SoftSling: A Soft Robotic Arm Control Strategy to Throw Objects With Circular Run-Ups

Diego Bianchi, *Graduate Student Member, IEEE*, Giulia Campinoti, Costanza Comitini, Cecilia Laschi, *Fellow, IEEE*, Alessandro Rizzo, *Senior Member, IEEE*, Angelo Maria Sabatini, *Senior Member, IEEE*, and Egidio Falotico, *Member, IEEE*

***Abstract*—In this letter, we present SoftSling, a soft robot control strategy designed for accurately throwing objects following circular run-ups. SoftSling draws inspiration from ancient slingers, who rotated a sling loaded with a projectile at high speeds to fight and hunt, releasing the object by letting go of the sling's end. Our study aims to replicate this behavior by exploiting the embodied intelligence of soft robots under periodic actuation input, that enables them to generate self-stabilizing motions. The periodic input parameters for moving along a circular-like path are generated by a neural network based on the weight of the object and the target position. Subsequently, a separate neural network model predicts the release time by considering the gripper opening delay and the object positions during motion. We tested this strategy on a modular soft robot, I-Support, by throwing three objects of varying weights into 140-mm square target boxes. We achieved a success rate ranging from 75% to 88% for different objects, with the heaviest object yielding the highest success rate. Our research contributes to integrating soft robots into everyday life, enabling them to perform complex and dynamic tasks.**

***Index Terms*—Embodied intelligence, high dynamics, neural networks, soft robotics, throwing task.**

I. Introduction

THROWING objects with a robot is a highly desirable advancement, allowing for the expansion of the robot workspace without requiring physical modifications. This capability is particularly promising for the industrial sector, where robots are increasingly used to enhance operational efficiency. Many logistics tasks can benefit from this development [1]. For instance, adopting a throw-based approach enables a robot to significantly extend its workspace [2], effectively covering more distance with a pick-and-throw method rather than the traditional pick-and-place strategy [3], [4]. This approach is particularly effective in scenarios like garbage sorting at recycling centers [4], operations within warehouses [1], or automated mail sorting facilities. By moving only to the release point instead of completing the entire path, this capability also enhances productivity [4].

Generating ballistic motions in robots poses a challenge due to the limitations of joint stiffness and the high acceleration required from the actuators [5]. However, research has shown that using compliant actuators to store elastic energy can enable rigid robots to perform dynamic tasks such as throwing [6]. Soft robots, with their flexible properties [7], may be even better suited for these activities. Their mechanical properties allow them to store potential energy and release it suddenly, akin to natural mechanisms found in living creatures [8]. Additionally, the use of compliant materials in these robots inherently improves safety for workers.

These robots are typically bioinspired, extracting engineering principles from biology to create more robust and versatile systems [9]. One such principle is embodied intelligence, which enhances the adaptability and resilience of these robots. Due to their innovative design and control architectures, soft robots have demonstrated a range of capabilities, including locomotion [10], climbing [11], jumping [12], [13], flying [14], combined bending and stiffening [15], self-healing [16], and self-morphing and growth [17].

However, controlling a soft robot remains an open challenge due to several factors, including hysteresis and stochasticity, mainly caused by the manufacturing process and materials [18]. Despite these challenges, both model-free [19] and model-based controllers [20] have been developed to achieve static and dynamic tasks that, usually, do not involve interaction with the environment, with a primary focus on trajectory-following tasks.

The potential of soft robots to perform dynamic tasks beyond trajectory following has been demonstrated by few recent studies [21], [22]. Among these tasks is the throwing task, achieved by releasing a held object during a linear trajectory.

Manuscript received 26 March 2024; accepted 25 July 2024. Date of publication 13 August 2024; date of current version 20 August 2024. This article was recommended for publication by Associate Editor Ryan Truby and Editor Yong-Lae Park upon evaluation of the reviewers' comments. This work was supported by the European Union's Horizon 2020 FET-Open program under Grant 863212 (PROBOSCIS Project). *(Corresponding authors: Diego Bianchi; Egidio Falotico.)*

Diego Bianchi, Angelo Maria Sabatini, and Egidio Falotico are with The BioRobotics Institute, Scuola Superiore Sant'Anna, 56025 Pontedera, Italy, and also with the Department of Excellence in Robotics and AI, Scuola Superiore Sant'Anna, 56127 Pisa, Italy (e-mail: diego.bianchi@santannapisa.it; angelo.sabatini@santannapisa.it; egidio.falotico@santannapisa.it).

Giulia Campinoti is with the Medical Microinstrument, 56121 Pisa, Italy (e-mail: giulia.campinoti97@gmail.com).

Costanza Comitini is with the Stellantis, 10135 Turin, Italy (e-mail: costanza.comitini@external.stellantis.com).

Cecilia Laschi is with the Department of Mechanical Engineering, National University of Singapore, Singapore 119077, on leave from the BioRobotics Institute, Scuola Superiore Sant'Anna, 56127 Pontedera, Italy (e-mail: mpeclc@nus.edu.sg).

Alessandro Rizzo is with the Department of Electronics and Telecommunications, Politecnico di Torino, 10129 Turin, Italy (e-mail: alessandro.rizzo@polito.it).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2024.3442535, provided by the authors.

