13

# HỌ TÊN:NGUYỄN LÊ HOÀNG THANH KÝ TÊN:

**TRƯỜNG: HVCNBCVT LỚP:D17CQPU01-N MSSV:N17DCPT058**

# MÔN: KỸ THUẬT LẬP TRÌNH ỨNG DỤNG ĐA PHƯƠNG TIỆN NGÀY: 27/10/2021… (BUỔI HỌC SỐ 7)

**NHIỆM VỤ BUỔI 7: CỬA SỔ TRƯỢT (**SLIDING WINDOW**)**

**HƯỚNG DẪN**

## CỬA SỔ TRƯỢT (SLIDING WINDOW) => nhận diện ảnh và trích đặc trưng ảnh

**===================**

+ Window size (kernel size): kích thước cửa sổ trượt = là số lẻ, tức 3, 5, 7, …

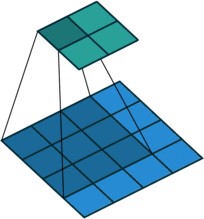
+ Padding: số pixel mở rộng thêm vào ảnh đầu vô.

+ Stride: khoảng cách lần trượt.

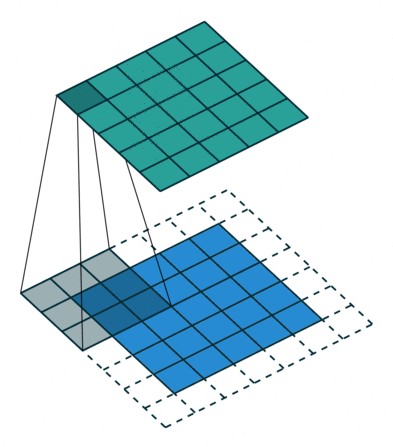
+ Dilation: khoảng cách của mỗi pixel trên cửa sổ

====================đơn vị đo pixel=========

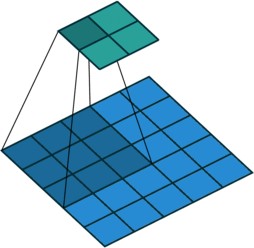
kernel\_size=3, padding=0, stride=1, dilation=1



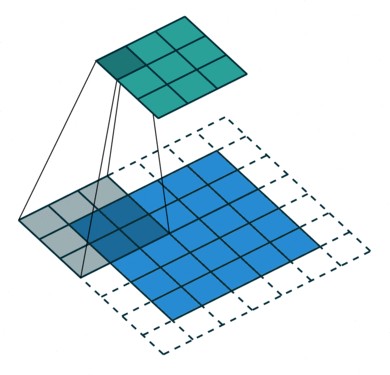
kernel\_size=3, padding=1, stride=1, dilation=1



kernel\_size=3, padding = 0, stride = 2, dilation=1



kernel\_size=3, padding = 1, stride = 2, dilation=1



## GIẢI THÍCH VỀ MẢNG PHÁT SINH TỪ TẬP HỢP enumerate

y\_pos = [v for idx, v in enumerate(list(range(k\_half, h-k\_half))) if idx % stride == 0]

enumerate = phát sinh tập hợp / list = danh sách : trong khoảng = range với ĐK idx chia hết cho k/c trượt

k\_half -> h – k\_half

v là phần tử chạy (tương tự i trong for); v bắt đầu là idx

**FULL CODE (tham khảo)**

import cv2

import numpy as np

INPUT\_IMAGE = "HE.jpg" # HẰNG = lưu Hình Input

def apply\_sliding\_window(img, kernel, padding=0, stride=1): #padding = số pixel mở rộng ảnh input

#Stride: khoảng cách trượt. #kernel: kích thước cửa sổ trượt

h, w = img.shape[:2] # lấy chiều cao & chiều rộng ảnh

img\_p = np.zeros([h+2\*padding, w+2\*padding]) # Numpy = lập ma trận trống zero = chiều cao|| rộng + 2 lần pixel mở rộng

img\_p[padding:padding+h, padding:padding+w] = img # gán ảnh vào khung ma trận nêu trên

kernel = np.array(kernel) # lập cửa sổ trượt

assert len(kernel.shape) == 2 and kernel.shape[0] == kernel.shape[1]

