Universidad de Buenos Aires
Facultad de Ciencias Exactas y Naturales

Tesis de Licenciatura

Inicialización y mantenimiento del modelo del fondo con fuerte oclusión

por

Francisco Facioni

L.U.: 004/04

fran6co@gmail.com

Director: Marta Mejail

Co-director: Julio Jacobo

Resumen

El proceso de segmentación es parte fundamental en muchos algoritmos de visión en robótica ya que permite separar los objetos del fondo en el cual se encuentran.

Existen muchos métodos de segmentación y se utilizan para analizar automáticamente vídeos de diversas naturalezas como partidos de futbol, tráfico, seguridad y vigilancia.

Hoy en dia hay otro campo que esta tomando mucha fuerza: la realidad aumentada. Al tener dispositivos móviles de alta capacidad computacional y, en futuro muy cercano, anteojos con cámara y pantalla (Google Glass) va a ser cada vez más necesario tener métodos para rápidamente y precisamente seleccionar los objetos a analizar.

Uno de los grandes problemas de la mayoría de los métodos de segmentación es que necesitan tener una imagen del fondo sin objetos para poder segmentar los objetos. Para que sean totalmente autónomos son necesarios métodos de inicialización de fondo.

En este trabajo se analizaron dos propuestas de inicialización de fondo basados en regiones. En el primer trabajo se proponen regiones solapantes y selección de la región correcta vía un método basado en graph-cut. En el segundo se utilizan regiones no solapantes y descomposición DCT con campos de Markov. A diferencia de los trabajos actuales en esta tesis se propone una mejora utilizando el índice SSIM para mejorar la robustez de los métodos.

Después de probar distintas variantes de los métodos analizados, los resultados obtenidos demuestran que el índice SSIM es la alternativa más robusta.

Índice general

1. Introducción

1.1. Segmentación de fondo

Segmentar el fondo de una imagen es el proceso por el cual se separan los objetos del fondo. Así dicho resulta una tarea fácil, ya que al ser humano esta tarea es algo que hace automáticamente todo el tiempo. Pero esconde una complejidad muy grande que empieza al momento de definir los conceptos de objeto y fondo.

La distinción de objeto y fondo es relativa al contexto que se tome y de las circunstancias. Por ejemplo para un video de seguridad se hace foco en los objetos en movimiento o que estuvieron en movimiento, con lo cual una maceta se podría clasificar como parte del fondo pero si algún objeto interactúa con ella (si la mueven, por ejemplo) podría pasar a ser un objeto.

Por más que se cuente con una definición de fondo y de objetos, su traducción a una implementación operativa es lejos de directa y suele contener ambigüedades, por ejemplo cambios bruscos de iluminación, movimiento de la cámara u objetos transparentes y/o camuflados generan casos donde objetos se toman como fondo y/o parte del fondo como objetos.

Para poder conseguir los mismos resultados que un humano sería necesario tener un modelo del mundo equivalente al de un humano, llevando la problemática de la segmentación al mundo de la inteligencia artificial. Por suerte existen atajos que logran resultados aproximados con niveles de precisión suficientes para muchas tareas. Por ejemplo la segmentación de objetos en una secuencia de imágenes es fundamental en campos muy diversos como seguridad, producción audiovisual, monitoreo de tráfico automotor, compresión de video [7] y robótica.

1.2. El problema de la inicialización del modelo de fondo

En general, se pueden separar en dos partes los métodos de segmentación: modelización del fondo y mantenimiento del modelo de fondo. Existe gran cantidad de bibliografía sobre segmentación de fondo [8-10], y mucha de ella da por descontado que se dispone de al menos una imagen donde se muestra solamente el fondo de la cual construir el modelo del fondo. Por lo general se simplifica a tal punto de suponer que los cuadros al principio de la secuencia de video representan el fondo. En otros casos es necesaria algún tipo de intervención humana en la selección de dichos cuadros.

Al proveer una imagen inicial del fondo implícitamente se está proveyendo una definición de lo que es fondo y de lo que son objetos. Pero en casos donde la oclusión del fondo es constante, como cámaras apuntadas a tráfico vehicular o a zonas muy activas, proveer de esa imagen inicial del fondo resulta poco práctico.

1.3. Objetivo del trabajo

Luego de analizar el estado del arte de la inicialización de fondo se seleccionaron los trabajos [2] y [3] ya que, según los autores, sus resultados eran muy superiores a métodos normalmente usados. En especial [3] provee una definición de fondo y de objetos muy concisa junto a las suposiciones por las cuales se rige el método, de esta manera se obtiene un marco teórico robusto desde el cual analizar los diversos métodos.

El objetivo de este trabajo es generar un nuevo método de inicialización de fondo que combine la performance de [2] y el marco teórico de [3]. El trabajo [2] propone un método con resultados comparables al de [3] sacrificando el marco teórico, al tomar un camino puramente práctico, para lograr una performance cercana al real-time.

1.4. Estructura del trabajo

En el capítulo 2 se presenta un detallado informe sobre los métodos utilizados. En el capítulo 3 se analiza el índice SSIM y como se aplica para generar un nuevo método de inicialización. En el capítulo 4, se realiza un análisis de los resultados obtenidos en pos de evaluar la variante introducida. Por último, en el capítulo 5 se concluye el presente informe y se plantea el trabajo a futuro. Por último un apéndice que describe en más detalle algunas técnicas utilizadas.

2. Inicialización de fondo basado en regiones

El problema de la inicialización de fondo se puede definir como:

Dada una secuencia de imágenes obtenidas con una cámara estática, obtener una imagen del fondo sin ningún objeto en movimiento, aunque nunca se muestre el fondo completamente en un cuadro.

Dicho de otra manera, el problema de inicialización es tomar un pixel del primer cuadro de la secuencia e imaginar la línea temporal que lo atraviesa y pasa por toda la secuencia de cuadros (*fig. 1*). Los pixeles por los cuales esta línea imaginaria pasa pueden ser parte del fondo o de un objeto, por lo tanto el problema de inicializar se puede pensar como tomar el pixel correcto para cada línea temporal.

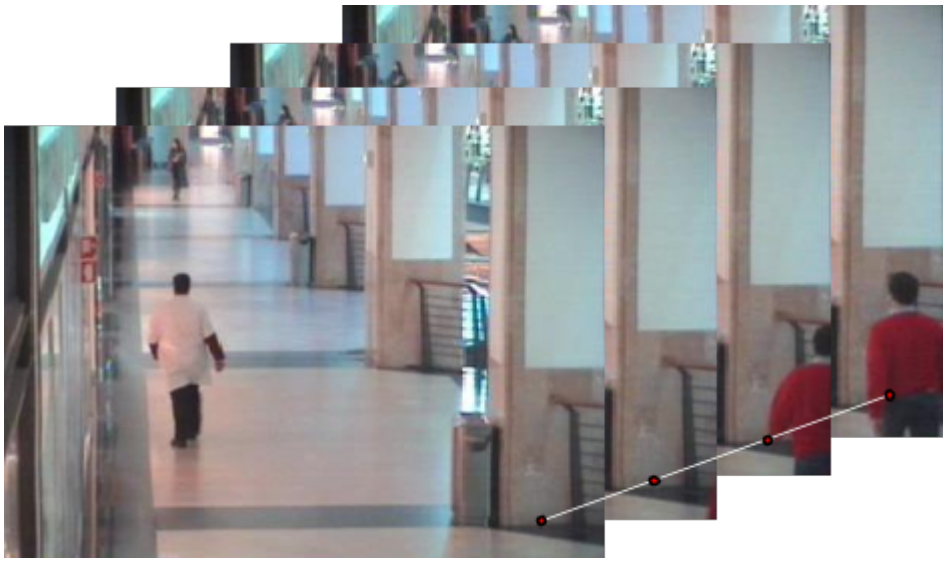


fig. 1: Ejemplo de misma posición en distintos frames, en los primeros dos se clasifican como fondo y los últimos dos no

Las suposiciones que guían las técnicas a analizar se

pueden resumir en:

- el fondo es estático y los objetos están en movimiento
- el fondo se muestra al menos en un cuadro en cada región del vídeo (se definirá el significado de región más adelante)
- los objetos introducen una discontinuidad con respecto al fondo

La mayoría de las técnicas reconstruyen el fondo analizando pixel por pixel independientemente (ej.: Mixture of Gaussian [1]) aunque esté demostrado que técnicas que aprovechan la correlación espacial de los pixeles entre sí logran mejores resultados [11-13].

Uno de los puntos donde los trabajos recientes ([2] y [3]) que se analizaron en esta tesis se diferencian de los precursores ([8]-[10]) es en la suposición de la estabilidad temporal, donde se da más importancia a pixeles, o regiones, que aparecen con más frecuencia en la secuencia. El caso donde esta suposición claramente falla es en el caso de objetos estáticos que aparecen en gran parte de la secuencia del video. Este caso se lo denomina “persona que duerme” o “persona que se despierta” (*fig. 2*), donde una persona aparece durmiendo en la mayoría de los cuadros del video (objeto estático) y en un momento se despierta y abandona la escena. Si se supone la estabilidad temporal como parámetro para definir el fondo, la persona durmiendo se tomaría como parte del fondo y no el fondo real que se muestra al despertarse la persona.

fig. 2: En el rectángulo rojo se puede ver como el televisor está presente por la mayor parte del video y en los últimos cuadros lo mueven

Los trabajos [2] y [3] están entre los que mejor capturan las dificultades de inicializar el fondo y ofrecen una solución consistente con lo dicho anteriormente. En este capítulo se analizan sus similitudes y diferencias.

En la primera sección se trata de la estructura general de los algoritmos propuestos en los dos trabajos y en las siguientes secciones se trata paso por paso de los algoritmos.

2.1. Estructura general de los algoritmos

Si se toma un video como una sucesión de F imágenes, se puede definir una región (o ventana) W de dimensión $NxNxF$ del video como el conjunto de píxeles de cada cuadro donde sus coordenadas espaciales están contenidas en un cuadrado de lado N. Entonces queda definido $W=[W_1...W_k...W_F]$, con parches W_k de dimensión NxN que representan una parte de la imagen en el momento k del video como se muestra en *fig. 3*.

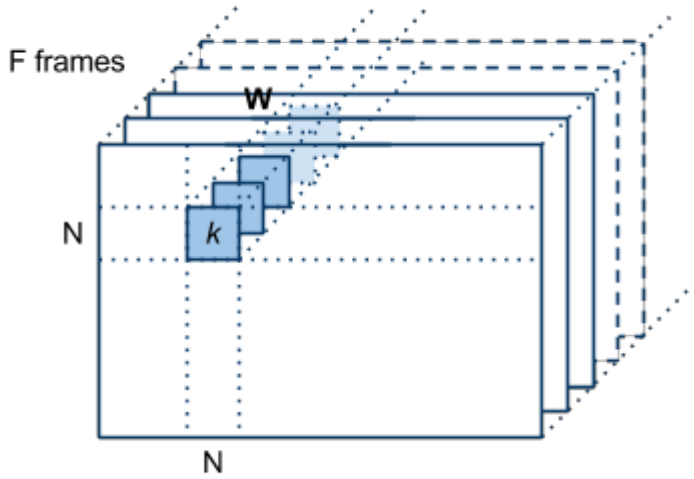


fig. 3: Parche W_k perteneciente a una región (o ventana) W en el orden k-ésimo de la secuencia de F parches de dimensión NxN

Utilizando estos elementos, los algoritmos de inicialización estudiados se pueden resumir en los siguientes pasos:

1. Separar en regiones el vídeo
2. Por cada región *W* agrupar los parches W_k similares según algún criterio
3. Establecer como fondo todas las regiones que produjeron sólo un grupo de parches en el paso anterior, es decir todos los parches W_k de la ventana *W* son similares (o sea, que no cambiaron durante todo el video)
4. Reconstrucción del fondo
    a. Seleccionar una región para la cual no se definió todavía un grupo como fondo
    b. Utilizar las regiones vecinas para determinar qué grupo de parches, de la región seleccionada, pertenece al fondo
    c. Volver al punto a hasta que no queden regiones sin un grupo de parches establecido como el fondo
5. Corrección del fondo

A continuación se analizan paso por paso los algoritmos propuestos.

