Robiot: A Design Tool for Actuating Everyday Objects with Automatically Generated 3D Printable Mechanisms

Jiahao Li
UCLA HCI Research
Los Angeles, United States
ljhnick@g.ucla.edu

Jeeeun Kim
CS&E Texas A&M University
College Station, United States
jeeeun.kim@tamu.edu

Xiang 'Anthony' Chen
UCLA HCI Research
Los Angeles, United States
xac@ucla.edu

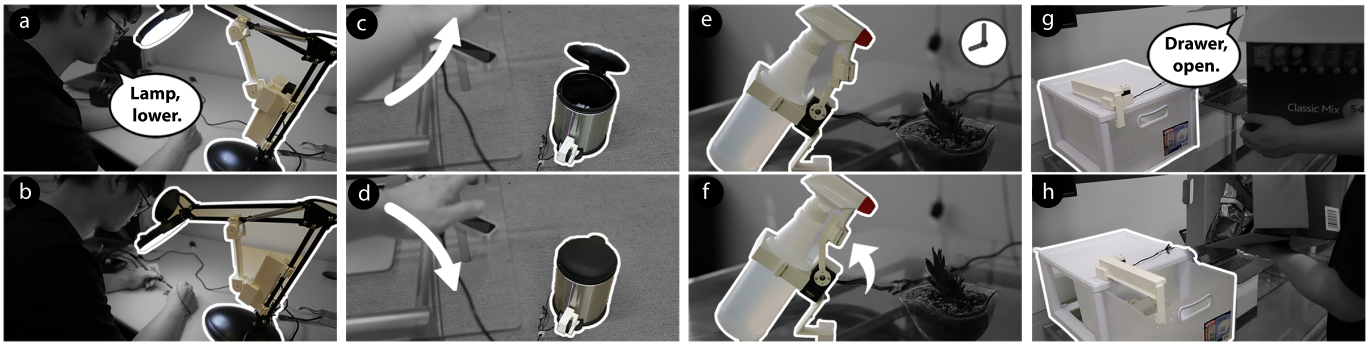

Figure 1. Robiot enables end-users to take a video and automatically generates mechanisms that can actuate everyday objects to perform simple physical tasks. All the four examples were designed by participants in our design session: lowering a lamp while busy soldering (a), waving a hand up to open a trashcan afar (c), setting up a scheduled house plant water spraying system (e), and asking the drawer to open itself when having no extra hand (g).

ABSTRACT

Users can now easily communicate *digital* information with an Internet of Things; in contrast, there remains a lack of support to automate *physical* tasks that involve legacy static objects, *e.g.*, adjusting a desk lamp's angle for optimal brightness, turning on/off a manual faucet when washing dishes, sliding a window to maintain a preferred indoor temperature. Automating these simple physical tasks has the potential to improve people's quality of life, which is particularly important for people with a disability or in situational impairment.

We present Robiot—a design tool for generating mechanisms that can be attached to, motorized, and actuating legacy static objects to perform simple physical tasks. Users only need to take a short video manipulating an object to demonstrate an intended physical behavior. Robiot then extracts requisite parameters and automatically generates 3D models of the enabling actuation mechanisms by performing a scene and motion analysis of the 2D video in alignment with the object's 3D model. In an hour-long design session, six participants used Robiot to actuate seven everyday objects, imbuing them with the robotic capability to automate various physical tasks.

ACM Classification Keywords

H.5.2. Information interfaces and presentation: User Interfaces-Input devices and strategies

Author Keywords

Design tool; actuation; everyday objects; generative design.

INTRODUCTION

An Internet of Things (IoT) is becoming increasingly ubiquitous in our everyday environments. Current IoT primarily focuses on sensing technologies that enable users to easily exchange *digital* information with spatially-distributed devices, *e.g.*, remotely turning on a desk lamp from your phone, feeding data from rooftop weather sensors to the sprinkler control.

In contrast, there remains a lack of support for *physical* tasks, *e.g.*, adjusting the angles of a desk lamp for optimal brightness as you perform a soldering task. Automating physical tasks has important implications for people with a disability or in a situational impairment. For example, turning on a manual faucet without touching it becomes useful when both of your hands are dirty, opening a pantry is helpful when you are holding bags of groceries, and it is convenient to have a window that closes by itself when sensing the temperature drops.

The recent development of actuatable appliances (*e.g.*, smart blinds [39], switches [29], and TV deck[31]) and reconfigurable furniture [15] suggests a future of ubiquitous robotic things: akin to how ubiquitous computing imagined a world of omnipresent computational power, we can envision a future of everyday objects and appliances equipped with robotic

Submitted to UIST 2019.

capabilities to interactively carry out a series of complex actions. With such physical tasks and dynamic actions that assist people in a variety of contexts, the future of everyday robotic IoT devices can open a door for a new world of everyday interactive systems.

To bridge us to this vision, instead of waiting to replace and upgrade a whole world of legacy static objects with fully automation-intended gadgets, one reasonable first step is to come up with solutions that allow end-users to augment these objects with actuatable behaviors. Leveraging a democratization of robotic kits and rapid prototyping machines, prior work has proposed the design of external [56, 30] or add-on [45, 53] actuation mechanisms to operate a physical control interface. To generalize the approach to a wider range of everyday things, past research primarily focuses on generating *passive* adaptations to reduce the effort of operating handheld objects [9]. In contrast, *active* mechanisms on everyday objects remains a nascent topic—very little is known how average users can create actuation mechanisms with desired motion to automate everyday objects for even the simplest physical tasks.

We present Robiot—a design-by-demonstration tool for end-users to fabricate add-on mechanisms to actuate legacy static objects for everyday physical tasks. Using Robiot, users only need to take a short video manipulating an object to demonstrate an intended physical behavior, such as lowering a desk lamp, squeezing a spray bottle, or opening a drawer. To generate the actuation mechanisms, Robiot extracts two requisite parameters: *(i)* first it performs a scene and motion analysis in the video domain to infer the **type of joint** corresponding to the demonstrated motion; *(ii)* then it retrieves the object's 3D model from a pre-constructed repository and identifies **maneuverable/ground parts**—where on the object to ground a mechanism and where to let it exert the actuation. These two parameters lead to a set of candidate mechanism designs, which can be further filtered by adjusting several design parameters, *e.g.*, range of motion, torque and speed. Figure 1 and 3-6 show several of exemplary applications designed using Robiot, fabricated and installed by our study participants. In an hour-long design session, six participants successfully designed 14 mechanisms. After 3D printing, they were able to assemble and install mechanisms on existing objects with instructions given by the system, finding its usefulness for future use in their environment to actuate own things.

Contribution

Our main contribution is an end-to-end pipeline that requires very minimal user input to automatically generate 3D printable actuation mechanisms by a novel combination of 2D video analysis and 3D geometry processing. Although we demonstrated 3D printing as the main fabrication technique, our pipeline's ability to extract mechanically meaningful information from user demonstration can potentially generalize to other manufacturing techniques.

Limitations

We currently focus on small rather than furniture-scale objects. Our end-to-end pipeline requires a pre-existed 3D model of the target legacy object (although the advancement in scanning technology might soon dispense with such requirement). We focus on one degree of freedom at a time (*e.g.*, a user would demonstrate rotating one joint of the lamp rather than multiple joints). We assume an ideal capturing angle of the camera, *i.e.*, its orientation as orthogonal as possible to the object's motion path to best extract movements from the video and there is no occlusion of the object in the video; Finally, our focus is on design and fabrication; techniques that interact with actuated objects is beyond the present scope and will be addressed in future work.

RELATED WORK

Robiot provides end-users with a design tool that can create mechanisms to attach to and actuate existing everyday objects for simple physical tasks. This goal cross-cuts three areas of prior work: *(i)* personal robots that interact with and assist people with their physical tasks; *(ii)* reality-based design tools that extract information from the real world to create designs that can in turns augment the real world; and *(iii)* computational methods of designing and prototyping functional objects.

