Pineal: Bringing Passive Objects to Life with Embedded Mobile Devices

David Ledo[1,2], Fraser Anderson[1], Ryan Schmidt[1],
Lora Oehlberg[2], Saul Greenberg[2], Tovi Grossman[1]

User Interface Research Group
Autodesk Research
{firstname.lastname}@autodesk.com

Department of Computer Science
University of Calgary
{firstname.lastname}@ucalgary.ca

Figure 1. Sample interactive objects created with Pineal. (A) Magic 8-ball, (B) Level, (C) Firetruck, (D) Twitter Display Planter, (E) Voice-Activated Light

ABSTRACT

Interactive, smart objects – customized to individuals and uses – are central to many movements, such as tangibles, the internet of things (IoT), and ubiquitous computing. Yet, rapid prototyping both the form and function of these custom objects can be problematic, particularly for those with limited electronics or programming experience. Designers often need to embed custom circuitry; program its workings; and create a form factor that not only reflects the desired user experience but can also house the required circuitry and electronics. To mitigate this, we created *Pineal*, a design tool that lets end-users: (1) modify 3D models to include a smart watch or phone as its heart; (2) specify high-level interactive behaviours through visual programming; and (3) have the phone or watch act out such behaviours as the objects' "smarts". Furthermore, a series of prototypes show how Pineal exploits mobile sensing and output, and automatically generates 3D printed form-factors for rich, interactive, objects.

Author Keywords

Fabrication; 3d printing; rapid prototyping; smart objects

ACM Classification Keywords

H.5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous;

INTRODUCTION

Many new objects, environments and devices now include digital capabilities with some form of sensing, connectivity or interaction capabilities. From consumer electronics and appliances to toys for children, these objects range in a variety of form factors and functionality. Additionally, these objects are becoming 'smarter' in that they can connect to the cloud, respond to richer interactions, and utilize sensors to better understand their environment. Still, prototyping these interactive devices is not a trivial process [4], as it requires understanding of software and hardware components: identifying and acquiring appropriate electronics, assembling and programming them, as well as creating 3D form-factors that will suit the task at hand. There have been several approaches within HCI research exploring novel methods to fabricating interactive devices [18, 21, 22, 24], but these typically require specialized components, programming or electronic assembly.

One possible solution is to utilize existing commodity mobile devices, such as smartphones and smartwatches, in combination with 3D fabrication, to rapidly prototype the form and functionality of smart objects. These high-computational devices contain a myriad of sensors capable of rich interactions, from their high-resolution touch displays to motion and environmental sensors. They also possess an integrated and long lasting power source as well as wireless connectivity. Whereas current prototyping approaches rely on a single sensor, mobile devices possess the necessary capabilities required by a significant proportion of today's smart objects all contained within one core [20].

Early examples of repurposing smartphones in such a way have already emerged. For example, Google cardboard makes a mobile device suitable to act as a head mounted Virtual Reality display, and the Smartboy [36] provides tangible buttons to transform a phone into a functioning Gameboy form factor. Work in Human–Robot Interaction (e.g. [10, 28]) is increasingly using mobile devices as the

CHI 2017, May 06-11, 2017, Denver, CO, USA
 ISBN 978-1-4503-4655-9/17/05…$15.00
DOI: http://dx.doi.org/10.1145/3025453.3025652

robot's face. In each of these cases, the smartphone provides the intelligence and sensing abilities (*soul*) of the device, while the physical objects (*body*) provide affordances, aesthetics and other functions. Though specific commercial examples of this concept exist, we are unaware of any explorations into the design space of these soul-body smart objects and the design tools which could enable this practice.

Our goal is to enable interaction designers to create rich interactive, self-contained, smart objects (Figure 1), using commercially available mobile devices and 3D printers. Specifically, this paper presents the following contributions:

1. Leveraging commonly available mobile devices (phones and watches) to rapidly prototype smart interactive objects without the need for custom electronics.
2. A design space that identifies common sensory and output modalities in current mobile devices, as well as modifiers for these capabilities to change their function.
3. Pineal, a system that enables interaction designers to prototype smart objects with little knowledge of circuitry, software development or 3D modelling. It encourages integrated designs that can explore look and feel, as well as implementation and role [11]. Based on user-defined interactive behaviours, the system automatically modifies an imported 3D model to fit the mobile device inside, and expose the necessary input and outputs.

The system is able to create a wide range of prototypes to showcase a range of supported functionality. While more work is needed to make the visual programming language fully featured, the current implementation is able to demonstrate a wide variety of different sensors and form factors across the described design space.

RELATED WORK

This work builds upon prior work in extending the capabilities of mobile devices, sensing techniques for prototyping, and automating CAD modelling processes.

Physical Extensions to Mobile Phones

Previous work has explored adapting the concepts of tangible interaction into mobile form factors. While primarily focusing on sensing mechanisms, this body of work discusses the addition of tangible tokens atop the interactive surface. For instance, SLAP Widgets [33] and Clip-On Gadgets [35] create physical controls for mobile interfaces. Capricate [26] integrates capacitive sensors to re-route touch from a printed, 3D object to a mobile device touch screen. Acoustruments [13] connects a smartphone's built-in microphone and speakers through a 3D printed tube. This tube acts as a passive acoustic transducer to other input methods (e.g., tilt sensors, sliders, etc.). This allows the mobile device to sense interactions with the tangible components without added electronics. These technologies add tangibility to hold meaning of the effect of actions, provide physical affordances, integrate physical tools or add reinforcement for the current interaction modes. Pineal extends and integrates some of these works. It simplifies the workflow to (1) create the extensions to the mobile device, (2) fitting a form-factor, and (3) reducing the barrier to entry for prototyping intelligent objects.

Sensing Techniques for Prototyping Interactive Devices

Several research projects have investigated various sensing techniques to enable rapid prototyping of interactive objects. For instance, Lamello [23] allows interaction designers to 3D print tangible interface components (e.g. slider potentiometers) with passive acoustic signatures that can be sensed with a microphone. Similarly, with Sauron [21], interaction designers import a form, and select where to place physical interface components. Using computer vision, Sauron is able to detect the movement of input devices and map those inputs to interactive behaviours.

