Congr. Int. en Ing. Electrónica. Mem. ELECTRO, Vol. 39, pp. 52-54, Oct 2017, Chihuahua, Chih. , México
http://electro.itchihuahua.edu.mx/memorias_electro/MemoriaElectro2017.zip
ISSN 1405-2172

ADAPTACIÓN DEL CÓDIGO MATLAB DEL FILTRO DE KALMAN APLICADO AL PROCESAMIENTO DE IMÁGENES

Ortiz Anchondo, Luis Ricardo[1], Espejel García, Daphne[1],
Espejel García, Vanessa Veronica[1], Villalobos Aragón, Alejandro[1]
[1]Universidad Autónoma de Chihuahua
Facultad de Ingeniería
Circuito No 1, Campus Universitario II Chihuahua, Chih. C.P. 31125, México
(+52) (614) 442-95-00
a291540@uach.mx, despejel@uach.mx

RESUMEN.

El procesamiento de imágenes es un conjunto de técnicas, en constante evolución, enfocadas al mejoramiento de las mismas y obtención de información. Uno de los problemas que se deben eliminar, es la presencia de ruido, el cual no permite obtener resultados confiables durante el procesamiento. Uno de los métodos para la reducción de este problema es el filtro de Kalman, el cual permite separar las señales aleatorias del ruido aleatorio. Es complicado crear un filtro Kalman, por lo que se propone adaptar el código que proporciona Matlab para construir, en este caso, un programa de aforo vehicular que permita contar los vehículos que transitan por un carril en una vialidad. Esta herramienta permitirá al usuario emplear un método de aforo de bajo costo, utilizando solo una cámara de video y un procesador de alto rendimiento.

***Palabras Clave: Kalman, procesamiento de imágenes, Matlab*.**

ABSTRACT.

Image processing is considered as a set of techniques in constant evolution, which allow to improve the images and to obtain information from them. One persistent problem during the process, is the presence of noise that commonly does not allow obtain reliable results. The Kalman filter minimizes this problem, by separating the random signals from the random noise. It is complicated to create a Kalman filter, it being the main reason to adapt the one in Matlab. In this case, for a vehicle count procedure in a street lane. This tool will help the user, in order to apply a low-cost vehicle counting technique, using only a video camera and a high-performance computer processor.

***Keywords: Kalman, image processing, Matlab*.**

1. INTRODUCCIÓN

En la actualidad, el procesamiento de sistemas de imágenes se considera una herramienta muy útil, ya sea para mejorar la calidad de las mismas, o para buscar información a partir de ellas. El análisis de imágenes en tiempo real involucra varios procesos, entre ellos el estudio del flujo de iconografías captadas por una cámara, y el control de un sistema basado en los resultados [1]. Uno de los problemas a considerar, es trabajar con secuencias de imágenes que se degradan debido a la presencia de ruido, el cual suele aparecer desde la adquisición de las imágenes [2].

El control del sistema está basado en el correcto procesamiento de las imágenes, por lo que es importante que las mismas estén libres de ruido y sean procesadas a tiempo [3]. Esto se puede lograr aplicando el filtro de Kalman, ya que el algoritmo logra separar las señales aleatorias del ruido aleatorio [4].

Aún y cuando ya existen de manera comercial software que procesan imágenes, el costo de los mismos es muy elevado, ya que desde la perspectiva de un ingeniero en software no es sencillo crear filtros Kalman [3]. Por lo anterior, se propone una adaptación del código Matlab del filtro Kalman, con el fin de procesar imágenes y realizar un aforo vehicular (conteo de vehículos que pasan por un punto en particular [5]). Esto es posible mediante el uso de herramientas comunes, como una cámara de video y un procesador de alto rendimiento (*e.g. core* i7).

2. FILTRO DE KALMAN

En 1960, Rudolf Kalman creó un filtro que soluciona problemas en comunicación y control estadístico, al separar señales aleatorias del ruido, obteniendo como resultado un método que identifica el estado no oculto de un sistema dinámico lineal, con o sin ruido [4]. Este filtro es ampliamente utilizado en los sistemas de navegación y procesamiento de imágenes [6]. El filtro de Kalman se implementó en el procesamiento de imágenes en la década de 1970 [7], y desde entonces se ha utilizado en varias aplicaciones, tales como restauración de imágenes [7], rastreo de objetos (como vehículos o peatones) [8], estimación de objetos en movimiento [9] o implementación del filtro por pixel en un FPGA (*Field Programable Gate Array*, por sus siglas en inglés) [10].

Referente a los aforos vehiculares, en 1994 se creó un sistema rastreador de contornos, donde el filtro de Kalman detecta las trayectorias vehiculares en una secuencia de imágenes [11]. Posteriormente se aplicó el filtro para rastrear vehículos [12] o para contarlos [13].

Actualmente, se puede adaptar el código para Matlab del filtro de Kalman a las necesidades del usuario, esto con respecto al procesamiento de imágenes y conteo de vehículos. Se eligió

Matlab por ser un software de fácil acceso para ingenieros, y se decidió modificarlo, para crear una herramienta accesible y de bajo costo. Cabe mencionar que el código debe ser modificado, ya que tal y como aparece en Matlab, no realiza ningún conteo vehicular.

