Teleoperation of the Allegro Hand via Kinematic Hand Synergies in Drake

A. Michael West Jr., Raul Largaespada, Kaymie Shiozawa

***Abstract*—This paper presents the design and implementation of a teleoperation scheme for a multi-finger robotic hand in the Drake simulation environment using human demonstrations captured by a single RGB camera. Specifically, teleoperation of the Allegro Hand was implemented using a custom script to obtain joint kinematics of the human hand via a software called MediaPipe. Moreover, through the addition of kinematic hand synergies in this pipeline, we were able to simplify the degrees of freedom used to control the robotic hand. Currently, full teleoperation is incomplete as we were unsuccessful in grasping and manipulating different objects. In the future, we plan to not only address this limitation but to also apply this teleoperation pipeline to conduct behavior cloning of various manipulation tasks based on human demonstration. Such research is important for the design and control of anthropomorphic prosthetic hands.**

***Index Terms*—Teleoperation, Robotic Manipulation, Dexterous Manipulation, Synergies.**

I. Introduction and Related Work

THANKS to the dexterity of our hands, humans have a unique ability to manipulate objects and tools to navigate the world around us. In robotics, multi-finger dexterous manipulation has been an important, yet challenging topic. Because of the many degrees of freedom (approximately 20) to control, dynamics and contact modeling are significantly more complex for a multi-finger robotic hand than for a traditional two-finger gripper.

Despite these challenges, successfully achieving robust multi-finger dexterous manipulation in a variety of scenarios could significantly improve prosthetic design. A 2008 study showed that approximately 700,000 Americans [1] have undergone amputation of their hand(s). With an expected increase of diabetic patients, this number is predicted to double by 2050 [1]. Furthermore, amputees have been reported to prefer the use of anthropomorphic prosthetic hands [2]. In a world designed for dexterous humans, an effective, anthropomorphic prosthetic device can markedly improve an amputee's quality of life [3].

One proposed solution to multi-finger dexterous manipulation control is to learn from human demonstration [4]. A teleoperation pipeline can be designed to collect this human-based data; the design of such a pipeline is the topic of this paper. Many of the state of the art hand-tracking and robotic teleoperation systems require full motion capture set ups [5], [6] or specialized gloves [7]–[10]. Han et al. showed impressive real-time hand tracking using motion capture, but did not extend their study to conduct teleoperation [5]. Another study tested subjects' ability to teleoperate a robotic arm via an optical motion capture system [6], but in general, motion capture setups are expensive and can take up a lot of physical space. Alternatively, specialized gloves like the CyberGlove [7], reduce the physical footprint and are some of the most accurate solutions to hand-tracking because resistive bend-sensors can be attached to the hand directly. However, these systems are still costly, and wearing the glove may shift the behavior of the human subject from their normal manipulation patterns. Furthermore, an off-the-shelf hand-tracking glove does not fit all hands equally, and this misfit may reduce its accuracy when collecting data from a variety of subjects.

Addressing the aforementioned limitations and trade-offs, vision-based hand-tracking has recently emerged as a promising method to inexpensively conduct human teleoperation experiments. RGB cameras are ubiquitous today and can be obtained at a low cost. For example, Ge et al. proposed a deep neural networks-based method to capturing hand kinematic data and showed outstanding results using a depth camera [11]. Although depth cameras may not be as expensive as motion capture systems or specialized gloves, they are still less prevalent than RGB cameras. Meanwhile, MediaPipe Hands [12] is a high-fidelity hand tracker that uses machine learning to capture 21 points of interest on the human hand. Their approach is effective, as it can achieve real-time tracking with a mobile phone (as opposed to hefty desktop machines or cloud clusters). Additionally, MediaPipe Hands is based on a dataset of hands from all over the world, various age groups, and in different lighting and background conditions, making it suitable to most use cases. Finally, self-occlusions and partially visible hands were a specific topic of interest for their team, so MediaPipe Hands is relatively more robust to these common pain points of vision-based hand tracking.

Applying vision-based tracking to robotic teleoperation, Handa et al. successfully implemented an impressive teleoperation pipeline that succeeded in fine motor tasks such as removing a bill from a wallet, albeit requiring a depth camera [13]. Sivakumar et al. instead focused on enabling robots to learn in the wild and emphasized the importance of creating a low-cost solution that works in a variety of environments and with inexperienced operators [14]. They were able to successfully teleoperate using a single uncaliberated camera. This paper has a similar approach, as we apply hand-tracking

M. West and K. Shiozawa are with the Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, R. Largaespada is with the Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, 02139 USA. E-mails: amwestjr@mit.edu, kaymies@mit.edu, rlargaes@mit.edu

Manuscript received December 12, 2022.

with an uncalibrated RGB camera through MediaPipe Hands to robotic teleoperation.

The ultimate goal of the multi-finger robotic teleoperation pipeline proposed in this paper is to apply it to the learning challenges of dexterous robotic manipulation. However, a major challenge in the manipulation of multi-finger robotic hands is the "curse of dimensionality" [15]. A human hand has more than 20 degrees of freedom. The individual coordination of each of those degrees of freedom makes the learning of the control inputs for robotic manipulation complex. Drawing upon the features of human biomechanics and sensory-motor control may simplify this problem. By using synergies, a coordinate-based dimensionality reduction, we can reduce the degrees of freedom in the human hand. Santello et al. first studied kinematic synergies of the human hand; they found that two synergies accounted for more than 80% of the variance in hand posture during grasping of a set of 57 imagined objects [16]. With this in mind, we postulate that knowledge of human kinematic hand synergies can reduce the learning requirements of dexterous hand manipulation. Therefore, incorporating previously studied hand synergies in our teleoperation pipeline will likely decrease the dimensionality of the learning problem, in turn reducing the complexity of the controller that needs to be learned.

