Universidad de Chile
Facultad de Ciencias Físicas y Matemáticas
Departamento de Ingeniería Eléctrica

Tarea 1

DETECCIÓN Y TRACKING DE OBJETOS BASADO EN MODELO DE HUBEL Y WIESEL

Integrantes: Vincko Fabres Á.
Profesor: Claudio A. Pérez
Ayudantes: Gabriel Cubillos F.
Jhon Pilataxi
Auxiliar: Jorge Zambrano I.
Ayudante de Laboratorio: Juan Pérez C.
Fecha de entrega:Viernes 9 de Julio de 2021
Santiago, Chile

Reporte

Se anexa junto a este reporte el documento jupyter notebook utilizado.

Utilizando las librerías cv2, matplotlib, numpy, os y natsorted.

Implementación de patrones

i

Se lee la primera imagen y se explora un umbral adecuado para su transformación a imagen binaria, siendo el umbral para este caso el valor 175, como se muestra a continuación en Figura 1.

Figura 1: Imagen 1 de escala de grises a binaria

Posteriormente se busca la ubicación del patrón deseado 'p' y se recorta para su posterior procesamiento, resultando Figura 2.

Figura 2: Imagen de patron p obtenido de imagen binaria

ii

Se genera un kernel balanceado de 3x3 de la forma:

$$K = \begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix} \tag{1}$$

Con este kernel se convoluciona el patrón 'p' para obtener su filtro, resultando el filtro de Figura 3:

Figura 3: Filtro resultante 'p'

iii

Una vez obtenido el filtro, se genera simetría tanto de forma vertical como horizontal para aplicarlo a la imagen y obtener los puntos de detección del patrón, esto debido a que la convolución hace una rotación del filtro al aplicarlo, lo cual se corrige con lo mencionado anteriormente. La corrección anterior vista visualmente, comparando el antes y el después del filtro utilizado se visualiza en Figura 4.

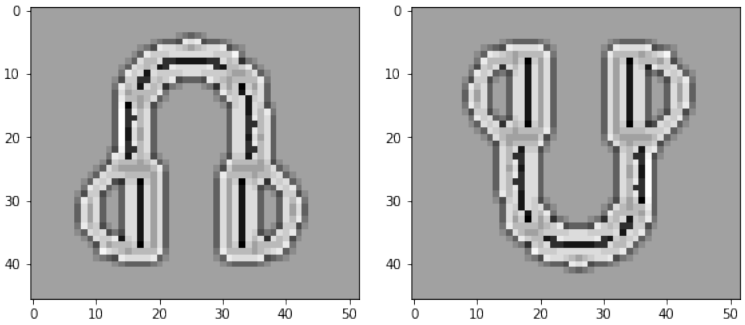

Figura 4: Corrección a filtro antes de aplicar detección

iv

Una vez realizada la corrección se convoluciona con la imagen para el posterior análisis y detección de patrones, la imagen binaria resultante de la convolución, en escala de grises y mediante mapas de calor se ven a continuación.

