregard other players' information when they become tired. One way of mitigating this problem is to support pairwise collaboration and competition as part of the game. If a user is paired with another user and asked to collaborate, we expect that they may feel less burdened since they do not need to track all of the other players' statuses. Exercise intensity should be carefully coordinated into 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 [Borg 1982]. This would allow the game logic to automatically adapt to a swimmer's changing conditions.

In general, when designing game content, we can leverage Schell's game design mechanics [Schell 2008]: (1) virtual space for game play (e.g., two/three dimensional, zero dimensional); (2) objects (e.g., characters, props, scoreboards), attributes of the objects (e.g., current position in the game space, min/max speed), and states (e.g., current speed, current position); (3) player actions that can be defined by mapping physical movements to virtual movements; (4) rules that define the space, the objects, the actions, the consequences of the actions, the constraints on the actions (operational rules and game states), and the goals; and (5) skill differences (physical/cognitive skills). With these characteristics, it is interesting to consider the key findings of user studies that attempt to understand group fitness swimming and design constraints. First, physical movements in swimming (e.g., stroke/turn and timing) must be mapped into a virtual space (or virtual movements). Second, it is important to design coordinated virtual movements that can facilitate social interactions (e.g., altering game states such as health point and position). In particular, game designers should consider designs that are less prone to connectivity loss and long latency issues. Third, lap swimming patterns and workout guidelines can be incorporated when dynamically generating workout schedules (e.g., style combination/duration, required physical efforts) [Park et al. 2012]. Fourth, players' physiological state, such as heart rate, can be considered to encourage balanced exertion [Sinclair et al. 2007].

8.5. 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 with our experiments are to an extent consistent with the measurement results for U.S. operators [Huang et al. 2012, 2013], 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 have exhibited gender bias. To evaluate whether this was the case, in one experiment we left the one female participant out, and trained and tested the model. However, we found that there was still a very high level of accuracy, indicating that gender bias may not have had any influence. Although 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 showed 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 5 to 10 groups of three to four users each and draw more certain conclusions from both qualitative and quantitative data analysis.

ELECTRONIC APPENDIX

The electronic appendix for this article can be accessed in the ACM Digital Library.

REFERENCES

Miru Ahn, Sungjun Kwon, Byunglim Park, Kyungmin Cho, Sungwon Peter Choe, Inseok Hwang, Hyukjae Jang, Jaesang Park, Yunseok Rhee, and Junehwa Song. 2009. Running or gaming. In *Proceedings of the International Conference on Advances in Computer Entertainment Technology (ACE'09)*. ACM, New York, NY, 345–348. DOI:http://dx.doi.org/10.1145/1690388.1690455

Marc Bächlin, Kilian Förster, and Gerhard Tröster. 2009. SwimMaster: A wearable assistant for swimmer. In *Proceedings of the 11th International Conference on Ubiquitous Computing (UbiComp'09)*. ACM, New York, NY, 215–224. DOI:http://dx.doi.org/10.1145/1620545.1620578

Gunnar A. Borg. 1982. Psychophysical bases of perceived exertion. *Medicine and Science in Sports and Exercise* 14, 5, 377–381.

Taj Campbell, Brian Ngo, and James Fogarty. 2008. Game design principles in everyday fitness applications. In *Proceedings of the 2008 ACM Conference on Computer Supported Cooperative Work (CSCW'08)*. ACM, New York, NY, 249–252. DOI:http://dx.doi.org/10.1145/1460563.1460603

Yu-Chen Chang, Jin-Ling Lo, Chao-Ju Huang, Nan-Yi Hsu, Hao-Hua Chu, Hsin-Yen Wang, Pei-Yu Chi, and Ya-Lin Hsieh. 2008. Playful toothbrush: Ubicomp technology for teaching tooth brushing to kindergarten children. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'08)*. ACM, New York, NY, 363–372. DOI:http://dx.doi.org/10.1145/1357054.1357115

Olivier Chapelle, Patrick Haffner, and Vladimir N. Vapnik. 1999. Support vector machines for histogram-based image classification. *IEEE Transactions on Neural Networks* 10, 5, 1055–1064. DOI:http://dx.doi.org/10.1109/72.788646

