Vision-Based Proprioceptive Sensing for Soft Robotic Fingers

Richard Li*
MIT
rli14@mit.edu

Annan Zhang*
MIT
annanz@mit.edu

Abstract

Handed shearing auxetics (HSAs) are novel soft actuators that promise compact, fast, and energy-efficient soft robotic manipulators. While the hardware exists for HSA-based soft robotic grippers, we need state information if we hope to write controllers using these grippers for manipulating objects. Unlike a rigid robotic finger, where a simple model of rigid body links allows us to accurately describe the internal state of a finger, and via forward kinematics, the tip position, it is difficult to model the internal state and tip position of a soft robot finger. In this project, we instead opt to install cameras inside the tip of the finger, pointing towards the base. This camera provides a relatively complete visual view of the internal configuration of the finger, and bypasses the need to define an analytical internal state. Using these internal fingertip camera readings, we estimate the tip position relative to the finger base. We propose and evaluate several methods for fingertip position estimation of an HSA finger in terms of their accuracy and computational speed.

1. Introduction

Soft robotics is a relatively new branch of robotics that deals with mechanically compliant robots that deform in response to forces [9]. The compliance in soft robotic systems comes from the use of soft materials for the robot body, as well as compliant actuators. In comparison to rigid systems, compliance gives us safety and robustness [8]. When using a rigid robotic arm to grasp a fragile item, or having non-compliant system that interacts regularly with humans, the slightest inaccuracies in sensor signals or actuation commands may lead to the item breaking or harm caused to humans. In other scenarios, due to absence of compliance, the robot itself may be damaged. Soft robots have the ability to adapt to constantly changing environments, which makes them viable for search and rescue applications. Recently developed snake-like soft robots can crawl and navigate through various terrains by locomotion through growth [4]. Furthermore, soft robots demonstrate extraordinary performance on complicated tasks involving uncertainty, like picking up objects of unknown sizes and shapes. Soft vacuum grippers can cling to the surfaces of the grasped objects by pumping out the air in the gripper, without specifying the exact displacement or forces of the gripper [5].

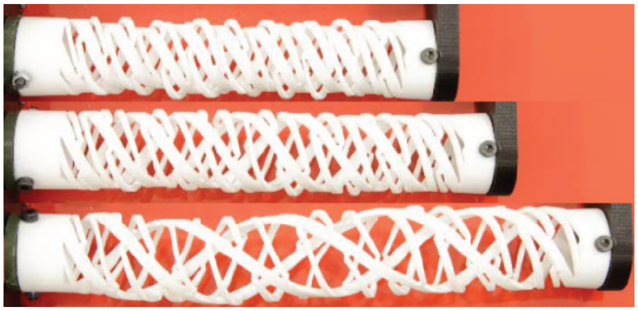

Figure 1. HSAs are polymer tubes with a unique pattern that makes them extend when twisted at the end [6].

Recently, a novel type of electrically-driven soft actuators called handed shearing auxetics (HSAs) have emerged [6]. HSAs are tubes made out of polymers with a unique pattern on their walls, which couples a twisting motion on their ends with an extension in the axial direction 1. Adding additional material along a diagonal in the cylinder wall constraints the HSA on one side and causes it to bend. To build a soft robotic finger, two constraint HSAs with mirrored patterns are rigidly joined together at their tops and their bottoms are rotated in opposite directions 2. Grippers built using two of those HSA fingers have outperformed conventional soft grippers in terms of energy efficiency, speed, strength, compactness, and puncture resistance [1]. Furthermore, HSA grippers are driven by commercially available servo motors, while soft grippers constructed with fluid-driven actuators require large pistons and heavy hardware 3. A robotic platform with 4 degrees of freedom, which can be used as base for a robotic wrist, is built by combining four unconstrained HSAs [6] 4.

