Ciencia y Tecnología
Secretaría de Ciencia, Humanidades, Tecnología e Innovación

CIMAT
UNIDAD MÉRIDA

ALGORITMOS DE ESTIMACIÓN DE MOVIMIENTO

Dr. Francisco J. Hernández López
SECIHTI – CIMAT-Mérida
fcoj23@cimat.mx, www.cimat.mx/~fcoj23

ESTIMACIÓN DE MOVIMIENTO GLOBAL

- El movimiento de la cámara afecta el movimiento de todos los pixeles de la secuencia de video
- Un algoritmo de compensación de movimiento global (CMG) puede ser incluido en algún compresor de video (como MPEG)
- Bajo el modelo de intensidad constante y el criterio de error cuadrático, tenemos el siguiente problema de optimización:

$$\min_{\vec{v}} \mathcal{E}(\vec{v}), \quad \mathcal{E}(\vec{v}) = \sum_{\vec{x}} \mathcal{E}^2(\vec{x}), \quad \mathcal{E}(\vec{x}) = I_k(\vec{x}) - I_{k-1}[\vec{x} - \vec{v}(\vec{x})(t_k - t_{k-1})]$$

asumiendo $t_k - t_{k-1} = 1$

Agunas elecciones de $\vec{v}(\vec{x})$ pueden ser:

$$\begin{cases} \vec{v}(\vec{x}) = \begin{pmatrix} p_1 \\ p_2 \end{pmatrix} \\ \vec{v}(\vec{x}) = \begin{pmatrix} p_1 \\ p_2 \end{pmatrix} + \begin{pmatrix} p_3 & p_4 \\ p_5 & p_6 \end{pmatrix} \vec{x} \end{cases}$$

Alan C. Bovik. 2009. *The Essential Guide to Video Processing* (2nd ed.). Academic Press

ESTIMACIÓN DE MOVIMIENTO GLOBAL (C1)

- Ya que la dependencia entre $\mathcal{E}(\vec{v})$ y $\vec{b}$ es no lineal, entonces se utiliza un procedimiento de minimización iterativo, por ejemplo:

$$\vec{b}^{k+1} = \vec{b}^{k} + B^{-1}\vec{c}$$

donde B es la matriz Hessiana de $K \times K$

$$B_{k_1 k_2} = \sum_{\vec{x}} \frac{\partial^2 \mathcal{E}^2(\vec{x})}{\partial b_{k1} \partial b_{k2}}$$

y $\vec{c}$ un vector

$$\vec{c} = -\sum_{\vec{x}} \mathcal{E}(\vec{x}) \frac{\partial \mathcal{E}(\vec{x})}{\partial b_{k1}}$$

EMPAREJAMIENTO DE BLOQUES

- Es el algoritmo más simple para estimación de movimiento local
- Utiliza un modelo de movimiento constante en espacio y lineal en el tiempo sobre una región de soporte rectangular
- Puede ser implementado en tiempo real usando VLSI (*Very Large Scale Integration*), CUDA (*Compute Unified Device Architecture*), etc.
- El objetivo es minimizar:

$$\min_{\vec{d}(\vec{x})\in S} \mathcal{E}\left(\vec{d}(\vec{x})\right), \quad \mathcal{E}\left(\vec{d}(\vec{x})\right) = \sum_{\vec{y}\in R_{\vec{x}}} \Phi[I_k(\vec{y}) - I_{k-1}(\vec{y} - d(\vec{x}))], \quad \forall\vec{x}$$

donde S es el área de búsqueda en la cual $d(\vec{x})\exists$

Alan C. Bovik. 2009. *The Essential Guide to Video Processing* (2nd ed.). Academic Press

CORRELACIÓN DE FASE, PHASE ONLY CORRELATION (POC)

- Método basado en el dominio de la frecuencia
- Se utiliza para estimar el movimiento global
- En combinación con el emparejamiento de bloques, POC explota las ventajas de los criterios tanto en el dominio espacial como en la frecuencia
- La idea es calcular los candidatos $\vec{d}$ en el dominio de la frecuencia y entonces realizar una búsqueda con emparejamiento de bloques solo para dichos candidatos (evitando la búsqueda exhaustiva)

Alan C. Bovik. 2009. *The Essential Guide to Video Processing* (2nd ed.). Academic Press

POC (C1)

- Dado el criterio de correlación cruzada

$$C(\vec{d}) = \sum_{\vec{x}} f(\vec{x}) g\left(\vec{x} - \vec{d}(\vec{x})\right),$$

- Usando Fourier puede ser expresado como

$$\mathcal{F}\left(C(\vec{d})\right) = \mathcal{F}\left[\sum_{\vec{x}} f(\vec{x}) g\left(\vec{x} - \vec{d}(\vec{x})\right)\right] = \mathcal{F}(\omega) G^{*}(\omega),$$

donde $G^{*}(\omega)$ es el conjugado de $G(\omega)$

- Normalizando $\mathcal{F}\left(C(\vec{d})\right)$ y tomando la inversa de Fourier

$$r(\vec{x}) = \mathcal{F}^{-1}\left\{\frac{\mathcal{F}(\omega) G^{*}(\omega)}{|\mathcal{F}(\omega) G^{*}(\omega)|}\right\}$$

Entonces $r(\vec{x})$ es la correlación normalizada entre f y g

POC (C2)

- En el caso especial de una traslación global
$$f(\vec{x}) = g(\vec{x} - \vec{d}),$$

- Usando la propiedad del corrimiento podemos que ver que la inversa de la transformada de Fourier de una exponencial compleja es una función delta de Kronecker:
$$r(\vec{x})\Big|_{f(\vec{x})=g(\vec{x}-\vec{d})} = \delta(\vec{x} - \vec{d})$$

- En la práctica $r(\vec{x})$ no es solo un impulso, tiene numerosos picos, los cuales pueden ser utilizados como candidatos en un emparejamiento por bloques

ALGORTIMO POC CON EMPAREJAMIENTO DE BLOQUES

1. Dividir I_{k-1} y I_k en bloques grandes I_{k-1}^m y I_k^m $(64 \times 64$ considerando movimientos de hasta ± 32 pixeles) para $m = 1, \ldots, M$, con M el número de bloques

