Hand Tracking for Air Drawing: A Gesture-Based Approach to Digital Art

Himanshu*, Kaustubh Goyal*, Nikita Malik*

Abstract: ***Traditional drawing and painting applications often require physical tools such as styluses, tablets, or mouse devices, which can be limiting or inconvenient. This creates a need for an innovative approach to enable intuitive and hands-free drawing using natural gestures. The work presented in this paper is an interactive air-drawing application utilizing computer vision and hand-tracking technology. Using the OpenCV and MediaPipe libraries, the program captures video from a webcam, allowing users to draw on a virtual canvas by moving their hands in the air. Different coloured drawing options (blue, green, red, and yellow) are available, and users can switch colours or clear the canvas by moving their hand to specific areas of the screen. The paper aims to develop a real-time application that allows users to draw and paint virtually using hand gestures detected through a webcam, providing a hands-free and user-friendly creative experience. It utilizes computer vision and machine learning libraries, specifically OpenCV and MediaPipe. A webcam captures video frames, and hand landmarks are detected using the MediaPipe Hands solution.***

Keywords: MediaPipe Hands, OpenCV, Virtual drawing, Hands-free interaction, Gesture-based painting

1. INTRODUCTION

Sometimes one draws their creative thoughts simply by deferring their fingers in air. Computer vision is an interdisciplinary field that provides provisions on how PCs (personal computers) can be made to attain a significant level of understanding by utilizing various computerized methods. In this study, a Virtual Reality (VR) Air Canvas is fabricated that can draw anything on it simply by capturing the movement of fingers with a webcam. A few examples of hand gestures and tracking are shown in figure 1 [1].

1.1 Purpose

The main objective of this paper is to leverage computer vision to recognize hand gestures and translate them into real-time actions, such as drawing on a digital canvas or interacting with a virtual interface. The purpose can vary depending on the specific application, but common objectives might include:

- **Creating a hands-free drawing interface:** Enabling users to create digital artwork, sketches, or diagrams using only their hand movements, without the need for a traditional input device like a mouse or touchscreen.
- **Interactive user experience:** Enhancing the interaction between humans and machines, making it more intuitive by allowing the user to control a system with natural movements (e.g., drawing or manipulating objects in 3D space).

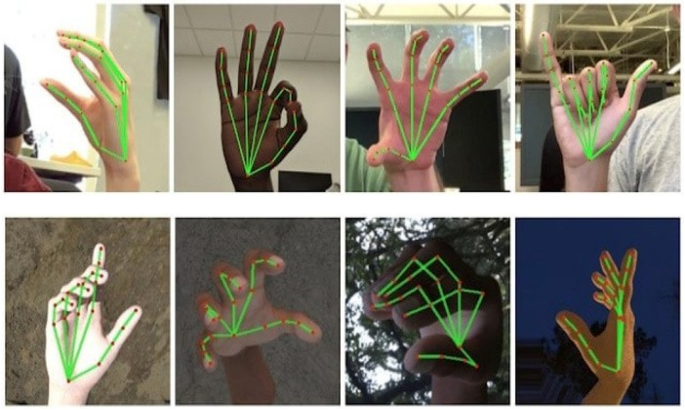

Fig. 1. Sample images of Hand Tracking

1.2 Computer Vision

Computer vision (CV) is a field of artificial intelligence (AI) which focuses on enabling machines to interpret and understand visual information from the world, just as humans do, as discussed in Section 4. In the context of this study, computer vision was used to analyze the video feed (usually from a camera) and detect hand movements or gestures, as shown in figure 1.

1.3 Drawing Applications

Drawing applications are software tools that allow users to create digital art or diagrams using various input methods, such as stylus, mouse, or touch gestures, as discussed in Section 4.4. In this study, a drawing application refers to a platform in which hand gestures are translated into visible output, such as lines, shapes, or color changes.

1.4 Entertainment or Educational Tools

Developing innovative software for creative activities, such as virtual painting or interactive learning tools, where users can draw or gesture to trigger specific actions or responses.

Therefore, this study aims to use computer vision techniques to recognize specific hand gestures, which are then interpreted to trigger drawing actions (such as creating lines, shapes, or colors) or other interactions in a computer-based environment.

