Drawing Application Using Mediapipe and Kalman Filter

1. Project Goals

- **Objective**:

Xây dựng một ứng dụng sử dụng webcam để nhận diện tay, cho phép người dùng vẽ trên màn hình bằng ngón tay trỏ. Ứng dụng sử dụng các công nghệ chính gồm:

    - Mediapipe: Phát hiện và theo dõi bàn tay.
    - Kalman Filter: Làm mượt dữ liệu đầu vào để cải thiện trải nghiệm vẽ.
    - OpenCV: Xử lý hình ảnh và hiển thị canvas.

- **Key Features**:
    - Bật/Tắt chế độ vẽ (d).
    - Xóa canvas (c).
    - Thay đổi màu bút (r, g, b).
    - Thay đổi độ dày bút (+, ``).

---

2. Theoretical Background

2.1 Mediapipe

Mediapipe là một framework mã nguồn mở được Google phát triển, tập trung vào các ứng dụng xử lý video thời gian thực như phát hiện bàn tay, khuôn mặt, và cơ thể.

- **Cách hoạt động trong ứng dụng**:
    - Mediapipe cung cấp mô hình phát hiện bàn tay, với các landmark cụ thể trên bàn tay (21 điểm).

    - Dữ liệu landmark được sử dụng để xác định vị trí đầu ngón tay trỏ, phục vụ việc vẽ.
- **Ưu điểm**:
    - Dễ sử dụng, hiệu suất cao.
    - Phù hợp cho cả CPU và GPU, thích hợp cho ứng dụng thời gian thực.
- **Áp dụng**:
    - Trong ứng dụng này, Mediapipe được dùng để:
        - Phát hiện bàn tay trong khung hình.
        - Trích xuất vị trí landmark của ngón tay trỏ (INDEX_FINGER_TIP).

2.2 Kalman Filter

Kalman Filter là một thuật toán toán học mạnh mẽ được sử dụng để dự đoán và lọc dữ liệu trong môi trường nhiễu.

- **Nguyên lý hoạt động**:
    - Dự đoán (Prediction): Dự đoán trạng thái tiếp theo của hệ thống dựa trên trạng thái hiện tại.
    - Hiệu chỉnh (Correction): Điều chỉnh dự đoán dựa trên dữ liệu quan sát.
- **Lợi ích trong ứng dụng**:
    - Xử lý dữ liệu vị trí ngón tay bị nhiễu, do rung tay hoặc môi trường không ổn định.
    - Tăng độ mượt mà của nét vẽ trên canvas.
- **Áp dụng trong ứng dụng**:
    - Kalman Filter được thiết lập với:

        - Trạng thái: (x, y, dx, dy) - Vị trí và vận tốc của ngón tay.
        - Quan sát: (x, y) - Dữ liệu vị trí thô từ Mediapipe.
    - Giúp làm mượt chuyển động của ngón tay khi vẽ trên canvas.

2.3 OpenCV

OpenCV (Open Source Computer Vision) là thư viện xử lý ảnh mã nguồn mở phổ biến.

- **Chức năng trong ứng dụng**:
    - Quản lý và hiển thị video từ webcam.
    - Xử lý và kết hợp canvas (khung vẽ) với khung hình webcam.
    - Hiển thị các thông báo và hỗ trợ tương tác với người dùng (ví dụ: "Drawing ON", "Pen Color: Red").

---

3. Design Overview

3.1 Workflow

Ứng dụng hoạt động theo quy trình sau:

1. **Nhận diện bàn tay**:
    - Sử dụng Mediapipe để phát hiện bàn tay và trích xuất vị trí đầu ngón tay trỏ.
2. **Làm mượt dữ liệu**:
    - Áp dụng Kalman Filter lên tọa độ ngón tay để giảm nhiễu và tạo chuyển động mượt mà hơn.
3. **Vẽ trên canvas**:
    - Khi chế độ vẽ được bật, tọa độ đã làm mượt sẽ được dùng để vẽ đường trên canvas.

4. **Hiển thị kết quả**:
    - Canvas được kết hợp với khung hình webcam và hiển thị cho người dùng.

3.2 System Architecture

Hệ thống được chia thành các module chính:

- **Hand Tracking**: Nhận diện tay và trích xuất tọa độ.
- **Drawing Control**: Xử lý đầu vào từ người dùng để bật/tắt chế độ vẽ, thay đổi màu và độ dày nét vẽ.
- **Canvas Management**: Quản lý và cập nhật canvas vẽ.
- **UI Display**: Hiển thị kết quả và thông báo lên màn hình.

