MITSUBISHI ELECTRIC RESEARCH LABORATORIES
https://www.merl.com

Stochastic Learning Manipulation of Object Pose With Under-Actuated Impulse Generator Arrays

Kong, Chuizheng; Yerazunis, William S.; Nikovski, Daniel

TR2023-151 December 20, 2023

Abstract

Robotic assembly systems are common in modern industry and a fixture of commerce. However, the robots themselves lack the adaptability of humans in terms of sin- gulating and grasping parts with uncontrolled pose. To this end, vibratory bowl feeder (VBF) devices are often employed to pre-orient the part for robot grasping. Unfortunately, VBFs themselves are inflexible (usually bespoken for one specific part), noisy, and very expensive to design and tune. We consider an alternative to the VBF — an array of impulse-generating solenoids positioned under a semi-rigid part-carrying platform that uses computer vision and self-supervised machine learning to generate a policy implementing a closed-loop controller to orient randomly positioned parts into a pose acceptable for robot grasping. Using a flat square wooden nut from a child's assembly toy as a test object, we were able to flip the nut into the desired orientation (standing vertically on the narrow edge) 21.1% of the time with a single impulse, and 35.4% of the time with two impulses, versus just 10.2% and 19.2% (respectively) of the time for a baseline policy of random choice of solenoid position and impulse duration, thus demonstrating black-box control of a process commonly considered too difficult to physically model.

International Conference on Machine Learning and Applications (ICMLA) 2023

Mitsubishi Electric Research Laboratories, Inc.
201 Broadway, Cambridge, Massachusetts 02139

Stochastic Learning Manipulation of Object Pose With Under-Actuated Impulse Generator Arrays

C. Kong
Mitsubishi Electric Research Labs
Cambridge, Massachusetts, 02139
email: kczttm@gmail.com

W. S. Yerazunis
Mitsubishi Electric Research Labs
Cambridge, Massachusetts, 02139
email: yerazunis@merl.com
correspondence author

D. Nikovski
Mitsubishi Electric Research Labs
Cambridge, Massachusetts, 02139
email: nikovski@merl.com

***Abstract*—Robotic assembly systems are common in modern industry and a fixture of commerce. However, the robots themselves lack the adaptability of humans in terms of singulating and grasping parts with uncontrolled pose. To this end, vibratory bowl feeder (VBF) devices are often employed to pre-orient the part for robot grasping. Unfortunately, VBFs themselves are inflexible (usually bespoken for one specific part), noisy, and very expensive to design and tune. We consider an alternative to the VBF — an array of impulse-generating solenoids positioned under a semi-rigid part-carrying platform that uses computer vision and self-supervised machine learning to generate a policy implementing a closed-loop controller to orient randomly positioned parts into a pose acceptable for robot grasping. Using a flat square wooden nut from a child's assembly toy as a test object, we were able to flip the nut into the desired orientation (standing vertically on the narrow edge) 21.1% of the time with a single impulse, and 35.4% of the time with two impulses, versus just 10.2% and 19.2% (respectively) of the time for a baseline policy of random choice of solenoid position and impulse duration, thus demonstrating black-box control of a process commonly considered too difficult to physically model.**

***Index Terms*—Control under uncertainty, stochastic modeling, learning control**

I. Introduction

Typical industrial assembly robots are flexible in that they can be reprogrammed — for example, building blenders one day, and assembling toasters the next day. To perform this automated assembly of goods means the robot depends on the availability of parts already in a graspable orientation, perhaps assisted by a machine vision system controlling one undetermined axis (usually Z rotation). However, more complex part pose variation or failed singulation is poorly accommodated, and so a separate device, called a vibratory bowl feeder (VBF) [1] is often used to singulate and orient the parts for robot grasping. A VBF uses a circular vibration of a large bowl with a rising spiral ramp containing cutouts and other obstructions to shake the parts either into the desired orientation, or to nudge mis-oriented parts off the ramp and back into the bowl at the bottom leaving only correctly oriented parts.

Unfortunately, VBF units, by their passive filtering nature, are generally capable of feeding only a single part design. Most VBFs are custom built for the particular part by highly skilled and specialized designers, so typical VBFs are priced in the tens to hundreds of thousands of dollars and have delivery times measured in months. VBFs are physically large and (due to the powerful vibration) noisy when in operation. To achieve more flexibility, several commercial entities have marketed simplified parts feeders where the parts are deposited onto a flat surface that can be commanded to vibrate in a fixed pattern; the vibration hopefully singulates and orients at least one of the parts so that it can be recognized by an industrial machine vision camera and then grasped by the robot. This method reduces the cost and lead time of the feeder, as the vibration pattern is standardized and needs no customization.

These fixed-pattern part singulators solve part of the problem, but they do not address the situation of the part lying on the wrong facet. As the robot can only approach the part in a direction contained in the hemisphere above the platform at best, the situation remains that the robot would need a custom program to grasp and flip over the inverted parts, and then grasp again (again requiring highly skilled programmers to create a custom program). This grasp/flip/re-acquire/grasp again process imposes a time cost on the robot, increasing the *takt* time of the assembly process and slowing production.

