Interactive Virtual Graphics with Physical Objects

ANONYMOUS AUTHOR(S)
SUBMISSION ID: 2691

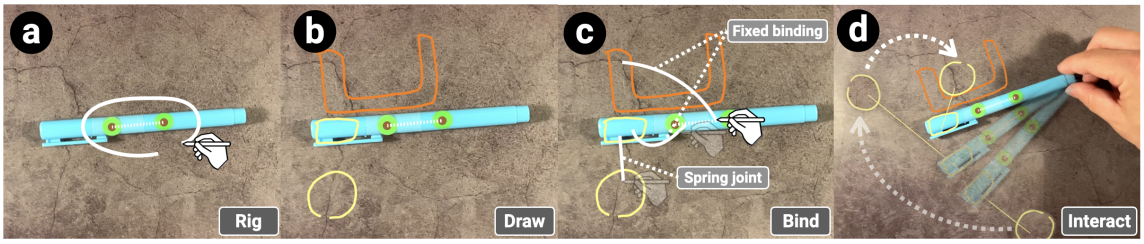


Fig. 1. *Interactive virtual graphics with everyday physical objects.* Under a touch tablet (e.g., iPad in this example), the user can rig a real object under the camera view (a), draw virtual graphics (b), annotate their relationship (c), and then animate the virtual graphics by moving the physical object (d). The annotation can be performed across multiple camera views analogous to keyframe-based animation, with effects ranging from handle-based deformation to physical simulation.

Everyday objects are commonly deployed for a variety of real-world applications, such as storytelling, components for game play, or props for concept explanation. Recent advances in tangible interface and augmented reality have shown promise in helping users authoring and performing combinations of real objects and virtual effects, such as body-driven graphics [44] and object-driven concept visualization [48]. However, it remains challenging to create expressive interactive graphical effects with physical objects. We present a system that enables the real-time creation of rich interactive augmented-reality effects with ordinary physical objects. The interface of our system allows users to map between real objects and virtual graphics so that tangible manipulations of the former can drive the movement of the latter, including handle-based deformation, key-frame animation, and trigger-based physics and interactions. We evaluate our system with a variety of applications in interactive art, tangible gaming, and concept explanation.

CCS Concepts: • **Human-centered computing → Human computer interaction (HCI)**.

Additional Key Words and Phrases: augmented reality; real-time authoring; sketching interfaces; tangible interaction;

ACM Reference Format:
Anonymous Author(s). 2022. Interactive Virtual Graphics with Physical Objects. In *CHI '22: ACM Conference on Human Factors in Computing Systems, April 30 – May 6, 2022, New Orleans, USA.* ACM, New York, NY, USA, 19 pages. https://doi.org/10.1145/1122445.1234567

1 INTRODUCTION

There is a rich history in HCI of using physical objects and spaces to interact with the digital world [17, 21, 54]. With the technological advancements and better understanding of the benefits of tangible and immersive interactions, researchers have explored such interactions to enhance learning, education [11, 43, 52], collaboration [6], design [50], storytelling

Manuscript submitted to ACM

[4] and artistic expression [12]. By leveraging our innate experiences with the physical world, tangible interactions enhances our immersion and engagement [21]. Despite the benefits and potential, crafting interactive experiences with everyday physical objects in improvisational ways is still challenging [36]. Existing systems are domain specific and often rely on pre-programmed behaviors without easy customization interfaces for novice users. Due to the lack of flexible and expressive improvisational tools, users often rely on post-production to augment videos with concept visualizations and graphical effects.

With the recent advancements in head-mounted display (HMD) and consumer grade augmented reality (AR) devices (such as smart phones and tablets), researchers have explored creating interactive AR experiences in real-time with sketching, gestures, and contextual (*e.g.*, 3D planes, geometry) information [3, 8, 23, 31, 32, 44, 48]. For instance, RealitySketch [48] facilitates the real-time creation of concept visualizations that are driven by physical objects. In this paper, our goal is to empower novice users (*e.g.*, , teachers, artists, amateurs with no programming expertise) to craft interactive AR experiences with arbitrary objects that goes beyond concept visualizations. Examples include crafting interactive game experiences, playful interaction with hand-drawn sketches, or demonstrating educational concepts in whimsical ways. However, there are both technical and interaction challenges to achieve this goal. For instance, to play a Pac-Man game with an articulated physical object, like scissor Figures 7 and 10, we need to reliably track the physical object (scissor), identify states (open vs. close), and track continuous parameters (e.g., the angle between the two metal blades). Further, from an interaction standpoint, one needs to specify the states (open or close), parameters (angle), and the corresponding actions and events to the graphical objects for these states and parameters, such as how the physical scissor angle will scale the virtual object being eaten.

In the paper, we present an end-to-end interactive system that enables the real-time creation of rich interactive graphical effects with everyday physical objects (e.g., scissors, pens, food items, linkage systems, or even human hands). With our system, users can sketch virtual graphics, map their spatial and temporal properties to physical objects, and start to manipulate the physical objects to drive the virtual graphical objects. Some example effects include virtual light particles reflected by a physical barrier (Figure 11) for explaining physics, a virtual parachute animated by a physical banana (Figure 17) for animated art, and a virtual ball game played with physical rods (Figure 14).

To facilitate the easy creation of virtual graphics that align with physical objects [37], we have designed and implemented a direct drawing interface that leverages the camera views and touch interfaces of AR enabled consumer devices (phone or tablet) which are accessible to billions of users. As shown in Figure 1, users can select and rig physical objects, draw virtual graphics, link them, assign physical properties, and then play. They can also rapidly iterate on their interactions by playing some and then adding, removing, and changing the existing pieces. Our authoring tool consists of two major (iterative) steps: rigging and mapping. First, in the rigging step, users first select a physical object, and physically attach colored dot stickers to track and specify points of interest. In the UI, these colored dot stickers are used to parameterize the desired variables (e.g., lines, angles, length) and states of the object. Second, in the mapping step, the user maps the tracked object parameters to graphical effects, such as handle-based deformation [20] as well as trigger-based interactions [24]. Once mapped, users can manipulate the physical object and see the corresponding virtual effects in real-time.

Essentially, our proposed solution provides a rigging abstraction with several key benefits. First, by simplifying the tracking problem, it enables the use of any arbitrary objects to interact with virtual graphics in improvisational ways. Second, it provides a general and flexible method for parameterization, enabling the user to map desired properties and parameters from complex and articulated physical objects (*e.g.*, length of a limb, angle between blades of a scissor).

Overall we aim to propose a general, simple, and accessible solution to create interactive virtual experiences with arbitrary physical objects.

We demonstrate applications of our tool in interactive art, tangible gaming, and concept explanation. In summary, the contributions of this work include:

- A flexible rigging mechanism for a wide range of everyday objects (*e.g.*, soft and rigid objects, simple to complex linkages, 3D objects, deformable objects such as ribbon and mesh) without having to rely on recognizable objects or recognition accuracy of computer vision algorithms;
- An end-to-end prototype that achieves novel functionalities and capabilities, including a mapping interface between physical variables and virtual effects that go beyond concept visualization;
- Results across applications in concept explanation, interactive art, and tangible gaming.

2 RELATED WORK

There is a long history of research in HCI about interacting with virtual environments via physical hand gestures [39, 49] as well as driving virtual performances with facial expressions [35] or body/hand gestures/postures [3, 32, 38, 44]. For example, ChalkTalk [38] and MagicalHands [3] can match interactive mid-air sketches and gestures with pre-built libraries to produce dynamic animated effects in real-time, while PoseTween [32] and Saquib et al. [44] can map body postures with drawn graphical effects for interactive AR performance. In addition to human bodies, physical objects have also been explored to affect virtual graphical effects across a variety of applications, including information/instruction/concept visualization [5, 19, 33, 48, 57], and physical proxies for manipulation [17, 58] or design [53]. In these works, the physical objects play the main role while the virtual graphics are augmentations. Our project instead focuses on the virtual effects and uses physical objects as tangible manipulators, as inspired by prior puppetry techniques for videos [4], objects [16], and articulated figures [15]. However, these prior works limit mappings between inputs and outputs such as object-to-object [16] or morphology-to-morphology [15]. We aim to continue this line of research with expanded scope of effects and flexibility of mapping and manipulation, in which everyday objects can be directly deployed to drive a variety of virtual graphical effects. In addition, inspired by static visual blends [10, 41], we would like our system to be able to integrate virtual effects with physical objects.

Our target applications are inspired by prior works in creating dynamic graphical effects that are responsive to various input mechanisms, such as direct manipulation for primary deformation [1, 20, 47] or secondary motion [55] to achieve dynamic animation effects. For example, optimization-based shape deformation can be combined with direct hand manipulation [20] or indirect video transfer [47], and the primary motions of virtual graphics can automatically drive detailed secondary motions [55] to reduce the need for manual authoring. Beyond animation, responsible graphics can also be an effective means to convey mathematics and physics concepts [7, 29, 45, 46, 51]. In general, the graphical effects should be customizable, with a potential wide variety of means such as sketches [30, 59], triggers [24], rigs/parameters [22, 25], and presets/templates [26, 56]. Our applications are also inspired by prior works in AR and tangible interfaces, such as education [14, 23, 40], visualization [9, 42, 50], and content creation [2, 13, 27, 28]. We continue this line of work by manipulating real-world objects through mobile AR for animating sketched graphical effects.

3 SYSTEM AND INTERFACE

This section introduces the design of our system and user interface which allows users to map between real objects and virtual graphics so that the tangible manipulations of the former can drive the movement of the latter.

3.1 System design

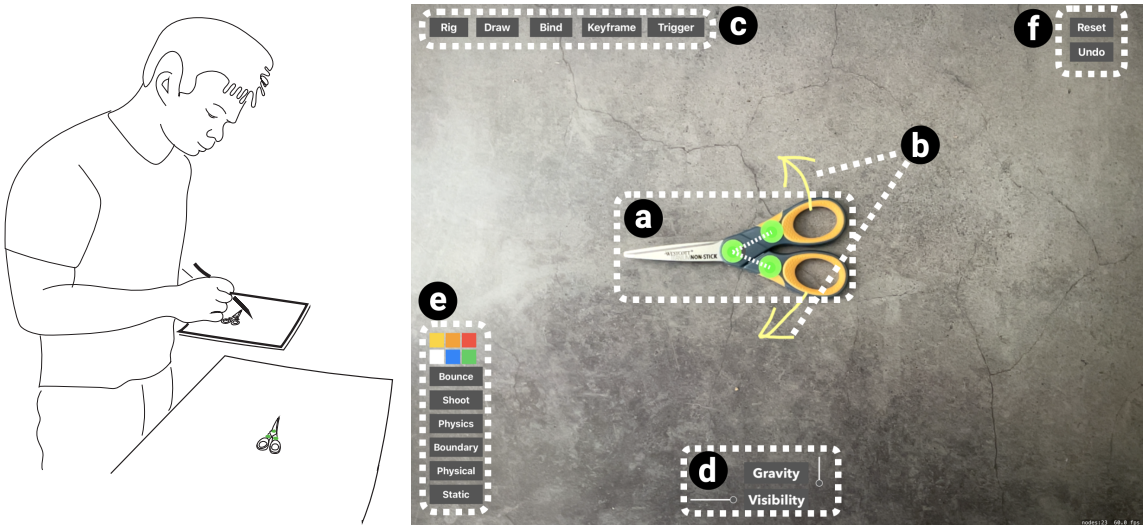
In order to take the spatial parameters of the physical objects as input to drive the virtual performance, the design of the user interface aims to solve the following questions: *(i)* how to designate and track different movable parts of a physical object to extract the relevant spatial information and *(ii)* how to map the tracked physical variables to the virtual graphics to achieve the intended interactive effects.

Based on the question above, the workflow of our system can be decomposed as follows:

Rigging and tracking real objects which may be rigid or have multiple mechanically movable rigid parts.
Drawing virtual graphics to be deformed, animated, or physics-simulated.
Mapping between real objects and virtual graphics so the tangible manipulations of the former can drive the movement of the latter,

To support the aforementioned goals and workflows, we have surveyed relevant techniques in computer graphics and human computer interaction, and found inspirations from deformation [20], simulation [22], keyframe animation, and trigger-based animation [24, 44]. Specifically, keyframe-based animation and deformation provides a familiar mechanism for users to specify the driving parameters of different stages via design-by-demonstration. To further leverage that, we also developed a trigger-based binding incorporating physics and other interactions. To facilitate expressiveness, we also add functionalities of directly adjusting the environment parameters such as global visibility and gravity, which are useful for our intended applications.

3.2 Basic setup



(i) System setup (ii) User interface

Fig. 2. *An overview of our system setup (i) and user interface (ii).* Our user interface consists of the physical object augmented by color dots (a), virtual graphics drawn by users (b), buttons to switch between different authoring and interaction stages (c), environment parameters to adjust (d), buttons for changing stroke colors and specific functions (e) and buttons for reset and undo (f).

The basic set-up consists of a touch tablet, target objects and colors stickers (see Figure 2i). Under the tablet (*e.g.*, an iPad in this example), the user can draw virtual graphics over the camera view of a physical object and create mappings between the two using a pen or bare finger. The physical objects are augmented by a set of color dots for tracking and rigging purpose as multiple mechanically movable parts of an object can be tracked independently in our system. Therefore, the object is not strictly constraint to the 2D plane and the mapping will still be valid as long as the color dots are visible.

Figure 2ii shows the interface of our system, which consists of a canvas for camera frames, and buttons for users to switch between different stages to interact with the system. Specifically, Figure 2iia is the physical object (a pair of scissor in the example) augmented by a set of green color dots and Figure 2iib is the virtual graphics drawn by a user. Figure 2iic consists of several buttons that enable users to switch between different interaction stages and Figure 2iid is the environment parameters that users could adjust. The user could also use the buttons in Figure 2iie to achieve specific goals of different applications which will discussed in the following.

3.3 Rigging physical objects

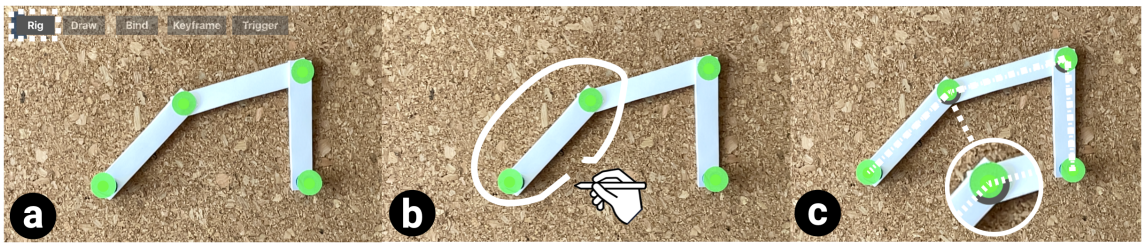


Fig. 3. *Rigging a physical object.* A user could start to rig the physical object in the *Rig* mode (a) by circling around the target color markers (b). If a marker is shared by multiple links, it will be detected as a joint showing the angle between the shared links (c).

In order to drive the animation of the virtual graphics with the movement of the real objects, the spatial information of the objects needs to be captured and extracted by the system as input parameters that affect the output performance.

In order to achieve the intended interactive effects (concept explanation, interactive art and tangible gaming), relevant spatial information of the physical object needs to be extracted and tracked. Under the category of everyday objects, we aim to capture and track three major types of spatial information of the objects:

Spatial configuration consisting of the position and orientation of the object.

Articulated motion of multiple components that connect with each other. While there are plenty of everyday objects that have multiple mechanically movable parts (*e.g.*, a pair of scissors has two movable parts, each of which contains a blade for the cutting), it is vital to track the relative position of one part to another. The physical point where multiple parts connect is called a *mechanical joint*. Since the movement of our targets objects lies in a 2D plane, the mechanical joints are limited to either a rotary joint (the object moves around the point) or a linear joint (the object moves in a sliding motion). Specifically, we need to capture the angle (of a rotary joint) or the sliding distance (of a linear joint).

Customized boundary of the objects which can be aligned with the real shape of the object or virtual by user specification. In order to immerse the object into the target application, we need to capture the boundary

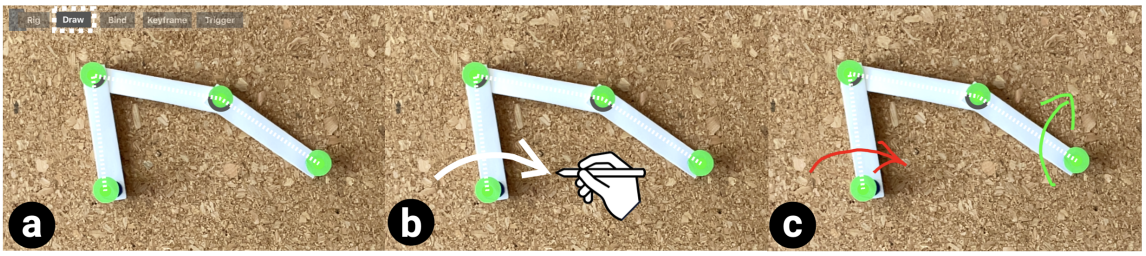


Fig. 4. *Drawing virtual graphics.* A user could draw virtual graphics over physical object such as arrows for annotating the moving direction.

specified by the user that binds with the rigid body of the object. The boundary can be specified by the user as described in Section 3.4 and the binding with the real object can be created as described in Section 3.5.1.

