

Trabajo Practico 3

Luis Fernando Recalde

Instrucciones: Utilizar la imágenes utilizadas en el practico anterior o generar nuevas si es necesario.

1. Resolver

- (a) Aplicar la detección de esquinas Harris usando OpenCV para las 4 imágenes de las prácticas anteriores. Girar, trasladar y escalar la imagen y verificar que las esquinas detectadas son invariantes a la translación y rotación, no así con el escalado.

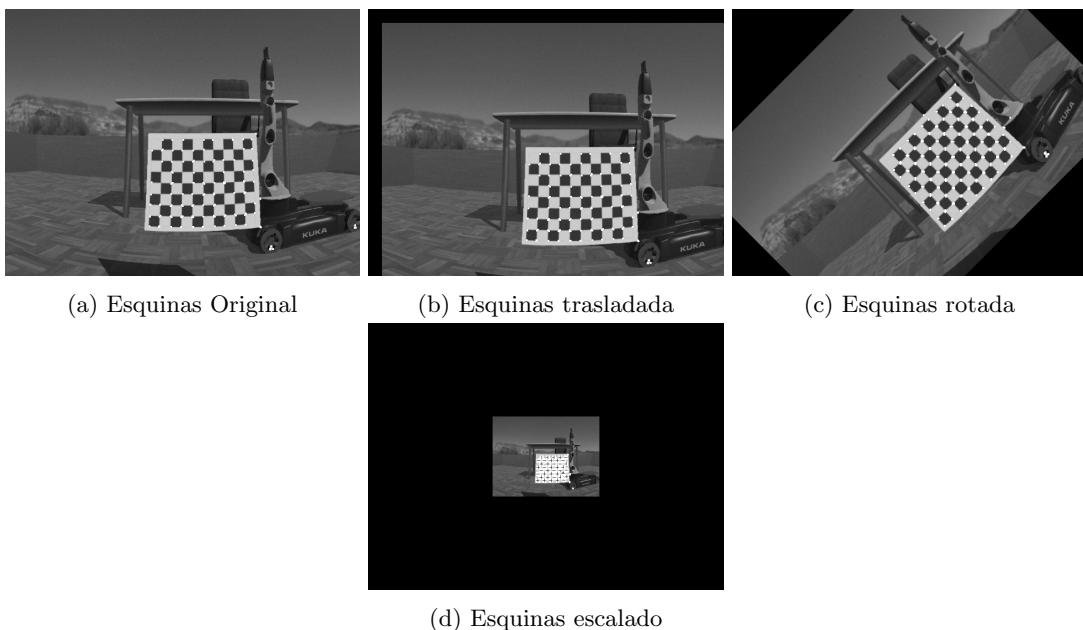


Figure 1: Detección Esquinas usando Harris.

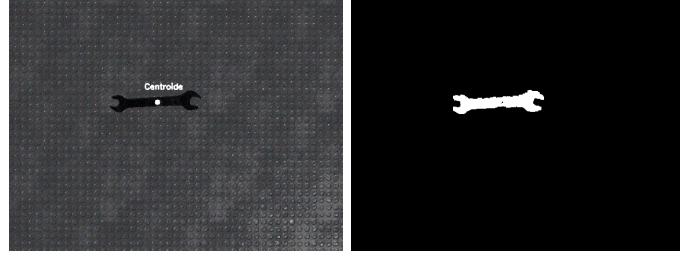
Los resultados de este algoritmo se muestran en Fig. 1, las esquinas detectadas en la imagen original se muestran en Fig. 1a, además rotando la imagen $\theta = 45^\circ$ se obtienen el mismo numero de esquinas mostrado en Fig. 1c, finalmente aplicando una translación a la imagen de $P_x = 25$ [pixels] y $P_y = 25$ [pixels] se obtuvieron los mismos resultados Fig. 1b.

Una variación significativa se presentó al escalar la imagen con un factor de $f = 0.3$, esto se presenta en Fig. 1d, donde el numero de esquinas aumentó significativamente.

- (b) Teniendo alguna imagen con un objeto de interés de un color, realizar la segmentación por color para separar ese objeto del resto de la imagen. Calcular el centroide.

