GRIPaint

Project Report

Team: **EXCJT**ing

Authors: Ellen Redgrave, Xinyan Ye, Chengyang Zhu, Jikai Wu, Tianxiao Wang

Date: March 30th, 2023

Table of Contents

The implementation of GRIPaint can be accessed via the following GitHub Repository:

https://github.com/xy2119/HCARD_Augmentation_2023

1. Introduction

The augmentation of the human body uses technology to extend or enhance a person's natural abilities. There are various types of augmentation, however, all augmentation is limited to targeting either the sensing or movement of the body. On the other hand, there are unlimited applications including being used to assist with daily activities like eating or for the rehabilitation of patients [1]. Our task was to design a product which augments the hand and compete against other groups.

Our goal as a group was to design a digital controller that motivates users to exercise their grip control, and therefore augment their capability of accessing digital interfaces such as drawing boards and games.

1.1 The Problem

Grip strength has been found to be an accurate marker of overall physical and psychological health, especially in the elderly. [2] Elderly people start to lose their grip strength as their muscle mass decreases, as well as other's such as people with carpal tunnel syndrome or arthritis. [3] Rehabilitation exercises that include grip strength have been found to be beneficial and increase the grip strength of an individual [4], however both elderly people and younger individuals can struggle to engage in rehabilitation exercises due to their challenging or boring nature.

Creativity and artistic expression have been found to have a great impact on the health of elderly people. Joining a choir improving participants' health compared to a baseline group not in a choir [5], and art has been found to have numerous benefits in the elderly population [6] Elderly people enjoy being creative and often have hobbies such as painting, writing, drawing, and knitting. However, as they get older, they may struggle to take part in activities that they used to enjoy due to reduction in mobility and strength. There is a gap in the market for a rehabilitation system focusing on accessible creativity and self-expression. This product would augment the users' grip strength by providing engaging training over time.

1.2 Design objectives and specifications

After analysing the problem and needs of our target users, the team outlined a working plan with three key milestones (figure 1) and specific tasks to fulfil our design objectives (in table 1). The overall objective is to design a novel hand controller and a digital application for human-computer interaction which puts emphasis on utilising *grip* actions to create paint strokes in a digital medium.

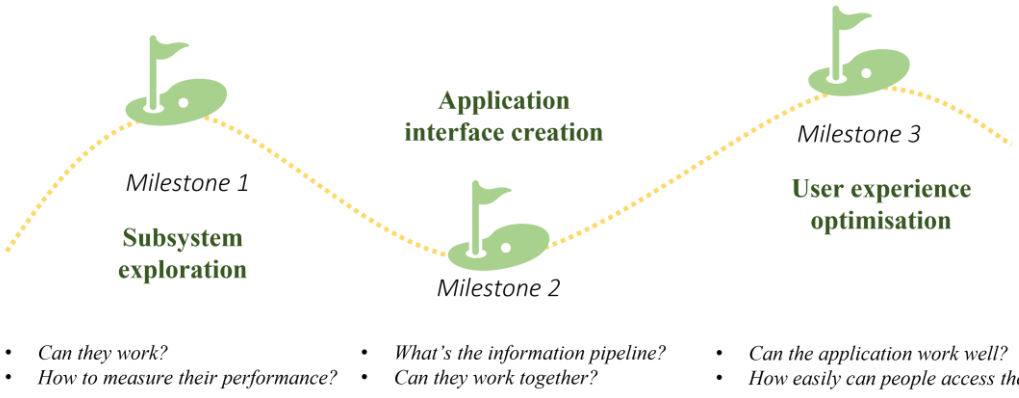


Figure 1: Key milestones.

Table 1: Objectives and specifications of tasks

Objective	Actions	Responsibilities
1. Create an ergonomic gripper design as the game controller (the hardware interface)	- prototype various forms of gripper and compare how intuitive they can be used - iterate the design to accommodate the engineering needs (design for manufacture and assembly)	Chengyang Ellen
2. Create an interactive digital medium as the visual display (the digital interface)	- define key user interfaces (UI) - implement the desired functions in Unity	Tianxiao Ellen
3. Test the feasibility and performance of using a force sensitive resistor to measure the grip strength	- construct a circuit to receive useful analogue reading using Arduino	Jikai
4. Test the feasibility and performance of position tracking using computer vision techniques	- explore methods to enable hand tracking - benchmark the performance	Xinyan
5. System Communication	- enable the Open Sound Control through user datagram protocol in Unity and Python for cross system information exchange	Tianxiao Xinyan

2. Design of GRIPaint

2.1 Concept summary

Our design demonstrates workspace and range augmentation by extending the natural range of the hand using motion tracking technology. The concept of our hand-held controller makes interaction with the computer more accessible to users with low dexterity such as the elderly. The painting software we designed is one example of the many high dexterity activities our users may struggle to partake in traditionally, that we have adapted to make more accessible and responsive. Furthermore, the gripper design in combination with the implementation of FSRs and buttons not only communicates with the software but also acts as a method to train grip strength.

Compared to similar products on the current market like GripAble [7], where design is primarily focused on mechanical elements, our design incorporates motion tracking which could allow us to extend the capability and the range of rehabilitation training.

2.2 Hardware

2.2.1 Shape of gripper design

When designing the shape of the gripper, it was important to consider the ergonomics of the product, to ensure that the design was comfortable and simple to use for elderly users and others with reduced

grip strength. Furthermore, the design must also fit all the required electronic components. Existing products and literature were reviewed to understand what shapes and features were effective. The two most of note were GripAble which utilises gentle curved and a specific grip area [8], and Grip-Ball, which is made from soft materials and a ball shape to make it more comfortable for the user to squeeze. [9]

Prototype 1 was a simple cube that had a hollow centre and holes in the sides for sensors. This design was used to test the initial feel of the gripper, and it was decided that the design needed to be curved for easier holding, and with larger grip areas.

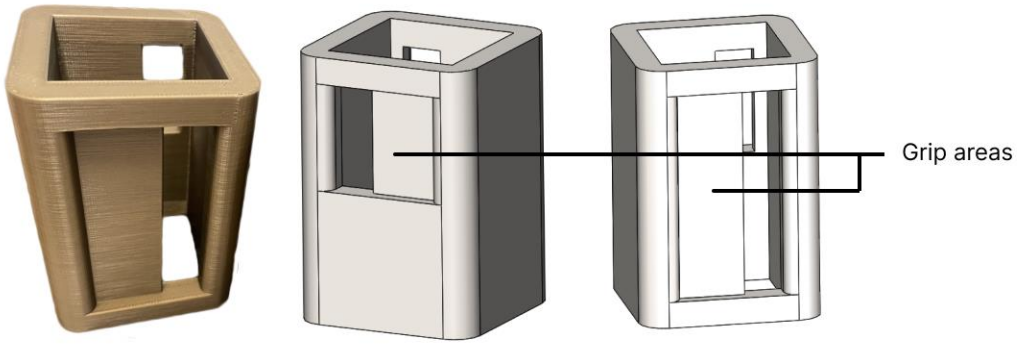


Figure 2: Prototype-1: CAD and 3D printed model.

The design was then developed to have a greater curvature in the side of the body for easier gripping. A greater inset was also added to leave room for FSR sensors and buttons. The other side of the design was left flat for the ArUco tag to be placed for tracking.