2.2. Separación en regiones

La separación del video en regiones permite tomar en cuenta la relación espacial que hay entre los píxeles cercanos, a diferencia de métodos que procesan el video pixel por pixel.

El tamaño de los parches es arbitrario pero se tienen que cumplir los siguientes criterios:

- tiene que ser lo suficientemente grande como para aprovechar la relación espacial entre píxeles cercanos
- tiene que ser lo suficientemente chico como para que al menos dos parches pertenezcan enteramente al fondo

El segundo punto no siempre es factible, ya que se puede tener el caso donde un objeto oscila siempre en el mismo lugar sin mostrar una parte del fondo (*fig. 4*). Videos que presenten este tipo de objetos van a producir fondos que contengan al objeto que oscila o parte del mismo.

fig. 4: En el rectángulo rojo se puede ver como la silla siempre ocluye el fondo

La forma en la cual se distribuyen espacialmente las ventanas determina la relación de vecindad entre las mismas. Básicamente hay dos tipos de distribución: solapantes y no solapantes.

Ventanas no solapantes simplemente son ventanas adyacentes unas a las otras que cubren la totalidad del vídeo. Por

cada ventana *W* se define su vecindad como el conjunto de ventanas (*V* en el gráfico *fig. 5*) adyacentes horizontalmente, verticalmente y diagonalmente.

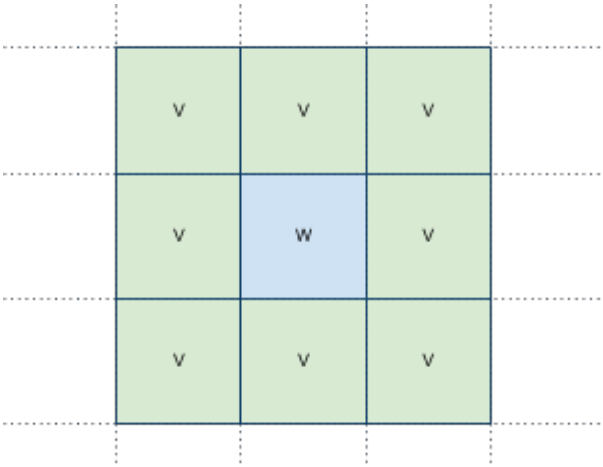


fig. 5: Ventana W y sus 8 vecinos (V en el gráfico)

Las regiones solapantes se distribuyen de tal manera que cada ventana *W* tenga 4 ventanas vecinas (*V* en *fig. 6*) con las cuales comparte la mitad de los pixeles. O sea, si tenemos una ventana de *NxN*, la ventana a la izquierda va a compartir *NxN/2* pixeles, y lo mismo para las otras tres direcciones (arriba, abajo y a la derecha).

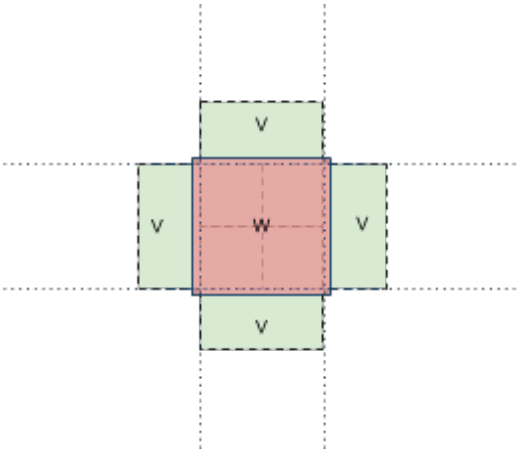


fig. 6: La ventana W y sus 4 vecinos (V en el gráfico)

La distribución a utilizar depende de los criterios que se

utilizan para analizar ventana por ventana con respecto a sus vecinos.

Cada ventana se procesa de tal manera de que los parches similares (según uno de los criterios de similitud definidos más adelante) pertenezcan al mismo grupo. La clusterización por ventana permite eliminar la influencia del supuesto de la estabilidad temporal, un grupo con muchos parches tiene igual de importancia a uno con pocos, y reduce el costo de procesamiento al trabajar con grupos en lugar de parches individuales.

A continuación se analiza las distintas técnicas de clusterización de parches.

2.3. Agrupación de parches

Cada región (o ventana) está compuesta por tantos parches como frames hay en el vídeo. Por lo general, todo video presenta una gran redundancia temporal (explotada por muchos métodos de compresión) donde muchos parches son similares entre sí, conviene agrupar los parches por similitud, para reducir la cantidad de parches redundantes y reducir la complejidad espacial y temporal de los algoritmos.

Una vez agrupados se crea un representante por cada grupo (*fig. 7*), reduciendo cada región a un conjunto de representantes en lugar de grupos de parches. Se puede utilizar la media o la mediana para la generación del representante de cada grupo. En los dos métodos propuestos se prefiere la media, ya que se puede calcular iterativamente a medida que nuevos frames se van incorporando sin tener que guardar todos los parches, a diferencia de la mediana que tiene una complejidad espacial más elevada.

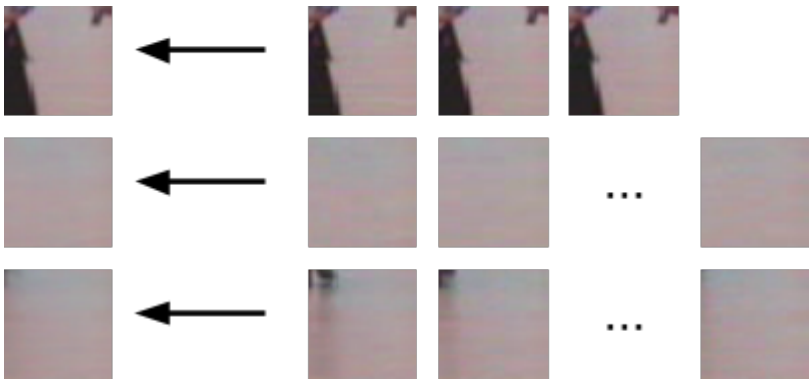


fig. 7: Varias imagenes son clusterizadas por parecido y se genera un representante. El último cluster muestra cómo pequeños cambios pueden ser considerados parecidos.

Para determinar si dos parches son similares entre sí se proponen dos medidas de similitud entre dos parches:

- la distancia de Mahalanobis entre los parches vectorizados (ver [3])
- el coeficiente de correlación y la media de la diferencia absoluta entre los parches vectorizados (ver [2])

Para vectorizar un parche se concatenan las componentes de cada pixel de la siguiente manera:

$$\bar{x}_W = (r_0, g_0, b_0, ..., r_i, g_i, b_i, ..., r_n, g_n, b_n)$$

donde $\bar{x}_W$ es el vector resultante para el parche (o representante) perteneciente a la ventana W y (r, g, b) son las componentes de cada pixel. n es la cantidad de pixeles en un parche (*NxN* salvo que pertenezca a un borde, en tal caso van a ser menos pixeles).

A continuación analizamos los criterios de similitud utilizados: Distancia de Mahalanobis y la media de la diferencia absoluta.

a. Distancia de Mahalanobis

Si tomamos la representación vectorial de las imágenes una posible métrica de diferencia entre dos imágenes es la distancia euclidiana.

$$d(\bar{x}_W, \bar{y}_W) = \sqrt{(\bar{x}_W - \bar{y}_W)^T \ . \ (\bar{x}_W - \bar{y}_W)}$$

donde $\bar{x}$ y $\bar{y}$ son dos parches vectorizados pertenecientes a la ventana W. La distancia es 0 para dos imágenes iguales y crece a medida que su diferencia pixel por pixel aumenta. El problema con esta métrica es que las imágenes tienen ruido con lo cual se corre el riesgo que se tomen como distintas imágenes que en realidad son iguales.

Con el fin de tener en cuenta el ruido en la métrica el trabajo [3] propone la distancia de Mahalanobis (*fig. 8*), que se define de la siguiente manera:

$$d(\bar{x}_W, \bar{y}_W) = \sqrt{(\bar{x}_W - \bar{y}_W)^T S^{-1} (\bar{x}_W - \bar{y}_W)}$$

donde $\bar{x}$ y $\bar{y}$ son dos parches vectorizados pertenecientes a la ventana W, S es la matriz de covarianza que determina la covarianza entre cada componente de los dos vectores. Se puede asumir que los videos se ven afectados por un ruido gaussiano con distribución $N(0, \sigma^2)$, lo que implica que $\bar{x}_W - \bar{y}_W$ tiene distribución $N(0, 2\sigma^2)$. Esto hace que la matriz de covarianza se reduzca a una matriz diagonal con $2\sigma^2$ como valor. Entonces la distancia de Mahalanobis se puede reescribir, eliminando la raíz, como la suma de las distancias al cuadrado:

$$SSD(\bar{x}_W, \bar{y}_W) = (\bar{x}_W - \bar{y}_W)^T I / 2\sigma^2 (\bar{x}_W - \bar{y}_W)$$

Se puede ver SSD tiene distribución Chi cuadrado con $3N^2$ grados de libertad (3 componentes y *NxN* pixeles), lo cual nos permite definir un punto de corte *t* en base a un índice de confianza α utilizando $\chi^{-1}_{3N}(\alpha)$. Definido un umbral *t*, dos parches se dicen que son similares si el SSD es menor a *t* y distintos si es mayor.

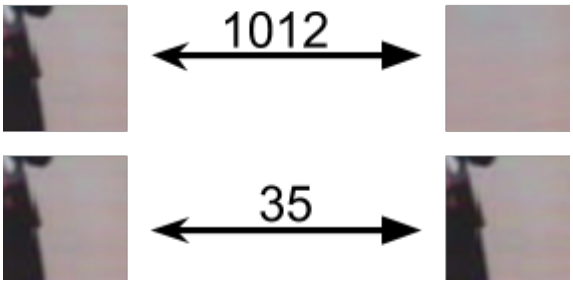


fig 8: Muestra la distancia Mahalanobis entre dos pares de imágenes. Se puede ver como imágenes parecidas tienen una distancia menor que imágenes diferentes

b. Media de la diferencia absoluta

En la distancia definida en el trabajo [2] se toman dos criterios para determinar si dos parches son similares:

1. $\frac{1}{N^2}\sum_{n=0}^{N^2-1}\left|\bar{x}_W(n) - \bar{y}_W(n)\right| < T_1$
2. $(\bar{x}_W - \mu_{\bar{x}_W})^T(\bar{y}_W - \mu_{\bar{y}_W})/(\sigma_{\bar{x}_W}\sigma_{\bar{y}_W}) > T_2$

$\mu_{\bar{x}_W}$ y $\sigma_{\bar{x}_W}$ son la media y la desviación estándar del parche $\bar{x}_W$, respectivamente.

