A Mirroring Hand That Is Not in a Mirror

Caden Chin, Kangrui Wu, Zechen Wang

Abstract

This research focuses on the development of a realistic robotic hand with gesture recognition and imitation capabilities to aid painters and other users in mastering various hand poses. Unlike traditional wooden hand models, the proposed robotic hand aims to simulate the subtle movements, strength, and adaptability of the human hand. The system utilizes computer vision algorithms for hand gesture recognition and Python for control program development. The research explores various techniques in the field of gesture recognition, including Convolutional Neural Networks (CNNs), to enhance the accuracy of gesture classification. The robotic hand offers preset gestures that can be selected and fine-tuned through a user interface. The potential applications of the realistic robotic hand extend beyond painting to areas like game development and medical rehabilitation. The ongoing efforts are focused on developing an accurate, intelligent, and efficient robotic hand to enhance gesture recognition and provide convenience and surprise to users.

1 Introduction

Painting people often use wooden hand models in order to better draw various hand poses. Such models enable painters to better understand the structure of the human body, but it cannot fully simulate the subtle movements and strength of the human hand, nor can they reflect the difficulty and proficiency of posing, and the need for imagination. Therefore, we decided to develop a realistic robotic hand that can perform gesture recognition and imitation under the guidance of the user, with a high degree of precision and adaptability to help people better grasp various gestures and postures.

[11][2]

For the simulation and modeling of our robotic hand, we employ PyBullet, chosen for its noted performance in simulations with a high degree of freedom [11]. PyBullet's ability to run parallel simulations can accelerate the learning process, enabling the concurrent execution of multiple training scenarios that could improve the realism of our robotic hand's movements [11]. We write control programs and convert skeleton data into driving information for the manipulator using Python.

Hand gesture recognition employs monocular camera-based computer vision algorithms. We aim to compare several algorithms and their combinations to enhance accuracy, ultimately improving the robotic hand's ability to mimic gestures and enhancing the controllability and adaptability of the manipulator. A manually controlled rotatable base also enables the manipulator's actions to be observed from multiple angles, providing a more comprehensive reference.

We anticipate that this realistic manipulator will offer considerable value, particularly for novice painters, by providing a more dynamic and realistic model for studying hand gestures. Potential applications extend beyond the art world to game development, medical rehabilitation, and other fields. Our ongoing efforts aim to refine our robotic hand into a more accurate, intelligent, and efficient tool, striving to bring convenience and surprises to users.

(a) (b)

Fig.1 (a) A robotic hand (b) A wooden hand model

2 Background Related Work

2.1 Wooden Hand Models in Painting

Wooden hand models have long been employed in the field of painting to assist artists in accurately depicting various hand poses. These models enable a better understanding of the human body's structure, acting as an essential tool for mastering the intricacies of hand gestures and postures in artistic works. However, the static nature of wooden models limits their ability to fully capture the subtle movements and strength of the human hand, necessitating the development of more advanced tools for simulating and mimicking these aspects [6].

2.2 Gesture Recognition

The advent of computer vision and machine learning has facilitated significant progress in the field of gesture recognition. Several techniques, including the utilization of Convolutional Neural Networks (CNNs), have shown promise in recognizing and classifying hand gestures [1][2]. Building upon this progress, the hand tracking system combines the power of Mediapipe and OpenCV, leveraging a pre-trained model to identify 21 distinct landmarks that accurately mark the hand skeleton. To ensure robust detection, conservative thresholds of 0.8 and 0.9 are set for detection and tracking confidence, respectively, prioritizing accuracy even at the potential expense of speed. The system efficiently operates using a single monocular camera, providing the necessary visual input for hand tracking and analysis. This research highlights the potential for implementing advanced gesture recognition algorithms in robotic systems, enhancing their capabilities [2].

2.3 Simulation and Robotic Hand Modeling

To develop a realistic robotic hand capable of simulating and mimicking human hand gestures, advanced simulation, and modeling techniques are essential. Pybullet, a versatile framework, is employed for simulation, offering access to 21 hinge joints that correspond to the majority of human hand joints. Notably, the MCP and TM joints, which possess 2 degrees of freedom (DOF), are implemented using two hinge joints. However, it should be noted that the simulation assumes a fixed palm configuration and does not incorporate CMC joints into the model. Pybullet is widely recognized for its utility in robotic simulation, modeling, and control. The work by Qian et al. [3] demonstrates the successful implementation of Pybullet for simulating manipulation tasks, providing valuable insights into its application for developing a realistic robotic hand [7]. These techniques and the Pybullet framework play a vital role in creating a robotic hand that can faithfully replicate human hand gestures.