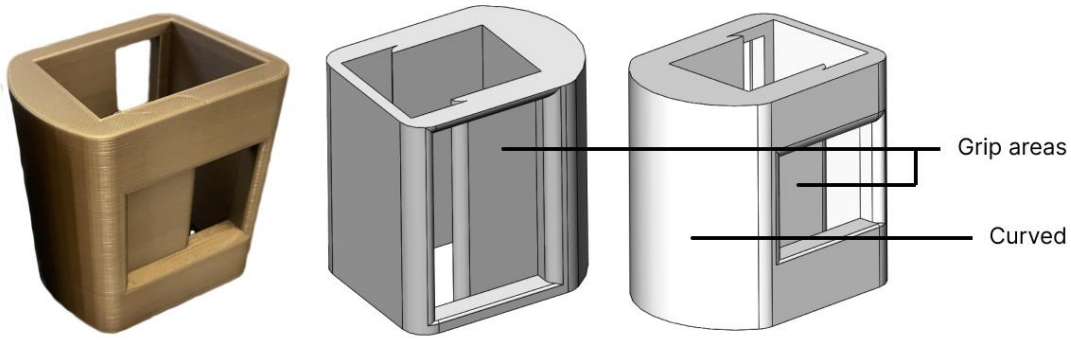


Figure 3: Prototype-2: CAD and 3D printed model.

It was determined that a more universal design should be implemented following the review of existing designs. Therefore, an ovoid prototype was developed. The grip area on the side of the design was

created so that the product was more like the Jamar dynamometer - the industry-standard position for grip strength testing [10]. Testing the prototype led us to reducing the size of the body to be more ergonomic for a wider range of users and moving the ArUco tag to above the body and rotating it 90° to reduce twisting wrist strain.

2.2.2 Final Design

The final design is detailed in the diagram below. The ArUco tag is placed on the top of the design, and resistive feedback is given through foam padding below the grip area. The design contains space inside for the sensors and is attached to the breadboard on the table through cables. The product was 3D printed and assembled using screws, and the tag holder was laser cut and attached using glue through a hole in the top of the design.

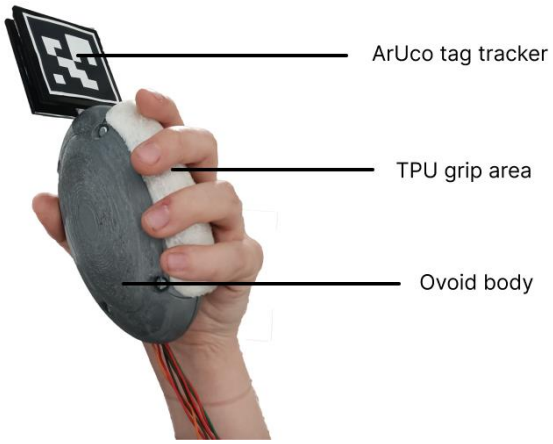


Figure 4: Final prototype.

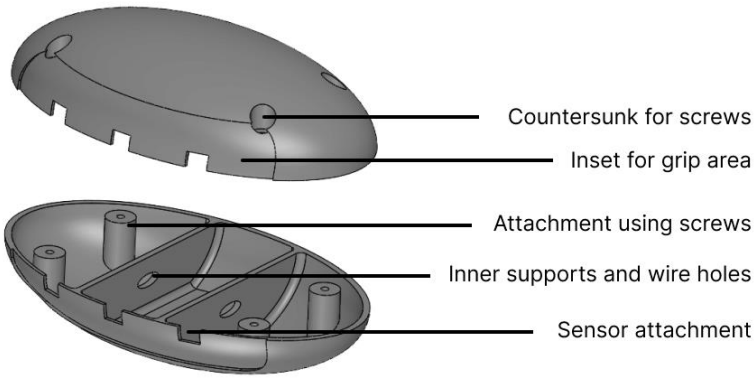


Figure 5: CAD design of final prototype.

2.2.3 Electronic components

The FSR was chosen to measure force as it is more compact and simpler to implement in comparison to load cells and strain gauges. The circuit diagram used to obtain a force reading between 0 and 1023 [11] is shown in figure 5. The value of the resistor in the voltage divider was chosen to be 1MΩ as it provides a linear and sensitive force range.

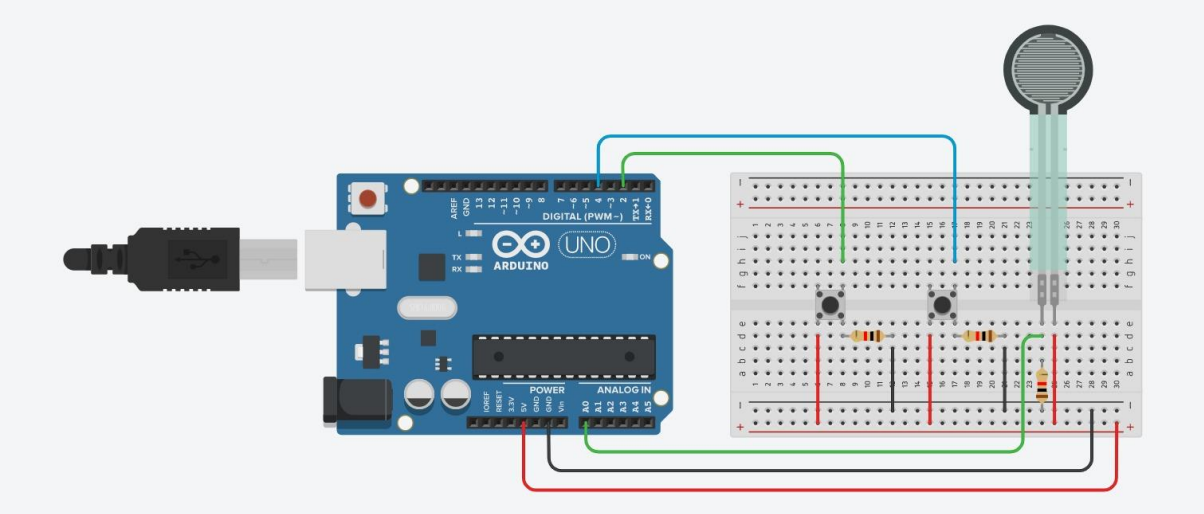


Figure 6: Circuit diagram.

A disadvantage of the FSR is the lack of user feedback. This was addressed with the design choice of using a mechanical button to simulate the mouse click and using the FSR exclusively for measuring force. The buttons provide tactile and audio feedback when pressed which improves user experience. Furthermore, the button is less prone to noise making it more reliable. Another advantage of the button is its binary serial output of 0s and 1s which make it simpler to define functions within the software.

One design aspect important to every electrical system is the reduction of noise and interference. Noise affects this sub-system by introducing fluctuations into the FSR reading which decreases the accuracy.

Table 2: Methods to reduce input noise.

Method	Observation	Final design choice
Increasing delay between readings	Significantly reduced fluctuation of the reading	Delay of 80ms
Implementation of a moving average filter	Reduced the range of fluctuation of the reading	Implementation of a first order moving average filter with a window size of 25
Twisting of the wires [12]	Slight improvement in reducing interference	The wires of each FSR or button were twisted together
Soldering of components onto stripboard	No significant improvement in reducing noise	Components were placed on a breadboard
Implementation of capacitors	Reduced the fluctuation of the reading significantly but the effect weakened as the capacitor was left to charge up	No capacitors used in the circuit

2.3 Software

2.3.1 Game interface design

User Interface

An intuitive user interface is essential for our elderly target users. Therefore, we adopted a simplistic layout for easy navigation, using large, clearly labelled buttons, straightforward icons, and minimal text

to reduce cognitive load. A high-contrast colour scheme is chosen in our next iteration, which ensures visibility of text and icons against the background.

