

UNIVERSIDAD AUTÓNOMA DE MADRID
MASTER'S PROGRAM IN RESEARCH AND INNOVATION IN INFORMATION
AND COMMUNICATION TECHNOLOGIES (I2-ICT)
Video Analysis Techniques for Surveillance (Course 2012/2013)

LAB 3: Blob extraction and classification

Augusto Bourgeat Terán
José Luis Carrillo Medina

1 Objetivo

El objetivo de este trabajo es el desarrollo de rutinas de análisis de Blob, utilizando un módulo de segmentación de primer plano que genera una máscara binaria (con valores 0 y 255) y la secuencias de vídeo de entrada capturada por las cámaras de vídeo fijas, y a través de funcionalidades básicas de Blob, obtener la extracción de blob, clasificación de blob y detección de objetos estacionario, ha este tipo de implementación se conoce como detección de objetos.

2 Problema

Características de la Implementación:

- Extracción de blob,
- Clasificación de blob, y
- Detección de objetos estacionarios.

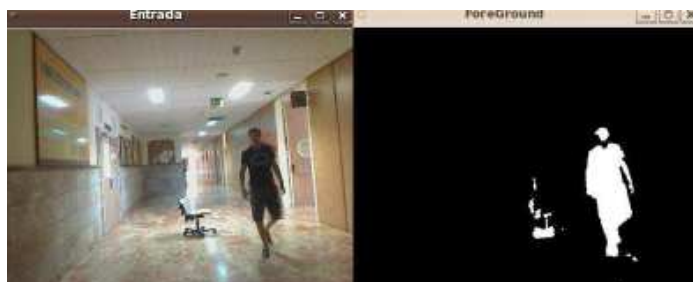
Las rutinas desarrolladas se integrarán en un solo programa que se implementa en C (o C + +) utilizando la biblioteca pública opencv bajo el sistema operativo Linux.

Palabras claves: blob (Binary Large Object), extracción, clasificación y objetos estacionarios.

3 Desarrollo

Un Binary Large OBject, denominado Blob, no es más que un conjunto conectado e impreciso de pixeles de una imagen cuya intensidad es igual a un umbral 255 ("1"), y que están unidos entre sí. A partir de él es posible calcular una serie de características de gran utilidad a la hora de su procesamiento y posterior detección de los objetos de una imagen.

El análisis de blob consiste en una serie de operaciones de procesamiento y funciones de análisis que producen información acerca de cualquier forma 2D dentro de una imagen.



Detección de objetos [1].

Para llevar a cabo la implementación de detección de objetos a través de Blob, se llevaron a cabo las siguientes etapas:

- Segmentación de fondo,
- Extracción de Blob,
- Clasificación de Blob, y
- Detección de objetos estacionarios.

a) Segmentación de frente.

Para la segmentación de Frente se implementaron varias técnicas de sustracción de fondo así como también aplicando funciones propias de opencv.

- Sustractor de fondo:
 - Running average, selectivo.
 - Inicio en Caliente, supresión de fantasmas, objetos estacionarios.
 - Funciones propias de opencv (Modelo de Gaussianas).

Todas las técnicas anteriormente mencionadas tienen el requisito de que deben ser capaces de detectar objetos en movimiento mediante secuencias de video. Además todos los algoritmos deben solventar problemas como: ruido (puede provocar la detección de zonas de foreground incorrectas), sombras y reflejos (en la mayoría de los casos generan interferencias que hacen que el algoritmo no actué de forma adecuada), cambios de iluminación (cambio de luminosidad a lo largo del día o distintas fuentes de iluminación), actualización del fondo de escena (la inicialización del fondo de la secuencia, generalmente no coincide con el background ya que puede haber algún objeto o persona en movimiento, por lo que debe ser capaz de no clasificarlo como fondo así como los cambios importantes en la secuencia), otro punto bien importante es que los objetos deben ser sólidos para así poderlos extraer y clasificarlos.

