Zooids: Building Blocks for Swarm User Interfaces

Mathieu Le Goc, Lawrence Kim, Ali Parsaei, Jean-Daniel Fekete, Pierre Dragicevic, Sean Follmer

Zooids: Building Blocks for Swarm User Interfaces

Mathieu Le Goc[1,3,4], Lawrence H. Kim[2], Ali Parsaei[2], Jean-Daniel Fekete[1,4] Pierre Dragicevic[1,4], Sean Follmer[2]

[1]Inria, [2]Stanford University, [3]Université Paris-Sud, [4]Université Paris-Saclay

{mathieu.le-goc, pierre.dragicevic, jean-daniel.fekete}@inria.fr, {lawkim, aparsaei, sfollmer}@stanford.edu

Figure 1. ***Zooids*** **can be held as tokens, manipulated collectively or individually, behave as physical pixels, act as handles and controllers, and can move dynamically under machine control. They are building blocks for a new class of user interface we call** ***swarm user interfaces*****.**

ABSTRACT

This paper introduces *swarm user interfaces*, a new class of human-computer interfaces comprised of many autonomous robots that handle both display and interaction. We describe the design of *Zooids*, an open-source open-hardware platform for developing tabletop swarm interfaces. The platform consists of a collection of custom-designed wheeled micro robots each 2.6 cm in diameter, a radio base-station, a high-speed DLP structured light projector for optical tracking, and a software framework for application development and control. We illustrate the potential of tabletop swarm user interfaces through a set of application scenarios developed with Zooids, and discuss general design considerations unique to swarm user interfaces.

ACM Classification Keywords

H.5.m. Information Interfaces and Presentation (e.g. HCI): Miscellaneous

Author Keywords

Swarm user interfaces; tangible user interfaces.

INTRODUCTION

This article contributes to bringing closer to reality the vision of Ivan Sutherland for the Ultimate Display as *"a room within which the computer can control the existence of matter"* [70], and Hiroshi Ishii's vision of Radical Atoms where people can interact with *"a new kind of matter capable of changing form dynamically"* [26].

Several significant steps have been recently made towards Sutherland's and Ishii's visions, particularly through research on actuated tangibles [48, 50, 78] and shape displays [55, 56, 15]. However, current systems suffer from a number of limitations. First, actuated tabletop tangibles generally only support the manipulation and actuation of a few (e.g., 3–4) solid objects, which is not enough to emulate physical matter that can change form. On the other hand, shape displays try to achieve surfaces that can be deformed and actuated, but current implementations do not support arbitrary physical topologies. Furthermore, both types of systems traditionally use physical objects primarily as *input*, while *output* is almost always provided through separate pixel-based display technology. Although video-projected overlays allows input and output to spatially coincide [12], they provide only a limited sense of physicality [5]. Likewise, many such systems require heavy hardware or displays to function, and are thus primarily meant to be operated in sterile environments rather than embedded in our own physical world [24, 77].

Our research work fills this current gap in user interface technologies by introducing *Zooids* and *swarm user interfaces* (see figure 1). A *Zooid* is a hardware and software system: a small wheel-propelled robot with position and touch sensing capabilities that can be freely arranged and repositioned on any horizontal surface, both through user manipulation and computer control.

A Zooid is defined in Wikipedia as *"a single animal that is part of a colonial animal. Zooids are multicellular; their structure is similar to that of other solitary animals."* Zooids build on work from swarm robotics [10, 68], adding interaction and speed. *Swarm user interfaces* are interfaces built using

UIST 2016, October 16-19, 2016, Tokyo, Japan

ACM ISBN ...

DOI: http://dx.doi.org/10.1145/2984511.2984547

collections of self-propelled physical objects (e.g., mini robots) that can move collectively and react to user input. Swarm user interfaces can be seen as a coarse-grained version of Sutherland's and Ishii's futuristic visions of user interfaces based on programmable matter.

Due to zooids' ability to freely and quickly reconfigure themselves spatially, a collection of zooids can act as a display and can provide meaningful user output. Due to their ability to sense user actions, zooids can also support rich input. For example, users can either move zooids one by one, or manipulate many zooids at once using "sweeping" gestures [35]. Sophisticated interactive behaviors can be implemented on the application side, e.g., zooids can act as controls or as handles for manipulating others zooids; they can even move other light objects. At the same time, since all input *and* output can be mediated through the same physical elements, the system is able to achieve a complete fusion between input and output and provide a full experience of physical manipulation. Finally, the system is relatively lightweight and only requires the use of a compact DLP projector ($122\,mm \times 115\,mm \times 48\,mm$) for optical tracking. Zooids can operate on any horizontal surface (e.g., a sheet of paper, a messy office desk, a dining table, or a game board), making it possible to blend swarm user interfaces with everyday physical environments. To stimulate future research on swarm user interfaces, we distribute our Zooids tabletop swarm user interface platform in open-source and open-hardware.

In summary, our contributions are:

- A working definition for swarm user interfaces with several implemented examples,
- The first open-source hardware/software platform for experimenting with tabletop swarm user interfaces,
- A set of scenarios to illustrate the unprecedented possibilities offered by our system and by tabletop swarm user interfaces in general,
- A discussion of some general design principles and design challenges for swarm user interfaces.

Furthermore, as benefits, Zooids:

- can coexist in large numbers, in comparison to previous actuated tangible user interfaces,
- can act as individual objects, while being small enough to also act as *"pixels"* of a physical display,
- can be manipulated either individually or collectively, including with physical tools such as rulers,
- are lightweight, can operate on any horizontal surface, and relatively cost-effective: about 50 USD each now, down to $20 if mass manufactured.

BACKGROUND

Our work is related to several research areas, namely: tabletop tangible user interfaces, shape displays, swarm robotics and data physicalization.

Tabletop Tangible User Interfaces

Although tangible user interfaces (TUIs) come in many different forms (including modular assemblies of sensors and actuators [19, 40]), *tabletop TUIs* are particularly common. Tabletop TUIs allow users to interact with digital information by moving physical objects (tangibles) on flat table surfaces [72, 73]. These systems have been used for a range of applications such as systems engineering control [51], musical composition [52], urban planning [74], and education [23].

One limitation with classical tabletop TUIs is the one-way mapping between digital and physical objects — if the former change, the latter can become inconsistent [26]. A number of technologies have been proposed to actuate tangibles, including 2D gantries [6, 38], arrays of electromagnets [48, 50, 78, 76], arrays of ultrasonic transducers [42], electro-static attraction [80, 4], vibration [57, 81] and mobile robots [60, 30, 58, 47, 43, 53, 49]. These systems also support position tracking through a variety of means such as optical tracking with cameras of LEDs or markers (including those using an optical multitouch table), or projector-based tracking. The tangibles range in size from coin-sized [48] to 10 cm [53].

A variety of interaction techniques have been explored on actuated tabletop TUIs, primarily based on the direct manipulation of a single tangible per hand [48], or of small groups of tangibles through multitouch input [53]. Patten [50] explored the use of passive tools in conjunction with actuated tangibles for specifying computational constraints. Other researchers have added dimensions such as vertical displacement to actuated tangibles [43]. These active tangibles can provide haptic feedback while interacting, by applying force directly to the user's hand as they translate along the surface [48, 41]. Actuated TUIs also provide opportunities for remote collaboration, as remote objects can be kept in sync [6, 58].