2.4 Applications of Gesture Recognition in Robotics

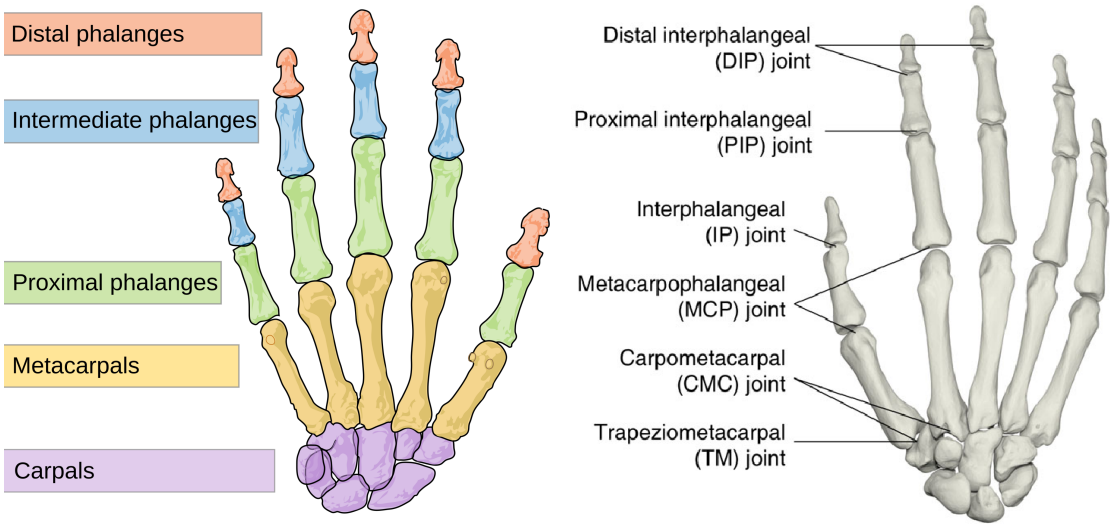
Gesture recognition has been applied to various aspects of robotics, including real-time interactions with mobile robots. The research by Konda et al. [4] explores the use of hand gestures for controlling mobile robots, demonstrating the potential for integrating gesture recognition into the development of a realistic robotic hand [10].

3 Technical Approach / Methodology / Theoretical Framework

3.1 Robot Hand Simulation

Our goal for the modeling of the robot hand in simulation is to replicate mechanical features of a human hand, either left or right, as best as we can. We use Solidworks to model such robot hand and use sw2urdf to transfer the model into the Pybullet package for simulation purposes. Current robot hand designs and wood hand models mostly focus on phalanges (finger bones) with MCP joints (knuckles). Our hand model features 23 hinge joints, using two standalone joints to replicate a real hand joint with two DOFs.

After balancing the pros and cons we decided to not model Carpometacarpal joints (CMC) since they will add significant complexity while contributing little to gesture accuracy. Also, we need some hard reference planes/vectors to establish the control angles.



(a) (b)

Fig.2 (a) Human hand skeleton [5] (b) Human hand joints [6]

3.2 Hand Gesture Recognition

For hand gesture recognition, we use Mediapipe and OpenCV (based on Python) to implement it. The algorithm relies only on a monocular camera and can show precise 3D skeleton layout. Inevitably, without a second camera, depth data generated from the machine learning algorithm is not trustworthy and needs to be enhanced. Our design doesn't require fast response speed and accuracy should always outweigh response time. After several tests, we chose to set the algorithm's detection confidence and tracking confidence set to 0.8 and 0.9. Refresh rate of the algorithm should vary depending on the hardware while for most personal computer it can likely achieve 20+ FPS.