Meng-Chieh Chiu, Shih-Ping Chang, Yu-Chen Chang, Hao-Hua Chu, Cheryl Chia-Hui Chen, Fei-Hsiu Hsiao, and Ju-Chun Ko. 2009. Playful bottle: A mobile social persuasion system to motivate healthy water intake. In *Proceedings of the 11th International Conference on Ubiquitous Computing (UbiComp'09)*. ACM, New York, NY, 185–194. DOI:http://dx.doi.org/10.1145/1620545.1620574

Woohyeok Choi, Jeungmin Oh, Taiwoo Park, Seongjun Kang, Miri Moon, Uichin Lee, Inseok Hwang, and Junehwa Song. 2014. MobyDick: An interactive multi-swimmer exergame. In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems (SenSys'14)*. ACM, New York, NY, 76–90. DOI:http://dx.doi.org/10.1145/2668332.2668352

Cecil Colwin. 2002. *Breakthrough Swimming*. Human Kinetics.

Franco Curmi, Maria Angela Ferrario, Jen Southern, and Jon Whittle. 2013. HeartLink: Open broadcast of live biometric data to social networks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'13)*. ACM, New York, NY, 1749–1758. DOI:http://dx.doi.org/10.1145/2470654.2466231

Neil Davey, Megan Anderson, and Daniel A. James. 2008. Validation trial of an accelerometer-based sensor platform for swimming. *Sports Technology* 1, 4-5, 202–207. DOI:http://dx.doi.org/10.1002/jst.59

Rodrigo de Oliveira and Nuria Oliver. 2008. TripleBeat: Enhancing exercise performance with persuasion. In *Proceedings of the 10th International Conference on Human Computer Interaction with Mobile Devices and Services (MobileHCI'08)*. ACM, New York, NY, 255–264. DOI:http://dx.doi.org/10.1145/1409240.1409268

Thomas Erickson and Wendy A. Kellogg. 2000. Social translucence: An approach to designing systems that support social processes. *ACM Transactions on Computer-Human Interaction* 7, 1, 59–83. DOI:http://dx.doi.org/10.1145/344949.345004

Jakob Eriksson, Hari Balakrishnan, and Samuel Madden. 2008. Cabernet: Vehicular content delivery using WiFi. In *Proceedings of the 14th ACM International Conference on Mobile Computing and Networking (MobiCom'08)*. ACM, New York, NY, 199–210. DOI:http://dx.doi.org/10.1145/1409944.1409968

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research* 9, 1871–1874.

Johannes Färber. 2002. Network game traffic modelling. In *Proceedings of the 1st Workshop on Network and System Support for Games (NetGames'02)*. ACM, New York, NY, 53–57. DOI:http://dx.doi.org/10.1145/566500.566508

Fina. 2015a. *Fina 25m Pool World Records (as of November 1, 2015)*. Technical Report. Available at http://www.fina.org/sites/default/files/wr_25m_nov_1_2015.pdf.

Fina. 2015b. *Fina 50m Pool World Records (as of October 1, 2015)*. Technical Report. Fédération Internationale de Natation. http://www.fina.org/sites/default/files/wr_50m_oct_1_2015_redesigned.pdf.

Jean-Marc Francois. 2010. Jahmm: An Implementation of Hidden Markov Models in Java. Retrieved September 9, 2015, from https://code.google.com/p/jahmm/.

Songfeng Guo, Garrett G. Grindle, Erica L. Authier, Rory A. Cooper, Shirley G. Fitzgerald, Annmarie Kelleher, and Rosemarie Cooper. 2006. Development and qualitative assessment of the GAME$^{\text{Cycle}}$ exercise system. *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 14, 1, 83–90. DOI:http://dx.doi.org/10.1109/TNSRE.2006.870493

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA data mining software: An update. *ACM SIGKDD Explorations Newsletter* 11, 1, 10–18. DOI:http://dx.doi.org/10.1145/1656274.1656278

Mark Andrew Hall and Geoffrey Holmes. 2003. Benchmarking attribute selection techniques for discrete class data mining. *IEEE Transactions on Knowledge and Data Engineering* 15, 6, 1437–1447. DOI:http://dx.doi.org/10.1109/TKDE.2003.1245283

Emmett Hines. 2008. *Fitness Swimming* (2nd. ed.). Human Kinetics.