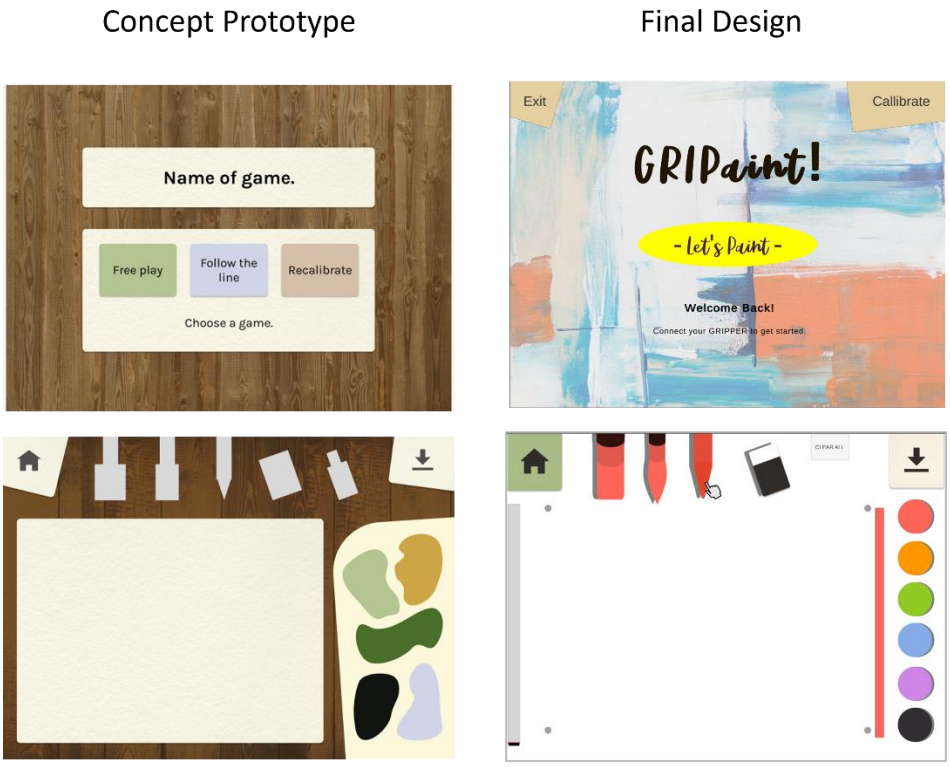


Figure 7: The initial UI design and final concept, for main page, and drawing page.

Affordance for improved user experience

Drawing from Self-determination Theory in HCI research [13, 14], our system supports players' intrinsic motivations through the affordance of **autonomy, competence, and relatedness**. Several design decisions were made to fulfil such needs.

1. *Autonomy*: Players can freely explore different paintbrush types and colours without strict rules, while drawing boundaries serve as creative constraints.

2. *Competence*: The cursor responds to hand position, creating a positive feedback loop. Calibration between the input space and camera perspective is achieved using a design trick to rescale the pointer size based on the size of ArUco tag recognised, visually prompting the player to stand at appropriate distance from the system.

3. *Relatedness*: We considered the option of save and download drawn images for sharing. In the future development roadmap, we aim to integrate more social features, such as using mini-games with leader boards and to motivate regular grip control practice.

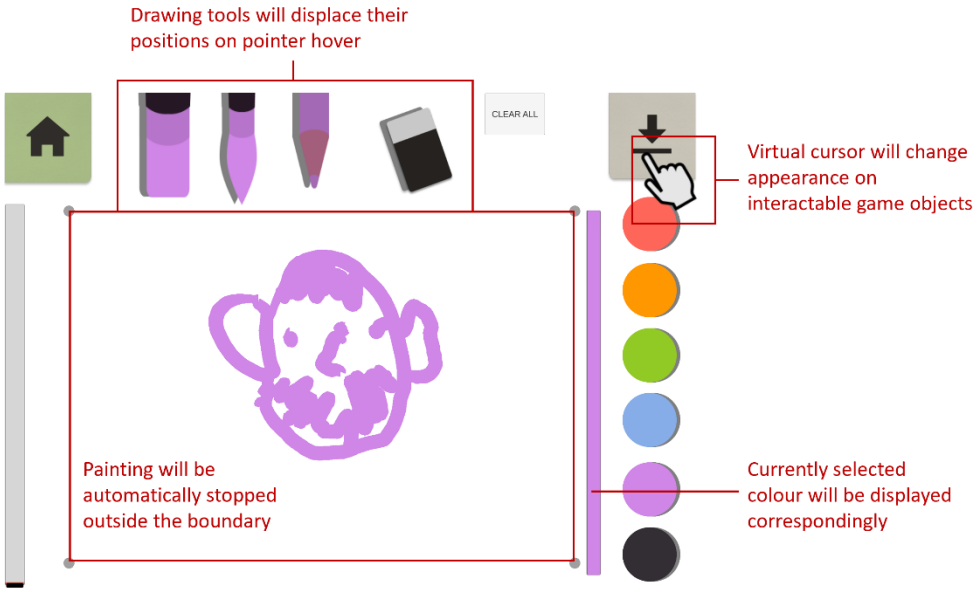


Figure 8: The painting UI with annotated design affordances.

Player Input handling

An OSC listener is set up using the user datagram protocol to process information [15]. Arduino board serial byte output is transmitted via a Python OSC package, splitting data into game states and FSR slider bar value updates for calibration. (e.g., pressed down, or up) and updates the FSR slider bar value in the calibration setting. Since reading high-rate serial output directly will result in significant graphic lag, OSC became an effective workaround and was implemented through an event system, which will only trigger when there's a value change, and is independent of the operation system our application is running on.

2.3.2 Motion Tracking

Motion Tracking System

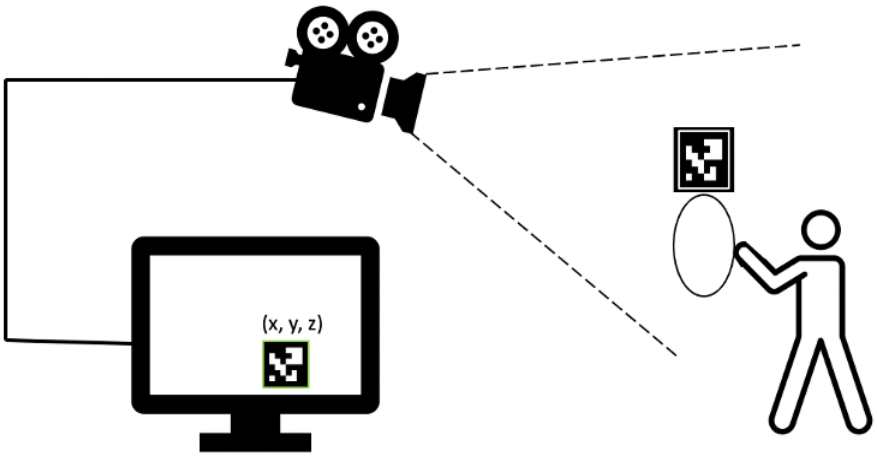


Figure 9: Illustration of the Tag Detection System

The motion tracking system aims to enable precise user movement tracking and facilitate mapping between user space and drawing space. Object detection has been a focal point in computer vision and robotics research, leading to numerous methods over time. Early systems faced challenges like occlusion and false detection, spurring the development of advanced fiducial marker systems such as ArUco Tags, QR Codes, and AprilTags [17]. ArUco tags use predefined binary patterns for unique identification and are easily detected and identified for real-time tracking, as shown in Figure 9. Explanation of algorithm is attached in Appendix II.

