



UNIVERSIDAD DEL BÍO-BÍO

12 Recortar Imagen webcam

Tutorial Procesamiento de Imagen con webcam

Estudiantes Practicantes:

Javiera Gutiérrez

Javiera Henríquez

Teresa Vidal

Profesor:

Luis Vera

Laboratorio CIMUBB

Recortar imagen webcam

Para este proyecto tomaremos una foto utilizando la webcam y la recortaremos indicando las coordenadas x e y que le asignemos.

Primero, creamos otro archivo .py y le añadimos las siguientes librerías.

Creamos nuestra ventana, le añadimos tamaño, que el usuario no pueda cambiar el tamaño y el título a la ventana.

```
09 Recortar Imagen webcam.py > ...
1  import tkinter as tk
2  from tkinter import *
3  from PIL import Image
4  from PIL import ImageTk
5  import imutils
6  import cv2
7
8  # Crea ventana, define tamaño y título
9  ventana = tk.Tk()
10 ventana.geometry("1070x450")
11 ventana.resizable(0,0)
12 ventana.title("Recortar foto webcam")
13
```

Añadimos las funciones para iniciar la cámara web y tomar foto que vimos en los tutoriales anteriores:

```
14 #Funciones cámara web
15 def camara():
16     global capture
17     capture = cv2.VideoCapture(0)
18     iniciar()
19
20 def iniciar():
21     global capture
22     if capture is not None:
23         ret, frame = capture.read()
24         if ret == True:
25             frame = imutils.resize(frame, width=311)
26             frame = imutils.resize(frame, height=241)
27             ImagenCamara = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
28             im = Image.fromarray(ImagenCamara)
29             img = ImageTk.PhotoImage(image= im)
30             LImagen.configure(image= img)
31             LImagen.image = img
32             LImagen.after(1,iniciar)
33         else:
34             LImagen.image = ""
35             capture.release()
36
37 #Función para tomar una foto
38 def Capturar():
39     global Captura
40     camara = capture
41     return_value, image = camara.read()
42     frame = imutils.resize(image, width=301)
43     frame = imutils.resize(frame, height=221)
44     Captura = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
45     im = Image.fromarray(Captura)
46     img = ImageTk.PhotoImage(image= im)
47     LImagenROI.configure(image= img)
48     LImagenROI.image = img
49
```

Agregamos la función de que detecte en que coordenada se hace clic con el mouse, que vimos en el tutorial anterior:

```
49
50 def mostrar_coordenadas(event):
51     coordenadas['text']=f'x = {event.x}    y = {event.y}'
52
```

Creamos la función para recortar la imagen, obteniendo las varias x1, y1, x2 e y2 ingresadas por el usuario, y creamos una nueva variable con imagen recortada.

La transformamos para que sea compatible para mostrarse en otro recuadro gris.

```
53  def recortar():
54      global ImgRec
55      vx1 = int(x1.get())
56      vy1 = int(y1.get())
57      vx2 = int(x2.get())
58      vy2 = int(y2.get())
59      ImgRec = Captura[vx1:vx2, vy1:vy2]
60      Im = Image.fromarray(ImgRec)
61      ImRec = ImageTk.PhotoImage(image=Im)
62      LImagenRecorte.configure(image= ImRec)
63      LImagenRecorte.image = ImRec
64
```

Creamos los botones para accionar las funciones **camara()**, **Capturar()** y **recortar()**, asignándoles un tamaño y posición.

```
66  #Botones
67  BCamara = tk.Button(ventana, text="Iniciar cámara", command=camara)
68  BCamara.place(x=80,y=330,width=90,height=23)
69  BCapturar = tk.Button(ventana, text="Tomar foto", command=Capturar)
70  BCapturar.place(x=220,y=330,width=91,height=23)
71  BRecortar = tk.Button(ventana, text="Recortar", command=recortar)
72  BRecortar.place(x=840,y=400,width=80,height=23)
```

Creamos tres cuadros grises, el primero se ocupará para mostrar la imagen de la webcam, el segundo para la foto y el tercero para mostrar la imagen recortada.

```
74  #Cuadros de imagen gris
75  LImagen = tk.Label(ventana, background="gray")
76  LImagen.place(x=50,y=50,width=300,height=240)
77  LImagenROI = tk.Label(ventana, background="gray")
78  LImagenROI.place(x=390,y=50,width=300,height=240)
79  LImagenRecorte = tk.Label(ventana, background="gray")
80  LImagenRecorte.place(x=730,y=50,width=300,height=240)
```

Se le añade el **bind** con la función que detecta hace clic al cuadro gris **LImagenRoi**, que es donde se muestra la foto tomada.

```
82     #Coordenadas x e y
83     LImagenROI.bind('<Button-1>', mostrar_coordenadas)
84
```

Se agregan **Label** para el texto de título Coordenada, un texto para que muestre las coordenadas detectadas y texto indicando x1, y1, x2 e y2.

```
84
85     #Label - Texto
86     coordenadasTitulo = tk.Label(ventana, text="Coordenadas")
87     coordenadasTitulo.place(x=505, y=310)
88     coordenadas = tk.Label(ventana, text="")
89     coordenadas.place(x=495, y=330)
90     Lx1 = tk.Label(ventana, text="x1")
91     Lx1.place(x=790, y=330)
92     Ly1 = tk.Label(ventana, text="y1")
93     Ly1.place(x=890, y=330)
94     Lx2 = tk.Label(ventana, text="x2")
95     Lx2.place(x=790, y=360)
96     Ly2 = tk.Label(ventana, text="y2")
97     Ly2.place(x=890, y=360)
98
```

Se agregan **Spinbox** para que el usuario le dé un valor a cada coordenada, se les asigna un máximo para que no pasen el tamaño de la foto. Posteriormente se recorta la foto en base a estas coordenadas.

Se termina el programa con el código **ventana.mainloop()** para que todo se muestre correctamente.

```
98
99     #Spinbox - Escoger número coordenadas para recortar la foto
100    x1 = tk.Spinbox(ventana, from_=0,to=298)
101    x1.place(x=810, y=330, width=42, height=22)
102    y1 = tk.Spinbox(ventana, from_=0,to=239)
103    y1.place(x=910, y=330, width=42, height=22)
104    x2 = tk.Spinbox(ventana, from_=1,to=298)
105    x2.place(x=810, y=360, width=42, height=22)
106    y2 = tk.Spinbox(ventana, from_=1,to=239)
107    y2.place(x=910, y=360, width=42, height=22)
108
109    ventana.mainloop()
```

Al compilar y ejecutar, el programa se verá así:



Y al recortar se verá así:

