Lenguajes de Programación: Trabajo Práctico I

M. Sc. Saúl Calderón Ramírez
Instituto Tecnológico de Costa Rica,
Escuela de Ingeniería en Computación,
PAttern Recognition and MAchine Learning Group (PARMA-Group)

28 de julio de 2016

El presente proyecto introduce el uso de lenguajes imperativos de alto nivel como MATLAB y Octave. Debe ser desarrollado en grupos de tres personas.

Fecha de entrega: 18 de agosto.

1. Introducción y motivación

El análisis automatizado de videos se ha visto estimulado con la masificación de internet, las plataformas de computación de alto rendimiento a bajo costo, y el diseño de nuevos algoritmos para su procesamiento eficiente. Una aplicación del análisis automatizado de videos es la obtención de datos de alto nivel a partir de videos digitales de futbol, estos datos corresponden al uso de estrategias y tácticas por cada uno de los equipos de futbol. El sistema ACE, actualmente desarrollado en el PRIS-Lab www.pris.eie.ucr.ac.cr, de la Universidad de Costa Rica tiene por objetivo analizar de manera automática videos de futbol. Sus etapas van desde la identificación de escenas con información útil en el video, la segmentación y rastreo de jugadores, hasta el análisis automático de los recorridos y comportamiento de los jugadores.

En el presente proyecto, se propone el desarrollo de la etapa de segmentación de jugadores, el cual se detalla a continuación. [1].

2. Algoritmo a implementar

El algoritmo a implementar realiza la «segmentación de jugadores» y se ilustra en la Figura 1. La región o «blob» de cada jugador en la imagen se define como $R_{i,t}$, correspondiente al jugador i en el cuadro del vídeo U_t (donde t define el número de cuadro) . La segmentación de los jugadores en un video $\mathcal{V} = \langle U_1, U_2, \ldots, U_T \rangle$, compuesto por T cuadros resulta entonces en un con-

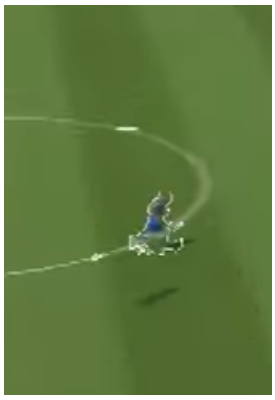

Figura 1: Segmentación de jugadores.

junto de regiones $\mathcal{R} = \langle \mathbb{R}_1, \mathbb{R}_2, \ldots, \mathbb{R}_T \rangle$, donde $\mathbb{R}_t$ se refiere al subconjunto de regiones de los L jugadores presentes en el cuadro t: $\mathbb{R}_t = \langle R_{1,t}, R_{2,t}, \ldots, R_{L,t} \rangle$.

2.1. Algoritmo de segmentación en detalle

El algoritmo de segmentación propuesto usa la información conocida de antenamo relacionada con el hecho de que los jugadores se encuentran siempre rodeados de pixeles verdes (campo de juego). Es por esto que el algoritmo se subdivide en dos etapas no necesariamente secuenciales:

- Detección del campo de juego o construcción de la «máscara del campo de juego» F_t para cada cuadro del vídeo U_t.
- Detección de las regiones candidatas de jugadores (tomando en cuenta al público) $C_{j,t}$ para cada cuadro del vídeo U_t, contenidas en la «máscara de candidatos a jugadores» P_t.

Para obtener las regiones correspondientes a los jugadores (blobs de personas dentro del campo de juego) se realiza un AND entre las máscaras ($P_t \& F_t$), por lo que entonces $R_{i,t} = C_{j,t} \& F_t$. A continuación se detalla el procedimiento, ilustrado con las funcionalidades en MATLAB correspondientes. La imagen con las regiones correspondientes a los jugadores se presenta en la Figura 5.

2.1.1. Detección del campo de juego

Para detectar el campo de juego, se siguen los siguientes pasos.

1. Convertir la imagen de entrada U_t a una imagen de cromaticidad H_t tomando la capa de cromaticidad H después de usar la función *rgb2hsv.*
2. Encontrar los valores α_{min} y $\alpha_{\text{máx}}$ que definan un rango de «verdosidad», para umbralizar la imagen en ese rango y obtener una máscara binaria de pixeles «verdosos» G_t a partir de H_t .

Figura 2: Imagen original y máscara del campo de juego F_t.

3. Rellenar los hoyos en la máscara G_t, posiblemente correspondientes los jugadores, usando la función *imfill*, resultando en G'_t.

