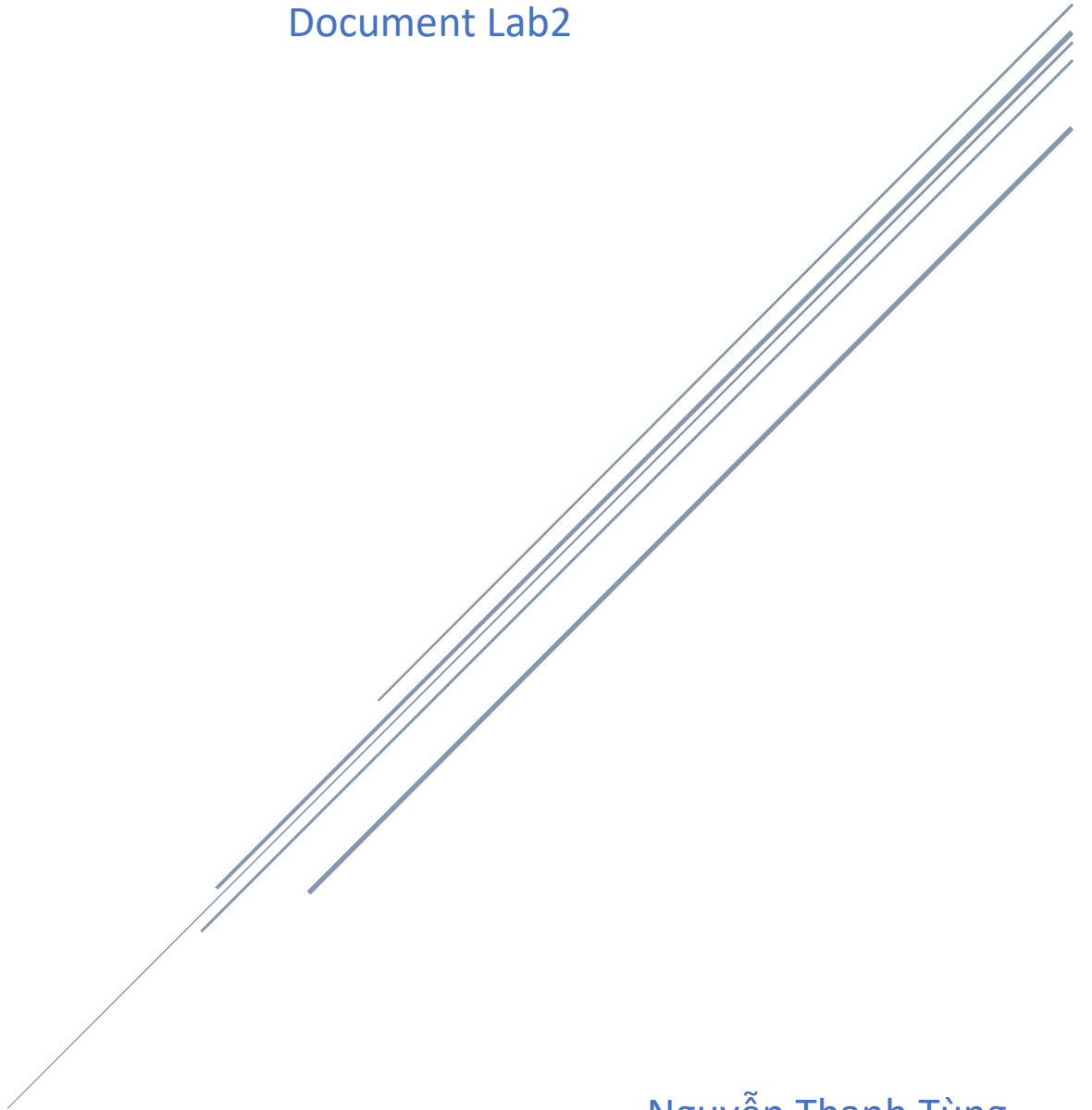


Trường đại học Khoa Học Tự Nhiên
Thành phố Hồ Chí Minh

XLAV

Document Lab2



Nguyễn Thanh Tùng
19127618

Contents

I. Thông tin sinh viên	1
II. Mô tả các hàm	1
III. Hướng dẫn sử dụng.....	5
IV. Tham khảo:.....	6

I. Thông tin sinh viên

1. MSSV: 19127618
2. Họ tên: Nguyễn Thanh Tùng

II. Mô tả các hàm

Class	Function	Description
Bilinear Interpolate	Interpolate	<p>Hàm nội suy màu theo phương pháp song tuyến tính.</p> <p>- Áp dụng công thức có trong slide lý thuyết: chiếu điểm a' của ảnh f' lên ảnh f ta được điểm a'' (chứa tọa độ thực), chiếu a'' lên 2 trục song song được 2 điểm $(x, k+1)$ và (x, k)</p> $\rightarrow f'(x', y') = bf(x, k + 1) + (1 - b)f(x, k)$ <p>- Tiếp tục nội suy 2 điểm này với 4 điểm quanh a'' ta được:</p> $f'(x', y') = (1 - a)(1 - b)f(l, k) + a(1 - b)f(l + 1, k) + (1 - a)bf(l, k + 1) + abf(l + 1, k + 1)$ <p>Với $l = \text{round}(x); k = \text{round}(y); a = x - l; b = y - k$</p> <p>- Phải đặt chặn trên và chặn dưới cho các giá trị sau khi tính toán vì sẽ có trường hợp tràn $[0, 255]$.</p>
Nearest Neighbor Interpolate	Interpolate	<p>Hàm nội suy màu theo phương pháp láng giềng gần.</p> <p>- Thuật toán cực kỳ đơn giản: sau khi chiếu x' lên ảnh f thì lấy điểm nguyên của f gần x' nhất.</p>

		<p>Với $l = \text{round}(x); k = \text{round}(y) \rightarrow f'(x', y')$ $= f(l, k)$</p> <p>- Ở phương pháp này không cần đặt chặn trên và dưới vì điểm ảnh cần xác định sẽ lấy thông tin của điểm ảnh gần nó nhất, mà ở ảnh gốc mọi điểm ảnh thuộc $[0, 255]$ thì điểm ảnh cũng sẽ sao chép những giá trị thuộc $[0, 255]$.</p>
