Identificador de manos MediaPipe Hands

Diego Andres Muñoz 6000470
Juan David Duarte Ramirez 6000412
Julian Camilo Tellez 6000453.

Resumen - Con base en la teoría del procesamiento de Imágenes se desarrollará un programa que logre la identificación de las manos y sus poses, logrando ver los factores comunes obtenidos en imágenes y videos, principalmente para poder facilitar tareas en la computadora, captadas por una cámara, tales como subir y bajar el volumen, pasar o retroceder canciones, o incluso el uso del mouse a partir del movimiento de la mano.

MediaPipe hands, openCV, Python, Imágenes, detección, puntos clave

1. INTRODUCCIÓN.

Para el desarrollo de este programa se tendrá en cuenta la identificación de manos y sus poses obtenidas en imágenes y videos en tiempo real, programación e implementación de la misma para facilitar tareas comunes cuando uno está frente a la computadora, se analizará cuales son estas tareas, como subir y bajar el volumen, pasar o retroceder canciones, abrir aplicaciones específicas, manipulación de textos entre otros. Funcionalidad similar a la ya implementada en los asistentes de búsqueda, que se activan mediante la voz y posteriormente usados para realizar búsquedas o dar órdenes a nuestros dispositivos se aplicará Mediapipe el cual usa inteligencia artificial para lograr la detección de objetos y más específicamente la detección de manos, Buscaremos que las manos tomen el comportamiento de mouse a partir de la detección de estas

2. Método.

Para este proyecto lo principal es el procesamiento de las manos humanas que se necesita para el funcionamiento principal de nuestro objetivo, una vez logrado dicho objetivo podremos empezar con las funciones para hacer que los gestos puedan acceder a las funciones, como las del sonido, de nuestro dispositivo (computador) mediante su cámara, haciendo posible poder modificar dichas funciones.

MediaPipe hands este permite la detección de manos con 21 puntos de referencias 3D, permitiendo identificar cada uno de los dedos de la palma de la mano para ello mediapipe emplea machine learning de donde han obtenido múltiples modelos que trabajan juntos para obtener los resultados que podemos ver en pantalla en la documentación de mediapipe se nos describe el proceso que realizaron para llegar a obtener estos resultados así que vamos a ver voy a un principio tenemos un modelo para la detección de la palma de la mano este es el que se aplica en toda la imagen intentando obtener alguna detección, en cuanto este detector haya encontrado una palma devolverá un cuadro delimitador orientado de la mano una vez obtenida el área en donde se encuentra la mano la imagen pasará por un hand landmark model que es el modelo que permitirá ubicar los 21 de referencia en la imagen dada, cada uno de estos tiene asociado un de los 21 puntos.

Este proceso de por sí es muy interesante de analizar y se puede aplicar a imágenes de entrada pero qué pasa con los vídeos pues bien en este caso han desarrollado un método en el cual se aplique la detección de palmas y el modelo handan marc en los primeros fotogramas una vez obtenidos los 21 puntos de referencia estos son rastreados para calcular la nueva ubicación de la mano de este modo no se está aplicando la detección de palmas a cada momento sino que se realiza seguimiento tracking del área ya he encontrado en primer lugar invocando al detector de palmas únicamente cuando los puntos no se puedan identificar reduciendo de este modo la latencia.

Exploración de las configuraciones del MediaPipe hands.

STATIC IMAGE MODE

En primer lugar tenemos a static image mode, que puede tener valores de True o False. Cuando se le asigna False, entonces trata a las imágenes de entrada como un videostream, de tal manera que aplica el modelo de detección de palma y el modelo hand landmarks en un principio, pero luego realiza tracking para obtener la nueva ubicación de la mano, basándose en los puntos de referencia. De este modo, solo se invocará nuevamente al detector de palmas cuando no se hayan identificado los 21 puntos.

Cuando se le asigna True, entonces los detectores están aplicándose en cada imagen, por lo que es mejor usarla en caso de que se trate de imágenes que no tengan que ver entre sí.

MAX_NUM HANDS

Número máximo de manos por detectar.

MIN DETECTION CONFIDENCE.

Valor mínimo de confianza del modelo de detección de manos, para que la detección sea considerada como exitosa. Sus valores comprenden de 0 a 1.