We propose a novel design for part orientation as shown in Figure 1. An array of solenoids is mounted vertically beneath a semi-rigid plate supporting the parts; the solenoids are computer controlled and can impart impulses of varying amplitudes to the semi-rigid surface, hopefully flipping the parts into the preferred pose. An industrial computer vision camera determines the current pose of the part, and a machine learning system is trained in a self-supervised learning mode, generating a policy that inputs the part's current pose, and outputs which solenoid and what impulse duration has the best chance of putting the part into an acceptable pose for robot grasping, thus creating a truly adaptive part orientation system. Human intervention is needed only for the training of the computer vision (CV) system to recognize "correct" part poses (anything not "correct" is presumed incorrect); from that point onward the ML training and policy generation are entirely self-supervised, requires no human intervention, and typically completes in one unattended overnight session.

A recent study on a task related to ours (rolling a cube on

Fig. 1: The 7-solenoid impact manipulation system "Thumper"; the seven solenoids beneath a transparent support surface are in the foreground, with the controller board in the background

a flat surface) showed that even after very careful calibration, the behavior of systems with contact-rich, 6-DoF objects was largely unpredictable and not very consistent across multiple physics engines [2]. In contrast, the application of machine learning technology has the potential to capture precisely the effect of the actuators of the system on the manipulated parts entirely in a self-supervising fashion, eliminating the need for long and costly manual system modelling and controller design. Pursuing this same problem — the non-prehensile manipulation of a cube, in physical reality rather than simulation, showed that a classic machine learning technique (the *k-Nearest-Neighbor*, or *kNN*) [3], [4] was capable of controlling a die roll with successfully choosing the next face over 30% of the time with a single impulse, and over 50% of the time with two successive impulses [5].

Section II describes the background and prior work on probabalistic models. Section III describes the mechatronic impulse manipulator including the sensor instrumentation and calibration. Section IV describes the test part — a flat wooden nut from a child's toy set. Section VI describes our initial experimentation with the thumper-nut interactions, and Section VII describes a controller policy based on the exploratory information. Section VIII describes the experimental results obtained using this controller, and Section IX summarises these results and proposes further directions for research and improvements to the system.

II. Background and Prior Methods

Learning control methods have been researched extensively, and their success largely depends on the nature of the control problem being solved. Some of the earliest work on learning control was on manipulation of parts by means of tilting a tray. It was based on learning a predictive model of the effect of the chosen direction of the tray's tilt, and using this model for planning and control [6]. The machine learning formalism this model was based on was that of stochastic learning automata (SLA), [7], and discretized the state space of the manipulated part (position on the tray) into quite coarse regions. This matched well the usual assumption of SLA for relatively few discrete states, and made the learning problem tractable, but led to the introduction of additional uncertainty and stochasticity in the model due to partial state observability, on top of the already significant stochasticity of the system due to complex contact and impact dynamics. Even though our control problem bears strong similarity to the one in [6], it is likely that an approach that does not quantize the state into a few coarse discrete states would be more effective.

Another popular modeling methodology is to learn a full state-space model of the system dynamics, using various system identification methods [8]. Although this approach has been very productive for linear systems, the complicated non-linear nature of contact dynamics has required the application of advanced methods for learning non-linear and possibly hybrid discrete/continuous dynamical models. A number of machine learning models and algorithms have been used to learn system dynamics, and neural networks in particular have been investigated extensively for a long time [9]–[11]. Recent interest in model-based reinforcement learning has renewed research efforts to find good methods for learning world models [12]. Recently proposed Contact Nets have improved considerably the accuracy of predictive models with respect to earlier dynamical models based on standard neural networks [13], [14]. However, learning such models is usually quite complicated, if higher accuracy is desired to predict the entire future motion of the manipulated object. However, this is probably not even necessary for our problem, where only the final resting state of the object is of primary interest.

For this reason, we focused on learning predictive models that predict only the resting state of the manipulated part as a result of a particular action (solenoid fired). Similar to SLA, these predictive models are probabilistic, with the goal of capturing the inherent stochasticity of the complex contact dynamics involved. However, in departure from the SLA formalism, our models use the full continuous state of the manipulated part in order to predict the resting state. In other words, our models focus on representing the significant aleatoric (inherent) uncertainty (mostly due to chaotic bifurcation dynamics and contact phenomena), but do not need to deal with significant epistemic (observational) uncertainty. As there is no reason to introduce such epistemic

uncertainty by quantizing the state, a better approach might be to measure the continuous state as accurately as possible, and then employ suitable machine learning methods that can work with the full continuous state.

III. Mechatronic Design of the Experimental System

Our ”smart bowl” test system (nicknamed ”Thumper”) is shown in Figure 3. The system uses seven solenoids with radiused striking heads mounted vertically under a semi-flexible support surface. All seven solenoids are individually controllable as to timing and pulse duration via an Arduino Mega activating PowerFET drivers. A calibrated HD web-cam with a ring light captures the bowl state at 30 frames per second (fps) in a world coordinate frame; the host PC (running Debian Linux) uses OpenCV to determine the pose of the part; one of several ML methods can then be applied to generate a policy or use a saved policy to select a solenoid and an impulse duration to manipulate the part as desired.

