BỘ XÂY DỰNG
HỌC VIỆN HÀNG KHÔNG VIỆT NAM
KHOA CÔNG NGHỆ THÔNG TIN

Xử lý ảnh và thị giác máy tính

Chương trình tạo ảnh động

Giảng viên hướng dẫn: TS. Trần Nguyên Bảo

Nhóm sinh viên thực hiện: Nhóm 04

Lớp: 010100086903

TP.Hồ Chí Minh, tháng 11/2025

BỘ XÂY DỰNG
HỌC VIỆN HÀNG KHÔNG VIỆT NAM
KHOA CÔNG NGHỆ THÔNG TIN

Xử lý ảnh và thị giác máy tính

Chương trình tạo ảnh động

Giảng viên hướng dẫn: TS. Trần Nguyên Bảo

Nhóm sinh viên thực hiện: Nhóm 04

Lớp: 010100086903

TP.Hồ Chí Minh, tháng 11/2025

Danh sách Nhóm:

STT	Họ và tên	MSSV	Lớp	Ghi chú
1	**Lê Quang Nguyên**	**2331540284**	**23ĐHTT01**	Thành viên
2	**Ao Trường Giang**	**2331540143**	**23ĐHTT03**	Thành viên
3	**Trần Phạm Minh Đức**	**2331540141**	**23ĐHTT03**	Nhóm Trưởng
4	**Hoàng Quốc Cường**	**2331540219**	**23ĐHTT04**	Thành viên

Cán bộ chấm thi 1 *(ký và ghi rõ họ tên)*	**Cán bộ chấm thi 2** *(ký và ghi rõ họ tên)*
Cán bộ chấm thi phúc khảo 1 *(ký và ghi rõ họ tên)*	**Cán bộ chấm thi phúc khảo 2** *(ký và ghi rõ họ tên)*

MỤC LỤC

CHƯƠNG 1. GIỚI THIỆU 1
1.1. Lý do chọn đề tài 1
1.2. Mục tiêu đề tài 1
1.3. Phạm vi đề tài 1
1.4 Đối tượng nghiên cứu 1
1.5. Phương pháp nghiên cứu 2
1.6 Bố cục 2
CHƯƠNG 2. CƠ SỞ LÝ THUYẾT 3
2.1. Cơ sở lý thuyết Ngôn ngữ Python 3
2.2. Cơ sở lý thuyết Thư viện OpenCV 3
2.3. Cơ sở lý thuyết Thư viện Pillow (PIL) 4
2.4. Cơ sở lý thuyết Thư viện ImageIO 4
2.5. Cơ sở lý thuyết Nguyên lý tạo ảnh động (Animation) 4
2.6. Cơ sở lý thuyết Thư viện Tkinter 5
2.7. Cơ sở lý thuyết Threading 5
CHƯƠNG 3. PHÂN TÍCH HỆ THỐNG VÀ XÂY DỰNG SẢN PHẨM 6
3.1. Phân tích hệ thống 6
3.1.1 Mục tiêu và yêu cầu của hệ thống 6
3.1.2 Mô hình hoạt động của hệ thống 7
3.2. Xây dựng giao diện các chức năng sản phẩm 11
3.2.1 Giao diện chính (GUI) 11
KẾT LUẬN 20
Kết luận 20
Hướng phát triển 21
Tài liệu tham khảo 22
PHỤ LỤC 23
Link GitHub 23

DANH MỤC CÁC KÝ HIỆU, CHỮ VIẾT TẮT

Hình 3.1: Biểu đồ usecase 8
Hình 3.2: Biểu đồ hoạt động cho các chức năng 9
Hình 3.3: Biểu đồ tuần tự cho các chức năng..... 10
Hình 3.4: Giao diện tạo GIF và Video..... 11
Hình 3.5: Nút chọn ảnh ở Tab1 12
Hình 3.6: Nút Xem GIF ở Tab1..... 12
Hình 3.7: Nút Lưu GIF ở Tab1 12
Hình 3.8: Nút Tạo video góc trái ở Tab1..... 12
Hình 3.9: Tab xem Video đã lưu 13
Hình 3.10: Các nút chức năng Xem Video đã lưu 13
Hình 3.11: Nút Tạo GIF từ video 14
Hình 3.12: Hộp thoại Tạo GIF từ video 14
Hình 3.13: Nút Chọn Video 15
Hình 3.14: Chức năng cơ bản của hộp thoại Tạo GIF từ video 15
Hình 3.15: Chức năng cắt video trong khoảng từ điểm A tới B..... 15
Hình 3.16: Nút Tạo GIF..... 16
Hình 3.17: Nút Xóa danh sách..... 16
Hình 3.18: Các nút điều chỉnh GIF tạo từ ảnh 16
Hình 3.19: Khu vực chứa ảnh và các GIF đầu ra..... 17
Hình 3.20: Giao diện xuất các frame từ Video Input..... 18
Hình 3.21: Giao diện xuất các frame từ Video Input..... 18
Hình 3.22: Chọn Số lượng ảnh mỗi giây 18
Hình 3.23: Chọn thời lượng ảnh sẽ lấy..... 19
Hình 3.24: Chọn thư mục các frame sẽ lưu vào..... 19
Hình 3.25: Nút Xuất Frames 19
Hình 3.26: Các frame ảnh đã xuất 19

MỞ ĐẦU

Trong thời đại công nghệ số phát triển mạnh mẽ, việc xử lý và hiển thị hình ảnh động đóng vai trò quan trọng trong nhiều lĩnh vực như thiết kế đồ họa, truyền thông, và học tập. Đề tài **"Xây dựng ứng dụng tạo ảnh động GIF và video từ ảnh tĩnh"** nhằm giúp người dùng có thể dễ dàng kết hợp nhiều hình ảnh để tạo thành một chuỗi ảnh động, hỗ trợ xuất ra định dạng GIF hoặc MP4.Ứng dụng cũng bổ sung chức năng **trích xuất khung hình (frames)** từ video, giúp người dùng tái sử dụng hoặc chỉnh sửa nội dung hình ảnh nhanh chóng.

Mục tiêu của đề tài là xây dựng một công cụ đơn giản, dễ sử dụng, hỗ trợ người dùng (đặc biệt là sinh viên ngành công nghệ thông tin và thiết kế đồ họa) có thể tạo ra ảnh động phục vụ học tập, minh họa hoặc trình bày sản phẩm một cách trực quan, sinh động.

CHƯƠNG 1. GIỚI THIỆU

1.1. Lý do chọn đề tài

Việc tạo ảnh động hoặc video từ ảnh tĩnh thường yêu cầu phần mềm chuyên dụng, vốn phức tạp và tốn thời gian. Với Python và thư viện mã nguồn mở, có thể tạo ra công cụ trực quan, nhẹ, dễ sử dụng mà vẫn đáp ứng tốt nhu cầu của người dùng cá nhân.

1.2. Mục tiêu đề tài

- Xây dựng **ứng dụng Python có giao diện đồ họa (GUI)** giúp người dùng:
    - Tạo **ảnh GIF** từ nhiều hình ảnh.
    - Xuất **video MP4** từ ảnh.
    - **Xem trước** GIF và video ngay trong ứng dụng.
    - **Trích xuất khung hình từ video** (extract frames).
- Ứng dụng chạy độc lập, không cần cài đặt môi trường phức tạp.

1.3. Phạm vi đề tài

- **Phạm vi không gian:** Ứng dụng chạy trên môi trường **máy tính cá nhân (desktop)**, có thể thực thi trên các hệ điều hành Windows, Linux hoặc macOS.
- **Phạm vi thời gian:** Thực hiện trong một học kỳ của môn học *Xử lý ảnh và Thị giác máy tính*.
- **Phạm vi kỹ thuật:**
    - Ngôn ngữ lập trình: **Python**
    - Thư viện chính: **Tkinter, Pillow, imageio, opencv-python (cv2), threading.**
    - Xuất đầu ra: **GIF** hoặc **video MP4**
    - Không tập trung vào xử lý video phức tạp, deep learning hay mô hình 3D.