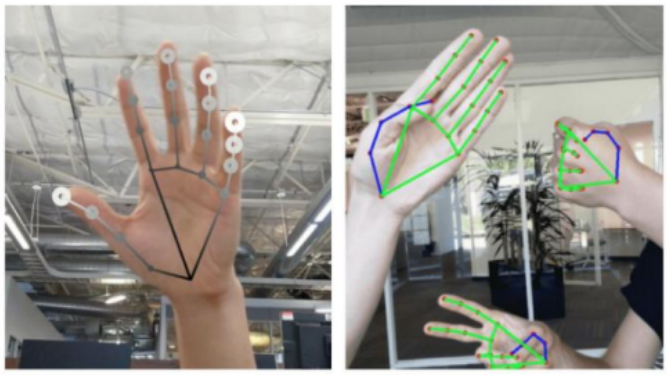
MIN TRACKING CONFIDENCE

Valor mínimo de confianza del modelo de rastreo de los landmark, para que el rastreo de los 21 puntos sea considerado como exitoso. En caso de no serlo, se invocará al detector de manos en la siguiente imagen.

Este es ignorado si static imagen de la imagen está en True.

Detector BlazePalm.

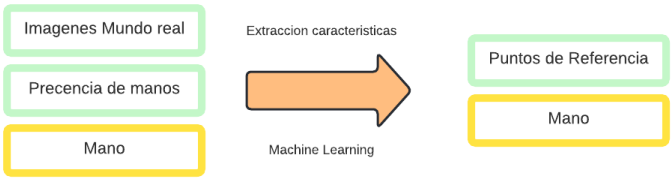
Para detectar las ubicaciones iniciales de las manos, empleamos un modelo detector de disparo único optimizado para aplicaciones móviles en tiempo real similar a Blaze Face[1], que también está disponible en MediaPipe. Detectar manos es una tarea decididamente compleja: nuestro modelo tiene que funcionar en una variedad de tamaños de manos con un intervalo de gran escala



Primero, entrenamos un detector de palmas en lugar de un detector de manos, ya que estimar cajas delimitadoras de objetos rígidos como palmas y puños es significativamente más simple que detectar manos con dedos articulados. Además, como las palmas de las manos son objetos más pequeños, el algoritmo de supresión no máxima funciona bien incluso para los casos de autooclusión de dos manos, como los apretones de manos. Usamos un extractor de características de codificador decodificador, para una mayor conciencia del contexto de la escena, incluso para objetos pequeños. Por último, minimizamos la pérdida focal durante el entrenamiento para admitir una gran cantidad de anclas resultantes de la variación de escala alta. La arquitectura del detector de palma es de alto nivel. Presentamos un estudio de ablación de nuestros elementos de diseño

Modelo de referencia de mano

Después de ejecutar la detección de la palma de la mano en toda la imagen, nuestro modelo de punto de referencia de la mano posterior realiza una localización precisa del punto de referencia de 21 coordenadas 2.5D dentro de las regiones de la mano detectadas a través de la regresión. El modelo aprende una representación de pose de mano interna consistente y es robusto incluso para manos parcialmente visibles. El modelo tiene tres salidas