2. Calcular la transformada de Fourier de cada bloque

3. Calcular $r^m(\vec{x})$ con los correspondientes bloques I_{k-1}^m y I_k^m

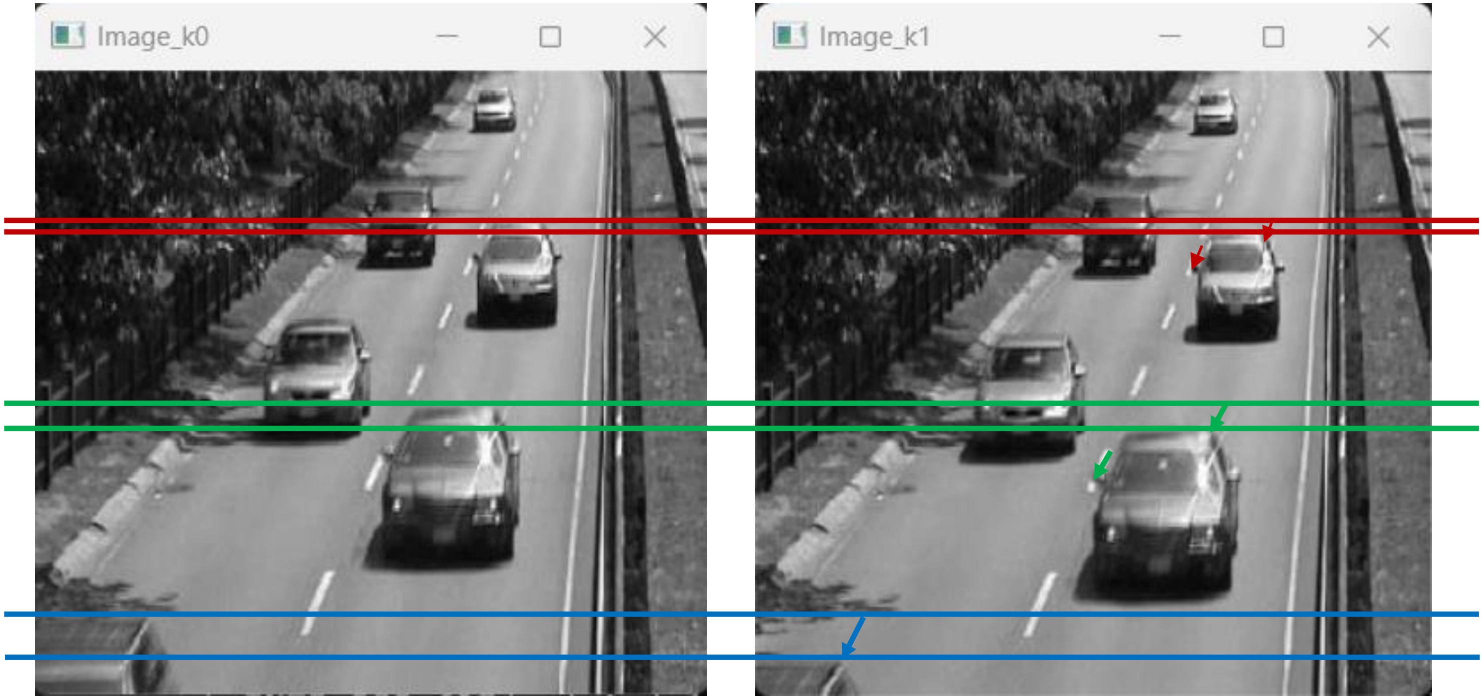
4. Estimar n desplazamientos candidatos $\vec{d}_n^m$ en cada bloque, a partir de los picos más altos de $r^m(\vec{x})$, (por ej. con $n = 5$)

5. Utilizar la estrategia de emparejamiento de bloques con los $\vec{d}_n^m$ candidatos para obtener el $\vec{d}^*$ óptimo en cada pixel (por ej. con ventanas de 16×16)

Alan C. Bovik. 2009. *The Essential Guide to Video Processing* (2nd ed.). Academic Press

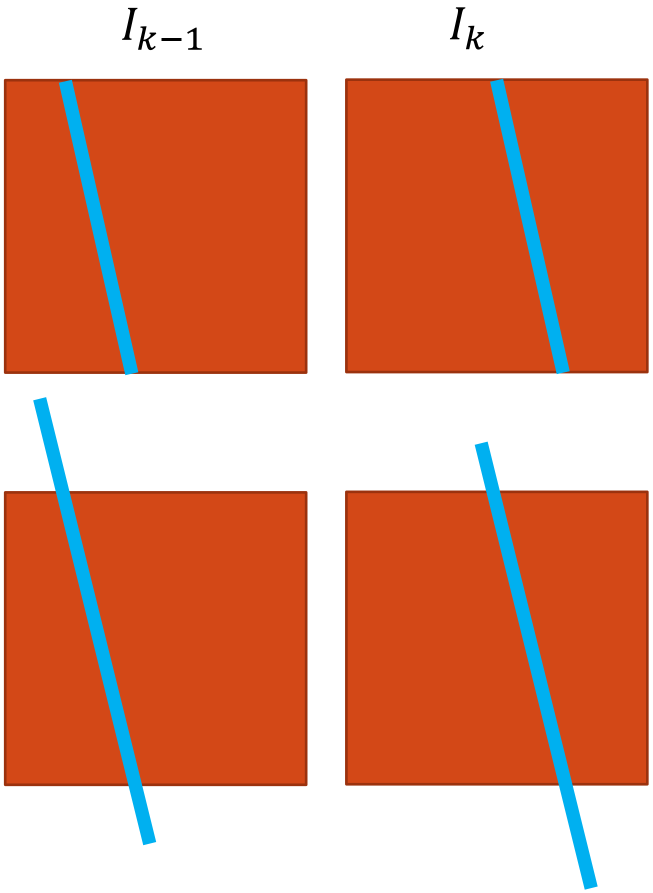
FLUJO ÓPTICO

- Refleja los cambios en la imagen debido al movimiento en un intervalo de tiempo dt, el campo del flujo óptico es el campo de velocidades que representan el movimiento 3D de los objetos a través de una imagen 2D



Sonka, Milan, Vaclav Hlavac, and Roger Boyle. Image processing, analysis, and machine vision. Cengage Learning, 2014.

