ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO LAB 02

Môn: Thị giác máy tính - CSC16004

Lớp: 21_22

Tên: Lê Nguyễn

Mã số sinh viên: 21120511

Thành phố Hồ Chí Minh - 2024

Mục lục

1 Hướng dẫn sử dụng

Hướng dẫn được thực hiện trên hệ điều hành **Ubuntu 22.04.4 LTS** [1].

1.1 Build source code

- Đầu tiên cài đặt những package quan trọng (gcc, g++, gdb, ...) và cmake (công cụ build chính):

```
$ sudo apt install build-essential gdb cmake
```

- Tiếp theo cài đặt thư viện OpenCV:

```
$ sudo apt install libopencv-dev
```

- Trong thư mục Sources (gồm source code chính và file CMakeLists.txt), tạo thư mục build và đi vào thư mục build [2]:

```
$ mkdir build
$ cd build
```

- Sau đó thực hiện build source code bằng:

```
$ cmake -DCMAKE_BUILD_TYPE=Debug ..
$ make
```

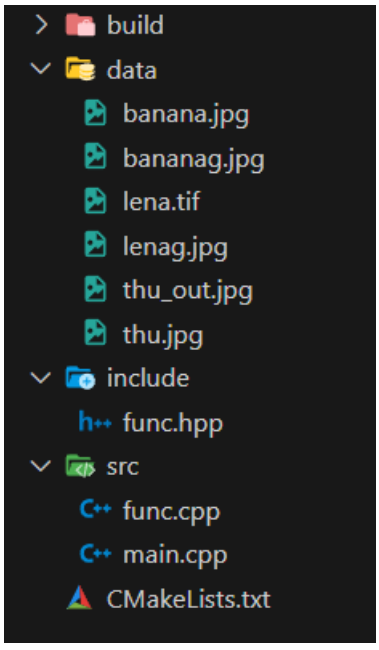
- Khi đã build source code thành công, thì có thể chạy bằng command sau:

```
$ ./21120511 -<command> <InputPath> <OutputPath> <Value (optional)>
```

Lưu ý: đường dẫn tới ảnh (hoặc bất cứ file nào trong thư mục Sources) được hiểu là relative path. Giả sử muốn dùng ảnh thu.jpg trong thư mục data làm ảnh input và ảnh thu_out.jpg làm ảnh output, ta có thể dùng command như sau:

```
$ ./21120511 -<command> data/thu.jpg
                        data/thu_out.jpg
                        <Value>

// Do giới hạn chỗ viết nên em xuống dòng
// nhưng khi gõ command thì ta gõ
// luôn một dòng
```



Hình 1: Cấu trúc của thư mục Sources

[1] Ubuntu này được em chạy trên WSL2 nhưng em nghĩ vẫn có thể áp dụng lên Ubuntu chạy trên máy thật.
[2] Mỗi một dấu `$` được hiểu là một command, hai dấu tức là thực hiện command phía trên xong rồi đến command phía dưới.

1.2 Chạy bằng file executable

Trong folder nộp bài sẽ gồm 4 folder nhỏ:

- **Document**: chứa báo cáo.
- **Sources**: chứa source code, có thể build và chạy như phía trên.
- **Data**: bao gồm ảnh input và các ảnh output (nếu muốn chạy bằng cách trên, hãy copy các ảnh này vào thư mục Sources).
- **Executable**: bao gồm file executable có tên là 21120511.

Để chạy bằng file executable, đầu tiên cho phép quyền chạy file bằng:

```
$ cd Executable
$ chmod +x 21120511
```

Sau đó có thể thực hiện chạy file bằng command như phần trên. Lưu ý, vị trí đường dẫn ảnh lúc này không còn là relative path. Giả sử muốn input ảnh sekiro.jpg nằm trong folder **Data** và chuyển thành ảnh xám với output là ảnh sekiro_out.jpg nằm trong folder hiện tại thì ta nhập như sau:

```
$ ./21120511 -rgb2gray ../Data/sekiro.jpg sekiro_out.jpg
```

