UNIVERSIDAD DEL CEMA
Buenos Aires
Argentina

Serie
DOCUMENTOS DE TRABAJO

Área: Ingeniería Informática

TECNOLOGÍA INTEL SSE APLICADA AL CÁLCULO DE FLUJO ÓPTICO

Javier Luiso

Julio 2018
Nro. 639

www.cema.edu.ar/publicaciones/doc_trabajo.html
UCEMA: Av. Córdoba 374, C1054AAP Buenos Aires, Argentina
ISSN 1668-4575 (impreso), ISSN 1668-4583 (en línea)
Editor: Jorge M. Streb; asistente editorial: Valeria Dowding jae@cema.edu.ar

TECNOLOGÍA INTEL SSE APLICADA AL CÁLCULO DE FLUJO ÓPTICO

Ing. Javier Luiso *

Universidad del CEMA

Buenos Aires, Argentina

* Los puntos de vista del autor no necesariamente representan la posición de la Universidad del Cema

RESUMEN

El presente trabajo expone la implementación de una algoritmo para el cálculo del flujo óptico a partir de procesar una secuencia de frames consecutivos (video) obtenido de un sensor de imagen. El foco de este trabajo esta puesto en aprovechar las capacidades de procesamiento SIMD disponible en la mayoría de los procesadores Intel. Nos referimos a la tecnología denominada SSEx. El tipo de algoritmo a implementar y en especial el tipo de procesamiento que se realiza sobre la secuencia de frames se adapta perfectamente al tipo de procesamiento paralelo que es posible realizar en dicha tecnología. Esto resulta de interés en especial en casos donde se este trabajando en sistemas embebidos donde las capacidades de computo de los procesadores en general suelen ser mas limitadas que las que se dispone en una computadora estándar de escritorio o portátil. Además hay que considerar que en una aplicación real el procesador no puede estar dedicado full-time a resolver el calculo del flujo óptico, sino que debe seguramente ser compartido por otros procesos.

INTRODUCCIÓN

Flujo Óptico

La imagen vista o capturada por un sensor de imagen, es el resultado de la proyección de una porción del espacio sobre un plano, que es el propio sensor de imagen. Cualquier movimiento que describa un objeto dentro de la zona de visión del sensor tendrá su correlato como un movimiento en dos dimensiones de una región de la imagen. Dicha región es la proyección del objeto sobre el plano del sensor.

Si es posible detectar el desplazamiento que una región presente entre frames consecutivos de una secuencia de imágenes, entonces se podrá definir una magnitud que represente la velocidad del movimiento que dicha región está teniendo. Esta velocidad caracteriza entonces al movimiento que se detecta en el plano del sensor. En caso que se detecte un movimiento también es posible determinar cuál es su dirección.

Este campo de velocidades cuyo dominio es toda la imagen, es lo que se denomina "flujo óptico" de la imagen o también "campo de velocidades" de la imagen. [1], [2]

El flujo óptico constituye entonces un *estimador del movimiento 2D* presente en la imagen.

El presente trabajo expone la implementación de un algoritmo que calcula el flujo óptico a partir de una secuencia de imágenes. Dicha implementación se realizará de manera de aprovechar la capacidad de procesamiento paralelo disponible en los procesadores desarrollados por Intel; más específicamente la tecnología conocida como SSEx. Otros fabricantes implementan similares capacidades en sus procesadores y por lo tanto el mismo algoritmo puede implementarse en otros procesadores.

ALGORITMOS PARA EL CÁLCULO DEL FLUJO ÓPTICO

Después de más de 30 años, la mayoría de los métodos para la estimación del flujo óptico que han surgido siguen fuertemente basados en lo formulado por Horn y Shunck. Estos métodos se pueden agrupar de acuerdo a la técnica que utilizan para la estimación, Barron et al [3]. Para este trabajo se ha seleccionado una técnica de cálculo diferencial.

Técnica diferencial

Se basa en la estimación de la velocidad de la imagen a partir de las derivadas espacio-temporales de la intensidad de una imagen o de una versión previamente filtrada de la misma, generalmente utilizando un filtro pasa-bajo o pasa-banda.

Existen soluciones que utilizan derivadas de primer orden, generalmente basadas en una traslación de la imagen que cumpla con:

$$I(\mathbf{x}, t) = I(\mathbf{x} - \mathbf{v}t, 0) \tag{1}$$

donde $\mathbf{x} = (x, y)$ y $\mathbf{v} = (u, v)$.

Se asume que la intensidad se mantiene constante y esto permite expresar la siguiente condición:

$$\nabla I(\mathbf{x}, t)\mathbf{v} + I_t(\mathbf{x}, t) = 0 \tag{2}$$

donde $I_t(\mathbf{x}, t)$ es la derivada temporal de la intensidad.

Se requiere una restricción adicional sobre la Intensidad, dado $\mathbf{v} = (u, v) \in \Re^2$. Los distin-

tos métodos basados en esta técnica en general se diferencian por la restricción adicional que adoptan.

