**BỘ MÔN THỊ GIÁC MÁY TÍNH – KHOA CÔNG NGHỆ THÔNG TIN**

**ĐẠI HỌC KHOA HỌC TỰ NHIÊN THÀNH PHỐ HỒ CHÍ MINH, ĐẠI HỌC QUỐC GIA TP HCM**

CN Phạm Minh Hoàng

CN Phạm Thanh Tùng

Sinh viên thực hiện: 19120106 – Nguyễn Ngọc Khôi Nguyên

GV phụ trách: PGS. TS Lý Quốc Ngọc

Đồ án/bài tập môn học  - XỬ LÝ ẢNH SỐ VÀ VIDEO SỐ

HỌC KỲ I – NĂM HỌC 2021-2022

**môn học XỬ LÝ ẢNH SỐ**

**BẢNG THÔNG TIN CÁ NHÂN**

|  |  |
| --- | --- |
| **Mã sinh viên:** | 19120106 |
| **Tên sinh viên:** | **Nguyễn Ngọc Khôi Nguyên** |
| **Avatar** |  |

**YÊU CẦU ĐỒ ÁN- BÀI TẬP**

|  |  |
| --- | --- |
| **Loại bài tập** | **Lý thuyết 🗹 Thực hành 🗹 Đồ án Bài tập** |
| **Ngày bắt đầu** | **04/11/2021** |
| **Ngày kết thúc** | **21/12/2021** |

# **Yêu cầu của Đồ án/Bài tập LAB 03**

In this assignment, you must implement some simple image manipulation using OpenCV in C++. Your program is called by command line arguments and perform the following functions.

# **Solutuion**

1. Load the input image by reading it from a file.

* Sử dụng cv::imread(<InputFilePath>, cv::IMREAD\_UNCHANGED)
* <InputFilePath>: đường dẫn của ảnh cần đọc
* cv::IMREAD\_GRAYSCALE: ý nghĩa là ảnh load lên d

1. Save the processed output image to a file.

* Sử dụng cv::imwrite(<OutputFilePath>, sourceImage)
* <OutputFilePath>: đường dẫn để lưu ảnh
* sourceImage: ảnh cần lưu

1. Convolution

* Sử dụng Convolution::DoConvolution(const Mat& sourceImage, Mat& destinationImage)
* sourceImage: ảnh ban đầu kích thước Height x Width
* destinationImage: ảnh sau khi nhân tích chập với kernel cho sẵn
* Khởi gán ảnh destinationImage là bản sao của ảnh ban đầu.
* Tạo vector offsets lưu vùng lân cận của 1 pixel với kích thước kHeight\*kWidth (kích thước của kernel).
* Khởi tạo xStart = kWidth / 2, xEnd = width - 1 - kWidth / 2

yStart = kHeight / 2, yEnd = height - 1 - kHeight / 2

là khoảng pixel nhân tích chập (các pixel ở biên giữ nguyên)

* Với mỗi pixel của ảnh gốc trong khoảng (xStart -> xEnd, yStart -> yEnd) ta gán nó bằng:

A picture containing shoji, public

Description automatically generated

Table

Description automatically generated

1. Filter an image using average filtering

* Sử dụng Blur::AverageOpt(const Mat& sourceImage, Mat& destinationImage, int kWidth, int kHeight)
* sourceImage: ảnh ban đầu kích thước Height x Width
* destinationImage: ảnh sau khi blur
* kWidth, kHeight: Kích thước kernel sử dụng
* Khởi gán ảnh destinationImage là bản sao của ảnh gốc.
* Áp dụng phép tích chập với ảnh sourceImage: kernel lọc trung bình (kích thước kWidth x kHeight) với giá trị mỗi phần tử là
* Ta thu được ảnh destinationImage sau khi áp dụng bộ lọc trung bình.

1. Filter an image using median filtering

* Sử dụng Blur::MedianOpt(const Mat& sourceImage, Mat& destinationImage, int kWidth, int kHeight)
* sourceImage: ảnh ban đầu kích thước Height x Width
* destinationImage: ảnh sau khi blur
* kWidth, kHeight: Kích thước kernel sử dụng
* Khởi gán ảnh destinationImage là bản sao của ảnh gốc.
* Ta làm như phép tích chập, với mỗi pixel sourceImage trong ta xét các giá trị pixel trong vùng lân cận (kWidth x kHeight) và lấy giá trị trung vị gán cho pixel tương ứng trong ảnh destinationImage (việc này như việc ta sắp xếp lại các pixel trong vùng lân cận và dùng tích chập với kernel toàn giá trị 1).
* Ta thu được ảnh destinationImage sau khi áp dụng bộ lọc trung bình.