3. MODIFICACIONES AL CÓDIGO DEL FILTRO DE KALMAN.

3.1. Delimitación del área de aforo (carril).

El uso de la función Built-in de MatLab "ginput()", misma que devuelve la posición (x,y) del cursor en el momento en el que este hace click (Fig. 1), permite al usuario indicar el ancho del carril que delimita el área que se va a tomar en cuenta para el conteo de autos.

```
%%
%desired es la cantidad de carriles a escanear
desired = inputdlg('Por favor, ingrese la cantidad de carriles que desea escanear')
%imshow es para mostrar la imagen, i es el indice (posicion) de la imagen,
%por lo que 1 es la primer imágen(frame) del video.
imshow(mov(1).cdata);
%str2num sirve para convertir 'uno' en 1, basicamente
desired = str2num(desired{1});
%parsear desired a numero o a Integer
coord=round(ginput (desired*2));
close();
%%

    if(desired == 1)
ub = {[coord(1,2):coord(2,2)];
     [coord(1,1):coord(2,1)]};
    end
```

Figura 1. Bloque de código que muestra cómo se facilita al usuario el delimitar el área examinada.

3.2. Eliminación de sombras.

Un problema que debe ser suprimido, es la presencia de sombras, las cuales provocan error en el conteo, ya que el programa las detecta como si fueran otro vehículo. Tomando en cuenta que los valores RGB de las sombras tienden a ser menores (entre más cerca de [0,0,0], más oscura será la imagen), se delimitó un rango dependiendo de la hora del día en que se tomó el video, y usando un ciclo, se convierten las sombras que entran dentro de ese rango a un color parecido al del asfalto, salteando la tolerancia del bloque de código que añadía autos al contador. Como es preciso recorrer todo el "frame", es necesario saber su tamaño, así que se asignó el tamaño de "X" a una variable "nz" para hacer un ciclo. Nótese que "X" corresponde al "frame" actual (Fig. 2).

```
%Nótese que la primer dimensión será el eje x, la segunda Y, y la tercera
%z, en otras palabras, X[x, y, z] = un valor
X = mov(i).cdata;
nz=size (X);
if strcmp(eliminarSombra, 'Si') == 1;
for X5=1:nz(1);
       for X6 =1:nz(2);
                if  X(X5,X6,1) > 35 &&  X(X5,X6,1) <83; %43 y 83
                    X(X5,X6,1) = 105;
                end
                if strcmp(blancoNegro, 'RGB') == 1
                     if  X(X5,X6,2) < 25;
                        X(X5,X6,2) = 135;
                     end
                     if  X(X5,X6,3) < 25;
                        X(X5,X6,3) = 135;
                     end
                end
       end
end
```

Figura 2. Bloque de código que muestra cómo se realiza la transformación de las sombras a un color similar al del asfalto.

3.3. Conteo de vehículos.

Con el área delimitada y las sombras eliminadas, se procede a la estimación de Kalman. Nuevamente, se debe establecer un rango de tolerancia para el resultado, de restar la imagen actual y el fondo en el que, con la ayuda de una variable bandera, el programa decide si lo que hay actualmente es un auto, si ya lo contó o si no es ninguno de los dos (Fig. 3).

```
 if predicted{j}(1)>25 && predicted{j}(2)>25
     if cont{j}==0
         FLAG{j}=0;
         disp('Permiso otorgado')
     end
     cont{j}=1;
     disp('----------------Hay un vehiculo en detección----------------')
     % FLAG da permiso de aumentar el contador FLAG{j}=0;
else
    %Como ya no hay autos a la vista, significa que el próximo auto que
    %detecte no ha sido contado, asi que se le dá permiso para dar
    %permiso para contarlo
    cont{j}=0;
end
```

Figura 3. Bloque de código que muestra cómo el programa decide si lo que detecta es un vehículo, y si fue contado o no.

4. DESARROLLO.

Se utilizó el código del filtro de Kalman que proporciona Matlab, y se ejecutó el algoritmo en una computadora con Windows 10, y un procesador *core i7*. Las imágenes se obtuvieron con una cámara de video HD handycam de 9.2 megapixeles, instalada en un tripié sobre los puentes peatonales de una vialidad. Los videos se convirtieron en archivos .AVI y .MP4, y se transformaron a imágenes en blanco y negro para facilitar su ejecución en Matlab. Se eligió un punto y una hora de bajo flujo vehicular, para verificar al mismo tiempo el conteo de vehículos que realiza el algoritmo, y así validar el programa.

5. RESULTADOS.

La adaptación el código para Matlab del filtro de Kalman permitió:

- Identificar en una imagen 2D, el área que cubre el frente de un vehículo.
- Delimitar el espacio (carril) que se desea aforar.
- Eliminar las sombras de los vehículos, que podían ser contadas como otro objeto.
- Realizar el conteo de cada vehículo que circula por un carril.
- Elaborar un cuadro de presentación donde se muestra el conteo vehicular (Fig. 4).