The design space of tabletop TUIs is vast, and a lot has been explored. However, previous systems have not considered interaction with many (e.g., 10, 30 or more) small actuated tangibles, which we show opens up possibilities for novel interactions and applications. Also, in many previous systems [48, 50, 60, 58, 47, 43, 53, 49, 81], tangibles are used in conjunction with a graphical display, so the tangibles primarily act as handles for digital information. We are interested in user interfaces where the tangibles are used not only as controllers, but also as representations of digital content.

Shape Displays and Programmable Matter

Shape displays are user interfaces involving physical surfaces or volumes that can sense user input and whose geometry can be computer-controlled [56, 61]. Many such systems support discretized shape control of 2.5D surfaces using arrays of motorized bars [54, 39, 15], while other systems support continuous shape control using pneumatic or hydraulic actuation [14, 82] or shape-memory alloys [61]. Currently, many of these systems require a heavy equipment and only allow limited control over physical geometry and topology. In particular, none of the previous systems can emulate separate, detached physical objects.

These research efforts are partly inspired by visions such as Sutherland's and Ishii's discussed before, where computers would be able to reconfigure physical matter to recreate any physical shape. Other fields such as robotics and material science have been interested in realizing this dream of "pro-

grammable matter", but most of the progress so far has been purely theoretical [18, 64]. Working prototypes rely on swarm robotics, that we discuss next.

Swarm Robotics

Swarm robots draw from natural swarms, where social animals such as birds or ants can produce complex collective behavior by moving and interacting with each other according to simple rules. The largest robotic swarm implemented so far involves as many as 1,000 robots although they move slowly (about 1 cm/s vs. ~50 cm/s for Zooids) [63]. Our paper is inspired from past research in swarm robotics [10, 9], but while the area of swarm robotics has been mostly interested in how to emulate swarm behavior using distributed intelligence and fully autonomous agents, we focus on direct physical interaction with small swarm robots, HCI applications, and employ a centralized system to coordinate robots.

Researchers in robotics have started to develop methods for interacting with swarm robots, but most of them have only been tested on mouse-operated computer simulations [29, 31]. Alonso-Mora and colleagues investigated the use of swarm robots as physical displays [2] and recently extended their system to support interaction through sketching [21], handheld tablet input [20] and mid-air gestures [3]. Their systems share many features with Zooids, but our paper instead focuses on direct *tangible* manipulation of swarm robots and explores a wider range of application scenarios.

Recently, Rubens et al [62] described a mid-air 3D physical display system based on drones, with which users can interact by directly manipulating drones. Despite their goal of ultimately converging towards a swarm user interface, each drone is rather large (8.9 cm) and the number of drones that can be simultaneously used is limited due to turbulence issues — their prototype currently consists of 3 drones. The Drone 100 [16] project involves a hundred drones called "spaxels". Each is light-equipped and can be positioned in three dimensions, resulting in a choreographed swarm capable of displaying dynamic images. However, the large operating volume prevents direct manipulation.

Data Physicalization

Based on research in cognitive science around embodied and distributed cognition, there has been recent interest in the information visualization field around physical data visualization [28, 25, 84]. Researchers have already shown that there can be benefits to passive physical representations of data to promote engagement [45], to better support data exploration [27], and for the vision impaired [36].

Less work has explored dynamic physical visualizations because they are more complex to build [28], but recent work has investigated the use of 2.5D shape displays for data exploration [71]. However, the range of visualization techniques that can be supported with 2.5D shape displays is limited. Swarm interfaces provide a promising platform to physicalize many traditional 2D information visualizations, as well as newer interactive data visualizations [83, 44].

Swarm User Interfaces

We propose to refer to *swarm user interfaces* (swarm UIs) as:

> *"human-computer interfaces made of independent self-propelled elements that move collectively and react to user input".*

Independent: the user interface elements need to be physically detached from each other and free to move. Counter-examples include graphical elements on a computer display, which are all part of a single physical object. Articulated models, 2.5D shape displays [15] and physical control panels such as mixing consoles also do not qualify, since the moving parts and controls are attached and not free to move.

Self-propelled: the elements need to be able to move without external forces. Counter-examples include passive physical tokens [25, 34].

Move collectively: by definition, swarming behavior involves collective motion. Thus the elements need to be able to move in a coordinated fashion, either by exchanging information with each other or with a centralized coordinator. In addition, the more elements a user interface contains, the more their motion can be qualified as collective, and thus the more "swarm-like" the interface is.

React to user input: the elements need to sense user input and react to this input. Thus, most swarm robotics systems are not swarm user interfaces, because they lack the user interaction element. A swarm display that is interactive but only takes user input from external sources — e.g., a mouse or a keyboard — is not a swarm user interface either according to our definition, because the elements themselves need to be able to react to user input. Systems that use computer vision to detect mid-air gestures such as DisplaySwarm [3] are in a gray area. For smooth interaction, speed of the system is critical: the elements of a swarm UI need to be fast enough for shape change to occur at a usable rate. The ideal transition time is in the order of one second, because this is about the limit where a system is perceived as interactive [46], and it is also the recommended duration for animated transitions on regular graphical displays [22].

The systems that come closest to swarm user interfaces according to our definition are self-propelled tangibles [60, 30, 58, 47, 43, 53, 49] and BitDrones [62], because they are made of independent self-propelled elements that can move in a coordinated fashion and can be directly manipulated. However, these systems involve few elements (i.e., around 4-5), and are thus at best low-fidelity prototypes of actual swarm user interfaces. While many such systems could have involved more units, a small form factor (e.g., zooids are more than $3\times$ smaller than Rosenfeld's [60] robots) enables different types of interactions. Users can manipulate many zooids at once, while several dozens of larger robots may not even fit on a regular table. Moreover, previous work does not discuss or demonstrate swarm user interfaces, which are our focus.

In principle, swarm user interfaces could take many forms and could be implemented in many different ways. For example, a swarm UI can consist of free-floating particles [66] or

drones [62] that are free to move in 3D space, or can consist of objects that evolve on a 2D surface [48]. In this article we focus on elements that move on a 2D surface, i.e., tabletop swarm user interfaces. Next, we discuss our implementation of zooids, then illustrate the possibilities offered by tabletop swarm interfaces through examples implemented using zooids.

SWARM UI EXAMPLES WITH ZOOIDS

In this section we illustrate and discuss possibilities Zooids offer through simple use cases and scenarios, before explaining their hardware and software design. Most examples can also be seen in the accompanying video.

Swarm Drawing

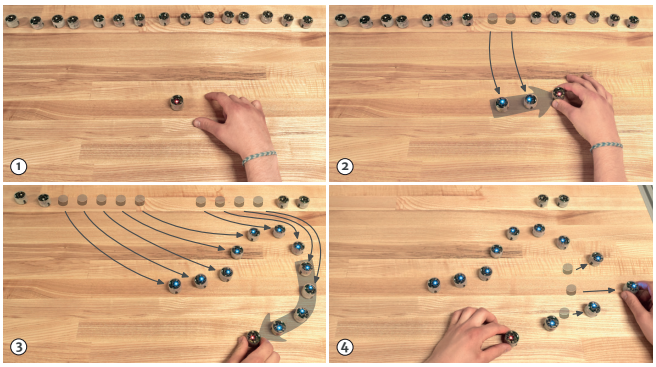

Figure 2. Freehand swarm drawing (1-3) and curve manipulation (4).