Los sustractores de fondos implementados con técnicas básicas (Running Average) e Inicio en Caliente, supresión de fantasmas y objetos estacionarios, se tuvo algunos problemas como: imágenes incompletas (personas segmentadas en varias partes), estelas no controladas. En cambio con el sustractor de fondo implementado con funciones de opencv (Gaussianas), se mejoro los problemas (personas mejor segmentadas, sólidas y sin estelas). Razón por la cual se adopto el sustractor realizado con funciones de opencv.



Detección del Frente (zonas donde se produce movimiento)

b) Extracción de Blob.

Para la extracción de Blob, se parte del requerimiento ya analizado anteriormente que es localizar las zonas donde se produce movimiento, detección del frente.

Esta etapa consiste en extraer los Blob que tiene cada imagen de la secuencia de video, segmentados en el frente como "255". Para extraer todos los blob de una imagen se utiliza la técnica de **análisis de componentes conectados**, función implementada en opencv llamada **cvFloodFill** con 8 conexiones, la misma que realiza un barrido pixel a pixel buscando compactar pixeles de un mismo color formando un objeto desconocido, denominado blob, los mismos que se almacenan en la lista de blobs encontrados en la imagen actual, los blob's son la entrada de la clasificación de objetos.

Dentro de los parámetros de la función **cvFloodFill** esta la estructura llamada **CvConnectedComp** la cual contendrá información del blob encontrado al procesar la imagen actual como son: la posición en donde empieza el Blob, X e Y, alto y ancho del blob, el área y otros datos del blob.

El valor encontrado de área no se podía almacenar en la clase **BasicBlob** ya que no existía como dato miembro de la clase, siendo un dato importante para la posterior clasificación de blob, por lo que se creó este dato en la clase **BasicBlob**, así como sus funciones asociadas, set y get.







Cabe notar que se realizó un filtrado de blob, estableciéndose como un objeto en este caso blob a aquellos objetos del frente que tuvieran un ancho y alto mayor a 10 pixeles respectivamente, si esto sucede se considera como blob.

c) Clasificación de Blob.

El algoritmo de clasificación destinado a determinar si un objeto es persona, coche, grupo, maleta o desconocido se denomina detector de objetos. Estos detectores pueden ser de distintos tipos. Pueden ser basados en el análisis de regiones: el aspect ratio, que no es otra cosa que la tasa de aspecto que es igual al ancho/altura y dependiendo de este valor se clasifica a cada uno de los blob de la imagen actual, otra forma es la compactness, que es igual al $\text{área_blob} / (\text{ancho} * \text{alto})$ del blob, otra forma es determinar la elipse más grande contenida en cada región obtenida, otra forma son los basados en las características intrínsecas de la persona (basados en el área o en las relaciones entre las medidas medias).

En el cálculo del **Aspect ratio**, se utilizó un dataset de personas (entregado por el Profesor) y dataset ad-hoc de coches, maletas contruidos a partir de imágenes capturadas, cuyos valores se obtuvieron mediante matlab encontrándose los estadísticos sobre la base de la media y varianza de las características.

Así como también se calculó para cada tipo de objeto su **Compacidad** parámetro que se va a utilizar como segunda característica para discriminar un objeto encontrado.

binaryDB		
data_autos		
data_maleta		

Aspect ratio:

CLASE	ESTADÍSTICOS	RANGO (99% DE CONFIANZA) $\Sigma= 3.0$
Persona	Media=0,3509 Desviación = 0,0639	0.16 a 0.50
Grupo	Media=1.23 Desviación = 0.12	0.87 a 1.60
Objeto (maleta)	Media=0,68 Desviación = 0,061	0.50 a 0.87
Auto	Media=2.6 Desviación = 0,33	1.60 a 3.60
Desconocido		Otro valores diferentes a las anteriores

Compactness.

