Ingeniería y
Competitividad
Vol 26 (1) 2024; doi: 10.25100/iyc.v26i1.13230

Vol 26 (1)

ISSN
0123-3033
e- 2027-8284

Artículo de investigación

Detección de acción humana para el control de inventario utilizando visión por computadora

Human action detection for inventory control using computer vision

Francisco Bernal-Baquero[1] Darwin E. Martínez[1]

[1]Departamento de Ingeniería, Universidad Sergio Arboleda, Bogotá, Colombia.

Resumen

La visión por computadora (VC) puede ser un proceso que facilite algunas tareas en la gestión de inventarios, por medio de este proceso se puede realizar un análisis permanente de un inventario y así mantener registro de todos los movimientos realizados, entregando un reporte instantáneo cuando sea requerido. Esto supone una mejora en la seguridad, ya que al mantener un control estricto de los elementos existentes en el inventario se puede saber si un elemento pertenece o no a un inventario o cuando se retira o agrega un elemento, tras esta necesidad de control de inventario, surge el objetivo de diseñar un sistema inteligente que pueda facilitar el control de inventarios. Mediante la combinación de 2 frameworks, se realiza la creación de un algoritmo capaz de realizar la identificación y conteo de objetos, así como la identificación de la mano para determinar cuándo se realiza una manipulación humana al inventario. Para lograr este objetivo, se utilizaron dos algoritmos: MediaPipe y YOLOv5 combinado con el dataset de COCO, el primero se usó para la detección de manos y el segundo identifica y cuenta los objetos. Después de las pruebas realizadas al algoritmo se determina que el reconocimiento de manos de MediaPipe tuvo una precisión del 96% y la detección y clasificación de objetos usando YOLO fue de 43.7%. Teniendo como retos el algoritmo la superposición, la oclusión/auto oclusión de los objetos, o la pérdida de foco de los elementos debido al sensor.

Abstract

Computer vision (CV) can be a process that facilitates some tasks in inventory management, through this process a permanent analysis of an inventory can be performed and thus keep record of all movements made, delivering an instant report when required. This means an improvement in security, since by keeping a strict control of the existing elements in the inventory it is possible to know if an element belongs or not to an inventory or when an element is removed or added, after this need for inventory control, the need arises to design an intelligent system that can facilitate inventory control. Through the combination of 2 frameworks, the creation of an algorithm capable of performing the identification and counting of objects, as well as the identification of the hand to determine when a human manipulation is performed to the inventory. To achieve this objective, two algorithms were used: MediaPipe and YOLOv5 combined with the COCO dataset, the first one was used for hand detection and the second one identifies and counts the objects. After testing the algorithm, it was determined that the hand recognition of MediaPipe had an accuracy of 96% and the detection and classification of objects using YOLO was 43.7%. Challenges for the algorithm were overlapping, occlusion/self-occlusion of objects, or loss of focus of items due to the sensor.

Palabras clave: conteo de objetos, Detección de manos, Detección de objetos, MediaPipe, YOLO.

Keywords: hand detection, Object counting, Object detection, Mediapipe, YOLO

¿Cómo citar?

Bernal-Baquero, F., Martínez, D.E. Human action detection for inventory control using computer vision, Ingeniería y Competitividad, 2024, 26(1): e-21813230.

https://doi.org/10.25100/iyc.v26i1.13230

Recibido: 14-09-23
Aceptado: 26-02-24

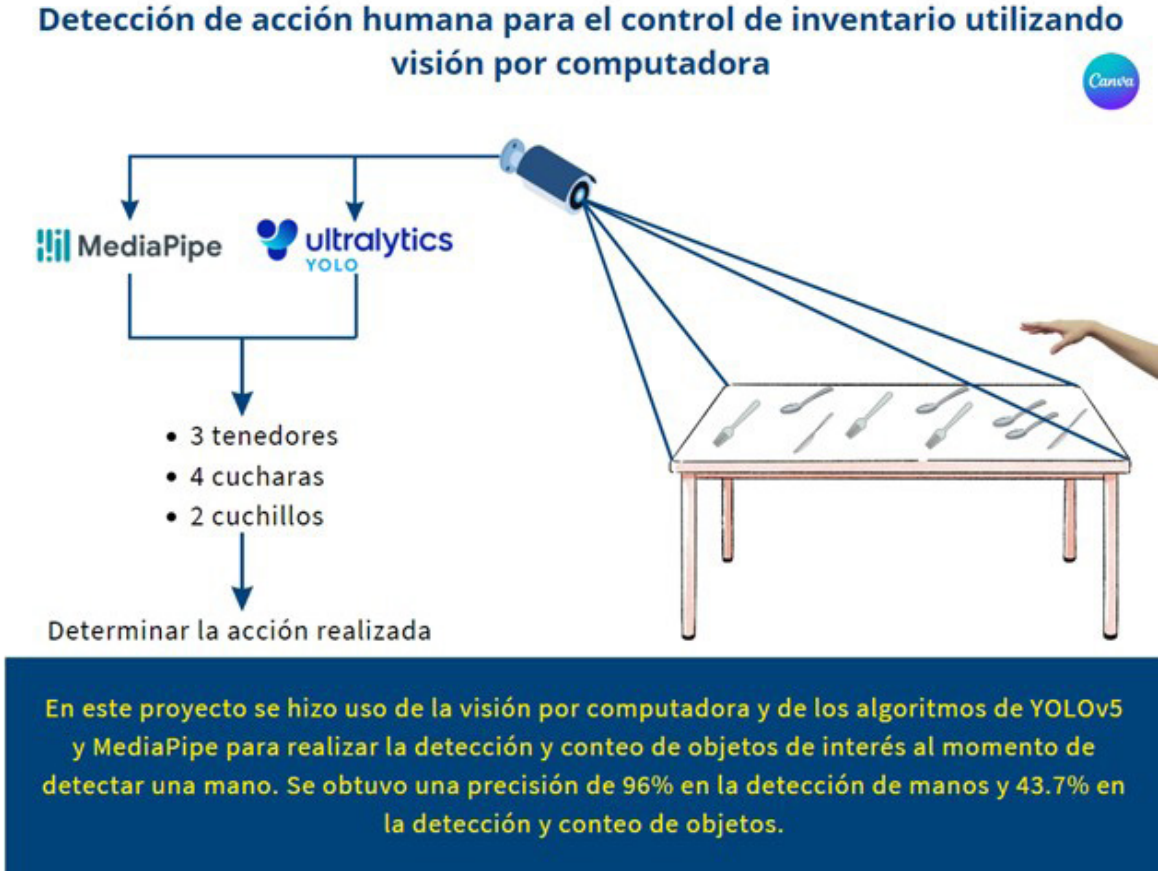
Correspondencia:
francisco.bernal02@usa.edu.co

