BRIAN A. SMITH | RESEARCH STATEMENT

I am broadly interested in human–computer interaction (HCI), the design of smart interactive systems that can enrich people's everyday lives. I love HCI because it combines the scientific approach to solving computer science problems with the humanitarian approach to understanding how people use and benefit from technology.

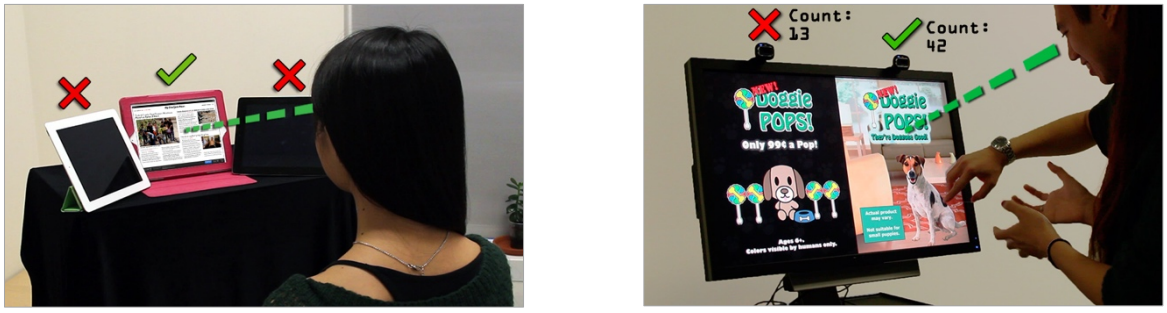
Today's HCI systems are driven largely by developers' intuitions of how users will behave, such as where they would like to see a button placed or how a keyboard should be laid out. Qwerty's home row, for example, is biased toward alphabetic order and includes "ADFGHJKL." In reality, however, human behavior — including motor control behavior — is too nuanced and complex for us to articulate in rules or guidelines.

My approach to HCI is different. I use a combination of machine learning and insightful system design to create what I believe to be the next generation of interactive systems: ones that can learn and understand the nuances of how people actually behave. Imagine a video game, for example, that can tailor its content or difficulty for a player just by recognizing the nuances of that player's controller inputs — no extra hardware or instrumentation required [3]. Such a game may also be able to tell if the player is stressed, scared, bored, or having fun using the player's controller inputs alone.

My work spans a wide breadth of topics, from computer vision [1] to text entry [2] to games [3,4] and assistive technology [4], with the same overarching research philosophy. Below are four examples of this research.

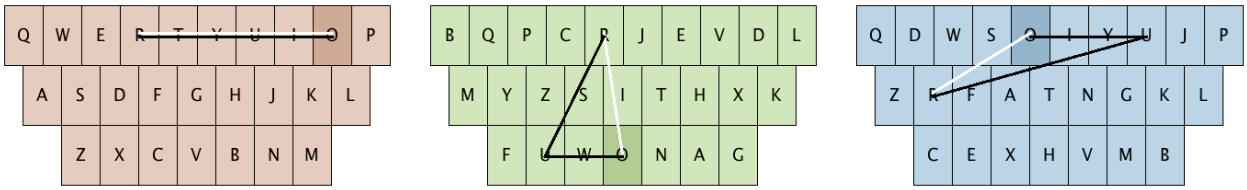
INTERACTING WITH OBJECTS BY LOOKING AT THEM

Gaze tracking systems, which estimate the exact angle at which someone is looking, suffer by only working at close range (< 80 cm) and often requiring calibration and special infrared illumination hardware. Interestingly, people themselves have difficulty determining the angle that someone else is looking as well, but seem to be very good at determining when someone else is looking *at them*. Moreover, most important interactions between people involve determining whether the other person is looking *at them*.



(a) (b)

Figure 1. Interacting with objects by looking at them. (a) My gaze locking system (eye contact detector) allows each iPad to sense when it is being looked at and wake up when it is. (b) Two ordinary webcams are placed above two ads for the same product. By counting the number of times each advertisement is viewed, we can gauge which one is more effective.



(a) Qwerty (b) Double-Optimized Layout (c) Triple-Optimized Layout

Figure 2. Keyboards optimized for gesture typing. (a) Many pairs of words such as "or" and "our" shown here share the same gesture on Qwerty, but have different gestures for (b) our "double-optimized" layout optimized for gesture clarity and gesture speed, and (c) our "triple-optimized" layout optimized for gesture clarity, gesture speed, and learnability.