With the future goal of enabling learning of the control inputs necessary for robotic manipulation, we hypothesize that a multi-finger robotic teleoperation system can be designed with reduced degrees of freedom based on hand synergies. To achieve this, we first aimed to create a simulation system in Drake with the LBR iiwa robot, the Allegro Hand, and some representative objects to manipulate. We then set out to integrate the MediaPipe Hands architecture to control the Allegro hand and simplify this control using previously proposed hand synergies. The final performance goal of this project was to grasp and manipulate the objects in the environment with the above pipeline. Key results include:

- The Allegro Hand in the simulation environment was able to be teleoperated via vision-based hand tracking, as it could follow different hand poses that the operator made with sufficient accuracy.
- Hand synergies were successfully applied to the teleoperation of the Allegro Hand, and multiple robot joints were shown to move simultaneously when the operator controlled one joint.
- The contact dynamics in the simulation environment proved to be challenging, and grasping objects via teleoperation was unsuccessful.

II. Methods

A. Implementation of Teleoperation

1) Obtaining Hand Kinematics: One of the goals of this project was to set up teleoperation of a simulated robotic hand in a virtual environment. The first step in doing so was obtaining a signal of human hand motion. This presented many immediate challenges that are briefly discussed in Section IV. We eventually settled on using MediaPipe to detect key points on the hand. The MediaPipe pipeline consists of two models: a palm detector, which provides a bounding box of a hand, and a hand landmark model, which uses the bounding box from the palm detector to predict the hand skeleton [12]. Specifically, the hand landmark model measures 21 keypoints of the hand: four at each finger (three at each joint and one at the tip) and one at the bottom of the palm of the hand. This is shown in Figure 1.

Having obtained the Cartesian coordinates of the hand, we needed to convert them to the relative joint angles of the four fingers as inputs to the controller. Thus, with the help of a labmate, we created a custom script to do so. Specifically, we defined the plane connecting the key points at the base of the palm, the middle finger metacarpal (MCP) joint, and the index finger MCP joint as our global xyz-coordinates. The vector connecting the base of the palm and the middle finger MCP joint served as the +y-axis, and the vector connecting the middle finger metacarpal MCP joint and the index finger MCP joint served as the +x-axis. Furthermore, we assumed that each forefinger has four rotational DOFs (degrees of freedom): flexion/extension at the distal interphalangeal (DIP), proximal interphalangeal (PIP), and MCP joints, and abduction/adduction (ABD) at the MCP joints. Additionally, the thumb had two rotational DOFs at the carpometacarpal joint and a single DOF each at the MCP and interphalangeal (IP) joints. Given the defined coordinate system and these assumptions about joint motion, the 21 hand key points were mapped to the 20 relative joint angles listed above.

2) Simulation Environment: Before implementing the teleoperation, the simulated environment was set up in pyDrake. Specifically, we used the iiwa robot with an Allegro Hand as its end effector. Those can be pictured in Figure 2. The iiwa has seven rotational DOFs while the Allegro Hand has 16 DOFs across four actuated fingers.

3) Contact Simulation of Objects in Scene: The simulation environment was implemented to allow for objects of various shapes and sizes, including spheres, boxes, and mustard bottles, to be placed at arbitrary positions in the world. To simulate contact dynamics, each object has a collision box

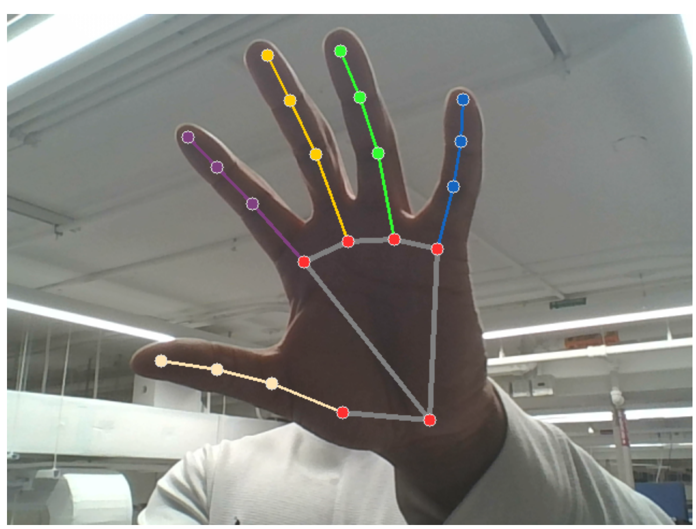

Fig. 1: Example of MediaPipe detecting 21 hand keypoints.