Digital Object Identifier 10.1109/LRA.2024.3442535

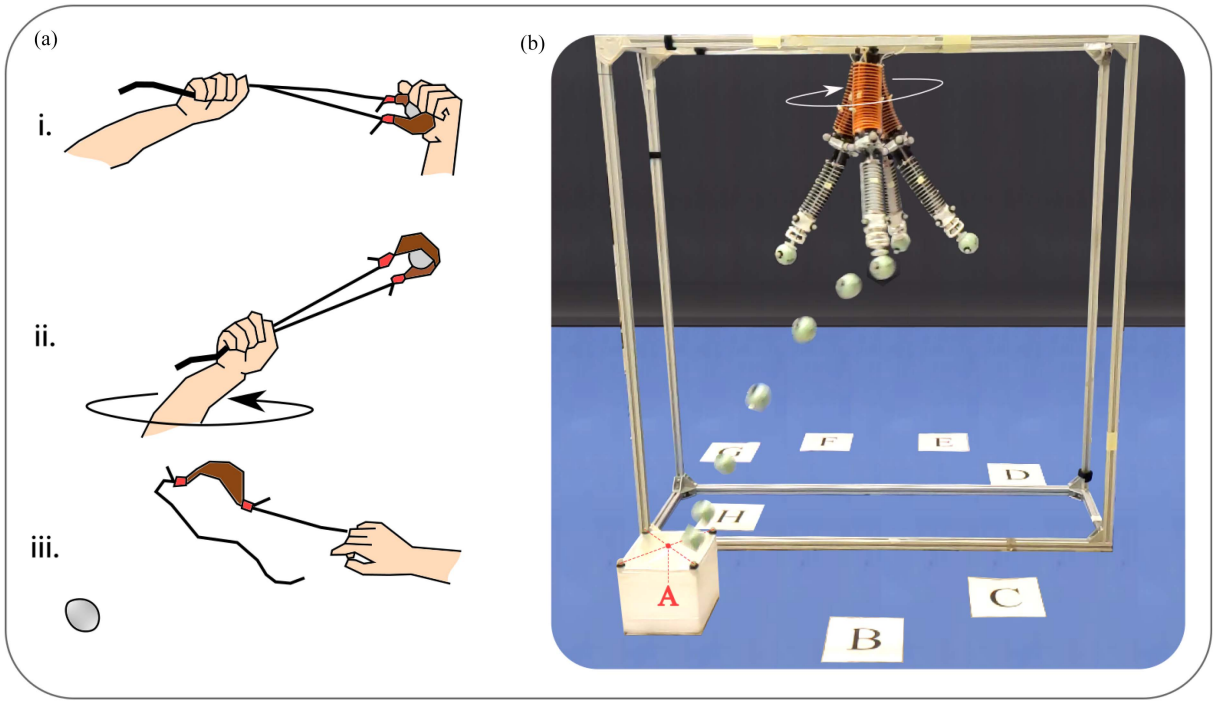

Fig. 1. *Sling*. (a) How to use a sling to throw an object. Using a sling to throw an object can be divided into three phases: i. The initial position involves loading and preparing the sling; ii. The object is accelerated by wrist rotation to reach a stable trajectory at high speed; iii. Release the sling, and let the object follow the tangent to the curve described in the previous step. (b) The soft robot throwing toward Target A.

Few controllers have been developed to deal with the throwing in a target position, such as the method described in [23], where the authors identified the actuation pattern for a simulated robot by solving an optimization process. However, this method is not suitable for real-time applications due to the iterative nature of the optimization method.

The first real-time open-loop controller for a soft robot based on reinforcement learning (RL), has been designed to perform straight throws, achieving a success rate of about 63% [24]. A similar level of performance has been attained with a learning-based inverse dynamic controller [25]. One intrinsic characteristic of open-loop systems is that they do not allow for corrections once an action has commenced, leading to reduced accuracy. Additionally, the performance of these controllers varies with directionality, (performing better when throwing in some directions and worse in others). Another limitation of the existing soft robot controllers for throwing tasks is the limited throwing distance [24], [25].

In this letter, we present SoftSling, a control strategy that enables a soft robot to perform the throwing task surpassing the performance of current state-of-the-art methods. Drawing inspiration from the mechanics of a sling (Fig. 1), SoftSling generates circular-like run-ups before releasing the object. Previous work has demonstrated that such trajectories executed by a soft arm are self-stabilizing [26].

To perform the throw into a target box, SoftSling addresses two key challenges: (i) generating a motion pattern that enables the held object to reach the necessary kinematic conditions (position and speed); and (ii) releasing the object at a precise instant to achieve the desired ballistic movement. For these purposes, as shown in Fig. 2, we leveraged model-free approaches by deploying two neural networks: (i) one to estimate the input parameters of the motion pattern based on the target position and object weight; and (ii) another one to estimate the release time by considering the moving object positions and the gripper release delay.

II. Materials and Methods

A. Methodology

Throwing an object requires reaching a point in space at a certain speed and releasing the object at a precise moment. Methods have been developed in the literature to perform this dynamic task by constraining the motion characteristics. For example, in [2], [23], [24], [27] the authors defined the type of trajectory (linear) and other task characteristics as the release point or the release instant.

To define the motion of our system we took inspiration from one of the most popular weapons of ancient times, the sling. Fig. 1(a) describes the procedure for throwing a projectile with a sling. After loading the projectile into the sling, a rotational motion of the wrist causes the sling to spin rapidly. After a few revolutions, the slinger releases the free end of the sling, and the projectile flies off on the tangent to the circle described by the sling [28].

Inspired by the sling, we decided to have the robot move in a circular trajectory while holding an object, as shown in Fig. 1(b). Since there is a delay associated with the gripper, we used a predictor to estimate the landing position of the object if it had been released after the opening delay. Based on the comparison

