
BENCHMARKING DETECCIÓN DE PASOS DESDE MODELOS ARTICULADOS

Hito 2 - IPD441

Integrantes

Ignacio Barrera
Francisca Soto

Fecha:

21-06-2024

Contents

1 Problema	3
2 Solución	4
3 Técnicas Estudiadas	5
4 Datasets	6
5 Métricas y Resultados	7
6 Análisis de Resultados	10
7 Código	10

1 Problema

Con el fin de entrenar a deportistas, se ha desarrollado una plataforma que busca realizar un análisis cognitivo y motor de estos, el funcionamiento de esta es mediante estímulos presentados a los deportistas los cuales le indican a posición moverse dentro de un esquema de nueve posiciones, esta plataforma se basa en una solución de análisis de video a través de una cámara frontal frente al circuito de reacción (ver Figura 1). Para la parte de análisis, se busca un algoritmo el cual logre detectar cuando el jugador pisa las marcas situadas en el circuito, hasta el momento, la solución implementada utiliza modelos de segmentación a través de procesamiento digital de imágenes para detectar las posiciones de los pies, la zona a pisar y la intersección de estas, sin embargo con el avance de tecnología y la inestabilidad de los modelos mencionados, surge la propuesta de utilizar modelos de Deep Learning, mas específicamente modelos 2D-SPPE[1] (Single Person Pose Estimation), el esquema explicativo de los tipos de modelos de HPE (Human Pose Estimation) se muestra en la figura 2.



Figure 1: Captura de escenario de Análisis Neuromuscular

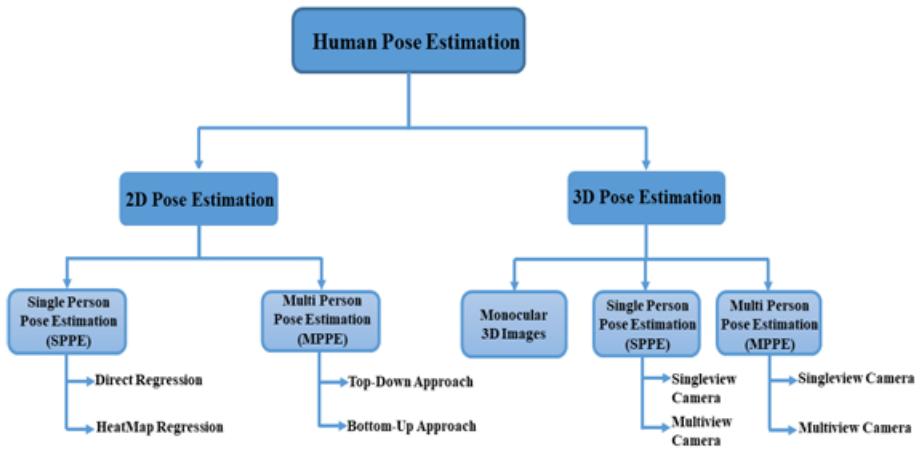


Figure 2: Cuadro resumen de tipos de estimación de pose

2 Solución

La estimación de pose humana presenta diferentes alternativas de implementación, para este trabajo nos enfocaremos en detectar una persona en 2D con modelos kinematicos (ver a. en figura 3), por lo que los modelos que nos serán útiles corresponden a los modelos de análisis por medio de mapas de calor y los modelos de regresión directa, notemos que modelos que posean tracking pueden aportar un gran valor a la solución gracias a la coherencia tiempo-espacial que presenta nuestro problema. Sin embargo, modelos de detección de múltiples persona también puede ser de interés, ya que en nuestro contexto solo aparece una persona en los vídeos por lo que el modelo no debería detectar mas de una persona de igual forma. Finalmente proponemos tres modelos principales para el análisis de la detección de pose, con el fin de encontrar el mejor para nuestro objetivo, nuestro primer paso sera validar los modelos a través de la obtención de métricas utilizando dos diferentes sub-datasets de COCO (datasets que poseen los pies etiquetados), posteriormente se procesaran los datos obtenidos por los modelos de pose en un algoritmo que busca detectar las pisadas de los jugadores basándose en un umbral de movimiento entre frame, para validar este segundo algoritmo, se utilizará un dataset propio obtenido de la plataforma de entrenamiento, nuestra propuesta se ve de manera resumida en la Figura 4.