Personal Robotics

Research in personal robotics has sought to democratize robots to assist people with a range of physical tasks in home and offices. Designing robots with motion enables users to communicate and dynamically engage with robots, such as having a robot aid a user's office tasks [19]. Further, to consider a broader audience, personal robots are taking an increasingly important role in enabling people with disabilities [7].

At a smaller scale, robots can perform grasping and manipulation of everyday objects. Grasping and manipulation remains a fundamental challenge that drives new algorithmic and system designs [50]. At the application level, grasping and manipulation also enables robots to assist people's daily living tasks, from fetching a mug [55] to taking medications [43]. On the other hand, robots can now support people's whole-body activity in a large scale, such as a soft assistive robot that provides scaffolding and protection for elderly people when they take a bath [35]. Robot assistance is also frequently used in navigation, from orienting oneself in a work area [54] to maneuvering in a shopping mall [17].

Despite such development, the range of physical tasks a personal robot can handle remains limited due to the large variety of tasks as well as the different objects involved. To make robot's ability flexible and scalable, prior work has been focusing on using programming by demonstration to teach robot custom behaviors [16, 6, 58]. Although in this way the physical tasks become programmable to a robot, the performance is by and large determined by the one-size-fits-all design of the robot that is available. Worse, at times such a robot might not even be available to perform a bespoke task. To overcome this limit, prior work such as Coros et al.'s provides an expressive means for specifying robotic behaviors by sketching a user-defined motion path, which generates a corresponding linkage design for an automata [12]. Instead of sketching, our design tool allows a user to directly manipulate an existing object to specify a desired physical behavior, and then automatically generates a behavior-specific robotic component that

can retrofit to and actuate everyday objects to assist users in automating simple physical tasks.

Reality-based Design Tools

Related to enabling design of actuating physical tasks, past research has explored a class of reality-based design tools that address two important issues: *(i)* how to extract information from the physical world that can lead to *(ii)* creating solutions that can leverage personal fabrication to augment the physical world.

The need to extract information from the physical world coincides with many research goals of Augmented Reality (AR). Early work such as DART allows designers to rapidly transition storyboards to a work experience based on a camera view and an live AR authoring environment [34] A recent surge of AR technologies gives rise to prototyping tools that incorporate real-world information in the design process, such as directly viewing, positioning and iterating a 2D sketch in a 3D real environment [3, 40, 57, 25, 44].

Obtaining an understanding of the physical world informs new ideas of augmentation, often manifested as the idea of 'mechanical hijacking' first demonstrated by Davidoff et al. [13] and later extended by Chen et al. in adapting hand-operated objects for easier manipulation [9]. AutoConnect enables the automatic generation of structures that connect (i.e., holding in place) two user-selected objects based on their scanned digital representations [27]. Printy allows novice users to fabricate fully-functional internet-connected object [4]. Patching provides a hybrid platform that scans, mills, and additively fabricate new components to replace part of an existing object with augmented functionality [38]. Facade uses the crowd to annotate a visually inaccessible physical interface (e.g., buttons without tactile feedback) and generates 3D printable tactile overlay to assist visually-impaired people to use these interfaces [18]. Perhaps the most related to our work is RetroFab, which offers an authoring tool to scan an existing physical interface and automate its controls by adding an enclosure consisting of mechanical and electronic devices [45]. However, Retrofab only addresses actuation specific to operating physical controls and does not consider a more general way to encompass motions from various other everyday objects.

To summarize, as shown in Table 1, the most related work in reality-based design tools either focuses on physical interfaces [18, 45], or only addresses everyday objects with passive add-on components [9]. Robiot complements existing research with a generative pipeline to create active actuation mechanisms on everyday objects.

Table 1. Robiot goes beyond prior work with active mechanisms to actuate a range of everyday objects.

	Physical Interfaces $\rightarrow$	Everyday Objects
Passive	Facade [18]	Reprise [9]
Active	Retrofab [45]	**Robiot**

Designing and Prototyping Functional Objects

The eventual goal of our tool is to generate fabrication-ready actuation mechanisms. To achieve such functional design, prior work has demonstrated two approaches: assembly-based and generative design.

Assembly-based solutions allow users to put together off-the-shelf components for a functional, often complex object. For example, consider a plethora of robotic kits, such as the popular LEGO Mindstorms [30] used in early work of mechanically 'hijacking' the control of physical devices [13]. TrussFormer enables users to 3D print large-scale kinetic structures [26]. Zykov et al.'s Molecubes is an open-source modular robotics kit that provides a low-cost, ruggedized and expandable platform with software support for visual and control design [59]. Schweikardt and Gross demonstrated the expressiveness of roBlocks—a reconfigurable modular robotic prototyping tool where small, magnetic, heterogeneous components can be snapped together to create large and complex constructs [52]. Grafter largely automates the process of extracting and recombining mechanical elements from 3D printed machines and affords extracting groups of mechanical elements that already work together, such as axles and their bearings or pairs of gears [49].

Generative design allows users to specify their high-level design goals while leaving the low-level functional considerations to a generative algorithmic process. Autodesk's Project DreamCatcher takes a data-driven approach to generate hundreds of thousands of design options based on input functional requirements [47]. To explore the many generated design alternatives, DreamLens provides a visualization platform built for exploring large-scale design datasets [36]. DreamSketch allows a user to integrate generatively designed components with the workflow of sketching [24]. To further incorporate users' intents, Forte loops user input into the optimization process to create structures that not only meet functional requirement but also mimic users' sketches [10]. Generative design also enables end-user design and fabrication of robots, from creating linkages to exhibit specific motion path [5], to the automation of a comprehensive set of design considerations, including different morphology, proportions, gait and motions [37].

It is also possible to take a hybrid approach that combines existing components as well as a generative process. Desai et al. propose an assembly-aware design pipeline that automatically lays out user-defined electromechanical components and creates 3D printable enclosures to assemble the robot [14]. Robiot employs a hybrid approach: the mechanisms design is generated based on the intrinsic geometry of the object as well as the extrinsic motion demonstrated by the user; then these generated components are assembled together with existing parts (eg/ motor) and installed on the object.

ROBIOT: MECHANISMS TO ACTUATE EVERYDAY THINGS

Overview of Robiot's Workflow

As shown in Figure 2, our technical contribution is an end-to-end pipeline that requires minimum user input to generate 3D printable mechanism, which actuates legacy static objects. Using Robiot, a user simply takes a short video demonstrating how they want an object in motion. A wide range of these rigid-body objects' motion can be expressed as linear or rotational

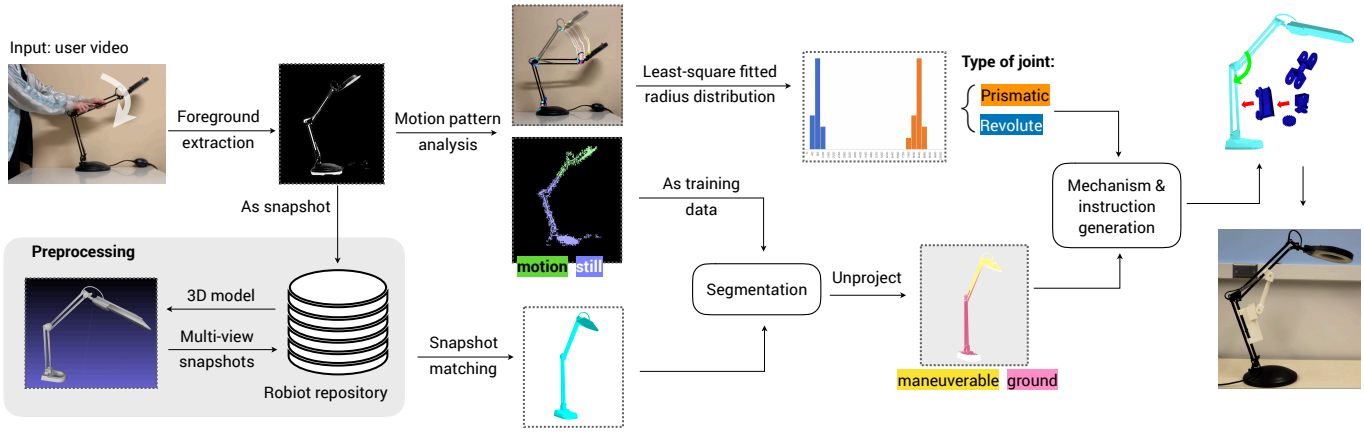

Figure 2. Overview of Robiot's end-to-end pipeline for generating actuation mechanisms from a user's input video.