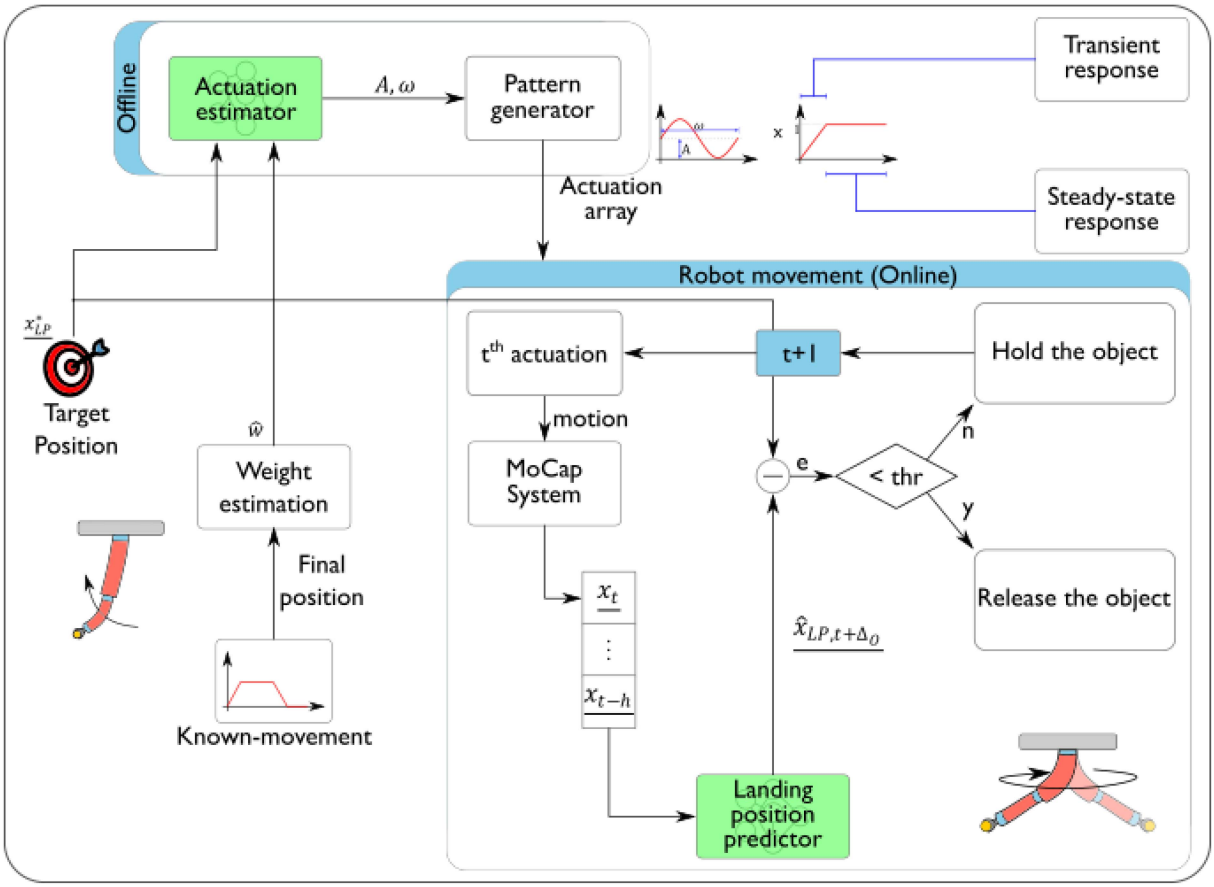

Fig. 2. *Throwing strategy*. The strategy is based on the desired position and the weight of the object held by the robot, which are given as input. The procedure is divided into two main steps. In the first offline step, the actuation estimator predicts the actuation parameters to follow the most appropriate periodic path. Given the periodic input provided by the pattern generator, the soft robot reaches a dynamic stable state. When this is reached, the second phase begins. This last phase occurs in real-time as the robot moves while the MoCap system tracks the object's position. During this phase, the landing position estimator, which takes into account the gripper opening delay Δ_O, predicts the landing position of the object at every t-th step based on a short history of the position of the gripped object ($\underline{x_t}$,..., $\underline{x_{t-h}}$). The object is either released or not released based on the distance between the predicted and the desired position. The movement continues if the "hold the object" condition is met ($t \rightarrow t+1$).

between this prediction and the desired target position, we decide whether to release the object or continue.

The whole procedure to perform this task is outlined in Fig. 2. The kinematics of a soft robot is strongly influenced by the payload attached to it. If the weight of the held object is unknown, it can be estimated by a regression based on the last position reached after a known movement. To estimate the parameters of the actuation inputs needed by the robot to follow the correct circular path, we must use both the desired position coordinate and the payload. The estimation is done by a double hidden-layer feed-forward neural network, which we call the actuation estimator. Once the robot starts moving, we track the position of the held object with a motion capture (MoCap) system. We feed this information to the landing position estimator to predict the landing position of the object if we were to release it after the gripper opening delay Δ_o. This estimator is a feed-forward neural network with two hidden layers. Using this prediction, we determine whether we should release the object now, despite intending to release it after Δ_o, or keep moving to find better conditions.

B. Experimental Setup

The control scheme shown in Fig. 2 has been tested on an experimental setup comprising three main components. These components are: (I) a robotic platform, which involved a sequence of the I-SUPPORT robotic arm [29] and a vacuum cup deployed as a gripper; (II) an actuation unit responsible for providing input signals to the platform; and (III) sensors, specifically a vision-based MoCap system from VICON Motion Capture Ltd with 8 Bonita cameras.

The I-Support robotic arm is a modular, soft manipulator [29]. Three McKibben-like actuators are integrated into each module and are positioned at 120° intervals from each other. For this study, two modules have been connected with a plexiglass interface that had a width of 3 mm. The modules are oriented with a 60° offset relative to one another. The modules are tapered to minimize the overall system weight. The proximal module uses pairs of McKibben-like actuators, while the distal module employs single chambers. In the experiments, only the proximal module is actuated, while the distal module remains passive. To control the pneumatic actuation of the robot, we used

three proportional pressure micro regulator valves (specifically Camozzi K8P-0-E522-0).

The robotic arm is attached to an aluminum support frame and is oriented downwards to counteract the effects of gravity. The robot arm measures a total length of 409 mm and weighs 230.5 g when it is in its relaxed state. It is suspended 1 m above the ground. The target container is positioned beneath the robot. The target is a square container measuring $[140 \times 140 \times 100]$ mm.

The gripper is one of the main components of the system. In the experiments, it is necessary to toss the object, and the interface between the robot and the object is crucial. To create the vacuum gripper, a silicone cup is molded and then utilized with a vacuum pump (Schwarzer Precision – SP570 EC-BL). The cup is crafted using DragonSkin30. The design resembles a hollow sector-of-sphere with a 20 mm inner radius. The vacuum cup is attached to the end effector of the soft robot. It adds 5 g to the robot's weight.

The time required to release the object has a massive impact on the experiments because it affects the landing position estimator. Preliminary studies were conducted to define this quantity. We used objects with different weights (7.10–19.33–33.37–46.47–61.58 g). To simplify, we will refer to the last two objects as weighing 45 and 60 g, respectively. Each object is a sphere with a diameter of 42 mm. We conducted three trials for each object, in which we let the robot move along a circular path for 25 s, after which we released the object. For each trial, the duration between sending the command and releasing the object was measured. The object is considered released when its distance from the end effector is 5% greater than before the release command. The releasing time is dependent on the weight of the object, as shown in Fig. 3. We chose a release time of 0.10 s, making it feasible for us to handle objects weighing over 45 g.