1.4 Đối tượng nghiên cứu

- Các phương pháp xử lý ảnh trong Python.
- Cách tạo ảnh động từ chuỗi ảnh.
- Cách đọc/ghi video bằng OpenCV và ImageIO.
- Thiết kế giao diện bằng Tkinter.

1.5. Phương pháp nghiên cứu

- **Phương pháp thu thập thông tin:**
  Tìm hiểu tài liệu về xử lý ảnh động, tham khảo các thư viện Python chuyên dụng như OpenCV, Pillow, imageio, Tkinter.

- **Phương pháp xử lý thông tin:**
  Phân tích nguyên lý hoạt động của các công cụ tạo ảnh động; so sánh hiệu suất giữa các phương pháp đọc/ghi ảnh; lựa chọn cấu trúc chương trình phù hợp với mục tiêu đề tài.

- **Phương pháp thực nghiệm:**
  Tiến hành lập trình thử nghiệm, kiểm thử các hàm xử lý chuỗi ảnh, tối ưu thời gian xuất ảnh động; khảo sát người dùng thử nghiệm (sinh viên) về mức độ dễ sử dụng và chất lượng kết quả.

1.6 Bố cục

Phần còn lại của báo cáo tiểu luận môn học này được tổ chức như sau:

Chương 2: *Cơ sở lý thuyết* — trình bày các khái niệm và công nghệ sử dụng, bao gồm xử lý ảnh bằng OpenCV/Pillow, nguyên lý hoạt động của ảnh động GIF, khái niệm khung hình (frame) và tốc độ phát (FPS), cùng tổng quan các định dạng video/ảnh động phổ biến.

Chương 3: *Phân tích và thiết kế hệ thống* — mô tả chức năng chương trình, sơ đồ use case, thiết kế giao diện, và luồng xử lý dữ liệu giữa các mô-đun.

Chương 4: *Kết luận và hướng phát triển* — tổng kết kết quả đạt được, hạn chế còn tồn tại, và đề xuất các hướng mở rộng như thêm hiệu ứng nâng cao, nhận dạng chuyển động tự động hoặc kết hợp AI để tạo animation thông minh.

CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

Đề tài "Xây dựng chương trình tạo ảnh động" thuộc lĩnh vực xử lý ảnh số và thị giác máy tính, do đó cần ứng dụng nhiều kiến thức liên quan đến việc biểu diễn, thao tác và tổng hợp ảnh. Để hiện thực hóa hệ thống, nhóm lựa chọn sử dụng ngôn ngữ Python cùng các thư viện mã nguồn mở chuyên dụng cho xử lý ảnh và video như OpenCV, Pillow (PIL) và ImageIO.

Lý do lựa chọn các cơ sở lý thuyết này:

- Python là ngôn ngữ dễ học, có cộng đồng lớn và hỗ trợ mạnh mẽ trong lĩnh vực xử lý ảnh, AI, và thị giác máy tính.
- OpenCV cung cấp các công cụ xử lý ảnh nhanh, mạnh và hiệu quả.
- Pillow hỗ trợ thao tác ảnh tĩnh, đọc/ghi nhiều định dạng và vẽ lên ảnh dễ dàng.
- ImageIO hỗ trợ đọc/ghi ảnh động (GIF) và video, rất phù hợp cho phần xuất sản phẩm của đề tài.

2.1. Cơ sở lý thuyết Ngôn ngữ Python

Python được phát triển bởi **Guido van Rossum** vào cuối những năm 1980 và chính thức phát hành năm 1991. Python nổi tiếng nhờ cú pháp đơn giản, dễ đọc và thư viện phong phú, trở thành lựa chọn hàng đầu cho các lĩnh vực như **xử lý ảnh, thị giác máy tính, trí tuệ nhân tạo và khoa học dữ liệu**.

Đặc điểm nổi bật:

- Cú pháp dễ hiểu, phù hợp cho người mới bắt đầu.
- Cộng đồng người dùng và tài nguyên học tập phong phú.
- Có thư viện mạnh cho xử lý ảnh/video như OpenCV, Pillow, Numpy, ImageIO, Matplotlib,…
- Dễ tích hợp với giao diện đồ họa (Tkinter, PyQt) hoặc môi trường web (Flask, Django).

2.2. Cơ sở lý thuyết Thư viện OpenCV

OpenCV (Open Source Computer Vision Library) là một thư viện mã nguồn mở ra đời vào năm 1999 bởi Intel, sau đó được hỗ trợ bởi Willow Garage và Itseez (sau này thuộc Intel). Thư viện này cung cấp hàng trăm hàm phục vụ xử lý ảnh, nhận dạng vật thể, thị giác máy tính và học sâu.

Các chức năng chính của OpenCV:

- Đọc, ghi và hiển thị ảnh/video.
- Thay đổi kích thước, cắt, xoay, lật ảnh.
- Áp dụng các phép biến đổi hình học và màu sắc.

- Lọc ảnh (Gaussian, Median, Bilateral).
- Xử lý biên, phát hiện vật thể, trích chọn đặc trưng.

2.3. Cơ sở lý thuyết Thư viện Pillow (PIL)

Pillow là phiên bản nâng cấp của **Python Imaging Library (PIL)** – thư viện xử lý ảnh ra đời từ năm 1995. Pillow hỗ trợ đọc, ghi, chỉnh sửa và hiển thị nhiều định dạng ảnh khác nhau (JPEG, PNG, BMP, GIF,…).

Các chức năng chính:

- Đọc và ghi ảnh (Image.open(), Image.save()).
- Chuyển đổi định dạng ảnh.
- Thao tác pixel (lấy giá trị, thay đổi màu sắc).
- Vẽ văn bản hoặc hình dạng lên ảnh (ImageDraw).
- Hỗ trợ tạo và lưu ảnh GIF.

2.4. Cơ sở lý thuyết Thư viện ImageIO

ImageIO là thư viện Python chuyên dùng để đọc và ghi nhiều định dạng ảnh, video và ảnh động (GIF, MP4, AVI, TIFF, v.v.). Thư viện này rất gọn nhẹ, dễ dùng và thường được sử dụng trong các ứng dụng xử lý ảnh nhanh.

Các chức năng chính :

- Đọc và ghi ảnh, video, ảnh động.
- Hỗ trợ ghi ảnh động từ danh sách ảnh (imageio.mimsave()).
- Tương thích với NumPy để thao tác ma trận ảnh.
- Hỗ trợ xuất video nhiều định dạng thông qua FFMPEG.

2.5. Cơ sở lý thuyết Nguyên lý tạo ảnh động (Animation)

Ảnh động (animated image) là chuỗi các ảnh tĩnh (frame) được hiển thị liên tiếp với tốc độ nhất định, tạo cảm giác chuyển động liên tục. Mỗi khung hình có thể có thời gian hiển thị riêng (frame duration).

Nguyên lý hoạt động

- Chuỗi ảnh được sắp xếp theo thứ tự thời gian.
- Mỗi ảnh được hiển thị trong một khoảng thời gian ngắn (ví dụ: 0.1 giây).
- Khi lặp lại liên tục, mắt người cảm nhận như vật thể đang chuyển động.
- Tốc độ hiển thị (FPS – frame per second) quyết định độ mượt của ảnh động.

2.6. Cơ sở lý thuyết Thư viện Tkinter

Tkinter là thư viện tiêu chuẩn của Python dùng để xây dựng **giao diện đồ họa người dùng** Vì Tkinter được tích hợp sẵn trong Python, nên khi cài Python người dùng không cần cài thêm gói phụ nào khác.

