ĐỒ ÁN XỬ LÝ ẢNH: HỆ THỐNG CHẤM ĐIỂM TRẮC NGHIỆM VÀ NHẬN DẠNG VĂN BẢN TỰ ĐỘNG (OMR & OCR)

Nhóm thực hiện: Nguyễn Thế Minh Nhật

Ngày 25 tháng 12 năm 2025

Mục lục

1 Giới thiệu chung

Trong bối cảnh chuyển đổi số giáo dục, việc tự động hóa các quy trình thủ công như chấm thi và quản lý thông tin thí sinh ngày càng trở nên cấp thiết. Đồ án này trình bày giải pháp xây dựng một hệ thống xử lý ảnh thông minh, tích hợp hai chức năng chính:

1. **Chấm điểm trắc nghiệm tự động (Optical Mark Recognition - OMR):** Nhận dạng và chấm điểm các câu trả lời trên phiếu trắc nghiệm.
2. **Nhận dạng văn bản tự động (Optical Character Recognition - OCR):** Tự động phát hiện và trích xuất thông tin văn bản (như Tên, MSSV, Mã đề) từ các vùng bên ngoài phiếu trả lời.

Hệ thống được thiết kế để tối ưu hóa hiệu suất, giảm thiểu sai sót của con người và cung cấp kết quả một cách nhanh chóng, trực quan.

1.1 Các tính năng chính

- **Xử lý từ file PDF:** Hệ thống có khả năng đọc và xử lý trực tiếp file PDF chứa ảnh scan của phiếu làm bài.
- **Tự động định vị và căn chỉnh phiếu:** Tự động tìm kiếm khung viền của phiếu trong ảnh và áp dụng biến đổi hình học để làm phẳng, loại bỏ biến dạng phối cảnh.
- **Thiết lập bán tự động thông minh:**
  - *OMR:* Cung cấp công cụ tương tác để người dùng chỉ cần chọn 2 điểm neo (góc trên-trái và dưới-phải) cho khối trắc nghiệm trong lần đầu tiên.
  - *OCR:* Hoàn toàn tự động phát hiện các vùng chứa văn bản (tên, mã số...) nằm bên ngoài khối trắc nghiệm mà không cần sự can thiệp của người dùng.
- **Chấm điểm OMR chính xác:** Chấm điểm và so sánh với đáp án, đưa ra điểm số cuối cùng.
- **Trích xuất thông tin OCR:** Sử dụng model EasyOCR để nhận dạng và trích xuất nội dung văn bản từ các vùng đã được tự động phát hiện.
- **Trực quan hóa và lưu trữ kết quả:** Hiển thị kết quả chấm (đúng/sai) trực tiếp trên ảnh, tạo ảnh báo cáo điểm và lưu toàn bộ kết quả (ảnh, dữ liệu OCR) vào thư mục đầu ra.

1.2 Các công nghệ sử dụng

- **Python 3:** Ngôn ngữ lập trình chính của dự án.
- **OpenCV (Open Source Computer Vision Library):** Thư viện mã nguồn mở hàng đầu cho các tác vụ xử lý ảnh, được sử dụng cho hầu hết các công đoạn từ đọc ảnh, biến đổi hình học, đến phát hiện đối tượng.
- **NumPy:** Thư viện nền tảng cho tính toán khoa học trong Python, cung cấp cấu trúc mảng đa chiều hiệu suất cao để biểu diễn và tính toán ma trận điểm ảnh.

- **EasyOCR:** Một thư viện OCR mạnh mẽ và dễ sử dụng, hỗ trợ nhiều ngôn ngữ (bao gồm tiếng Việt), được dùng để trích xuất văn bản.
- **pdf2image:** Thư viện tiện ích giúp chuyển đổi các trang của file PDF thành đối tượng ảnh mà OpenCV có thể xử lý. Nó hoạt động như một lớp vỏ (wrapper) cho tiện ích Poppler.
- **Poppler:** Một thư viện render PDF, là phụ thuộc hệ thống bắt buộc của `pdf2image`.