¿Por qué se llevó a cabo?
Intentamos responder una pregunta: ¿Cómo mediante la visión por computadora podemos contar los elementos de un inventario cuando por una interacción humana este se modifica? La respuesta a esta pregunta nos brindó una alternativa tecnológica para el control de inventarios, aumentando la seguridad y manteniendo un control permanente sobre los elementos que lo componen, ayudando así a llevar un seguimiento de los artículos que entran o salen del inventario.

¿Cuáles fueron los resultados más relevantes?
La combinación de MediaPipe para el reconocimiento de manos y el uso de un modelo entrenado YOLOv5 con un conjunto de datos personalizado para el reconocimiento de objetos, dio como resultado una muy buena herramienta que cumplió con los requisitos del algoritmo, logrando un reconocimiento de manos con una precisión del 96% y 43,7%. para objetos, cabe destacar que el modelo YOLOv5 fue entrenado con muy pocas imágenes (alrededor de 320) por lo que se considera que el modelo de reconocimiento de objetos funcionó mejor de lo esperado.

¿Qué aportan estos resultados?
El algoritmo se presenta como una ayuda en el control de inventario porque con él se puede tener un mayor control de la gestión del inventario y saber rápidamente cuántos artículos tienes. El algoritmo presentado, además de contar los objetos, es capaz de detectar las manos que modifican el inventario y también detectar si se han añadido o quitado objetos del inventario, adicionando un extra al control de objetos del inventario.

Graphical Abstract



doi: 10.25100/iyc.v26i1.13230

Introducción

La visión por computadora (VC) es un campo de la informática que se centra en el desarrollo de algoritmos y técnicas, la cual permite a las computadoras “ver” y comprender el mundo que nos rodea. La VC incluye el procesamiento de imágenes, aprendizaje automático y el reconocimiento de patrones para extraer información de interés en los datos que se encuentran en las imágenes; de igual manera puede ser utilizada para realizar tareas como: la detección de objetos, la clasificación de imágenes, la detección de bordes, el seguimiento de objetos, entre otras (1).

La visión por computadora es cada vez más utilizada en el mundo de los inventarios (2) y la seguridad (3). Según Arias, la visión por computadora resulta muy útil para realizar control de inventarios, así como para detectar objetos no deseados, permitiendo mejorar la seguridad ya que reconoce objetos potencialmente peligrosos como armas o bombas y alertar a los usuarios (3). Por otro lado, Kalahiki propone el uso de la VC para lograr un control de inventario más fiable por medio de varias técnicas existentes como lo son: la recopilación, el preprocesamiento de los datos y la transferencia de aprendizaje (2).

En este trabajo se busca dar solución a la siguiente pregunta: ¿Cómo por medio de la VC podemos realizar un conteo de los elementos de un inventario cuando por medio de una interacción humana este es modificado? La respuesta a esta pregunta da una alternativa tecnológica al control de inventario, logrando aumentar la seguridad y llevando un control permanente sobre los elementos, ayudando de esta manera a que las empresas puedan ser más efectivas en llevar un registro de los ítems que entran o salen de sus inventarios o identificando objetos que no deberían estar en su inventario.

Metodología

MediaPipe

MediaPipe es un marco de trabajo para construir pipelines aplicados de aprendizaje automático multimodales (como de video, audio, datos de series de tiempo, etc.), y para diferentes plataformas (por ejemplo, Android, iOS, Web, y dispositivos perimetrales). MediaPipe provee múltiples características, incluyendo detección de rostros, seguimiento de manos, detección de gestos, y detección de objetos (4).

Seguimiento de manos

El seguimiento de manos de MediaPipe utiliza un proceso de aprendizaje automático (ML) con dos modelos que trabajan en conjunto:

- Un detector de palma que opera en una imagen completa de entrada y localiza las palmas mediante un cuadro delimitador de mano orientado.
- Un modelo de puntos de referencia de mano que opera en el cuadro delimitador de mano recortado proporcionado por el detector de palma y devuelve puntos de referencia 2.5D de alta fidelidad.

Al identificar la imagen de palma recortada con el modelo de puntos de referencia de mano, ver figura 3, se reduce drásticamente la necesidad de aumentar los datos (por ejemplo, mediante rotaciones, traslaciones y escalas) y permite que la red se centre más en la precisión de la localización de los puntos de referencia. En un escenario de seguimiento en tiempo real, se obtiene un cuadro delimitador a partir de la predicción de puntos de referencia del fotograma anterior como entrada para el fotograma actual, evitando así

doi: 10.25100/iyc.v26i1.13230

aplicar el detector en cada fotograma. En cambio, el detector solo se aplica en el primer fotograma o cuando la predicción de la mano indica que se ha perdido la mano (5).

