An artificial potential field approach for velocity control and object manipulation on shape displays

Brian K. Johnson, J. Sean Humbert, Mark E. Rentschler

***Abstract*—Shape displays, devices which can actively alter their surface topology, have promising applications for human-robot-collaboration, haptic interaction, and manufacturing. However, these use-cases are limited due to existing challenges in dynamic object control on the shape display surface. We propose a general-purpose, extendable potential field algorithm that enables complete trajectory control of spherical objects on shape displays via surface rolling. The algorithm uses two potential field layers to control object velocity and energy, driving motion through a potential to kinetic energy transfer. We extend the base-case algorithm to enable the control of multiple agents on separate trajectories with object avoidance, as well as cooperative formation control. We quantitatively and experimentally evaluate the capabilities of this approach on a shape display driven by a 10x10 actuator array, demonstrating ball accelerations from rest within 0.5 seconds, high directional accuracy of velocity, and control stability in the presence of obstacles and limit-cycle trajectories. We also demonstrate Lyapunov stability of a three-ball formation which can be formed within 2.5 seconds. The qualitative and quantitative results suggest that the application of potential fields for spatial and velocity control is highly effective on deformable shape display systems.**

***Note to Practitioners*—The algorithm we propose and experimentally demonstrate in this article allows an actuated shape display to control spherical ball motion with high precision even on a small-scale 10x10 array device, with expected increased performance for higher-resolution devices in haptic or manufacturing settings. The demonstrations are run in real time with a 26 Hz feedback loop using 260 Hz filtered image input from a USB camera; better camera quality and loop frequencies should also lead to improved performance. Since the generalized algorithm that we present can be applied to many kinds of shape display systems, we express experimental results in a normalized format to allow for performance predictions at different length scales.**

***Index Terms*—Soft sensors and actuators; sensor-based control.**

I. Introduction

SHAPE displays, shape-changing interfaces, or shape-morphing surfaces are robotic systems which locally change their physical topology along discrete or continuous points of the surface [1], [2]. A common method of changing the surface topology is by using an array of linear actuators which either directly comprise the surface or interact with a deforming surface layer [3]–[10]. These deformable surfaces can be used as visual information displays [8], [10], [11], tele-operative or haptic devices [3], [8], [9], [12], or for dynamic motion of objects on the surface such as object conveyance or fluid mixing [3]–[7], [10], [13], [14]. A common demonstration is to apply shape deformation to roll a ball, showing a use-case for sorting fruits, vegetables, or consumer items. This type of manipulation has been demonstrated on both rigid and soft shape displays [3], [4], [10].

However, control strategies to achieve object rolling are largely unexplored to date. Existing works achieve object manipulation through open-loop pre-programming [5], neural networks [6], undisclosed algorithms [3], [4], or position-based feedback [10]. Most existing schemes [3], [4], [6] are relatively slow and limited to single point position setting (e.g. moving a ball from one position on the surface to another). However, a robust, simple algorithm that describes the complete trajectory of a ball on the surface is currently not present in literature.

We propose a closed-loop algorithm for ball velocity and trajectory based on artificial potential field (APF) theory. APF theory, first proposed by O. Khatib as a means for mobile robots to navigate through their environment [15], has become widely popular in robotic control systems [16], [17]. In APF theory, robot goals in the environment are mapped to negatively-valued potential fields while obstacles or avoidance points are mapped to positively-valued fields; the achieved robot path-planning is the result of a simple gradient descent from the robot's position to the goal. APFs allow for built-in obstacle avoidance [18]–[20], trajectory planning [16], [21]–[25], and multi-agent cooperation [23], [26]–[31] with minimal computational overhead. However, a common problem in APF applications is dead-lock, which occurs when the robot follows the potential field gradient into local rather than global minima [17], [24], [32], [33].

We apply the mathematical simplicity of APF theory by reframing the problem of object manipulation to that of robotic navigation: rather than the shape display manipulating the object on the surface, the object is considered as an agent navigating through its environment with control inputs provided by the shape display topology. Our APF algorithm applies two separate field layers: the first is a purely attractive APF which forms the ball's desired velocity gradient at each point along the surface, and the second is a purely repulsive potential energy field realized via deformation of the surface. The separation of the APF layers prevents dead-lock for positional control. The algorithm also easily extends to global trajectory control by shaping the velocity gradient APF to the desired trajectory, and extends to multiple balls as well, enabling enable multi-agent motion and shape formation control.

The paper is organized as follows. In Section II, we discuss the general dynamics of a ball rolling on a shape display surface and describe the basic case of our two-layer-APF algorithm: positional control of a single agent. We also formu-

This work was supported by the National Science Foundation under Grants 1739452 and 1650115. (*Corresponding author: Brian K. Johnson*)

B. K. Johnson, J. S. Humbert, M. E. Rentschler are with the Paul M. Rady Deptartment of Mechanical Engineering, University of Colorado Boulder, Boulder, CO 80309 USA (e-mail: brian.k.johnson@colorado.edu).

M.E. Rentschler is a IEEE Senior Member.

late the functional extensions of the algorithm for multi-agent and trajectory control. Section III describes our experimental hardware and software to validate the algorithm's performance. We detail the experiments and introduce a new method of performance quantification. In Section IV we discuss the experiment results of four capabilities introduced by our proposed algorithm: (i) point-to-point ball position control, (ii) trajectory control, (iii) multi-agent expansions of (i) and (ii), and finally (iv) multi-agent formation control. Lastly, Section V presents the key results and conclusions of this work.

II. Potential Field Algorithm

A. Ball dynamics on a deforming surface

The motion of a ball on a deforming surface is described by energy gradients and energy conversion. If the local surface under the ball deforms upward (raising the ball), the ball's potential energy V increases by

$$V_{ball} = Mgh_{ball} \tag{1}$$

with ball mass M, gravitational acceleration constant g, and change of ball height h_{ball}. If the ball rolls down the slope of the newly deformed surface, the potential energy converts into both linear and rotational kinetic energy. The sum of kinetic energy T is

$$V_{ball} \rightarrow T_{ball} = \frac{1}{2}M\|\dot{\boldsymbol{x}}_{ball}\|^2 + \frac{1}{2}I(\|\dot{\boldsymbol{x}}_{ball}\|/r)^2 \tag{2}$$

where $\dot{\boldsymbol{x}}_{ball}$ is the planar velocity of the ball for $\boldsymbol{x}_{ball} \in \mathbb{R}^2$ (no vertical component), $\|\cdot\|$ denotes the 2-norm, I is the ball moment of inertia, and r is the ball radius. For this analysis, we assume a hollow sphere with $I = (2/3)Mr^2$ (as is the case with our experimental validation discussed in Section III). The velocity $\dot{\boldsymbol{x}}_{ball}$ increases in the direction of the potential energy gradient:

$$\hat{\dot{\boldsymbol{x}}}_{ball} = -\frac{\nabla V_{ball}}{\|\nabla V_{ball}\|} \tag{3}$$

where $\hat{\dot{\boldsymbol{x}}}_{ball}$ indicates the normalized ball velocity. We also assume a no-slip condition between the ball and surface. The generation of horizontal velocity of an object via potential energy conversion is the governing concept of our control algorithm for the shape display, and it has in practice been demonstrated on existing shape displays [4], [6], [10].

B. Artificial Potential Fields (APFs)

Our algorithm uses two potential fields: The first is a Kinetic Energy APF (KPF) and the second is a Potential Energy PF (PPF). The KPF is part of the outer control loop which dictates ball velocity and therefore trajectory, while the PPF in the inner loop controls active surface deformation to achieve the desired velocities of the KPF. Following Khatib [15], our general approach is to use the negative gradient of the potential field to control the velocity of a ball along the shape display surface. The steps of the algorithm are outlined in Figure 1.