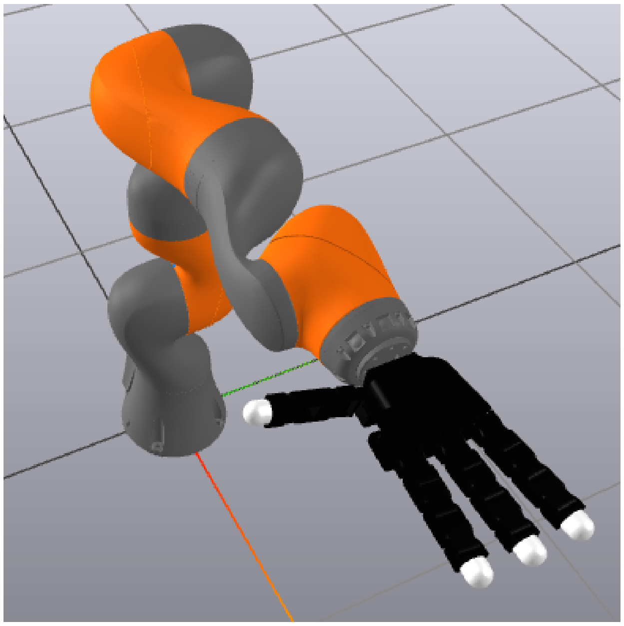

Fig. 2: 7-DOF iiwa Robot with 16-DOF Allegro Hand as its end effector in the Drake simulation environment.

geometry occupying a volume slightly smaller than the visual geometry defined for the object. Additionally, for box-shaped objects, small collision spheres were placed on each corner to supplement existing collision geometry and increase accuracy when in contact with a flat plane.

These contact geometries, in conjunction with the contact geometry on the surface and tips of the Allegro Hand, were sufficient to simulate contact forces when grasping and manipulating objects.

4) Robot Inverse Dynamics Controller: With the measured joint angles of the joints of the hand and the simulation environment setup, a controller for the robot hand and arm were designed. To do so, we used Drake's embedded `InverseDynamicsController` function. This controller takes the estimated and desired states as inputs, and outputs control torques at each individual DOF. Mathematically, the controller solves the following equations:

$$\tau = inverseDynamics(q, \dot{q}, \dot{q}_{command})$$
$$\dot{q}_{command} = kp(q_d - q) + kd(\dot{q}_d - \dot{q}) + ki \int (q_d - q) + \ddot{q}_d. \quad (1)$$

where, the subscript d denotes the desired values. In our simulation we used kp, kd, and ki gains of 20, 5, and 1 respectively. With this control formulation, we inputted the desired positions and velocities for the iiwa and the Allegro Hand separately.

We commanded the pose of the iiwa or the position of the Allegro Hand end-effector using meshcat sliders [1]. Then, we used Drake's `InverseKinematics` function to solve for the joint kinematics that satisfy the position and orientation constraints defined by the desired end-effector position and orientation. These joint kinematics could be used as the input to the inverse dynamics controller.

To command the Allegro Hand, we mapped the hand joint positions from MediaPipe and our custom code to the joint positions of the Allegro Hand. However, because the Allegro Hand's kinematics are not exactly anthropomorphic and due to limitations on the accuracy of the joint angle tracking from MediaPipe, this task required some filtering and adjustment to the joint angle inputs. The mapping from joint angles to Allegro command angles is shown in Table I. Reordering the joint angles using the table produced our position command input to the Allegro Hand. Since the Allegro hand only has four fingers, angles from the pinky were ignored.

[1]Ideally, we would like to have been able to map direct hand translations and motions to the robot end-effector. However, without a depth camera this was infeasible.

B. Implementation of Kinematic Hand Synergies

The second goal of this project was to test the use of synergies as a subspace to control the 16-DOF Allegro Hand. In simulation we implemented synergy reconstruction under five control conditions. The conditions specified the number of DOFs the Allegro Hand had by projecting the human motion into a synergistic subspace. The DOFs that we implemented included: All Allegro DOF, 1 synergy, 2 synergies, 3 synergies, 4 synergies. The synergies were obtained from [17]. To project hand motions into the synergy space, the following process was followed:

1) Collect the hand joint kinematics via MediaPipe, $q_0 \in R^{20\,\times\,1}$
2) Find a least squares projection, $\Lambda \in R^{j\,\times\,1}$ to a subset of known j synergy(s) $V \in R^{20\,\times\,j}$ obtained from [17], such that $V \times \Lambda \approx q_0$.
3) Return the new hand motion projected into the synergy space, $q_{0,syn} = V \times \Lambda$.

With the new hand motions q_0, the Allegro Hand can be controlled as described in Section II-A.

Input Human Angle	Output Allegro Joint Angle
$\frac{\pi}{10}$	I_{ABD}
I_{MCP}	I_{MCP}
I_{PIP}	I_{PIP}
I_{DIP}	I_{DIP}
$\max(-2.4T_{ABD} - 0.35, 0)$	T_{ROT}
$\frac{\pi}{4}$	T_{ABD}
$\max(2.4T_{MCP}, 0)$	T_{MCP}
T_{IP}	T_{IP}
0	M_{ABD}
M_{MCP}	M_{MCP}
M_{PIP}	M_{PIP}
M_{DIP}	M_{DIP}
$-\frac{\pi}{10}$	R_{ABD}
R_{MCP}	R_{MCP}
R_{PIP}	R_{PIP}
R_{DIP}	R_{DIP}

TABLE I: A table mapping the human joint angles measured from MediaPipe to the commanded joint angles of the Allegro Robot. Some Allegro joints were set to constant values or were scaled from the measured human joints. All values are in radians.

