Análisis de Video en Biomecánica

Andréi Guchin, Gonzalo Pereira,
Guillermo Ottado, Mauricio Ramos, and Juan Cardelino.

Instituto de Ingeniería Eléctrica, Facultad de Ingenieria,
Universidad de la República, Uruguay.
proyecto.iie.biomecanica@gmail.com
http://www.fing.edu.uy

Resumen En este artículo se propone un sistema óptico de captura de movimiento basado en marcadores para facilitar la tarea en el análisis biomecánico del movimiento de las personas.
Dicho sistema se compone de bloques independientes, dando una solución general que posibilita modificar o sustituir sus componentes fácilmente.
Se describen lineamientos para generar un laboratorio virtual a partir del cual obtener una base de datos con secuencias de videos sintéticas y se exploran las características que debe tener un laboratorio de captura óptico basado en marcadores para facilitar el procesamiento.
Se reune un conjunto de métricas para medir la performance de cada bloque y del sistema en su totalidad.
Las pruebas realizadas sobre el software implementado reflejaron que el mismo tiene una precisión del orden del centímetro sobre secuencias obtenidas en ambientes controlados. Estos resultados son buenos teniendo en cuenta que los algoritmos utilizados en cada bloque son de complejidad baja y se pueden optimizar.

Keywords: Biomecánica, calibración, detección de marcadores, reconstrucción, seguimiento, laboratorio virtual.

1. Introducción

El análisis de video es una herramienta fundamental para la recolección y estudio de datos. El seguimiento de puntos de referencia se utiliza para el cálculo de posición y otras variables asociadas como son la velocidad, la aceleración y por ende desplazamientos. Trabajar con video permite además estudiar secuencialmente situaciones estáticas en el tiempo. El seguimiento de dichos puntos resultaría tedioso si se hiciera manualmente por lo que resulta necesario contar con una herramienta que realice esta tarea automáticamente. Algunos ejemplos correspondientes a distintas áreas que ilustran estas necesidades son a *nivel asistencial* en el área de *fisioterapia*, *investigación académica en biomecánica, medidas de performance* en el deporte de alto nivel y *Animación 3D*, entre otras. Los ejemplos mencionados anteriormente definen distintos casos de uso con características disímiles, de manera que la búsqueda de una solución única que abarque las necesidades particulares de todos ellos resulta compleja. Contar con

este tipo de herramientas es fundamental para las necesidades de los equipos de profesionales, cuya alternativa son productos comerciales de alto costo.

En virtud de estas necesidades, este proyecto buscó realizar una aplicación básica y funcional de código abierto de análisis de video, que proporcione solución a las necesidades que se describieron ya sea utilizando como base algún proyecto de software libre existente, o en su defecto, desarrollando un prototipo de software básico completo que abarque el problema en forma general, para luego estudiar extender la aplicación hacia otros casos de uso.

El sistema creado pretende bajo ciertas condiciones controladas, obtener las coordenadas espaciales de un número de puntos de interés sobre un paciente. Una de las formas de obtener esto, y la estudiada en este trabajo, es colocar al sujeto con traje negro con marcadores en un ambiente con iluminación adecuada, filmar con varias cámaras a lo largo del tiempo, adquirir esta información en la computadora y mediante un posterior procesamiento obtener la posición 3D de cada uno de los puntos de interés (los marcadores) a lo largo de toda la secuencia.

2. Base de datos

Con el fin de implementar, testear y comparar los distintos tipos de algoritmos desarrollados por el sistema, es deseable poseer múltiples secuencias de videos 2D de movimiento obtenidas a partir de cámaras situadas en un entorno 3D cerrado, previamente acondicionado. Así como también contar con el correspondiente ground truth 2D y 3D de los datos de movimiento disponibles, así como la información de calibración de las cámaras utilizadas para efectuar las capturas.

2.1. Características de Laboratorio

A continuación se enumeran algunas variables que es necesario tener en cuenta a la hora de diseñar un laboratorio adecuado para un sistema de captura óptica basado en marcadores.