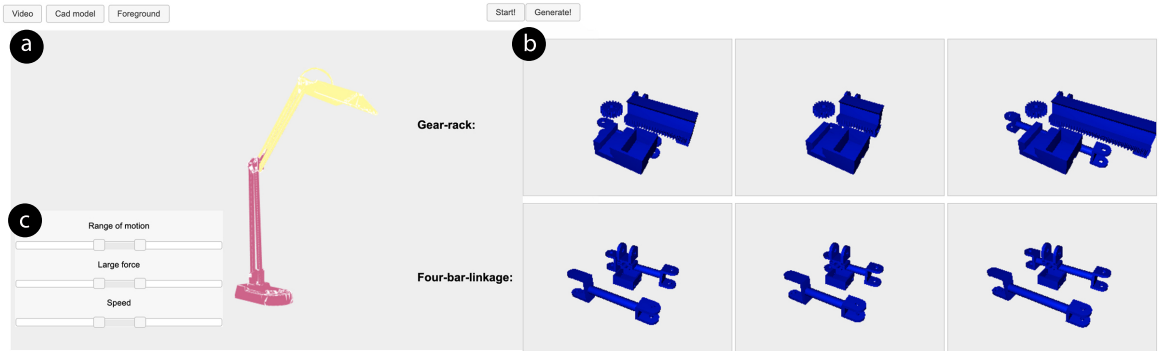

Figure 3. A screenshot of the user interface provided to the user once Robiot generated a set of candidate mechanisms from a given video demonstration: the corresponding 3D model (a), the generated mechanisms (b), and sliders to filter the list by range of motion, torque and speed (c).

[9]. Thus to generate the enabling mechanisms, Robiot models the actuated components in two types of *joints*: prismatic (linear) and revolute (rotational). As shown in Figure 2, Robiot performs a scene and motion analysis of the input video to extract the following information.

Type of joint - The object is extracted from the video and an optical flow technique analyzes the motion, which is used to classify whether the motion is linear (prismatic) or rotational (revolute). This information leads to specific mechanisms that can actuate the object to behave as the user demonstrates in the video.

Maneuverable vs. Ground parts - First a 3D model is retrieved from an existing repository that contains preprocessed information for matching which 3D model best corresponds to the object as viewed in motion. Further, results from the above optical analysis are used to segment the 3D model into maneuverable and ground parts. Robiot then automatically generates a list of possible mechanism designs that also raise implicit constraints inferred from the input video, e.g., the size of the object's components, the required minimal torque. As shown in Figure 3, Robiot also provides more advanced features that allow a user to tweak and filter design options by adjusting preferred range of motion, torque, and speed.

Below we first showcase a series of examples designed using Robiot's workflow while leaving the technical details later in following sections.

Examples Generated by Robiot

We present a series of examples created by Robiot, which automatically generates the 3D models of the mechanisms from a user input video. As we focus on the design tool, all the subsequent interaction was developed ad hoc (*e.g.*, using commercially available voice or gesture sensing input devices) as a way to demonstrate Robiot's potential to perform simple physical tasks for users.

Figure 1 showcases exemplary applications with mechanisms installed to everyday objects at home and offices. When both of user's hands are occupied and tied to a task, such as for soldering, an automatically adjustable lamp with a reading lens can come closer to him, assisting his delicate task such as soldering (Figure 1a,b). In the similar vein, as shown in Figure 1, when user's hands are full with garbage grabbed from a counter top (c) and a heavy box with clutter (g), automatically opening trash can (d) and drawer (h) become convenient for her to ease the cleaning task. For a busy office worker who is likely to forget to water the plant regularly, a squeezer mechanism attached to a spray bottle will do the watering tasks according to a predefined schedule (e-f).

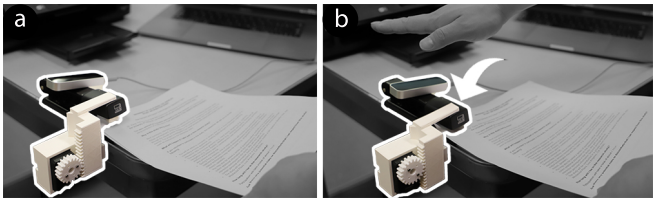

Figure 4. Automatic stapler is one common office appliance, that Robiot can robotize from a cheap manual stapler

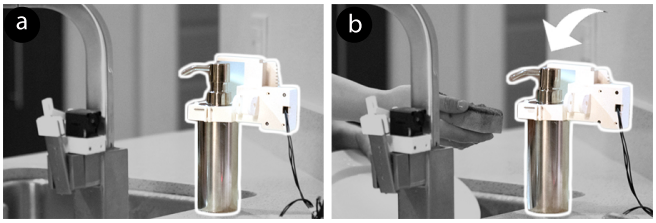

Figure 5. Robiot mechanism attached on top of a soap bottle performs pressing to squeeze liquids without touching

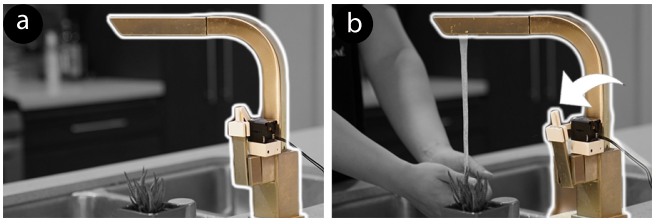

Figure 6. A manual faucet can automatically turn to release water by attaching a mechanism to pull the handle.

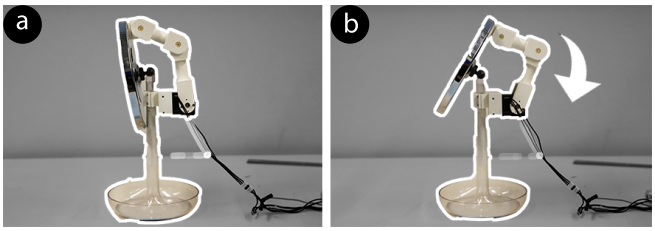

Figure 7. Automatically adjusting mirror can aid a user who wears make-ups and contact lens using both hands

Figure 4-7 further demonstrate a wider range of example applications with actuation mechanism attachments. There exist many commercial automated staplers, which we can replicate using a Robiot-created mechanism that performs the same task using a cheap stapler (Figure 4). When the user's both hands are dirty, a soap bottle that automatically presses the top to squeeze liquid soap (Figure 5) on her hand and a water tap turning on water help those who do not want to spread the mess (Figure 6). As another example, wearing make-ups is one of the most complex tasks, from which people can benefit from a mirror with automatable angle adjustment so that its usage becomes hands-free.

IMPLEMENTATION

In this section, we detail step by step process of creating a mechanism, from user input to ready-to-print 3D model for end users to assemble and install with the instruction.

Preprocessing

Robiot's analysis of real-world objects starts from a repository of 3D models corresponding to it. As opposed to 3D scanning of the object, the models in the repository offer clearer and better-defined mesh information than the currently-limited scanned data that often requires additional post-scan processing. Such a repository can also be populated with manufacturers cataloging the CAD models at design time and 3rd party dataset. However, the advancement in scanning technology might soon provide a viable alternative in lieu of a repository.

As shown in Figure 2, for each object, Robiot performs a one-time preprocessing step by taking snapshots of the 3D model at a set of predefined locations spherically around the object. Snapshots are stored together with the object's 3D model and will be used for retrieving the 3D model as detailed below.

#1 Extracting an object's 2D and 3D representations

The input of Robiot's generative pipeline is a video of a user manually manipulating an object to demonstrate the action, expected to be produced by some mechanisms. Our first step is to extract the object's 2D and 3D representations from the video (based on the preprocessed 3D repository).