Based on the above requirements of the system, we adopted a vision-based tracking system based on color markers. The first step is to augment the real object by sticking color markers onto it. The augmentation follows the following rule: two markers represent a single rigid part (link) and each joint will be represented by a single marker shared by two links. With this method, it is effortless and low-cost to rig any rigid object that moves mechanically. The user specifies each link by circling around the two corresponding markers in the camera view under the *rig* mode (Figure 3). And if one marker is share by multiple links, this marker will be detected as a joint.

3.4 Drawing virtual graphics

After the user established the tracking of the target objects, the next step is to draw some virtual graphics whose role will be assigned as part of the final application in later steps. Users have freedom of drawing the graphics based on the application they would like to create. By clicking the *draw* button (Figure 4), the user enters the drawing mode, in which the user could sketch strokes with fingers or a pen.

3.5 Mapping objects and graphics

With the tracked real physical objects and the drawn virtual graphics, the next vital step is to create the mapping between the physical objects and the virtual graphics so that the tangible manipulation of the former could drive the movement of the latter. Our system provides different mapping modes for the users to fit different applications. In general, the virtual graphics play two different roles in different applications:

Annotation where the graphics become part of the object showing the originally invisible content or information of the object (*e.g.*, annotating a linkage system for educational purpose as shown in Figure 5).

Integration where the object is integrated into the virtual graphics where the object facilitate the application through tangible manipulation (*e.g.*, a mobile pendulum system using a pen as shown in Figure 6).

Based on the two different roles of the graphics, we support three types of mapping method: direct mapping, indirect mapping with keyframes and indirect mapping with triggers, while these methods can be combined together to create various examples.

3.5.1 Direct mapping. By clicking the *Bind* button, the user enters the direct mapping mode. Intuitively, under direct mapping, the virtual graphics is expected to follow the part of the physical object in a relatively fixed position. For

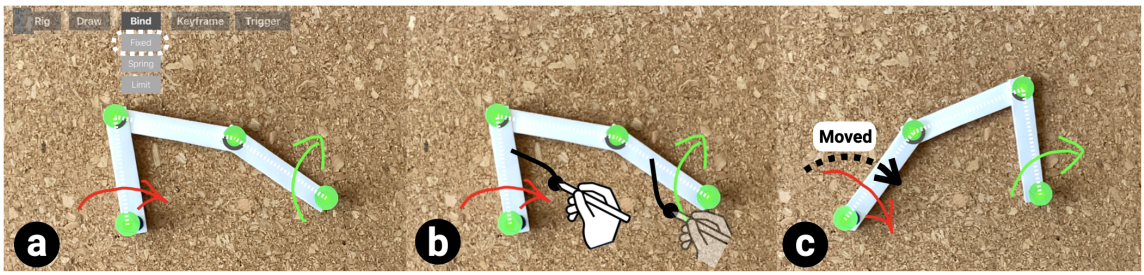


Fig. 5. *Fixed binding.* A user could bind the virtual graphics and the physical object by connecting them together (b). When the physical object moves, the graphics will follow accordingly (c).

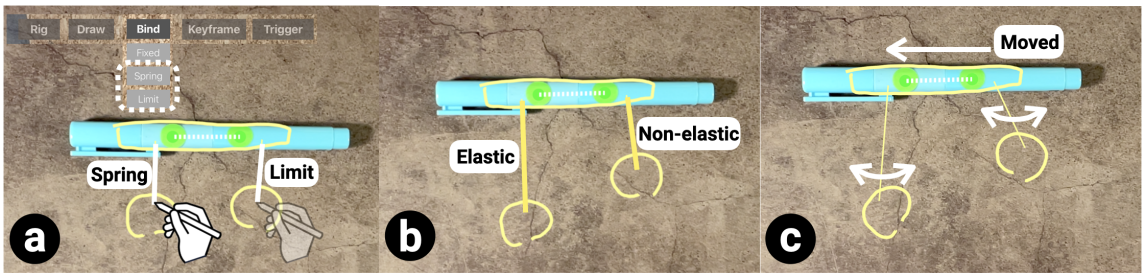


Fig. 6. *Direct mapping: spring and limit joints.* A user could specify spring and limit joints between two graphic elements, where the spring joint is elastic with adjustable frequency and damping and the limit joint is non-elastic with a maximal length.

example, a user could draw annotation over a linkage system to visualize how different links move while the motion is being transferred (Figure 5). In this case, the annotation (*e.g.*, an arrow indicating the rotating direction of the links) needs to move along with object with fixed relative position and orientation. This type of direct mapping is call **fixed binding**. Other than fixed binding, users may need to create some mechanical connection between the virtual graphics and the physical objects. For example, if a user wants to create a mobile pendulum system using a pen, the virtual ball and the physical object need to be mechanically connected by a virtual spring or string (Figure 6). We call such a connection between a physical link and a virtual graphics a *joint*. Our system provides three types of joints for users to directly bind the virtual graphics with the physical object: *fixed joint*, *spring joint* (elastic) and *limit joint* (non-elastic) . The fixed binding in the previous paragraph adopts the *fixed joint* when creating the binding. To create a *joint*, a user can draw a stroke to connect the target graphics with the rigged link or another set of graphics (as shown in Figures 5 and 6). Noted that for *spring* and *limit joint*, the connection have to be specified between two graphic elements as the rigged link does not provide the spatial information of *customized boundary*. The damping and frequency of *spring* and the maximal length of the *limit joint* can adjusted afterwards.

3.5.2 Indirect mapping with key-frames. Other than direct mapping, customized animation or transformation may be required in specific applications. For example, a user may expect a PacMan example using a pair of scissors such that when the pair of scissors closes, the virtual face will shrink and disappear to indicate it has been eaten (Figure 7). The goal here for the user is to specify several keyframes via design-by-demonstration to specify the controlling parameters and then the virtual graphics will be animated or deformed based on the current value of the controlling parameters.

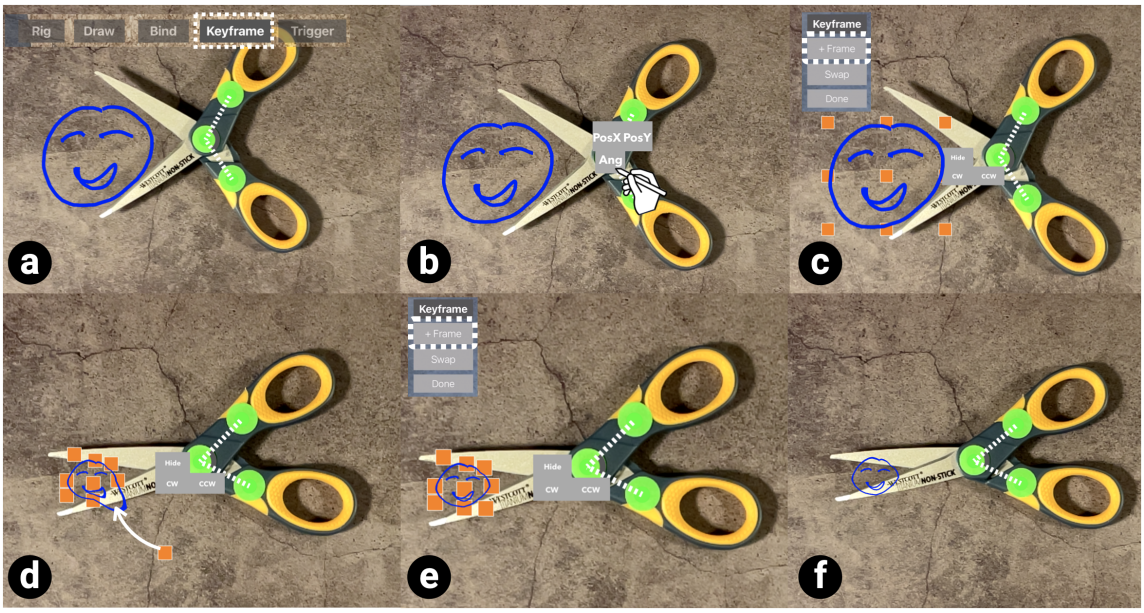


Fig. 7. An example of using keyframes to achieve an effect of using a pair of scissors to eat a smiling face as Pac-man. The user can press the *Keyframe* button (a) and select the target link and the target input parameter (b), and then specify different keyframes for different input parameters and using the control points (c-e). The smiling face will shrink when the pair of scissors closes (f).

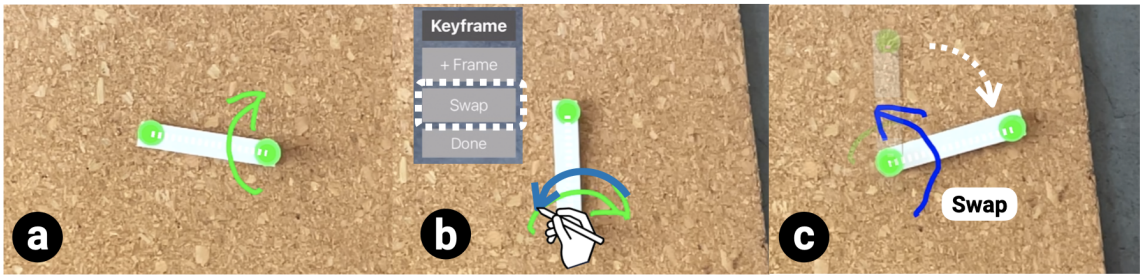


Fig. 8. A swapping action can be specified instead of linear interpolation during the keyframes. A user could click on the *Swap* button and draw the graphic element to replace at the specific configuration (b).

