SUR40 Linux: + =

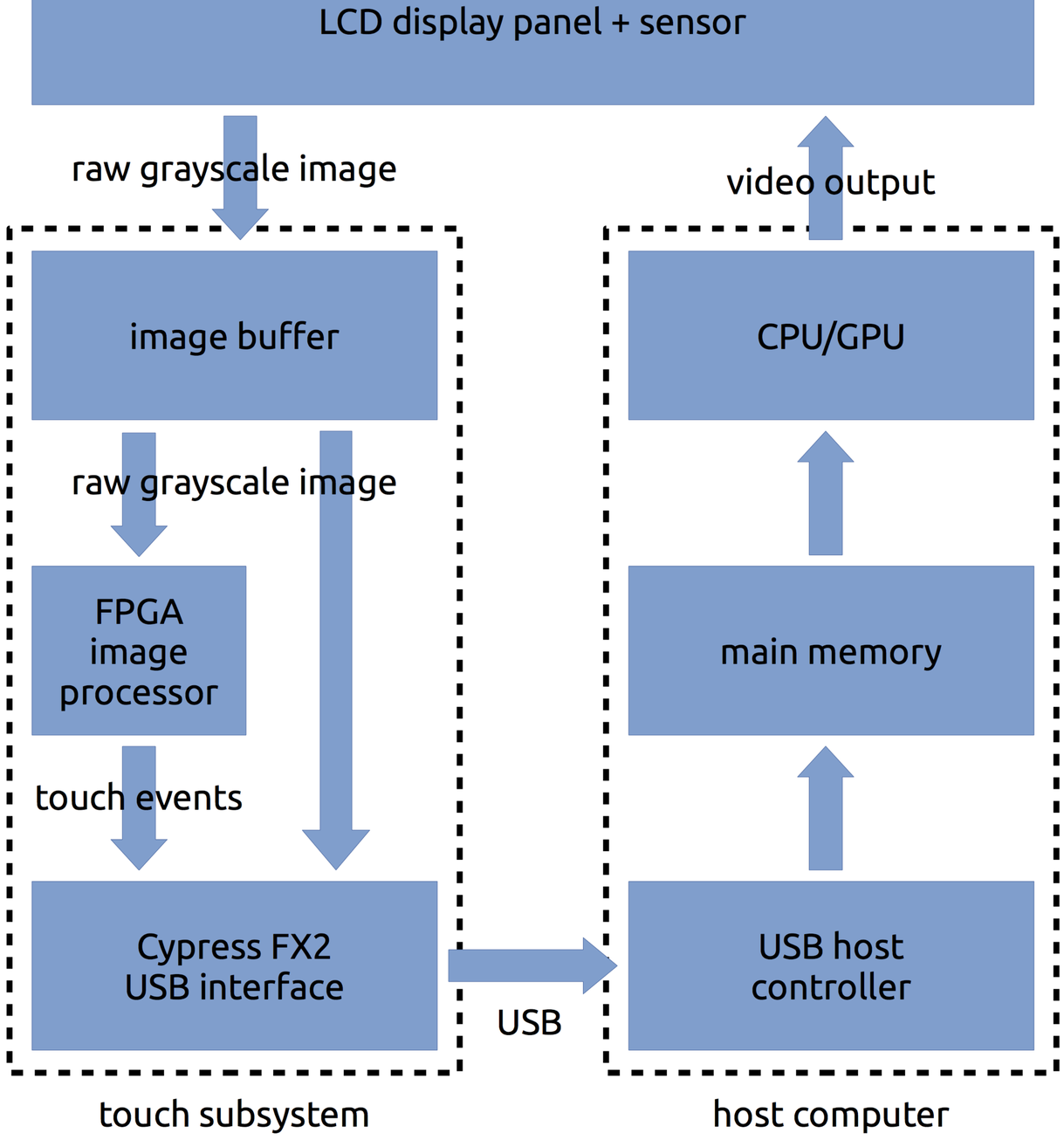
Reanimating an Obsolete Tangible Interaction Platform

Florian Echtler
Bauhaus-Universität Weimar
Contact: florian.echtler@uni-weimar.de