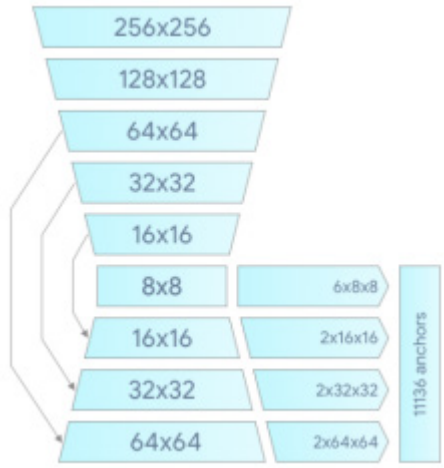


Figura 1 Arquitectura del modelo de detector de palma. (5)

Modelo de detección de palma

De acuerdo con Zhang (5) detectar manos es una tarea compleja, ya que para ello es necesario enfrentar varios retos, el modelo debe de ser capaz de reconocer las manos en una variedad de tamaños y ser capaz de detectarlas aun cuando se encuentran ocluidas o auto ocluidas. A diferencia de las caras que tienen patrones de contraste, por ejemplo, alrededor de los ojos y la boca, la falta de esas características en las manos hace que sea comparativamente más difícil detectarlas de manera confiable solo a partir de sus características visuales. Para solucionar estos retos, Zhang (5) propone abordarlos con diferentes estrategias, la primera es realizar el entrenamiento de palmas, en lugar de un detector de manos, ya que, según ellos estimar cuadros delimitadores de objetos rígidos como palmas y puños es mucho más sencillo que detectar manos con dedos articulados. Al ser las palmas más pequeñas, el algoritmo de “supresión no máxima” funciona incluso en casos complejos como un apretón de manos.

Las palmas pueden modelarse utilizando solo cuadros delimitadores, ignorando otras relaciones de aspecto, lo que reduce el número de anclajes. Luego de este proceso, utilizan un extractor de características codificador-decodificador como FPN para tener una mayor conciencia del contexto de la escena, incluso para objetos pequeños y, por último, minimizan la perdida focal para manejar una gran cantidad de anclajes resultantes de la alta variabilidad de escala. La arquitectura de detector de palma de alto nivel se muestra en la Figura 2.

doi: 10.25100/iyc.v26i1.13230

Modelo de referencia de la mano

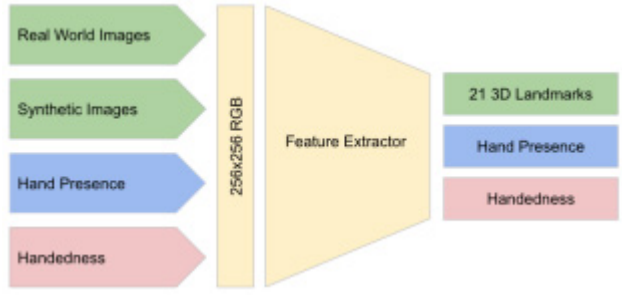


Figura 2 Arquitectura del modelo de detección de manos de MediaPipe

El modelo tiene tres salidas que comparten un extractor de características. Cada cabeza es entrenada por conjuntos de datos correspondientes marcados en el mismo color (5) Después de ejecutar la detección de palmas en toda la imagen, el modelo subsiguiente de puntos de referencia de mano realiza una localización precisa de 21 coordenadas 2.5D dentro de las regiones de mano detectadas a través de la regresión. El modelo aprende una representación de postura interna de la mano consistente y es robusto, incluso, ante manos parcialmente visibles y auto ocultaciones. El modelo tiene tres salidas, ver Figura 2:

- 21 puntos de referencia de la mano que consisten en coordenadas “x, y” y profundidad relativa, ver Figura 3.
- Una indicación de la probabilidad de presencia de mano en la imagen de entrada.
- Una clasificación binaria de lateralidad de la mano, por ejemplo, si es izquierda o derecha.

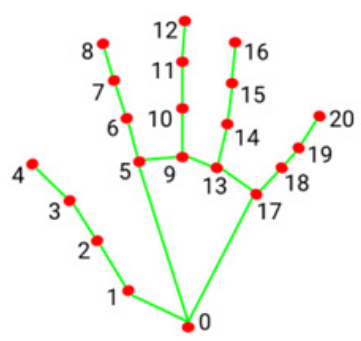


Figura 3 Puntos de manos que toma MediaPipe para el reconocimiento y seguimiento de manos. Imagen tomada de la documentación de Hand Landmarks detection guide de MediaPipe.

You Only Look Once (YOLO)

YOLO se presentó al público en el año 2015 (6), debido a su velocidad, exactitud y capacidad para correr en tiempo real, su uso para detección de objetos se masificó. Se basa en la arquitectura de Googlenet (7), el funcionamiento de YOLO consta en la división de la imagen en una rejilla, donde en cada celda se verifica si existe un objeto de interés en su interior. Con este proceso, se generan varios cuadros delimitadores de diferente tamaño, se debe definir con qué nivel de confianza se hace una predicción de la existencia de un objeto. Un nivel de confianza alto supone que el algoritmo será más estricto en detectar y clasificar un objeto y un nivel de confianza bajo hará que el algoritmo sea más permisivo en la detección y clasificación.

doi: 10.25100/iyc.v26i1.13230

Versiones

YOLOv4 y YOLOv5 introducen cambios que generan una mejoría en cuanto a la velocidad y el rendimiento probado en diferentes benchmark. YOLOv5 entrega resultados similares a YOLOv4, pero, su tiempo de entrenamiento es aún menor. Además, el desarrollo de esta arquitectura se implementa nativamente en Pytorch, a diferencia de las versiones anteriores basadas en Densenet.

Adicionalmente, YOLOv5 consta de diferentes modelos, los cuales tienen distintas configuraciones ver Figura 4, de manera que se debe escoger el modelo adecuado de acuerdo con las necesidades específicas de cada proyecto, así se debe determinar si se requiere mayor precisión o velocidad. Los modelos como YOLOv5x y YOLOv5x6 producirán mejores resultados en cuanto a precisión y detección, pero tiene más parámetros, requiere más memoria CUDA para entrenar y son más lentas para ejecutarse. Ultralytics recomienda usar los modelos más pequeños para desarrollos en móviles, y los modelos grandes para desarrollos con arquitecturas basados en la nube.