1) KPF for positional control: We construct an APF based on kinetic energy whose gradient gives desired agent velocity, an approach which has been used in prior works [21], [22] for mobile robot navigation. For positional control (i.e. setting a target position/goal on the surface for the ball), we define the KPF as a quadratic attractor:

$$KPF = a_v\|\boldsymbol{x}_{ball} - \boldsymbol{x}_{goal}\|^2 \tag{4}$$

where a_v is a constant scaling factor and $\boldsymbol{x}_{goal}$ is the [x, y] surface position of the goal. Taking the negative gradient gives the desired ball velocity $\dot{\boldsymbol{x}}_{des}$:

$$\dot{\boldsymbol{x}}_{des} = -\nabla KPF = -2a_v(\boldsymbol{x}_{ball} - \boldsymbol{x}_{goal}). \tag{5}$$

This desired velocity determines the required velocity vector $\dot{\boldsymbol{x}}_{req}$:

$$\dot{\boldsymbol{x}}_{req} = \dot{\boldsymbol{x}}_{des} - \dot{\boldsymbol{x}}_{ball}. \tag{6}$$

Adding the required velocity to the current velocity achieves the desired velocity for the KPF. The magnitude $\|\dot{\boldsymbol{x}}_{req}\|$ and normalized direction $\hat{\dot{\boldsymbol{x}}}_{req}$ of the required velocity vector are the inputs to the PPF.

2) PPF to generate velocity: The PPF is a single repulsive potential function in the form of a 2D Gaussian which describes the desired potential energy across the deformable surface. Following the dynamics derived in Section II-A, the PPF represents both the energy input to the ball and the desired topology of the shape display.

The Gaussian repulsor magnitude and position are calculated to generate the PPF. Combining Eqs. (1, 2) gives the relationship between the desired height of the surface deformation and the resulting ball velocity $\dot{\boldsymbol{x}}_{req}$:

$$\begin{aligned} V_{ball} = Mgh_{req} &= \frac{1}{2}M\|\dot{\boldsymbol{x}}_{req}\|^2 + \frac{1}{2}I(\|\dot{\boldsymbol{x}}_{req}\|/r)^2 = T_{ball} \\ Mgh_{req} &= \frac{1}{2}M\|\dot{\boldsymbol{x}}_{req}\|^2 + \frac{2}{6}M\|\dot{\boldsymbol{x}}_{req}\|^2 \\ h_{req} &= \frac{5}{6g}\|\dot{\boldsymbol{x}}_{req}\|^2. \end{aligned} \tag{7}$$

Note that mass drops out of the equations. However, on a soft deformable surface, higher mass will correspond to increased surface friction and frictional energy losses [34]; due to the complexity of predicting these effects, we neglect them in this analysis. We define the standard 2D Gaussian function as the PPF:

$$PPF(\boldsymbol{x}_{surf}) = h(\boldsymbol{x}_{surf}) \tag{8}$$

$$= h_{rep}\text{exp}\left(\frac{-1}{2\sigma^2}\|\boldsymbol{x}_{surf} - \boldsymbol{x}_{rep}\|^2\right), \tag{9}$$

where $\boldsymbol{x}_{surf} \in \mathbb{R}^2$ spans the surface, h_{rep} is the peak height of the repulsor, σ^2 is the repulsor variance (equal in both dimensions), and $\boldsymbol{x}_{rep} \in \mathbb{R}^2$ is the repulsor peak position. We enforce the following relationship between the required height and the peak height of the repulsor:

$$h_{rep} = \frac{3}{2}h_{req}. \tag{10}$$

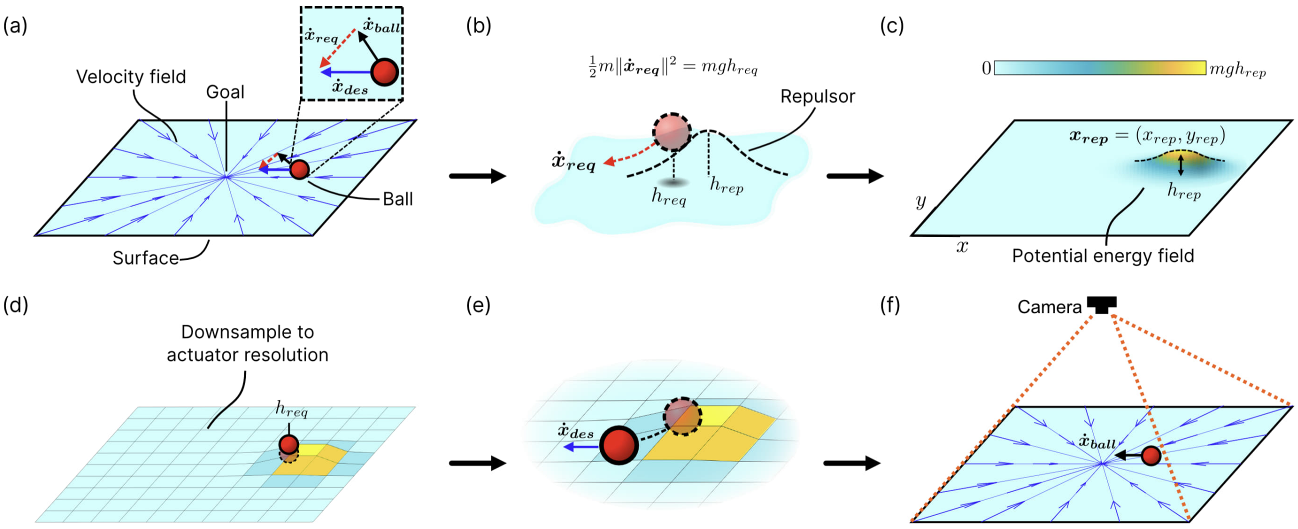

(a)
Velocity field
Goal
$\dot{x}_{req}$
$\dot{x}_{ball}$
$\dot{x}_{des}$
Ball
Surface
(b)
$\frac{1}{2}m\|\dot{\boldsymbol{x}}_{\boldsymbol{req}}\|^2 = mgh_{req}$
Repulsor
$\dot{\boldsymbol{x}}_{\boldsymbol{req}}$
h_{req}
h_{rep}
(c)
0
mgh_{rep}
$\boldsymbol{x}_{\boldsymbol{rep}} = (x_{rep}, y_{rep})$
y
x
h_{rep}
Potential energy field
(d)
Downsample to actuator resolution
h_{req}
(e)
$\dot{\boldsymbol{x}}_{\boldsymbol{des}}$
(f)
Camera
$\dot{\boldsymbol{x}}_{ball}$

Fig. 1. Positional control using the two-layer APF algorithm. **(a)** The Kinetic Artificial Potential Field (KPF) determines the desired and required ball velocity at each point on the surface ($\dot{\boldsymbol{x}}_{\boldsymbol{des}}$, $\dot{\boldsymbol{x}}_{\boldsymbol{req}}$). **(b)** A hypothetical 2D Gaussian repulsor with height h_{rep} is generated to increase ball velocity by $\dot{\boldsymbol{x}}_{\boldsymbol{req}}$. **(c)** The position $\boldsymbol{x}_{\boldsymbol{rep}}$ of the repulsor forms the global Potential Energy Potential Field (PPF). **(d)** The PPF is down-sampled to the surface actuator resolution; in this case, a 10x10 actuator array. Actuation of the surface lifts the ball by height h_{req}. **(e)** The ball rolls down the gradient of the repulsor, and it's velocity increases by $\dot{\boldsymbol{x}}_{\boldsymbol{req}}$ to achieve the total velocity $\dot{\boldsymbol{x}}_{\boldsymbol{des}}$. **(f)** The surface is returned to its undeformed shape, and an overhead camera measures the new ball position and velocity. The algorithm repeats from step (a).

This ensures the ball is not positioned at the unstable peak. Using h_{req} and $\hat{\dot{x}}_{req}$ as inputs, we calculate the position of the repulsor $\boldsymbol{x}_{rep}$ via

$$\boldsymbol{x}_{rep} = \boldsymbol{x}_{ball} - \left[-2\sigma^2 \ln(h_{req}/13)\right]^{1/2} \hat{\dot{\boldsymbol{x}}}_{req} + \alpha \dot{\boldsymbol{x}}_{ball} \quad (11)$$

where α is a scaling factor. We offset the repulsor from the ball position by the $[\cdot]^{1/2}$ term so $h(\boldsymbol{x}_{ball}) = h_{req}$, and by the $\alpha\dot{\boldsymbol{x}}_{ball}$ term to account for time delay between the algorithm output and actuating the physical surface. α depends on the closed-loop control frequency and the response time of the shape display being used.