As a result, I felt that HCI systems would benefit by simplifying the continuous gaze *tracking* problem into a binary gaze *locking* problem: that is, detecting eye contact instead of exact gaze direction. My colleagues and I developed a computer vision system [1] to detect eye contact directly from an image or video, exploiting the special appearance of direct eye gaze, which is a subtle difference from slightly averted gaze. Our resulting system, shown in Figure 1, is calibration-free, requires no extra hardware, and is over 90% accurate at detecting eye contact from a distance of 18 meters.

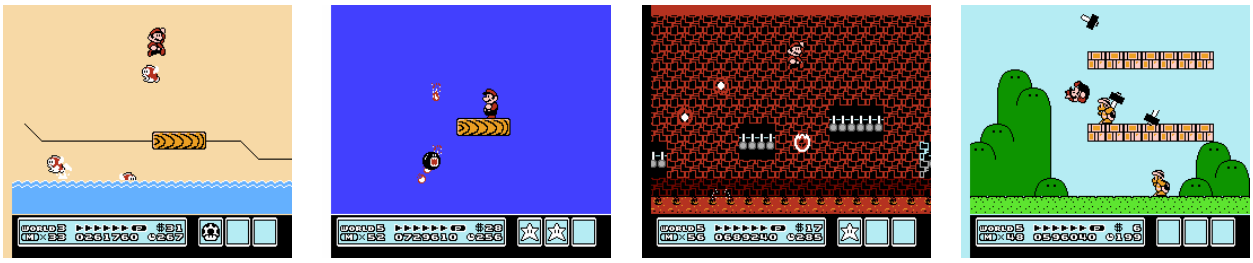
OPTIMIZING KEYBOARDS FOR GESTURE TYPING

Gesture typing, the concept of drawing word gestures on a touchscreen by swiping to connect words' letters, has been proven to be faster than touch typing but suffers from a major drawback not present in touch typing: word gesture ambiguity. Many words such as "or" and "our" have identical gestures on the Qwerty keyboard and many more such as "pretty" and "prey" have very similar gestures, all because users must swipe over unintended letters to reach intended ones. Completely changing the keyboard layout to make gestures more distinct, however, would force users to learn how to type all over again.

To see if a touchscreen keyboard layout can be changed in a beneficial and easy-to-learn way, I developed three models for predicting a given keyboard layout's worth [2] during a summer internship at Google. The models were: (1) gesture clarity, which models how distinct a keyboard layout's word gestures are; (2) gesture speed, which models how quickly users can type on a given keyboard layout based on human motor control theory; and (3) learnability, which models how easy a given keyboard layout would be to learn. By performing a rigorous optimization procedure using these models, I found that error rates can be reduced by 52% over Qwerty. Figure 2 shows two keyboards optimized for gesture typing.

A NETFLIX FOR VIDEO GAMES AND VIRTUAL REALITY EXPERIENCES

We will soon live in a world in which people can shop among many different types of virtual reality (VR) experiences on demand in pay-per-view or Netflix-type marketplaces. These experiences could include going on tours, going to the arcade, playing escape room puzzles, and visiting theme park attractions. When that happens, finding experiences that we would like based on the history of experiences that we liked before will be difficult.



(a) World 3-2 (b) World 5-9 (c) World 5-Fortress2 (d) Hammer Bros. Stage

Figure 3. Stage recommendations from controller inputs. Although these stages look quite different at first glance, they all share the same primary type of gameplay: jumping around flying enemies or projectiles. Here we see (a) Flying Cheep-Cheeps, (b) Fire Chomps and their fireballs, (c) Podoboos, and (d) Hammer Bros.

Such a problem already exists in the realm of video games. *Super Mario Maker* for the Wii U and Nintendo 3DS, for example, features a marketplace with hundreds of thousands of player-created levels that others can download and play. These levels are not curated in any way, however, making it difficult for players to find levels that they might enjoy playing.