# assert =kiểm tra đàm bảo (tương tự if) square kernel = số chiều = 2 và 2 chiều bằng nhau

assert kernel.shape[0] % 2 != 0

# kernel size is odd number = số chiều cao là lè (đương nhiên chiều w cũng lẻ ) = khung vuông 2 chiều có kích thước là số lẻ

k\_size = kernel.shape[0] # chiều cao của khung trượt k\_half = int(k\_size/2) # nửa chiều cao của khung trượt

y\_pos = [v for idx, v in enumerate(list(range(k\_half, h-k\_half))) if idx % stride == 0] # tập vị trí y (dọc)

x\_pos = [v for idx, v in enumerate(list(range(k\_half, w-k\_half))) if idx % stride == 0] # tập vị trí x (ngang)

new\_img = np.zeros([len(y\_pos), len(x\_pos)]) #lập ma trận khung trồng (chuần bị điền ảnh vào khung trượt)

for new\_y, y in enumerate(y\_pos): #chạy vị trí y trong tập vị trí y đã xác định ở trên

for new\_x, x in enumerate(x\_pos): #chạy vị trí x trong tập vị trí x đã xác định ở trên if k\_half == 0: #bắt đầu điền ảnh từ vị trí 1/2 ảnh đầu (đã xác định ở trên)

pixel\_val = img\_p[y, x] \* kernel # element-wise multiply = nhân -> mở rộng phần ảnh

else:

pixel\_val = np.sum(img\_p[y-k\_half:y-k\_half+k\_size, x-k\_half:x-k\_half+k\_size] \* kernel)

# mở rộng = tích vô hướng 2 vector

new\_img[new\_y, new\_x] = pixel\_val

# gán vị trí ảnh phù hợp vào vị mới => chuẩn trượt tiếp

return new\_img

# HÀM TRƯỢT 3 GAM MÀU = KHÔNG GIAN MÀU RGB

def apply\_sliding\_window\_on\_3\_channels(img, kernel, padding=0, stride=1): #làm mờ ảnh

layer\_blue = apply\_sliding\_window(img[:,:,0], kernel, padding, stride) layer\_green = apply\_sliding\_window(img[:,:,1], kernel, padding, stride) layer\_red = apply\_sliding\_window(img[:,:,2], kernel, padding, stride)

new\_img = np.zeros(list(layer\_blue.shape) + [3])

new\_img[:,:,0], new\_img[:,:,1], new\_img[:,:,2] = layer\_blue, layer\_green, layer\_red return new\_img

# GỌI HÀM VỚI THAM SỐ THỰC

if name == " main ":

img = cv2.imread(INPUT\_IMAGE) cv2.imshow("Hinh goc",img)

new\_img = apply\_sliding\_window\_on\_3\_channels(img, kernel=[[1]], padding=0, stride=2) # làm mở RGB

cv2.imshow("Hinh moi",new\_img) cv2.imwrite('H\_new.jpg', new\_img)

print('Kich thuoc hinh GOC:', img.shape) # kích thước hình gốc print('Kich thuoc hinh moi:', new\_img.shape) #kích thước hình mới sau xử lý

# print('Luu hinh @ 3DF\_new.jpg') # in tên file hình đã lưu

cv2.waitKey(0) cv2.destroyAllWindows()

print(' ')

lighten\_blur\_img = apply\_sliding\_window\_on\_3\_channels(img, kernel=[[0.33, 0.33, 0.33], [0.33, 0.33, 0.33], [0.33, 0.33, 0.33]], padding=1, stride=1)

cv2.imshow("Hinh lighten\_blur",lighten\_blur\_img)

cv2.imwrite('H\_new\_blur.jpg', lighten\_blur\_img)

print('Kích thước hình gốc:', img.shape)

print('Kích thước sau khi lighten\_blur:', lighten\_blur\_img.shape)

# print('Luu @ 3DF\_new\_blur.jpg') cv2.waitKey(0) cv2.destroyAllWindows()

**chú ý:**

. Dùng chỉ số để lấy phần tử mảng (hoặc ký tự trong chuỗi); chỉ số âm tính từ phải Lấy mảng / chuỗi con: từ i -> j a[i:j]