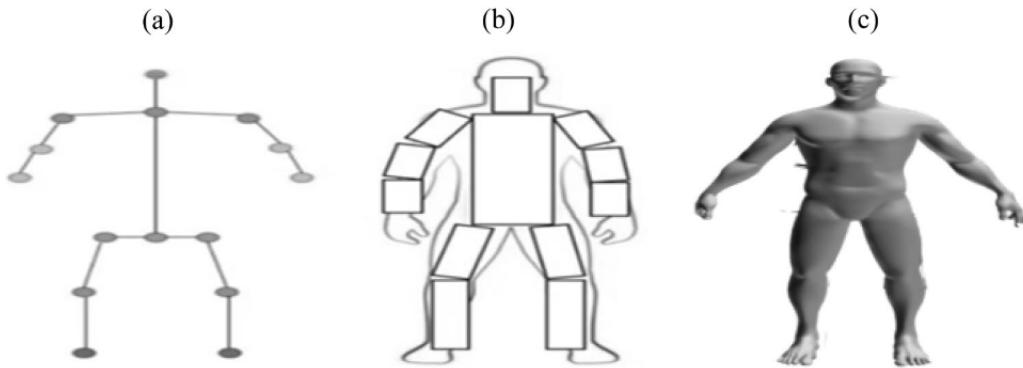


Figure 3: Tipos de esqueletos

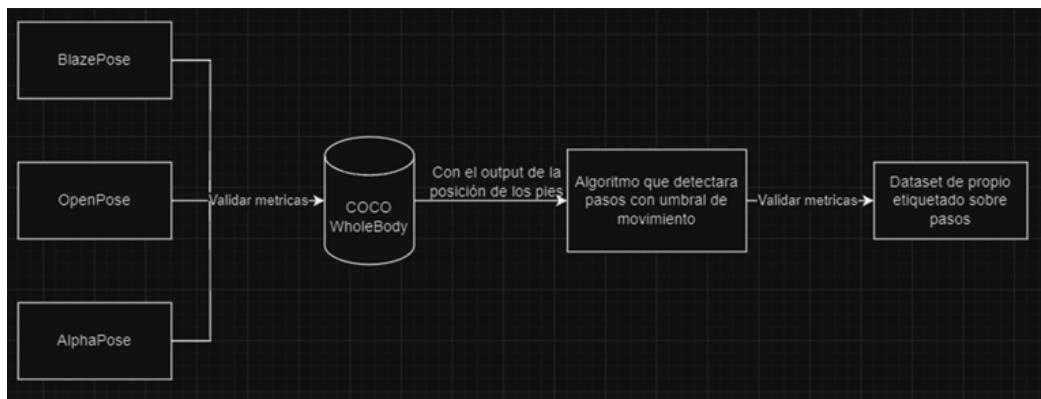


Figure 4: Propuesta de solución

3 Técnicas Estudiadas

Como se mencionó previamente, se utilizarán tres técnicas (modelos) de detección de pose, la elección de estos modelos se deben principalmente a como distribuyen los keypoints del cuerpo, los tres modelos elegidos poseen tres keypoints para cada pie. esto es fundamental ya que es ahí donde nos enfocaremos para lograr el objetivo de este trabajo, de igual manera para comparar la estimación de pose se comprobó la detección con otros algoritmos, por ejemplo YoloV8, sin embargo este algoritmo detecta más de una persona y establece solo un keypoint en el pie como se puede ver en la Figura 5.