Other approaches have examined how to enable and extend touch-input with prototypes. As an end-to-end prototyping tool, Midas offers interaction designers ways to prototype capacitive touch sensors via attaching adhesive-backed copper foil to physical objects [25] to reroute and relocate touch sensors. Touché [19] and Touch & Activate [16] enable touch interaction with objects by sending signals through the objects and recording how touches modify those signals. Touché uses electrical signals, whereas Touch & Activate uses ultrasonic signals.

While these works extend the possibilities of prototyping interactive devices, many of them require programming knowledge, specialized hardware, or a background in electronics. In contrast, Pineal automates many of the difficulties inherent in this prototyping process while integrating multiple sensors.

Automating 3D CAD Modelling

Creating 3D models from scratch is difficult, especially when those models have to match existing, real-world geometries. To simplify this process, many research projects have provided automated tools to address specific domains or problems.

A number of projects have examined integrating electronic components with 3D models. For instance, RetroFab [18] automatically generates enclosures for electronic components which are used to actuate existing physical interfaces. PipeDream [24] allows users to easily author internal pipe structures within 3D printed objects, which can be applied to a range of input (capacitive sensing) and output (haptic feedback, light pipes [37]) techniques but relies on the author to define the location and purpose of these pipes. MakersMarks [22] allows users to physically sculpt and create an object and use tags in order to specify physical input mechanisms. When the object is 3D scanned, the model combines the tagged markers to create a solid model that reflects both the overall form as well as function. Enclosed [31] allows designers to create enclosures for electronic objects. While it allows for custom shape creation, its focus is on incorporating the shape of the electronic components, not on working with existing 3D models.

Several approaches have examined how to print and model around existing geometries. One approach examines how to continue building on a previous print [29], by scanning the object after it has been mounted inside of a 3D printer and determining planes that can be extended with additional material. In a similar vein, Encore [5] allows users to print and model around existing objects which are not 3D printed. Lastly, Reprise [6] enables end-users to customize accessibility-focused adaptations to existing objects by allowing users to specify simple, high-level specifications.

MixFab [32], CopyCAD [7], and KidCAD [8] all allow novices to begin to perform 3D modelling operations using real-world objects. The objects are scanned in via cameras or sensors and brought into a 3D modelling environment which simplifies the modelling process. In many of these approaches, the modelling techniques are simple enough for novice end-users to use.

Pineal builds on these prior works by automating the 3D modelling tasks inherent in embedding devices into 3D objects. Pineal leverages prior algorithms [24], while introducing new functionality such as automatic splitting. This removes the need for users to understand 3D modelling and simplifies the creation of smart objects.

DESIGN SPACE OF SOUL–BODY PROTOTYPING

Currently, mobile devices have a wide range of sensors and outputs. Given the broad range of applications, they can be thought of as multi-tasking devices, similar to a swiss army knife. We envision that by creating a new form factor around them, the mobile device could become a new, single-task specialized device. This inspired the notion of soul-body prototyping, with the name Pineal referring to Descartes' belief that the soul and body meet in the pineal gland.

The core concept of soul-body prototyping is to embed a mobile device (*soul*) into a custom fabricated target object (*body*) to prototype the form and function of a smart interactive object. Current mobile devices contain a wide assortment of input modalities and abilities for output which they can pass on to the target object. Additionally, these abilities can be augmented with simple hardware modifications to transform the input and output, leading to new interaction possibilities. Thus, the passive enclosure can dictate the meaning and functionality of the mobile device. In this section, we present a design space of the capabilities enabled by this soul-body prototyping paradigm.

Mobile Input

Mobile devices are rich in input capabilities, enabling users to interact via touch, voice, movement and vision – properties which can be passed onto the target objects.

Capacitive: Modern smartwatches and phones are equipped with arrays of capacitive sensors overlaid on their displays. If mobile devices are embedded within the body such that their displays are mounted on the surface, then the objects can be capable of sensing touch.

Acoustic: Through their integrated microphones, mobile devices can be used to add acoustic input, such as audio communication, to the target objects. Using speech and audio recognition, context sensing and voice commands could also be supported.

Motion: Accelerometers, gyroscopes and magnetometers can be used to determine the orientation at which the object is being held, or provide information regarding the context of use. Motion data can also be used to sense more explicit input, such as gestures [27].

Vision: Cameras are now available on almost all smartphones and some smart watches. In addition to supporting photo and video capture, these sensors can be used to give its target object the ability to scan barcodes [15] and perform context or activity recognition [2]. For such capabilities to be passed on to the target object, the camera sensor must be on the surface of the object or have a hollow channel to its exterior.

Additional Sensors: In addition to these common input modalities, emerging mobile devices are including many other sensors such as GPS, NFC, temperature, IR, and force. These new sensors could enable many new capabilities for prototyping responsive, interactive objects.

Mobile Output

Current mobile devices are also equipped with a variety of output modalities. Designers can leverage them to prototype devices that can convey rich information to the user.

Visual: Most devices are equipped with a high-resolution, full-colour display, and are able to render text, images and video. Mounting a mobile device on the surface of the target object would enable these visual output capabilities.

Audio: Many mobile devices contain one or more speakers with the ability to output both human-perceivable audio, as well as ultrasonic frequencies. This can be used to prototype objects that provide audio feedback for interactions, notifications, as well as speech-based audio.

Vibration: Small, vibrating motors are embedded into most commercial mobile devices to allow them to provide haptic feedback to the user. Depending on how the mobile device is embedded into the target object, the tactile information can be passed on the entire target object, or localized areas. This feedback can be used to enable discreet notifications, or direct feedback to user operations.

Power and Connectivity

Connectivity: Currently, most mobile devices have a variety of wireless capabilities, allowing them to connect to the internet and other devices via Wi-Fi and Bluetooth. With this connectivity, devices are able to retrieve information about current events, the environment, and a wide range of other information that the user or system can act on. When embedded on a target object, the object inherently becomes wireless and web-enabled.

Power: A core requirement of the vast majority of interactive devices is power. By leveraging a mobile device as the soul, its own power supply can be used to enable the above-described input and output capabilities.

Input and Output Modification

In addition to using the sensors of a mobile device for their intended purpose, one can add *modifiers* to expand the capabilities of the mobile device. Modifications can take the form of *rerouting* modifiers, or *transducing* modifiers.

Rerouting Modifiers

Rerouting modifiers preserve the nature of the sensing or actuation, but change the location where it takes place. Rerouting can be particularly useful if input and output is desired on different locations of the target object, which extend beyond the physical size of the mobile device. For instance, light pipes can be used to reroute light output to different physical locations [3].