2 Cấu trúc dự án

Cây thư mục của dự án được tổ chức một cách khoa học để dễ dàng quản lý và mở rộng:

`main.py`: File thực thi chính, điều phối toàn bộ luồng hoạt động của hệ thống.

`requirements.txt`: Danh sách các thư viện Python cần thiết.

`src/`: Thư mục chứa mã nguồn xử lý lõi.

- `config.py`: File cấu hình tập trung, chứa tất cả các đường dẫn, ngưỡng và tham số quan trọng.
- `pre_processing.py`: Các hàm tiền xử lý ảnh (chuyển đổi PDF, tìm và bẻ thẳng phiếu).
- `omr_logic.py`: Logic chấm điểm trắc nghiệm (tạo lưới tọa độ, phân tích ô được tô).
- `ocr_logic.py`: Logic trích xuất văn bản (sử dụng EasyOCR trên các vùng ảnh đầu vào).
- `measure_tool.py`: Công cụ tương tác cho việc thiết lập tọa độ lần đầu.
- `utils.py`: Các hàm tiện ích chung (sắp xếp điểm, biến đổi phối cảnh, xử lý contour).
- `ui.py`: Các hàm quản lý giao diện người dùng (hộp thoại, hiển thị kết quả).

`data/`: Chứa dữ liệu đầu vào.

- `raw/`: Chứa file PDF phiếu làm bài.
- `answer/`: Chứa file đáp án cho phần OMR.
- `template/`: Chứa file tọa độ được lưu sau lần thiết lập đầu tiên.

`output/`: Thư mục lưu trữ tất cả các kết quả đầu ra (ảnh, file JSON).

`poppler-25.11.0/`: Thư viện Poppler cần thiết cho việc xử lý PDF trên Windows.

3 Cấu trúc mô hình và Thuật toán

Hệ thống sử dụng một phương pháp lai (hybrid) kết hợp giữa thuật toán xử lý ảnh truyền thống và mô hình học sâu đã được huấn luyện trước.

3.1 Mô hình OMR: Thuật toán xử lý ảnh kinh điển

Phần chấm trắc nghiệm (OMR) không sử dụng mô hình học máy. Thay vào đó, nó dựa trên một chuỗi các bước xử lý ảnh kinh điển từ thư viện OpenCV để đảm bảo tốc độ và độ chính xác trên một mẫu phiếu đã xác định:

1. **Tiền xử lý ảnh:** Ảnh phiếu sau khi được làm phẳng (warped) sẽ được chuyển đổi sang ảnh thang độ xám (grayscale).

2. **Nhị phân hóa (Binarization):** Áp dụng phương pháp ngưỡng của Otsu (`THRESH_BINARY_INV | THRESH_OTSU`) để tạo ra một ảnh nhị phân. Trong ảnh này, các vùng được tô mực (đáp án, chữ viết) sẽ có giá trị pixel là 255 (màu trắng) và nền giấy là 0 (màu đen).

3. **Phân tích vùng đáp án:** Với mỗi ô đáp án (A, B, C, D) trong một câu hỏi:
   - Một mặt nạ hình tròn (`cv2.circle`) được tạo ra tại tọa độ của ô đáp án.
   - Hệ thống đếm số lượng pixel trắng (khác 0) nằm trong vùng mặt nạ này bằng hàm `cv2.countNonZero`.

4. **Xác định đáp án được chọn:** Lựa chọn có số pixel trắng cao nhất và vượt qua một ngưỡng nhiễu (`PIXEL_THRESHOLD`) sẽ được coi là đáp án mà thí sinh đã chọn. Nếu không có lựa chọn nào vượt ngưỡng, câu hỏi được xem là bỏ qua.

Phương pháp này hiệu quả vì nó không đòi hỏi dữ liệu huấn luyện và hoạt động rất nhanh, phù hợp cho bài toán có cấu trúc rõ ràng như phiếu trắc nghiệm.