Por ejemplo hay trabajos que usan derivadas de segundo orden, imponiendo la condición:

$$\frac{d\nabla I(\mathbf{x},t)}{dt} = 0 \tag{3}$$

esto lleva a la expresión

$$\begin{bmatrix} I_{xx}(\mathbf{x},t) & I_{xy}(\mathbf{x},t) \\ I_{yx}(\mathbf{x},t) & I_{yy}(\mathbf{x},t) \end{bmatrix} \begin{pmatrix} u \\ v \end{pmatrix} + \begin{pmatrix} I_{tx}(\mathbf{x},t) \\ I_{ty}(\mathbf{x},t) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \tag{4}$$

Encontramos métodos que combinan las restricciones 1 y 4.

Los trabajos de Horn y Shunck [2], Lucas y Kanade [4], Simoncelli et al [5] y Nagel [6] están basados en esta técnica.

IMPLEMENTACIÓN DE LA TÉCNICA DIFERENCIAL

La técnica elegida para implementar es la abordada por Horn y Schunck, [2]. Su enfoque enuncia que el Flujo Óptico no puede ser calculado localmente, dado que en una punto (x, y) de la imagen solamente se dispone de un dato independiente, mientras que el flujo es una magnitud de dos componentes. Entonces se requiere una segunda restricción y ellos proponen asumir que la velocidad aparente del patrón del brillo varia suavemente prácticamente en toda la imagen.

Condiciones de restricción

El enfoque seleccionado se basa en las restricciones que fueron planteadas por Horn y Shucnk:

$$\nabla I(\mathbf{x}, t)\mathbf{v} + I_t(\mathbf{x}, t) = 0 \tag{5}$$

$$\nabla^2 u + \nabla^2 v = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} = 0 \tag{6}$$

Estimación de las derivadas parciales

Para aplicar las restricciones es necesario poder estimar, a partir de la información obtenida en las imágenes, las derivas parciales del brillo $\frac{\partial I}{\partial x}$, $\frac{\partial I}{\partial y}$ y $\frac{\partial I}{\partial t}$, como también el valor de los laplacianos.

Para la estimación de las derivadas parciales se utiliza un kernel de 2x2x2 pixels. Es decir

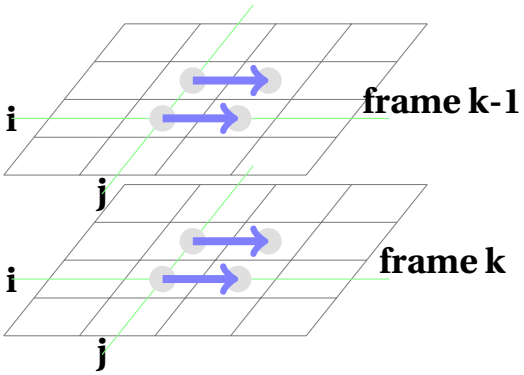

Figure 1: Estimación de la derivada parcial en x

que estamos considerando las dimensiones (i, j, k) donde i, j representan la posición del pixel en el frame y el k representa la coordenada discreta del tiempo, donde se ordenan los sucesivos frames.

La estimación de la derivada parcial en x se realiza conforme lo que indica la ecuación 20 y la figra 1.

$$I_x = \frac{1}{4}\{(I_{i,j+1,k} - I_{i,j,k}) + (I_{i+1,j+1,k} - I_{i+1,j,k}) +$$

$$(I_{i,j+1,k-1} - I_{i,j,k-1}) + (I_{i+1,j+1,k-1} - I_{i+1,j,k-1})\} \tag{7}$$

La estimación de la derivada parcial en y se realiza conforme lo que indica la ecuación 8 y la figura 2.

$$I_y = \frac{1}{4}\{(I_{i+1,j,k} - I_{i,j,k}) + (I_{i+1,j+1,k} - I_{i,j+1,k}) +$$

$$(I_{i+1,j,k-1} - I_{i,j,k-1}) + (I_{i+1,j+1,k-1} - I_{i,j,k-1})\} \tag{8}$$

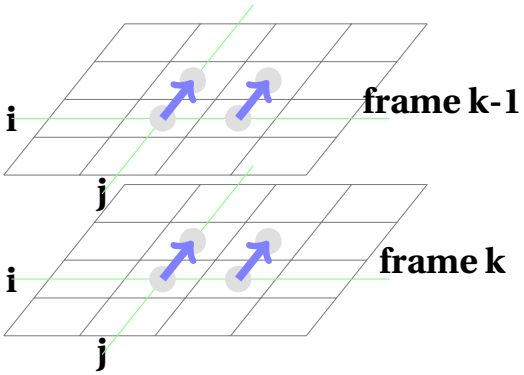

Figure 2: Estimación de la derivada parcial en y

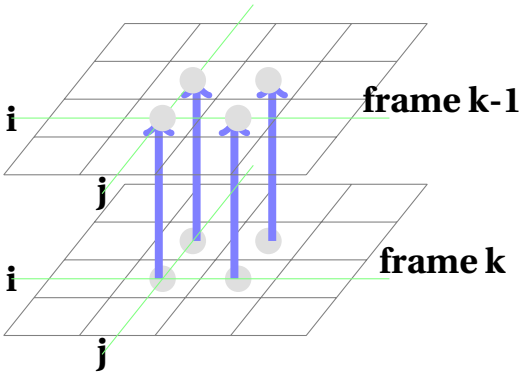

Figure 3: Estimación de la derivada parcial en t

La estimación de la derivada parcial en y se realiza conforme lo que indica la ecuación 8 y la figura 2.

$$I_y = \frac{1}{4}\{(I_{i+1,j,k} - I_{i,j,k}) + (I_{i+1,j+1,k} - I_{i,j+1,k}) + (I_{i+1,j,k-1} - I_{i,j,k-1}) + (I_{i+1,j+1,k-1} - I_{i,j,k-1})\} \tag{9}$$

$$I_t = \frac{1}{4}\{(I_{i,j,k-1} - I_{i,j,k}) + (I_{i+1,j,k-1} - I_{i+1,j,k}) + (I_{i,j+1,k-1} - I_{i,j+1,k}) + (I_{i+1,j+1,k-1} - I_{i+1,j+1,k})\} \tag{10}$$

$\frac{1}{12}$	$\frac{1}{6}$	$\frac{1}{12}$
$\frac{1}{6}$	-1	$\frac{1}{6}$
$\frac{1}{12}$	$\frac{1}{6}$	$\frac{1}{12}$

Figure 4: El laplaciano es estimado sustrayendo el valor en un punto determinado (i, j) de un promedio ponderado de los valores vecinos

Estimación de los laplacianos

Para la estimación del laplaciano se utiliza un kernel de 3x3 pixels, donde los pesos relativos del kernel se muestran en la figura 4.

La expresión de la estimación es:

$$\nabla^2 u \approx K(\overline{u}_{i,j,k} - u_{i,j,k}) \tag{11}$$

$$\nabla^2 v \approx K(\overline{v}_{i,j,k} - v_{i,j,k}) \tag{12}$$

donde $\overline{u}$ y $\overline{v}$ se definen como (ver figura 4):

$$\overline{u}_{i,j,k} = \frac{1}{6}\{u_{i-1,j,k} + u_{i,j+1,k} + u_{i+1,j,k} + u_{i,j-1,k}\} +$$

$$\frac{1}{12}\{u_{i-1,j-1,k} + u_{i-1,j+1,k} + u_{i+1,j+1,k+u_{i+1,j-1,k}}\} \tag{13}$$

$$\overline{v}_{i,j,k} = \frac{1}{6}\left(v_{i-1,j,k} + v_{i,j+1,k} + v_{i+1,j,k} + v_{i,j-1,k}\right) +$$

$$\frac{1}{12}\left(v_{i-1,j-1,k} + v_{i-1,j+1,k} + v_{i+1,j+1,k+v_{i+1,j-1,k}}\right) \tag{14}$$

Función de minimización

La función de minimización adoptada es la que plantean en su trabajo Horn y Schunck y que combina ambas restricciones:

$$\int_D \left\{ (\nabla I(\mathbf{x}, t)\mathbf{v} + I_t)^2 + \alpha^2 \left(||\nabla u||_2^2 + ||\nabla v||_2^2 \right) \right\} dx \tag{15}$$

donde λ representa la influencia o factor de peso del término asociado a la restricción de la suavidad de la Intensidad.

El proceso de minimización implica calcular un valor de (u, v) que haga mínimo el error. A partir de la ecuación 15 obtenemos:

$$I_x^2 u + I_x I_y v = \alpha^2 \nabla^2 u - I_x I_t$$

$$I_x I_y u + I_y^2 v = \alpha^2 \nabla^2 v - I_y I_t$$

Reemplazando los laplacianos por las aproximaciones detalladas anteriormente,

$$\left(\alpha^2 + I_x^2\right) u + I_x I_y v = \left(\alpha^2 \overline{u} - I_x I_t\right)$$

$$I_x I_y u + \left(\alpha^2 + I_y^2\right) v = \left(\alpha^2 \overline{v} - I_y I_t\right)$$

El sistema se puede resolver para las incógnitas u y v,

$$\left(\alpha^2 + I_x^2 + I_y^2\right) u = + \left(\alpha^2 + I_y^2\right) \overline{u} - I_x I_y \overline{v} - I_x I_t \tag{16}$$

$$\left(\alpha^2 + I_x^2 + I_y^2\right) v = - I_x I_y \overline{u} + \left(\alpha^2 + I_x^2\right) \overline{v} - I_y I_t \tag{17}$$

Solución iterativa

La solución iterativa que se plantea para calcular los valores de (u, v) es la siguiente:

$$u^{n+1} = \overline{u}^n - I_x \left[I_x \overline{u}^n + I_y \overline{v}^n + I_t\right] / \left(\alpha^2 + I_x^2 + I_y^2\right) \tag{18}$$

$$v^{n+1} = \overline{v}^n - I_y \left[I_x \overline{u}^n + I_y \overline{v}^n + I_t\right] / \left(\alpha^2 + I_x^2 + I_y^2\right) \tag{19}$$

Condiciones Iniciales y de contorno

Se requieren dos imágenes para poder calcular un primer par (u, v) dado que la estimación de las derivadas parciales necesita disponer del frame actual y del anterior.