To use the finger for manipulation, we would like to understand what the position of the finger tip is with respect to the finger base. Deriving this relationship between the

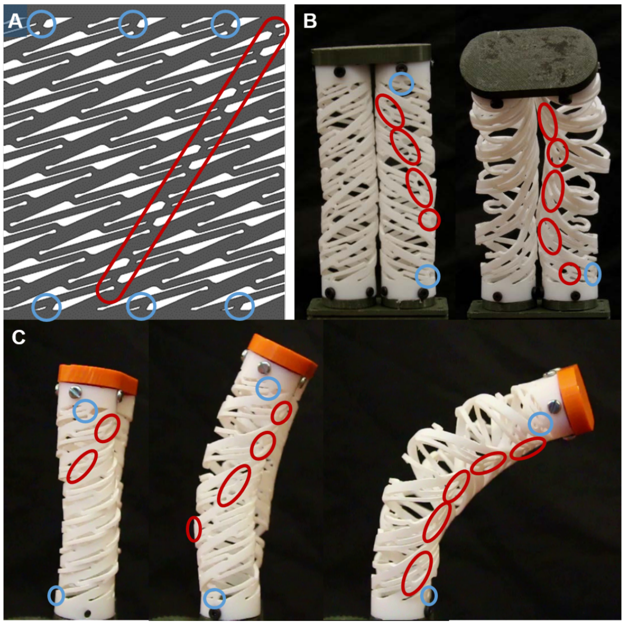

Figure 2. Adding additional material on the pattern (highlighted in red) constraints the HSA on one side and makes them bend instead of extend when twisted [3].

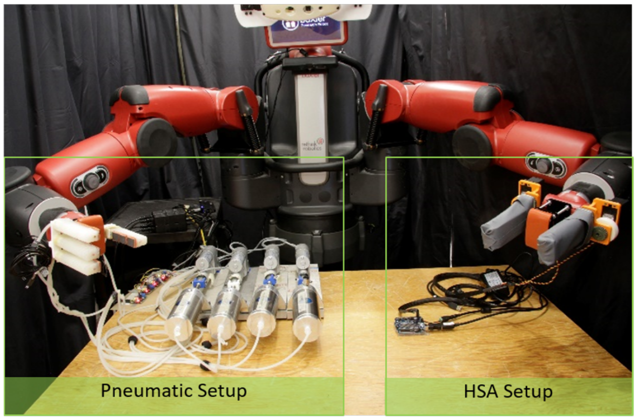

Figure 3. Conventional soft pneumatic grippers driven by fluidics require bulky hardware involving pistons and pumps, while soft HSA grippers only need the power supply for the servo [1].

finger tip and base analytically requires an analytical model of the internal state of the finger. The mechanical behavior of the internals of an HSA is inherently difficult to model, since they posses a high number of degrees of freedom. Furthermore, they exhibit highly nonlinear material behavior both on the constitutive level in terms of the stress-strain-response and on the structural level in terms of buckling. Due to the viscoelasticity of the polymers, HSAs mechanically respond to time-dependent effects, like strain rate, hysteresis, stress relaxation, and creep [11]. Thus, we currently do not have good analytical models for the internal state of an HSA.

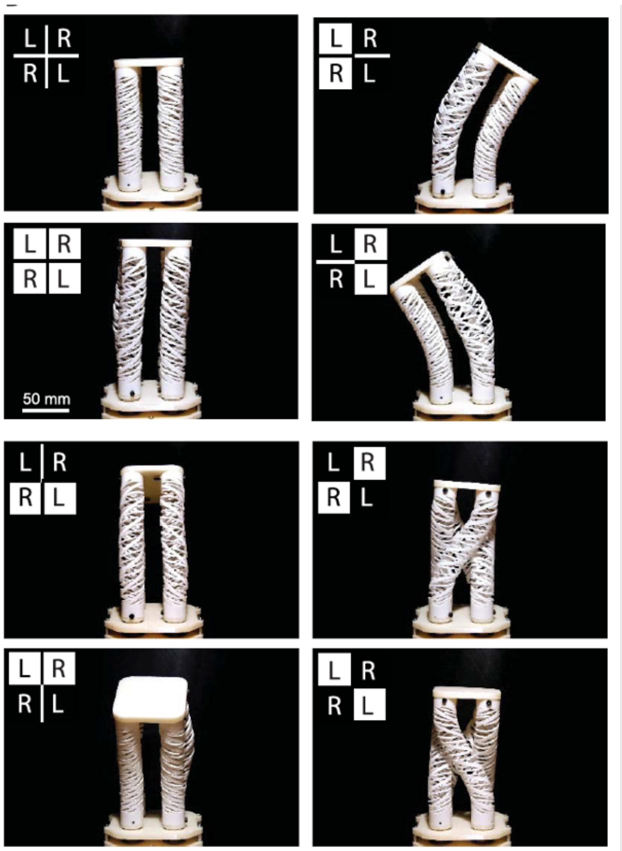

Figure 4. The four degrees-of-freedom robotic platform is made out of four unconstrained HSAs and can be used as base for a robotic wrist [6].