Input can be rerouted in a number of ways, for instance, capacitive touch sensing can be moved to off-screen buttons using conductive pathways [26, 35]. This allows for the touch sensor to be placed in a more convenient or appropriate location. Similarly, light pipes can be placed over the camera lens to allow light to be sensed in several remote locations which may otherwise be out of the view of the camera.

Transducing Modifiers

Transducing modifiers are those which allow for different types of interaction that extend beyond the base capabilities of the mobile device. This is often done to enable new types of interaction, or to allow for different sensing techniques. For instance, Acoustruments [13] uses the microphone and speakers to sense how tangible controls are being manipulated therefore adding novel input mechanisms to a mobile device. Microphones can also enable gestural or tangible interactions with devices, as in SoundWave, Lamello and Acoustruments [9, 13, 23]. The speakers can also be used for interaction and sensing by emitting audio chirps which can be used to measure distance [14], or ultrasonic patterns which can be affected by the environment and used to sense explicit user input [13].

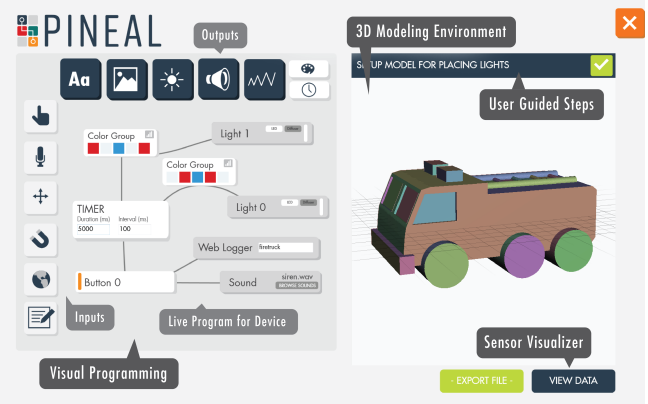

Figure 2. Pineal Interface – the left side is a visual programing panel that allows designers to prototype and test behaviours. The right shows a 3D modeling environment where the user imports their model, which the tool dynamically modifies to embed the mobile device.

Beyond audio, the magnetometer can be used to sense the movement of a magnetic field as input from buttons, such as with Nenya [1] and Google Cardboard [38]. Additionally, the Wi-Fi and Bluetooth signals that are received by mobile devices can be used for indoor localization rather than explicit data communication. This can provide applications with more information regarding the context of use.

To transduce output, the rhythmic vibration patterns can be used to achieve translational or rotational motor movement, causing a device to move in the environment [39]. This can be used to give an object the ability to move throughout a space, such as rotating around an environment [17, 34].

PINEAL

Pineal is a tool that allows hobbyists, makers, and interaction designers to rapidly prototype interactive smart devices. These smart devices are the result of programming the inputs and outputs from a mobile *device* (the "soul"), which is then embedded into a static 3D printed *target object* (the "body"). The system contains a visual programming authoring interface (Figure 2, left) and a 3D modeling workspace (Figure 2, right). After authoring the behaviours, the system guides the users through simplified modelling tasks within the workspace, enabling end-users to quickly prototype and test different behaviours in real-time without the need for advanced 3D modelling, programming or electronic assembly skills.

Sample Walkthrough: Creating a Toy Firetruck

To illustrate how a designer would use Pineal to create an interactive device we provide a walkthrough of the various steps a designer takes to go from ideation to functional prototype. In this example, the designer would like to create an interactive firetruck toy where the lights illuminate in alternating red-and-white colours and a siren sound plays when a button on the truck model is pressed. Authoring the behaviour is done on a desktop computer and the logic is streamed to the phone via wireless connection.

1. Importing the Form Factor

Prior to interacting with Pineal, the designer acquires a 3D model of a firetruck they wish to make interactive and imports it into Pineal. This model can be downloaded from common online repositories such as Thingiverse [30] or TurboSquid [40], or can be created using other tools.

2. Authoring the Behaviour

After importing the model, the designer begins to create the behaviour of the device using the visual programming interface. The designer first selects the 'button press' input from the palette and drags it onto the canvas. Then, they drag the 'play sound' output from the toolbar onto the canvas and select a siren-sound from Pineal's existing library. Similarly, the designer drags two 'lights', along with corresponding

'colour' modules onto the canvas. The designer then specifies that the lights are white and red, to emulate the light patterns of a firetruck. Lastly, they connect the button press to the siren and lights by dragging links between them. As this is done, the system streams the code to the mobile phone to process the inputs and generate the required outputs.

3. Guiding the Modelling Process

Once the user has defined a behavioural description of the object, Pineal guides them through the process of placing the physical components within the virtual model. The order of modeling steps is pre-determined, using a fixed rule-set. For instance, the phone is always placed before the lights as the location of the phone must be known to properly route the endpoints of the light tubes to the phone. Given that the firetruck does not require a display, the system automatically places the model of the phone in the centre of the firetruck.

Pineal then instructs the user to specify the location of the two lights, which the designer does by directly clicking on the surface of the 3D model within the modelling workspace. Following this, Pineal automatically creates a curved path from the surface of the model, through the object, terminating at the screen of the mobile device. This path provides a channel where the designer will insert light pipes that will direct light from the screen to the desired location.

Next, the user adds the button by selecting the desired region on the surface of the model. This selection defines the shape and location of the button, which will be printed out of conductive material. An additional, optional *spring* model is automatically generated, which can be attached to the bottom of the button to provide a more realistic button feel if desired.

Finally, since the user will be using audio output, Pineal creates an array of 'speaker holes' through the object to better allow sound to travel out of the device.

4. Object Generation

Once complete, Pineal automatically generates the following objects that the designer can export (Figure 3): (1) the top of the fire truck (with alignment pins, a button cavity, and two hollowed channels for the lights); (2) the bottom of the fire truck (with a phone cavity, holes for the speakers, and alignment pin holes); (3) a button; and (4) a spring. These models can be exported as standard STL files and sent to a consumer-grade 3D printer.

5. Fabrication and Assembly

The top and bottom of the firetruck can be printed in any material, while the button requires conductive material (e.g., conductive PLA). The optional *spring* is printed in an elastic material (e.g., NinjaFlex).