As shown in Figure 2, after identifying the first stable frame we perform a scene analysis—a foreground extraction to obtain the object's 2D representation as a binary mask M_{video} (with the 1's representing the object and 0's the background). We implement this step using GrabCut [48], although other methods (e.g., deep learning based direct object segmentation [28]) can also be used to replace this component in Robiot's pipeline.

Next, we use M_{video} to retrieve a 3D model from the Robiot repository by snapshot matching, *i.e.*, finding a 3D model that has a snapshot that best matches M_{video}. For each 3D model, we perform a stepwise searching process. Specifically, for each snapshot we first binarize it into $M_{snapshot}$ and scale it to match the aspect ratio of M_{video}. We then measure how well $M_{snapshot}$ matches M_{video} by computing a matching score:

$$s_{mathcing} = \frac{sum(M_{video} \wedge M_{snapshot})}{\sqrt{sum(M_{video}) \cdot sum(M_{snapshot})}} \quad (1)$$

We identify the 3D model that has the highest $s_{matching}$ as the object's 3D representation; we also save the corresponding $M_{snapshot}$ for latter processing.

#2 Determining the type of joint

Once the area containing the object is extracted, the next step is to perform a video-based motion pattern analysis. First we extract feature points using the Shi-Tomasi corner detector [23], and then use the Lucas-Kanade method [33] to calculate the optical flow of these feature points during the course of the video. As shown in Figure 8, each feature point is 'tracked' frame by frame, resulting in a 2D trajectory comprised of an array of X/Y coordinates. Next we filter out noises and jitters by setting an empirically defined threshold to cut off feature points whose trajectory coordinates with a low covariance.

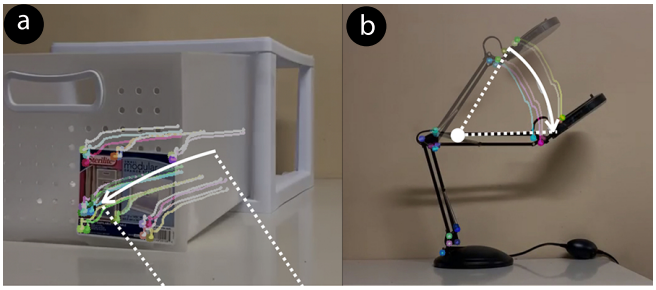

Figure 8. Optical flow analysis shows distinct difference between prismatic (a) and revolute (b) joints by comparing the least-square fitted radius (prismatic joint's fitted radius is much larger, extending beyond the camera view).

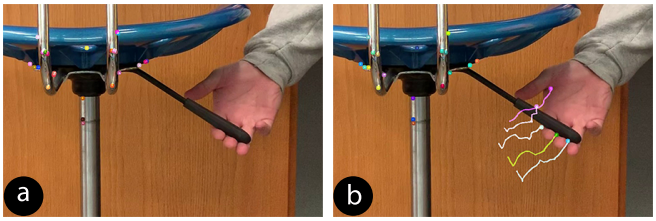

Figure 9. Some objects due to their geometry and texture provide very few feature points. To solve this problem, we incorporate a user's operating hand whose motion correlates to the manipulated object's.

Now that we have collected trajectories representing how the object should be actuated, the next step is to determine whether such motion can be enabled using a revolute or prismatic joint. For each trajectory, we use a least-square method to fit it to an arc. The rationale is that if the actuation is revolute (rotational), the pivot should physically be part of the object thus the fitted radius must be significantly smaller than the fitted radius of a prismatic actuation. Now we compare the radius and the distance between two states– initial vs. final. Note that without occlusion, the initial/final state can be robustly extracted from the first/last frame of the demonstration video, respectively. As the length of an arc is approximately equal to $len = R\sin\alpha$ (where α is the rotation angle) if $len < \frac{R}{4}$, resulting in $\alpha < 15^\circ$, the joint is regarded as prismatic joint, otherwise it is regarded as revolute joint. As shown in Figure 8, we compute a distribution of the fitted radii from all the trajectories, which exhibit a clear separation between the two types of joint.

One challenge here is that some objects might have very few 'sharp corners' that can be used as feature points for the optical flow analysis. To address this, we incorporate the user's hand, which provides ample feature points. As shown in figure 9, as the hand is used to manipulate the object, its motion must match that of the object's. Thus we use the hand as a supplement when there is a lack of feature points detectable from the target object. We detect the hand's position using a skin color based method [46].

#3 Finding maneuverable parts and grounds

Having identified the type of joint, the next step is to find out where and how to attach a mechanism to an object. The joint should be fixed to part of the object that does not move, *ground*, and should 'grab on' to part of the object that can move, *maneuverable parts*. The key of the following steps is to identify maneuverable parts and grounds, not just in the video domain but also in the 3D space as eventually we will generate 3D models of mechanisms attached to the actual object.

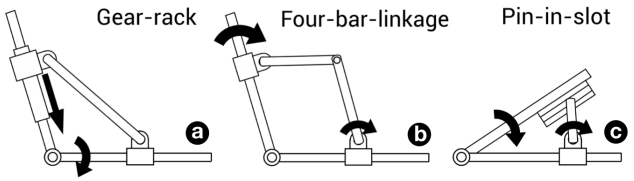

Figure 10. Technical sketches of the three mechanisms showing how they are actuated.

Recall that the above motion analysis (Figure 8) has already identified a set of feature points with significant motion trajectory. We use these feature points to train a segmentation model (we use k-nearest neighbor [2]) and apply it to the snapshot ($M_{snapshot}$) of the object's 3D model, marking each pixel either as maneuverable or ground. As shown in Figure 2, we then unproject pixels of the snapshot back to the 3D model using ray casting. As a result, each face of the 3D model is associated with one or more snapshot pixels. We take a majority vote to determine whether the face should be considered maneuverable or ground.

#4 Generating Actuation Mechanisms & Instructions

As described in an overview, Robiot provides a list of possible mechanisms as options from the library—gear-rack, four-bar linkage, and pin-in-slot—that can afford actuating various everyday objects. Figure 10 shows how the three mechanisms are actuated basically. Below we describe how mechanisms are chosen based on constraints and generated from a core set of parameters.

Prismatic joint (linear motion) . Gear rack is the only mechanism that we use for both types of joints. As shown in Figure 11(a), the gear-rack system translates the rotary motion of the motor into linear motion, which enables a lamp height (linear) to be adjusted by a motor (rotation). Most of the gear-rack components are standardized, except for the length or rack to be determined by the range of motion, computed from the union of trajectories from the optical flow analysis.

Revolute joint (rotational motion) . When the type of joint is revolute, all three (See Figure 11) mechanisms can be considered and modeled after the same set of parameters. Each of the three mechanisms can be generated based on three parameters and their relations: the lengths of the two links l_1 and l_2 (see Figure 11) and the range of motion (rotation) α.

Despite their similarity, the choice of one mechanism over another is based on intrinsic constraints. Gear-rack allows larger motion and larger torque at a cost of larger installation space, whereas pin-in-slot could be installed in a smaller space but cannot provide large torques; four-bar-linkage stands in between and has much fewer applicability constraints.

Specifically, for gear-rack as a revolute joint, two constraints contribute: *(i)* the size of the motor (currently set to default in the system) and the two connecting components limit the

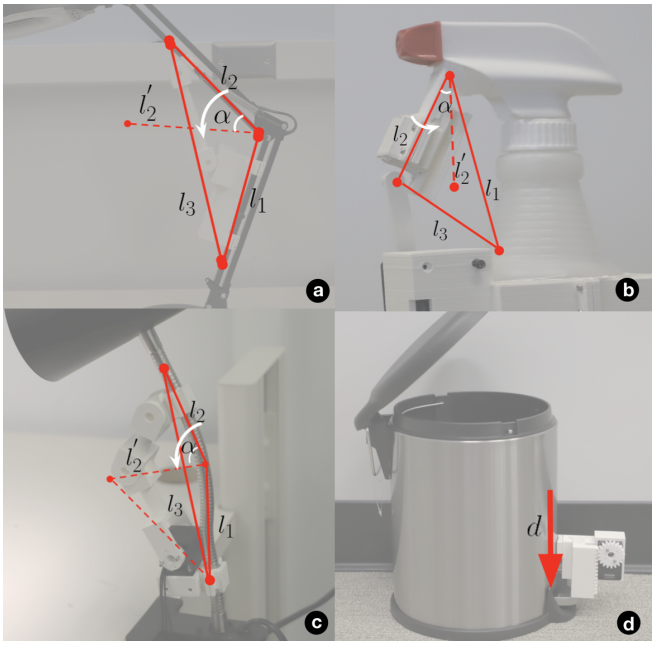

Figure 11. Three mechanisms and their parameters (gear rack appears twice for both types of joints). For revolute joints, different mechanisms can be considered and modeled after the same set of parameters – l_1, l_2, l_3 and α.