- **Cámaras**. La resolución espacial de los datos en el procesamiento condiciona la resolución de la cámara, la resolución temporal para la marcha requiere como mínimo 30 cuadros por segundo, con tiempos de obturación de al menos $1/2000\,s$, esto último permite evitar efectos de distorsión debidos a falta de nitidez.
- **Marcadores**. El color de los marcadores debe contrastar claramente con la vestimenta y el fondo del espacio de captura, se recomienda una forma esférica. En capturas con cámaras ubicadas a menos de 12 metros del movimiento a relevar, un tamaño aceptable para el marcadores es de $3\,cm$ de diámetro.
- **Vestimenta**. Debe ser ajustada, para despreciar fluctuaciones en la posición de los marcadores y preferiblemente de igual color que el fondo.
- **Iluminación**. debe ser uniforme, si se utiliza iluminación artificial con focos puntuales es habitual colocar pantallas difusoras delante de los focos.

- **Espacio de captura**. debe contrastar con los marcadores, y sus dimensiones varían según el tipo de marcha a relevar. En caso de marcha rectilínea sobre una plataforma de $3\,m \times 5\,m$ se encuentra que 4 son el mínimo número de cámaras que permiten relevar el movimiento de manera satisfactoria. Mientras que en el caso de la marcha libre sobre una plataforma circular de $5\,m$ de diámetro, se recomienda la utilización de al menos 8 cámaras.

2.2. Laboratorio Virtual

Utilizando la suite de animación 3D gratuita y de código abierto de *Blender*, se genera un laboratorio de captura de movimiento virtual, donde se obtienen secuencias de movimiento sintéticas con sus respectivos videos a partir de un modelo virtual 3D. Si bien las secuencias de video obtenidas son lo único necesario para el análisis posterior, al generar dichas secuencias a través de un entorno virtual controlado como lo es *Blender*, se puede probar múltiples configuraciones contando en cada caso, con la información exacta del ambiente de captura.

Blender cuenta con una interfaz de programación de aplicaciones flexible, que permite extender su funcionalidad a través de programas en Python. Lo cual permite automatizar varias etapas en el desarrollo de nuevas secuencias, así como gestionar la exportación de información desde *Blender* a otros lenguajes.

Disposición del Laboratorio Un laboratorio con 17 cámaras dispuestas uniformemente sobre un ambiente rectangular de $10 \times 15\ m$ permite probar diferentes combinaciones de captura a la hora de generar secuencias. Para iluminar el laboratorio basta con 8 focos puntuales de luz omnidireccional sobres los límites del mismo, rodeando la escena y a un nivel de 3 metros de altura. Cuidando que ninguno de los focos sea tomado directamente por las cámaras y los marcadores se iluminen correctamente.

Modelo virtual La relevancia del modelo en las capturas es simular las oclusiones de marcadores debida a los miembros del sujeto en una captura real. El modelo virtual utilizado se basa en un maniquí de madera convencional, el mismo se ajusta con facilidad a una posición particular, siendo cada miembro fácilmente correlacionado con un hueso específico, sin perder las particularidades propia de un sujeto real.

Esqueleto El esqueleto del modelo contiene la información de movimiento, el mismo se obtuvo de la base de datos *MotionBuilder-friendly versión* ofrecidas por *cgspeed* [1], donde se cuenta con las fuentes BVH que provienen de las capturas de movimiento de *Carnegie Mellon University Motion Capture Database*. Con el fin de normalizar las secuencias BVH [1] provenientes de MotionBuilder se ha utilizado la herramienta de edición de archivos BVH *bvhacker* [13]. La

[1] Biovision Hierarchical data, formato de captura de movimiento desarrollado por Biovision.

misma permite centrar las secuencias sobre un mismo punto en el primer cuadro removiendo los offset globales. Una vez gestionado lo anterior se importa en *Blender* la secuencia BVH y se procede a generar los marcadores en las articulaciones del esqueleto, dado que se tiene la posición exacta del origen de cada hueso en el esqueleto y la articulación es la unión entre dos huesos consecutivos, se puede obtener la posición exacta de los marcadores a partir de la secuencia BVH. La generación de los marcadores y el enlazado del esqueleto al modelo virtual se realiza a través de un código python. Alguno de los factores que justifican esta elección son que C.M.U. dispone de un gran número de capturas de movimiento de acceso público, varias utilidades de software que permiten llevar a otros formatos y es utilizado ampliamente en el ámbito de la animación por computadora.