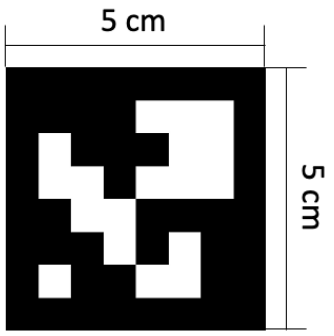


Figure 10: ArUco tag with 5cm x 5cm dimensions.

We chose ArUco tags due to their excellent performance in handling changes in scale, rotation, and their built-in error detection and correction capabilities. Through testing various tag sizes, we found that 5cm x 5cm tags were optimal for our camera with a resolution of 1920 x 1440 pixels. When considering a user standing 1-2 meters in front of the camera, 5cm x 5cm tag size achieves an ideal balance between accurate tracking and range of motion. This size prevents the tag from being neither too small, which would make recognition challenging, nor too large, causing the recognition target to take up a significant portion of the lens' field of view and increase the chance of it going out of frame.

Key components of the entire tracking system are shown in Figure 11:

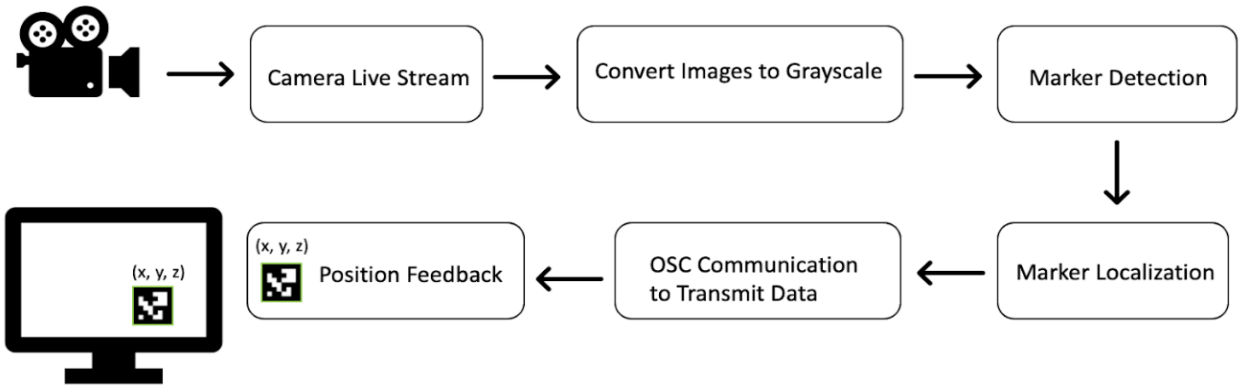


Figure 11: Workflow of the motion tracking system (further explanation in Appendix II)

Painting with Precision: How ArUco Tags Optimized Motion Tracking Performance

ArUco tag-based tracking exhibits high accuracy and robustness in diverse background environments and positioning angles, significantly outperforming finger detection and color marker detection, particularly when users move forwards and backwards.

Finger detection, our initial approach, provides intuitive interaction but faces challenges. Successful implementation requires detecting the full hand skeleton and distinguishing individual fingers. In practice, it may suffer from inaccuracies and occlusion issues, especially when fingers are close together or partially obscured.

Color marker detection is another alternative but is heavily affected by factors such as background environments and lighting conditions. Cameras capturing colours with chromatic aberrations can hinder accurate detection.

In brief, ArUco tag-based tracking has proven to be the most accurate and robust solution among tracking methods, making it the ideal choice for the current GRIPaint version.

However, after the demonstration day, two emergent limitations deserve further discussion and improvements.

1. Extreme tag deflection occurs when users extend their upper limb without twisting their wrist to maintain tag's orientation towards the camera, causing the camera to lose sight of the full tag and interrupt data transmission, resulting in a "lagging" brush position in the user experience. Future improvements could explore alternative tag placement strategies or algorithms capable of detecting multiple tags at different orientations on the drawing handle.
2. Tag reflection under sunlight or other light sources can impact tracking accuracy. Future improvements could Investigate various lighting conditions on tag detection, with potential solutions including specialized lighting setups or algorithms compensating for challenging conditions (further discussed in Appendix III).

2.3.3 System design and integration

Using one main controller to control every part directly is impractical for this project because it will take up a great number of memories, resulting in crashes easily. Hence, peripheral controllers are added for controlling the camera and button. The camera will capture images in real-time and the peripheral controller will calculate the position of the ArUco tag, which is based on the python ArUco detection algorithm. Then the position is sent back to the main controller, and the communication is achieved by the local area network (LAN).

Except for the ArUco tag, there are two buttons and one force sensing resistor (FSR) on the gripper, which are controlled by Arduino [18]. The button status and FSR reading will be controlled by Arduino, and it will integrate this information into a 1×3 matrix. The matrix will be sent back to the main controller, and the communication is achieved by serial communication, where the Universal Asynchronous Receiver/Transmitter (UART) interface [19] is used. The UART interface is a hardware component that allows serial communication to occur, which is natively supported by Arduino.

The main controller will integrate the position and sensor status together and send the information into Unity software. Since the difference between camera resolution and Unity canvas size, a mapping

function is required, then the position of the gripper can be reflected on the canvas in real-time. The control schematic diagram is illustrated in figure 12.

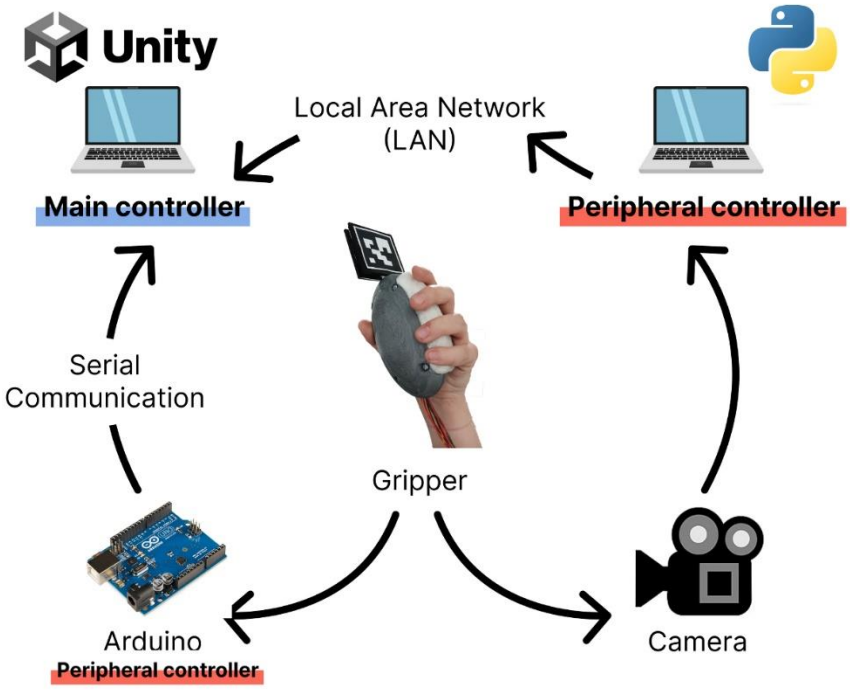


Figure 12: Control Diagram

3. Future work, limitations, and reflections

1. The size of the gripper is not adjustable, and there may be some incompatibilities for users with different hand sizes. In the future, we will design a variety of grippers of different sizes. Measure the user's hand size and select the appropriate gripper before use. Since the gripper contains electronic components, changing different grippers will lead to a repeated connection of the circuit, which is a waste of time. We propose a modular design, which means the internal circuit devices can be easily removed and installed, which is well adapted to grippers of different sizes, making it more convenient for users to change grippers.
2. There is a certain probability that the force sensing resistor (FSR) will be unstable, resulting in some functions in the system failing. There are several possible reasons for failure, one is an unstable circuit connection, since the FSR is very sensitive to small changes in the circuit, any circuit error will be magnified. The second is that it is impossible to ensure that the user applies all the force on the FSR, because the area where the user applies pressure may be staggered from the sensing area of the FSR. In this case, the user may apply a lot of force, but the sensor reading is small, resulting in a bad user experience. One possible solution is designing an additional mechanical structure to guide the user to place the finger completely on the sensing area to minimize force dispersion. Additionally, we will test different types of force sensors and not be limited to FSR. We will use the performance of the original FSR as a benchmark, compare it with the new force sensor, and select the one with better performance to improve the device.