Affine Transform	Translate	Hàm tạo matrix transform cho phép tịnh tiến theo vector (dx, dy) .
		Theo tài liệu matrix transform của hàm Translate sẽ được tính bằng cách nhân với matrix $\begin{bmatrix} 1 & 0 & d_x \\ 0 & 1 & d_y \\ 0 & 0 & 1 \end{bmatrix}$ (d_x, d_y) là vector tịnh tiến
	Rotate	Hàm tạo matrix transform cho phép xoay 1 góc angle
		Theo tài liệu matrix transform của hàm Translate sẽ được tính bằng cách nhân với matrix $\begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$ θ là góc xoay tuyến vào
	Scale	Hàm tạo matrix transform cho phép tỉ lệ theo hệ số
		Theo tài liệu matrix transform của hàm Translate sẽ được tính bằng cách nhân với matrix $\begin{bmatrix} c_x & 0 & 0 \\ 0 & c_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$ c_x, c_y là hệ số scale
	TransformPoint	Hàm transform 1 điểm $(x, y) \rightarrow (x', y')$ theo matrix transform đã có.
		Theo tài liệu matrix transform của hàm Translate sẽ được tính bằng cách nhân với matrix $\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$, với điểm x tại $(0, 0)$ và y tại $(0, 1)$ là điểm sau biến đổi.
	flipHorizontally	Hàm tạo matrix transform cho phép lật ngang
		Theo tài liệu matrix transform của hàm Translate sẽ được tính bằng cách nhân với matrix $\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
	flipVertically	Hàm tạo matrix transform cho phép lật dọc