After printing, the designer can begin assembly. Assembly is relatively quick and simple, with this example taking approximately 5 minutes to assemble. The phone is placed in the cavity, and the two halves of the model snap together. The alignment pins, as well as implicit affordances (e.g., overall shape) help guide the assembly. After assembly, the

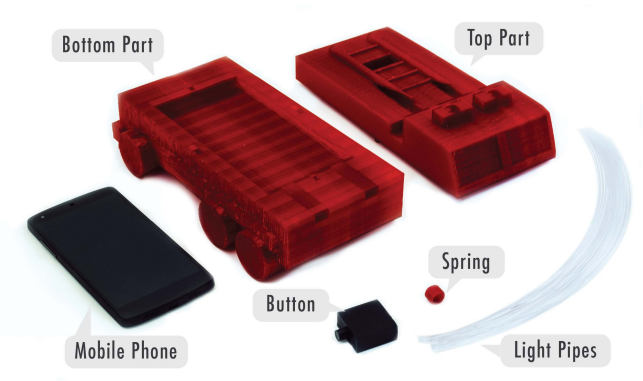

Figure 3. The components of the complete firetruck model.

light pipes can be inserted into the channels carved out for them, and the button can be placed into the corresponding hole in the back of the truck. The firetruck is now ready for use, and when the button is pressed, it will play a siren sound and flash its lights.

IMPLEMENTATION

Pineal is comprised of a visual programming language, allowing designers to author high-level behaviours using a drag and drop interface. An interactive 3D modelling canvas updates as the user modifies these high-level behaviours. When changes in the visual programming language require modifications to the model, the interface displays the required actions above the modelling environment to guide the user through the process.

System Overview

The Pineal system (Figure 4) includes a smartwatch, smartphone and desktop PC, each running custom software. Currently, only the Google Nexus 5 phone and a Samsung Gear Live smartwatch are supported and both run a custom application implemented in Java for Android (SDK 23). The smartphone application connects to a NodeJS server on the desktop PC, which in turn connects to a C# (WPF) application where the visual programming is done. This C#

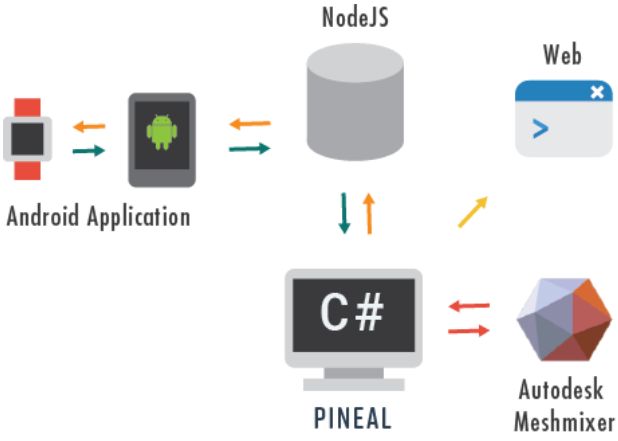

Figure 4. System architecture of Pineal, showing connections between smart phone, watch, Node JS server, C# client and Meshmixer.

application connects to an instance of Autodesk Meshmixer, using the Meshmixer API. This API allows Pineal to perform the automated modelling steps.

The application running on the mobile devices streams all sensor data to the C# application, where the data is processed. If any action is to be taken (e.g., play a sound, display an image), the C# application sends a message back to the mobile device indicating what action is to be taken.

This constant streaming of data through the central C# application allows for live debugging and interactive re-programming of the smart objects.

Visual Programming Language

Designers author the behaviour of their smart object using a high-level visual programming language interface. Inputs can be placed on the canvas and linked to outputs to enable the creation of basic behaviours. The visual programming language is composed of the following components.

Input Modules

Pineal provides support for several inputs commonly found on mobile devices, including discrete inputs (those that have an explicit 'trigger') and continuous inputs (those that respond to changing, always-available values).

Button (discrete) – Triggers an event when the generated button is pressed by remapping the user's touch directly to the touch screen via conductive material. Buttons can be of arbitrary shapes, and must be printed in conductive plastic.

Speech (discrete) – Triggers an event when the specified word is sensed. Speech recognition is done by the mobile device, with events being sent to the server once the word is detected. Multiple speech modules can be placed onto the canvas to detect and respond to multiple words.

Shake (discrete) – Triggers an event when a device is shaken. To recognize shakes, Pineal processes the accelerometer data and triggers the event when the magnitude of the acceleration exceeds a pre-defined threshold.

Orientation (discrete & continuous) – Triggers an event when the specified axis of orientation (azimuth, pitch or roll) exceeds a given value, or provides a continuous value that can be mapped to another module.

Web (discrete & continuous) – Triggers an event when a specified web event occurs (e.g., a hashtag is tweeted about), or provides a continuous value corresponding to the temperature forecast for a given city. Currently, only these two web functionalities are supported, but this feature could be extended to other services.

Output Modules

A number of types of output are supported by Pineal to allow the smart objects to have expressive characteristics.

Text Display – Displays a text sequence when an action takes place. This sequence is fetched from a specified sequence every time the module is activated.

Image – Displays an image selected from the pre-loaded library of images. The image location (x and y) can be updated via the input. The input is assumed to be screen coordinates, and appropriate mapping must be done by the module providing the input.

Simulated LEDs – Simulates the effect of an LED turning on when triggered by routing light from the screen to another location on the object via light pipes. The module takes a list of colours as an input, and each time the module is triggered it reads from the associated sequence.

Light Diffusers – Changes the colour of a light diffuser when triggered to enable the smart object to function as a low resolution ambient display. Conceptually, this is similar to the simulated LED module, except it allows the user to import a model to be placed on top of the base model as a light rather than having the light reroute to appear as an LED.

Sound – Plays a WAV file selected from the pre-loaded library when triggered.

Mapping Modules

Mapping modules support the input and output modules by providing means to store values and invoke timers. Currently Pineal supports two mapping modules:

Retrievable Sequences – text sequences and colour groups are lists that contain multiple entries. A calling module (e.g. input module) retrieves a value, which can be provided in increasing order, decreasing order, or random order, as specified by the user.

Timer – fetches a retrievable value from a sequence for a specified duration at a given interval in milliseconds. A discrete input value can start a timer.

Automated Model Configuration

Pineal steps the user through different actions that modify the 3D model to fit the mobile device and any new modifications (e.g. buttons). Some of these actions require explicit user input (e.g. positioning the screen). The required actions are determined by the behaviours generated in the visual programming panel. Figure 5 shows the different operations that Pineal supports.