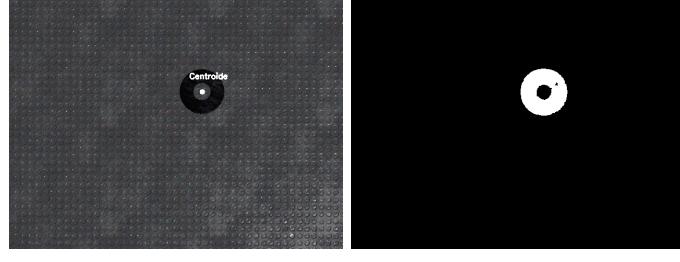
Los resultados de este filtro se muestran en Fig. 2 , se utilizó una banda transportadora y una serie de herramientas de color muy similar, las cuales deben ser segmentadas Fig. 2b y así calcular el centroide Fig. 2a.

Los resultados con otro objeto de interés se muestran en Fig. 3, se observa que se logra estimar el centroide del objeto de interés.



(a) Original con centroide (b) Objeto segmentado

Figure 2: Filtro de un primer objeto interés



(a) Original con centroide (b) Objeto segmentado

Figure 3: Filtro de un segundo objeto interés

- (c) Detectar las vocales en la siguiente imagen con texto. Contar cuantas vocales A hay.

Para este algoritmo se utilizó los momentos de Hu y la norma 2, la letras a detectar fueron A y Á, por lo que se generaron dos momentos deseados y finalmente una máscara de convolución la que nos ayudó a movernos entre las letras.

K	Á	L	U	L	É	B	I	L	F
T	U	C	Á	N	Á	T	Ú	N	I
O	P	L	U	P	L	W	J	R	O
O	L	L	I	L	U	Z	Á	I	L
R	C	K	S	O	T	T	W	J	E
E	S	T	O	R	N	I	D	O	M
Y	P	S	F	T	Á	G	H	R	M
Y	O	Ñ	I	M	R	Á	C	U	I
C	Z	L	O	C	Á	R	Á	C	N
O	I	M	I	S	T	O	L	M	G

Figure 4: Filtro de un segundo objeto interés

Los resultados de este algoritmo se muestran en Fig. 4, las letras detectadas son marcadas con color negro, el número final de letras detectadas es $n = 8$.

- (d) Seleccionar 5 hojas de árboles o plantas diferentes. Clasificar la hojas usando los momentos de Hu y la distancia Euclídea.

Para este algoritmo se utilizó una base de datos de imágenes de hojas (Hojas), de las cuales se seleccionaron 5.

Las imágenes utilizadas se muestran en Fig. 5, con su respectivo numero de identificación

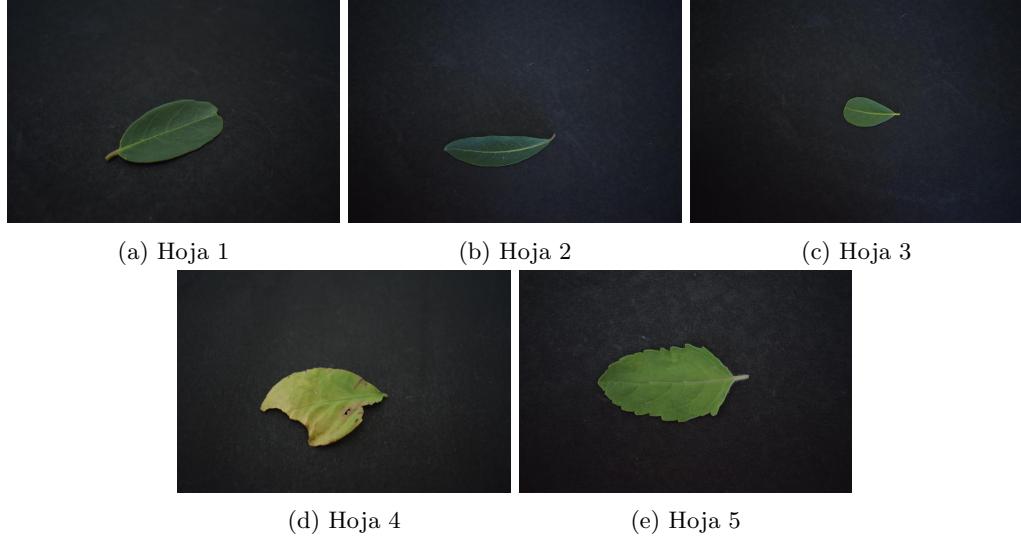


Figure 5: Hojas a clasificar

Los resultados se muestran en Fig. 6, se puede observar la imagen con su respectiva clasificación Fig. 6a, el objeto segmentado de la imagen en Fig. 5b y finalmente la distancia entre el frame y la base de datos Fig. 5c, notando que la distancia menor es a la clase que pertenece.