To be implemented in a real-world system, Eq. (9) must be sampled at the resolution of the given shape display, and the shape display control system must be able to achieve the given h_{rep} value.

C. Closed loop algorithm summary

Figure 1 outlines the algorithm. The full procedure is outlined in Algorithm 1.

Although the algorithm uses a 2D Gaussian function to build the PPF, other PPF forms could also be implemented by replacing Eq. (9). For example, we might achieve similar control by inverting the surface topology; the 2D Gaussian thus forming a valley rather than a peak. While the control stability of the inverted Gaussian would be increased, it would likely be a more energy-intensive control strategy, as most shape displays require power to hold positive deformations.

D. Stability analysis

We analyze stability of the algorithm in a BIBO (bounded input bounded output) sense by defining the storage function of the undeformed surface at time t_0 as

Algorithm 1 Two-layer APF velocity control.

```
while active do
    Measure $\boldsymbol{x}_{ball}$ and $\dot{\boldsymbol{x}}_{ball}$
    Calculate KPF $\dot{\boldsymbol{x}}_{des}$ (Eq. (5))
    Calculate $\dot{\boldsymbol{x}}_{req}$ (Eq. (6))
    Calculate PPF magnitude $h_{rep}$ (Eqs. (7, 10))
    Calculate PPF position $x_{rep}$ (Eq. (11))
    Sample shape display surface coordinates $\boldsymbol{x}_{surf}$
    (specific to device)
    Calculate pixel height across $\boldsymbol{x}_{surf}$ (Eq. 9)
    Apply height commands to each actuator
    (specific to device)
end while
```

$$E(t_0) = V(t_0) + T(t_0) \quad (12)$$

where V and T are the total potential and kinetic energies of the ball on the surface, respectively. When the surface deforms according to the PPF function at some time t_k, we impart some potential energy into the storage function: $V(t_k) - V(t_0) = \Delta V(t_k) > 0$. $\Delta V(t_k)$ is bounded because the displacement of the shape display has a finite limit. We assume that this does not change the kinetic energy of the ball, i.e. $\Delta T(t_k) = 0$. The storage function at time t_k becomes

$$\begin{aligned} E(t_k) &= \Delta V(t_k) + V(t_0) + \Delta T(t_k) + T(t_0) \\ &= \Delta V(t_k) + E(t_0). \end{aligned} \quad (13)$$

As the ball rolls down the PPF gradient, the increased potential energy is converted to kinetic energy. That is, at time t_{k+1},

$$\Delta T(t_{k+1}) = -\Delta V(t_{k+1}), \tag{14}$$

and

$$\begin{aligned} E(t_{k+1}) &= \Delta V(t_{k+1}) + V(t_k) + \Delta T(t_{k+1}) + T(t_k) \\ &= E(t_k). \end{aligned} \tag{15}$$

The energy conversion does not increase the overall energy of the system. Brearley et al. showed that the deformation of a surface due to the ball's mass induces frictional energy losses [34]. A deformable surface without continued inputs is thus dissipative with respect to objects on the surface. We therefore expect that if no additional inputs are provided, $\dot{E}(t) < 0$ for $t > t_{k+1}$ and thus the system is BIBO stable.

The algorithm is also free of dead-lock for positional control. By separating the attractive (velocity) and repulsive (height) features into separate potential fields, we create a trivial case with only one global minimum and no local minima, thus avoiding dead-lock condition.

E. Trajectory control

Thus far, the KPF is formed by a quadratic attractor at an agent's goal position (Fig. 2 (a)). However, one can also directly specify the KPF to create arbitrary velocity vector fields across the surface.

For example, given the vector field equation of a stable limit cycle with radius r, scaling factor γ, centered on $(0, 0)$:

$$f\left(\begin{bmatrix} x \\ y \end{bmatrix}\right) = \gamma \begin{bmatrix} -y + x(r^2 - x^2 - y^2 - 1) \\ x + y(r^2 - x^2 - y^2 - 1) \end{bmatrix}, \tag{16}$$

the KPF gradient can be replaced with this function to generate a circular trajectory for the ball. We can offset the (x, y) positions by (x_c, y_c) to shift the center position of the limit cycle:

$$x_d = \boldsymbol{x}_{ball}(1) - x_c, \; y_d = \boldsymbol{x}_{ball}(2) - y_c. \tag{17}$$

Rather than an infinitesimally narrow stable orbit of the limit cycle, we also choose an inner radius r_i and outer radius r_o to form a stable orbit (Fig. 2 (b)). The resulting desired velocity from the KPF is

$$\dot{\boldsymbol{x}}_{des} = \begin{cases} \gamma \begin{bmatrix} -x_d + x_d(v_i - 1) \\ x_d + y_d(v_i - 1) \end{bmatrix}, & v_i > 0 \\ \gamma \begin{bmatrix} -x_d + x_d(v_o - 1) \\ x_d + y_d(v_o - 1) \end{bmatrix}, & v_o < 0 \\ \gamma \begin{bmatrix} -y_d \\ x_d \end{bmatrix}, & \text{otherwise} \end{cases} \tag{18}$$

where

$$\begin{aligned} v_i &= r_i^2 - x_d^2 - y_d^2, \\ v_o &= r_o^2 - x_d^2 - y_d^2. \end{aligned}$$

By control of the PPF, the ball following the vector fields according to Eq. (18) will orbit around the center (x_c, y_c) at a radius between r_i and r_o. The speed at which the ball orbits is controlled by γ.

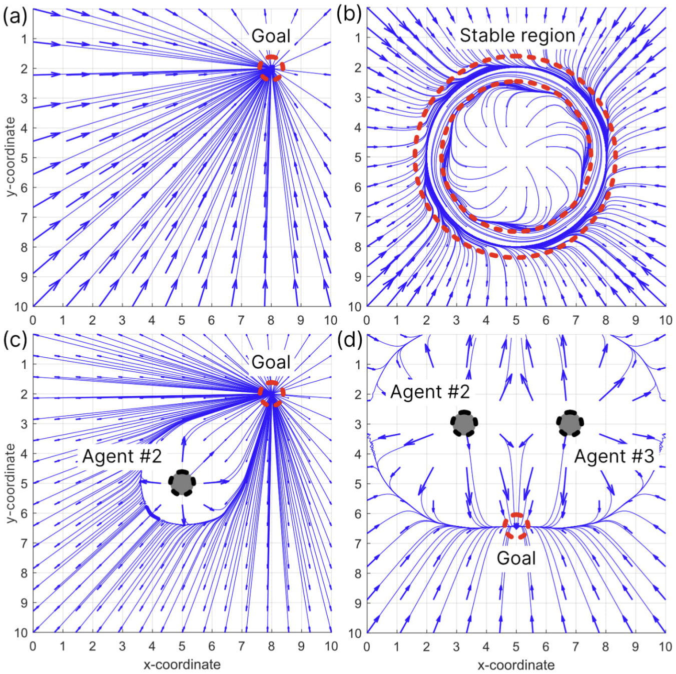


Fig. 2. Desired ball velocity vectors generated by the Kinetic Artificial Potential Fields for **(a)** positional control, **(b)** trajectory control using a stable limit cycle, **(c)** multi-agent control with one agent posing as an obstacle, and **(d)** formation control of a three-agent system. The size of the velocity vector indicates its relative magnitude.

