MOD: A Portable Instrument for Mixing Analog and Digital Drawing for Live Cinema

Ali Momeni
Carnegie Mellon University
Pittsburgh, USA
momeni@cmu.edu

Daniel McNamara
CalArts
Valencia, USA
danielmcnamara@alum.calarts.edu

ABSTRACT
This paper describes the design and fabrication of MOD (Mobile Object for Drawing)–a portable instrument for combining analog and digital drawing. MOD is intended for live performance and content creation efforts that mix common analog drawing interfaces (i.e. paper, transparency, pencil, marker) with digital cameras (webcams, scientific imaging cameras, digital magnifiers and microscopes), custom software (for keying, thresholding, looping, layer) and digital projectors. The iteration of the instrument described here combines all of these components into a single portable battery powered package that embeds the computation on a small linux computer, includes a small laser projector, and integrates custom tactile controllers. The intended uses of this instrument include experimental performance and rapid content creation; the instrument is intended to be suitable for formal (concert hall, theater) and informal (street performance, busking, parade, protest) settings, classrooms and maker spaces.

ACM Classification Keywords
H.5.m Information Interfaces and Presentation (e.g. HCI): Miscellaneous; H.5.1 Information Interfaces and Presentation (e.g. HCI): Animation

Author Keywords
Drawing; Animation; Live Cinema; Project; Projection Mapping; Performance; Interactive; Mobile; Portable; Intervention; Instrument

INTRODUCTION
This paper describes the design and fabrication of MOD (Mobile Object for Drawing)–a portable instrument for combining analog and digital drawing (see Figure 1). The instrument's design resulted from a decade of collaborative "live-drawing" performance practice by the author and Jenny Schmid, an artist with expertise in drawing and illustration. These performances were primarily achieved by combining common analog drawing interfaces (i.e. paper, transparency, pencil, marker) with digital cameras (webcams, scientific imaging cameras, digital magnifiers and microscopes), custom software on a laptop (for keying, thresholding, looping, layer) and digital projectors. The iteration of the instrument described here combines all of these components into a single portable battery powered package that embeds the computation on a small linux computer (Raspberry Pi 3), includes a small laser projector, and integrates custom tactile controllers (knobs, buttons, other sensors). The intended uses of this instrument include experimental performance and rapid content creation; the instrument is intended to be suitable for formal (concert hall, theater) and informal (street performance, busking, parade, protest) settings, classrooms and maker spaces.

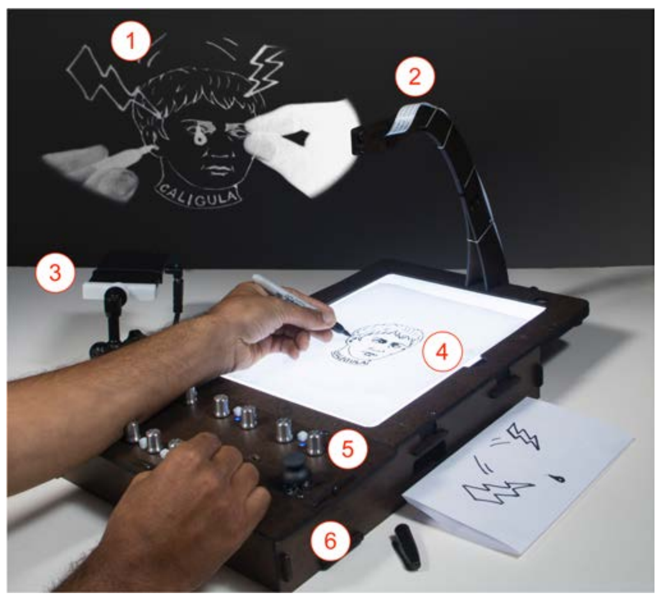

Figure 1. MOD (Mobile Object for drawing): 1) Projected image, can show live camera feed as well as three recorded layers, 2) Camera module, connected to Raspberry Pi with ribbon cable, 3) Pico projector attached with miniature magic arm, 4) back-lit drawing area, suitable for paper or transparency and pencil or marker, 5) UI panel with knobs, buttons, LEDs, power switch, 6) space between base and top surface contains Raspberry Pi, battery and voltage regulation

BACKGROUND

Motivation
The author–a musician, designer and software developer–and the collaborating artist–a master printmaker, drawer, illustrator, animator–developed a style of live performance that combines drawing, real-time video processing, sound and an outdoor

TEI '18, March 18–21, 2018, Stockholm, Sweden

DOI: https://doi.org/10.1145/3173225.3173254

Figure 2. Performance of *Gutless Warrior* by Momeni and Schmid: (Top) Architectural projection, combining short recorded video loops of participants, and drawings made on-site; (Bottom) Networked collaborative drawing table

public setting; elements of this performance genre have been described as "live cinema" by [10]. MOD's development is also a continuation of a decade of instrument building, set into motion by an urban projection collective named MAW [1] which Momeni founded in 2007. MAW performed several hundred outdoor projection performances throughout the country and abroad. In creating these events, the MAW collective created a range of mobile instruments for participatory projection events; they included mobile projection bikes, tricycles, trailers, baskets and backpacks, as well as mobile video capture instruments for gathering drawings and mugshots[2].

The author has produced a number of site-specific participatory live projection works whose documentations delve more deeply into the aesthetic, social and technical concerns of this medium. The Battle of Everyouth [3] utilized multiple mobile and wireless media capture devices intended for allowing audience participation and was conducted by 15-16 year olds (see Figure 3); Exquisite Corpse/Lavish Martyr [4] utilized three mobile device for gathering and processing paper/marker drawings from audience members during a street performance at a festival; Gutless Warrior [5] required a panoramic projection system coupled with multiple live cameras with video sampling capabilities (see Figure 2).