PROBLEMA DE LA APERTURA



Movimiento percibido

Solo puedo ver en dirección del gradiente

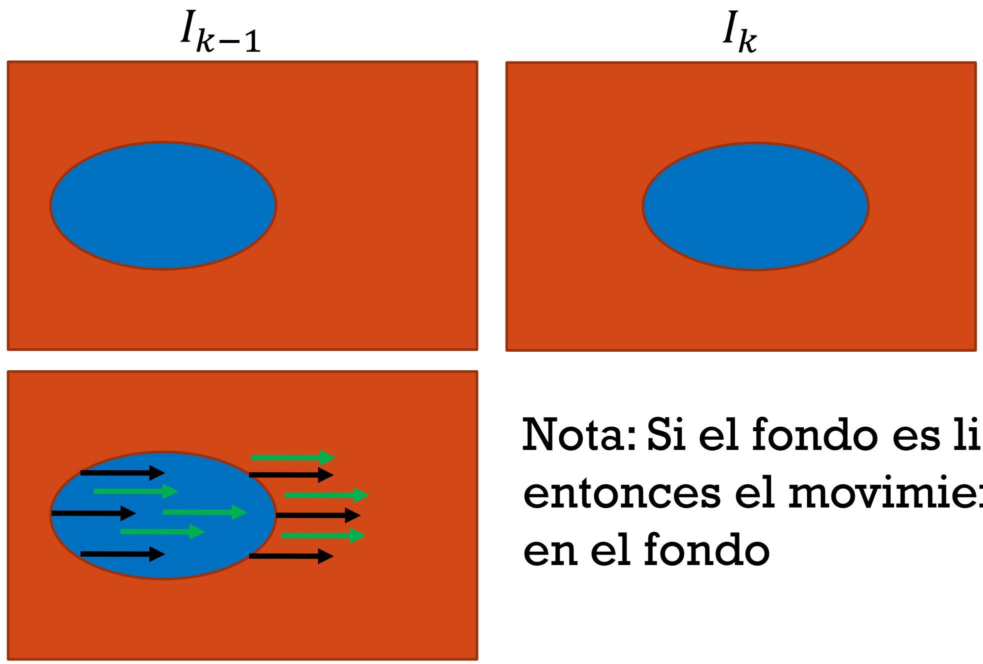
Movimiento real

barber's pole
https://en.wikipedia.org/wiki/Barberpole_illusion

REGULARIZACIÓN

- Por el problema de la apertura, podemos agregar un término de regularización

$$\vec{d}(\vec{x}) \approx \vec{d}(\vec{y}), con\ \|\vec{x} - \vec{y}\| = 1$$



Nota: Si el fondo es liso (no tiene textura), entonces el movimiento se va a extender en el fondo

FLUJO ÓPTICO (HORN AND SCHUNK, 1981)

- Sea $I(x, y, t)$ una secuencia de video y $(x(t), y(t))$ la trayectoria de un punto proyectado en el plano de la imagen
- La suposición de brillo constante establece que las intensidades del pixel (x, y) permanecen constantes a través del tiempo

$$I(x + \delta x, y + \delta y, t + \delta t) = I(x, y, t)$$

- Usando Taylor de primer orden, tenemos que

$$I(x + \delta x, y + \delta y, t + \delta t) \approx I(x, y, t) + \frac{\partial I(x, y, t)}{\partial x}\delta x + \frac{\partial I(x, y, t)}{\partial y}\delta y + \frac{\partial I(x, y, t)}{\partial t}\delta t$$

Klette, R. (2014). *Concise computer vision* (Vol. 233, pp. 2-1). London: Springer.

FLUJO ÓPTICO (HORN AND SCHUNK, 1981) (C1)

- Queremos que

$$\frac{\partial I(x,y,t)}{\partial x}\delta x + \frac{\partial I(x,y,t)}{\partial y}\delta y + \frac{\partial I(x,y,t)}{\partial t}\delta t = 0$$

- Dividiendo entre δt llegamos a la ec. de restricción del flujo óptico de Horn-Schunk

$$\frac{\partial I(x,y,t)}{\partial x}u(x,y,t) + \frac{\partial I(x,y,t)}{\partial y}v(x,y,t) + \frac{\partial I(x,y,t)}{\partial t} = 0$$

con $u(x,y,t) = \frac{\delta x}{\delta t}$ y $v(x,y,t) = \frac{\delta y}{\delta t}$

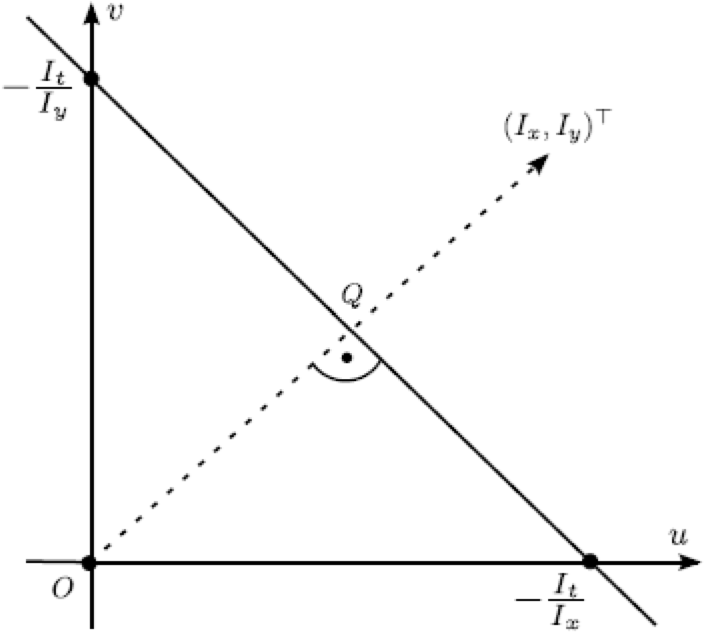
Klette, R. (2014). *Concise computer vision* (Vol. 233, pp. 2-1). London: Springer.