F. Multi-agent extension

The two-layer APF algorithm can also be extended to control the velocity of multiple balls across the surface. Each ball independently follows the original algorithm, with the addition that the KPF contains repulsors of each other ball on the surface, allowing each agent on the surface to maintain distance separation (Fig. 2 (c)). Given this, Eq. (4) is modified to

$$KPF_n = a_{n,v}\|\boldsymbol{x}_{n,ball} - \boldsymbol{x}_{n,goal}\|^2 + \sum_{\substack{i=1 \\ i \neq n}}^{m} R(i) \tag{19}$$

for m total agents and current agent n. The repulsor $R(i)$ towards agent n from each other agent i is

$$R(i) = r_v \exp\left(\frac{-1}{2{\sigma_v}^2}\|\boldsymbol{x}_{n,ball} - \boldsymbol{x}_{i,ball}\|^2\right) \tag{20}$$

where r_v and ${\sigma_v}^2$ is the repulsor magnitude and variance. Taking the negative gradient results in the new desired velocity of each agent, modified from Eq. (5):

$$\begin{aligned} \dot{\boldsymbol{x}}_{n,des} &= -\nabla KPF_n \\ &= -2a_{n,v}(\boldsymbol{x}_{n,ball} - \boldsymbol{x}_{n,goal}) \\ &\quad + \sum_{\substack{i=1 \\ i \neq n}}^{m} \frac{1}{{\sigma_v}^2}(\boldsymbol{x}_{n,ball} - \boldsymbol{x}_{i,ball})R(i). \end{aligned} \tag{21}$$

Note that the addition of the repulsor terms to the KPF can create instantaneous local minima. However, dead-lock is only possible if multiple agents share the same goal positions, as they would repel each other away from the goal. Also note that the KPF gradient can be arbitrarily replaced with another function as in Section II(E), provided unique trajectory control over each agent.

The $\dot{\boldsymbol{x}}_{n,des}$ for each agent n can then be used to calculate each $\dot{\boldsymbol{x}}_{n,req}$ following Eq. (6). Similarly, following Eqs. (7-11) one can derive the $h_{n,rep}$ and $\boldsymbol{x}_{n,rep}$ for each agent. These values from each agent are input to Eq. (9) to obtain $PPF_n(\boldsymbol{x}_{surf})$ for each agent. The complete set of PPFs is

$$F(\boldsymbol{x}_{surf}) = \{PPF_n(\boldsymbol{x}_{surf}) | n = 1, ..., m\}, \qquad (22)$$

and the global PPF applied to the surface is the maximum value from F at every point in $\boldsymbol{x}_{surf}$,

$$PPF_{global}(\boldsymbol{x}_{surf}) = \max(F(\boldsymbol{x}_{surf})). \qquad (23)$$

Accordingly, if the individual PPFs of two or more agents overlap on $\boldsymbol{x}_{surf}$, the PPF with the largest magnitude at that location is used for the global value. As in the single-agent algorithm, the global PPF is used to generate the new surface topology of the shape display to drive the motion of all agents.

G. Formation control

We further expand the algorithm via a small addition to the multi-agent equations to enable cooperative formations between agents. This can be done by replacing the user-set goal position of each agent with an attractor located at the global centroid of the agents on the surface. Eq. (19) is modified by replacing $\boldsymbol{x}_{goal}$ with the centroid $\boldsymbol{x}_c$:

$$\boldsymbol{x}_c = \frac{1}{m}\sum_{i=1}^{m} \boldsymbol{x}_{i,ball}. \qquad (24)$$

Eq. (19) thus becomes

$$KPF_n = c_v\|\boldsymbol{x}_{n,ball} - \boldsymbol{x}_c\|^2 + \sum_{\substack{i=1 \\ i\neq n}}^{m} R(i) \qquad (25)$$

with scaling factor c_v applied to the centroid attractor. Similar techniques have been used in literature to create multi-agent formations [35], [36].

For m agents placed at the vertices of an m-sided regular polygon each agent experiences an attractive force in the KPF towards the centroid (center of the polygon) and $m-1$ repulsive forces from the other agents. At a set distance from the centroid these forces result in a net zero force in the KPF, resulting in a minimum in the field and zero velocity vector in the gradient at the vertices of the polygon (Fig. 2 (d)). Adjusting the scaling factor c_v controls the size of the polygonal formation created by the agents.

III. Experimental Validation

Having described the mathematical approach and capabilities of our two-layer APF algorithm, we implemented the algorithm on a robotic shape-changing surface to validate its effectiveness in each use-case: (i) point-to-point control, (ii) trajectory control, (iii) multi-agent control, and (iv) formation control.

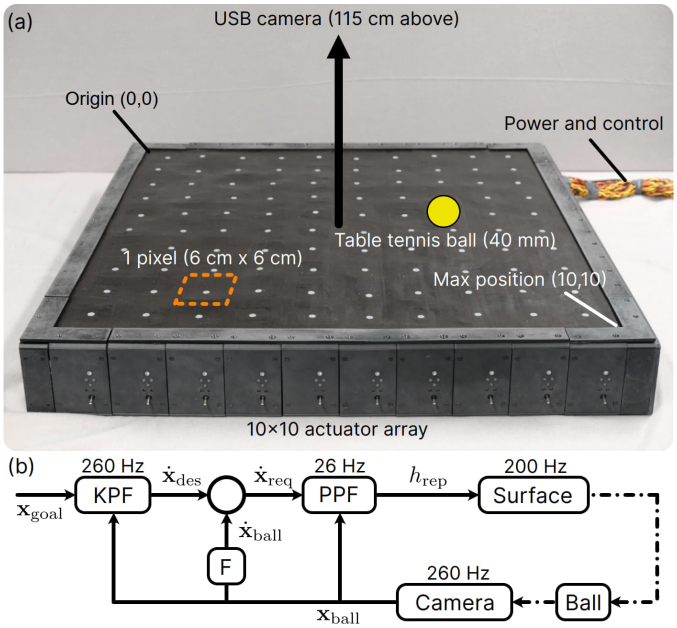

(a)
USB camera (115 cm above)
Origin (0,0)
Power and control
Table tennis ball (40 mm)
1 pixel (6 cm x 6 cm)
Max position (10,10)
10×10 actuator array
(b)
260 Hz
KPF
$\dot{\mathbf{x}}_{\text{des}}$
$\dot{\mathbf{x}}_{\text{req}}$
26 Hz
PPF
h_{rep}
200 Hz
Surface
$\mathbf{x}_{\text{goal}}$
$\dot{\mathbf{x}}_{\text{ball}}$
F
260 Hz
Camera
Ball
$\mathbf{x}_{\text{ball}}$

Fig. 3. Hardware and control loop for experimental validation. **(a)** The shape display is 10 x 10 actuator array with a deformable silicone surface skin. Each actuator forms a 6 cm x 6 cm pixel. A standard table tennis ball is used for all experiments, and ball position is measured with an overhead USB camera. **(b)** Block diagram of the APF algorithm implemented on the device. Block F samples ball position to obtain velocity (Eq. (27)). Blocks KPF (Eq. (4)) and PPF (Eq. (9)) form the two potential fields of the algorithm. Height commands are sent to local controllers in the shape display to deform the surface. The dashed line indicates the physical portion of the diagram; the remaining control takes place via computer.

A. Hardware

We use a soft shape display developed in part by the authors [10] to validate the APF algorithm (Fig. 3 (a)). The display is driven by a 10 x 10 array of electrostatic actuators which transduce high voltage to force and displacement [37]. Each actuator occupies a 6 cm x 6 cm area. Closed-loop control of each actuator's displacement is achieved using local microcontrollers and magnetic-based displacement sensors beneath each pixel [10]. The displacement loop runs at 200 Hz. Above the array, the display uses a thin silicon skin to form a continuous, deformable surface for object manipulation.

In addition, a high speed USB color camera (2.9mm Wide Angle, ELP) is placed 115 cm above the device and images the surface at 260 Hz. This is used to measure ball position and to identify multiple balls based on color [10]. A desktop computer interfaces with the USB camera, runs the APF algorithm, and sends control commands to the shape display microcontrollers. Standard table tennis balls with a diameter of 40 mm are used for all experiments (Fig. 3).

B. Software

The computer-side software uses Julia [38] for computation and communication, and the display microcontrollers use C++ using the same code as reported in the original work describing the shape display [10]. We also implement the two-layer APF algorithm in Julia, and the USB camera interfaces with Julia using a Python script and OpenCV [10], [39].