Additionally, it is difficult to add internal sensors throughout HSAs. The difficulty sensorizing HSAs lies in the fact that all of cylindrical wall moves. Existing efforts include fixating a strain gauge on the outer radius of the bending [2], as well as laminating a conductive material onto the outer wall of the HSA. The former method mechanically interferes with the actuation of the HSA and only delivers one dimensional strain data. The latter method, in essence a piezoelectric sensor, fails because of variability in adhesion and significant drift. In this work, we propose using a camera that looks at the inside of an HSA as sensor. We leverage the high dimensional camera data to get rich sensor readings, which gives us the flexibility to learn models that predict different information about the HSA state. Furthermore, having a sensor that does not mechanically interfere with the HSA enables virtual data collection by simulating HSAs using finite element methods (FEM). This is not possible for interfering sensors, since modeling contact with the HSAs makes the FEM simulation infeasible.

In this project we are concerned with the estimation of the tip position of an aforementioned soft robotic finger, a pair of HSA cylinders joined together at their tips. Knowing the current tip position of the finger will allow us to write controllers for accomplishing downstream tasks, such as grasping objects. Conventional robot fingers are composed of rigid links with sensors providing joint readings at each link. The tip position of a robotic finger relative to the finger base can be calculated by the matrix product of a sequence of transformation matrices. This calculation is known as forward kinematics.

In contrast, soft robotic fingers do not have a simple internal model relating the finger base pose and finger tip pose. Thus, forward kinematics is not applicable. Instead, we aim to do visual pose estimation for our soft robot finger by mapping the high-dimensional images of the internals of the finger to the tip position using neural networks. These images are taken with Raspberry Pi cameras that are mounted into the top of the HSAs, while the ground truth tip position is recorded by an external motion capture system.

2. Related Work

In related work, Werner 2020 [12] developed a camera sensor to estimate the tip position of a pneumatically-driven soft bellow actuator. For their inference pipeline to run in real time, they preprocessed the images with non-learned algorithms and hand-designed six 3×3 features for a SVM that performs kernel regression. Similarly, She 2020 [10] developed exoskeleton-covered soft fingers with built-in cameras, which capture high-resolution images for estimating the finger tip position. They employed a simple, large CNN, and demonstrated its capability to predict the bending angles between the piece-wise straight sections of the exoskeleton-covered fingers.

Like Werner 2020, we desire a light-weight state estimation system that can run in real-time. However, our HSAs are more complicated than the actuators considered by Werner, which may necessitate the usage of more powerful function approximators like neural networks. Compared to She 2020, we aim to design a much faster system which can be applied on mobile robots. Additionally, our HSA soft actuators are not the same as the fingers used in She's work. Because we have holes in our actuators, there is more ambient light affecting our camera images, which may make our task harder.

3. Approach

Our method consists of two main components: designing the hardware system to collect data and implementing the models and algorithms for mapping the image of the inside of the finger to the fingertip position.

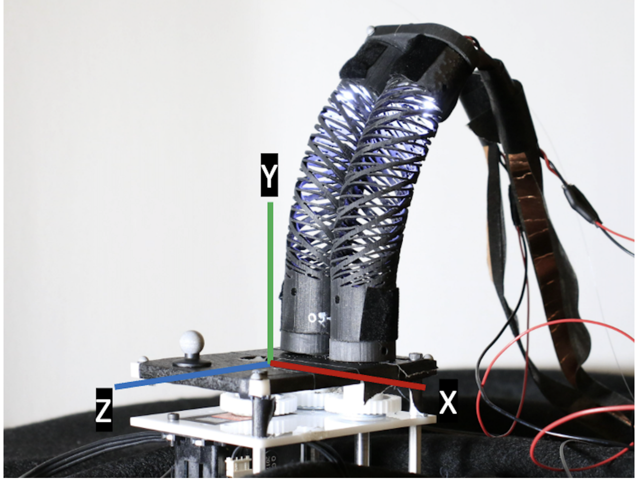

Figure 5. Our setup involves an HSA finger that has LED boards and cameras in the tip. The HSA tubes are connected to rotating adapters, which are driven by counter-rotating gears connected to one servo motor. The coordinate frame of reference is defined to be the center of the base platform.