For naming convenience, in this paper we will use the term ”thumper” to indicate the entire apparatus. We will also use the term ”thumper” with a number to indicate one of the seven sets of PowerFET, solenoid, and striker heads. The actual layout and numbering of the solenoids are shown in Figure 2.

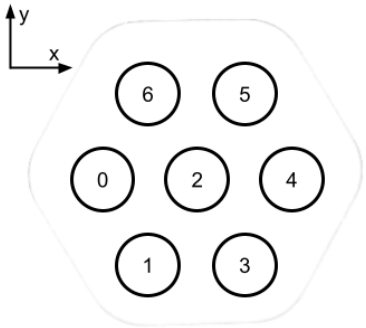

Fig. 2: The geometrical layout of the solenoid array as seen by the downfacing camera; the inter-solenoid spacing is 60mm.

The vocabulary of commands that this manipulation system can execute is very limited compared to a conventional robotic system. The command specifies only a solenoid number (0 to 6) and a duration in milliseconds. Durations of 8ms or less are all equivalent to no command at all, as the solenoid slug does not get enough energy to actually rise into contact with the part support surface; durations over 25ms are all equivalent, as the solenoid slug remains in contact with the solenoid’s core stop in the elevated position, but the impact plate continues upward and ceases contact with the solenoid, so no further energy can be transferred.

In all, the command vocabulary contains, at best, 7 solenoid choices multiplied by 17 impulse durations or a total of 119 possible commands. The mechatronic parts of the system can be seen in Figure 3 and the assembled system in Figures 4 and 5.

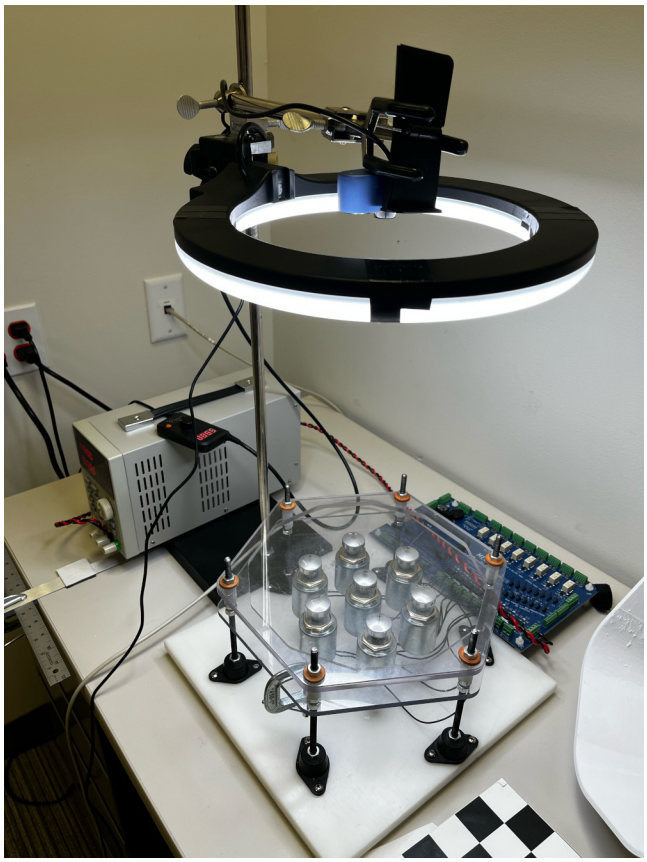

Fig. 3: The Thumper showing the ringlight and downfacing camera

To confine the parts onto the bowl floor, a white PETG 3-D printed hexagonal corral is mounted 5mm above the bowl floor. The corral is easily removed for calibration and is only needed when in operation, as shown in Figure 5.

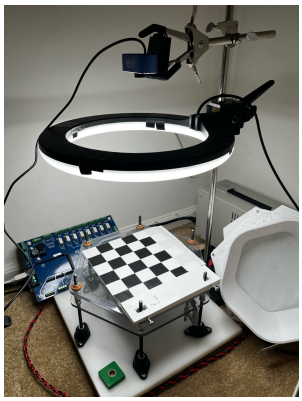

Fig. 4: Thumper with calibration jig

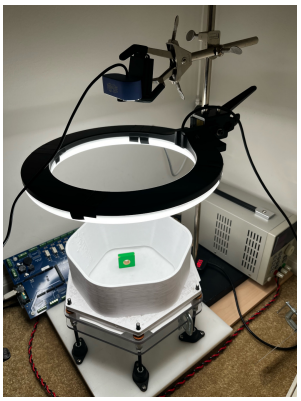

Fig. 5: Thumper in operation mode

IV. Test Article: Flat Wooden Nut

To evaluate this device as a robot part feeder, we chose the manipulation task of standing an imprecise flat square nut from a children's assembly set. Its behavior is similar to that of nuts and bolts with hexagonal shape, but its size and shape make its pose easier to estimate from camera images. (This part of the experiment has only a supporting role, and in a real factory deployment this function will be performed by an industrial machine vision system that can easily handle more complicated shapes.) However, increasing the challenge, unlike a regular nut or bolt, the nut is imprecise in almost every dimension with side lengths varying by $0.5-1$ mm. These conditions of the nut has led to several issues:

1) As the nut is rather thin (about 10mm thick), the desired goal state of the nut standing on edge is metastable and is statistically rare (thermodynamically unfavorable), even if there was adequate kinetic energy supplied in the solenoid impulse. This leads to heavily imbalanced training classes, with many more failures than successes ($Psucc \sim 13\%$).
2) Since the nut is slightly asymmetric, the force needed to stand the nut is different on each edge; however, it is hard for the vision system to identify such asymmetry on every location of the impulse bowl. This ambiguity leads to some degree of inaccuracy in the training process.
3) We also found that the apparatus must be well levelled and not tilted, as a tilt as small as 0.4° will cause the nut to "migrate" to the lowest corner of the apparatus.