**Department of Computer Applications, Maharaja Surajmal Institute, New Delhi*
*Corresponding author: *nikitamalik@msijanakpuri.com*

1.5 Key Components of Hand Gesture Recognition

- Hand Tracking- Determining the precise position and movement of the hand over time, often through detecting key points on the hand (such as the fingertips, knuckles, and wrist), as shown in section 4.1. Technologies like MediaPipe, a Google framework, or OpenCV (with models trained for hand detection) are popular for hand trackings.
- Feature Extraction- Once the hand is detected; certain features like angles between fingers, relative positions of fingers, or overall hand shape are extracted to identify the gesture, as shown in section 4.1.
- Gesture Classification- Machine learning (ML) algorithms like SVM (Support Vector Machines), Random Forests, or Deep Learning (DL) models can be used to classify the gesture. This involves training the system on various hand gestures such as open hand, fist, pointing, waving, etc.
- Real-time Processing- When it comes to data entry, real-time processing means that data can be processed and checked quickly and without delay. This means that once a product has been created or acquired, it can be processed, modified and made available for further use or research.

2. LITERATURE REVIEW

The development of gesture-based interactive applications has gained momentum with the advancements in computer vision and machine learning. Virtual drawing applications offer a way for users to create art or annotate visuals in real time, often without traditional input devices, such as a mouse or stylus. Instead, these applications may leverage hand-tracking or gesture-recognition technologies, creating immersive user experiences in digital art and design.

2.1 Robust Hand Recognition with Kinect Sensor

The system described in [3] utilized the Kinect sensor's depth and color data to identify hand shapes. Despite the implementation of a Kinect sensor, gesture recognition remains a significant challenge. The Kinect sensor's resolution was 640×480, which is adequate for monitoring large objects such as the human body. However, tracking smaller objects, particularly fingers, proves to be problematic. The complexity of gesture recognition persists even with the capabilities afforded by the Kinect sensor.

2.2 LED fitted finger movements

In [4], researchers proposed a technique that utilizes a finger-mounted LED and a web camera to track finger movements. The system compares the drawn characters to a database and identifies the closest matching letter. This method necessitates a red LED light source affixed to the user's finger. Moreover, it presupposes that no other red objects are present within the camera's field of view apart from the LED light.

2.3 Augmented Desk Interface

A novel approach for interactive computing was introduced in [5], presenting an enhanced segmented desk interface. This system incorporates a video projector and a charge-coupled device (CCD) camera, enabling users to control desktop applications through fingertip interaction. The configuration utilizes a dual-hand operation method, wherein the left hand navigates radial menus and the right hand selects objects for manipulation. An infrared camera facilitates this functionality. To address the computational demands of fingertip detection, the system implements predefined search windows for fingertip localization.

2.4 Applications of Virtual Drawing Technologies

Beyond recreational use, gesture-based virtual drawing has applications in education, virtual meetings, and remote learning, where such tools are used to annotate or illustrate concepts in real-time.

This technology is particularly valuable in fields like telemedicine, where practitioners can illustrate diagnoses or treatments over video streams.

2.5 Evolution of Virtual Drawing and Painting Applications

The transition from traditional drawing tools to virtual applications has been driven by the need for more flexible, user-friendly, and interactive digital solutions. Early software in this domain primarily relied on mouse or stylus input.

However, with advancements in computer vision and the rise of touchless interaction, systems now leverage cameras and algorithms to interpret user gestures as drawing inputs.

2.6 Computer Vision in Gesture Recognition

Computer vision plays a central role in the development of gesture-based drawing applications. Frameworks such as OpenCV and MediaPipe are key players, providing powerful tools for hand movement detection and tracking. OpenCV supports various image processing features, such as edge detection, segmentation, and morphological transformations, aiding in the accurate detection of hand contours and other gestures. On the other hand, MediaPipe makes it a piece of cake with the help of machine learning models that are trained to recognize essential hand landmarks for more accurate

Vision-based systems have been shown to enhance engagement and lower the interaction barrier, as they offer a more realistic interface that is more in tune with real-world hand movements.

2.7 Advances in Colour and Brush Selection Mechanisms

Much of the experience of digital drawing is related to the responsiveness of the application and the ease with which you select a color and pick a brush. Moreover, it was shown in previous research that the more options available to control

motion by gestures, the more creative and fluid the application becomes [12]

Gesture-controlled mechanisms for color and brush selection allow artists to focus on their creations rather than manually setting up workspaces.

Impulse assignment- The gesture of drawing itself is a common one, and many gesture-based drawing programs assign various gestures such as switching colors or changing brush sizes, which would, of course, also allow users to change their tools during a session. A responsive listening and amplifying mechanism for movement makes it easier for users to draw through the lines faster which ultimately speed up draws and leads to a more agile drawing interaction improving both user experience and output.

2.8 Use of Deque Data Structures for Efficient Point Storage

In virtual drawing programs, tracking and storing multiple points during a drawing session require an efficient data structure.

The use of deque (double-ended queue) data structures in Python for storing drawing points optimizes the process because deques offer fast append and pop operations on both ends.

This efficiency is critical in real-time applications where delays in point storage can lead to interruptions or stuttering in the drawing line, diminishing user experience.

2.9 Applications and Impacts of Gesture-Based Drawing Programs

Gesture-based virtual drawing applications have significant potential in various fields [8].

- **IT and Remote Work:** Virtual whiteboards that allow you to draw without a pen just by using your hands have been integrated into online learning platforms to create an interactive feel. Studies have shown that these tools support both instructors and students by enabling the real-time annotation and visualization of complex topics.
- **Healthcare:** In the field of telemedicine, gesture-based drawing tools that allow a user to draw over streaming videos have been successfully used by professionals to demonstrate how to perform procedures or annotate regions of interest on digital medical images.
- **Art Therapy and Rehabilitation:** Digital art platforms that use gestures have been explored as therapeutic tools in settings where traditional art tools may be inaccessible. The engaging, hands-free nature of these applications has made them particularly beneficial for individuals undergoing motor-skill rehabilitation.
- **Future Directions in Gesture-Based Virtual Drawing Applications:** Gesture-based drawing is an area of research that looks at making programs more precise and sensitive to user input.

Work remains to be done, in particular, to increase the stability and robustness of hand tracking under different lighting conditions, reduce the latency of gesture detection, and to include more advanced drawing features such as multi-touch support and pressure sensitivity emulation.

Another future advancement to enhance measures that increase user satisfaction is the suggestion of AI-driven personalization, in which the application adapts to a specific drawing style and user preferences.

3. TECHNOLOGIES USED

Air painting is a novel interaction method that allows users to draw or paint objects in a virtual environment using hand gestures. This approach leverages advancements in computer vision and machine learning to provide an intuitive touchless interface. It has applications in art, education, virtual reality, and interactive systems.

3.1 Hardware used

- A webcam is a type of video camera that may gradually send or receive images or videos to or through a PC network, such as the Internet.
- Images produced by a PC or other electronic device are displayed on a visual display unit (VDU). VDU is sometimes used synonymously with "screen," however it can also refer to another type of display, such as a computerised projector.

3.2 Key Technologies used

The program uses:

- MediaPipe Hands: Detection and tracking of hand landmarks with high accuracy.
- OpenCV: Real-time video capture and drawing functionalities.
- Deque Data Structure: The sequence of points for each colour trial is stored.
- Gesture Recognition: Hand gestures form the foundation of this study. The fingertip position was identified and tracked to enable air-drawing. Gestures such as moving fingers or selecting virtual buttons are interpreted to perform specific tasks, such as choosing colors or clearing the canvas [11].
- Interactive graphics allow users to draw different colors and styles in a simulated environment. The drawing application uses a graphical interface to virtually replicate the traditional painting experience [11].

4. WORKING AND IMPLEMENTATION

The code implements an air drawing program that uses hand

gestures detected through a camera feed to allow users to draw various colors. This modular breakdown covers the entire code structure, including variable initialization, frame processing, gesture recognition, and drawing functions, as shown in figure 2 and figure 3.