Como condiciones de borde se toma las estimaciones de las derivadas $I_x = 0$, $I_y = 0$ sobre el borde de la imagen.

La solución para la condición de borde en la estimación del laplaciano se explica más adelante cuando se muestre como se resuelven los algoritmos.

IMPLEMENTACIÓN DEL ALGORITMO DE CÁLCULO DEL FLUJO ÓPTICO

A continuación se detallará los aspectos de diseño y el modo en que se implementó la estimación del Flujo Óptico.

Uno de los objetivos iniciales de este trabajo fue analizar las capacidades de procesamiento SIMD disponibles en la plataforma Edison de Intel.

Intel extensiones streaming SIMD

La tecnología SSE (Streaming SIMD Extension) fue introducida por Intel en 1999 para su línea de procesadores Pentium III. SIMD hace referencia a Single Instruction Multiple Data. Esta tecnología permite a las instrucciones la manipulación de múltiples elementos de datos simultáneamente, logrando así un rendimiento superior. Estas instrucciones operan con paquetes de operandos en punto flotante de precisión simple (FP).

Hay varios tipos de instrucciones SSE

- Instrucciones SSE de Transferencia de datos.
- Instrucciones SSE de Conversión.
- Instrucciones SSE Aritméticas.
- Instrucciones SSE Lógicas.

Con la tecnología SSE, los microprocesadores x86 fueron dotados de setenta nuevas instrucciones y de ocho registros nuevos: del xmm0 al xmm7. Estos registros tienen una

extensión de 128 bits (es decir que pueden almacenar hasta 16 bytes de información cada uno).

Las estructuras de datos y algoritmos involucrados en el procesamiento de imágenes a menudo son candidatos adecuados para optimizaciones utilizando estos conjuntos de instrucciones. Este es el caso de los algoritmos involucrados en la estimación del Flujo Óptico.

En las secciones subsiguientes se detalla como se ha aprovechado la tecnología SSE.

Separación del canal de luminancia

El formato en que el sensor de imagen entrega los frames capturados es el YUYV o YUV422, en donde la información de luminancia se alterna con las componentes U y V de crominancia. La figura 5 muestra la organización de la información.

La primera operación necesaria sobre un frame es separar la información de luminancia de los canales de crominancia. Ver figura 6.

Utilizando los registros SSE se consigue procesar de a 8 pixels en paralelo. El algoritmo primero separa la información de luminancia y luego la convierte a formato de punto flotante en 32 bits.

Las figuras 7, 8, 9 y 10 muestran el proceso realizado en los registros SSE.

Estimación de las derivadas parciales

De acuerdo a lo detallado en la operación de estimar las derivadas parciales consiste en realizar 4 restas y luego un promedio entre los cuatro resultados.

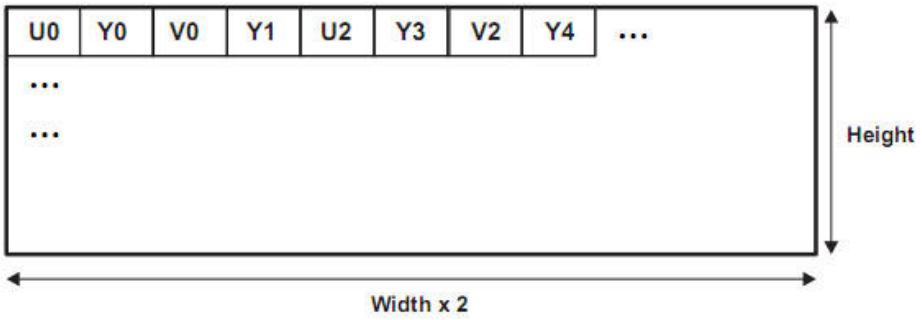

Figure 5: Organización de la información en el formato YUV422. En este formato un frame 160x120 px requiere un buffer de 38.4 Kbytes.