3.1. Hardware

3.1.1 Actuators

Our actuator consists of two 3D-printed HSA cylinders made out of polyurethanes, which have a length of 101.6 mm and an outer diameter of 25.6 mm. Each of the HSAs is printed with an additional constraint in its structure, which causes it to bend, instead of extend, when twisted. The bottoms of the HSAs are mounted onto counter-rotating sockets, which are both actuated by the same ROBOTIS Dynamixel MX-28 servo motor. At the top, the HSAs are rigidly joined together so the cylinders provide a reaction torque for each other when the sockets are rotated 5.

3.1.2 Camera Setup

A customary fixed-focus Raspberry Pi camera (Arducam IMX219) is attached to the top of each HSA (e.g. at the fingertip), recording its interior with a resolution of 480×360 pixels at 30 frames per second. The cameras are connected to an NVIDIA Jetson Nano board, on which the final neural network will be deployed after the finger is installed onto a mobile robot.

Early prototypes have shown that the HSAs are very permeable to light, making image data recorded at different ambient lighting conditions incomparable. To alleviate this issue, we designed a printed circuit board with six white surface-mount LEDs 6. These LED boards are connected to an external power source and illuminate the inside of an HSA from the tip, providing a robuster light environment for recording reproducible images 7.

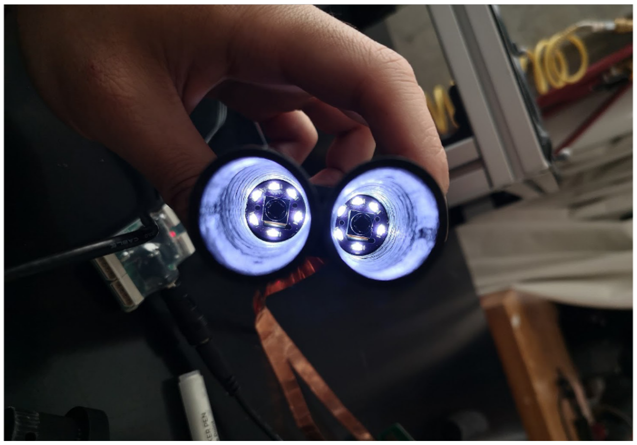

Figure 6. View inside the HSA, where our custom-made LED board, mounted at the tip, provides internal lighting for the cameras.

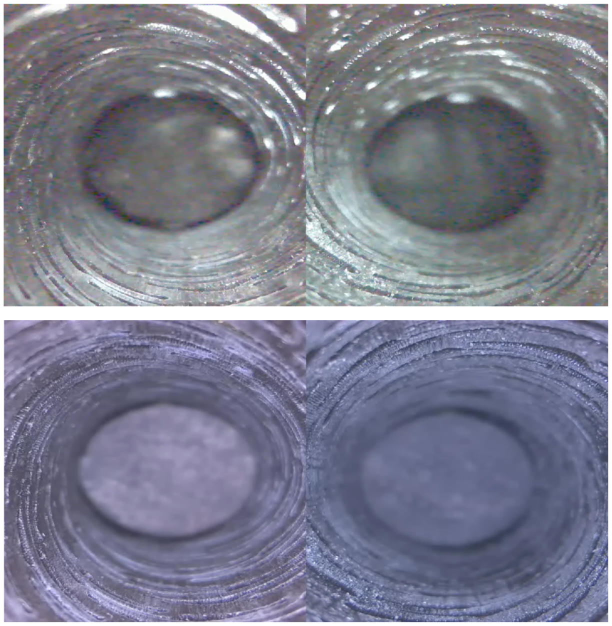

Figure 7. Top: An image recorded by the cameras when the LED lights are turned off is noisy and varies highly with ambient lighting. Bottom: With the LEDs turned on, this is a typical image in our data set of images with non-marked bases.