Our system supports the two types of output effects: *deformation* and *animation*, using control-point based transformation specified at key-frames. By clicking the *Keyframe* button, the user enters the keyframing mode. The first step is select the driving input parameters by selecting one of the rigged links. Recall that each link consists of two key points (markers) and the point may become a joint if it is shared by multiple links. Therefore, if a user wants to select the angle of a joint as the input parameters, the joint is also part of the selected link. The second step is to select the target graphics by drawing a stroke circling around the target elements. This action will group all the graphic elements in the circle to be transformed together.

After that, nine control points will be visualized over the target graphic element. Our system utilizes a design-by-demonstration method to specify each keyframe. To do so, the user should set the object to the desired configuration

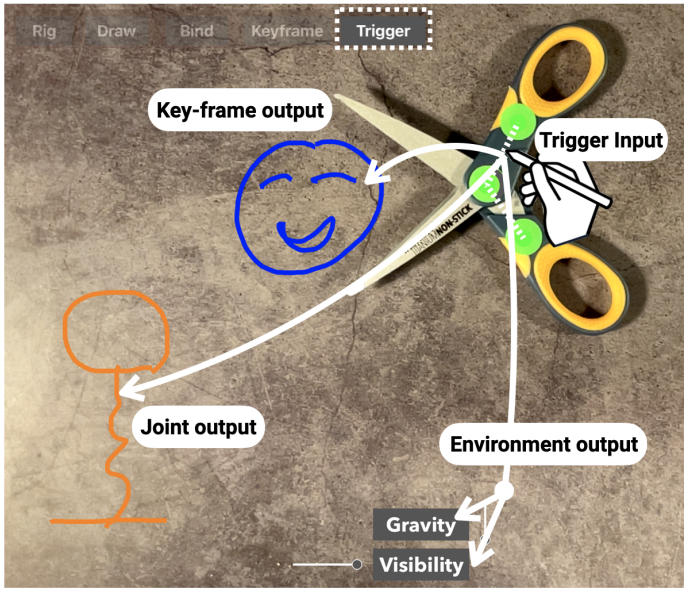


Fig. 9. *Indirect mapping with triggers.* A user could connect the driving link with the target triggering output including environment output, key-frame output and joint output.

(angle or position) and configure the graphic element by dragging the control points. After multiple key-frames being specified, our system interpolate the position of each control point based on current input value to perform deformation and animation. The deformation and animation is implemented with a grid-based deformation method. Our system provides several shortcuts for hiding, rotation, etc. (see Figure 7e) to eliminate the tedious manipulation on each control point. Our system also supports swapping between graphics instead of interpolating a transformation. The user could specify a fresh keyframe to swap with the existing keyframes at the desired configuration by clicking the *Swap* button (see Figure 8). Users could specify multiple key-frame mappings one-by-one to create expressive interactions.

3.5.3 Indirect mapping with triggers. Another level of the indirect mapping is using triggers. While the keyframing only works for graphic elements that could deform or animate using control points, a wider range of output effect can be achieved using trigger-based mapping.

The trigger-based mapping in our system supports four types of output parameters: visibility and global forces from the environment, and the physics joint and keyframing of the graphic elements (Figure 9). Users could specify different triggers to change the visibility (*e.g.*, hide or show graphics) or global force (*e.g.*, wind or gravity), which are parts of the environment parameters. Besides the environment parameters, output parameters of the graphic elements can also be affected based on the triggers. Such parameters includes the keyframing created in the previous steps and, the modification of the physics joint connected to the graphic elements.

For the input, similar to the mapping with keyframes, the user could specify the range of the triggering value of the target input link by demonstration. Differently, our system supports three triggering type: linear, step and pulse (Figure 10a). The user needs to specify two data points for each target input parameter for all the triggering types (Figure 10b-c). The output is a unit-less value indicating how much the output is triggered. For the linear triggering, the

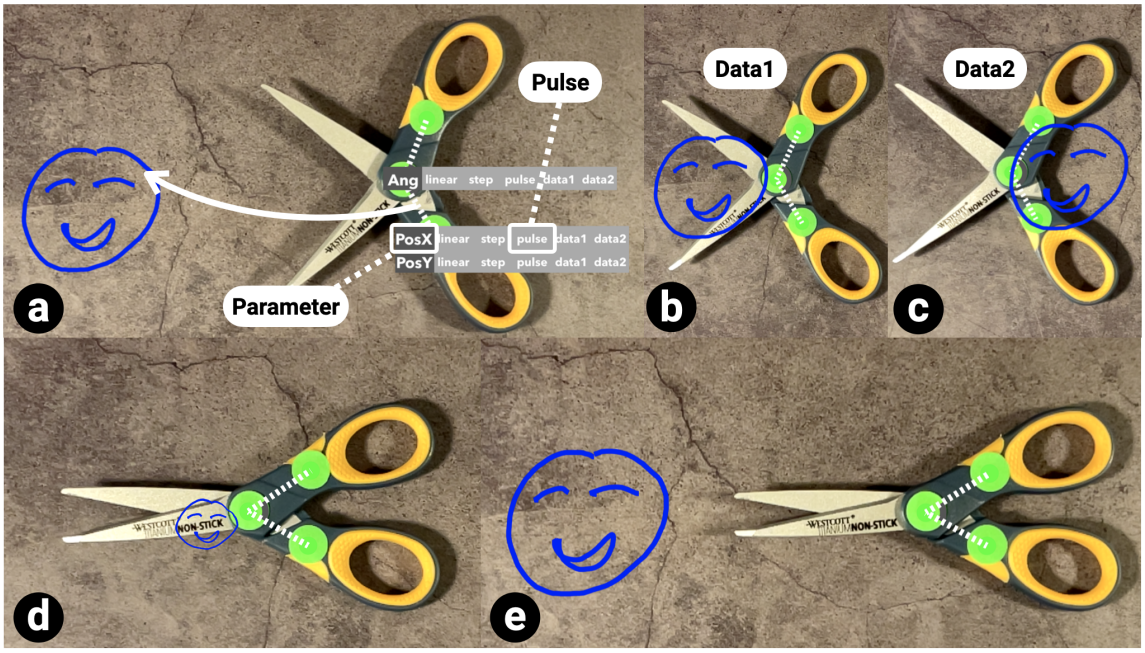


Fig. 10. *A walkthrough of adding trigger to the scissors-as-Pacman example.* The user could specify the target input parameter (position X of the lower link) and the target triggering type (pulse) to trigger the keyframe animation (a) and then specify the two sets of data for the triggering (b, c). The final result shows that the keyframe will be triggered only if the pair of scissors is close to the smiling face (d, e).

output is interpolated based on the input value comparing to the values specified (ranged from 0-1). For the step trigger, the first value specified the user would be the switching point and the second defines on which side will there be a positive output (=1). For the pulse trigger, the output will be 1 between the two values specified. Noted that the user could create multiple layers of the triggers by blending different triggers together. For example, it is possible to create a logic that "when the pair of scissors (as Pac-man) *approaches to* (pulse trigger of the position X and Y) the virtual face and close, it will eat the smiling face (trigger the keyframing deformation)" as shown in Figure 10. However, each of the output parameters may not support all the triggering types, which will be discussed below.

Visibility and environment force. These are the parameters that affect most of the element in the scene globally. These two parameters support all the triggering type. By adjusting the slider shown in Figure 2iid, the user could specify different state of the two output parameters.

Physics joint. The user could specify the triggering output to create a new physics joint or remove an existing joint. For example, a user could remove the joint between the balloon and the holder just like cutting the string based on the scissors' location and configuration. There is no phase in-between for the physics joint and therefore it only supports the step and pulse triggers.

Keyframing. The key-frame mapping created in previous step can also be one of the output parameters in the triggering mapping. Similar to the physics joint, there are only two states of the keyframing here — activated or not, and therefore it only supports the step and pulse triggers.

3.6 Other features

Other than the basic features above, our system also support some other features to provide a better experience of creating customized applications. For example, the user could modify the global visibility and force at anytime (Figure 2iie), or activate or deactivate the physical boundary of the scene for different scenarios by toggling the *Boundary* button (Figure 2iie). For a physics experiment example — how the light will be reflected on different surface, user could activate the ball shooting mode by toggling the *Shoot* button to mimic the photon and or the user could set the global force to zero to prevent the gravity effect. The user could also draw *dynamic* (*e.g.*, ball in a pong game) or *static* (obstacles) objects by toggling the buttons.