While the contents and the venues for these performances varied a great deal, the goals and methodologies were consistent: to activate public spaces by using playful and collaborative technologies with large-scale architectural video projections. Of all the interfaces for public engagement that MAW's performances tried, one persisted as uniquely and reliably accessible, intuitive, expressive, and limitless: Drawing. Performances usually consisted of an artist drawing with pens and markers on white paper or transparency, a camera for digitizing the image, and custom software enabling recording, playback, layering, scaling and various visual effects. Several iterations of this hybrid hardware-software system for interactive drawing and performance were developed and tested over the course of dozens of performances.

Throughout these performance experiences, the system's complexities posed many challenges for live performance in public spaces. This system used a wide range of off-the-shelf and custom pieces of hardware, including professional photography rigging equipment, climbing and maritime fasteners, digital and analog cameras and digitizers, specialized lighting equipment, game and gestural controllers, and personal computers. The wide variety of devices and interfaces was a weak-point in the system; changes in computing platform or system updates often affected system performance in unforeseen ways; professional rigging/climbing/sailing equipment required considerable practice before mastery. Similarly, the responsiveness and latency were difficult to solve challenges. Similarly, initial versions of the software enabling video capture, processing and playback was created in Cycling '74 Max. While Max allowed for considerable flexibility and rapid development, it too posed challenges in performance (e.g. framerates), compatibility (moving from Mac, to Windows to Linux) and updatability (e.g. new Max updates often affected system performance).

MOD's design responds to these limitations by implementing all functionality onto a low-cost embedded platform, and coupling the platform with custom hardware that provides all the necessary I/O. The embedded platforms and custom hardware in MOD offer a trade-off: we gain in affordability, reliability, size and ease-of-use, while we lose in overall system flexibility and computational horsepower.

Related Works

Drawing, as an interface, has received limited attention within the TEI community. [8] and [1] describe interfaces for musical composition and performances based on drawing. [18] proposes an interface for programming that uses crayon-drawn colors, and a color-sensor based real-time system. [16] proposes a novel interface for capturing and manipulating drawings made with ink that combines the versatility of the analog interface, with the extensibility of digital image manipulation techniques. [19] describes a method for creating large-scale pictures in public spaces using directional reflections and a custom roller device. Outside the TEI community, a wide range relevant of contributions highlight the advantages of physical interfaces to digital or hybrid content creation or design processes. [9], [15] provide an overview of the theoretical framework for designing and analyzing such interfaces, as well consideration of specific use cases like video-collage, low-fidelity prototyping or user-interface design.

As an autonomous hybrid drawing interface, the MOD is inspired by the versatility of a traditional overhead projection. This omnipresent interface has seen multiple digital makeovers in devices like the digital document viewers by Epson or Elmo (e.g. Epson DC-21 or Elmo MX-1/P30HD); the most recent innovation in this area may be the HP Sprout, a hybrid interface for manipulating physical and digital worlds that combines

[1] http://maw.c-uir.org
[2] http://alimomeni.net/projection-instruments
[3] http://alimomeni.net/bey
[4] http://alimomeni.net/eclm
[5] http://alimomeni.net/gutless

a high resolution camera, a touch interface and 2D and 3D scanning. The medium of urban projection builds on interventionist [21] and situationist [11] practices that regard the urban environment as a site for activation, serendipitous interactions [17] and a canvas for visual expression [13]. This medium owes a great deal to several contemporary artists with a body of works that define this territory. Krzysztof Wodiczko (b. 1943) is a Polish/American artist renown for large-scale architectural projection works; notable in his practice is the use of custom instruments for public interactions, or what he calls the "thing-in-between" [24]. The experience of being *alien* or *strange* dominates Wodiczko's early works like the *Alien Staff (1996)* or the *Mouth Piece (1997)* [24] exemplify his recurring strategy of using tangible and embedded interactive systems as a mechanism for engaging the people in public spaces.

Jenny Holzer (b. 1950) is an iconic media artist with a wide range of large scale architectural projection works [22]. Holzer's works are particularly relevant here in two ways: first, they re-purpose features of the urban landscape that are typically at the service of capitalist messaging (i.e. road signs, buildings facades, billboards), to instead trigger the imagination and invite reflection on the state of society. Second, they often employ a graphic style consisting of block text and white-on-black. This design choice allows for a heightened visibility and legibility of projected text on non-ideal surfaces, an implied objectivity through reference to journalistic and scientific texts [6].

Graffiti Research Labs (GRL) [7] builds on the the site specificity that is at the core of Wodiczko and Holzer's work, by adding mobility into the equation. In this regard, GRL's mobile projection tricycle represents an important step in the evolution of mobile projection devices [20]. Other artists with relevant works include Rafael Lozano-Hemmer, Karolina Sobecka, Rebecca Smith, Pablo Valbuena, Chris O'Sheo, The Illuminator.

In addition to the above artistic works within the medium, numerous academic publications have identified opportunities for public engagement offered by interactive projections and media facades ([2], [12], [23], [7], [5], [3], [4]).

DESIGN PROCESS

This section describes the MOD's design requirements and iterative development process, which were derived from many software and hardware iterations informed by live performance experiences by the authors and feedback from collaborators and participants in public performance.

Design Requirements

Interactive instruments for engaging broad audiences in some creative and performative activity pose a unique set of technical and social challenges and design requirements. Our analysis of these design requirements are divided into three categories: Affordances, Mobility and Robustness, and Customizability.