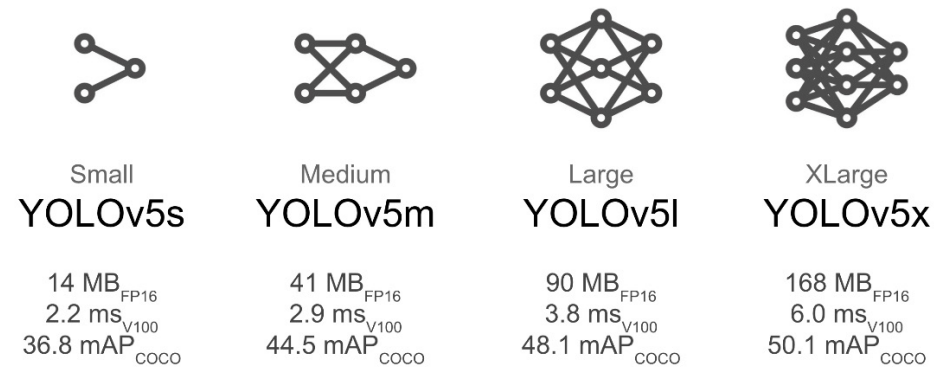


Figura 4 Velocidad de análisis, mAP y peso de los diferentes modelos de YOLOv5. Imagen tomada del repositorio de ultralitycs en github.

A continuación, se presenta el diagrama con la metodología general, ver Figura 5.

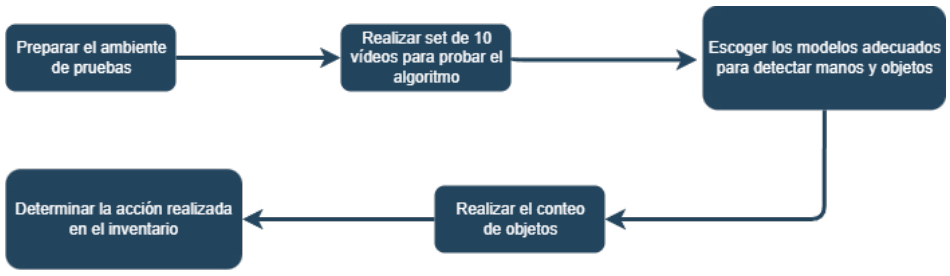


Figura 5 Metodología realizada para detectar las manos e identificar los objetos en una escena que permita el control de objetos en el inventario

Preparar el ambiente de pruebas

La preparación del ambiente de pruebas consistió en adecuar la superficie para realizar el experimento. Se utilizó un mantel de color rojo y azul sobre una mesa para que contrastara con los utensilios de cocina. Adicionalmente, se realizó un control lumínico con dos lámparas halógenas. Con la finalidad de reducir las sombras que ocasionaran un conteo y detección errada por parte del algoritmo.

doi: 10.25100/iyc.v26i1.13230

Realizar set de 10 vídeos para probar el algoritmo

Se construyó un conjunto de vídeos con el fin de realizar varias pruebas y con ellas identificar los diferentes estados de los elementos de interés. Cada vídeo presenta un reto diferente para los modelos, como lo son: obstrucción o superposición de objetos, objetos alejados del campo visual del sensor, agregar o retirar dos o más objetos a la vez o reconocer cuando entran y salen 1 ó 2 manos.

Escoger los modelos adecuados para detectar manos y objetos

Se realizaron pruebas con diferentes modelos para la detección de manos y objetos, estas pruebas utilizaron el conjunto de vídeos para determinar que algoritmos funcionaban mejor en la detección de los objetos de interés. La medida que se utilizó para seleccionar el modelo más idóneo fue la precisión.

Realizar el conteo de objetos

Se realizó el conteo de objetos por medio de la definición de una lista de objetos de interés, esta lista sirve como elemento de control que permite al algoritmo filtrar los objetos de interés de los demás elementos, una vez realizada la individualización de los objetos se realiza un conteo, de tal manera el algoritmo tuvo la capacidad de realizar el conteo de los objetos que son de interés del algoritmo.

Determinar la acción realizada en el escenario para el control del inventario

El algoritmo es capaz de determinar dos tipos de acciones: la primera, consiste en añadir elementos al inventario, y la segunda acción, retira los elementos del campo visual del sensor.

Para determinar estas acciones se realiza el conteo de los objetos en dos puntos, al momento de detectar la mano y al momento de ya no detectar la mano, cuando no se logra detectar la mano, el algoritmo realiza una comparación de los conteos previos y posteriores a la intervención humana, si el conteo de objetos posterior es mayor se determina la acción de añadir objetos al inventario, caso contrario se determina que se tomaron objetos. La respuesta esperada debería ser la siguiente, ver Figura 6:

Figura 6 A la izquierda, organización de los elementos del inventario y su respectivo conteo al ya no detectar la mano por medio del sensor; a la derecha, el mensaje del algoritmo indicando la clasificación y conteo de los objetos y la acción realizada en el inventario. En este caso: tomar objetos.

doi: 10.25100/iyc.v26i1.13230

Resultados y discusión

En el desarrollo del presente trabajo se usaron diferentes modelos y algoritmos, para determinar cuál era el modelo ideal en términos de detección, robustez y fiabilidad, obteniendo los siguientes resultados:

Métricas detección de manos

Para la detección y seguimiento de manos se utilizó el dataset de EgoHands, un dataset especializado en manos, etiquetado y realizado por Sven B, Stefan L, David C y Chen Y de la universidad de Indiana (9), se decidió poner a prueba el algoritmo de reconocimiento de manos. Se realizaron modificaciones en el dataset para cambiar las características y categorías de las imágenes ya que de estas se puede descartar si es una mano izquierda o una mano derecha u otras características que este modelo no necesita para su funcionamiento. Estos datos nos dan una información respecto al rendimiento del modelo en términos de su capacidad para identificar correctamente las muestras positivas y negativas.