Freehand Drawing

Inspired from vector graphics authoring tools, we have implemented a swarm version of a freehand drawing tool, shown in Figure 2: initially, the freehand drawing zooid stands in the center of the working surface, while unassigned zooids wait at the top, in an idle state (Figure 2-⓪1). When the user drags the freehand drawing zooid, the previously idle zooids move to the path of the drawing zooid to form a physical trail (Figure 2-⓪2 and ⓪3). When the system runs out of idle zooids, the trail follows the freehand drawing tool like a snake. The curve can also be deformed by dragging its constituent zooids individually (Figure 2-⓪4), or by moving many of them simultaneously, e.g., by pushing them with the side of the arm.

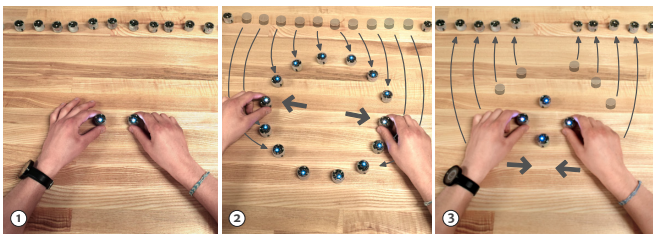

Figure 3. Circle swarm drawing, where zooids are automatically inserted (2) or discarded (3) depending on the circle's diameter.

Shapes

We also experimented with tools for drawing lines, rectangles and circles, based on the standard rubber band technique from desktop applications. Each of these tools employs two zooids as control points. Figure 3 shows the example of a circle tool, where two control points are used to define the circle's diameter, and idle zooids are automatically positioned to complete the circular shape. Zooids are also automatically added or removed depending on how many of them are necessary to construct the shape. Another zooid at the bottom of the table (not shown) allows users to switch between shapes.

Bézier Curves

In traditional vector drawing editing tools, Bézier curves allow for accurate shaping using discrete control points. We developed a physical curve editing tool where a collection of zooids are positioned to represent a curve. While shaping using the previously introduced drawing tool requires to manipulate many zooids at once, this tool uses specific zooids as control points to adjust the curvature represented by the collection. Each control point consists of two zooids, where one sets the anchor point and the other adjusts the tangent.

It is important to note that although GUIs currently support far higher information resolution, Zooids enable richer physical gestures. We believe that technology advances will allow higher resolution of swarm UIs in the future.

Interactive Swarm Visualization

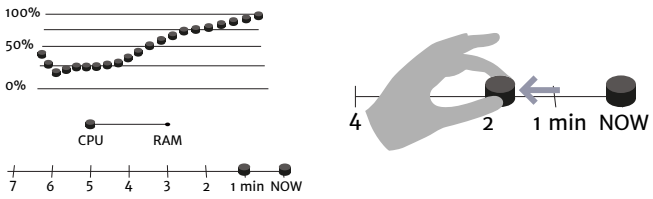

Figure 4. A line chart visualization using zooids.

Time-Series Navigation

We used zooids to visualize and navigate in time-series data. The physical interface illustrated in Figure 4-⓪1 shows with a line chart the evolution of CPU usage on a computer. Decorations such as axes and labels are static (e.g, printed on paper), while the data visualization itself is dynamic and continuously updated – the line chart appears to move to the left as new data arrives. At the bottom of the interface, zooids take on the role of widgets to let users customize the display and navigate the data. The two zooids at the bottom right specify the time range to visualize – they act like the two thumbs of a range slider [1]. If the user moves the left thumb to the left (see Figure 4-⓪2), the line chart stretches to show a wider time range. Moving both zooids to the left scrolls in the past. Finally, another zooid (see center of Figure 4-⓪1) lets users switch the visualization between CPU usage and memory usage.

Multiple Scatterplots

Scatterplots are one of the most common way to visualize data points. Looking more specifically at multivariate data, the ability to represent multiple dimensions at the same time is particularly relevant to better understand the data, identify trends and isolate clusters. Our scatterplot physicalization tool allows for multiple juxtaposed representations of a dataset, each representing a unique couple of dimensions. One representation can help identify a group of points. The user can

then pick up these zooid, and place in another scatterplot. As each zooid embodies a specific data point, moving it into a different chart allows users to probe different dimensions. In addition, the users can place the zooid on a active graphical display, such as a mobile phone or tablet, to find out additional parameters and information about that data point.

Stop Motion Swarm Animation

Inspired by traditional stop motion animation tools, we implemented a tool enabling users to author physical animations. The user positions each zooid to form the desired layout. Moving the timeline zooid a step forward saves the current layout as a key frame. Once the desired layouts have been recorded, toggling the second control zooid switches the mode to playback and plays consecutively the different keyframes.

In-the-Wild Scenarios

Although we have not implemented specific applications, we have begun to experiment with in-the-wild scenarios, in which zooids could be embedded with real-world environments. For example, they could be placed on a user's working desk to act as ambient displays (e.g., to show progress in downloads), extra controls, or as notification devices (e.g., they could hit a metallic or glass object when an important event starts or to remind you to drink water). Enough zooids can even move objects such as smartphones.

ZOOIDS HARDWARE AND SOFTWARE DESIGN

Elaborating from the examples of uses of zooids just presented, this section explains their hardware and software design.

Hardware

Robot Design

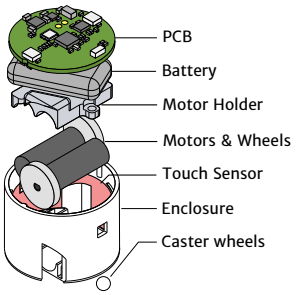

Figure 5. Exploded view of a zooid.

Zooids are small custom-made robots as shown in Figure 5; their dimensions are 26 mm in diameter, 21 mm in height and they weight about 12 g. Each robot is powered by a 100 mAh LiPo battery and uses motor driven wheels. The motors are placed non-colinearly to minimize the diameter. Even though the motors do not rotate around the same axis, the robot has the same net force and moment as would a robot with colinear motors. To drive the robot, a motor driver chip (Allegro A3901) and two micro motors (FA-GM6-3V-25) are used. With this combination, the robot has a maximum speed of approximately 74 cm/s. However, for controllability and smoothness of the motion, the robots move at a slower average speed of 44 cm/s for our applications.

A flexible electrode is wrapped inside the 3D printed enclosure to provide capacitive touch sensing capabilities. An integrated capacitive touch sensing circuit is included (Atmel AT42QT1070) to detect user's touch.

Embedded custom electronics, shown in the PCB layer of Figure 5, allows for robot control. A 48MHz ARM microcontroller (STMicroelectronics STM32F051C8) manages the overall logic computation and communicates wirelessly with the main master computer using a 2.4GHz radio chip (Nordic nRF24L01+). As part of the projector-based tracking system (explained in the next section), two photodiodes are placed at the top of the robot. Placed between the photodiodes, a color LED is used for robot identification and feedback.

Most of the power in the robots are consumed by (in order) the motors, radio module, micro-controller, and LED. When stationary, each robot consumes approximately 40 mA and 100 mA when moving. Thus, with a 100 mAh battery, robots are capable of moving for one hour, and can work even longer under normal usage.

Radio Communication

Each robot communicates with the radio receiver using the NRF24L01+ chip. Using a teensy 3.1 microcontroller as the master and Arduino Pro mini as the slave, we tested the total communication times for different numbers of slaves per master and packet sizes. From the experiment, we found that the total time is linearly dependent of both packet size and number of slaves, and that we could have up to 18 slaves per master for a packet size of 12 bytes. Zooids uses 10 slaves per master for a safety factor of about 2.