La *desigualdad 1* es la media de la diferencia absoluta (MeanAD) entre las dos imágenes. Cumple el mismo rol que la distancia euclidiana definida anteriormente, pero el ruido en las imágenes está representado por el punto de corte T_1. Podemos estimar T_1 de la siguiente manera: sea L y N_b la cantidad de cuadros a considerar y la cantidad de ventanas, respectivamente

1. Para cada ventana se calcula el MeanAD entre los parches sucesivos, obteniendo $(L-1)N_b$ valores
2. Se ordenan los valores de manera ascendente y se dividen en cuartiles
3. Se calcula la media y la desviación standard del cuartil Q_2 y se estima $T_1 \simeq (\mu_{Q_2} + 2\sigma_{Q_2})$

En el primer cuartil se encuentran todos los valores cercanos o iguales a 0 (donde no hay ruido), y en el tercer cuartil contiene los valores más altos, que son debidos a movimientos

de los objetos. Por lo tanto el cuartil del medio es el que contiene los valores consistentes con el ruido en el video.

Existen casos donde la diferencia entre dos imágenes es consistente con el ruido pero un humano las tomaría como imágenes distintas (*fig. 9*). Por ejemplo, dos imágenes del fondo pero una con una parte pequeña de un objeto.

fig. 9: Estos dos parches cumplen con los criterios de la media absoluta pero claramente son distintos, ya que el de la izquierda contiene una esquina más oscura

Para resolver este problema se introduce la ecuación 2 que determina el coeficiente de correlación entre los dos parches. El umbral T_2 es determinado empíricamente (0.8 en el paper) de tal manera que dos parches distintos no se consideren iguales por el ruido.

Una vez establecido un criterio de similitud, se analizan todos los parches de una ventana (o región) y se agrupan en clusters. En los métodos analizados se utiliza single linked cluster [14] o por representante.

Single linked clustering define que un parche pertenece a un grupo si la similitud entre el parche más similar, al parche a clasificar, de un grupo esta dentro de las condiciones de corte. Esto implica que a medida que se analizan los parches dos grupos distintos se pueden unificar si existe un parche que pertenece a ambos (*fig. 10*).

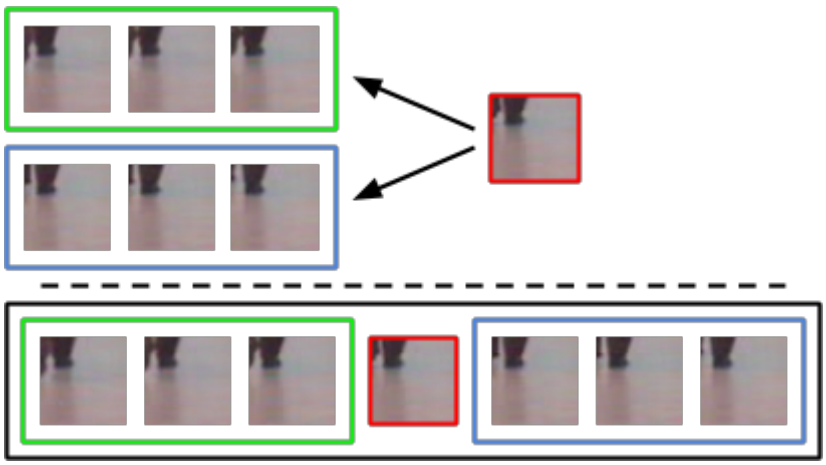


fig. 10: Una imagen cumple con las condiciones de corte para dos clusters (recuadros verde y azul), con lo cual se juntan todos en un único cluster (recuadro negro)

En el segundo método se dice que un parche pertenece a un grupo si este cumple con las condiciones de similitud con el representante (*fig. 12*). Si no se encuentra ningún grupo se crea uno nuevo. En el trabajo analizado no está definido que pasa si un parche pertenece a más de un grupo, pero en la implementación proporcionada toman el primer grupo que se encuentra y no lo agrega al resto (*fig. 11*).

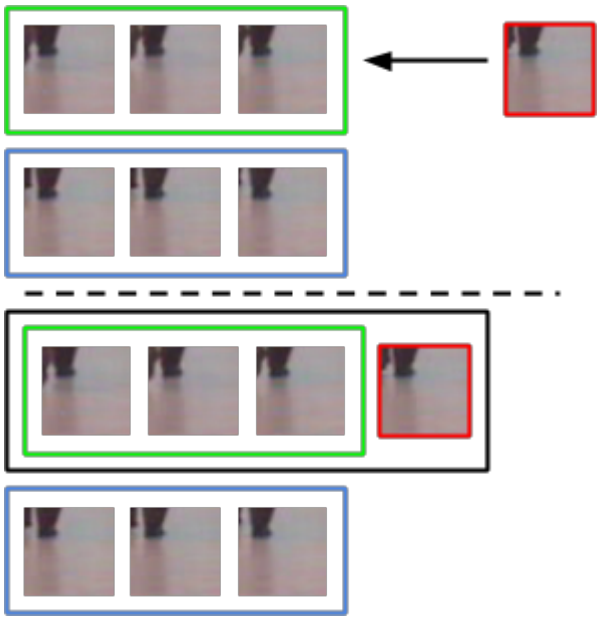


fig. 11: Una imagen cumple con las condiciones de corte del primer cluster (recuadro verde), con lo cual se lo agrega y no se sigue revisando el resto de los clusters

Para calcular los representantes a medida que se crean los grupos (en el caso de single linked cluster se pueden calcular al final, ya que no los necesita) se utiliza la media de la siguiente manera:

$$R_{W,i} \leftarrow (\left|R_{W,i}\right| R_{W,i} + P_W)/(\left|R_{W,i}\right| + 1)$$

$$\left|R_W\right| \leftarrow \left|R_W\right| + 1$$

donde $R_{W,i}$ es el representante del grupo i en la ventana W, $\left|R_{W,i}\right|$ es la cantidad de parches contenidos en el grupo, P_W es un parche de la ventana W.

fig. 12: Una imagen nueva se compara solamente contra el representante del cluster (recuadro azul)

La manera de crear los grupos es fundamental para compensar el ruido al momento del vídeo y para reducir la cantidad de elementos con los cuales se trabaja. También hace que sea independiente de la distribución temporal de los parches, ya que la agrupación no tiene en cuenta el cuando sino solamente la similitud entre los parches.

Los métodos descritos tienen serios problemas al manejar cambios de iluminación, y son muy dependendientes de una buena estimación del ruido.

La distancia de Mahalanobis tiene una definición de corte muy clara, que permite usar conceptos de estadística (grado de confianza 0.95 o 0.99) en lugar de coeficientes arbitrariamente elegidos en base a un set reducido de pruebas. Lo dos métodos tienen una complejidad muy baja $O(N^2)$ lo cual hace que no impacten fuertemente en los tiempos de la generación del fondo.

2.4. Fondo inicial

En este paso del algoritmo se definen las ventanas que van servir como “seed” para reconstruir el fondo. Para poder reconstruir el fondo es necesario establecer un grupo de ventanas iniciales, que permiten poder seleccionar el fondo correcto en sus vecinos.

Un criterio trivial es tomar como fondo todas las regiones donde no se produjo cambio durante todo el video. Se puede ver que las regiones sin cambios, o cambios consistentes con el criterio de similitud elegido, son las que tienen solo un grupo de parches.

Si no existen ventanas estáticas, una alternativa es utilizar una heurística basada en la frecuencia temporal, o sea, se toma

como ventana inicial la que tenga el grupo con mayor cantidad de parches [3]. Obviamente esto lleva a problemas, ya que existen casos donde un objeto aparece estático por casi toda la duración del vídeo y, en los últimos cuadros, se mueve. En ese caso se tomaría erróneamente como fondo el grupo que contiene el objeto.

Si la falta de regiones estáticas es debida a un cambio generalizado en el vídeo (en pocos frames sucesivos), se puede reinicializar el modelo del fondo.

En el trabajo [2] se utiliza una versión refinada de la heurística antes descrita, se toma el grupo con mayor cantidad de parches pero teniendo en cuenta solamente las 4 esquinas. La lógica detrás de tal cambio es que por lo general las cámaras se apuntan de tal manera de que la mayor parte de la actividad es en el centro de la cámara, por lo tanto las esquinas deberían ser menos activas.

2.5. Reconstrucción del fondo

En este paso del algoritmo se reconstruye el fondo en base a las ventanas “seed” del paso anterior.

Una vez establecidas las regiones iniciales se busca entre las ventanas, sin procesar, una que tenga en su vecindad suficientes partes del fondo como para poder analizar sus grupos.

Los criterios para establecer un grupo de parches de una región como fondo se basan en cuán “bien continúan” el fondo conocido hasta el momento. Los métodos analizados presentan distintos criterios para determinar la “buena continuación” del fondo por parte de un representante.

En el trabajo [3] se utilizan conceptos de inpainting y graph-cuts para determinar la mejor continuación dada una vecindad. En cambio en el trabajo [2] se utiliza la descomposición discreta de cosenos para cuantificar la suavidad de la composición resultante de los vecinos con los distintos representantes.

A continuación se analiza cada método por separado.

a. Graph-cuts y “sin costuras”

Utilizando ventanas solapantes, la vecindad de una región

es definida por los 4 vecinos que comparten la mitad del espacio. Alcanza con solo un vecino perteneciente al fondo para determinar el fondo en una región.

De todos los grupos se toman aquellos donde su representante que no presenta “costuras” con respecto a la vecindad. Por costura entendemos la parte compartida con el vecino esté dentro los criterios de similaridad establecidos anteriormente. Se itera por todos los representantes y se eliminan los que tengan alguna parte distinta con algún vecino. Este paso sirve para asegurar que el nuevo fondo sea consistente con los vecinos (*fig. 13*).

fig. 13: La imagen de la izquierda es parte del fondo, para que no forme costuras la imagen, que debería continuar a la derecha, tiene que ser parecida en la zona solapante (marcada en rojo).

Los candidatos remanentes juegan un torneo round-robin (todos juegan contra todos, siempre de a pares, en el peor caso son *L*(L-1)/2* partidos con *L* la cantidad de cuadros del video) donde el ganador es el que menos discontinuidades tiene con respecto a los otros. Para determinar qué candidato tiene menos discontinuidades se utilizan conceptos de graph-cuts e inpainting para cuantificar la integración con los vecinos (tomar nota que no se realiza realmente el cálculo del corte óptimo).

La idea base de graph-cuts es que se puede modelar una imagen como un grafo donde sus nodos son los pixeles y sus aristas conectan todos los pixeles entre sí. Estableciendo como pesos $w_{i,j}$ de las aristas que conectan los pixeles *i* y *j* la siguiente fórmula:

$$w_{i,j} = e^{-(f_i - f_j)^T (2\Lambda)^{-1} (f_i - f_j)}$$

donde $f_i = [x_i, y_i, r_i, g_i, b_i]$ es el vector que contiene la

posición y los componentes de cada pixel y $\Lambda = diag(N^2/16, N^2/16, \sigma^2, \sigma^2, \sigma^2)$ es la matriz diagonal con las componentes de normalización. De esta manera se asigna un peso alto a la arista entre dos pixeles que son muy parecidos y muy cercanos y se reduce a medida que se alejan los pixeles y/o menos se parecen.

Si tomamos la diferencia entre las dos imágenes a comparar, podemos establecer dos conjuntos de coordenadas de tal manera que uno contenga las que no presentan diferencias sustancial entre las dos imágenes y en otro las que sí. Los dos conjuntos definen un corte del grafo (*fig. 14*).

Utilizando la distancia de Mahalanobis pixel por pixel y aprovechando que su distribución es Chi cuadrado, podemos definir el punto de corte para determinar si una diferencia es sustancial o no en base un índice de confianza, los dos conjuntos A y B quedarían definidos como:

$$A = \{(x,y) : \frac{1}{\sigma_{k_1}^2 + \sigma_{k_2}^2} \left|\left| u_{x,y,k_1} - u_{x,y,k_2} \right|\right|^2 < \chi_3^{-1}(\alpha) \}$$

$$B = \{(x,y) : \frac{1}{\sigma_{k_1}^2 + \sigma_{k_2}^2} \left|\left| u_{x,y,k_1} - u_{x,y,k_2} \right|\right|^2 \geq \chi_3^{-1}(\alpha) \}$$

donde σ_k^2 es la varianza para el representante k, $u_{x,y,k}$ es el pixel en la posición x, y para el representante k y $\chi_3^{-1}(\alpha)$ es el punto de corte dando por Chi cuadrado con un índice de confianza α.