CLASE	ESTADÍSTICOS	RANGO (99% DE CONFIANZA) $\Sigma= 3.0$
Persona	Media=0,6357 Desviación = 0,0824	0.39 a 0.88
Grupo	Media=0,6357 Desviación = 0,0824	0.39 a 0.88
Objeto (maleta)	Media=0,6945 Desviación = 0,0785	0.46 a 0.93
Auto	Media=0.6045 Desviación = 0,0950	0.32 a 0.89
Desconocido		Otro valores diferentes a las anteriores

Para obtener estos valores se ha programado en matlab:

Tasa_aspecto_Persona2.m

Tasa_aspecto_maleta.m

Tasa_aspecto_autos.m

Compactness_Persona.m
Compactness_maleta.m
Compactness_autos.m

De las tablas anteriores (rangos de clasificación) tanto para aspect ratio como para compactness se puede indicar que:

Aspect Ratio: Es un atributo que si permite una clasificación de acuerdo con los rangos de valores para cada clase.

Compactness: Este atributo tiene valores promedios aproximados para todas las clases que deseamos clasificar, por tanto no es una buena propiedad para separar o para poder identificar los blobs de diferentes clases.

Otros valores de atributos realizados fueron la extracción de promedios de Histograma de pixeles Horizontal - y pixeles Vertical de personas, Histograma de pixeles Horizontal - y pixeles Vertical de autos, Histograma de pixeles Horizontal - y pixeles Vertical de maletas. Con ellos calculamos un tasa de medias de histograma: $\text{RateHist} = \text{Hist_Vert_media} / \text{Hist_Horiz_media}$

Hist_HorVer_Persona.m
Hist_HorVer_autos.m
Hist_HorVer_maleta.m

Se ha probado utilizando por separado los dos parámetros y usando también una fusión de los mismos, quedándose solo con la tasa de aspecto.

Por lo que para cada tipo de objeto, se cuenta con dos parámetros a analizar en la clasificación de objetos: **aspect ratio**, y **compactness**, una vez determinados los objetos se etiquetan como persona, grupo, maleta, carros y desconocido, obteniéndose los siguientes resultados:

Resultado 1: Aspect ratio



Clasificación de personas

Interpretación:

La ejecución del programa presenta tres en las ventanas, el procesamiento de los videos a través de sus frame : 1) Imagen original a color, 2) Imagen de Frente, 3) Etiquetado de Objetos.

Analizando los resultados obtenidos se observa:

- Que etiqueta bien a una persona adulta y a una niña.

Resultado 2: Aspect ratio:



Clasificación de personas

Interpretación:

Analizando los resultados obtenidos se observa:

- Que etiqueta bien a la persona que está en el medio de la escena, en cambio a la persona que se está asomando la cabeza la etiqueta como grupo.

Resultado 3: Aspect ratio:



Clasificación de personas

Interpretación:

Analizando los resultados obtenidos se observa:

- Que etiqueta las persona que está cruzando el pasillo como grupo, y a la persona que esta ingresando lo detecta como persona, de igual manera, las personas que caminan en el fondo y están segmentadas en pedazos (solo detecta ciertas partes del cuerpo), no se etiquetan.

Clasificador y etiquetado por redes Neuronales

Formamos una base de datos con los siguientes atributos ***aspect ratio***, ***compactness***, ***RateHist*** y **etiqueta (0 = persona 1=auto 2=maleta/objeto)**. Tabla en la que hay 400 casos para personas, 20 casos para autos y 20 casos para maletas. Con esta base de entrenamiento queremos predecir una de las 3 clases **etiqueta (0 = persona 1=auto 2=maleta)**.

Utilizando un programa de red neuronal tipo back propagation que tiene 3 capas y 440 datos de entrenamiento, 2 atributos (***aspect ratio***, ***compactness***) y 3 clases o etiqueta (0 = persona 1=auto 2=maleta), se obtiene el modelo para clasificar, datos que ingresaran en cada frame y serán extraídos como blob, para calcular sus atributos y poder ingresar a la red neuronal el cual proporcionará la clase, para posteriormente colocar la etiqueta respectiva.

Los programas por separado funcionan, la red entrena y clasifica, al integrar los programas (extracción/clasificación Red Neuronal), funciona, con la limitante que los datos deben ser decimales con separador de punto (ejemplo 0.5), para hacer esto se puede configurar por el sistema.

