



Centro de Investigación en Matemáticas A.C.

# Substracción de Fondo Basado en Movimiento

Tesis

que para obtener el grado de

**Maestro en Ciencias con Especialidad en  
Computación y Matemáticas Industriales**

presenta

**Juana Edith Santoyo Morales**

Director de Tesis

**Dr. Rogelio Hasimoto Beltrán**

*Guanajuato, Gto.*

*Julio de 2008*



*A mi mamá.*



---

# Contenido

---

<b>Contenido</b>	<b>i</b>
<b>1 Introducción</b>	<b>11</b>
1.1 Motivación . . . . .	11
1.2 Planteamiento del problema . . . . .	11
1.3 Estado del arte . . . . .	12
<b>2 Modelos de substracción de fondo</b>	<b>15</b>
2.1 Introducción . . . . .	15
2.2 Modelación utilizando la media como estadístico . . . . .	16
2.3 Modelación utilizando la mediana como estadístico . . . . .	18
2.4 Modelo de mezclas . . . . .	20
2.4.1 Entendiendo el modelo de mezclas . . . . .	20
2.4.2 Mezcla de gaussianas con velocidad de aprendizaje adaptativa	25
2.4.3 Número de distribuciones del modelo de mezcla . . . . .	30
2.4.4 Método en <i>Cascada</i> . . . . .	34
2.4.5 Otras variantes . . . . .	35
<b>3 Factores externos al movimiento</b>	<b>41</b>
3.1 Identificación de variación de la iluminación y ruido . . . . .	42
3.2 Cambios rápidos en la iluminación . . . . .	47
3.3 Detección de sombras . . . . .	48
3.3.1 Correlación con la intensidad de imágenes . . . . .	49
3.3.2 Correlación de imágenes a color . . . . .	51
<b>4 Trabajo desarrollado y resultados</b>	<b>55</b>
4.1 Umbral para $\sigma^2$ . . . . .	55
4.2 Velocidad de aprendizaje . . . . .	57
4.3 Número de distribuciones adaptativo . . . . .	62
4.4 Píxel vs. Estadístico de región . . . . .	67
4.5 Espacios de color . . . . .	71
4.6 Prueba de ruido . . . . .	79
4.7 Modificación del estadístico . . . . .	82

4.8	Algoritmo final . . . . .	84
<b>5</b>	<b>Conclusiones y trabajo futuro</b>	<b>91</b>
5.1	Conclusiones . . . . .	91
5.2	Trabajo futuro . . . . .	91
	<b>Bibliografía</b>	<b>93</b>
<b>A</b>	<b>Algoritmo Expectation-Maximization</b>	<b>101</b>
A.1	Aplicando EM al modelo de mezcla de gaussianas . . . . .	102
A.2	Modificación a ecuaciones online . . . . .	104
<b>B</b>	<b>Conversión de espacios de color</b>	<b>107</b>
	<b>Lista de Figuras</b>	<b>117</b>
	<b>Lista de Tablas</b>	<b>118</b>

---

# Agradecimientos

---

Agradezco a Conacyt por el apoyo otorgado en la realización de esta tesis. Al Cimat por los recursos académicos proporcionados durante mis estudios de maestría. A mi director de tesis Dr. Rogelio Hasimoto Beltrán ya que sin su apoyo y comprensión hubiera sido difícil la conclusión del presente trabajo. A Johan Van Horebek, Dr. Jean Bernard Hayet por aceptar ser mis sinodales. A Maximino Tapia, que fué mi tutor de maestría, agradezco su apoyo, comprensión y amistad.

A mis maestros por su apoyo durante mis estudios de maestría, en especial al Dr. Mariano Rivera Meraz y Dr. Arturo Hernández Aguirre ya que sin sus consejos, comprensión e interés hubiera sido sumamente difícil la finalización de la maestría. Al Dr. Jean Brnard Hayet, por su paciencia, interés, comprensión e introducción al trabajo de tesis.

A mi familia: mamá Mary, papá Francisco, hermanas Dulce y Susana, sobrino Carlos, cuñado Enrique por su apoyo y amor que me ayuda cada día para continuar con mis sueños. A Miguel Angel por sus consejos, apoyo, preocupación y amor. Estoy muy orgullosa de tí y te has vuelto en una parte primordial e importante en mi vida. Te amo!

A mi amiga de tanto años Aurea, que ni la distancia afecta lo mucho que la quiero. A mis compañeros y amigos de la maestría, muy especialmente: Cyntia, Josué, Gerardo, Carlos Angulo que me ayudaron muchísimo a sobrevivir a la experiencia de la maestría en Cimat. A mis compañeros de Cimat, que hicieron más fácil la terminación de mi trabajo de maestría con su apoyo y amistad: Laura, Migue, Haydey, Isabel, Lenin. Y a Ivvan, por su ayuda, apoyo y amistad. A mis amigos de la Gerencia de Software: Lupita, Esteban, Israel, Jaime, por todo su apoyo, amistad y preocupación.

Y finalmente, pero no menos importante, a Dios. Que siempre me da la oportunidad y fortaleza de realizar mis planes.





---

# Descripción de símbolos e imágenes

---

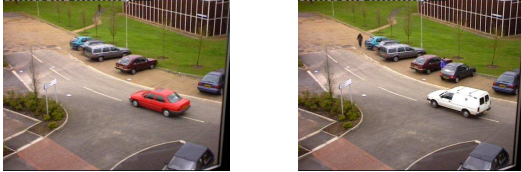


## Lista de símbolos





Variable	Descripción
$I_{(i,j)}^t$	píxel en la posición $(i, j)$ de la imagen actual, $I$ , en el tiempo $t$
$\hat{B}_{(i,j)}^t$	píxel en la posición $(i, j)$ de la imagen estimada del fondo, $\hat{B}$ , en el tiempo $t$
$Etiquetas_{(i,j)}^t$	píxel en la posición $(i, j)$ de la imagen binaria, $Etiquetas$ , donde los píxeles etiquetados con 1 indican <i>fondo</i> y con 0 se indica <i>no fondo</i>
$I_{(i,j)}^0$	píxel en la posición $(i, j)$ de la <b>primer</b> imagen de la secuencia de video
$\hat{B}_{(i,j)}^0$	píxel en la posición $(i, j)$ de la <b>primer</b> estimación de la imagen de fondo, $\hat{B}$
$pixel_{(i,j)}$	píxel en la posición $(i, j)$ de la imagen $I$
$\hat{b}_{(i,j)}$	píxel en la posición $(i, j)$ de la imagen $\hat{B}$
$\alpha$	velocidad de aprendizaje
$Th$	umbral para la segmentación de un píxel en fondo o no fondo; este umbral se aplica en la diferencia absoluta
$Th_{avg}$	umbral para la segmentación de un píxel en fondo o no fondo; este umbral se aplica en el método de la media
$m_{(i,j)}^t$	mediana del píxel en la posición $(i, j)$ en el tiempo $t$
$Th_m$	umbral para la segmentación de un píxel en fondo o no fondo; este umbral se aplica en el método de la aproximación de la mediana
$\beta$	velocidad para la actualización de la mediana
$t$	número de secuencia actual del video que se está procesando
$K$	número de distribuciones de la mezcla de gaussianas
$\omega_i^t$	peso de la distribución $i$ de la mezcla de gaussianas en el tiempo $t$





Variable	Descripción
$\mu_i^t$	media de la distribución $i$ de la mezcla de gaussianas en el tiempo $t$
$\Sigma_i^t$	matriz de varianzas de la distribución $i$ de la mezcla de gaussianas en el tiempo $t$
$\eta(X   \mu, \Sigma)$	función de densidad de la gaussiana con parámetros $x, \mu, \Sigma$
$n$	dimensión de los componentes de la mezcla de gaussianas
$N$	número de elementos de la vecindad de un píxel bajo evaluación
$ \Sigma $	determinante de $\Sigma$
$\Theta^t$	parámetros del modelo en el tiempo $t$ , $\Theta = (\omega_1^t, \dots, \omega_K^t, \mu_1^t, \dots, \mu_K^t, \Sigma_1^t, \dots, \Sigma_K^t)$
$\theta_k^t$	parámetros de la $k$ -ésima función de densidad en el tiempo $t$ , en el caso de función de densidad de una gaussiana, $\theta_k = (\mu_k^t, \Sigma_k^t)$
$\rho_k$	velocidad de aprendizaje para la actualización de los parámetros $\theta$ de la $k$ -ésima gaussiana de la mezcla
$\hat{k}$	distribución que tuvo concordancia positiva con el píxel actual
$match$	variable booleana que indica si el píxel actual encontró concordancia con alguna de las distribuciones
$\sigma_{INICIAL}^2$	valor de varianza con la que se inicializa la mezcla de gaussianas
$Th_\sigma$	umbral con el que se decidirá si existe concordancia del píxel actual con alguna de las distribuciones gaussianas
$B$	conjunto de distribuciones que modelan el fondo en el método de Stauffer y Grimson
$Th_B$	umbral para obtener el conjunto de distribuciones $B$
$\omega_{CHICO}$	valor de los pesos $\omega$ con que se inicializa la mezcla de gaussianas
$\rho_k(t)$	función que actualiza el parámetro $\rho$ , el cual es la velocidad de aprendizaje utilizada para la actualización de los parámetros de la $k$ -ésima distribución gaussiana
$Th_F$	umbral para la segmentación de un píxel en fondo o no fondo; este umbral se aplica en el método efectivo de Lee
$c_k$	número de píxeles que han concordado con la gaussiana $k$
$p_k$	probabilidad que tiene un dato de pertenecer a la distribución $k$ , $p_k = \omega_k \cdot \eta(X   \mu, \Sigma)$
$q_k$	probabilidad posterior de la gaussiana $k$ , $q_k = \frac{p_k}{\sum_{j=1}^K p_j}$
$Winner\_Take\_All$	booleano que indica si se actualizarán los valores de la distribución que tuvo <b>mayor</b> concordancia con el píxel actual o se actualizarán todas las que tuvieron concordancia.
$inf$	valor con el que se inicializan las medias de las distribuciones de la mezcla, se elige un valor que no este en el rango de valores de los datos, $inf \notin [0, 255]$




Variable	Descripción
$L$	número de últimos frames, esta variable se utiliza en el método con número de distribuciones adaptativa
$\delta$	velocidad de olvido, esta variable se utiliza en el método con número de distribuciones adaptativa
$Th_{\omega}$	umbral para la eliminación de componentes gaussianos en el método de mezcla de gaussianas con $K$ adaptativo
$N_k^t$	estadístico que cuenta el número de muestras que pertenecen al componente Gaussiano $k$
$S_k^t$	estadístico que representa la suma de las muestras que pertenecen al componente Gaussiano $k$
$Z_k^t$	estadístico que representa la suma del producto externo de las muestras que pertenecen al componente Gaussiano $k$
$Th_N$	umbral para la prueba de ruido
$Th_I$	umbral para la prueba de variación en iluminación

## Descripción de secuencias de prueba

Secuencia		Descripción
Workshop PETS (Performance Evaluation of Tracking and Surveillance)		
<i>PETS2000</i>	Nombre Núm secuencias Características	PETS 2000 1452 Personas y automóviles en movimiento, variaciones leves en la iluminación, asimilación de objetos que se detienen
	Muestra	
<i>PETS2001_DS1_TE1_C1</i>	Nombre Núm secuencias Características	PETS 2001 DataSet1 Testing 1 Camera 1 2687 Personas y automóviles en movimiento, asimilación de objetos al fondo
	Muestra	
<i>PETS2001_DS1_TE1_C2</i>	Nombre Núm secuencias Características	PETS 2001 DataSet1 Testing 1 Camera 2 2688 Personas y automóviles en movimiento, asimilación de objetos que se incorporan al fondo, objetos en movimiento que se pueden confundir con el fondo debido a la similitud en color que existe
	Muestra	
<i>PETS2001_DS3_TE1_C1</i>	Nombre Núm secuencias Características	PETS 2001 DataSet3 Testing 1 Camera 1 5336 Personas y automóviles en movimiento, asimilación de objetos que se incorporan al fondo, objetos en movimiento que se pueden confundir con el fondo debido a la similitud en color que existe, cambios grandes en la iluminación debido al movimiento de las nubes e intensidad del sol
	Muestra	

Secuencia		Descripción	
	Muestra		
Diversos			
<i>WaterSurface</i>	Nombre Núm secuencias Características  Muestra	WaterSurface 634 Persona en movimiento, variaciones en la iluminación, movimiento en el fondo debido al mar  	
<i>HighWay1</i>	Nombre Núm secuencias Características  Muestra	HighWay1 441 Movimiento de automóviles frecuente, variaciones en la iluminación, sombras densas  	
<i>HighWay2</i>	Nombre Núm secuencias Características  Muestra	HighWay2 501 Movimiento de automóviles frecuente, sombras densas, objetos en movimiento que se pueden confundir con el fondo debido a la similitud en color que existe  	
<i>Fountain</i>	Nombre Núm secuencias Características	Fountain 524 Movimiento de personas y automóviles, movimiento del fondo debido a la fuente y árboles, objetos en movimiento que se pueden confundir con el fondo debido a la similitud en color que existe	

Secuencia		Descripción	
	Muestra		
<i>Campus</i>	Nombre Núm secuencias Características	Campus 1440 Movimiento de personas y automóviles, movimiento del fondo ocasionado por gran viento, cambios en la iluminación	
	Muestra		
<i>Lobby</i>	Nombre Núm secuencias Características	Lobby 1547 Movimiento de personas, objetos en movimiento que se pueden confundir con el fondo debido a la similitud en color que existe, cambios bruscos en la iluminación ocasionado por apagar y encender lámparas	
	Muestra		
<i>Escalator</i>	Nombre Núm secuencias Características	Escalator 2150 Movimiento frecuente de personas, cambios en la iluminación, movimiento del fondo debido a las escaleras eléctricas	
	Muestra		
<i>Rain</i>	Nombre Núm secuencias Características	Rain 201 Movimiento de automóviles, ruido ocasionado por la lluvia	

	Secuencia	Descripción
	Muestra	
<i>Bootstrap</i>	Nombre Núm secuencias Características	Bootstrap 3056 Movimiento frecuente de personas, cambios en la iluminación, sombras 
<i>Umd</i>	Nombre Núm secuencias Características	Umd 626 Movimiento de personas, asimilación de objetos al fondo, sombras 
	Muestra	





# Capítulo 1

---

## Introducción

---

### 1.1 Motivación

La identificación de objetos en movimiento en una secuencia de video es una tarea fundamental y crítica, debido a que la detección de estas regiones es el primer paso a realizar en la arquitectura de la mayoría de las aplicaciones de visión computacional, como lo son: vigilancia, análisis y monitoreo vehicular, detección y seguimiento de objetos, etc. Lograr un buen resultado en la detección de movimiento es una tarea muy complicada, debido a diversos factores que afectan la calidad del video recibido, como los son, cambios en la luminosidad, ruido de la cámara, condiciones climáticas, fondo no estático, asimilación de objetos en movimiento al fondo, sombras, similitud de objetos y áreas que presentan objetos en movimiento con frecuencia, los cuales no permiten el modelado correcto del fondo, entre otros.

### 1.2 Planteamiento del problema

Considérese una secuencia de video adquirida de una cámara estacionaria, la cual observa una intersección de tráfico, existen varios factores a los que un buen algoritmo para detección de cambios se debe de adaptar, como por ejemplo: **cambios en la iluminación** que ocurren debido al movimiento del sol, condiciones climáticas adversas como nieve, neblina, lluvia, etc., **fondo no estático**, como las ramas de los árboles, **objetos integrados al fondo**, por ejemplo, continuando con el la escena de la intersección de tráfico , cuando una persona se mueve en frente de un carro recién estacionado, se debería de poder detectar solo a la persona y no a ambos, carro y persona; con **sombras**, **camuflaje de objetos**, apariencias muy similares de los objetos a detectar y del fondo ocasionará que la distinción entre los dos sea difícil, **áreas de mucho tráfico**, cuando una escena es ocupada frecuentemente por muchos objetos en movimiento, puede ser

difícil obtener una referencia del fondo estable.

### 1.3 Estado del arte

Diversos métodos han sido desarrollados para manejar los problemas mencionados anteriormente. La mayoría de los que existen en la actualidad utilizan información espacial o temporal de la secuencia de imágenes y pueden ser categorizados en tres grupos [Hu et al., 2004]: **diferencia temporal**, en estas técnicas la diferencia de píxeles de 2 ó 3 frames consecutivos será significativamente grande en regiones que presentan movimiento comparado con regiones estáticas; así que esta suposición será usada para segmentar las regiones estáticas y las que están en movimiento; **flujo óptico**, determinando el movimiento de regiones en imágenes consecutivas se pueden segmentar los objetos en movimiento del fondo estático. Objetos con un patrón de flujo diferente del movimiento general serán etiquetados como objetos en movimiento (por ejemplo, el flujo del fondo causado por el movimiento de la cámara, también puede ser distinguido); **substracción del fondo**, un modelo del fondo es creado y píxeles de la imagen actual que son significativamente diferentes del modelo son clasificados como píxeles en movimiento.

Lo más significativo de los métodos de diferencia temporal es el comportamiento robusto que se tiene en situaciones con cambios rápidos en la iluminación; los métodos basados en el flujo óptico han probado su efectividad en sistemas con cámaras en movimiento. Aunque ambos métodos detectan objetos en movimiento no son convenientes en algunas situaciones, como lo es seguimiento de personas dado que las personas se mantienen quietas al realizar alguna actividad.

Una manera muy común de realizar la detección de objetos en movimiento es utilizar la **substracción del fondo**, en donde cada frame del video es comparado contra una imagen de referencia o un modelo de fondo. Las ventajas de los métodos de substracción de fondo dependen de la manera en que se modela el fondo. En la literatura se han propuesto diferentes modelos, los cuales contienen los siguientes tres módulos: *modelo de fondo*, *mecanismo de actualización* y *entrenamiento para la inicialización del modelo*. Los buenos resultados que se tengan en los métodos de substracción de fondo dependen fuertemente de la representación que se tenga del color, por ejemplo, representaciones de color que separan la información de brillo y de cromaticidad usualmente tienen mayor robustez ante cambios rápidos en la iluminación.

Una manera simple de modelar el fondo es modelar la variación de las intensidades o color de un píxel como una sola distribución unimodal; la aplicación Pfinder [Wren et al., 1997] utiliza este modelo estadístico multiclase para seguir objetos. Después de un periodo de inicialización donde la escena está vacía, el sistema reporta buenos resultados. No hay reportes de situaciones al aire libre.

[Horprasert et al., 1999] usa cuatro valores para modelar un píxel, la media, la desviación estándar de los valores en RGB, la variación en el brillo y la variación en la cromacidad. [Monnet et al., 2003], propusieron un método online basado en predicción para modelar las escenas dinámicas. Se han tenido buenos resultados en escenas con olas del mar y con movimiento de los árboles; sin embargo, se necesitan demasiadas imágenes sin objetos en movimiento para aprender el modelo de fondo y los objetos en movimiento no se pueden detectar si éstos se mueven en la misma dirección que las olas del mar. [Mittal and Paragios, 2004] presentaron una técnica de sustracción de fondo basada en movimiento usando una estimación de la densidad con kernels adaptativos. En su método, se calcula el flujo óptico y es utilizado como característica. Ellos pueden manejar fondos complejos, pero el costo computacional es muy alto. [Huwer and Niemann, 2000] propusieron un método que combina las diferencias y la modelación del fondo adaptativo, este método puede manejar cambios de luminosidad, sin embargo, ninguno de estos métodos puede lidiar con variaciones rápidas en la iluminación. Recientemente, [Li et al., 2004] propusieron una arquitectura Bayesiana que incorpora características espectrales, espaciales y temporales para caracterizar la apariencia del fondo en cada píxel. Su método puede manejar fondos estáticos y dinámicos. [Ridder et al., 1995] modelan cada píxel con un filtro de Kalman lo cual provoca que la aplicación sea más robusta a cambios en la iluminación, pero la desventaja que presenta este trabajo es que no maneja fondos bimodales muy bien. [Elgammal et al., 2000] un modelo no paramétrico es utilizado, donde una función basada en kernel es usada para representar la distribución de cada color por píxel. En los trabajos de [Piccardi and Jan, 2004] y [Han et al., 2007] se utiliza el método mean-shift para modelar el fondo. Recientemente, la mezcla de gaussianas se ha vuelto muy popular ya que puede manejar cambios lentos en la iluminación, movimientos periódicos, movimiento lento de los objetos, ruido de la cámara, etc. [Friedman and Russell, 1997] implementaron una estructura de EM por píxel para detectar vehículos, este método clasifica cada valor de píxel en tres clases predeterminadas: color del camino, sombras y vehículos. [Stauffer and Grimson, 1999] modelaron cada píxel como una mezcla de Gaussianas y usaron una aproximación del EM para actualizar el modelo. Su sistema puede lidiar con cambios en la iluminación, movimiento lento de los objetos e introducir o remover objetos de la escena.

En este trabajo se estudian y analizan los métodos de sustracción de fondo que utilizan la mezcla de gaussianas por píxel y cuyos parámetros son actualizados utilizando una modificación del algoritmo EM. Debido a que pueden lidiar con la mayoría de los problemas que se tienen en la detección de objetos en movimiento, a su simplicidad y a los resultados reportados, entre otros motivos, los métodos de sustracción de fondo serán el núcleo del análisis realizado en este trabajo.

En el capítulo 2 se mencionan varios métodos de sustracción de fondo, entre ellos el método de Stauffer y Grimson, el cual se ha convertido un estándar en métodos de sustracción de fondo que utilizan la mezcla de gaussianas para modelar cada píxel, así como variaciones que se han realizado en el método para

aumentar su efectividad. En el capítulo 3, se mencionan algunas técnicas utilizadas para eliminar factores que afectan el buen desempeño de los algoritmos, como lo son luminosidad, ruido y sombras. Modificaciones realizadas a algunos métodos, algunas propuestas y resultados de la tesis se mencionan en el capítulo 4. Conclusiones y trabajo futuro se presentan en el capítulo 5.

## Capítulo 2

---

# Modelos de substracción de fondo

---

Este capítulo introduce el concepto de modelos adaptativos de fondo para imágenes de video y describe varios métodos para modelar el fondo. El primero es un método simple el cual aproxima la media de cada píxel; mientras que el segundo utiliza una aproximación de la mediana. El tercero es más complejo y poderoso, el cual utiliza una mezcla de gaussianas para modelar cada píxel. Los siguientes métodos que se revisarán son una modificación del método de la mezcla de gaussianas, los cuales mejoran notablemente los resultados reportados. Al final de cada sección se muestra el pseudo-código de cada método así como también algunos ejemplos.

### 2.1 Introducción

En visión computacional la modelación del fondo se refiere a estimar una imagen o estimar estadísticos del fondo de una escena que aparece en una secuencia de imágenes o video. La manera más simple de modelar el fondo es tal vez tomar una imagen de la escena cuando no está presente algún objeto y entonces usar esta imagen como modelo de fondo. Los objetos que no pertenecen al fondo pueden ser determinados por medio de la diferencia absoluta de frames, esto es, comparar cada píxel de la secuencia actual,  $I^t$ , con la imagen estimada del fondo,  $\hat{B}$ , y si el valor absoluto de la diferencia está debajo de algún umbral,  $Th$ , el píxel es clasificado como fondo,

$$Etiquetas_{(i,j)} = |\hat{B}_{(i,j)} - I^t_{(i,j)}| < Th. \quad (2.1)$$

Esta solución debería de ser suficiente en ambientes controlados, pero en condiciones arbitrarias como escenas al aire libre, las condiciones de iluminación varían con el tiempo. Además, podría ser difícil o imposible poder tomar una imagen de

la escena a modelar sin algún objeto presente. Por lo tanto es conveniente tener un modelo de fondo que se adapte a la escena.

Esta sección se enfoca a modelos de fondo adaptativos que pueden ejecutarse en tiempo real. Los métodos adaptativos de la literatura explicados en este trabajo son clasificados como técnicas recursivas, ya que el modelo de fondo se actualiza recursivamente en cada iteración.

## 2.2 Modelación utilizando la media como estadístico

En este método, el fondo es modelado como un promedio de todos los frames anteriores y del actual,

$$\hat{B}_{(i,j)}^t = \frac{1}{t} \sum_{t'=1}^t I_{(i,j)}^{t'},$$

donde  $I_{(i,j)}^t$  y  $\hat{B}_{(i,j)}^t$  son el píxel en la posición  $(i, j)$  en el tiempo  $t$  de la imagen actual y fondo estimado, respectivamente. La ecuación anterior también se puede calcular de manera incremental:

$$\hat{B}_{(i,j)}^t = \frac{(t-1)}{t} \hat{B}_{(i,j)}^{t-1} + \frac{1}{t} I_{(i,j)}^t,$$

Un problema con este enfoque son los cambios de iluminación que ocurren con el tiempo en la escena. Esto se puede manejar utilizando un olvido exponencial, esto es, la contribución de la imagen de fondo es tomada de tal manera que decrezca exponencialmente a medida que nos alejamos del fondo que fué calculado con los frames anteriores. Este enfoque es implementado usando una velocidad de aprendizaje  $\alpha$ , entonces cada píxel  $(i, j)$  del fondo estimado en el tiempo  $t$  estaría dado por:

$$\hat{B}_{(i,j)}^t = \alpha I_{(i,j)}^t + (1 - \alpha) \hat{B}_{(i,j)}^{t-1}.$$

Ya que se tiene el fondo, la parte que es No Fondo se obtiene utilizando la ecuación 2.1. Este método es probablemente el más simple, además que tiene una rápida y sencilla implementación pero los resultados que se obtienen no son buenos en especial en fondos complejos.

El procedimiento de este método se encuentra en el Algoritmo 1.

---

**Algoritmo 1:** Aproximación del filtro de la media
 

---

**Entrada:**  $I^t, \hat{B}^{t-1}, \alpha, Th_{avg}$

**Salida :**  $\hat{B}^t, Etiquetas$

```

1 #Inicialización
2 forall  $pixel_{(i,j)} \in I^0$  y  $\hat{b}_{(i,j)} \in \hat{B}^0$  do
3   |  $\hat{b}_{(i,j)} = x_{(i,j)}$ 
4 end
5  $t = 1$ 
6 while se tenga una secuencia  $I^t$  do
7   #Actualización del fondo
8   forall  $pixel_{(i,j)} \in I^t$  y  $\hat{b}_{(i,j)} \in \hat{B}^{t-1}$  do
9     |  $\hat{b}_{(i,j)} = \alpha \cdot pixel_{(i,j)} + (1 - \alpha) \cdot \hat{b}_{(i,j)}$ 
10  end
11  #Segmentación del fondo
12  forall  $pixel_{(i,j)} \in I_t$  y  $\hat{b}_{(i,j)} \in \hat{B}^t$  do
13    | if  $|\hat{b}_{(i,j)} - pixel_{(i,j)}| < Th_{avg}$  then
14      |  $Etiquetas_{(i,j)} = 1$ 
15    else
16      |  $Etiquetas_{(i,j)} = 0$ 
17    end
18  end
19   $t = t + 1$ 
20 end

```

---

## 2.3 Modelación utilizando la mediana como estadístico

La mediana es un estadístico que representa el número central de un conjunto de datos ordenados, por ejemplo, considérese  $x_1, x_2, \dots, x_N$ , un conjunto de datos ordenados de forma creciente, entonces la mediana de estos datos esta dada por

$$Me = \begin{cases} x_{\frac{N+1}{2}} & \text{si } N \text{ es impar} \\ \frac{x_{\frac{N}{2}} + x_{\frac{N+1}{2}}}{2} & \text{si } N \text{ es par} \end{cases}$$

La modelación del fondo utilizando la mediana puede categorizarse dentro de los modelos de fondo de una sola moda, esto es porque mantienen información de la moda de los valores del píxel de la imagen de fondo.

En esta técnica se supone que el píxel se mantiene en el fondo más de la mitad de los últimos  $N$  frames. Debido a que es demasiado costoso calcular la mediana de todas o de las últimas  $N$  imágenes de un video, se utiliza una aproximación iterativa de la mediana, la cual fue descrita originalmente por [McFarlane and Schofield, 2005].