Projector-based Tracking System

A projector-based tracking system similar to Lee [37] is used for robot position tracking. As opposed to camera based systems, our projector based tracking system does not add any latency from networking for the local feedback control on each robot, making position control more stable. Our system setup is demonstrated in Figure 6. Using a high frame-rate (3000 Hz) projector (DLP LightCrafter) from Texas Instruments Inc., a sequence of gray-coded patterns are projected onto a flat surface. Then, the photodiodes on the robot independently decodes the gray code into a location within the projected area, and sends its position and orientation to the master computer. Due to the number of the patterns, the position refresh rate is approximately 73 Hz (1/(41 images per pattern $\times$ 1/3000)). Due to the diamond pixels of the projector, the horizontal and vertical resolutions slightly differ. In the current setup in which the projector is placed 1.25 m above the table producing a 1 m $\times$ 0.63 m projection area, the horizontal and vertical resolutions are 1.15 mm and 1.12 mm, respectively.

Calibration

Due to the discrepancies of the hardware, all robots do not exactly behave in the same manner and thus calibration for crucial elements is needed.

Minimum Speed Duty Cycle Each robot has a minimum speed or Pulse Width Modulation (PWM) duty cycle that is needed to overcome the static friction between the wheels

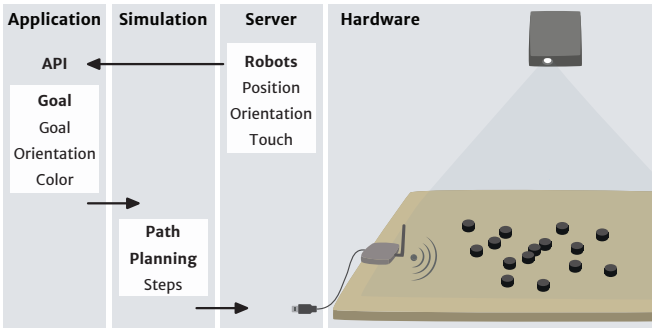

Figure 6. Software Architecture.

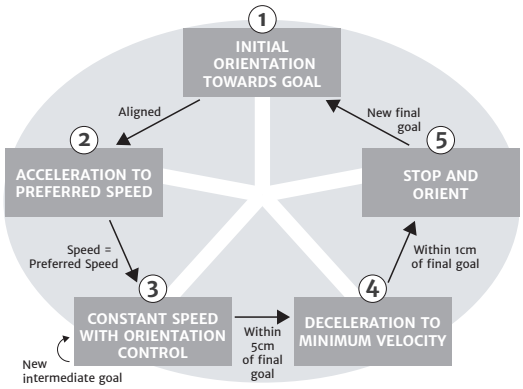

Figure 7. Structure of local PID position control

and the ground surface. While the robots have similar minimum duty cycle, they do not behave identically. Thus, during a startup phase, each robot goes through an initialization and calibration process to find their own parameters. This is achieved by incrementing the PWM duty cycle until it achieves moving the robot by 5 mm in 100 ms.

Preferred Speed Duty Cycle For most of their active time, robots move at their preferred speed. Similar to the minimum speed, there is a need for calibrating the preferred speed duty cycle. This is achieved again incrementing the PWM duty cycle until it moves at the nominal preferred speed of 44 cm/s.

Gain between Motors As each robot behaves differently, the motors within the robot also behave differently and thus, a calibration between the motors is needed. The calibration process is as follows: record the initial orientation, let the robot move for 0.5 s, compare the final and initial orientation and either increase or decrease the motor gain accordingly. This process is repeated until the initial and final orientations are less than 5 degrees apart.

Software

As shown in Figure 6, the communication structure consists of four main layers from highest to lowest level: Application, Simulation, Server, and Hardware.

At the application level, the desired positions of the robots are computed. These desired positions are transmitted to the simulation layer through a network socket. The application programmer can choose between two control strategies: *Proportional-Integral-Derivative* (PID) position control or *Hybrid Reciprocal Velocity Obstacles* (HRVO) combined with PID (these options are explained in the next paragraphs). Based on the chosen control strategy, the simulation layer computes the goal positions of the robots, either final positions for PID or intermediate points for HRVO, and sends them to the server. Finally, the server layer dispatches commands to the individual zooids, while at the same time monitoring their status and position.

The control procedure for our system consists of three steps:

- Hungarian goal assignment (optional)
- HRVO global control strategy (optional)
- PID position control.

Before any movement, each robot first needs to be assigned its final position. The final positions may be specific for each robot or they can be dynamically assigned to move in a more efficient manner. The Hungarian algorithm [32], a well-known optimization method for one-to-one task-agent problems, can be used to assign the goal positions to robots in an optimal fashion. The cost function to be optimized is the summation of the squared distances from the initial to the final positions.

After the goal assignment step, robots need to move toward their goals, while minimizing possible collisions with each other robot. We chose to use the HRVO control strategy [67, 68] due to its fast real-time path planning capabilities. With HRVO, a robot moves at the user-defined preferred speed unless it detects possible collisions. In that case, it uses the notion of velocity obstacle, i.e., the set of all robot velocities that will result in a collision with another robot. While HRVO does not guarantee collision-free, oscillation-free control, it reduces the number of collisions dramatically compared to other velocity obstacle strategies while providing real-time updates, essential to natural and smooth user interactions. To implement HRVO, we used a slightly modified version of the HRVO library created by Snape et al. [67, 68].

With the HRVO control strategy, we can derive the incremental goal positions along a path for each robot. These positions are sequentially sent to each robot which independently controls its motion through a PID controller based on the state machine shown in Figure 7. Given a final goal, the robot initially turns itself in the right direction and, once aligned, accelerates to its user-defined preferred speed. When it reaches the speed, it maintains it with a PID control on the orientation to ensure its direction towards the final goal. When a new incremental goal is given, it will still move at same speed but the PID control on orientation will direct the robot towards the new intermediate goal. When the robot arrives within 5 cm of the final goal, it slows down to its minimum velocity and once within 1 cm of the final goal, it stops and orients itself as commanded by the application programmer. To enable smooth transitions between the incremental goal positions, robots are given their next position at 60 Hz.

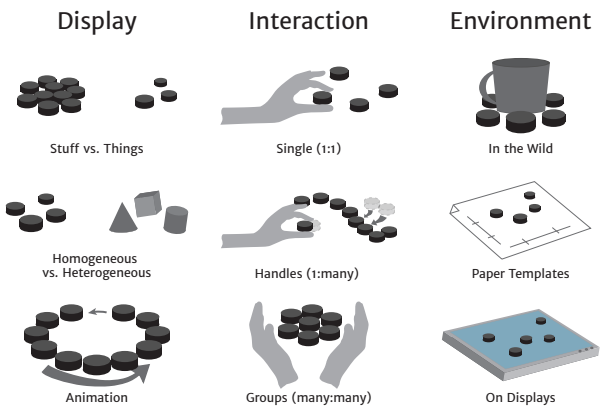

Figure 8. Design Space explored with Zooids.

SWARM UIS: DESIGN PRINCIPLES AND CHALLENGES

Swarm UIs radically change the way we think of user interfaces, not only from an end user's perspective but also from an application designer's perspective. We discuss new concepts, and highlight the major differences here.

Figure 8 gives an overview of the design space of Swarm Interfaces. They can be organized into an *interaction* aspect (interacting with one zooid, controlling many with one zooid, or with groups), a *display* aspect, and an *environment* aspect (operating in a neutral area, in a crowded desk populated with external objects, over a static background layer, or over a dynamic display). We expand on some of these aspects below.