[6]http://unprojects.org.au/magazine/issues/issue-4-1/jenny-holzer/
[7]http://www.graffitiresearchlab.com/blog/

Figure 3. (Left) Drawing Case: Portable wireless interface used for gathering drawings from audience members; used in *Battle of Everyouth* by Momeni and Schmid; (Right) Projection/Drawing Jib: Mobile projection and drawing interface; used in *Musée Itinerant* by Momeni and David Bithell

Affordances

In his 2003 response to Norman's seminal 1999 publication problematizing the concept of affordance [14] Hartson outlines four types of affordances for interaction design: Cognitive, physical, sensory and functional [6]. The MOD's design has been accordingly concerned with affordances because they all impact the experience for the performers, the participating audience members and the bystanders. Regarding cognitive and functional affordances, the key consideration is the threshold for cognitive overload for a performer or participating audience member when interacting with a multi-channel system. Expert performers engaged in high-end media work for a large-scale concert or Broadway show may comfortably use a video projection system with several dozen content layers. On the other hand, the impromptu, ephemeral and outdoor nature of our use-case is often satisfied with a much smaller number of layers, combined with greater variety of physical interaction and image manipulation techniques. To make for an instrument that can be "played" while "drawing", the MOD opts for physical interfaces (real pen/makers/paper/transparency as opposed to a digital pen, physical buttons and knobs as opposed to a screen) that need no explanation, and can be manipulated intuitively and in the dark.

Mobility and Robustness

The physical and sensory affordances of our design are motivated by the range of real-world situations in which performances occur, including situations with limited lighting, unfavorable weather, overcrowded spaces, no power from the grid, or the need to clear out of the performance space very quickly. Instruments for street performances and on-the-go shows impose additional design constraints concerning weight and portability, power usage, robustness and repairability. In order to meet these requirements, the MOD uses a highly modular design in which different components are each relatively low-cost and easy to replace. Special attention was paid to making the device easy to repair (i.e. components are all easily accessible, thickness-pitch-drive of all hardware is consistent across the entire design, each module is easy to vary or customize),

Customizability
MOD was developed to not only meet the needs of the author and his collaborators in a public performance practice, but also as a versatile and extensible architecture for creating other hybrid instruments. To these ends, the hardware allows for independent real-time control of multiple continuous and discrete parameters. The software allows for independent control of multiple layers of video, each with a set of transformation and visual effects. The modular hardware and software design (see Modular Design section below) allows makers to mix-and-match components and functionalities to meet the project's needs, and to design and build new compatible modules.

Precursors and Iterative Design
The MOD is an evolution of a number of precursor instruments for achieving similar goals. Previous versions of the instrument have relied on custom software (in Cycling 74 Max, OpenFrameworks, TouchDesigner, Quartz Composer and Millumin) on a powerful personal computer, coupled with digital cameras capable of high resolution low-latency capture. The first iteration of this system created in 2008 (see Figure 4) utilized a fluorescent slide-viewing light-box , an Imaging Source scientific digital camera, a laptop and custom software that allowed for recording, playback, scaling, and masking of several layers of video (see Figure 5).

Figure 4. Livedraw Hardware v1: Momeni (left) and Schmid (right) collaborating on a livedrawing performance; System consists of a light box, digital camera with C-Mount lens and custom software. Schmid draws with pens/markers on transparency paper; digital camera captures image for manipulation in software

Later iterations of the instrument reconsidered the drawing interface (see Figure 6). This version leveraged advances in LED lighting, low-cost webcams and affordable digital fabrication to create an instrument that integrated the capture device, the mounting mechanism and the back-lighting all into a single object that was light-weight, collapsible and low-cost. By stabilizing the lighting and rigging, this iteration dramatically improved the reliability of high-quality image capture from drawings.

Figure 5. Livedraw Software v1: Screenshot of software developed in Max to allow for recording and manipulation of multiple multiple layers of video (labeled 0-10). Digital camera input ("monitor" window) is masked, scaled and placed on a video layer ("output" window).

Figure 6. Livedraw 2015; System consists of a thin USB-powered LED lightbox, encased in a custom fabricated casing made of Masonite that includes an appendage for holding the camera in the ideal location, and a USB HD Webcam

Opensource Hardware and Software and Mostly In-House Fabrication

The MOD is designed with makers, hackers and students in mind. All hardware and software developments for the project are shared with an opensource license on GitHub, and readers are encouraged to view, make or vary the designs to suit their own needs. The design and fabrication process for the MOD relied on flipping back-and-forth between in-house rapid prototyping and outsourced fabrication; *In-house:* all software elements prototyped in PureData and OpenFrameworks, all sculptural elements made with 3d printers and lasercutters, all circuit boards designed as 1-layer boards with Eagle and fabricated using an OtherMill small CNC-router; *Outsourced:* 2-layer printed circuit boards.

HARDWARE DESIGN

Overview

MOD's hardware is implemented as a modular system around the Raspberry Pi 3 computer and the Teensy 3.6 microcontroller (MCU). The Pi provides all video processing functionalities (video capture, layering, processing), while the Teensy provides all physical computing capabilities (buttons, knobs, joysticks, actuator control). A small LED projector handles video output, and power is provided to the entire system from a 12000mAH rechargeable Lithium Polymer battery.