We obtain ball velocity by differentiating a weighted mean of the latest 10 ball position values sampled from the camera, which is used in the KPF calculation at 260 Hz to match the camera sample rate. The smoothed velocity derived from $\boldsymbol{x}_{ball}$ at sample time $[k]$ with sample frequency f_s (Hz) using a 10-sample weighted moving average is

$$\dot{\boldsymbol{x}}_{ball}[k] = (\boldsymbol{x}_{ball}[k] - \boldsymbol{x}_{ball}[k-1])\, f_s \tag{26}$$

$$\dot{\boldsymbol{x}}_{ball,smth}[k] = \frac{\sum_{i=1}^{10} W[i]\dot{\boldsymbol{x}}_{ball}[k-(i-1)]}{\sum_{i=1}^{10} W[i]} \tag{27}$$

$$\begin{aligned} W = \big[1,\quad 0.917,\quad 0.833,\quad 0.75,\quad 0.667,\\ 0.584,\quad 0.5,\quad 0.417,\quad 0.334,\quad 0.25,\big] \end{aligned} \tag{28}$$

Because the weighted mean introduces a lag in the sensor measurement, the PPF is calculated at 26 Hz compared to 260 Hz for the KPF. This is a sufficient loop frequency to control object motion without excessive lag. The closed loop block diagram describing the application of the algorithm and filter in relation to the hardware is shown in Fig. 3 (b).

C. Performance quantification

To validate the algorithm's capabilities, we define quantifiable performance measures. For all experiments, we normalize positions and velocities to the side length of a single actuator "pixel" (6 cm). Therefore, an input speed of 1 unit/s corresponds to 6 cm/s on our shape display. This allows the results to be interpreted in the context of other shape displays with different pixel sizes.

1) Step inputs: We evaluate a step input on the KPF in the form of a uniform velocity gradient which steps from zero to constant velocity:

$$\dot{\boldsymbol{x}}_{des} = \mathbf{0} \rightarrow v\hat{\boldsymbol{e}} \tag{29}$$

where v is the scalar speed $\hat{\boldsymbol{e}}$ is the unit direction vector of the field. For example, $\hat{\boldsymbol{e}} = (1,0)$ corresponds to a uniform velocity field along the x-axis of the shape-changing surface.

We expect worse performance for off-axis field directions because the shape of each actuator is square. This results in non-uniform surface deformation which will perturb ball velocity. With this in mind, we evaluate the step response of the algorithm under 5 directions and 7 speeds with the speed and direction vector specified by

$$\begin{aligned} &v = \{2,3,4,5,6,7,8\}\; unit/s \\ &\hat{\boldsymbol{e}} = \left\{ \left(cos\left(n_\theta \frac{\pi}{8}\right), sin\left(n_\theta \frac{\pi}{8}\right)\right) | n_\theta = 0,1,...,4 \right\}. \end{aligned} \tag{30}$$

For each direction n_θ, 6 trials are performed using a constant speed $v = 4$ for each n_θ. Likewise, for each speed v, 6 trials are performed using a constant direction $n_\theta = 0$ for each v. The ball is placed in different locations for each trial to validate the algorithm's spatial robustness.

Velocity data are processed by measuring the 10-90% rise time of the ball velocity from 0 to target velocity v for each trial. The direction data are processed by (i) subtracting the start position of each trial so that each trial begins at (0,0), (ii) normalizing the data in time, (iii) re-sampling to 501 data points, (iv) calculating the mean position of the 6 trials over time, and (v) normalizing the mean positions to a unit circle.

TABLE I
ALGORITHM PARAMETER VALUES FOR THE SOFT SHAPE DISPLAY

Equation	Parameter	Value
(5)	a_v	1.5
(9)	σ^2	0.5625
(11)	α	0.25
(19)	$a_{n,v}$	1.5
(20)	r_v	100
(20)	${\sigma_v}^2$	0.25
(18)	γ	1

2) Positional control testing for single and multiple agents: We evaluate the algorithm's positional control capability in both a single- and multi-agent scenario. For the single agent, we position the ball at a starting surface position of (2,8) with the KPF set to a goal position of (8,2) (Fig. 2 (a)). Initial ball velocity is zero. As the ball travels along the surface we record position and time data. The test is terminated after the ball reaches the goal position. This is repeated 10 times.

For the multi-agent case, we use the same initial and goal positions as the single-agent case. However, we also add a static (non-moving) ball at position (5,5) which acts as an obstacle (Fig. 2 (c)). As the ball travels along the surface we record position and time data, and the test is terminated after the ball reaches the goal position. This is repeated 10 times.

3) Trajectory analysis using a limit cycle: We evaluate the algorithm's capability to control global trajectories using the limit cycle KPF described in Eq. (18). We set $r_i = 2.5$ and $r_o = 2.5$. The orbit is centered at $x_c = 5, x_d = 5$, the center of the display surface (Fig. 2 (c)).

The ball has a random starting position within the stable orbit (between 2.5-3.5 radius from center). As the ball moves across the surface we record position and time data. We record this data for 20 orbits along the limit cycle taken across 4 experimental trials.

4) Multi-agent formation control: Lastly, we test the formation capabilities of the multi-agent algorithm using a three-ball system, corresponding to an ideal formation of an equilateral triangle with one agent at each vertex. Fig. 2 (d) shows an example velocity field which would be generated by this experiment. The desired formation is an equilateral triangle with area 6.25 pixel^2 and side length 3.8 pixel.

We position each agent near a different corner of the display surface with zero initial velocity. Time and position of each agent was recorded over each trial. We repeated this for 7 trials. To quantify the performance, we evaluate both the mean side length (of each side separately) and mean area over time of the triangle formed by the three agents.

5) Algorithm parameters: Table I lists the parameter values used in the above experiments using the equations in Section II. The parameters are unique to our display and must be experimentally tuned for other displays.

Due to the limited actuation height of each pixel on the shape display [10], we also limit h_{req} to a maximum value of 10 mm.

TABLE II
ALGORITHM PARAMETERS FOR FORMATION CONTROL

Equation	Parameter	Value
(20)	r_v	150
(20)	${\sigma_v}^2$	1.96
(25)	c_v	1.5

For the formation controller, some values from Table I are modified to obtain the desired formation shape. Table II lists the parameter values for the formation control experiments.

IV. Results and Discussion

A. Uniform field step inputs

Under the effects of a uniform vector field, the algorithm can drive ball motion with median 10-90% rise times below 0.9 s for all tested speeds (Fig. 4 (a)). For speeds below 8 pixel/s (48 cm/s on our shape display) the median rise times are below 0.5 s, showing a rapid ability to control ball velocity using surface deformation. The smallest distribution of rise times occurs for speeds between 5-7 pixel/s; we believe that at faster speeds the controller is more susceptible to the effects of feedback delay, while at slower speeds the ball is more susceptible to perturbations from local surface imperfections.

The experimental results for directional performance show that the algorithm generally maintains the desired direction of ball motion (Fig. 4 (b)). Important to note is that the non-orthogonal directions exhibit a similar performance to that of the orthogonal fields, suggesting a negligible performance impact from actuator shape. Imperfect positional accuracy can be attributed to inconsistencies among the actuators, which do not share identical voltage-displacement curves [10]. We expect the results would be further improved using a higher-resolution array with greater actuator consistency.

B. Positional control

Positional control results for a single agent are shown in Fig. 5 (a), with a sample trial also shown in the Supplementary Material. Each trial shows consistent trajectory and overshoot performance despite minor variations in initial ball position. The spread of trajectories over the 10 trials has a width of about 1 pixel, which is the expected performance limit, and the decrease in velocity error towards zero over time shows the system is Lyapunov stable.

The addition of the static obstacle agent is alters the ball trajectory as predicted by the shape of the KPF (Fig. 5 (b)). All 10 trials show the ball remaining at least 1 pixel distance away from the obstacle. Compared to the single-agent system, the trajectories exhibit a wider spread and variability, likely due to the larger gradients introduced by the presence of the static agent. As in the single-agent case, the velocity error approaches zero over time.

C. Trajectory control

Using the specified limit-cycle equations for the KPF, the algorithm achieves circular ball trajectories (Fig. 5 (c)) without requiring a parameterized time equation (i.e. where the goal position is a function of time). A sample orbit is shown in the Supplementary Material.