Of these issues, the asymmetry issue was underestimated in our initial exploration; the consequences of this will be described in more depth below.

V. Sensing and Control of the Solenoid Array

Setting up the system to stand the flat nuts on edge was relatively straightforward. We set up the ring light, webcam, and OpenCV system to track the state of the nut. We created a feature extraction pipeline that uses the width / length ratio and the area of the nut's top view to distinguish between the standing and laying status; an industrial automation camera would provide this directly. Figure 6 shows the CV system output identifying the standing and laying states of the 35mm flat square wooden nut. The standing state is our desired state for robotic picking.

VI. Exploratory Testing and Verification

Having built the necessary infrastructure for operation, we began exploratory testing to verify that the full system behaved according to our expectations. To gather a starting dataset, we ran 20,000 impulses, randomly choosing both the solenoid number (from 0 to 6) and the duration from the range 8 to 25 milliseconds, and saving the raw image from the camera. We found the OpenCV pipeline can label 99.6% of the cases as either "standing" (a successful manipulation, thus "positive") or "laying" (not successful, hence "negative"), leaving only 0.4% as "unknown" (most "unknown"

Fig. 6: OpenCV Recognized Flat Nut in standing and laying states.

examples are cases where the nut is leaning on the bowl's wall).

Each of the 20,000 sample images was accompanied by the returned data from the CV system, specifically the CV-determined start pose $(x, y, \theta,$ (stand / lie)), the commanded solenoid (uniform distribution from 0 to 6), the commanded impulse duration (uniform distribution, ranging from 8ms to 25ms), and the CV-determined new pose $(x_n, y_n, \theta_n,$ (stand OR lie)$_n)$ for the nut. Of the 99.6% cases labeled by the OpenCV system as either "standing" or "laying", there was a zero false-positive rate (false identification as standing up) and a ∼1.0% false-negative rate (false identification of laying down). We then verified the OpenCV system's saved image results manually for all 20,000 samples; this manually-verified dataset became our ground truth training data.

Analyzing the data, we found a soft threshold behavior — for durations less than 14ms, the success in standing the nut up was essentially zero. At pulse durations of 18ms or longer, the typical success rate over the range of tested impulse durations in the run of 20,000 was nearly constant. Additionally, the ground truth showed a severe class imbalance (worse than 7:1, favoring "laying"), as shown in Figure 7.

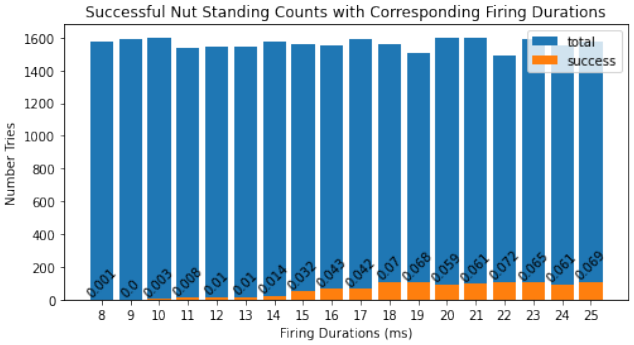

Fig. 7: Nut standing (success) counts and rates versus firing duration

We also found that, although the CV system could determine the (x, y) centroid of the square nut to about one millimeter repeatability and the gross inclination of one facet of the nut between 0 and 90° to within one degree, the

CV system was unable to resolve the position of the nut with respect to the eightfold corner symmetry of a flattened cuboid (that is, which of the eight corners of the flattened cuboid was in a particular corner of the image.)

To determine if this slight asymmetry in the flat nut would be significant, we used a three-point kinematic jig to place the nut at a fixed position (x, y, θ) laying down as shown in Figure 8 (a small dot of marker was added on one corner to allow a human to disambiguate the visual eightfold symmetry of the square nut).

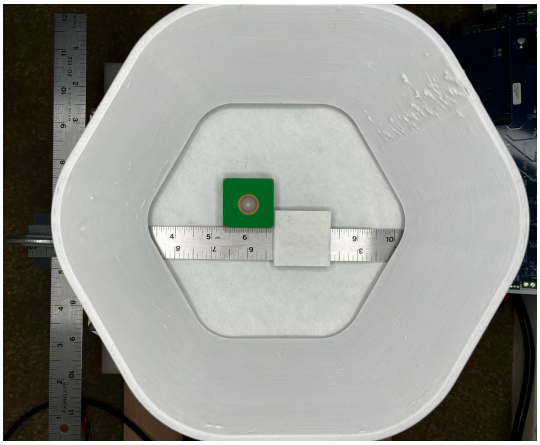

Fig. 8: Nut positioned with kinematic jig for symmetry importance testing.