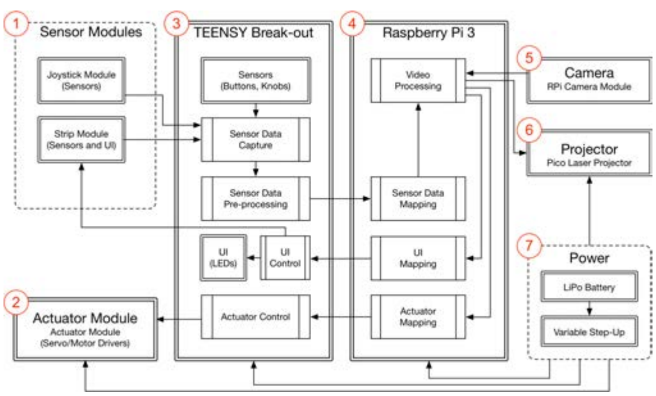

Figure 7. Hardware Overview; Double-stroke boxes labeled indicate discrete hardware components: 1) Custom PCB with sensors and UI elements, 2) Custom PCB with voltage regulation and motor drivers, 3) Custom PCB with Teensy 3.6 microcontroller, sensors and UI elements, driving circuitry for all UI elements, and connectors for secondary modules; 4) Raspberry Pi 3B, 5) Raspberry Pi Camera module, 6) Pico Projector, 7) Power management including LiPo battery and high-current voltage regulator

Physical Computing

A Teensy 3.6 Microcontroller (MCU) is used as a bridge between the software processes running on the Raspberry Pi, and physical inputs (sensors) and outputs (user interfaces and actuators) in the physical world (see Figure 10). In order to achieve these I/O requirements, the Teensy is used as a USB MIDI device slave to the Raspberry Pi that manages incoming and outgoing "Note" (discrete) and "Control Change" (continuous) MIDI Messages. Specifically, all sensor inputs are interpreted as incoming MIDI messages, while UI and actuator controls are achieved via outgoing MIDI messages. This implementation offers two notable advantage: 1) MCU programming is simplified thanks to the Teensy 's existing C/C++ classes for handling incoming and outgoing MIDI messages, 2) MIDI messages are handled with relatively low-latency/jitter by the operating system and provided to many software environments with little development overhead. This approach is also burdened by the typical disadvantages of MIDI: continuous controls are limited to the MIDI protocol's 7-bit resolution regardless of the specifications of the ADC.

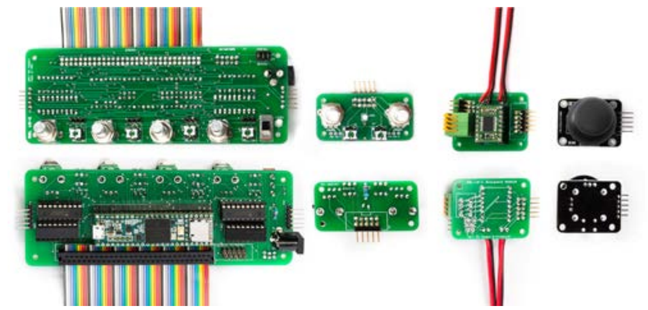

Figure 8. Modular Hardware Design: Custom printed circuit boards for the physical computing subsystem; top and bottom rows show top and bottom of circuit boards respectively. From Left to Right: A) *Teensy breakout* board; includes MCU, transistor for driving circuitry, knobs, buttons, LEDs, and ribbon cables for connecting with the *Strip* and *Actuator* modules; B) *Strip module*, includes two knob-LED-Button sets and an additional status LED; C) *Actuator module*: Includes TB6612FNG dual-motor driver and terminals; D) *Joystick modules*

Modular Design

The physical computing subsystem designed around the Teensy is implemented in several modules: a *Teensy Breakout Module*, multiple *Strip Modules* and *Joystick Modules*, and an *Actuator Module* (see Figure 8). The modular design significantly reduces fabrication costs (smaller boards are much cheaper), while allowing a wider variety of future applications where individual interface elements can be redesigned or placed differently without impacting the Teensy breakout board. Together, these modules offer many input and output possibilities (see 9). These modules can be interchangeably combined to create a unified interface for the user (see Figure 10).

Teensy Breakout Module

The main board (referred to as the *Teensy Breakout* containing the MCU itself provides the following functionalities: 1) Connect all I/O pins of the Teensy to other components and/or connectors, 2) provide high-current LED driving capability with two 8-channel source driver IC's (Allegro Systems 2982) that can deliver up to 500mA to any of the PWM outputs; this allows the system to avoid drawing too much current from the MCU itself, 3) provide ribbon-cable interconnects between the *Teensy Breakout* and three other modules that implement various sensor input, UI and actuator output capabilities. The *Teensy Breakout* provides the following:

- 4 x analog inputs (for knobs)
- 4 x 3-pin connectors for additional analog sensors
- 4 x digital inputs (for buttons)
- 4 x PWM outputs (for button LEDs)
- 6 x 10-pin ribbon connectors for *Strip Modules*