3.3 Driving Algorithm

The driving data for the robotic hand is derived from angles obtained by converting coordinates. Two base vectors play a crucial role in this process. The finger base vector, obtained from the middle finger metacarpal (0-9), helps establish a reference for finger movements. Additionally, the palm normal vector, derived from the plane formed by the landmarks 0-5-17, provides orientation information. The hand's joints are categorized into three distinct groups: wrist joints, which are manually controlled by the user to achieve desired poses; grasping joints, responsible for mimicking gripping actions; and waving joints, enabling the fingers to perform waving gestures. By employing this framework, the robotic hand becomes capable of replicating a wide range of human hand movements with high precision and adaptability.

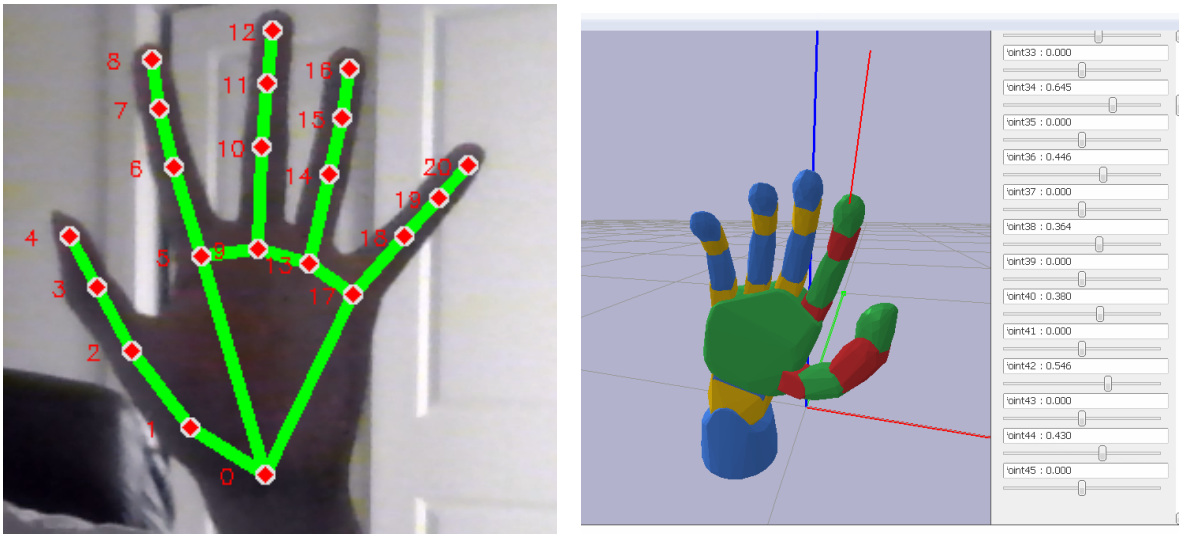
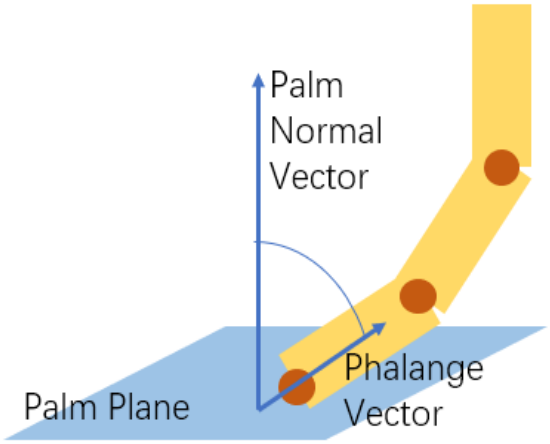


Fig.3 (a) 21 Hand landmarks (b) the Hand model in Pybullet

In the grasping joints category, several specific joints are involved: DIP (distal interphalangeal), PIP (proximal interphalangeal), IP (interphalangeal), and one degree of freedom (DOF) of the MCP (metacarpophalangeal) joints. The driving angles for the DIP, PIP, and IP joints are calculated using two vectors associated with the corresponding phalanges. These angles determine the flexion and extension of these joints, enabling realistic grasping movements. On the other hand, the MCP joint angles are calculated based on the vector of the proximal phalange and the palm normal vector. This calculation ensures accurate positioning and movement of the MCP joints in accordance with the hand's overall orientation. By considering these angles and vectors, the robotic hand can achieve natural and precise grasping motions, closely resembling human hand movements.