El costo del corte queda definido de la siguiente manera

$$cut(A,B) = \sum_{i \in A, j \in B} w_{i,j}$$

donde A y B son los dos conjuntos que conforman el corte.

fig. 14: Dos parches en la posición x,y en los momentos k_1 y k_2 respectivamente y su respectivo corte A, B en blanco y negro

Como la función de costo asigna un mayor valor a pixeles parecidos y cercanos, es fácil ver que el costo del corte va a ser

alto cuando la imagen no presenta discontinuidades entre los dos conjuntos y va a ser bajo cuando hay una discontinuidad (pixeles muy diferentes). Dicho en otras palabras el costo de separar dos regiones diferentes es bajo y el costo de separar dos regiones parecidas es alto. Como una de las suposiciones es que los objetos introducen una discontinuidad con respecto al fondo, con lo cual parches que pertenecen al fondo van a tener un costo más alto con respecto a los que no.

El ganador del torneo round robin es el que obtuvo un mayor costo de corte contra la mayoría de los otros representantes.

b. Suavidad y superbloques

El concepto de suavidad se puede definir como la falta de saltos abruptos en un contexto. Si se aplica la transformación discreta del coseno (DCT) se puede cuantificar la "suavidad" en una imagen por los coeficientes en las altas frecuencias que son los que determinan los bordes o saltos abruptos en la imagen. Más fuertes son los coeficientes en las altas frecuencias más discontinuidades hay en el imagen. Con lo cual se puede utilizar como criterio de mejor continuación el candidato que genere los coeficientes más bajos con respecto a los otros, dada una vecindad.

La vecindad en una configuración de ventanas no solapantes son todas las regiones adyacentes, ya sea horizontalmente y verticalmente como diagonalmente. Se define un superbloque como la ventana W a evaluar más uno de los grupos de 3 vecinos (*V en fig. 15*) en las esquinas, que pertenezcan todos al fondo.

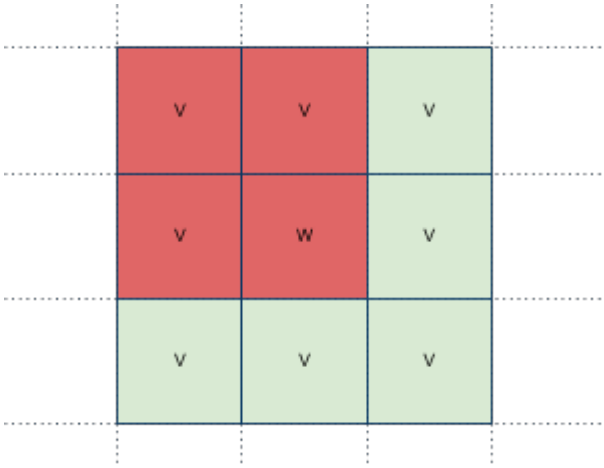


fig. 15: En rojo un superbloque compuesto por la ventana W y 3 vecinos V

De esta manera un superbloque contiene las tres direcciones (horizontal, vertical y diagonal) en las cuales se evalúa la "suavidad" de los candidatos.

Por cada grupo de parches de la región a evaluar, se toma el representante y se realizan los siguientes pasos:

1. Calcular el DCT del superbloque con la región a evaluar en 0 si la media del representante es mayor a 128 y en 255 si es menor (esto es para generar coeficientes lo suficientemente distintos si los valores son cercanos a 0)
2. Se pone en 0 el valor DC de los coeficientes antes evaluados (interesan solo las variaciones espaciales)
3. Se genera el DCT del superbloque pero esta vez con los vecinos en 0 si la media del representante es mayor a 128 y en 255 si es menor
4. Se pone en 0 el valor DC de los coeficientes antes evaluados
5. Combinar los dos grupos de coeficientes y calcular el costo

El costo se calcula de la siguiente manera para cada representante k:

$$cost(k) = \lambda_k \sum_{v=0}^{M-1} \sum_{u=0}^{M-1} \left| C_{v,u} + D_{k,v,u} \right|$$

donde C es el conjunto de los coeficientes de los vecinos, D_k los coeficientes del representante k y $\lambda_k = e^{-\alpha w_k}$ con $\alpha \in [0,1]$ un valor de ajuste y $w_k = W_k / \sum_{i=1}^{S} W_i$ con W_i la cantidad de parches en el grupo i y S la cantidad total de grupos. El coeficiente λ_k ajusta el costo en base a la permanencia temporal de cada grupo.

El representante con el menor costo es el que produce los coeficientes más bajos, con lo cual se toma como parte del fondo (*fig. 16*).

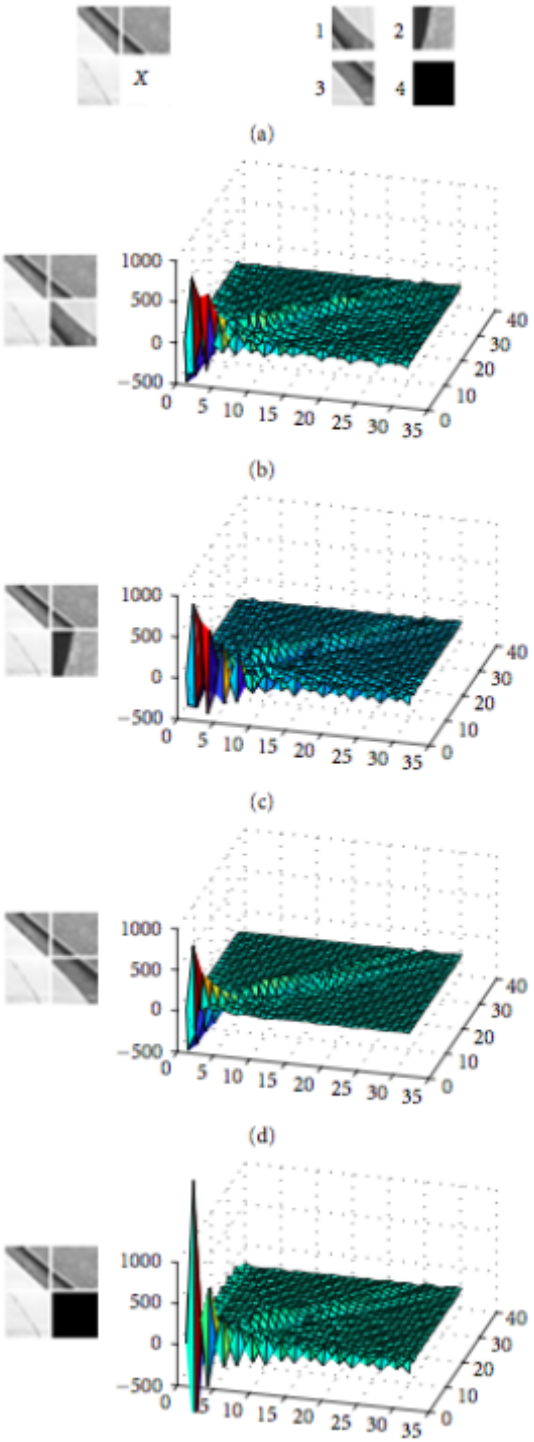


fig. 16: Muestra la discontinuidad introducida por los varios candidatos

Como la evaluación de cada ventana utiliza la vecindad disponible al momento, puede ser que partes del fondo no sean consistentes con la nueva vecindad. Para amortizar los efectos de definir el fondo en base a una vecindad potencialmente incompleta, en las últimas versiones del trabajo [2] se agregó el uso de campos aleatorios Markovianos (MRF) (ver A.1.).

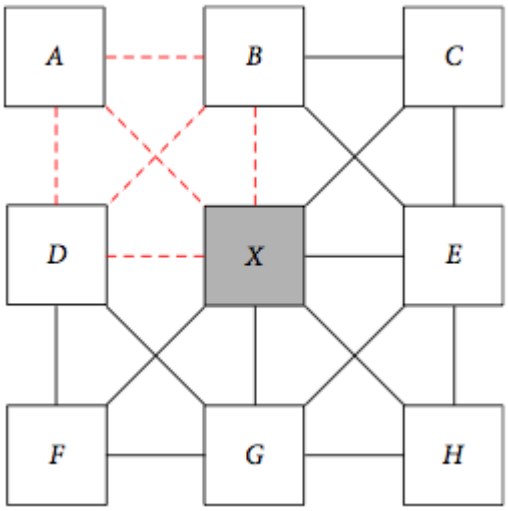


fig. 17: Ejemplo de grafo de dependencias de la vecindad de una ventana. Cada ventana se puede tomar como una variable aleatoria donde los posibles valores son los candidatos a fondo.

El problema se traduce a un MRF generando un grafo cuyos nodos son las ventanas que componen el video y las posibles etiquetas para cada nodo los representantes disponibles. Los nodos están conectados entres sí respetando la relación de vecindad definida para ventanas no solapantes (*fig. 17*). La probabilidad a maximizar es definida por:

$$P(r_k) = l(r_k)p(r_k)$$

donde r_k es el representante *k* del conjunto de los representantes pertenecientes a una ventana *W* y $l(r_k)$ es la función de similitud y $p(r_k)$ es la densidad de probabilidad a priori dada por la vecindad.

Gracias al sampler de Gibbs (ver A.1.3.) podemos decir que la densidad de probabilidad a priori se define como:

$$p(r_k) = e^{-U(w)/T}/Z$$

donde

$$Z = \sum_{w} e^{-U(w)/T}$$

es la constante de normalización, T es una constante para reducir el impacto de outliers y la distribución U(w) es la función de energía que es la suma de los potenciales V_c de todos los cliques posibles compuestos por la vecindad de una ventana dado el conjunto de soluciones posibles w.

La inicialización del MRF es dado por el mecanismo anteriormente descrito, donde se define como etiqueta perteneciente al fondo la que sea más “suave”.

Una vez establecidas las etiquetas de fondo para toda ventana se utiliza el método ICM (modo condicional iterativo, ver A.2.) para maximizar la probabilidad global de las etiquetas.

2.6. Resumen de algoritmo: suavidad, superbloques y cadenas de Markov

Este algoritmo utiliza regiones no solapadas para poder aplicar MRF, la distancia dada por la diferencia absoluta y análisis de los coeficientes del DCT para reconstruir el fondo. Se puede trabajar en base a frame por frame.

Los pasos son:

1. Cortar el vídeo en ventanas no solapadas
2. Por cada ventana agrupar los cuadros por similitud usando la distancia de diferencia absoluta. Se utiliza el método de Single-linkage clustering para agrupar los cuadros.
3. Se calcula un representante por cada cluster utilizando la media
4. Se busca la ventana donde exista solo un cluster (quiere decir que en todos los cuadros la ventana no cambia o cambia pero consistente con el ruido) y se usa como semilla

5. Se toma una ventana que tenga tres vecinos, cercanos entre si, definidos como background
6. Se calcula el DCT de la imagen dada por el super bloque compuesto por los vecinos
7. Por cada candidato se calcula el DCT y se lo combina con el DCT del superbloque, se toma el candidato que produzca la menor discontinuidad
8. Se vuelve al punto 5 hasta que no queden más ventanas a procesar
9. Utilizar ICM para reajustar la MRF

Este método tiene varios problemas y limitaciones:

- Supone que el fondo se muestra al menos una vez en cada ventana
- La eficacia es muy dependiente de cuán bien se tomen los coeficientes para la distancia y para el índice de continuidad, [2] recomienda unos valores
- El tamaño de la ventana es arbitrario y no es claro el criterio con el cual se selecciona
- No se manejan pequeños cambios explícitamente, los únicos que automáticamente se manejan son los que son consistentes con los coeficientes arbitrarios

2.7. Resumen de algoritmo: regiones solapadas y Mahalanobis

Este algoritmo utiliza regiones solapadas, la distancia de Mahalanobis y graph-cuts para reconstruir el fondo. Se trabaja con el video completo, con lo cual no esta pensado para usarse en real-time o con partes del video.