FLUJO ÓPTICO (HORN AND SCHUNK, 1981) (C2) PRIMERA RESTRICCIÓN

- Podemos escribir la ec. de restricción del flujo óptico de Horn-Schunk de la sig. manera

$$-I_t = uI_x + vI_y = \vec{u}\nabla I$$

- Define una recta en el espacio (u, v)
- La solución que queremos es un punto en esa recta
- Tenemos solo una ecuación con dos incognitas, por lo que necesitamos otra restricción para poder resolver el problema



Klette, R. (2014). *Concise computer vision* (Vol. 233, pp. 2-1). London: Springer.

FLUJO ÓPTICO (HORN AND SCHUNK, 1981) (C3) SEGUNDA RESTRICCIÓN

- Movimiento espacial constante. Se asume que pixeles adyacentes en algún pixel (x, y) tienen el mismo vector del flujo óptico (u, v)
- El problema se puede escribir como

$$E_T = \underbrace{\sum_{\Omega}\left[uI_x + vI_y + I_t\right]^2}_{\substack{\text{Término de datos} \\ E_D}} + \lambda \underbrace{\sum_{\Omega}\left[\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial u}{\partial y}\right)^2 + \left(\frac{\partial v}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2\right]}_{\substack{\text{Término de restricción de suavidad} \\ E_S}}$$

Klette, R. (2014). *Concise computer vision* (Vol. 233, pp. 2-1). London: Springer.

FLUJO ÓPTICO (HORN AND SCHUNK, 1981) (C4)

- Derivadas del término de datos c.r.a u_{xy} y v_{xy}

$$\frac{\partial E_D}{\partial u_{xy}}(u,v) = 2\big[I_x(x,y)u_{xy} + I_y(x,y)v_{xy} + I_t(x,y)\big]I_x(x,y)$$

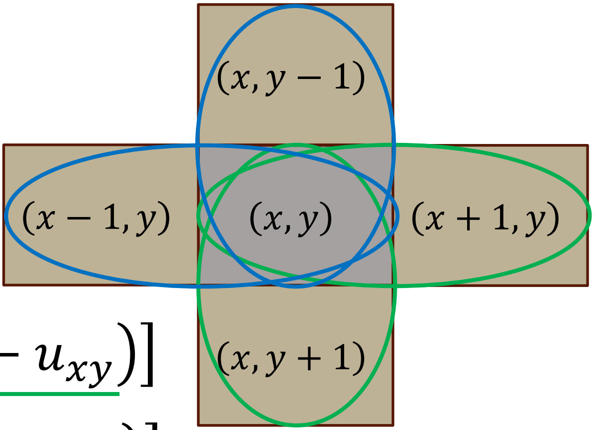
$$\frac{\partial E_D}{\partial v_{xy}}(u,v) = 2\big[I_x(x,y)u_{xy} + I_y(x,y)v_{xy} + I_t(x,y)\big]I_y(x,y)$$

- Podemos aprox. las derivadas del término de suavidad como

$$E_S = \sum_{x,y} \begin{matrix} \left(u_{x+1,y} - u_{xy}\right)^2 + \left(u_{x,y+1} - u_{xy}\right)^2 \\ +\left(v_{x+1,y} - v_{xy}\right)^2 + \left(v_{x,y+1} - v_{xy}\right)^2 \end{matrix}$$

Klette, R. (2014). *Concise computer vision* (Vol. 233, pp. 2-1). London: Springer.

FLUJO ÓPTICO (HORN AND SCHUNK, 1981) (C5)



- Der. del término de suavidad c.r.a u_{xy} y v_{xy}

$$\frac{\partial E_S}{\partial u_{xy}}(u, v) = -2\left[\left(u_{x+1,y} - u_{xy}\right) + \left(u_{x,y+1} - u_{xy}\right)\right]$$
$$+2\left[\left(u_{xy} - u_{x-1,y}\right) + \left(u_{xy} - u_{x,y-1}\right)\right]$$

$$= 2\begin{bmatrix} \left(u_{xy} - u_{x+1,y}\right) + \left(u_{xy} - u_{x,y+1}\right) \\ +\left(u_{xy} - u_{x-1,y}\right) + \left(u_{xy} - u_{x,y-1}\right) \end{bmatrix}$$

- Eso se puede reducir a

$$\frac{1}{4}\frac{\partial E_S}{\partial u_{xy}}(u, v) = 2\left[u_{xy} - \overbrace{\frac{1}{4}\left(u_{x+1,y} + u_{x,y+1} + u_{x-1,y} + u_{x,y-1}\right)}^{\bar{u}_{xy}}\right]$$

Klette, R. (2014). *Concise computer vision* (Vol. 233, pp. 2-1). London: Springer.

FLUJO ÓPTICO (HORN AND SCHUNK, 1981) (C6)

- Igualando a cero las derivadas, llegamos a estas ecuaciones

$$\left[I_x(x,y)u_{xy} + I_y(x,y)v_{xy} + I_t(x,y)\right]I_x(x,y) + \lambda\left[u_{xy} - \bar{u}_{xy}\right] = 0$$

$$\left[I_x(x,y)u_{xy} + I_y(x,y)v_{xy} + I_t(x,y)\right]I_y(x,y) + \lambda\left[v_{xy} - \bar{v}_{xy}\right] = 0$$

- Por regla de Cramer tenemos

$$u_{xy}^{n+1} = \bar{u}_{xy}^{n} - I_x(x,y)\,\frac{I_x(x,y)\,\bar{u}_{xy}^{n} + I_y(x,y)\,\bar{v}_{xy} + I_t(x,y)}{\lambda + I_x^2(x,y) + I_y^2(x,y)}$$

$$v_{xy}^{n+1} = \bar{v}_{xy}^{n} - I_y(x,y)\,\frac{I_x(x,y)\,\bar{u}_{xy}^{n} + I_y(x,y)\,\bar{v}_{xy} + I_t(x,y)}{\lambda + I_x^2(x,y) + I_y^2(x,y)}$$

Klette, R. (2014). *Concise computer vision* (Vol. 233, pp. 2-1). London: Springer.