The system currently has preset models for the watch and the phone, marked with specific locations (*pivot points*): screen center, screen corners, speakers, and camera. To fit the device models, the system automatically splits the base model into two pieces, creates a cavity for the device to house it, and adds corresponding alignment pins to each half. With the exception of the screen placement operation, all other operations take place once the base object has been divided into two parts. Each of these operations are separate steps, which the user completes in order when prompted by the system.

Screen Positioning and Device Placement

By default, the device is placed in the centre of the 3D model. Images, text and light diffuser modules require the display to

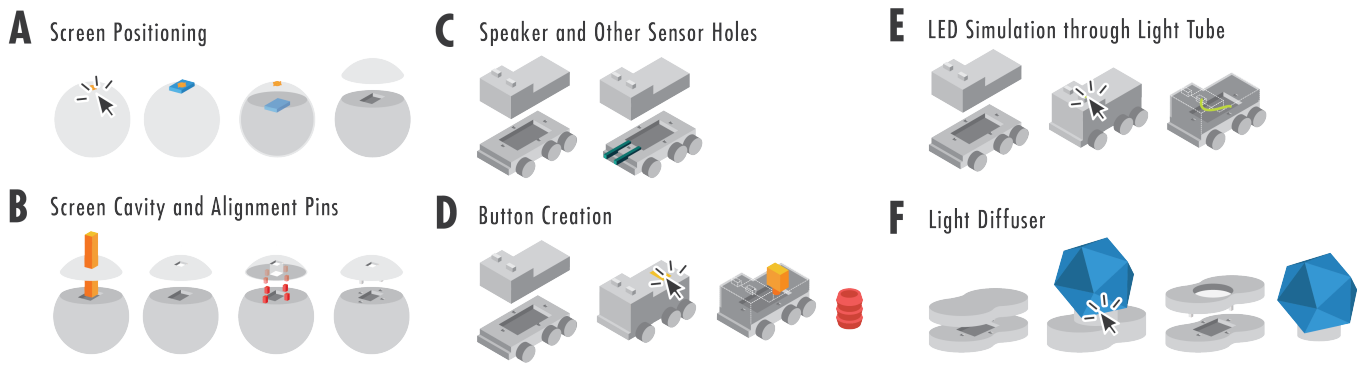

Figure 5. Operations performed by Pineal on the 3D models, which include (A) positioning the screen; (B) carving the screen and placing alignment pins; (C) Speaker or holes; (D) Buttons; (E) Simulated LED lights through light pipes; and (F) Light diffuser.

be exposed, and thus are configured by the user in the modelling workspace. As shown in Figure 5a once the user places the screen on the model, the device model is moved 2 cm towards the centre of the base model. This distance is the result of testing to ensure the screen is visible while still being securely fastened in the object. The system then cuts a plane on the base model about the top surface of the device model. The device model is subtracted from the bottom piece of the cut to create a cavity.

Screen Cavity and Alignment Pins

As shown on Figure 5b, Pineal can create openings on the device, as well as alignment pins. This is done through Boolean operations using models generated at runtime: 3 x 3 x 5 cm for the watch screen, 0.9 x 0.9 x 11 cm for the alignment pins. The system aligns the watch screen carving object with the screen centre of the device model, and then subtracts it from the top piece of the base model.

Alignment pins work in a similar fashion. Each alignment pin is duplicated to create one post and one hole for each alignment point. The hole is scaled at a rate of 1.1 in every dimension for clearance, so that the new printed object will fit together. Finally, the alignment objects are centered on the division plane in different locations with respect to the mobile device model (3 mm from the edges of the device cavity). The number of alignment pins can be changed programmatically up to 9. However, we found 4 to be enough for generating stable attachments. The system performs a Boolean union between the top model and the posts, and a Boolean subtract between the bottom model and the holes.

Speaker and Sensor Holes

In the same way that Pineal supports alignment pins and screen carvings, it also supports carvings for other sensors (Figure 5c). Currently supported configurations include speakers and camera, but one can also imagine creating holes for the microphone, volume buttons, etc. which can be easily integrated into the current system.

Button Creation

Pineal can accommodate modifiers, such as buttons, as shown in Figure 5d. To place a button, the user performs a brush selection in a region above the display location (since it requires capacitive input). This selection is smoothed by the system and then is extracted as a flat, separate object. This object is then extended downward to meet the screen, and is duplicated. The duplicate is subtracted from the top piece of the base model, while the original, now the new button, is rescaled by a factor of 0.9 for clearance. The system finds the centroid for the button face touching the screen, and adds a 7.5 x 7.5 x 5 mm cylinder. The button model can then be printed using a conductive material (in our case 1.75 mm Proto Pasta PLA). The conductive PLA proved a reliable trigger when contacting with human skin. To create restoring force, Pineal creates a spring model to be printed with a flexible material such as Thermoplaster Elastomer (in our case, NinjaFlex). This spring model can also be imported into the current instance of Meshmixer.

LEDs Simulated with Light Pipes

Another modifier is achieved using light pipes (fiber optic cables). In our case, we use 1.5 mm diameter fiber optic cables from Industrial Fiber Optics [37]. We found these to have the best light transfer and variety of available thicknesses. We also found the light transfer to be dependent on the screen brightness, with newer phone models emitting much brighter lights. While the light pipes are only 1.5 mm wide, the system generates 5 mm tubes, so that the user can fit multiple pipes within one opening.

To add the lights, the user selects a location on the model, as shown in Figure 5e. The Meshmixer client then creates a tube from that location to a screen coordinate of the mobile device. The system attempts to have the lights distant from each other, and will favour the corners of the screen. The mobile application then subdivides the screen into the number of lights placed, as determined by the program logic. For example, if two lights are placed, the screen will be divided in half.

Diffusers

As shown in Figure 5f, a user can import models to create custom shaped ambient lights that are illuminated by the mobile device's screen. These ambient light models attach on top of the base model that have already been cut by the system. The user can import a new model and place it on the base model within the 3d modelling workspace. The system then creates an opening so that the mobile device light can shine through it. The light structure model is hollowed to a depth of 1 mm, and then printed using transparent material (in our case MakerBot Natural PLA).