4. Conclusion

In conclusion, GRIPaint design enhances the natural range of the hand, facilitating computer interactions for users with limited dexterity, such as the elderly. By utilizing motion tracking technology and developing user-friendly painting software, we enable users to participate in high-dexterity activities with greater ease. The ergonomic gripper design not only interfaces with the software but also serves as a tool for grip strength training. Through the integration of a peripheral controller and the optimization of hardware and software components, we have created an intuitive user interface tailored to our target demographic. Although certain limitations exist, the scope for future improvements and adaptability positions our design as a significant contribution to the field of accessible technology.

Reference

1. JONATHAN. EDEN, "Principles of movement augmentation," in Human Centred Design of Assistive and Rehabilitation Devices Lecture 2, 27-Jan-2023.
2. C. R. Gale, C. N. Martyn, C. Cooper, and A. A. Sayer, "Grip strength, body composition, and mortality," International journal of epidemiology, vol. 36, no. 1, pp. 228–235, 2007.
3. J. O. Holloszy, "The biology of aging," in Mayo Clinic Proceedings, Elsevier, vol. 75, 2000, S3–S9.
4. M. Azeez, C. Clancy, T. O'Dwyer, C. Lahiff, F. Wilson, and G. Cunnane, "Benefits of exercise in patients with rheumatoid arthritis: A randomized controlled trial of a patient- specific exercise programme," Clinical rheumatology, vol. 39, pp. 1783–1792, 2020.
5. G. Cohen, "Creativity and aging study: Initial results for chorale," Retrieved November, vol. 27, p. 2006, 2006.
6. H. Vaartio-Rajalin, R. Santamaïki-Fischer, P. Jokisalo, and L. Fagerströ m, "Art making and expressive art therapy in adult health and nursing care: A scoping review," Interna- tional journal of nursing sciences, vol. 8, no. 1, pp. 102–119, 2021.
7. Gripable, Gripable, https://gripable.co/, Accessed: 2023-03-30, 2023
8. M. Mace, S. A. Mutalib, M. Ogrinc, N. Goldsmith, and E. Burdet, "Gripable: An accurate, sensitive and robust digital device for measuring grip strength," Journal of Rehabilitation and Assistive Technologies Engineering, vol. 9, p. 20 556 683 221 078 455, 2022.
9. R. Jaber, D. J. Hewson, and J. Duchˆene, "Design and validation of the grip-ball for mea- surement of hand grip strength," Medical engineering & physics, vol. 34, no. 9, pp. 1356– 1361, 2012.
10. E. Fess, "Clinical assessment recommendations," American society of hand therapists, pp. 6–8, 1981.
11. Arduino, analogRead(), https://www.arduino.cc/reference/en/language/functions/analog-io/analogread/, Accessed: 2023-03-30, 2023
12. D. Barnett, D. Groth, and J. McBee, Cabling: the complete guide to network wiring. John Wiley & Sons, 2006
13. R. M. Ryan and E. L. Deci, Self-determination theory: Basic psychological needs in mo- tivation, development, and wellness. Guilford Publications, 2017.

14. D. Peters, R. A. Calvo, and R. M. Ryan, “Designing for motivation, engagement and wellbeing in digital experience,” Frontiers in psychology, p. 797, 2018.
15. G. Scavone, Open Sound Control (OSC), https://www.music.mcgill.ca/~gary/306/week9/osc.html, Accessed: 2023-03-29, 2020.
16. V. Sigalkin, extOSC - Open Sound Control Protocol for Unity, https : // github . com / Iam1337/extOSC, Accessed: 2023-03-29, 2022.
17. S. Garrido-Jurado, R. Mu˜noz-Salinas, F. J. Madrid-Cuevas, and M. J. Marın-Jim´enez, “Automatic generation and detection of highly reliable fiducial markers under occlusion,” Pattern Recognition, vol. 47, no. 6, pp. 2280–2292, 201
18. Arduino, Arduino Uno Rev3, https://store.arduino.cc/products/arduino-uno-rev3, Accessed: 2023-03-29, 2023
19. U. Nanda and S. K. Pattnaik, “Universal asynchronous receiver and transmitter (uart),” in 2016 3rd international conference on advanced computing and communication systems (ICACCS), IEEE, vol. 1, 2016, pp. 1–5.
20. Bradski, G. (2000). The OpenCV Library. Dr. Dobb. Journal of Software Tools.
21. Elgendy, Mostafa, et al. ‘Identification of Markers in Challenging Conditions for People with Visual Impairment Using Convolutional Neural Network’. Applied Sciences, vol. 9, no. 23, Nov. 2019, p. 5110. https://doi.org/10.3390/app9235110.

Appendix I – OSC unity implementation

The core implementation methods are attached here. Please note that OSC requires clients and servers to be connected under the same network. The IP address needs to be updated every time to establish the correct connection.

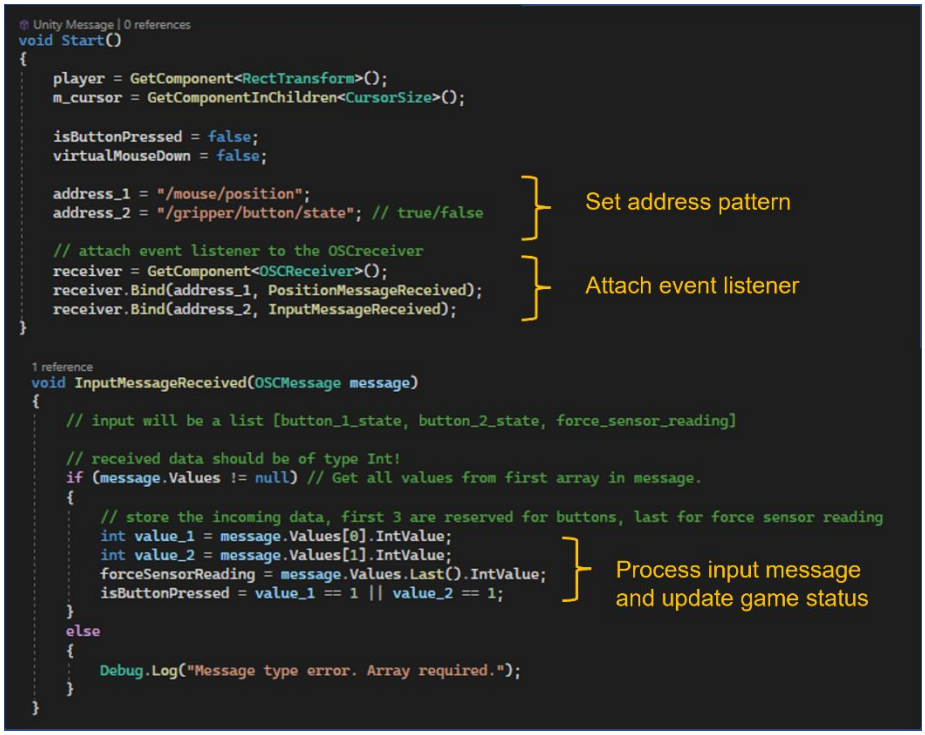


Figure I.I Implementation of Open Sound Control using extOSC package [16]

Appendix II – Explanation on Tracking Algorithm

This document presents a detailed explanation of the tracking algorithm used in our system, which involves ArUco tag generation, video stream setup, ArUco tag detection, and Open Sound Control communication. By understanding each component's functionality, we can effectively optimize and adapt the algorithm to enhance tracking performance and broaden its applicability.