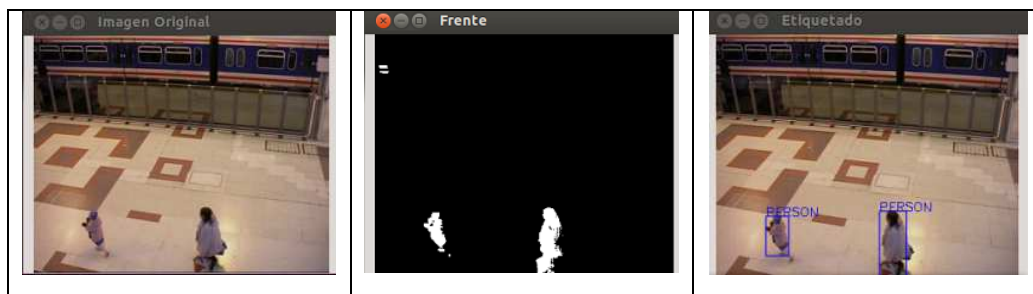


Fig. Utilizando clasificador por Redes Neuronales



Fig. Utilizando clasificador por Redes Neuronales

Interpretación:

Analizando los resultados obtenidos se observa:

- Que etiqueta bien a la persona que está en el medio de la escena.

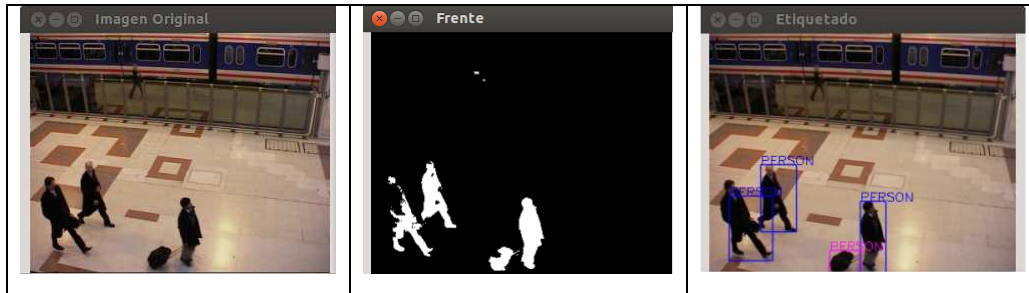


Fig. Utilizando clasificador por Redes Neuronales

Como se puede observar la primera imagen representa la imagen original, la segunda el frente o mascara y la tercera el clasificador por redes neuronales y etiquetado de blob.

Encuentra las etiquetas mejor que el algoritmo de clasificación por bloques (tasa de aspecto) pudiendo mejorarse utilizando un gran número de imágenes de entrenamiento para cada clases, además el costo computacional es más elevado.

Se probó el funcionamiento con dos atributos aspect ratio, compactness, pudiendo extenderse para más atributos como por ejemplo (RateHist, y otros), implementación que se trató de realizar.

Se generaron dos archivos `redn.cpp` y `redn.h` el cual contiene las siguientes funciones:

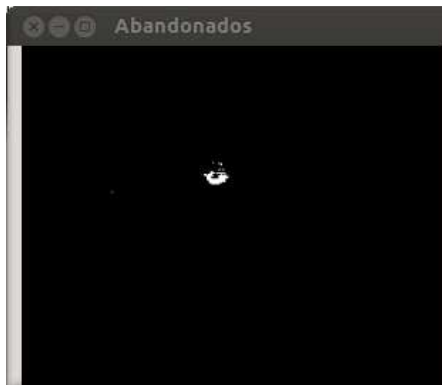
- función para leer datos desde un archivo plano. `read_data_from_txt()`
- Función para entrenar la red neuronal. `Training_network`
- Función para predecir tipos de clases: `test_network`

Estas funciones fueron implementados mediante la clase `red`.

d) Detección de objetos estacionarios.

El algoritmo de detección de objetos estacionarios permite identificar situaciones anómalas en el video, esto se lo implementó mediante la detección de objetos abandonados. El algoritmo consiste en verificar que un objeto que se está moviendo, que no es parte del fondo, se detiene en un periodo de tiempo se lo detecta como objeto del frente y se reporta como objeto abandonado. Para este caso se tomó un período de tiempo de 10 seg., de acuerdo a lo especificado en el código entregado.