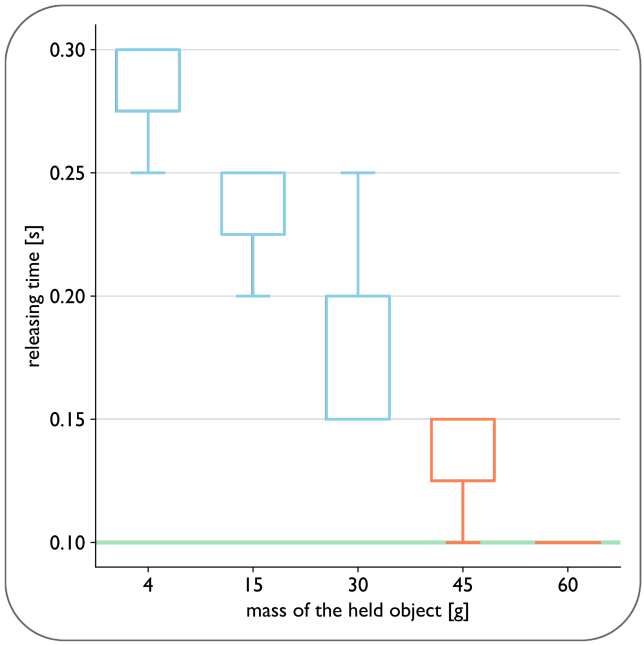

Fig. 3. *Characterization of gripper dynamics*. Release delay of the vacuum gripper with objects of different weights. The release delay depends on the weight of the object and it has an intrinsic variability. As a reference, we have chosen an opening delay of: $\Delta_O = 0.10$ s. Considering this value and the actuation frequency (20 Hz) of the system, the objects on which this delay is valid are those with weights of 45 and 60 g.

C. Dataset Collection and Artificial Neural Network Training

To produce a circular pattern with the I-Support robot Fig. 4(a), 120° phase-shifted sinusoidal pressure signals must be sent to its chambers. We kept the distal module passive, i.e., not actuated, to simplify the system's actuation process. Equation (1) describes the input values that are sent to the i-th chamber over time.

$$p_i(t) = A(1 + \sin(\omega t + \phi)) \tag{1}$$

Each sine wave is described by two parameters: amplitude A and angular velocity ω. During our experiments, we used a ramp signal to modulate this motion to avoid any sudden response by the robot. As shown in Fig. 4(b), this modulation enables a prolonged transient period prior to achieving the circular path. This was merely our proposal to achieve a steady condition. This is similar to the behavior that inspired us, the sling. As in the ancient weapon, after a transient phase, the object reaches a stable but high velocity regime. This embodied behavior of soft robots when subjected to periodic input was already demonstrated by [26] and then emphasized by [9]. Both suggest that when a soft robot is subjected to periodic input, it will eventually converge to a periodic orbit over a sufficient period of time. Among the different combinations of amplitude and angular velocity tested, Fig. 4(a) and (b) show the case where the maximum velocity is reached. It is worth noting that this is one of the highest speeds recorded with this type of robotic arms (∼2 m/s) [22].

TABLE I
SINEWAVE PARAMETERS. VALUES OF AMPLITUDE AND ANGULAR VELOCITY USED IN THE DATASET COLLECTION PHASE

Amplitude [bar]	0.40	0.45	0.50	0.55	0.60	
Angular velocity [rad/s]	5.5	6	6.5	7	7.5	8

To collect the dataset, we used the objects of 45 and 60 g. For each weight, we consider different combinations of amplitude and angular velocity. The values used are collected in Table I. We divided the combinations into six categories to increase the variability and reduce the conditioning of the dataset. We kept the angular speed fixed and randomly varied the amplitude between the values listed. The robot is actuated and the MoCap system run at 20 Hz. The total duration of each trial is 45 s, of which the first 15 s are used for the ramp modulation. For every attempt, the weight of the object (w_i), its position ($\underline{x_O}$), and actuation parameters were recorded. After collecting the dataset, we used the projectile equations to derive the landing positions ($\underline{x_{LP}}$). We only considered the influence of gravity, disregarding other factors such as air drag. The resulting dataset is represented in Fig. 4(c). We trained the neural networks shown in Fig. 2 using the new dataset.

In this task, the robot is given the weight of the object and the desired landing position. Using the method illustrated in Fig. 2,