### Algoritmo

Sea  $pixel_{(i,j)}$  el valor del píxel en la posición  $(i, j)$  de la imagen actual y sea  $m_{(i,j)}^t$  la aproximación actual de la mediana en la posición  $(i, j)$ ,  $m_{(i,j)}^{t+1}$  es aproximada por:

$$m_{(i,j)}^{t+1} = \begin{cases} m_{(i,j)}^t + \beta & \text{si } pixel_{(i,j)} > m_{(i,j)}^t \\ m_{(i,j)}^t - \beta & \text{si } pixel_{(i,j)} < m_{(i,j)}^t \\ m_{(i,j)}^t & \text{si } pixel_{(i,j)} == m_{(i,j)}^t \end{cases} \quad (2.2)$$

donde  $\beta$  depende de la convergencia que se desee. Por ejemplo, en el trabajo propuesto por [Cheung and Kamath, 2004] se utiliza un  $\beta = 1$ , el cual es un valor grande. Considérese un píxel en escala de grises con valores en el rango  $[0, 255]$  y clasifíquese como fondo si cae dentro de  $Th_m$  unidades de la mediana. Tratemos de entender como funciona este método con el siguiente ejemplo: considérese un objeto que se mueve en un píxel  $(i, j)$  determinado. Si el objeto se mantiene en ese píxel como mínimo  $Th_m$  frames, el fondo estimado se pierde, dado que el objeto que se detuvo se vuelve parte del fondo, pero si el objeto empieza a moverse otra vez, la imagen de fondo volverá a retomarse, pero se requerirá de cierto tiempo para que el valor de fondo sea el real. Ahora, la elección del valor de  $\beta$  depende de la situación; un valor alto permite que el fondo se adapte a cambios rápidos, pero en el caso de escenas con objetos que se detienen temporalmente se debe de tomar alguna consideración.

### Implementación

Siguiendo la notación que se presentó en la ecuación 2.2, el algoritmo en pseudo-código se muestra en el algoritmo 2.



La inicialización de la mediana se puede hacer de diversas maneras:

1. La primera imagen de la secuencia del video puede ser utilizada como la mediana inicial.
2. Con el valor de 0.

---

**Algoritmo 2:** Aproximación del filtro de la mediana

---

**Entrada:**  $I^t$ ,  $\beta$ ,  $m^{t-1}$ ,  $Th_m$

**Salida :**  $m^t$ , *Etiquetas*

```

1 #Inicialización
2 forall  $pixel_{(i,j)} \in I^0$  do
3   |  $m^0_{(i,j)} = pixel_{(i,j)}$ 
4 end
5  $t = 1$ 
6 while se tenga una secuencia  $I^t$  do
7   #Aproximación de la mediana
8   forall  $pixel_{(i,j)} \in I^t$  do
9     | if  $pixel_{(i,j)} > m^{t-1}_{(i,j)}$  then
10      | |  $m^t_{(i,j)} = m^{t-1}_{(i,j)} + \beta$ 
11      | else if  $pixel_{(i,j)} < m^{t-1}_{(i,j)}$  then
12      | |  $m^t_{(i,j)} = m^{t-1}_{(i,j)} - \beta$ 
13      | end
14   end
15   #Segmentación del fondo.
16   forall  $pixel_{(i,j)} \in I_t$  do
17     | if  $|pixel_{(i,j)} - m^t_{(i,j)}| \leq Th_m$  then
18     | |  $Etiquetas_{(i,j)} = 1$ 
19     | else
20     | |  $Etiquetas_{(i,j)} = 0$ 
21     | end
22   end
23    $t = t + 1$ 
24 end
```

---

## 2.4 Modelo de mezclas

El método para la modelación del fondo utilizado en este trabajo fué introducido por [Stauffer and Grimson, 1999]. Stauffer y Grimson notaron que el desempeño computacional de las computadoras de su época, 1999, habían alcanzado un nivel donde metodos más complejos y robustos para modelado del fondo en tiempo real podrían considerarse. El enfoque es usar un modelo de mezclas para representar los estadísticos de la escena. A comparación de la aproximación de la media y mediana revisadas en la sección anterior, este método permite tener una representación multimodal del fondo, lo cual puede ser más útil, por ejemplo para remover movimientos repetitivos como lo son: lo reluciente del agua, el movimiento de las ramas, la oscilación de una bandera, etc.

### 2.4.1 Entendiendo el modelo de mezclas

Considérense los valores de un píxel como un proceso aleatorio, el cual es una serie de tiempo de los valores del píxel, por ejemplo, escalares para imágenes en escala de gris o vectores para imágenes a color. En el tiempo  $t$ , lo que se conoce de un píxel en particular  $(x_0, y_0)$  es su historia

$$\{X^1, \dots, X^t\} = \{I^i_{(x_0, y_0)} : 1 \leq i \leq t\},$$

donde  $I$  es la secuencia de imágenes.

La historia reciente de cada píxel,  $\{X^1, \dots, X^t\}$ , es modelada por una mezcla de  $K$  distribuciones Gaussianas. La probabilidad de observar el valor del píxel actual es

$$P(X^t | \Theta^t) = \sum_{i=1}^K w_i^t * \eta(X^t | \mu_i^t, \Sigma_i^t),$$

donde

$K$  Número de distribuciones gaussianas a utilizar en la mezcla

$\Theta^t$  Parámetros del modelo en el tiempo  $t$ ,  $\Theta = (\omega_1^t, \dots, \omega_K^t, \mu_1^t, \dots, \mu_K^t, \Sigma_1^t, \dots, \Sigma_K^t)$

$\omega_i^t$  Estimación del peso de la gaussiana  $i$  de la mezcla en el tiempo  $t$ , esto es qué porción de los datos han sido generados por la gaussiana  $i$

$\mu_i^t$  Media de la gaussiana  $i$  en el tiempo  $t$

$\Sigma_i^t$  Matriz de covarianza de la gaussiana  $i$  en el tiempo  $t$

$\eta()$  Función de densidad de probabilidad de la distribución gaussiana

$$\eta(X_t | \mu, \Sigma) = \frac{1}{(2\pi)^{\frac{n}{2}} \cdot |\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{1}{2} (X_t - \mu)^T \Sigma^{-1} (X_t - \mu)\right) \quad (2.3)$$

$K$  se determina de acuerdo a la memoria disponible y el poder computacional, generalmente de 3 a 5 distribuciones se utiliza. Además, por razones computacionales, la matriz de covarianza se asume de la forma:

$$\Sigma_k^t = \sigma_k^{2t} \mathbf{I}$$

Si el proceso aleatorio del píxel pudiera considerarse como un proceso estacionario, se podría utilizar un método estándar para maximizar la verosimilitud de los datos observados, el algoritmo *Expectation-Maximization (EM)*. Pero, el inconveniente es que cada proceso del píxel varía con el tiempo por lo que se necesita una aproximación del método EM. Además, debido a que hay un modelo de mezcla para cada píxel, implementar un algoritmo EM exacto en una ventana de los datos más recientes sería muy costoso, por lo cual es más factible utilizar una aproximación del algoritmo EM, llamado *on-line K-means*.

### Derivación del procedimiento de actualización de los parámetros

Ya que se decidió la representación de la función de densidad de probabilidad de  $X$ , el problema que sigue es, dado un conjunto de datos de tamaño finito de muestras independientes de  $X$ , ¿cómo elegir el vector de los parámetros que mejor representan a los datos?. Una manera común de resolver este problema es estimar los parámetros que dan la máxima verosimilitud de los datos. Recuérdese la definición del problema de estimación de máxima verosimilitud: sea  $X$  una variable aleatoria con función de densidad de probabilidad  $p(x|\Theta)$ ; además sea  $\mathcal{X} = \{x_1, \dots, x_N\}$  la que denota a un conjunto de  $N$  datos independientes que son muestras tomadas de  $X$  y sea  $\Theta$  el vector de parámetros. La función de verosimilitud es entonces,

$$\mathcal{L}(\Theta | \mathcal{X}) = \prod_{i=1}^N p(x_i | \Theta).$$

El vector de parámetros óptimo, según la máxima verosimilitud, se obtiene maximizando la función de verosimilitud con respecto a  $\Theta$ ,

$$\Theta^* = \arg \max_{\Theta} \mathcal{L}(\Theta | \mathcal{X}). \quad (2.4)$$

Resolver 2.4 es una tarea difícil, un truco usado comúnmente es maximizar la  $\log \mathcal{L}(\Theta | \mathcal{X})$  con respecto de  $\Theta$ , ya que el producto se convierte en sumatoria; la  $\log \mathcal{L}(\Theta | \mathcal{X})$  es llamada la función log-verosimilitud. El algoritmo EM puede ser de gran ayuda para resolver este problema. En el apéndice A se explica a detalle el algoritmo EM aplicado a mezcla de Gaussianas.

### Algoritmo de Stauffer y Grimson

El primer paso es estimar cual de las  $K$  distribuciones generó el valor de píxel actual  $x_t$  y actualizar los parámetros. Lo que se realiza en el método propuesto en [Stauffer and Grimson, 1999], y que es una gran aportación al estado del arte, es encontrar la **concordancia**,  $p(k | x_t, \Theta^{old})$ , que tiene el píxel  $X_t$  con cada una de las distribuciones de la mezcla y se considerará una concordancia positiva si  $X_t$  se encuentra a menos de  $2.5\sigma$  de la media  $\mu$ , lo cual puede expresarse en

términos de la distancia de Mahalanobis como

$$d_k^2 = (x_t - \mu_k)^T \Sigma_k^{-1} (x_t - \mu_k), \quad (2.5)$$

donde se tendrá **concordancia** positiva si  $d_k < 2.5$ . El valor de 2.5 nos indica un nivel de confianza del 98.76% de una distribución normal en 1 dimensión. Para generalizar lo anterior a más de una dimensión se utilizarán los valores de la función de distribución  $\chi^2$  con  $n$  grados de libertad y un nivel de confianza  $\gamma$ ; obteniéndose la **concordancia** positiva de la siguiente manera:  $d_k^2 < \chi^2_{n,\gamma}$ .

La probabilidad  $p(k | x_t, \Theta^{old})$  se aproxima utilizando el resultado obtenido de la distancia anterior y se expresa con  $q_k$  dentro de este trabajo,

$$p(k | x_t, \Theta^{old}) = q_k = \begin{cases} 1 & \text{para la distribución que tuvo la menor distancia,} \\ 0 & \text{para las demás.} \end{cases}$$

Si varias distribuciones concuerdan, se elige la distribución que maximice  $\omega_k / \|\Sigma_k\|_F$ , donde  $F$  denota la norma de Frobenius, debido a que este valor será grande para distribuciones que generan valores frecuentemente, ya que tendrán un peso grande y una varianza pequeña. Los parámetros de la distribución seleccionada serán actualizados de acuerdo a las ecuaciones A.6 - A.9. En el trabajo de Stauffer y Grimson se utiliza una aproximación de  $\rho$  de la siguiente manera:

$$\rho = \alpha \cdot \eta(X | \mu_k, \Sigma_k).$$

Notése que en el algoritmo de Stauffer y Grimson utilizan un valor de  $\rho$  diferente al resultado obtenido del algoritmo EM A.8 y del resultado que se tendría si se utilizara la aproximación de  $p(k | x_t, \Theta^{old})$  propuesto en este trabajo, ya que se tendría:

$$\rho = \begin{cases} \frac{\alpha}{\omega_{k,t}} & \text{si hubo concordancia,} \\ 0 & \text{en otro caso,} \end{cases} \quad (2.6)$$

el cual es más rápido que la aproximación utilizada por Stauffer y Grimson. Pero si se analiza la ecuación 2.6 se puede observar que en regiones con probabilidad baja, es posible que se tengan valores  $\rho \geq 1$ , lo cual ocurre si  $\omega_{k,t} \leq \alpha$  y esto provocaría introducir “ruido“ en las actualizaciones de las ecuaciones A.6-A.9. Por lo que la aproximación que utiliza Stauffer y Grimson es adecuada computacionalmente [Power and Schoonees, 2002].

### Estimación del Fondo

En el caso de que ninguna de las distribuciones concuerde, la menos probable de las distribuciones es reemplazada por una nueva distribución con valor de media  $\mu$  igual al valor del píxel actual  $X_t$ , una varianza  $\sigma^2$  alta y un peso  $\omega$  chico, así es como objetos nuevos se incorporan al fondo.

Stauffer y Grimson ordenan las distribuciones con respecto a  $\omega_k / \|\Sigma_k\|_F$ . Entonces, las primeras  $B$  distribuciones que combinadas superen un umbral  $Th$  son elegidas como modelo del fondo, esto es

$$B = \arg \min_b \left( \sum_{k=1}^b \omega_k > Th \right). \quad (2.7)$$

Para determinar la imagen binaria, se compara el conjunto de distribuciones  $B$  de la ecuación 2.7 con la que concordó el píxel actual, llámese  $\hat{k}$  a esa distribución, entonces

$$\begin{aligned} \text{si } \hat{k} \in B & \quad \text{el píxel es parte del fondo} \\ \text{si } \hat{k} \notin B & \quad \text{el píxel no es parte del fondo} \end{aligned}$$

El algoritmo 3 muestra la implementación del trabajo de Stauffer y Grimson.

---

**Algoritmo 3:** Modelo de mezclas de Stauffer y Grimson

---

**Entrada:**  $K, \alpha, \sigma_{INICIAL}^2, Th_\sigma, Th_B, \omega_{CHICO}$

**Salida :**  $\hat{B}, Etiquetas$

```

1 #Inicialización de parámetros en cada píxel
2  $\forall_{k=1\dots K} \quad \sigma_k^2 = \sigma_{INICIAL}^2 \quad \mu_k = 0 \quad \omega_k = \frac{1}{K} \quad t = 1$ 
3 while se tenga una secuencia  $I^t$  do
4   forall  $pixel_{(i,j)} \in I_t$  do
5     #Actualización de parámetros.
6     Aprendizaje_Mezclas() (ver algoritmo 4)
7     if  $match \neq 0$  then
8       | Ordenar  $\omega, \mu, \sigma$  con respecto a  $\frac{\omega_1}{\sqrt{\|\sigma_1^2\|}}, \dots, \frac{\omega_K}{\sqrt{\|\sigma_K^2\|}}$ 
9     end
10    #Segmentación del fondo
11     $B = \arg \min_b \left( \sum_{k=1}^b \omega_k > Th_B \right)$ 
12    #Imagen binaria
13    si  $\hat{k} \in B \Rightarrow Etiquetas = 1$  el píxel es parte del fondo
14    si  $\hat{k} \notin B \Rightarrow Etiquetas = 0$  el píxel no es parte del fondo
14  end
15 end

```

---

---

**Algoritmo 4:** Aprendizaje\_Mezclas()

---

**Entrada:**  $pixel_{(i,j)}$ ,  $\omega$ ,  $\mu$ ,  $\sigma^2$ ,  $Th_\sigma$ ,  $K$ ,  $\alpha$ **Salida :**  $\omega$ ,  $\mu$ ,  $\sigma^2$ ,  $Th_\sigma$ ,  $\hat{k}$ ,  $match$ 

```

1  match = 0
2  forall  $k \in [1, \dots, K]$  do
3       $d_k^2 = \sum_{d=1}^n \frac{(pixel_{(i,j,d)}^t - \mu_{k,d})^2}{\sigma_{k,d}^2}$ 
4      if  $d_k < Th_\sigma$  then
5          if match == 0 then
6               $\hat{k} = k$ 
7              match = 1
8          else if  $\frac{\omega_k}{\sqrt{\|\sigma_k^2\|}} > \frac{\omega_m}{\sqrt{\|\sigma_m^2\|}}$  then
9               $\hat{k} = k$ 
10         end
11     end
12 end
13 if match == 1 then
14     forall  $k \in [1, \dots, K]$  &  $k \neq \hat{k}$  do
15          $\omega_{\hat{k}} = (1 - \alpha) \cdot \omega_{\hat{k}}$ 
16     end
17      $\omega_{\hat{k}} = (1 - \alpha) \cdot \omega_{\hat{k}} + \alpha$ 
18      $\rho_{\hat{k}} = \alpha \cdot p(\hat{k} | pixel_{(i,j)}, \Theta^{old})$ 
19      $\mu_{\hat{k}} = (1 - \rho_{\hat{k}}) \cdot \mu_{\hat{k}} + \rho_{\hat{k}} \cdot pixel_{(i,j)}$ 
20      $\sigma_{\hat{k}}^2 = (1 - \rho_{\hat{k}}) \cdot \sigma_{\hat{k}}^2 + \rho_{\hat{k}} \cdot (pixel_{(i,j)} - \mu_{\hat{k}})^T (pixel_{(i,j)} - \mu_{\hat{k}})$ 
21 else
22     #Asúmase que las distribuciones estan ordenadas de manera
23     decreciente de acuerdo a  $\frac{\omega_1}{\sqrt{\|\sigma_1^2\|}}$ 
24      $\hat{k} = K$ 
25      $\omega_{\hat{k}} = \omega_{CHICO}$ 
26      $\mu_{\hat{k}} = pixel_{(i,j)}$ 
27      $\sigma_{\hat{k}}^2 = \sigma_{INICIAL}^2$ 
28 end
29 #Normalización de los pesos  $\omega$ 
 $\forall_{k=1 \dots K} \quad \omega_k = \frac{\omega_k}{\sum_{k=1}^K \omega_k}$ 

```

---

### 2.4.2 Mezcla de gaussianas con velocidad de aprendizaje adaptativa

Un problema común en los modelos adaptativos de mezclas de gaussianas es equilibrar entre la rapidez de convergencia y estabilidad. La mayoría de las aplicaciones que involucran la mezcla de gaussianas son entrenadas con alguna variante del algoritmo EM. Las aplicaciones de vigilancia necesitan de operaciones en tiempo real, por lo que se prohíbe el uso del EM normal [Dempster et al., 1977] o alguna variante incremental [Neal and Hinton, 1999]. Lo que se necesita es una aproximación online que aprenda incrementalmente con cada nueva observación y que se adapte sin la necesidad de grandes requerimientos de memoria.

El trabajo de [Stauffer and Grimson, 1999] utiliza una aproximación online del algoritmo EM, basada en el filtro recursivo que se mencionó en 2.4.1, para entrenar el modelo de la mezcla de gaussianas (MoG). La velocidad de aprendizaje es controlada por un parámetro global  $\alpha \in [0, 1]$ . En el artículo mencionado se utiliza un valor de  $\alpha$  fijo muy pequeño, esto es para tener una estabilidad en el modelo; desafortunadamente esto resulta en una convergencia muy lenta cada vez que una distribución se adapta a un nuevo grupo o cluster; afectando la detección de cambios en el fondo y no fondo. Si se elige un valor de  $\alpha$  muy grande, podría mejorar la velocidad de convergencia, pero el modelo resultaría muy sensible al ruido y conservaría muy poco de la historia acumulada.

En [Lee, 2005] se propone un esquema efectivo para mejorar la velocidad de convergencia sin comprometer la estabilidad del modelo. Esto se logra reemplazando el factor global y estático de aprendizaje por una velocidad de aprendizaje calculada para cada gaussiana en cada frame.

La aproximación del aprendizaje de mezclas de manera recursiva y con una velocidad de aprendizaje se puede formular de la siguiente manera:

$$\nu(t) = (1 - \rho(t)) \cdot \nu(t - 1) + \rho(t) \cdot \nabla(x(t); \nu(t - 1)),$$

donde el modelo al tiempo  $t$ ,  $\nu(t)$ , es actualizado utilizando una estimación a partir de los datos actuales  $\nabla(x(t); \nu(t - 1))$  utilizando una velocidad que es controlada por  $\rho(t)$ .

Si la velocidad  $\rho(t) = 1/t$ , [Friedman and Russell, 1997], [Sato and Ishii, 2000], los parámetros actuales son actualizados tomando en cuenta el número de observaciones hasta el momento actual, permitiendo aproximarse rápidamente al valor esperado y converger a la estimación óptima en un número infinito de observaciones en una **distribución estacionaria**. Si  $\rho(t) = \alpha$  fija, como en [Harville et al., 2001], [Stauffer and Grimson, 1999] la estimación de los parámetros reflejaría la información de las observaciones más actuales dentro de un tiempo de  $L = 1/\alpha$  con un deterioro exponencial, sin embargo, esto le toma demasiado tiempo a los parámetros para converger.

Una solución que se presenta en [Lee, 2005] para lograr una convergencia rápida y adaptativa es utilizar una velocidad de aprendizaje tal que  $\rho(t)$  converja a  $\alpha$  en lugar de 0 y calcular  $\rho(t)$  para cada gaussiana basado en su estimación

actual de la verosimilitud de una manera que sea consistente con el algoritmo EM. Para tomar en cuenta, que no todas las gaussianas son actualizadas con cada dato y que pueden ser reiniciadas,  $\rho(t)$  es calculado para cada gaussiana independientemente de la estimación esperada de la verosimilitud acumulada.

El algoritmo base en [Lee, 2005] sigue la formulación propuesta en el trabajo de [Stauffer and Grimson, 1999] con algunas diferencias, entre las más importantes se encuentran:

1. El cálculo de la velocidad de aprendizaje  $\rho_k$ ,

$$\rho_k = q_k \cdot \left( \frac{1 - \alpha}{c_k} + \alpha \right),$$

donde  $q_k$  representa la probabilidad posterior de la gaussiana  $k$ .

2. La introducción de la variable  $c_k$ , que cuenta el número de observaciones efectivas de la gaussiana  $k$  hasta el tiempo actual; esta variable servirá para la actualización de la velocidad  $\rho_k$ ; además esta variable se reinicia a 1 cuando la distribución anterior se haya deteriorado y una nueva distribución se haya creado debido al dato actual.
3. Por razones de eficiencia, se introduce la opción *Winner\_Take\_All* ya que generalmente la “mejor” distribución es seleccionada para la actualización de sus parámetros, pero si la mezcla contiene dos grupos significativos, la estrategia utilizada puede guiar a una de las distribuciones a la extinción debido a que a una de las distribuciones se le da mayor peso para dominar a las otras. Así que una solución es que todas las gaussianas que concuerden con el dato actual sean actualizadas de manera proporcional a la probabilidad posterior estimada  $P(k | x)$  [Nowlan, 1991].
4. Se modifica la manera de actualizar  $\hat{P}(F | k)$ .

Se puede observar que  $c_k$ , la cual es la suma de la probabilidad posterior esperada de la gaussiana  $k$ , es uno de los estadísticos calculados en el algoritmo EM incremental

$$c_k^t = \sum_{i=1}^t q_k^i = \sum_{i=1}^t P(k | x^i, \theta^{i-1}).$$

Esto sirve como reloj individual para el aprendizaje de la actualización de cada gaussiana. De las ecuaciones de actualización de la velocidad de aprendizaje es claro que la etapa inicial de aprendizaje, cuando solo pocas muestras han sido observadas,  $\rho_k \approx 1/c_k$ , y la actualización de los parámetros se aproximan mucho al valor esperado calculado con los estadísticos del EM incremental. Mientras más muestras son incluidas en la estimación,  $\rho_k$  se aproxima a  $\alpha$  y se comporta como un aprendizaje recursivo típico. El algoritmo de [Lee, 2005] difiere en la actualización del parámetro  $\rho_k$  con los métodos existentes en lo siguiente:

- En el algoritmo reportado en [Stauffer and Grimson, 1999] se usa una velocidad de aprendizaje  $\rho_k(t) = \alpha \cdot \eta(X | \mu, \Sigma)$ .



- Dado que  $\eta(X | \mu, \Sigma) = P(x | k)$ , es un valor muy pequeño, se presenta una convergencia más lenta a si se utilizara sólomente  $\rho(t) = \alpha$ , [Harville et al., 2001], [KaewTraKulPong and Bowden, 2001].
- Otra modificación razonable es usar  $\rho_k(t) = \alpha \cdot P(k | x)$ , pero se tendría la misma convergencia a si se utiliza  $\alpha$  estática.
- En el método propuesto en [KaewTraKulPong and Bowden, 2001], el aprendizaje en la etapa inicial es calculado directamente de los estadísticos del EM incremental para obtener  $\rho_k(t) = q_k^t / c_k^t$ . Después de las primeras  $L = 1/\alpha$  muestras, el aprendizaje cambia a la regla de ventana-L, lo cual se puede ver que es equivalente a:

$$\begin{aligned}\mu_k^t &= (1 - \alpha) \cdot \mu_k^{t-1} + \rho_k(t-1) \cdot x^t, \\ \rho_k(t) &= \alpha \cdot \frac{q_k^t}{\omega_k^t},\end{aligned}$$

dado que  $\omega_k^t = \frac{c_k^t}{t}$ , entonces  $\rho_k(t) \approx \alpha \cdot \frac{t}{c_k^t}$ .

Sin embargo, estas ecuaciones sólo se aplican en la etapa de actualización y no beneficia a distribuciones nuevas o reasignadas que se adaptan a nuevos clusters. Además, como  $\eta_k$  no tiene límites y  $(1 - \alpha) + \eta_k$  puede exceder al valor 1 en la regla de ventana-L, el algoritmo algunas veces muestra un comportamiento divergente. En contraste, el algoritmo de [Lee, 2005] considera cada gaussiana separadamente y aplica la velocidad de aprendizaje apropiada independientemente de la fase de aprendizaje en la que se encuentre.

La segmentación del fondo se refiere a un problema de clasificación binaria basada en  $P(F | X)$ , donde  $x$  es el valor del píxel en el tiempo  $t$ , y  $F$  representa la clase Fondo. Con la representación de la distribución  $P(X)$  como una mezcla,

$$P(X) = P(X | \Theta) = \sum_{k=1}^K P(k)P(x | k) = \sum_{k=1}^K \omega_k \cdot \eta(X | \mu, \Sigma),$$

la probabilidad posterior puede ser expresada en término de los componentes gaussianos  $P(k)$  y  $P(X | k)$ , y una estimación de la densidad  $P(F | k)$  como sigue:

$$P(F | x) = \sum_{k=1}^K P(F | k)P(k | x) = \frac{\sum_{k=1}^K P(x | k)P(k)P(F | k)}{\sum_{k=1}^K P(x | k)P(k)}. \quad (2.8)$$

Los componentes de la mezcla son obtenidos con el algoritmo de aprendizaje descrito en [Lee, 2005]. La estimación de  $P(F | k)$  envuelve sin excepción conocimiento heurístico del desconocido proceso de fondo.  $P(F | k) = 1$  en el artículo propuesto por [Stauffer and Grimson, 1999], para todas las gaussianas

con un valor grande de  $\omega/\sigma$ , para un cierto porcentaje de las observaciones y 0 para las otras, dicho en otras palabras, todas las distribuciones que estan en el conjunto  $B$ , donde  $B$  se obtiene con la ecuación 2.7. En este trabajo se entrena una función sigmoïdal en  $\omega/\sigma$  para aproximar la  $P(F | k)$  usando regresión logística:

$$\hat{P}(F | k) = f\left(\frac{\omega_k}{\sigma_k}; a, b\right) = \frac{1}{\left(1 + e^{-a \cdot \frac{\omega_k}{\sigma_k} + b}\right)}$$

Una vez que  $P(X)$  y  $P(F | k)$  son estimadas, la región del primer plano es compuesta de píxeles donde  $P(F | x) < 0.5$ .

Para obtener una representación visual del modelo de fondo, una solución intuitiva es calcular el valor de la esperanza de las observaciones que son fondo:

$$E[X | F] = \sum_{k=1}^K E[X | k] \cdot P(k | F) = \frac{\sum_{k=1}^K \mu_k \cdot P(F | k)P(k)}{\sum_{j=1}^K P(F | j)P(j)} \quad (2.9)$$

El pseudocódigo se muestra en el algoritmo 5.