- **Tích hợp sẵn trong Python:** Không cần cài đặt thêm thư viện ngoài.
- **Đa nền tảng:** Chạy được trên Windows, macOS và Linux.
- **Dễ học, dễ dùng:** Cấu trúc rõ ràng, hỗ trợ lập trình hướng đối tượng.
- **Hỗ trợ nhiều widget:** Cho phép tạo các thành phần giao diện như nút bấm, nhãn, hộp nhập liệu, khung vẽ hình ảnh, thanh cuộn,…
- **Có thể kết hợp với các thư viện khác:** như **Pillow, OpenCV, imageio, threading** để xử lý ảnh, hiển thị video, hoặc thực thi đa luồng.

2.7. Cơ sở lý thuyết Threading

Là thư viện tiêu chuẩn dùng để xử lý đa luồng (multithreading). Mục tiêu chính của nó là cho phép chương trình thực hiện nhiều tác vụ cùng lúc, giúp tăng tốc độ phản hồi và hiệu suất, đặc biệt trong các ứng dụng có giao diện đồ họa (GUI) hoặc cần thực hiện nhiều công việc song song (như xử lý ảnh, tải dữ liệu, phát video,…).

- **Thread (Luồng):** Là một đơn vị nhỏ nhất của quá trình thực thi trong chương trình.
- **Main thread:** Luồng chính – nơi chương trình bắt đầu chạy.
- **Worker thread:** Các luồng phụ, thực hiện tác vụ song song mà không ảnh hưởng đến luồng chính

CHƯƠNG 3. PHÂN TÍCH HỆ THỐNG VÀ XÂY DỰNG SẢN PHẨM

3.1. Phân tích hệ thống

Hệ thống được xây dựng nhằm **xử lý ảnh và tạo ảnh động (GIF)** từ các chuỗi ảnh tĩnh, phục vụ cho các mục đích như trình diễn, minh họa hoặc thị giác máy tính.
Ứng dụng tập trung vào việc cho phép người dùng **nhập nhiều ảnh đầu vào**, **xử lý chúng** (như thay đổi kích thước, áp dụng bộ lọc, chuyển đổi định dạng, v.v.), và cuối cùng **ghép lại thành một ảnh động (GIF)** có thể lưu hoặc xem trực tiếp.

3.1.1 Mục tiêu và yêu cầu của hệ thống

- **Yêu cầu chức năng:**

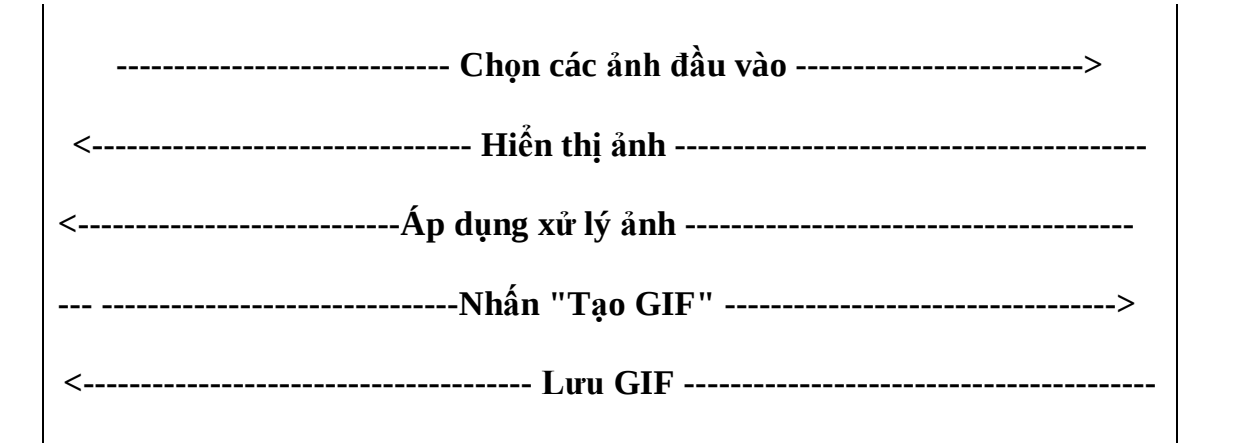
  - Cho phép người dùng chọn nhiều ảnh từ thư mục.
  - Hiển thị ảnh xem trước
  - Chọn số khung hình/giây (FPS), hiệu ứng chuyển cảnh (**fade, slide, none**).
  - Xem thử ảnh động (GIF Preview).
  - Lưu GIF hoặc xuất thành video MP4.
  - Chọn video và trích xuất khung hình theo FPS mong muốn.
  - Xem trước danh sách ảnh được trích ra.
  - Xuất các frame từ video đầu vào (dựa vào số giây trong video mà chuyển tuỳ theo người dùng chọn 12,24,60 FPS)

- **Yêu cầu phi chức năng:**

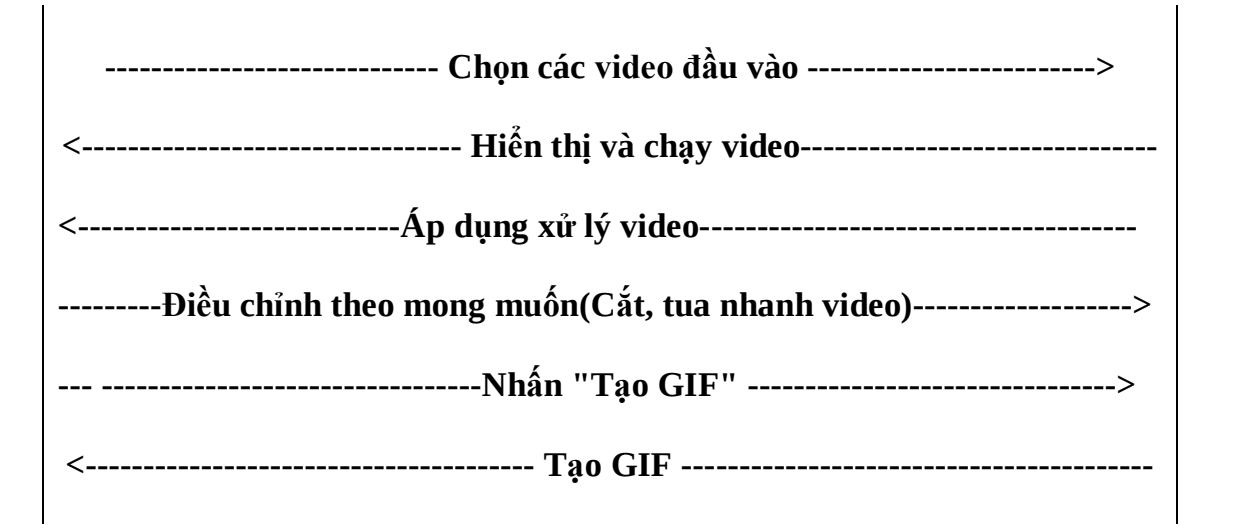
  - Ứng dụng thân thiện, dễ sử dụng.
  - Xử lý nhanh, không bị treo giao diện.
  - Kết quả xuất ra đúng định dạng, chất lượng tốt.