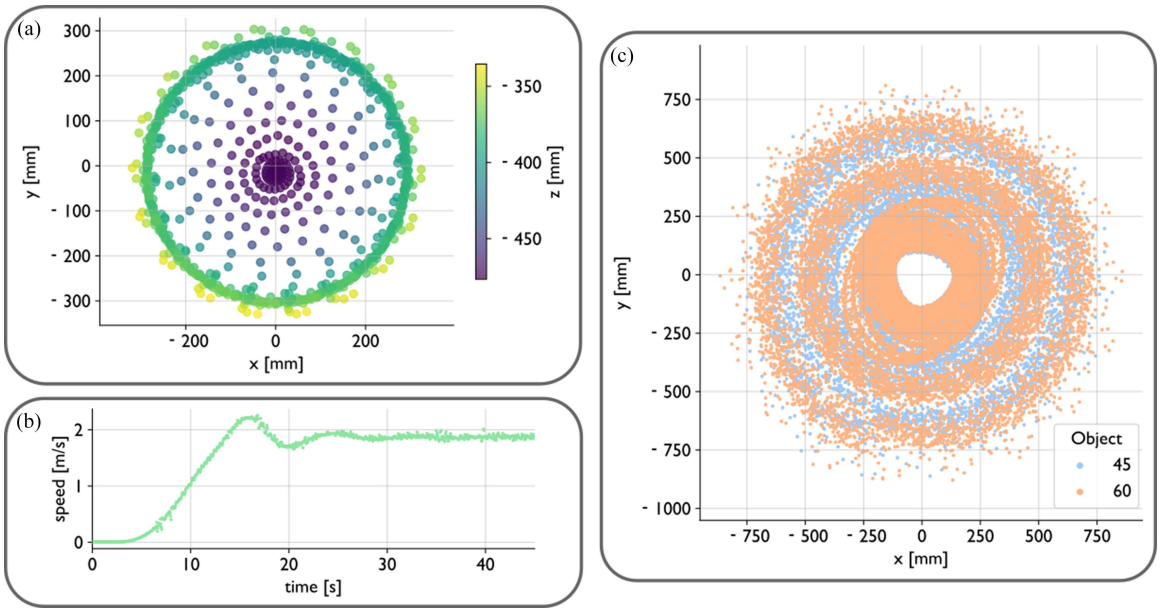

Fig. 4. *Dataset*. (a) Circular Trajectory. This is one of the many trajectories included in the training dataset. With respect to (1), it is characterized by an amplitude (A) of 0.50 bar and an angular velocity (ω) of 6.0 rd/s. After a transient phase, the robot stabilizes on a circular path. (b) Velocity Trend. Trend of the norm of the speed of the object during the trajectory shown in (a). It is possible to notice a transient in the velocity (in terms of the norm). In its dynamic stable state the robot reaches an outstanding speed of ∼2 m/s. (c) Dataset used to train the neural networks. Two objects have been taken into account in the data collection phase, in particular, the ones that weigh 45 and 60 g.

TABLE II
MODEL SELECTION. BEST COMBINATION OF HYPERPARAMETERS FOR THE ACTUATION ESTIMATOR AND THE LANDING POSITION PREDICTOR

Model	**Actuation estimator**	**Landing position predictor**
Number of units	128	128
Activation function	ReLU	ReLU
Number of hidden layers	2	2
Normalization (I&O)	Standard	Standard
Performance (MAE)	0.043 bar 0.249 rad/s	13.86 mm

the robot can then throw the object into the target container. The initial step involves determining the combination of actuation parameters needed to move the object along a circular path. To achieve this, we utilized the actuation estimator, which is an approximation of (2).

$$A, \omega = f(w, \underline{x^*_{LP}}) \tag{2}$$

(2) is derived by a feedforward neural network that predicts the actuation parameters (A, ω) using the weight of the object (w) and the desired landing position coordinates($\underline{x^*_{LP}}$). To choose the best set of hyper-parameters, we performed model selection based on the mean absolute error in the two values. The best combination is described in Table II.

Thanks to the prediction of (2), the system can start moving along a circular path. However, it is also necessary to determine whether to release the gripper or not. This led to the need for the landing position predictor, which approximates (3).

$$\underline{\hat{x}_{LP,\Delta_O}} = g(\underline{x_t}, \ldots, \underline{x_{t-h}}) \tag{3}$$

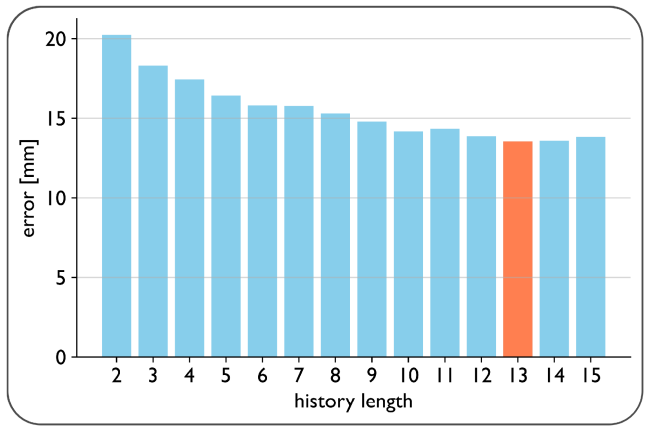

Fig. 5. *Model selection for the input size of the landing position predictor*. The performance of the predictor, as measured by mean absolute error (Euclidean distance between the predicted and the reference values), varies based on changes in input size dimensions.

(3) is derived with a feedforward neural network that predicts the landing position ($\underline{\hat{x}_{LP,\Delta_O}}$) of an object tracked by the MoCap system when released after a 0.10 s gripper releasing delay. The network uses a brief history of the object's positions ($\underline{x_t}$,..., $\underline{x_{t-h}}$) as input. We performed a double model selection to determine the optimal combination of hyperparameters for the neural network and the number of input positions. Both model selections were based on the Euclidean distance. The results are summarized in Table II and Fig. 5, respectively.

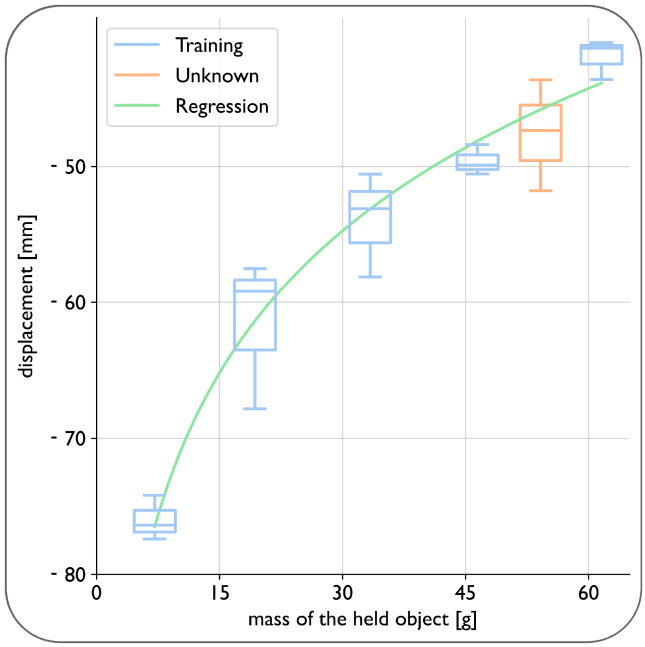

Fig. 6. *Weight estimation*. The change in displacement along x of the end-effector due to the change of the held objects can be modeled through a logarithmic curve. The same can be employed to estimate the weight of the object, which is one of the required input to predict the sinewave parameters, as shown in Fig. 2.