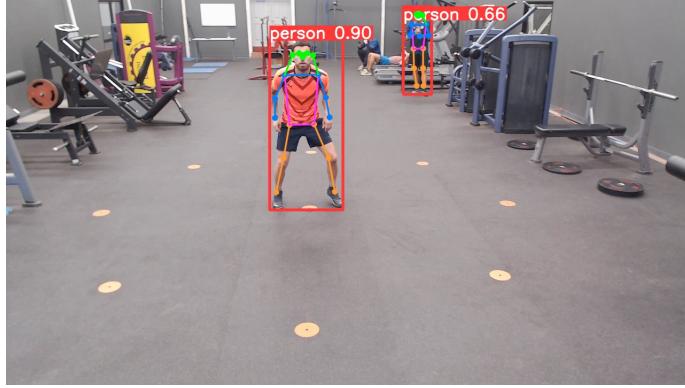


Figure 5: Imagen procesada por YoloV8

Ahora bien, enfocándonos en la cantidad de keypoints que se utilizan en los pies se utilizará **OpenPose** [2] ya que permite el tracking para una o varias personas y utiliza tres keypoints en el pie de manera moderada, pues en esta técnica la arquitectura propuesta codifica el contexto global, permitiendo un paso de análisis ascendente que mantiene una alta precisión y al mismo tiempo logra un rendimiento en tiempo real, independientemente del número de personas en la imagen, en la Figura 6 se aprecia una imagen procesada por el algoritmo.

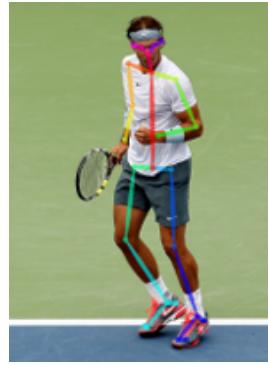


Figure 6: Ejemplo imagen procesada por OpenPose

La siguiente técnica es utilizando **BlazePose** [3] donde realiza tracking de una persona con tres keypoints en el pie, mediante una arquitectura de una red convolucional ligera que produce 33 puntos claves en una persona, además permite que el uso en tiempo real mediante

keypoints como con mapas térmicos. La asociación de los 33 keypoints de BlazePose se pueden ver en la Figura 7.

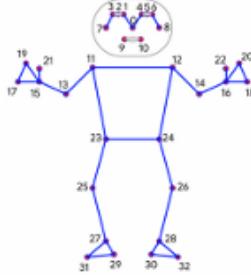


Figure 7: Ejemplo imagen procesada por BlazePose

Finalmente la última técnica estudiada es **AlphaPose** [4] que es un sistema que puede realizar una estimación precisa de la pose de todo el cuerpo y un seguimiento conjunto mientras se ejecuta en tiempo real, utilizando diversas técnicas. Un ejemplo de imagen procesada se puede ver en la Figura 8.

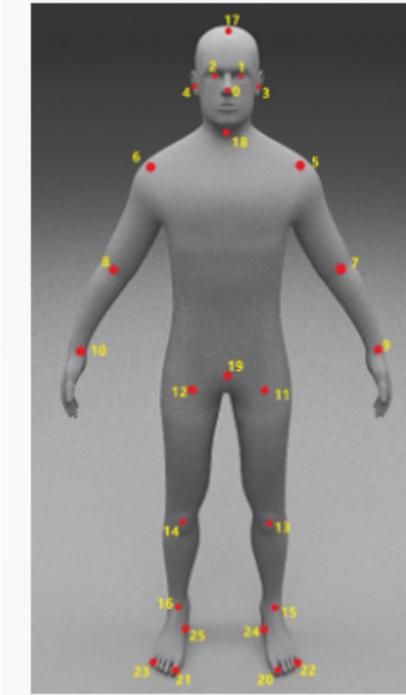


Figure 8: Ejemplo imagen procesada por AlphaPose

4 Datasets

En este trabajo utilizaremos dos sub-datasets que son extensiones de COCO (Common Objects in Context)[5] que incluye 80 categorías de objetos comunes como personas, vehículos, animales, entre otros. El primer [dataset](#) llamado foot-keypoint es de interés nos es útil

ya que incluye anotaciones de keypoints que indican la ubicación de partes del cuerpo, específicamente para el caso de estudio se etiquetan seis puntos claves en el pie como se puede apreciar en la Figura 9. Además utilizaremos COCO-WholeBody [6] que es un subdataset de COCO con cinco mil anotaciones de cuerpo entero y contiene 133 puntos de referencia en todo el cuerpo, específicamente 68 en la cara, 42 en las manos, 17 en el cuerpo y 6 en el pie, este presenta una estructura muy similar a foot-keypoint. Aclaráramos que por el momento no hemos logrado tener acceso al dataset foot-keypoint, ya que los servidores de descarga se encuentran caídos, por lo que hasta ahora, solo hemos trabajado sobre WholeBody.