3.1.2 Mô hình hoạt động của hệ thống

Luồng hoạt động của tạo GIF từ ảnh:



Luồng hoạt động của tạo GIF từ Video:



Luồng hoạt động của tạo các ảnh từ Video:

Người dùng **Hệ thống**

---------- Chọn video đầu vào ---------->

----------Chọn số lượng ảnh tạo ra mỗi giây của video FPS---------->

----------Chọn thời lượng (s) để tạo ảnh---------->

--- ----------Chọn thư mục lưu các frame ảnh ---------->

---------- Nhấn "Xuất frame" ---------->

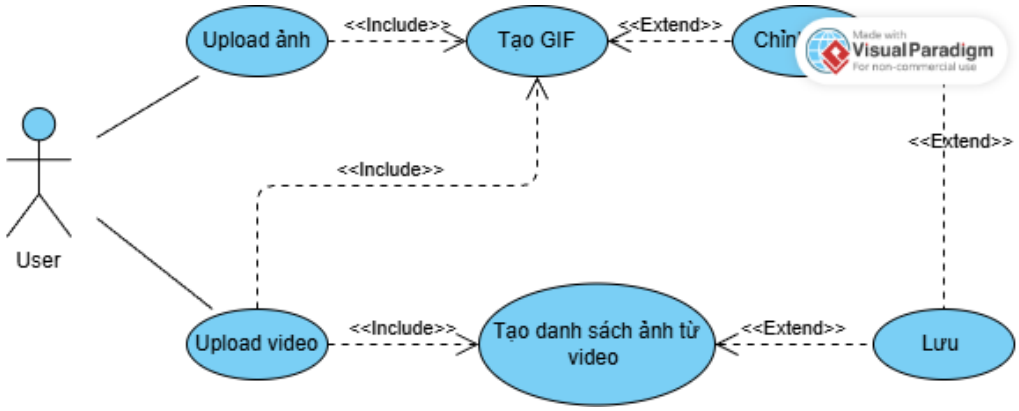
<---------- Trả về số ảnh đã lưu trong thư mục ----------

3.1.3 Tác nhân

Các chức năng mà **User** tương tác

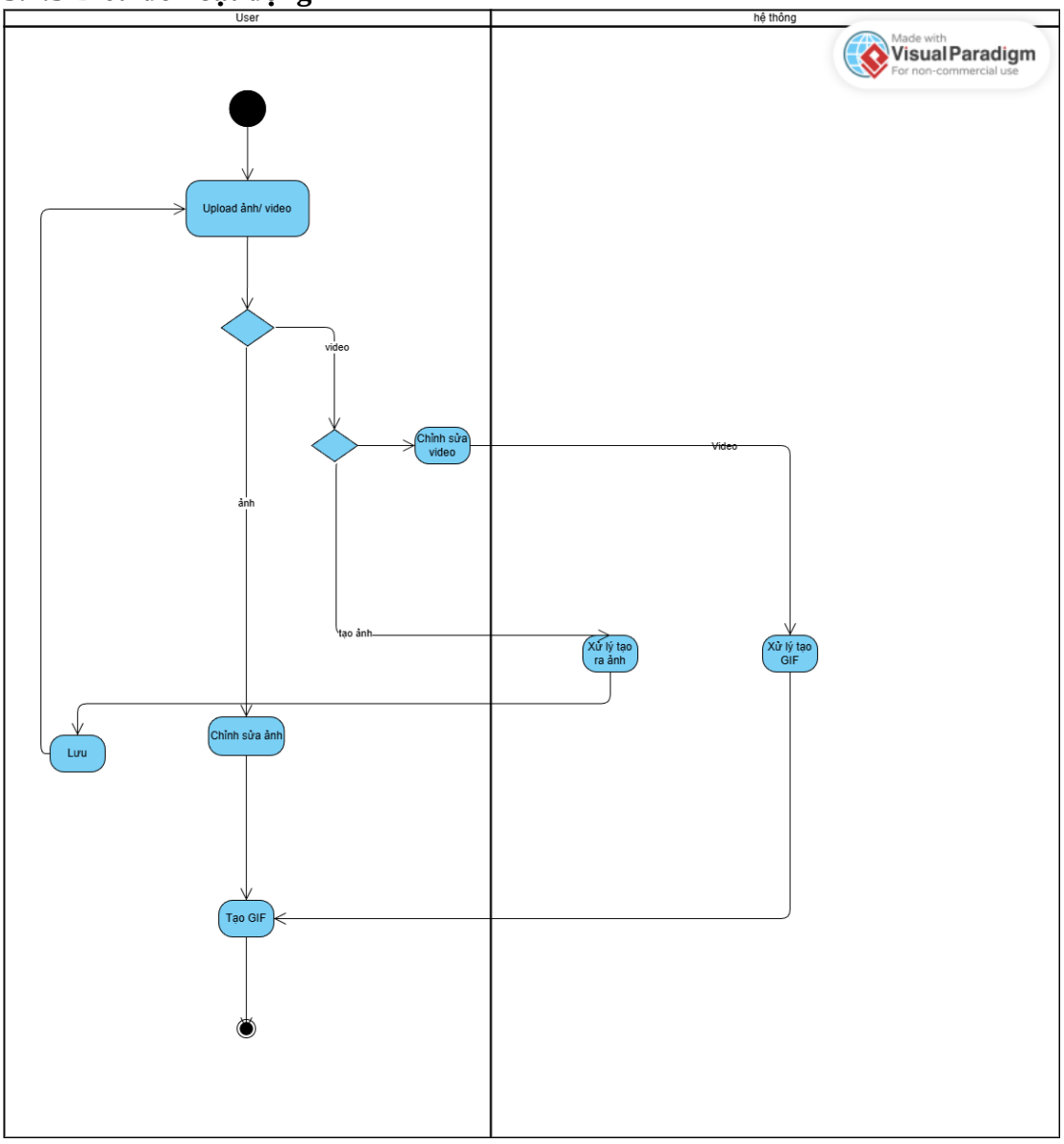
- Upload ảnh / video
- Chỉnh sửa
- Xóa danh sách đầu vào
- Tạo GIF và Lưu GIF