**Algoritmo 5:** Modelo de mezclas efectivas para mezcla de gaussianas**Entrada:**  $K, \sigma_{INICIAL}^2, \alpha, Th_\sigma, Th_F$ **Salida :** *Etiquetas*

```

1 #Inicialización de parámetros en cada píxel
2  $\forall_{j=1\dots K} \quad \omega_j = 0 \quad \mu_j = \text{inf} \quad \sigma_j = \sigma_{INICIAL} \quad c_j = 0 \quad t = 0$ 
3 while se tenga una secuencia  $I^t$  do
4   forall  $pixel_{(i,j)} \in I_t$  do
5     #Obtener la concordancia del dato con respecto a cada
     una de las distribuciones.
6      $\forall_{j=1\dots K} \quad p_j = \begin{cases} \omega_j \cdot \eta_j(pixel_{(i,j)}; \mu_j, \sigma_j) & \text{si } \frac{|pixel_{(i,j)} - \mu_j|}{\sigma_j} < Th_\sigma \\ 0 & \text{en otro caso} \end{cases}$ 
7     if  $\sum_{j=1}^K p_j > 0$  then
8       forall  $k = 1 \dots K$  do
9         #Probabilidad posterior de gaussiana  $k$ 
10         $q_k = \frac{p_k}{\sum_{j=1}^K p_j}$ 
11        if Winner Take All then
12           $q_k = \begin{cases} 1 & \text{si } k = \arg \max_j \{p_j\} \\ 0 & \text{en otro caso} \end{cases}$ 
13        end
14         $\omega_k = (1 - \alpha) \cdot \omega_k + \alpha \cdot q_k$ 
15        if  $q_k > 0$  then
16           $c_k = c_k + q_k$ 
17           $\rho_k = q_k \cdot \left( \frac{1-\alpha}{c_k} + \alpha \right)$ 
18           $\mu_k = (1 - \rho_k) \cdot \mu_k + \rho_k \cdot pixel_{(i,j)}$ 
19           $\sigma_k = (1 - \rho_k) \cdot \sigma_k^2 + \rho_k \cdot (pixel_{(i,j)} - \mu_k)^2$ 
20        end
21      end
22    else
23       $\forall_{j=1\dots K} \quad \omega_j = (1 - \alpha) \cdot \omega_j$ 
24       $\hat{k} = \arg \min_j \{\omega_j\} \quad \omega_{\hat{k}} = \alpha \quad \mu_{\hat{k}} = pixel_{(i,j)} \quad \sigma_{\hat{k}} =$ 
25       $\sigma_{INICIAL} \quad c_{\hat{k}} = 1$ 
26    end
27    #Normalización de los pesos  $\omega$ 
28     $\forall_{j=1\dots K} \quad \omega_k = \frac{\omega_k}{\sum_{k=1}^K \omega_k}$ 
29    #Segmentación del fondo
30    Si  $\begin{cases} P(F | pixel_{(i,j)}) > Th_F & \text{Etiquetas}_{(i,j)} = \text{Es fondo} \\ P(F | pixel_{(i,j)}) \leq Th_F & \text{Etiquetas}_{(i,j)} = \text{Es No Fondo} \end{cases}$ 
31     $t = t + 1$ 
32 end

```

### 2.4.3 Número de distribuciones del modelo de mezcla

En el trabajo de [Stauffer and Grimson, 1999] y [Lee, 2005] se utiliza una modelo de mezcla de  $K$  gaussianas, donde  $K$  es una constante predefinida para cada píxel. El método propuesto en [Richardson and Green, 1997], el cual utiliza cadenas de Markov, utiliza un modelo de mezclas con un número desconocido de componentes, pero no existe una versión para tiempo real. En el trabajo [Tan et al., 2006] se propone una manera de tener un modelo de mezclas con un número de componentes,  $K$ , adaptativo. Como resultado, regiones complicadas son descritas con un mayor número de componentes y regiones simples con pocos componentes.

Debido a los requerimientos de tiempo real que se requiere, se utiliza una versión del algoritmo EM-online. En esta variante del EM, se consideran tres estadísticos:

- $N_k$ , el cual representa el número de muestras que pertenecen al componente gaussiano  $C_k$ ,
- $S_k$ , es la suma de las muestras que pertenecen al componente gaussiano  $C_k$

$$S_k = \sum_{x_{(i,j)} \in C_k} x_{(i,j)},$$

- $Z_k$ , representa la suma del producto externo de las muestras que pertenecen a  $C_k$ .

$$Z_k = \sum_{x_{(i,j)} \in C_k} \frac{x_{(i,j)}^2}{n},$$

donde  $k$  denota la distribución  $k$ -ésima de la mezcla de Gaussianas del píxel actual. Como consecuencia, los parámetros del modelo pueden ser calculados a partir de estos tres estadísticos como sigue:

$$\begin{aligned} \omega_k &= \frac{N_k}{\sum_k^K N_k} \\ \mu_k &= \frac{S_k}{N_k} \\ \sigma_k^2 &= \frac{1}{N_k} Z_k - \mu_k^2. \end{aligned} \tag{2.10}$$

Cuando una nueva muestra  $x_{(i,j)}$  llega, los tres estadísticos son actualizados como sigue:

$$\begin{aligned} N_k^t &= N_k^{t-1} + P(X \in C_i | X = x_{(i,j)}, \Theta^{t-1}), \\ S_k^t &= S_k^{t-1} + x_{(i,j)}^t P(X \in C_k | X = x_{(i,j)}, \Theta^{t-1}), \\ Z_k^t &= Z_k^{t-1} + x_{(i,j)}^{t2} P(X \in C_k | X = x_{(i,j)}, \Theta^{t-1}), \end{aligned} \tag{2.11}$$

donde

$$\begin{aligned} P(X \in C_k | X = x_{(i,j)}, \Theta^{t-1}) &= \frac{P(X \in C_k, X = x_{(i,j)} | \Theta)}{P(X = x_{(i,j)} | \Theta)} \\ &= \frac{\omega_k \cdot \eta(x_{(i,j)} | \Theta_k)}{\eta(x_{(i,j)} | \Theta^{t-1})}, \end{aligned}$$

para iniciar el algoritmo se eligen valores de los tres estadísticos  $\{N_k^0, S_k^0, Z_k^0\}_{k=1}^K$  y al actualizar estos estadísticos, pueden obtener los parámetros  $\Theta$ .

La modelación del fondo con número de distribuciones de la mezcla adaptativo se realiza de la siguiente manera: cuando el primer frame del video se tiene, es creado un nuevo componente gaussiano por cada píxel, los cuales tienen como media el valor del píxel actual, una varianza alta y un peso chico. Con cada valor nuevo que se tenga, si se encontró concordancia (la concordancia se maneja tal cual se manejó en el trabajo de Stauffer y Grimson), se actualiza el modelo utilizando 2.11; si no se encuentra alguna concordancia, un nuevo componente gaussiano es creado y no se dispone de algún componente existente.

Entonces, dos problemas surgen: esos estadísticos,  $\{N_k, S_k, Z_k\}$ , incrementan ilimitadamente mientras más imágenes son procesadas;  $K$  también incrementa ilimitadamente en un píxel en particular; así que el costo computacional incrementa drásticamente. Primero, si  $\sum_{k=1}^K N_k^{t-1} < L$ ,  $\{N_k, S_k, Z_k\}$  son actualizados utilizando 2.11; en otro caso se define una velocidad de olvido como sigue:

$$\delta = \frac{\sum_{k=1}^K N_k^{t-1}}{\sum_{k=1}^K N_k^{t-1} + 1},$$

entonces, los estadísticos son actualizados de la siguiente manera:

$$\begin{aligned} N_k^t &= \delta [N_k^{t-1} + P(X \in C_k | X = x_{(i,j)}, \Theta^{t-1})], \\ S_k^t &= \delta [S_k^{t-1} + x_{(i,j)}^t P(X \in C_k | X = x_{(i,j)}, \Theta^{t-1})], \\ Z_k^t &= \delta [Z_k^{t-1} + x_{(i,j)}^{t^2} P(X \in C_k | X = x_{(i,j)}, \Theta^{t-1})], \end{aligned} \quad (2.12)$$

Como resultado  $\sum_{k=1}^K N_k$  será una constante cercana a  $L$ .

Segundo, cada  $L$  imágenes, el peso de cada componente gaussiano, esto es  $\omega_k$ , es comparado con un umbral predefinido  $Th_\omega$ . Si  $\omega_k < Th_\omega$  el componente  $C_k$  es eliminado ya que la desigualdad indica que hay poca evidencia que respalde a ese componente como modelo del fondo. Un valor razonable para  $Th_\omega$  es  $1/L$ , ya que el componente es soportado por menos de una evidencia en  $L$  frames.

La estimación del fondo se realiza de la misma manera que el trabajo de Stauffer y Grimson. El pseudo-código se muestra en el algoritmo 6.

---

**Algoritmo 6:** Modelo de mezclas adaptativas

---

**Entrada:**  $I^t, N_k^0, S_k^0, Z_k^0, \omega_{CHICO}, \sigma_{GRANDE}^2, L$ **Salida :** *Etiquetas*

```

1 #Inicialización
2  $K = 1$ 
3  $Th_\omega = \frac{1}{L}$ 
4  $N_k^0, S_k^0, Z_k^0$ 
5 forall  $pixel_{(i,j)} \in I^t$  do
6    $\omega_1 = \omega_{CHICO}$ 
7    $\mu_1 = pixel_{(i,j)}$ 
8    $\sigma_1^2 = \sigma_{GRANDE}^2$ 
9    $t = 0$ 
10 end
11 while se tenga una secuencia  $I^t$  do
12   forall  $pixel_{(i,j)} \in I^t$  do
13     if  $\sum_{k=1}^K N_k < L$  then
14       Actualizar los estadisticos del EM como en 2.11
15     else
16        $\delta = \frac{\sum_{k=1}^K N_k}{\sum_{k=1}^K N_{k+1}}$ 
17       Actualizar los estadisticos del EM como en 2.12
18     end
19     if  $t \bmod L == 0$  then
20       forall  $k = 1 \dots K$  do
21         if  $\omega_k < Th_\omega$  then
22           Se elimina el componente gaussiano  $k$ 
23         end
24       end
25     end
26     #Actualización de parámetros.
27     Aprendizaje_Mezclas_Dinamicas() (ver algoritmo 7)
28      $B = \arg \min_b \left( \sum_{k=1}^b \omega_k > T \right)$ 
29     si  $\hat{k} \in B \Rightarrow Etiquetas_{(i,j)} = 1$  el píxel es parte del fondo
30     si  $\hat{k} \notin B \Rightarrow Etiquetas_{(i,j)} = 0$  el píxel no es parte del fondo
31   end
32    $t = t + 1$ 
33 end

```

---

---

**Algoritmo 7:** Aprendizaje\_Mezclas\_Dinamicas()
 

---

**Entrada:**  $pixel_{(i,j)}$ ,  $\omega$ ,  $\mu$ ,  $\sigma^2$ ,  $Th_\sigma$ ,  $K$ 
**Salida :**  $\omega$ ,  $\mu$ ,  $\sigma^2$ ,  $\hat{k}$ ,  $K$ 

```

1 match = 0
2 forall  $k \in [1, \dots, K]$  do
3    $d_k^2 = \sum_{d=1}^n \frac{(pixel_{(i,j,d)}^t - \mu_{k,d})^2}{\sigma_{k,d}^2}$ 
4   if  $d_k < Th_\sigma$  then
5     if match == 0 then
6        $\hat{k} = k$ 
7       match = 1
8     else if  $\frac{\omega_k}{\sqrt{\|\sigma_k^2\|}} > \frac{\omega_m}{\sqrt{\|\sigma_m^2\|}}$  then
9        $\hat{k} = k$ 
10    end
11  end
12 end
13 if match == 1 then
14   #Se actualizan los parámetros con ecuación 2.10
15 else
16   #Se crea un nuevo componente gaussiano
17    $K = K + 1$ 
18    $\hat{k} = K$ 
19    $\omega_{\hat{k}} = \omega_{CHICO}$ 
20    $\mu_{\hat{k}} = pixel_{(i,j)}$ 
21    $\sigma_{\hat{k}}^2 = \sigma_{GRANDE}^2$ 
22 end

23 #Normalización de los pesos  $\omega$ 
24  $\forall_{k=1 \dots K} \quad \omega_k = \frac{\omega_k}{\sum_{k=1}^K \omega_k}$ 

```

---

### 2.4.4 Método en Cascada

El trabajo [Teixeira and Corte-Real, 2007] utiliza en conjunto de pruebas en cascada para la detección de cambios, entre las pruebas que integra se tienen la detección de ruido, la variación en la luminosidad y la detección de cambios estructurales. Las pruebas de ruido y variación en la iluminación se explican más a detalle en la sección 3.1 y el algoritmo que utiliza para el aprendizaje de objetos en movimientos es el algoritmo de mezclas efectivas 5. El pseudo-código se muestra en el algoritmo 8.

---

#### Algoritmo 8: Algoritmo en cascada

---

**Entrada:**  $\alpha, K, T_N, T_I, T_F$

**Salida :**  $\hat{B}, Etiquetas$

```

1 while se tenga una secuencia  $I^t$  do
2   forall  $pixel_{(i,j)} \in I^t$  do
3     #Clasificacion
4     if  $Prueba\_Ruido() > T_N$  then
5       | Clasifíquese el píxel como FONDO
6     else if  $Prueba\_Ruido() > 10^{-10}T_N$  then
7       | if  $Prueba\_Iluminacion() > T_I$  then
8         | | Clasifíquese el píxel como FONDO
9         | end
10      else if  $P(F | X)(2.8) > T_F$  then
11        | Clasifíquese el píxel como FONDO
12      else
13        | Clasifíquese el píxel como NO_FONDO
14      end
15      #Actualización del fondo
16      Actualizar  $\omega_k, \mu_k$  y  $\sigma_k$  como en el algoritmo 5.
17      Actualizar la imagen de referencia utilizando 2.9
18    end
19     $t = t + 1$ 
20 end
```

---



### 2.4.5 Otras variantes

Es importante mencionar algunas variantes de las mezclas de gaussianas y del trabajo analizado en esta sección que existe en la literatura. El trabajo de [Heikkilä et al., 2004] se basa en el algoritmo de [Stauffer and Grimson, 1999] para realizar el modelo del fondo, pero en lugar de utilizar el vector de color o valor de niveles de gris se utiliza una medida de textura: LBP (Local Binary Pattern); donde cada bloque de la imagen es modelado como un grupo de histogramas adaptativos ponderados de LBP's, se utiliza esta medida debido a que es invariante a cambios monotónicos en escala de grises.

El trabajo de [Teixeira et al., 2007] está basado completamente en el trabajo de *Cascada* mencionado en la sección 2.4.4, el único cambio que existe es el algoritmo de aprendizaje de mezclas que se utiliza; mientras que en *Cascada* se utiliza el algoritmo de mezclas efectivas 5 en el otro se utiliza el algoritmo de filtro de media<sup>1</sup>.

En las tablas 2.1, 2.2, 2.3 y 2.4 se muestran los resultados obtenidos con los métodos de la media, mediana, MoG de Stauffer, MoG efectivas y MoG con pruebas en *Cascada*.



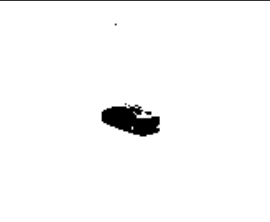



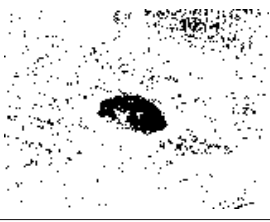
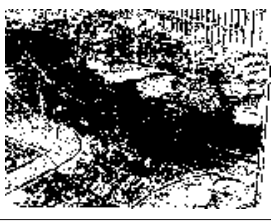


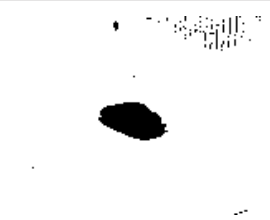

	Secuencias PETS2000	
	Seq-146	Seq-816
Real		
M. Media		
M. Mediana		
MoG Stauffer		
MoG Efectivas		
Cascada		

Tabla 2.1: Detección de objetos en movimiento de los métodos media, mediana, MoG Stauffer, MoG efectivas, MoG en cascada en secuencias *PETS2000*






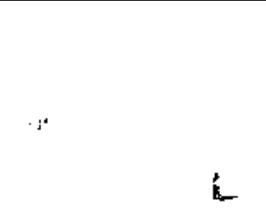
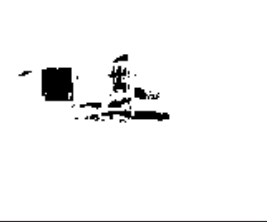

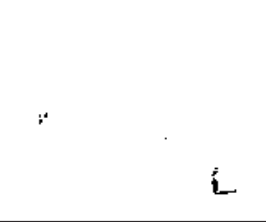


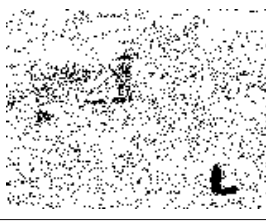

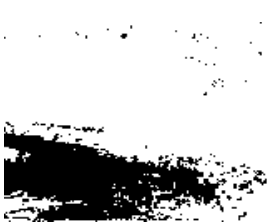
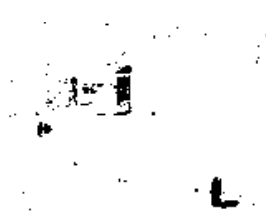


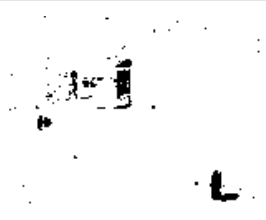
Secuencias PETS2001_DS3_TE1_C1			
	Seq-201	Seq-824	Seq-2096
Real			
M. Media			
M. Mediana			
MoG Stauffer			
MoG Efectivas			
Cascada			

Tabla 2.2: Detección de objetos en movimiento de los métodos media, mediana, MoG Stauffer, MoG efectivas, cascada en secuencias *PETS2001\_DS3\_TE1\_C1*



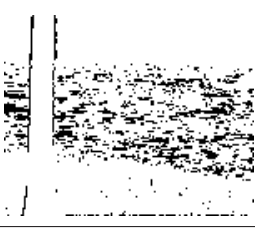
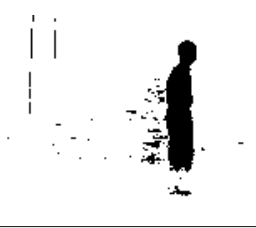
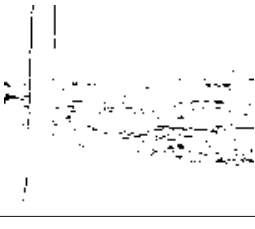
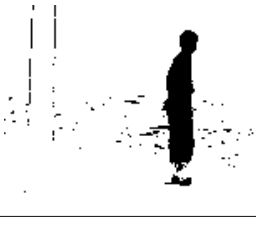
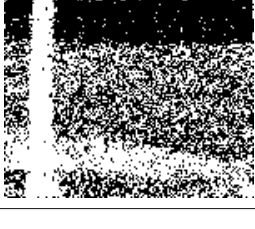
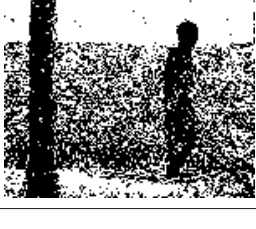
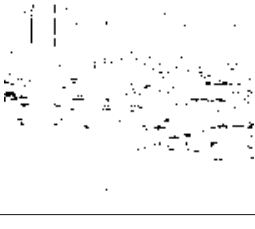
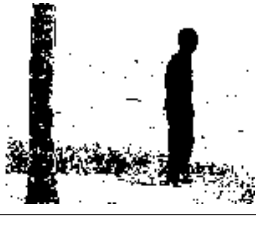
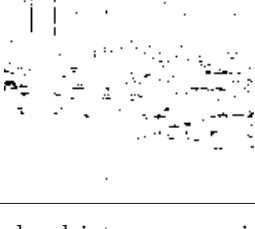

Secuencias WaterSurface		
	Seq-324	Seq-598
Real		
M. Media		
M. Mediana		
MoG Stauffer		
MoG Efectivas		
Cascada		

Tabla 2.3: Detección de objetos en movimiento de los métodos media, mediana, MoG Stauffer, MoG efectivas, cascada en secuencias *WaterSurface*













		Secuencias HighWay1	
		Seq-100	Seq-365
Real			
M. Media			
M. Mediana			
MoG Stauffer			
MoG Efectivas			
Cascada			

Tabla 2.4: Detección de objetos en movimiento de los métodos media, mediana, MoG Stauffer, MoG efectivas, cascada en secuencias *HighWay1*



## Capítulo 3

---

# Factores externos al movimiento

---

La detección de objetos en movimiento en una secuencia de imágenes se ve afectada generalmente por varios factores: cambios en la iluminación, ruido, movimientos repetitivos del fondo, sombras, etc. Lo que éstos ocasionan es que píxeles que pertenecen al fondo sean mal clasificados como píxeles en movimiento y esto se debe a que el valor del píxel actual es modificado drásticamente.

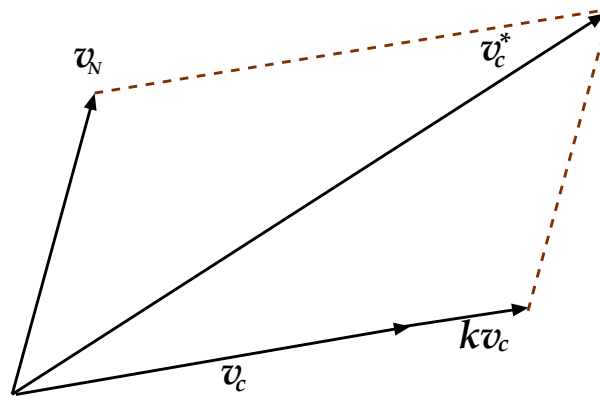


Figura 3.1: Efectos de variación en iluminación y ruido aditivo

Por ejemplo, obsérvese en la figura 3.1 el efecto que ocasionan los factores de ruido e iluminación en el píxel de color real  $v_c$ , dando como resultado el píxel de color observado  $v_c^*$ , donde  $v_N$  denota el vector aditivo del factor ruido y  $k$  es el factor multiplicativo de iluminación.

En este capítulo se mencionan algunas técnicas utilizadas para detectar la influencia de la iluminación, ruido, cambios rápidos y sombras en una secuencia de imágenes.

### 3.1 Identificación de variación de la iluminación y ruido

Debido a los problemas que ocasionan la iluminación y el ruido en la exacta detección de objetos en movimiento, diferentes enfoques han sido propuestos para detectar píxeles que han sido afectados por estos factores: geopixel [Jain et al., 1977], QPF (Quadratic Picture Image) [Hsu et al., 1984], Shading Model [Skifstad and Jain, 1989], Momentos Circulares Recorridos (Circular Shift Moments) [Liu et al., 1998], etc. Existe gran variedad de técnicas para resolver estos problemas, en esta sección se mencionan algunas de ellas.

En el método cascada [Teixeira and Corte-Real, 2007] se asume que la variación en la iluminación es ocasionada por un factor multiplicativo  $k$  que afecta al píxel real, entonces el píxel resultante es un vector de color necesariamente colineal al vector de referencia. La prueba de colinealidad consiste en evaluar el ángulo entre el píxel actual  $v^c$  y el píxel de referencia  $v^r$ ,

$$\cos \theta = \frac{v^c \cdot v^r}{\|v^c\| \|v^r\|}. \quad (3.1)$$

Si el  $\cos \theta$  es mayor a un umbral  $Th_I \approx 1$ , el vector es considerado como **colineal**, ocasionando que se afirme que el cambio detectado en estos píxeles es ocasionado por algún cambio en la iluminación. En la tabla 3.1 se muestra el resultado de esta prueba en algunas secuencias de imágenes, los píxeles de las imágenes del segundo renglón son aquellos que el algoritmo identificó como objetos en movimiento; los píxeles de color rojo expresan que la prueba de iluminación los detectó como píxeles colineales de aquí que se eliminan como píxeles en movimiento y se identifican como píxeles que fueron afectados por una variación en la iluminación; los píxeles azules no fueron identificados como colineales, por lo tanto se consideran píxeles en movimiento. El tercer renglón muestra los píxeles en movimiento despues de la prueba de variación en la iluminación.

[Mester et al., 2001] modelan la imagen obtenida del brillo como el producto de la iluminación y la reflectancia sobre la superficie de los objetos; además se asume que la iluminación es generalmente un función que varía lentamente en el espacio. Para detectar el cambio, se compara la imagen actual con la imagen de fondo en escala de grises; si no existe cambio estructural en la escena, las diferencias observadas pueden ser causadas por:

- un factor multiplicativo positivo  $k$  el cual modula la señal.
- ruido que puede ser modelado como ruido gaussiano o laplaciano independiente idénticamente distribuido (i.i.d.).

Considérese el caso en el que la hipótesis nula  $H_0$  (*colinealidad*) es verdadera. Ordénense los valores de las ventanas  $W_1$  y  $W_2$  en los vectores columna  $x_1$  y  $x_2 \in \mathbb{R}^N$ ; donde las ventanas  $W_1, W_2$  están centradas en el píxel bajo evaluación de la imagen actual y fondo respectivamente y  $N$  es el número de píxeles de las






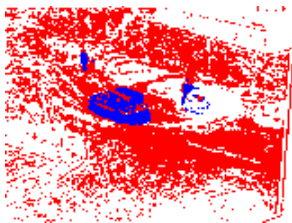
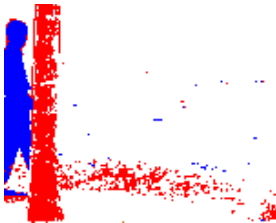
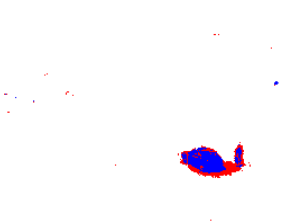



	Pets2000	WaterSurface	PETS2001_DS1_TE1_C1
Imagen Real			
Iluminación			
Movimiento			

Tabla 3.1: Variación en la luminosidad

ventanas.

Los vectores que contienen los valores de la vecindad de la imagen actual y de fondo,  $x_1, x_2$  respectivamente, pueden ser representados como  $x_1 = s + \varepsilon_1$  y  $x_2 = k \cdot s + \varepsilon_2$ , donde  $\varepsilon_i, i = 1, 2$  son vectores aditivos de ruido con

$$\begin{aligned} E[\varepsilon_1] &= E[\varepsilon_2] = 0, \\ Cov[\varepsilon_1] &= Cov[\varepsilon_2] = \sigma_d^2 \cdot I_N \end{aligned} \quad (3.2)$$

y sea  $s$  el vector de la señal. Desafortunadamente, el vector de la señal  $s$  es desconocido. En un caso ideal,  $x_1$  y  $x_2$  son paralelos dado  $H_0$ . Así que se formulará la detección de cambio probando si  $x_1$  y  $x_2$  son alguna versión de vectores paralelos, con algun factor  $k$ .

Enfoques anteriores y más simples han probado la colinealidad enfocándose en la diferencia entre los dos vectores observados  $x_i$  o utilizando la diferencia angular entre ellos. Sin embargo, basar la decisión de detección de cambio en el ángulo entre vectores (por ejemplo, usar el coeficiente de correlación normalizado) no es recomendado.