1. Filter an image using Gaussian filtering

* Sử dụng Blur::GaussianOpt(const Mat& sourceImage, Mat& destinationImage, int kWidth, int kHeight)
* sourceImage: ảnh ban đầu kích thước Height x Width
* destinationImage: ảnh sau khi blur
* kWidth, kHeight: Kích thước kernel sử dụng
* Khởi gán ảnh destinationImage là bản sao của ảnh gốc.
* Ta các giá trị sigmaX và sigmaY như sau:
* Tạo kernel như sau:

Table

Description automatically generated

* Áp dụng tích chập với kernel vừa tạo ta được ảnh kết quả sau khi áp Gaussian filter.

1. Detect edges using Sobel operator.

* Sử dụng EdgeDetector::SobelOpt(const Mat& sourceImage, Mat& destinationImage)
* sourceImage: ảnh ban đầu kích thước Height x Width
* destinationImage: ảnh sau khi nhận dạng biên cạnh
* Khởi gán ảnh destinationImage là bản sao của ảnh gốc.
* Tạo 2 kernel như sau:

A picture containing text, object, clock

Description automatically generated

* Ta tính 2 ma trận và là giá trị gradient với 2 hướng X và Y bằng cách lấy tích chập ảnh gốc với 2 kernel tương ứng ở trên.
* Tính ma trận giá trị gradient bằng cách:
* Với mỗi pixel ở ảnh destinationImage nếu giá trị gradient tương ứng ở ảnh sourceImage của pixel đó lớn hơn ngưỡng threshold (ta lấy giá trị gradient lớn nhất của ảnh nhân với một tỉ số đặt trước, với tùy ảnh). Đặt giá trị của pixel đó thành 255, ngược lại đặt là 1:
* Ta thu được ảnh đầu ra là đen trắng là ảnh biên cạnh nhận dạng được bằng phép toán tử Sobel.

1. Detect edges using Prewitt operator

* Sử dụng EdgeDetector::PrewittOpt(const Mat& sourceImage, Mat& destinationImage)
* sourceImage: ảnh ban đầu kích thước Height x Width
* destinationImage: ảnh sau khi nhận dạng biên cạnh
* Khởi gán ảnh destinationImage là bản sao của ảnh gốc.
* Tạo 2 kernel như sau:

A picture containing text, object, clock

Description automatically generated

* Ta tính 2 ma trận và là giá trị gradient với 2 hướng X và Y bằng cách lấy tích chập ảnh gốc với 2 kernel tương ứng ở trên.
* Tính ma trận giá trị gradient bằng cách:
* Với mỗi pixel ở ảnh destinationImage nếu giá trị gradient tương ứng ở ảnh sourceImage của pixel đó lớn hơn ngưỡng threshold (ta lấy giá trị gradient lớn nhất của ảnh nhân với một tỉ số đặt trước, với tùy ảnh). Đặt giá trị của pixel đó thành 255, ngược lại đặt là 1:
* Ta thu được ảnh đầu ra là đen trắng là ảnh biên cạnh nhận dạng được bằng phép toán tử Prewitt.

1. Detect edges using Laplace operator

* Sử dụng EdgeDetector::LaplaceOpt(const Mat& sourceImage, Mat& destinationImage)
* sourceImage: ảnh ban đầu kích thước Height x Width
* destinationImage: ảnh sau khi nhận dạng biên cạnh
* Khởi gán ảnh destinationImage là bản sao của ảnh gốc.
* Tạo kernel như sau:

A picture containing text, clock

Description automatically generated

* Ta tính giá trị đạo hàm bậc 2 của từng pixel trong ảnh đầu vào bằng cách lấy tích chập ảnh gốc với kernel đã tạo
* Với mỗi pixel ở ảnh destinationImage nếu giá trị gradient tương ứng ở ảnh sourceImage của pixel đó lớn hơn ngưỡng threshold (ta lấy giá trị gradient lớn nhất của ảnh nhân với một tỉ số đặt trước, với tùy ảnh). Đặt giá trị của pixel đó thành 255, ngược lại đặt là 1:
* Ta thu được ảnh đầu ra là đen trắng là ảnh biên cạnh nhận dạng được bằng phép toán tử Laplace.