3.1.4 Biểu đồ use-case



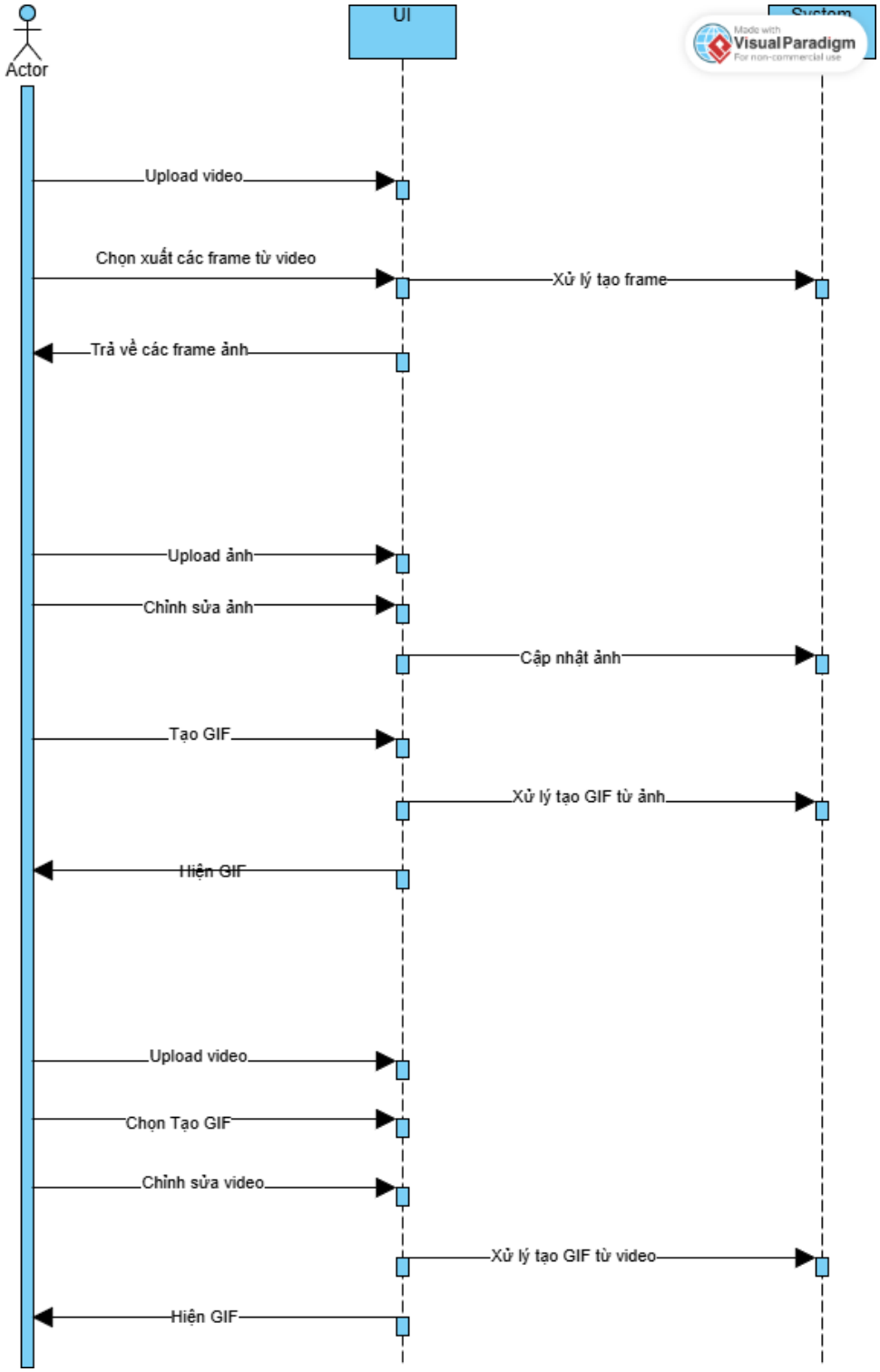
Hình 3.1: Biểu đồ usecase

3.1.5 Biểu đồ hoạt động



Hình 3.2: Biểu đồ hoạt động cho các chức năng

3.1.6 Biểu đồ hoạt động

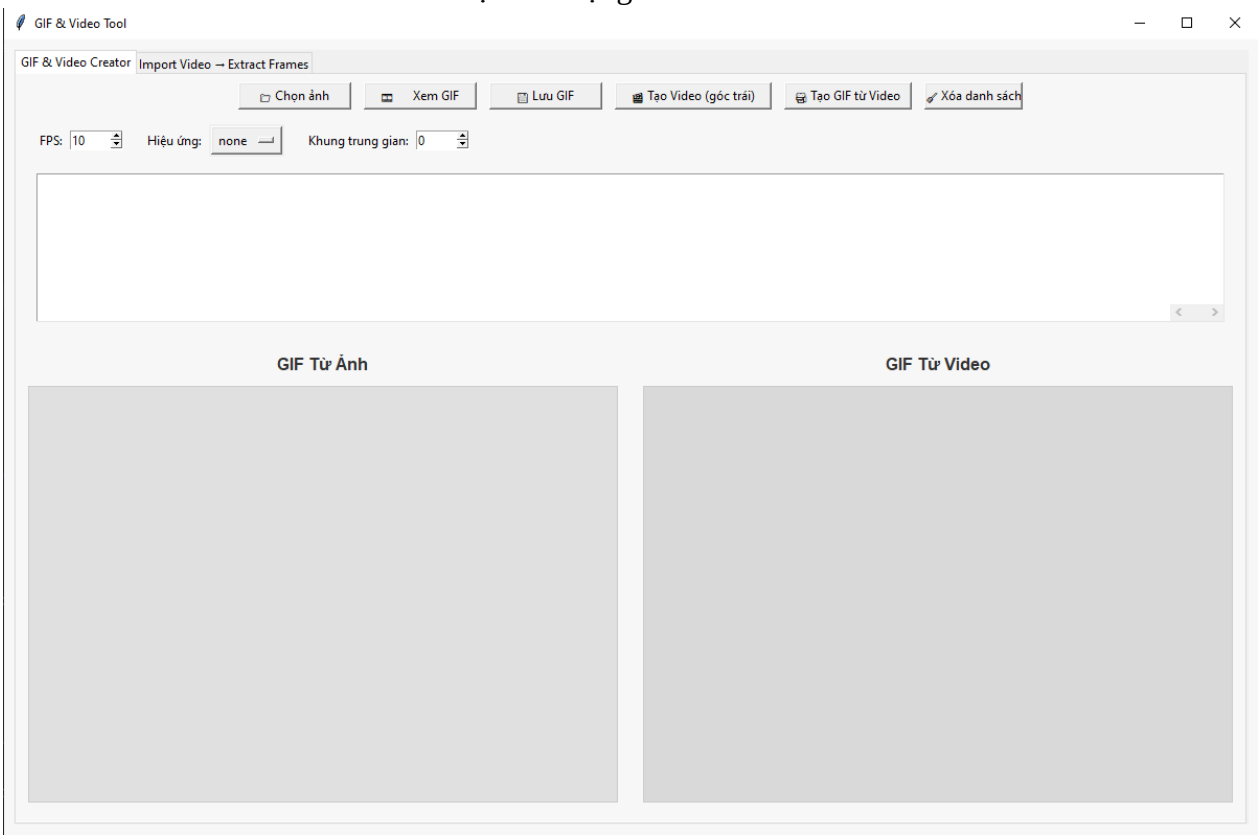


Hình 3.3: Biểu đồ tuần tự cho các chức năng

3.2. Xây dựng giao diện các chức năng sản phẩm

3.2.1 Giao diện chính (GUI)

- **Tab 1:** "GIF & Video Creator" – tạo ảnh động và video.



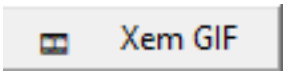
Hình 3.4: Giao diện tạo GIF và Video

- Các nút chức năng :
- Nút Chọn ảnh
- Xem GIF
- Lưu GIF
- Tạo Video(góc trái)
- Tạo GIF từ Video
- Xóa danh sách
- Nơi chọn FPS
- Khung trung gian
- Khu vực xem các ảnh đầu vào
- Khu vực xem GIF tạo từ ảnh (bên trái),GIF từ video(bên phải)

*Các nút điều khiển ở **Tab1**:

Hình 3.5: Nút chọn ảnh ở Tab1

- **Có thể load ảnh từ máy tính (tối thiểu ít nhất 2 ảnh)**



Hình 3.6: Nút Xem GIF ở Tab1

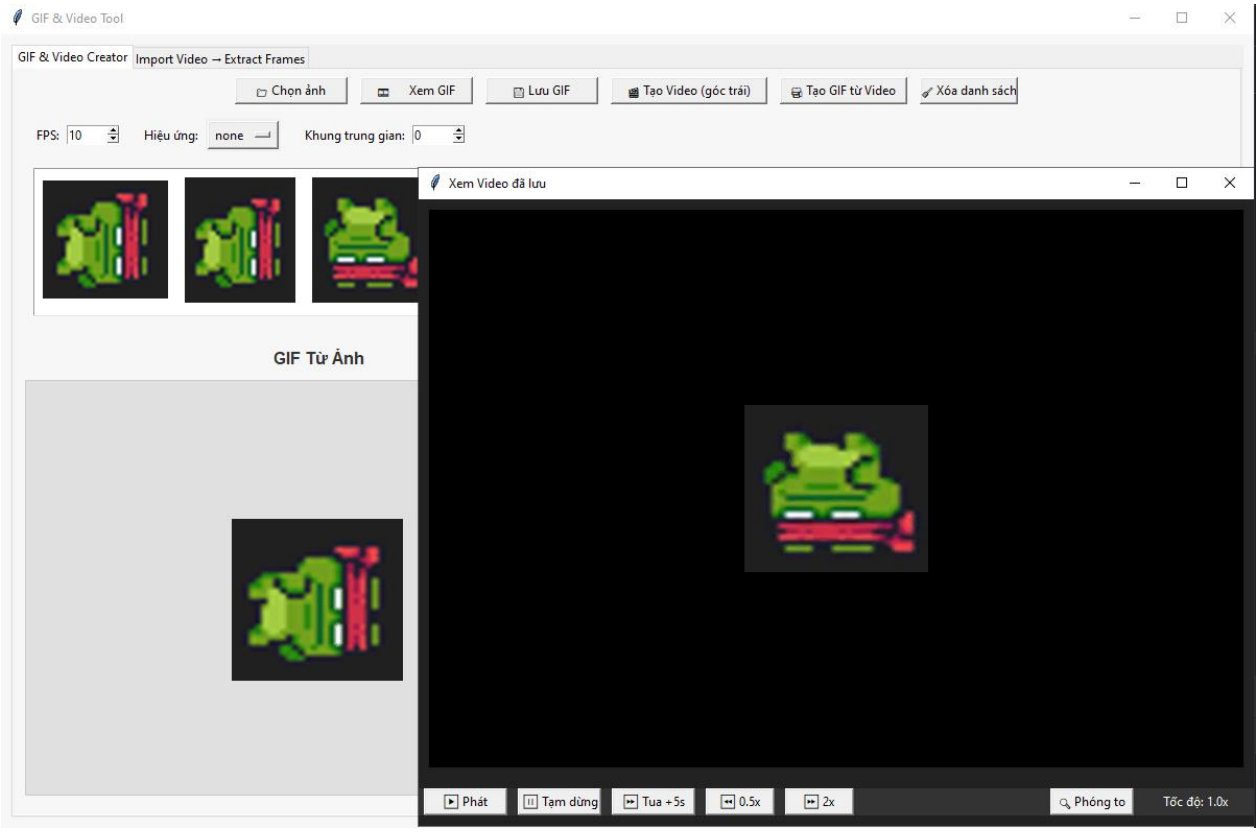
- **Xem được GIF đã qua chỉnh sửa do mình tạo ra , hiện ở góc trái**