El enfoque que se propone en [Mester et al., 2001] se ilustra en la figura 3.2. Dadas las observaciones  $x_i, i = 1, 2$  y asumiendo ruido gaussiano i.i.d., la esti-

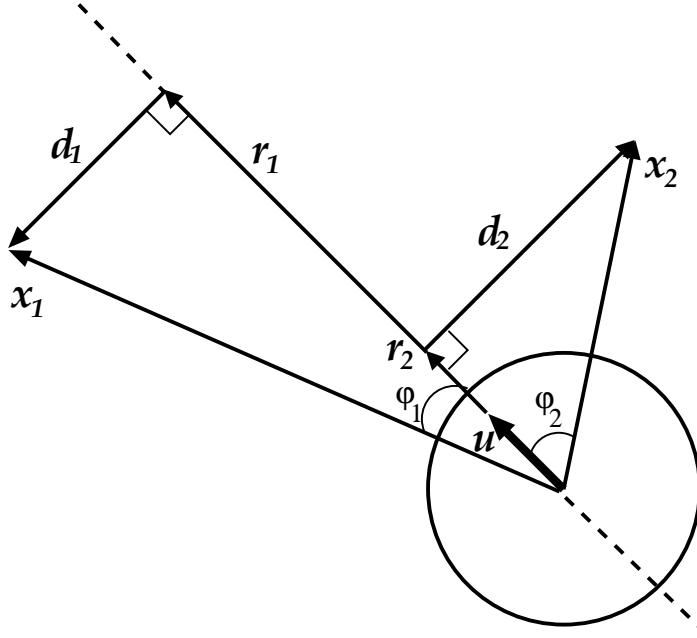


Figura 3.2: Interpretación geométrica de la prueba de colinealidad de los vectores  $x_1, x_2$ .

mación de la máxima verosimilitud de la dirección real de la señal (representado por el vector unitario  $u$ ) se obtiene minimizando la suma  $D^2 = |d_1|^2 + |d_2|^2$  del cuadrado de las distancias  $d_i$  de los vectores observados  $x_i$  al eje dado por el vector  $u$ . Se puede observar que si  $x_1$  y  $x_2$  son colineales, la diferencia de los vectores y de ahí la suma de sus normas es cero. Las proyecciones  $r_i, i = 1, 2$ , son estimaciones de máxima verosimilitud de los vectores de la señal correspondientes. Es evidente que se tiene,

$$\begin{aligned} |d_i|^2 &= |x_i|^2 - |r_i|^2 \quad \text{para } i = 1, 2 \\ |r_i| &= ||x_i| \cdot \cos \varphi_i| = |x_i^T \cdot u| \quad (|u| = 1) \\ \implies |d_i|^2 &= |x_i|^2 - |x_i^T \cdot u|^2 \\ \implies D^2 \stackrel{def}{=} |d_1|^2 + |d_2|^2 &= |x_1|^2 + |x_2|^2 - |x_1^T \cdot u|^2 - |x_2^T \cdot u|^2. \end{aligned}$$

Fórmese, la matriz  $X_{2 \times N}$ ,

$$X \stackrel{def}{=} \begin{pmatrix} x_1^T \\ x_2^T \end{pmatrix} \quad X \cdot u = \begin{pmatrix} x_1^T \cdot u \\ x_2^T \cdot u \end{pmatrix}$$

$$\implies |X \cdot u|^2 = u^T \cdot X^T \cdot X \cdot u = |x_1^T \cdot u|^2 + |x_2^T \cdot u|^2$$

Entonces,  $D^2$  quedaría de la siguiente manera,

$$D^2 = |d_1|^2 + |d_2|^2 = |x_1|^2 + |x_2|^2 - u^T \cdot X^T \cdot X \cdot u$$

y el vector  $u$  que minimiza  $D^2$  es el mismo vector que maximiza

$$u^T \cdot X^T \cdot X \cdot u \longrightarrow \max \quad \text{con } |u| = 1$$

lo cual es un problema de eigenvalores con respecto a la matriz  $X^T \cdot X$ . Debido a la manera en que esta construida la matriz  $X$ , se tienen solamente dos eigenvalores. Se está interesado solamente en el valor del estadístico de prueba  $D^2$  y se puede mostrar que  $D^2$  es idéntico al eigenvalor mas pequeño de la matriz  $X^T \cdot X$ . Además, se puede mostrar que los eigenvalores que no son cero, de la matriz  $X^T \cdot X$  y  $X \cdot X^T$  son idénticos, de donde el eigenvalor buscado es el más pequeño de los dos eigenvalores de la matriz de dimensión  $2 \times 2$ ,  $X \cdot X^T$ , el cual se puede calcular sin utilizar algún método iterativo. Así que el valor mínimo de  $D^2$  puede ser determinado sin tener que conocer  $u$ . (Esta derivación esta relacionada con el problema de Mínimos Cuadrados Totales (Total Least Squares (TLS)) [Mester and Muhlich, 2001] y tareas de reducción de rango en matrices utilizado en teoría de estimación moderna).

Para construir un procedimiento de decisión que tenga significado matemática y estadísticamente se debe conocer la distribución de  $D^2$ , por lo menos para la hipótesis nula  $H_0$ (=colinealidad). Supóngase que la norma del vector de la señal  $s$  es mucho más grande que la norma del vector del ruido (lo cual debería ser verdadero en la mayoría de los casos prácticos), se puede mostrar que la suma  $|d_1|^2 + |d_2|^2$  es proporcional a una variable  $\chi^2$  con  $N - 1$  grados de libertad y un factor proporcional a  $\sigma_d^2$

$$D^2 \sim \sigma_d^2 \cdot \chi_{N-1}^2. \quad (3.3)$$

Probar la hipótesis nula puede expresarse verificando si  $D^2$  puede ser explicado por el modelo de ruido dado, lo cual se puede realizar aplicando una prueba y nivel de confianza,  $t$  y  $\alpha$ , respectivamente, lo cual se puede expresar con la siguiente probabilidad condicional  $Prob(D^2 > t | H_0) = \alpha$ . Bajo la prueba alternativa  $H_1$  ( vectores  $x_1$  y  $x_2$  no son colineales), se modela la función de probabilidad condicional  $p(D^2 | H_1)$  como

$$p(D^2 | H_1) = \left( \frac{1}{\sqrt{2\pi}\sigma_c} \right)^N \cdot \exp\left(-\frac{D^2}{2 \cdot \sigma_c^2}\right) \quad (3.4)$$

con  $\sigma_c^2 \gg \sigma_d^2$ . Además, los cambios se modelan utilizando *Markov Random Fields* de tal manera que las regiones con cambios detectados tiendan a ser compactos. De este modelo, las probabilidades apriori  $Prob(c)$  y  $Prob(u)$ , para las etiquetas  $c$ -(cambios) y  $u$ -(no cambios) pueden ser obtenidas. La regla de decisión máxima apriori (MAP), dadas las etiquetas de la vecindad, esta dada por

$$\frac{p(D^2 | H_1)}{p(D^2 | H_0)} \underset{u}{\overset{c}{\gtrless}} \frac{Prob(u)}{Prob(c)}. \quad (3.5)$$

Con algo de álgebra se tiene la regla de decisión adaptativa:

$$D^2 \underset{u}{\overset{c}{\gtrless}} T + (4 - p_c) \cdot B \quad (3.6)$$

donde  $D^2$  es el estadístico de prueba introducido anteriormente,  $T$  es un umbral fijo, el parámetro  $p_c$  denota el número de píxeles que han sido etiquetados como  $c$ .

Lo anterior es realizado en la vecindad de tamaño  $3 \times 3$  del píxel bajo evaluación tomándose los valores de los píxeles de la vecindad que ya han sido procesados, los cuales están representados con valores de gris en la figura 3.3, mientras que de los píxeles que faltan por evaluar se tomará el valor obtenido previamente.

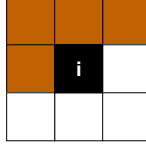


Figura 3.3: Vecindad a evaluación.

Para detectar ruido introducido a la imagen de referencia debido al proceso de captura el trabajo propuesto por [Aach et al., 1993] puede ser utilizado, en el cual es posible calcular cual es la probabilidad que tiene un valor en una posición dada de la imagen de ser causado por ruido en lugar de otras causas al ser comparado con una imagen de referencia. Se asume que el ruido que afecta a la imagen proviene de un proceso gaussiano con media  $\mu_N$  y desviación estándar  $\sigma_N$ ; además, el ruido que afecta a imágenes sucesivas en la secuencia se considera decorrelacionado. La desviación estándar  $\sigma_N$  puede ser obtenida calculando la diferencia  $d_{(i,j)}$  de cada píxel  $(i,j)$  entre la imagen de referencia y la imagen actual. Ahora, considérese un ventana  $W_{i,j}^n$  que contiene  $n$  píxeles alrededor del píxel  $(i,j)$  bajo evaluación con  $\Delta_{(i,j)}^2 = \sum_{(k,l) \in W_{i,j}^n} d_{(k,l)}^2$ . Se puede demostrar que la variable aleatoria  $\Delta^2$  sigue una distribución  $\chi^2$ . Entonces, dada la hipótesis  $H_0$  de que  $\Delta_{(i,j)}^2$  resulta de ruido en vez de otro factor, la probabilidad que la hipótesis  $H_0$  se cumpla esta dada por:

$$P\left(\Delta^2 > \Delta_{(i,j)}^2 \mid H_0\right) = \frac{\Gamma\left(\frac{n}{2}, \frac{\Delta_{(i,j)}^2}{2\sigma_c^2}\right)}{\Gamma\left(\frac{n}{2}\right)} \quad (3.7)$$

con  $\sigma_c^2 = 2\sigma_N^2$  y donde  $\frac{\Gamma(n,a)}{\Gamma(n)}$  es la función Gamma incompleta regularizada. Cuando la probabilidad estimada de la ecuación 3.7 es menor a un umbral  $Th_N$  se considera que la hipótesis  $H_0$  no se satisface; sin embargo, para píxeles que la cumplen se garantiza que los cambios son originados solamente por ruido de la cámara y que estos píxeles son necesariamente parte del fondo. Generalmente se elige un tamaño de ventana  $n = 25$  y en [Teixeira and Corte-Real, 2007] se elige un umbral del ruido  $Th_N = 10^{-4}$ .

Otro trabajo que utiliza una prueba de ruido para eliminar falsos positivos se menciona en el trabajo de [Grest et al., 2003], en donde se mide la varianza del ruido del píxel sobre la imagen de diferencias completa sobre cada nuevo

frame. Sea  $d(x, y) = r(x, y) - a(x, y)$  la imagen diferencia, con  $r(x, y)$  la imagen de referencia o fondo y  $a(x, y)$  la intensidad de la imagen actual en la posición  $(x, y)$ . La varianza del ruido en la imagen de tamaño  $M \times N$  es,

$$\sigma^2 = \frac{1}{MN} \sum_{i=1}^{M-1} \sum_{j=1}^{N-1} (d(i, j) - \bar{d})^2, \quad (3.8)$$

donde  $\bar{d}$  es la media de la imagen diferencia. Para cálculos eficientes la ecuación 3.8 puede ser reescrita como:

$$\sigma^2 = \frac{1}{MN} \sum_{i=1}^{M-1} \sum_{j=1}^{N-1} d(i, j)^2 - \left( \sum_{i=1}^{M-1} \sum_{j=1}^{N-1} d(i, j) \right)^2. \quad (3.9)$$

Asumiendo el ruido como una distribución gaussiana se asigna el umbral para la segmentación en  $2\sigma^2$ . Así que cada píxel pertenece al fondo si,

$$|d(x, y) - \bar{d}| < 2\sigma^2. \quad (3.10)$$

La principal desventaja que tiene este enfoque es la rápida adaptación a los cambios en iluminación, porque el umbral es calculado por cada frame y es independiente del frame anterior y como el valor de la media en las diferencias se incorpora aunque existan variaciones en la iluminación. Sin embargo, es importante calcular el nuevo umbral de los píxeles que son fondo solamente y no de sombras, partes de personas u otros objetos en movimiento.

## 3.2 Cambios rápidos en la iluminación

Debido a que el método de mezcla de gaussianas genera gran cantidad de regiones en movimiento incorrectas cuando hay cambios rápidos en la iluminación, en el trabajo de [Tian et al., 2005] se integra la información de la textura a las regiones en movimiento detectadas para remover falsos positivos. La idea básica es que la textura de regiones afectadas por un cambio rápido de iluminación debería de ser similar a la textura en el fondo.

El valor del gradiente es menos sensible a cambios en la iluminación rápidos y es capaz de derivar una medida de diferencia de texturas locales exactas. Se define una medida para medir la similitud de texturas, correlación, para cada píxel  $X$  entre el frame actual y la imagen de fondo como

$$S(X) = \frac{\sum_{u \in W_x} 2 \|g(u)\| \cdot \|g_b(u)\| \cos \theta}{\sum_{u \in W_x} (\|g(u)\|^2 + \|g_b(u)\|^2)}, \quad (3.11)$$

donde  $S(X)$  denota la correlación de textura de la imagen actual y de fondo,  $W_x$  la vecindad de  $M \times N$  centrada en el píxel  $X$ ;  $g$  y  $g_b$  son el vector gradiente del frame actual y la imagen de fondo, respectivamente, y  $\theta$  es el ángulo entre los vectores. El vector gradiente  $g(X) = (g^x(X), g^y(X))$  y las derivadas parciales  $g^x(X)$  y  $g^y(X)$  son obtenidas con el operador Sobel. En las áreas de falsos positivos

del primer plano causadas por cambios rápidos en la iluminación se espera que no existan cambios en la textura entre el frame actual y el de fondo. Entonces,  $S(X) \approx 1$ . La máscara del primer plano será removida de áreas con  $S(X) \geq Th_s$ , un umbral  $Th_s = 0.7$  se utiliza en [Tian et al., 2005].

Lo robusto de la medida 3.11 con respecto a cambios de iluminación y al ruido puede encontrarse en [Li and Leung, 2002].

### 3.3 Detección de sombras

En la literatura existen diversos algoritmos para la detección de sombras, la mayoría de éstos están basados en características de color que no son afectadas por condiciones de la iluminación. [McKenna et al., 2000] utilizó información de píxel y bordes de cada canal del espacio RGB normalizado (rgb). [Elgammal et al., 2002] también utilizó el espacio rgb pero incluyó una medida de iluminación para detectar sombras proyectada en el fondo. [Cucchiara et al., 2003] y [Prati et al., 2001] utilizaron el espacio de color HSV y clasifican como sombras aquellos píxeles que tuviesen el canal de tinte (hue) y saturación similares pero baja luminosidad. [KaewTrakulPong and Bowden, 2003] utiliza una medida de distorsión cromática y un umbral de brillo en el espacio RGB para determinar píxeles no fondo afectados por sombras. [Salvador et al., 2004] adoptaron el espacio de color  $c1c2c3$  y exploraron características geométricas de las sombras.

Para distinguir las sombras de una secuencia de video se asume que éstas tienen las siguientes propiedades [Grest et al., 2003]:

- un píxel sombra es mas oscuro que el píxel correspondiente en la imagen de fondo,
- la textura de la sombra es correlacionada con la textura correspondiente a la imagen de fondo.

Las sombras pueden ser removidas en la imagen segmentada realizando los siguientes pasos [Grest et al., 2003]:

1. Se aplica la operación morfológica matemática, erosión, para eliminar píxeles que son causados generalmente por el ruido.
2. Para cada píxel en la imagen actual, que sea más oscura que el correspondiente píxel de la imagen de referencia, se calcula una correlación cruzada normalizada de  $7 \times 7$  entre la imagen de referencia y la imagen actual.

En regiones que son sombras, se espera que una parte  $\beta$  de la luz sea bloqueada [Elgammal et al., 2002], aunque existen varios factores que afectan la intensidad de un píxel en sombra [Salvador et al., 2004], se asume que la intensidad observada de los píxeles en sombra es directamente proporcional a la luz incidente. Como consecuencia, píxeles sombra son versiones escaladas (más oscuras) de los

píxeles considerados como fondo.

### 3.3.1 Correlación con la intensidad de imágenes

Para medir la similitud de regiones de una imagen, es posible calcular la correlación cruzada sobre un tamaño de ventana dado, ya que esta medida puede identificar versiones escaladas de la misma señal. Usualmente se desea que no exista variación en el brillo, como se presenta en la covarianza cruzada. En [Grest et al., 2003], la covarianza cruzada que utiliza un tamaño de ventana  $M \times N$  para dos imágenes  $a, b$  en una posición dada  $(x, y)$ , está definida como:

$$CC_{a,b}(x, y) = \frac{1}{MN} \sum_{i,j} (a_{i,j} - \bar{a}) (b_{i,j} - \bar{b}), \quad (3.12)$$

donde  $i$  abarca de  $(x - \frac{M-1}{2})$  a  $(x + \frac{M-1}{2})$  y  $j$  de  $(y - \frac{N-1}{2})$  a  $(y + \frac{N-1}{2})$ . Las variables  $\bar{a}, \bar{b}$  denotan el promedio sobre la ventana de  $M \times N$  de las imágenes  $a$  y  $b$  respectivamente.

La covarianza cruzada es invariante al brillo, pero sensible a cambios en el contraste. Para lograr también invarianza en el contraste, la correlación es normalizada por la varianza de cada ventana, lo cual lleva a la conocida correlación cruzada normalizada (NCC). Para eficientes cálculos la NCC puede expresarse como,

$$NCC = \frac{\sum_{i,j} a_{i,j} b_{i,j} - \frac{1}{MN} \sum_{i,j} a_{i,j} \sum_{i,j} b_{i,j}}{\sqrt{\left(\sum_{i,j} a_{i,j}^2 - MN \bar{a}^2\right) \left(\sum_{i,j} b_{i,j}^2 - MN \bar{b}^2\right)}} \quad (3.13)$$

con

$$\bar{a} = \frac{1}{MN} \left( \sum_{i,j} a_{i,j} \right)$$

Para píxeles que son sombra, esta correlación es muy cercana a 1, así que eliminamos los píxeles oscuros en la imagen segmentada si la NCC entre las imágenes de referencia y actual es mayor a un umbral  $\theta_{NCC}$ . Después de remover los píxeles se aplica el operador morfológico de cerradura. En [Grest et al., 2003] se utiliza un umbral  $\theta_{NCC} = 0.4$ , el cual elimina la mayoría de las sombras pero no todas. Si se reduce el umbral  $\theta_{NCC}$  se eliminan más sombras, pero también remueve partes de los objetos en movimiento. Esto se debe al tamaño de ventana tan pequeño que se utiliza para tener cálculos más rápidos. En regiones que son muy homogéneas, la NCC tiende a variar mucho, haciendo muy difícil distinguir regiones sin textura de las sombras.

En [Silveira Jacques Jr. et al., 2005] utilizan la correlación en niveles de gris entre las imágenes  $a$  y  $b$  para detectar píxeles *candidatos* de sombra de manera similar al trabajo de [Grest et al., 2003] con la diferencia que **no se centran los datos**.

La NCC en la posición de píxel  $(i, j)$  se define de la siguiente manera:

$$NCC(i, j) = \frac{ER(i, j)}{EB(i, j) ET(i, j)}, \quad (3.14)$$

donde

$$\begin{aligned} ER(i, j) &= \sum_{n=-N}^N \sum_{m=-N}^N b(i+n, j+m) a(i+n, j+m), \\ EB(i, j) &= \sqrt{\sum_{n=-N}^N \sum_{m=-N}^N b(i+n, j+m)^2}, \\ ET(i, j) &= \sqrt{\sum_{n=-N}^N \sum_{m=-N}^N a(i+n, j+m)^2}, \end{aligned}$$

pero además se verifica que la varianza de la región actual,  $ET(i, j)$ , sea menor a la varianza de la región del fondo,  $EB(i, j)$ , de donde, la detección de sombras *candidatas* sería:

$$NCC \geq \theta_{NCC} \quad y \quad ET < EB.$$

Además, para remover falsos positivos, se propone verificar si la proporción  $\frac{I(i, j)}{B(i, j)}$  en una vecindad alrededor de cada píxel candidato es aproximadamente constante, esto se verifica calculando la desviación estándar de  $\frac{I(i, j)}{B(i, j)}$ , donde  $I(i, j)$  y  $B(i, j)$  es el píxel en la posición  $i, j$  de las imágenes actual y fondo, respectivamente.

De manera más específica, considérese una región  $R$  con  $(2M + 1) \times (2M + 1)$  píxeles (se utiliza  $M = 1$ ) centrado en cada píxel candidato.

Se clasifica como píxel fondo si

$$std_R \left( \frac{I(i, j)}{B(i, j)} \right) < Th_{std} \quad \& \quad Th_{low} \leq \left( \frac{I(i, j)}{B(i, j)} \right) < 1$$

donde  $std_R \left( \frac{I(i, j)}{B(i, j)} \right)$  es la desviación estándar de los valores  $\frac{I(i, j)}{B(i, j)}$  en la región  $R$  y  $Th_{std}$  es un umbral que controla la máxima desviación dentro de una vecindad,  $Th_{low}$  es un umbral que previene la mal clasificación de objetos oscuros con intensidades de píxel muy baja. En [Silveira Jacques Jr. et al., 2005] utilizan un umbral  $\theta_{NCC} = 0.95$ , un tamaño de ventana  $M = N = 9$ ,  $Th_{std} = 0.05$  y  $Th_{low} = 0.5$ .

En la figura 3.2 se muestran los resultados obtenidos al utilizar la correlacion de la intensidad para la identificacion de las sombras. En el primer renglon de las imágenes se muestran los píxeles que fueron detectados como píxeles en movimiento, los píxeles que se encuentran de color cafe son píxeles que fueron



identificados como sombras, mientras que los píxeles de color azul representan los píxeles que se encuentran en movimiento. El segundo renglon muestra los píxeles que se encuentran en movimiento al eliminar los píxeles que fueron etiquetados como sombra.





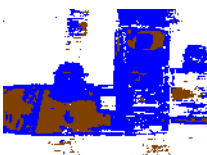
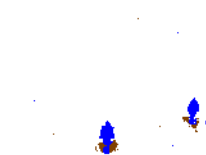

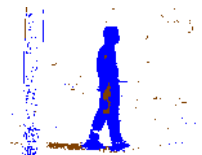

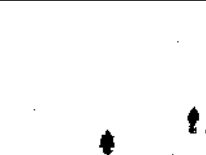


	HighWay1	Pets2000	Pets2000	WaterSurface
Imagen Real				
Sombras				
Movimiento				

Tabla 3.2: Detección de sombras con Correlación de la Intensidad

### 3.3.2 Correlación de imágenes a color

Para mejorar la calidad en la detección de sombras, se agrega la información que proporciona el color [Grest et al., 2003], esto si los valores de la intensidad no proporcionan información de correlación. Para separar la información del color y del brillo se utiliza el espacio de color HSL (Hue, Saturation, Luminance), el cual tiene las siguientes propiedades:

- El valor del tinte (hue) del espacio HSL es el ángulo en el plano cromático alrededor del eje acromático del espacio RGB. El valor del tinte es independiente del brillo.
- La saturación  $S$  del color es calculada como  $\max(R, G, B) - \min(R, G, B)$  y es sin embargo dependiente del brillo. Una normalización en la saturación resultaría en valores de saturación independientes del brillo, pero esto no se desea ya que ocasionaría que la saturación varíe demasiado en regiones oscuras debido al ruido.
- La intensidad  $L$  es calculada como  $L = \frac{\max(R, G, B) + \min(R, G, B)}{2}$ . Se pueden utilizar otras fórmulas pero no se tendría la forma bicónica como en la figura 3.4.

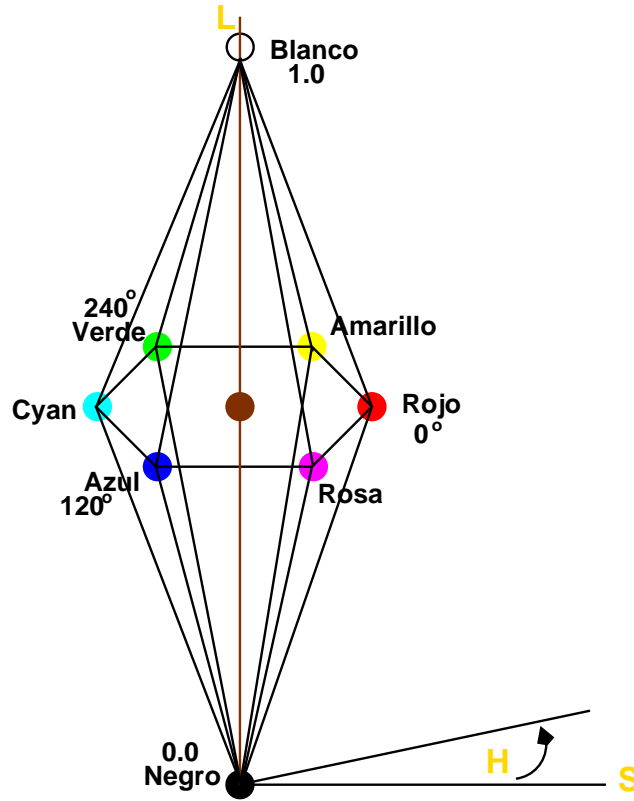


Figura 3.4: Espacio de color HSL

La proyección de todo el espacio RGB en el plano cromático es un hexágono como lo muestra la figura 3.5. Para medir la similitud se calcula el espacio de color HSL no en coordenadas polares sino que se utiliza la proyección del vector  $(R, G, B)$  en el plano crómico  $(H, S)$  para calcular los valores euclidianos del color y la saturación (estas conversiones se muestran en el apéndice B). Se denotará la representación de las coordenadas euclidianas como  $(h, s)$ , llamándose al espacio de color donde los valores H,S han sido convertidos a coordenadas euclidianas como el espacio hsL [Angulo and Serra, 2004]. La parte proyectada h,s es escalada de tal manera que la longitud es igual a la saturación del espacio HSL,

$$c = S \cdot \frac{c}{\|c\|} \quad c \in \{h, s\}.$$

Se propone para la correlación de dos píxeles de color  $c^a = (h^a, s^a, L^a)$  y  $c^b = (h^b, s^b, L^b)$  el producto escalar  $c^a c^b$ . El producto escalar de dos vectores  $(h, s, L)$  diferentes resulta en una correlación alta para píxeles de color que tienen tinte (hue) similar, esto es ángulo pequeño entre ellos, y además tienen grande saturación.

Como la saturación de un vector es equivalente a la longitud del vector  $(h, s)$ , los píxeles de color tienen una correlación pequeña, si alguna de las saturaciones lo es. Sin embargo, el producto escalar es conveniente para medir la correlación

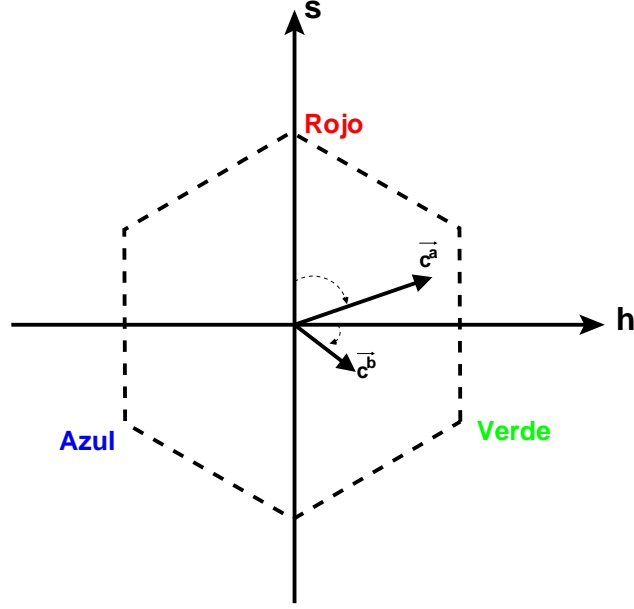


Figura 3.5: Plano cromático de espacio de color HSL

de píxeles de color. El producto escalar para  $(h, s)$  es calculado separadamente de las intensidades y resultados negativos son igualados a cero. Eso considera el hecho que dos colores con un ángulo de tinte de más de 90 grados entre ellos, son interpretados como no similares.

Se define la covarianza cruzada de color (CCC) sobre una ventana  $M \times N$  para dos píxeles de color  $c_{xy}^a, c_{xy}^b$  de dos imágenes  $a, b$  en una posición de la imagen  $(x, y)$  como,

$$CCC_{a,b}(x, y) = \frac{1}{MN} \sum_{i,j} \left( c_{ij}^a \bullet c_{ij}^b \right) - \bar{L}^a \bar{L}^b \quad (3.15)$$

con

$$c_{ij}^a \bullet c_{ij}^b = (h_{ij}^a, s_{ij}^a) \circ (h_{ij}^b, s_{ij}^b) + L_{ij}^a L_{ij}^b \quad (3.16)$$

donde  $i$  abarca de  $(x - \frac{M-1}{2})$  a  $(x + \frac{M-1}{2})$  y  $j$  de  $(y - \frac{N-1}{2})$  a  $(y + \frac{N-1}{2})$  y  $\bar{L}^a$  es el promedio de las intensidades en la imagen  $a$  sobre la ventana de  $M \times N$ . El operador  $\circ$  denota el producto escalar con los valores negativos resultados igualados a cero.

La extensión de NCC para imágenes a color es de manera directa. Los valores de correlación de color (tinte y saturación) son solamente normalizados para que la correlación cruzada normalizada de color (CNCC) se encuentre entre  $[-1 \dots 1]$ . Se define CNCC sobre una ventana de tamaño  $M \times N$  para dos píxeles de color  $c_{xy}^a, c_{xy}^b$  en una posición de la imagen  $(x, y)$  como,

$$CNCC_{x,y} = \frac{\sum_{i,j} \left( c_{ij}^a \bullet c_{ij}^b \right) - \bar{L}^a \bar{L}^b}{\sqrt{VAR^a VAR^b}} \quad (3.17)$$

con

$$VAR^k = \left( \sum_{i,j} (c_{ij}^k \bullet c_{ij}^k) - MN\bar{L}^{k^2} \right)$$

donde  $i, j, \bar{L}^a, \bar{L}^b$  ya han sido definidas anteriormente y  $k \in \{a, b\}$ .

Las saturaciones de los colores tienen la misma escala que las intensidades, tal que la contribución conjunta del tinte y la saturación tenga tanta influencia en el valor resultante de  $CNCC$  como solo el valor intensidad. En la tabla 3.3 se muestran algunos resultados de la detección de sombras utilizando la correlación de imágenes a color. El segundo renglón muestra los píxeles que fueron identificados como sombras en color café y en azul aquellos que se etiquetaron como píxeles en movimiento. El tercer renglón muestra las imágenes de regiones en movimiento después de eliminar las regiones que se identificaron como sombras.