D. Weight Estimation

As described in Fig. 2, if the weight of the object to be thrown is not known, there is a way to estimate it. The estimation method exploits regression and the soft arm's characteristics, whose dynamics rely on the attached payload (see [30], [31]). Essentially, if the object's mass increases, the robot's workspace tends to decrease. For each object used in the gripper characterization phase, the robot performed a predefined movement three times. In terms of actuation, only one chamber is inflated at 1 bar for 30 s. To minimize arm oscillation, a ramp signal was utilized for half of the trial while maintaining constant pressure for the remainder. The MoCap system tracked the final position of the soft arm's end effector.

Fig. 6 illustrates the end effector's displacement along the x-axis from its initial position with varying weights. The x-axis is perpendicular to the robot axis and passes through the inflated chamber. We utilized a regression to enable the reconstruction of the weight of an unknown object by replicating the experiment. This procedure was utilized to estimate the weight of an *unknown* object that was later used along with the 45 and 60 g projectiles to test the method. The predicted weight was 48.94 g, while the measured one was 54.16 g. This prediction served as input for (2).

III. Results

Our results indicate that the SoftSling strategy achieves a performance in successfully throwing objects to target boxes as high as 75−88%, depending on the weight of the held object. The robot performs better on this task than a real-time RL-based controller [24] in terms of both success rate and throwing distance. Additionally, as it does not require the added training of an RL agent, we only require a fraction of the time from data collection to throwing.

We conducted experiments with three distinct objects, two of which had known weights (the 45 g and 60 g spheres) and the other being unknown. We chose eight targets near the robot, on which we placed four markers at each vertex for locating with the MoCap system. For each combination of parameters (object type-target location), three trials were completed resulting in a total of 72 throws. We tracked the object trajectory and recorded the success or failure of each trial. Supplementary Material includes some of these recordings.

The potential results of the throws can be qualitatively categorized as follows:

- Entered (E): the object lands within the target box.
- Hit (H): the object is thrown in the correct direction, but it hits the box without landing inside.
- Outside (O): the object is thrown in the wrong direction.
- Not thrown (N): this occurs when the throwing conditions are not met.

The evaluation metrics that we used are:

1) *Success rate:* the percentage rate of successful throws (E) relative to the total number throws.
2) *Distance from the center of the box*, calculated as the centroid given the markers on the target box.

The controller's performance is presented in Fig. 7 through a summary of evaluation metrics. Both qualitative and quantitative information, including the impact of different thrown objects, is reported.

Fig. 7(a) confirms that the controller can effectively throw objects in all directions without preferring any particular direction. Yet, this strategy appears to fail when the lightest object is utilized, as indicated by the chart. However, it is evident from Fig. 7(b) that even with the object having the lowest success rate, it either lands inside the target container (E) or hits the box before falling outside (H) when released. If the releasing conditions are not met, the object is not thrown (N). This outcome has only been observed in one trial, and the robot has never intentionally thrown the ball in the wrong direction (O). The strategy achieves success rates of 75%, 88% and 83%, respectively, with objects weighing 46.47, 61.58, and 54.16 g.

The error in Fig. 7(c) refers to the distance between the centroid of the target container and the landing position of the projectile across the different trials and targets. In Fig. 7(c), a violin plot is displayed. A box plot is included within each violin plot, using the same data. Only cases in which the object is released are considered; therefore, results from cases (just one) where the projectile is not thrown have been removed and replaced with the median value of the other results. This graph illustrates that the error distributions are alike in all three cases. Additionally, the median values for all cases are approximately ∼52 mm, which is slightly above the threshold of 50 mm that we established.

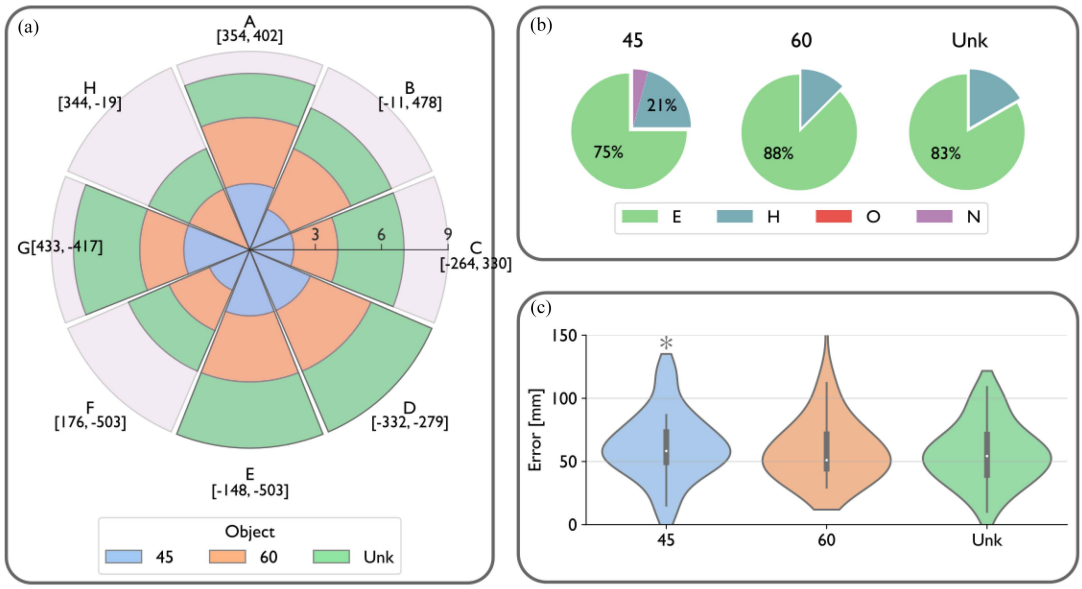

Fig. 7. *Qualitative and quantitative results*. The SoftSling control strategy was tested during experiments with three different objects. Two of these objects, weighing ∼45 g and ∼60 g, were also used during the training phase. The third object, whose weight was unknown (54 g), was used as a test of the controller generalization capability. (a) Number of successful attempts for each target (coordinates are in millimeters). A successful throw is one in which the object lands inside the target box. This chart displays the strategy's capability to throw objects in varying directions. The successful trial count for each object is denoted by the radius of the angular sector. Since three attempts have been made for each projectile, there are a maximum of nine tosses for each direction. (b) Qualitative results with different objects. The pie charts, one for each projectile, summarize the qualitative results obtained. Four outcomes have been considered: (E) Entered - successful attempt, the object lands directly inside the target box; (H) Hit – correct direction but the object hits the target box without landing inside; (O) Out - the object is thrown in a wrong direction; and (N) Not thrown - the throwing conditions were never reached (see Fig. 2). (c) Comparison of controllers for different objects based on distance in millimeters from the target position. The asterisk on the distribution of object 45 indicates the existence of an outlier (an attempt that was not thrown) that has been replaced by the median value.