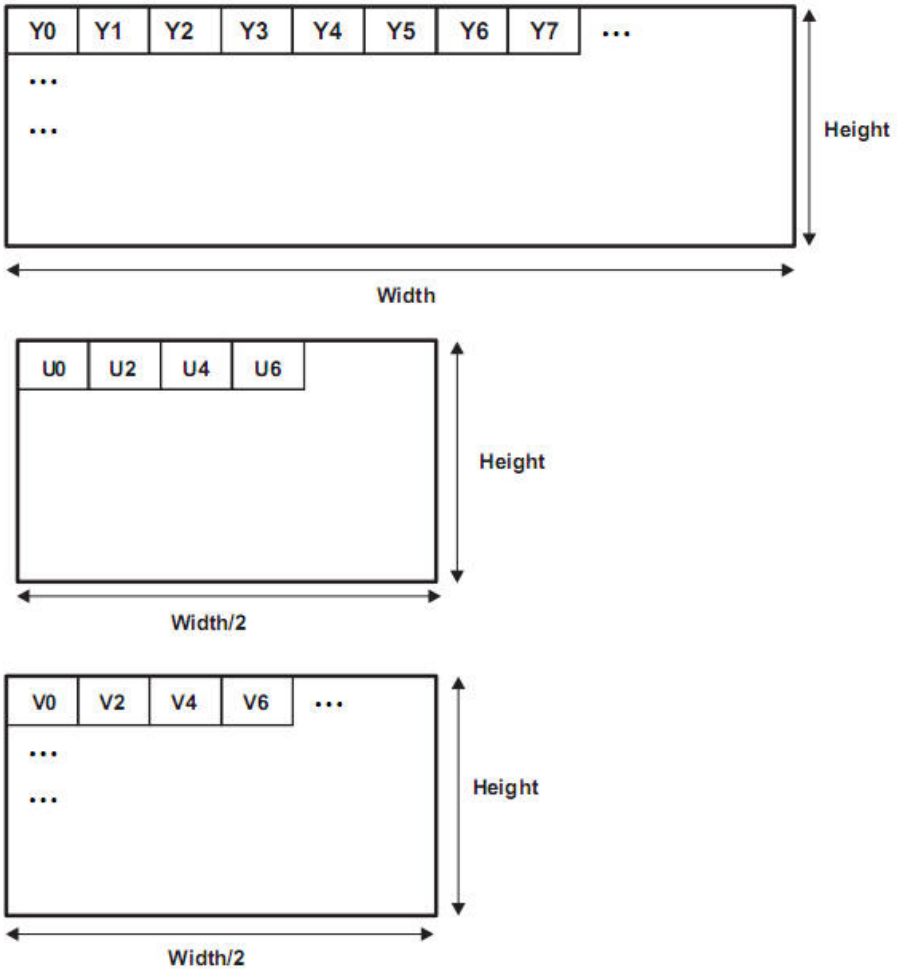

Figure 6: Canales de luminancia y crominancia separados. El buffer de luminancia es de 19.2 Kbytes. Los canales de crominancia se descartan porque la estimación del Flujo Óptico se hace con la información de luminancia.

U_0	Y_0	V_0	Y_1	U_2	Y_2	V_2	Y_3	U_4	Y_4	V_4	Y_5	U_6	Y_6	V_6	Y_7

Figure 7: Se cargan 8 pixels en un registro SSE

Y_0	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7								

Figure 8: Se eliminan los bytes con información de crominancia y se obtiene la luminancia de 8 pixels en formato 8bits

Y_4	Y_5	Y_6	Y_7				
Y_0	Y_1	Y_2	Y_3				

Figure 9: Se expanden los 8 valores de luminancia a 16 bits y se almacenan de a 4 valores en sendos registros SSE

Y_4	Y_5	Y_6	Y_7
Y_0	Y_1	Y_2	Y_3

Figure 10: Luego se expanden a un formato de 32 bits. La información contenida en los dos registros SSE se guarda en un buffer, que contendrá la información de luminancia en formato de punto flotante de 4 bytes

$$I_x = \frac{1}{4}\{(I_{i,j+1,k} - I_{i,j,k}) + (I_{i+1,j+1,k} - I_{i+1,j,k}) +$$

$$(I_{i,j+1,k-1} - I_{i,j,k-1}) + (I_{i+1,j+1,k-1} - I_{i+1,j,k-1})\} \tag{20}$$

Las operaciones de resta se realizan entre los pixels vecinos para el frame t_k y t_{k-1}. Nuevamente el objetivo es poder lograr estimar las derivadas procesando varios pixels en paralelo.

En las siguientes imágenes se muestra el modo en que se ordenan la información de luminancia para poder calcular en paralelo las derivadas parciales. La figura 11 muestra una porción de dos frames consecutivos con información de luminancia I. Dicha información se carga en registros SSE para realizar el cálculo de I_X. Se han de procesar en paralelo de a 4 pixels.

El procedimiento que se detalla en 12 se repite 4 veces, dos para cada frame t_k, t_{k-1}. Se obtiene entonces en 4 registros SSE el resultado de las 4 restas que requiere el cálculo de I_x. Ver figura 13.

El procedimiento es similar para la estimación de las derivadas parciales I_y y I_t. En todos los casos se logra una paralelización del cálculo en un factor x4.