En general, la detección de sombras utilizando el color elimina más sombras que usando solamente intensidades pero también la  $CNCC$  elimina gran parte de los objetos que no son sombra lo cual no es conveniente para la buena identificación de regiones en movimiento.













	HighWay1	Pets2000	Pets2000	HighWay1
Imagen Real				
Sombras				
Movimiento				

Tabla 3.3: Detección de sombras con Correlación utilizando el color

Existen diversas técnicas que permiten detectar el efecto que ocasionan factores externos como el ruido, iluminación, etc.. En este capítulo se mencionaron algunas maneras de mejorar la detección de objetos en movimiento.

Pero es importante recalcar que aunque en la mayoría de las simulaciones se tuvieron mejores resultados, todavía no existe técnica que funcione para todas las situaciones que existen.

## Capítulo 4

---

# Trabajo desarrollado y resultados

---

El trabajo [Teixeira and Corte-Real, 2007] fué el algoritmo de mezcla de Gaussianas que presentó los mejores resultados de entre los algoritmos que se analizaron [Stauffer and Grimson, 1999], [Lee, 2005]. La mejora del método en Cascada se debe a las pruebas de ruido e iluminación introducidas, dando como resultado una detección más exacta de las regiones en movimiento. Por lo cual, el método en Cascada se tomó como base para el análisis de los métodos de substracción de fondo con modelo de mezclas de Gaussianas desarrollado en esta tesis. Se analizaron diversos parámetros para verificar el efecto que tienen dentro del modelo de mezcla de Gaussianas así como las pruebas de cambios no estructurales.

En esta sección se menciona el análisis y cambios realizados en el algoritmo de Cascada para aumentar su efectividad. El trabajo realizado dió como resultado una buena detección de regiones en movimiento, eliminando lo más posible los píxeles que son afectados por iluminación y ruido. Los parámetros que se analizaron fueron el valor de la varianza  $\sigma^2$ , la velocidad de aprendizaje  $\alpha$ , el número de distribuciones  $K$  para la estimación de la función de densidad de la historia de cada pixel, el espacio de color utilizado, el valor de pixel que es modelado en la mezcla de Gaussianas, entre otros. Además se analizaron diversas pruebas para incrementar la exactitud de la detección del movimiento, entre ellas están la prueba de ruido, textura y sombras. Los resultados obtenidos se mencionan en este capítulo.

### 4.1 Umbral para $\sigma^2$

Los valores de los parámetros  $\sigma^2$  y  $\omega$ , resultan de gran importancia en los métodos de substracción de fondo con modelo de mezcla de gaussianas, ya que estos parámetros aportan gran información para la segmentación del fondo y no fondo. En los píxeles que pertenecen al fondo, se presenta un valor de  $\sigma^2$  pequeño y un valor de  $\omega$  grande, ya que se espera que en los píxeles que pertenecen al fondo

no se tenga mucha variación en el valor del píxel y además el peso grande indica que varios de los píxeles que han sido procesados en el tiempo han encontrado concordancia o han sido generados por esa distribución.

Pero si no se tuviera control sobre el valor que toma el parámetro  $\sigma$ , en una secuencia de imágenes en donde la mayoría de los píxeles han sido fondo, se tendrían valores demasiado pequeños (menores a  $1 \times 10E - 8$ ), lo cual no filtraría el ruido que pueda existir en la imagen, así como pequeñas variaciones del color que se presenten.

Debido a lo anterior, en este trabajo se fija un umbral para el valor de la varianza. El algoritmo 9 muestra las condiciones:

---

**Algoritmo 9:** Condición para  $\sigma$

---

```

forall  $k \in \{1 \text{ to } K\}$  do
  | forall  $z \in \{1 \text{ to } N\text{Canales}\}$  do
  | | if  $\sigma_{k,z} < \text{SigmaMenor}_z$  then
  | | |  $\sigma_{k,z} = \text{SigmaMenor}_z$ 
  | | end
  | end
end

```

---

Recuérdese que  $K$  es el número de distribuciones de la mezcla de Gaussianas,  $N\text{Canales}$  es el número de canales que tiene el espacio de color utilizado (3 en este caso, ya que se utiliza el espacio  $YUV$ ). El valor del umbral que se utilizó es el siguiente:  $\text{SigmaMenor} = [2.0, 2.0, 2.0]$ .

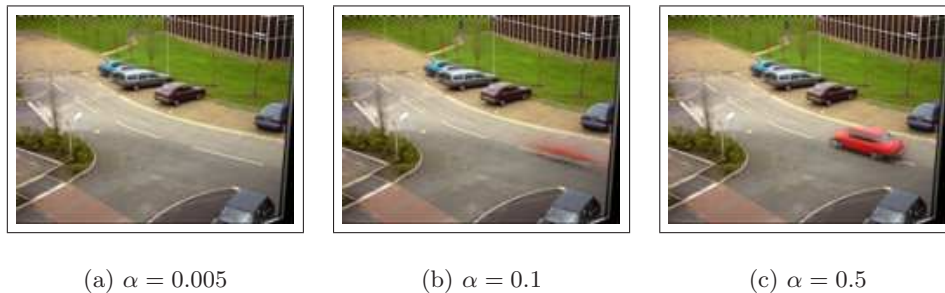
Los resultados mejoraron notablemente al incluir esta restricción, como ejemplo obsérvense los resultados obtenidos sin control de  $\sigma^2$  y los resultados que se tienen al incluir la condición, en las tablas 4.3 y 4.4, respectivamente.

## 4.2 Velocidad de aprendizaje

La velocidad de aprendizaje, esto es, la velocidad con la que las variables son actualizadas, es un factor muy importante para el buen modelado del fondo, algunas maneras de actualización se mencionaron en la sección 2.4.2. En la mayoría de los enfoques se utiliza una velocidad de aprendizaje  $\alpha$  fija con un valor muy pequeño, esto se establece debido a que de esa manera los objetos en movimiento tardan más en ser parte del fondo. Por ejemplo, utilizando la secuencia de imágenes PETS2000 se pueden observar los resultados utilizando un valor  $\alpha = 0.005$  [Stauffer and Grimson, 1999] y valores de alfa grande  $\alpha = \{0.1, 0.5\}$ .



Figura 4.1: Secuencia 130 de PETS2000



(a)  $\alpha = 0.005$

(b)  $\alpha = 0.1$

(c)  $\alpha = 0.5$

Figura 4.2: Modelos de fondo con alfa fija en secuencia 130 de PETS2000

Como se observa en la figura 4.2, mientras más grande sea el valor de  $\alpha$ , los objetos en movimiento se incorporan más rápido al fondo; lo cual la mayoría de las veces no es algo que se desee. Pero existen situaciones en las que es bueno que se tenga rápida asimilación; entre algunas de estas situaciones se tienen la variación en la iluminación. Véase como ejemplo la secuencia PETS2001\_DS3\_TE1\_C1 (PETS2001-DataSet3-Training1-Camera1), donde existen cambios rápidos en la iluminación.

Y ahora obsérvense los resultados que se tendrían con diferentes valores de  $\alpha$  en las figuras 4.1 y 4.2; la figura muestra en sus columnas: la imagen de fondo, diferencia, etiquetas con pruebas, etiqueta de movimiento.

La primera idea que se tuvo en este tipo de problemas fue utilizar un alfa que fuera decrementando conforme se vayan procesando los frames; lo cual ayudaría

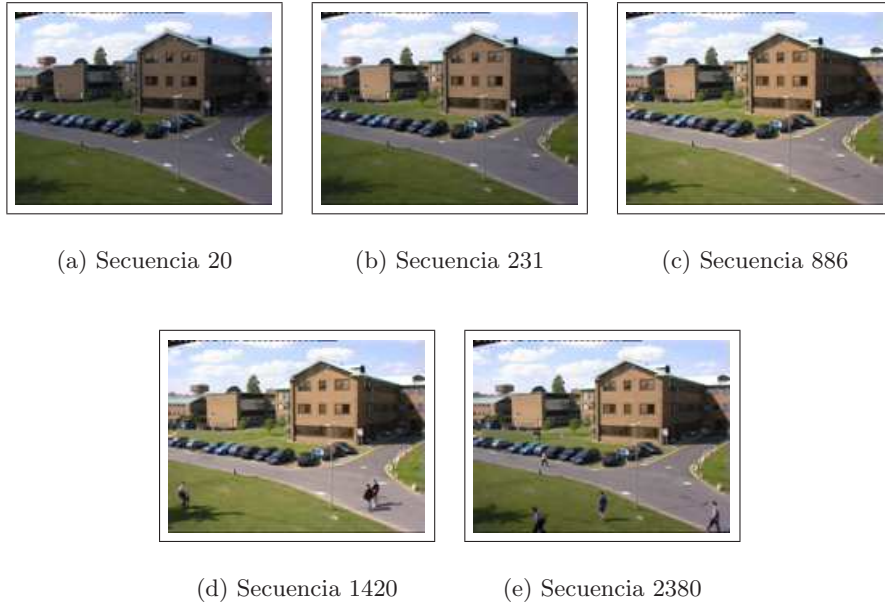


Figura 4.3: Secuencias de PETS2001\_DS3\_TE1\_C1

a la inicialización del modelo.

Después de realizar simulaciones con diversas formas de actualizar  $\alpha$ , la que presentó mejores resultados fue la siguiente:

$$\alpha = \frac{1}{1 + \log t^2}, \quad (4.1)$$

pero todavía se tenía el problema de la asimilación de los objetos en movimiento al fondo. Para solucionar esto, se decidió utilizar además un valor de  $\alpha$  pequeño para que fuese aplicado en las regiones donde se estaban detectando cambios.

Las ventajas que tiene esta alfa dual son las siguientes:

- Dado que en los primeros frames se tiene un alfa grande, en secuencias donde en los primeros frames se tiene mucho tráfico de objetos en movimiento, el fondo se actualiza más rápido.
- Se puede lidiar con cambios en la iluminación.
- No se tiene el problema de que las regiones en movimiento se asimilen al fondo tan rápido.

Para incorporar las ventajas de un valor pequeño de  $\alpha$ , se verificó si el píxel fué identificado en movimiento en la iteración anterior, y en esos casos se utiliza una alfa pequeña fija  $\alpha = 0.005$  para la actualización de los parámetros. Los





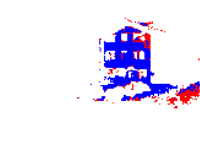



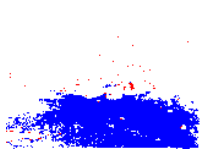



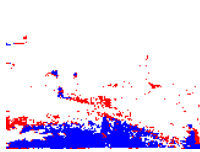

Secuencias PETS2001_3.1				
	Fondo	Diferencia	Filtros	Movimiento
Seq-231				
Seq-886				
Seq-2380				

Tabla 4.1: Modelos de fondo con  $\alpha = 0.005$  de PETS2001\_DS3\_TE1\_C1






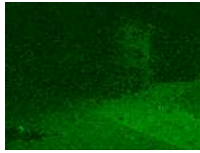






Secuencias PETS2001_DS3_TE1_C1				
	Fondo	Diferencia	Filtros	Movimiento
Seq-231				
Seq-886				
Seq-2380				

Tabla 4.2: Modelos de fondo con  $\alpha = 0.1$  de PETS2001\_DS3\_TE1\_C1

resultados obtenidos con estas modificaciones se muestran en la figura 4.3.

Notéanse las mejoras que muestran los resultados al aplicar el umbral en los valores de  $\sigma$  y  $\omega$  en la figura 4.4.

El pseudo-código para la utilización del alfa dual se muestra en el algoritmo 10.



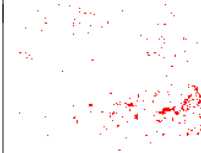



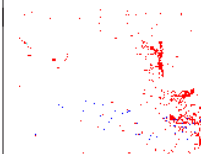



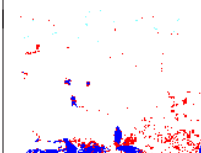

Secuencias PETS2001_DS3_TE1_C1				
	Fondo	Diferencia	Filtros	Movimiento
Seq-231				
Seq-886				
Seq-2380				

Tabla 4.3: Modelos de fondo con  $\alpha$  *dual* sin umbral en  $\sigma$  de PETS2001\_DS3\_TE1\_C1

---

**Algoritmo 10:** Alfa\_Dual()

---

```

1  $\alpha = \frac{1}{1+\log t^2}$ 
2 if  $\alpha < Th_\alpha$  then
3   |  $\alpha = Th_\alpha$ 
4 end
5 #El parámetro blnAlfa se utiliza en el algoritmo 13
6 blnAlfa = false
7 if etiquetast-1 == 0 & etiquetast-2 == 0 then
8   | #Se utilizará la alfa fija  $\alpha = 0.005$ 
9   | blnAlfa = true
10 end

```

---







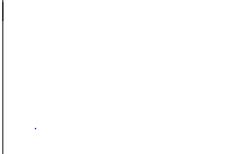





Secuencias PETS2001_DS3_TE1_C1				
	Fondo	Diferencia	Filtros	Movimiento
Seq-231				
Seq-886				
Seq-2380				

Tabla 4.4: Modelos de fondo con  $\alpha$  *dual* y con umbral en  $\sigma$  de PETS2001\_DS3\_TE1\_C1

### 4.3 Número de distribuciones adaptativo

El número de distribuciones es un parámetro muy importante en los métodos de sustracción de fondo con modelo de mezclas de distribuciones adaptativas esto debido a que mientras mayor sea el número de distribuciones se tendrá una mejor estimación de la función de densidad de los datos. Por otro lado, debido al costo computacional y el requerimiento de tiempo real no es posible aumentar demasiado esta variable. En el trabajo [Tan et al., 2006] se propone tener un número de distribuciones adaptativas, por lo que se adaptó esta propuesta al algoritmo de [Lee, 2005], el cual es el algoritmo utilizado para el aprendizaje de la mezcla en [Teixeira and Corte-Real, 2007]. Las ventajas que tiene el utilizar un número de distribuciones adaptativas está en que el costo computacional se reduce. En efecto, píxeles que no tienen demasiada complejidad para su modelado, utilizan un menor número de distribuciones para representar el comportamiento de dicho píxel; tal como se observa en las tablas 4.5 y 4.6 ; donde los píxeles de las imágenes representan el número de distribuciones utilizadas para modelar el valor del píxel; el valor blanco significa que ese píxel se modela con 1 gaussiana y dependiendo de los valores de gris aumenta el número de distribuciones utilizadas. Como resultado, en la mayoría de las secuencias utilizadas, el número de distribuciones utilizadas para estimar la densidad de los datos de forma dinámica es menor a utilizar un número fijo de distribuciones.





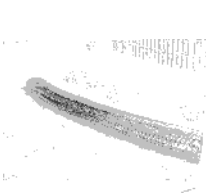


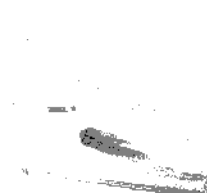
		Secuencias PETS2000		Secuencias PETS2001.3.1	
		Seq-196	Seq-1419	Seq-922	Seq-1685
Real					
Num. Distrib.					

Tabla 4.5: Número de distribuciones de cada píxel

Para observar las diferencias al utilizar un número de distribuciones fija y adaptativa, veáanse las imágenes 4.4, 4.5, 4.6 que muestran una comparación de la estimación de la función de densidad de la historia de un píxel en específico con un número fijo de distribuciones y con un número dinámico.



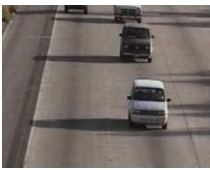
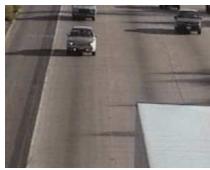
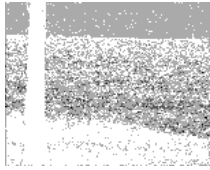
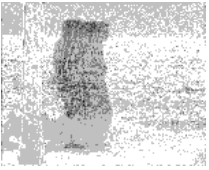

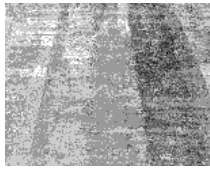
	Secuencias WaterSurface		Secuencias HighWay1	
	Seq-83	Seq-514	Seq-174	Seq-371
Real				
Num. Distrib.				

Tabla 4.6: Número de distribuciones de cada pixel

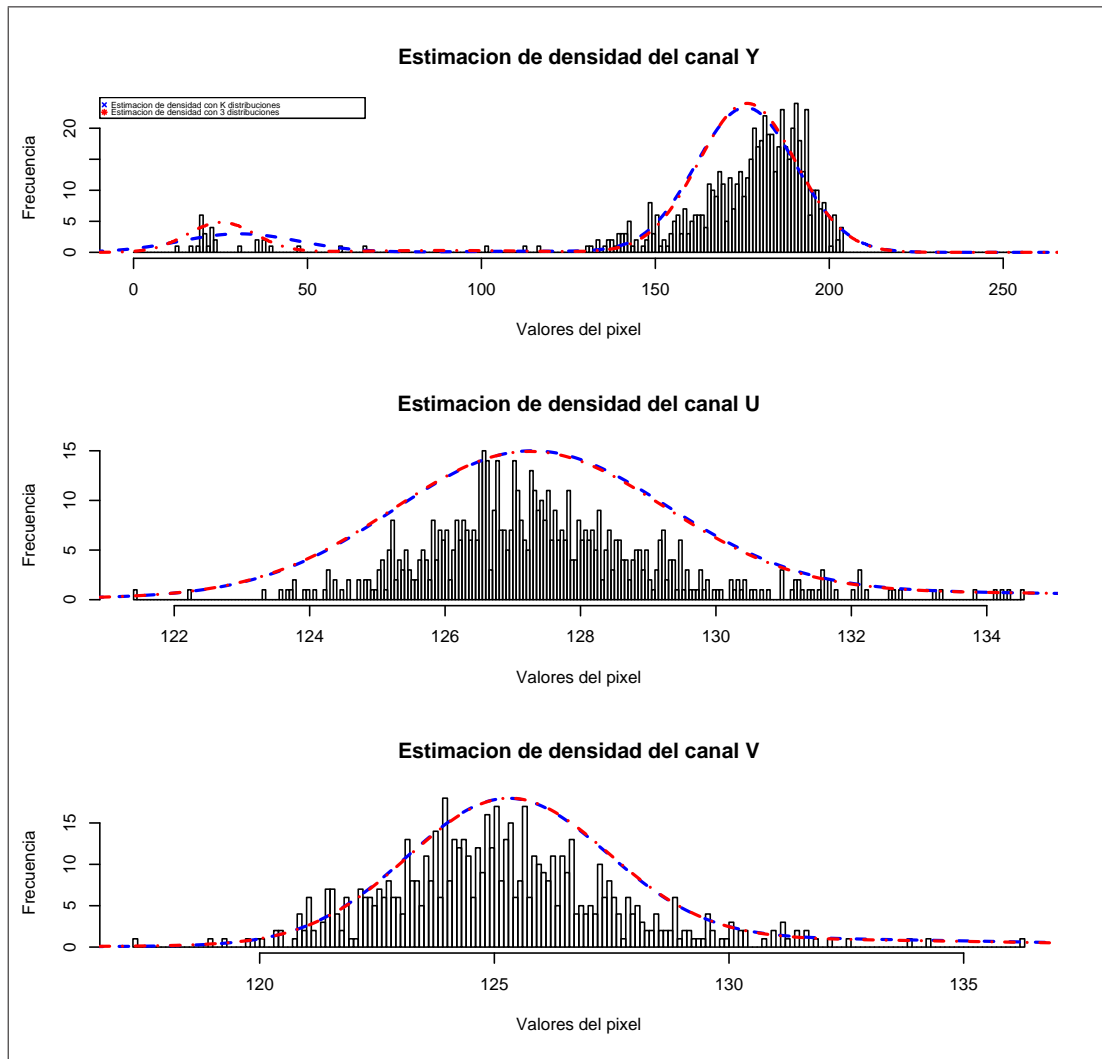


Figura 4.4: Función de densidad resultante secuencia WaterSurface pixel (113, 70)

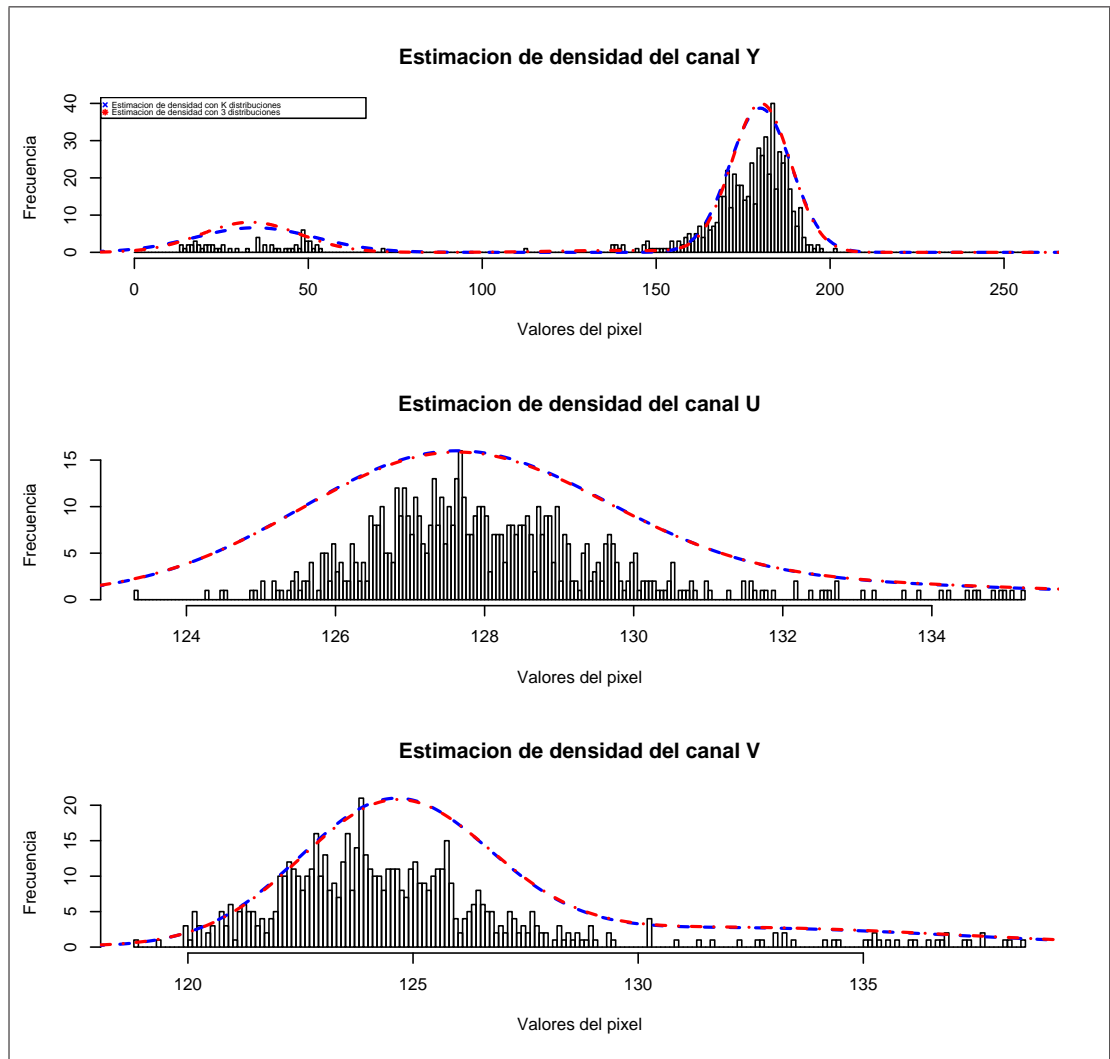


Figura 4.5: Función de densidad resultante secuencia WaterSurface pixel (102, 56)

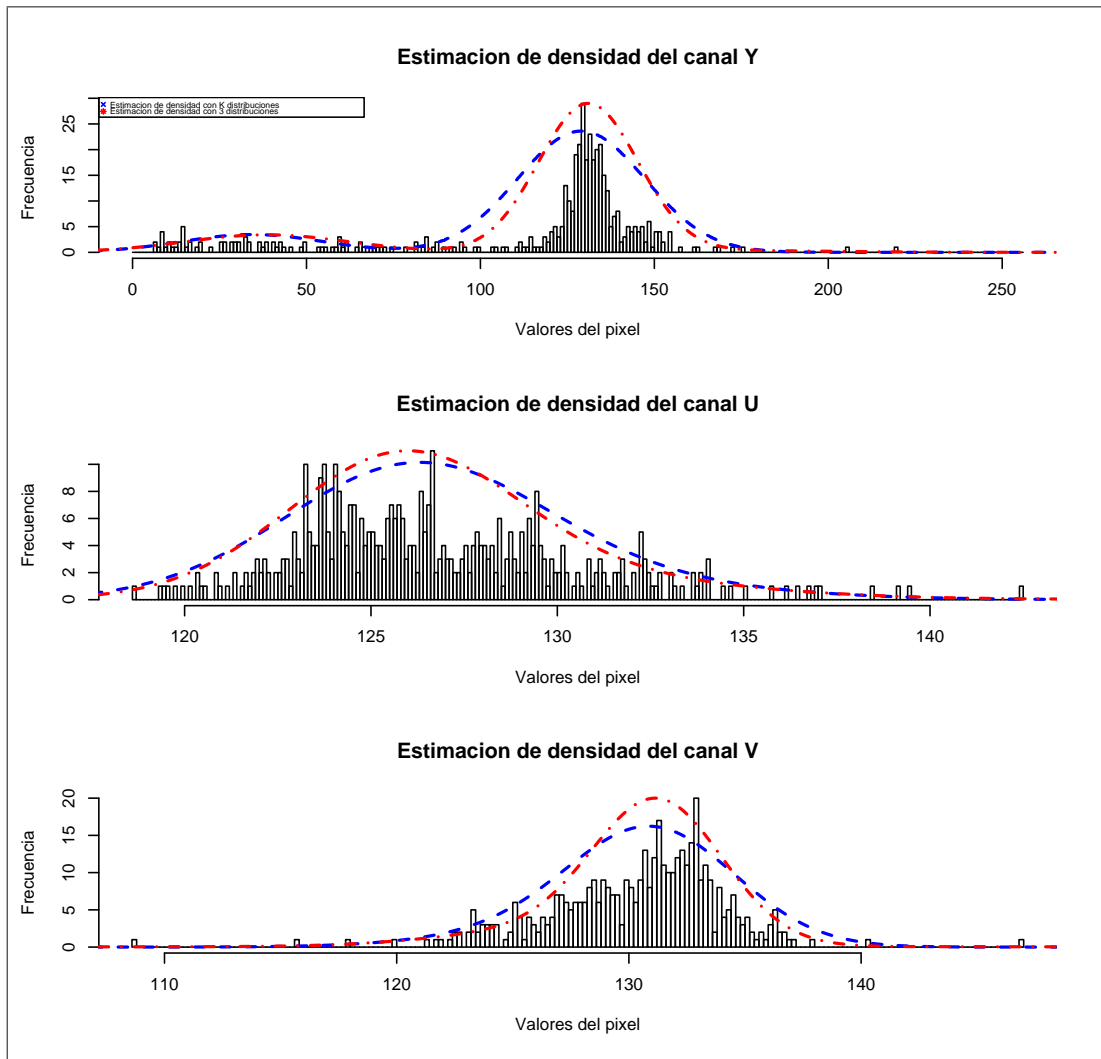


Figura 4.6: Función de densidad resultante secuencia HighWay1 pixel (110, 70)



## 4.4 Píxel vs. Estadístico de región

Se utilizaron diferentes estadísticos para el modelado del píxel de la imagen, y así verificar si se podrían obtener mejores resultados que al utilizar el valor del píxel tal cual, esto es, para determinar la función de densidad de la historia de los datos, se utilizaron los estadísticos de una ventana del píxel bajo evaluación en lugar del valor del píxel. Entre los estadísticos que se utilizaron se tienen:

**Media aritmética:** La media aritmética o promedio, de una cantidad finita de números, es igual a la suma de todos ellos dividida entre el número de sumandos. Dados los  $n$  números  $a_1, a_2, \dots, a_n$ , la media aritmética se define simplemente como:

$$\bar{x} = \frac{\sum_{i=1}^n a_i}{n} = \frac{a_1 + \dots + a_n}{n}$$

**Mediana:** La mediana de un conjunto finito de valores es aquel valor que divide al conjunto en dos partes iguales, de forma que el número de valores mayor o igual a la mediana es igual al número de valores menores o igual a estos. Considerando  $x_1, x_2, x_3, \dots, x_n$  los datos de una muestra ordenada en orden creciente y mediana como  $M_e$ :

$$M_e = \begin{cases} x_{(n+1)} & , \text{ si } n \text{ es impar} \\ \frac{x_{\frac{n}{2}} + x_{\frac{n}{2}+1}}{2} & , \text{ si } n \text{ es par} \end{cases} \quad (4.2)$$

**Media corregida:** La media corregida es obtener la media aritmética pero eliminando los  $m$  primeros y últimos valores de los datos ordenados. Considerando  $x_1, x_2, x_3, \dots, x_n$  los datos de una muestra ordenada, la media corregida esta dada por:

$$\bar{x}_c = \frac{\sum_{i=m}^n a_i}{n - 2m} = \frac{a_m + \dots + a_{n-m}}{n - 2m} \quad (4.3)$$

En las tablas 4.7-4.9 se muestran algunos resultados obtenidos con los estadísticos mencionados anteriormente. En general se observó que en secuencias que presentan fondo dinámico se observaron mejores resultados, como lo son escenas en las que se tiene movimiento de las ramas de los arboles, de las olas del mar, etc. El tamaño de ventana utilizado fue  $n = 5$ .