ALGORITMO DE FLUJO ÓPTICO (HS)

1. Inicializar $u(x, y) = 0$ y $v(x, y) = 0$ para $(x, y) \in \Omega$
2. Dado un máximo número de iteraciones N, calcular los valores de u^{n+1} y v^{n+1} para todos los pixeles (x, y)

$$u_{xy}^{n+1} = \bar{u}_{xy}^{n} - I_x(x, y) \frac{I_x(x, y)\,\bar{u}_{xy}^{n} + I_y(x, y)\,\bar{v}_{xy} + I_t(x, y)}{\lambda + I_x^2(x, y) + I_y^2(x, y)}$$

$$v_{xy}^{n+1} = \bar{v}_{xy}^{n} - I_y(x, y) \frac{I_x(x, y)\,\bar{u}_{xy}^{n} + I_y(x, y)\,\bar{v}_{xy} + I_t(x, y)}{\lambda + I_x^2(x, y) + I_y^2(x, y)}$$

3. Parar si

$$\sum_{x,y} E_T^2(x, y) < \epsilon,$$

donde ϵ es el mínimo error permitido, en otro caso ir al paso 2

Sonka, Milan, Vaclav Hlavac, and Roger Boyle. Image processing, analysis, and machine vision. Cengage Learning, 2014.

FLUJO ÓPTICO (LUCAS-KANADE, 1981)

- Asumiendo intensidad constante
$$I_k(x+\delta x, y+\delta y) - I_{k-1}(x,y) = 0$$
- Asumiendo que los desplazamientos son pequeños entonces (usando Taylor hasta el término de primer orden)
$$I_k(x+\delta x, y+\delta y) \approx I_k(x,y) + \frac{\partial I_k}{\partial x}\delta x + \frac{\partial I_k}{\partial y}\delta y$$
$$\overbrace{I_k(x,y) - I_{k-1}(x,y)}^{\frac{\partial I}{\partial t}} + \frac{\partial I_k}{\partial x}\delta x + \frac{\partial I_k}{\partial y}\delta y = 0$$
- Asumiendo que en la vecindad de un pixel (x,y) los desplaz. son similares:
  - Por ejemplo, en una ventana de 3×3 centrada en (x,y), tendríamos 9 ecuaciones lineales para calcular los desplaz. $(\delta x, \delta y)$

Sonka, Milan, Vaclav Hlavac, and Roger Boyle. Image processing, analysis, and machine vision. Cengage Learning, 2014.

FLUJO ÓPTICO (LUCAS-KANADE, 1981) (C1)

- Podemos construir el sistema lineal

$$A\begin{pmatrix}\delta x\\ \delta y\end{pmatrix} = \vec{c}$$

con A una matriz de 9×2, donde cada fila es una estimación de $\frac{\partial I_k}{\partial x}$, $\frac{\partial I_k}{\partial y}$ y $\vec{c}$ es un vector de 9×1 en donde cada componente es la diferencia de intensidades entre I_k y I_{k-1} en los respectivos pixeles

