PROJECT CUỐI KỲ
HỌC PHẦN XỬ LÝ ẢNH SỐ

DIPR430685

[Học kỳ II/Năm học: 2024-2025]

Giảng viên phụ trách: Huỳnh Thế Thiện

Deadline nộp bài: 23:59, Thứ Ba, ngày 06 tháng 05 năm 2025

1 MỤC TIÊU PROJECT

Project này nhằm mục đích giúp sinh viên củng cố kiến thức đã học và áp dụng các kỹ thuật xử lý ảnh số vào giải quyết các bài toán thực tế bằng ngôn ngữ lập trình Python và các thư viện phổ biến. Sinh viên sẽ thực hành các kỹ thuật về trích xuất đặc trưng (PCA, SIFT, LBPH), biến đổi hình học, khớp ảnh và ứng dụng học máy cơ bản trong lĩnh vực xử lý ảnh.

2 YÊU CẦU CHUNG

- **Ngôn ngữ lập trình:** Python 3.x.
- **Thư viện chính:** Sinh viên được phép và khuyến khích sử dụng các thư viện Python phổ biến cho xử lý ảnh và khoa học dữ liệu, bao gồm nhưng không giới hạn:
  - OpenCV (`opencv-python` và `opencv-contrib-python` khi cần SIFT).
  - NumPy (`numpy`).
  - Matplotlib (`matplotlib`).
  - Scikit-learn (`scikit-learn` cho PCA, RandomForestClassifier, StandardScaler, metrics).
  - Scikit-image (`scikit-image` cho LBPH).
  - Kaggle (`kaggle`, `kagglehub`) và Pandas (`pandas`) cho Tác vụ 4.
  - Các thư viện chuẩn khác của Python (`os`, `math`, `time`, `glob`...).
- **Tính di động (Portability):** Mã nguồn phải được viết sao cho có thể chạy được trên các hệ điều hành khác nhau (Windows, macOS, Linux) mà chỉ cần cài đặt các thư viện phụ thuộc nêu trên. **Tuyệt đối tránh sử dụng đường dẫn tuyệt đối (hard-coded absolute paths)** trong code. Sử dụng đường dẫn tương đối hoặc các hàm từ thư viện `os` (ví dụ: `os.path.join()`, `os.getcwd()`) để xử lý đường dẫn file một cách linh hoạt.
- **Chất lượng mã nguồn:**
  - Code phải rõ ràng, dễ đọc, có cấu trúc tốt (sử dụng hàm, lớp nếu cần).
  - Đặt tên biến, hàm có ý nghĩa.
  - **Bắt buộc:** Phải có chú thích (comment) đầy đủ, giải thích rõ ràng mục đích của các khối lệnh, các hàm, các tham số quan trọng, và các bước xử lý phức tạp. Sử dụng docstring cho các hàm tự định nghĩa.
- **Xử lý lỗi cơ bản:** Code cần có các kiểm tra lỗi cơ bản như kiểm tra sự tồn tại của file ảnh đầu vào trước khi đọc (`os.path.exists`), kiểm tra kết quả trả về `None` từ các hàm OpenCV (`cv2.imread`), v.v.

3 CÁC TÁC VỤ CỤ THỂ (Yêu cầu thực hiện đủ 4 tác vụ)

Sinh viên cần viết 04 chương trình Python riêng biệt (file `.py`) hoặc trong cùng 01 file Jupyter Notebook (file `.ipynb`) nhưng phải có phân tách và giải thích rõ ràng cho từng tác vụ.