ArUco Tag Generation

```python
def create_aruco_tag(output_path="ARUCO_ORIGINAL.png", id_num=0, tag_type="DICT_ARUCO_ORIGINAL"):
    """
    This function creates an ArUco tag using the OpenCV librarqy.

    :param output_path: The path and filename of the output file. Default is 'ARUCO_ORIGINAL.png'.
    :param id_num: The ID number of the tag. Default is 0.
    :param tag_type: The type of ArUco tag to create. Default is 'DICT_ARUCO_ORIGINAL'.
    :return: None
    """
    # Define names of each possible ArUco tag OpenCV supports
    ARUCO_DICT = {
        "DICT_4X4_50": cv2.aruco.DICT_4X4_50,
        "DICT_4X4_100": cv2.aruco.DICT_4X4_100,
        "DICT_4X4_250": cv2.aruco.DICT_4X4_250,
        "DICT_4X4_1000": cv2.aruco.DICT_4X4_1000,
        "DICT_5X5_50": cv2.aruco.DICT_5X5_50,
        "DICT_5X5_100": cv2.aruco.DICT_5X5_100,
        "DICT_5X5_250": cv2.aruco.DICT_5X5_250,
        "DICT_5X5_1000": cv2.aruco.DICT_5X5_1000,
        "DICT_6X6_50": cv2.aruco.DICT_6X6_50,
        "DICT_6X6_100": cv2.aruco.DICT_6X6_100,
        "DICT_6X6_250": cv2.aruco.DICT_6X6_250,
        "DICT_6X6_1000": cv2.aruco.DICT_6X6_1000,
        "DICT_7X7_50": cv2.aruco.DICT_7X7_50,
        "DICT_7X7_100": cv2.aruco.DICT_7X7_100,
        "DICT_7X7_250": cv2.aruco.DICT_7X7_250,
        "DICT_7X7_1000": cv2.aruco.DICT_7X7_1000,
        "DICT_ARUCO_ORIGINAL": cv2.aruco.DICT_ARUCO_ORIGINAL,
    }

    # Verify that the supplied ArUCo tag exists and is supported by OpenCV
    if ARUCO_DICT.get(tag_type, None) is None:
        print("[INFO] ArUCo tag of '{}' is not supported".format(tag_type))
        sys.exit(0)

    # Load the ArUCo dictionary
    aruco_dict = cv2.aruco.Dictionary_get(ARUCO_DICT[tag_type])

    # Allocate memory for the output ArUCo tag and then draw the ArUCo tag on the output image
    print("[INFO] generating ArUCo tag type '{}' with ID '{}'".format(tag_type, id_num))
    tag = np.zeros((300, 300, 1), dtype="uint8")
    cv2.aruco.drawMarker(aruco_dict, id_num, 300, tag, 1)

    # Write the generated ArUCo tag to disk
    cv2.imwrite(output_path, tag)

    # Display the generated ArUCo tag
    cv2.imshow("Generated ArUco Tag", tag)
    cv2.waitKey(0)
    cv2.destroyAllWindows()

create_aruco_tag(output_path="my_aruco_tag.png", id_num=0, tag_type="DICT_6X6_250")
```

```
[INFO] generating ArUCo tag type 'DICT_6X6_250' with ID '0'
```

Figure II.I ArUco tag generation code

ArUco Tag Generation is carried out using the OpenCV library [20]. The aruco module provides convenient functions for generating ArUco Tags. We used “DICT_6X6_250” and used the code below to generate 6 by 6 pixel markers.

Open Sound Control Communication

```python
# OSC communication parameters
address = "/mouse/position"
address_2 = "/gripper/button/state"

# Print OSC server information
print("The ip of the OSC server:" + ip)
print("The port the OSC server is listening on:" + ip)

# Create OSC client
client = udp_client.SimpleUDPClient(ip, port)
```

Figure II.II Open Sound Control communication code

Open Sound Control [15,16] is a handy tool for sending the position and size of detected ArUco markers to a Unity application. By using the pythonosc library, the code sets up an OSC client that shares the marker's location and dimensions with a specific IP address and port (where Unity is running). This data is then employed to guide the movements of an on-screen painter in the virtual world.

Aruco Tag Detection & Video Stream Setup

```python
# Main loop
while True:
    # Grab and resize the frame
    frame = vs.read()
    frame = imutils.resize(frame, width=fx + w)

    # Detect ArUco markers
    (corners, ids, rejected) = cv2.aruco.detectMarkers(frame, arucoDict, parameters=arucoParams)

    # Process detected markers
    if len(corners) > 0:
        # Flatten the ArUco IDs list
        ids = ids.flatten()

        # Loop over detected corners
        for (markerCorner, markerID) in zip(corners, ids):
            # Extract and process marker corners
            corners = markerCorner.reshape((4, 2))
            (topLeft, topRight, bottomRight, bottomLeft) = corners

            # Convert each of the (x, y)-coordinate pairs to integers
            topRight = (int(topRight[0]), int(topRight[1]))
            bottomRight = (int(bottomRight[0]), int(bottomRight[1]))
            bottomLeft = (int(bottomLeft[0]), int(bottomLeft[1]))
            topLeft = (int(topLeft[0]), int(topLeft[1]))

            # Draw the bounding box of the ArUCo detection
            cv2.line(frame, topLeft, topRight, (0, 255, 0), 2)
            cv2.line(frame, topRight, bottomRight, (0, 255, 0), 2)
            cv2.line(frame, bottomRight, bottomLeft, (0, 255, 0), 2)
            cv2.line(frame, bottomLeft, topLeft, (0, 255, 0), 2)


            # Compute and draw the center (x, y)-coordinates
            k=1.3
            cX = int((topLeft[0] + bottomRight[0]) / 2.0)
            cY = int((topLeft[1] + bottomRight[1]) / 2.0)
            size=np.sqrt((topLeft[0] - topRight[0])*(topLeft[0] - topRight[0])+(topLeft[1] - topRight[1])*(topL

            cv2.circle(frame, (cX, cY), 4, (0, 0, 255), -1)


            # Send OSC message with marker information
            mouse_xys = [fx - float(cX), fy - float(cY * k), float(abs(size))]
            client.send_message(address, mouse_xys)
            print([fx - int(cX), fy - int(cY * k), int(abs(size))])

            # Draw the ArUco marker ID on the frame
            cv2.putText(frame, str(markerID),
                (topLeft[0], topLeft[1] - 15),
                cv2.FONT_HERSHEY_SIMPLEX,
                0.5, (0, 255, 0), 2)

    # Show the output frame
    cv2.imshow("Frame", frame)
    key = cv2.waitKey(1) & 0xFF

    # Break the loop if 'q' key is pressed
    if key == ord("q"):
        break
```

Figure II.III Aruco tag detection code

The code initiates a video stream by using the WebcamVideoStream class from the imutils.video package, capturing frames from an attached webcam. These frames are then resized to a specified width and analyzed to identify any ArUco markers in the image. OpenCV's aruco module offers built-in support for detecting ArUco markers. The detection process consists of the following stages:

1. Grayscale transformation: The input image is converted to grayscale, which simplifies computations and boosts detection effectiveness.
2. Adaptive thresholding: The grayscale image is binarized using adaptive thresholding, distinguishing the marker from its background.
3. Contour extraction: Contours are extracted from the binary image, and potential ArUco tag candidates are spotted based on their geometric traits, such as having four corners and a convex shape.
4. Candidate refinement: Marker candidates are further refined by looking for a binary pattern in the marker's inner region.
5. Decoding: Each detected marker's unique ID is obtained by decoding its binary pattern.