Jianying Hu, Michael K. Brown, and William Turin. 1996. HMM based on-line handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 18, 10, 1039–1045. DOI:http://dx.doi.org/10.1109/34.541414

Junxian Huang, Feng Qian, Alexandre Gerber, Z. Morley Mao, Subhabrata Sen, and Oliver Spatscheck. 2012. A close examination of performance and power characteristics of 4G LTE networks. In *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services (MobiSys'12)*. ACM, New York, NY, 225–238. DOI:http://dx.doi.org/10.1145/2307636.2307658

Junxian Huang, Feng Qian, Yihua Guo, Yuanyuan Zhou, Qiang Xu, Z. Morley Mao, Subhabrata Sen, and Oliver Spatscheck. 2013. An in-depth study of LTE: Effect of network protocol and application behavior on performance. In *Proceedings of the 2013 ACM SIGCOMM Conference (SIGCOMM'13)*. ACM, New York, NY, 363–374. DOI:http://dx.doi.org/10.1145/2486001.2486006

Xuedong Huang, Yasuo Ariki, and Mervyn Jack. 1990. *Hidden Markov Models for Speech Recognition*. Columbia University Press, New York, NY.

Daniel A. James, Neil Davey, and Tony Rice. 2004. An accelerometer based sensor platform for insitu elite athlete performance analysis. In *Proceedings of IEEE Sensors 2004*. IEEE, Los Alamitos, CA, 1373–1376. DOI:http://dx.doi.org/10.1109/ICSENS.2004.1426439

Dinesh B. Jayagopi, Hayley Hung, Chuohao Yeo, and Daniel Gatica-Perez. 2009. Modeling dominance in group conversations using nonverbal activity cues. *IEEE Transactions on Audio, Speech, and Language Processing* 17, 3, 501–513. DOI:http://dx.doi.org/10.1109/TASL.2008.2008238

Shan Jiang and Stavros Georgakopoulos. 2011. Electromagnetic wave propagation into fresh water. *Journal of Electromagnetic Analysis and Applications* 3, 7, 261–266. DOI:http://dx.doi.org/ 10.4236/jemaa.2011.37042

Thorsten Joachims. 1998. Text categorization with support vector machines: Learning with many relevant features. In *Machine Learning: ECML-98*. Springer, Berlin, Germany, 137–142. DOI:http://dx.doi.org/10.1007/bfb0026683

Holger Junker, Oliver Amft, Paul Lukowicz, and Gerhard Tröster. 2008. Gesture spotting with body-worn inertial sensors to detect user activities. *Pattern Recognition* 41, 6, 2010–2024. DOI:http://dx.doi.org/10.1016/j.patcog.2007.11.016

Kevin Karplus. 2009. SAM-T08, HMM-based protein structure prediction. *Nucleic Acids Research* 37, 2, W492–W497. DOI:http://dx.doi.org/10.1093/nar/gkp403

Dagmar Kern, Mark Stringer, Geraldine Fitzpatrick, and Albrecht Schmidt. 2006. Curball—a prototype tangible game for inter-generational play. In *Proceedings of the 15th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE'06)*. IEEE, Los Alamitos, CA, 412–418. DOI:http://dx.doi.org/10.1109/WETICE.2006.27

Ron Kohavi and George H. John. 1997. Wrappers for feature subset selection. *Artificial Intelligence* 97, 1-2, 273–324. DOI:http://dx.doi.org/10.1016/S0004-3702(97)00043-X

Mathieu Lacage, Mohammad Hossein Manshaei, and Thierry Turletti. 2004. IEEE 802.11 rate adaptation: A practical approach. In *Proceedings of the 7th ACM International Symposium on Modeling, Analysis, and Simulation of Wireless and Mobile Systems (MSWiM'04)*. ACM, New York, NY, 126–134. DOI:http://dx.doi.org/10.1145/1023663.1023687

Liu Lanbo, Zhou Shengli, and Cui Jun-Hong. 2008. Prospects and problems of wireless communication for underwater sensor networks. *Wireless Communications and Mobile Computing* 8, 8, 977–994. DOI:http://dx.doi.org/10.1002/wcm.654

Belinda Lange, Sheryl Flynn, Albert Rizzo, Mark Bolas, Michael Silverman, and Anna Huerta. 2009. Breath: A game to motivate the compliance of postoperative breathing exercises. In *Proceedings of the 2009 Virtual Rehabilitation International Conference (ICVR'09)*. IEEE, Los Alamitos, CA, 94–97. DOI:http://dx.doi.org/10.1109/ICVR.2009.5174212