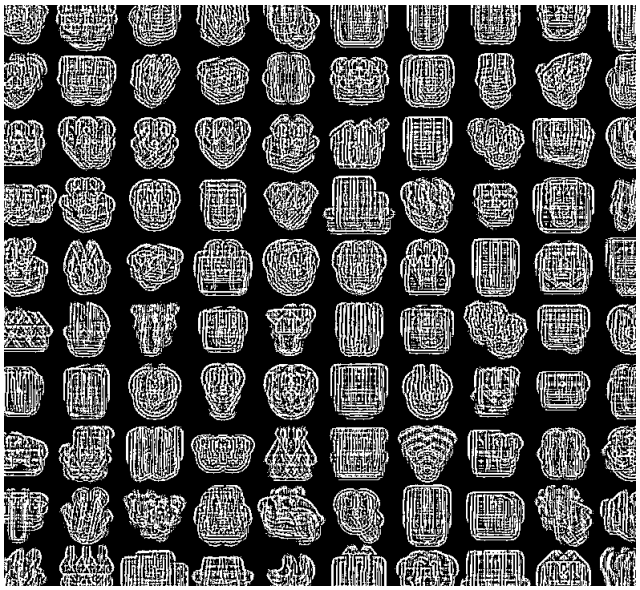

Figura 5: Convolución, imagen binaria

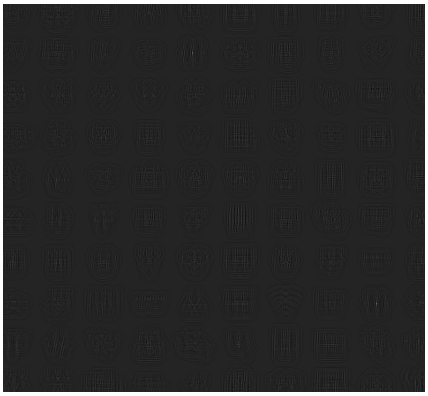

Figura 6: Convolución, escala de grises

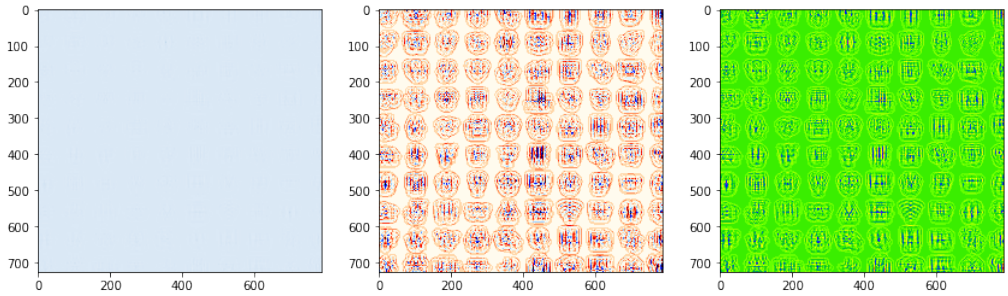

Figura 7: Convolución visualizada mediante mapas de calor

Analizando cada visualización es posible ejercer diversos comentarios; en primera instancia con la imagen binaria se logra visualizar el filtro aplicado a cada imagen, en escala de grises, reajustando

los valores es posible apreciar como se verá a continuación, que los valores peak (puntos blancos) son los centroides de los cuadros de detección, y por último dadas las diferentes exploraciones de mapas de calor disponibles, sirviendo de ejemplo el primero, se hace visible la poca visualización y por tanto entregta de información de la gran mayoría de estos, siendo excepciones los mapas 'flag' y 'prism' de matplotlib.

v

La matriz resultante de la convolución (que se visualiza como imagen) es pasada a un vector fila, ploteando sus valores en una gráfica como se muestra a continuación en Figura 8.