Display: Things vs. Stuff

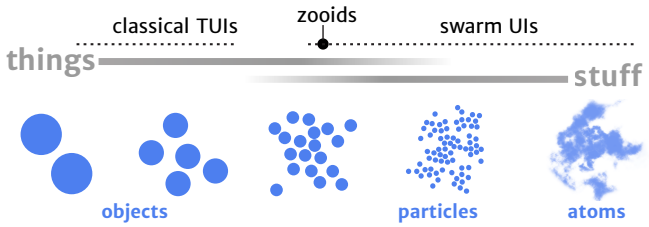

Figure 9. The continuum between "things" and "stuff".

Designing swarm UIs requires thinking both in terms of "things" and of "stuff". In our previous examples, a zooid can stand for an individual object (e.g., a widget) or be part of a larger collection of objects (e.g., a circle). Figure 9 illustrates the continuum between these two paradigms: *things* are physical entities experienced as individual, solid objects; *Stuff* consist in physical entities experienced as shapes and material that can be reshaped, divided, merged, or temporarily solidified to emulate things. The elements making up stuff can be large enough to be visible (particles) or too small to be visible (atoms). Typical TUIs are located to the left of the continuum — they are made of things. In contrast, Swarm UIs occupy the right half of the continuum. As a low-resolution swarm UI implementation, zooids stand in the gray area of the continuum and have both the affordance of things and stuff.

The thing-stuff continuum also applies to traditional graphical displays. Many computer displays from the 80's were very low

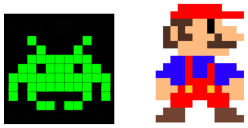

Figure 10. Alien from the game Space Invaders from Taito (1978) and main character from the game Mario Bros by Nintendo (1983).

resolution (semi-graphics from the Sinclair ZX-81 and Tandy TRS-80 were 64×48 pixels), thus pixels were discernible particles much like the zooids in our previous examples (see Figure 10). Now with ultra-high resolution displays pixels became practically invisible, i.e., they became atoms. There are however major conceptual differences between pixel-based displays and swarm UIs, which we discuss next.

Display: Fixed vs. Movable Elements

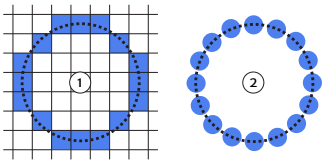

Figure 11. A circle obtained by assembling 16 elements using (1) Bresenham's algorithm and (2) free object positioning.

We are used to program graphics on computer displays where the elements (pixels) are arranged on a regular grid, and only their color is controlled. Although elements of swarm UIs can also have different colors (in our system, each zooid embeds a color LED), a major difference is that they can be positioned freely. Even at equal resolution between the two systems, the way elements can be combined into shapes is very different (see Figure 11). In general, free positioning allows finer shape control than simply turning pixels on and off. At the same time, this extra flexibility comes at the cost of slower response time and higher engineering complexity, with algorithmic problems such as collision avoidance and optimal element-target assignment. In addition, with systems with few elements such as zooids, designers need to think carefully about how to use every zooid optimally, the same way designers from the 80's had to think carefully about how to best use every pixel. It will become less of a concern as the resolution of swarm UIs increases, but on the other hand, engineering and algorithmic challenges will likely become harder. In addition, as shown in Figure 8, the display elements may be homogeneous, as with zooids, or heterogeneous.

Display: Fixed vs. Variable Numbers of Elements

On regular graphical displays the total number of pixels is generally fixed, and the illusion of having more or less content on the screen is achieved by simply manipulating pixel color (e.g., having more or less dark pixels on a white background). In contrast, many swarm applications (e.g, our drawing application) require the number of elements to actually change over time. Zooids cannot be created or destroyed, but as we saw, unassigned zooids can be placed in a dedicated region and moved to the working area whenever they are needed. This type of object persistence contributes to realism and can help

users remain oriented across view changes [7]. As a result, object persistence is often implemented as a metaphor in modern GUIs (e.g., [44]). Swarm UIs support these metaphors natively, and they force designers to think about how to animate appearance and disappearance [7]. However, when true appearance and disappearance are needed, swarm UIs may be impractical and the motions produced by elements constantly arriving and departing can be distracting to end users.

Display: Elements with an Identity vs. Interchangeable Elements

One important distinction to be made is between swarm UI elements that have a fixed identity, and elements that are interchangeable. In general, elements used as "things" have a fixed identity, whereas elements making up "stuff" are interchangeable. For example, in our shape drawing application, the zooids that make up a circle or a line do not have an identity of their own and could be freely swapped. As explained in the implementation section, this makes it possible to optimize the swarm interface so that the motion of zooids remain minimal even during complex transitions. At the same time, swapping a widget (e.g., one of the handles) with another zooid is not desirable, as this might be disorienting to a user, especially if she was about to grasp it. Similarly, in systems where each zooid has a stable meaning (e.g., a visualization system where a zooid represent a data point), swapping zooids can produce confusing or misleading animations. Therefore, the designer of a swarm UI should think carefully about which elements are interchangeable, and which elements should be given a fixed identity. Finally, elements that are manipulated should never be reassigned, which is ensured automatically in our current Zooid implementation.

Interaction: Element Manipulation

Regular graphical displays do not allow pixels to be physically manipulated. Although direct touch displays give a decent illusion of direct manipulation, the subjective experience and the level of expressiveness fall short of true physical object manipulation [75]. In contrast, zooids can be grasped and directly manipulated, allowing to tap into the richness of human hands [34]. For example, in our swarm drawing scenario, users can not only manipulate curves using surrogates such as control points, they can also shape the curves directly. Our system explicitly supports such interactions by registering when a zooid is touched and by constantly updating its goal based on its position. Generally, swarm UI designers should not only focus on the design of "synthetic" interactions, but also consider what is possible in terms of purely physical interactions [28]. Due to their form factor, zooids can be manipulated both as collections of objects (stuff), and as individual objects (things). As swarm UI elements get smaller though, affordances will change dramatically. For example, grains of rice can be manipulated individually, but rice better affords being manipulated as "stuff". While object manipulation is supported natively in systems with large enough elements like ours, future swarm UIs will need to be able to coalesce multiple elements into solid objects to be able to support similar manipulations.

Interaction: Differing Roles of Elements

Different swarm UI elements can be given different roles. For example in our time series visualization application, the top zooids are used for display purposes only, while the bottom ones are used as controllers. On the drawing application, in contrast, zooids are used both for input and output, although different zooids interpret input differently. For example, in our rectangle drawing tool, moving the two control points reshapes the rectangle, while moving any other zooid translates it. Giving different roles to different swarm UI elements allows for more design flexibility, but it also poses the problem of how to convey affordances. In our example applications we assign different LED colors to different functions, but the color mappings are arbitrary and this approach assumes a user who is already familiar with the system. Better affordances could be conveyed by giving different zooids different shapes, consistent with the tradition of TUI design [13]. These different shapes could be clippable, or alternatively, zooids could change their shape dynamically. For a high-resolution swarm UI however, a more natural approach would consist of producing objects of different shapes by assembling many particles or atoms together, as we discussed previously.