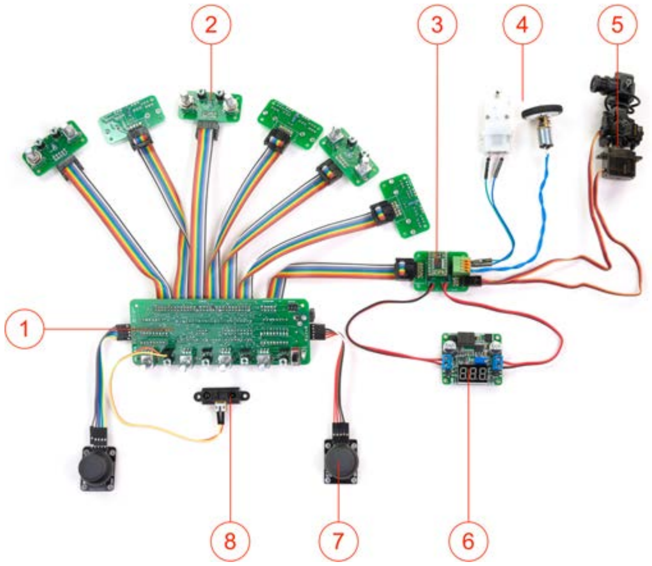

Figure 9. MOD modular design: 1) *Teensy Breakout* board, includes micro controller, buttons, knobs, LEDs, power switch, and ribbon connectors to other modules; 2) *Strip Module*, includes two knobs, two buttons, three LEDS; connects to breakout board with 10-conductor ribbon; 3) *Actuator Module*, includes dual motor driver, and connectors to/from step-up module, 10-conductor ribbon to breakout board; 4) two DC motors; 5) two hobby servos (here used in a pan-tilt configuration for a camera); 6) High-current Step-up regulator for DC actuators; 7) *Joystick modules*; 8) an IR distance sensor connected directly to the breakout board

- 1 x 10-pin ribbon connector for *Actuator Module*
- 2 x 5-pin ribbon connectors for *Joystick Modules*
- Barrel connector for 5V power from the LiPo battery
- Main power switch and indicator LED
- 2 x 8-channel source drivers

Strip Module

Based on the analogy of a "channel strip" found in audio mixers, this module is intended to be used in multiples to control "layers" or "voices" of a multi-channel system. This module connects to the *Teensy Breakout Module* with a 10-pin ribbon, and incorporates the following:

- 1 x 10-pin ribbon connector for the *Teensy Breakout*
- 2 x analog inputs (for knobs or continuous analog sensors)
- 2 x digital inputs (for buttons or discrete sensors)
- 2 x pwm outputs (for button LEDs)
- 1 x digital output (for status LED)

Actuator Module

Our performance experiences showed that props, including simple robotic gadgets combine well with drawings to create interesting imagery. This feature allows the user to generate and control continuous visual motion without the need to continually interact with objects under the camera. This instrument allows for independent bi-direction and variable-speed of two DC actuators (motors, solenoids) and two hobby-servos via MIDI note and control-change messages. The *Actuator Module* PCB is built around the TB6612FNG motor driver IC and controls are provided from the microcontroller using a 10-pin IDC connector. All actuators are powered by a variable high-current step-up regulator (see Figure 11) in order to allow control of actuators that require more than the 250mA the Teensy can provide. The actuator module consists of the following

- 1 x 10-pin ribbon connector for the *Teensy Breakout*
- 1 x TB6612FNG dual DC motor driver
- 1 x logic-inverter IC (to save an MCU pin)
- 2 x 3-pin connectors for attaching hobby servos
- Spring-loaded quick connectors for attaching actuators

Joystick Module

This module is built around the common and low-cost thumb-joystick found in PlayStation controllers. The module connects with the *Teensy Breakout* with a 5-pin ribbon and provides two continuous controls (horizontal and vertical joystick movement) and one digital input (joystick button).

Video Input and Output

In order to optimize performance, a Raspberry Pi Camera Module V2 is used. [8]. This modules offers a Sony IMX219 8-megapixel sensor capable of 1080p30 and decent low-light performance, and a high-speed CSI interface for connecting to the Pi with a ribbon cable.

A Sony MP-CL1A pico projector with an LED light-source is used as video output. While this projector is far less bright than a full-sized home- or pro-theater projector, it offers a number of invaluable features that make it well suited to our use-case: very small and lightweight (77.0 x 149.5 x 13.0 mm , 210 g), battery powered (2 hours at 5V/1.5A; rechargeable), short throw (range 1.1 - 3.4 m) and infinite focus (as with all laser projectors). MOD also includes an optional HDMI-splitter that allows the Pi's HDMI output to be routed to a larger projector if needed.

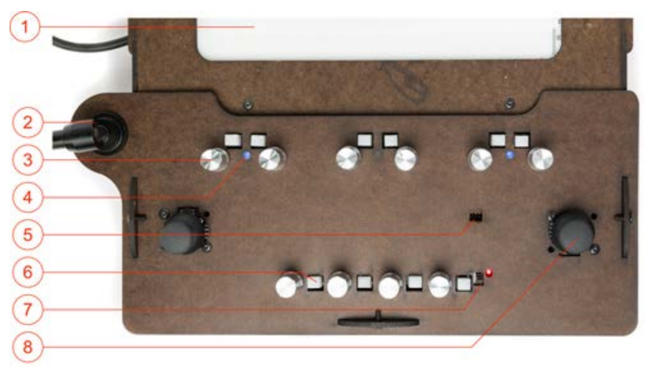

Figure 10. MOD UI Panel: 1) Back-lit drawing area; 2) mounting hole for projector; 3) knobs; 4) LED; 5) servo connectors; 6) Illuminated buttons; 7) Power button; 8) Thumb joystick

Power Management

The MOD is powered from an off-the-shelf 5V 12000mAh rechargeable LiPo battery.