In the waving joints category, two types of joints are involved: MCP (metacarpophalangeal) joints and CMC (carpometacarpal) joints. For the thumb, its two waving joints (TM and MCP) are simplified into a single joint. The waving angle for this joint is determined by calculating the degree between the finger base vector and the projection of the proximal phalange on the palm plane. This measurement captures the flexion and extension of the joints, allowing the robotic hand to perform realistic waving gestures. By considering these angles and projections, the robotic hand can mimic waving motions, providing a more natural and expressive interaction capability.



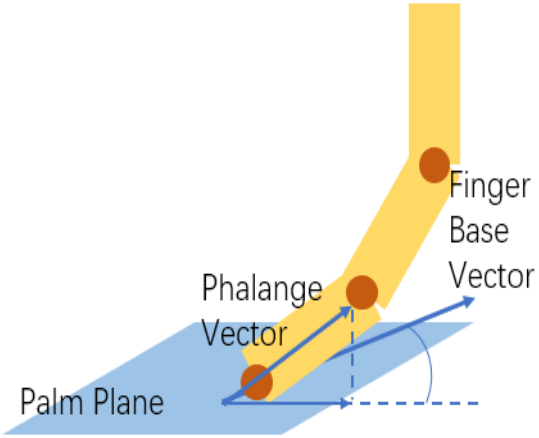


Fig.4 (a) Vector figure for grasping joints **(b) Vector figure for waving joints**

To enhance the recognition depth data, an improved approach is implemented using the fixed length ratio between bones. The index finger metacarpal vector is selected as the reference vector due to its relatively limited maneuverability. By establishing this reference, the fixed length ratio between different bones is utilized to obtain more accurate and reliable depth information. In most cases when the measured length is either longer or shorter than the ideal length, the depth data (z-axis coordinate) will be adjusted to fit the desired length (see Fig.5(b)).

However, when the ideal length is shorter than the projection of the measured length on the palm plane, which often happens when the palm is not clear to the camera or it's leaning at huge angles, it's impossible to achieve the ideal length by simply adjusting depth and our solution is to change its coordinates in all directions to shorten the length while maintaining the orientation. The performance can be further improved if the user can use some hard reference like a colored bar attached to the hand with a known length. Additionally, a hand size measure step should be taken to register the user's hand size features. This step ensures that the system adapts to individual variations in hand sizes, further improving the precision and adaptability of the hand recognition process.

$$\frac{l_{Ideal}}{r_{Measured}} = \frac{l_{Known}}{r_{Known}}$$

l, phalange length; r, reference length

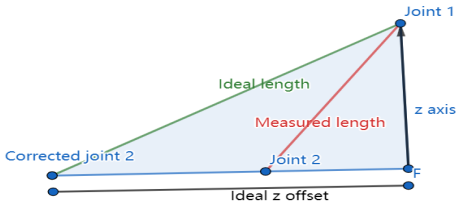


Fig.5 (a) Formula for ideal length **(b) Method for improving depth data**

3.4 Filter Adjustment of Hand Joint Data

To enhance the accuracy and stability of the hand joint data, we employ two filtering algorithms: the Kalman filter and the Particle filter. These filters refine the raw measurements by incorporating past observations and probabilistic modeling.

The Kalman filter is a recursive estimation algorithm that operates based on a linear dynamical system model. It combines the current measurement with the predicted state estimate to generate an optimal estimation of the true state. In our implementation, we initialize a Kalman filter object for each index joint. The filter's parameters, such as the initial state mean, initial state covariance, transition matrices, observation matrices, transition covariance, and observation covariance, are set to control the filter's behavior. These parameters are tuned to ensure effective estimation and tracking of the hand joint angles.

In our filtering process, we update the Kalman filter for each joint by providing it with the observed angle measurement. The filter utilizes the transition and observation models to predict and correct the state estimate. The filtered angle is obtained as the mean of the estimated angles from the Kalman filter. By continuously updating the filter with new measurements, it adapts to changes in the hand joint angles and reduces the impact of noise and measurement errors.