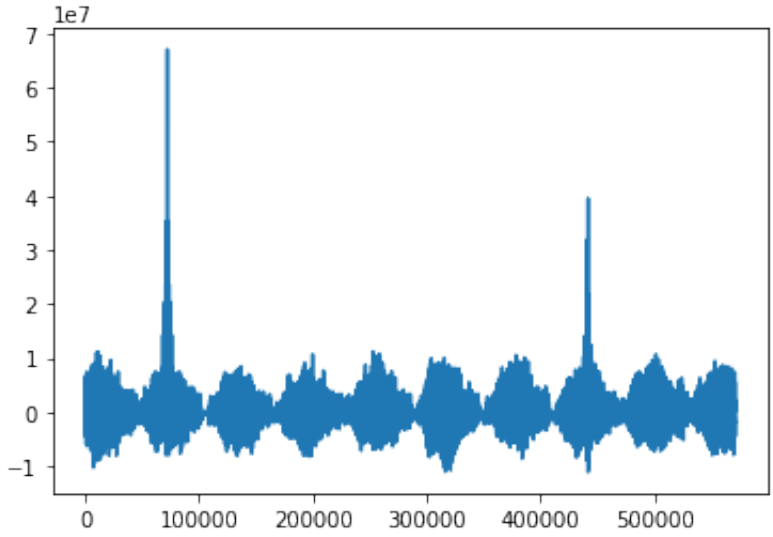

Figura 8: Matriz resultante en forma de vector

Dada la visualización de los valores es posible apreciar 2 picos, los que corresponden al lugar de la imagen donde la deteccion del patron es mayor, dada la imagen se decide de forma experimental un valor umbral para detecciones, en este caso el valor escogido corresponde a $3.5 * 10^7$, el cual al ser fijado resulta la siguiente gráfica:

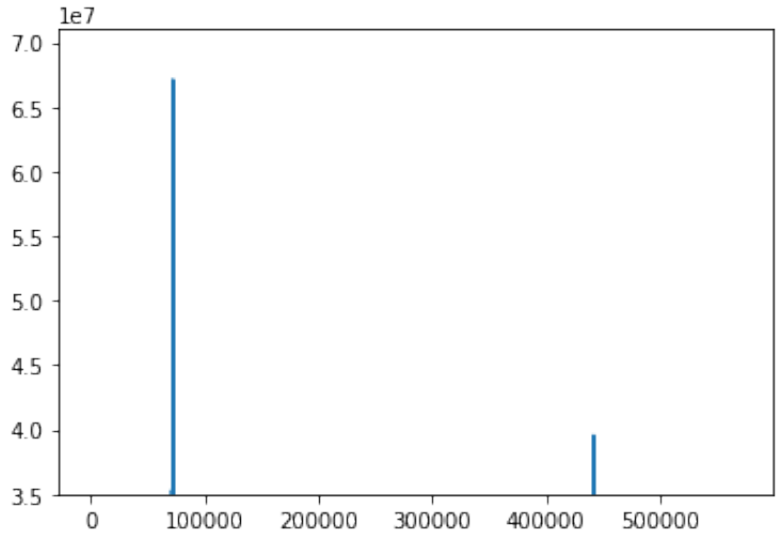

Figura 9: Matriz resultante en forma de vector con umbral aplicado

vi

Una vez aplicado el umbral, en la matriz se buscan los valores tales que su valor corresponda a uno igual o mayor que el umbral, resultando en este caso 2 valores, los cuales corresponden al centroide del cuadro de detección a aplicar, que posee dimensiones iguales al tamaño del filtro aplicado para generar las detecciones. Esto se realiza mediante la función *np.where*.

vii

Una vez realizado lo anterior se obtiene lo deseado,es decir, detección del patrón 'p' en imagen 1 como se muestra en Figura 10

Figura 10: Detección de patrón 'p'

Con este método, si se decide por ejemplo, utilizar de umbral un 50 % del valor máximo se genera un mayor número de detecciones, en este caso 4, las cuales corresponden a la superposición de cuadros, dado que el filtro se aplica con strides de valor 1, la convolución vecina al mayor valor también tendrá un valor alto. La visualización del ejemplo mencionado se ve en Figura 11:

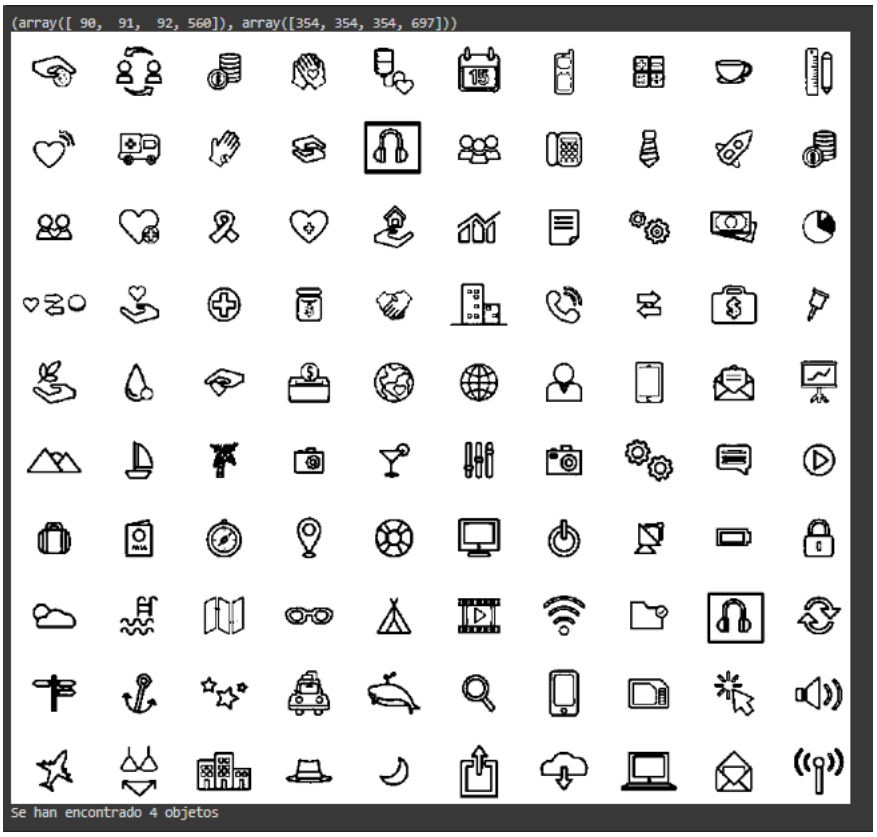

Figura 11: Detección con umbral al 50 %, generando superpsición

Utilizando el procedimiento explicado en las secciones anteriores, esta vez para imagen 2, siendo esta Figura 12:

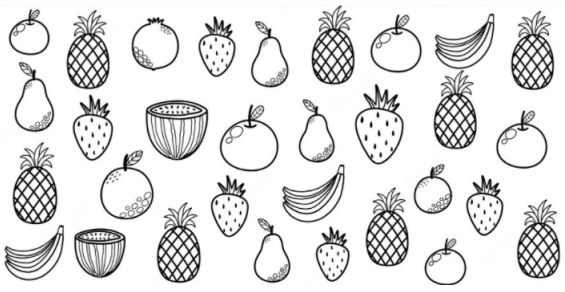

Figura 12: Imagen 2

Se busca detectar piñas y frutillas, para las piñas al aplicar un umbral del 20 % aún no es suficiente para generar la totalidad de detecciones, contando 37 objetos debido a la superposición, una vez bajado el umbral a 10 % es posible encerrar todas, contando con 161 detecciones, esto se visualiza a continuación:

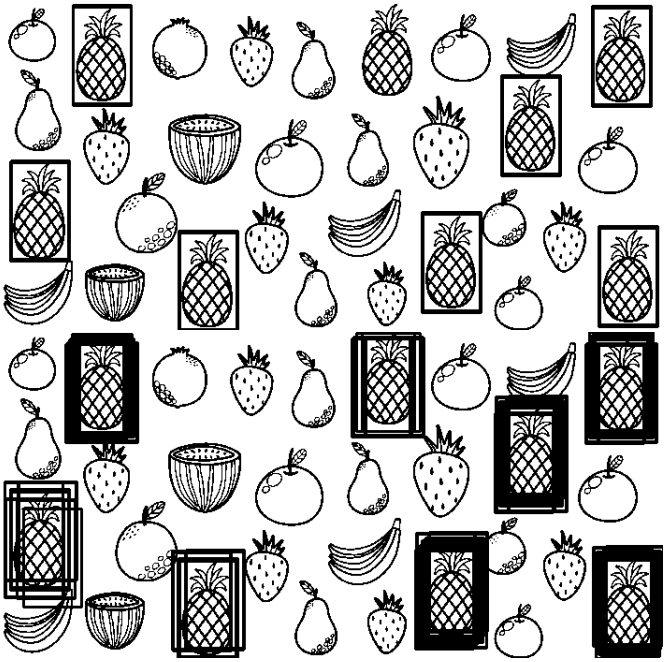

Figura 13: Detección de piñas

Para el caso de la frutilla con umbral al 50 % se detecta solo una, con 20 % se detectan 7 debido a la superposición, con 13 % 20 objetos, con un falso positivo y finalmente si se baja al 10 % 52 objetos con una cantidad cuantiosa de falsos positivos, los resultados mostrados en orden se muestran a continuación:

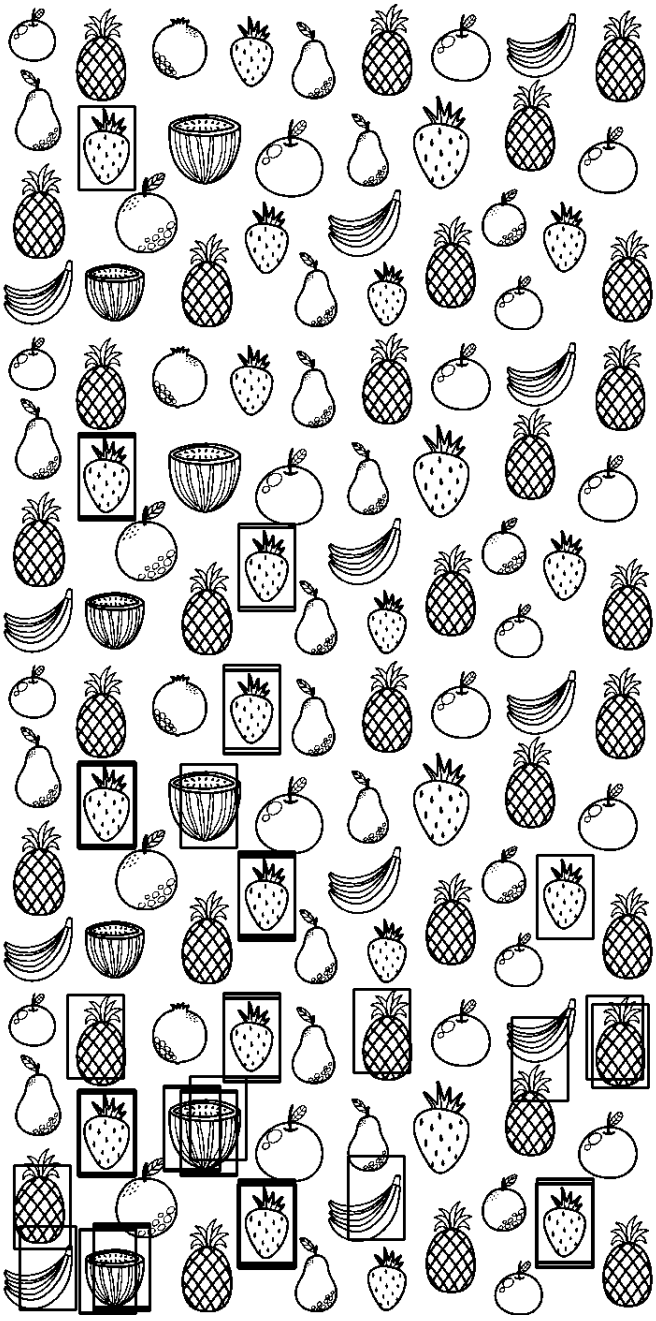

Figura 14: Detección de frutillas

Como es posible apreciar para el caso de las frutas, se da un contraste, la detección de piñas si bien posee superposición, todas las detecciones son verdaderos positivos con variaciones en la posición del centroide del cuadro de detección, en cambio, para las frutillas al llegar a un valor unmbral de 13 % del máximo es posible apreciar detecciones erroneas y omisiones de verdaderas frutillas, esto se debe al tamaño de las 2 frutillas ignoradas en la detección, dado que el método solo soporta detecciones de la imagen tal cual se ingresa, sin ser invariante a tamaño o rotaciones.