Anna Larmo, Magnus Lindstrom, Michael Meyer, Ghyslain Pelletier, Johan Torsner, and Henning Wiemann. 2009. The LTE link-layer design. *IEEE Communications Magazine* 47, 4, 52–59. DOI:http://dx.doi.org/10.1109/MCOM.2009.4907407

Haechan Lee, Miri Moon, Taiwoo Park, Inseok Hwang, Uichin Lee, and Junehwa Song. 2013. Dungeons and swimmers: Designing an interactive exergame for swimming. In *Proceedings of the 2013 ACM Conference on Pervasive and Ubiquitous Computing Adjunct Publication (UbiComp'13 Adjunct)*. ACM, New York, NY, 287–290. DOI:http://dx.doi.org/10.1145/2494091.2494180

Jonathan Lester, Tanzeem Choudhury, and Gaetano Borriello. 2006. A practical approach to recognizing physical activities. In *Proceedings of the 4th International Conference on Pervasive Computing (PERVASIVE'06)*. 1–16. DOI:http://dx.doi.org/10.1007/11748625_1

Yuanqing Lin, Fengjun Lv, Shenghuo Zhu, Ming Yang, Timothee Cour, Kai Yu, Liangliang Cao, and Thomas Huang. 2011. Large-scale image classification: Fast feature extraction and SVM training. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'11)*. IEEE, Los Alamitos, CA, 1689–1696. DOI:http://dx.doi.org/10.1109/CVPR.2011.5995477

Jaime Lloret, Sandra Sendra, Miguel Ardid, and Joel J. P. C. Rodrigues. 2012. Underwater wireless sensor communications in the 2.4GHz ISM frequency band. *Sensors* 12, 4, 4237–4264. DOI:http://dx.doi.org/10.3390/s120404237

Jin-Ling Lo, Tung-Yun Lin, Hao-Hua Chu, Hsi-Chin Chou, Jen-Hao Chen, Jane Yung-Jen Hsu, and Polly Huang. 2007. Playful tray: Adopting Ubicomp and persuasive techniques into play-based occupational therapy for reducing poor eating behavior in young children. In *Proceedings of the 9th International Conference on Ubiquitous Computing (UbiComp'07)*. 38–55. DOI:http://dx.doi.org/10.1007/978-3-540-74853-3_3

Tony Manninen and Tomi Kujanpää. 2002. Non-verbal communication forms in multi-player game session. In *People and Computers XVI—Memorable Yet Invisible*. Springer, London, England, 383–401. DOI:http://dx.doi.org/10.1007/978-1-4471-0105-5_23

Joe Marshall. 2013. Smartphone sensing for distributed swim stroke coaching and research. In *Proceedings of the 2013 ACM Conference on Pervasive and Ubiquitous Computing Adjunct Publication (UbiComp'13 Adjunct)*. ACM, New York, NY, 1413–1416. DOI:http://dx.doi.org/10.1145/2494091.2496036

William P. Morgan and Michael L. Pollock. 1977. Psychologic characterization of the elite distance runner. *Annals of the New York Academy of Sciences* 301, 1, 382–403. DOI:http://dx.doi.org/10.1111/j.1749-6632.1977.tb38215.x

Dan Morris, T. Scott Saponas, Andrew Guillory, and Ilya Kelner. 2014. RecoFit: Using a wearable sensor to find, recognize, and count repetitive exercises. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'14)*. ACM, New York, NY, 3225–3234. DOI:http://dx.doi.org/10.1145/2556288.2557116

Florian 'Floyd' Mueller and Stefan Agamanolis. 2005. Sports over a Distance. *Computers in Entertainment* 3, 3, Article No. 4E. DOI:http://dx.doi.org/10.1145/1077246.1077261

Florian 'Floyd' Mueller, Martin R. Gibbs, and Frank Vetere. 2009. Design influence on social play in distributed exertion games. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'09)*. ACM, New York, NY, 1539–1548. DOI:http://dx.doi.org/10.1145/1518701.1518938

Florian 'Floyd' Mueller, Shannon O'Brien, and Alex Thorogood. 2007. Jogging over a Distance: Supporting a "Jogging Together" experience although being apart. In *CHI '07 Extended Abstracts on Human Factors in Computing Systems (CHI EA'07)*. ACM, New York, NY, 1989–1994. DOI:http://dx.doi.org/10.1145/1240866.1240937