Hình 3.7: Nút Lưu GIF ở Tab1

- **Lưu GIF ở góc trái vào thư mục tự chọn**

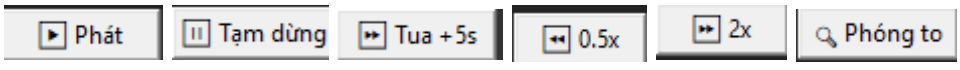
Hình 3.8: Nút Tạo video góc trái ở Tab1

- **Tạo Video(góc trái) – Tạo và đồng thời hiện tab xem Video đã lưu**



Hình 3.9: Tab xem Video đã lưu

***Gồm các nút điều chỉnh video đã tạo từ các tệp ảnh đầu vào**

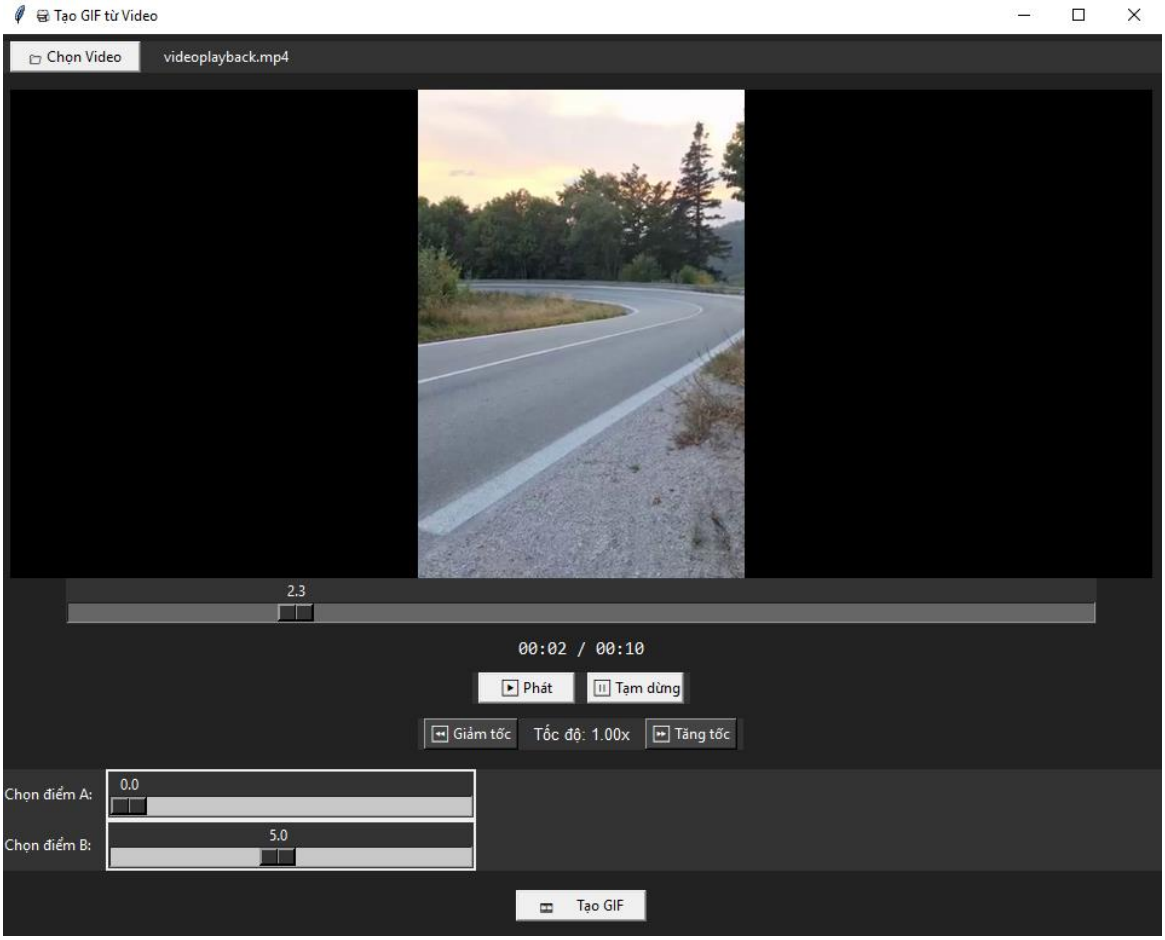


Hình 3.10: Các nút chức năng Xem Video đã lưu

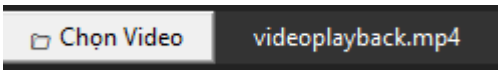
- Chạy video
- Dừng video
- Tua 5 giây video
- Lùi lại 5 giây video
- Chỉnh tốc độ 1x, 2x ,4x
- Phóng to video

Hình 3.11: Nút Tạo GIF từ video

- **Tạo GIF từ video – sẽ hiện ra 1 hộp thoại Tạo GIF từ video**

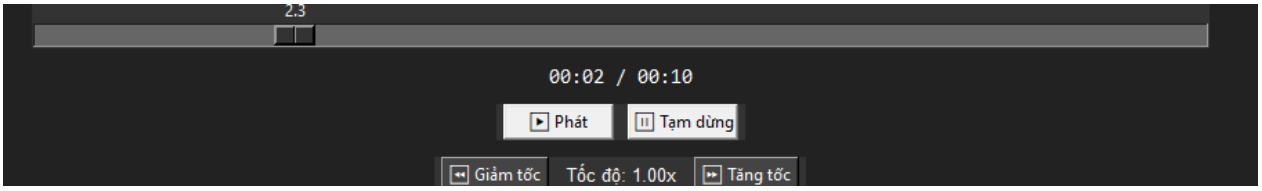


Hình 3.12: Hộp thoại Tạo GIF từ video



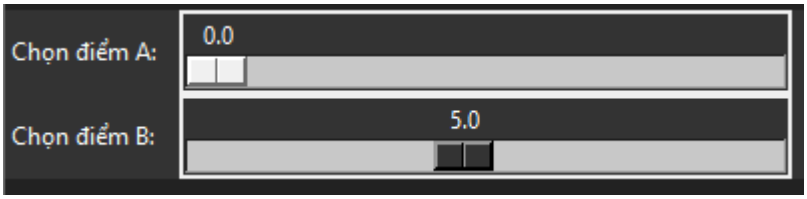
Hình 3.13: Nút Chọn Video

- **Nút chọn video (Tên video sẽ hiện ra ở kế bên)**



Hình 3.14: Chức năng cơ bản của hộp thoại Tạo GIF từ video

- **Khu vực nơi video chạy**
- **Thanh trượt thời gian của video**
- **Nút Phát và Tạm Dừng video**
- **Nút giảm và tăng tốc video**