3.1 Tác vụ 1: Nén ảnh sử dụng PCA (Principal Component Analysis)

- **Mục tiêu:** Áp dụng PCA trên các khối ảnh để thực hiện nén ảnh mất mát.
- **Input:** Một ảnh màu bất kỳ (sẽ được cung cấp hoặc sinh viên tự chọn và ghi rõ trong báo cáo).
- **Output:**
  - Ảnh màu đã được nén và tái tạo, lưu thành file ảnh mới.
  - Hiển thị so sánh ảnh gốc và ảnh tái tạo bằng Matplotlib.
  - In ra giá trị MSE (Mean Squared Error) và/hoặc PSNR (Peak Signal-to-Noise Ratio) giữa ảnh gốc và ảnh tái tạo.
  - In ra tỷ lệ nén ước lượng (dựa trên số lượng hệ số PCA và thông tin cần lưu so với số pixel gốc).
- **Yêu cầu kỹ thuật:**
  - Cho phép người dùng cấu hình `block_size` (ví dụ: 8) và `n_components` (số thành phần chính giữ lại trên mỗi khối, ví dụ: 10, 20, 30,... và phải $\leq$ `block_size*block_size`).
  - Xử lý từng kênh màu riêng biệt (nếu là ảnh màu).
  - Chia kênh ảnh thành các khối vuông không chồng lấn (có thể cần padding ảnh gốc).
  - Áp dụng `sklearn.decomposition.PCA` cho tập hợp các vector khối (đã làm phẳng).
  - Tái tạo lại các khối từ `n_components` thành phần chính và các hệ số tương ứng.
  - Ghép các khối tái tạo thành ảnh hoàn chỉnh (cắt bỏ padding nếu có).
  - Lưu ảnh kết quả với tên file có ý nghĩa (ví dụ: `[ten_anh_goc]_pca_k[so_components].jpg`).
  - **Comment rõ ràng** các bước, đặc biệt là cách chia khối, áp dụng PCA và tái tạo ảnh.

3.2 Tác vụ 2: Ghép ảnh Panorama sử dụng SIFT

- **Mục tiêu:** Ghép hai ảnh có vùng chồng lấn thành một ảnh panorama rộng hơn.
- **Input:** Hai ảnh màu có vùng nhìn chồng lấn (sẽ được cung cấp).
- **Output:**
  - Ảnh panorama kết quả, lưu thành file ảnh mới.
  - Hiển thị ảnh panorama bằng Matplotlib.
  - (Tùy chọn) Hiển thị ảnh thể hiện các cặp điểm khớp được sử dụng.
- **Yêu cầu kỹ thuật:**
  - Sử dụng SIFT (ví dụ: `cv2.SIFT_create()` từ `opencv-contrib-python`) để phát hiện keypoints và tính descriptors cho cả hai ảnh.
  - Sử dụng `cv2.BFMatcher` với `knnMatch` (k=2) và **Lowe's Ratio Test** để lọc ra các cặp khớp tốt (good matches).
  - Sử dụng `cv2.findHomography` với thuật toán **RANSAC** để ước lượng ma trận Homography (H) từ các cặp khớp tốt. Kiểm tra số lượng khớp tốt và inliers tối thiểu (ví dụ: > 10).
  - Sử dụng `cv2.warpPerspective` để biến đổi (warp) một ảnh vào hệ quy chiếu của ảnh còn lại dựa trên ma trận H.

- Kết hợp (ghép/trộn) ảnh gốc và ảnh đã warp thành ảnh panorama cuối cùng (có thể dùng phương pháp ghép đơn giản hoặc các kỹ thuật blending nâng cao hơn).
- Lưu ảnh panorama kết quả (ví dụ: `panorama_result.jpg`).
- **Comment rõ ràng** các bước SIFT, matching, ratio test, RANSAC, warping và stitching.

3.3 Tác vụ 3: Khớp ảnh / Đăng ký ảnh sử dụng SIFT

- **Mục tiêu:** Tìm và định vị một ảnh đối tượng (nhỏ) bên trong một ảnh cảnh (lớn).
- **Input:** Một ảnh cảnh màu và một ảnh đối tượng màu (ảnh đối tượng là một phần trích ra từ ảnh cảnh, sẽ được cung cấp).
- **Output:**
  - Ảnh cảnh với một **khung bao (bounding box)** được vẽ xung quanh vị trí đối tượng được tìm thấy. Lưu ảnh này ra file.
  - Hiển thị ảnh thể hiện các cặp điểm khớp tốt (inliers) giữa ảnh đối tượng và ảnh cảnh (có thể phân biệt màu inliers/outliers).
- **Yêu cầu kỹ thuật:**
  - Thực hiện các bước phát hiện, mô tả SIFT và khớp đặc trưng (KNN + Ratio Test) tương tự Tác vụ 2.
  - Ước lượng ma trận Homography (H) từ ảnh đối tượng sang ảnh cảnh bằng `cv2.findHomography` với RANSAC.
  - Lấy tọa độ 4 góc của ảnh đối tượng.
  - Sử dụng hàm `cv2.perspectiveTransform` và ma trận H để tìm tọa độ tương ứng của 4 góc đó trên ảnh cảnh.
  - Sử dụng `cv2.polylines` để vẽ khung bao nối 4 điểm góc tìm được lên ảnh cảnh.
  - Lưu ảnh cảnh với khung bao (ví dụ: `scene_with_detected_object.jpg`).
  - Hiển thị ảnh các cặp điểm khớp (sử dụng `status` trả về từ `findHomography` để phân biệt inliers/outliers bằng màu sắc).
  - **Comment rõ ràng** các bước, đặc biệt là cách tính toán và vẽ khung bao dựa trên Homography và cách sử dụng status mask.