[8] https://www.raspberrypi.org/products/camera-module-v2/

Table 1. Hardware modules for MOD Physical Computing Modules

Module	*Components*	#	Function
Teensy Breakout	Teensy 3.6	1	IO
	A2982	2	Drive LEDs/actuators
	Knob	4	User Input
	ButtonLED	4	User Input/Interface
	Strip Connector	6	Connect to strips
	Actuator Connector	1	Drive actuators
	Servo Outputs	2	Control Servos
	Switch	1	Power Switch
	Power	1	Power from battery
Strip Module	Knob	2	User Input
	Button	2	User Input
	LED		User Interface
	LED	1	User Interface
Actuator Module	TB6612FNG Motor Driver	1	Actuator Control
	Variable Step-up	1	Convert 5V to 5-30V
	Servo Connectors	2	Drive two servos
	Motor Connectors	1	Drive two motors
Joystick Module	Playstation ThumbJoystick	1	User Input (continuous controls)
	Button	1	User Input (switch)

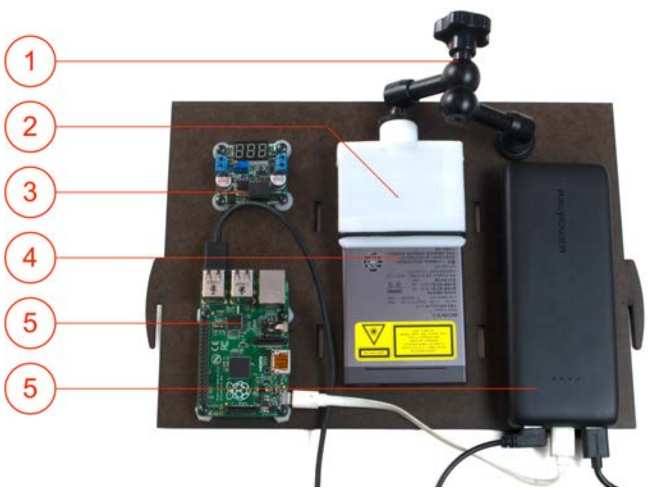

Figure 11. Hardware inside the MOD: 1) small "magic arm", 2) custom printed projector mount, 3) step-up voltage regulator for high-voltage actuators, 4) Pico laser projector, 5) Raspberry Pi 3, 6) LiPo Battery

Enclosure

The body of the instrument is fabricated from Masonite that is cut to form using a laser cutter (see Figure 12. Total material cost for the shell of the instrument is under $5.

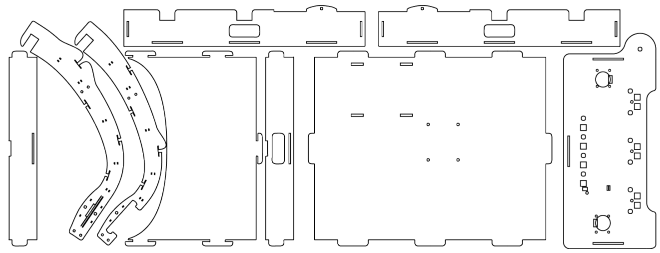

Figure 12. CAD designs for the MOD: all components are lasercut from low-cost 1/8" sheet material; minimal hardware is required for assembly

SOFTWARE DESIGN

Overview

Software for this system is developed using three environments:

1. Autodesk Eagle: Primarily used to design PCBs; also used to define the relationship between MCU hardware pins and MCU software variables; as well as the relationship between physical input devices and UI elements, and their corresponding MIDI note and control-change numbers. This feature is described in the section titled "Linking Hardware and Software Design".

2. Arduino IDE: Used to program the Teensy MCU; requires the *Teensyduino* add-on that allows programming Teensy MCUs, and provides USB-MIDI device functionality

3. OpenFrameworks: A C++ framework used to program all video processing

4. GLSL: OpenGL shader language, used to optimize video processing with GPU-based hardware acceleration

Figure 13 shows the software sub-processes and the hardware platforms that manage them.

Video Processing

The video processing functionality of MOD is based on *Livedraw* [9], a visual performance software developed by the author over the last decade. *Livedraw* offers the following core capabilities:

1. Managing multiple layers: This includes independent control of scale and position for layers of superimposed video
2. Keying and Thresholding: GPU-based image processing that renders parts of the image transparent so as to allow overlaying multiple video layers on top of one another
3. Live Looping: CPU- or GPU-based recording and looping of video frames
4. Controller mappings and UI: Triggers and parameters for the system are mapped to physical controllers and UI elements in order to allow the user to interact with the system gesturally

[9] http://maw.c-uir.org/software/

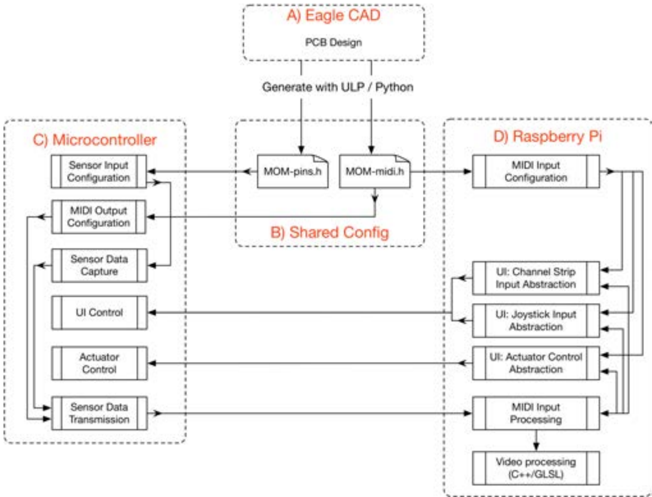

Figure 13. Software Overview; dashed-boxes indicated independent programming environments: A) Autodesk Eagle CAD was used to design all custom circuit boards; B) two header files are generated by a python script post processor a text file that describes signal connections in the "Teensy Breakout" board; C) Microcontroller code (in C on a Teensy 3.6) uses the two header files to configure sensor capture and MIDI-output configuration; D) The video processing engine (implemented in C++ with OpenFrameworks on the RPi) uses the MIDI header file to configure all MIDI I/O from system

The openFrameworks implementation of this graphics engine is built around a custom class (`vidLayer.cpp` and `vidlayer.h`), which is instantiated once for the live video layer and once for each looping video layer. This class optimizes performance by making use of the openGL `framebuffer` for all recording and processing. This class also defines the initialization and rendering behavior of each video layer, applies the openGL shader interactions. The shaders (`livedraw.frag` and `livedraw.vert`) are implemented in GLSL ES2 for compatibility with embedded platforms, and manage thresholding, keying, inverting and alpha-transparencies for all video layers.