Florian 'Floyd' Mueller, Frank Vetere, Martin R. Gibbs, Darren Edge, Stefan Agamanolis, and Jennifer G. Sheridan. 2010. Jogging over a distance between Europe and Australia. In *Proceedings of the 23nd Annual ACM Symposium on User Interface Software and Technology (UIST'10)*. ACM, New York, NY, 189–198. DOI:http://dx.doi.org/10.1145/1866029.1866062

Kartik Muralidharan, Azeem Javed Khan, Archan Misra, Rajesh Krishna Balan, and Sharad Agarwal. 2014. Barometric phone sensors: More hype than hope! In *Proceedings of the 15th Workshop on Mobile Computing Systems and Applications (HotMobile'14)*. ACM, New York, NY, Article No. 12. DOI:http://dx.doi.org/10.1145/2565585.2565596

Bonnie Nardi and Justin Harris. 2006. Strangers and friends: Collaborative play in World of Warcraft. In *Proceedings of the 2006 20th Anniversary Conference on Computer Supported Cooperative Work (CSCW'06)*. ACM, New York, NY, 149–158. DOI:http://dx.doi.org/10.1145/1180875.1180898

Ville Nenonen, Aleksi Lindblad, Ville Häkkinen, Toni Laitinen, Mikko Jouhtio, and Perttu Hämäläinen. 2007. Using heart rate to control an interactive game. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'07)*. ACM, New York, NY, 853–856. DOI:http://dx.doi.org/10.1145/1240624.1240752

Patrick Ng and Keith Nesbitt. 2013. Informative sound design in video games. In *Proceedings of the 9th Australasian Conference on Interactive Entertainment: Matters of Life and Death (IE'13)*. ACM, New York, NY, Article No. 9. DOI:http://dx.doi.org/10.1145/2513002.2513015

Taiwoo Park, Inseok Hwang, Uichin Lee, Sunghoon Ivan Lee, Chungkuk Yoo, Youngki Lee, Hyukjae Jang, Sungwon Peter Choe, Souneil Park, and Junehwa Song. 2012. ExerLink: Enabling pervasive social exergames with heterogeneous exercise devices. In *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services (MobiSys'12)*. ACM, New York, NY, 15–28. DOI:http://dx.doi.org/10.1145/2307636.2307639

Taiwoo Park, Jinwon Lee, Inseok Hwang, Chungkuk Yoo, Lama Nachman, and Junehwa Song. 2011. E-Gesture: A collaborative architecture for energy-efficient gesture recognition with hand-worn sensor and mobile devices. In *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems (SenSys'11)*. ACM, New York, NY, 260–273. DOI:http://dx.doi.org/10.1145/2070942.2070969

Taiwoo Park, Chungkuk Yoo, Sungwon Peter Choe, Byunglim Park, and Junehwa Song. 2012. Transforming solitary exercises into social exergames. In *Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work (CSCW'12)*. ACM, New York, NY, 863–866. DOI:http://dx.doi.org/10.1145/2145204.2145332

Ioannis Pefkianakis, Suk-Bok Lee, and Songwu Lu. 2013. Towards MIMO-aware 802.11n rate adaptation. *IEEE/ACM Transactions on Networking* 21, 3, 692–705. DOI:http://dx.doi.org/10.1109/TNET.2012.2207908

James W. Pennebaker and Jean M. Lightner. 1980. Competition of internal and external information in an exercise setting. *Journal of Personality and Social Psychology* 39, 1, 165–174. DOI:http://dx.doi.org/0.1037/0022-3514.39.1.165

J. Ross Quinlan. 1993. *C4.5: Programs for Machine Learning*. Morgan Kaufmann.

Sasank Reddy, Min Mun, Jeff Burke, Deborah Estrin, Mark Hansen, and Mani Srivastava. 2010. Using mobile phones to determine transportation modes. *ACM Transactions on Sensor Networks* 6, 2, Article No. 13. DOI:http://dx.doi.org/10.1145/1689239.1689243

Sreemananath Sadanand and Jason J. Corso. 2012. Action bank: A high-level representation of activity in video. In *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'12)*. IEEE, Los Alamitos, CA, 1234–1241. DOI:http://dx.doi.org/10.1109/CVPR.2012.6247806

Kartik Sankaran, Minhui Zhu, Xiang Fa Guo, Akkihebbal L. Ananda, Mun Choon Chan, and Li-Shiuan Peh. 2014. Using mobile phone barometer for low-power transportation context detection. In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems (SenSys'14)*. ACM, New York, NY, 191–205. DOI:http://dx.doi.org/10.1145/2668332.2668343

Jesse Schell. 2008. *The Art of Game Design: A Book of Lenses*. CRC Press, Boca Raton, FL.