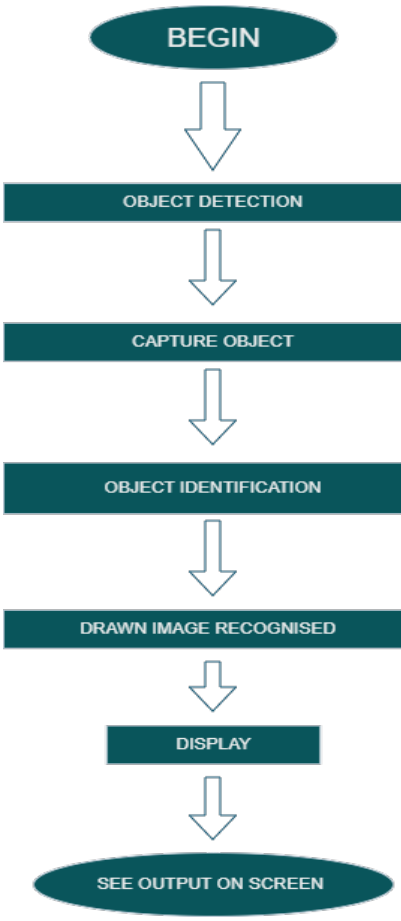

Fig. 2. Flow diagram

4.1 Initialization of Variables and Colors

This section covers the setup of essential components like deques for drawing points, colour indexes for selecting colours, and the paint window.

- **Deques for Point Tracking:** Four deques (bpoints, gpoints, rpoints, and ypoints) are created for each color: blue, green, red, and yellow. Each deque holds the points drawn with the selected color with a maximum length of 1024 points, ensuring smooth real-time drawing without overwhelming memory. Each deque is a list of points (coordinates) drawn by the program, allowing separate tracking for each color.
- **Index Tracking for Colors:** Variables such as blue_index, green_index, red_index, and yellow_index track the current drawing point within each color deque. These indexes ensure that points are appended correctly, as new lines are drawn in each color.
- **Color Definition and Color Index:** A list of colors [(255, 0, 0), (0, 255, 0), (0, 0, 255), (0, 255, 255)] corresponding to blue, green, red, and yellow was defined. A color index variable is used to switch between these colors.
- **Paint Window Setup:** The paint window (paintWindow) was initialized as a blank white canvas, with buttons created using rectangles in OpenCV.

The "CLEAR" button and each color button are displayed at the top of the paint window. The buttons were visually distinguished by drawing colored rectangles and adding labels, allowing the user to interact with them using gestures.

4.2 Looping and Frame Processing

This section describes the program's main loop, which continuously captures video frames and processes each one to detect hand landmarks and draw actions based on gestures as shown in figure 2.

- **Frame Capture:** The main loop captures frames from the camera using OpenCV. Each frame is preprocessed and encoded before it hits MediaPipe. This technology automatically tracks different body parts with fingers and does not distort them.
- **MediaPipe Hand Landmark Detection:** MediaPipe locates hand landmarks in each frame, and the tips of the fingers are of particular interest. These provide the (x, y) coordinates to obtain the hand position to create the drawing and button presses, as shown in figure 3.
- **Drawing Actions Based on Frame Processing:** For every recognized hand landmark, the program checks whether the user is to paint the window or through the buttons. Points are added to the selected colour deque relative to where it is placed in the hand.

4.3 Gesture Recognition Logic

In this section, the code logic that interprets specific gestures (like clearing the screen or changing colours) is explained in detail as shown in figure 3.

- **Color Selection:** Based on the location of the finger, if the program identifies that it is over a color button, then the user's finger is in. (clientX, clientY) this.colorIndex = colorIndex.tolist() that sets specific coordinates (e.g., on x, y coordinates); it changes the color index to the selected color. This enables users to vary the colors on the fly during sketching. With such information, you will constantly alter your color as you draw by moving your hand over different buttons.
- **Clear Screen Gesture:** When program detects a gesture above the "CLEAR" button, all deques are reset; thus, the paint window is cleared. As a result of this feature, there is no overlapping of old drawings with new inputs and users can start drawing absolutely touchless.

In addition, it makes the system more reliable over time. It can be rewritten with a slightly different style: fresh without restarting program.

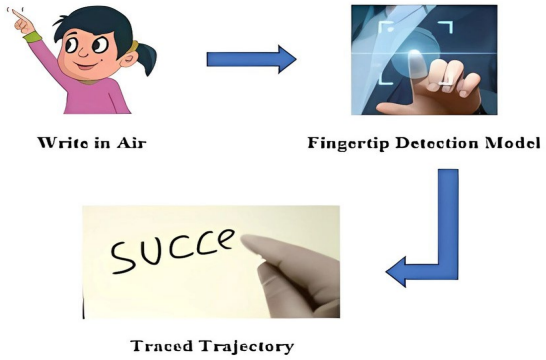

Fig. 3. Workflow of Air Canvas

4.4 Draw Functionality

This section details how the program applies colors based on hand gestures and displays the drawings in real time, as shown in figure 4.