3.1.3 Motion Capture System and Data Diversity Considerations

Our goal is to learn a tip position estimation system that is highly accurate, such that we can write fine-grained controllers for manipulating objects. In order for our machine learning system to be accurate, we need very accurate ground truth labels of the tip position and orientation. Motion capture (mocap) systems are one way of collecting high accuracy state readings. In our work, we obtain the ground truth state readings from a motion capture system using six OptiTrack Flex 13 cameras connected to an external PC. We attach special bead markers to the fingertip so the mocap system can detect our fingertip. The tip position of the HSA finger is recorded with submillimeter acccuracy at 60 Hz.

One major consideration of our data collection process is data variation. The typical demands of a pose estimation include accurate estimation in the face of sensor noise, variations in background texture, and occlusions from other objects. Because our camera is looking from the top of the fingertip down into the internals of the finger, many of issues that make external pose estimation difficult are alleviated. However, there are still variations in the internal environment of the finger that we attempt to model with our data collection process. While most of the outside world is occluded by the finger casing, there are still holes in the mesh-like HSA material. These holes allow significant amounts of ambient lighting to potentially enter. Therefore, we take care to provide different ambient lighting conditions by recording our trajectories at different times of day and by varying the lights that are turned on or off in the room we use for collecting data.

Our final goal of pose estimation is to provide the necessary input to a controller is tasked with picking up objects. As such, the trajectory of the finger should not be constant from recording to recording. We thus generate varied trajectories of the finger during mocap data collection.

We would hope to design a vision system that can be used from finger to finger without being retrained. Therefore, we collect trajectories with different builds of the finger and with slightly different materials.

Finally, the HSA materials are currently not extremely durable. Over the course of dozens of interactions, the material may considerably degrade. This degradation is likely to be a significant factor influencing the distribution of image readings. We account for this by using the same finger for many takes at a time, thus naturally collecting trajectories over varied levels of mechanical wear.

3.1.4 Physical Base Marking

Prior work found that painted patterns of dots assisted in pose estimation [12]. The basic idea is an easily distinguishable physical feature allows a computer vision system to more easily localize itself. In our system, we cut out a small white circular cardboard piece, and taped it to the base of the finger. This marking manifested itself as a bright white circle in the fingertip camera images. We created two equally sized datasets to test the effect of pose estimation with and without this physical base marking.

3.2. Software

3.2.1 MoCap Data Processing

We perform a series of data processing steps on the fingertip camera images and mocap data. The fingertip camera collects images of the inside of the finger / HSA cylinders, while the mocap system collects state readings of the fingertip position and orientation. The images are considered the input, and the mocap state readings are considered the output; together, (image, state) tuples form the samples for our machine learning system. Our goal is to synchronize these two data sources in time, to form the samples of our machine learning dataset. Since the images and mocap data are collected on different computer platforms (Jetson board and external PC, respectively), we use a large light perturbation to synchronize the data in time. At the beginning of every data recording session, we turn on a very bright flood light, that shines into the HSA finger. We detect when this flash happens in the stream of fingertip camera images by searching for the largest intensity drop between frames, and in the mocap data stream by looking at when the mocap system stopped receiving signals from the flood light.

To fill in single missing data points in the mocap tip position and orientation, we linearly interpolate the tip from existing time steps. We perform a coordinate frame transformation to obtain the tip position in a coordinate frame fixed to the HSA finger base, which makes the data independent of position and orientation of the whole HSA setup inside the mocap booth. Furthermore, we smoothen the mocap data with a 1D Gaussian filter with kernel size 3 to reduce noise and subsample at a factor of 2. The final output of our mocap data collection process is a collection of MP4 videos, formed from the sequence of fingertip camera images, and a corresponding collection of CSVs (one per video), where each row of the CSV contains the tip position and tip orientation at every timestep in the video.

3.2.2 Machine Learning Setup

Besides our hardware setup, we want to design a machine learning system that can 1) learn from a relatively small number of samples, 2) achieve high accuracy on a test set of held out trajectories, and 3) have fast inference time for single samples.