Estimación del Laplaciano

De acuerdo a lo desarrollado en la sección , para estimar el laplaciano se utiliza el kernel que se muestra en la figura 14, que implica sumar el valor de los pixels vecinos con la ponderación

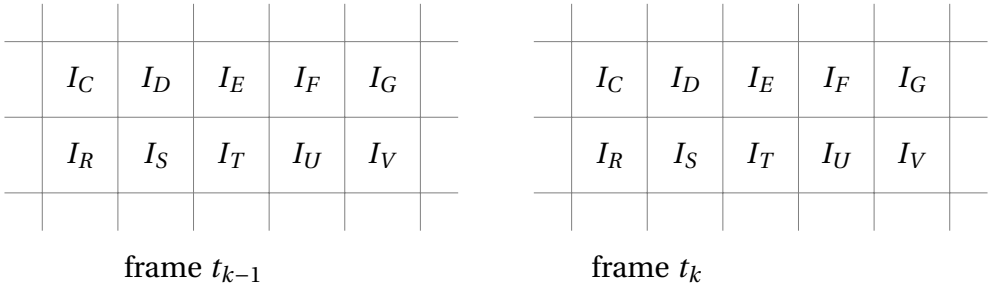

Figure 11: Misma área de 4x2 pixels de dos frames consecutivos en el tiempo
Para el pixel correspondiente a I_C tenemos que
$I_x = \frac{1}{4}\{(I_D - I_C)_k + (I_S - I_R)_k + (I_D - I_C)_{k-1} + (I_S - I_R)_{k-1}\}$

$XMM0$	I_C	I_D	I_E	I_F
$XMM1$	I_D	I_E	I_F	I_G
$XMM2$	$I_D - I_C$	$I_E - I_D$	$I_F - I_E$	$I_G - I_F$

Figure 12: En dos registros SSE se cargan 4 pixels consecutivos, desplazados en 1 columna, de modo que al restar los registros se realiza simultáneamente el cálculo de la primer resta de la expresión I_X para estos 4 pixels.

$XMM5$	$(I_S - I_R)_{k-1}$	$(I_T - I_S)_{k-1}$	$(I_U - I_T)_{k-1}$	$(I_V - I_U)_{k-1}$
$XMM4$	$(I_D - I_C)_{k-1}$	$(I_E - I_D)_{k-1}$	$(I_F - I_E)_{k-1}$	$(I_G - I_F)_{k-1}$
$XMM3$	$(I_S - I_R)_k$	$(I_T - I_S)_k$	$(I_U - I_T)_k$	$(I_V - I_U)_k$
$XMM2$	$(I_D - I_C)_k$	$(I_E - I_D)_k$	$(I_F - I_E)_k$	$(I_G - I_F)_k$
$XMM8$	$(I_X)_D$	$(I_X)_E$	$(I_X)_F$	$(I_X)_F$

Figure 13: Los 4 registros SSE que contiene las 4 restas se suman en entre sí y luego se divide por el factor $\frac{1}{4}$ obteniendo así de manera simultánea las estimación de I_X para 4 pixels.

$\frac{1}{12}$	$\frac{1}{6}$	$\frac{1}{12}$
$\frac{1}{6}$	-1	$\frac{1}{6}$
$\frac{1}{12}$	$\frac{1}{6}$	$\frac{1}{12}$

Figure 14: El laplaciano es estimado sustrayendo el valor en un punto determinado (i, j) de un promedio ponderado de los valores vecinos

correspondiente y al resultado restarle el valor del pixel.

La paralelización del algoritmo no se logra indexando los pixels vecinos en cada cálculo sino con un enfoque similar al utilizado para las derivadas parciales. La figura 15 muestra un conjunto de pixels de un buffer que contiene alguna de las dos componentes estimadas del Flujo Óptico (u, v). Se detallan las expresiones de la estimación para los 4 pixels consecutivos K,L,M,N. En las expresiones que se listan debajo de la figura, se puede observar siempre hay conjuntos de 4 pixels consecutivos que intervienen en el cálculo de cada uno. Los pixels quedan ordenados en columnas. Esta es la condición que se aprovecha a la hora de implementar el algoritmo. Estos conjuntos de 4 pixels consecutivos se cargan en registros SSE y se efectúan las operaciones de suma, de escalado por los factores $\frac{1}{6}$ y $\frac{1}{12}$ respectivamente. Esta operación se realiza para ambas componentes (u, v) del Flujo Óptico. Nuevamente en ambos casos se logra una paralelización del cálculo en un factor x4.

Estimación del Flujo Óptico

Como se expresa en , la estimación del Flujo Óptico se realiza con una algoritmo iterativo que implementa el siguiente cómputo:

	A	B	C	D	E	F	
	J	K	L	M	N	O	
	R	S	T	U	V	W	

Figure 15: Según eq **??** el valor del laplaciano para los pixels K,L,M,N es:
$\overline{u}_K = \frac{1}{6}\{B+L+S+J\} + \frac{1}{12}\{A+C+R+T\}$
$\overline{u}_L = \frac{1}{6}\{C+M+T+K\} + \frac{1}{12}\{B+D+S+U\}$
$\overline{u}_M = \frac{1}{6}\{D+N+U+L\} + \frac{1}{12}\{C+E+T+V\}$
$\overline{u}_N = \frac{1}{6}\{E+O+V+M\} + \frac{1}{12}\{D+F+U+W\}$

$XMM3$	E	O	V	M
$XMM2$	D	N	U	L
$XMM1$	C	M	T	K
$XMM0$	B	L	S	J

$XMM4$	$\frac{1}{6}\{B+L+S+J\}$	$\frac{1}{6}\{C+M+T+K\}$	$\frac{1}{6}\{D+N+U+L\}$	$\frac{1}{6}\{E+O+V+M\}$

Figure 16: En primer lugar se resuelve la suma que es escalada por el factor de $\frac{1}{6}$ y el resultado se guarda en un registro SSE.