- **Point Collection for Drawing:** By tracking the user's gestures from frame to frame, new points are added to the color deque to correspond to the selected color.
- **Dynamic Line Drawing:** OpenCV's line function draws lines between consecutive points in each color deque. By iterating through each deque (for each color), the program can render all drawn points, preserving the user's drawing as they switch colors. The program uses a loop to check and draw points in each color deque, creating a seamless multicolor drawing experience.
- **Color Application:** This value is used in combination with the color index to set the color of the lines drawn into the paint window. And by changing the colorIndex each time the user clicks a button available in the list of colours, the selected colour is taken for drawing new lines.

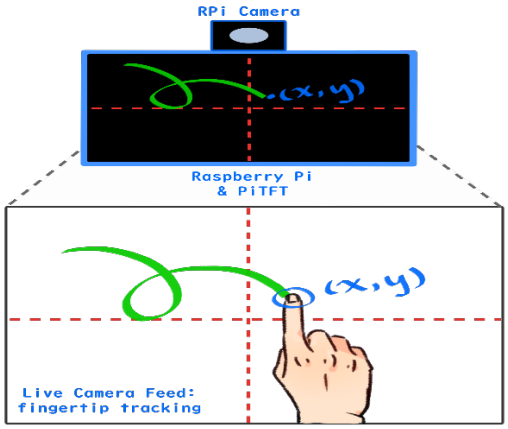

Fig. 4. Training the finger recognition model

5. RESULT

After considerable proof reading and editing, the content scraped through Air Canvas was accurate and good quality. Notably, the content also converted successfully into editable text thanks to OCR tools [9].

The air-drawing application demonstrated high accuracy in recognizing gestures and performed well under various lighting conditions and hand orientations. The program achieved a stable frame rate of 25 to 30 FPS on standard hardware, with minimal latency in gesture detection and colour changes. Resource consumption was moderate, ensuring smooth performance even on average computing systems.

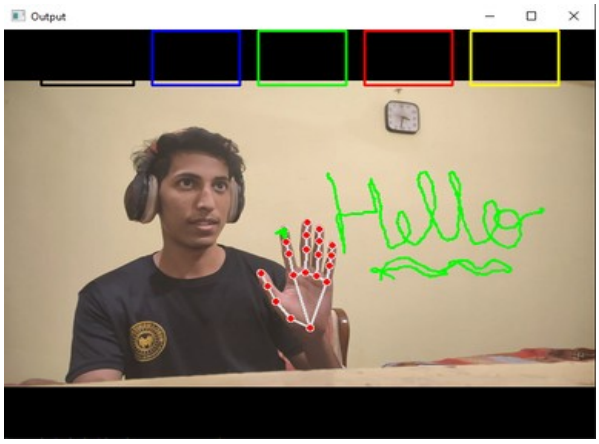

Fig. 5. Interface and drawing window

Figure 5 and 6 present the results of this work. OpenCV module has been used to run the code. The code is executed in Anaconda, Jupiter Notebook, where the necessary packages are first installed and then the code is run for output. Here, two windows are displayed- one window is the camera page where the user's pen will be tracked using coordinates which are specified in the code and, the second window is for drawing the recorded coordinates and components like clear or change the colour can be used.

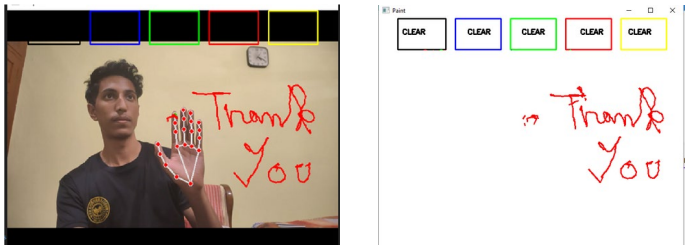

Fig. 6. (a) Drawing window (b) Paint window

6. FINDINGS

- **Hand Tracking:** The program utilizes the MediaPipe library to detect and track hand landmarks in real-time through a webcam feed. It specifically tracks the positions of the index finger (forefinger) and thumb [10].