Compare to the Kalman filter, we also implement the particle filter to further improve the estimation accuracy. The particle filter, also known as a sequential Monte Carlo method, represents the state estimate as a set of particles (samples). Each particle corresponds to a possible state hypothesis, and their distribution approximates the true state distribution. In our implementation, we initialize a particle filter object for each index joint.

To update the particle filter, we define a motion model that captures the dynamics of the hand joint angles. The particles' positions are adjusted based on this motion model. We then update the particle weights using the observed angle measurement. The weights reflect the likelihood of each particle being in the true state based on the measurement. After updating the weights, we perform resampling, where new particles are drawn from the existing particles based on their weights. The resampling step ensures that particles with higher weights have a higher chance of being selected, thus improving the representation of the true state distribution.

The filtered angle in the particle filter is calculated as the mean of the particles' angles. By considering multiple samples (particles), the particle filter can handle non-linearities and uncertainties in the hand joint angles. It provides a robust estimation that accounts for the inherent variability and noise present in the measurements.

Throughout the filtering process, we calculate the mean squared error (MSE) between the filtered angles and the unfiltered (observed) angles. The MSE values serve as a quantitative measure of

the filtering performance, allowing us to assess the effectiveness of the filters in reducing the error between the estimated angles and the ground truth.

4 Evaluation

4.1 Evaluation Method

One of the main objectives of this study is to compare the Particle Filter (PF) and Kalman Filter (KF) in a robot hand experiment, focusing on their practical application and effectiveness. We evaluated the performance by measuring the Mean Squared Error (MSE) between the actual and predicted angles over multiple iterations.

The results showed that the KF had a slightly lower average MSE (0.244) than the PF (0.257). Additionally, the consistency of the KF was marginally better, with a slightly lower standard deviation of MSE (0.120) compared to the PF (0.121). This suggests that the KF may offer more precise and reliable results.

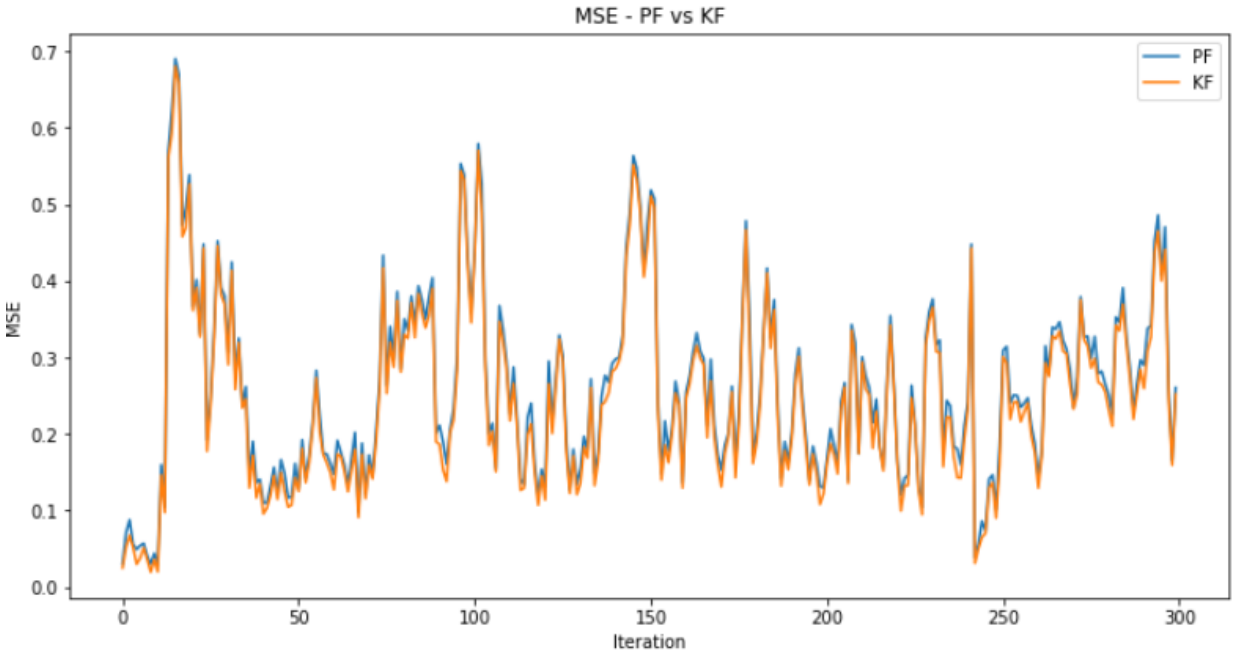


Fig.6 Comparison of MSE data between Kalman filter(Orangle) and Particle filter(Blue)

Further analysis revealed that the KF's benefits became more apparent when observing each joint index. The KF data was smoother, exhibited fewer fluctuations than the PF data, and had overall lower values compared to the unfiltered data. This indicates better accuracy with the KF.