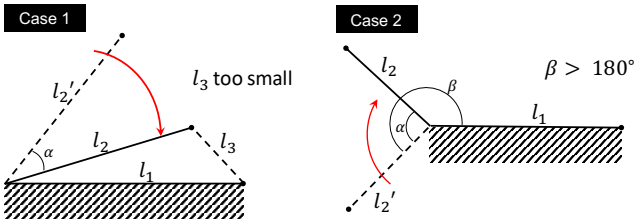

Figure 12. Two cases where gear rack does not apply

minimum length of l_3; *(ii)* the range of rotation α cannot exceed 180°(figure 12). For pin-in-slot, the main constraint is whether it can meet the required torque on the pivot τ_p. For the same motor torque τ_m, pin-in-slot generates a much smaller τ_p ($\tau_p = \tau_m \sin(\beta - 90°)/l_2$) compared to the other two mechanisms.

Programmatically generating 3D models of mechanisms All mechanisms are generated procedurally using basic primitives and boolean operations. For example, for gear-rack, we use trapezoid shape to generate the teeth of the rack and use rectangle or self-defined closed curve to extrude to get the 3D model; the gear is based on an existing example provided by OpenJSCAD. For pin-in-slot, the link attached to the motor to transmit the power from the motor contains a wheel and a connecting rod, which are generated using cylinders. For four-bar-linkage, the link is cylinder-shaped and we use cylinders to generate the connecting joints between two links.

Installation: fasteners and instructions For fastening, we employ Chen *et al.*'s method [8] to compute the circumference of a cross section corresponding to a maneuverable/ground part. We then generate a pipe clamp as part of the mechanism that can be bolted to fasten the mechanism onto existing objects. Other attachment techniques (*e.g.*, [8, 27]) can be also applied based on the target shape.

User interface After the system automatically generates a set of recommended mechanisms, Robiot's UI allows a user to further specify desired strengths and properties (e.g. range of motion, required torque and speed of the motion).

Finally, we generate instructions to assist end-users to install the generated mechanisms onto existing objects. We provide standard instructions for fastening a pipe clamp with bolts and configuring a motor; for each specific case, as shown in figure 14, we also visualize where on the object to install which part.

Software and Hardware

Robiot's front end is written in JavaScript using jQuery[1] for UI development, three.js[2] for 3D graphics, and OpenJSCAD[3] for procedurally generating the geometry of actuation mechanisms. The back end Python. Everything runs on a MacBook Pro (15-inch, 2016 year) with a 2.7 GHz Intel i7 and 16 GB 2133 MHz LPDDR3 memory. In our design session and demonstrations, the front end runs on a Google Chrome web browser. We use Dynamixel XL-430 W-250 motors to power the actuation mechanisms, which were all 3D printed using an Ultimaker S5 using primarily white PLA.

DESIGN SESSIONS

To validate Robiot, we conducted informal, qualitative design sessions with six participants (aged 20-25, female=3, male=3). The objective of the study is to let participants create mechanism designs to actuate a set of everyday objects using Robiot's generative pipeline. In so doing, we try to elicit users' initial reaction and feedback to the system in order to validate Robiot's easiness to use, its usefulness for automating physical tasks, as well as what to further improve to enhance its efficiency in robotizing things.

Participants

We recruited participants from the university. One participant had a Mechanical Engineering background and one an Electrical and Computer Engineering background, both of which self-reported that they were knowledgeable in mechanical engineering. The other participants did not have any engineering background. Amongst all participants, three had experienced CAD systems, while the others did not. One participant did not even know what CAD means.

Apparatus, Tasks and Procedure

There were two different sessions in two days to budget time for 3D printing mechanisms that participants designed on the first day, which they continued to assemble and interact with them on the second day.

Design Session (Day 1) - started with a five-minute quick tutorial. We introduced to a participant how Robiot works step by step using a simple educational example—making an

[1] https://jqueryui.com/
[2] https://threejs.org/
[3] https://openjscad.org/

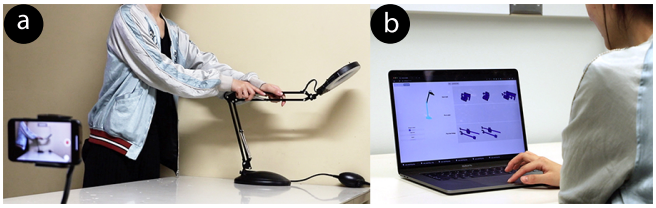

Figure 13. Participants in our study took videos of them manipulating everyday objects (a), whereby Robiot generated corresponding actuation mechanisms for participants to explore on our user interface (b)

old-school stapler automatic (Figure 4). Once the participant understood the concept and the process of Robiot, they proceeded to try out our design tool. Participants were free to choose at least two from a set of seven objects we provided, including lamps (×3), spray bottle, squeeze bottle, trash can, make-up mirror, and drawer. These objects strike a balance between prismatic and revolute joints, also variations of the same object (lamps), and between different actuation types for similar functionalities (spray vs. squeeze bottles).

The main tasks consisted of participants using Robiot to create an actuation mechanism by taking a video (using an iPhone XS max running iOS 12.1.4) provided by us) as they manipulated each object. To avoid leading the participant, for each object we showed them images of the initial state (*e.g.*, drawer closed) and final state (*e.g.*, drawer open). The participant was asked to manipulate each object to achieve the final state.

After taking the video, participants explored and selected from a number of mechanism designs generated by Robiot from a viewer using a laptop (Figure 13b). As we would fabricate user-created mechanisms, we had to budget the printing time, thus allowing each participant to choose two objects, each of which with one mechanism design. The first session took about 45 minutes.

Assembly & Interaction Session (Day 2) took place after we 3D printed participants' designs. On Day 2, the participants were given instructions for assembly generated by Robiot Figure 14, based on which they assembled the printed mechanisms and attach to the corresponding objects. Participants could try out interaction with the actuated objects using predefined gestures implemented via a Leap Motion[4]. Our main goal is to let participants experience their created mechanisms in action as they act; studying different types of techniques to interact with such mechanisms and letting them assign desired interactions to actuate them is beyond the scope of this paper, which we leave for future work.

At the end of the session on Day 1, participants filled out a questionnaire regarding overall user experience, including the difficulty to learn how to use Robiot and whether the process required extra effort than what they had expected (in the Likert scale 1-7). At the end of the Day 2, they filled out another questionnaire to answer *(i)* how difficult it was to assemble the mechanisms; *(ii)* whether the installed mechanisms behave as they expected; *(iii)* perceived usefulness of having such mechanisms to actuate legacy objects. Then we solicited feedback about the entire design and fabrication process and any suggestions for improvement.