2 Giải thích chi tiết

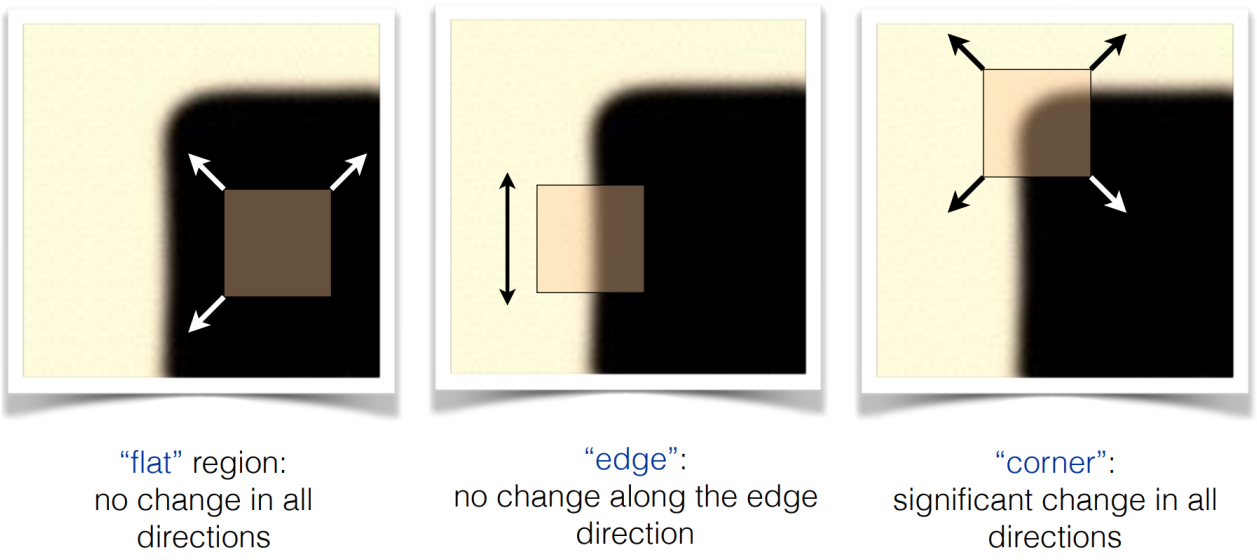
Hình 2: Ảnh được dùng để chạy thuật toán Harris Corner Detection

Hình 3: Ảnh 11 sau khi chạy thuật toán

Command:

```
$ ./21120511 -harris data/tulips.png data/out.png
```

2.1 Định nghĩa góc



Hình 4: Khác biệt giữa góc, biên cạnh và vùng "phẳng" [1]

Dựa vào ảnh 4, khi nhìn ảnh bằng một "cửa sổ" nhỏ thì ta có thể nhận thấy rằng:

- Khi cửa sổ di chuyển ở vùng "phẳng" thì giá trị của các điểm ảnh gần như không thay đổi (vẫn giữ màu đen).
- Khi cửa sổ di chuyển ở biên cạnh thì giá trị của các điểm ảnh không thay đổi theo hướng của cạnh.
- Khi cửa sổ di chuyển ở góc thì giá trị của các điểm ảnh ở bất kỳ hướng nào đều có thay đổi.

Vậy để xác định được góc, ta cần dùng một cửa sổ và tìm những nơi mà điểm ảnh thay đổi ở mọi hướng.

2.2 Nền tảng

Gọi ảnh là I, khi đó một điểm ảnh tại vị trí (x, y) được kí hiệu là $I(x, y)$. Giả sử ảnh I là một ảnh xám, khi đó $I(x, y) \in \mathbb{R}$. Xét một cửa sổ W của ảnh, khi W di chuyển một đoạn (u, v) thì ta có được độ thay đổi của ảnh là:

$$E(u, v) = \sum_{(x,y)} w(x, y) \left[I(x + u, y + v) - I(x, y)\right]^2 \tag{1}$$

Trong đó $I(x + u, y + v)$ là giá trị của điểm ảnh sau khi di chuyển một đoạn, còn $w(x, y)$ là hàm cửa sổ, ta chọn hàm $w(x, y)$ sao cho:

$$w(x, y) = \begin{cases} 1 & \text{nếu } (x, y) \in W \\ 0 & \text{ngược lại} \end{cases} \tag{2}$$

Đặt $I_x(x,y)$ là đạo hàm của ảnh theo hướng x tại (x,y), tương tự $I_y(x,y)$ là đạo hàm của ảnh theo hướng y tại (x,y). Khai triển Taylor cho giá trị của ảnh sau khi dịch chuyển một đoạn, ta có:

$$I(x+u,y+v) \approx I(x,y) + uI_x(x,y) + vI_y(x,y) \tag{3}$$

Thay phương trình 3 vào 1, ta được:

$$E(u,v) \approx \sum_{(x,y)} w(x,y)\left[uI_x(x,y) + vI_y(x,y)\right]^2 \tag{4}$$

Phân tích phương trình 4, ta có thể đưa về dạng toàn phương:

$$\left[uI_x(x,y) + vI_y(x,y)\right]^2 = u^2I_x(x,y)^2 + 2uvI_x(x,y)I_y(x,y) + v^2I_y(x,y)^2 \tag{5}$$

$$= \begin{bmatrix} u & v \end{bmatrix} \begin{bmatrix} I_x(x,y)^2 & I_x(x,y)I_y(x,y) \\ I_x(x,y)I_y(x,y) & I_y(x,y)^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \tag{6}$$