Environment: Extra Visual Feedback

Although the drawing application we illustrated is a pure swarm UI, in practice many coarse-grained swarm UIs would need the display of extra visual information (such as text) to be really usable. We illustrated two ways of doing this: one can craft a support surface that contains all the necessary annotations, provided these are stable over time (as in, e.g., board games). When annotations need to change over time, zooids can be placed on a regular graphical display, or alternatively, top projection can be used if optical tracking is in the IR spectrum. Despite the current need for extra display hardware, zooids can convey more visual information on their own than traditional TUIs that only involve a few tangibles as controllers and typically convey most visual information through additional graphical overlays. One can imagine that future swarm UIs will be high-resolution enough to be able to act as displays of their own, thereby entirely eliminating the need for virtual information overlays that suffer from many problems such as (for top projection) occlusion, difficulty of calibration, and difficulty of projecting on shiny or dark surfaces [17].

LIMITATIONS AND FUTURE WORK

There are a number of technical limitations with the Zooids system that limit its capabilities and performance as a swarm user interface. These range from the scale and speed of the device to the cost.

One significant limitation is that our robots have a non-holonomic drive, meaning that they cannot move freely in two-dimensional space and instead must turn to a specific heading like a car. Having a holonomic system with an omni-direction drive would allow the robots to move more smoothly and more easily respond to user interaction. Unlike the case of using robots as displays, where movement paths can be pre-computed [63], our interactive systems may not be able to find a simple or comprehensible path, especially when the movements are over small distances.

Currently, our sensing of input is limited to capacitive touch input on each robot around its circumference. When interacting with many zooids at once, not all touch sensors will be activated, only the ones directly touching a user's hand. Sensor fusion techniques could be used to identify and match the motion between two or more robots that are being moved in unison. This would allow for richer interaction techniques leveraging the direct manipulation of many robots at once.

Another technical limitation of our system is its use of an external projector for tracking. This requirement adds cost and also requires additional hardware and set up to use the system, impeding the scalability of zooids. In addition, like all optical tracking systems, our system is limited by occlusions which may often happen when interacting with the system. Finally, our projector and photodiodes operate in the optical light spectrum, making it hard to use with too much ambient light (this could be improved some with the use of an IR projector and Photodiodes). A number of different approaches could improve our tracking. Using rotating IR laser line beacons, similar to Valve's Vive Lighthouse tracker (http://www.htcvive.com) could significantly reduce the cost, and having multiple beacons could solve some occlusion problems. However, we see great potential in wireless tracking, which could reduce setup to adding a small number of fixed beacons (anchors) for localization with received radio signal strength. Alternatively, future work on improving dead-reckoning location techniques with sensor fusion between wheel encoders and IMUs, coupled with either IR or RSSI anchor free localization between elements, could reduce the need for external tracking completely. We believe that advances in technology will benefit swarm UIs, allowing for more ubiquitous installations.

Power and charging management of many robots presents many challenges. Currently, our system relies on individual chargers in which each robot must be placed manually. An automated system, potentially with integrated wireless charging coils in each robot could allow robots to charge autonomously when needed by returning to a charging base station.

The scale and number of elements in our current system limits the type of interaction and applications that can be created — smaller and more elements may allow for radically different and richer styles of interaction with "stuff" instead of "things". In order to achieve smaller elements we will need to move away from geared DC motors with wheels for locomotion to other actuation, such as piezo actuators. Other micro-robots have been developed which utilize compliant linkages with piezo actuation to create locomotion similar to that of small insects at much smaller scales [65], however power electronics at this scale remain challenging [69]. Another contributing factor which limits the number of robots is cost. Our current robot design at small scales of production is around $50 USD per robot in cost for parts and assembly. This makes swarms larger than 30-40 cost prohibitive outside of research applications. With further design for manufacturing at larger scales, the price per robot could be reduced, but other fabrication techniques such as printable and foldable robots [11] may ultimately enable much cheaper swarm interface systems.

Another large limitation is the interaction area, as well as the type of surfaces on which the elements can move. Since zooids's movement relies on a set of small rubber wheels, our system can only work for relatively flat surfaces with a minimal amount of traction. This limits our system to 2D interactions. Obviously, work on swarms of aerial drones [33] present opportunities to make fully 3D interfaces. However, we see great opportunity in further exploration of ground based swarm interfaces that may be able to reconfigure into 2.5D or even 3D displays. Taking inspiration from ants and other insects, they could form complex shapes by interweaving and connecting, or even rolling on top of each other [8, 59]. We also see great potential for different classes of robots which could help construct more 3D shapes, such as a ramp robot, or other passive building blocks that could allow swarm interfaces to form more complex structures similar to recent work in swarm robotics [79].

Finally, we want to explore more application domains; now that we have created a scalable platform we can explore and quickly prototype. We believe information visualization is an exciting area, especially for creating engagement and for educational domains. It is also important to better understand the benefits and drawbacks of swarm user interfaces compared to traditional GUIs. For this purpose, conducting user studies will identify favorable conditions for the use of swarm user interfaces. We hope that our open source platform with also encourage other researchers, designers, and educators to explore a range of applications, and will enable further evaluation and study of tangible interaction principles.

CONCLUSION

We introduced *swarm user interfaces*, a new class of user interfaces made of "independent self-propelled elements that move collectively and react to user input". We described the technical implementation of *Zooids*, a novel open-source platform for building swarm user interfaces, and illustrated its possibilities through concrete examples. We hope that this article and the Zooids platform will spur more research in swarm user interfaces, and bring us closer to Sutherland and Ishii's visions of the ultimate user interface that is able to fully combine human capabilities for physical manipulation with the power of computing.

All necessary material and documentation for implementing Zooids can be found at

https://github.com/PhysicalInteractionLab/SwarmUI/.

ACKNOWLEDGMENTS

This work was partially funded by the Région Ile de France, DIM ISC-PIF. We would also like to thank Alexa Siu, Shenli Yuan, Ernesto Ramirez and Pham Minh Hieu for investing so much time and efforts.

REFERENCES

1. Ahlberg, C., Williamson, C., and Shneiderman, B. Dynamic queries for information exploration: An implementation and evaluation. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM (1992), 619–626.