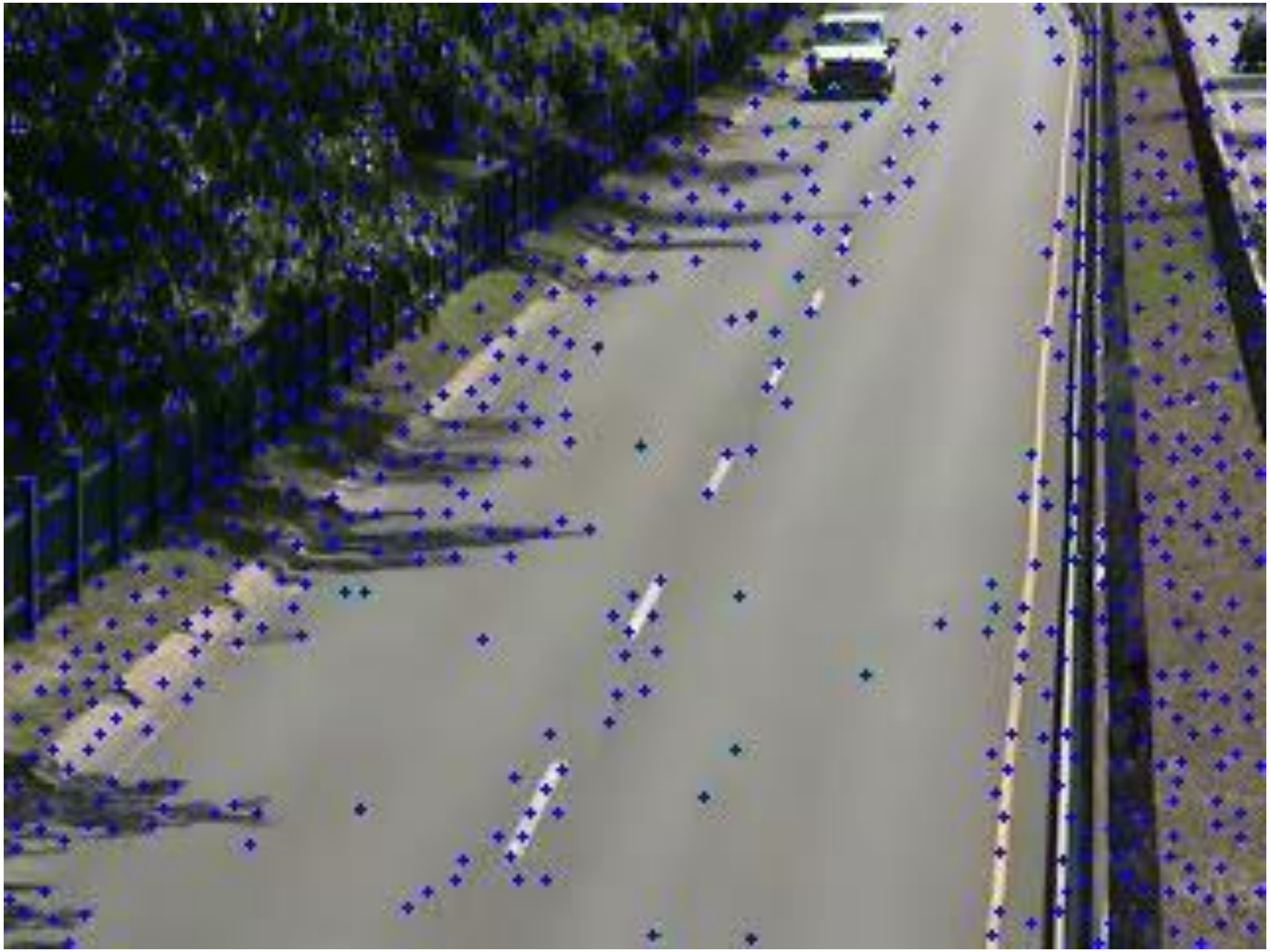
- La solución de este sistema puede ser

$$A^T A\begin{pmatrix}\delta x\\ \delta y\end{pmatrix} = A^T\vec{c}, \qquad A^T A = \begin{pmatrix}\sum \frac{\partial I_k}{\partial x}\frac{\partial I_k}{\partial x} & \sum \frac{\partial I_k}{\partial x}\frac{\partial I_k}{\partial y}\\ \sum \frac{\partial I_k}{\partial x}\frac{\partial I_k}{\partial y} & \sum \frac{\partial I_k}{\partial y}\frac{\partial I_k}{\partial y}\end{pmatrix}$$

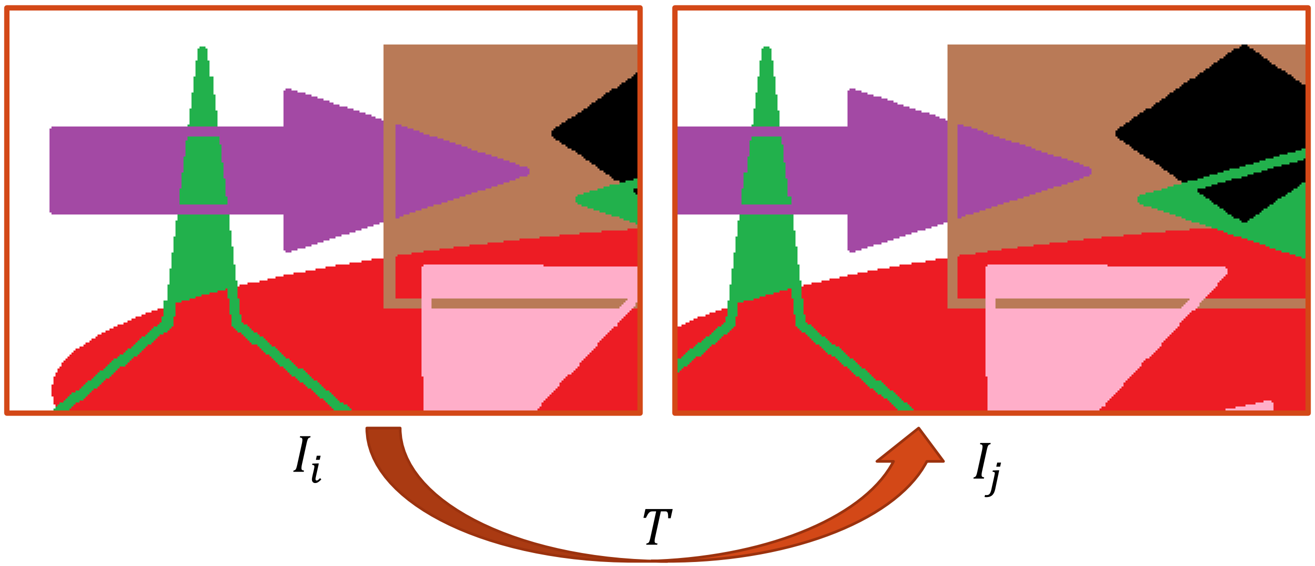
el cual es un sistema de 2×2 que tiene solución si $(A^T A)^{-1}$ $\exists$

Sonka, Milan, Vaclav Hlavac, and Roger Boyle. Image processing, analysis, and machine vision. Cengage Learning, 2014.

EJEMPLO DE FLUJO ÓPTICO USANDO PUNTOS CARACTERÍSTICOS Y LK



REGISTRO O ALINEAMIENTO DE IMÁGENES



La idea es encontrar una transformación T, que relacione ambas imágenes, con el fin de poder traslaparlas. Existen diferentes métodos, estos se aplican o se construyen a partir de los diferentes niveles de procesamiento de la imagen:

- Por pixel
- Por regiones
- Por características, etc.

ALINEAMIENTO DE IMÁGENES

- Problema:

$$\min_{\vec{p}} \sum_{\vec{x}} \left[I\left(W(\vec{x};\vec{p})\right) - T(\vec{x}) \right]^2$$

$W(\vec{x};\vec{p})$ → Transformación 2D

$\vec{x}$ → Coordenadas 2D

$\vec{p} = \{p_1, \dots, p_N\}$ → Parámetros de la transf.

$I(\vec{x}') = I\left(W(\vec{x};\vec{p})\right)$ → Imagen transformada o deformada

$T(\vec{x})$ → Imagen de referencia o *template*

Baker, S., & Matthews, I. (2004). Lucas-kanade 20 years on: A unifying framework. International journal of computer vision, 56, 221-255.
Computer Vision Lectures: http://www.cs.cmu.edu/~16385/

ALINEAMIENTO USANDO LUCAS-KANADE

- Dada una buena inicialización de los parámetros $\vec{p}$
- Podemos escribir el problema de la siguiente manera:

$$\sum_{\vec{x}} \left[I\left(W\left(\vec{x}; \vec{p} + \overrightarrow{\Delta p}\right)\right) - T(\vec{x}) \right]^2, \qquad (1)$$

en donde la idea es estimar el pequeño incremento $\overrightarrow{\Delta p}$

- Usando la aprox. de Taylor de primer orden:

$$f(x, y) \approx f(a, b) + f_x(a, b)(x - a) + f_y(a, b)(y - b)$$

temenos que

$$I\left(W\left(\vec{x}; \vec{p} + \overrightarrow{\Delta p}\right)\right) \approx I\big(W(\vec{x}; \vec{p})\big) + \frac{\partial I\big(W(\vec{x}; \vec{p})\big)}{\partial \vec{p}} \overrightarrow{\Delta p}$$