Lấy mảng / chuỗi con: từ i đến cuối a[i:] Lấy mảng / chuỗi con: từ đầu đến i a[:i]

. in lặp lại chuỗi s với n lần : s\*n => có thể “lồng” việc này vào nhau:

VD m = 3; n = 5; print((“\*”\*m+”\n”)\*n) => in ma trận dấu \* với m cột và n dòng

. Định dạng chuỗi: str.format(): VD: a = 9 b = 10 str.format(“{} + {} = {}”,a, b, a+b)

str.format(“{2} = {1} – {0}”, a, b, b – a)

str.format(“{x} - {y} = {z}”, x =b, y = a, z = b -a)

str.format(“{1} - {0} = {z}”, a, b, z = b -a)

-> thêm định dạng: str.format(“{0} / {1} = {z:.2f}”, a, b, z = a/b)

-> dùng từ điển: pr = {“A”: 5, “B”: 9, “C”:15}

str.format("AA = {0[A]}; CC = {0[C]}" , pr)

**BÀI TẬP**

**Bài 1:** Viết chương trình cho NSD nhập lựa chọn và tương ứng thực hiện: Thực hiện bài tập mẫu nêu trên:

Dùng khung trượt (Slide Windows) trên ảnh gốc để nhận diện ảnh phát hiện phát hiện đặc trưng

1. Chọn 1 trong 2 phương án:

+ Nạp file Ảnh : liệt kê danh sách tên file (có chỉ số thứ tự)

Cho người sử dụng chọn tên file Ảnh Input bằng 1 trong 2 phương án:

-> Nhập chỉ số thứ tự của file đó OR

-> Dùng giọng nói (đọc số thứ tự của file) để nạp {tham khảo bài: Voice Assistant}

+ Chụp ảnh từ Camera của PC

1. NHẬP THƯ MỤC LƯU SẢN PHẨM các file ảnh Output theo từng ảnh đầu vào

+ Nhập tên thư mục

+ Mẫu tên file: …

1. CHO NSD NHẬP CÁC THAM SỐ KHUNG TRƯỢT

+ Window size (kernel size): kích thước cửa sổ trượt = là số lẻ, tức 3, 5, 7, …

+ Padding: số pixel mở rộng thêm vào ảnh đầu vô (thường 0, 1, 2, 3, …)

+ Stride: khoảng cách lần trượt (thường 0, 1, 2, 3, …)

+ Dilation: khoảng cách của mỗi pixel trên cửa sổ (thường 0, 1, 2, 3, …)