Los resultados consideran a la hoja 2, con una traslación de $P_x = 25$ [pixels] y $P_y = 25$ [pixels], una rotación de $\theta = 0^\circ$ y un escalado de $f = 1$.

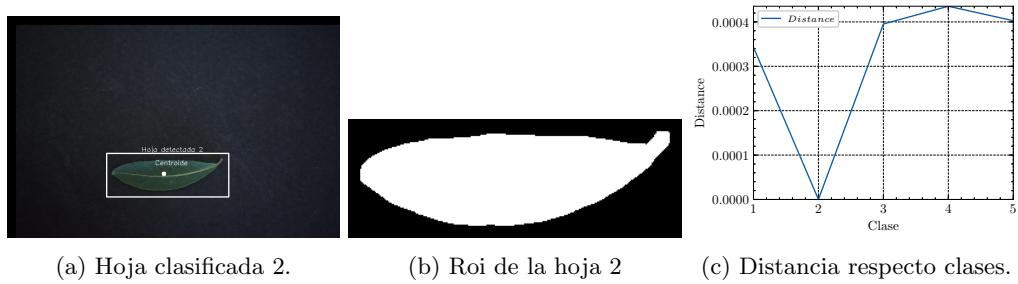


Figure 6: Hojas a clasificar 2.

Los resultados usando la hoja 4 se muestran en Fig. 7, la imagen con su respectiva clasificación se muestra en Fig. 7a, la región de interés se muestra en Fig. 7b y finalmente la distancia con respecto a la base de datos en Fig. 7c.

Los resultados consideran una traslación de $P_x = 25$ [pixels] y $P_y = 25$ [pixels], una rotación de $\theta = 180^\circ$ y un escalado de $f = 1.2$.

- (e) Verificar el algoritmo HOG de detección de personas usando las funciones de OpenCV.

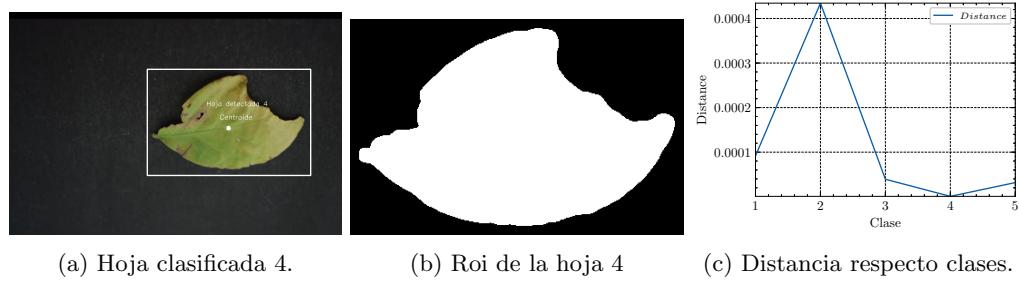


Figure 7: Hojas a clasificar 4.

Para este algoritmo se utilizo las funciones desarrolladas por Opencv `cv2.HOGDescriptor()`, además como método de clasificación se utilizo Support Vector Machine (SVM) el cual es una función implementada en el procesador de imágenes `hog.setSVMClassifier()`.

Los resultados se muestran en Fig. 8, los resultados no son los mejores ya que presenta falsos positivos, pero esto se produce al algoritmo el cual no es muy robusto, y debe tener un pos-procesamiento para garantizar una correcta detección de personas.

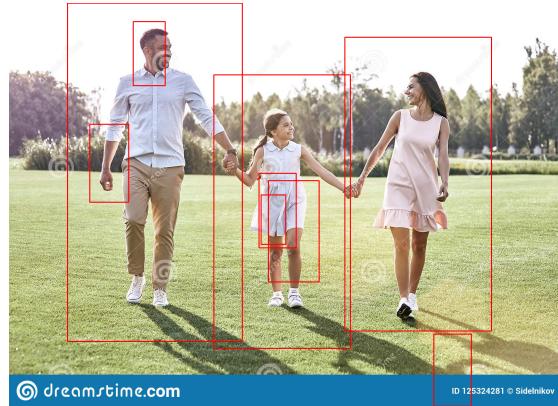


Figure 8: Detección personas algoritmo HOG.