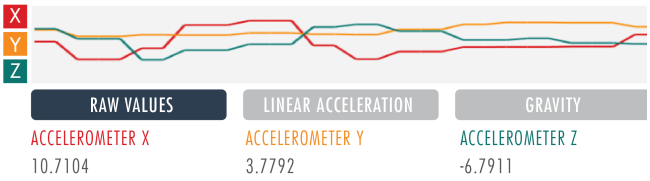

Figure 6. Live accelerometer data from mobile phone visualized by Pineal. The visualization plots the X, Y and Z

Raw Sensor View

To aid in the debugging and understanding of the constructed smart objects, Pineal includes an interface which displays raw sensor values that are live-streaming from the mobile device (Figure 6). Currently, Pineal provides raw views to acceleration, orientation and touch input data.

EXAMPLE SOUL-BODY OBJECTS

To illustrate the breadth of use-cases and functionality supported with Pineal, we built five prototype objects (Figure 1). These sample objects were chosen to cover a range of the design space of the soul-body prototyping paradigm, as illustrated in (Figure 7).

Toy Fire Truck

The sample walkthrough, described earlier, describes the workflow used to create the toy fire truck, which lights up and plays a sound when a tactile button is pressed. The firetruck is an example of using the mobile device to enable visual and audio output. It is also an example of transducing input – translating the phone's capacitive sensing into a physical button that provides tactile feedback when pressed.

Magic 8 Ball

A Magic 8 Ball is a ball with a display that reveals a random answer to a question when shaken. To construct this smart object, a designer first creates a model of a sphere and loads it into Pineal, which will adapt the model to a smartwatch. Once the sphere model is loaded the designer uses the visual programming language to create a module to sense when the object is shaken, and sets an output to display one of the following strings randomly: '*yes*; *'no'*; *'maybe'*; '*try again'*; '*never'*. The modeling steps can then be carried out, in this case consisting of specifying the location for the display and then clicking '*generate model*'. The model is automatically generated and can be exported for printing. The print creates two halves, with alignment pins, that house the watch. When the ball is shaken, a new response appears on screen. After testing the device, the designer can add a new response by editing the text field within the output module to include '*wrong question*'. The logic updates in real-time. This example shows the use of discrete motion input triggers and visual output.

Level

A level provides visual feedback to indicate when a surface is parallel with the ground. Currently there are existing mobile apps to simulate the functionality of a level, however, the form factor of a phone is not well-suited to being used as a level as it will easily fall over. To create a level in Pineal, a designer imports a model for a level, to which they add the smartwatch. The visual programming interface is used to map the horizontal location of a bubble image to be proportional to the watch's sensed orientation. The designer can then select the location of the screen on the model and export it for printing. This prototype demonstrates how Pineal is able to create linear mappings between inputs and outputs, and shows how motion-based input and visual image-based output can be utilized.

Ambient Display Planter

An ambient display changes its colour in response to data – in this case, live Twitter data. To develop an ambient display with Pineal, the designer first imports a base model (a rounded box). They can then use the visual programming language to follow the Twitter hashtag *#CHI2017*. They add a diffuser module and import another model for the light diffuser, a Bulbasaur planter. Finally, they add a colour group module and populate it with a list of colours, connect the Twitter module to the colour group module, and connect the colour group module to the diffuser module. Each time a new tweet is detected, the light scrolls through to the next colour from the list. Pineal also supports other live web data sources such as weather information.

Voice-Activated Light-bulb

A voice controlled light bulb could add ambience to a room, allowing the user to change the colour of the light by

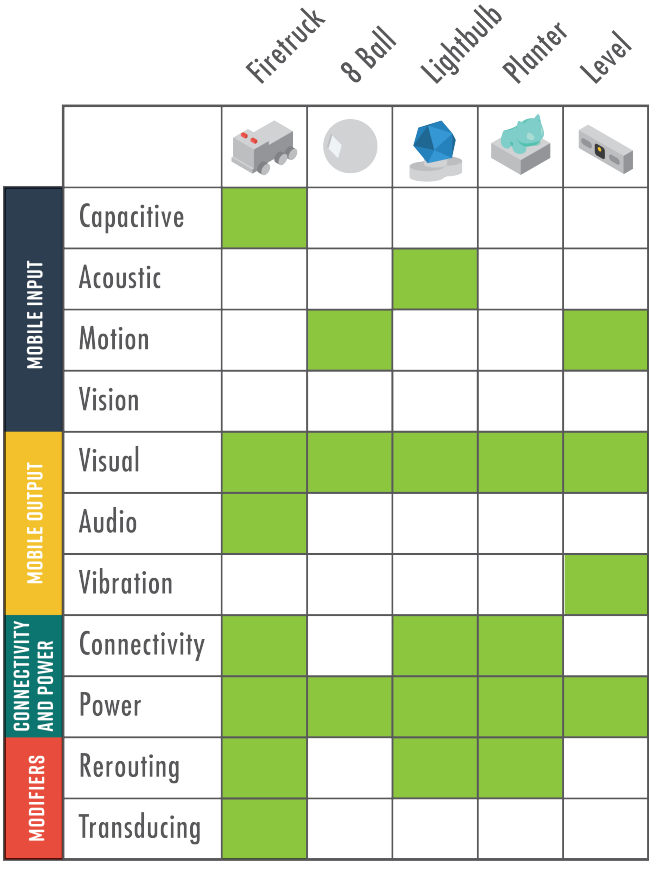

Figure 7. Illustration of design space, highlighting which areas are realized with each of the example prototypes.

speaking to it. To create a smart lightbulb that responds to voice commands, the designer creates different speech modules with different words: *"off", "yellow", "blue", "red", "white", "green"*. Each of these is respectively mapped to a different colour group containing an individual item: black, yellow, blue, red, white, and green. All of these colour groups attach to the light diffuser module. Now, each time a word is recognized, Pineal will send the appropriate colour to the light command, to which the phone will respond by changing the screen's colour. The designer then modifies the model by importing a lightbulb model and placing it on top of a round base model which houses the phone.

DISCUSSION AND FUTURE WORK

While the primary purpose of Pineal is to enable designers to rapidly prototype and iterate on ideas, it is possible that the created objects are of high-enough fidelity to serve as a permanent, functioning object. For instance, the Magic 8 Ball is of sufficient quality that it could be used as a customized novelty object for a child's birthday party, for instance. Future work is needed to explore what requirements vary when designing a 'body' that is intended to permanently house a mobile device (e.g., considerations for charging cables). That way, it may encourage reuse and repurposing of old mobile devices.