1. XỬ LÝ CÁC ẢNH TRONG THƯ MỤC Output {xem bài trước}
   1. Cắt ảnh theo thông tin về kích thước ảnh được cắt do NSD nhập vào
   2. Xoay ảnh theo yêu cầu của NSD (NSD nhập yêu câu)
   3. Thay đổi kích thước ảnh theo yêu cầu của NSD
2. CHO NSD LỰA CHỌN DẠNG CỬA SỔ TRƯỢT
   1. …= apply\_sliding\_window\_on\_3\_channels(img, kernel=[[1]], padding=0, stride=2) # làm mở RGB
   2. …. = apply\_sliding\_window\_on\_3\_channels(img, kernel=[[0.33, 0.33, 0.33], [0.33, 0.33, 0.33], [0.33, 0.33, 0.33]], padding=1, stride=1)
   3. …
3. # -\*- coding: utf-8 -\*-
4. """
5. Created on Mon Nov  1 11:01:37 2021
7. @author: thanh
8. """
9. **import** cv2
10. **import** speech\_recognition as sr
11. **import** os
12. **import** numpy as np
13. """
14. Khai báo hằng
15. """
16. path = '13NguyenLeHoangThanh'
17. """
18. Nạp fileVideo bằng giọng nói
19. """
20. **def** nlht\_nhapfile():
21. **print**("""
22. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
23. Chọn chế độ nhập:
24. 1. Nhập từ image
25. 2. Nhập từ camera
26. """)
27. c=input("Chọn: ")
28. **if** c=="1":
29. anh = []
30. classNames  = []
31. myList = os.listdir(path)#danh sách tệp tin
32. #Lấy anh và tên
33. **for** cl **in** myList:
34. img = cv2.imread(f'{path}/{cl}',0)
35. anh.append(img)
36. classNames.append(os.path.basename(cl))#lấy tên tệp tin
37. **for** i **in** classNames:#liệt kê file
38. **print**("{} : {}".format(classNames.index(i),i))
39. **print**("""
40. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
41. Chọn chế độ nhập:
42. 1. Dùng giọng nói
43. 2. Nhập số thứ tự
44. """)
45. ch=input("Chọn: ")
46. **if** ch=="1":
47. **while** (True):
48. **print**("Hãy nói số thứ tự video bạn muốn mở(ví dụ: số 0)")
49. r = sr.Recognizer()
50. with sr.Microphone() as source:
51. **print**("Điều chỉnh tiếng ồn ")
52. r.adjust\_for\_ambient\_noise(source, duration=1)
53. **print**("Nói bằng tiếng Việt đi bạn 5s sau sẽ in ra Text...")
54. # read the audio data from the default microphone
55. audio\_data = r.record(source, duration=5)
56. **print**("Kết quả nhận diện...")
57. # convert speech to text
58. **try**:
59. query = r.recognize\_google(audio\_data,language="vi")
60. **except**:
61. **continue**
62. **print**(query)
63. query=query.strip('số ')#bỏ từ 'số'
64. **try**:
65. query=int(query)
66. **break**
67. **except**:
68. **continue**
69. imgout = cv2.imread("{}/{}".format(path,classNames[query]),cv2.IMREAD\_COLOR)
70. **if** ch=="2":
71. query=int(input("Số thứ tự: "))
72. imgout = cv2.imread("{}/{}".format(path,classNames[query]),cv2.IMREAD\_COLOR)
73. **if** c=="2":
74. cap=cv2.VideoCapture(0)
75. **while**(cap.isOpened()):
76. ret, frame = cap.read()
77. cv2.imshow('Bam q de chup',frame)
78. **if** cv2.waitKey(10) & 0xFF == ord('q'):
79. cv2.imwrite('thanh\_camera.jpg',frame)
80. **break**
81. cap.release()
82. cv2.destroyAllWindows()
83. imgout = cv2.imread('thanh\_camera.jpg',cv2.IMREAD\_COLOR)
84. **return** imgout
85. **def** apply\_sliding\_window(img, kernel, padding=0, stride=1):
86. h, w = img.shape[:2]
87. img\_p = np.zeros([h+2\*padding, w+2\*padding])
88. img\_p[padding:padding+h, padding:padding+w] = img
89. kernel = np.array(kernel) # lập cửa sổ trượt
90. **assert** len(kernel.shape) == 2 **and** kernel.shape[0] == kernel.shape[1]
91. **assert** kernel.shape[0] % 2 != 0
93. k\_size = kernel.shape[0]
94. k\_half = int(k\_size/2)
96. y\_pos = [v **for** idx, v **in** enumerate(list(range(k\_half, h-k\_half))) **if** idx % stride == 0]
97. x\_pos = [v **for** idx, v **in** enumerate(list(range(k\_half, w-k\_half))) **if** idx % stride == 0]
99. new\_img = np.zeros([len(y\_pos), len(x\_pos)])