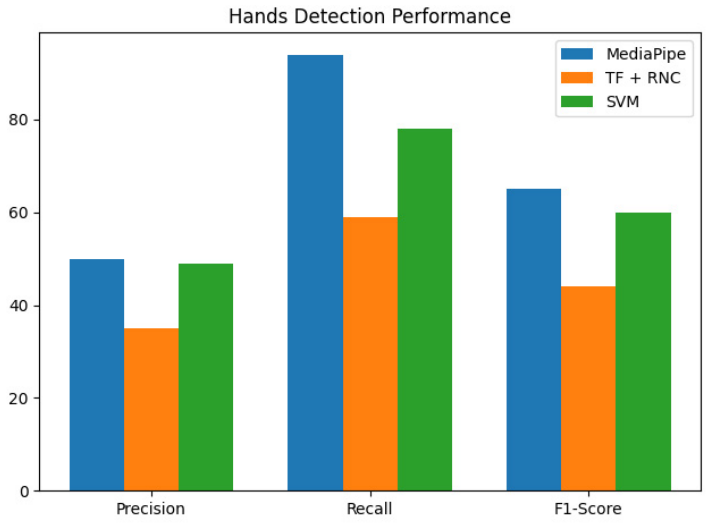


Figura 7. Gráfica de evaluación de rendimiento de detecciones de manos, utilizando diferentes algoritmos: MediaPiPe, Tensorflow combinado con Redes Neuronales Convolucionales y maquinas de vectores de soporte.

En general, una precisión y un recall más altos indican un mejor rendimiento del modelo, mientras que un F1-score más alto indica un equilibrio entre la precisión y el recall. Por lo tanto, es importante tener en cuenta tanto la precisión como el recall para obtener una evaluación completa del rendimiento del modelo.

Métricas detección de objetos

Al no encontrar un dataset específico para poder probar el rendimiento en la detección de los objetos de interés, se decidió generar uno que se acoplara a las necesidades de nuestro caso de uso.

doi: 10.25100/iyc.v26i1.13230

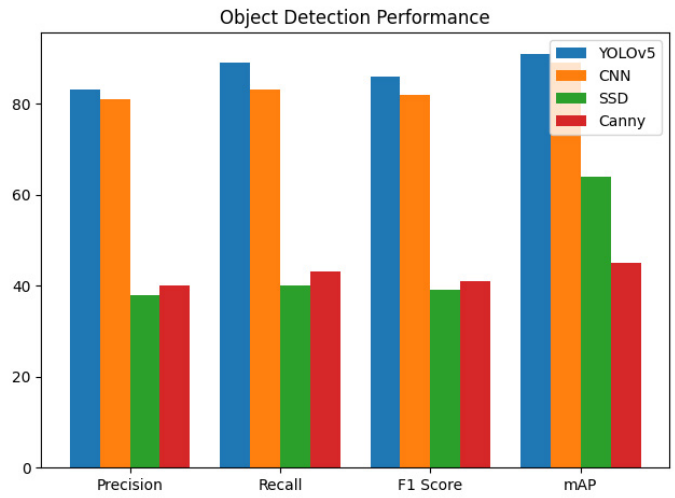


Figura 8. Gráfica de evaluación de rendimiento de detección de objetos, utilizando diferentes algoritmos: YOLOv5, CNN, SSD y canny.

Análisis de resultados

Para la detección y seguimiento de manos se utilizaron 3 modelos, los cuales fueron: MediaPipe, TensorFlow y SVM. En la tabla 1, se presenta el rendimiento de los modelos, con el dataset de EgoHand. Dichas métricas revelan que el mejor modelo para la detección de manos es MediaPipe, a pesar de que la precisión fue muy baja en todos los modelos, y para este caso el que presento el rendimiento más bajo fue TensorFlow combinado con Redes Neuronales Convolucionales.

Modelo	Precisión	Recall	F1 Score
MediaPipe	50%	94%	65%
TF + RNC	35%	59%	44%
SVM	49%	78%	60%

Tabla 1. Comparación de rendimiento entre MediaPipe, TensorFlow, SVM para la detección de manos con el dataset de EgoHands.

Con base en estos resultados se ha determinado que MediaPipe presenta el mejor rendimiento en la detección y seguimiento de manos.

- Precisión: la precisión de 0.509 significa que aproximadamente la mitad de las detecciones positivas del modelo son correctas, mientras que la otra mitad son falsas alarmas.
- Recall: el recall de 0.945 significa que el modelo es muy bueno identificando las manos en las imágenes, ya que detecta el 94.5% de las manos presentes en las imágenes.
- F1-Score: el F1-score de 0.65 muestra un equilibrio entre la precisión y el recall. Puede indicar que el modelo está realizando un buen trabajo al mantener un recall alto mientras mantiene un nivel aceptable de precisión.

doi: 10.25100/iyc.v26i1.13230

Por otro lado, para la detección de conteo de objetos se realizaron pruebas con 4 modelos: YOLOv5, redes neuronales convolucionales y SSD MobileNet V2. Presentando los siguientes resultados:

Tabla 2. Comparación de rendimiento entre YOLOv5, CNN, SSD y Canny para la detección y conteo de objetos entrenado con un dataset propio

Modelo	Precisión	Recall	F1 Score	mAP
YOLOv5	83%	89%	86%	91%
CNN	81%	83%	82%	89%
SSD	38%	40%	39%	64%
Canny	40%	43%	41%	45%

Para la detección de objetos se decidió optar por YOLOv5 a pesar de que con CNN las métricas eran muy parecidas y presentaba un F1 Score más alto, en términos generales YOLOv5 presentaba métricas más altas y el F1 Score de este modelo indicaba un rendimiento muy bueno.

Comparativa de modelos

La detección de manos vs. la detección de objetos supone un cambio muy alto uno del otro en cuanto a precisión y a F1 Score (Aproximadamente un 30% de diferencia), esto se debe a que EgoHands al ser un dataset más especializado, con mayor trayectoria, presentaba un reto para los modelos al buscar manos en lugares poco comunes o de difícil locación o tamaño de mano, este dataset consta de 3841 imágenes de entrenamiento, de diferentes lugares y situaciones, por otro lado, el entrenamiento para la detección de objetos se presenta como un dataset construido de acuerdo a las necesidades, presentando como una base de imágenes de 322, en situaciones más fáciles que no representaban un reto para el modelo.

