end-user frustration. *International Journal of Human-Computer Interaction*, 2004.

[6] K. T. Claypool and M. Claypool. On frame rate and player performance in first person shooter games. *Multimedia Systems*, 2007.

[7] M. Claypool, K. Claypool, and F. Damaa. The effects of frame rate and resolution on users playing first person shooter games. In *Electronic Imaging*. Int. Society for Optics and Photonics, 2006.

[8] J. Deber, R. Jota, C. Forlines, and D. Wigdor. How much faster is fast enough?: User perception of latency & latency improvements in direct and indirect touch. In *Proc. ACM CHI*, 2015.

[9] D. E. Dilger. Agawi TouchMark contrasts iPad's fast screen response to laggy Android tablets. `http://appleinsider.com/articles/13/10/08/agawi-touchmark-contrasts-ipads-fast-screen-response-to-laggy-android-tablets`, 2013.

[10] Dot-Tec. Dot pen. `http://dot-tec.com`.

[11] Y. Endo and M. Seltzer. Improving interactive performance using TIPME. In *Proc. ACM SIGMETRICS*, 2000.

[12] Y. Endo, Z. Wang, J. B. Chen, and M. I. Seltzer. Using latency to evaluate interactive system performance. In *Proc. USENIX OSDI*, 1996.

[13] A. Goel, L. Abeni, C. Krasic, J. Snow, and J. Walpole. Supporting time-sensitive applications on a commodity OS. In *Proc. USENIX OSDI*, 2002.

[14] Google. GLSurfaceView. `http://developer.android.com/reference/android/opengl/GLSurfaceView.html`.

[15] Google. Graphics architecture. `http://source.android.com/devices/graphics/architecture.html`.

[16] Google. Implementing graphics. `http://source.android.com/devices/graphics/implement.html`.

[17] Google. invalidate(). `https://developer.android.com/reference/android/view/View.html`.

[18] M. S. Gordon, D. K. Hong, P. M. Chen, J. Flinn, S. Mahlke, and Z. M. Mao. Accelerating mobile applications through Flip-Flop replication. In *Proc. ACM MobiSys*, 2015.

[19] M. S. Gordon, D. A. Jamshidi, S. Mahlke, Z. M. Mao, and X. Chen. COMET: Code offload by migrating execution transparently. In *Proc. USENIX OSDI*, 2012.

[20] K. Ha, Z. Chen, W. Hu, W. Richter, P. Pillai, and M. Satyanarayanan. Towards wearable cognitive assistance. In *Proc. ACM MobiSys*, 2014.

[21] M. Ham, I. Dae, and C. Choi. LPD: Low power display mechanism for mobile and wearable devices. In *Proc. USENIX ATC*, 2015.

[22] B. F. Janzen and R. J. Teather. Is 60 fps better than 30?: The impact of frame rate and latency on moving target selection. In *Proc. ACM CHI*, 2014.

[23] M. B. Jones, D. Roşu, and M.-C. Roşu. CPU reservations and time constraints: Efficient, predictable scheduling of independent activities. In *Proc. ACM SOSP*, 1997.

[24] Khronos Group. glScissor(). `https://www.khronos.org/opengles/sdk/docs/man/xhtml/glScissor.xml`.

[25] S. Kolokowsky and T. Davis. *Not All Touchscreens are Created Equal - How to ensure you are developing a world class touch product*. Planet Analog: `http://www.cypress.com/file/98261`, 2010.

[26] E. Lank, Y.-C. N. Cheng, and J. Ruiz. Endpoint prediction using motion kinematics. In *Proc. ACM CHI*, 2007.

[27] J. J. LaViola. Double exponential smoothing: An alternative to Kalman filter-based predictive tracking. In *Proc. Eurographics Wrkshp. Virtual Environments*, 2003.

[28] K. Lee, D. Chu, E. Cuervo, Y. Degtyarev, S. Grizan, J. Kopf, A. Wolman, and J. Flinn. Outatime: Using speculation to enable low-latency continuous interaction for mobile cloud gaming. In *Proc. ACM MobiSys*, 2015.

[29] W. R. Mark, L. McMillan, and G. Bishop. Post-rendering 3D warping. In *Proc. SIGGRAPH Symp. Interactive 3D Graphics and Games (I3D)*, 1997.

[30] Monsoon. Monsoon power monitor. `https://www.msoon.com`.

[31] Mozilla. Off main thread compositing. `https://wiki.mozilla.org/Platform/GFX/OffMainThreadCompositing`, 2015.

[32] A. Ng, M. Annett, P. Dietz, A. Gupta, and W. F. Bischof. In the blink of an eye: Investigating latency perception during stylus interaction. In *Proc. ACM CHI*, 2014.

[33] A. Ng, J. Lepinski, D. Wigdor, S. Sanders, and P. Dietz. Designing for low-latency direct-touch input. In *Proc. ACM UIST*, 2012.

[34] NVIDIA. G-sync. `http://www.geforce.com/hardware/technology/g-sync`, 2014.

[35] P. T. Pasqual and J. O. Wobbrock. Mouse pointing endpoint prediction using kinematic template matching. In *Proc. ACM CHI*, 2014.

[36] T. Petersen. GPU boost 3 and SLI. `https://www.technopat.net/sosyal/konu/video-what-is-nvidia-fast-sync.329258`, 2016.

[37] E. Petillon. Demystify DSI I/F: `http://www.ti.com/lit/an/swpa225/swpa225.pdf`. Texas Instruments, 2012.

[38] S. C. Seow. *Designing and engineering time: The psychology of time perception in software*, chapter 3. Addison-Wesley Professional, 2008.

[39] The Chromium Projects. Multi-process architecture. `https://www.chromium.org/developers/design-documents/multi-process-architecture`, 2008.

[40] P. Tsoi and J. Xiao. Advanced touch input on iOS: Increasing responsiveness by reducing latency. The Apple Worldwide Developers Conference `https://developer.apple.com/videos/play/wwdc2015/233`, 2015.

[41] C. Velazco. Microsoft envisions a future with super-fast touchscreens. `http://techcrunch.com/2012/03/09/microsoft-demos-super-fast-touchscreen-but-will-they-ever-make-it-to-market`, 2012.

[42] WebKit. Webkit2 - high level document. `https://trac.webkit.org/wiki/WebKit2`, 2016.

[43] T. Yang, T. Liu, E. D. Berger, S. F. Kaplan, and J. E. B. Moss. Redline: First class support for interactivity in commodity operating systems. In *Proc. USENIX OSDI*, 2008.

[44] M. H. Yun, S. He, and L. Zhong. Polypath: Supporting multiple tradeoffs for interaction latency. *arXiv preprint arXiv:1608.05654*, 2016.

[45] D. W. Zimmerman. A note on interpretation of the paired-samples t test. *Journal of Educational and Behavioral Statistics*, 22(3):349–360, 1997.