Renderizado y ground truth Una vez se dispone de la secuencia de movimiento en el entorno virtual de *Blender*, lo que resta es renderizar dicha secuencia sobre las cámaras del laboratorio. Para contar con el ground truth final de la secuencia animada se debe exportar desde *Blender* la información del esqueleto en un BVH, cuidando de habilitar que se mantenga la misma escala de tiempos que en el momento del renderizado. De esta manera se tienen sincronizados los videos y el ground truth 3D.

3. Implementación

Un sistema de captura de movimiento con las características necesarias para cumplir el objetivo de este proyecto debe estar formado por cuatro bloques generales: *calibración*, *detección de marcadores*, *reconstrucción* y *seguimiento*. En la figura 1 se muestra un esquema del sistema a implementar.

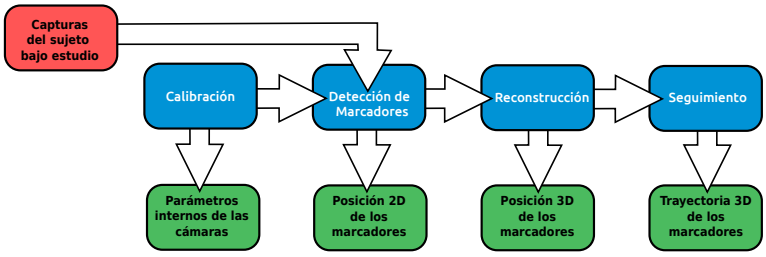


Figura 1: Diagrama de bloques del sistema completo.

Es importante destacar el hecho de poder separar el sistema en bloques independientes. Esto asegura que el funcionamiento de uno de ellos no dependa del funcionamiento de otro. Por otro lado, da la posibilidad que en etapas futuras se pueda realizar el estudio de uno de los bloques de la Figura 1 individualmente y así poder modificarlo u optimizarlo sin afectar al resto.

En los capítulos que siguen, se explicará el funcionamiento de cada bloque de forma detallada, así como su implementación y el análisis de resultados de cada uno de ellos.

4. Calibración

Con el objetivo de establecer una metodología de calibración que fuera válida para la configuración de cámaras con las que se diseñó el entorno virtual descrito en 2, se probaron distintas implementaciones existentes. Para esto se evaluaron dos toolbox elaborados en *Matlab*. La metodología de calibración fue simulada en *Blender* mediante *scripts Python* y las imágenes obtenidas como resultado se procesaron con dichos toolbox. Para cada uno de los toolbox se describe la metodología y las simulaciones realizadas.

Automatic Multi-Camera Calibration Toolbox (amcctoolbox) [12]

Esta herramienta automatiza ciertas funciones del toolbox *Camera Calibration Toolbox*, reduciendo la intervención del usuario. Utiliza como objeto de calibración un damero.

Para la calibración de conjuntos de varias cámaras, el damero debe ser colocado en múltiples ubicaciones y orientaciones cubriendo el mayor volumen de trabajo tal que, en cada una de estas posiciones, el damero pueda ser visto por más de una cámara simultáneamente.

La calibración del entorno virtual se realiza simulando un damero real. Para cada par de cámaras adyacentes se coloca el damero en algún punto dentro del volumen de trabajo, con una orientación tal que la figura plana del damero sea visible en ambas cámaras. Luego, a partir de dicha posición, se toman capturas del damero cambiando ligeramente la ubicación y orientación del mismo. Para cada par de cámaras adyacentes se consideran entre 15 y 20 posiciones distintas del damero.

Una vez realizadas las capturas se procesan las imágenes mediante el toolbox, obteniéndose los parámetros intrínsecos de cada una de las cámaras y la posición relativa de cada uno de los pares de cámaras adyacentes y por lo tanto las posición relativa de todas las cámaras.