3.4 Tác vụ 4: Nhận dạng Biểu cảm Khuôn mặt dùng LBPH và Random Forest

- **Mục tiêu:** Xây dựng mô hình nhận dạng biểu cảm khuôn mặt cơ bản sử dụng đặc trưng LBPH và bộ phân loại Random Forest.
- **Input:** Tự động tải bộ dữ liệu FER2013 từ Kaggle sử dụng thư viện `kagglehub`.
- **Output:**
  - Độ chính xác (accuracy) của mô hình trên tập kiểm tra (test set từ dataset).
  - Báo cáo phân loại chi tiết (`classification_report` bao gồm precision, recall, f1-score cho từng lớp).
  - (Tùy chọn) Hiển thị một vài dự đoán trên ảnh kiểm tra.

- **Yêu cầu kỹ thuật:**
  - **Bắt buộc** sử dụng `kagglehub` để tải bộ dữ liệu `msambare/fer2013`. Sinh viên cần tự cài đặt thư viện `kaggle`, `kagglehub` và **thiết lập Kaggle API credentials** (`kaggle.json`) trên máy của mình theo hướng dẫn của Kaggle.
  - Code phải xử lý được cấu trúc thư mục `train`/`test` chứa các thư mục con đặt tên theo biểu cảm được Kaggle Hub tải về.
  - Đọc ảnh grayscale từ các thư mục tương ứng.
  - Trích xuất đặc trưng LBPH từ mỗi ảnh sử dụng `skimage.feature.local_binary_pattern`. **Khuyến khích** sử dụng `method='uniform'`. Cho phép cấu hình tham số `P` (số điểm lân cận) và `R` (bán kính).
  - Tính toán histogram của mã LBP cho mỗi ảnh để tạo vector đặc trưng LBPH. Chuẩn hóa histogram (ví dụ: L1 norm).
  - Sử dụng `sklearn.preprocessing.StandardScaler` để chuẩn hóa các vector đặc trưng LBPH trước khi huấn luyện.
  - Sử dụng bộ phân loại **Random Forest** (`sklearn.ensemble.RandomForestClassifier`) để huấn luyện mô hình trên tập `train` (đặc trưng LBPH và nhãn). Cho phép điều chỉnh các tham số của Random Forest (ví dụ: `n_estimators`, `max_depth`).
  - Đánh giá mô hình trên tập `test` (sử dụng `accuracy_score` và `classification_report`).
  - **Comment rõ ràng** các bước tải dữ liệu, xử lý cấu trúc thư mục, trích xuất LBPH (ý nghĩa tham số P, R, method, số bins), chuẩn hóa, huấn luyện và đánh giá Random Forest.

4 DỮ LIỆU THỬ NGHIỆM

- Đối với Tác vụ 1, 2, 3: Giảng viên sẽ cung cấp một bộ ảnh mẫu để sinh viên thử nghiệm và kiểm tra chương trình trong buổi báo cáo (nếu có) hoặc để chấm điểm. Sinh viên cũng nên tự tìm thêm ảnh để thử nghiệm và đưa vào báo cáo.
- Đối với Tác vụ 4: Chương trình phải tự động tải bộ dữ liệu FER2013 từ Kaggle khi chạy lần đầu (nếu chưa có trong cache) và đọc từ cấu trúc thư mục do Kaggle Hub tạo ra. Sinh viên cần đảm bảo đã cài đặt và xác thực Kaggle API.