The visual programming language implemented in Pineal is not fully featured. The language was sufficiently developed to support the example prototypes, but it is not a feature-complete language (e.g., there are no variables, operators, etc.). Additionally, the vision component of the design space was left unexplored as many of the adaptations are similar to those explored in existing prototypes (e.g., routing light pipes to the camera lens).

Another limitation of Pineal is that the size of the mobile device limits the size of the object. Currently, only two devices are supported and their hardware limits the types of prototypes that are possible. For instance, the smart watch does not have rich audio output abilities, so designing a very small prototype that plays sounds is currently not possible. This limitation will be addressed as devices gain richer capabilities and more devices are added to the system. Next, for very complex prototypes, not all desired functionality may be satisfied with a single device. For instance, if an object requires an LCD display on the top as well as the side, a second device would need to be added. This is not currently supported in the system. Additionally, the current version does not optimize placement of the mobile device in an effort to minimize light pipe length, or distance from the speaker to the surface of the model.

Currently, only mobile devices are supported in Pineal, not custom electronics. In the future, devices such as .NET Gadgeteer or other microcontrollers could be integrated. This would allow for functionality not yet available on mobile devices (e.g., mechanical actuation).

Although a user evaluation is outside of the scope of this paper, it will be an important future step. In our current effort, the interactive prototypes validate the tool in showing that it is able to create a wide range of devices (as well as variations), as well as its coverage of the design space. Future work could explore the usability of the system, the effect of the workflow integration, and the expressivity.

Lastly, while Pineal aims to make the fabrication of these devices easier, it still requires some technical skill in operating a 3D printer, assembling components, and manipulating a visual program. However, there are currently efforts to understand how to make 3D printing more approachable and reliable [12], as well as increasing technical skill among designers.

CONCLUSION

Smart objects are ubiquitous, yet their development requires substantial effort and knowledge. Current mobile devices, such as smart watches and phones, possess a range of input and output capabilities that can be leveraged to prototype interactive devices, though accessing them and embedding them within objects requires some skill. With Pineal, users are able to rapidly prototype smart objects and modify both their form and function without substantial technical skill. The example prototype devices demonstrate the range of utility and use-cases that are enabled by this approach.

ACKNOWLEDGEMENTS

We would like to thank Nathaniel Hudson and Nobuyuki Umetani for their help with 3D printing and finding conductive and flexible materials. We would also like to thank the UI Research Group at Autodesk Research for their insightful discussions and feedback.

REFERENCES

1. Daniel Ashbrook, Patrick Baudisch, and Sean White. 2011. Nenya: Subtle and Eyes-free Mobile Input with a Magnetically-tracked Finger Ring. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, 2043–2046. http://doi.org/10.1145/1978942.1979238
2. Martin Azizyan, Ionut Constandache, and Romit Roy Choudhury. 2009. SurroundSense: mobile phone localization via ambience fingerprinting. *Proceedings of the 15th annual international conference on Mobile computing and networking*, ACM, 261–272. Retrieved September 18, 2016 from http://dl.acm.org/citation.cfm?id=1614350
3. Patrick Baudisch, Torsten Becker, and Frederik Rudeck. 2010. Lumino: Tangible Blocks for Tabletop Computers Based on Glass Fiber Bundles. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, 1165–1174. http://doi.org/10.1145/1753326.1753500
4. Tracey Booth, Simone Stumpf, Jon Bird, and Sara Jones. 2016. Crossed Wires: Investigating the Problems of End-User Developers in a Physical Computing Task. *Proceedings of the 2016 CHI*

Conference on Human Factors in Computing Systems, ACM, 3485–3497. Retrieved September 18, 2016 from http://dl.acm.org/citation.cfm?id=2858533

5. Anthony Chen, Stelian Coros, Jennifer Mankoff, and Scott Hudson E. 2015. Encore: 3D Printed Augmentation of Everyday Objects with Printed-Over, Affixed and Interlocked Attachments. *To appear in Proceedings of the ACM symposium on User Interface Software and Technology*.
6. Anthony Chen, Jeeun Kim, Jennifer Mankoff, Tovi Grossman, Stelian Coros, and Scott Hudson. 2016. Reprise: A Design Tool for Specifying, Generating, and Customizing 3D Printable Adaptations on Everyday Objects. *ACM UIST*.
7. Sean Follmer, David Carr, Emily Lovell, and Hiroshi Ishii. 2010. CopyCAD: Remixing Physical Objects with Copy and Paste from the Real World. *Adjunct Proceedings of the 23Nd Annual ACM Symposium on User Interface Software and Technology*, ACM, 381–382. http://doi.org/10.1145/1866218.1866230
8. Sean Follmer and Hiroshi Ishii. 2012. KidCAD: digitally remixing toys through tangible tools. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, 2401–2410. http://doi.org/10.1145/2207676.2208403
9. Sidhant Gupta, Daniel Morris, Shwetak Patel, and Desney Tan. 2012. SoundWave: Using the Doppler Effect to Sense Gestures. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, 1911–1914. http://doi.org/10.1145/2207676.2208331
10. John Harris, Stephanie Law, Kazuki Takashima, Ehud Sharlin, and Yoshifumi Kitamura. 2014. Calamaro: Perceiving Robotic Motion in the Wild. *Proceedings of the Second International Conference on Human-agent Interaction*, ACM, 59–66. http://doi.org/10.1145/2658861.2658891
11. Stephanie Houde and Charles Hill. 1997. What do prototypes prototype. *Handbook of human-computer interaction* 2: 367–381.
12. Nathaniel Hudson, Celena Alcock, and Parmit K. Chilana. 2016. Understanding Newcomers to 3D Printing: Motivations, Workflows, and Barriers of Casual Makers. *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, ACM, 384–396. http://doi.org/10.1145/2858036.2858266
13. Gierad Laput, Eric Brockmeyer, Moshe Mahler, Scott E. Hudson, and Chris Harrison. 2015. Acoustruments: Passive, Acoustically-driven, Interactive Controls for Handheld Devices. *ACM SIGGRAPH 2015 Emerging Technologies*, ACM, 3:1–3:1. http://doi.org/10.1145/2782782.2792490
14. Patrick Lazik and Anthony Rowe. 2012. Indoor pseudo-ranging of mobile devices using ultrasonic chirps. *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems*, ACM, 99–112. Retrieved September 18, 2016 from http://dl.acm.org/citation.cfm?id=2426667
15. E. Ohbuchi, H. Hanaizumi, and L. A. Hock. 2004. Barcode readers using the camera device in mobile phones. *2004 International Conference on Cyberworlds*, 260–265. http://doi.org/10.1109/CW.2004.23
16. Makoto Ono, Buntarou Shizuki, and Jiro Tanaka. 2013. Touch & Activate: Adding Interactivity to Existing Objects Using Active Acoustic Sensing. *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology*, ACM, 31–40. http://doi.org/10.1145/2501988.2501989
17. Jifei Ou, Gershon Dublon, Chin-Yi Cheng, Felix Heibeck, Karl Willis, and Hiroshi Ishii. 2016. Cilllia: 3D Printed Micro-Pillar Structures for Surface Texture, Actuation and Sensing. *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, ACM, 5753–5764. http://doi.org/10.1145/2858036.2858257
18. Raf Ramakers, Fraser Anderson, Tovi Grossman, and George Fitzmaurice. 2016. RetroFab: A Design Tool for Retrofitting Physical Interfaces Using Actuators, Sensors and 3D Printing. *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, ACM, 409–419. http://doi.org/10.1145/2858036.2858485
19. Munehiko Sato, Ivan Poupyrev, and Chris Harrison. 2012. Touché: Enhancing Touch Interaction on Humans, Screens, Liquids, and Everyday Objects. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, 483–492. http://doi.org/10.1145/2207676.2207743
20. Valkyrie Savage. 2016. Fabbed to Sense: Integrated Design of Geometry and Sensing Algorithms for Interactive Objects.
21. Valkyrie Savage, Colin Chang, and Björn Hartmann. 2013. Sauron: Embedded Single-camera Sensing of Printed Physical User Interfaces. *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology*, ACM, 447–456. http://doi.org/10.1145/2501988.2501992
22. Valkyrie Savage, Sean Follmer, Jingyi Li, and Björn Hartmann. 2015. Makers' Marks: Physical Markup for Designing and Fabricating Functional Objects. *Proceedings of User Interfaces and Software Technology*.
23. Valkyrie Savage, Andrew Head, Björn Hartmann, Dan B. Goldman, Gautham Mysore, and Wilmot Li. 2015. Lamello: Passive Acoustic Sensing for Tangible Input Components. *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, ACM, 1277–1280. http://doi.org/10.1145/2702123.2702207
24. Valkyrie Savage, Ryan Schmidt, Tovi Grossman, George Fitzmaurice, and Björn Hartmann. 2014. A Series of Tubes: Adding Interactivity to 3D Prints