# **Hướng dẫn sử dụng**

1. Load the input image by reading it from a file.

Sử dụng trong chương trình để đọc ảnh input của các lệnh khác.

1. Save the processed output image to a file.

Sử dụng trong chương trình để lưu ảnh output của các lệnh khác.

1. Filter an image using average filtering

Command line:

<Program.exe> -avg <> <> <InputFilePath> <OutputFilePath>

* Program.exe: tên của file exe
* <InputFilePath>: đường dẫn của file input
* <OutputFilePath>: đường dẫn của file output
* -avg: tên lệnh
* : kích thước kernel theo chiều ngang
* : kích thước kernel theo chiều dọc

Lưu ý:

* Nếu nhập bắt đầu là số thì chương trình sẽ lấy giá trị cho đến khi gặp một ký tự không phải số.
* Nếu nhập là ký tự chữ, dấu,… thì chương trình tự chuyển sang bằng 0.

VD: với kernel 3x3

Input:  Output: 

1. Filter an image using median filtering

Command line:

<Program.exe> -med <> <> <InputFilePath> <OutputFilePath>

* Program.exe: tên của file exe
* <InputFilePath>: đường dẫn của file input
* <OutputFilePath>: đường dẫn của file output
* -med: tên lệnh
* : kích thước kernel theo chiều ngang
* : kích thước kernel theo chiều dọc

Lưu ý:

* Nếu nhập bắt đầu là số thì chương trình sẽ lấy giá trị cho đến khi gặp một ký tự không phải số.
* Nếu nhập là ký tự chữ, dấu,… thì chương trình tự chuyển sang bằng 0.

VD: với kernel 3x3

Input: A picture containing person

Description automatically generated Output: A person wearing a hat

Description automatically generated with medium confidence

1. Filter an image using Gaussian filtering

Command line:

<Program.exe> -gau <> <> <InputFilePath> <OutputFilePath>

* Program.exe: tên của file exe
* <InputFilePath>: đường dẫn của file input
* <OutputFilePath>: đường dẫn của file output
* -gau: tên lệnh
* : kích thước kernel theo chiều ngang
* : kích thước kernel theo chiều dọc

Lưu ý:

* Nếu nhập bắt đầu là số thì chương trình sẽ lấy giá trị cho đến khi gặp một ký tự không phải số.
* Nếu nhập là ký tự chữ, dấu,… thì chương trình tự chuyển sang bằng 0.

VD: với kernel 3x3

Input: A picture containing person

Description automatically generated Output: 

1. Detect edges using Sobel operator

Command line:

<Program.exe> -sobel <InputFilePath> <OutputFilePath>

* Program.exe: tên của file exe
* <InputFilePath>: đường dẫn của file input
* <OutputFilePath>: đường dẫn của file output
* -sobel: tên lệnh

VD: với kernel 3x3

Input:  Output: A picture containing outdoor object

Description automatically generated

1. Detect edges using Prewitt operator

Command line:

<Program.exe> -prew <InputFilePath> <OutputFilePath>

* Program.exe: tên của file exe
* <InputFilePath>: đường dẫn của file input
* <OutputFilePath>: đường dẫn của file output
* -prew: tên lệnh

VD: với kernel 3x3

Input:  Output: A picture containing outdoor object

Description automatically generated

1. Detect edges using Laplace operator

Command line:

<Program.exe> -lap <InputFilePath> <OutputFilePath>

* Program.exe: tên của file exe
* <InputFilePath>: đường dẫn của file input
* <OutputFilePath>: đường dẫn của file output
* -lap: tên lệnh

VD: với kernel 3x3

Input: A person wearing a hat

Description automatically generated with medium confidence Output: A close-up of a person's head

Description automatically generated with low confidence

# **Tài liệu tham khảo**

### [1]: *Learning Image Processing with OpenCV, by Gloria Bueno Garcia (Author), Oscar Deniz Suarez (Author), Jose Luis Espinosa Aranda (Author), Jesus Salido Tercero (Author), Ismael Serrano Gracia (Author)*

### [2]: *https://docs.opencv.org/4.x/d4/d13/tutorial\_py\_filtering.html*