3.2 Mô hình OCR: Mạng nơ-ron tích chập và LSTM

Phần nhận dạng văn bản (OCR) sử dụng thư viện **EasyOCR**, một mô hình học sâu mạnh mẽ đã được huấn luyện trước (pre-trained). Cấu trúc của EasyOCR là sự kết hợp của nhiều thành phần hiện đại:

1. **Phát hiện văn bản (Text Detection):** EasyOCR sử dụng một mô hình dựa trên mạng nơ-ron tích chập (CNN), thường là CRAFT (Character-Region Awareness for Text Detection), để xác định vị trí của các dòng hoặc các từ văn bản trong ảnh. Kết quả là các hộp giới hạn (bounding box) bao quanh vùng chứa chữ.

2. **Nhận dạng văn bản (Text Recognition):** Với mỗi hộp giới hạn đã được phát hiện, ảnh cắt (cropped image) của vùng văn bản đó sẽ được đưa vào một mô hình nhận dạng. Mô hình này thường là một kiến trúc ResNet + LSTM + Attention.
   - **ResNet (Residual Neural Network):** Đóng vai trò là tầng trích xuất đặc trưng (feature extractor) từ ảnh.
   - **LSTM (Long Short-Term Memory):** Một loại mạng nơ-ron hồi quy (RNN), xử lý chuỗi đặc trưng từ ResNet để hiểu được ngữ cảnh tuần tự của các ký tự.
   - **CTC (Connectionist Temporal Classification) hoặc Attention:** Là tầng giải mã (decoder) để chuyển đổi đầu ra của LSTM thành chuỗi ký tự cuối cùng.

Trong đồ án này, EasyOCR được cấu hình để sử dụng các mô hình đã được huấn luyện cho cả tiếng Việt và tiếng Anh, cho phép nó đọc được các thông tin như họ tên, mã số sinh viên, và các ghi chú khác.

4 Cài đặt và Hướng dẫn sử dụng

4.1 Yêu cầu về môi trường

1. **Python:** Phiên bản 3.8 trở lên.
2. **Poppler:** Do hệ thống cần xử lý file PDF, Poppler là một yêu cầu bắt buộc.
   - Dự án đã đi kèm thư mục `poppler-25.11.0` cho Windows.
   - **Quan trọng:** Người dùng cần thêm đường dẫn đầy đủ tới thư mục `poppler-25.11.0/Librar` vào biến môi trường `Path` của hệ thống. Nếu không có bước này, chương trình sẽ báo lỗi không tìm thấy Poppler.

4.2 Cài đặt các thư viện Python

Mở một cửa sổ dòng lệnh (terminal) trong thư mục gốc của dự án và chạy lệnh sau để cài đặt tất cả các thư viện cần thiết:

```
pip install -r requirements.txt
```

Lưu ý: Trong lần chạy đầu tiên, thư viện ***`easyocr`*** *có thể sẽ tự động tải về các model nhận dạng. Đây là quá trình chỉ diễn ra một lần và cần có kết nối Internet.*

4.3 Cấu hình hệ thống

Tất cả các cấu hình quan trọng đều được tập trung tại file `src/config.py`.

- `PDF_PATH`: Đường dẫn tới file PDF phiếu làm bài đầu vào.
- `ANSWER_KEY_PATH`: Đường dẫn tới file `.csv` chứa đáp án của phần trắc nghiệm.
- `COORDINATES_PATH`: Đường dẫn để lưu file template tọa độ sau lần thiết lập đầu tiên.

Người dùng cần đảm bảo các file dữ liệu (phiếu làm bài, đáp án) được đặt đúng vị trí và các đường dẫn trong file config là chính xác.

4.4 Hướng dẫn vận hành

1. **Chạy chương trình:**

```
python main.py
```

2. **Thiết lập lần đầu (First-Time Setup):**

   Nếu hệ thống không tìm thấy file tọa độ template, một hộp thoại sẽ hiện lên hỏi quyền được bắt đầu quá trình thiết lập.