Given the MP4 and CSV files from the mocap data collection step, we must write a dataloader. It is difficult to write a fast dataloader directly pulling frames from the MP4 videos at each training minibatch. Instead, before training starts, we uniformly sample frames at 33.3ms intervals (corresponding to 30FPS), and serialize these frames to disk as individual images. Our dataloader crawls the folder structure of our data folder and loads the file paths of images and the corresponding mocap state readings into a pandas dataframe. Every minibatch, the dataloader loads images from disk into GPU memory, to be passed to the neural network.

Before images are passed into our neural network, we apply a few transforms. One important transform is resizing the 480x720 original images (pulled from the MP4) to a 224x224 downsized image. This was important for significantly reducing the size of the network, and did not result in significant accuracy decrease. Besides resizing, we only had two other simple operations. The first, was a ColorJitter data augmentation from Pytorch, which augmented the brightness, contrast, saturation, and hue of the image. This augmentation was designed to increase generalization performance given our small training set size. Finally, we normalized the image along the x, y, and z channels, such that the resulting distribution of pixels along each channel were unit Gaussian. To do this, we calculated the mean and standard deviation of pixel intensities across all pixels of all images in the training set. All of these transformations were critically important to the final accuracy and speed of the model.

Due to our desire for fast inference, we designed a small, simple architecture, and we called it SimpleCNN. The basic module in our architecture consists of: 2D convolution layer, batchnorm, ReLU activation function, and 2D maxpooling layer. The SimpleCNN architecture consists of this basic module stacked three times depthwise, with two final fully connected (FC) embedding layers, which eventually map to a 3D vector representing the XYZ position output. Symbolically, the architecture can be expressed as: Conv-BN-ReLU-MaxPool-Conv-BN-ReLU-MaxPool-Conv-BN-ReLU-MaxPool-FC-FC.

4. Experimental Results

Our first experiment was using only the SimpleCNN architecture without any data augmentation or other training techniques. Our initial setup of the train/val/test datasets was pooling together the frames from all trajectories, and randomly partitioning the frames according to a 60/20/20 split. In fact, in this setting even this most basic training setup worked well. The real challenge was setting up a train/val/test split based on the trajectories, and not individual frames. In other words, we pool the trajectory IDs into a set, and randomly partition the *trajectories* according a 60/20/20 split. In this setting, the test set contains unseen trajectories. The initial dataset split only tested our network's capability of interpolating amongst frames in trajectories it already saw during training time. In this new dataset, our network is tested on tis ability to generalize to unseen trajectories, which means unseen variations in environmental lighting, tip position, hardware model of the finger, and levels of finger mechanical wear. The setting of generalizing over trajectories is much harder.

Immediately upon changing to this new dataset split over trajectories, we noticed much higher variance in the validation error during training. The validation mean squared error (MSE) would fluctuate from 5mm to 260mm. Even with early stopping based on the validation error, we were not able to achieve good results. The X, Y, Z MSE were 4.641, 8.222, and 6.497 respectively. For the X and Y dimension, the MSE was greater than the range between the minimum and maximum values, meaning the network was not even making predictions within the support region of the data.

The next change we implemented was normalizing the input data. We normalized all the pixels along the X, Y, and Z dimension such that the resulting distribution was unit Gaussian. Despite already having BatchNorm between CNN layers, this input normalization made a significant difference. Our new X, Y, Z MSE were 3.654, 1.104, and 3.511. The fluctuation in the validation error was also greatly decreased. Using input normalization, the maximum validation error was 43mm instead of 260mm.

We noticed at this point that the generalization gap would start increasing as the network overfit to the training data. The training error would quickly converge close to 0, but the validation error would start diverging. We introduce our data augmentation now as a form of regularization to reduce the generalization gap. We only apply data augmentation over the color intensities and brightness values of the image, and not any translational or rotational transforms. While these motion transforms may improve classification performance, they are not appropriate for our pose regression task, where the absolute position of pixels gives us important information about the tip position. We apply this augmentation using the ColorJitter augmentation in Pytorch, which takes in 4 scalar hyperparameters determining the amount of brightness, contrast, saturation, and hue jitter. We use the same hyperparameter across all four types of ColorJitter. We compare performance over two instantiations of this hyperparameter: [0.2, 0.4]. We found that 0.2 performed the best and that 0.4 was too extreme and led to a decrease in test accuracy. Using ColorJitter with a 0.2 hyperparameter, we ended up with a new X, Y, Z MSE of 3.654, 1.104, and 3.511.