Los pasos son:

1. Estimar el ruido en el video usando MedianAD
    a. Tomar cada frame y hacer la diferencia con los siguientes 3 cuadros
    b. Computar la mediana de todas las diferencias en conjunto
    c. Computar la mediana del valor absoluto de todas

las diferencias entre cuadros menos la mediana
    d. El MedianAD es directamente proporcional a la desviación estándar, con lo cual se puede obtener la varianza sin problemas
2. Cortar el video en ventanas solapadas
3. Por cada ventana agrupar los cuadros por similitud usando la distancia de Mahalanobis con la varianza antes estimada. Se utiliza el método de Single-linkage clustering para agrupar los cuadros.
4. Se calcula un representante por cada cluster utilizando la mediana
5. Se buscan todas las ventanas donde exista solo un cluster (quiere decir q en todos los cuadros la ventana no cambia o cambia pero consistente con el ruido) y se usan como semillas, si no existen se usa una heurística
6. Se toma una ventana que tenga por vecino una ventana ya definida como background
7. Se descartan todos los clusters que no continúen las ventanas background vecinas
8. De los que quedan se juega un torneo utilizando Graph-cuts, el ganador es el fondo
9. Se vuelve al punto 6 hasta que no queden más ventanas a procesar

Este método tiene varios problemas y limitaciones:

- Supone que el fondo se muestra al menos una vez en cada ventana
- La eficacia es muy dependiente de la estimación del ruido
- Necesita tener todo el vídeo disponible, no esta pensado para trabajar iterativamente
- El tamaño de la ventana es arbitrario y no es claro el criterio con el cual se selecciona
- Muy dependiente de la mediana para eliminar casos que el algoritmo no maneja
- No se manejan pequeños cambios explícitamente, los únicos que automáticamente se manejan son los que son consistentes con el ruido

3. Criterios de clusterización, búsqueda de un método más robusto

3.1. Índice SSIM

Idealmente un índice de similitud entre dos imágenes debería ser directamente proporcional con la percepción humana [4]. Los índices clásicos de similitud MSE (mean squared error) y PSNR (peak signal to noise ratio) distan mucho de ese ideal, ya que dos imágenes parecidas pueden ser marcadas como distintas y viceversa (*fig. 18*).

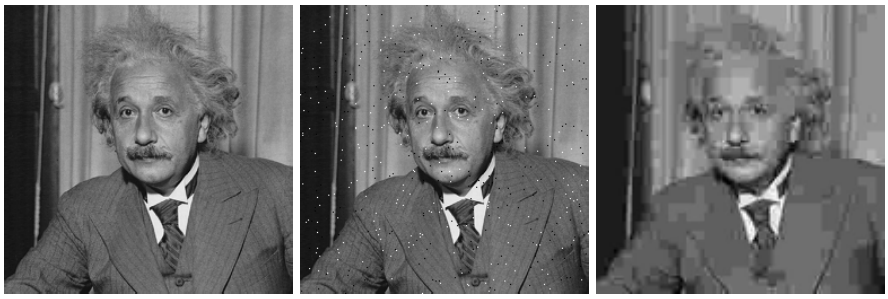


fig. 18: Las dos imágenes de la derecha tiene ruido salt&pepper y artifacts de JPG, respectivamente. El MSE para las dos es de 144 pero el índice SSIM es de 0.84 y 0.662, logrando discernir entre los dos ruidos

La idea general del índice es que cuantifica las diferencias estructurales entre dos imágenes, partiendo de la base que la percepción humana (HVS, human visual system) es muy sensible a tales diferencias. La estructura es el remanente después de eliminar la influencia de la iluminación y el contraste en la imagen. Por lo tanto se miden y se eliminan como se muestra en la figura (*fig. 19*).

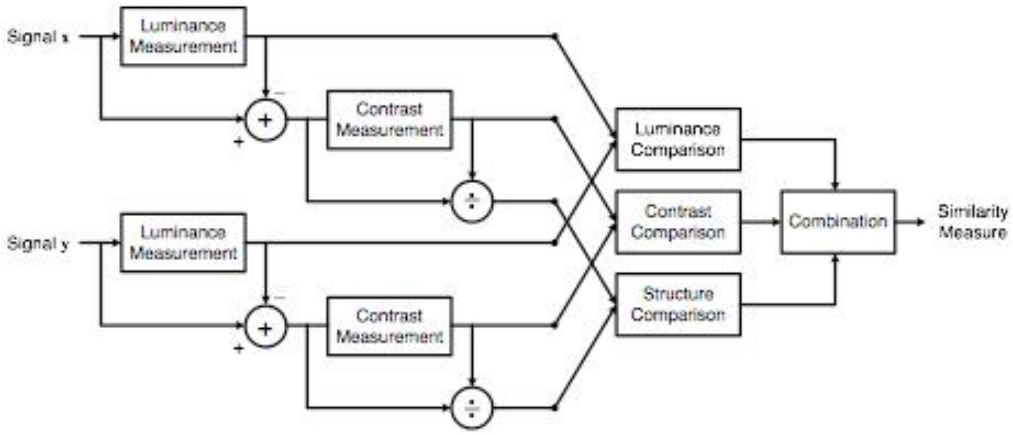


fig. 19: Estructura del índice SSIM

La iluminación se evalúa utilizando la siguiente fórmula:

$$l(x,y) = (2\mu_x\mu_y + C_1)/(\mu_x^2 + \mu_y^2 + C_1)$$

donde x y y son la componente "Luma" de las imágenes a comparar (canal Y de la descomposición YUV o YCbCr), μ_x y μ_y son la media de x y y respectivamente calculadas como:

$$\mu_x = 1/N \sum_{i=1}^{N} x_i$$

con N la cantidad de pixeles y x_i el valor de la componente Y del pixel i, C_1 es una constante incluida para evitar inestabilidad cuando $\mu_x^2 + \mu_y^2$ es cercano a 0. Tomamos C_1 como:

$$C_1 = (K_1 L)^2$$

donde L es el rango de posibles valores para x_i, para YUV y YCbCr son 8-bits por canal así que L es igual a 255, y $K_1 << 1$ una constante pequeña. Esta medida de iluminación es consistente con la ley de Weber, que describe la sensibilidad del HVS a cambios de la iluminación relativa y no absoluta, se puede ver mejor con el siguiente cambio:

$$l(x,y) = 2(1+R)/(1+(1+R)^2 + C_1/\mu_x^2)$$

donde $R = \mu_y^2/\mu_x^2 - 1$ la magnitud de cambio de

iluminación. Si suponemos que C_1 es lo suficientemente chico con respecto a μ_x como para ser ignorado, $l(x,y)$ depende solamente de R.

La comparación de contraste toma una forma muy parecida a la iluminación, pero utiliza la varianza en lugar de la media:

$$c(x,y) = (2\sigma_x\sigma_y + C_2)/(\sigma_x^2 + \sigma_y^2 + C_2)$$

con $C_2 = (K_2L)^2$ y $K_2 << 1$. Esta definición de $c(x,y)$ es consistente con las características de HVS donde para casos con la misma diferencia de contraste ($\Delta\sigma = \sigma_y - \sigma_x$), es menos sensible a altos valores de contraste base (σ_x) que a valores bajos.

Una vez obtenidos los valores de la iluminación y del contraste, se eliminan sus efectos de las imágenes para obtener la estructura subyacente utilizando $(x - \mu_x)/\sigma_x$ y $(y - \mu_y)/\sigma_y$. Una buena estimación de la diferencia estructural es el coeficiente de correlación entre $(x - \mu_x)/\sigma_x$ y $(y - \mu_y)/\sigma_y$, pero se puede ver que es equivalente al coeficiente de correlación entre x y y. Con lo cual queda definido la componente estructural como:

$$s(x,y) = (\sigma_{xy} + C_3)/(\sigma_x\sigma_y + C_3)$$

donde C_3 es una constante para evitar la inestabilidad dada cuando $\sigma_x\sigma_y$ es cercano a 0 y $\sigma_{xy} = 1/(N-1)\sum_{i=1}^{N}(x_i - \mu_x)(y_i - \mu_y)$

Combinando las tres componentes obtenemos

$$SSIM(x,y) = l^{\alpha}(x,y) \cdot c^{\beta}(x,y) \cdot s^{\gamma}(x,y)$$

con $\alpha > 0$, $\beta > 0$ y $\gamma > 0$ factores para establecer la importancia relativa de cada componente, pero se pueden tomar $\alpha = \beta = \gamma = 1$, y $C_3 = C_2/2$. Quedarían por definir K_1 y K_2 que por lo general se toman respectivamente iguales a 0.01 y 0.03. La elección de estos valores es arbitraria, pero el índice SSIM es

insensible a ellos ya que más que nada sirven para evitar los casos de inestabilidad.

Se puede ver que cada componente y, en consecuencia, el índice final cumplen con las siguientes reglas:

1. Simetría $S(x,y) = S(y,x)$
2. Acotado $S(x,y) \leq 1$
3. Único máximo $S(x,y) = 1 \Leftrightarrow x = y$

Existen otras variantes del método que mejoran su precisión y lo hacen más robusto con respecto a la translación y rotación (CW-SSIM, STSIM, STSIM-2). Se están desarrollando nuevos índices basados en STSIM que utilizan la información de color para diferenciar imágenes parecidas estructuralmente pero con colores muy diferentes.

3.2. Aplicación del índice SSIM a la clusterización

La distancia de Mahalanobis es un método estadístico que ofrece mejores resultados pero es muy sensible a cambios de iluminación y pequeños movimientos. Su fuerte dependencia a una buena estimación del ruido, hace que en la práctica se dan muchos casos no consistentes con la visión humana. A pesar de estas limitaciones, la distancia de Mahalanobis permite definir un punto de corte utilizando técnicas estadísticas bien conocidas.

El método de la media absoluta trata de compensar los problemas de la distancia de Mahalanobis agregando un índice de correlación, pero utiliza cotas determinadas empíricamente y hereda la sensibilidad a los cambios de iluminación. Por ejemplo dos imágenes iguales pero una más clara que la otra van a tener perfecta correlación pero su media de la diferencia absoluta va a ser más grande que el ruido de la cámara permitiendo diferenciarlas. En cambio dos imágenes iguales pero donde en una aparece una parte de un objeto, la media diferencia absoluta puede estar dentro del ruido de la cámara pero la correlación va a ser muy baja. La base teórica de este criterio es muy débil en el

paper, que el índice SSIM logra suplir.

Como alternativa, en este trabajo se propone el índice SSIM (Structural Similarity). El índice SSIM fue desarrollado como una alternativa a los índices MSE (mean squared error) y PSNR (peak signal to noise ratio) para la evaluación de calidad de imágenes. Los buenos resultados obtenidos en [4] abre las puertas a utilizar el índice en campos como el de base de datos de imágenes.