Al observar los resultados obtenidos, se decide **utilizar el estadístico de la media**.


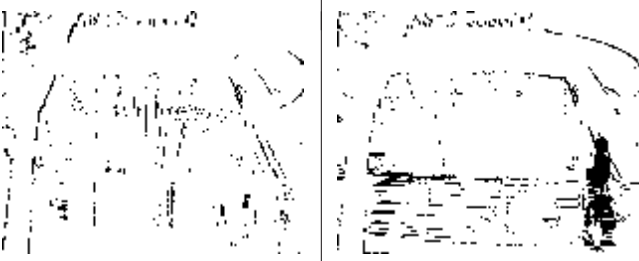
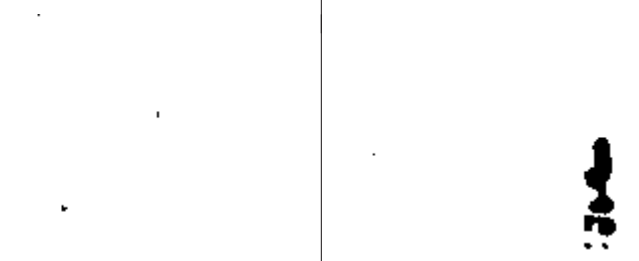
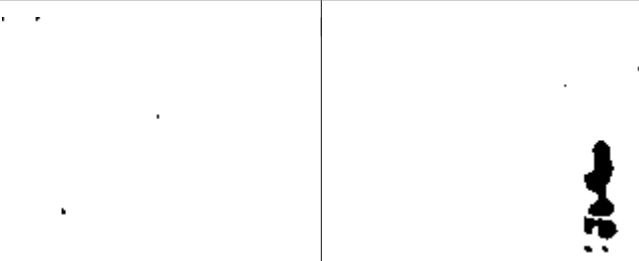
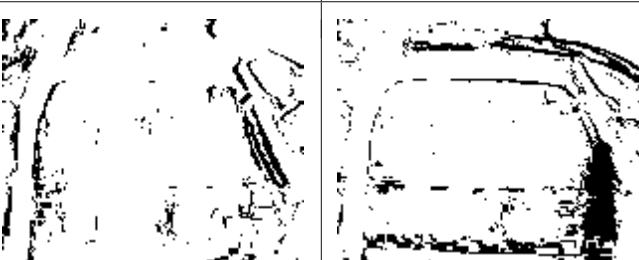
Secuencias <i>Fountain</i>	
Real	
Pixel	
Media	
MediaC	
Mediana	

Tabla 4.7: Etiquetas en secuencias *Fountain*, utilizando estadísticos


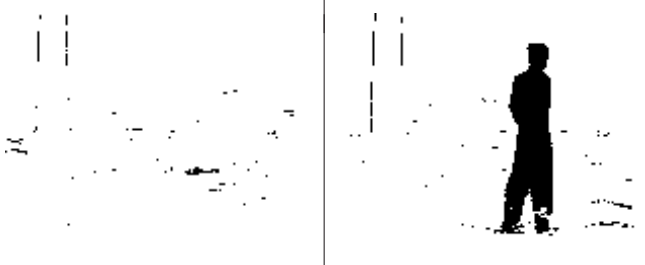
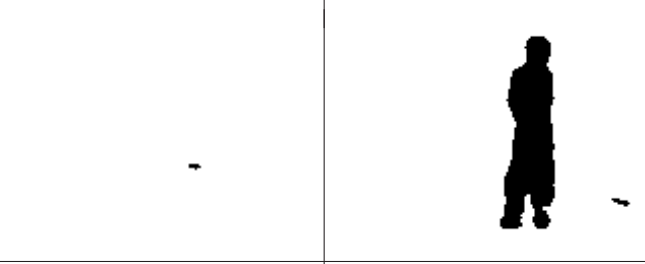
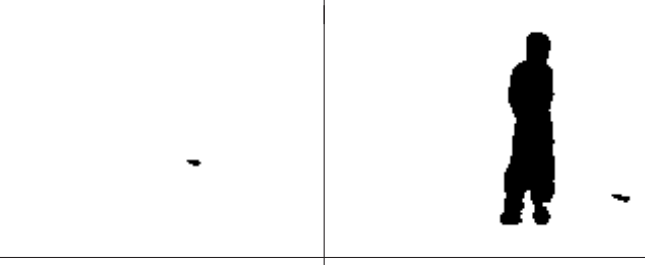
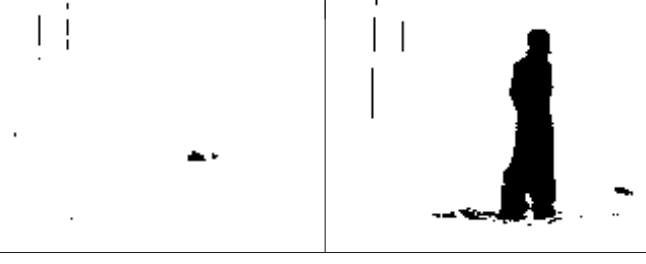
Secuencias <i>WaterSurface</i>	
Real	
Pixel	
Media	
MediaC	
Mediana	

Tabla 4.8: Etiquetas en secuencias *WaterSurface*, utilizando estadísticos


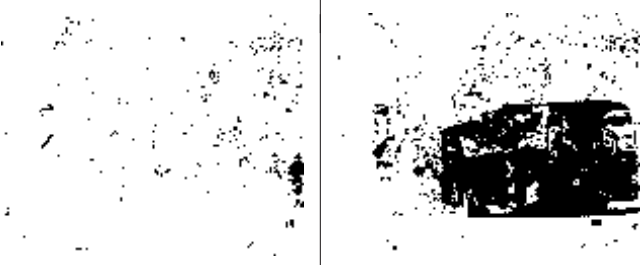


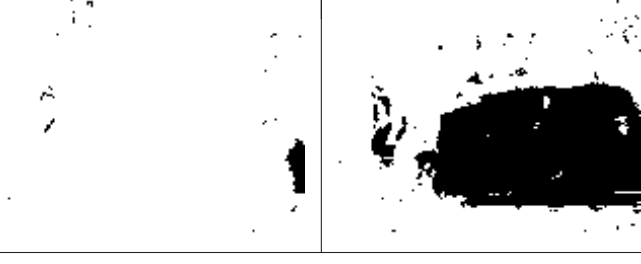
Secuencias <i>Campus</i>	
Real	
Pixel	
Media	
MediaC	
Mediana	

Tabla 4.9: Etiquetas en secuencias *Campus*, utilizando estadísticos

## 4.5 Espacios de color

El espacio de color RGB generalmente no es utilizado al realizar comparaciones o cálculos, esto debido a que pequeñas perturbaciones que se apliquen al los canales RGB ocasionaría que el color fuera muy diferente. El trabajo que se tomó de base [Teixeira and Corte-Real, 2007] utiliza el espacio de color YUV y a comparación con el espacio RGB, los resultados mejoran notablemente. Se utilizaron diversos espacios de color esperando minimizar el efecto que tienen los efectos de iluminación en los resultados, pero desafortunadamente el espacio de color YUV fué el que presentó los mejores resultados. Las transformaciones entre los espacios de color se encuentran en el apéndice B. Entre los espacios de color utilizados se encuentran los siguientes:

**HSV:** El espacio de color HSV (hue, saturation, value), pertenece al grupo de sistemas de color orientados al color (Hue) y que se asimilan de manera más cercana a la percepción del color. El sistema de coordenadas del color HSV, propuesto originalmente por Smith [Smith, 1978], es un cilindro y por conveniencia esta representado con un hexágono, tal como se observa en la figura 4.7(a). El tinte ( $H$ ) se mide con ángulos alrededor del eje vertical y su rango de valores esta entre 0 y  $360^\circ$ , iniciando con rojo ( $0^\circ$ ). La saturación  $S$  es el radio que toma valores entre 0 y 1; estos componentes indican que tan lejos el color esta del gris de igual iluminación. El valor  $V$  también toma valores entre 0 y 1, y es la medida del brillo. En el origen  $V = 0$  y este punto corresponde al negro; en este valor en particular,  $H$  y  $S$  son indefinidos [Guan et al., 2000].

**HSI:** El espacio de color HSI (hue, saturation, intensity) separa la información de color de la información de intensidad; la información de color está dada por las componentes  $H$  y  $S$ ; mientras que la de intensidad por  $I$ . El sistema coordenados del color HSI es un bicono con base triangular. La descripción geométrica del espacio de color HSI según [Gonzales and Woods, 1992] aparece en la figura 4.7(b).

**HSL:** El espacio de color HSL está representado por un doble cono hexagonal [Grest et al., 2003] como se muestra en la figura 4.7(c) con blanco en la cima y negro en el fondo y con colores completamente saturados alrededor del borde y con grises en el centro. Notese que el canal del tono (hue) en los espacios HSL y HSV es de la misma manera, pero la manera en la que se representa la saturación es difiere dramáticamente.

**L\*a\*b, L\*u\*v:** Los espacios de color desarrollados por la Comisión Internacional de Iluminación (CIE - Commission International de l'Éclairage) tienen la capacidad de representar las diferencias de color percibidas a través de la distancia euclideana y son considerados como una aproximación del sistema visual humano. El espacio CIE XYZ es el resultados de mediciones directas sobre el ojo humano hechas a finales de 1920 por W. Davis Wright [Wright, 1929] y John Guild [Guild, 1932]. A partir de XYZ, por una transformación no lineal, se calculan los valores en los espacios L\*u\*v y L\*a\*b.

El componente  $L^*$  es la luminancia,  $L^*=0$  indica negro y  $L^*=100$ , blanco. Los componentes  $a^*$  y  $u^*$  representan la posición entre el morado y verde; valores negativos de  $a^*$  y  $u^*$  indican cuanto verde hay, mientras que valores positivos indican cuanto hay de morado. Los componentes  $b^*$  y  $v^*$  representan la posición entre amarillo y azul. Valores negativos de  $b^*$  y  $v^*$  indican cuanto hay de azul, mientras que valores positivos indican cuanto hay de amarillo. La relación no lineal en las expresiones que definen los espacios  $L^*a^*b^*$  y  $L^*u^*v^*$  tienen como objetivo simular la respuesta logarítmica del ojo humano [Martínez, 2007]. La representaciones geométricas de estos espacios se muestran en las figuras 4.7(d), 4.7(e).

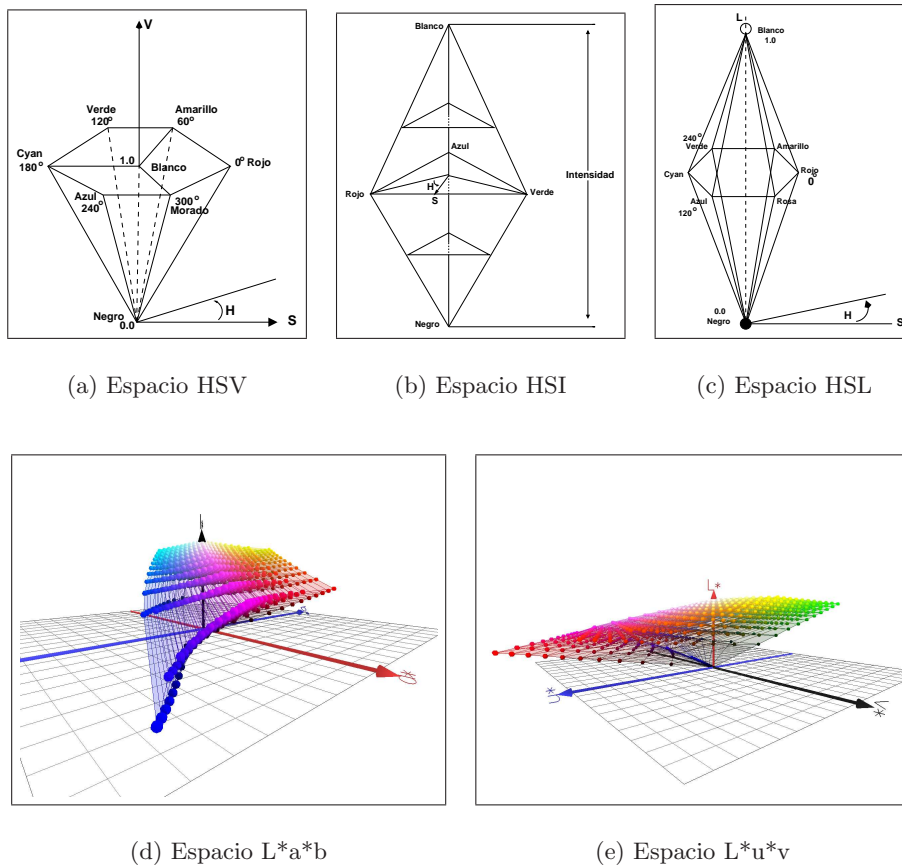


Figura 4.7: Espacios de color

Algunos resultados comparativos entre los espacios de color se muestran en las tablas 4.10 - 4.14. El algoritmo utilizado fue el Algoritmo Effective (algoritmo 5). Se puede observar en la figura que el espacio Lab tiene resultados muy buenos, ya que en general no se ve tan afectado por las variaciones en la iluminación, pero en algunas de las secuencias no detecta las regiones en movimiento; por lo cual se utilizará el espacio de color YUV. Por ejemplo, obsérvense los

buenos resultados obtenidos con el espacio de color Lab en las secuencias *Fountain*, *PETS2001\_DS3\_TE1\_C1*, *PETS2000*, *WaterSurface*, pero en la secuencia *HighWay1* la detección de objetos en movimiento es incompleta.













		Secuencias <i>Fountain</i>	
		Seq-192	Seq-521
Real			
HSI			
HSL			
HSV			
LAB			
YUV			

Tabla 4.10: Resultados de detección de movimiento en varios espacios de color















		Secuencias <i>HighWay1</i>	
		Seq-99	Seq-361
Real			
HSI			
HSL			
HSV			
LAB			
YUV			

Tabla 4.11: Resultados de detección de movimiento en varios espacios de color










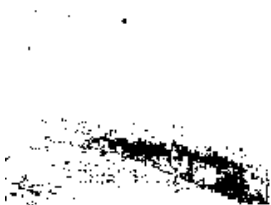
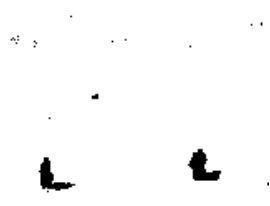

Secuencias <i>PETS2001_DS3_TE1_C1</i>		
	Seq-1447	Seq-870
Real		
HSI		
HSL		
HSV		
LAB		
YUV		

Tabla 4.12: Resultados de detección de movimiento en varios espacios de color





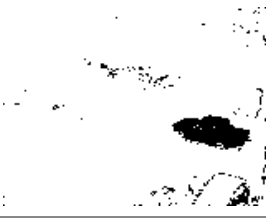
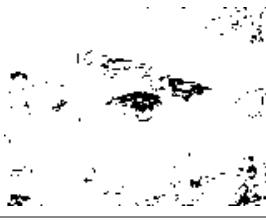


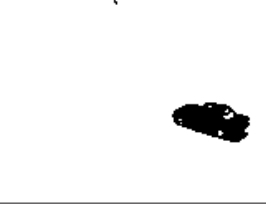

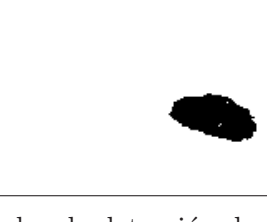
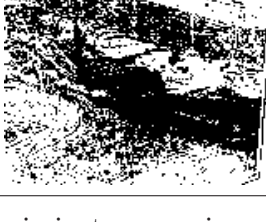
		Secuencias <i>PETS2000</i>	
		Seq-125	Seq-824
Real			
HSI			
HSL			
HSV			
LAB			
YUV			

Tabla 4.13: Resultados de detección de movimiento en varios espacios de color



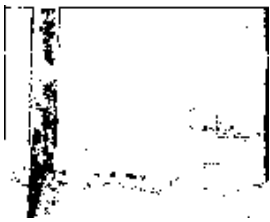
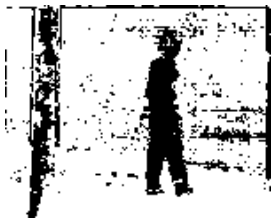
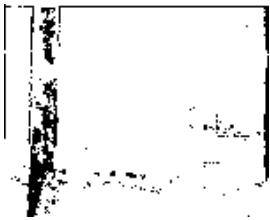

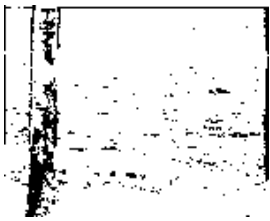





		Secuencias <i>WaterSurface</i>	
		Seq-337	Seq-559
Real			
HSI			
HSL			
HSV			
LAB			
YUV			

Tabla 4.14: Resultados de detección de movimiento en varios espacios de color

## 4.6 Prueba de ruido

En el trabajo de Teixeira [Teixeira and Corte-Real, 2007] utilizan una prueba de ruido introducida por Aach [Aach et al., 1993], la cual fué mencionada en la sección 3.1. Se buscó encontrar una prueba de ruido que resultara menos compleja y que mostrara los mismos resultados. En el trabajo de Grest [Grest et al., 2003] se presenta otra prueba de ruido (en la sección 3.1 se explica más a detalle esta prueba):

$$Ruido = \left(x_t - \hat{d}\right)^2 > 2\sigma_N^2. \quad (4.4)$$

donde  $x_t$  es el valor del pixel actual y  $\sigma_N^2$  es la varianza del ruido. Debido a que el costo computacional de 4.4 es mucho menor que la prueba utilizada en [Aach et al., 1993], se realizaron pruebas para realizar una comparación entre las dos pruebas. El resultado fué que el algoritmo del método de Cascada tenía resultados mejores, por lo cual no se modificó la prueba de ruido. Algunos resultados en donde se tuvieron detecciones de ruido se presentan en las tablas 4.15 y 4.16.




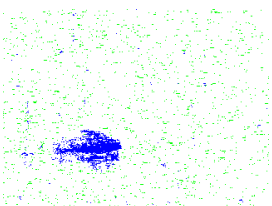
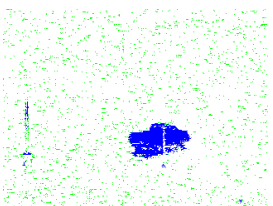
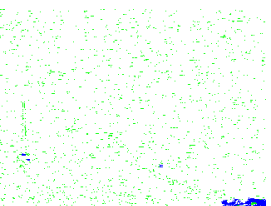

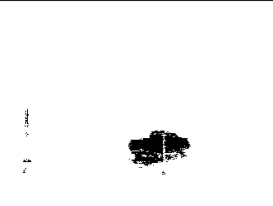
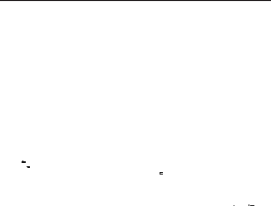
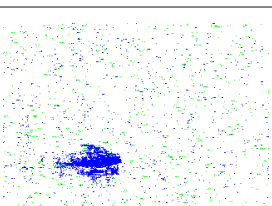
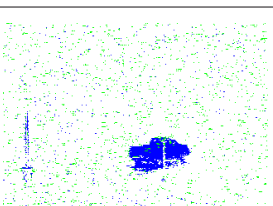
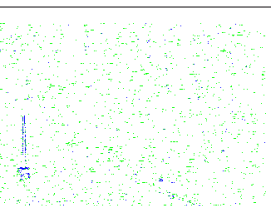
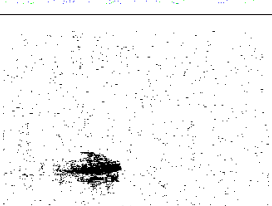
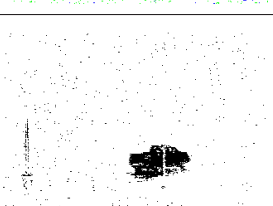
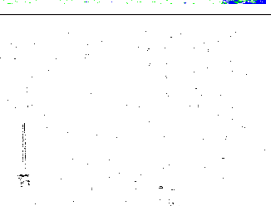
	Secuencias <i>Rain</i>		
Real			
Ruido1			
Movimiento1			
Ruido2			
Movimiento2			

Tabla 4.15: Pruebas de ruido en las secuencias *Rain*




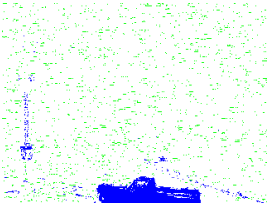
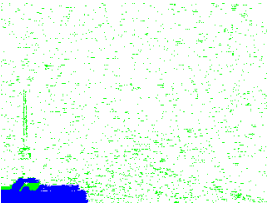
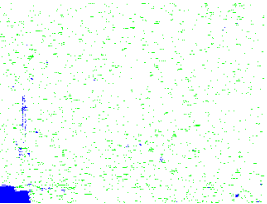



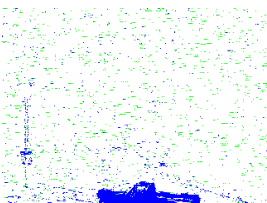
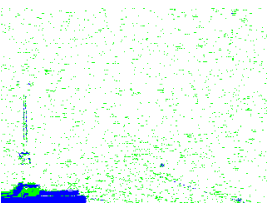
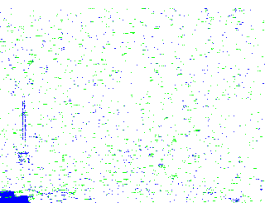


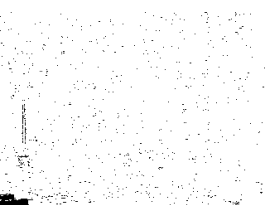
	Secuencias <i>Rain</i>		
Real			
Ruido1			
Movimiento1			
Ruido2			
Movimiento2			

Tabla 4.16: Pruebas de ruido en las secuencias *Rain*

## 4.7 Modificación del estadístico

Para determinar a que distribución de la mezcla pertenece un píxel en particular, en los trabajos de [Stauffer and Grimson, 1999], [Lee, 2005] y [Teixeira and Corte-Real, 2007] se utiliza la distancia de Mahalanobis (ecuación 2.5) y un umbral  $Th = 2.5\sigma$ , lo cual es equivalente a un nivel de confianza del 99.5% de una distribución normal. Lo anterior es utilizado cuando el píxel se obtiene de una imagen en niveles de gris; mientras que en imágenes a color, se tiene un umbral  $Th = 12.84$  equivalente al 99.5% de confianza de una distribución  $\mathcal{X}^2$ . Pero debido a la información que proporciona cada canal del espacio de color utilizado (YUV), se desea que el nivel de confianza sea mayor en los componentes que tienen la información del color al componente que contiene la información de luminosidad. Por lo cual, el estadístico utilizado para encontrar la concordancia de un píxel con alguna distribución en particular es el siguiente:

$$D_{k,j}^{*2} = \frac{(x_j - \mu_{k,j})^2}{\sigma_{k,j}}, \quad (4.5)$$

donde el índice  $k$  representa la distribución de la mezcla a utilizar en la evaluación,  $j$  representa el componente de color a utilizar. El nivel de confianza a utilizar es del 99.9% para el canal de luminosidad y 99.5% para los canales de color, quedando el umbral de la siguiente manera:  $Th = \{10.83, 7.88, 7.88\}$ ; se espera que al utilizar estos niveles de confianza personalizados para cada componente las variaciones en la iluminación no afecten demasiados a cada píxel. El algoritmo de esta prueba se muestra en el algoritmo 11. Comparación de resultados obtenidos utilizando el estadístico  $D_{k,j}^{*2}$  y el estadístico  $D_{k,j}^2$  se muestran en la tabla 4.17, donde se observa que la distancia propuesta (ecuación 4.5) no se ve afectada tan fácilmente por los cambios en la iluminación.

---

**Algoritmo 11:** Prueba de concordancia.

---

```

1 forall  $k$  do
2    $match = 0$ 
3   forall  $j \in \{YUV\}$  do
4      $D_{k,j}^{*2} = \frac{(x_j - \mu_{k,j})^2}{\sigma_{k,j}^2}$ 
5     if  $D_{k,j}^{*2} < Th_j$  then
6        $match ++$ 
7     end
8   end
9   if  $match == 3$  then
10    Hubo Concordancia
11  end
12 end
```

---




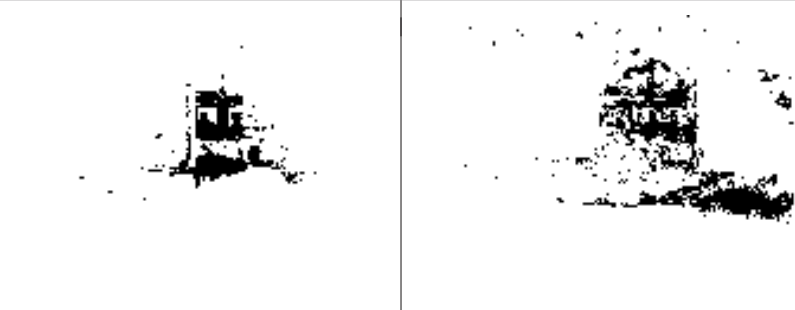

Secuencias <i>PETS2001_DS3_TE1_C1</i>	
Real	
Etiquetas $D^2$	
Etiquetas $D^{2*}$	

Tabla 4.17: Imágenes resultado al cambiar el nivel de confianza.

## 4.8 Algoritmo final

El algoritmo final de esta tesis es el resultado del análisis de parámetros del modelo de mezcla de Gaussianas y técnicas para detectar factores externos al movimiento. Entre las mejoras encontradas al analizar los parámetros se tienen las siguientes:

- Eliminación de ruido y mejor estimación de la función de densidad de los datos al umbralizar la varianza  $\sigma^2$ .
- Reducir el efecto de variaciones en la iluminación al actualizar  $\alpha$  en cada iteración.
- Reducir el efecto de variaciones en la iluminación al modificar la distancia para obtener la concordancia del píxel y la distribuciones de la mezcla de Gaussianas.
- Realizar la elección del número de distribuciones de la mezcla de manera dinámica independiente para cada píxel.
- Modelar la media de la vecindad del píxel bajo evaluación en lugar de sólo el valor.

Aunque se analizaron diversas técnicas para la detección de variación en la iluminación, ruido, cambios bruscos en la iluminación y sombras. No se tuvieron los resultados que se esperaban, por lo cual el algoritmo final presenta las pruebas de variación en la iluminación y ruido presentadas en el método de Cascada. Pero con las modificaciones realizadas se logra una mejora notable a comparación de los resultados obtenidos de la implementación realizada del método de Cascada.

El pseudocódigo final se muestra en el algoritmo 12.