Fig.7 Comparison of changes of robot hand joint 7 (Thumb_CMC) among unfiltered data(Blue), Kalman filtered data(Green) and Particle filtered data(Orangle)

When we applied these results to a virtual machine, the KF's advantage was clearly visible. The robot hand-tracked the user's gestures more smoothly and with less noticeable wobble when using the KF, enhancing user experience and performance.

A t-test further confirmed the KF's superior performance, resulting in a p-value of 0.192. The effect size, represented by Cohen's d value of 0.107, suggested that the KF performed about 1.05 times better than the PF.

To sum up, while the statistical differences between KF and PF may seem small, their practical implications are significant. The KF's smooth and stable performance in the robot hand experiment underlines its potential advantages over the PF in precise and consistent movement 11 tracking applications. Therefore, when it comes to tasks that require smooth gesture tracking and improved user experience, the Kalman Filter is the optimal choice.

4.2 Evaluation Conclusion

In summary, although the statistical differences between the KF and PF may appear small, their practical implications are significant. The KF's smooth and stable performance in the robot hand

experiment underscores its potential advantages over the PF in precise and consistent movement tracking applications. Therefore, for tasks that require smooth gesture tracking and improved user experience, the Kalman Filter emerges as the optimal choice.

These findings contribute to the understanding of filtering techniques in robotics and highlight the practical benefits of the Kalman Filter. Further research could explore variations of these filters and their application in different contexts, expanding the knowledge base for optimal filtering solutions in robotics and related fields.

In conclusion, this report focused on the development of a realistic robotic hand with gesture recognition and imitation capabilities to assist painters and users in mastering various hand poses. The proposed robotic hand aimed to simulate the subtle movements, strength, and adaptability of the human hand, surpassing the limitations of traditional wooden hand models.

The system incorporated computer vision algorithms, utilizing monocular camera-based hand gesture recognition techniques. Various algorithms, including Convolutional Neural Networks (CNNs), were explored to enhance gesture classification accuracy. The robotic hand provided preset gestures that could be selected and fine-tuned through a user interface. The potential applications of the realistic robotic hand extended beyond painting to areas such as game development and medical rehabilitation.

The technical approach involved modeling the robotic hand using PyBullet for simulation, with control programs developed in Python. Hand gesture recognition was achieved through Mediapipe and OpenCV, enabling an accurate 3D skeleton layout. The driving algorithm utilized angles derived from coordinates and vectors, allowing the robotic hand to replicate human hand movements with precision and adaptability.

The evaluation compared the performance of the Kalman Filter (KF) and Particle Filter (PF) in terms of Mean Squared Error (MSE) between actual and predicted angles. The results demonstrated that the KF exhibited a slightly lower average MSE and better consistency compared to the PF, indicating more precise and reliable results. Further analysis revealed that the KF produced smoother and more accurate joint movements compared to the unfiltered data.

Applying the results to a virtual machine showcased the advantages of the KF, as the robotic hand-tracked user gestures smoothly and with the reduced wobble, enhancing the user experience and performance.

Statistical analysis, including a t-test and effect size measurement, confirmed the KF's superior performance over the PF. The practical implications of these findings emphasized the KF's

potential advantages in precise and consistent movement-tracking applications, particularly in tasks that require smooth gesture tracking and improved user experience.

In summary, this research successfully developed a realistic robotic hand with gesture recognition and imitation capabilities. The comparison between the KF and PF highlighted the KF's superiority in terms of accuracy, stability, and user experience. The findings contribute to the advancement of robotic systems in various fields and provide a valuable tool for painters, game developers, and medical rehabilitation practitioners. Future work aims to refine the robotic hand further, focusing on accuracy, intelligence, and efficiency to enhance gesture recognition and provide convenience and surprises to users.