Resultados de experimentación

En el contexto de este proyecto se buscó implementar un algoritmo que lograra reconocer objetos de interés en una secuencia de vídeo en vivo, y que contara los objetos que detectó. Para poder probar el algoritmo se creó un conjunto de 10 vídeos y con los vídeos pregrabados se realizó la toma de métricas, obteniendo los siguientes resultados, ver tabla 3.

Nota: en la tabla 3 se toma como 0 un Falso Positivo (FP) o Falso Negativo (FN), Un FP se determina como una cantidad de conteo exitoso pero etiquetados como diferentes clases, y un FN se determina como un caso en el que ni el conteo ni la clase coinciden; y los casos marcados como 1 se cuentan como Verdaderos positivos (TP), un TP se cuenta como un conteo y una clasificación de elementos exitosa; por último, un guion (-) indica que no se realizaron más tomas de prueba para probar el modelo en ese vídeo. Los “test” se definen como cada detección de entrada y salida de mano, en la cual el modelo realizaba el conteo de los objetos en el inventario, todos los vídeos tuvieron diferentes “test” siendo el que más tiene el vídeo 8 con 11 interacciones de inventario y el vídeo 10 el menor con una interacción.

doi: 10.25100/iyc.v26i1.13230

Tabla 3. Resultados de los 10 vídeos de prueba de experimentación

Video	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	Test 11
Video 1	1	1	1	1	-	-	-	-	-	-	-
Video 2	0	0	1	0	-	-	-	-	-	-	-
Video 3	0	1	1	0	1	-	-	-	-	-	-
Video 4	1	0	0	1	0	0	-	-	-	-	-
Video 5	1	1	1	0	1	-	-	-	-	-	-
Video 6	0	0	0	1	0	0	0	1	-	-	-
Video 7	1	0	0	0	1	0	-	-	-	-	-
Video 8	1	1	1	1	1	1	1	1	1	0	0
Video 9	0	0	0	-	-	-	-	-	-	-	-
Video 10	0	-	-	-	-	-	-	-	-	-	-

Estados del algoritmo

De acuerdo a las pruebas realizadas se determinó que el algoritmo tiene tres posibles estados como respuesta, estos estados son caso exitoso en el que la clasificación, conteo y acción realizada son correctas; caso no exitoso con mala clasificación, en el cual el conteo y la acción realizada son correctos pero la clasificación es errónea; y por último, caso no exitoso con mala clasificación y conteo, en el cual la clasificación, el conteo y la acción realizada no es correcta.

Caso exitoso

Un caso exitoso se define cuando un conteo y una clasificación de objetos se realiza correctamente, como se muestra en la Figura 10.a tomado del vídeo 1, notamos que el algoritmo luego de no detectar la mano realiza una clasificación, conteo y acción correcta ya que en el estado anterior del inventario había 2 cucharas y un tenedor, luego de la intervención humana se detectaron de acuerdo con la salida del programa 3 tenedores y dos cucharas, señalando la acción como agregar elementos.

Caso no exitoso, mala clasificación y conteo

El segundo estado erróneo se puede observar en la Figura 10.b tomado del vídeo 9, en el cual ni el conteo, ni la clasificación coinciden con lo observado en la imagen. En este vídeo ninguna clasificación fue exitosa ya que se decidió probar el algoritmo en condiciones usuales como lo son superposición de objetos o ubicaciones no muy comunes de los objetos, esta configuración de inventario impactó negativamente la precisión del algoritmo en la detección y clasificación de los objetos.

doi: 10.25100/iyc.v26i1.13230

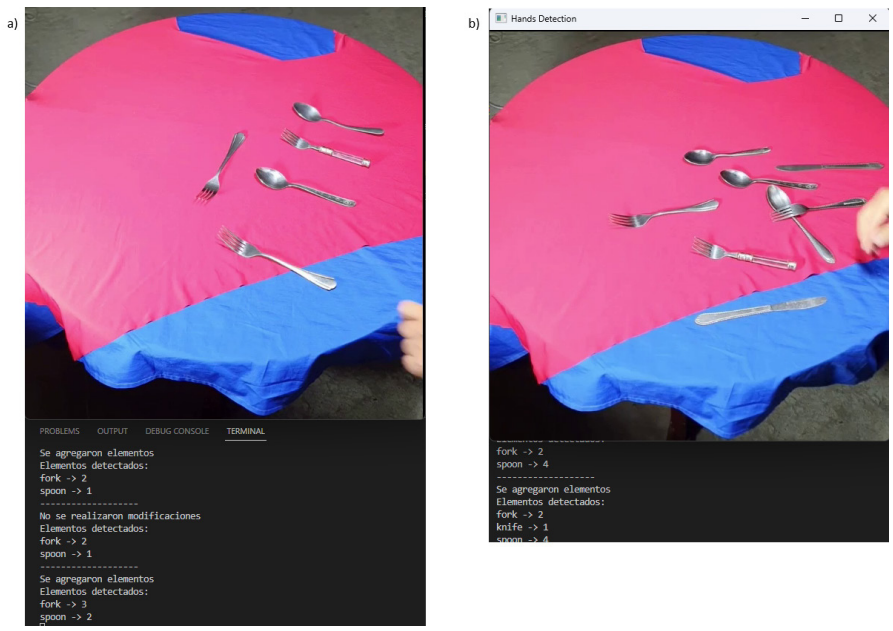


Figura 9. Posibles resultados determinados por el algoritmo, a) resultado correcto en el que tanto la clasificación como el conteo de objetos es correcto; b) resultado incorrecto en el que ni el conteo ni la clasificación de objetos es correcto.