- **Giai đoạn 1 (Thủ công):** Một cửa sổ hiển thị ảnh phiếu sẽ mở ra. Người dùng cần click chuột vào 2 vị trí theo hướng dẫn trên console: tâm của ô đáp án **trên cùng bên trái** và tâm của ô **dưới cùng bên phải** của khối trắc nghiệm.
- **Giai đoạn 2 (Tự động):** Ngay sau khi 2 điểm được chọn, cửa sổ sẽ đóng lại. Hệ thống sẽ tự động chạy nền để tìm các vùng văn bản OCR.
- Toàn bộ tọa độ sau đó được lưu vào file `coordinates.json`.

3. **Các lần chạy sau:**

   Chương trình sẽ tự động đọc file `coordinates.json` đã lưu và thực hiện toàn bộ quy trình chấm điểm, nhận dạng mà không cần bất kỳ tương tác nào.

5 Kết quả thực nghiệm và Phân tích

Phần này trình bày chi tiết về dữ liệu được sử dụng và các kết quả số liệu thu được từ quá trình chạy thực nghiệm hệ thống.

5.1 Tập dữ liệu (Dataset)

Một điểm quan trọng cần làm rõ là hệ thống này **không yêu cầu quá trình huấn luyện (training)** cho các tác vụ chính. Do đó, không có khái niệm tập huấn luyện (train set) và tập kiểm thử (test set) như trong các bài toán học máy truyền thống.

- **Đối với OMR:** Thuật toán dựa trên xử lý ảnh kinh điển (đếm pixel), không cần dữ liệu để học.
- **Đối với OCR:** Hệ thống sử dụng mô hình `EasyOCR` đã được huấn luyện trước trên một tập dữ liệu lớn gồm nhiều ngôn ngữ (bao gồm tiếng Việt). Chúng ta chỉ sử dụng lại mô hình này (inference).

Do đó, "dữ liệu" trong bối cảnh của đồ án này bao gồm:

- **Dữ liệu đầu vào:** Một file PDF mẫu (`Mau_de_thi_co_dap_an.pdf`) chứa ảnh scan của một phiếu trả lời đã được điền.
- **Đáp án:** Một file CSV (`answer_key.csv`) chứa danh sách các đáp án đúng để hệ thống so sánh và chấm điểm.

Việc kiểm thử hệ thống được thực hiện bằng cách chạy xử lý trên file PDF mẫu này và so sánh kết quả đầu ra với đáp án đã biết.

5.2 Kết quả số liệu và Phân tích

Hệ thống được chạy với dữ liệu đầu vào đã mô tả. Dưới đây là các kết quả số liệu cụ thể thu được.