Sandra Sendra, Jaime Lloret, Joel J. P. C. Rodrigues, and Javier M. Aguiar. 2013. Underwater wireless communications in freshwater at 2.4GHz. *IEEE Communications Letters* 17, 9, 1794–1797. DOI:http://dx.doi.org/10.1109/LCOMM.2013.072313.131214

SFIA. 2013. *2013 Sports, Fitness, and Leisure Activities Topline Participation Report*. Technical Report. Available at https://www.sfia.org/reports/301_2013-Sports,-Fitness,-and-Leisure-Activities-Topline-Participation-Report.

Sangho Shin, Andrea G. Forte, Anshuman Singh Rawat, and Henning Schulzrinne. 2004. Reducing MAC layer handoff latency in IEEE 802.11 wireless LANs. In *Proceedings of the 2nd International Workshop on Mobility Management and Wireless Access Protocols (MobiWac'04)*. ACM, New York, NY, 19–26. DOI:http://dx.doi.org/10.1145/1023783.1023788

M. D. Shuster and J. Oh. 1981. Three-axis attitude determination from vector observations. *Journal of Guidance, Control, and Dynamics* 4, 1, 70–77. DOI:http://dx.doi.org/10.2514/3.19717

Pekka Siirtola, Perttu Laurinen, Juha Röning, and Hannu Kinnunen. 2011. Efficient accelerometer-based swimming exercise tracking. In *Proceedings of the 2011 IEEE Symposium on Computational Intelligence and Data Mining (CIDM'11)*. IEEE, Los Alamitos, CA, 156–161. DOI:http://dx.doi.org/10.1109/CIDM.2011.5949430

Jeff Sinclair, Philip Hingston, and Martin Masek. 2007. Considerations for the design of exergames. In *Proceedings of the 5th International Conference on Computer Graphics and Interactive Techniques in Australia and Southeast Asia (GRAPHITE'07)*. ACM, New York, NY, 289–295. DOI:http://dx.doi.org/10.1145/1321261.1321313

Sian E. Slawson, Laura M. Justham, Andrew. A. West, Paul P. Conway, Mike P. Caine, and Robert Harrison. 2008. Accelerometer profile recognition of swimming strokes (P17). In *The Engineering of Sport 7*. Springer, Paris, France, 81–87. DOI:http://dx.doi.org/10.1007/978-2-287-09411-8_10

Tadeusz Stach, T. C. Nicholas Graham, Jeffrey Yim, and Ryan E. Rhodes. 2009. Heart rate control of exercise video games. In *Proceedings of the 2009 Conference on Graphics Interface (GI'09)*. 125–132.

Yu Ukai and Jun Rekimoto. 2013. Swimoid: Interacting with an underwater buddy robot. In *Proceedings of the 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI'13)*. IEEE, Los Alamitos, CA, 243–244. DOI:http://dx.doi.org/10.1109/HRI.2013.6483592

Lloyd R. Welch. 2003. Hidden Markov models and the Baum-Welch algorithm. *IEEE Information Theory Society Newsletter* 53, 4, 10–13.

Kylie Wilson and Darren Brookfield. 2009. Effect of goal setting on motivation and adherence in a six-week exercise program. *International Journal of Sport and Exercise Psychology* 7, 1, 89–100. DOI:http://dx.doi.org/10.1080/1612197X.2009.9671894

Bogdian Wozniak and Jerzy Dera. 2007. *Light Absorption in Sea Water*. Springer, New York, NY. DOI:http://dx.doi.org/10.1007/978-0-387-49560-6

Paweł Woźniak, Kristina Knaving, Staffan Björk, and Morten Fjeld. 2015. RUFUS: Remote supporter feedback for long-distance runners. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI'15)*. ACM, New York, NY, 115–124. DOI:http://dx.doi.org/10.1145/2785830.2785893

Bei Yuan, Eelke Folmer, and Frederick C. Harris Jr. 2011. Game accessibility: A survey. *Universal Access in the Information Society* 10, 1, 81–100. DOI:http://dx.doi.org/10.1007/s10209-010-0189-5

Received March 2015; revised September 2015; accepted January 2016