

`  

**ĐẠI HỌC QUỐC GIA TP HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**------------------------**

**BÁO CÁO ĐỒ ÁN**

**MỘT VÀI HIỆU ỨNG TRÊN VIDEO BẰNG OPENCV**

*Môn học*: **Nhập môn thị giác máy tính**

*Giảng viên:* ***TS. Nguyễn Vinh Tiệp***

*Danh sách thành viên*:

Võ Huy Khôi 18520949

Nguyễn Thịnh Quyền 18521322

Hứa Văn Sơn 18521344

**PHÂN CÔNG THÀNH VIÊN**

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên** | **Võ Huy Khôi** | **Hứa Văn Sơn** | **Nguyễn Thinh Quyền** |
| **Mã số sinh viên** | 18520949 | 18521344 | 18521322 |
| **Công việc thực hiện** | Lọc data, tiền xử lý ban đầu, xây dựng bảng kết quả. | Cài đặt thuật toán truy vấn | Cào data, xây dựng giao diện web |

1. Tổng quan đề tài

Giải trí luôn là lĩnh vực được mọi người quan tâm, nhất là trong khoảng thời gian công nghệ phát triển như hiện nay. Các xu hướng giải trí ngày càng phát triển và được ưu tiên sử dụng ngày càng nhiều điển hình như tiktok và các trang mạng xã hội khác. Một trong những hình thức giải trí mà được nhiều người yêu thích và tin chọn nhiều nhất là chỉnh sửa ảnh trên video bằng những hiệu ứng thú vị ví dụ như : làm đẹp, chỉnh màu , chèn thêm hình ảnh, … Nắm bắt được nhu cầu giải trí của người dùng trên thế giới đang gia tăng nên nhóm quyết định chọn đề tài làm một số hiệu ứng trên video bằng OpenCv.

1. Tổng quan về OpenCv
2. OpenCv

OpenCV (Open Computer Vision) là một thư viện mã nguồn mở hàng đầu cho xử lý về thị giác máy tính, machine learning, xử lý ảnh. OpenCV đươc viết bằng C/C++, vì vậy có tốc độ tính toán rất nhanh, có thể sử dụng với các ứng dụng liên quan đến thời gian thực. Opencv có các interface cho C/C++, Python Java vì vậy hỗ trợ được cho Window, Linux, MacOs lẫn Android, iOS OpenCV có cộng đồng hơn 47 nghìn người dùng và số lượng download vượt quá 6 triệu lần. Phạm vi sử dụng từ nghệ thuật tương tác, cho đến lĩnh vực khai thác mỏ, bản đồ trên web hoặc công nghệ robot.

Trang chủ chính thức của OpenCv: <http://opencv.org/>

Trang download các phiên bản của OpenCv: <http://opencv.org/downloads.html>

Hiện tại bản mới nhất của thư viện OpenCv là bản 2.4.13

1. Xử lí video bằng OpenCv

Video thực chất là các frame được thực hiện liên tiếp. Mỗi frame là một tensor 3 chiều gồm chiều dài, chiều rộng và chiều sâu ( kênh màu ). Một frame một ảnh, nhiều frame sẽ tạo nên một video, vì vậy sử dụng các hàm, các kĩ thuật có trong OpenCv xử lí cho từng ảnh và từ đó có thể áp dụng lên toàn bộ ảnh trong video.