Mobile Media Group

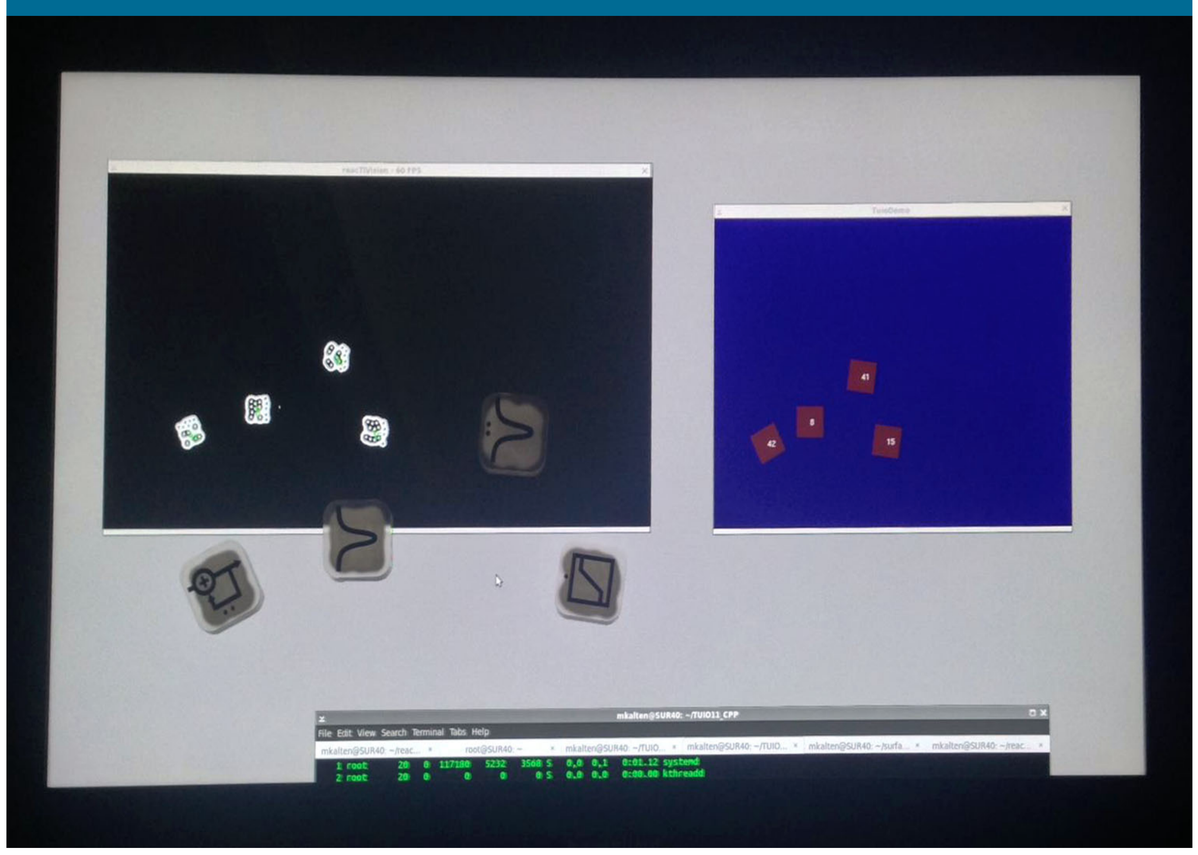
Martin Kaltenbrunner
Kunstuniversität Linz, Austria
Contact: martin.kaltenbrunner@ufg.at

interface culture



ABSTRACT

We present our research into modernizing and extending the Pixelsense/SUR40 system. By switching to a Linux operating system running a custom video driver, we are able to provide lower latency, support other types of optical tags and improve the system's robustness, particularly regarding external lighting conditions. We present an analysis of the device's internals, a comparison of quantitative performance measurements, and an outlook into extending the tangible interaction capabilities with an improved cross-platform development framework.



I Kernel Driver

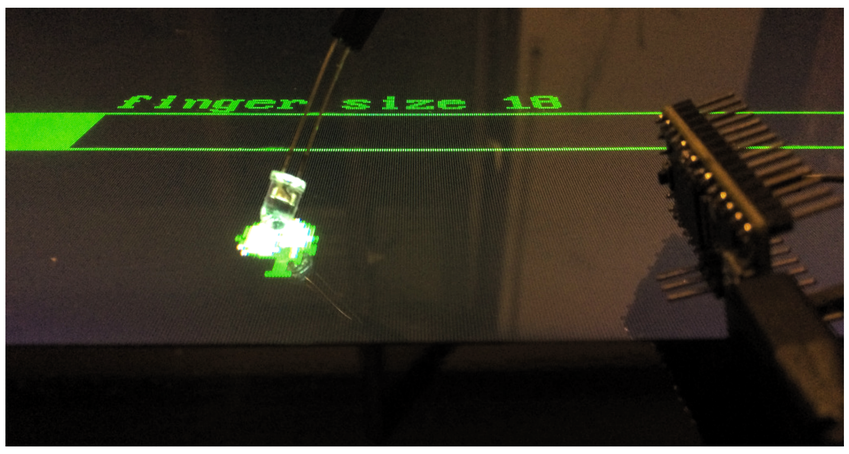
A native Linux driver provides access to both the processed touch data and the raw video stream via standard interfaces (evdev and V4L2). This driver has been included into the official mainline Linux kernel and enables users direct access to all features of the SUR40 device, including native touch support.

II User Space

By having direct access to the raw video stream, it is now possible to run vision-based interaction frameworks such as reacTIVision directly on the SUR40. In turn, this enables the detection of alternative fiducial markers in addition to the original byte tags. This more modular and open software stack allows researchers to experiment on a high-performance platform.

III Open Source

As we will continue the ongoing development of the kernel-level drivers as well as the user-space TUI tools, we are also looking for collaborators, testers and users on this platform. At this occasion we'd also like to encourage the original Samsung and Microsoft developers to support the completion of this open source effort.



LCD panel refresh (sensor readout) at 60 Hz	1 frame
internal image buffer + processing	1 frame
USB transfer (506 kB/frame at 480 MBps)	1 frame
reacTIVision and TUIO	1 frame
66ms Total Latency (before display)	4 frames

Table 1: Latency analysis (average case)

Latency Measurements & Performance Evaluation

We measured the "headless" latency with a modified TUIO client, which provided the total delay for the above V4L2 image acquisition, plus the reacTIVision image analysis and TUIO transport without display. This measurement exposed an expected latency of roughly one additional frame, resulting in an average minimum latency of 66 ms and a maximum of 82 ms by average.

Based on our experiments the perceived overall latency can be interpreted as a combination of individual delays accumulated by data acquisition, image processing, API communication and finally the display. We can also conclude that the majority of the total latency is caused by the sensor processing itself as well as a considerable delay of the display, which both eventually could be improved with an extended knowledge of the hardware internals. The Table on the left analyzes the average latency of these individual sources and how much they contribute to the observed total latency.

Find SUR40 Linux on GitHub: https://github.com/floe/surface-2.0