Computer Vision Lectures: http://www.cs.cmu.edu/~16385/

Alineamiento usando Lucas-Kanade (C1)

- Usando la regla de la cadena

$$I\left(W\left(\vec{x};\vec{p}+\overrightarrow{\Delta p}\right)\right) = I\left(W(\vec{x};\vec{p})\right) + \frac{\partial I\left(W(\vec{x};\vec{p})\right)}{\partial \vec{x}'}\frac{\partial W(\vec{x};\vec{p})}{\partial \vec{p}}\overrightarrow{\Delta p}$$

$$= I\left(W(\vec{x};\vec{p})\right) + \nabla I(\vec{x}')\frac{\partial W}{\partial \vec{p}}\overrightarrow{\Delta p}$$

- Sustituyendo en la ec. (1)

$$\sum_{\vec{x}}\left[I\left(W(\vec{x};\vec{p})\right) + \nabla I(\vec{x}')\frac{\partial W}{\partial \vec{p}}\overrightarrow{\Delta p} - T(\vec{x})\right]^2, \qquad (2)$$

ahora la función es lineal en los parámetros $\overrightarrow{\Delta p}$

Computer Vision Lectures: http://www.cs.cmu.edu/~16385/

¿CÓMO SE CALCULA EL JACOBIANO?

- Podemos escribir la transformación W como

$$W = \begin{bmatrix} w_x\,(x,y) \\ w_y\,(x,y) \end{bmatrix}$$

- Entonces el Jacobiano se puede representar de la sig. manera

$$\frac{\partial W}{\partial \vec{p}} = \begin{bmatrix} \frac{\partial w_x}{\partial p_1} & \frac{\partial w_x}{\partial p_2} & \cdots & \frac{\partial w_x}{\partial p_N} \\ \frac{\partial w_y}{\partial p_1} & \frac{\partial w_y}{\partial p_2} & \cdots & \frac{\partial w_y}{\partial p_N} \end{bmatrix}$$

- Por ej., para el modelo Affine:

$$W(\vec{x};\vec{p}) = \begin{bmatrix} p_1 x + p_2 y + p_3 \\ p_4 x + p_5 y + p_6 \end{bmatrix}, \frac{\partial W(\vec{x};\vec{p})}{\partial \vec{p}} = \begin{bmatrix} x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \end{bmatrix}$$

Computer Vision Lectures: http://www.cs.cmu.edu/~16385/

ALINEAMIENTO USANDO LUCAS-KANADE (C2)

- El problema lo podemos escribir también de esta forma

$$\min_{\overrightarrow{\Delta p}} \sum_{\vec{x}} \left[\nabla I(\vec{x}') \frac{\partial W}{\partial \vec{p}} \overrightarrow{\Delta p} - \left[T(\vec{x}) - I\big(W(\vec{x};\vec{p})\big) \right] \right]^2 \qquad (3)$$

- Aprox. por mínimos cuadrados

$$\widehat{\overrightarrow{\Delta p}} = \arg\min_{\overrightarrow{\Delta p}} \left\| A\overrightarrow{\Delta p} - \vec{b} \right\|^2$$

$$\overrightarrow{\Delta p} = (A^T A)^{-1} A^T \vec{b}$$

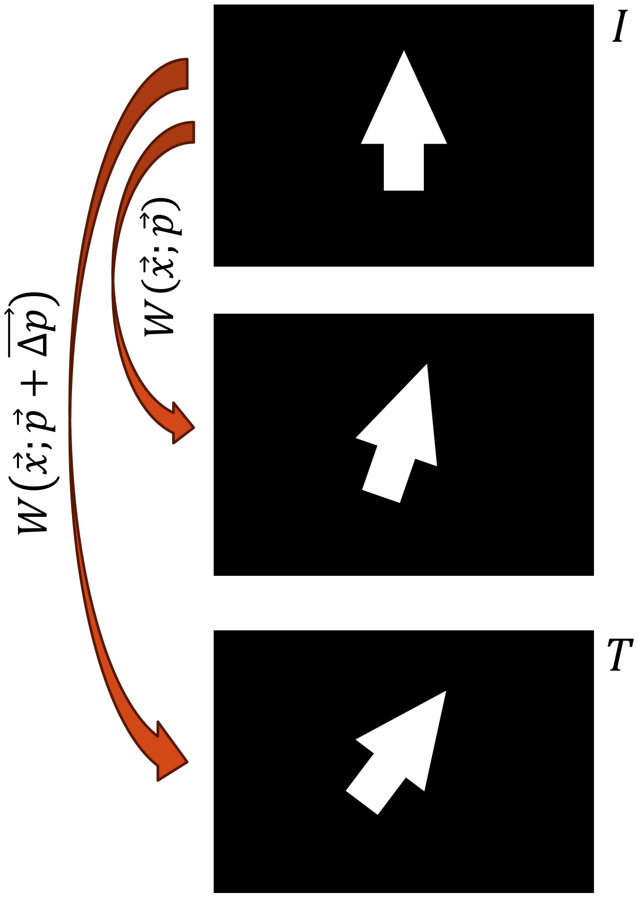
$$\overrightarrow{\Delta p} = H^{-1} \sum_{\vec{x}} \left[\nabla I \frac{\partial W}{\partial \vec{p}} \right]^T \left[T(\vec{x}) - I\big(W(\vec{x};\vec{p})\big) \right]$$

con $H = \sum_{\vec{x}} \left[\nabla I \frac{\partial W}{\partial \vec{p}} \right]^T \left[\nabla I \frac{\partial W}{\partial \vec{p}} \right] = A^T A$

Computer Vision Lectures: http://www.cs.cmu.edu/~16385/

ALGORITMO LK – ALINEAMIENTO ADITIVO

1. Transf. la imagen $I(W(\vec{x};\vec{p}))$
2. Calcular el error
$$E(\vec{x}) = \left[T(\vec{x}) - I\big(W(\vec{x};\vec{p})\big)\right]$$
3. Calcular el gradiente
$$\nabla I(\vec{x}'), \vec{x}' = W(\vec{x};\vec{p})$$
4. Evaluar el Jacobiano $\frac{\partial W}{\partial \vec{p}}$
5. Calcular el Hessiano
$$H = \sum_{\vec{x}} \left[\nabla I \frac{\partial W}{\partial \vec{p}}\right]^T \left[\nabla I \frac{\partial W}{\partial \vec{p}}\right]$$
6. Calcular la solución aprox.
$$\overrightarrow{\Delta p} = H^{-1} \sum_{\vec{x}} \left[\nabla I \frac{\partial W}{\partial \vec{p}}\right]^T E(\vec{x})$$
7. Actualizar los parámetros $\vec{p} \leftarrow \vec{p} + \overrightarrow{\Delta p}$