Hình 3.15: Chức năng cắt video trong khoảng từ điểm A tới B

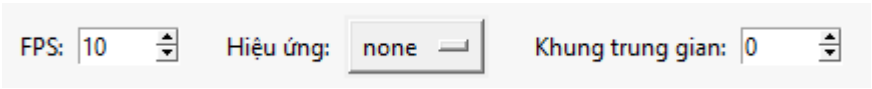
- **Chọn điểm A: Nút bắt đầu điểm cắt**
- **Chọn điểm B: Nút kết thúc điểm cắt**
- **Điều kiện là A luôn nhỏ hơn B**

Hình 3.16: Nút Tạo GIF

- **Tạo GIF – sẽ hiện ra thư mục để lưu như chức năng đã kể trên**

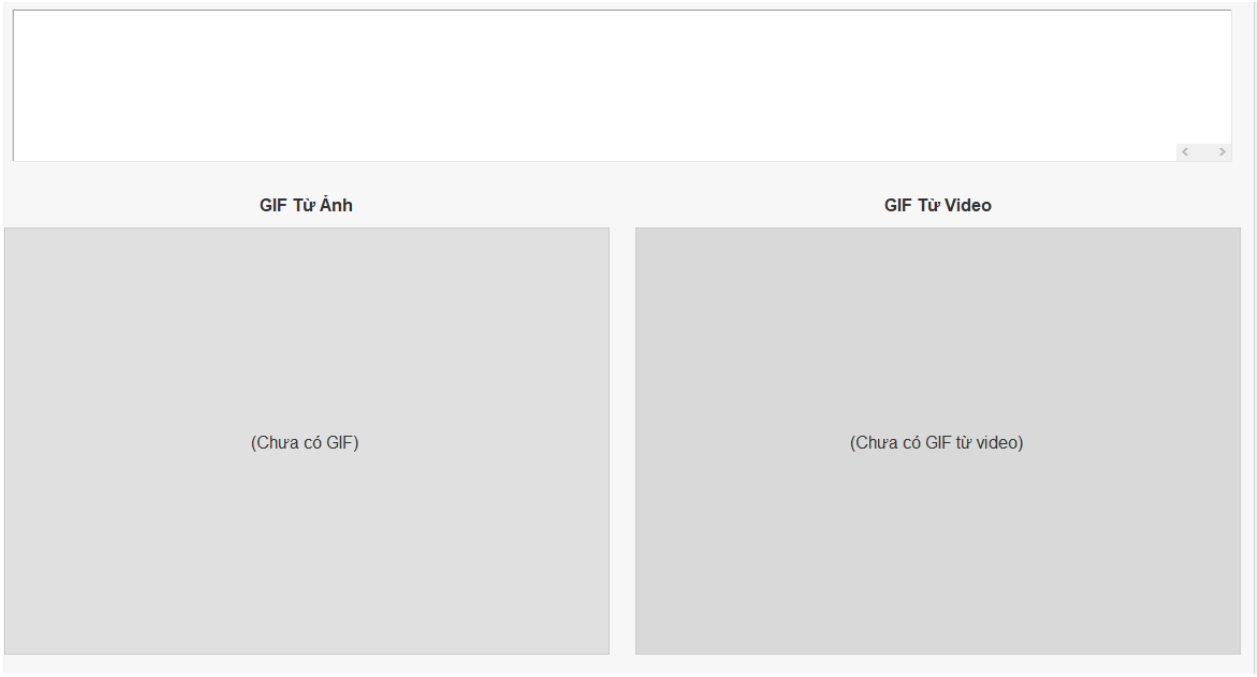
Hình 3.17: Nút Xóa danh sách

- **Xóa danh sách – xoá đầu vào các GIF ảnh/video đã tạo**



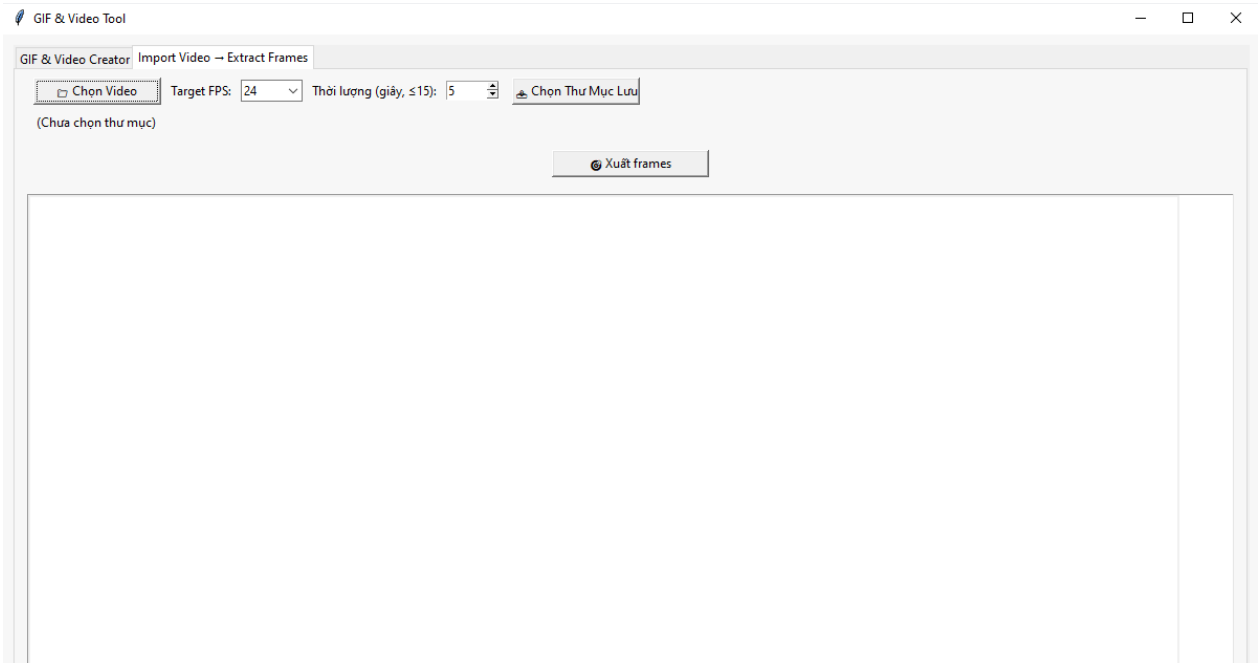
Hình 3.18: Các nút điều chỉnh GIF tạo từ ảnh

- FPS: số lượng ảnh hiện ra mỗi giây
- Hiệu ứng : Không hiệu ứng, Fade, Slide
- Khi áp dụng hiệu ứng thì khung trung gian sẽ được sử dụng
- *Khung trung gian là khoảng các giữa các ảnh với nhau VD: Ảnh 1 và Ảnh 2
- Nhập 10 khung thì giữa 2 ảnh đó sẽ hiện 10 khung



Hình 3.19: Khu vực chứa ảnh và các GIF đầu ra

- **Tab 2**: "Import Video → Extract Frames" – trích xuất ảnh từ video.