viii

Para corregir el problema a la superposición de detecciones se genera una función auxiliar la cual se cerciora se si 2 cuadros vecinos coinciden, utilizando de supuesto que las detecciones no deben estar superpuestas, por lo cual la distancia minima entre centroides corresponde al tamaño de la ventana del filtro.

ix

Una vez modificado el método, con la corrección mencionada, conocida como NMS (Non Maximum Supression), se aplican nuevamente a las imágenes para comparar resultados, por lo cual se aplicarán los mismos umbrales a cada detección.

Para el caso del patrón 'p' con umbral al 50 % ya no existe superposición, hayando solo 2 detecciones, como debería ser.

Para el caso de las piñas al 20 % se generan 9 detecciones, cuando deberían ser 7, esto se debe a que el método compara solo con el cuadro vecino y no de forma 'all vs all', al aplicar el umbral al 10 % se encuentran 37 objetos, siendo todos verdaderos positivos, aunque en total solo son 8, debido a lo explicado anteriormente.

Para el caso de las frutillas con umbral al 50 % no hay cambios, al 20 % se generan 2 detecciones, al 13 % se detectan 7 objetos, con superposición y 1 Falso Positivo y finalmente con umbral al 10 % existen 28 detecciones, de las cuales 11 son Falsos Positivos, habiendo superposición tanto de Falsos como Verdaderos Positivos.

Los resultados anteriores son explicados tanto a la implementación del metodo NMS, como limitaciones del filtrado y fijación de umbral adecuado, aunque cabe destacar que la mejoría con NMS es sustancial en comparación a la no utilización de este.

P4

i

Lo primero es leer el video y generar una secuencia de frames, los cuales se generan con funciones de la librería de cv2.

ii

Para realizar tracking se utiliza el método generado anteriormente, considerando esta vez múltiples entradas, por lo cual se aplica este metodo a cada frame y posteriormente se generará una secuencia de imágenes, lo que se traduce en un video.

iii

Para realizar los filtros se debe buscar la imagen correspondiente a la detección, en este caso 3 filtros, los cuales corresponden a la Posición del salto hacia delante de Mario en el frame 145, Goomba en el frame 297 y Cara de Mario hacia adelante frame 10.

Los recortes de imagen en formato imagen binaria son los siguientes:

Figura 15: Imágenes de tracking deseado

Dado como se implementa el método, todo el proceso siguiente se genera en un paso, el cual tarda por filtro alrededor de una hora, por lo que sólo se comentarán los resultados finales.

En primer lugar se puede apreciar el costo computacional del método dado el tiempo que tarda, y como se puede apreciar en cada tracking, este posee varias fallas.

Para el caso del tracking de cara se puede apreciar que esta es bien detectada sólo cuando Mario avanza en un sentido, de caso contrario la convolución posee valores mayores en el piso, dado que el método como se discutió anteriormente no es invariante a rotaciones o en este caso una simetría en eje vertical, la detección en piso se puede deber a la continuidad de valores tanto en cara como en este.

Para el caso de Goomba la detección es correcta cada vez que este esta presente, esto se puede deber a la forma, ya que es poco frecuente, en el caso en que Goomba está ausente se presentan variadas detecciones erroneas, debido a que el valor máximo es considerado frame a frame, por lo que debe ser el ruido presente y un ajuste de este umbral fue dificil de implementar dado el tiempo (aunque se pudo considerar un frame de Goomba ausente como representativo).

Para el último caso, donde Mario salta pasa el mismo problema que con la cara, el sentido afecta a la detección.