Figure 9: Keypoints pie de COCO

5 Métricas y Resultados

Hasta el momento se han ejecutado los modelos y se han obtenido las siguientes métricas, Avarage Precision (AP), Avarage Recall (AR) y AP según tamaño del objeto, que se detallan a continuación:

- AP @ IoU=0.50:0.95 (mAP) - El promedio de precisiones en 10 umbrales de IoU, desde 0.50 hasta 0.95 con un paso de 0.05.
- AP @ IoU=0.50 - La precisión cuando el umbral de IoU es 0.50 (AP más permisivo).
- AP @ IoU=0.75 - La precisión cuando el umbral de IoU es 0.75 (AP más estricto).
- AP @[IoU=0.50:0.95 — area=medium — maxDets= 20] - El promedio de precision con umbrales de IoU desde 0.50 hasta 0.95 de objetos medianos con un máximo de 20 detecciones por imagen.
- AP @[IoU=0.50:0.95 — area=large — maxDets= 20] - Idem caso anterior pero para objetos grandes.
- AR @[IoU=0.50:0.95 — area=all — maxDets= 20] - media de recall a través de múltiples umbrales de IoU desde 0.50 hasta 0.95 para todos los tamaños de objetos con un máximo de 20 detecciones por imagen.

- AR @[IoU=0.50 — area=all — maxDets= 20] - Recall con umbral fijo de IoU de 0.50.
- AR @[IoU=0.75 — area=all — maxDets= 20] - Se calcula recall a un umbral fijo de IoU de 0.75.
- AR @[IoU=0.50:0.95 — area=medium — maxDets= 20] - Similar a la primera métrica de recall, pero solo para objetos medianos.
- AR @[IoU=0.50:0.95 — area=large — maxDets= 20] - Idem caso anterior pero con objetos grandes.

En estos casos AP mide la precisión a través de diferentes umbrales de IoU, representando la capacidad del modelo para detectar correctamente las poses de las personas con diferentes niveles de precisión, mientras que AR mide el recall, indicando la capacidad del modelo para detectar la mayoría de las poses de las personas en diferentes umbrales de IoU.

Los resultados obtenidos son:

- OpenPose: Las métricas resultantes se aprecian en Figura 10 y una imagen resultante de la estimación de pose en la Figura 11.

Average Precision (AP) @[IoU=0.50:0.95 area= all maxDets= 20] = 0.545
Average Precision (AP) @[IoU=0.50 area= all maxDets= 20] = 0.737
Average Precision (AP) @[IoU=0.75 area= all maxDets= 20] = 0.582
Average Precision (AP) @[IoU=0.50:0.95 area=medium maxDets= 20] = 0.538
Average Precision (AP) @[IoU=0.50:0.95 area= large maxDets= 20] = 0.724
Average Recall (AR) @[IoU=0.50:0.95 area= all maxDets= 20] = 0.614
Average Recall (AR) @[IoU=0.50 area= all maxDets= 20] = 0.770
Average Recall (AR) @[IoU=0.75 area= all maxDets= 20] = 0.645
Average Recall (AR) @[IoU=0.50:0.95 area=medium maxDets= 20] = 0.539
Average Recall (AR) @[IoU=0.50:0.95 area= large maxDets= 20] = 0.728