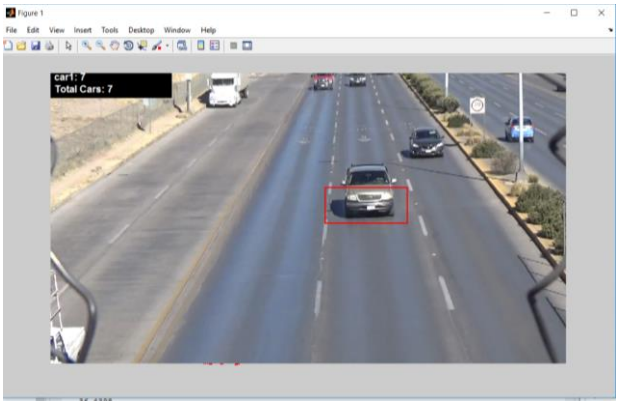

Figura 4. Imagen del conteo vehicular en Matlab. El recuadro negro en la parte superior izquierda muestra el total de vehículos contados.

6. CONCLUSIONES Y RECOMENDACIONES.

El procesamiento de imágenes es una herramienta eficaz para obtener información a partir de las mismas, especialmente cuando los datos requeridos son difíciles de obtener manualmente. El filtro de Kalman es útil para eliminar los errores que provoca el ruido de las imágenes. Matlab es un software comúnmente utilizado, que puede programar y procesar archivos. Esta combinación permite realizar programas de bajo costo que solucionan un problema, como lo es en este caso, un aforo vehicular.

Debido al tamaño de los archivos de las imágenes, por el momento solo se recomienda realizar aforos un carril a la vez, para evitar el colapso del programa. El trabajo a futuro incluye adaptar el código, e incorporarlo a un procesador de mayor rendimiento para obtener resultados en tiempo real.

7. REFERENCIAS.

[1] P.A. Laplante y C.A. Neill. A class of Kalman filters for real-time image processing, Proceedings of the real-time imaging conference, Santa Clara: SPIE, Enero 2003, 22-29.

[2] F. Schneider, S.M. Easterbrook, J.R. Callahan y G.J. Hotzmann. Validating requirements for fault tolerant systems using model checking, Third IEEE conference on requirements engineering, Colorado Springs, Abril 1998.

[3] M. Piovoso y P.A. Laplante. Kalman filter récipes for real-time image processing, Real-time Imaging, 9, 2003, 433-439.

[4] R.E. Kalman. A new approach to linear filtering and prediction problems, Journal of basic Engineering, 82(1), 35-45.

[5] N.J. Garber y L.A. Hoel. Ingeniería de Tránsito, New York, Thomson ed., 2005, 1170 p.

[6] M.S. Grewal. Kalman filtering, Spring Berling Heidelberg, 705-708.

[7] J. Biedmond, J. Riesek y J.J. Gerbrands. A fast Kalman filter for images degraded by both blur and noise, IEEE Transactions on Acoustics, Speech, and Signal Processing, 35(1), 1983, 1248-1256.

[8] Y. Xia, X. Shi, G. Song, Q. Geng y Y. Liu. Towards improving quality of video-based vehicle counting method for traffic flow estimation, Signal Processing, 120, 642-681.

[9] V. Lipiello, B. Siciliano y L.A. Villani. A new method of image features pre-selection for real-time pose estimation based on Kalman filter, Proceedings of the IEEE/RSJ international conference on intelligent robots and systems, EPFL, Lausanne, Suiza, Octubre 2002, 372-377.

[10] R.D. Turney, A.M. Resa y J.G.R. Delva. FPGA implementation of adaptive temporal Kalman filter for real time video filtering, Proceedings of the IEEE International Conference Transactions on Acoustics, Speech, and Signal Processing, 4, 1999, 2231-2234.

[11] D. Koller, J. Weber, T. Huang, J. Malik, G. Ogasawara, B. Rao y S. Russell. Towards robust automatic traffic scene analysis in real-time, Proceedings of the 12th IAPR International Conference, Computer Vision and Image Processing, 1, 1994, 126-131.

[12] Y.K. Jung y Y.S. Ho. Traffic parameter extraction using video-based vehicle tracking, Proceedings of IEEE/IEEJ/JSAI International Conference in Intelligent Transportation Systems, 1, 1999, 764-769.

[13] R. Bas, A.M. Tekalp y F.S. Salman. Automatic vehicle counting from video for traffic flow analysis, Proceedings IEEE International Conference, Intelligent Vehicle Symposium, 2007, 392-397.

8. AGRADECIMIENTOS

Un enorme agradecimiento a mi tutora, la profesora Daphne Espejel, a la Dra. Vanessa Espejel y al Dr. Alejandro Villalobos, cuya pericia y experiencia aportaron una gran e indispensable asistencia y colaboración en cada trance de asesoría que acontecía a lo largo de la elaboración del presente artículo.