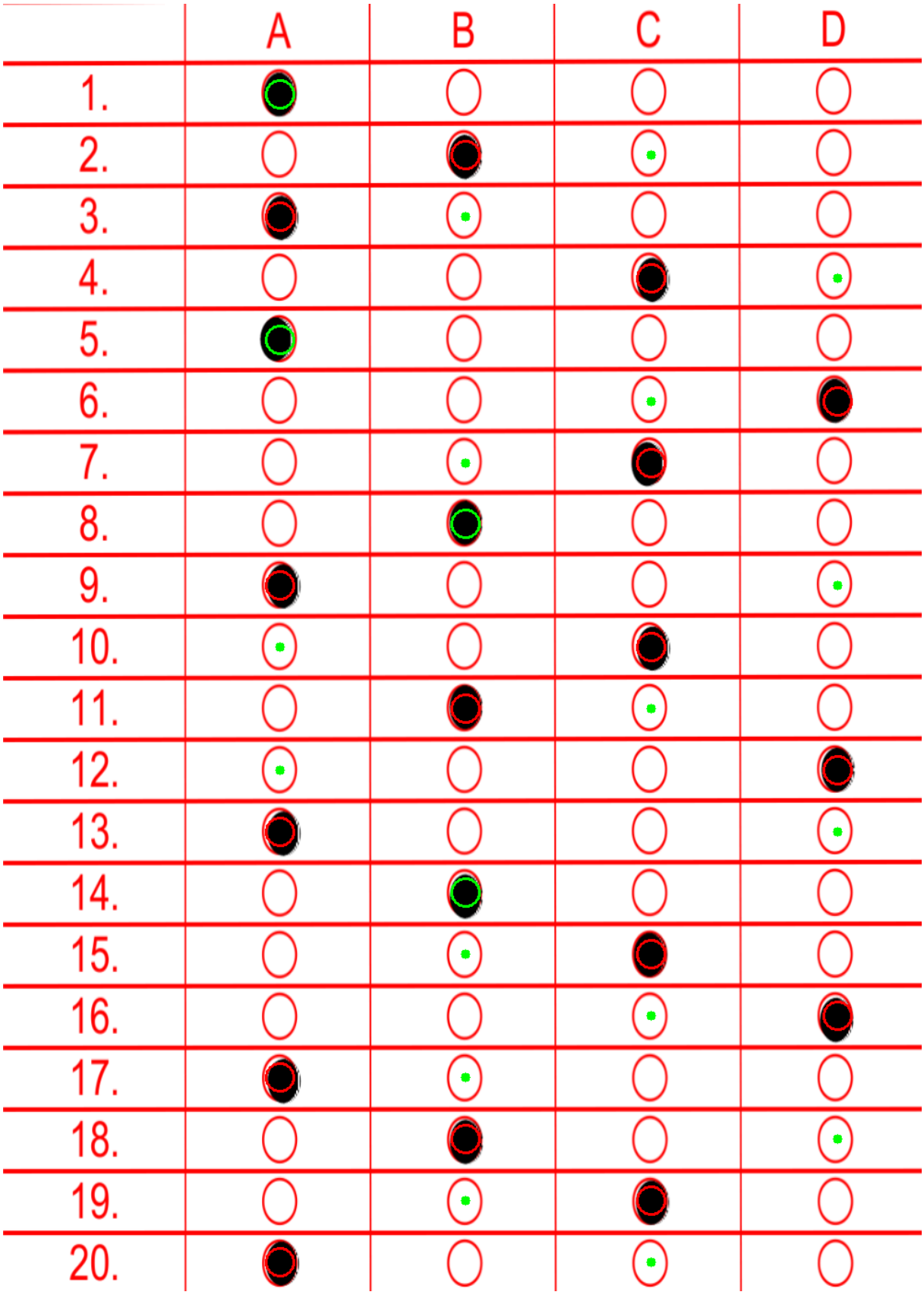
5.2.1 Kết quả chấm điểm OMR

Sau khi xử lý phần trắc nghiệm, hệ thống ghi nhận các kết quả sau:

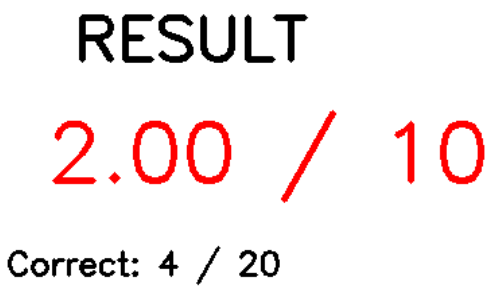
- **Tổng số câu hỏi:** 20
- **Số câu trả lời đúng:** 4
- **Điểm số (thang 10): 2.00 / 10**

Các kết quả này được trực quan hóa qua hai tệp ảnh trong thư mục `output/`:

- `scoring_result.png`: Ảnh phiếu làm bài với các đáp án được tô màu (Xanh: đúng, Đỏ: sai). (Xem Hình 1)
- `score.png`: Ảnh thẻ điểm tóm tắt kết quả. (Xem Hình 2)



Hình 1: Kết quả chấm điểm được trực quan hóa trên phiếu làm bài.