Figure 10: Metricas obtenidas en OpenPose



Figure 11: Estimación de pose con OpenPose

- BlazePose: Las métricas y la estimación de pose se aprecian en las figuras 12 y 13, respectivamente.
- AlphaPose: hasta el momento de este informe aun no se tienen las metricas resultantes utilizando este método, sin embargo la estimación de pose se aprecia en la figura 14

```

Average Precision (AP) @[ IoU=0.50:0.95 | area= all | maxDets= 20 ] = 0.088
Average Precision (AP) @[ IoU=0.50 | area= all | maxDets= 20 ] = 0.164
Average Precision (AP) @[ IoU=0.75 | area= all | maxDets= 20 ] = 0.087
Average Precision (AP) @[ IoU=0.50:0.95 | area=medium | maxDets= 20 ] = 0.029
Average Precision (AP) @[ IoU=0.50:0.95 | area= large | maxDets= 20 ] = 0.170
Average Recall (AR) @[ IoU=0.50:0.95 | area= all | maxDets= 20 ] = 0.132
Average Recall (AR) @[ IoU=0.50 | area= all | maxDets= 20 ] = 0.201
Average Recall (AR) @[ IoU=0.75 | area= all | maxDets= 20 ] = 0.140
Average Recall (AR) @[ IoU=0.50:0.95 | area=medium | maxDets= 20 ] = 0.039
Average Recall (AR) @[ IoU=0.50:0.95 | area= large | maxDets= 20 ] = 0.258

```

Figure 12: Metricas obtenidas en BlazePose



Figure 13: Estimación de pose con BlazePose



Figure 14: Estimación de pose con AlphaPose

6 Análisis de Resultados

Si bien se han obtenido métricas interesantes, creemos que es necesario complementarlas con OKS (Object Keypoint Similarity) que si bien es calculada como métrica base para las demás calculadas, creemos que es necesario imprimirlas para favorecer la comprensión de la detección de pose y el resultado de las otras métricas.

Además se debe considerar un trade-off de precisión versus el costo computacional que este genera, ya que obtener métricas exactas con los dataset genera un largo tiempo de ejecución y este costo debe ser considerado pensando en que el siguiente paso de nuestro trabajo y en la futura implementación de estos modelos en la plataforma.

Otra situación crítica es por medio del uso del formato de COCO ya que cada modelo genera anotaciones diferentes por lo que se debe convertir las anotaciones a un formato específico para realizar la validación y comparación de los modelos. Sin embargo, a pesar de la conversión, modelos como BlazePose generan un tipo de distribución diferente de keypoints a los demás por lo que esto nos genera un problema para realizar una comparación objetiva con este tipo de modelos, es por esto que planeamos como trabajo futuro idear un método que permita convertir esta distribución a una aceptada por la notación COCO. Debido a lo anterior, es que al momento de obtener las métricas de BlazePose obtenemos valores bajos.

Debido a todo lo mencionado previamente, no logramos aún elegir un modelo específico para trabajar y avanzar con los siguientes pasos, sin embargo ya con las métricas obtenidas y completas, se podrá decidir qué modelo(s) utilizar para comenzar el proceso de detección de los pasos.

7 Código

Los códigos desarrollados se pueden encontrar en el siguiente enlace https://github.com/ignacio-barrera/pose_tracker.

References

- [1] S. Dubey and M. Dixit, “A comprehensive survey on human pose estimation approaches,” *Multimedia Systems*, vol. 29, no. 1, pp. 167–195, 2023.
- [2] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, “Realtime multi-person 2d pose estimation using part affinity fields,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7291–7299.
- [3] V. Bazarevsky, I. Grishchenko, K. Raveendran, T. Zhu, F. Zhang, and M. Grundmann, “Blazepose: On-device real-time body pose tracking,” *arXiv preprint arXiv:2006.10204*, 2020.
- [4] H.-S. Fang, J. Li, H. Tang, C. Xu, H. Zhu, Y. Xiu, Y.-L. Li, and C. Lu, “Alphapose: Whole-body regional multi-person pose estimation and tracking in real-time,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.
- [5] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft coco: Common objects in context,” *European conference on computer vision*, pp. 740–755, 2014.
- [6] S. Jin, L. Xu, J. Xu, C. Wang, W. Liu, C. Qian, W. Ouyang, and P. Luo, “Whole-body human pose estimation in the wild,” in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IX 16*. Springer, 2020, pp. 196–214.