C. Grasping in the Simulated Environment

Upon implementing synergies in the simulated hand controller, we aimed to better understand how these synergies may lead to better grasps. We designed an experiment where subjects were tasked with grasping and lifting various objects in the simulated environment. The objects of interest were: a sphere with low ($\mu = 0.2$) and high ($\mu = 0.8$) friction, a cube with low and high friction, and a mustard bottle.

In this experiment the iiwa was set to follow a specific trajectory. Start above the object at 0.8m, go toward the object at a sufficient height, stay there for several seconds, and raise back up to the starting position. To obtain this trajectory, the positions between the start and end poses were interpolated. While the arm was held at a position immediately above the object, the subject could attempt to grasp the object. Given this trajectory in end-point Cartesian space, the robot motion could be commanded as described in section II-A4.

The experiment had two dependent variables: grasp success rate – where success is defined by the subject's ability to effectively grasp the object before the arm started moving and (2) lift success rate – where success is defined by the subject's ability to effectively hold onto the object during the entire duration the arm was moving.

III. Results

A. Implementation of Teleoperation

As described in Section II-A, we were able to teleoperate the Allegro Hand in Drake using a control input that found joint kinematics from a custom script developed via MediaPipe. This implementation worked very well, and some example hand configurations are shown in Figure 3. The Allegro Hand was able to imitate the peace sign, fist, and grasp closure of the human hand.

Moreover, Figure 4 demonstrates the success of the `InverseKinematics` solver used to find the joint kinematics required to achieve the specified endpoint position. Specifically, the highlighted Meshcat slider in Figure 4 denotes the vertical z-position of the Allegro Hand. As the z-value was changed, the position of the hand in space changed accordingly.

B. Teleoperation using Kinematic Hand Synergies

We were also able to successfully implement teleoperation using Kinematic Hand Synergies as described in Section II-B. This result is best shown in Figure 5. In the top subfigure, we see the human teleoperating the hand using all 16 DOFs prescribed to the Allegro Hand. In the case where there is one synergy, the control input was projected into the subspace prescribed by one synergy. That is, all the joints were moving in a coupled manner simultaneously. As we increased the number of synergies in the controller, the Allegro Hand began to look more like the peace sign in the top part of the figure. Notably, the first synergy is often akin to a power grasp (i.e. closure of the hand, for example when holding a water bottle.); thus, in the case where we are reconstructing a peace sign with the first synergy, the robot's hand pose resembles a power grasp.

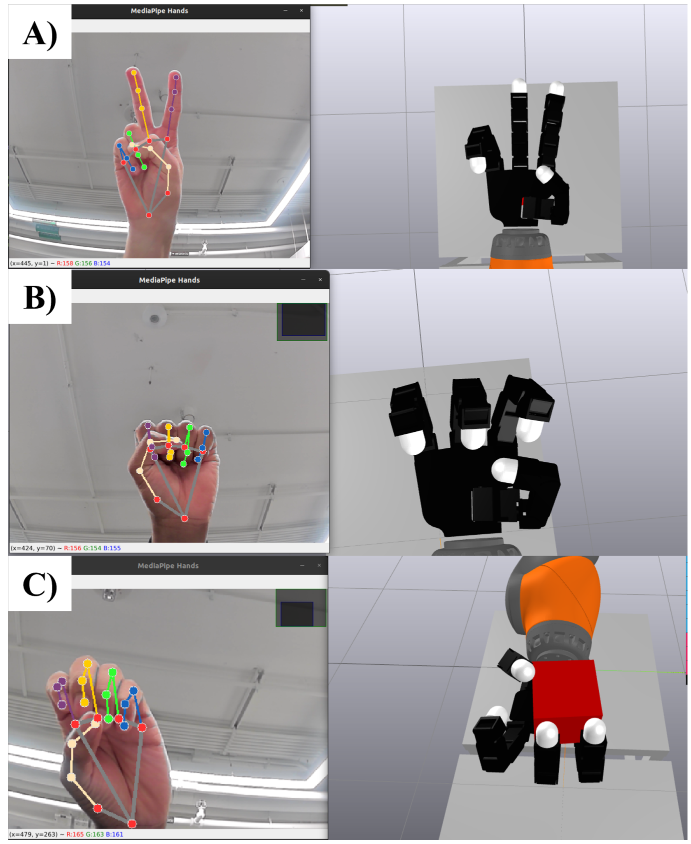

Fig. 3: Example of human (left) teleoperation of the Allegro Hand (Right). (a) A peace sign. (b) A fist. (c) Grasp closure of a cube.

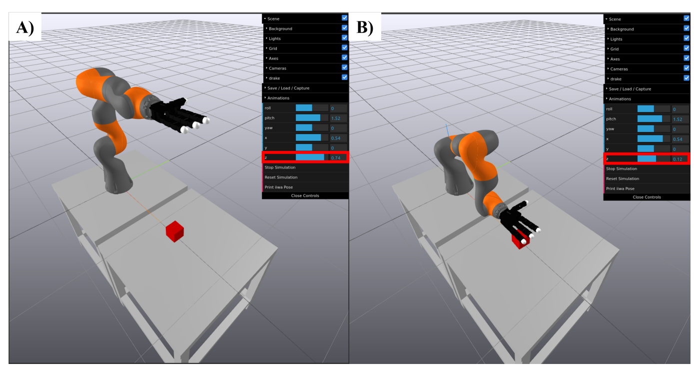

Fig. 4: Example of teleoperation of the iiwa arm using Meshcat Sliders to specify endpoint position. The highlighted Meshcat slider denotes the vertical z position of the Allegro Hand; as this value was changed, the position of the hand in space changed accordingly.