TABLE III
COMPARISON WITH THE STATE OF THE ART

Characteristic	**[24]**	**[25]**	**[24]**	**SoftSling**
Soft arm controller	optimization	supervised learning	reinforcement learning	periodic motion pattern
Throwing area [mm]	~ 300	~ 300	~ 300	~ 600
Duration of the throw [s]	~ 60	2	2	16.0 ± 6.4
Real time	✗	✓	✓	✓
Number of different objects	4	4	4	3
Number of attempts	3	3	3	3
Number of target positions	10	10	10	8
Mean success rate across objects [%]	63	68	63	82

IV. CONCLUSION

In this letter we present SoftSling, a control strategy, inspired by the slingers, to perform the throwing task with a soft robotic arm after a circular run-up. This ability has been demonstrated by performing experiments with different weights, with which we achieved a success rate ranging from 75% to 88%. The lower value has been obtained with the lightest object (45 g), while the upper limit is achieved with the heaviest object. This discrepancy may be due to the delay in releasing the vacuum cup used as the gripper. In fact, even a delay (or advancement) of just a few tenths of a second in releasing the object, which rotates at high speeds, can determine the success or failure of a trial. Indeed, slingers must undergo tough training before they can master this skill [28].

However, performance seems to improve as the weight of the object increases, reaching 88% with the heaviest weight. The choice of releasing time during our experiments may have contributed to this outcome. Fig. 3 illustrates that this variable is weight-dependent. As we established the reference point using the heaviest object, any reduction in weight leads to reduced performance. Moreover, the impact of regression error resulting from estimating the unknown weight appears to be negligible, as we achieved performances that are comparable to those obtained with objects in the training phase. The achieved success rates surpass those found in the state-of-the-art using a modular soft robot, as shown in Table III. While other works in the literature have demonstrated the feasibility of the task [21], [22], we exploit the embodied intelligence of soft robots to throw objects to shift our attention from the robot motion to the gripper releasing. Indeed, we did not define the trajectory over time because the robot automatically stabilizes on a periodic motion, which gave us the opportunity to focus only on the release instant. We compared this strategy with other controllers designed for this task available in the literature. [24] presented a real-time open-loop controller based on RL to achieve linear trajectories of throws, with a success rate of about $\sim 62 \div 63\%$. Similar performance has been achieved using a learning-based inverse

dynamic controller in [25] and with an optimization-based controller (non-real-time), as demonstrated in [24]. The authors compared the RL-controller previously mentioned with one tested only in simulation in [23]. Our work has achieved a more than 15% increase in controller overall accuracy compared to the first throwing controller developed for a soft robot. Then, objects weighing up to 6 times more than the heaviest item used in [24] are thrown over a larger distance. In this instance, the maximum throwing distance of the target from the robot's initial position is ∼600 mm, as opposed to ∼300 mm in the state-of-the-art. Additionally, while the robot's initial workspace is only a small fraction of the rigid robot's one used in [2], the success rate is comparable. In this study utilizing a conventional manipulator, the achievement of successful pick-and-throw of various objects was recorded at 82.3%. The authors noted that this figure is significantly affected by the effectiveness of the grasping phase. It is worth mentioning that in our own experiment, we positioned the object manually underneath the vacuum cup for grasping.

In contrast to [24], another important aspect of our strategy is its lack of missed throws or preferred directions. The results, as illustrated in Fig. 7(b), show that the *outside* condition never occurs. This means that when the projectile is thrown, it will hit the target container, even if the likelihood of entering the box is considerably higher. This makes our strategy fail-safe because, when the throwing conditions are not met, the soft robot will continue to hold onto the object. This can be viewed as an inherent safety feature of our approach.

This highlights a potential improvement in the SoftSling control strategy. By incorporating dynamic planning of the circular run-ups, the workspace can be explored more effectively when the threshold condition is not met. This aims to relocate the object along a different circular trajectory, to perform a successful throw. In addition, to ensure consistent performance across objects of different weights, it is necessary to develop a controller that automatically adjusts the gripper opening delay based on the weight of the object.

Author Disclosure Statement

All authors declare that they have no conflicts of interest.

References

[1] H. Frank, D. Barteit, and F. Kupzog, "Throwing or shooting - a new technology for logistic chains within production systems," in *Proc. IEEE Int. Conf. Technol. Practical Robot Appl.*, 2008, pp. 62–67.

[2] A. Zeng, S. Song, J. Lee, A. Rodriguez, and T. Funkhouser, "TossingBot: Learning to throw arbitrary objects with residual physics," *IEEE Trans. Robot.*, vol. 36, no. 4, pp. 1307–1319, Aug. 2020.

[3] Z. Fang, Y. Hou, and J. Li, "A pick-and-throw method for enhancing robotic sorting ability via deep reinforcement learning," in *Proc. 36th Youth Academic Annu. Conf. Chin. Assoc. Automat.*, 2021, pp. 479–484.

[4] F. Raptopoulos, M. Koskinopoulou, and M. Maniadakis, "Robotic pick-and-toss facilitates urban waste sorting," in *Proc. IEEE 16th Int. Conf. Automat. Sci. Eng.*, 2020, pp. 1149–1154.

[5] D. Büchler, R. Calandra, and J. Peters, "Learning to control highly accelerated ballistic movements on muscular robots," *Robot. Auton. Syst.*, vol. 159, 2019, Art. no. 104230.

[6] D. J. Braun, M. Howard, and S. Vijayakumar, "Exploiting variable stiffness in explosive movement tasks," *Robotics: Sci. Syst. VII*, vol. 7, pp. 25–32, 2012.