[4] https://www.leapmotion.com/

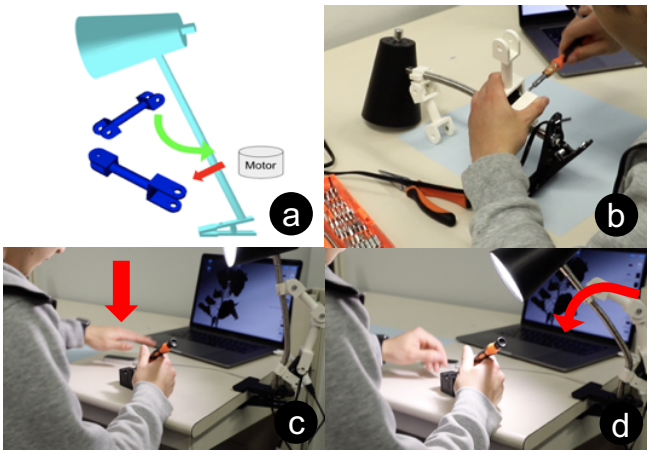

Figure 14. Robiot generates an instruction for assembly (a), then a participant can assemble and install following that instruction (b), to interact with the robotic things using gesture (c) lowering the lamp height to work with optimal brightness when soldering (d)

* 1: Strongly disagree – 7: Strongly agree

1	2	3	4	5	6	7	Mean
Q1: It is easy to learn how to use Robiot							
1	-	1	-	-	3	1	4.8
Q2: The process require less effort than I had expected							
1	-	-	-	1	2	2	5.3
Q3: It is easy to assembly the mechanisms							
-	-	-	1	2	1	2	5.7
Q4: The installed mechanism behave as you expected							
-	-	-	-	1	2	3	6.3
Q5: It is useful to have such mechanisms to actuate legacy objects							
-	-	-	-	1	5	-	5.8

Table 2. Selected statements with survey scores, counts in each cell indicate how many participants rated their scores

Results and Findings

Participants created a total of 14 mechanism designs using Robiot. All but one participant successfully assembled their designs (the only failure case was due to a critical component broken before the second session).

We first analyzed participants' questionnaire responses for the quantitative analysis. Table 2 summarizes the results on each question, gauging easiness and usefulness of the tool and design pipeline. Then we transcribed video recordings to observe key insights from participants' behaviors for qualitative analysis. We transcribed video data based on the context (e.g., taking video, using Robiot desktop system, assembly, etc.), logging participants' spoken responses (e.g., "why does this look [the] same?") and description on their behaviors (e.g., P1 tried to handle 2 DOF at a time). Then we classified this data to identify findings as follows.

Easiness of using the System - The most participants reported that the pipeline is simple and the user interface is easy to use, being able to operate without much background knowledge (Q1-2). Participants commented *"I've never thought that machines can be made through that simple steps"* (P4) and *"I thought designing robotic tools involves heavy measurement, designing and trial and errors. But using Robiot, I simply record the video and it automatically did all the work"*. However, one participant reported an important aspect about instruction *"I won't know I can rotate and zoom if I am using it independently"* (P6), which suggest user interface improvement to better inform possible functionalities, what design options are available when users perform a design task.

Accessible Pipeline and Design Assistance - Participants had no problem assembling and installing the mechanisms, interacting with objects actuated by the mechanisms (Q3-4), reporting *"The part is straightforward, I just need to put the joints together and tighten screws. Then everything works well"* (P1) as well as *"It's only few components with clear instructions"* (P3). Also, all participants were satisfied with generated motions, because robotized objects behaved as they expected from the beginning of the design. However, one participant mentioned *"It works well though not very sensitive"* (P4), raising concerns on the granularity of motion Robiot can generate. We will discuss in more detail later in the paper.

Usefulness of the Tool and Aesthetics - All participant were in favor of the tool as they foresee potentials (Q5) to help them *"customize [my] own things that meet my own needs"* (P3), *"on many objects around me"* (P4). Though, participants also have higher expectations in aesthetics and design to fully utilize the system in their everyday lives, commenting that *"Without proper design, these modern mechanisms will look strange on old-time objects"* (P1) and *"The design part could be previously done by specialists, instead of automatic algorithm"* (P2). We expect addressing concerns around design aspects would expand future use case of Robiot, by involving more users who care aesthetics in creating custom robotic things.

Remaining Challenges - There were a few common challenges among participants. Some participants struggled to understand what 'initial state' and 'final state' meant. In hindsight, we realized such wording was too technical, and perhaps an alternative expression such as 'before/after' would have been more understandable. Also, participants did not react positively to the sliders that can further adjust design parameters and filter a subset of generated mechanisms. Although participants were satisfied when finishing the process, they did comment on a lack of understanding of how things work, such as *"I cannot understand why I choose one mechanism, or how to choose one option"* (P6). In addition to the instructions we provided for assembly, in the design phase, animated previews of each suggested mechanism in action and step by step instructions for users to manipulate a design interface would help them feel more engaged in the design process with less confusion.

DISCUSSION

In this section, we discuss existing issues, limitations, and opportunities for future work.

Future Technical Work

There are several technical details Robiot needs to focus on in the future. In cases where a user accidentally blocks the camera at the beginning or the end of the video, one future direction will be employing computer vision techniques to detect such occlusion and providing a simple UI for the user to select a better, unblocked start/end frame.

As there are usability issues with sliders in user interface, one future direction of user interface will be providing interactive tutorials to help users understand the mechanical effect of adjusting each slider.

Scale of Mechanisms

Robiot suggests mechanisms that best suit user-specified action and generate a 3D printable model by a desktop 3D printer. The scale of mechanisms and mechanical elements (e.g., size of gear teeth and the length of the rack) are dependent on the capability of the printer, with common hardware settings. Investigating possibilities for the system (1) to design larger mechanisms to support furniture scale objects' actuation and (2) to handle granularity of the motion, which is defined by the size of mechanical elements, could be an interesting extension of our work.

Camera Angle to Capture Desired Motion

To best extract series of motion path captured from a video, users need to capture the objects in motion *orthogonal* to the movement paths. Because Robiot currently lets users design mechanisms in one degree of freedom at a time, motion extraction and generating mechanisms are based on 2D, where an orthogonal scene best derives the motion. Prior work has investigated retaining 3D information when converting 2D videos [21, 32], by estimating depth from 2D scenes. Another future direction of Robiot is extending its capability to extract features for motions with depth in 3D from 2D videos, and generate mechanisms in multiple degrees of freedom at a time that addresses 3D motions.

Designing Motion Beyond Given Affordances

Currently, Robiot helps users to create 3D printable actuation mechanisms to perform *expected* action of legacy objects. For example, users are likely to design height adjusting mechanisms for a chair and a rotating mechanism for an old water faucet. Users' choices on actuating mechanisms are decided by the existing affordances of a physical interface, it is hard to imagine a user would change these affordances, such as making a linear switch to attach on a rotating faucet and turning pulling drawer opening in rotational angle. Reprise is an approach that allows users to change the type of required movement to perform actions on physical handheld objects, particularly for people with fine motor impairments [9]. One interesting future direction could be applying this technique to generate mechanisms that enable users to alter the type of motion to perform a fixed physical task, such as rotational to linear motion and vise versa.

Sensing and Designing Custom Interactions

The main contribution of ours is an end-to-end pipeline, enabling designing fabricable mechanisms; sensing and defining

custom interactions, and mapping them to desired actions are beyond the scope of this paper. Nonetheless, there exist commercial kits that welcome average users to install add-on motion sensors to trigger actuating home IoT (e.g., [1, 22]), and it becomes common to perform such tasks using connected devices or voice assistance using home intelligence devices (e.g., [20]). We demonstrated the feasibility of mapping custom gesture interactions using LeapMotion, which opens future opportunity to implement novel interfaces for end users to define custom user interactions to fine-control the motion, given different user requirements. With these interfaces, existing work on novel sensing techniques on everyday objects to detect unique object touch [51] or direct sensing of a human body to detect motion [11] can be applied to enrich user experiences in everyday use of robotic things.

Performing Ungrounded Action

Recent advancements in ubiquitous computing have presented the future vision of moving objects that do not require explicit user intention to perform such actions. For example, Nissan showcased their visionary self driving cars through smart slippers [42] and chairs [41] that self-organize. It is also viable to imagine salt and pepper bottles shaking by themselves, as having them on a soup bowl is a common routine. Envisioning the future with omnipresent robotic things that accommodate people's everyday routines and expected activities as triggers (e.g., sandals coming to you when you come into the door, desktop self-adjusting height as you stand up to refresh your posture) by predefined activity sensing, Robiot sheds lights on a new possibility for end-users to robotize objects at home and office to make their everyday life easier towards functional home/office.