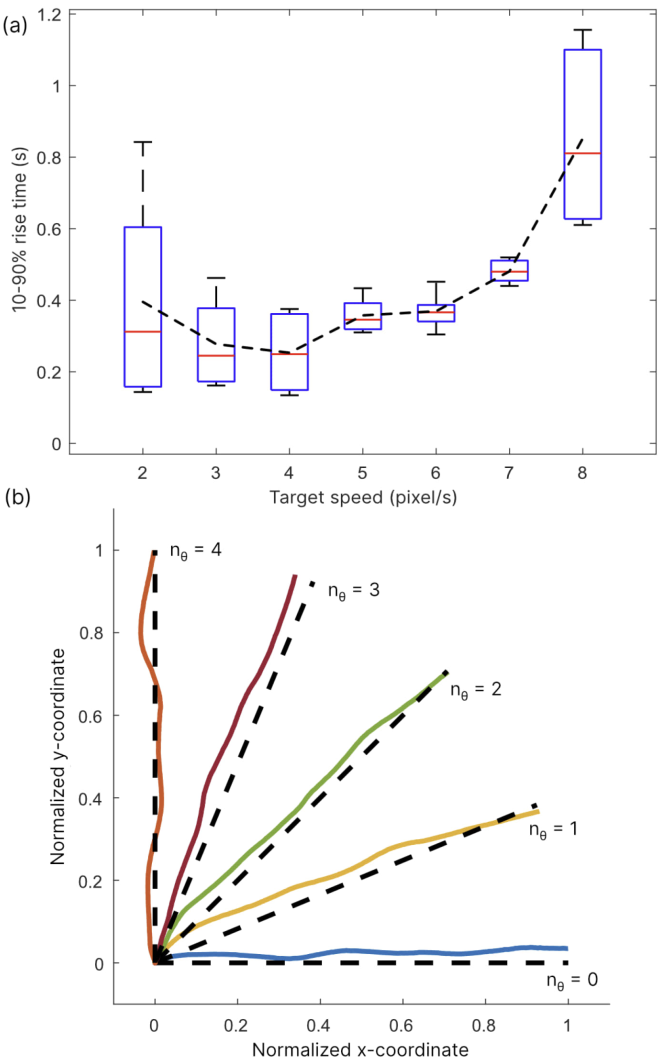


Fig. 4. ”Step response” results for uniform velocity fields. **(a)** The rise time to reach the target speed (from initial speed = 0) was typically less than 1 s. The box plot shows the median value, the 25 and 75 percentiles, and outliers (n = 6). The dashed line indicates the mean rise time for each speed. **(b)** Mean trajectory (n = 6) for each uniform field direction. The results show that pixel orientation relative to the desired velocity vector does not affect ball motion control. On our device, 1 pixel = 6 cm.

The shape display pixel size plays an especially important role in the performance of this experiment. Tightly curved trajectories with a small radius of curvature necessarily contain more degrees of arc across a single pixel compared to broadly curved trajectories. These tightly curved trajectories are therefore harder to accurately maintain given the reduced number of actuators at play.

The general trend shows the agent tracing out the desired trajectory over a period of 20 cycles, showing stability in the Lyapunov sense (Fig. 5 (c)). However, the radius of the ball trajectory is highly oscillatory within the stable region. This is likely due to the large gradients in the KPF outside the stable orbit, resulting in a greater tendency to overshoot velocity if the ball moves to close to the edges of the stable region.

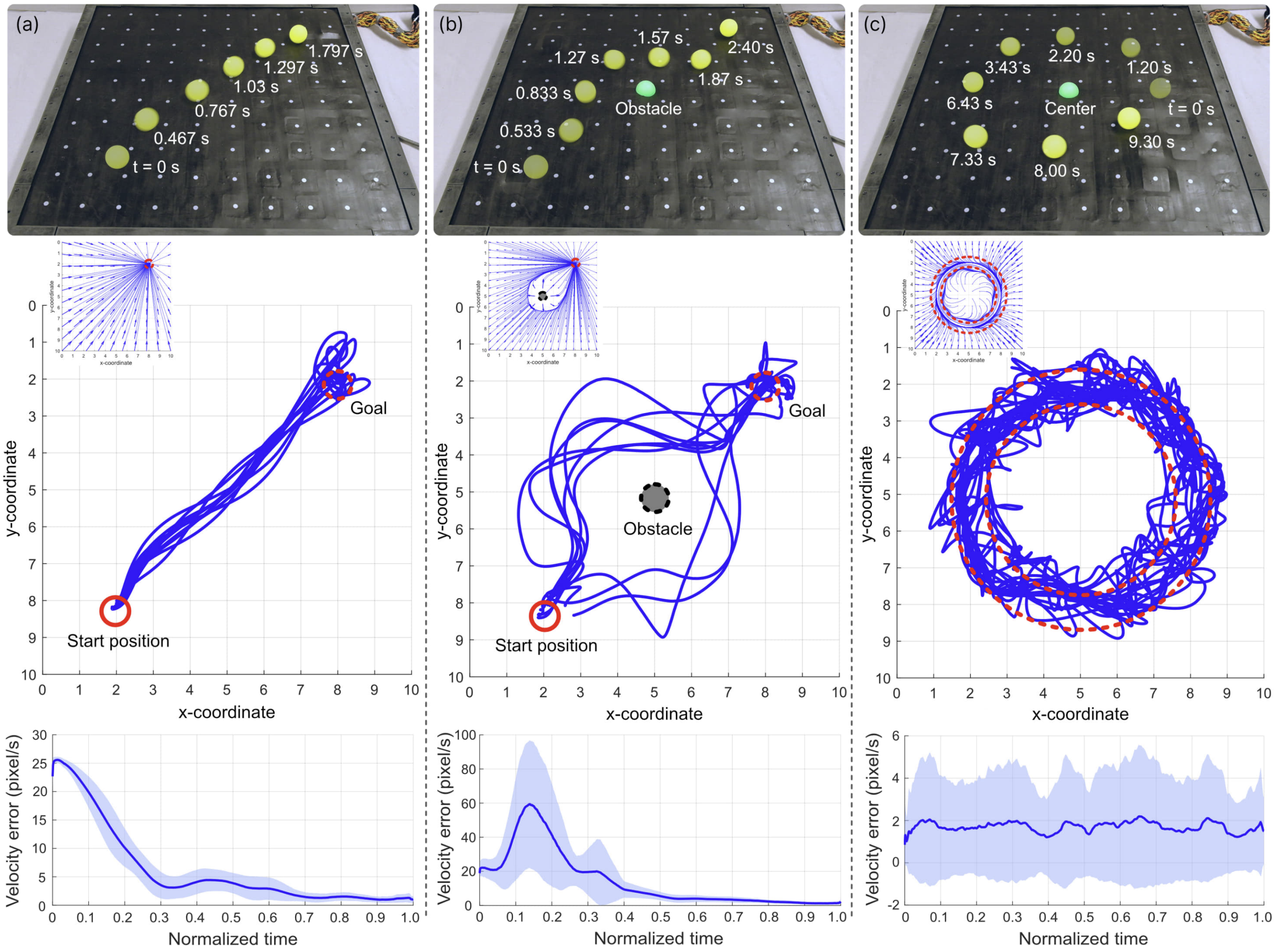


Fig. 5. Validation of algorithm capabilities including (top) a timelapse of one trial, (middle) recorded ball position of all trials with inlays showing the corresponding vector field from Fig. 2, and (bottom) magnitude of ball velocity error normalized over time (solid line = mean, $\pm\sigma$ = shaded area, 1 pixel = 6 cm) plotted for **(a)** single-agent positional control (n = 10), **(b)** multi-agent positional control with one agent acting as an obstacle (n = 10), and **(c)** single-agent trajectory control using a stable limit cycle (n = 5, 20 cycles).

This also increases the variability and spread of trajectories beyond the stable region; i.e., the trace of the trajectories has a diameter greater than 1 pixel. As a result of the greater variability and continuous nature of the limit cycle, the velocity error does not decrease over time.