2. Alonso-Mora, J., Breitenmoser, A., Rufli, M., Siegwart, R., and Beardsley, P. Multi-robot system for artistic pattern formation. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, IEEE (2011), 4512–4517.
3. Alonso-Mora, J., Lohaus, S. H., Leemann, P., Siegwart, R., and Beardsley, P. Gesture based human-multi-robot swarm interaction and its application to an interactive display. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE (2015), 5948–5953.
4. Amano, K., and Yamamoto, A. Tangible interactions on a flat panel display using actuated paper sheets. In *Proceedings of the 2012 ACM international conference on Interactive tabletops and surfaces*, ACM (2012), 351–354.
5. Bennett, E., and Stevens, B. The effect that touching a projection augmented model has on object-presence. In *Information Visualisation, 2005. Proceedings. Ninth International Conference on*, IEEE (2005), 790–795.
6. Brave, S., Ishii, H., and Dahley, A. Tangible interfaces for remote collaboration and communication. In *Proceedings of the 1998 ACM conference on Computer supported cooperative work*, ACM (1998), 169–178.
7. Chang, B.-W., and Ungar, D. Animation: from cartoons to the user interface.
8. Cucu, L., Rubenstein, M., and Nagpal, R. Towards self-assembled structures with mobile climbing robots. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, IEEE (2015), 1955–1961.
9. Ducatelle, F., Di Caro, G., Pinciroli, C., and Gambardella, L. Self-organized cooperation between robotic swarms. *Swarm Intelligence Journal 5*, 2 (2011), 73–96.
10. Dudek, G., Jenkin, M., Milios, E., and Wilkes, D. A taxonomy for swarm robots. In *Intelligent Robots and Systems' 93, IROS'93. Proceedings of the 1993 IEEE/RSJ International Conference on*, vol. 1, IEEE (1993), 441–447.
11. Felton, S. M., Tolley, M. T., Onal, C. D., Rus, D., and Wood, R. J. Robot self-assembly by folding: A printed inchworm robot. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, IEEE (2013), 277–282.
12. Fishkin, K. P. A taxonomy for and analysis of tangible interfaces. *Personal Ubiquitous Comput. 8* (September 2004), 347–358.
13. Fitzmaurice, G. W., and Buxton, W. An empirical evaluation of graspable user interfaces: towards specialized, space-multiplexed input. In *Proc. CHI 1997*, 43–50.
14. Follmer, S., Leithinger, D., Olwal, A., Cheng, N., and Ishii, H. Jamming User Interfaces: Programmable Particle Stiffness and Sensing for Malleable and Shape-Changing Devices. In *ACM Symposium on User Interface Software and Technology* (2012), 519–528.
15. Follmer, S., Leithinger, D., Olwal, A., Hogge, A., and Ishii, H. inform: Dynamic physical affordances and constraints through shape and object actuation. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology*, UIST '13, ACM (New York, NY, USA, 2013), 417–426.
16. Futurelab, A. E. Drone 100 – the world record for intel 2015. http://tinyurl.com/drone100, 2016.
17. Gervais, R. *Interaction and introspection with tangible augmented objects*. Phd dissertation, Université de Bordeaux, Dec. 2015.
18. Goldstein, S. C., Campbell, J. D., and Mowry, T. C. Programmable matter. *Computer 38*, 6 (2005), 99–101.
19. Greenberg, S., and Fitchett, C. Phidgets: easy development of physical interfaces through physical widgets. In *Proceedings of the 14th annual ACM symposium on User interface software and technology*, ACM (2001), 209–218.
20. Grieder, R., Alonso-Mora, J., Bloechlinger, C., Siegwart, R., and Beardsley, P. Multi-robot control and interaction with a hand-held tablet. In *Workshop Proc. Int. Conf. Robotics and Automation*, vol. 131, Citeseer (2014).
21. Hauri, S., Alonso-Mora, J., Breitenmoser, A., Siegwart, R., and Beardsley, P. Multi-robot formation control via a real-time drawing interface. In *Field and Service Robotics*, Springer (2014), 175–189.
22. Heer, J., and Robertson, G. G. Animated transitions in statistical data graphics. *Visualization and Computer Graphics, IEEE Transactions on 13*, 6 (2007), 1240–1247.
23. Horn, M. S., Solovey, E. T., Crouser, R. J., and Jacob, R. J. Comparing the use of tangible and graphical programming languages for informal science education. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, 975–984.
24. Hornecker, E., and Buur, J. Getting a grip on tangible interaction: a framework on physical space and social interaction. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, ACM (2006), 437–446.
25. Huron, S., Jansen, Y., and Carpendale, S. Constructing visual representations: Investigating the use of tangible tokens. *Visualization and Computer Graphics, IEEE Transactions on 20*, 12 (2014), 2102–2111.
26. Ishii, H., Lakatos, D., Bonanni, L., and Labrune, J.-B. Radical atoms: beyond tangible bits, toward transformable materials. *interactions 19*, 1 (Jan. 2012), 38–51.
27. Jansen, Y., Dragicevic, P., and Fekete, J.-D. Evaluating the efficiency of physical visualizations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM (2013), 2593–2602.

28. Jansen, Y., Dragicevic, P., Isenberg, P., Alexander, J., Karnik, A., Kildal, J., Subramanian, S., and Hornbæk, K. Opportunities and challenges for data physicalization. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, ACM (2015), 3227–3236.

29. Kira, Z., and Potter, M. A. Exerting human control over decentralized robot swarms. In *Autonomous Robots and Agents, 2009. ICARA 2009. 4th International Conference on*, IEEE (2009), 566–571.

30. Kojima, M., Sugimoto, M., Nakaruma, A., Tomita, M., Inami, M., and Nii, H. Augmented coliseum: An augmented game environment with small vehicles. *Horizontal Interactive Human-Computer Systems, International Workshop on 0* (2006), 3–8.

31. Kolling, A., Nunnally, S., and Lewis, M. Towards human control of robot swarms. In *Proceedings of the seventh annual ACM/IEEE international conference on human-robot interaction*, ACM (2012), 89–96.

32. Kuhn, H. W. The hungarian method for the assignment problem. *Naval research logistics quarterly 2*, 1-2 (1955), 83–97.

33. Kushleyev, A., Mellinger, D., Powers, C., and Kumar, V. Towards a swarm of agile micro quadrotors. *Autonomous Robots 35*, 4 (2013), 287–300.

34. Le Goc, M., Dragicevic, P., Huron, S., Boy, J., and Fekete, J.-D. Smarttokens: Embedding motion and grip sensing in small tangible objects. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, ACM (2015), 357–362.

35. Le Goc, M., Dragicevic, P., Huron, S., Boy, J., and Fekete, J.-D. A better grasp on pictures under glass: Comparing touch and tangible object manipulation using physical proxies. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, ACM (2016), 76–83.

36. Lederman, S. J., and Campbell, J. I. Tangible graphs for the blind. *Human Factors: The Journal of the Human Factors and Ergonomics Society 24*, 1 (1982), 85–100.

37. Lee, J. C., Hudson, S. E., Summet, J. W., and Dietz, P. H. Moveable interactive projected displays using projector based tracking. In *Proceedings of the 18th annual ACM symposium on User interface software and technology*, ACM (2005), 63–72.

38. Lee, N., Kim, J., Lee, J., Shin, M., and Lee, W. Molebot: mole in a table. In *ACM SIGGRAPH 2011 Emerging Technologies*, ACM (2011), 9.

39. Leithinger, D., and Ishii, H. Relief: a scalable actuated shape display. In *Proceedings of the fourth international conference on Tangible, embedded, and embodied interaction*, ACM (2010), 221–222.

40. Lifton, J., Broxton, M., and Paradiso, J. A. Experiences and directions in pushpin computing. In *IPSN 2005. Fourth International Symposium on Information Processing in Sensor Networks, 2005.*, IEEE (2005), 416–421.

41. Marquardt, N., Nacenta, M. A., Young, J. E., Carpendale, S., Greenberg, S., and Sharlin, E. The haptic tabletop puck: tactile feedback for interactive tabletops. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*, ACM (2009), 85–92.

42. Marshall, M., Carter, T., Alexander, J., and Subramanian, S. Ultra-tangibles: creating movable tangible objects on interactive tables. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM (2012), 2185–2188.

43. Mi, H., and Sugimoto, M. Hats: interact using height-adjustable tangibles in tabletop interfaces. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*, ACM (2011), 71–74.

44. Microsoft. Sanddance: Visually explore, understand, and present data. Online. http://research.microsoft.com/en-us/projects/sanddance/, 2016.

45. Moere, A. V. Beyond the tyranny of the pixel: Exploring the physicality of information visualization. In *Information Visualisation, 2008. IV'08. 12th International Conference*, IEEE (2008), 469–474.

46. Nielsen, J. Usability engineering.

47. Nowacka, D., Ladha, K., Hammerla, N. Y., Jackson, D., Ladha, C., Rukzio, E., and Olivier, P. Touchbugs: Actuated tangibles on multi-touch tables. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM (2013), 759–762.