---

**Algoritmo 12:** Algoritmo final para subtraer el fondo usando mezcla de gaussianas.

---

**Entrada:**  $V_0$

**Salida :**

```

1 #Inicialización de parámetros en cada píxel
2 forall  $i = 1 \dots N_{\text{Píxeles}}$  do
3    $MG[i].K = 1$ 
4    $MG[i].CG[1].\omega_1 = 1$ 
5    $MG[i].CG[1].\mu_1 = x_{j,1}$ 
6    $MG[i].CG[1].\sigma_j = V_0$ 
7    $MG[i].CG[1].c_j = 0$ 
8 end
9 forall  $I_t$  do
10  #Valor de alfa a utilizar
11   $Alfa\_Dual()$  (Ver algoritmo 10)
12  forall  $i = 1 \dots N_{\text{Píxeles}}$  do
13    if  $t\%L == 0$  then
14      | Eliminar componentes que tengan  $\omega < Th_{wt}$ 
15    end
16    #Obtener la concordancia del dato con respecto a cada
17    una de las distribuciones.
18     $Concordancia(blnAlfa)$  (ir a algoritmo 13)
19     $etiqueta = Obtener\_Etiqueta\_Effective()$ 
20    if  $etiqueta == No\_Fondo$  then
21      | if  $Prueba\_Ruido() > T_N$  then
22        | |  $etiqueta == Fondo$ 
23      else if  $Prueba\_Iluminacion() > T_I$  then
24        | |  $etiqueta == Fondo$ 
25      end
26    end
27     $Etiquetas_{i,t} = etiqueta$ 
28  end
29 end

```

---

**Algoritmo 13:** Concordancia(blnAlfa)

---

```

1 forall  $k = 1 \dots K$  do
2    $match = 0$ 
3   forall  $j \in \{YUV\}$  do
4     if  $\left(\frac{x - \mu_j}{\sigma_j}\right)^2 < Th[j]$  then  $match ++$ 
5      $p_j = \begin{cases} \omega_j \cdot g_j(x; \mu_j, \sigma_j) & \text{si } match == 3 \\ 0 & \text{en otro caso} \end{cases}$ 
6   end
7 end
8 if  $\sum_{j=1}^K p_j > 0$  then
9   forall  $k = 1 \dots K$  do
10    #Se calcula la probabilidad posterior de  $G_k$ 
11     $q_k = \frac{p_k}{\sum_{j=1}^M G[i].K p_j}$ 
12    if Winner_Take_All then  $q_k = \begin{cases} 1 & \text{si } k = \arg \max_j \{p_j\} \\ 0 & \text{en otro caso} \end{cases}$ 
13    if blnAlfa then
14       $\omega_k = (1 - \alpha_{fija}) \cdot \omega_k + \alpha_{fija} \cdot q_k$ 
15    else
16       $\omega_k = (1 - \alpha) \cdot \omega_k + \alpha \cdot q_k$ 
17    end
18    if  $q_k > 0$  then
19      forall  $j \in \dots YUV\}$  do
20         $c_k = c_k + q_k$ 
21        if blnAlfa then
22           $\eta_k = q_k \cdot \left(\frac{1 - \alpha_{fija}}{c_k} + -\alpha_{fija}\right)$ 
23        else
24           $\eta_k = q_k \cdot \left(\frac{1 - \alpha}{c_k} + \alpha\right)$ 
25        end
26         $\mu_k = (1 - \eta_k) \cdot \mu_k + \eta_k \cdot x$ 
27         $\sigma_k = (1 - \eta_k) \cdot \sigma_k^2 + \eta_k \cdot (x - \mu_k)^2$ 
28        if  $\sigma_{k,z} < SigmaMenor_z$  then  $\sigma_{k,z} = SigmaMenor_z$ 
29      end
30    end
31  end
32   $\forall_{j=1 \dots K} \omega_j = (1 - \alpha) \cdot \omega_j$ 
33  Se crea un nuevo componente gaussiano con
34   $\mu_{\hat{k}} = x \quad \sigma_k = V_0 \quad c_k = 1 \quad K ++$ 
35 end
36 #Normalización de los pesos  $\omega$ 
37  $\forall_{j=1 \dots K} \omega_k = \frac{\omega_k}{\sum_{k=1}^K \omega_k}$ 

```

---

Obsérvense algunos ejemplos de secuencias en los que era sumamente difícil la detección de objetos en movimiento y que mejoraron notablemente sus resultados.

En la tabla 4.18 se muestran ejemplos de las secuencias *HighWay1* y *PETS2000*, en donde los resultados del método Cascada se muestran en la segunda columna y resultados de este trabajo en la tercera. La detección de regiones en movimiento del método propuesto no se ve tan afectado por las variaciones en la iluminación y ruido que presentan las imágenes; además la actualización del parámetro  $\alpha$  mejora la inicialización del modelo de fondo, lo cual se observa más claramente en la secuencia *HighWay1*, la contiene objetos en movimiento durante toda la secuencia.

En la tabla 4.19 se muestran ejemplos de las secuencias *WaterSurface* y *PETS2001\_DS3\_TE1\_C1*. Lo complicado de la secuencia *WaterSurface* se debe al movimiento de las olas del mar, pero al modificar la estimación de la función de densidad del valor del pixel por el valor de la media de la vecindad del píxel bajo evaluación, los resultados mejoraron notablemente. Y la secuencia *PETS2001\_DS3\_TE1\_C1* presenta grandes cambios en la iluminación por lo cual es difícil el obtener buenos resultados. Pero el método propuesto logra adaptarse a los cambios en la iluminación y sigue teniendo detecciones de los objetos en movimiento.

Un problema que presenta este método es que se ve afectado por objetos en movimiento que presenten color similar al fondo.













		Métodos		
		Real	Cascada	Cascada Final
<i>HighWay1</i>	Seq-100			
	Seq-365			
<i>Pets2000</i>	Seq-146			
	Seq-816			

Tabla 4.18: Detección de objetos en movimiento de los métodos cascada y cascada final


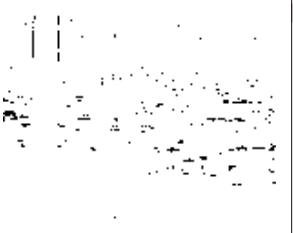











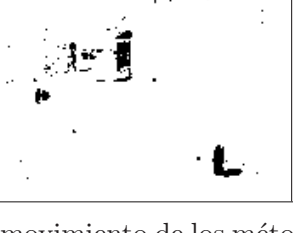
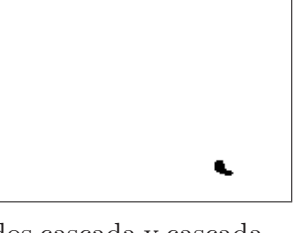
		Métodos		
		Real	Cascada	Cascada Final
<i>WaterSurface</i>	Seq-324			
	Seq-598			
<i>PETS2001_3_1</i>	Seq-201			
	Seq-824			
	Seq-2096			

Tabla 4.19: Detección de objetos en movimiento de los métodos cascada y cascada final





## Capítulo 5

---

# Conclusiones y trabajo futuro

---

### 5.1 Conclusiones

El análisis realizado en este trabajo mejoró de manera notable los resultados reportados anteriormente en métodos basados en mezclas de Gaussianas. Entre las aportaciones están las siguientes: se eliminaron los falsos positivos ocasionados por variaciones rápidas en la iluminación, lo cual se realizó de una manera sencilla que no implica un costo computacional alto, la modificación del estadístico utilizado para encontrar la concordancia entre las distribuciones se modificó de tal manera que el efecto del canal Y (luminosidad) no afecte a píxeles que son afectados por una variación en la iluminación, el umbral establecido en el parámetro  $\sigma$  aportó que los resultados filtren los efectos del ruido, la modificación realizada en el valor del píxel que se modela, esto es, utilizar la media de la vecindad del píxel bajo evaluación permitió que en escenas que presentan un fondo dinámico se redujera el porcentaje de falsos positivos; la utilización de un número de distribuciones de la mezcla que dependiera del comportamiento del píxel ocasionó que el costo computacional fuera menor además que mejoró la estimación de la función de densidad de los datos.

### 5.2 Trabajo futuro

Segmentar el fondo y los objetos en movimiento es una tarea complicada y más aún lo es el proporcionar un algoritmo que se adapte a todos los factores que afectan la detección de movimiento. Existe varias aportaciones que se pueden realizar para aumentar la efectividad de los algoritmo de sustracción de fondo:

- detección de sombras,
- estudiar el algoritmo EM e intentar utilizar las extensiones que se han desarrollado,

- investigar una manera efectiva de detectar las oscilaciones que existen entre píxeles, las cuales se presentan en fondos que son dinámicos,
- investigar la manera de realizar una etiquetación del estatus que tienen los píxeles en la escena, por ejemplo: píxel en movimiento, integrado al fondo, removido del fondo, dinámico, etc.,
- utilizar una técnica para la detección de la variación en la iluminación más robusta,
- analizar el parámetro  $\alpha$  para actualizarlo de tal manera que se involucre el número de aciertos que ha tenido cada píxel en las distribuciones de la mezcla.

---

# Bibliografía

---

- [Aach et al., 1993] Aach, T., Kaup, A., and Mester, R. (1993). Statistical model-based change detection in moving video. *Signal Process.*, 31(2):165–180. [cited at p. 46, 79]
- [Angulo and Serra, 2004] Angulo, J. and Serra, J. (2004). Traitements des images de couleur en représentation luminance/saturation/teinte par norme l1 = colour image processing from luminance/saturation/hue in l1 representation. *Traitement du Signal*, 21(6):583–604. [cited at p. 52]
- [Bilmes, 1998] Bilmes, J. A. (1998). A gentle tutorial on the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models. Technical Report TR-97-021, Department of Electrical Engineering and Computer Science U.C. Berkeley. [cited at p. 103]
- [Cheung and Kamath, 2004] Cheung, S.-c. S. and Kamath, C. (2004). Robust techniques for background subtraction in urban traffic video. In Panchanathan, S. and Vasudev, B., editors, *Visual Communications and Image Processing 2004. Edited by Panchanathan, Sethuraman; Vasudev, Bhaskaran. Proceedings of the SPIE, Volume 5308, pp. 881-892 (2004).*, volume 5308 of *Presented at the Society of Photo-Optical Instrumentation Engineers (SPIE) Conference*, pages 881–892. [cited at p. 18]
- [Cucchiara et al., 2003] Cucchiara, R., Grana, C., Piccardi, M., and Prati, A. (2003). Detecting moving objects, ghosts, and shadows in video streams. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(10):1337–1342. [cited at p. 48]
- [Dempster et al., 1977] Dempster, A., Laird, N., and Rubin, D. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38. [cited at p. 25, 101, 102]
- [Elgammal et al., 2002] Elgammal, A., Duraiswami, R., Harwood, D., and Davis, L. S. (2002). Background and foreground modeling using nonparametric kernel density for visual surveillance. In *Proceedings of the IEEE*, volume 90, pages 1151–1163. [cited at p. 48]
- [Elgammal et al., 2000] Elgammal, A. M., Harwood, D., and Davis, L. S. (2000). Non-parametric model for background subtraction. In *ECCV '00: Proceedings of the 6th European Conference on Computer Vision-Part II*, pages 751–767, London, UK. Springer-Verlag. [cited at p. 13]
- [Friedman and Russell, 1997] Friedman, N. and Russell, S. (1997). Image segmentation in video sequences : A probabilistic approach. In *Annual Conference on Uncertainty in Artificial Intelligence*, pages 175–181. [cited at p. 13, 25]

- [Gonzales and Woods, 1992] Gonzales, R. C. and Woods, R. E. (1992). *Digital Image Processing*. Addison-Wesley Publishing Company. [cited at p. 71]
- [Grest et al., 2003] Grest, D., Frahm, J.-M., and Koch, R. (2003). A color similarity measure for robust shadow removal in real time. In Ertl, T., editor, *Proceedings of the Vision, Modeling, and Visualization Conference 2003 (VMV 2003)*, pages 253–260. Aka GmbH. [cited at p. 46, 48, 49, 51, 71, 79]
- [Guan et al., 2000] Guan, L., Kung, S. Y., and Larsen, J. (2000). *Multimedia Image and Video Processing*. CRC Press. [cited at p. 71]
- [Guild, 1932] Guild, J. (1932). The colorimetric properties of the spectrum. *Philosophical Transactions of the Royal Society of London*, 230:149–187. [cited at p. 71]
- [Han et al., 2007] Han, B., Comaniciu, D., Zhu, Y., and Davis, L. (2007). Sequential kernel density approximation through mode propagation: Applications to background modeling. *IEEE Trans. Pattern Analysis Machine Intelligence*. [cited at p. 13]
- [Harville et al., 2001] Harville, M., Gordon, G. G., and Woodfill, J. (2001). Foreground segmentation using adaptive mixture models in color and depth. In *IEEE Workshop on Detection and Recognition of Events in Video*, pages 3–11. [cited at p. 25, 27]
- [Heikkilä et al., 2004] Heikkilä, M., Pietiläinen, M., and Heikkilä, J. (2004). A texture-based method for detecting moving objects. In *Proc. the 15th British Machine Vision Conference (BMVC 2004)*, pages 187–196. [cited at p. 35]
- [Horprasert et al., 1999] Horprasert, T., Harwood, D., and Davis, L. S. (1999). A statistical approach for real-time robust background subtraction and shadow detection. In *In IEEE ICCV99 Frame-rate workshop*. [cited at p. 13]
- [Hsu et al., 1984] Hsu, Y., Nagel, H., and Rekers, G. (1984). New likelihood test methods for change detection in image sequences. *Computer Vision Graphics Image Processing*, 26(1):73–106. [cited at p. 42]
- [Hu et al., 2004] Hu, W., Tan, T., Wang, L., and Maybank, S. (2004). A survey on visual surveillance of object motion and behaviors. *SMC-C*, 34(3):334–352. [cited at p. 12]
- [Huwer and Niemann, 2000] Huwer, S. and Niemann, H. (2000). Adaptive change detection for real-time surveillance applications. In *Proceedings of the Third IEEE International Workshop on Visual Surveillance (VS'2000)*, pages 37–45, Washington, DC, USA. IEEE Computer Society. [cited at p. 13]
- [Jain et al., 1977] Jain, R., Miltzer, D., and Nagel, H. (1977). Separating non-stationary from stationary scene components in a sequence of real world tv images. In *Proceedings of the 5th IJCAI*, pages 612–618, Cambridge, MA. [cited at p. 42]
- [KaewTraKulPong and Bowden, 2001] KaewTraKulPong, P. and Bowden, R. (2001). An improved adaptive background mixture model for realtime tracking with shadow detection. *European Workshop on Advanced Video-based Surveillance Systems*. [cited at p. 27]
- [KaewTrakulPong and Bowden, 2003] KaewTrakulPong, P. and Bowden, R. (2003). A real time adaptive visual surveillance system for tracking low-resolution colour targets in dynamically changing scenes. *Image and Vision Computing*, 21(10):913–929. [cited at p. 48]
- [Lee, 2005] Lee, D.-S. (2005). Effective gaussian mixture learning for video background subtraction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(5):827–832. [cited at p. 25, 26, 27, 30, 55, 62, 82]

- [Li et al., 2004] Li, L., Huang, W., Gu, I. Y.-H., and Tian, Q. (2004). Statistical modeling of complex backgrounds for foreground object detection. *IEEE Transactions on Image Processing*, 13(11):1459–1472. [cited at p. 13]
- [Li and Leung, 2002] Li, L. and Leung, M. K. H. (2002). Integrating intensity and texture differences for robust change detection. *IEEE Transactions on Image Processing*, 11(2):105–112. [cited at p. 48]
- [Liu et al., 1998] Liu, S.-C., Fu, C.-W., and Chang, S. (1998). Statistical change detection with moments under time-varying illumination. *IEEE Transactions on Image Processing*, 7(9):1258–1268. [cited at p. 42]
- [Martínez, 2007] Martínez, T. E. A. (2007). *Segmentación lingüística del color mediante un modelo bayesiano jerárquico*. PhD thesis, Centro de Investigación en Matemáticas A.C. [cited at p. 72]
- [McFarlane and Schofield, 2005] McFarlane, N. and Schofield, C. (2005). Segmentation and tracking of piglets in images. *Machine Vision and Applications*, 8(3):187–193. [cited at p. 18]
- [McKenna et al., 2000] McKenna, S. J., Jabri, S., Duric, Z., Rosenfeld, A., and Wechsler, H. (2000). Tracking groups of people. *Computer Vision and Image Understanding: CVIU*, 80(1):42–56. [cited at p. 48]
- [Mester et al., 2001] Mester, R., Aach, T., and Dömbgen, L. (2001). Illumination-invariant change detection using a statistical colinearity criterion. In Radig, B. and Florczyk, S., editors, *Pattern Recognition: Proceedings 23rd DAGM Symposium, Lecture Notes in Computer Science 2191, Springer Verlag*, volume 2191/2001, pages 170–177, Munich, Germany. Springer-Verlag. [cited at p. 42, 43]
- [Mester and Muhlich, 2001] Mester, R. and Muhlich, M. (2001). Improving motion and orientation estimation using an equilibrated total least squares approach. In *2001 International Conference on Image Processing (ICIP)*, volume 2, pages 929–932. [cited at p. 45]
- [Mittal and Paragios, 2004] Mittal, A. and Paragios, N. (2004). Motion-based background subtraction using adaptive kernel density estimation. *2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'04)*, 2:302–309. [cited at p. 13]
- [Monnet et al., 2003] Monnet, A., Mittal, A., Paragios, N., and Ramesh, V. (2003). Background modeling and subtraction of dynamic scenes. In *ICCV '03: Proceedings of the Ninth IEEE International Conference on Computer Vision*, pages 1305–1312, Washington, DC, USA. IEEE Computer Society. [cited at p. 13]
- [Neal and Hinton, 1999] Neal, R. M. and Hinton, G. E. (1999). A view of the em algorithm that justifies incremental, sparse, and other variants. In Jordan, M. I., editor, *Learning in graphical models*, pages 355–368. MIT Press. [cited at p. 25]
- [Nowlan, 1991] Nowlan, S. J. (1991). *Soft competitive adaptation: neural network learning algorithms based on fitting statistical mixtures*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, USA. [cited at p. 26]
- [Piccardi and Jan, 2004] Piccardi, M. and Jan, T. (2004). Mean-shift background image modelling. *International Conference on Image Processing (ICIP'04)*, 5:3399–3402. [cited at p. 13]

- [Power and Schoonees, 2002] Power, P. W. and Schoonees, J. A. (2002). Understanding background mixture models for foreground segmentation. In *Proceedings Image Vision Computing New Zealand 2002*, pages 267–271. [cited at p. 22]
- [Prati et al., 2001] Prati, A., Mikic, I., Grana, C., and Trivedi, M. (2001). Shadow detection algorithms for traffic flow analysis: A comparative study. In *Proceedings IEEE Intelligent Transportation Systems*, pages 340–345, Oakland, CA, USA. [cited at p. 48]
- [Richardson and Green, 1997] Richardson, S. and Green, P. (1997). On bayesian analysis of mixtures with an unknown number of components. *Journal of the Royal Statistical Society*, 60(Series B):731–792. [cited at p. 30]
- [Ridder et al., 1995] Ridder, C., Munkelt, O., and Kirchner, H. (1995). Adaptive background estimation and foreground detection using kalman filtering. In *Proceedings of International Conference on recent Advances in Mechatronics (ICRAM'95)*, pages 193–199. [cited at p. 13]
- [Salvador et al., 2004] Salvador, E., Cavallaro, A., and Ebrahimi, T. (2004). Cast shadow segmentation using invariant color features. *Computer Vision Image Understanding*, 95(2):238–259. [cited at p. 48]
- [Sato and Ishii, 2000] Sato, M.-A. and Ishii, S. (2000). On-line em algorithm for the normalized gaussian network. *Neural Computation*, 12(2):407–432. [cited at p. 25]
- [Silveira Jacques Jr. et al., 2005] Silveira Jacques Jr., J. C., Jung, C. R., and Musse, S. R. (2005). Background subtraction and shadow detection in grayscale video sequences. In *SIBGRAPI '05: Proceedings of the XVIII Brazilian Symposium on Computer Graphics and Image Processing*, page 189, Washington, DC, USA. IEEE Computer Society. [cited at p. 49, 50]
- [Skifstad and Jain, 1989] Skifstad, K. and Jain, R. (1989). Illumination independent change detection for real world image sequences. *Computer Vision Graphics Image Processing*, 46(3):387–399. [cited at p. 42]
- [Smith, 1978] Smith, A. R. (1978). Color gamut transform pairs. In *SIGGRAPH '78: Proceedings of the 5th annual conference on Computer graphics and interactive techniques*, pages 12–19, New York, NY, USA. ACM. [cited at p. 71]
- [Stauffer and Grimson, 1999] Stauffer, C. and Grimson, W. (1999). Adaptive background mixture models for real time tracking. *IEEE International Conference on Computer Vision and Pattern Recognition*, 19:246–252. [cited at p. 13, 20, 21, 25, 26, 27, 30, 35, 55, 57, 82]
- [Tan et al., 2006] Tan, R., Huo, H., Qian, J., and Fang, T. (2006). Traffic video segmentation using adaptive-k gaussian mixture model. In Heidelberg, S. B. ., editor, *Advances in Machine Vision, Image Processing, and Pattern Analysis*, volume 4153/2006 of *Lecture Notes in Computer Science*, pages 125–134. [cited at p. 30, 62]
- [Teixeira et al., 2007] Teixeira, L. F., Cardoso, J. S., and Corte-Real, L. (2007). Object segmentation using background modelling and cascaded change detection. *Journal of Multimedia (JMM)*, 2:55–65. [cited at p. 35]
- [Teixeira and Corte-Real, 2007] Teixeira, L. F. and Corte-Real, L. (2007). Cascaded change detection for foreground segmentation. *IEEE Motion and Video Computing*, page 6. [cited at p. 34, 42, 46, 55, 62, 71, 79, 82]

- [Tian et al., 2005] Tian, Y.-L., Lu, M., and Hampapur, A. (2005). Robust and efficient foreground analysis for real-time video surveillance. *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1*, pages 1182–1187. [cited at p. 47, 48]
- [Wren et al., 1997] Wren, C. R., Azarbayejani, A., Darrell, T., and Pentland, A. P. (1997). Pfnder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):780–785. [cited at p. 12]
- [Wright, 1929] Wright, W. D. (1929). A re-determination of the trichromatic coefficients of the spectral colours. *Transactions of the Optical Society*, 30:141–164. [cited at p. 71]





# Apéndices



## Apéndice A

---

# Algoritmo Expectation-Maximization

---

El algoritmo EM, es un método iterativo para encontrar la estimación de máxima verosimilitud de los parámetros cuando los datos son incompletos. Un conjunto de datos se dice que es incompleto cuando la distribución de la que proviene tiene parámetros escondidos, desconocidos o perdidos. El algoritmo fue introducido por [Dempster et al., 1977].

Sea  $\mathcal{X}$  un conjunto de datos incompletos, asumiendo la existencia de un conjunto de datos completos  $\mathcal{Z} = (\mathcal{X}, \mathcal{Y})$ , una función de densidad conjunta puede ser especificada,

$$p(z|\Theta) = p(x, y|\Theta) = p(x|\Theta) p(y|x, \Theta).$$

donde  $\mathcal{Y}$  denota los datos desconocidos de  $\mathcal{Z}$  y  $x, y, z$  son elementos de  $X, Y, Z$ . Usando esta función de densidad una nueva función de verosimilitud, llamada función de verosimilitud de los datos completos, puede ser definida como

$$\mathcal{L}(\Theta|\mathcal{Z}) = \mathcal{L}(\Theta|\mathcal{X}, \mathcal{Y}) = \prod_{i=1}^N p(x_i|\Theta) p(y_i|x_i, \Theta).$$

Dejando  $\Theta$  fija, la función de verosimilitud de los datos completos se convierte en función de la variable  $\mathcal{Y}$ , esto es

$$\mathcal{L}(\Theta|\mathcal{X}, \mathcal{Y}) = l(\mathcal{Y} | \mathcal{X}, \Theta).$$

Asumiéndose que  $\mathcal{Y}$  es una instancia de la variable aleatoria  $Y$ , se puede calcular el valor esperado de  $\log l(Y | \mathcal{X}, \Theta)$ . El algoritmo EM usa un conjunto dado de parámetros  $\Theta_{t-1}$  para evaluar la densidad marginal de  $Y$  dado  $\mathcal{X}$  y usa esta función para calcular el valor esperado de la función de verosimilitud de los datos completos; este paso del algoritmo es conocido como el paso de Esperanza o paso-E. Dado que  $\Theta$  es fijo, el valor esperado será una función determinista de

$\Theta$ . [Dempster et al., 1977] definen una función  $Q$  como

$$Q(\Theta^t, \Theta^{t-1}) = E[\log l(Y|\mathcal{X}, \Theta^t) | \mathcal{X}, \Theta^{t-1}], \quad (\text{A.1})$$

donde  $\Theta^t = \Theta$  y la parte de la derecha es evaluada como

$$E[\log l(Y|\mathcal{X}, \Theta^t) | \mathcal{X}, \Theta^{t-1}] = \int_{y \in D_Y} \log l(y|\mathcal{X}, \Theta^t) p(y|\mathcal{X}, \Theta^{t-1}) dy,$$

donde  $D_Y$  es el espacio de valores que  $y$  puede tomar.

El segundo y último paso del EM es maximizar  $Q$  con respecto de  $\Theta^t$ , esto es

$$\Theta^* = \arg \max_{\Theta^t} Q(\Theta^t, \Theta^{t-1}).$$

Este paso del algoritmo es conocido como el paso de Maximización o paso-M.

El algoritmo EM entonces usa  $\Theta^*$  como  $\Theta^{t-1}$  en A.1 y los dos pasos del algoritmo son repetidos. Cada iteración del algoritmo garantiza incrementar la función de verosimilitud hasta que un máximo local es alcanzado. Una prueba matemática de lo anterior, así como de información de la velocidad de convergencia pueden ser encontrada en [Dempster et al., 1977].

## A.1 Aplicando EM al modelo de mezcla de gaussianas

Una manera de resolver el problema de estimar los parámetros de máxima verosimilitud de un modelo de mezclas es usar el algoritmo EM. Considérese una variable aleatoria  $X$  representada por una mezcla que consiste de  $K$  procesos aleatorios, cada uno con su propia función de densidad de probabilidad, la cual es:

$$p(x | \Theta) = \sum_{k=1}^K \omega_k p_k(x | \theta_k),$$

donde  $\theta_k$  denota el conjunto de parámetros de la  $k$ -ésima función de probabilidad.