1. Một vài hiệu ứng trên video
2. Hiệu ứng ảnh chì
   * 1. Ý tưởng

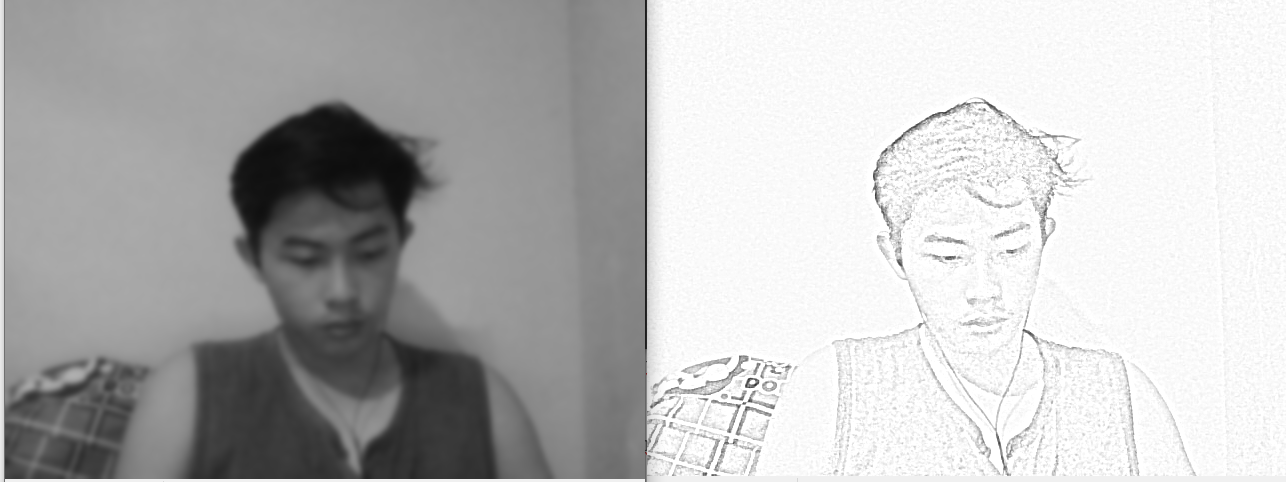
* Ta có thể thấy ảnh chì là ảnh gồm 2 phần: pixel màu đen ở những vùng biên cạnh và pixel màu trắng ở nhưng vùng còn lại.
* Dựa vào đặc điểm trên ta có thể dễ dàng tạo ra ảnh chì bằng python.
  + 1. Thực hiện
* Bước 1: ta tạo 1 ảnh xám.
* Bước 2: ta tạo 1 ảnh blur của ảnh xám đã tạo trên.
* Bước 3: Ảnh chì sẽ bằng ảnh xám chia cho ảnh blur.

**Giải thích**: sau khi blur ảnh xám, những pixel vị trí biên cảnh ở ảnh blur sẽ có giá trị **nhỏ hơn** các pixel ở ảnh xám, và các các pixel ở vị trí còn lại sẽ nhỏ hơn. Do đó khi ta lấy ảnh xám chia cho ảnh blur, những vị trị ở biên cạnh sẽ có giá trị lớn hơn 1, còn những pixel ở các vùng còn lại sẽ có giá trị nhỏ hơn 1. Kết quả tạo thành ảnh chỉ với những pixel tối màu ở biên cạnh và trắng ở các vùng khác.

Ảnh chì sẽ khác khác với các ảnh đã dùng filter tìm cảnh ở chỗ các pixel ở các đường biên cạnh của ảnh chì sẽ có giá trị nhỏ hơn, nên ảnh sẽ có màu hơi nhạt. Do độ lệch giữa các giá trị của ảnh xám và ảnh blur là không nhiều nên khi chia sẽ không ra các giá trị quá lớn.

* + 1. Kết quả

Kết quả sau khi blur và sau khi lấy ảnh xám chia ảnh blur :



1. Hiệu ứng phát hiện chuyển động

a. Ý tưởng

Ý tưởng dựa trên kỹ thuật Frame Subtraction

* Ý tưởng của Frame Subtraction (tạm dịch: trừ khung hình) rất đơn giản, đó là ta trừ hai khung hình với nhau sau đó chuẩn hóa về miền giá trị mong muốn. Ví dụ: Nếu muốn trực quan hóa sự khác biệt giữa hai ảnh thì ta có thể lấy giá trị tuyệt đối. Hoặc cũng có thể áp dụng các kiểu chuẩn hóa khác tùy mục đích muốn xử lý là gì.
* Khi áp dụng Frame Subtraction cho video thì ta sẽ lấy 2 khung hình liền kề (pair) trừ cho nhau để tìm điểm khác biệt ở đồ án này nhóm chọn lấy 2 khung cách nhau frame. Nói về bản chất của video thì có thể xem nó là một chồng ảnh, bởi video chính là việc trình chiếu các khung ảnh theo thời gian. Do đó, việc áp dụng xử lý ảnh trên video sẽ là việc áp dụng xử lý ảnh trên từng khung hình (frame) bằng cách lần lượt lặp cho đến hết tất cả khung hình có trong video. Sau đó sử dụng hàm cv2.absdiff trong OpenCV để làm rõ sự khác biệt giữa 2 frame.