4 IMPLEMENTATION

We leverage the 2D general-purpose framework (SpriteKit) to drive the performance of virtual graphics using physical objects.

Tracking. Currently we use a color-based tracking algorithm implemented in OpenCV. We used low-cost green markers purchased from Amazon as our default markers which are solid and distinct. The algorithm track the 2D position of the markers using the embedded color-based masking function and overlay it onto the camera frames. The tracking is fast enough for our applications (*e.g.*, 60 FPS with iPad Pro 12.9 inch).

Drawing. Each user-drawn stroke is represented as an *SKShapeNode* in SpriteKit and considered a single graphic element. While the properties of the graphic element can be specified in later steps (*e.g.*, physical property), the user is able to create such properties at need. For example, the user could directly create obstacles which have physical property to collide with other elements or balls falling from drawing point in a gaming application by turning the *Physical* mode on (see Figure 2iie).

5 RESULTS

In this section we present results created by multiple users across different application domains.

5.1 Educational demonstration

In classroom education, physical props are widely used to help the students understand the basic concept of the knowledge because it is easier for student to create connections between the knowledge and the physical phenomena [14, 18, 34, 48]. However, it could be insufficient to explain the concept with only the physical props. For example, to do a lesson about mirrors and optics, a complex set-up is required including laser lights and mirrors of different shapes (*e.g.*, concave and convex). It is extremely challenging to show the motion of a mechanical system (*e.g.*, linkage systems) with physical annotations following the corresponding parts; and the teacher does not have the ability to stylize the environment parameters, to visualize how a pendulum system would work on the moon (low gravity). Our system leverages the tangibility and expressive of using physical props and empowers users with the ability to map the properties of real objects to drive the performance of virtual graphics.

Physics is the first category that our system supports which can expand the possibilities of using physical props. As shown in Figure 11, a teacher could use an everyday physical object (a ruler in the example) to mimic the motion of a mirror. In this example, the laser light is simulated as a stream of photons and different shape of the surface can be drawn by the teacher to visualize how the light is reflected on different shapes of surface. By drawing different shapes (flat, convex and concave), the user could bind the graphics with the target object using direct mapping and assign physics property to the graphics. Then the physical object can be moved around (so are the graphics) to simulate the light shot from different angles.

Besides providing an easy and customizable method of creating virtual objects for physics education, our tool also offers the ability to stylized the physics, *e.g.*, modifying the gravity in the environment. As shown in Figure 12, by drawing a circle as a ball and bind it with a marker using a spring joint, it becomes a mobile pendulum system and the user could move the marker to simulate the behavior of it in 2D plane. By adjusting the gravity (force applied to all the physical object in the scene), the user could simulate how the pendulum system works in different environment, *e.g.*, upside down world or with a strong side wind. With this method, the audience could easily get a sense of how pendulum system behaves differently in different environment.

Other than physics education, tutorial example using customized annotation is the second category that our system focuses on. Figure 13 illustrates a visualization of the force applied onto the target link of a four-bar-linkage system. The four-bar-linkage system consists of an input link and an output link. The user could annotate the link applied by the input link (in red arrow) and the force applied onto the output link (in green arrow), where the size of the annotation corresponds to the amount of the force. Seeing the size of the arrow would be straightforward for the audience to find the configuration where the target link is applied the largest force.

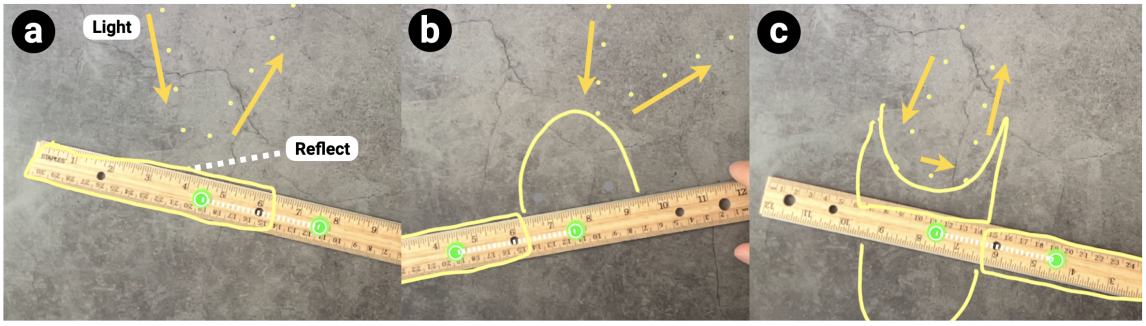


Fig. 11. *Using a ruler for optics example.* A flat (a), convex (b) and concave (c) virtual mirror can bind with the ruler to visualize how light reflects on different surface shapes.

5.2 Gaming with tangible input

Our system can also provide gaming experience using the physical objects as tangible input. We leverage the capability of our system to blend the interaction between real physical objects and the virtual graphics, a variety of applications can be implemented by the system on the fly. Figure 14 illustrates a series of gaming examples using markers. To play a ball bouncing game, the user could start with drawing a custom shape mapped with the pen as the paddle and draw a ball in the middle of the scene to bounce (Figure 14a). To make the game more interesting, the user could use the same set-up as in Figure 12 and draw a cup on the pen. By binding it with the pen, the user could play a cup-the-ball game

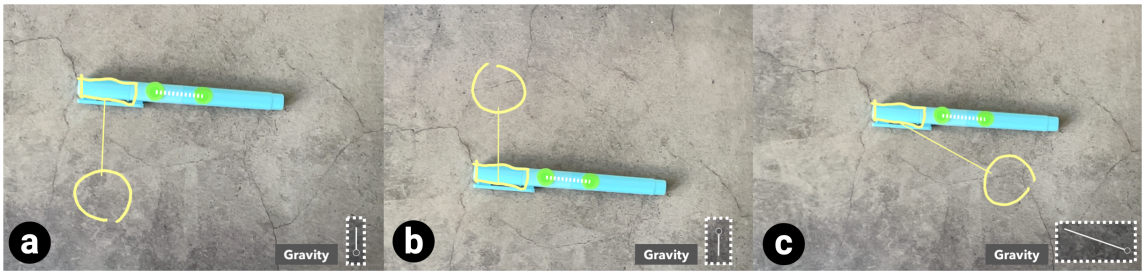


Fig. 12. *A mobile pendulum system using marker is simulated under different environmental force conditions:* normal gravity (a), anti-gravity (b), and normal gravity with a strong side wind (c).

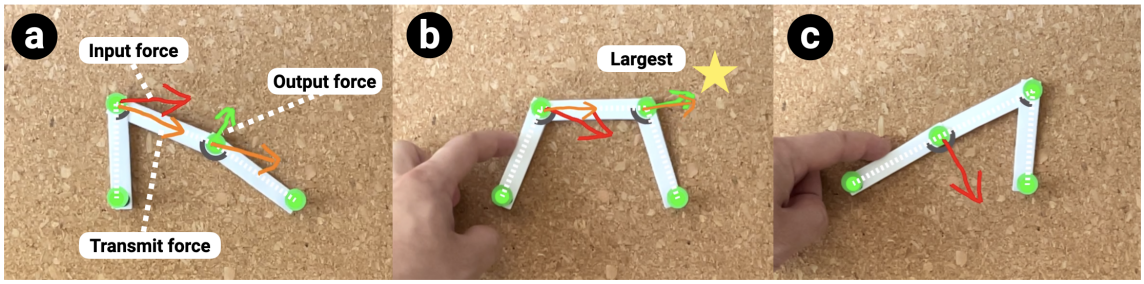


Fig. 13. *Forces applied on each link of the four-bar-linkage system are visualized (a).* The largest output force is can be found based on the size of the arrow (b) and the output force decreased to zero in certain configuration (c).

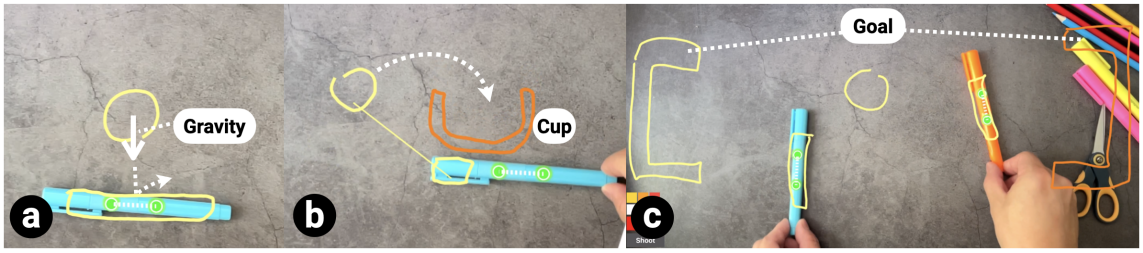


Fig. 14. *Different gaming applications using a marker.* Ball bouncing game (a), cup-the-ball (b) and multiplayer pong (c).