We tested firing durations on the solenoid from 15ms to 25ms in increments of 1 ms. For every firing duration, we repeated the positioning/firing cycle five times and recorded whether the nut remained laying or stood up. We found a pose and firing duration and with a local maximum in the success rate, but when we rotated the nut 180 degrees on the Z axis and placed it back to its original (x, y, θ) location (representing the CV system's θ ambiguity), the best mode changed to a different firing duration, even while forcing the use of the same solenoid. Similar changes were seen in the success rates and preferred impulse durations for all eight possible nut orientations, showing that there was a fourth significant state variable representing the eightfold cuboid symmetry of the nut. For convenience we will call this variable s (for spin) and so the actual state of the nut is (x, t, θ, s) and (unfortunately) s is not observable by our current CV system.

This lack of observability of a demonstrably significant state variable implies several obstacles to be overcome:

- Difficulty in establishing an effective regression algorithm to pick the best impulse duration for the nut.
- Difficulty in determining the most effective solenoid to fire
- Even assuming a perfect choice (100% success rate) of solenoid number and duration exists, we have only a 1 in 8 probability of matching the unobservable spin of that perfect choice, so any success rate better than 1 in 8 implies multiple nonzero-valued solutions for solenoid number and firing duration.

In further exploratory runs using a random-firing strategy, we determined that the system did exhibit a sixfold symmetry with respect to the relative locations of the solenoid and the nut for a successful standing operation. Figure 9 shows the successful "standing" (x, y) impulse locations for 18, 20, and 23 millisecond impulses:

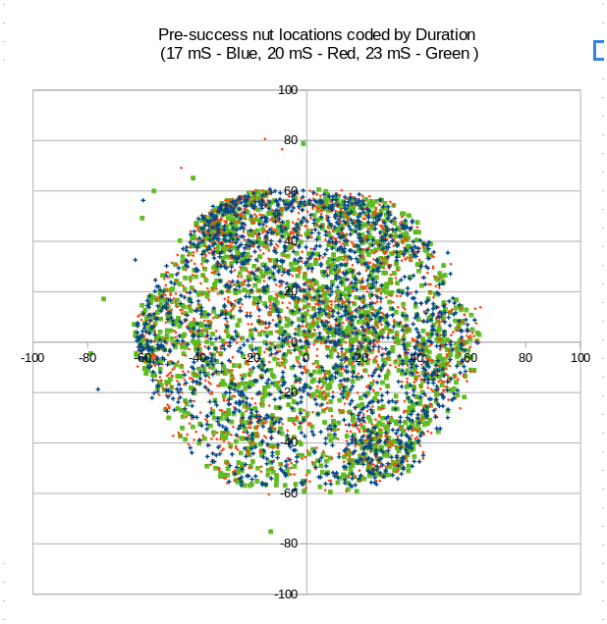

Fig. 9: Success locations for impulses of 17, 20, and 23 millisecond durations with the impulse chosen randomly.

In a total of 20,000 random firing, only 1,960 impulses were successful, giving an overall success rate of 0.098. From the result graph we see that the region of successful impulses exhibited a sixfold symmetry (as expected) and that variation of duration also had an impact, albeit a smaller impact. The sixfold symmetry is expressed in six visible clusters of nut centroids from which successful standing up was performed. As these clusters are about 60^o apart from each other, viewed from the center of the bowl, it can be surmised that this symmetry is related to the placement of the six solenoids below the plate, also at 60^o increments about the center of the bowl. Furthermore, it can be hypothesized that the nut would react similarly to an impulse of the same duration imparted by two different solenoids, if its position and angle are relatively the same with respect to coordinate frames with origin at the center and rotated by the angle between the lines connecting the positions of the two solenoids and the origin.

VII. Learning a Control Policy

Given this information, we could then make an informed choice of strategy. We considered two approaches:

1) train seven separate binary classifiers (one per solenoid) that outputs a probability of the nut changing its status from laying to standing, given its current state (x, y, θ)

2) train one binary classifier that utilizes the six-fold symmetry of the thumper bowl, combining all six peripheral solenoids together with suitable rotations to place the effective thumper at (-60,0). Output is also the optimal thumper to fire given the nut's current state.

Although the first approach provides seven classifiers that can account for the minor differences between each solenoid, the sparse and heavily imbalanced success / fail ($\sim$ 320 / 2500) samples would likely render poorly performing models. Choosing to trust that the actual thumper response is as symmetrical as Figure 9 suggests, we follow the second approach, effectively lumping all samples from the exploratory run of 20,000 into a single solenoid experiment and yielding 1,953 usable ground-truth successes out of 20,000 ground-truth samples, comfortably larger than the typical 1953 / 6 = $\sim$ 320 raw successes per solenoid of option 1.

Notice that the center thumper (thumper 2) was not included in this sample set, as its flexural environment is different from the peripheral thumpers; second, its class imbalance from the exploratory run was too severe (only 7 successes out of $\sim$ 2800 impulses applied to the center solenoid) to lend much significance.