ACKNOWLEDGMENTS

We thank the reviewers for their valuable feedback. We also thank participants for participating in the user study. This work was funded in part by Adobe.

REFERENCES

[1] Gift a lot. 2019. Smart Wifi Electric Curtains Tracks and Motorized Roller Blinds. https://giftalot.com/. (2019). (Accessed on 04/01/2019).

[2] N. S. Altman. 1992. An introduction to kernel and nearest-neighbor nonparametric regression. *American Statistician* (1992). DOI: http://dx.doi.org/10.1080/00031305.1992.10475879

[3] Rahul Arora, Rubaiat Habib Kazi, Tovi Grossman, George Fitzmaurice, and Karan Singh. 2018. SymbiosisSketch. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems - CHI '18*. ACM Press, New York, New York, USA, 1–15. DOI:http://dx.doi.org/10.1145/3173574.3173759

[4] Daniel Ashbrook, Shitao Stan Guo, and Alan Lambie. 2016. Towards augmented fabrication: Combining fabricated and existing objects. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*. ACM, 1510–1518.

[5] M Bacher, S Coros, and B Thomaszewski. 2015. LinkEdit: Interactive Linkage Editing using Symbolic Kinematics. *Acm Transactions on Graphics* (2015). DOI: http://dx.doi.org/10.1145/2766985

[6] Aude Billard, Sylvain Calinon, Ruediger Dillmann, and Stefan Schaal. 2008. Robot programming by demonstration. *Springer handbook of robotics* (2008), 1371–1394.

[7] Mayara Bonani, Raquel Oliveira, Filipa Correia, André Rodrigues, Tiago Guerreiro, and Ana Paiva. 2018. What My Eyes Can't See, A Robot Can Show Me. In *Proceedings of the 20th International ACM SIGACCESS Conference on Computers and Accessibility - ASSETS '18*. ACM Press, New York, New York, USA, 15–27. DOI:http://dx.doi.org/10.1145/3234695.3239330

[8] Xiang 'Anthony' Chen, Stelian Coros, Jennifer Mankoff, and Scott E Hudson. 2015. Encore: 3D printed augmentation of everyday objects with printed-over, affixed and interlocked attachments. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software and Technology*. ACM, 73–82.

[9] Xiang 'Anthony' Chen, Jeeeun Kim, Jennifer Mankoff, Tovi Grossman, Stelian Coros, and Scott E Hudson. 2016. Reprise: A Design Tool for Specifying, Generating, and Customizing 3D Printable Adaptations on Everyday Objects. In *the 29th Annual ACM Symposium on User Interface Software and Technology*. ACM.

[10] Xiang 'Anthony' Chen, Ye Tao, Guanyun Wang, Runchang Kang, Tovi Grossman, Stelian Coros, and Scott E Hudson. 2018. Forte: User-Driven Generative Design. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, 496.

[11] Gabe Cohn, Sidhant Gupta, Tien-Jui Lee, Dan Morris, Joshua R. Smith, Matthew S. Reynolds, Desney S. Tan, and Shwetak N. Patel. 2012. An Ultra-low-power Human Body Motion Sensor Using Static Electric Field Sensing. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing (UbiComp '12)*. ACM, New York, NY, USA, 99–102. DOI: http://dx.doi.org/10.1145/2370216.2370233

[12] Stelian Coros, Bernhard Thomaszewski, Gioacchino Noris, Shinjiro Sueda, Moira Forberg, Robert W. Sumner, Wojciech Matusik, and Bernd Bickel. 2013. Computational design of mechanical characters. *ACM Transactions on Graphics* 32, 4 (jul 2013), 1. DOI: http://dx.doi.org/10.1145/2461912.2461953

[13] Scott Davidoff, Nicolas Villar, Alex S Taylor, and Shahram Izadi. 2011. Mechanical hijacking: how robots can accelerate UbiComp deployments. In *Proceedings of the 13th international conference on Ubiquitous computing*. ACM, 267–270.

[14] Ruta Desai, James McCann, and Stelian Coros. 2018. Assembly-aware Design of Printable Electromechanical Devices. In *The 31st Annual ACM Symposium on User*

Interface Software and Technology - UIST '18. ACM Press, New York, New York, USA, 457–472. DOI: http://dx.doi.org/10.1145/3242587.3242655

[15] dezeen. 2018. Reconfigurable apartment allows residents to transform their living spaces. https://www.dezeen.com/2018/02/15/video-white-arkitekter-dream-home-reconfigurable-apartment-movie/. (2018). (Accessed on 04/01/2019).

[16] Markus Ehrenmann, Oliver Rogalla, Raoul Zöllner, and Rüdiger Dillmann. 2001. Teaching service robots complex tasks: Programming by demonstration for workshop and household environments. In *Proceedings of the 2001 International Conference on Field and Service Robots (FSR)*, Vol. 1. 397–402.

[17] Chaitanya P. Gharpure and Vladimir A. Kulyukin. 2008. Robot-assisted shopping for the blind: issues in spatial cognition and product selection. *Intelligent Service Robotics* 1, 3 (jul 2008), 237–251. DOI: http://dx.doi.org/10.1007/s11370-008-0020-9

[18] Anhong Guo, Jeeeun Kim, Xiang 'Anthony' Chen, Tom Yeh, Scott E Hudson, Jennifer Mankoff, and Jeffrey P Bigham. 2017. Facade: Auto-generating tactile interfaces to appliances. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM, 5826–5838.

[19] Guy Hoffman and Wendy Ju. 2014. Designing Robots With Movement in Mind. *Journal of Human-Robot Interaction* (2014). DOI: http://dx.doi.org/10.5898/jhri.3.1.hoffman

[20] Google Home. 2019. Smart Speaker & Home Assistant - Google Store. https://store.google.com/us/product/google_home?hl=en-US. (2019). (Accessed on 04/04/2019).

[21] Xiaojun Huang, Liangh Wang, Junjun Huang, Dongxiao Li, and Ming Zhang. 2009. A Depth Extraction Method Based on Motion and Geometry for 2D to 3D Conversion. In *2009 Third International Symposium on Intelligent Information Technology Application*, Vol. 3. 294–298. DOI:http://dx.doi.org/10.1109/IITA.2009.481

[22] Philips Hue. 2019. Smart home automation light. https://www2.meethue.com/en-us/smart-home-automation-light. (2019). (Accessed on 04/04/2019).

[23] Jianbo Shi and Tomasi. 2002. Good features to track. DOI:http://dx.doi.org/10.1109/cvpr.1994.323794

[24] Rubaiat Habib Kazi, Tovi Grossman, Hyunmin Cheong, Ali Hashemi, and George Fitzmaurice. 2017. DreamSketch: Early Stage 3D Design Explorations with Sketching and Generative Design. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology - UIST '17*. DOI: http://dx.doi.org/10.1145/3126594.3126662

[25] Han-Jong Kim, Chang Min Kim, and Tek-Jin Nam. 2018. SketchStudio: Experience Prototyping with 2.5-Dimensional Animated Design Scenarios. *Proceedings of the 2018 on Designing Interactive Systems Conference 2018* (2018). DOI: http://dx.doi.org/10.1145/3196709.3196736

[26] Robert Kovacs, Alexandra Ion, Pedro Lopes, Tim Oesterreich, Johannes Filter, Philipp Otto, Tobias Arndt, Nico Ring, Melvin Witte, Anton Synytsia, and others. 2018. Trussformer: 3d printing large kinetic structures. In *The 31st Annual ACM Symposium on User Interface Software and Technology*. ACM, 113–125.

[27] Yuki Koyama, Shinjiro Sueda, Emma Steinhardt, Cal Poly, and I D C Herzliya. 2015. AutoConnect : Computational Design of 3D-Printable Connectors. *ACM Transactions on Graphics* (2015).

[28] Andrey Kurenkov. DeepCrop : Directed Object Segmentation with Deep Learning. In *arXiv*.

[29] TP-Link Laos. 2019. Smart Wi-Fi Light Switch. https://www.tp-link.com/la/products/details/cat-5622_HS200.html. (2019). (Accessed on 04/01/2019).