Este método aunque sus resultados tiene buena precision [7], puede no ser suficientemente flexible para un sistema de muchas cámaras, ya que, entre otras cosas, puede ser necesaria la intervención manual en algunos casos.

Multi-Camera Self-Calibration Toolbox [11]

El procedimiento utilizado en este toolbox consiste en capturar el movimiento de una fuente puntual de luz que recorra el volumen de trabajo. Por lo tanto para cada cuadro se tiene un punto 3D en el espacio en una posición distinta y en cada una de las cámaras su correspondiente proyección si dicho punto es visible desde la cámara. Para esto debe existir un contraste suficiente de la fuente puntual de luz respecto al laboratorio. La fuente de luz puede ser por ejemplo, una lámpara led.

La simulación de este procedimiento se logra creando un punto 3D que toma para cada cuadro, distintas posiciones en forma aleatoria dentro del volumen de trabajo. Para cada cuadro se *renderiza* su posición en las 17 cámaras. En este caso se han tomado 500 posiciones distintas.

El error de re-proyección promedio obtenido es menor a 0.13 píxeles para todas las cámaras. Este método plantea una forma simple de calibrar un conjunto de muchas cámaras adecuado para el sistema de 17 cámaras del laboratorio virtual desarrollado en *Blender*.

5. Detección de marcadores

Una vez realizada la captura de video del paciente, el primer paso en el procesamiento de las secuencias de video es reconocer los marcadores en el cuerpo del sujeto.

El bloque de detección de marcadores, se puede dividir en dos partes: la **segmentación** y el **filtrado de objetos**. En la Figura **??** se muestra el resultado del procesamiento de cada etapa.

El algoritmo realiza la detección siguiendo el siguiente proceso:

1. Se recibe como entrada un video y este es separado en cada uno de sus cuadros.
2. Se toma un cuadro y se lo segmenta utilizando umbralización de Otsu.
3. A partir de la imagen segmentada, se identifican los marcadores.
4. Se escribe la posición de los marcadores detectados para este cuadro en un archivo con formato XML.
5. Se toma el siguiente cuadro y se repite el proceso a partir del paso 2.

Este bloque fue implementado en lenguaje C++, debido a que es uno de los lenguajes de programación que cuenta con mayor cantidad de recursos para procesamiento de imágenes. En particular, se utilizaron las librerías *OpenCV* [8] y CVBlob [2] ya que funcionan para las plataformas principales de PC y dispositivos móviles, y están diseñadas para tener una gran eficiencia computacional en las implementaciones.

5.1. Descripción de las etapas de detección

Segmentación En lo que respecta al bloque de segmentación, se eligió utilizar la umbralización generando umbrales con el método de Otsu[9] de tres clases.

Trabajar con tres clases permite ser un poco más flexible con los contrastes entre los marcadores y el resto de la imagen por lo que no sería estrictamente necesario, por ejemplo, que el traje del paciente y el fondo sean del mismo color.

Filtrado La etapa de filtrado no es más que una clasificación de los objetos segmentados. Dado que los objetos a detectar tienen formas relativamente sencillas (círculos blancos sobre fondo oscuro) y las condiciones de laboratorio son controladas al realizar la captura, esta etapa no requirió implementar algoritmos muy complejos. En particular, se implementó un detector de objetos circulares en base a momentos geométricos[4] y un filtro según el área de los mismos.

5.2. Resultados

Para la etapa de segmentación se observó que el resultado obtenido depende fuertemente de las condiciones de captura y del umbral calculado. Se debe tener especial atención en las condiciones de captura ya que de no cumplir con las establecidas los resultados no son del todo satisfactorios (ver Figura 2). Por otro lado, si las capturas se hacen dentro de las condiciones establecidas, los resultados obtenidos son mucho mejores (ver Figura 2).