$XMM3$	D	F	U	W
$XMM2$	C	E	T	V
$XMM1$	B	D	S	U
$XMM0$	A	C	R	T

$XMM5$	$\frac{1}{12}\{A+C+R+T\}$	$\frac{1}{12}\{B+D+S+U\}$	$\frac{1}{12}\{C+E+T+V\}$	$\frac{1}{12}\{D+F+U+W\}$

Figure 17: En primer lugar se resuelve la suma que es escalada por el factor de $\frac{1}{12}$ y el resultado se guarda en un registro SSE.

$XMM6$	K	L	M	N
$XMM4$	$\frac{1}{6}\{B+L+S+J\}$	$\frac{1}{6}\{C+M+T+K\}$	$\frac{1}{6}\{D+N+U+L\}$	$\frac{1}{6}\{E+O+V+M\}$
$XMM5$	$\frac{1}{12}\{A+C+R+T\}$	$\frac{1}{12}\{B+D+S+U\}$	$\frac{1}{12}\{C+E+T+V\}$	$\frac{1}{12}\{D+F+U+W\}$
$XMM7$	$\overline{u}_K$	$\overline{u}_L$	$\overline{u}_M$	$\overline{u}_K$

Figure 18: Teniendo los factores escalados se realiza la suma para así calcular de manera paralela el valor del laplaciano para 4 pixels.

$$u^{n+1} = \overline{u}^n - I_x \left[I_x \overline{u}^n + I_y \overline{v}^n + I_t \right] / \left(\alpha^2 + I_x^2 + I_y^2 \right) \tag{21}$$

$$v^{n+1} = \overline{v}^n - I_y \left[I_x \overline{u}^n + I_y \overline{v}^n + I_t \right] / \left(\alpha^2 + I_x^2 + I_y^2 \right) \tag{22}$$

La condición inicial para la primera iteración $n = 1$ es que $\overline{u}^n = \overline{v}^n = 0$. Atendiendo esta condición particular para el cálculo en la primera iteración se determinó eficiente separar el algoritmo para $n = 1$ (eq. 23 y 24) y para $n > 1$ (eq. 21 y 22). Esta consideración redunda en una simplificación del cálculo en la primera iteración ya que no es necesario estimar el Laplaciano del campo de velocidades porque se supone nulo. Como se verá más adelante esta decisión impacta directamente en el tiempo de cálculo del Flujo Óptico.

Las ecuaciones 23, 24, 21 y 21 representan siempre operaciones escalares entre todos los valores para una coordenada (i, j) de un pixel. Es decir que dicha operación se debe efectuar para cada uno de los 19200 pixels de un frame. Claramente ambas ecuaciones permiten ser paralelizadas. Y trabajando de manera similar a como se detalló para la estimación de las derivadas parciales y del laplaciano, se consigue un grado de paralelización x4.

$$u^1 = (-I_x I_t) / \left(\alpha^2 + I_x^2 + I_y^2\right) \tag{23}$$

$$v^1 = \left(-I_y I_t\right) / \left(\alpha^2 + I_x^2 + I_y^2\right) \tag{24}$$

ENSAYO DEL ALGORITMO

El algoritmo desarrollado fue probado en una plataforma Intel Edison, procesando los frames obtenidos de un sensor de imagen de bajo costo (webcam). En particular el algoritmo fue empleado en el desarrollo de un sensor de flujo óptico para ser utilizado en un vehículo multirotor, [7].

De acuerdo a los ensayos y mediciones realizas se puede caracterizar la carga que representa el sensor a la placa Edison. Las gráficas 19, 20 y 21 muestran la relación entre el porcentaje de uso del procesador en función de la cantidad de iteraciones que realice el algoritmo y la tasa FPS a la que esté trabajando el sensor.

Con esta información y en función de la carga de la aplicación usuaria del sensor se puede determinar un punto de trabajo en términos de la tasa de FPS y la cantidad de iteraciones a realizar.

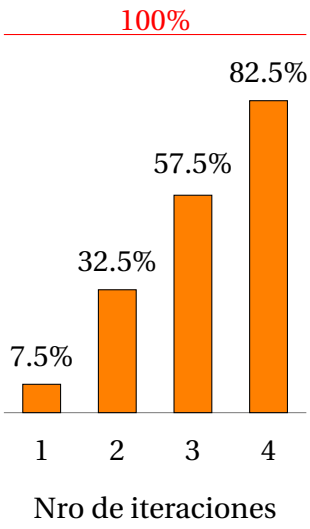

Figure 19: Porcentaje de carga del procesador para distintos niveles de iteración del algoritmo trabajando a 25 FPS.