Para utilizar SSIM como índice de similitud para clusterizar es necesario definir un punto o rango de corte. El índice toma valores entre 0 y 1 (el método en el paper va de -1 y 1 pero se puede llevar a 0 y 1), donde 1 es cuando las dos imágenes comparadas son iguales y 0 cuando son totalmente distintas. Si se toman dos imágenes idénticas y se le aplica ruido a una, el índice SSIM va a ser directamente proporcional a tal ruido, con lo cual se pueden tomar valores cercanos al 1 como punto de corte. 0.95 se puede tomar como el equivalente de un test de confianza al 5%.

4. Resultados

4.1. Métricas

Se utilizó la métrica de AGE (average grey level error) [6] para clasificar los distintos algoritmos y sus variantes. AGE se calcula tomando la media de la diferencia absoluta entre el fondo reconstruido y el real. Quedaría definido como:

$$AGE(I_{bkg}, I_{method}) = 1/N \sum_{i=1}^{N} \left| I_{bkg}(i) - I_{method}(i) \right|$$

donde I_{bkg} es la imagen real de fondo, I_{method} es la generada por el método y N es la cantidad de píxeles de las imágenes. El algoritmo se puede reducir a los siguientes pasos:

1. Convertir a escala de grises el fondo generado y el fondo real
2. Hacer la diferencia entre los dos
3. Calcular la media de la diferencia

4.2. Set de datos

Se utilizó el set de datos CAVIAR (http://groups.inf.ed.ac.uk/vision/CAVIAR/CAVIARDATA1/) que contiene videos de seguridad en un shopping y provee información frame por frame del área que ocupan los objetos. Con lo cual se utilizó tal información para segmentar los objetos y obtener el fondo real frame por frame y se calculó el AGE evitando las zonas que contienen objetos.

4.4. Pruebas realizadas

Se tomó el método de superbloques y se variaron los tipos de clusterización y de tamaño de los parches. “full” se refiere a que se clusterizó con single linkage y “online” con respecto a los representantes. “reddy”, “mahalanobis” y “ssim” se refieren a los criterios de similitud utilizados para clusterizar. Ver apéndice A.3.

para los datos utilizados para crear el gráfico.

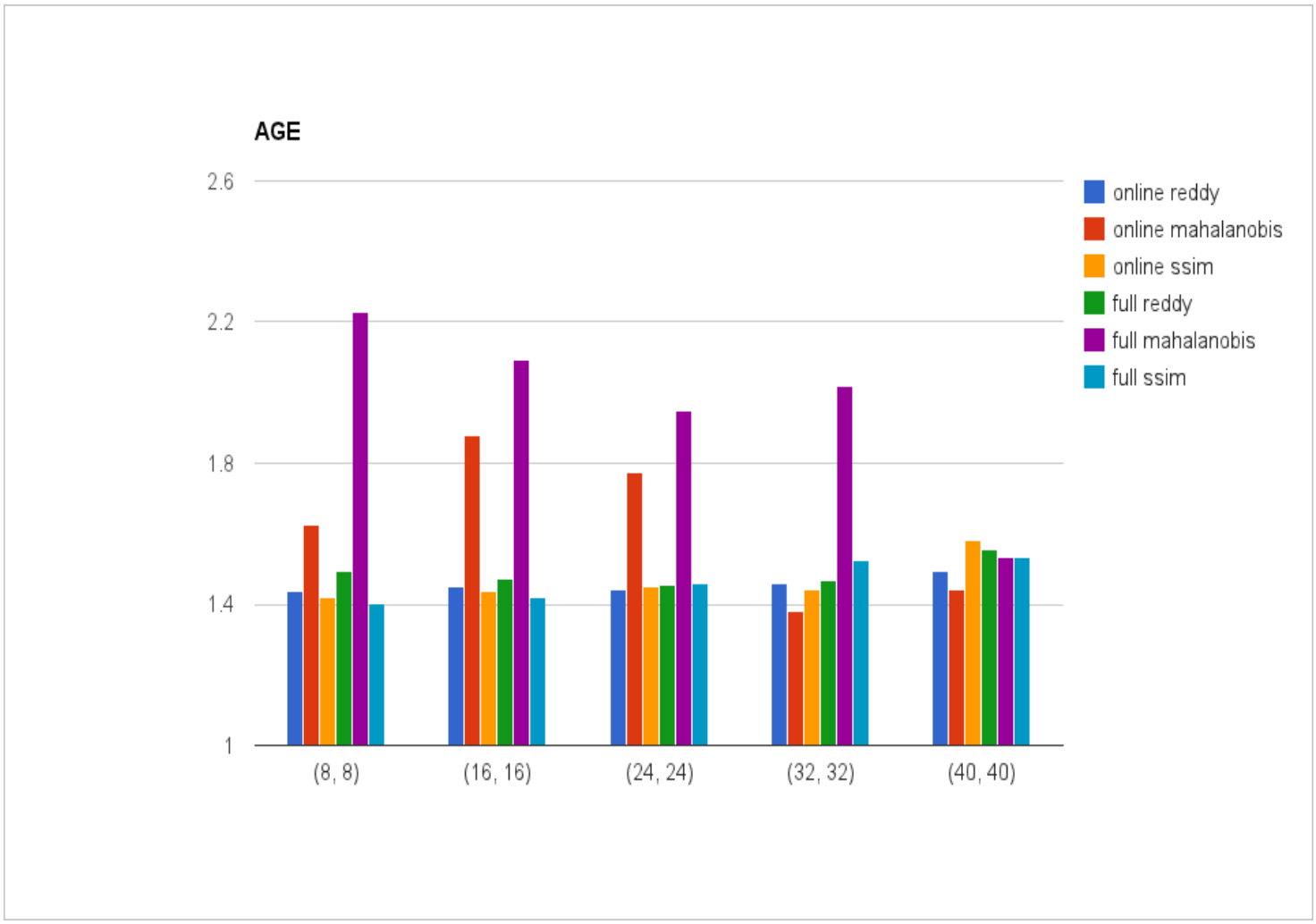
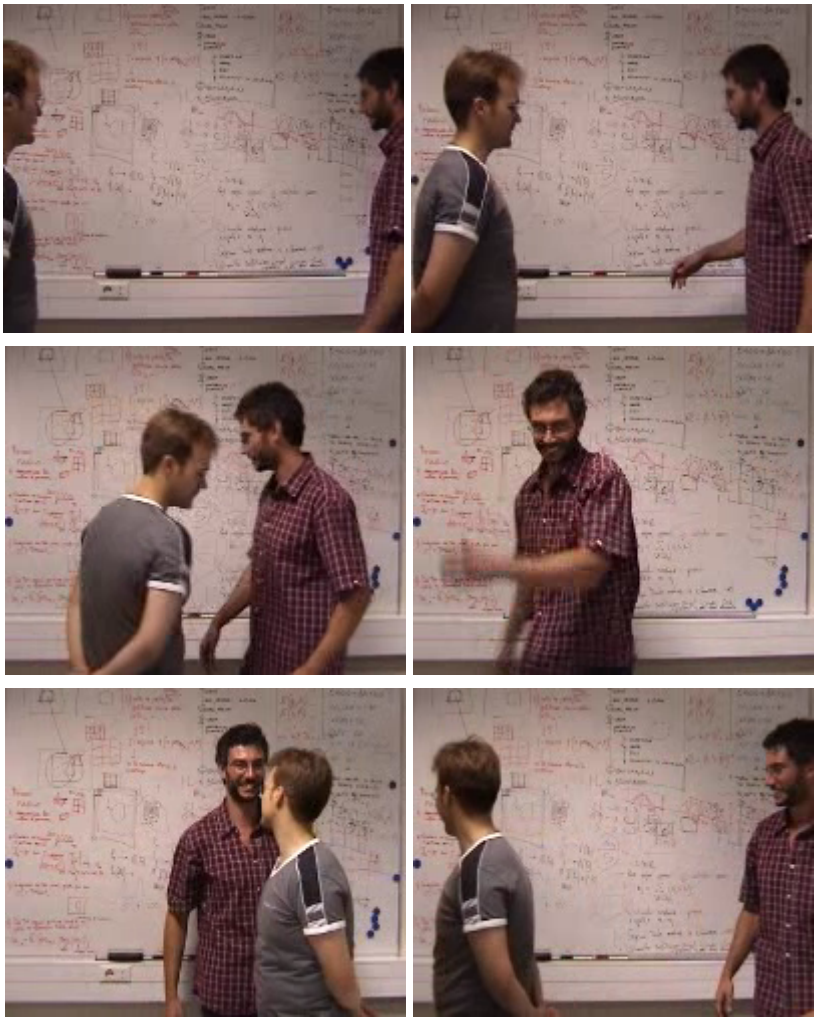


fig. 20: Gráfico de AGE de los distintos métodos

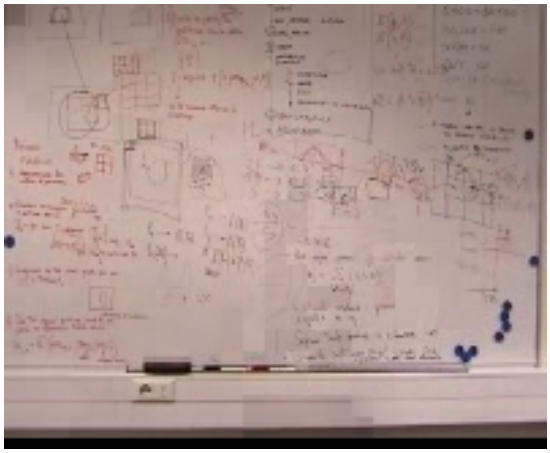
4.5. Ejemplos de corridas

A continuación se muestran ejemplos de videos utilizados para evaluar los algoritmos. Si no se especifica lo contrario los resultados obtenidos por los distintos métodos no muestran significativas diferencias.

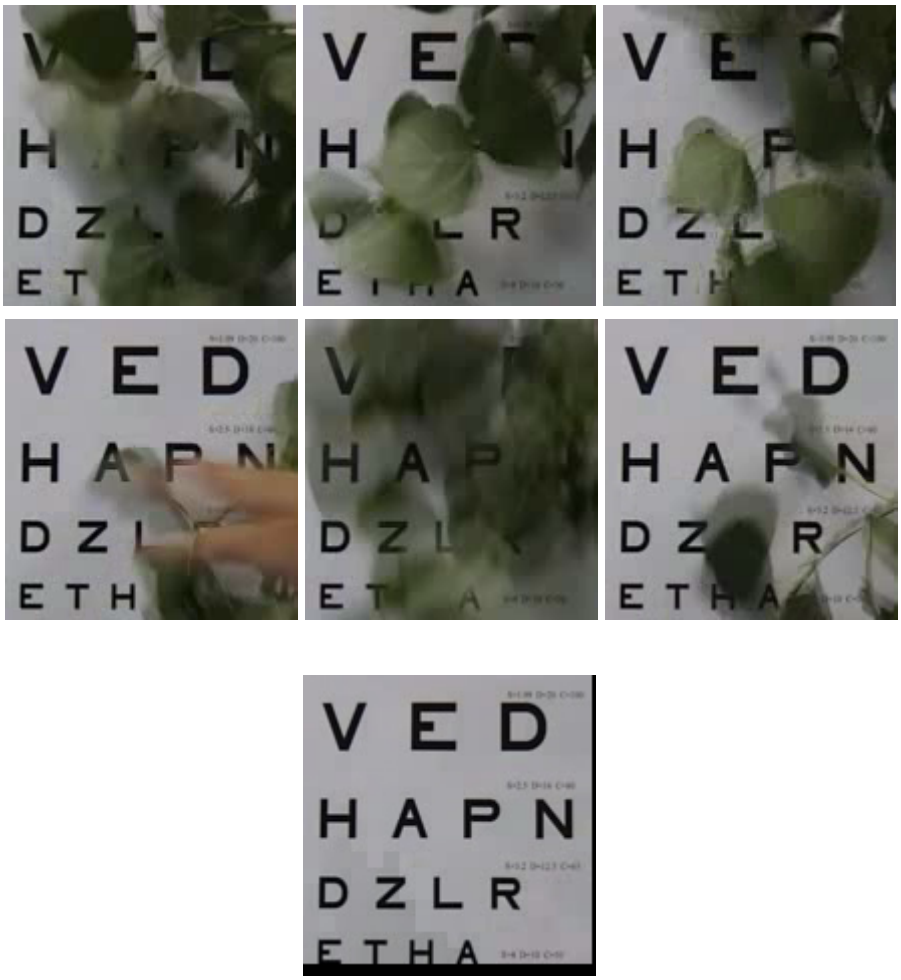
En esta escena dos personas caminan delante de un pizarrón. En ningún frame aparece solamente el pizarrón.