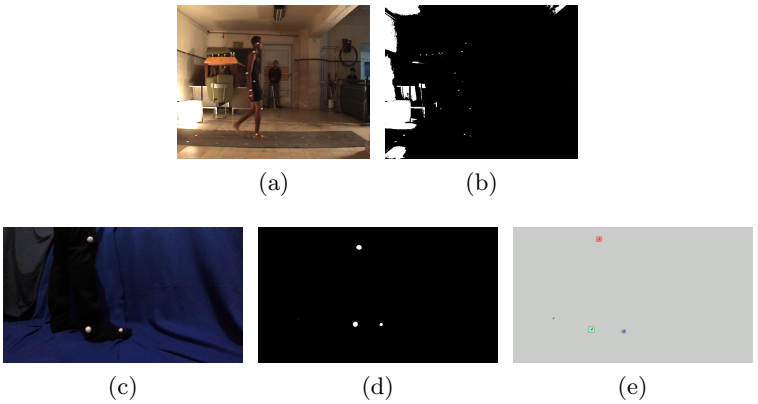


Figura 2: Entradas y salidas de cada etapa del bloque de deteción. **(2a)** Imagen original de una secuencia fuera de las hipótesis de captura. **(2b)** Imagen segmentada con el umbral de Otsu. **(2c)** Captura original de una secuencia real bajo las hipótesis de captura. **(2d)** Imagen filtrada con el umbral de Otsu. **(2e)** Marcadores detectados.

6. Reconstrucción

A la salida del bloque de detección de marcadores se tiene, para cada cámara y para cada cuadro de una secuencia adquirida, un conjunto de coordenadas en dos dimensiones (x, y) que ubican la posición en la imagen de aquellos marcadores que fueron detectados. El proceso de reconstrucción consiste en obtener las

coordenadas en tres dimensiones de la posición de los marcadores en el espacio, a partir de la posición de los marcadores en al menos dos retinas.

El proceso de reconstrucción que se presenta fue inspirado en el trabajo de Herda [3] y consiste en tres pasos fundamentales:

1. Encontrar la correspondencia entre puntos en retinas diferentes.
2. Seleccionar la mejor correspondencia.
3. Reconstruir y verificar en el resto de las retinas.

6.1. Algoritmo

El algoritmo implementado recibe como entrada los puntos 2D de los marcadores detectados y devuelve como salida los puntos 3D reconstruidos. En la Figura 3 se presenta un diagrama del algoritmo.

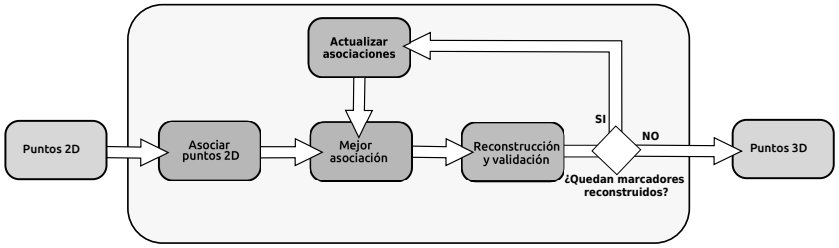


Figura 3: Diagrama de bloques del algoritmo de reconstrucción.

Asociar puntos 2D Este bloque recibe como entrada las coordenadas de los puntos detectados en cada una de las cámaras, parámetros de las mismas tales como sus matrices de proyección y devuelve para cada punto una lista ordenada por relevancia, de las asociaciones existentes con puntos en otras cámaras. Los pasos a seguir son los siguientes:

- Seleccionar dos cámaras y considerar un punto en una de ellas, por ejemplo el punto x_{11} de la cámara 1.
- Proyectar la recta epipolar l_{11} correspondiente al punto x_{11} sobre la cámara 2.
- Tomar las distancias de los puntos detectados en la cámara 2 a la recta l_{11}.
- Generar una lista de posibles asociaciones ordenada por distancia.

Se asume que los puntos de la cámara 2 que tengan mayor posibilidad de corresponder con el punto x_{11}, son los más próximos a l_{11}. Repitiendo el procedimiento de manera inversa, esto es, de la cámara 2 a la cámara 1, se obtiene igualmente para cada punto de la cámara 2 los puntos de la cámara 1 ordenados según su

proximidad a la recta epipolar correspondiente. A continuación se toman otros pares de cámaras y se vuelve a repetir el proceso.