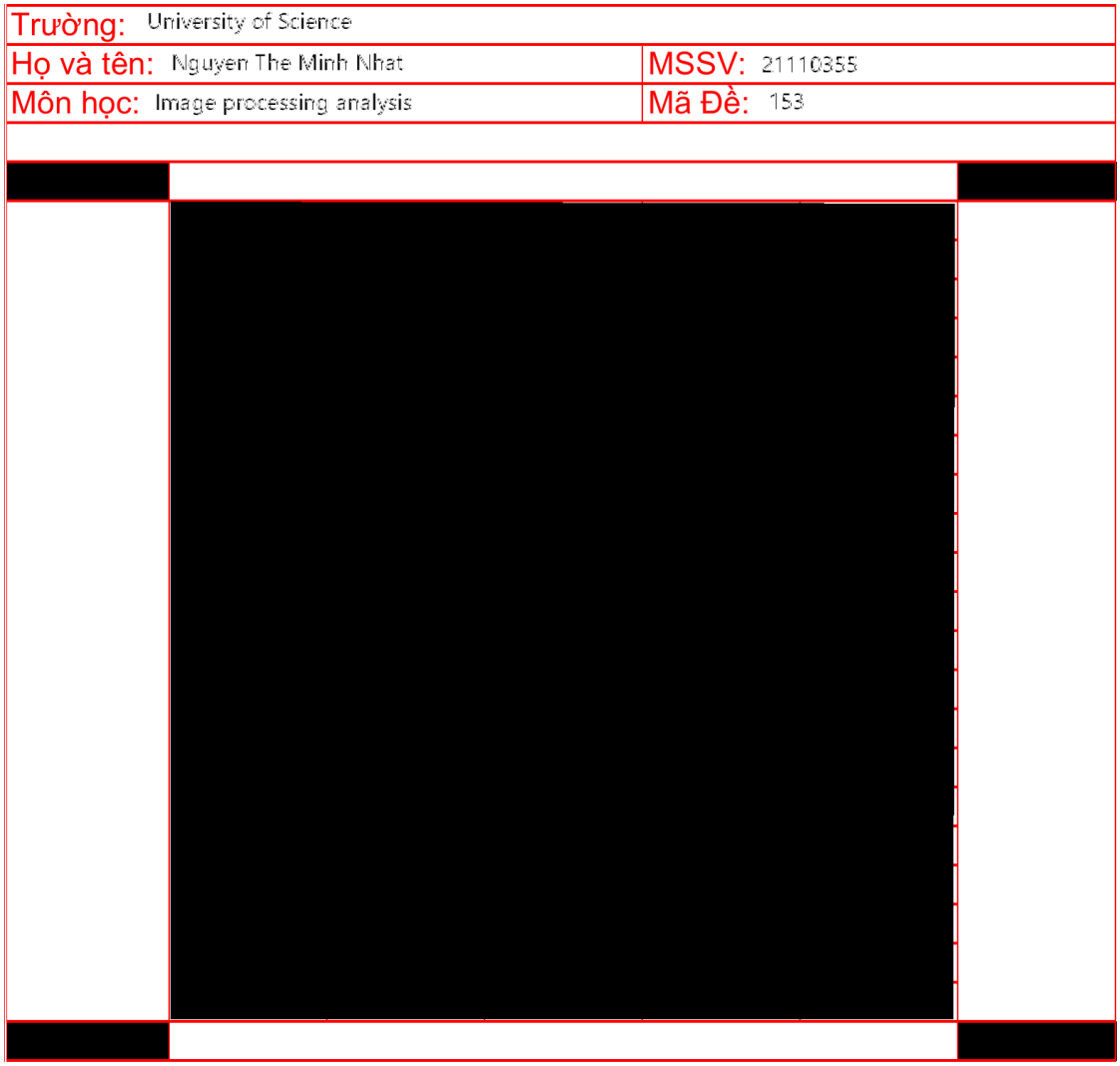
Hình 2: Thẻ điểm được tạo tự động.

5.2.2 Kết quả nhận dạng OCR

Phần OCR được thực thi trên vùng ảnh bên ngoài phiếu trắc nghiệm để trích xuất thông tin thí sinh.

- **Văn bản trích xuất được từ vùng thông tin:** `'2111Ũ355 153'`

Kết quả này được lưu tại tệp `output/ocr_results.json`. Phân tích kết quả: Hệ thống đã nhận dạng được dãy ký tự. Tuy nhiên, có thể thấy một lỗi nhận dạng nhỏ: ký tự `'Ũ'` có khả năng là một sự nhầm lẫn của model. Dựa vào ngữ cảnh, ký tự này có thể là '0', 'D', hoặc một ký tự khác, cho thấy thách thức của việc nhận dạng trong điều kiện ảnh scan không hoàn hảo.