(Figure 14b). The user could move the pen around trying to swing the ball into the cup. It is also possible for the user to invite a friend to play a 2D pong game. The user could rig two different pens and draw the corresponding shapes over the pen as in the previous steps. Now with the ball floating in the scene, the two players can move their pens to play the pong game (Figure 14c). They can also draw a goal on each side to make the game more immersive.

Besides the applications mentioned above, the user could go beyond just manipulating the object on a black background below the AR device by panning the camera up and appear in the scene with the objects and the created virtual graphics. That opens more possibilities for the application, for example, the user could rig the hand and interact with the virtual graphics together with the physical objects.

Fig. 15. The pair of scissors becomes the beak of a bird and when it moves, the wings of the bird will flap.

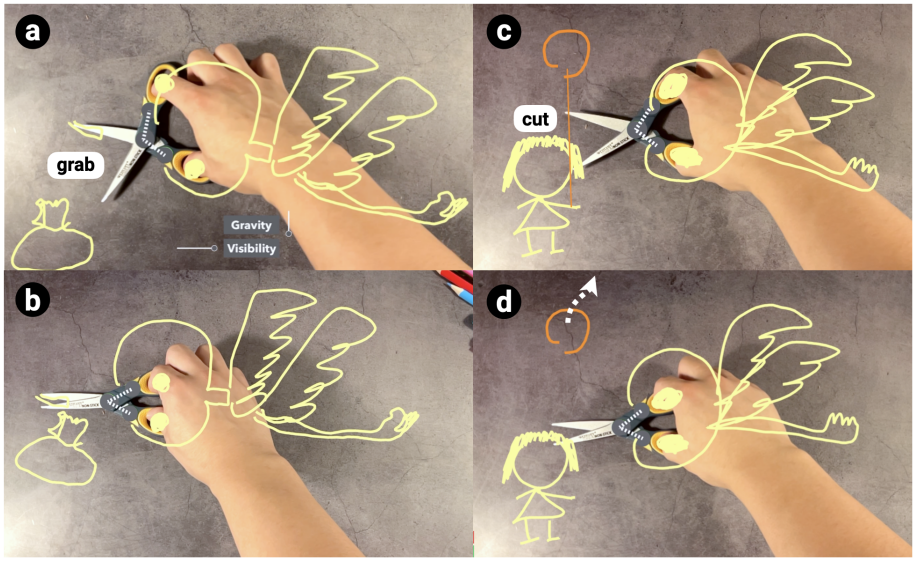


Fig. 16. The bird can grab a bag on the ground (a, b) and cut the string of the balloon held by a girl (c, d) when the pair of scissors closes.

5.3 Artistic storytelling

Physical objects are commonly used for different applications such as parts of a creative sketch (Figures 15 to 17). With the ability to use these physical objects to drive live virtual effect with different types and levels of control, we could animate the graphics of the art and achieve an immersive experience of the storytelling. Figure 17 illustrates three examples of food arts where the real objects play important roles in telling the story. As shown in Figure 17a, a little monster is holding a flower as a flashlight which could lighten its hidden friend. Also in Figure 17b, multiple eggs in the scene have smiling face. If someone takes one of the egg, the other eggs will start to cry because they realize that one of them is going to be eaten. The user could achieve such storytelling by adding a swapping key-frame between the smiling and crying face and use the motion of the egg as the driving parameters. In Figure 17c, the banana becomes a parachute of the skydiver drawn by the user. The movement of the banana will carry and swing the diver. By adjusting the global force, the diver encounters a bad situation — blown away by large wind.

A series of story can be created using a single pair of scissors which becomes the beak of the bird. When the scissors move, the bird will flap its wings (Figure 15) using keyframes specified by the user. By adding mappings with trigger

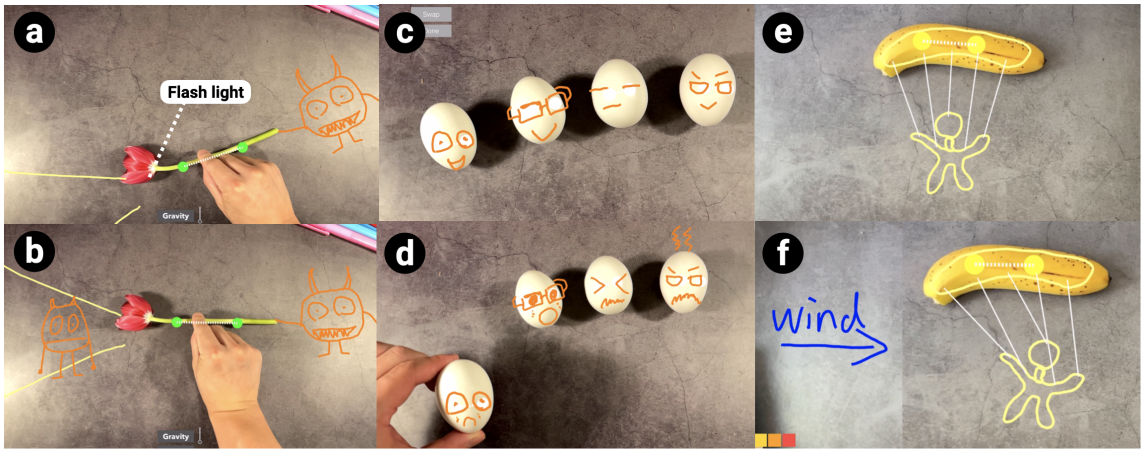


Fig. 17. *Food arts.* The flower becomes a flash light held by the little monster (a, b); the eggs will start to cry when one of them is grabbed away (c, d); and the banana becomes a parachute of a skydiver (e, f).

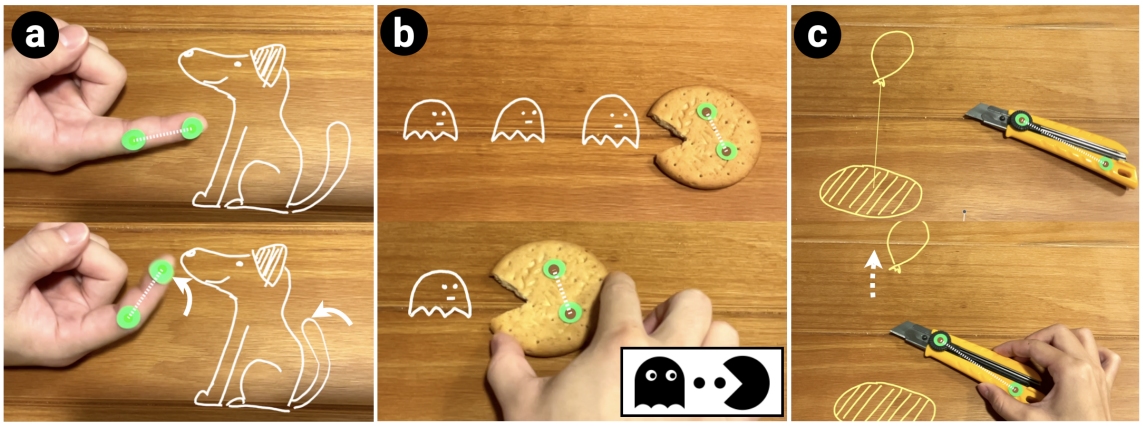


Fig. 18. *Results made by an external designer:* curving a finger to wag a virtual dog's tail (a), sliding a biscuit Pac-Man to eat virtual ghosts (b), and using a physical knife to cut the virtual string connecting a virtual rock and a virtual balloon to make it float away (c).

using the position of the bird, the bird could hold the bag on the ground in the mouse while flying over it (Figure 16ab) by creating a physical joint between the beak and the bag. The naughty bird may also bother the little girl who is holding a balloon (Figure 16cd) and the balloon flying away since the string is broke by the bird (removing the physical joint between the balloon and the bird using the angle of the scissors as trigger).

5.4 External design

To broaden the potential applications of our system, we had an informal design session with one external designer (female, age=29) who is an student of industrial design with experience in authoring software including Adobe After Effects and Unity. She was firstly introduced the workflow of our system by walking through the example of scissor-as-Pacman (Figure 10). After that, the designer took approximately an hour of brainstorming and came up with three

different applications as shown in Figure 18, including petting a dog (a), turning a piece of biscuits into a Pac-man (b) and cut the string of the balloon with a cutter (c).

After the brainstorming and the design stage, we had a informal interview with the participant about her thought about our system. The external designer agrees that our system can achieve various output effect and is easy to learn, as she commented - "It is very open and the user could a lot of different output effect with very simple manipulation" and "The manipulation is very simple and easy to learn". However, she also pointed out that the feedback while interacting with the interface could be improved – "The feedback of each button during the interaction is not good enough as I do not know if I press the button successfully". She also thought that some of the functionality is not straightforward to the user such as the gravity – "The function of adjusting gravity is not so user-friendly as I cannot know that I could use it to achieve an effect of floating the balloon without instruction". Finally, she gave some suggestions regarding the future development of the tool – "It would be super great if it has different drawing tool like those in the commercial software, and definitely more powerful output effect can be achieved using that."