3.3 Interaction Design

- **Hotkeys**:
    - d: Bật/Tắt chế độ vẽ.
    - c: Xóa toàn bộ canvas.
    - r/g/b: Thay đổi màu bút (Đỏ/Xanh lá/Xanh dương).
    - +/-: Tăng/Giảm độ dày nét vẽ.
    - Esc: Thoát ứng dụng.

---

4. Conclusion

4.1 Achieved Results

- **Functionality**:
    - Ứng dụng đã đạt được mục tiêu cho phép người dùng vẽ trên màn hình bằng ngón tay trỏ với các tính năng cơ bản như bật/tắt

chế độ vẽ, thay đổi màu bút, điều chỉnh độ dày nét vẽ, và xóa toàn bộ canvas.

    - Tích hợp thành công Mediapipe để phát hiện bàn tay và Kalman Filter để làm mượt chuyển động, mang lại trải nghiệm người dùng ổn định và mượt mà.

- **Performance**:
    - Ứng dụng hoạt động ổn định với tốc độ khung hình phù hợp (~30 FPS), đảm bảo khả năng xử lý thời gian thực.
    - Các tính năng như thay đổi màu bút, độ dày nét vẽ, và thông báo hiển thị tạm thời giúp tăng tính tương tác và thân thiện với người dùng.

4.2 Future Development Directions

1. **Feature Enhancements**:
    - **Chế độ vẽ đa màu**: Cho phép vẽ bằng nhiều màu sắc khác nhau trên cùng một canvas thay vì chỉ sử dụng một màu tại một thời điểm.
    - **Hỗ trợ vẽ hình học**: Cung cấp các tùy chọn như vẽ hình tròn, hình chữ nhật, hoặc đường thẳng khi người dùng thực hiện các cử chỉ tay đặc biệt.
2. **Multi-Hand Support**:
    - Mở rộng khả năng nhận diện nhiều bàn tay, cho phép sử dụng cả hai tay để điều khiển (một tay chọn công cụ, tay còn lại vẽ).
3. **Integration with AI Models**:
    - Tích hợp mạng học sâu để nhận diện các cử chỉ tay phức tạp, tạo ra các tính năng tương tác mới như xoay, phóng to/thu nhỏ canvas, hoặc điều chỉnh nét vẽ theo lực ấn giả định.

4. **Export Options**:
    - Cho phép người dùng lưu lại bản vẽ dưới dạng ảnh PNG hoặc JPEG.
    - Hỗ trợ tính năng undo/redo để cải thiện trải nghiệm vẽ.
5. **Cross-Platform Support**:
    - Triển khai ứng dụng trên các nền tảng khác như Android hoặc iOS, giúp người dùng tận dụng chức năng trên các thiết bị di động.
6. **Performance Optimization**:
    - Tối ưu hóa hiệu suất để hỗ trợ độ phân giải cao hơn và tăng tốc độ xử lý khung hình, đảm bảo hoạt động mượt mà ngay cả trên các thiết bị cấu hình thấp.
7. QuickDraw:
    - Tiến hành dự đoán sau khi người dùng hoàn thành bản vẽ

4.3 Final Remarks

Dự án này là một bước tiến thú vị trong việc sử dụng các công cụ hiện đại như Mediapipe, Kalman Filter, và OpenCV để xây dựng các ứng dụng tương tác thời gian thực. Với tiềm năng mở rộng và phát triển, ứng dụng không chỉ phục vụ mục đích giải trí mà còn có thể được tích hợp trong giáo dục, nghệ thuật kỹ thuật số, hoặc các nền tảng sáng tạo khác.

Hướng phát triển trong tương lai sẽ tập trung vào tăng cường trải nghiệm người dùng, mở rộng tính năng, QuickDraw và đảm bảo tính khả dụng trên các nền tảng khác nhau.

---

5. References

1. **Mediapipe Documentation**:

https://google.github.io/mediapipe/solutions/hands

2. **Kalman Filter Explanation**:

https://en.wikipedia.org/wiki/Kalman_filter

3. **OpenCV Documentation**:

https://docs.opencv.org

4. GitHub

https://github.com/MinHiuLe?tab=repositories