D. Formation control

Formation control of three agents is shown in Fig. 6, with a sample trial shown in the Supplementary Material. This demonstration shows that multi-agent formation control is achievable with the algorithm, and the formation is rapidly achievable (within 2.5 seconds as shown in Fig. 6 (a)). The resulting equilateral triangle formation has the desired side lengths and area (Fig. 6 (b,c)), corresponding to the desired triangle shape. The results tend to be oscillatory just like the those of the limit cycle experiment, suggesting that this method of control is not asymptotically stable but at least Lyapunov stable. This oscillatory behavior is likely due to a rubber-banding effect where the deviation of any single agent away from its position will move the formation centroid and thus ”pull” the other agents in that direction according to the changed potential fields.

The formation algorithm can be combined with the other demonstrated KPFs. For example, one agent can use a positional control KPF to control it’s position, while the other agents will maintain formation control and move with the positional-controlled agent in the desired formation. The multi-agent system could also follow a trajectory-based KPF or other KPFs, enabling spatial and pose control of the formation. These capabilities are a result of the independent KPF generation of each agent.

V. Conclusions

In this work we formulate and validate a potential field algorithm to manipulate the positions, velocities, and trajectories of spherical objects on the surface of shape displays. Object motion in the algorithm is driven by the conversion of a velocity vector field to a potential energy field realized

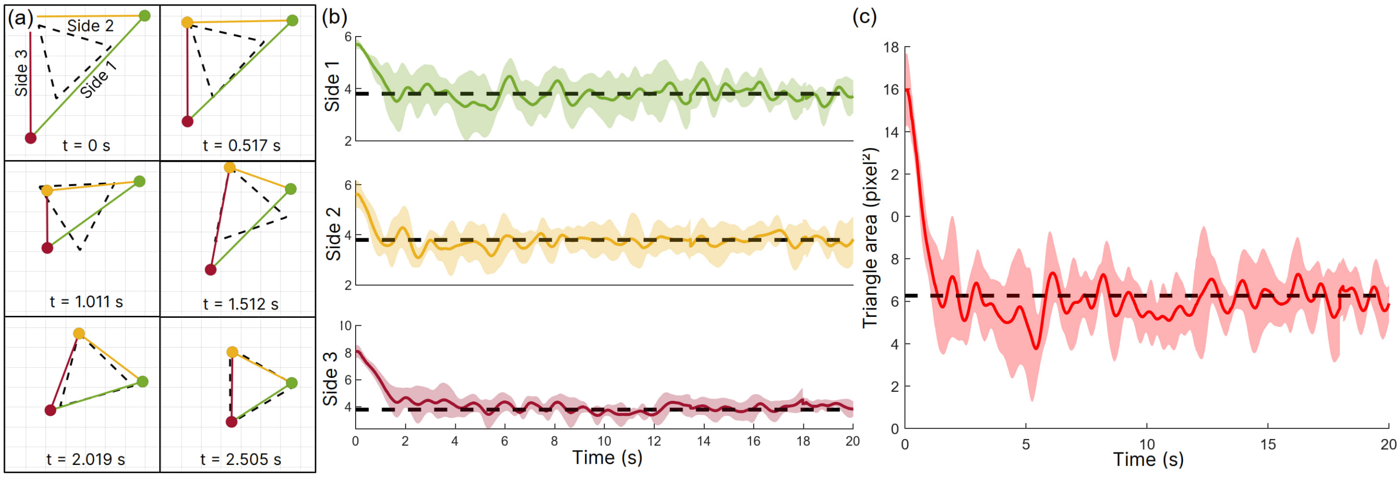


Fig. 6. Formation control using three agents to form an equilateral triangle. **(a)** Snapshots of the shape formed by the 3 balls (solid line triangle) versus the target formation (dashed line triangle) for a single trial. The target shape is achieved in 2.5 seconds. **(b)** Triangle side length (pixel) over time (mean = solid line, $\pm\sigma$ = shaded area, n = 7). The dashed line denotes the target side length (3.8 pixel). **(c)** Triangle area over time (mean = solid line, $\pm\sigma$ = shaded area, n = 7). The target area is 6.25 pixel^2. On our device, 1 pixel = 6 cm.

via deformation of the shape display. We present four distinct capabilities which stem from the algorithm: (i) positional control of a ball, (ii) multi-agent/multi-ball simultaneous control and avoidance, (iii) global trajectory control of a ball, and (iv) formation control of a multi-agent system.

Our experimental results using a 10x10 array soft shape display verify the success of the algorithm in achieving all four capabilities. We demonstrate rapid velocity control with step response rise times on the order of 0.5 seconds as well as high directional accuracy regardless of the shape display pixel orientation. We also validated positional control of objects, obstacle avoidance on the display surface, stable control of the global trajectory of a ball using a stable limit cycle, and the ability to create multi-agent formations within approximately 2.5 seconds. We expect the performance of all capabilities can be increased by using a higher resolution shape display relative to object diameter, using a higher resolution/higher frequency camera and control loop, and fine-tuning the algorithm parameters (Tables I and II).

This work provides a promising approach to global object manipulation with shape displays and suggests new applications for artificial potential fields. By validating the algorithm in a real-world experiment, we show the promise of this approach in accomplishing practical work. We anticipate that this control approach could be used for new haptic interactions or games for small-scale shape displays, and at the industrial-scale could enable increased capabilities in object conveyance, sorting, and assembly in manufacturing and agriculture.

References

[1] M. K. Rasmussen, E. W. Pedersen, M. G. Petersen, and K. Hornbæk, “Shape-changing interfaces: a review of the design space and open research questions,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2012, pp. 735–744.

[2] J. Alexander, A. Roudaut, J. Steimle, K. Hornbæk, M. Bruns Alonso, S. Follmer, and T. Merritt, “Grand challenges in shape-changing interface research,” in *Proceedings of the 2018 CHI conference on human factors in computing systems*, 2018, pp. 1–14.

[3] S. Follmer, D. Leithinger, A. Olwal, A. Hogge, and H. Ishii, “inform: dynamic physical affordances and constraints through shape and object actuation.” in *Uist*, vol. 13, no. 10.1145, 2013, pp. 2 501 988–2 502 032.

[4] “Wave-handling,” Festo, Brochure No. 54817, 2013, accessed Aug. 17, 2021. [Online]. Available: https://www.festo.com/PDF_Flip/corp/Festo_WaveHandling/en/index.html

[5] T. Wang, J. Zhang, J. Hong, and M. Y. Wang, “Dielectric elastomer actuators for soft wave-handling systems,” *Soft robotics*, vol. 4, no. 1, pp. 61–69, 2017.

[6] R. Hashem, B. Smith, D. Browne, W. Xu, and M. Stommel, “Control of a soft-bodied xy peristaltic table for delicate sorting,” in *2016 IEEE 14th International Workshop on Advanced Motion Control (AMC)*. IEEE, 2016, pp. 358–363.

[7] A. De Acutis, L. Calabrese, A. Bau, V. Tincani, N. Pugno, A. Bicchi, and D. De Rossi, “Design and proof of concept for multi degree of freedom hydrostatically coupled dielectric elastomer actuators with roto-translational kinematics for object handling,” *Smart Materials and Structures*, vol. 27, no. 7, p. 074005, 2018.

[8] A. F. Siu, E. J. Gonzalez, S. Yuan, J. B. Ginsberg, and S. Follmer, “Shapeshift: 2d spatial manipulation and self-actuation of tabletop shape displays for tangible and haptic interaction,” in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, 2018, pp. 1–13.

[9] M. Coelho and J. Zigelbaum, “Shape-changing interfaces,” *Personal and Ubiquitous Computing*, vol. 15, no. 2, pp. 161–173, 2011.

[10] B. K. Johnson, M. Naris, V. Sundaram, A. Volchko, K. Ly, S. K. Mitchell, E. Acome, N. Kellaris, C. Keplinger, N. Correll, J. S. Humbert, and M. E. Rentschler, “A multifunctional soft robotic shape display with high-speed actuation, sensing, and control,” *Nature Communications*, vol. 14, no. 4516, 2023.

[11] D. Leithinger, S. Follmer, A. Olwal, and H. Ishii, “Shape displays: Spatial interaction with dynamic physical form,” *IEEE computer graphics and applications*, vol. 35, no. 5, pp. 5–11, 2015.