6 CONCLUSIONS

We have presented a prototype system that enables users to rig and track everyday objects so that they can be tangibly manipulated to drive the deformation and animation of virtual graphical effects. With commodity hardware (e.g., a smart phone or tablet equipped with a camera), users can create a variety of experiences with our system, including handle-based deformation, key-frame animation, and trigger-based interactions with stylized or physically plausible effects, across application domains such as interactive art, tangible gaming, and concept explanation. We have taken one more step following the recent line of work [44, 48] and believe that exciting future opportunities exist for democratizing the power of interactive, mixed-reality experiences across a range of application domains.

7 LIMITATIONS AND FUTURE WORK

Our first prototype for driving virtual graphical effects with physical objects has a limited feature set and can be expanded in terms of tracking and authoring capabilities. Some potential future directions are as follows.

Combining with multi-modal input. Our mapping strategy could potentially be applied to other modalities such as sound and lighting in addition to space, which already provides large enough design space for the scope of this paper. Exploring other modalities and the interaction/combination of multiple modalities can be a fruitful future direction.

Semantic meaning for the physical object. While the current interaction between the physical object and the virtual graphics all rely on user improvisation, one interesting future direction could be adding semantic meaning for the physical object during the authoring. For example, a physical rubber could erase drawn virtual graphics, or a physical pendulum swing could knock out virtual balls.

REFERENCES

[1] Adobe. 2018. Puppet Warp. https://helpx.adobe.com/photoshop/using/warp-images-shapes-paths.html

[2] Rahul Arora, Rubaiat Habib Kazi, Tovi Grossman, George Fitzmaurice, and Karan Singh. 2018. *SymbiosisSketch: Combining 2D & 3D Sketching for Designing Detailed 3D Objects in Situ.* 1–15. https://doi.org/10.1145/3173574.3173759

[3] Rahul Arora, Rubaiat Habib Kazi, Danny M. Kaufman, Wilmot Li, and Karan Singh. 2019. MagicalHands: Mid-Air Hand Gestures for Animating in VR. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New Orleans, LA, USA) *(UIST '19)*. 463–477. https://doi.org/10.1145/3332165.3347942

[4] Connelly Barnes, David E. Jacobs, Jason Sanders, Dan B Goldman, Szymon Rusinkiewicz, Adam Finkelstein, and Maneesh Agrawala. 2008. Video Puppetry: A Performative Interface for Cutout Animation. *ACM Trans. Graph.* 27, 5, Article 124 (Dec. 2008), 9 pages. https://doi.org/10.1145/1409060.1409077

[5] Leonardo Bonanni, Chia-Hsun Lee, and Ted Selker. 2005. CounterIntelligence: Augmented reality kitchen. In *CHI 2005 Extended Abstract*, Vol. 2239. 45.

[6] Scott Brave, Hiroshi Ishii, and Andrew Dahley. 1998. Tangible interfaces for remote collaboration and communication. In *Proceedings of the 1998 ACM conference on Computer supported cooperative work*. 169–178.

[7] Nicky Case. 2015. Explorable Explanations. https://explorabl.es

[8] Jiawen Chen, Shahram Izadi, and Andrew Fitzgibbon. 2012. KinÊTre: Animating the World with the Human Body. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology* (Cambridge, Massachusetts, USA) *(UIST '12)*. 435–444. https://doi.org/10.1145/2380116.2380171

[9] Zhutian Chen, Yijia Su, Yifang Wang, Qianwen Wang, Huamin Qu, and Yingcai Wu. 2020. MARVisT: Authoring Glyph-Based Visualization in Mobile Augmented Reality. *IEEE Transactions on Visualization and Computer Graphics* 26, 8 (2020), 2645–2658. https://doi.org/10.1109/TVCG.2019.2892415

[10] Lydia B. Chilton, Savvas Petridis, and Maneesh Agrawala. 2019. *VisiBlends: A Flexible Workflow for Visual Blends*. 1–14. https://doi.org/10.1145/3290605.3300402

[11] CNN Business. 2018. This is what a trade war looks like. https://youtu.be/VA-LdvH35Uk.

[12] Diego Cusano. 2015. Food and Everyday Objects Turned Into Creative Illustrations. https://www.behance.net/Diego_Cusano

[13] Tobias Drey, Jan Gugenheimer, Julian Karlbauer, Maximilian Milo, and Enrico Rukzio. 2020. *VRSketchIn: Exploring the Design Space of Pen and Tablet Interaction for 3D Sketching in Virtual Reality*. 1–14. https://doi.org/10.1145/3313831.3376628

[14] David Furió, Stéphanie Fleck, Bruno Bousquet, Jean-Paul Guillet, Lionel Canioni, and Martin Hachet. 2017. *HOBIT: Hybrid Optical Bench for Innovative Teaching*. 949–959. https://doi.org/10.1145/3025453.3025789

[15] Oliver Glauser, Wan-Chun Ma, Daniele Panozzo, Alec Jacobson, Otmar Hilliges, and Olga Sorkine-Hornung. 2016. Rig Animation with a Tangible and Modular Input Device. *ACM Trans. Graph.* 35, 4, Article 144 (July 2016), 11 pages. https://doi.org/10.1145/2897824.2925909

[16] Robert Held, Ankit Gupta, Brian Curless, and Maneesh Agrawala. 2012. 3D puppetry: a kinect-based interface for 3D animation. In *UIST '12*. 423–434.

[17] Anuruddha Hettiarachchi and Daniel Wigdor. 2016. Annexing Reality: Enabling Opportunistic Use of Everyday Objects as Tangible Proxies in Augmented Reality. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. 1957–1967. https://doi.org/10.1145/2858036.2858134

[18] Derek Hodson. 1996. Laboratory work as scientific method: Three decades of confusion and distortion. *Journal of Curriculum studies* 28, 2 (1996), 115–135.

[19] Adrian Iftene, Diana Trandabăț, and Vlad Rădulescu. 2020. Eye and Voice Control for an Augmented Reality Cooking Experience. *Procedia Computer Science* 176 (2020), 1469–1478. https://doi.org/10.1016/j.procs.2020.09.157 Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 24th International Conference KES2020.

[20] Takeo Igarashi, Tomer Moscovich, and John F. Hughes. 2005. As-Rigid-as-Possible Shape Manipulation. *ACM Trans. Graph.* 24, 3 (July 2005), 1134–1141. https://doi.org/10.1145/1073204.1073323

[21] Hiroshi Ishii and Brygg Ullmer. 1997. Tangible Bits: Towards Seamless Interfaces between People, Bits and Atoms. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems* (Atlanta, Georgia, USA) *(CHI '97)*. 234–241. https://doi.org/10.1145/258549.258715

[22] Ben Jones, Jovan Popovic, James McCann, Wilmot Li, and Adam Bargteil. 2015. Dynamic sprites: artistic authoring of interactive animations. *Computer Animation and Virtual Worlds* 26, 2 (2015), 97–108. https://doi.org/10.1002/cav.1608

[23] Seokbin Kang, Ekta Shokeen, Virginia L. Byrne, Leyla Norooz, Elizabeth Bonsignore, Caro Williams-Pierce, and Jon E. Froehlich. 2020. *ARMath: Augmenting Everyday Life with Math Learning*. 1–15. https://doi.org/10.1145/3313831.3376252

[24] Rubaiat Habib Kazi, Fanny Chevalier, Tovi Grossman, and George Fitzmaurice. 2014. Kitty: Sketching Dynamic and Interactive Illustrations. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology* (Honolulu, Hawaii, USA) *(UIST '14)*. 395–405. https://doi.org/10.1145/2642918.2647375

[25] Rubaiat Habib Kazi, Fanny Chevalier, Tovi Grossman, Shengdong Zhao, and George Fitzmaurice. 2014. Draco: Bringing Life to Illustrations with Kinetic Textures. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Toronto, Ontario, Canada) *(CHI '14)*. 351–360. https://doi.org/10.1145/2556288.2556987

[26] Rubaiat Habib Kazi, Tovi Grossman, Nobuyuki Umetani, and George Fitzmaurice. 2016. *Motion Amplifiers: Sketching Dynamic Illustrations Using the Principles of 2D Animation*. 4599–4609. https://doi.org/10.1145/2858036.2858386

[27] Kin Chung Kwan and Hongbo Fu. 2019. Mobi3DSketch: 3D Sketching in Mobile AR. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. 1–11. https://doi.org/10.1145/3290605.3300406

[28] David Lakatos, Matthew Blackshaw, Alex Olwal, Zachary Barryte, Ken Perlin, and Hiroshi Ishii. 2014. T (ether): spatially-aware handhelds, gestures and proprioception for multi-user 3D modeling and animation. In *Proceedings of the 2nd ACM symposium on Spatial user interaction*. 90–93.