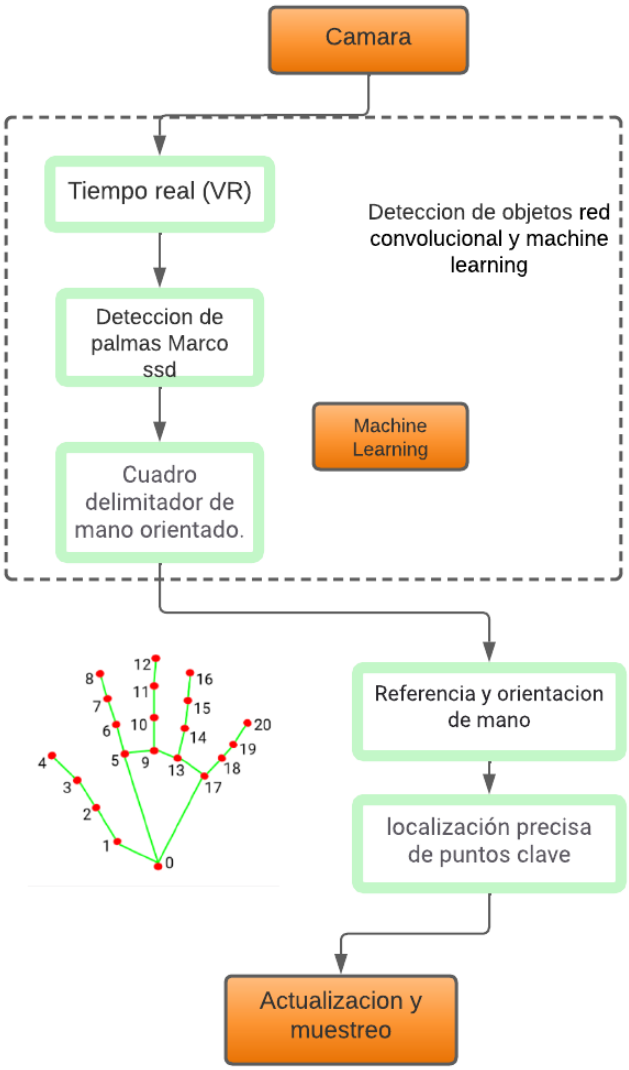


1. 21 puntos de referencia de mano que consisten en x, y y relativo
2. Una bandera de mano que indica la probabilidad de presencia de la mano en la imagen de entrada.
3. . Una clasificación binaria de lateralidad, por ejemplo, izquierda o derecha

Implementación en MediaPipe

Mediapipe viene con un conjunto ampliable de calculadoras para resolver tareas como inferencia de modelos, procesamiento de medios y transformaciones de datos en una amplia variedad de dispositivos y plataformas. Calculadoras individuales como recorta, Los cálculos de renderizado y redes neuronales se optimizan aún más para utilizar la aceleración de GPU. Por ejemplo, empleamos

Controles de salida del modelo de puntos de referencia de la mano cuando se activa el modelo de detección de mano. Este comportamiento se logra mediante la poderosa sincronización de MediaPipe bloques de construcción, lo que resulta en un alto rendimiento y óptimo rendimiento de la canalización de ML



Principalmente se tiene la cámara con la que se captará las imágenes en tiempo real, al mismo tiempo es algoritmo y Machine Learning ya estará llevando a cabo el proceso de detección de objetos en este caso de palmas. Una vez se detecte la palma con ayuda de la inteligencia artificial que ya tiene MediaPipe comenzará la búsqueda de puntos de referencia dando como resultado la localización 2.5 D donde se tendrá en cuenta la profundidad de cada uno de estos 21 puntos clave de la mano. Para finalizar ya se actualizara y se mostrará el muestreo de estos puntos, logrando una imagen donde se resaltan las manos, se podría decir que con esqueleto mallado que representará a la misma.

Frameworks y librerías implementadas.

- OpenCV: lo implementamos para identificar y leer las imágenes 2D y 3D de los frames del video por cámara.
- Pyautogui: Esta librería la implementamos para utilizar la función del mouse en las manos.
- Numpy: Esta librería se implementa para algunas operaciones matemáticas y su correcto funcionamiento en el programa.
- Mediapipe: Es el frameworks en el cual se trabajó todo proyecto.

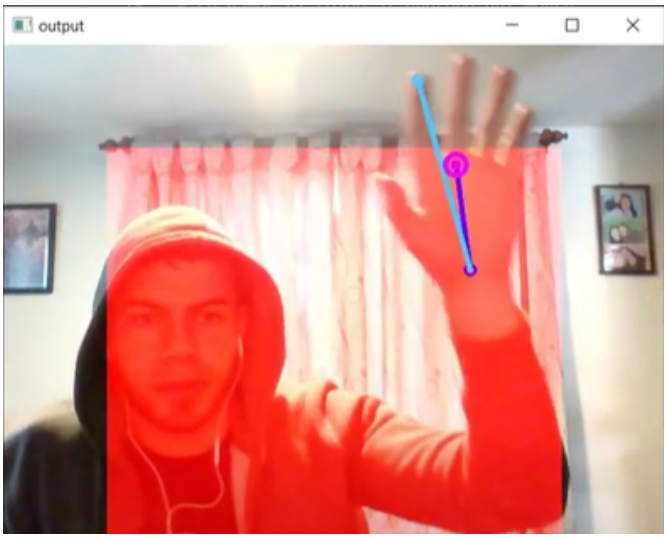
3. Resultados y discusión.