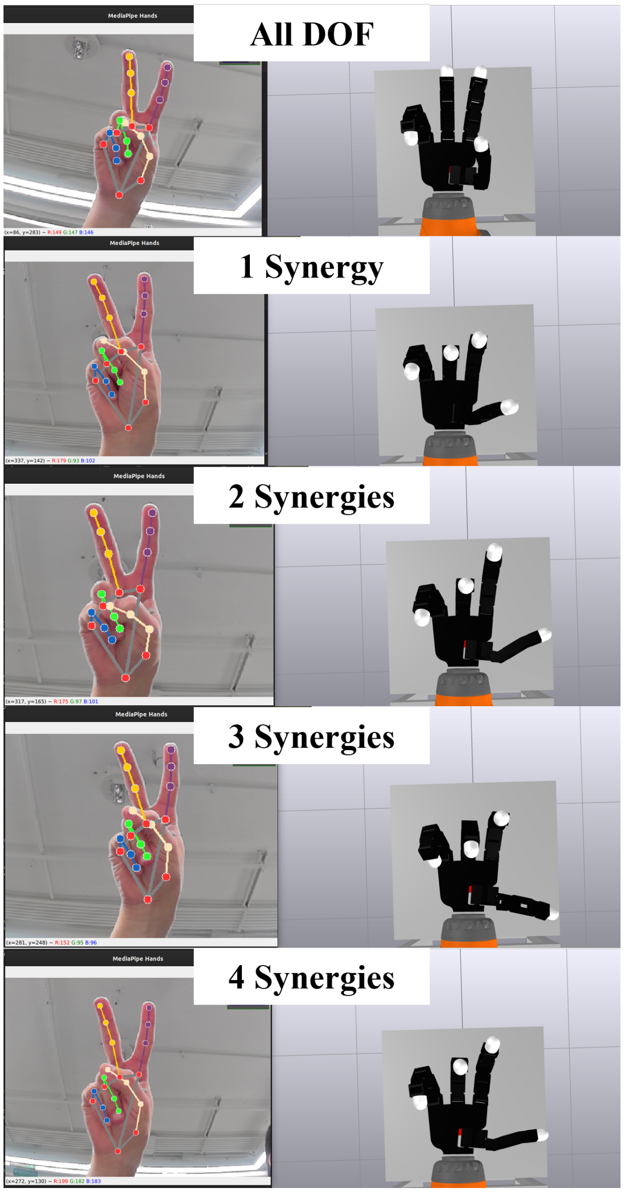

Fig. 5: Example of human (left) teleoperation of the Allegro Hand (right) using varying levels of synergies in the controller.

When we added the subspace described by the second synergy, more individuation of the thumb motion could be observed. This is because the second synergy is akin to a pinch grasp, where the thumb, index, and middle finger come together (for example, when holding a pencil). In the peace sign, while the thumb is rotated inwards, the middle and index fingers were extended. Thus, the reconstruction of the peace sign using two synergies had more outward thumb rotation and index and middle finger extension than the reconstruction of the peace sign using one synergy. It is evident that when fewer DOFs are employed (by coupling the motion using synergies), our ability to reconstruct the hand motion suffers. However, when grasping an object, synergies reduce the complexity of the grasp. Coupling this strategy with hand compliance may lead to finding stable grasps more easily.

C. Teleoperation to Grasp Various Objects

As described in Section II-C, we aimed to test the hypothesis that various objects in the simulated environment can be grasped using a subset of synergies. Unfortunately, our teleoperation method of using a standard RGB camera to track hand motion was not sensitive enough to recreate stable grasps around the objects we tested. In particular, controlling the Allegro Hand such that its fingers enclosed around an object to form an antipodal grasp with all four fingers was challenging. Furthermore, the lack of force feedback from teleoperation made it difficult to determine when sufficient force was applied to grip an object. The robot's motions were often exaggerated to compensate for these limitations, leading to our simulation becoming unstable (and our test objects flying away) when excessive force was applied from the Allegro Hand. This behavior is depicted in Figure 6; as the hand closes around the cube, the forces caused the cube to accelerate away and the Allegro Hand to diverge from the prescribed kinematics by the human operator.

IV. Discussion

This project set out to build a teleoperation pipeline that can ultimately be used to provide a basis for conducting behavior cloning on various manipulation tasks. We aimed to explore how using kinematic hand synergies could be employed to reduce the DOFs required to produce stable grasps in an anthropomorphic gripper. Specifically, our project was divided into 3 sub-tasks:

1) Teleoperate the Allegro Hand.
2) Teleoperate the Allegro Hand using known kinematic hand synergies.
3) Grasp various objects in simulation using teleoperation of the Allegro Hand.

We were able to successfully implement the first two tasks, but we were unfortunately unable to implement the third task.

Before using MediaPipe to extract human hand kinematics, we aimed to use the CyberGlove [7]. Unfortunately, CyberGlove only runs on Windows 10 while Drake only runs on Mac and Linux. Before abandoning the CyberGlove, we considered running the project in C++ using MuJoCo [18] as our simulation software. However, shortly after continuing the project in that direction, we realized that without Drake, we would not be able to use the class' resources (i.e., the professor and TAs) for help with debugging. Thus, we finally settled on exploring vision-based hand key point detection alternatives, which led us to MediaPipe.