- **Virtual Drawing:** Users can draw on a virtual canvas by moving their finger in the air. The drawing is facilitated by identifying the position of the index finger, which acts as a virtual brush [10].
- **Color Selection:** The application allows users to select different drawing colours (Blue, Green, Red, Yellow) by hovering their index finger over specific areas on the screen [10].
- **Clear Canvas:** A "CLEAR" button is provided, which can be activated by hovering over it, to reset the canvas [10].
- **Real-Time Feedback:** The program displays the drawing in real-time on two windows: One shows the live video feed with drawing elements. The other shows only the virtual canvas.
- **Drawing Persistence:** Different colours are drawn on separate layers (queues) to maintain clarity and avoid mixing during drawing [10].
- **Intuitive Gestures:** The Intuitive Gesture has been developed in answer to the very real need for a more expressive means of communication, for all facets of our lives, to penetrate beyond predictable patterns of communication that, in large part, quite frankly aren't working [10].
- **Robustness:** It is the ability of a system to cope with erroneous input and errors during execution.

7. CONCLUSION

The Air Canvas project was planned and tested. All materials and application processes were designed by integration. The existence of each module was considered and carefully structured to achieve the optimum performance. Second, we used the traditional programming language Python and NumPy Python library, which works with the help of presentation and innovation, and the project was completed successfully. Using Air Canvas, we developed a program that does not require drawing and recognizes the user's fingerprint using OpenCv. Beautiful lines can be drawn wherever the customers want. There is no need to search the PC while the program is running.

8. FUTURE SCOPE

If we had more time to work on this project, we could have improved the hands-on experience, explored our VR Air Canvas goals for the first time, and tried to better understand the basic modes. There are many different methods for contour analysis, however, in this particular algorithm, it is necessary to look at the histogram of colors used to create the contour in question. Different methods can also be used. Allowing users to save their final work or animate their drawing process could be a unique feature similar to real creative software. Finally, we can achieve meaningfulness by analyzing how much work has been done using the data.

REFERENCES

[1] Shinde, H. A., Jagtap, S. M., Kalpund, A. A., More, P. B., & Parkale, A. A. (2021). Air Canvas: Draw in Air Using AI. *International Journal of Scientific Research & Engineering Trends. 7*(6). 3553-3554.

[2] Saoji, S. U., Dua, N., Choudhary, A. K., & Phogat, B. (2021). Air canvas application using Opencv and numpy in python. *IRJET*, *8*(08).

[3] Baig, F., Fahad Khan, M., & Beg, S. (2013). Text writing in the air. *Journal of information display*, *14*(4), 137-148.

[4] Yilmaz, A., Javed, O., & Shah, M. (2006). Object tracking: A survey. *ACM computing surveys (CSUR)*, *38*(4), 13-es.

[5] Chang, Y. H., & Chang, C. M. (2010). Automatic Hand-Pose Trajectory Tracking System Using Video Sequences. In *User Interfaces*. IntechOpen.

[6] Delafontaine, M., Chavoshi, S. H., & Van de Weghe, N. (2019). Representing Moving Point Objects in Geospatial Sketch Maps.

[7] Bach, B., Sicat, R., Pfister, H., & Quigley, A. (2017, October). Drawing into the AR-CANVAS: Designing embedded visualizations for augmented reality. In *Workshop on Immersive Analytics, IEEE Vis* (Vol. 4).

[8] Pavlovic, V. I., Sharma, R., & Huang, T. S. (1997). Visual interpretation of hand gestures for human-computer interaction: A review. *IEEE Transactions on pattern analysis and machine intelligence*, *19*(7), 677-695.

[9] Salina, A., Kaivalya, K., Sriharsha, K., Praveen, K., & Nirosha, M. (2022). Creating Air Canvas Using Computer Vision. *International Journal of Advances in Engineering and Management (IJAEM). 4*(6), 443-451.

[10] Create Air Canvas using Python OpenCV. (2023). *GeeksforGeeks*. Available online: https://www.geeksforgeeks.org/create-air-canvas-using-python-opencv.

[11] More, S., Mhatre, P., Pakhare, S., & Khot, S. (2022). Air Canvas: Draw in Air. *International Research Journal of Engineering and Technology (IRJET) Volume*, *9*.

[12] Yang, L., Hong, X., Li, J., Ji, C. Y., Han, Y., Chen, S., ... & Fang, D. (2022). Rechargeable metasurfaces for dynamic color display based on a compositional and mechanical dual-altered mechanism. *Research*.