El fondo resultante elimina a las personas como esperado. Tiene unos artifacts debido a la sombra generada por las personas.



Otro ejemplo de escena con alto nivel de oclusión:



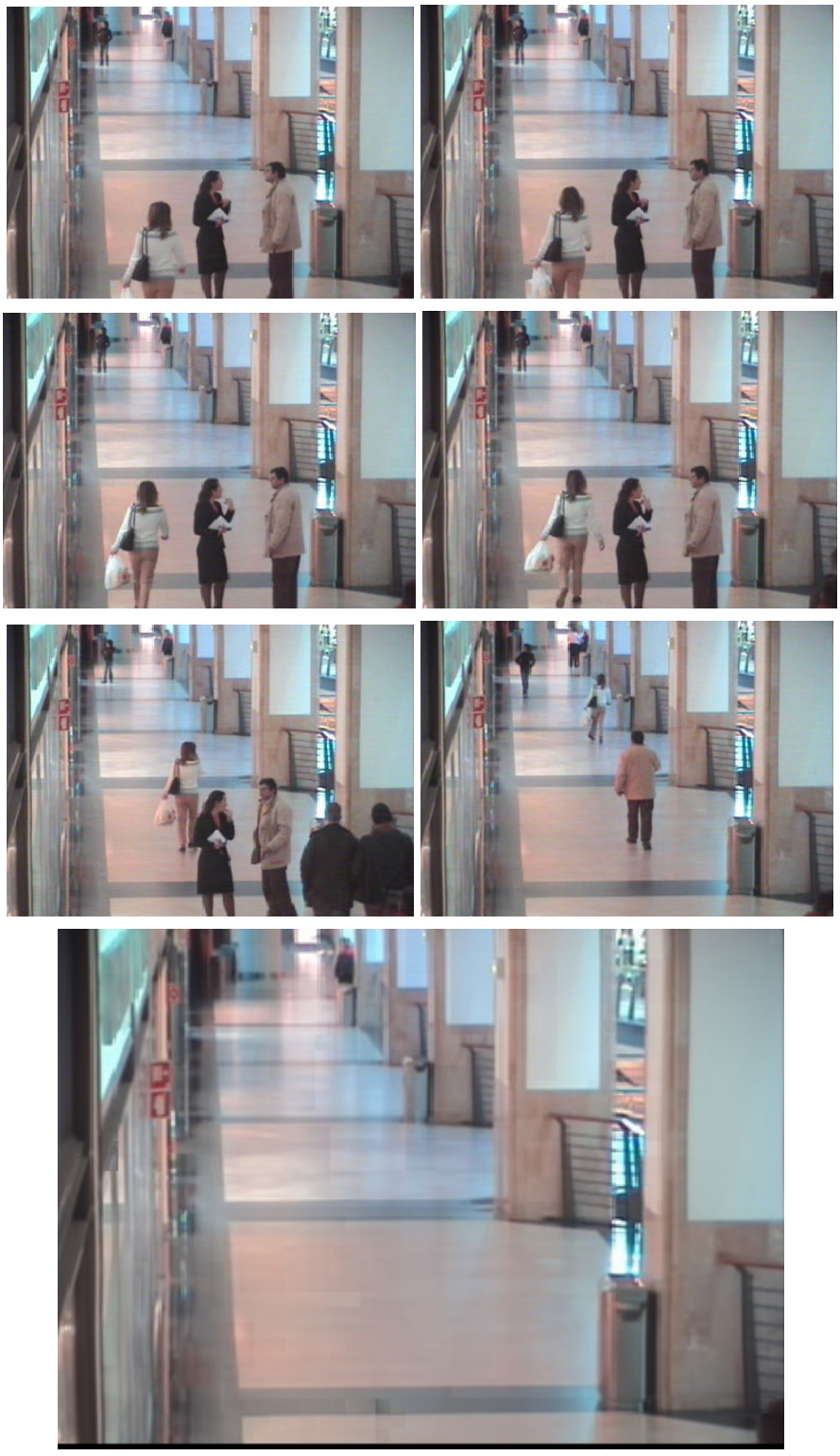
Caso donde el uso de DCT produce artifacts no queridos al tener superficies muy homogéneas.

El resultado utilizando DCT presenta fuertes artifacts a diferencia de utilizar sin costuras.

Resultado utilizando DCT

Resultado utilizando algoritmo sin costuras y "Graph-cut"

Video de la base de datos CAVIAR.



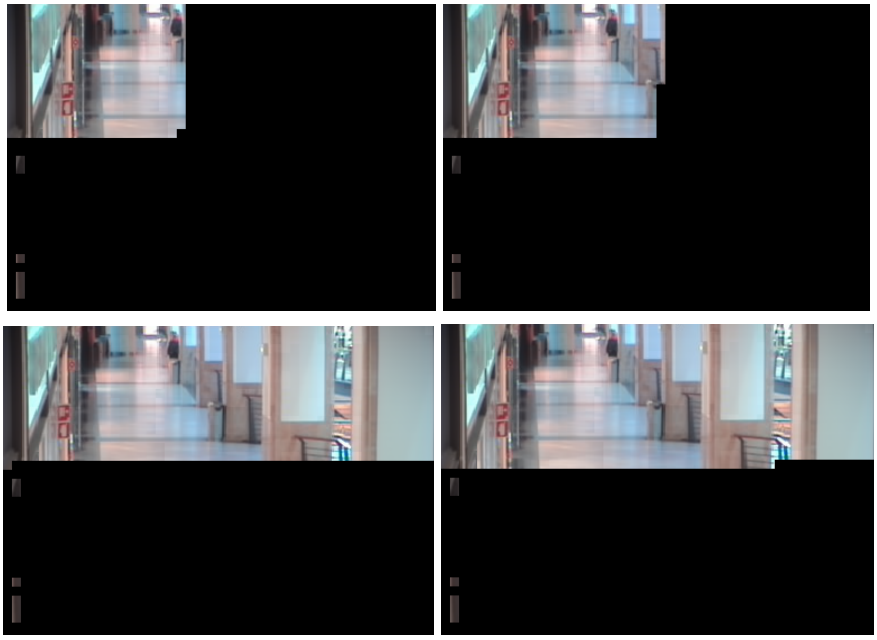
Reconstrucción final

Algoritmo paso a paso:

Regiones estáticas

Generación del fondo en base a un vecino

Generación del fondo en base al fondo reconstruido en el paso anterior



Pasos intermedios de la reconstrucción

5. Conclusiones

Por los resultados obtenidos se puede ver que el uso de SSIM logra niveles de AGE muy similares al de la media de la diferencia absoluta pero con un criterio de corte no empírico. En cambio mahalanobis se normaliza con respecto a los otros métodos recién a partir de parches más grandes, en congruencia con lo obtenido en [3].

No hay una diferencia significativa entre agrupamiento single link y el método online propuesto por [2] salvo utilizando la distancia mahalanobis, donde el método online obtiene mejores resultados. Esto se debe a que mahalanobis sobre parches chicos no logra diferenciar cambios pequeños entre los parches (significativos para tamaños comparables al de los cambios), genera clusters con parches erróneos. Utilizando el método online se reduce este tipo de errores ya que pone un límite a la agrupación temporal, o sea si hay dos clusters parecidos pero separados por otro cluster en la línea temporal, estos no se agrupan como si se haría en método “full”.

El método de agrupamiento en línea (online clustering) permite procesar los frames a medida que llegan, reduciendo de esta manera la complejidad espacial del método y mejorando la performance. Con lo cual es preferido al método más riguroso.

5.1. Trabajo futuro

5.1.2. Refinaciones a la segmentación

Los métodos anteriormente descritos se basan en que la cámara es estática. Para hacer los métodos aplicables a cámaras en movimiento se utilizan técnicas de generación de fotos panorámicas, donde se relacionan los cuadros entre sí y se los junta de tal manera que se obtiene un vídeo estático.

Para estimar el movimiento de la cámara se utilizan feature points obtenidos con métodos como SURF, FAST, MSER, etc... y

se relacionan los frames entre sí utilizando RANSAC. La estimación se ajusta utilizando bundle adjustment y se obtiene la posición de la cámara en cada frame. Para generar una imagen fija del video se determina un nuevo punto de cámara desde donde el error de proyección es menor y se proyectan los frames sobre el nuevo plano.

Una vez obtenida la imagen se puede analizar con los métodos anteriormente descritos ya que la posición de la cámara ahora es fija.

5.1.2. Paralelización e implementación en GPU

El método de Reddy es muy rápido con vídeos de baja resolución (normales en cámaras de seguridad) pero no tanto con videos de alta resolución. El algoritmo es altamente paralelizable, ya que cada ventana se puede clusterizar por separado del resto y lo mismo se puede hacer cuando se analizan los candidatos de fondo.

Con la librería OpenCV se tiene acceso a la implementación en GPU (openCL y CUDA) de muchas de las operaciones básicas entre matrices. Combinando con la paralelización se pueden obtener mejoras de performance muy grandes en los algoritmos.

6. Bibliografía

[1] P. KaewTraKulPong and R. Bowden, “An Improved Adaptive Background Mixture Model for Realtime Tracking with Shadow Detection”, 2001

[2] Vikas Reddy, Conrad Sanderson and Brian C. Lovell, “A Low-Complexity Algorithm for Static Background Estimation from Cluttered Image Sequences in Surveillance Contexts”, 2010

[3] Andrea Colombari and Andrea Fusiello, “Patch-based Background Initialization in Heavily Cluttered Video”, 2010

[4] Z. Wang, A. C. Bovik, H. R. Sheikh and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity", 2004

[5] David L. Donoho Iain M. Johnstone, “Adapting to Unknown Smoothness via Wavelet Shrinkage”, 1994

[6] D. Gutchess, M. Trajković, E. Cohen-Solal, D. Lyons and A. K. Jain, “A background model initialization algorithm for video surveillance”, 2001

[7] P.Nunes, P.Correia and F.Pereira, “Coding video objects with the emerging mpeg-4 standard”, 1997

[8] C. Stauffer and W. E. L. Grimson, “Learning patterns of activity using real-time tracking,” IEEE Transactions on Pattern Analysis and Machine Intelligence, 2000

[9] I. Haritaoglu, D. Harwood, and L. Davis, “W^4: Who? When? Where? What? a real time system for detecting and tracking people,” in Proceedings of the 3rd International Conference on Face and Gesture Recognition, 1998

[10] C. Wren, A. Azarbayehani, T. Darrell, and A. Pentland, “Pfinder: Real-time tracking of the human body,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 19, pp. 780–785, 1997

[11] D. Gutchess, M. Trajkovic, E. Cohen-Solal, D. Lyons, and A. Jain, “A background model initialization algorithm for video surveillance,” in Proceedings of the IEEE International Conference on Computer Vision, 2001, pp. 733–740

[12] A. Colombari, M. Cristani, V. Murino, and A. Fusiello, “Exemplar based Background Model Initialization,” in Proceedings

of the 3rd ACM International Workshop on Video Surveillance & Sensor Networks, 2005

[13] D. Farin, P. H. N. de With, and W. Effelsberg, “Robust background estimation for complex video sequences,” in Proceedings of the IEEE International Conference on Image Processing, vol. 1, 2003, pp. 145– 148

[14] A. K. Jain, M. N. Murty, and P. J. Flynn, “Data clustering: a review,” ACM Computing Surveys, vol. 31, no. 3, pp. 264–323, 1999

Apéndice

A.1. Campos aleatorios de Markov

Para poder entender lo que es un campo aleatorio de Markov hay que entender antes que es la propiedad markoviana.