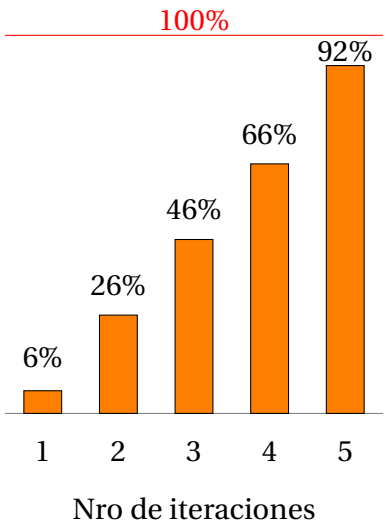

Figure 20: Porcentaje de carga del procesador para distintos niveles de iteración del algoritmo trabajando a 20 FPS.

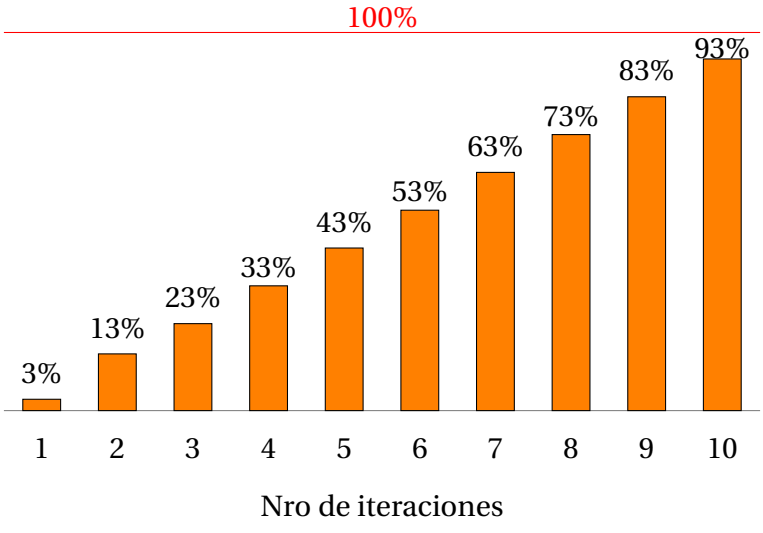

Figure 21: Porcentaje de carga del procesador para distintos niveles de iteración del algoritmo trabajando a 10 FPS.

El sensor puede trabajar con bajos tiempos de exposición en condiciones normales de iluminación. También se efectuó un ensayo en un ambiente de baja iluminación y los resultados obtenidos fueron satisfactorios. Para estos casos se puede modificar el valor del parámetro α presente en la solución iterativa:

$$u^{n+1} = \overline{u}^n - I_x \left[I_x \overline{u}^n + I_y \overline{v}^n + I_t \right] / \left(\alpha^2 + I_x^2 + I_y^2 \right) \tag{25}$$

$$v^{n+1} = \overline{v}^n - I_y \left[I_x \overline{u}^n + I_y \overline{v}^n + I_t \right] / \left(\alpha^2 + I_x^2 + I_y^2 \right) \tag{26}$$

En la práctica este parámetro termina actuando como un ajuste de la sensibilidad del sensor. El valor por default es $\alpha = 100.0$. Cuánto más chico es el valor, más sensibilidad tendrá el sensor.

CONCLUSIONES

Los resultados muestran que la utilización de las capacidades de procesamiento SIMD de la plataforma permitió lograr una implementación de los algoritmos apropiada para ser incorporada a otras aplicaciones que requerían calcular el flujo óptico, dado que el algoritmo deja suficiente tiempo de procesador disponible para ser utilizado por otros procesos.

Referenciasmkboth

REFERENCESREFERENCES

[1] J. J. Gibson, *The Perception of the Visual World*, Houghton Mifflin,1950.

[2] Berthold K. P. Horn y Brian G. Schunck, *Determining Optical Flow*, Artificial Intelligence 17 (1981) p185-203.

[3] J. L. Barron and D. J. Fleet and S. S. Beauchemin, *Performance of optical flow techniques*, International Journal of Computer Vision 1994, vol. 12, p43-77.

[4] Lucas, Bruce D. and Kanade, Takeo, *An Iterative Image Registration Technique with an Application to Stereo Vision*, Proceedings of the 7th International Joint Conference on Artificial Intelligence (1981), vol. 2 p674-679.

[5] E. P. Simoncelli and E. H. Adelson and D. J. Heeger, *Probability distributions of optical flow*, Computer Vision and Pattern Recognition 1991, Proceedings CVPR '91., IEEE Computer Society Conference on, p310-315

[6] Nagel, H., *Displacement vectors derived from second-order intensity variations in image sequences*, Computer Vision, Graphics, and Image Processing 1993, vol.21 p85.117.

[7] J. E. Luiso y J. I. Giribet, *Sensor de Flujo Óptico*, RPIC 2017 - XVII Reunión de trabajo en Procesamiento de la Información y Control - 20-21 de Septiembre de 2017, Mar del Plata.