Discusión

El algoritmo se presenta como una ayuda en el control de inventarios y la seguridad ya que con él se puede tener un mayor control del manejo del inventario y conocer cuándo un elemento fue agregado o retirado al momento de tener una interacción humana. El algoritmo presentado aparte de realizar el conteo de los objetos es capaz de detectar las manos que modifican el inventario, y así mismo, detectar si agrego o quito objetos del inventario, adicionando un extra al control de objetos en el inventario. Teniendo un dataset construido el cual se utilizó para el entrenamiento y pruebas del modelo de YOLOv5, y un dataset especializado en manos para pruebas del modelo de Mediapipe, el algoritmo en un ambiente real obtuvo una precisión de 43.7% en el reconocimiento de objetos y un 96% en el reconocimiento de manos, dichos resultados muestran que efectivamente se puede realizar un control de los objetos y detectar cuándo un actor humano interfiere en el inventario, y asi lograr definir la acción realizada.

Con el fin de mejorar la precisión del algoritmo en el reconocimiento de objetos por medio del dataset, Kalahiki propone un enfoque de entrenamiento constante y asi, aumentar la precisión del algoritmo en reconocimiento de objetos preparándolo para que reconozca objetos en distintas posiciones, ambientaciones de luz o lugares en el que el dataset original no se hubiese contemplado. Realizar estos cambios supone un trabajo de más en el entrenamiento, pero a su vez, implica un modelo cada vez más eficiente en cuanto a reconocimiento de acuerdo con los resultados expuestos por Kalahiki (ver Tabla 4). Donde se puede observar que la precisión aumenta cada semana a medida que se entrena con nuevas imágenes tomadas de los mismos vídeos y que tienen dificultad para ser reconocidos. En el trabajo de Kalahiki, empieza con una precisión de 61.92% para la función de activación lineal y en la semana 3 aumenta un 3.68% dejando la precisión en un 65.6%, este modelo resulta dar una buena alternativa ya que a medida que se entrena con más imágenes su algoritmo se volverá más robusto.

doi: 10.25100/iyc.v26i1.13230

Tabla 4. Precisión promedio en el conjunto de datos de generalización por función de activación. Datos tomados de (2)

Función de activación	Promedio inicial	Promedio en la semana 1	Promedio en la semana 2	Promedio en la semana 3
Lineal	61.92%	62.4%	63.83%	65.6%
ReLU	62.49%	62.89%	64.24%	66.39%
Leaky ReLU	63.16%	62.35%	63.58%	66.59%
SELU	62.62%	62.7%	63.75%	65.77%

Por otro lado, este trabajo plantea centrarse en la clasificación y conteo de objetos siendo activado cuando una mano pase por el campo de acción del sensor, y determine la acción realizada, este algoritmo tuvo una precisión de 43.7% en reconocimiento de objetos y un 96% en reconocimiento de manos, con lo cual se podría realizar el mismo tratamiento propuesto por Kalahiki y hacer el algoritmo más robusto y preparado a diferentes estados no contemplados en el dataset original.

Arias propone el uso de la VC por medio de las máquinas de vectores de soporte (SVM) para un reconocimiento más ágil de objetos potencialmente peligrosos en un contexto de aeropuertos, identificando armas cortas, Arias obtuvo aumento en el rendimiento en un 93.36% respecto a una implementación anterior usando la programación paralela al igual que una clasificación satisfactoria de 93.02% en imágenes que contenían armas de fuego. El algoritmo propuesto en este trabajo ejecuta los algoritmos de detección de manos y de objetos en paralelo, pero solo hace uso de YOLOv5 al momento de ser detectada la mano, evitando asi realizar un uso constante de la detección de objetos y reduciendo el costo computacional de analizar los frames de cada video, esto supone una mejora en el rendimiento del algoritmo, presentando los siguientes resultados (ver figura 11).

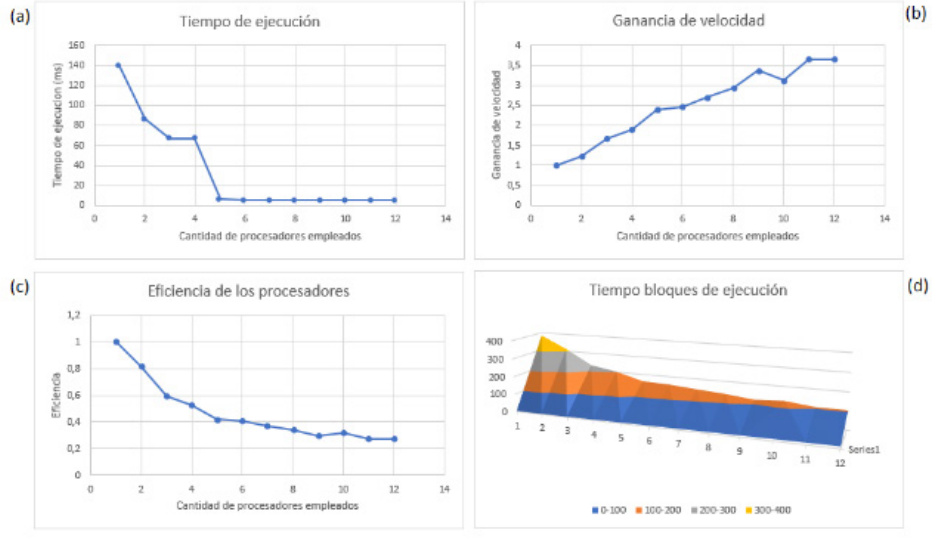


Figura 11. Gráfica rendimiento computacional del modelo detección con MediaPipe y YOLOv5; a) tiempo de ejecución en ms del código, con diferentes procesadores en uso; b) ganancia de velocidad con base a los procesadores empleados; c) eficiencia del algoritmo con diferentes procesadores en uso; d) tiempo en ms de la ejecución de los bloques de código

doi: 10.25100/iyc.v26i1.13230

Con un conjunto de datos y un entrenamiento focalizado este algoritmo podría servir como modelo para el reconocimiento y conteo de objetos diferentes a los propuestos en este artículo. Presentándose como una buena oportunidad para el control de inventario en tiendas, en bodegas de almacenamiento o de otros lugares en los que se requiera un control de inventario.