[7] D. Rus and M. Tolley, "Design, fabrication and control of soft robots," *Nature*, vol. 521, pp. 467–75, May 2015.

[8] A. Sakes, M. van der Wiel, P. W. Henselmans, J. L. van Leeuwen, D. Dodou, and P. Breedveld, "Shooting mechanisms in nature: A systematic review," *PLoS One*, vol. 11, 2016, Art. no. e0158277.

[9] C. Laschi, "Embodied intelligence in soft robotics: Joys and sorrows," *IOP Conf. Series: Mater. Sci. Eng.*, vol. 1261, Oct. 2022, Art. no. 012002.

[10] S. Seok, C. D. Onal, K.-J. Cho, R. J. Wood, D. Rus, and S. Kim, "Meshworm: A peristaltic soft robot with antagonistic nickel titanium coil actuators," *IEEE/ASME Trans. Mechatron.*, vol. 18, no. 5, pp. 1485–1497, Oct. 2013.

[11] S. Kim, M. Spenko, S. Trujillo, B. Heyneman, D. Santos, and M. R. Cutkosky, "Smooth vertical surface climbing with directional adhesion," *IEEE Trans. Robot.*, vol. 24, no. 1, pp. 65–74, Feb. 2008.

[12] C. Laschi, M. Cianchetti, B. Mazzolai, L. Margheri, M. Follador, and P. Dario, "Soft robot arm inspired by the octopus," *Adv. Robot.*, vol. 26, no. 7, pp. 709–727, 2012.

[13] B. Gorissen, D. Melancon, N. Vasios, M. Torbati, and K. Bertoldi, "Inflatable soft jumper inspired by shell snapping," *Sci. Robot.*, vol. 5, May 2020, Art. no. eabb1967.

[14] Y. Chen et al., "Controlled flight of a microrobot powered by soft artificial muscles," *Nature*, vol. 575, pp. 324–329, Nov. 2019.

[15] T. Ranzani, G. Gerboni, M. Cianchetti, and A. Menciassi, "A bioinspired soft manipulator for minimally invasive surgery," *Bioinspiration Biomimetics*, vol. 10, May 2015, Art. no. 035008.

[16] S. Terryn, G. Mathijssen, J. Brancart, D. Lefeber, G. V. Assche, and B. Vanderborght, "Development of a self-healing soft pneumatic actuator: A first concept," *Bioinspiration Biomimetics*, vol. 10, Jul. 2015, Art. no. 046007.

[17] E. W. Hawkes, L. H. Blumenschein, J. D. Greer, and A. M. Okamura, "A soft robot that navigates its environment through growth," *Sci. Robot.*, vol. 2, Jul. 2017, Art. no. eaan3028.

[18] M. S. Nazeer, C. Laschi, and E. Falotico, "RL-based adaptive controller for high precision reaching in a soft robot arm," *IEEE Trans. Robot.*, vol. 40, pp. 2498–2512, 2024.

[19] C. Laschi, T. G. Thuruthel, F. Lida, R. Merzouki, and E. Falotico, "Learning-based control strategies for soft robots: Theory, achievements, and future challenges," *IEEE Control Syst. Mag.*, vol. 43, no. 3, pp. 100–113, Jun. 2023.

[20] C. D. Santina, C. Duriez, and D. Rus, "Model-based control of soft robots: A survey of the state of the art and open challenges," *IEEE Control Syst. Mag.*, vol. 43, no. 3, pp. 30–65, Jun. 2023.

[21] O. Fischer, Y. Toshimitsu, A. Kazemipour, and R. K. Katzschmann, "Dynamic task space control enables soft manipulators to perform real-world tasks," *Adv. Intell. Syst.*, vol. 5, 2022, Art. no. 2200024.

[22] D. A. Haggerty et al., "Control of soft robots with inertial dynamics," *Sci. Robot.*, vol. 8, no. 81, 2023, Art. no. eadd6864.

[23] D. Bianchi, M. Antonelli, C. Laschi, and E. Falotico, "Open-loop control of a soft arm in throwing tasks," in *Proc. 19th Int. Conf. Inf. Control Automat. Robot.*, 2022, pp. 138–145.

[24] D. Bianchi, M. G. Antonelli, C. Laschi, A. M. Sabatini, and E. Falotico, "SofToss: Learning to throw objects with a soft robot," *IEEE Robot. Automat. Mag.*, early access, Sep. 29, 2023, doi: 10.1109/MRA.2023.3310865.

[25] D. Bianchi, M. Antonelli, C. Laschi, A. M. Sabatini, and E. Falotico, "Learning-based inverse dynamic controller for throwing tasks with a soft robotic arm," in *Proc. 20th Int. Conf. Inf. Control Automat. Robot.*, Nov. 2023, pp. 424–432.

[26] T. G. Thuruthel, E. Falotico, M. Manti, and C. Laschi, "Stable open loop control of soft robotic manipulators," *IEEE Robot. Automat. Lett.*, vol. 3, no. 2, pp. 1292–1298, Apr. 2018.

[27] D. J. Braun, M. Howard, and S. Vijayakumar, "Exploiting variable stiffness in explosive movement tasks," *Robotics: Sci. Syst.*, vol. 7, pp. 25–32, 2012.

[28] M. Korfmann, "The sling as a weapon," *Sci. Amer.*, vol. 229, pp. 34–42, Oct. 1973.

[29] M. Manti, A. Pratesi, E. Falotico, M. Cianchetti, and C. Laschi, "Soft assistive robot for personal care of elderly people," in *Proc. 6th IEEE Int. Conf. Biomed. Robot. Biomechatronics*, 2016, pp. 833–838.

[30] A. Centurelli, L. Arleo, A. Rizzo, S. Tolu, C. Laschi, and E. Falotico, "Closed-loop dynamic control of a soft manipulator using deep reinforcement learning," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 4741–4748, Apr. 2022.

[31] F. Piqué, H. T. Kalidindi, L. Fruzzetti, C. Laschi, A. Menciassi, and E. Falotico, "Controlling soft robotic arms using continual learning," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 5469–5476, Apr. 2022.