Una vez que ya implementamos el código necesario para lograr abrir la cámara y la detección de manos con MediaPipe, procedimos a probar la eficiencia del mismo, Probamos principalmente con 2 manos y funciono correctamente hasta con 6 manos.

Además de probar si se comportaba de manera correcta en el momento de abrir o cerrar las manos.Lo cual funcionó correctamente, Podemos decir que la inteligencia artificial que usa MediaPipe está muy enriquecida y logra detectar las manos a la perfección.

Se adjuntará el código con el cual se logra la detección de manos básico.Pero posteriormente decidimos aplicar un

código que logre interactuar de mejor manera y decidimos probar que con el seguimiento de las manos se lograse el uso del Puntero del mouse y a su vez se pudiera dar click, para este se teniendo en cuenta la ubicación de 3 de los puntos clave de la mano. Usamos el punto 6 siendo este el central de la mano para tomar el seguimiento del puntero. Con los puntos 8 y 0 aplicamos el click donde se tendrán en cuenta la distancia que hay entre ellos cuando se cierre la mano estos estarán más cerca y se hará la acción de click



Así mismo el código en el que se implementa el seguimiento de la mano para el uso de Puntero se anexará además de un video donde se muestra el funcionamiento de estos dos códigos.

https://www.youtube.com/watch?v=oPgwsFYJAJc

CONCLUSIÓN.

1. Se logró la identificación de las manos y sus movimientos en tiempo real.
2. Se implementó las funciones de mouse en la mano mediante la cámara.
3. Se analizó las diferentes funciones que tiene mediapipe para la lectura de los movimientos de las manos y la gran variedad de utilidades que esta nos ofrece como lo eran la detección de poses, rostros, objetos.
4. Se alcanzó a identificar un máximo de 6 manos al mismo tiempo.

REFERENCES

[1] VALENTÍN BAZAREVSKY, YURY KARTYNNIK, ANDREY VAKUNOV, KARTHIK RAVEENDRAN Y MATTHIAS GRUNDMANN. BLAZE FACE: DETECCIÓN NEURAL DE ROSTROS EN SUB MILISEGUNDOS EN GUS MÓVILES.

[2] Liuhao Ge, Hui Liang, Junsong Yuan y Daniel Thalmann. . Conferencia IEEE sobre visión artificial y reconocimiento de patrones, páginas 3593–3601, 2016. 4321.

[3] ***Christian Zimmermann, Thomas Brox*** ; Actas de la Conferencia internacional IEEE sobre visión artificial (ICCV), 2017, págs. 4903-4911

[4] ***Xiong Zhang, Qiang Li, Hong Mo, Wenbo Zhang, Wen Zheng*** ; Actas de la Conferencia internacional IEEE/CVF sobre visión artificial (ICCV), 2019, págs. 2354-2364

[5] Liuhao Ge, Zhou Ren, Yuncheng Li, Zehao Xue, Yingying Wang, Jianfei Cai y Junsong Yuan. forma de mano 3d y estimación de la pose a partir de una sola imagen rgb. En Actas de la conferencia IEEE sobre visión artificial y reconocimiento de patrones, páginas 10833–10842, 2019. 4321

[6] Fan Zhang Valentin Bazarevsky Andrey Vakunov Andrei Tkachenka George Sung Chuo-Ling Chang Matthias Grundmann Google Research 1600 Amphitheatre Pkwy, Mountain View, CA 94043, EE. UU.

[7] Iason Oikonomidis, Nikolaos Kyriazis y Antonis A Argyros. Seguimiento 3D eficiente basado en modelos de las articulaciones de la mano utilizando Kinect. 4321

[8] Andrea Tagliasacchi, Matthias Schroder, Anastasia Tkach, Sofien Bouaziz, Mario Botsch y Mark Pauly. Robusto

[9] https://codepen.io/mediapipe/pen/RwGWYJw

[10] https://google.github.io/mediapipe/solutions/hands

[11] https://google.github.io/mediapipe/

[12] https://omes-va.com/mediapipe-hands-python/

[13] https://learnopencv.com/introduction-to-mediapipe/