Using MediaPipe, key points of the human hand were extracted and a custom script was written to turn those key points into joint kinematics. The joint kinematics from the human hand were used to control the joints of the Allegro Hand (Table I) via an inverse dynamics controller. Often times in prosthetic design, a similar controller is implemented. Specifically, some desired motion trajectory of the hand is extracted from the user and is used as the desired controller trajectory. However, to be consistent with the hypothesis of human motor control (which is hypothesized to lead to easier

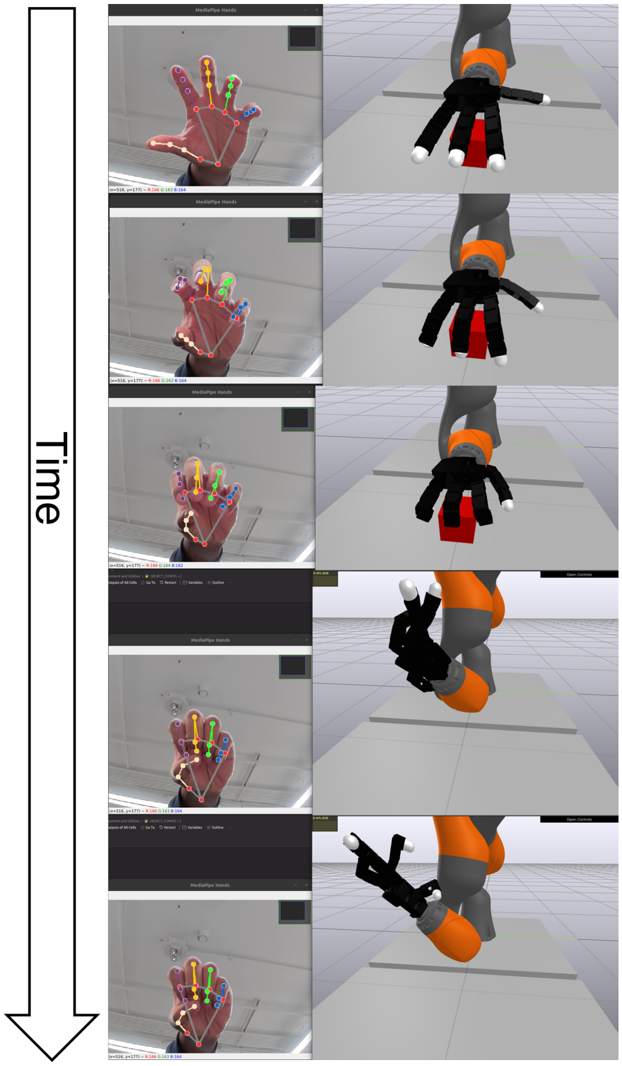

Fig. 6: Allegro Hand contact with the cube created an unstable simulation. As the hand closes around the cube, once the cube was touched, the cube accelerated away and the Allegro Hand no longer followed the prescribed human kinematics.

control for the user) [19], [20], we would have liked to implement Drake's `JointStiffnessController` function. It implements a controller,

$$\tau_{command} = -\tau_g - \tau_{applied} + kp(q_d - q) + kd(\dot{q}_d - \dot{q}) \quad (2)$$

where τ_g and $\tau_{applied}$ are the torques due to gravity and external forces respectively. There are key differences between the inverse dynamics controller and the joint stiffness controller. The inverse dynamics controller compensates for error with an integral term that predicts the next state. This can lead to instabilities during sudden dynamic changes. Meanwhile, the joint stiffness controller estimates measures of perturbations via the torques due to gravity and external forces. This formulation removes the need for an integral term, as direct force feedback can does not necessitate estimating the next state. In this work, we did not use Drake's joint stiffness controller because it was extremely sensitive to the time step value. Reducing the time step to avoid instabilities resulted in not being able to use MediaPipe to control the Allegro Hand in real-time. Thus, we continued with the inverse dynamics controller.

With this set up, we incorporated teleoperation via kinematic hand synergies (Task 2). The implementation is shown in 5. When less DOFs were employed (by coupling the motion using synergies), reconstructing the hand motions suffers. However, synergies reduce the complexity of the grasp, and coupled with hand compliance, it may lead to finding stable grasps more easily. This understanding can be useful in prosthetic hand design.

Since amputees prefer the use of prosthetic hands that look and function like biological hands [2], the study of the manipulation of dexterous robotic hands is essential. Previous work on using synergies in robotic hands suggests that the control of anthropomorphic hands can be simplified using synergies [21]. As synergies decrease the DOFs in the hand, it also decreases the number of actuators required to control the hand. In turn, this reduces the cost, size, and weight of the prosthetic device. Unfortunately, conducting tests on hardware devices can be time-consuming and limiting. We foresee that our proposed teleoperation pipeline using synergies may aid in testing the control of multi-finger robotic hands in simulation.