Una vez más, sea  $\mathcal{X} = \{x_1, \dots, x_N\}$  una variable que denota un conjunto de  $N$  muestras tomadas independientemente de  $X$ . Ya que las muestras son independientes e idénticamente distribuidos por  $p(x | \Theta)$ , la función log-verosimilitud sería

$$\log \mathcal{L}(\Theta | \mathcal{X}) = \log \prod_{i=1}^N p(x_i, | \Theta) = \sum_{i=1}^N \log \sum_{k=1}^K \omega_k p_k(x_i | \theta_k). \quad (\text{A.2})$$

La ecuación A.2 es difícil de maximizar. Sin embargo, asumiéndose el proceso adyacente como disjunto, la ecuación A.2 puede ser simplificada si se conoce qué proceso generó cada muestra; sea  $k_i \in \{1, \dots, K\}$  una variable que es conocida,

la cual indica esa información, por ejemplo, si  $k_i = k$ , la muestra  $i$  –ésima fue generada por el proceso  $k$  –ésimo. La expresión puede entonces reducirse a

$$\log \mathcal{L}(\Theta | \mathcal{X}) = \sum_{i=1}^N \log \omega_{k_i} p_{k_i}(x_i | \theta_{k_i}).$$

Por consiguiente, asúmase que los datos  $\mathcal{X}$  son incompletos en el sentido que existe un dato no observado que indica cual proceso generó cada muestra. Llámese a ese conjunto de datos no conocidos como  $\mathcal{K} = \{k_1, \dots, k_N\}$  y sea  $\mathcal{Z} = (\mathcal{X}, \mathcal{K})$  el conjunto de datos completos. Si  $\mathcal{K}$  es conocido la log-verosimilitud completa de los datos sería

$$\log \mathcal{L}(\Theta | \mathcal{X}, \mathcal{Z}) = \sum_{i=1}^N \log \omega_{k_i} p_{k_i}(x_i | \theta_{k_i}).$$

Sin embargo, si  $\mathcal{K}$  es desconocida y se asume que  $\mathcal{K}$  es una instancia de una variable aleatoria  $K$ , se puede proceder con el paso-E del algoritmo. Todo lo que se necesita es una expresión de la densidad marginal de  $K$  dado  $X$ . Nótese, que en este caso, una muestra  $k$  de  $K$  será un vector de  $N$  elementos. Usando la regla de Bayes es fácil encontrar una expresión de la probabilidad condicional de  $k_i$  dado  $x_i$  y un vector de parámetros  $\Theta^{t-1}$

$$p(k_i | x_i, \Theta^{t-1}) = \frac{\omega_{k_i} p_{k_i}(x_i | \theta_{k_i}^{t-1})}{\sum_{k=1}^K \omega_k p_k(x_i | \theta_k^{t-1})}.$$

Los parámetros desconocidos  $\omega_k$  pueden obtenerse, en este caso, como probabilidades a priori  $p(k_i | \theta^{t-1})$ . Ya que las muestras son independientes, la densidad marginal es

$$p(\mathcal{K} | \mathcal{X}, \Theta^{t-1}) = \prod_{i=1}^N p(k_i | x_i, \Theta^{t-1}).$$

La ecuación A.1 toma la forma

$$\begin{aligned} Q(\Theta^t, \Theta^{t-1}) &= E[\log l(K | \mathcal{X}, \Theta^t) | \mathcal{X}, \Theta^{t-1}] \\ &= \sum_{k \in D_k} \log l(k | \mathcal{X}, \Theta^t) p(k, | \mathcal{X}, \Theta^{t-1}) \\ &= \text{después de un poco de álgebra...} \\ &= \sum_{k=1}^K \sum_{i=1}^N \log(\omega_k p_k(x_i | \theta_k^t)) p(k | x_i, \Theta^{t-1}). \end{aligned}$$

Con la excepción de determinar  $\omega_k$ , calcular el paso-M es cuestión de elegir componentes de funciones de densidad y diferenciar  $Q(\Theta^t, \Theta^{t-1})$  con respecto de varios parámetros e igualar a 0. No es necesario especificar componentes de funciones de densidad para determinar  $\omega_k$ . [Bilmes, 1998] muestra que cuando se usan componentes de funciones de densidad gaussianas, es posible encontrar una expresión analítica para todos los nuevos parámetros  $\Theta^t$  a partir de los parámetros

anteriores  $\Theta^{t-1}$ . Usando una función de densidad Gaussiana con parámetros  $\theta_k = (\mu_k, \Sigma_k)$ ,

$$p_k(x | \mu_k, \Sigma_k) = \eta(x | \mu_k, \Sigma_k) = \frac{1}{(2\pi)^{n/2} |\Sigma_k|^{1/2}} \exp\left(-\frac{1}{2}(x - \mu_k)^T \Sigma_k^{-1} (x - \mu_k)\right),$$

las expresiones analíticas resultantes serían

$$\omega_k^t = \frac{1}{N} \sum_{i=1}^N p(k | x_i, \Theta^{t-1}), \quad (\text{A.3})$$

$$\mu_k^t = \frac{\sum_{i=1}^N x_i p(k | x_i, \Theta^{t-1})}{\sum_{i=1}^N p(k | x_i, \Theta^{t-1})}, \quad (\text{A.4})$$

$$\Sigma_k^t = \frac{\sum_{i=1}^N p(k | x_i, \Theta^{t-1}) (x_i - \mu_k^t) (x_i - \mu_k^t)^T}{\sum_{i=1}^N p(k | x_i, \Theta^{t-1})}. \quad (\text{A.5})$$

## A.2 Modificación a ecuaciones online

Dado que se espera modelar escenas que cambian con el tiempo, el píxel  $X$  no se puede considerar que se comporte como un proceso estacionario por un largo periodo de tiempo; por lo que se especificará un número  $N$  finito de muestras para la estimación de los parámetros  $\Theta$ . Las ecuaciones A.3-A.5 se tienen que evaluar en cada píxel de cada frame de la secuencia de imágenes, entonces lo que se necesita es tener ecuaciones que se puedan evaluar de manera recursiva. La ecuación recursiva derivada de A.3 para  $N + 1$  observaciones se puede obtener de la siguiente manera:

$$\begin{aligned} \omega_k^{N+1} &= \frac{1}{N+1} \sum_{i=1}^{N+1} p(k | x_i, \Theta^{t-1}) \\ &= \frac{1}{N+1} \sum_{i=1}^N p(k | x_i, \Theta^{t-1}) + \frac{1}{N+1} p(k | x_{N+1}, \Theta^{t-1}) \\ &= \frac{1}{N+1} \sum_{i=1}^N \omega_k^N + \frac{1}{N+1} p(k | x_{N+1}, \Theta^{t-1}) \\ &= \left(1 - \frac{1}{N+1}\right) \omega_k^N + \frac{1}{N+1} p(k | x_{N+1}, \Theta^{t-1}) \\ &= (1 - \alpha) \omega_k^N + \alpha p(k | x_{N+1}, \Theta^{t-1}) \end{aligned} \quad (\text{A.6})$$

donde  $\alpha = \frac{1}{N+1}$ . Aplicando la misma técnica en la ecuación A.4:

$$\begin{aligned}
\mu_k^{N+1} &= \frac{\sum_{i=1}^{N+1} x_i p(k | x_i, \Theta^{t-1})}{\sum_{i=1}^{N+1} p(k | x_i, \Theta^{t-1})} \\
&= \frac{\sum_{i=1}^N x_i p(k | x_i, \Theta^{t-1})}{\sum_{i=1}^{N+1} p(k | x_i, \Theta^{t-1})} + \frac{p(k | x_{N+1}, \Theta^{t-1})}{\sum_{i=1}^{N+1} p(k | x_i, \Theta^{t-1})} x_{N+1} \\
&= \frac{\sum_{i=1}^N p(k | x_i, \Theta^{t-1})}{\sum_{i=1}^{N+1} p(k | x_i, \Theta^{t-1})} \mu_k^N + \frac{p(k | x_{N+1}, \Theta^{t-1})}{\sum_{i=1}^{N+1} p(k | x_i, \Theta^{t-1})} x_{N+1} \\
&= \frac{\sum_{i=1}^{N+1} p(k | x_i, \Theta^{t-1}) - p(k | x_{N+1}, \Theta^{t-1})}{\sum_{i=1}^{N+1} p(k | x_i, \Theta^{t-1})} \mu_k^N + \frac{p(k | x_{N+1}, \Theta^{t-1})}{\sum_{i=1}^{N+1} p(k | x_i, \Theta^{t-1})} x_{N+1} \\
&= \left(1 - \frac{p(k | x_{N+1}, \Theta^{t-1})}{\sum_{i=1}^{N+1} p(k | x_i, \Theta^{t-1})}\right) \mu_k^N + \frac{p(k | x_{N+1}, \Theta^{t-1})}{\sum_{i=1}^{N+1} p(k | x_i, \Theta^{t-1})} x_{N+1} \\
&= (1 - \rho_k) \mu_k^N + \rho_k x_{N+1}, \tag{A.7}
\end{aligned}$$

donde

$$\rho_k = \frac{p(k | x_{N+1}, \Theta^{t-1})}{\sum_{i=1}^{N+1} p(k | x_i, \Theta^{t-1})} = \frac{p(k | x_{N+1}, \Theta^{t-1})}{(N+1) \omega_k^{N+1}} = \frac{\alpha \cdot p(k | x_{N+1}, \Theta^{t-1})}{\omega_k^{N+1}}. \tag{A.8}$$

Aplicando los mismos trucos algebraicos, de la ecuación A.5 se obtiene:

$$\begin{aligned}
\Sigma_k^{N+1} &= \frac{\sum_{i=1}^{N+1} p(k | x_i, \Theta^{t-1}) (x_i - \mu_k^{N+1}) (x_i - \mu_k^{N+1})^T}{\sum_{i=1}^{N+1} p(k | x_i, \Theta^{t-1})} \\
&= \dots \\
&= (1 - \rho_k) \Sigma_k^N + \rho_k (x_{N+1} - \mu_k^{N+1}) (x_{N+1} - \mu_k^{N+1})^T. \tag{A.9}
\end{aligned}$$

Escribiendo las ecuaciones recursivas obtenidas A.6-A.9 en términos de nuevos parámetros se obtiene:

$$\begin{aligned}
\omega_k^{new} &= (1 - \alpha) \omega_k + \alpha \cdot p(k | x_t, \Theta^{old}), \\
\mu_k^{new} &= (1 - \rho_k) \mu_k + \rho_k x_t, \\
\Sigma_k^{new} &= (1 - \rho_k) \Sigma_k + \rho_k (x_t - \mu_k^{new}) (x_t - \mu_k^{new})^T
\end{aligned}$$

donde

$$\begin{aligned}
\alpha &= \frac{1}{N+1}, \\
\rho_k &= \frac{\alpha \cdot p(k | x_t, \Theta^{old})}{\omega_k^{new}}.
\end{aligned}$$





## Apéndice B

---

# Conversión de espacios de color

---

---

**Algoritmo 14:** Conversión del espacio de color RGB al espacio HSI

---

**Entrada:**  $r, g, b \in \{0, 1\}$

**Salida :**  $h, s, i \in \{ \}$

$min = \text{Minimo}(r, g, b)$

$max = \text{Maximo}(r, g, b)$

$h = \text{acos} \left( \frac{0.5 * (r - g) + (r - b)}{(r - g)^2 + (r - b) * (g - b)} \right)$

**if**  $b > g$  **then**

  |  $h = 2\pi - h$

**end**

$s = \frac{max - min}{max}$

$i = max$

---

---

**Algoritmo 15:** Conversión del espacio de color HSI al espacio RGB
 

---

**Entrada:**  $h, s, i \in \{\}$

**Salida :**  $r, g, b \in \{0, 1\}$

**if**  $h \geq 0 \ \&\& \ h < 120$  **then**

$$b = \frac{1-s}{3}$$

$$r = \frac{1}{3} * \frac{1+(s*\cos h)}{\cos 60-h}$$

$$g = 1 - b - r$$

$$r = 3 * i * r$$

$$g = 3 * i * g$$

$$b = 3 * i * b$$

**else if**  $h \geq 120 \ \&\& \ h < 240$  **then**

$$h- = 120$$

$$r = \frac{1-s}{3}$$

$$g = \frac{1}{3} * \frac{1+s*\cos h}{\cos 60-h}$$

$$b = 1 - g - r$$

$$r = 3 * i * r$$

$$g = 3 * i * g$$

$$b = 3 * i * b$$

**else if**  $h \geq 240 \ \&\& \ h < 360$  **then**

$$h- = 240$$

$$g = \frac{1-s}{3}$$

$$b = \frac{1}{3} * \frac{1+s*\cos h}{\cos 60-h}$$

$$r = 1 - g - b$$

$$r = 3 * i * r$$

$$g = 3 * i * g$$

$$b = 3 * i * b$$

**end**

---

---

**Algoritmo 16:** Conversión del espacio de color RGB al espacio HSL
 

---

**Entrada:**  $r, g, b \in \{0, 1\}$

**Salida :**  $h, s, l \in \{ \}$

$min = \text{Minimo}(r, g, b)$

$max = \text{Maximo}(r, g, b)$

$delta = max - min$

**if**  $max == min$  **then**

  |  $h = 0$

**else if**  $max == r \ \&\& \ g \geq b$  **then**

  |  $h = 60 * ((g - b)/delta) + 0$

**else if**  $max == r \ \&\& \ g < b$  **then**

  |  $h = 60 * ((g - b)/delta) + 360$

**else if**  $max == g$  **then**

  |  $h = 60 * ((b - r)/delta) + 120$

**else if**  $max == b$  **then**

  |  $h = 60 * ((r - g)/delta) + 240$

**end**

$l = 0.5 * (max + min)$

**if**  $max == min$  **then**

  |  $s = 0$

**else if**  $*l \leq 0.5$  **then**

  |  $s = \frac{delta}{max+min}$

**else if**  $l > 0.5$  **then**

  |  $s = \frac{delta}{2-max-min}$

**end**

---

---

**Algoritmo 17:** Conversión del espacio de color HSL al espacio RGB

---

**Entrada:**  $h, s, l \in \{ \}$ **Salida** :  $r, g, b \in \{0, 1\}$ 

```

if  $s == 0$  then
  |  $r = l$     $g = l$     $b = l$ 
else
  | if  $l < 0.5$  then
  |   |  $q = l * (1 + s)$ 
  | else if  $l \geq 0.5$  then
  |   |  $q = l + s - (l * s)$ 
  |   |  $p = 2 * l - q$ 
  |   |  $hk = \frac{h}{360}$ 
  |   |  $t_R = hk + \frac{1}{3}$ 
  |   |  $t_G = hk$ 
  |   |  $t_B = hk - \frac{1}{3}$ 
  |   | forall  $C \in \{R, G, B\}$  do
  |     | if  $t_C < 0$  then
  |       |  $t_C = t_C + 1.0$ 
  |     | end
  |     | if  $t_C > 1$  then
  |       |  $t_C = t_C - 1.0$ 
  |     | end
  |     | if  $t_C < \frac{1}{6}$  then
  |       |  $C = p + (q - p) * 6 * t_C$ 
  |     | else if  $t_C \geq \frac{1}{6} \ \&\& \ t_C < 0.5$  then
  |       |  $C = q$ 
  |     | else if  $t_C \geq 0.5 \ \&\& \ t_C < \frac{2}{3}$  then
  |       |  $C = p + (q - p) * 6 * (\frac{2}{3} - t_C)$ 
  |     | else
  |       |  $C = p$ 
  |     | end
  |   | end
  | end
end

```

---

---

**Algoritmo 18:** Conversión del espacio de color RGB al espacio HSV
 

---

**Entrada:**  $r, g, b \in \{0, 1\}$   
**Salida :**  $h, s, v \in \{ \}$   
 $min = \text{Minimo}(r, g, b)$   
 $max = \text{Maximo}(r, g, b)$   
 $delta = max - min$   
**if**  $max == 0$  **then**  
 |  $s = 0$   
**else**  
 |  $s = 255 * \frac{delta}{max}$   
**end**  
 $v = max$   
**if**  $s \neq 0$  **then**  
 | **if**  $max == r$  **then**  
 | |  $h = (g - b)/delta$   
 | **end**  
 | **else if**  $maximo == g$  **then**  
 | |  $h = 2.0 + (b - r)/delta$   
 | **end**  
 | **else if**  $maximo == b$  **then**  
 | |  $h = 4.0 + (r - g)/delta$   
 | **end**  
**end**  
**else**  
 |  $h = -1$   
**end**  
 $h = *h * 60$   
**if**  $h < 0.0$  **then**  
 |  $h+ = 360$   
**end**

---

**Algoritmo 19:** Conversión del espacio de color HSV al espacio RGB**Entrada:**  $h, s, i \in \{\}$ **Salida** :  $r, g, b \in \{0, 1\}$  $hi = h/60.0f$      $s/ = 100.0f$      $v/ = 100.0f$ **if**  $s == 0$  **then** Escala de grises|  $r = g = b = v$ **else**|  $f = h/60 - hi$   $p = v * (1 - s)$   $q = v * (1 - s * f)$   $t = v * (1 - s * (1 - f))$ | **switch**  $hi$  **do**|    **case** 0|    |  $r = v; g = t; b = p; break$ |    **case** 1|    |  $r = q; g = v; b = p; break$ |    **case** 2|    |  $r = p; g = v; b = t; break$ |    **case** 3|    |  $r = p; g = q; b = v; break$ |    **case** 4|    |  $r = t; g = p; b = v; break$ |    **case** 5|    |  $r = v; g = p; b = q; break$ | **end****end****Algoritmo 20:** Conversión del espacio de color RGB al espacio LAB**Entrada:**  $r, g, b \in \{0, 1\}$ **Salida** :  $h, s, i \in \{\}$  $RGB2XYZ(r, g, b, x, y, z)$     Ir a algoritmo 22 $XYZ2LAB(x, y, z, l, a, lab_b)$     Ir a algoritmo 23**Algoritmo 21:** Conversión del espacio de color LAB al espacio RGB**Entrada:**  $h, s, i \in \{\}$ **Salida** :  $r, g, b \in \{0, 1\}$  $LAB2XYZ(l, a, lab_b, x, y, z)$     Ir a algoritmo 24 $XYZ2RGB(x, y, z, r, g, b)$     Ir a algoritmo 25**Algoritmo 22:** Conversión del espacio de color RGB al espacio LAB**Entrada:**  $r, g, b \in \{0, 1\}$ **Salida** :  $x, y, z \in \{0, 1\}$ 

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 0.4125 & 0.3576 & 0.1804 \\ 0.2125 & 0.7154 & 0.0721 \\ 0.0193 & 0.1192 & 0.9502 \end{bmatrix} \begin{bmatrix} r \\ g \\ b \end{bmatrix}$$

---

**Algoritmo 23:** Conversión del espacio de color XYZ al espacio LAB
 

---

**Entrada:**  $x, y, z \in \{ \}$   
**Salida :**  $l, a, b \in \{0, 1\}$   
 $X_n = 0.95050$   
 $Y_n = 1.00000$   
 $Z_n = 1.08870$   
 $L_t = 0.008856$   
 $L_0 = \frac{y}{255 * Y_n}$   
**if**  $L_0 > L_t$  **then**  
    |  $l = 116.0 * L_0^{\frac{1}{3}} - 16.0$   
    |  $fdeYY_n = L_0^{\frac{1}{3}}$   
**else**  
    |  $l = 903.3 * L_0$   
    |  $fdeYY_n = 7.787 * L_0 + \frac{16}{116}$   
**end**  
 $L_1 = \frac{x}{255 * X_n}$   
**if**  $L_1 > L_t$  **then**  
    |  $fdeXX_n = L_1^{\frac{1}{3}}$   
**else**  
    |  $fdeXX_n = 7.787 * L_1 + \frac{16}{116}$   
**end**  
 $L_2 = z / (255.0 * Z_n)$   
**if**  $L_2 > L_t$  **then**  
    |  $fdeZZ_n = L_2^{\frac{1}{3}}$   
**else**  
    |  $fdeZZ_n = 7.787 * L_2 + \frac{16}{116}$   
**end**  
 $a = 500 * (fdeXX_n - fdeYY_n)$   
 $b = 200 * (fdeYY_n - fdeZZ_n)$

---

**Algoritmo 24:** Conversión del espacio de color LAB al espacio XYZ

---

**Entrada:**  $r, g, b \in \{0, 1\}$   
**Salida :**  $x, y, z \in \{0, 1\}$

$$fY = \left(\frac{l+16.0}{116}\right)^3$$

**if**  $fY < 0.008856$  **then**  
  |  $fY = \frac{l}{903.3}$   
**end**  
 $y = fY$   
**if**  $fY > 0.008856$  **then**  
  |  $fY = fY^{\frac{1}{3}}$   
**else**  
  |  $fY = 7.787 * fY + \frac{16}{116}$   
**end**  
 $fX = \frac{a}{500} + fY$   
**if**  $fX > 0.206893$  **then**  
  |  $x = fX^3$   
**else**  
  |  $x = \frac{fX - \frac{16}{116}}{7.787}$   
**end**  
 $fZ = fY - \frac{b}{200}$   
**if**  $fZ > 0.206893$  **then**  
  |  $z = fZ^3$   
**else**  
  |  $z = \frac{fZ - \frac{16}{116}}{7.787}$   
**end**  
 $x = x * 0.950456 * 255$   
 $y = y * 255$   
 $z = z * 1.088754 * 255$

---

**Algoritmo 25:** Conversión del espacio de color XYZ al espacio RGB

---

**Entrada:**  $x, y, z \in \{0, 1\}$   
**Salida :**  $r, g, b \in \{0, 1\}$

$$\begin{bmatrix} r \\ g \\ b \end{bmatrix} = \begin{bmatrix} 3.240479 & -1.537150 & -0.498535 \\ -0.969256 & 1.875992 & 0.041556 \\ 0.055648 & -0.204043 & 1.057311 \end{bmatrix} + \begin{bmatrix} 0.5 \\ 0.5 \\ 0.5 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

**forall**  $c \in \{r, g, b\}$  **do**  
  | **if**  $c < 0$  **then**  
  |  |  $c = 0$   
  | **else if**  $c > 255$  **then**  
  |  |  $c = 255$   
  | **end**  
**end**

---



---

**Algoritmo 26:** Conversión del espacio de color RGB al espacio YUV

---

**Entrada:**  $x, y, z \in \{0, 1\}$

**Salida :**  $l, a, b \in \{0, 1\}$

$$\begin{bmatrix} y \\ u \\ v \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.168736 & -0.331264 & 0.5 \\ 0.5 & -0.418688 & -0.081312 \end{bmatrix} + \begin{bmatrix} 0 \\ 128 \\ 128 \end{bmatrix} \begin{bmatrix} r \\ g \\ b \end{bmatrix}$$


---

---

**Algoritmo 27:** Conversión del espacio de color YUV al espacio RGB

---

**Entrada:**  $x, y, z \in \{0, 1\}$

**Salida :**  $l, a, b \in \{0, 1\}$

$$r = y + 1.402 * (v - 128)$$

$$g = y - 0.34414 * (u - 128) - 0.71414 * (v - 128)$$

$$b = y + 1.772 * (u - 128)$$

```
forall c ∈ {r, g, b} do
  if c < 0 then
    | c = 0
  else if c > 255 then
    | c = 255
  end
end
end
```

---

---

**Algoritmo 28:** Conversión del espacio de color RGB al espacio hsL

---

**Entrada:**  $r, g, b$

**Salida :**  $h, s, L$

$suma = r + g + b$

**if**  $suma \neq 0$  **then**

$$\begin{cases} r = \frac{r}{suma} \\ g = \frac{g}{suma} \\ b = \frac{b}{suma} \end{cases}$$

**end**

$maximo = Maximo(r, g, b)$

$minimo = Minimo(r, g, b)$

$med = Mediana(r, g, b, maximo, minimo)$

$L = \frac{1}{3} \cdot (r + g + b)$

**if**  $L \geq med$  **then**

|  $s = \frac{3}{2} \cdot (maximo - L)$

**else**

|  $s = \frac{3}{2} \cdot (L - minimo)$

**end**

**if**  $r \geq g \&\& g \geq b$  **then**

|  $\lambda = 0$

**else if**  $g \geq r \&\& r \geq b$  **then**

|  $\lambda = 1$

**else if**  $g \geq b \&\& b \geq r$  **then**

|  $\lambda = 2$

**else if**  $b \geq g \&\& g \geq r$  **then**

|  $\lambda = 3$

**else if**  $b \geq r \&\& r \geq g$  **then**

|  $\lambda = 4$

**else**

|  $r \geq b \&\& b \geq g$

**end**

$\lambda = 5$

$h = 0$

**if**  $s \neq 0$  **then**

|  $h = \frac{\pi}{3.0} [\lambda + \frac{1}{2} - (-1)^\lambda \frac{maximo+minimo-2med}{2s}]$

**end**

$norma.hs = \sqrt{h^2 + s^2}$

**if**  $norma.hs \neq 0$  **then**

$$\begin{cases} h = \frac{h \cdot s}{norma.hs} \\ s = \frac{s^2}{norma.hs} \end{cases}$$

**end**

---

---

# Lista de Figuras

---

3.1	Efectos de variación en iluminación y ruido aditivo . . . . .	41
3.2	Interpretación geométrica de la prueba de colinealidad de los vectores $x_1, x_2$ . . . . .	44
3.3	Vecindad a evaluación. . . . .	46
3.4	Espacio de color HSL . . . . .	52
3.5	Plano cromático de espacio de color HSL . . . . .	53
4.1	Secuencia 130 de PETS2000 . . . . .	57
4.2	Modelos de fondo con alfa fija en secuencia 130 de PETS2000 . . . . .	57
4.3	Secuencias de PETS2001_DS3_TE1_C1 . . . . .	58
4.4	Función de densidad resultante secuencia WaterSurface pixel (113, 70)	64
4.5	Función de densidad resultante secuencia WaterSurface pixel (102, 56)	65
4.6	Función de densidad resultante secuencia HighWay1 pixel (110, 70) . .	66
4.7	Espacios de color . . . . .	72

---

# Lista de Tablas

---

2.1	Detección de objetos en movimiento de los métodos media, mediana, MoG Stauffer, MoG efectivas, MoG en cascada en secuencias <i>PETS2000</i> . . . . .	36
2.2	Detección de objetos en movimiento de los métodos media, mediana, MoG Stauffer, MoG efectivas, cascada en secuencias <i>PETS2001_DS3_TE1_C1</i>	37
2.3	Detección de objetos en movimiento de los métodos media, mediana, MoG Stauffer, MoG efectivas, cascada en secuencias <i>WaterSurface</i>	38
2.4	Detección de objetos en movimiento de los métodos media, mediana, MoG Stauffer, MoG efectivas, cascada en secuencias <i>HighWay1</i> . . .	39
3.1	Variación en la luminosidad . . . . .	43
3.2	Detección de sombras con Correlación de la Intensidad . . . . .	51
3.3	Detección de sombras con Correlación utilizando el color . . . . .	54
4.1	Modelos de fondo con $\alpha = 0.005$ de <i>PETS2001_DS3_TE1_C1</i> . . . . .	59
4.2	Modelos de fondo con $\alpha = 0.1$ de <i>PETS2001_DS3_TE1_C1</i> . . . . .	59
4.3	Modelos de fondo con $\alpha$ dual sin umbral en $\sigma$ de <i>PETS2001_DS3_TE1_C1</i>	60
4.4	Modelos de fondo con $\alpha$ dual y con umbral en $\sigma$ de <i>PETS2001_DS3_TE1_C1</i>	61
4.5	Número de distribuciones de cada pixel . . . . .	62
4.6	Número de distribuciones de cada pixel . . . . .	63
4.7	Etiquetas en secuencias <i>Fountain</i> , utilizando estadísticos . . . . .	68
4.8	Etiquetas en secuencias <i>WaterSurface</i> , utilizando estadísticos . . . . .	69
4.9	Etiquetas en secuencias <i>Campus</i> , utilizando estadísticos . . . . .	70
4.10	Resultados de detección de movimiento en varios espacios de color .	74
4.11	Resultados de detección de movimiento en varios espacios de color .	75
4.12	Resultados de detección de movimiento en varios espacios de color .	76
4.13	Resultados de detección de movimiento en varios espacios de color .	77
4.14	Resultados de detección de movimiento en varios espacios de color .	78
4.15	Pruebas de ruido en las secuencias <i>Rain</i> . . . . .	80
4.16	Pruebas de ruido en las secuencias <i>Rain</i> . . . . .	81
4.17	Imágenes resultado al cambiar el nivel de confianza. . . . .	83
4.18	Detección de objetos en movimiento de los métodos cascada y cascada final . . . . .	88

4.19 Detección de objetos en movimiento de los métodos cascada y cascada final . . . . . 89



---

# Lista de Algoritmos

---

1	Aproximación del filtro de la media . . . . .	17
2	Aproximación del filtro de la mediana . . . . .	19
3	Modelo de mezclas de Stauffer y Grimson . . . . .	23
4	Aprendizaje_Mezclas() . . . . .	24
5	Modelo de mezclas efectivas para mezcla de gaussianas . . . . .	29
6	Modelo de mezclas adaptativas . . . . .	32
7	Aprendizaje_Mezclas_Dinamicas() . . . . .	33
8	Algoritmo en cascada . . . . .	34
9	Condición para $\sigma$ . . . . .	56
10	Alfa_Dual() . . . . .	60
11	Prueba de concordancia. . . . .	82
12	Algoritmo final para subtraer el fondo usando mezcla de gaussianas. . . . .	85
13	Concordancia(blnAlfa) . . . . .	86
14	Conversión del espacio de color RGB al espacio HSI . . . . .	107
15	Conversión del espacio de color HSI al espacio RGB . . . . .	108
16	Conversión del espacio de color RGB al espacio HSL . . . . .	109
17	Conversión del espacio de color HSL al espacio RGB . . . . .	110
18	Conversión del espacio de color RGB al espacio HSV . . . . .	111
19	Conversión del espacio de color HSV al espacio RGB . . . . .	112
20	Conversión del espacio de color RGB al espacio LAB . . . . .	112
21	Conversión del espacio de color LAB al espacio RGB . . . . .	112
22	Conversión del espacio de color RGB al espacio LAB . . . . .	112
23	Conversión del espacio de color XYZ al espacio LAB . . . . .	113
24	Conversión del espacio de color LAB al espacio XYZ . . . . .	114
25	Conversión del espacio de color XYZ al espacio RGB . . . . .	114
26	Conversión del espacio de color RGB al espacio YUV . . . . .	115
27	Conversión del espacio de color YUV al espacio RGB . . . . .	115
28	Conversión del espacio de color RGB al espacio hsL . . . . .	116