[30] LEGO. 2019. Homes - Mindstorms LEGO.com. (2019). https://www.lego.com/en-us/mindstorms

[31] TV Lift. 2019. Motorized TV Lift & TV Automation System - Nexus 21. https://www.tvlift.com/. (2019). (Accessed on 04/01/2019).

[32] Chao Liu and Lauren Christopher. 2012. Depth map estimation from motion for 2D to 3D conversion. In *2012 IEEE International Conference on Electro/Information Technology*. 1–4. DOI: http://dx.doi.org/10.1109/EIT.2012.6220749

[33] Bruce D. Lucas and Takeo Kanade. 1981. An Iterative Image Registration Technique with an Application to Stereo Vision. *Robotics* (1981). DOI: http://dx.doi.org/10.1109/HPDC.2004.1323531

[34] Blair MacIntyre, Maribeth Gandy, Steven Dow, and Jay David Bolter. 2005. DART: A Toolkit for Rapid Design Exploration of Augmented Reality Experiences Blair. *ACM Transactions on Graphics* (2005). DOI: http://dx.doi.org/10.1145/1073204.1073288

[35] M. Manti, A. Pratesi, E. Falotico, M. Cianchetti, and C. Laschi. 2016. Soft assistive robot for personal care of elderly people. In *2016 6th IEEE International Conference on Biomedical Robotics and Biomechatronics (BioRob)*. IEEE, 833–838. DOI: http://dx.doi.org/10.1109/BIOROB.2016.7523731

[36] Justin Matejka, Ali Hashemi, Michael Glueck, Tovi Grossman, Erin Bradner, and George Fitzmaurice. 2018. Dream Lens. DOI: http://dx.doi.org/10.1145/3173574.3173943

[37] Vittorio Megaro, Bernhard Thomaszewski, Maurizio Nitti, Otmar Hilliges, Markus Gross, and Stelian Coros. 2015. Interactive design of 3D-printable robotic creatures. *ACM Transactions on Graphics* 34, 6 (oct 2015), 1–9. DOI: http://dx.doi.org/10.1145/2816795.2818137

[38] Stefanie Mueller, Alexander Teibrich, Stefan Neubert, Patrick Baudisch, François Guimbretière, and Robert Kovacs. 2015. Patching Physical Objects. DOI: http://dx.doi.org/10.1145/2807442.2807467

[39] MySmartBlinds. 2019. Make your blinds smart. https://www.mysmartblinds.com/. (2019). (Accessed on 04/01/2019).

[40] Michael Nebeling, Janet Nebeling, Ao Yu, and Rob Rumble. 2018. ProtoAR. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems - CHI '18*. ACM Press, New York, New York, USA, 1–12. DOI: http://dx.doi.org/10.1145/3173574.3173927

[41] BBC News. 2016. Nissan's self-parking robot chairs tidy up offices. https://www.youtube.com/watch?v=FLEgvD7iG-M. (2016). (Accessed on 04/04/2019).

[42] RT News. 2018. Smart slippers: Nissan extends its self-parking technology. https://www.youtube.com/watch?v=BtGqi0C2SSU. (2018). (Accessed on 04/04/2019).

[43] Georgia Peleka, Ioannis Kostavelis, Andreas Kargakos, Dimitrios Giakoumis, Manolis Vasileiadis, Dimitrios Tzovaras, and Evangelos Skartados. 2019. RAMCIP Robot: A Personal Robotic Assistant; Demonstration of a Complete Framework. Springer, Cham, 96–111. DOI: http://dx.doi.org/10.1007/978-3-030-11024-6_7

[44] Huaishu Peng, Jimmy Briggs, Cheng-Yao Wang, Kevin Guo, Joseph Kider, Stefanie Mueller, Patrick Baudisch, and François Guimbretière. 2018. RoMA: Interactive fabrication with augmented reality and a robotic 3D printer. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, 579.

[45] Raf Ramakers, Fraser Anderson, Tovi Grossman, and George Fitzmaurice. 2016. RetroFab. 409–419. DOI: http://dx.doi.org/10.1145/2858036.2858485

[46] Ahunzyanov Rasim and Tropchenko Alex. 1999. Hand Detection Based on Skin Color Segmentation and Classification of Image Local Features. (1999).

[47] Autodesk Research. 2019. Project Dreamcatcher. https://www.autodeskresearch.com/projects/dreamcatcher. (2019). (Accessed on 04/01/2019).

[48] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. 2004. "GrabCut"- Interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics – Proceedings of ACM SIGGRAPH 2004* (2004). DOI:http://dx.doi.org/10.1145/1015706.1015720

[49] Thijs Jan Roumen, Willi Mueller, and Patrick Baudisch. 2018. Grafter: Remixing 3D-printed machines. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, 63.

[50] Iason Sarantopoulos, Yannis Koveos, and Zoe Doulgeri. 2018. Grasping Flat Objects by Exploiting Non-Convexity of the Object and Support Surface. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 1–6. DOI: http://dx.doi.org/10.1109/icra.2018.8461192

[51] Munehiko Sato, Ivan Poupyrev, and Chris Harrison. 2012. Touché: Enhancing Touch Interaction on Humans, Screens, Liquids, and Everyday Objects. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12)*. ACM, New York, NY, USA, 483–492. DOI: http://dx.doi.org/10.1145/2207676.2207743

[52] Eric Schweikardt and Mark D. Gross. 2008. Learning about Complexity with Modular Robots. In *2008 Second IEEE International Conference on Digital Game and Intelligent Toy Enhanced Learning*. IEEE, 116–123. DOI:http://dx.doi.org/10.1109/DIGITEL.2008.49

[53] Evan Strasnick, Jackie Yang, Kesler Tanner, Alex Olwal, and Sean Follmer. 2017. shiftIO: Reconfigurable Tactile Elements for Dynamic Affordances and Mobile Interaction. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 5075–5086. DOI: http://dx.doi.org/10.1145/3025453.3025988

[54] Xiang Zhi Tan and Aaron Steinfeld. 2017. Using Robot Manipulation to Assist Navigation by People Who Are Blind or Low Vision. In *Proceedings of the Companion of the 2017 ACM/IEEE International Conference on Human-Robot Interaction - HRI '17*. ACM Press, New York, New York, USA, 379–380. DOI: http://dx.doi.org/10.1145/3029798.3034808

[55] Markus Vincze, Tobias Koertner, Astrid Weiss, Walter Wohlkinger, Peter Einramhof, Paul Panek, Konstantinos Papoutsakis, Antonis Argyros, David Fischinger, Stefan Hofmann, and Peter Mayer. 2014. Hobbit, a care robot supporting independent living at home: First prototype and lessons learned. *Robotics and Autonomous Systems* 75, PA (jan 2014), 60–78. DOI: http://dx.doi.org/10.1016/j.robot.2014.09.029

[56] Luke Vink, Viirj Kan, Ken Nakagaki, Daniel Leithinger, Sean Follmer, Philipp Schoessler, Amit Zoran, and Hiroshi Ishii. 2015. TRANSFORM As Adaptive and Dynamic Furniture. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '15)*. ACM, New York, NY, USA, 183–183. DOI: http://dx.doi.org/10.1145/2702613.2732494

[57] Philipp Wacker, Simon Voelker, Adrian Wagner, and Jan Borchers. 2018. Physical Guides. In *Proceedings of the Symposium on Spatial User Interaction - SUI '18*. ACM Press, New York, New York, USA, 25–35. DOI: http://dx.doi.org/10.1145/3267782.3267788

[58] Weitian Wang, Rui Li, Yi Chen, Z. Max Diekel, and Yunyi Jia. 2018. Facilitating Human-Robot Collaborative Tasks by Teaching-Learning-Collaboration From Human Demonstrations. (2018). DOI: http://dx.doi.org/10.1109/TASE.2018.2840345

[59] Victor Zykov, Andrew Chan, and Hod Lipson. 2007. Molecubes: An open-source modular robotics kit. In *IROS-2007 Self-Reconfigurable Robotics Workshop*. 3–6.