A.1.1. Propiedad markoviana

Dado un proceso estocástico (una familia arbitraria de variables aleatorias) $\{X_t\}_{t\in T}$, en donde cada X_t es una función del espacio muestral en algún conjunto E (el espacio de estados). Se distinguen diferentes casos, según T sea un conjunto continuo (generalmente en los reales R) o discreto (en general en los naturales N), y según la cardinalidad de E. Es usual interpretar a t como el tiempo transcurrido desde el instante inicial ($t = 0$), y que en cada instante $t \in T$ se lleva a cabo un experimento de cuyo resultado queda determinado el valor de X_t. Principalmente, interesa saber con qué probabilidad cada X_t asume valores en ciertos subconjuntos de E, y también si esta probabilidad está influenciada de alguna manera por los valores observados en los instantes de tiempo anteriores (la historia del proceso). Nos interesa aquí el caso en que $T = N = \{0,1,2,3, \ldots\}$ y E es un conjunto a lo sumo numerable.

Definición: Sea E un conjunto a lo sumo numerable (de estados) y $\{X_n\}_{n\in N}$ un proceso estocástico a valores en E. Se dice que el proceso satisface la propiedad markoviana si para todo entero $n \geq 0$ y todos $j, i_n, \ldots, i_0 \in E$, se cumple:

$$P(X_{n+1} = j \mid X_n = i_n, \ldots, X_0 = i_0) = P(X_{n+1} = j \mid X_n = i_n)$$

Una cadena de Markov es un proceso estocástico que satisface la condición markoviana. Es decir, en una cadena de Markov el conocimiento de los valores de $X_0, \ldots, X_{n-1}$ no agrega información a lo que puede esperarse como valor de X_{n+1} si se

sabe el valor de X_n.

A.1.2. Campos aleatorios de Markov

Dado un grafo no dirigido *G = (V,E)* de dependencia entre variables aleatorias $\{X_v\}_{v \in V}$, se dice un campo aleatorio de Markov si cumple con las propiedades markovianas locales:

Propiedad markoviana de a pares: todo par de variable que no dependientes entre sí son condicionalmente independientes:

$$X_u \perp X_v \mid X_{V \setminus \{u,v\}} \text{ si } \{u,v\} \notin E$$

Propiedad markoviana local: Una variable es condicionalmente independiente de todas las otras variables dada su vecindad

$$X_u \perp X_{V \setminus \{u\} \cup ne(u)} \mid X_{ne(u)}$$

donde *ne(u)* es el conjunto de la vecindad de *u*.

Propiedad markoviana global: Todo subconjunto de variables son condicionalmente independientes dado un subconjunto de separación:

$$X_A \perp X_B \mid X_S$$

donde todo camino de un nodo de *A* a uno de *B* pasa por *S*.

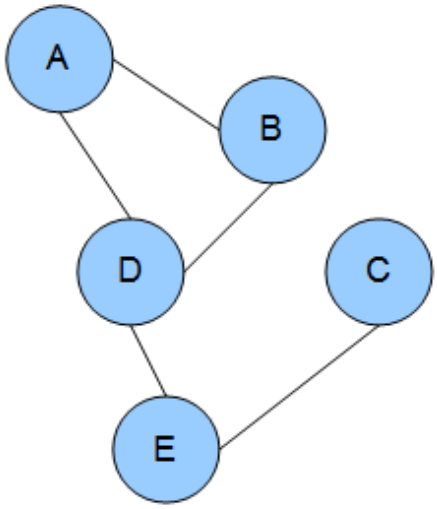


fig. 21: ejemplo de grafo de dependencias

A.1.3. Inferencia en campos aleatorios de Markov

Como en las redes bayesianas, se puede calcular la distribución condicional de un subconjunto de nodos V' = {v_1, ..., v_i} dados los valores de otro subconjunto de nodos W' = {w_1, ..., w_j} en un campo aleatorio de Markov sumando todas las posibles asignaciones a u V', W'. En su caso general esto es un problema #P-complete debido a la gran cantidad de posibles combinaciones. Para evitar este problema y resolver la inferencia se aplica el teorema Hammersley–Clifford que permite trabajar los campos aleatorios de Markov con el sampler de Gibbs. El sampler de Gibbs permite evaluar la distribución condicional para una configuración específica sin tener la distribución exacta.

El sampler permite definir para un conjunto de de variables aleatorias X con una configuración w

$$p(X = w) = e^{-U(w)/T}/Z$$

donde

$$Z = \sum_{w} e^{-U(w)/T}$$

es la constante de normalización conocida como la función de

partición, T es una constante para reducir el impacto de outliers y la distribución U(w) es la función de energía que es la suma de los potenciales V_c de todos los cliques posibles compuestos por la vecindad de una ventana dado el conjunto de soluciones posibles w.

A.2. Modo condicional iterativo

En estadística se denomina modos condicional iterativo (ICM) al algoritmo determinista para obtener la configuración que maximiza la probabilidad conjunta de un campo aleatorio de Markov.

El algoritmo ICM usa la estrategia *greedy* en la maximización iterativa local. Dados los datos d y las etiquetas $f^{(k)}_{S-\{i\}}$ (donde S son todas los nodos del campo, k es la iteración actual e i es el nodo a evaluar), el algoritmo actualiza secuencialmente cada etiqueta $f^{(k)}_i$ y la pasa a $f^{(k+1)}_i$ maximizando $P(f_i \mid d, f_{S-\{i\}})$, la probabilidad condicional (posteriori), con respecto a f_i.

A.3. Tabla de datos

A continuación están los datos en crudo utilizados para crear el gráfico de resultados.

Scene	Method	(8,8)	(16,16)	(24,24)	(32,32)	(40,40)
TwoEnterShop1cor	online reddy	0.982663224897	0.989755504153	0.999529871409	0.982191319077	1.01290534265
TwoEnterShop1cor	online mahalanobis	1.17274182281	1.38376644546	1.28621090688	0.985513643705	1.00685237383
TwoEnterShop1cor	online ssim	0.991081645014	0.973062932331	0.973893222105	0.984835822086	0.960859130562
TwoEnterShop1cor	full reddy	0.844266690402	0.840785920216	0.853455306473	0.854487861695	0.855253573999
TwoEnterShop1cor	full mahalanobis	1.51734788686	1.44029600925	1.34238373895	1.4113500647	0.915467855097
TwoEnterShop1cor	full ssim	0.935674585559	0.940118667775	0.939825594635	0.972073714243	0.904684540082
OneStopEnter1cor	online reddy	1.25675796652	1.27297563089	1.25831303614	1.29883801181	1.31808311457
OneStopEnter1cor	online mahalanobis	1.42516008092	1.68212326771	1.58456765523	1.19757854177	1.24675787571
OneStopEnter1cor	online ssim	1.21863344728	1.25946602549	1.28692559873	1.28646625659	1.48571115579
OneStopEnter1cor	full reddy	1.39088988513	1.36889293253	1.34381179948	1.36427771869	1.47219081338
OneStopEnter1cor	full mahalanobis	2.09623859461	1.95908679771	1.8167669084	1.88717394944	1.43887413147
OneStopEnter1cor	full ssim	1.21500723559	1.24390477952	1.31124693554	1.38852318346	1.42769525694
OneStopEnter2cor	online reddy	1.5257010092	1.54974943387	1.53103840824	1.58985840483	1.61083985451
OneStopEnter2cor	online mahalanobis	1.69314790148	1.95200669046	1.84440162768	1.43121494218	1.51973767371
OneStopEnter2cor	online ssim	1.49389447471	1.52343146304	1.54679990589	1.53531297365	1.79293116208
OneStopEnter2cor	full reddy	1.71633530268	1.70140331472	1.66327850298	1.68215384664	1.82475937986
OneStopEnter2cor	full mahalanobis	2.3962045255	2.26471251499	2.10778535186	2.1629453425	1.76259947297
OneStopEnter2cor	full ssim	1.50551336624	1.52016102472	1.59107712017	1.69588463177	1.76039549994
OneStopMoveEnter	online reddy	2.41313487245	2.41810637201	2.41200662257	2.3237925181	2.46237692205
OneStopMoveEnter	online mahalanobis	2.67615400729	2.95027769868	2.83205874663	2.39900992386	2.48136540089
OneStopMoveEnter	online ssim	2.46960125044	2.44260151798	2.39693557994	2.38389369597	2.28224249007
OneStopMoveEnter	full reddy	2.20535018055	2.16065079907	2.1675812058	2.14820118771	2.1746700553
OneStopMoveEnter	full mahalanobis	3.13978501839	2.92344544246	2.78369597133	2.89990578321	2.17223206649
OneStopMoveEnter	full ssim	2.39351519528	2.41977712786	2.35255893689	2.37538347746	2.24460455021
OneStopMoveNoEr	online reddy	1.58363535495	1.60192164668	1.56887380337	1.63216163194	1.6355280141
OneStopMoveNoEr	online mahalanobis	1.7556527834	2.01272980483	1.89631182957	1.48589039873	1.5421141973
OneStopMoveNoEr	online ssim	1.54466763262	1.57451784368	1.59827958865	1.58645008351	1.82430158837
OneStopMoveNoEr	full reddy	1.76192915732	1.73782884872	1.69724696202	1.71334718594	1.82984555714
OneStopMoveNoEr	full mahalanobis	2.45247150784	2.31920093626	2.15916730576	2.21342387227	1.7795297855
OneStopMoveNoEr	full ssim	1.55370565496	1.56975259126	1.63440104646	1.71623155914	1.77717600912
OneStopNoEnter1c	online reddy	1.04950922778	1.07287800064	1.07288160605	1.09432994819	1.10602341993
OneStopNoEnter1c	online mahalanobis	1.23256295033	1.49079893129	1.40178619152	1.00742102291	1.04184686761
OneStopNoEnter1c	online ssim	1.01392433124	1.04820639112	1.07327767986	1.06161175917	1.23594912767
OneStopNoEnter1c	full reddy	1.13162236183	1.12639081406	1.10819232045	1.13167241974	1.21027106237
OneStopNoEnter1c	full mahalanobis	1.86900620297	1.75543634585	1.60614574303	1.67928073943	1.19108120956
OneStopNoEnter1c	full ssim	1.0005365465	1.01766039151	1.07049120127	1.12699795509	1.16410163353

OneStopNoEnter2c	online reddy	1.2416866483	1.2548434354	1.23751750461	1.28746323548	1.3264834658
OneStopNoEnter2c	online mahalanobis	1.40826327372	1.66521232691	1.57035738021	1.15342858955	1.24551960314
OneStopNoEnter2c	online ssim	1.21043545065	1.24400395862	1.26685856218	1.26257865087	1.49995467866
OneStopNoEnter2c	full reddy	1.40631898147	1.37821758688	1.35265957332	1.36973067741	1.50973136746
OneStopNoEnter2c	full mahalanobis	2.1270702941	1.98836051222	1.81549894919	1.87839128857	1.45885468226
OneStopNoEnter2c	full ssim	1.2030282036	1.22921297534	1.29902292364	1.41111747471	1.46378975921