Es importante resaltar que para la elección de los pares de cámaras se han considerado dos casos. El primero de ellos evalúa cada cámara respecto a todas las restantes y el segundo considera la disposición de las cámaras en el espacio y empareja las cámaras adyacentes de manera consecutiva.

6.2. Mejor asociación

A partir de la lista con asociaciones entre puntos de dos vistas generada anteriormente, es necesario elegir aquella que posea mayor probabilidad de conformar la pareja de imágenes correspondiente a la proyección de un marcador 3D sobre dichas vistas.

Recordando que todas las asociaciones de puntos entre pares de cámaras se encuentran ordenadas por distancia, se toma aquella asociación que posea la menor distancia y contenga puntos válidos, descartando las restantes. Un punto se considera válido si en iteraciones anteriores no se a podido asociar a ningún punto 3D reconstruido (ver Sección 3). De esta forma cada punto de una cámara es asociado, si existen puntos válidos, con un punto en otra de las cámaras.

Para evaluar de todas las relaciones entre cámaras, cuál de los pares de puntos asociados disponibles es el par que posee mayor posibilidad de corresponder a las proyecciones de un punto 3D, se proyectan los rayos de proyección de todos los pares de puntos disponibles y se toma el par que genere rayos de proyección con la menor distancia entre sí.

6.3. Reconstrucción 3D y validación

Luego de encontrar la mejor asociación de puntos se procede a reconstruir y validar la misma. Si se encuentra al menos un punto en otra cámara que valide la reconstrucción se asume que ésta es correcta, se retira a la pareja que genera la reconstrucción así como también a los puntos que lograron validarla y se itera nuevamente repitiendo el proceso con la siguiente mejor pareja asociada entre dos cámaras.

La información que contiene un punto en una retina se mapea en el espacio 3D sobre el rayo de proyección que contiene a dicho punto y al centro de la cámara correspondiente. Supongamos que se tienen los rayos de proyección en el espacio 3D de todos los puntos contenidos en las retinas y que x_1 es un punto en la cámara 1 y x_2 es un punto en la cámara 2 que se encuentran asociados y reconstruyen al punto X_{12}. El algoritmo implementado asume que un punto en una cámara, valida a X_{12} si junto a x_1 reconstruye un punto 3D que se encuentra dentro de la esfera $B(X_{12}, \delta)$ de centro X_{12} y radio δ, donde δ es un cierto valor umbral.

Idealmente X_{12} se genera al interceptar los rayos de proyección de los puntos x_1 y x_2, pero debido a incertidumbres en la detección de marcadores o la calibración, comúnmente los rayos se van a cruzar. La reconstrucción, se estima como el punto del espacio de menor distancia a ambos rayos.

6.4. Actualizar asociaciones

Del bloque anterior se tiene, un punto X reconstruido y sus correspondientes proyecciones en cada una de las cámaras. Dichas proyecciones no deben ser consideradas nuevamente, por tanto estos puntos 2D que reconstruyen X, no se consideran como puntos válidos en las siguientes iteraciones.

Finalmente el proceso iterativo se detiene cuando no hay más marcadores para reconstruir, lo cual implica que se cumple alguna de las siguientes condiciones:

- el número de marcadores reconstruidos es igual al número de marcadores que tiene colocada la persona, o igual al número máximo de marcadores reconstruidos que se haya indicado.
- No existen puntos 2D válidos tal que pueda establecerse una asociación entre puntos de distintas vistas.

7. Seguimiento

El seguimiento de trayectoria se realiza sobre una ventana deslizante de tres a cuatro cuadros donde se enlazan los puntos reconstruidos de manera de mantener un movimiento lo mas suave posible.

Esta metodología fue la utilizada por Herda [3] en su trabajo basándose en los estudios de Malik, Drako, Papantoniou [6] .

7.1. Algoritmo

Sea una trayectoria de un marcador enlazada hasta el instante [f] la cual desea buscarse su próximo punto en [f+1], el movimiento entre [f-1] y [f] es prolongado para establecer un centro de búsqueda y encontrar el punto reconstruido que mejor continúa la trayectoria como se muestra en la Figura 4 .