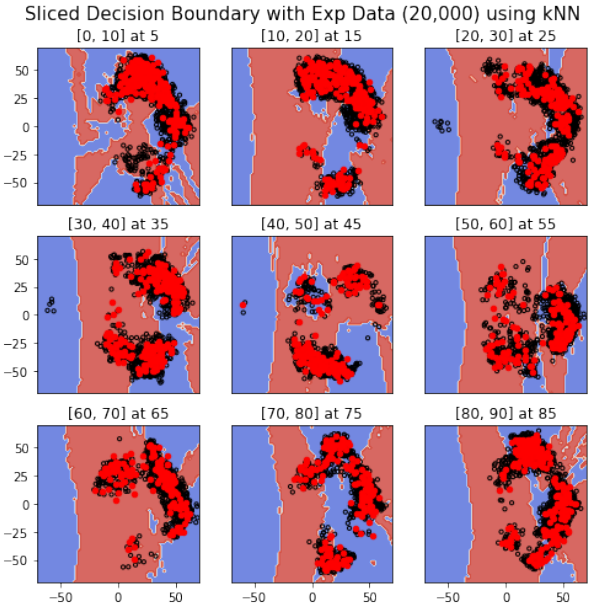

Fig. 10: Flat nut kNN decision boundaries sliced at 5, 15, ..., 85 degrees

We defined our core learning control method as a k Nearest Neighbor (kNN) [3], [4], with a weighted Euclidean distance $||(\Delta x, \Delta y, \Delta\theta)||_2$ defined in terms of the differences between the pose of a test object and that of an example in the training data set.

As the Δx and Δy have the dimensions of millimeters, but $\Delta\theta$ has the dimensions of degrees, we chose an input weighting of 1.0, 1.0, and 1.0 mm/deg (one degree of rotation is equivalent to 1mm of translation in the distance measuring

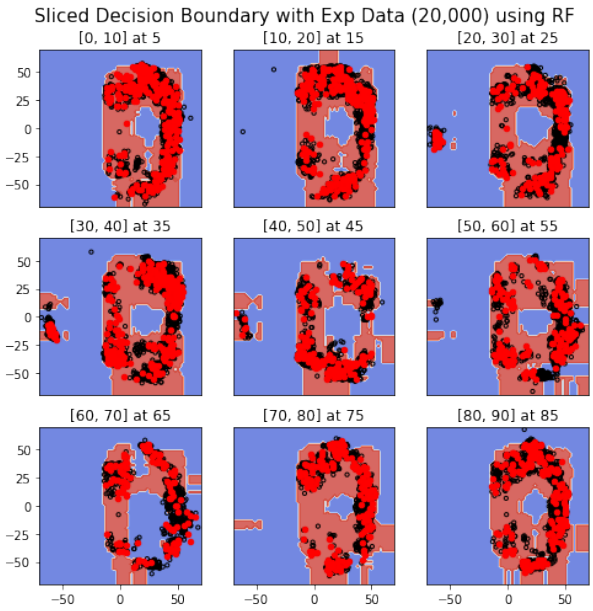

Fig. 11: Flat nut Random Forest Classifier decision boundaries sliced at 5, 15, ..., 85 degrees

function). Performing an optimization of the area under the receiver operating characteristic (AUROC) curve by tenfold validation of the resulting kNN, varying k from 1 to 30, we found the optimal AUROC $= 0.71$ at $k = 24$ [1]. This policy can be visualized for (x, y, θ) as a series of xy slices for varying θ, as shown in Figure 10. These nine slices show the decision boundaries as the θ of the experiment data varies at 10^o steps between 5^o and 85^o of the nut's angle with respect to the coordinate frame of the thumper. Red dots are successful impulses and black dots are failed impulses in the training data set. Red areas indicate positions where activating the thumper if the nut has this orientation is predicted to be successful, and blue areas indicate positions where it is predicted to be unsuccessful. At first glance, it is alarming to see how the decision boundaries of kNN are oddly shaped with the experimental result almost completely detached from the boundary. However, the nature of the kNN is to partition the space in ways similar to a Voronoi diagram, so even small changes in the near field data can make large changes in the far field.

We also tested RF (Random Forest) and SVM (Support Vector Machine) classifiers, as shown in Figures 11 and 12. Although the RF and SVM decision boundaries look more intuitively correct, we found that in actual use neither performed quite as well as the kNN, as will be discussed below. The complete control policy can now be stated: the core kNN control policy is evaluated six times, once for each of the six 60-degree symmetry rotations of the nut position. The highest scoring kNN output of the six rotations then determines choice of solenoid. Duration of impulse is then

[1]Many implementations of kNN classifiers restrict k to be odd, thus avoiding ties; we allowed an even k as it produced a higher AUROC and resolved ties by random selection between the two best choices.