Đặt $I_x^2 = I_{xx}, I_y^2 = I_{yy}$ và $I_x * I_y = I_{xy}$. Ta lại có:

$$E(u,v) \approx \begin{bmatrix} u & v \end{bmatrix} \sum_{(x,y)} w(x,y) \begin{bmatrix} I_{xx} & I_{xy} \\ I_{xy} & I_{yy} \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \tag{7}$$

Đặt M là vế giữa của phương trình 7 và đặt $S_{xx} = \sum_{(x,y)} w(x,y)I_{xx}$, tương tự với S_{yy}, S_{xy}. Ta được:

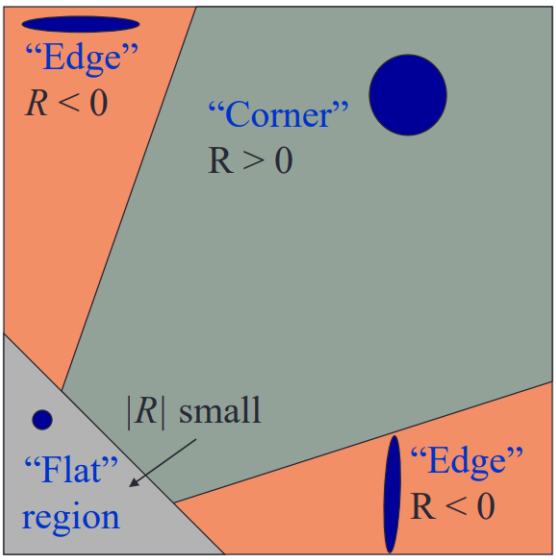
$$M = \begin{bmatrix} S_{xx} & S_{xy} \\ S_{xy} & S_{yy} \end{bmatrix} \tag{8}$$

Để xác định được góc [2], ta tính giá trị R, gọi là **Harris Response Value**:

$$R = det(M) - k \cdot trace(M)^2 \tag{9}$$

$$= S_{xx} \cdot S_{yy} - k \cdot (S_{xy} + S_{xy})^2 \tag{10}$$

trong đó k là giá trị mà ta tự điều chỉnh, thông thường k nằm trong đoạn $[0.04, 0.06]$.



Hình 5: Dựa vào giá trị R ta có thể xác định góc [3]

Góc sẽ có giá trị R lớn và dương, cạnh thường có giá trị R lớn nhưng âm trong khi đó những vùng "phẳng" sẽ có giá trị $|R|$ nhỏ.

2.3 Thuật toán

Bước 1 : Chuyển ảnh về thành ảnh xám.

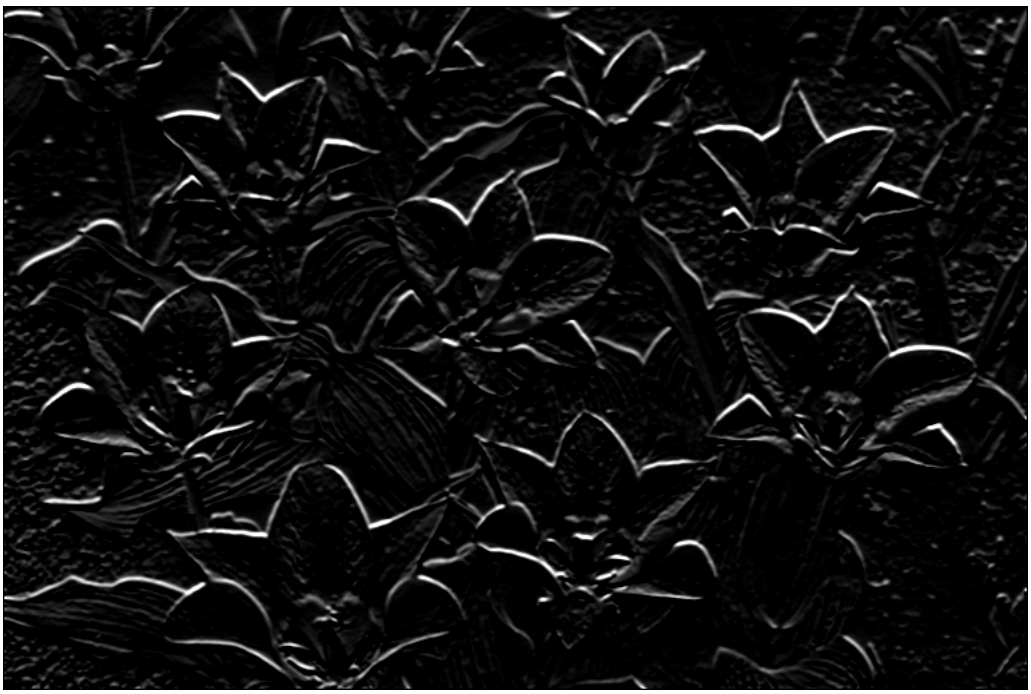
Hình 6: Chuyển về ảnh xám

Bước 2 : Loại bỏ nhiễu của ảnh, thông thường ta sẽ dùng lọc Gaussian với kernel kích thước 5×5.