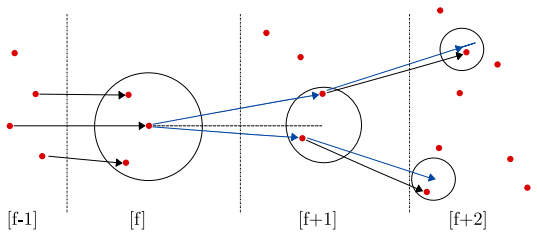


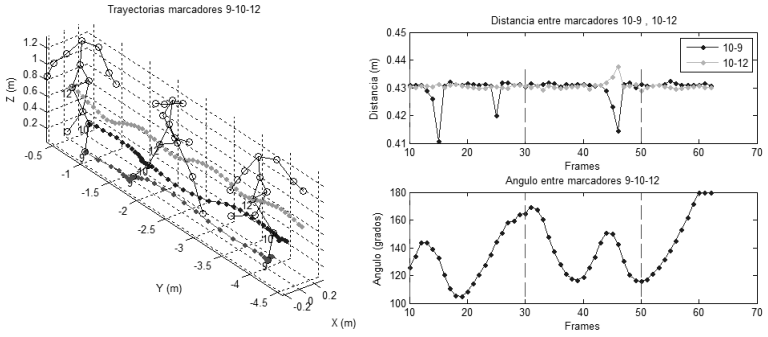
Figura 4: Seguimiento en cuatro cuadros, siendo [f] el cuadro actual que queremos seguir en [f+1]. (Fuente Human movement science 20(3), 313–341 [3]) .

Se presentan tres posibles casos al buscar puntos reconstruidos:

- Si solo se encuentra un punto reconstruido se agrega a la trayectoria para el cuadro [f+1], buscando el mas cercano a la estimación calculada como aquella que mejor se aproxima a una trayectoria de tres puntos con aceleración mínima
- En el caso de encontrar mas de un punto cada posible candidato es evaluado para realizar una segunda estimación hacia [f+2] de forma que la aceleración entre [f-1], [f] y el candidato en [f+1] sea la misma que entre [f], el candidato en [f+1] y la estimación en [f+2]. Luego de todos los posible caminos en cuatro cuadros, se elige el de menor variación de aceleración.
- Si no se encuentra ningún punto, se procede a aumentar de forma limitada el radio de búsqueda en [f+1] de forma excepcional. Esto se hace para continuar trayectorias que entran en estado de reposo y el ultimo movimiento conocido es nulo o muy pequeño.

Si una trayectoria queda trunca durante el enlazado, se intenta recuperar prolongando el movimiento en próximas cuadros para encontrar puntos reconstruidos cercanos a las estimaciones y extrapolar los puntos intermedios. Por otro lado, se implementan umbrales para definir limites sobre la aceleración de los enlaces obtenidos y detectar discontinuidades durante el seguimiento.

Con estas medidas implementadas es posible detectar las trayectorias individuales sobre los puntos reconstruidos, detectar de forma simple posibles discontinuidades, y estimar reemplazos en casos de perdidas. La captura mostrada en la Figura 5 corresponde a la marcha y se resalta las trayectorias individuales de puntos de la pierna así como un esqueleto simple generado simplemente para visualizar la evolución entre marcadores



(a) Trayectorias de marcadores de pierna.

(b) Distancia y Angulo entre marcadores de la pierna.

Figura 5: Posibles restricciones en ángulo y distancia, para el caso de la pierna en marcha.

El conjunto de puntos reconstruidos puede ser sometido a otros algoritmos de seguimiento como por Kalman [5] requiriendo la inicialización de modelos, o algoritmos basados en restricciones mas fuertes a las presentadas en este trabajo como podrían ser las distancias relativamente constantes entre marcadores de los miembros y ángulos continuos entre articulaciones pero requieren un estudio considerable de las características de cada sujeto y movimiento a capturar. La Figura 5 muestra algunas posibles restricciones en la marcha sobre los huesos de la pierna.