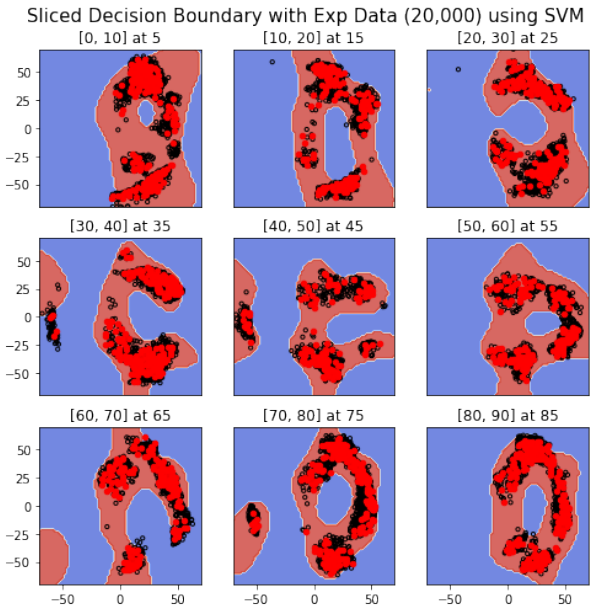

Fig. 12: Flat nut Support Vector Machine Classifier decision boundaries sliced at 5, 15, ..., 85 degrees

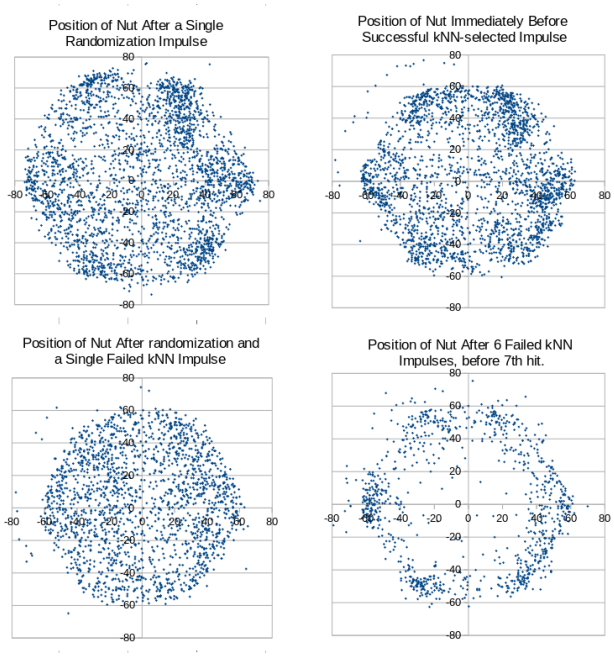

Fig. 13: (x, y) positions of the flat nut after a successful standing trial, immediate position before a successful trial, position after a single impulse after randomization, and position after six failed impulses. Note that multiple impulses seem to form a distinct attractor at the outer rim of the hexagonal corral.

chosen by selection of what duration of the three (17, 20, and 23 milliseconds) for that solenoid's k-neighborhood yielded the highest number of successful standing nuts.

If the impulse succeeds, the nut is knocked back down by applying repeating random impulses until the nut falls, then the entire procedure repeats until a total of 20,000 impulses have been applied.

VIII. Model Testing and Results

With our control policy fully defined, we can now test the actual performance of the self-supervised kNN policy versus the benchmark pure random policy. By executing a set of 20,000 impulses, we observe the scatter plots of success and failure shown in Figure 13.

We note several interesting phenomena here. First, there appears to be vague hot-spots for success; as expected, this hot-spot pattern has six-fold rotational symmetry. Secondly, the position distribution after a first kNN-selected pulse that did not stand the nut up appears to be visually more uniform than after a single randomization pulse.

Finally, comparing the benchmark pure random policy (as implemented by all known commercial offerings versus the kNN policy, we see the individual trial success rates shown in Figure 14 and the cumulative success rates in Figure 15.

The kNN per-impulse success rate exhibits a declining pattern, with the first impulse having a 0.211 success probability, the second impulse having an 0.182 probability, and so on. One possible explanation for this phenomenon is the evolving distribution of the position of the nut after repeated failed attempts. Figure 13 shows that the position of the nut before the first attempt (upper-left plot) is quite uniform and very different from the position of the nut after six failed attempts (lower-right plot); a clear grouping of the position

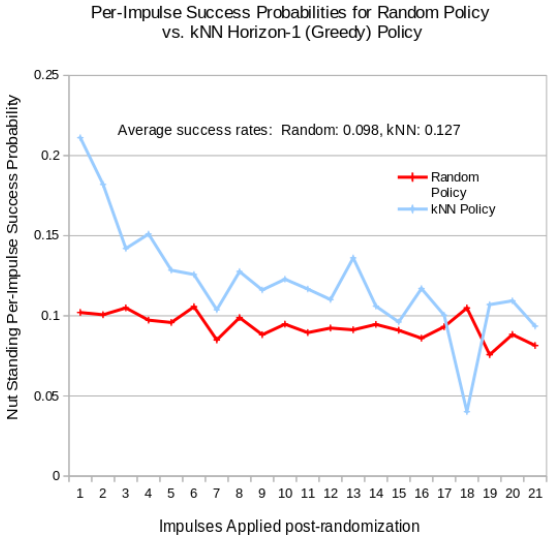

Fig. 14: Per-impulse success rates for the benchmark random policy and the self-supervised kNN policy.

around the edges of the bowl is visible, from where standing the nut is presumably more difficult.