5 SẢN PHẨM NỘP

Sinh viên cần nộp một file nén duy nhất (`.zip` hoặc `.rar`) chứa các thành phần sau:

1. **Mã nguồn:**
   - Các file mã nguồn Python (`.py`) riêng biệt cho từng tác vụ, đặt tên file rõ ràng (ví dụ: `task1_pca.py`, `task2_panorama.py`,...).
   - Hoặc một file Jupyter Notebook (`.ipynb`) duy nhất, nhưng phải có cấu trúc rõ ràng, phân tách code và giải thích markdown cho từng tác vụ.
   - **Quan trọng:** Code phải chạy được và kèm theo file `requirements.txt` liệt kê các thư viện cần thiết (hoặc ghi rõ trong báo cáo).

2. **Báo cáo Project (Report):** File báo cáo chi tiết dưới dạng **PDF**. Nội dung báo cáo cần tuân theo cấu trúc đề xuất (xem mục 6).

3. **Dữ liệu (Nếu có):**

   - Chỉ nộp các ảnh đầu vào mẫu mà sinh viên *tự tìm thêm* (nếu có) để minh họa trong báo cáo. Không cần nộp lại các ảnh do giảng viên cung cấp.
   - Không nộp bộ dữ liệu FER2013.
   - Các ảnh kết quả chính do chương trình sinh ra (ảnh nén, panorama, ảnh khớp, ảnh có khung bao) nên được chèn vào báo cáo hoặc nộp kèm trong thư mục riêng (`results/`).

Quy cách đặt tên file nộp: `MSSV_HoTen_ProjectXLAS.zip`
(Ví dụ: `2001101_NguyenVanA_ProjectXLAS.zip`)

6 YÊU CẦU BÁO CÁO (REPORT)

Báo cáo cần trình bày rõ ràng, mạch lạc, đầy đủ các nội dung sau (định dạng PDF):

- **Trang bìa:** Tên học phần, Tên Project, Tên Giảng viên, Tên Sinh viên, MSSV, Lớp, Ngày nộp.
- **Mục lục.**
- **Chương 1: Giới thiệu**
  - Mục tiêu tổng quan của project.
  - Phạm vi các bài toán được giải quyết.
  - Cấu trúc của báo cáo.
- **Chương 2, 3, 4, 5: Nội dung thực hiện từng tác vụ** (Lặp lại cấu trúc này cho 4 tác vụ)
  - **Mô tả bài toán:** Nêu rõ yêu cầu và mục tiêu của tác vụ cụ thể.
  - **Cơ sở lý thuyết:** Trình bày ngắn gọn, súc tích về lý thuyết của phương pháp chính được sử dụng (PCA, SIFT, Homography, LBPH, Random Forest...). Giải thích tại sao phương pháp đó phù hợp.
  - **Phương pháp thực hiện:**
    * Mô tả các bước chính đã thực hiện trong code (có thể dùng lưu đồ hoặc mô tả bằng lời).
    * Giải thích các thư viện, hàm quan trọng đã sử dụng.
    * Nêu rõ các tham số đã chọn và lý do lựa chọn (ví dụ: `block_size`, `n_components` trong PCA; `ratio_thresh`, `reproj_thresh` trong SIFT; `P`, `R`, `method` trong LBPH; các tham số của Random Forest...). Có thể trình bày kết quả thử nghiệm với các tham số khác nhau (nếu có).
  - **Kết quả thực nghiệm:**
    * Trình bày các kết quả đạt được một cách trực quan: chèn ảnh kết quả (ảnh nén/tái tạo, panorama, ảnh khớp, ảnh khung bao), các chỉ số đo lường (MSE, PSNR, Accuracy, Báo cáo phân loại...).