Development Environment

Creative work with embedded systems is a mixed blessing. The ability to design, customize and build one's own hardware is rewarding. The ability to create robust and reliable turn-key systems that are unaffected by regular software updates, virus infections, or interruptions from email or the like also rewarding. Ideally, such systems lead to more fulfilling creative experiences as the user's time is mostly spent working with the tools and the content, as opposed to developing and repairing them. That said, creating complex instruments with embedded systems is considerably more difficult and error-prone than a straight-forward software development task for personal computers. For this reason, several design considerations and development tools were utilized to make the overall workflow more stream-line and less prone to human error. These areas are briefly outlined below

Networking and Configuration Management

The Raspbian distribution of Linux comes with a range of Debian tools that facilitate development on the Pi, and/or the ability to quickly install additional packages using `apt-get`. The following packages were critical to building an efficient development environment:

- samba: debian package that enables easy networking between personal computers and the Pi
- ansible: debian package that enables configuration-management, application deployment, cloud provisioning, ad-hoc task-execution, and multinode orchestration

Linking Hardware and Software Design

From the user's perspective, each tangible interaction element in MOD is represented by a MIDI note- or control-change-input or output message. This association, however, must be persistently maintained in several disparate development environments: 1) The PCB Schematic and Board files that define the hardware modules, 2) the MCU code running on the Teensy that samples sensors and sends out MIDI messages, or receives MIDI messages to control lights and actuators, 3) the graphics code running on the Raspberry Pi that receives incoming MIDI messages from the MCU and maps them to various processing parameters. In order to maintain consistency across these environments and reduce chances of human error, a series of python text-processing scripts were created that automatically generate header files for the Arduino code and an XML file for the openFrameworks application with all the necessary I/O information. Specifically, we employ the `pinlist.ulp` script that ships with Eagle CAD to generate a text file that describes which signals are connected to which pins of the MCU. We then process this text file with a custom python script that uses regular expressions to create a well-formatted C header file for the Arduino program with all the necessary signal-to-MIDI-message mappings. At the same time, the script also generates a well-formatted XML file that is used to initialize the settings of the openFrameworks application; calibrating it to correlate the incoming MIDI information with its proper functionality control.

EVALUATION

The technical functions of this instrument were evaluated in a laboratory setting for hardware and software performance. According to our tests, the physical computing hardware is easily capable of handling all 58 channels of analog and digital input and output that enable buttons, knobs, additional analog sensors, and actuator control. Timing performance is comparable to off-the-shelf MIDI-devices. Software performance is dramatically reduced as compared with a personal computer–as expected. With a performance requirement of at least 15fps for rendering, the current iteration of the video processing software can accommodate 2 layers of video at 1080p, 3-4 layers at 720p and 5-10 layers at 640x480.

The creative potential of this instrument was evaluated by four performing artists with an active live cinema practice. The first two artists, who often perform as a pair and integrate traditional drawings, sand drawings, and puppetry into their work pointed to several strengths and weaknesses of this instrument. They identified the most desirable feature of this instrument to be its reliance on analog drawing (as opposed to a digitizing tablet), its portability and its autonomy (no need for a laptop). The ability to to combine analog drawing with digital effects, layering and transformation was deemed highly desirable, although the performers saw many occasions where some of these features would not be necessary. More specifically, the artist identified a distinct advantage in using ink and markers for mark making, as it allows them to build on their existing drawing skills, while allowing them to improvise with mark making in ways that are not possible with digital drawing interfaces. This group also identified opportunities to use this interface as a more generalized digital viewer, akin to a a traditional overhead projector. Building on this function, they fashioned a small container for sand and water that allowed them to use the instrument to capture, layer and project sand drawings as well as ink-and-water visual compositions. The third artists had similar positive feedback, but was more cognoscente of the computational limitations of this system as compared with the laptop-based live cinema system with which she typically performs. Specifically, limitations in the number of layers and the length of recorded video loops that are imposed by the memory limitations of our small embedded computer. All artists agreed on a user-interface weaknesses in the system: specifically, while the instrument portends to function as a one-person device, simultaneous drawing, looping and manipulation requires more hands than a single person has available. All artists also agreed that while the pico-projector that is included in the system may be effective for rehearsal or prototyping, real-world performance situations impose requirements (distance, brightness, etc) that will rarely be met by our small projector.

CONCLUSION AND FUTURE WORK

With the development of this mobile and easy-to-use hybrid drawing instrument, we expect this type of production to continue and expand in the coming year. We are presently in the process of fabricating multiple MODs for field experimentation: One unit will travel to the celebrated Bread and Puppet festival with artist Davey Steinman who has a practice of media-rich puppetry. A set of MODs will be used in an upcoming research project in collaboration with the Children's Museum of Pittsburgh to study the possibilities of integrating drawing into the workflow of a maker space situated in a museum. We are actively seeking new collaborations and opportunities for experimenting with new use-cases of this instrument.