		Theo tài liệu matrix transform của hàm Translate sẽ được tính bằng cách nhân với matrix $\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
	Inverse	Hàm tạo ma trận nghịch đảo. Xây dựng bằng cách gọi hàm inv() có sẵn của std::Mat.
Geometric Transformer	Transform	Hàm biến đổi ảnh theo 1 phép biến đổi affine đã có - Bước 1: xác định những thông tin cần thiết của ảnh gốc và ảnh sau chỉnh sửa như chiều dài, chiều rộng, step width, channel... - Bước 2: nghịch đảo ma trận affine. Vì chúng ta làm phép affine ngược nên phải nghịch đảo lại ma trận affine, nếu không nghịch đảo thì ảnh sau xử lý sẽ bị sai lệch, ví dụ như phóng to sẽ thành thu nhỏ, xoay góc sẽ bị lệch.... - Bước 3: duyệt qua từng điểm ảnh trong ảnh gốc, áp dụng matrix affine cho từng điểm ảnh.
	RotateKeepImage	Hàm xoay bảo toàn nội dung ảnh theo góc xoay cho trước - Bước 1: xác định lại kích thước ảnh đích vì khi xoay bảo toàn nội dung ta sẽ có độ cao và độ rộng mới tùy thuộc vào góc xoay. - Bước 2: tạo ảnh đích với kích thước vừa xác định và loại ảnh trùng với ảnh gốc. - Bước 3: tạo 1 đối tượng affineTransform mới. - Bước 4: căn chỉnh lại toạ độ của hình sao cho hình sau khi xoay sẽ ở chính giữa. Từ gốc toạ độ cộng thêm cho trục dx 1 khoảng (newHeight/2 - oldHeight/2) tương tự trục dy (newWidth/2 - oldWidth/2). - Bước 5: đưa matrix affineTransform về (0, 0) để thực hiện các phép biến đổi cơ bản. Sau đó ta dùng phép xoay (rotate) 1 góc cho trước và lại tịnh tiến trở lại vị trí ban đầu. - Bước 6: tạo ảnh hoàn chỉnh từ hàm Transform với matrix affineTransform ta vừa biến đổi phía trên. - Hàm này có dùng những công thức lượng giác nên phải đặc biệt cẩn thận với những trường hợp

		đặc biệt (góc 180 độ nó sẽ làm việc tính toán newWidth và newHeight bị sai).
	RotateUnkeepImage	Hàm xoay không bảo toàn nội dung ảnh theo góc xoay cho trước
		Tương tự như xoay ảnh bảo toàn nội dung ảnh nhưng ở hàm này chúng ta bỏ qua bước 1. Ảnh mới sẽ cùng kích thước với ảnh cũ và sẽ bị mất thông tin ở các góc.
	Scale	Hàm phóng to, thu nhỏ ảnh theo tỉ lệ cho trước
		<ul style="list-style-type: none"> - Bước 1: tạo ảnh đích với kích thước và loại ảnh trùng với ảnh gốc. - Bước 2: tạo 1 đối tượng affineTransform mới. - Bước 3: đưa matrix affineTransform về (0, 0) để thực hiện các phép biến đổi cơ bản. Sau đó ta dùng phép thu phóng (scale) để thu phóng ảnh theo trục dx và dy sau đó lại tịnh tiến trở lại vị trí ban đầu. - Bước 4: tạo ảnh hoàn chỉnh từ hàm Transform với matrix affineTransform ta vừa biến đổi phía trên.
	Resize	Hàm thay đổi kích thước ảnh
		Tương tự với hàm Scale nhưng khác ở <ul style="list-style-type: none"> - Bước 1: ảnh đích sẽ được tạo với kích thước bằng ảnh gốc nhân với hệ số resize (kích thước mới chia cũ). - Bước 3: sau khi đã dùng hàm scale thì lúc đưa ảnh lại tọa độ ban đầu chúng ta phải nhân thêm với hệ số resize.
	Flip	Hàm lấy ảnh đối xứng
		<ul style="list-style-type: none"> - Bước 1: tạo ảnh đích với kích thước và loại ảnh trùng với ảnh gốc. - Bước 2: tạo 1 đối tượng affineTransform mới. - Bước 3: đưa matrix affineTransform về (0, 0) để thực hiện các phép biến đổi cơ bản. Sau đó ta dùng phép lật ngang (flipHorizontally) hoặc lật dọc (flipVertically) để lật matrix rồi lại tịnh tiến trở lại vị trí ban đầu. - Bước 4: tạo ảnh hoàn chỉnh từ hàm Transform với matrix affineTransform ta vừa biến đổi phía trên.