    - Sử dụng bảng biểu, đồ thị để minh họa nếu cần thiết. Chú thích rõ ràng cho hình ảnh, bảng biểu.
  - **Thảo luận và Nhận xét:**
    - Phân tích kết quả: Kết quả có tốt không? Có đạt được mục tiêu không? Tại sao?
    - Ưu điểm, nhược điểm của phương pháp đã sử dụng trong bối cảnh bài toán cụ thể này.
    - Các yếu tố ảnh hưởng đến kết quả (chất lượng ảnh đầu vào, tham số, giới hạn của thuật toán...).
    - So sánh các phương pháp (nếu có thể, ví dụ so sánh kết quả SIFT và ORB nếu SV tìm hiểu thêm, hoặc so sánh chất lượng nén với các giá trị k khác nhau).
    - Đề xuất hướng cải tiến cho từng tác vụ (nếu có).

- **Chương 6: Kết luận chung**
  - Tóm tắt các kết quả chính đạt được của toàn bộ project.
  - Những kiến thức, kỹ năng đã học hỏi và vận dụng được.
  - Khó khăn gặp phải trong quá trình thực hiện và cách giải quyết (nếu có).
  - Hướng phát triển trong tương lai (nếu có).
- **Tài liệu tham khảo (Nếu có):** Liệt kê các sách, bài báo, trang web đã tham khảo.
- **Phụ lục (Nếu cần):** Có thể đính kèm mã nguồn chính vào phụ lục nếu muốn, nhưng việc nộp file code riêng vẫn là bắt buộc.

7 TIÊU CHÍ ĐÁNH GIÁ

Project sẽ được đánh giá dựa trên các tiêu chí sau (thang điểm 10):

- **Mã nguồn (4.0 điểm):**
  - Chương trình chạy đúng chức năng, không phát sinh lỗi với dữ liệu thử nghiệm hợp lệ (2.0 điểm).
  - Hoàn thành đầy đủ các yêu cầu kỹ thuật của từng tác vụ (1.0 điểm).
  - Code rõ ràng, dễ đọc, có cấu trúc tốt, comment đầy đủ và đúng chuẩn (1.0 điểm).
- **Báo cáo (6.0 điểm):**
  - Đầy đủ nội dung theo yêu cầu, đúng cấu trúc (1.5 điểm).
  - Trình bày khoa học, logic, rõ ràng, mạch lạc, không lỗi chính tả/ngữ pháp, định dạng tốt (1.5 điểm).
  - Giải thích lý thuyết và phương pháp thực hiện chính xác, thể hiện sự hiểu biết sâu sắc về các thuật toán và thư viện (1.5 điểm).
  - Phân tích, thảo luận kết quả sâu sắc, có cơ sở, đưa ra nhận xét và đề xuất hợp lý (1.5 điểm).

8 THỜI HẠN VÀ PHƯƠNG THỨC NỘP BÀI

- **Deadline nộp bài: 23:59, Thứ Ba, ngày 06 tháng 05 năm 2025.**
- **Hình thức nộp bài:** [Ghi rõ phương thức: Ví dụ: Nộp 01 file nén `.zip` hoặc `.rar` qua hệ thống LMS / Google Classroom / Email GV ([Email của bạn]) với tiêu đề theo định dạng `MSSV - HoTen - Nop Project XLAS`]. File nén phải chứa đầy đủ các sản phẩm yêu cầu (mã nguồn, báo cáo PDF, dữ liệu/kết quả mẫu nếu có).
- Nộp bài sau thời hạn sẽ bị trừ điểm theo quy định của học phần/khoa.

9 QUY ĐỊNH VỀ LIÊM CHÍNH HỌC THUẬT

- Sinh viên **bắt buộc** phải tự thực hiện project này. Nghiêm cấm mọi hành vi sao chép mã nguồn hoặc nội dung báo cáo từ các nguồn khác (bạn bè, internet, project khóa trước...) mà không được phép hoặc không trích dẫn rõ ràng.
- Mọi trường hợp vi phạm liêm chính học thuật sẽ bị xử lý nghiêm theo quy định của nhà trường, bao gồm cả việc nhận điểm 0 cho project này.
- Sinh viên được phép tham khảo tài liệu, sách, bài báo, và các mã nguồn mở trên Internet. Tuy nhiên, sinh viên phải hiểu rõ những gì mình tham khảo, tự cài đặt lại, và phải ghi rõ nguồn tham khảo trong báo cáo nếu sử dụng lại ý tưởng hoặc các đoạn mã nguồn đáng kể.

Chúc các bạn sinh viên hoàn thành tốt project!