We made one final change to significantly improve our test set performance. We took all the previous techniques inclduing normalization and data augmentation, and applied them to training a new model over the dataset collected with the physical base marking. The physical base marking is a white circle that provides an easily localizable feature in the image. This marking greatly improved performance. Our new X, Y, Z MSE became 0.064, 0.258, and 2.227.

In summary, we started off with a MSE error on the test set of 4.641, 8.222, 6.497 and were able to bring these errors down to the sub-millimeter level with new accuracies of 0.064, 0.258, and 2.227, by using a combination of BatchNorm, early stopping based on validation error, input normalization, data augmentation of color and brightness, and using a physical marker for the finger base. The Y and Z error decreased by a factor of 31.868 and and 2.917, respectively.

Additionally, we performed some small experiments regarding inference speed. We quantified the difference in inference speed between the SimpleCNN model and two other models: SimpleCNN5 and SimpleCNNMini. SimpleCNN5 contained two more convolutional layers before the fully-connected layers. It is described symbolically as such: Conv-BN-ReLU-MaxPool-Conv-BN-ReLU-MaxPool-Conv-BN-ReLU-MaxPool-Conv-BN-ReLU-MaxPool-Conv-BN-ReLU-MaxPool-FC-FC. SimpleCNNMini replaces one of the fully connected layers (which had many more weights than a convolutional layer) with two convolutional layers: Conv-BN-ReLU-MaxPool-Conv-BN-ReLU-MaxPool-Conv-BN-ReLU-MaxPool-Conv-BN-ReLU-MaxPool-Conv-BN-ReLU-MaxPool-FC. Compared to these architectures, SimpleCNN demonstrated a 34 percent and 32 percent decrease in inference time of a single image. The final speed for the forward pass for a single image was 0.47ms, which means the network forward pass should not be the bottleneck in a system implementing real time control.

5. Conclusion

We show that our pose estimation system is able to accuratly predict the fingertip position, with sub-millimeter accuracy, for completely unseen trajectories. These trajectories were taken in different environmental conditions, and require generalization over lighting, finger hardware, and motion. These results are promising and indicate that we may be able to design fine-grained controllers for picking up objects using the detected tip poses. For example, depending on the shape and scale of an object, we may modulate how far the fingers open or close, with extremely high accuracy. With respect to our current work, there are still some limitations. Although the forward pass of our network is extremely fast, there may be other computational bottlenecks between the image being produced by the camera and the image going into the neural network. For example, downsizing large images may be a relatively slow operation in practice. Secondly, although, we did our best to provide varied lighting and environmental conditions, our data was still largely collected in one of the laboratory rooms in the MIT CSAIL building. It may be difficult for a network trained on our data to generalize to open world environments or environments with very bright sunlight.

We desired a very simple state estimation in our current system, the XYZ position of the fingertip position. There may be other representations that produce better accuracy, or provide more information to a downstream controller. For example, instead of predicting the fingertip position, we may predict various keypoints along the finger, or predict the parameters of a spline. Finally, while we collected mocap readings of the orientation, we did not have enough compute or time to run training experiments on the orientation. Rotation is often the more difficult aspect of pose estimation. Difficulties related to discontinuous rotation representations and object symmetry may affect HSA fingertip rotation estimation [13] [7].

Furthermore, because of the difficulty of manually writing manipulation controllers, an interesting direction may be simultaneously learning the latent visual representation and policy, using imitation learning or reinforcement learning. For example, we could develop a teleoperation system representing two HSA fingers, and allow a human operator to teleop the robot to pickup various objects. An end-to-end pipeline may be able to bypass the need for accurate state estimation entirely.

Appendix

Fabrication of HSAs

Up until recently, fabricating an HSA involved cutting out the HSA pattern from off-the-shelf polymer tubes on a rotary laser machine. Reliance on customary products, combined with the mechanical requirement to have a material with high elongation at break, restricted the material choice practically to extruded teflon (polytetrafluoroethylene, PTFE) tubes. Inconsistencies in quality from off-the-shelf material, as well as inaccuracies in the laser cutting equipment, limited design flexibility. In recent work, 3D printing HSAs via digital projection lithography enables more flexible design, a greater range of materials, and a generally wider adoption of HSAs [11]. State-of-the-art HSAs are printed out of Flexible Polyurethane 50 (FPU 50), a proprietary polymer resin mixture by Carbon Inc.