In addition, although a simple gripper can achieve many tasks reliably as we have seen in class, the identification of colinear antipodal grasp points is important for a two-finger gripper to have a large contact wrench cone [22]. For example, when grasping a rigid sphere, without finding the colinear antipodal grasp, the gripper may slip off. On the other hand, a compliant multi-finger hand may be able to engulf the sphere with multiple contact points. At each contact point, the compliance of the fingers may help to form the hand around the object. In fact, Bicchi et al. found that an anthropomorphic hand controlled with just one synergy can often obtain a stable grasp due to this embedded controller compliance [23].

Unfortunately, we were unable to implement the task of grasping various objects in simulation using teleoperation of the Allegro Hand. Thus, we were unable to conduct the experiment outlined in Section II-C. Our teleoperation methods failed to reliably produce suitable grasps around our test objects, and the lack of force feedback made it difficult to determine when sufficient grip was achieved. As shown in Figure 6, excessive forces on our test objects and our end-effector led the simulation to become unstable. It is possible that these large forces were due to our use of the inverse dynamics controller. When the hand comes into contact with the box, the hand undergoes an instantaneous change in contact dynamics. This sudden change in contact force greatly affects the integral term in the controller. This behavior may lead to large torques in the control input leading to instabilities in the simulation. We hypothesize that with sufficient external force feedback coupled with a joint stiffness controller, we may not have these problems. Exploring this idea is a topic of future work.

In developing this project, the authors learned a great deal about common control concepts in manipulation, as well as the implementation of these concepts in Drake. Creating a working system that combined an inverse dynamics controller with an inverse kinematics solver for a robot with 23 state variables, while adding in arbitrary objects to manipulate and test with, involved extensive research on programming with Drake, to the level of examining the source code for utilities found in the course textbook. Connecting our inverse dynamics controller with the inverse kinematics solver gave us experience with the benefits and drawbacks of both approaches, which helped us shape our programmatic and meshcat graphical interfaces to give feedback when either of these systems failed. Integrating external inputs in the form of hand joint angles from a camera, and adding objects separate from the robot but which still affected the simulation state, required making full use of the input and output connections between subsystems defined in Drake, cementing our knowledge of this platform.

We designed this teleoperation pipeline with the ultimate goal of applying it to learn the controller for dexterous robotic manipulation. Recently, the works of [4] and [24] have shown that human demonstrations can be used for imitation learning to complete various manipulation tasks. Using the collected demonstrations in simulation, their work augments the reinforcement learning objective to achieve robust results in real robots. With our work on teleoperation using synergies, we would like to further simplify this learning problem in the future. From the given data in teleoperation, we can extract the specific kinematic synergies in joint space used in this task [25]. Moreover, rather than tackling this problem using a reinforcement learning approach, behavior cloning — a method by which human subcognitive skills can be captured and reproduced in a computer program [26] — may be employed. In future work we plan to leverage behavior cloning as a form of supervised learning that will allow the mapping of synergistic inputs to an output torque on a cube.

V. Conclusion

Through this project, we have designed and implemented the teleoperation scheme for a multi-finger robotic hand in the Drake environment using human demonstrations captured by a single RGB camera. Through the addition of kinematic hand synergies in this pipeline, we aimed to simplify the degrees of freedom used to control the robotic hand. Although we were able to teleoperate the Allegro Hand using synergies, we were unsuccessful in grasping and manipulating different objects with the same pipeline. A limitation in the current design is our use of the inverse dynamics controller on the Allegro hand. When the hand comes into contact with an object, the instantaneous change in dynamics induced instabilities in the simulation environment. In the future, we plan to not only address this limitation but to also apply this teleoperation pipeline to conduct behavior cloning of various manipulation tasks based on human demonstration. This work can be further applied to the design and control of anthropomorphic prosthetic hands. Finally, in simulating this teleoperation environment in Drake, each of us gained valuable experience in building a simulation environment, implementing different control mechanisms, and integrating the simulation with other pipelines, such as MediaPipe.

Acknowledgment

We would like to thank our lab mate Maggie Wang who helped us write the code to extract joint kinematics from MediaPipe. Additionally, we would like to acknowledge Professor Russ Tedrake for teaching such an interesting class that gave us the knowledge to compose and conduct this project. Lastly, we would like to thank the course TAs Boyuan Chen, Anthony Simeonov, Ria Sonecha, and Lirui Wang who all helped us when trying to debug issues in Drake.

Individual Contributions

Raul was responsible for constructing the coding environments, which included a combination of a Deepnote project for rapid prototyping and a local setup running Drake on Ubuntu 20.04 for integration and test of the full teleoperation pipeline. Raul also integrated MediaPipe image processing with the subsystems developed by Michael and Kaymie in Deepnote into the full pipeline running locally, and set up the local environment for testing different configurations and control schemes. Raul created demo videos used in the final presentation and contributed to sections II, III, and IV.

Michael was responsible for the project conception, implementing the allegro and iiwa controllers, and implementing the inverse kinematics solver on DeepNote prior to Raul integrating the code locally on Ubuntu 20.04. He also produced the final presentation and contributed to writing sections I, II, III, and IV of the final report.

Kaymie implemented the simulation environment and worked on developing the position control of the iiwa prior to Raul's integration of the whole system. Initially, she also explored camera integration with DeepNote among other tasks. She also contributed to the final presentation, wrote section I, contributed to writing section IV, and revised the entire paper in detail.

References

[1] K. Ziegler-Graham, E. J. MacKenzie, P. L. Ephraim, T. G. Travison, and R. Brookmeyer, "Estimating the Prevalence of Limb Loss in the United States: 2005 to 2050," *Archives of Physical Medicine and Rehabilitation*, vol. 89, no. 3, pp. 422–429, 2008.