48. Pangaro, G., Maynes-Aminzade, D., and Ishii, H. The actuated workbench: Computer-controlled actuation in tabletop tangible interfaces. In *Proceedings of the 15th Annual ACM Symposium on User Interface Software and Technology*, UIST '02, 181–190.

49. Patten, J. Thumbles - robotic tabletop user interface platform. *TED.com* (2014).

50. Patten, J., and Ishii, H. Mechanical constraints as computational constraints in tabletop tangible interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '07, ACM (New York, NY, USA, 2007), 809–818.

51. Patten, J., Ishii, H., Hines, J., and Pangaro, G. Sensetable: A wireless object tracking platform for tangible user interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '01, ACM (New York, NY, USA, 2001), 253–260.

52. Patten, J., Recht, B., and Ishii, H. Audiopad: A tag-based interface for musical performance. In *Proceedings of the 2002 Conference on New Interfaces for Musical Expression*, NIME '02, National University of Singapore (Singapore, Singapore, 2002), 1–6.

53. Pedersen, E. W., and Hornbæk, K. Tangible bots: interaction with active tangibles in tabletop interfaces. In *Proc. CHI*, ACM (2011), 2975–2984.

54. Poupyrev, I., Nashida, T., Maruyama, S., Rekimoto, J., and Yamaji, Y. Lumen: Interactive visual and shape display for calm computing. In *ACM SIGGRAPH 2004 Emerging Technologies*, SIGGRAPH '04, ACM (New York, NY, USA, 2004), 17–.

55. Poupyrev, I., Nashida, T., and Okabe, M. Actuation and tangible user interfaces: the vaucanson duck, robots, and shape displays. In *TEI '07*, 205–212.

56. Rasmussen, M. K., Pedersen, E. W., Petersen, M. G., and Hornbaek, K. Shape-changing interfaces: a review of the design space and open research questions. In *CHI '12*, 735–744.

57. Reznik, D., and Canny, J. A flat rigid plate is a universal planar manipulator. In *IEEE ICRA 1998*, vol. 2, IEEE (1998), 1471–1477.

58. Richter, J., Thomas, B. H., Sugimoto, M., and Inami, M. Remote active tangible interactions. In *Proceedings of the 1st international conference on Tangible and embedded interaction*, ACM (2007), 39–42.

59. Romanishin, J. W., Gilpin, K., and Rus, D. M-blocks: Momentum-driven, magnetic modular robots. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, IEEE (2013), 4288–4295.

60. Rosenfeld, D., Zawadzki, M., Sudol, J., and Perlin, K. Physical objects as bidirectional user interface elements. *Computer Graphics and Applications, IEEE 24*, 1 (2004), 44–49.

61. Roudaut, A., Karnik, A., Löchtefeld, M., and Subramanian, S. Morphees: toward high shape resolution in self-actuated flexible mobile devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM (2013), 593–602.

62. Rubens, C., Braley, S., Gomes, A., Goc, D., Zhang, X., Carrascal, J. P., and Vertegaal, R. Bitdrones: Towards levitating programmable matter using interactive 3d quadcopter displays. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, ACM (2015), 57–58.

63. Rubenstein, M., Ahler, C., and Nagpal, R. Kilobot: A low cost scalable robot system for collective behaviors. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, IEEE (2012), 3293–3298.

64. Rus, D. Programmable matter with self-reconfiguring robots. In *Proceedings of the 7th ACM international conference on Computing frontiers*, CF '10, 51–52.

65. Sahai, R., Avadhanula, S., Groff, R., Steltz, E., Wood, R., and Fearing, R. S. Towards a 3g crawling robot through the integration of microrobot technologies. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, IEEE (2006), 296–302.

66. Seah, S. A., Drinkwater, B. W., Carter, T., Malkin, R., and Subramanian, S. Dexterous ultrasonic levitation of millimeter-sized objects in air. *IEEE transactions on ultrasonics, ferroelectrics, and frequency control 61*, 7 (2014), 1233–1236.

67. Snape, J., van den Berg, J., Guy, S. J., and Manocha, D. The hybrid reciprocal velocity obstacle. *Robotics, IEEE Transactions on 27*, 4 (2011), 696–706.

68. Snape, J., van den Berg, J. P., Guy, S. J., and Manocha, D. Independent navigation of multiple mobile robots with hybrid reciprocal velocity obstacles. In *IROS* (2009), 5917–5922.

69. Steltz, E., Seeman, M., Avadhanula, S., and Fearing, R. S. Power electronics design choice for piezoelectric microrobots. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, IEEE (2006), 1322–1328.

70. Sutherland, I. E. The ultimate display, 1965.

71. Taher, F., Hardy, J., Karnik, A., Weichel, C., Jansen, Y., Hornbæk, K., and Alexander, J. Exploring interactions with physically dynamic bar charts. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, ACM (2015), 3237–3246.

72. Ullmer, B., and Ishii, H. The metadesk: Models and prototypes for tangible user interfaces. In *Proceedings of the 10th Annual ACM Symposium on User Interface Software and Technology*, UIST '97, ACM (New York, NY, USA, 1997), 223–232.

73. Ullmer, B., Ishii, H., and Jacob, R. J. K. Token+constraint systems for tangible interaction with digital information. *ACM Trans. Comput.-Hum. Interact. 12*, 1 (Mar. 2005), 81–118.

74. Underkoffler, J., and Ishii, H. Urp: A luminous-tangible workbench for urban planning and design. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '99, 386–393.

75. Victor, B. A brief rant on the future of interaction design. http://tinyurl.com/bvrant, 2011.

76. Wakita, A., Nakano, A., and Kobayashi, N. Programmable blobs: a rheologic interface for organic shape design. In *Proceedings of the fifth international conference on Tangible, embedded, and embodied interaction*, ACM (2011), 273–276.

77. Weiser, M. Some computer science issues in ubiquitous computing. *Communications of the ACM 36*, 7 (1993), 75–84.

78. Weiss, M., Schwarz, F., Jakubowski, S., and Borchers, J. Madgets: Actuating widgets on interactive tabletops. In *Proceedings of the 23Nd Annual ACM Symposium on User Interface Software and Technology*, UIST '10, 293–302.

79. Werfel, J., Petersen, K., and Nagpal, R. Designing collective behavior in a termite-inspired robot construction team. *Science 343*, 6172 (2014), 754–758.

80. Yamamoto, A., Tsuruta, S., and Higuchi, T. Planar 3-dof paper sheet manipulation using electrostatic induction. In *Industrial Electronics (ISIE), 2010 IEEE International Symposium on*, IEEE (2010), 493–498.

81. Yamanaka, S., and Miyashita, H. Vibkinesis: notification by direct tap and'dying message'using vibronic movement controllable smartphones. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*, ACM (2014), 535–540.

82. Yao, L., Niiyama, R., Ou, J., Follmer, S., Della Silva, C., and Ishii, H. Pneui: Pneumatically actuated soft composite materials for shape changing interfaces. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology*, UIST '13, 13–22.

83. Yi, J. S., Melton, R., Stasko, J., and Jacko, J. A. Dust & magnet: multivariate information visualization using a magnet metaphor. *Information Visualization 4*, 4 (2005), 239–256.

84. Zhao, J., and Moere, A. V. Embodiment in data sculpture: a model of the physical visualization of information. In *Proceedings of the 3rd international conference on Digital Interactive Media in Entertainment and Arts*, ACM (2008), 343–350.