Averaged over the first 20 impulses following a successful knockdown from a standing pose (covering 93% of the total 20,000 impulses) the kNN policy has a 0.127 success rate,

and a per-impulse weighted average success rate of 0.154 [2]. Corresponding values for the benchmark random policy are a fairly constant 0.098 probability and a per-impulse weighted average of 0.096.

Fig. 15: Cumulative success rates of the benchmark random policy and the self-supervised kNN policy

The cumulative success rates show the effectiveness of the kNN control policy. The kNN policy has the nut standing >50% of the time in about 3.6 impulses, whereas the random policy (corresponding with state-of-the-art commercial impulse and vibratory systems) requires an average of roughly 6.8 impulses to reach the 50% threshold. In terms of *takt* time of an assembly process using the manipulated parts, this implies the kNN policy would be about twice as fast as the benchmark random policy.

We also compared the kNN results with the Random Forest and SVM classifiers; we noted that although the RF and SVM classifiers outperformed the random (control baseline) policy, neither performed as well as the kNN policy (typically worse by 6-7%). Another advantage of the kNN is that it requires no significant post-processing computation and can be incrementally built.

IX. Conclusions and Further Work

These experiments show that learned probabilistic models can be useful to learn control policies for systems that are too complex or intractable to model physically from first principles, including systems where the process being controlled is a completely black box. Additionally, the black box may contain state components whose very existence is completely hidden from the controller policy (as exemplified here by the spin s eightfold position ambiguity of the flat nut). Furthermore, policy generation can be automated, and the collection of sample data can be performed in a self-supervised manner, given a machine vision system of moderate capability, thus requiring a minimum of human attention and programming skill.

Unexpectedly, the learned policy effectiveness declines after repeated failures, whereas no such effect is observed for the random policy. A more advanced learning controller might be able to maximize the success rate over a longer horizons by considering not only the expected success rate of the next impulse, but also what position would the part end up with in case of failure. A model predictive control (MPC) scheme with a longer horizon might be able to accomplish this, if a predictive model for the successor position of the part can be estimated reliably.

Other future extensions of this work that we are considering include controlling multiple objects in the bowl simultaneously, firing multiple solenoids simultaneously or with inter-firing delays on the order of the flexure propagation time of the bottom surface, the testing of other shapes, and the integration of the Thumper system with an industrial robot to verify the effectiveness of the part manipulation system end-to-end.

[2]Here the success rate is defined as the average of the success rates for the first impulse, the second impulse, and onward up to the 20th impulse; the weighted average is weighted by the number of times that an N^{th} impulse was applied.

References

[1] M. T. Sgriccia, "Feeder bowl, US Patent 2654465," 1950.

[2] S. Chung and N. Pollard, "Predictable behavior during contact simulation: a comparison of selected physics engines," *Computer Animation and Virtual Worlds*, vol. 27, no. 3-4, pp. 262–270, 2016.

[3] E. Fix and J. L. Hodges, "Discriminatory analysis. nonparametric discrimination: Consistency properties," in *USAF School of Aviation Medicine, Randolph Field, Texas.*, 1951. [Online]. Available: https://apps.dtic.mil/dtic/tr/fulltext/u2/a800276.pdf

[4] T. M. Cover and P. E. Hart, "Nearest neighbor pattern classification," in *IEEE Transactions on Information Theory. 13 (1): 21–27*, 1967. [Online]. Available: http://ssg.mit.edu/cal/abs/-2000_spring/np_dens/classification/cover67.pdf

[5] C. Kong, W. S. Yerazunis, and D. Nikovski, "Learning object manipulation with under-actuated impulse generator arrays," in *American Control Conference (ACC)*, May 2023. [Online]. Available: https://www.merl.com/publications/TR2023-053

[6] A. D. Christiansen, M. T. Mason, and T. M. Mitchell, "Learning reliable manipulation strategies without initial physical models," *Robotics and Autonomous Systems*, vol. 8, no. 1-2, pp. 7–18, 1991.

[7] K. S. Narendra and M. A. L. Thathachar, "Learning automata - a survey," *IEEE Transactions on systems, man, and cybernetics*, no. 4, pp. 323–334, 1974.

[8] L. Ljung, *System identification: Theory for the user*. Pearson, 1997.

[9] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Transactions on neural networks*, vol. 1, no. 1, pp. 4–27, 1990.

[10] M. I. Jordan and D. E. Rumelhart, "Forward models: Supervised learning with a distal teacher," *Cognitive science*, vol. 16, no. 3, pp. 307–354, 1992.

[11] K. S. Narendra and K. Parthasarathy, "Neural networks and dynamical systems," *International Journal of Approximate Reasoning*, vol. 6, no. 2, pp. 109–131, 1992.

[12] T. M. Moerland, J. Broekens, and C. M. Jonker, "Model-based reinforcement learning: A survey," *arXiv preprint arXiv:2006.16712*, 2020.

[13] S. Pfrommer, M. Halm, and M. Posa, "Contactnets: Learning discontinuous contact dynamics with smooth, implicit representations," *arXiv preprint arXiv:2009.11193*, 2020.

[14] M. Parmar, M. Halm, and M. Posa, "Fundamental challenges in deep learning for stiff contact dynamics," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 5181–5188.