[12] K. Nakagaki, D. Fitzgerald, Z. Ma, L. Vink, D. Levine, and H. Ishii, “inforce: Bi-directionalforce’shape display for haptic interaction,” in *Proceedings of the thirteenth international conference on tangible, embedded, and embodied interaction*, 2019, pp. 615–623.

[13] P. Schoessler, D. Windham, D. Leithinger, S. Follmer, and H. Ishii, “Kinetic blocks: Actuated constructive assembly for interaction and display,” in *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, 2015, pp. 341–349.

[14] M. Stommel and W. Xu, “Optimal, efficient sequential control of a soft-bodied, peristaltic sorting table,” *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 2, pp. 858–867, 2015.

[15] O. Khatib, “Real-time obstacle avoidance for manipulators and mobile robots,” in *IEEE International Conference on Robotics and Automation*, St. Louis, MO, USA, 1985, pp. 500–505.

[16] J. Barraquand, B. Langlois, and J.-C. Latombe, “Numerical potential

field techniques for robot path planning," *IEEE transactions on systems, man, and cybernetics*, vol. 22, no. 2, pp. 224–241, 1992.

[17] Y. Koren, J. Borenstein *et al.*, "Potential field methods and their inherent limitations for mobile robot navigation." in *ICRA*, vol. 2, 1991, pp. 1398–1404.

[18] J.-H. Chuang and N. Ahuja, "An analytically tractable potential field model of free space and its application in obstacle avoidance," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 28, no. 5, pp. 729–736, 1998.

[19] R. Zappulla, H. Park, J. Virgili-Llop, and M. Romano, "Real-time autonomous spacecraft proximity maneuvers and docking using an adaptive artificial potential field approach," *IEEE Transactions on Control Systems Technology*, vol. 27, no. 6, pp. 2598–2605, 2018.

[20] M. C. Lee and M. G. Park, "Artificial potential field based path planning for mobile robots using a virtual obstacle concept," in *Proceedings 2003 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM 2003)*, vol. 2. IEEE, 2003, pp. 735–740.

[21] L. Huang, "Velocity planning for a mobile robot to track a moving target—a potential field approach," *Robotics and Autonomous Systems*, vol. 57, no. 1, pp. 55–63, 2009.

[22] X. Xu, Y. Hu, J. Zhai, L. Li, and P. Guo, "A novel non-collision trajectory planning algorithm based on velocity potential field for robotic manipulator," *International Journal of Advanced Robotic Systems*, vol. 15, no. 4, p. 1729881418787075, 2018.

[23] P. Song and V. Kumar, "A potential field based approach to multi-robot manipulation," in *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*, vol. 2. IEEE, 2002, pp. 1217–1222.

[24] T. Tsuji, Y. Tanaka, P. G. Morasso, V. Sanguineti, and M. Kaneko, "Bio-mimetic trajectory generation of robots via artificial potential field with time base generator," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 32, no. 4, pp. 426–439, 2002.

[25] W. Afzal and A. A. Masoud, "Model-based navigation control for communication-aware motion using harmonic potential fields," *IEEE Transactions on Automation Science and Engineering*, 2022.

[26] C. M. Saaj, V. Lappas, and V. Gazi, "Spacecraft swarm navigation and control using artificial potential field and sliding mode control," in *2006 IEEE International Conference on Industrial Technology*. IEEE, 2006, pp. 2646–2651.

[27] M. Zhang, Y. Shen, Q. Wang, and Y. Wang, "Dynamic artificial potential field based multi-robot formation control," in *2010 IEEE Instrumentation & Measurement Technology Conference Proceedings*. IEEE, 2010, pp. 1530–1534.

[28] K. H. Kowdiki, R. K. Barai, and S. Bhattacharya, "Leader-follower formation control using artificial potential functions: A kinematic approach," in *IEEE-International Conference On Advances In Engineering, Science And Management (ICAESM-2012)*. IEEE, 2012, pp. 500–505.

[29] E. Fiorelli, N. E. Leonard, P. Bhatta, D. A. Paley, R. Bachmayer, and D. M. Fratantoni, "Multi-auv control and adaptive sampling in monterey bay," *IEEE journal of oceanic engineering*, vol. 31, no. 4, pp. 935–948, 2006.

[30] R. Sepulchre, D. A. Paley, and N. E. Leonard, "Stabilization of planar collective motion: All-to-all communication," *IEEE Transactions on automatic control*, vol. 52, no. 5, pp. 811–824, 2007.

[31] W. Pang, D. Zhu, and C. Sun, "Multi-auv formation reconfiguration obstacle avoidance algorithm based on affine transformation and improved artificial potential field under ocean currents disturbance," *IEEE Transactions on Automation Science and Engineering*, 2023.

[32] T. Weerakoon, K. Ishii, and A. A. F. Nassiraei, "An artificial potential field based mobile robot navigation method to prevent from deadlock," *Journal of Artificial Intelligence and Soft Computing Research*, vol. 5, no. 3, pp. 189–203, 2015.

[33] K. Sato, "Deadlock-free motion planning using the laplace potential field," *Advanced Robotics*, vol. 7, no. 5, pp. 449–461, 1992.

[34] M. N. Brearley and N. J. De Mestre, "Rolling of a rigid ball on a horizontal deformable surface," *The ANZIAM Journal*, vol. 46, no. 2, pp. 249–264, 2004.

[35] F. E. Schneider and D. Wildermuth, "A potential field based approach to multi robot formation navigation," in *IEEE International Conference on Robotics, Intelligent Systems and Signal Processing, 2003. Proceedings. 2003*, vol. 1. IEEE, 2003, pp. 680–685.

[36] E. Fiorelli, N. E. Leonard, P. Bhatta, D. A. Paley, R. Bachmayer, and D. M. Fratantoni, "Multi-auv control and adaptive sampling in monterey bay," *IEEE journal of oceanic engineering*, vol. 31, no. 4, pp. 935–948, 2006.

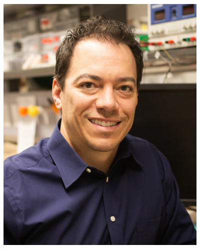
[37] S. K. Mitchell *et al.*, "An Easy-to-Implement Toolkit to Create Versatile and High-Performance HASEL Actuators for Untethered Soft Robots," *Advanced Science*, vol. 6, 2019, Art. no. 1900178.

[38] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, "Julia: A fresh approach to numerical computing," *SIAM review*, vol. 59, no. 1, pp. 65–98, 2017.

[39] G. Bradski and A. Kaehler, "Opencv," *Dr. Dobb's Journal of Software Tools*, vol. 3, p. 120, 2000.

Brian K. Johnson received a B.S. degree in mechanical engineering from Cornell University, Ithaca, NY, USA and both M.S. and Ph.D degrees in mechanical engineering from the University of Colorado Boulder, Boulder, CO, USA. He was also a National Science Foundation Graduate Research Fellow. His research focuses on the sensing and control of soft electrostatic actuator systems.

James Sean Humbert is the Denver Business Challenge Professor in the Department of Mechanical Engineering and the founding director of the Robotics Graduate Program at the University of Colorado, Boulder. He holds a B.S. degree in Mechanical Engineering from the University of California, Davis, along with M.S. and Ph.D. degrees in Mechanical Engineering from Caltech. His main research areas are robotics and autonomy, with a focus in perception, reduction and feedback principles in biology.



Mark E. Rentschler (Senior Member, IEEE) received the B.S. degree in mechanical engineering from the University of Nebraska, Lincoln, NE, USA, in 2001, the M.S. degree in mechanical engineering from the Massachusetts Institute of Technology (MIT), Cambridge, MA, USA, in 2003, and the Ph.D. degree in biomedical engineering from the University of Nebraska, in 2006.

He is currently a Professor in mechanical engineering with the University of Colorado Boulder, Boulder, CO, USA, with a secondary appointment at the Department of Surgery, University of Colorado Anschutz Medical Campus, Aurora, CO, USA. His research interests are in medical devices and surgical robotics.