La falta de imágenes para realizar el entrenamiento y la adecuada integración con la herramienta YOLOv5, es una de las causas principales por la cual el conteo y reconocimiento de cubiertos fue baja, con un adecuado conjunto de datos las métricas de precisión mejorarían, haciendo que el algoritmo sea más útil, siendo capaz de superar los retos que en su versión actual, no es capaz de solucionar, como la auto oclusión, la superposición de objetos, o los objetos parcialmente detectados por el campo de acción del sensor, por esa razón se espera contar con un dataset mejor preparado y así mejorar el reconocimiento de objetos.

Conclusiones

Para el reconocimiento de las manos es necesario configurar adecuadamente MediaPipe, esto quiere decir que se deben nivelar los umbrales, uno muy alto significará una detección más minuciosa por parte del algoritmo, lo que conlleva a que el reconocimiento de manos se dé bajo ciertas condiciones muy específicas, el caso contrario supone que el algoritmo será más permisivo en reconocer las manos y cualquier objeto en el campo visual del sensor podría ser reconocido como tal.

Al igual que con MediaPipe, YOLOv5 debe ser configurado correctamente en cuanto a la clasificación de umbrales, y confidencia de detección, esto permite tener un mayor control al momento de descartar o reconocer algún objeto dentro del inventario y lograr reconocerlo correctamente.

El conteo de los objetos recae fundamentalmente en el modelo de clasificación, una mala clasificación supone un mal conteo de objetos, que a su vez, indica una acción errónea por parte del usuario, se debe realizar un entrenamiento más exhaustivo en pro de que el algoritmo pueda reconocer y clasificar los objetos y esto significará automáticamente un mejor conteo de los elementos del inventario.

Al momento de obtener las métricas de MediaPipe, el conjunto de imágenes de prueba demostró un bajo puntaje en cuanto a precisión, recall y F1, pero, al momento de realizar la experimentación, las métricas demostraron un alto porcentaje en todos los aspectos, caso contrario ocurrió con las métricas de MediaPipe, esto se debe a la calidad de los datos de prueba versus los datos de experimentación.

La experimentación también demostró un punto débil en cuanto a reconocimiento en superficies con un color parecido al de los objetos de interés, en la experimentación se comprobó que con una superficie blanca YOLO no reconocía los objetos de interés, por eso, se tomó la decisión de tener una superficie que contrastara los objetos de interés.

Referencias

1. Athanasios V, Nikolas D, Anastasios D, Eftychios P. Deep Learning for Computer Vision: A Brief Review. Computational Intelligence and Neuroscience [Internet]. 2018 feb [citado 7 ago 2023]. Disponible en: https://doi.org/10.1155/2018/7068349.

doi: 10.25100/iyc.v26i1.13230

2. Christopher Bradley K. Computer Vision for Inventory Management [tesis maestría en Internet]. Lousiana Tech University; 2020 [citado 7 ago 2023]. Disponible en: https://digitalcommons.latech.edu/theses/40/.
3. Felipe R, Leaned Q, Frank S, David C, Enrique M. Software component for weapons recognition in X-ray images [Internet]. 2017 abr [Citado 7 ago 2023]. Disponible en: https://www.researchgate.net/publication/316628078_Software_component_for_weapons_recognition_in_X-ray_images.
4. Equipo de MediaPipe. MediaPipe Framework [Internet]. [Lugar desconocido] [Citado 8 ago 2023]. Disponible en: https://developers.google.com/mediapipe/framework.
5. Fan Z, Valentin B, Andrey V, Andrei T, George S, Chuo-Ling C, Matthias G. MediaPipe Hands: On-device Real-time Hand Tracking. 2020 jun [citado 8 ago 2023]. Disponible en: https://arxiv.org/abs/2006.10214.
6. Joseph R, Santosh D, Ross G, Ali F. You Only Look Once: Unified, Real-Time Object Detection. 2015 jun [citado 8 ago]. Disponible en: https://arxiv.org/abs/1506.02640.
7. Christian S, Wei L, Yangqing J, Pierre S, Scott R, Dragomir A, Dumitru E, Vincent V, Andrew R. Going Deeper with Convolutions. 2014 sep [citado 8 ago 2023]. Disponible en: https://arxiv.org/abs/1409.4842.
8. Glenn J. why do I need to train from the pt model you have trained? · Issue #2990 · ultralytics/yolov5 · GitHub [Internet]. [Lugar desconocido] [Citado 8 ago 2023]. Disponible en: https://github.com/ultralytics/yolov5/issues/2990.
9. Bambach, S., Lee, S., Crandall, D., and Yu, C. EgoHands Object Detection Dataset [Internet]. [Lugar Desconocido] [Citado 8 ago 2023]. Disponible en: https://public.roboflow.com/object-detection/hands.
10. Narendra A., Sinisa T. Learning the Taxonomy and Models of Categories Present in Arbitrary Images. 2007 dec [Citado 8 ago 2023]. Disponible en: https://ieeexplore.ieee.org/document/4409039.
11. Ming Jin C., Zaid O., Mohamed H. A review of hand gesture and sign language recognition technique. 2017 ago [Citado 8 ago 2023]. Disponible en: https://link.springer.com/article/10.1007/s13042-017-0705-5.
12. Loïc C., Benoît M., Philippe T. Object Detection with Spiking Neural Networks on Automotive Event Data. 2022 may [Citado 8 ago 2023]. Disponible en: https://arxiv.org/abs/2205.04339.
13. Pedro F., Ross B., David M., Deva R. Object Detection with Discriminatively Trained Part-Based Models. 2010 sep [Citado 8 ago 2023]. Disponible en: https://ieeexplore.ieee.org/document/5255236.
14. Sanja F., Ales L. Towards Scalable Representations of Object Categories: Learning a Hierarchy of Parts. 2007 jul [Citado 8 ago 2023]. Disponible en: https://ieeexplore.ieee.org/document/4270294.
15. Arpita H., Akshit T. Real-time Vernacular Sign Language Recognition using MediaPipe and Machine Learning. 2021 may [Citado 8 ago 2023]. Disponible en: https://www.researchgate.net/publication/369945035_Real-time_Vernacular_Sign_Language_Recognition_using_MediaPipe_and_Machine_Learning.

doi: 10.25100/iyc.v26i1.13230