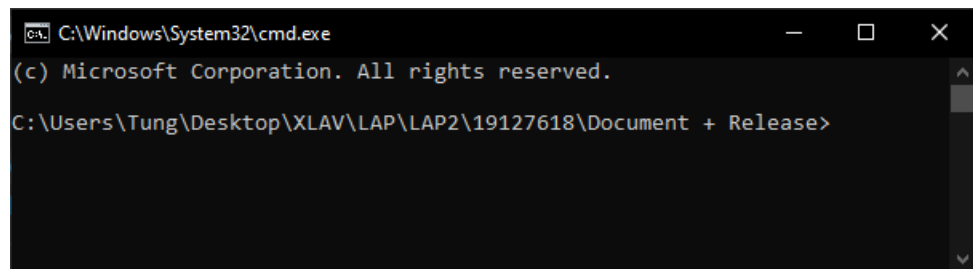
III. Hướng dẫn sử dụng

1. Cấu trúc thư mục:

```
|-19127618:
|   -Document+Release:
|       -19127618_lab2.docx
|       -19127618_lab2.pdf
|       -Program.exe
|       -test.jpg
|   -source:
|       -GeometricTransformer.h
|       -Lib.h
|       -SourceLab2.cpp
|       -MainLab2.cpp
```

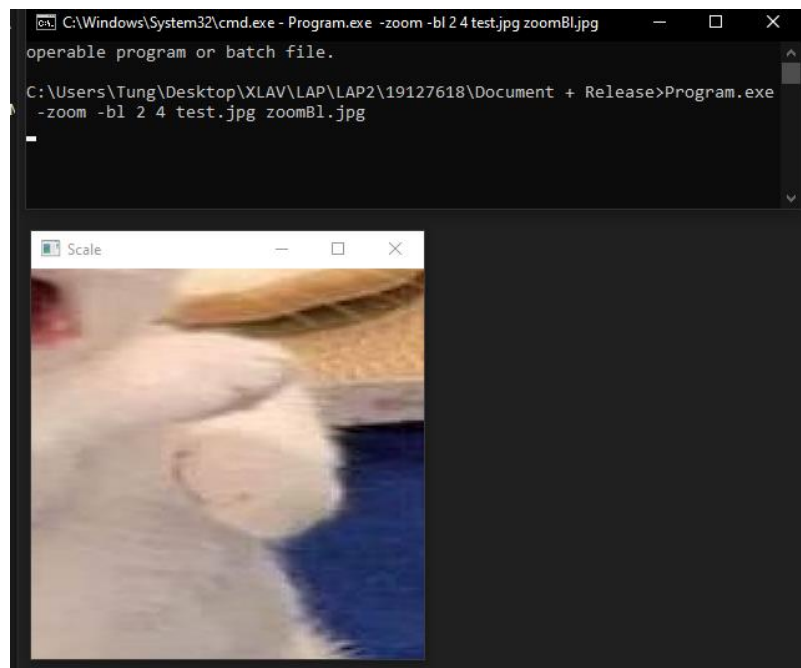
2. Cách sử dụng:

- Khởi chạy cmd, đi tới thư mục chứa file Program.exe.



```
C:\Windows\System32\cmd.exe
(c) Microsoft Corporation. All rights reserved.
C:\Users\Tung\Desktop\XLAV\LAP\LAP2\19127618\Document + Release>
```

- Nhập cmd theo hướng dẫn trong đề.



- Sau khi tắt ảnh preview ảnh sẽ được lưu lại.

3. Lưu ý:

- Phải nhập path chính xác nếu không chương trình sẽ không hoạt động.
- Những giá trị truyền vào bắt buộc phải lớn hơn 0.
- Trong file GeometricTransformer.h những hàm Interpolate em có thêm 1 tham chiếu uchar* des. Lý do vì hàm Interpolate có kiểu trả về là uchar chỉ có thể trả về 1 giá trị uchar nhưng qua hàm này em cần nó trả về một matrix dạng array uchar. Có 2 lựa chọn sửa lại hàm Interpolate là sửa kiểu trả về hoặc thêm biến tham chiếu -> em chọn thêm biến tham chiếu.
- Trong file GeometricTransformer.h class AffineTransform em có thêm 3 function flipHorizontally và flipVertically dùng để phụ trợ cho hàm flip của class GeometricTransformer, hàm Inverse để tạo ma trận nghịch đảo trong function Transform.

IV. Tham khảo:

- https://en.wikipedia.org/wiki/Affine_transformation
- https://users.soict.hust.edu.vn/trungtt/uploads/slides/CG05_ModelingTrans.pdf
- Slide lý thuyết và video hướng dẫn.

--HẾT--

Chúc thầy một ngày vui vẻ!