Resultado 1:



Detección objeto estacionario



Etiquetado

Interpretación:

Analizando los resultados obtenidos se observa:

- Que se etiqueta bien a la persona que sale de la escena caminando en el pasillo
- El objeto estacionario es detectado como abandonado.
- Resultado 2:



Imagen



Mascara



Detección objeto estacionario



Etiquetado

Interpretación:

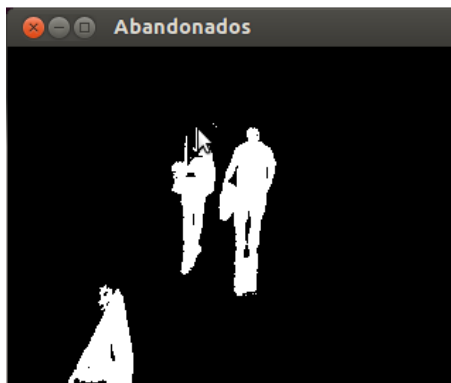
Analizando los resultados obtenidos se observa:

- Que se etiqueta bien a las personas que cruzan en la escena caminando en el pasillo
- El objeto estacionario se mantienen detectado como abandonado.

4 Limitaciones

De acuerdo a lo observado en este programa se puede indicar que tiene las siguientes limitaciones:

1. - El programa no detecta bien los Blob cuando se utiliza videos con inicio en caliente, las imágenes iniciales se detectan como objetos abandonados.



- 2.- El programa clasifica objetos completos. Cabezas, piernas, brazos, los etiquetas como grupo de personas cuando esta no aparece la imagen completa por las occlusiones o objetos estacionarios.

- 3.- El programa presenta problemas de camuflaje, cuando personas o objetos se detienen o tienen vestuario similar al fondo de la escena.



4.- Cuando el objeto es robado se detecta con la etiqueta abandono.



Conclusiones

Los resultados obtenidos muestran la eficacia de etiquetados, clasificador y detector de objetos estacionarios, su rendimiento depende principalmente de las mejoras y robustez del sistema de detección del frente.

Se ha conseguido detectar objetos abandonados a partir de secuencias de vídeo-seguridad. Este objetivo involucra la detección de regiones estáticas en la imagen que no pertenezcan al fondo.

La implementación de la red neuronal funciona, teniendo sus limitantes por el número de datos de entrenamiento.

Se han implementado varios algoritmos para la clasificación utilizando clasificadores de bloque fusionando atributos, y redes neuronales, dando resultados aceptables, con sus limitaciones.

Se puede usar este análisis para hallar información estadística, tal como el tamaño de los blobs o el número, ubicación y presencia de regiones de blobs. Con esta información, se pueden realizar numerosas tareas de video vigilancia, inspección de visión de maquinaria. También se pueden ubicar objetos en aplicaciones de control de movimientos donde existe una variación significativa de la forma u orientación de las piezas.

En esta práctica se han desarrollado diversas técnicas e implementado algoritmos de extracción de Blob utilizando OpenCv.

References

- [1] Apuntes de Clase, Video Analysis Techniques for Surveillance Unit 2: Video Analysis for Surveillance (Part II y III), Juan Carlos San Miguel Avedillo, juancarlos.sanmiguel@uam.es
- [2] Procesamiento de IMAGEN DIGITAL - Machine Vision
- [3] Grass-Fire Algorithm
- [4] Sadiye Guler Member IEEE and Matthew K. Farrow Abandoned Object Detection in Crowded Places

- [5] Álvaro Bayona, Juan C. SanMiguel, José M. Martínez STATIONARY FOREGROUND DETECTION USING BACKGROUND SUBTRACTION ANDTEMPORAL DIFFERENCE IN VIDEO SURVEILLANCE*

Realizado por:

Augusto Bourgeat

José Luis Carrillo

Program: Master's program in Research and Innovation in Information and Communication Technologies (I2-CIT)

Center: Escuela Politécnica Superior

University: Universidad Autónoma de Madrid
Madrid 2013