Using Internal Pipes. *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, ACM, 3–12. http://doi.org/10.1145/2642918.2647374

25. Valkyrie Savage, Xiaohan Zhang, and Björn Hartmann. 2012. Midas: Fabricating Custom Capacitive Touch Sensors to Prototype Interactive Objects. *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*, ACM, 579–588. http://doi.org/10.1145/2380116.2380189
26. Martin Schmitz, Mohammadreza Khalilbeigi, Matthias Balwierz, Roman Lissermann, Max Mühlhäuser, and Jürgen Steimle. 2015. Capricate: A Fabrication Pipeline to Design and 3D Print Capacitive Touch Sensors for Interactive Objects. *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, ACM, 253–258. http://doi.org/10.1145/2807442.2807503
27. Jeremy Scott, David Dearman, Koji Yatani, and Khai N. Truong. 2010. Sensing Foot Gestures from the Pocket. *Proceedings of the 23Nd Annual ACM Symposium on User Interface Software and Technology*, ACM, 199–208. http://doi.org/10.1145/1866029.1866063
28. E. Short, K. Swift-Spong, J. Greczek, et al. 2014. How to train your DragonBot: Socially assistive robots for teaching children about nutrition through play. *The 23rd IEEE International Symposium on Robot and Human Interactive Communication*, 924–929. http://doi.org/10.1109/ROMAN.2014.6926371
29. Alexander Teibrich, Stefanie Mueller, Francois Guimbretiere, Robert Kovacs, Stefan Neubert, and Patrick Baudisch. 2015. Patching Physical Objects. *To appear in Proceedings of the ACM symposium on User Interface Software and Technology*.
30. Thingiverse.com. MakerBot Thingiverse. Retrieved September 18, 2016 from http://www.thingiverse.com/
31. Christian Weichel, Manfred Lau, and Hans Gellersen. 2013. Enclosed: A Component-centric Interface for Designing Prototype Enclosures. *Proceedings of the 7th International Conference on Tangible, Embedded and Embodied Interaction*, ACM, 215–218. http://doi.org/10.1145/2460625.2460659
32. Christian Weichel, Manfred Lau, David Kim, Nicolas Villar, and Hans W. Gellersen. 2014. MixFab: A Mixed-reality Environment for Personal Fabrication. *Proceedings of the 32Nd Annual ACM Conference on Human Factors in Computing Systems*, ACM, 3855–3864. http://doi.org/10.1145/2556288.2557090
33. M. Weiss, R. Jennings, R. Khoshabeh, et al. 2009. SLAP widgets: bridging the gap between virtual and physical controls on tabletops. ACM, 3229–3234.
34. Shota Yamanaka and Homei Miyashita. 2014. Vibkinesis: Notification by Direct Tap and "Dying Message" Using Vibronic Movement Controllable Smartphones. *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, ACM, 535–540. http://doi.org/10.1145/2642918.2647365
35. Neng-Hao Yu, Sung-Sheng Tsai, I-Chun Hsiao, et al. 2011. Clip-on Gadgets: Expanding Multi-touch Interaction Area with Unpowered Tactile Controls. *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, ACM, 367–372. http://doi.org/10.1145/2047196.2047243
36. Smartboy Development Kit. *Hyperkin Lab*. Retrieved September 18, 2016 from http://hyperkinlab.com/smartboy-development-kit/
37. Industrial Fiber Optics. Retrieved September 21, 2016 from http://www.i-fiberoptics.com/
38. Google Cardboard – Google VR. Retrieved September 18, 2016 from https://vr.google.com/cardboard/
39. Cycloramic. Retrieved September 18, 2016 from http://www.cycloramic.com/
40. 3D Models for Professionals: TurboSquid. Retrieved September 18, 2016 from http://www.turbosquid.com/