4. Eliminar «regiones espurias» pequeñas (menos del 10 % de la imagen), usando la función *bwareaopen*, resultando en la máscara G''_t.

    a) Posiblemente queden hoyos no rellenados desde la máscara G_t (usualmente asociados a pequeñas regiones cerca de los bordes), por lo que se puede calcular el complemento de la máscara G''_t con *imcomplement* y eliminar de nuevo regiones pequeñas con *bwareaopen.* Calcular de nuevo el complemento, almacenando el resultado en G'''_t

5. Eliminar el logo del «marcador del partido», resultando en la máscara F_t.

2.1.2. Detección de regiones candidatas a jugadores

Para obtener las M regiones candidatas a corresponder a jugadores $C_{j,t}$ con $j = 1, \ldots, M$ del cuadro U_t el algortimo propuesto consiste en:

1. Convertir la imagen de entrada U_t a una imagen de cromaticidad H_t tomando la capa de cromaticidad H después de usar la función *rgb2hsv*. Se aconseja normalizar de 0 a 255 la imagen.

2. Calcular la imagen de *varianza local* S_t usando como entrada la imagen H_t. En MATLAB ello se puede realizar con la función *stdfilt* (a cuyo resultado se le debe calcular el cuadrado a cada elemento), **sin embargo, para este trabajo práctico debe implementarla manualmente siguiendo los siguientes pasos**:

   a) La varianza local, calcula, para una imagen U de $A \times B$ pixeles la varianza para toda ventana de $C \times D$ pixeles, de la siguiente forma:

   $$S(x,y) = \frac{1}{C\,D} \sum_{i=x-\Delta_1}^{x+\Delta_1} \sum_{j=y-\Delta_2}^{y+\Delta_2} \left(U(i,j) - \mu(x,y)\right)^2$$

   donde $\mu(x,y)$ se define como el valor medio en la ventana de $C \times D$ pixeles centrada en (x,y), con $\Delta_1 = \frac{C-1}{2}$ y $\Delta_2 = \frac{D-1}{2}$:

   $$\mu(x,y) = \frac{1}{C\,D} \sum_{i=x-\Delta_1}^{x+\Delta_1} \sum_{j=y-\Delta_2}^{y+\Delta_2} U(i,j)$$

   b) Si la imagen H_t fue normalizada de 0 a 255, truncar la imagen de varianza local S_t a 255, de modo que $S_t(x,y) \leq 255$. El resultado se muestra en la Figura 3.

3. Calcular el umbral óptimo τ para la imagen S_t, usando el algoritmo de Otsu o Kittler de umbralización por máxima verosimilitud, implementado en la función *graythresh* de MATLAB.

   a) Aplicar el umbral con la función *im2bw* y rellenar los hoyos con *imfill* obteniendo la máscara P_t con todas las regiones candidatas $C_{j,t}$, mostrada en la Figura 4.

3. Implementación

La implementación del presente proyecto debe realizarse en el lenguaje Octave. Se deben seguir estandares de documentación interna, modularidad, pero sobre todo, **utilizar al máximo las facilidades que provee el lenguaje**. **El programa debe guardar un video con el resultado de la segmentación.** Además

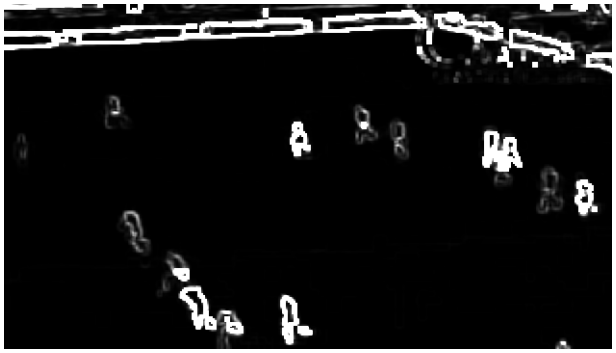

Figura 3: Resultado el cálculo de la varianza local y su truncamiento.

Figura 4: Máscara P_t.

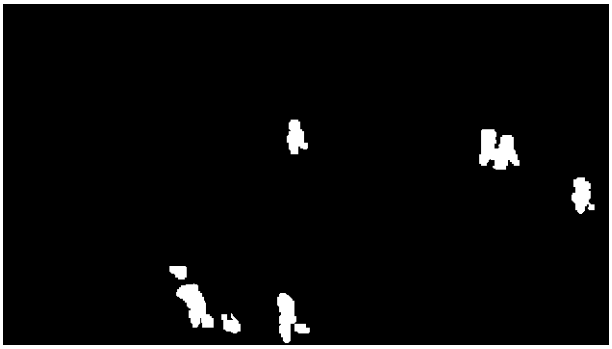

Figura 5: Resultado final de la segmentación espacial.

la documentación interna debe seguir el formato especificado en la carta al estudiante.

La documentación final del proyecto debe incluir todos los estándares implementados a lo largo del curso, incluyendo los realizados en tareas, de manera consistente. **Por cada paso del algoritmo, debe agregar una figura, ilustrando los resultados parciales.** Es el obligatorio el uso de LaTeX o algún editor basado en esa tecnología como LyX para la documentación del proyecto. Para facilitar la edición colaborativa, se recomienda el uso de la herramienta *Overleaf*.

Referencias

[1] Francisco Siles Canales. Ace-football, analysing football from tv broadcasting. *24th German Soccer Conference DVS – Fusball in Lehre und Forshung*, 2013.