b. Thực hiện

* Đọc video trong OpenCV: điểm mấu chốt chính là cv2.VideoCapture và lặp trên cap.read(). Vòng lặp các frame ảnh kết thúc khi hết video hoặc bị lỗi. Như vậy cứ mỗi lần lặp ta sẽ có khung hình để xử lý.
* Tiếp theo cần phải tạo 1 biến để lưu giá trị frame trước, đương nhiên nếu ở frame đầu tiên thì sẽ không có frame trước để xử lí, ta có thể bỏ qua frame đầu tiên.
* Sau khi trừ 2 frame cách nhau 1 khung hình ta chuẩn hóa bằng cách lấy giá trị tuyệt đối. Khi sử dụng OpenCV chỉ cần sử dụng đúng một hàm sau : cv2.absdiff (<https://docs.opencv.org/4.0.0/d2/de8/group__core__array.html#ga6fef31bc8c4071cbc114a758a2b79c14>) trừ điểm với điểm giữa 2 ảnh, sau đó lấy giá trị tuyệt đối.
* Cuối cùng để làm cho hiệu ứng thêm sinh động hơn nhóm quyết định cứ mỗi 60fps hay cứ mỗi 60 khung hình trên giây ( thường thì tốc độ video bình thường sẽ khoảng 29 30 khung hình trên giây ) tức là cứ mỗi 2s sẽ thực hiện Frame subtraction.

c. Kết quả :

Frame sau khi xử lí sẽ như thế này và video sẽ là những frame tương tự như vậy:



1. Hiệu ứng xoắn ảnh khi chuyển động

a. Ý tưởng

Chia mỗi frame thành các phần bằng nhau , mỗi frame lấy 1 phần và gộp lại.

b. Thực hiện

Đối với hiệu ứng này thì nhóm không sử dụng hàm hỗ trợ trong OpenCV.

* Đầu tiên nhóm sẽ tính toán các thông số của frame và video ví dụ như sử dụng hàm get(cv2.CAP\_FROP\_FPS) để tính fps của video , hàm get(cv2.CAP\_PROP\_FRAME\_HEIGHT) để tính chiều cao của frame, tính số lượng frame lồng vào nhau: segment\_count = fps\*3(có thể chọn nhân mấy tùy thích, segment\_count càng lớn thì ảnh sẽ càng xoắn nhiều vì có nhiều phần của các frame khác trên cùng 1 khung hình cụ thể ở đây là 90) , tính khoảng cách giữa 2 frame liền kề nhau khi lưu trên cùng 1 khung hình : segment\_height = (height/segment\_count) ở đây ví dụ height của frame là 900 thì segment\_height sẽ là 10.
* Đọc video để mỗi lần lặp ta sẽ có khung hình để xử lý.
* Tạo một mảng lưu trữ các frame, sau khi đủ số lượng frame = segment\_count sẽ bắt đầu xử lý.
* Tạo một mảng trong vòng lặp của mảng lưu trữ các frame trước đó, lưu giá trị pixel lần lượt của từng frame theo khoảng cách segment\_height ví dụ: segment\_height = 10 thì sẽ lưu vị trí các pixel frame[0:10] ở lần lặp đầu tiên, tiếp theo là lưu vị trí các pixel frame[10:20] của frame kế tiếp trong mảng lưu trữ các frame trước đó và cứ tiếp tục như vậy cho đến hết vòng lặp.
* Sau khi đã lặp qua hết tất cả các frame thì nối tất cả các vị trí của tất cả pixel được lưu trước đó vào 1 frame và xuất frame đó ra màn hình.

c. Kết quả

* Kết quả của hiệu ứng:



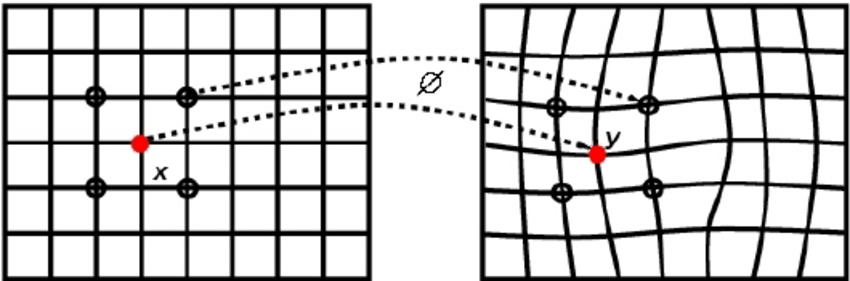
1. Hiệu ứng làm méo ảnh (Image warping)

a. Ý tưởng

Kỹ thuật xử lý làm biến dạng hình ảnh cho nhiều mục đích khác nhau. Ở đây, nhóm sử dụng cho mục đích giải trí, biến dạng các frame của video để tạo hiệu ứng giống như trong nhà cười làm biến dạng khuôn mặt.