Finally, the detection method pinpoints the marker corners and extracts their coordinates, which are then used to calculate the marker's central position and size.

Anticipated Outcomes

Running the algorithm will automatically open the camera, and the anticipated log output should appear as follows:

```
The ip of the OSC server:146.169.221.111
The port the OSC server is listening on:146.169.221.111
[INFO] detecting 'DICT_6X6_250' tags...
[INFO] starting video stream...
[709, 589, 256]
[712, 588, 257]
[712, 589, 259]
[713, 588, 258]
[716, 588, 256]
[715, 588, 258]
[716, 588, 260]
[718, 586, 258]
[716, 589, 260]
[716, 589, 258]
[716, 589, 260]
```

Figure II.IV Log output of the tracking system

Appendix III – Discussion on Future Improvements for Marker Detection in Challenging Conditions

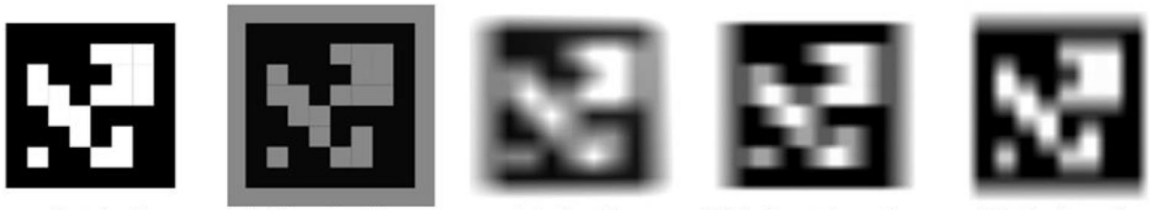
This discussion focuses on addressing the challenges associated with lighting variations, marker distortion, and blurring effects in detection. We also consider potential improvements using transfer learning to enhance the tracking algorithm's performance under these conditions.

Current tracking algorithms are found to be difficult under the following conditions:

1. Lighting variations: Sunlight or other light sources can cause reflections on the tags. An improved tracking algorithm should account for such variations and adapt to different lighting conditions.
2. Marker Distortion: When users extend their upper limb without twisting their waist, the marker's orientation may change, resulting in partial visibility or distortion of the tag.
3. Blurring effects: Rapid limb movements can cause blur in video frames, leading to inaccurate tag detection. Improved method could be using camera with a higher framerate (current framerate limited to 30 frames/second)

Transfer Learning for Enhanced Detection

Transfer learning offers a potential solution to improve detection performance under challenging conditions. By generating transformed samples of the original image with various effects, such as lighting changes and motion blur, the model can be trained to recognize and locate markers in diverse situations [21]. By retaining the existing model weights, the training continues with the transformed samples, enabling the model to adapt to the challenging scenarios it has already encountered.



(I) (II) (III) (IV) (V)

Figure III (I) Original tag (II) Lighting variation (III) Motion blur (IV) Horizontal motion blur (V) Vertical motion blur

Smooth Out Tracking with Interpolation

In some cases, the tracking system may still experience target loss, leading to temporary freezing as it stops updating without receiving position data. To enhance system performance, implementing interpolation techniques during these target loss stages could be an alternative solution. This will provide users with a smoother, more continuous tracking experience by compensating for momentary disruptions in data transmission.