[2] G. Caserta, N. Boccardo, M. Freddolini, G. Barresi, A. Marinelli, M. Canepa, S. Stedman, L. Lombardi, M. Laffranchi, E. Gruppioni, and L. De Michieli, "Benefits of the Cybathlon 2020 experience for a prosthetic hand user: a case study on the Hannes system," *Journal of NeuroEngineering and Rehabilitation*, vol. 19, no. 1, p. 68, 2022.

[3] C. Tabor, "The Impact of Prosthetics on Amputee Quality of Life," 2018.

[4] Y. Qin, H. Su, and X. Wang, "From one hand to multiple hands: Imitation learning for dexterous manipulation from single-camera teleoperation," 2022. [Online]. Available: https://arxiv.org/abs/2204.12490

[5] S. Han, B. Liu, R. Wang, Y. Ye, and C. D. Twigg, "Online optical marker-based hand tracking with deep labels," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 4, 2018.

[6] K. Z. Zhuang, N. Sommer, V. Mendez, S. Aryan, E. Formento, E. D'Anna, F. Artoni, F. Petrini, G. Granata, G. Cannaviello, W. Raffoul, A. Billard, and S. Micera, "Shared human–robot proportional control of a dexterous myoelectric prosthesis," *Nature Machine Intelligence*, vol. 1, pp. 400–411, 2019.

[7] "CyberGlove Systems LLC." [Online]. Available: http://www.cyberglovesystems.com/

[8] "Shadow Teleoperation System." [Online]. Available: https://www.shadowrobot.com/teleoperation/

[9] "HaptX." [Online]. Available: https://haptx.com/

[10] B. Dorner, “Chasing the Colour Glove: Visual Hand Tracking,” 1994.

[11] L. Ge, Z. Ren, and J. Yuan, “Point-to-point regression pointnet for 3D hand pose estimation,” *Lecture Notes in Computer Science*, pp. 489–505, 2018.

[12] F. Zhang, V. Bazarevsky, A. Vakunov, A. Tkachenka, G. Sung, C.-L. Chang, and M. Grundmann, “MediaPipe Hands: On-device Real-time Hand Tracking,” 2020.

[13] A. Handa, K. Van Wyk, W. Yang, J. Liang, Y.-W. Chao, Q. Wan, S. Birchfield, N. Ratliff, and D. Fox, “Dexpilot: Vision based teleoperation of dexterous robotic hand-arm system,” 2019.

[14] A. Sivakumar, K. Shaw, and D. Pathak, “Robotic telekinesis: Learning a robotic hand imitator by watching humans on youtube,” 2022.

[15] R. Bellman, “Dynamic programming and stochastic control processes,” *Information and Control*, vol. 1, no. 3, pp. 228–239, 1958.

[16] M. Santello, M. Flanders, and J. F. Soechting, “Postural hand synergies for tool use,” *The Journal of Neuroscience*, vol. 18, no. 23, pp. 10 105–10 115, 1998.

[17] A. M. West, Jr., “Towards a non-invasive measurement of human motion, force, and impedance during a complex physical-interaction task : wire-harnessing,” 2020.

[18] E. Todorov, T. Erez, and Y. Tassa, “MuJoCo: A physics engine for model-based control,” in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, oct 2012, pp. 5026–5033.

[19] N. Hogan and D. Sternad, “Dynamic primitives of motor behavior,” pp. 727–739, 2012.

[20] N. Hogan, “Physical interaction via dynamic primitives,” in *Geometric and Numerical Foundations of Movements.*, J. Laumond, N. Mansard, and J. Lasserre, Eds. Springer International Publishing AG, 2017, pp. 269–299.

[21] C. Y. Brown and H. H. Asada, “Inter-finger coordination and postural synergies in robot hands via mechanical implementation of principal components analysis,” *IEEE International Conference on Intelligent Robots and Systems*, pp. 2877–2882, 2007.

[22] R. Tedrake, *Robotic Manipulation*, 2022. [Online]. Available: http://manipulation.mit.edu

[23] A. Bicchi, M. Gabiccini, and M. Santello, “Modelling natural and artificial hands with synergies,” *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 366, no. 1581, pp. 3153–3161, 2011.

[24] S. P. Arunachalam, S. Silwal, B. Evans, and L. Pinto, “Dexterous imitation made easy: A learning-based framework for efficient dexterous manipulation,” 2022. [Online]. Available: https://arxiv.org/abs/2203.13251

[25] C. R. Mason, J. E. Gomez, and T. J. Ebner, “Hand synergies during reach-to-grasp,” *Journal of Neurophysiology*, vol. 86, no. 6, pp. 2896–2910, 2001.

[26] P. Munro, H. Toivonen, G. I. Webb, W. Buntine, P. Orbanz, Y. W. Teh, P. Poupart, C. Sammut, C. Sammut, H. Blockeel, D. Rajnarayan, D. Wolpert, W. Gerstner, C. D. Page, S. Natarajan, and G. Hinton, “Behavioral Cloning,” *Encyclopedia of Machine Learning*, pp. 93–97, 2011.

GitHub Link: https://github.mit.edu/rlargaes/64212_Project
Project video: https://youtu.be/IImbUH8T4I4