Về cơ bản, ý tưởng của kỹ thuật này thay đổi vị trí của các pixel trong ảnh gốc nhưng đồng thời vẫn giữ nguyên các giá trị của nó.



b. Thực hiện

Nhóm thực hiện biến đổi frame dựa trên công thức:

dst(x,y) = src(map\_x(x,y),map\_y(x,y))

* Tạo ra 2 ma trận map\_x, map\_y có cùng kích thước với frame gốc src nhưng chỉ có 2 chiều. Trong đó, pixel của map\_y chứa thông tin về vị trí pixel của src theo hàng (row) và pixel của map\_x chứa thông tin về vị trí pixel của src theo cột (column).
* Khi đó mỗi pixel của dst sẽ mang giá trị pixel của scr tại các vị trí theo map\_x và map\_y.

**Ví dụ:**

Src = array([[230, 45, 153, 233, 172, 153, 46, 29],

[172, 209, 186, 30, 197, 30, 251, 200],

[175, 253, 207, 71, 252, 60, 155, 124],

[114, 154, 121, 153, 159, 224, 146, 61],

[ 6, 251, 253, 123, 200, 230, 36, 85],

[ 10, 215, 38, 5, 119, 87, 8, 249],

[ 2, 2, 242, 119, 114, 98, 182, 219],

[168, 91, 224, 73, 159, 55, 254, 214]])

map\_y = np.array([[0, 1], [2, 3]])

map\_x = np.array([[5, 6], [7, 10]])

dst(0,0) = src (map\_x(0,0) , map\_y(0,0)) = src (5,0) = 153.

* Để thực hiện làm méo ảnh trên python, tạo 2 ma trận zeros map\_x và map\_y, gán các giá trị của map\_x và map\_y theo công thức (tham khảo trên mạng):

map\_x[y,x] = x + int(20.0 \* math.sin(2 \* 3.14 \* x / 150))

map\_y[y,x] = y + int(20.0 \* math.cos(2 \* 3.14 \* y / 150))

trong đó x được chạy từ 0 -> width, y chạy từ 0->height của frame gốc.

* Sử dụng hàm cv2.remap() để gán các giá trị pixel frame gốc cho frame dst theo công thức trên đồng thời vẫn giữ được màu sắc. Nếu việc biến đổi khiến giá trị pixel tại x, y của 2 map lớn hơn width và height của frame gốc thì dst(x,y) = 0.

c. Kết quả

Kết quả trước và sau khi áp dụng hiệu ứng :



**TÀI LIỆU THAM KHẢO**

[1] <https://stackoverflow.com/questions/46520123/how-do-i-use-opencvs-remap-function?fbclid=IwAR3o50c0gYrvoofa5LA0ZRWU-eTBr3R1C2yNKBMJyd9V1uc7dJ1dwR5HL8Y>

[2] <https://subscription.packtpub.com/book/application_development/9781785283932/1/ch01lvl1sec16/image-warping?fbclid=IwAR1ptiyV4JwX0OUmB8IDthGgW8zISmhDp5gKf77TiQvAt6Df8WC5AhxDJcI>

[3] <https://minhng.info/tutorials/frame-subtraction-phat-hien-chuyen-dong-video.html?fbclid=IwAR0KIFqTAv_L5fKa-21AaG8seGvjFYwY6XIdgCcKnqwm-aOMG8Z8VXavKhE#:~:text=%C3%9D%20t%C6%B0%E1%BB%9Fng%20c%E1%BB%A7a%20Frame%20Subtraction,mi%E1%BB%81n%20gi%C3%A1%20tr%E1%BB%8B%20mong%20mu%E1%BB%91n.&text=Khi%20%C3%A1p%20d%E1%BB%A5ng%20Frame%20Subtraction,%C4%91%E1%BB%83%20t%C3%ACm%20%C4%91i%E1%BB%83m%20kh%C3%A1c%20bi%E1%BB%87t>

[4] <https://github.com/Thekegman/noodledance?fbclid=IwAR3Fnh7YDhON19CTSrKktgQ8DA6kiWwkWVkyuZoyf6Nm-wNm3galXIM1pzY>

[5] <https://medium0.com/m/global-identity?redirectUrl=https%3A%2F%2Ftowardsdatascience.com%2Fbuilding-an-image-cartoonization-web-app-with-python-382c7c143b0d&fbclid=IwAR0ycoEwPOz4YNT--KXWHYsfyYduQE2GGfQuLOVGW2-kzJqnzEyj1F8sk0o>