Appendix IV – Arduino Recordings

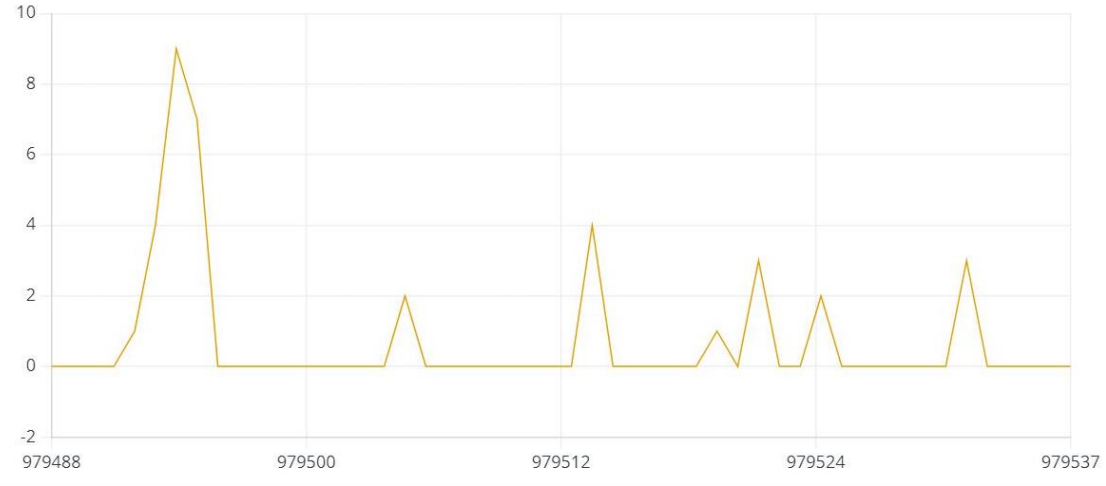


Figure IV.I 10K Resistor Value, No delay, Suggests non-linear operation of FSR

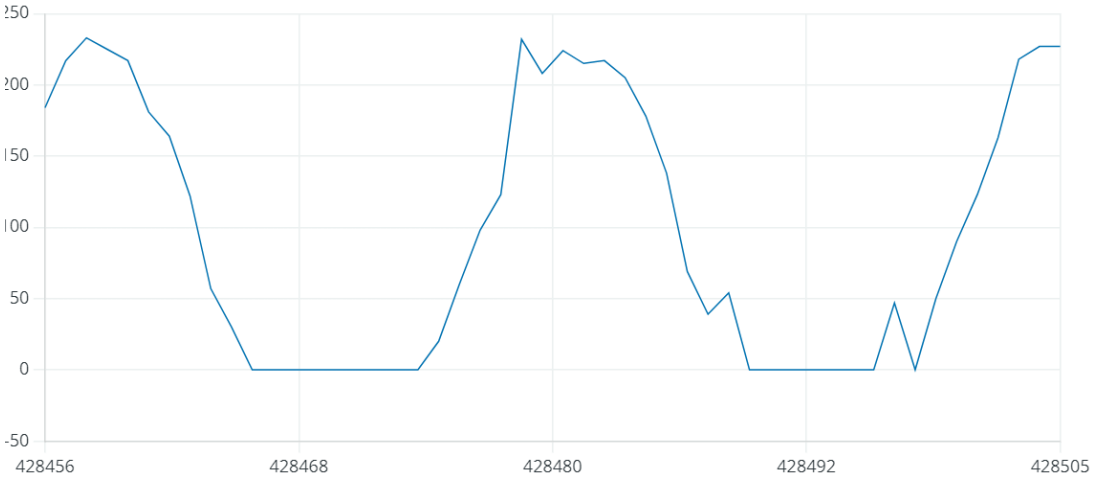


Figure IV.II 1MΩ Resistor, No delay, Breadboard

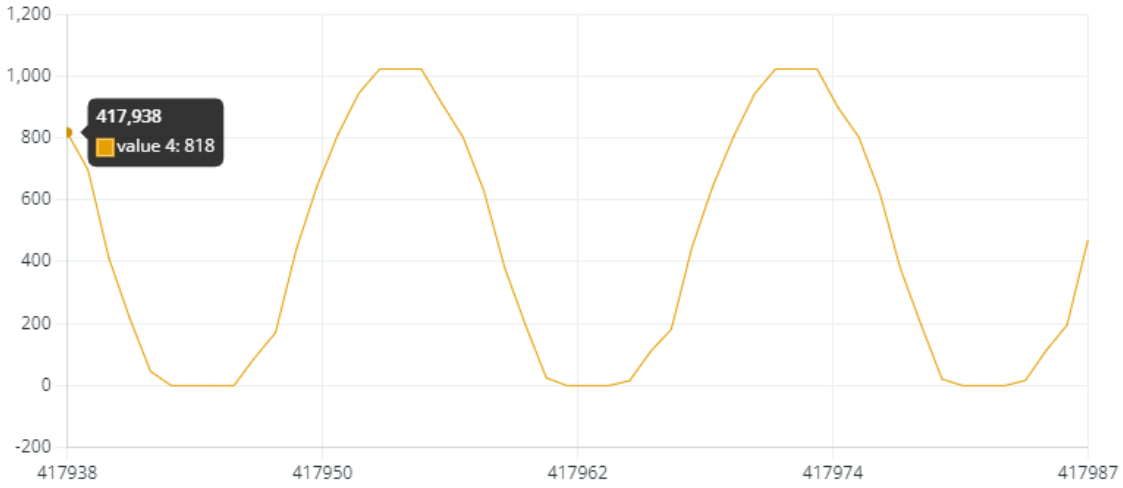


Figure IV. III 1MΩ Resistor, No delay, Stripboard

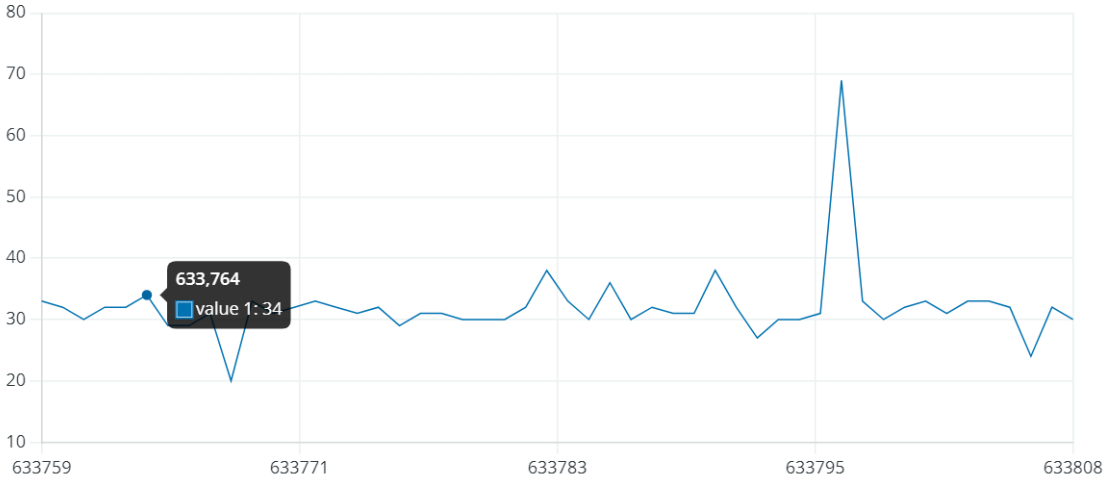


Figure IV. IV 1MΩ Resistor, No delay, 10µF Capacitor

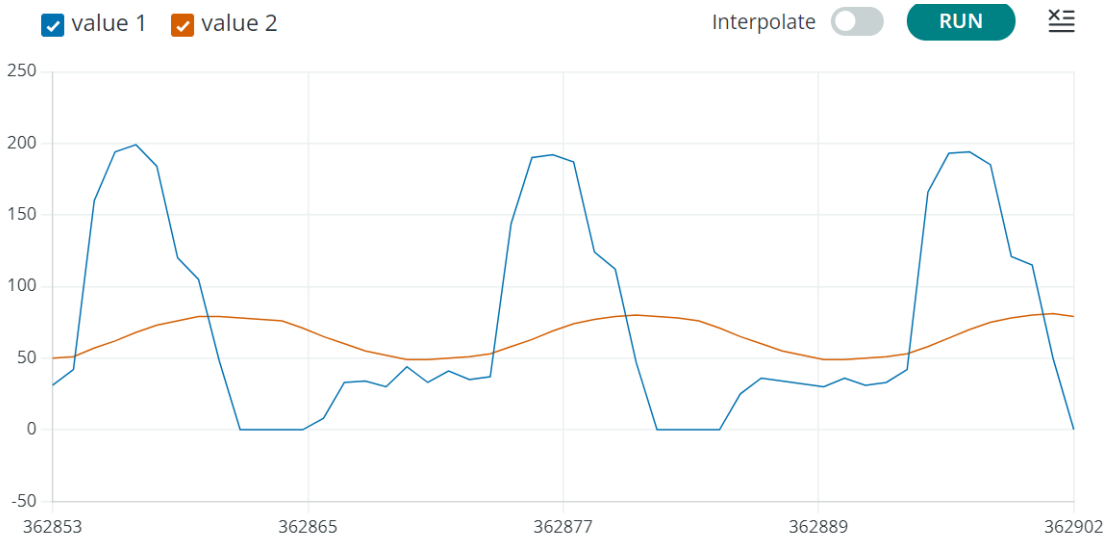


Figure IV. V 1MΩ Resistor, No delay, First order Moving average filter (Orange)

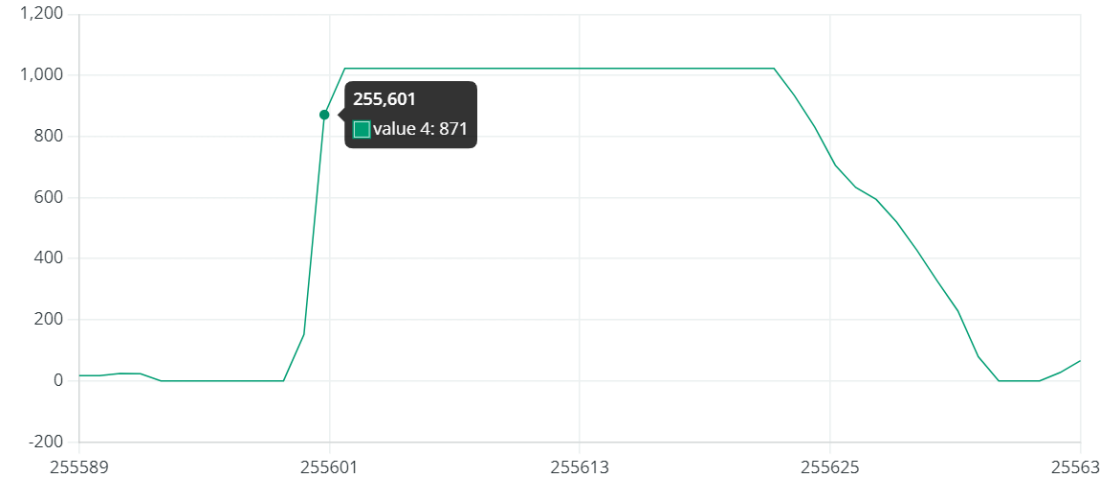


Figure IV. VI 1MΩ Resistor, 80ms delay