Our design allows for a range of future applications that we have yet to explore. In particular, the actuation possibilities of the MOD in combination with live video processing present a promising area of exploration in creating live animations. Moreover, since the *Teeny Breakout board* is capable of driving up to 16 actuators (in addition to the dedicated *Actuator Module*), we expect a range of new module designs that focus on miniature robotics.

We envision the MOD as part of a family of portable performance and content creation instruments that leverage embedded computing, gestural interaction, and intuitive interfaces. A similar instrument for music-making (named MOM for Mobile Object for Music[10]) is currently in its fourth iteration and will be further iterated to leverage the developments from the MOD.

[10]http://alimomeni.net/MOM

ACKNOWLEDGMENTS

This work is dedicated to all members, collaborators, and fans of MAW–Minneapolis Art On Wheels. Special thanks to Drew Anderson whose early work as a research assistant on mobile project instruments helped set this project in motion. Special thanks also to Jenny Schmid, Andrew Steudel, Sinan Goknur and Davey Steinman whose on-the-road work with urban projection within and beyond the context of MAW was invaluable and inspiring.

REFERENCES

1. Chiang Chenwei, Shu-Chuan Chiu, Anak Agung Gede Dharma, and Kiyoshi Tomimatsu. 2012. Birds on Paper: An Alternative Interface to Compose Music by Utilizing Sketch Drawing and Mobile Device. In *Proceedings of the Sixth International Conference on Tangible, Embedded and Embodied Interaction*. 1–4.
2. Peter Dalsgaard and Kim Halskov. 2010. Designing urban media façades: cases and challenges. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM Press, New York, New York, USA, 2277.
3. Claude Fortin, Steve DiPaola, Kate Hennessy, and Jim Bizzocchi. 2013. Medium-specific properties of urban screens: towards an ontological framework for digital public displays. In *Proceedings of the 9th ACM Conference on Creativity Cognition*.
4. C Fortin and K Hennessy. 2015. The dual skins of a media façade: Explicit and implicit interactions. *ACM SIGGRAPH Art Papers* (2015).
5. Claude Fortin, Kim Hennessy, and Hughes Sweeney. 2014. Roles of an interactive media façade in a digital agora. In *Proceedings of The International Symposium on Pervasive Displays*.
6. Rex Hartson. 2003. Cognitive, physical, sensory, and functional affordances in interaction design. *Behaviour & Information Technology* 22, 5 (Sept. 2003), 315–338.
7. Luke Hespanhol, Martin Tomitsch, and Oliver Bown. 2014. Using embodied audio-visual interaction to promote social encounters around large media façades. In *Proceedings of the 2014 conference on Designing interactive systems*.
8. Kazuhiro Jo. 2008. DrawSound: A Drawing Instrument for Sound Performance. In *Proceedings of the nd international conference on Tangible and embedded interaction*. 1–4.
9. Adriaan Ianus Keller. 2005. For Inspiration Only; Designer interaction with informal collections of visual material. (2005).
10. Michael Lew. 2004. Live Cinema: designing an instrument for cinema editing as a live performance. In *Proceedings of the 2004 conference on New interfaces for musical expression*.
11. Tom McDonough. 2004. *Guy Debord and the Situationist International: Texts and Documents (October Books)*. The MIT Press.

12. David Molyneaux and Hans Gellersen. 2009. Projected interfaces: enabling serendipitous interaction with smart tangible objects. *TEI '09: Proceedings of the 3rd International Conference on Tangible and Embedded Interaction* (Feb. 2009).
13. Ali Momeni and Stephanie Sherman. 2015. A Manual for Urban Projection.
14. Don Norman. 2004. Affordances and Design; Affordance, Conventions and Design (Part 2)–Two Essays on Design.
15. Daniel Saakes. 2005. Material light: exploring expressive materials. *Personal and Ubiquitous Computing* 10, 2-3 (Oct. 2005), 144–147.
16. Kayato Sekiya and Shinpei Chihara. 2011. iNkDraw: Physical Ink-Based Interface for Capturing and Manipulating Drawings on Digital Display. In *Proceedings of the fifth international conference on Tangible, embedded, and embodied interaction*. 1–4.
17. Mark Shepard. 2009. Sentient City Survival Kit: Archaeology of the Near Future. *Digital Arts and Culture 2009* (Dec. 2009).
18. Jay Silver and Eric Rosenbaum. 2010. Twinke: Programming with Color. In *Proceedings of the fourth international conference on Tangible, embedded, and embodied interaction*. 1–2.
19. Yuta Sugiura, Koki Toda, Takashi Kikuchi, Takayuki Hoshi, Youichi Kamiyama, Takeo Igarashi, and masahiko Inami. 2017. Grassffiti. In *the Tenth International Conference*. ACM Press, New York, New York, USA, 413–417.
20. Abigail Susik. 2012. Sky Projectors, Portapaks, and Projection Bombing: The Rise of a Portable Projection Medium. *Journal of Film and Video* 64, 1 (2012), 79–92.
21. Nato Thompson, Gregory Sholette, Joseph Thompson, Arjen Noordeman, and Nicholas Mirzoeff. 2004. *The interventionists*. The MIT Press.
22. Diane Waldman. 1989. *Jenny Holzer*. Solomon R Guggenheim Museum.
23. Alexander Wiethoff, Thomas Bauer, and Sven Gehring. 2014. Investigating multi-user interactions on interactive media façades. *Proceedings of the 2nd Media Architecture Biennale Conference* (2014).
24. Krzysztof Wodiczko. 1999. *Critical Vehicles*. The MIT Press.