Hình 7: Dùng lọc Gaussian

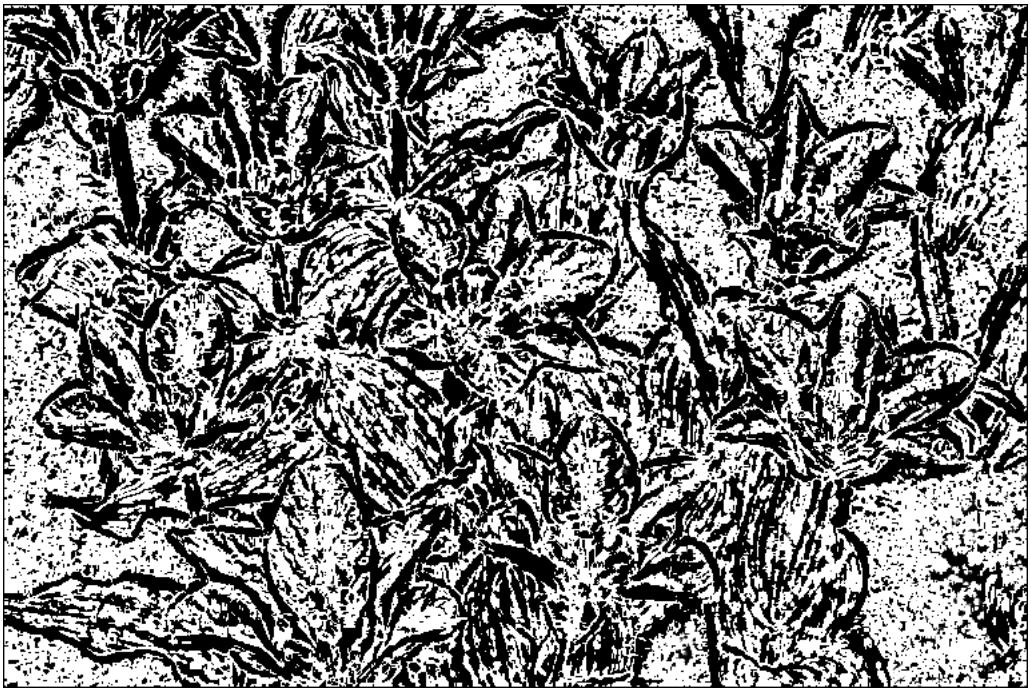
Bước 3 : Tính đạo hàm của ảnh theo 2 hướng x và y. Ta có thể dùng kernel Sobel cho 2 hướng x và y để tính I_x và I_y (ở trong code em dùng kernel Sobel).

Hình 8: Đạo hàm theo hướng x



Hình 9: Đạo hàm theo hướng y

Bước 4 : Sau đó tính ma trận M bằng cách tính từng giá trị S_{xx}, S_{yy} và S_{xy}, sau khi có ma trận M, ta tính từng giá trị R cho từng vị trí (x, y) (xem như là một cửa sổ xung quanh (x, y)).



Hình 10: Sau khi tính giá trị R cho từng vị trí (x, y), ta có được ảnh như đây

Bước 5 : Thế nhưng ta vẫn chưa xong, không phải vị trí (x, y) nào có $R > 0$ đều là góc. Để tìm được vị trí (x, y) là góc, ta dùng **Non-maximum suppression**. Ở dạng đơn giản nhất của thuật toán này, ta định nghĩa một cửa sổ kích thước $k \times k$ sau đó tìm giá trị lớn nhất trong cửa sổ ấy, gọi là m. Ta định nghĩa thêm giá trị ε, thông thường $\varepsilon = \max R \cdot \alpha$ với $\max R$ là giá trị reponse lớn nhất của ảnh còn α là giá trị ta tự định nghĩa (trong code $\alpha = 0.001$), nếu $m > \varepsilon$ thì ta xem điểm ấy là góc. Lúc này, ta được ảnh cuối cùng (những hình tròn màu xanh là góc, với $m > \varepsilon$):

Hình 11: Sau khi dùng Non-maximum suppression

Tài liệu

[1] 6.2 harris corner detection. URL: https://www.cs.cmu.edu/~16385/s17/Slides/6.2_Harris_Corner_Detector.pdf.

[2] Christopher G. Harris and M. J. Stephens. A combined corner and edge detector. In *Alvey Vision Conference*, 1988. URL: https://api.semanticscholar.org/CorpusID:1694378.

[3] Cs4495-features1. URL: https://faculty.cc.gatech.edu/~afb/classes/CS4495-Fall2014/slides/CS4495-Features1.pdf.