8. Performance

Fueron utilizadas las métricas establecidas en HumanEva [10] para comparar los conjuntos de datos obtenidos a la salida de cada bloque individual con aquel conjunto de referencia obtenido en el ground truth de base de datos, buscando primero la correspondencia entre puntos y luego la distancia euclidiana (en 2D para cámaras y 3D para espacio reconstruido) entre los puntos de ambos conjuntos.

Las secuencias sintéticas generadas para múltiples situaciones son más complejas que del caso ideal y contemplan problemas presentes en la realidad siendo el principal ejemplo que ciertos marcadores se encuentran ocultos por el propio cuerpo del sujeto a capturar.

El error de detección de los marcadores en cada una de las cámaras no supera un par de pixeles para el caso de cámaras con resolución en imagen de 1600x600. Es posible reducir la resolución de las cámaras hasta 800x300 manteniendo los mismos resultados, pero en resoluciones menores comienza a degradarse la tasa de detección de marcadores en cámaras individuales lo cual perjudica la siguiente etapa de reconstrucción. Fueron realizadas pruebas inyectando ruido en el bloque de detección y su impacto en etapas posteriores encontrando que se puede trabajar con un poco mas de ruido (no mas de tres pixeles de error) sin comprometer significativamente el error final.

Si las cámaras se encuentran en las condiciones de error anteriores, una cobertura total con 17 cámaras rodeando el movimiento a capturar permite reconstruir todas las trayectorias con un error por debajo del centímetro con errores máximos cercanos a tres centímetros. Los resultados se mantienen dentro de estos limites si se reduce a ocho cámaras colocando pares de cámaras en cada esquina del espacio de captura.

Reducir aún más el conjunto de cámaras sin degradar la performance es posible con ciertas condiciones: un conjunto de seis cámaras colocadas en dos costados de a tres, y cada costado tiene dos cámaras próximas con una tercera mas alejada. De esta manera se tiene las condiciones mínimas por las cuales se estableció la

reconstrucción, tener dos cámaras para reconstruir y una tercera para confirmar.

Este caso limite de reconstrucción muestra comportamiento global similares a los anteriores, pero presenta degradaciones puntuales para ciertas trayectorias de marcadores que superan el error promedio y deberían ser reparadas con medidas avanzadas.

9. Conclusiones

The conclusion goes here.

Referencias

1. cgspeed. http://www.cgspeed.com/, accedido 6-12-2014
2. Cvblob-blob library for opencv. https://code.google.com/p/cvblob/, accedido 29-11-2014
3. Herda, L., Fua, P., Plänkers, R., Boulic, R., Thalmann, D.: Using skeleton-based tracking to increase the reliability of optical motion capture. Human movement science 20(3), 313–341 (2001)
4. Hu, M.K.: Visual pattern recognition by moment invariants. Information Theory, IRE Transactions on 8(2), 179–187 (1962)
5. Kalman, R.E.: A new approach to linear filtering and prediction problems. Journal of Fluids Engineering 82(1), 35–45 (1960)
6. Malik, N., Dracos, T., Papantoniou, D.: Particle tracking in three-dimensional turbulent flows - part ii: Particle tracking. Experiments in Fluids 15, 297–294 (1993)
7. Medioni, G., Kang, S.B.: Emerging topics in computer vision. Prentice Hall PTR (2004)
8. Opencv. http://opencv.org/, accedido 29-11-2014
9. Otsu, N.: A threshold selection method from gray-level histograms. Automatica 11(285-296), 23–27 (1975)
10. Sigal, L., Balan, A.O., Black, M.J.: Humaneva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. International journal of computer vision 87(1-2), 4–27 (2010)
11. Svoboda, T., Martinec, D., Pajdla, T.: A convenient multi-camera self-calibration for virtual environments. PRESENCE: Teleoperators and Virtual Environments 14(4), 407–422 (August 2005)
12. Warren, M., McKinnon, D., Upcroft, B.: Online Calibration of Stereo Rigs for Long-Term Autonomy. In: International Conference on Robotics and Automation (ICRA). Karlsruhe (2013)
13. Wooldridge, D.: bvhacker: The free bvh file editing tool. http://davedub.co.uk/bvhacker/, accedido 30-11-2014