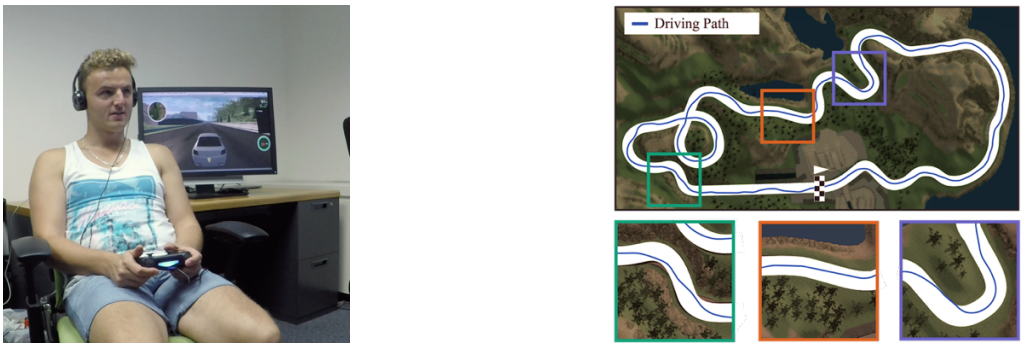
Using advanced statistical inference techniques, I built a system [3] that observes a player's controller inputs (raw actions) as they play a video game level to infer information about both the game level and the player themselves. Regarding the game level, the system infers the types of action that it fosters, such as puzzle solving and jumping on narrow platforms, and uses that information to recommend levels that play similarly. Figure 3 shows an example of this.

Regarding the player themselves, the system can learn their unique playing style and subsequently recognize them in just 20 seconds of gameplay. Neither form of understanding is obvious to a human observer, but the system can nonetheless infer these from the nuances of players' controller input behaviors.

MAKING VIDEO GAMES ACCESSIBLE TO PEOPLE WHO ARE BLIND

My most recent work is the *racing audio display (RAD)* [4], a system that makes it possible for people who are blind to play racing games as well as casual sighted players can. The RAD, shown in Figure 4, comprises two novel sonification techniques: the *sound slider* for understanding a car's speed and trajectory on a racetrack and the *turn indicator system* for alerting players to the direction, sharpness, length, and timing of upcoming turns. The RAD is an example of how an assistive technology's design can incorporate a deep understanding of people's moment-to-moment decision-making process to provide people with disabilities — in this case, blind users — with stimuli that is semantically identical to that which other users base their decisions on.

More specifically, the RAD distills many pieces of information — the car's lateral position on the track, its heading with respect to the track's, its speed, the track's width, whether the track is about to immediately turn, and more — into a single measure that is no less relevant to the process of racing than all of that information put together. Moreover, it does so in a way that gives players the freedom to decide how riskily they would like to race: whether they should cut corners by racing close to the track's inside edge or stay safe by racing closer to the track's



(a) (b)

Figure 4. Playing a racing game without sight. (a) A blind study participant playing a racing game using the racing auditory display (RAD), which outputs spatialized sound through a standard pair of headphones. Using the RAD, players can understand their car's pose, their car's speed, and the direction, sharpness, length, and timing of upcoming turns. (b) A sample driving path of this participant using the RAD. The RAD gives him enough information to cut corners consistently, even ess turns as in the lower-left insert.

center. I liken this process of distilling the many pieces of information to a more compact, salient form to that of dimensionality reduction in machine learning and statistics.

CONCLUSIONS AND FUTURE WORK

My research has been focused on building the next generation of interactive systems: systems that can understand the nuances of how people behave — how we interact with objects, our motor control process, what we prefer, and how we make decisions — on a subliminal level that we ourselves cannot articulate. I approach every problem with the same philosophy: by using a combination of machine learning and insightful system design, we can respond to very specific aspects of how people behave. This guiding philosophy has allowed me to explore a broad range of domains not typical for experts in this field. Going forward, I am excited to continue following this philosophy in order to help technology enrich our lives in ways we never thought possible.

REFERENCES

1. Smith, B. A., Yin, Q., Feiner, S. K., & Nayar, S. K. (2013). Gaze Locking: Passive Eye Contact Detection for Human–Object Interaction. *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology (UIST 2013)*. 271–280.
2. Smith, B. A., Bi, X., & Zhai, S. (2015). Optimizing Touchscreen Keyboards for Gesture Typing. *Proceedings of the 2015 CHI Conference on Human Factors in Computing Systems (CHI 2015)*. 3365–3374.
3. Smith, B. A. & Nayar, S. K. (2016). Mining Controller Inputs to Understand Gameplay. *Proceedings of the 29th Annual ACM Symposium on User Interface Software and Technology (UIST 2016).* 157–168.
4. Smith, B. A. & Nayar, S. K. (2018). The RAD: Making Racing Games Equivalently Accessible to People Who Are Blind. *To appear in the Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI 2018).*