100. **for** new\_y, y **in** enumerate(y\_pos):
101. **for** new\_x, x **in** enumerate(x\_pos):
102. **if** k\_half == 0:
103. pixel\_val = img\_p[y, x] \* kernel
104. **else**:
105. pixel\_val = np.sum(img\_p[y-k\_half:y-k\_half+k\_size, x-k\_half:x-k\_half+k\_size] \* kernel)
106. new\_img[new\_y, new\_x] = pixel\_val
107. **return** new\_img
108. **def** apply\_sliding\_window\_on\_3\_channels(img, kernel, padding=0, stride=1):   #làm mờ ảnh
109. layer\_blue = apply\_sliding\_window(img[:,:,0], kernel, padding, stride)
110. layer\_green = apply\_sliding\_window(img[:,:,1], kernel, padding, stride)
111. layer\_red = apply\_sliding\_window(img[:,:,2], kernel, padding, stride)
112. new\_img = np.zeros(list(layer\_blue.shape) + [3])
113. new\_img[:,:,0], new\_img[:,:,1], new\_img[:,:,2] = layer\_blue, layer\_green, layer\_red
114. **return** new\_img
115. **def** nhapthamso():
116. **print**("Nhập tham số :")
117. **print**("+Window size (kernel size): kích thước cửa sổ trượt = là số lẻ, tức 3, 5, 7, …")
118. kernel=int(input())
119. **print**("+ Padding: số pixel mở rộng thêm vào ảnh đầu vô (thường 0, 1, 2, 3, …)")
120. padding=int(input())
121. **print**("+ Stride: khoảng cách lần trượt (thường 0, 1, 2, 3, …)")
122. stride=int(input())
123. **print**("+ Dilation: khoảng cách của mỗi pixel trên cửa sổ (thường 0, 1, 2, 3, …)")
124. dilation=int(input())
125. **return** kernel,padding,stride,dilation
126. """
127. def thaydoikichthuoc(img,ty\_le):#kernel= 1 ,stride= 100/ti le=> anh thu nho theo ti le
128. padding=0
129. p=ty\_le/100
130. (kernel,stride)=p.as\_integer\_ratio()
131. newimg=apply\_sliding\_window\_on\_3\_channels(img, [[kernel]],padding,stride)
132. return newimg
133. """
134. **def** rotate(img,g):#xoay ảnh
135. (h, w,d) = img.shape
136. center = (w // 2, h // 2)
137. M = cv2.getRotationMatrix2D(center, g, 1.0)
138. rotated = cv2.warpAffine(img, M, (w, h))
139. **return** rotated
140. **def** cutimg(img,h1,h2,w1,w2):#cắt ảnh
141. p= img[h1:h2,w1:w2]
142. **return** p
143. **def** resize(img,s):#thay đổi kích thước
144. (h, w,d) = img.shape
145. w=int(w\*s/100)
146. h=int(h\*s/100)
147. dim=(w,h)
148. resized = cv2.resize(img, dim)
149. **return** resized
150. **def** main():
151. img=nlht\_nhapfile()
152. img\_name=input("Tên file lưu: ")
153. data\_adress=input("Thư mục lưu: ")
154. os.makedirs(data\_adress, exist\_ok=True) #Tao thu muc
155. **print**("""
156. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
157. Chọn yêu cầu xử lý :
158. 1 .Cắt ảnh theo thông tin về kích thước ảnh được cắt do NSD nhập vào
159. 2 .Xoay ảnh theo yêu cầu của NSD (NSD nhập yêu câu)
160. 3 .Thay đổi kích thước ảnh theo yêu cầu của NSD
161. 4 .Chinh anh theo kernel,padding,stride nhap vao
162. 5 .Lam mo anh
163. """)
165. ch  = input("Chọn :  ")
166. **if** ch=="1":
167. h1=int(input("Cắt ảnh từ chiều cao:"))
168. h2=int(input(" đến "))
169. w1=int(input("Cắt ảnh từ chiều rộng :"))
170. w2=int(input(" đến "))
171. newimg=cutimg(img, h1, h2, w1, w2)
172. **if** ch=="2":
173. g=float(input("Góc xoay theo độ(>0 quay ngược kim đồng hồ ):"))
174. newimg=rotate(img, g)
175. **if** ch=="3":
176. s=int(input("Tỉ lệ ảnh: "))
177. newimg=resize(img, s)
178. **if** ch=="4":
179. (kernel,padding,stride,dilation)=nhapthamso()
180. newimg=apply\_sliding\_window\_on\_3\_channels(img, [[kernel]],padding,stride)
181. **if** ch=="5":
182. newimg=apply\_sliding\_window\_on\_3\_channels(img, [[1]],padding=0,stride=2)
183. **print**("Dang luu...")
184. cv2.imwrite(os.path.join(data\_adress, '{}.jpg'.format(img\_name)), newimg)
185. **print**("Hoan thanh xu li")
186. main()