5 Timeline and Individual Responsibilities

Table 1 shows our time frame and individual responsibilities.

Table 1 Timeline and Individual Responsibilities

Timeframe	Work Task	Person in Charge
April 5-9	Environment setup and debugging	All
April 10-16	Manipulator modeling and debugging	Kangrui Wu, Zechen Wang
April 17-23	Gesture recognition algorithm establishment	Caden Chin, Zechen Wang
April 24-30	Gesture recognition algorithm optimization	Caden Chin, Zechen Wang
May 1-4	Integration of gesture recognition and robot	All
May 5-6	Establishment and debugging of preset gestures	Kangrui Wu
May 7	Project report and summary	All

6. Video

Video Link:https://youtu.be/2MV2wlbh46w

Reference

[1] M. Oudah, A. Al-Naji, and J. Chahl, “Hand Gesture Recognition Based on Computer Vision: A Review of Techniques,” Journal of Imaging, vol. 6, no. 8, p. 73, Jul. 2020, doi: 10.3390/jimaging6080073.

[2] F. Zhan, "Hand Gesture Recognition with Convolution Neural Networks," 2019 IEEE 20th International Conference on Information Reuse and Integration for Data Science (IRI), Los Angeles, CA, USA, 2019, pp. 295-298, doi: 10.1109/IRI.2019.00054.

[3] W. Qian et al., "Manipulation task simulation using ROS and Gazebo," 2014 IEEE International Conference on Robotics and Biomimetics (ROBIO 2014), Bali, Indonesia, 2014, pp. 2594-2598, doi: 10.1109/ROBIO.2014.7090732.

[4] K. Konda, H. Schulz, A. Königs and D. Schulz, "Real time interaction with mobile robots using hand gestures," 2012 7th ACM/IEEE International Conference on Human-Robot Interaction (HRI), Boston, MA, USA, 2012, pp. 177-178, doi: 10.1145/2157689.2157743.

[5] Wikimedia Commons, File:Scheme human hand bones-en.svg, https://commons.wikimedia.org/wiki/File:Scheme_human_hand_bones-en.svg

[6] I. Bullock, J. Borràs, and A. Dollar, "Assessing assumptions in kinematic hand models: A review," in Proceedings of the IEEE RAS and EMBS International Conference on Biomedical Robotics and Biomechatronics, 2012, pp. 1239-1246, doi: 10.1109/BioRob.2012.6290879.

[7] W. Qian et al., "Manipulation task simulation using ROS and Gazebo," 2014 IEEE International Conference on Robotics and Biomimetics (ROBIO 2014), Bali, Indonesia, 2014, pp. 2594-2598, doi: 10.1109/ROBIO.2014.7090732.

[8] Quentin De Smedt, Hazem Wannous, Jean-Philippe Vandeborre, Heterogeneous hand gesture recognition using 3D dynamic skeletal data, Computer Vision and Image Understanding, Volume 181, 2019, Pages 60-72, ISSN 1077-3142, https://doi.org/10.1016/j.cviu.2019.01.008.

[9] Corradini, A. (2002). Real-Time Gesture Recognition by Means of Hybrid Recognizers. In: Wachsmuth, I., Sowa, T. (eds) Gesture and Sign Language in Human-Computer Interaction. GW 2001. Lecture Notes in Computer Science(), vol 2298. Springer, Berlin, Heidelberg. https://doi-org.ezproxy.library.tufts.edu/10.1007/3-540-47873-6_4

[10] N. N. Hoang, G. -S. Lee, S. -H. Kim and H. -J. Yang, "Continuous Hand Gesture Spotting and Classification Using 3D Finger Joints Information," 2019 IEEE International Conference on Image Processing (ICIP), Taipei, Taiwan, 2019, pp. 539-543, doi: 10.1109/ICIP.2019.8803813.

[11] Körber, Marian, et al. "Comparing popular simulation environments in the scope of robotics and reinforcement learning." arXiv preprint arXiv:2103.04616 (2021). https://doi.org/10.48550/arXiv.2103.04616Links to an external site.