Hình 3: Vùng ảnh được sử dụng cho tác vụ OCR (phần màu đen đã bị loại bỏ).

5.3 Đánh giá chung

- **OMR:** Tỷ lệ đúng 4/20 cho thấy sự khác biệt giữa câu trả lời của thí sinh trên phiếu và đáp án. Về mặt kỹ thuật, thuật toán đã xác định đúng các ô được tô, và

kết quả điểm số phản ánh chính xác sự so khớp này.

- **OCR:** Model đã thành công trong việc trích xuất một chuỗi văn bản có ý nghĩa, mặc dù có lỗi nhỏ. Điều này cho thấy `EasyOCR` hoạt động tốt nhưng không phải lúc nào cũng hoàn hảo 100%, đặc biệt với các ký tự không rõ ràng.
- **Hiệu năng:** Toàn bộ quá trình xử lý cho một phiếu (bao gồm OMR và OCR) diễn ra trong vài giây trên một máy tính thông thường (không có GPU), cho thấy hiệu quả của giải pháp.

6 Kết luận và Hướng phát triển

6.1 Kết luận

Đồ án đã xây dựng thành công một hệ thống hoàn chỉnh có khả năng tự động hóa hai tác vụ quan trọng trong chấm thi: chấm điểm trắc nghiệm (OMR) và trích xuất thông tin thí sinh (OCR). Bằng cách kết hợp các thuật toán xử lý ảnh kinh điển của OpenCV và sức mạnh của model học sâu từ EasyOCR, hệ thống đã giải quyết được bài toán đặt ra với các kết quả chính:

- Tự động hóa toàn bộ quy trình từ đọc file PDF đến xuất kết quả.
- Giảm thiểu tối đa sự can thiệp của người dùng nhờ cơ chế thiết lập template thông minh và tự động phát hiện vùng OCR.
- Cung cấp kết quả đầu ra đa dạng, trực quan và dễ dàng lưu trữ, quản lý.

Dự án là một minh chứng rõ ràng cho tiềm năng của Xử lý ảnh trong việc giải quyết các bài toán thực tế trong lĩnh vực giáo dục và xa hơn nữa.

6.2 Hạn chế của hệ thống

- **Phụ thuộc vào cấu trúc phiếu:** Logic sinh tọa độ OMR hiện tại được thiết kế cho một mẫu phiếu nhất định. Việc thay đổi sang một mẫu phiếu có bố cục hoàn toàn khác (ví dụ: 2 cột câu hỏi) sẽ cần can thiệp vào mã nguồn.
- **Chất lượng ảnh hưởng lớn:** Chất lượng của ảnh scan (độ phân giải, độ tương phản, ánh sáng) ảnh hưởng trực tiếp đến độ chính xác của cả OMR và OCR.
- **Giao diện người dùng:** Hệ thống vẫn hoạt động chủ yếu trên giao diện dòng lệnh và các cửa sổ của OpenCV, chưa có một giao diện đồ họa (GUI) hoàn chỉnh.

6.3 Hướng phát triển

- **Xây dựng giao diện đồ họa (GUI):** Phát triển một ứng dụng desktop (sử dụng PyQt, Tkinter) hoặc web (sử dụng Flask, Django) để người dùng có thể dễ dàng tải lên file, cấu hình và xem kết quả một cách thân thiện hơn.
- **Nâng cao khả năng thích ứng:** Phát triển các thuật toán thông minh hơn để hệ thống có thể tự động nhận dạng bố cục của các loại phiếu trắc nghiệm khác nhau mà không cần cấu hình lại.

- **Xử lý theo lô (Batch Processing):** Cho phép người dùng tải lên nhiều file cùng lúc và hệ thống tự động xử lý hàng loạt.
- **Cải thiện độ chính xác OCR:** Thử nghiệm tích hợp các engine OCR khác hoặc tinh chỉnh model để cải thiện độ chính xác với chữ viết tay.