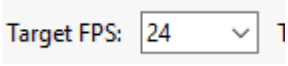


Hình 3.20: Giao diện xuất các frame từ Video Input

*Các nút điều khiển ở **Tab2**:

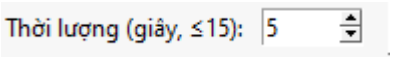
Hình 3.21: Giao diện xuất các frame từ Video Input

- **Chọn video – chọn video từ thư mục**



Hình 3.22: Chọn Số lượng ảnh mỗi giây

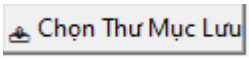
- **Chọn FPS(Frame mỗi giây) – chọn 12,24,60 FPS**



Hình 3.23: Chọn thời lượng ảnh sẽ lấy

- **Thời lượng của video (Giới hạn ở 15s)**
- **Thời lượng luôn bắt đầu từ 0s trở đi – thời gian và lượng frame chọn càng lớn thời gian tạo càng lâu theo công thức**

  Số lượng ảnh = FPS * thời lượng chọn (VD 10FPS * 5s = 50 tấm ảnh)

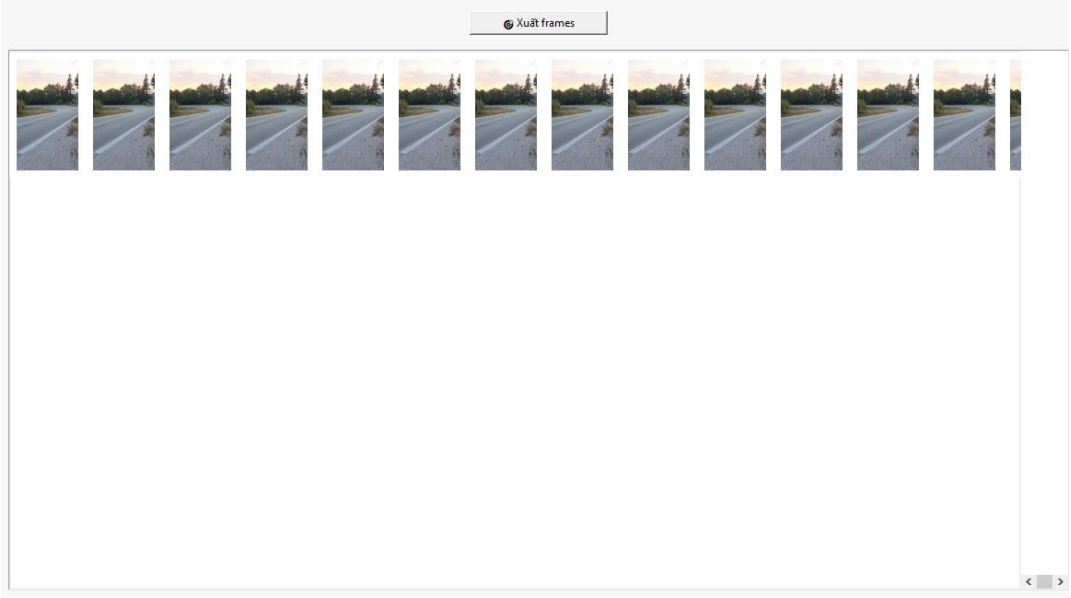


Hình 3.24: Chọn thư mục các frame sẽ lưu vào

- **Chọn thư mục Lưu các frame**
- **Khi đã lưu hoàn tất các ảnh sẽ hiện thông báo**

Hình 3.25: Nút Xuất Frames

- **Xuất các ảnh vào thư mục**
- **Đồng thời hiện thị lên khung ảnh đã xuất**



Hình 3.26: Các frame ảnh đã xuất

- **Khung hiển thị ảnh**

KẾT LUẬN

Kết luận

Đề tài **"Xây dựng chương trình tạo ảnh động"** đã hoàn thành mục tiêu đề ra là xây dựng được một ứng dụng xử lý ảnh cơ bản, cho phép người dùng:

Tải lên nhiều ảnh từ máy tính.

Áp dụng các thao tác xử lý ảnh cơ bản như chuyển ảnh sang thang xám, lọc nhiễu, thay đổi kích thước, v.v.

Tự động ghép chuỗi ảnh đã xử lý thành một ảnh động định dạng **GIF**.

Hiển thị kết quả trực quan ngay trên giao diện đồ họa thân thiện sử dụng **Tkinter**.

Chương trình được xây dựng hoàn toàn bằng **Python**, tận dụng sức mạnh của các thư viện **OpenCV**, **Pillow** và **ImageIO**, giúp xử lý và tạo ảnh động một cách nhanh chóng và chính xác. Qua quá trình triển khai, nhóm (hoặc sinh viên) đã:

Nắm vững quy trình đọc – xử lý – hiển thị ảnh bằng Python.

Hiểu rõ cách chuyển đổi giữa định dạng ảnh OpenCV (NumPy array) và ảnh hiển thị giao diện (PIL Image).

Thiết kế giao diện người dùng trực quan, dễ thao tác cho cả người không chuyên về kỹ thuật.

So với các công cụ tạo ảnh động phổ biến hiện nay, sản phẩm của đề tài tuy đơn giản nhưng có ưu điểm ở chỗ: mã nguồn mở, dễ mở rộng, và cho phép tích hợp thêm nhiều kỹ thuật xử lý ảnh khác nhau tùy nhu cầu học tập và nghiên cứu.

Trong quá trình thực hiện, đề tài đã đạt được hầu hết các mục tiêu ban đầu, tuy nhiên vẫn còn một số hạn chế như:

- Chưa hỗ trợ nhiều hiệu ứng chuyển cảnh giữa các ảnh.
- Chưa cho phép xem trước ảnh động trước khi lưu.
- Giao diện còn đơn giản, chưa tối ưu cho nhiều độ phân giải màn hình.

Thông qua quá trình nghiên cứu và lập trình, sinh viên đã tích lũy được nhiều kinh nghiệm thực tiễn về xử lý ảnh, lập trình Python, tư duy thiết kế phần mềm, cũng như khả năng tự học, tìm hiểu tài liệu kỹ thuật.

Hướng phát triển

Trong tương lai, đề tài có thể được mở rộng và hoàn thiện theo các hướng sau:

1. **Bổ sung hiệu ứng xử lý ảnh nâng cao:**
    - Thêm các bộ lọc làm mịn, làm nét, phát hiện biên, biến đổi màu, v.v.
    - Cho phép người dùng tự chọn loại hiệu ứng trước khi tạo ảnh động.
2. **Thêm các hiệu ứng chuyển động động học:**
    - Hỗ trợ hiệu ứng **fade-in / fade-out**, **zoom-in**, **slide**, hoặc **morphing** giữa các khung hình để tạo ảnh động mượt mà và chuyên nghiệp hơn.
3. **Xem trước ảnh động (Preview GIF):**
    - Trước khi lưu, người dùng có thể xem trước ảnh GIF trong cửa sổ giao diện, giúp dễ điều chỉnh tốc độ hoặc thứ tự ảnh.
4. **Tùy chỉnh tốc độ khung hình (Frame Duration):**
    - Thêm thanh điều chỉnh tốc độ phát hoặc lựa chọn khoảng thời gian giữa các ảnh.
5. **Nâng cấp giao diện người dùng (UI/UX):**
    - Thiết kế lại bằng thư viện **Tkinter nâng cao (ttk)** hoặc **PyQt5 / PySide6** để có giao diện hiện đại hơn, hỗ trợ nhiều kích cỡ màn hình.
6. **Tích hợp công nghệ trí tuệ nhân tạo (AI):**
    - Ứng dụng AI để tự động chọn ảnh phù hợp theo nội dung, nhận diện khuôn mặt hoặc tạo hoạt ảnh dựa trên chuyển động.
    - Dùng mô hình học sâu (Deep Learning) để sinh ảnh động từ video ngắn hoặc chuỗi ảnh chụp liên tiếp.
7. **Triển khai ứng dụng lên Web hoặc Mobile:**
    - Chuyển chương trình sang nền tảng web bằng **Flask / Django** hoặc tích hợp vào ứng dụng di động với **Kivy** hoặc **React Native + Python API**, giúp người dùng thao tác mọi lúc mọi nơi.

Tài liệu tham khảo

[1] “OpenCV – Open Source Computer Vision Library,” *OpenCV Documentation,* [Online]. Available: https://opencv.org/. [Accessed: 16-Oct-2025].

[2] “Pillow – The friendly PIL fork (Python Imaging Library),” *Pillow Documentation,* [Online]. Available: https://pillow.readthedocs.io/. [Accessed: 16-Oct-2025].

[3] ChatGPT Available: https://chatgpt.com /. [Accessed: 16-Oct-2025].

PHỤ LỤC

Link GitHub

Duckkeip/xulyanh2