[29] Joseph J. LaViola and Robert C. Zeleznik. 2004. MathPad2: A System for the Creation and Exploration of Mathematical Sketches. *ACM Trans. Graph.* 23, 3 (Aug. 2004), 432–440. https://doi.org/10.1145/1015706.1015741

[30] Bongshin Lee, Rubaiat Habib Kazi, and Greg Smith. 2013. SketchStory: Telling more engaging stories with data through freeform sketching. *IEEE Transactions on Visualization and Computer Graphics* 19, 12 (2013), 2416–2425.

[31] Germán Leiva, Cuong Nguyen, Rubaiat Habib Kazi, and Paul Asente. 2020. Pronto: Rapid Augmented Reality Video Prototyping Using Sketches and Enaction. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. 1–13. https://doi.org/10.1145/3313831.3376160

[32] Jingyuan Liu, Hongbo Fu, and Chiew-Lan Tai. 2020. PoseTween: Pose-driven Tween Animation. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*. 791–804.

[33] Markus Löchtefeld, Sven Gehring, Ralf Jung, and Antonio Krüger. 2011. GuitAR: Supporting Guitar Learning through Mobile Projection. In *CHI '11 Extended Abstracts on Human Factors in Computing Systems* (Vancouver, BC, Canada) *(CHI EA '11)*. Association for Computing Machinery, New York, NY, USA, 1447–1452. https://doi.org/10.1145/1979742.1979789

[34] William F McComas, Michael P Clough, and Hiya Almazroa. 1998. The role and character of the nature of science in science education. In *The nature of science in science education*. Springer, 3–39.

[35] Koki Nagano, Jaewoo Seo, Jun Xing, Lingyu Wei, Zimo Li, Shunsuke Saito, Aviral Agarwal, Jens Fursund, and Hao Li. 2018. PaGAN: Real-Time Avatars Using Dynamic Textures. *ACM Trans. Graph.* 37, 6, Article 258 (Dec. 2018), 12 pages. https://doi.org/10.1145/3272127.3275075

[36] Michael Nebeling and Maximilian Speicher. 2018. The Trouble with Augmented Reality/Virtual Reality Authoring Tools. In *2018 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*. 333–337. https://doi.org/10.1109/ISMAR-Adjunct.2018.00098

[37] Benjamin Nuernberger, Eyal Ofek, Hrvoje Benko, and Andrew D. Wilson. 2016. *SnapToReality: Aligning Augmented Reality to the Real World*. Association for Computing Machinery, New York, NY, USA, 1233–1244. https://doi.org/10.1145/2858036.2858250

[38] Ken Perlin, Zhenyi He, and Karl Rosenberg. 2018. Chalktalk: A Visualization and Communication Language–As a Tool in the Domain of Computer Science Education. *arXiv preprint arXiv:1809.07166* (2018).

[39] Jing Qian, Jiaju Ma, Xiangyu Li, Benjamin Attal, Haoming Lai, James Tompkin, John F. Hughes, and Jeff Huang. 2019. Portal-Ble: Intuitive Free-Hand Manipulation in Unbounded Smartphone-Based Augmented Reality. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New Orleans, LA, USA) *(UIST '19)*. 133–145. https://doi.org/10.1145/3332165.3347904

[40] Iulian Radu and Bertrand Schneider. 2019. What Can We Learn from Augmented Reality (AR)? Benefits and Drawbacks of AR for Inquiry-Based Learning of Physics. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–12. https://doi.org/10.1145/3290605.3300774

[41] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. 2021. Zero-shot text-to-image generation. *arXiv preprint arXiv:2102.12092* (2021).

[42] Ramesh Raskar, Paul Beardsley, Jeroen van Baar, Yao Wang, Paul Dietz, Johnny Lee, Darren Leigh, and Thomas Willwacher. 2004. RFIG Lamps: Interacting with a Self-Describing World via Photosensing Wireless Tags and Projectors. *ACM Trans. Graph.* 23, 3 (Aug. 2004), 406–415. https://doi.org/10.1145/1015706.1015738

[43] Hans Rosling. 2015. Why the world population won't exceed 11 billion. https://youtu.be/2LyzBoHo5EI.

[44] Nazmus Saquib, Rubaiat Habib Kazi, Li-Yi Wei, and Wilmot Li. 2019. Interactive Body-Driven Graphics for Augmented Video Performance. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. 1–12. https://doi.org/10.1145/3290605.3300852

[45] Nazmus Saquib, Rubaiat Habib Kazi, Li-yi Wei, Gloria Mark, and Deb Roy. 2021. Constructing Embodied Algebra by Sketching. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21)*. Article 428, 16 pages. https://doi.org/10.1145/3411764.3445460

[46] Jeremy Scott and Randall Davis. 2013. Physink: Sketching Physical Behavior. In *Proceedings of the Adjunct Publication of the 26th Annual ACM Symposium on User Interface Software and Technology (UIST '13 Adjunct)*. 9–10. https://doi.org/10.1145/2508468.2514930

[47] Qingkun Su, Xue Bai, Hongbo Fu, Chiew-Lan Tai, and Jue Wang. 2018. *Live Sketch: Video-Driven Dynamic Deformation of Static Drawings*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3173574.3174236

[48] Ryo Suzuki, Rubaiat Habib Kazi, Li-yi Wei, Stephen DiVerdi, Wilmot Li, and Daniel Leithinger. 2020. RealitySketch: Embedding Responsive Graphics and Visualizations in AR through Dynamic Sketching. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology (UIST '20)*. 166–181. https://doi.org/10.1145/3379337.3415892

[49] Xiao Tang, Xiaowei Hu, Chi-Wing Fu, and Daniel Cohen-Or. 2020. GrabAR: Occlusion-Aware Grabbing Virtual Objects in AR. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (Virtual Event, USA) *(UIST '20)*. 697–708. https://doi.org/10.1145/3379337.3415835

[50] John Underkoffler and Hiroshi Ishii. 1999. Urp: A Luminous-Tangible Workbench for Urban Planning and Design. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Pittsburgh, Pennsylvania, USA) *(CHI '99)*. 386–393. https://doi.org/10.1145/302979.303114

[51] Bret Victor. 2011. Explorable explanations. http://worrydream.com/ExplorableExplanations/

[52] Wall Street Journal. 2021. What Is Turbulence? A Pilot Explains How It Happens, Even in Clear Skies. https://youtu.be/5wQ9nAlO12E.

[53] Christian Weichel, Manfred Lau, David Kim, Nicolas Villar, and Hans W. Gellersen. 2014. MixFab: A Mixed-Reality Environment for Personal Fabrication. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Toronto, Ontario, Canada) *(CHI '14)*. Association for Computing Machinery, New York, NY, USA, 3855–3864. https://doi.org/10.1145/2556288.2557090

[54] Eric Whitmire, Hrvoje Benko, Christian Holz, Eyal Ofek, and Mike Sinclair. 2018. *Haptic Revolver: Touch, Shear, Texture, and Shape Rendering on a Reconfigurable Virtual Reality Controller*. 1–12. https://doi.org/10.1145/3173574.3173660

[55] Nora S. Willett, Wilmot Li, Jovan Popovic, Floraine Berthouzoz, and Adam Finkelstein. 2017. Secondary Motion for Performed 2D Animation. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology* (Québec City, QC, Canada) *(UIST '17)*. Association for

Computing Machinery, New York, NY, USA, 97–108. https://doi.org/10.1145/3126594.3126641

[56] Jun Xing, Rubaiat Habib Kazi, Tovi Grossman, Li-Yi Wei, Jos Stam, and George Fitzmaurice. 2016. Energy-Brushes: Interactive Tools for Illustrating Stylized Elemental Dynamics. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology* (Tokyo, Japan) *(UIST '16)*. 755–766. https://doi.org/10.1145/2984511.2984585

[57] Masahiro Yamaguchi, Shohei Mori, Peter Mohr, Markus Tatzgern, Ana Stanescu, Hideo Saito, and Denis Kalkofen. 2020. Video-Annotated Augmented Reality Assembly Tutorials. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*. 1010–1022. https://doi.org/10.1145/3379337.3415819

[58] Qian Zhou, Sarah Sykes, Sidney Fels, and Kenrick Kin. 2020. Gripmarks: Using Hand Grips to Transform In-Hand Objects into Mixed Reality Input. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. 1–11. https://doi.org/10.1145/3313831.3376313

[59] Bo Zhu, Michiaki Iwata, Ryo Haraguchi, Takashi Ashihara, Nobuyuki Umetani, Takeo Igarashi, and Kazuo Nakazawa. 2011. Sketch-Based Dynamic Illustration of Fluid Systems. *ACM Trans. Graph.* 30, 6 (Dec. 2011), 1–8. https://doi.org/10.1145/2070781.2024168