Individual Contributions

Richard: I helped with the overall design of the project during the planning phase as we decided how to frame the HSA state estimation problem as a project we could tackle in this class. After Annan setup the physical data collection process, and collected videos of camera images and state readings, I was responsible for writing the corresponding ML infrastructure to process and train the images. I wrote the dataloader, training scripts, models, and ran experiments over various hyperparameters and algorithms to achieve good performance over the test set of unseen trajectories. I also contributed significantly all written and presentation portions of the project.

Annan Zhang: I was responsible for the whole hardware setup (HSA fingers, LED board, camera, base marking, motion capture system). I was responsible for recording the camera and motion capture data, as well as synchronizing, postprocessing, and setting it up for the learning part. I helped designing some experiments and evaluation. I was responsible for the respective parts in the project presentation and report.

References

[1] Lillian Chin, Jeffrey Lipton, Robert MacCurdy, John Romanishin, Chetan Sharma, and Daniela Rus. Compliant electric actuators based on handed shearing auxetics. In *2018 IEEE International Conference on Soft Robotics (RoboSoft)*, pages 100–107. IEEE, 2018. 1, 2

[2] Lillian Chin, Michelle C Yuen, Jeffrey Lipton, Luis H Trueba, Rebecca Kramer-Bottiglio, and Daniela Rus. A simple electric soft robotic gripper with high-deformation haptic feedback. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 2765–2771. IEEE, 2019. 2

[3] Lillian Tiffany Chin. *A high-deformation electric soft robotic gripper via handed shearing auxetics*. PhD thesis, Massachusetts Institute of Technology, 2019. 2

[4] Elliot W Hawkes, Laura H Blumenschein, Joseph D Greer, and Allison M Okamura. A soft robot that navigates its environment through growth. *Science Robotics*, 2(8), 2017. 1

[5] Shuguang Li, John J Stampfli, Helen J Xu, Elian Malkin, Evelin Villegas Diaz, Daniela Rus, and Robert J Wood. A vacuum-driven origami “magic-ball” soft gripper. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 7401–7408. IEEE, 2019. 1

[6] Jeffrey Ian Lipton, Robert MacCurdy, Zachary Manchester, Lillian Chin, Daniel Cellucci, and Daniela Rus. Handedness in shearing auxetics creates rigid and compliant structures. *Science*, 360(6389):632–635, 2018. 1, 2

[7] Giorgia Pitteri, Michaël Ramamonjisoa, Slobodan Ilic, and Vincent Lepetit. On object symmetries and 6d pose estimation from images. In *2019 International Conference on 3D Vision (3DV)*, pages 614–622. IEEE, 2019. 7

[8] Panagiotis Polygerinos, Nikolaus Correll, Stephen A Morin, Bobak Mosadegh, Cagdas D Onal, Kirstin Petersen, Matteo Cianchetti, Michael T Tolley, and Robert F Shepherd. Soft robotics: Review of fluid-driven intrinsically soft devices; manufacturing, sensing, control, and applications in human-robot interaction. *Advanced Engineering Materials*, 19(12):1700016, 2017. 1

[9] Daniela Rus and Michael T Tolley. Design, fabrication and control of soft robots. *Nature*, 521(7553):467–475, 2015. 1

[10] Yu She, Sandra Q Liu, Peiyu Yu, and Edward Adelson. Exoskeleton-covered soft finger with vision-based proprioception and tactile sensing. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10075–10081. IEEE, 2020. 3

[11] Ryan L Truby, Lillian Chin, and Daniela Rus. A recipe for electrically-driven soft robots via 3d printed handed shearing auxetics. *IEEE Robotics and Automation Letters*, 6(2):795–802, 2021. 2, 7

[12] Peter Werner, Matthias Hofer, Carmelo Sferrazza, and Raffaello D'Andrea. Vision-based proprioceptive sensing for soft inflatable actuators. *arXiv preprint arXiv:1909.09096*, 2019. 3, 4

[13] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5745–5753, 2019. 7