Computer Vision Lectures: http://www.cs.cmu.edu/~16385/

ALINEAMIENTO POR COMPOSICIÓN (SHUN-SZELISKY)

- El problema lo podemos escribir también de esta forma

$$\min_{\overrightarrow{\Delta p}} \sum_{\vec{x}} \left[I\left(W\left(W\left(\vec{x}; \overrightarrow{\Delta p}\right); \vec{p} \right) \right) - T(\vec{x}) \right]^2 \tag{4}$$

- Al linealizar nos queda como

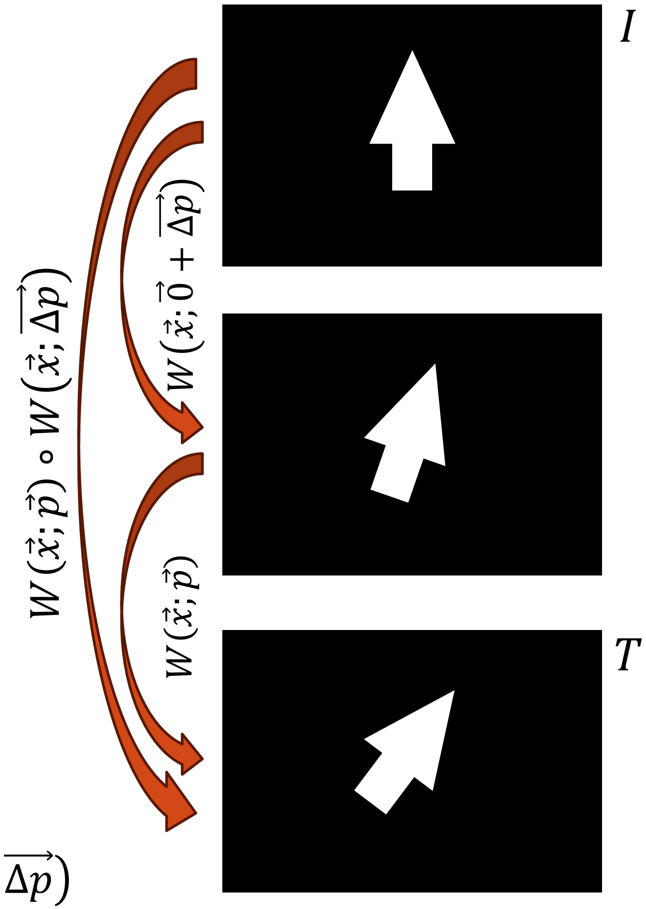
$$\sum_{\vec{x}} \left[I\left(W(\vec{x};\vec{p})\right) + \nabla I(\vec{x}') \frac{\partial W\left(\vec{x};\vec{0}\right)}{\partial \vec{p}} \overrightarrow{\Delta p} - T(\vec{x}) \right]^2$$

- Aquí el Jacobiano se calcula una sola vez

Computer Vision Lectures: http://www.cs.cmu.edu/~16385/

ALGORITMO SS – ALINEAMIENTO POR COMPOSICIÓN

1. Transf. la imagen $I(W(\vec{x};\vec{p}))$
2. Calcular el error
$$E(\vec{x}) = \left[T(\vec{x}) - I\big(W(\vec{x};\vec{p})\big)\right]$$
3. Calcular el gradiente
$$\nabla I(\vec{x}'), \vec{x}' = W(\vec{x};\vec{p})$$
4. Evaluar el Jacobiano $\frac{\partial W(\vec{x};\vec{0})}{\partial \vec{p}}$ (Precalculado)
5. Calcular el Hessiano
$$H = \sum_{\vec{x}} \left[\nabla I \frac{\partial W}{\partial \vec{p}}\right]^T \left[\nabla I \frac{\partial W}{\partial \vec{p}}\right]$$
6. Calcular la solución aprox.
$$\overrightarrow{\Delta p} = H^{-1} \sum_{\vec{x}} \left[\nabla I \frac{\partial W}{\partial \vec{p}}\right]^T E(\vec{x})$$
7. Actualizar los parámetros $W(\vec{x};\vec{p}) \leftarrow W(\vec{x};\vec{p}) \circ W\big(\vec{x};\overrightarrow{\Delta p}\big)$



Computer Vision Lectures: http://www.cs.cmu.edu/~16385/

ALINEAMIENTO POR COMPOSICIÓN INVERSA (BAKER-MATTHEWS)

- ¿Qué pasa si también se transforma la imagen de referencia T?
- Ahora el problema es

$$\min_{\overrightarrow{\Delta p}} \sum_{\vec{x}} \left[T\left(W\left(\vec{x}; \overrightarrow{\Delta p}\right)\right) - I\left(W\left(\vec{x}; \vec{p}\right)\right) \right]^2 \tag{5}$$

- Al linealizar nos queda como

$$\sum_{\vec{x}} \left[T\left(W\left(\vec{x}; \vec{0}\right)\right) + \nabla T \frac{\partial W}{\partial \vec{p}} \overrightarrow{\Delta p} - I\left(W\left(\vec{x}; \vec{p}\right)\right) \right]^2$$

- Solución aprox.

$$\underbrace{H = \sum_{\vec{x}} \left[\nabla T \frac{\partial W}{\partial \vec{p}} \right]^T \left[\nabla T \frac{\partial W}{\partial \vec{p}} \right]}_{\text{Precalculado}}, \overrightarrow{\Delta p} = H^{-1} \sum_{\vec{x}} \underbrace{\left[\nabla T \frac{\partial W}{\partial \vec{p}} \right]^T}_{\text{Precalculado}} \left[T(\vec{x}) - I\left(W\left(\vec{x}; \vec{p}\right)\right) \right]$$

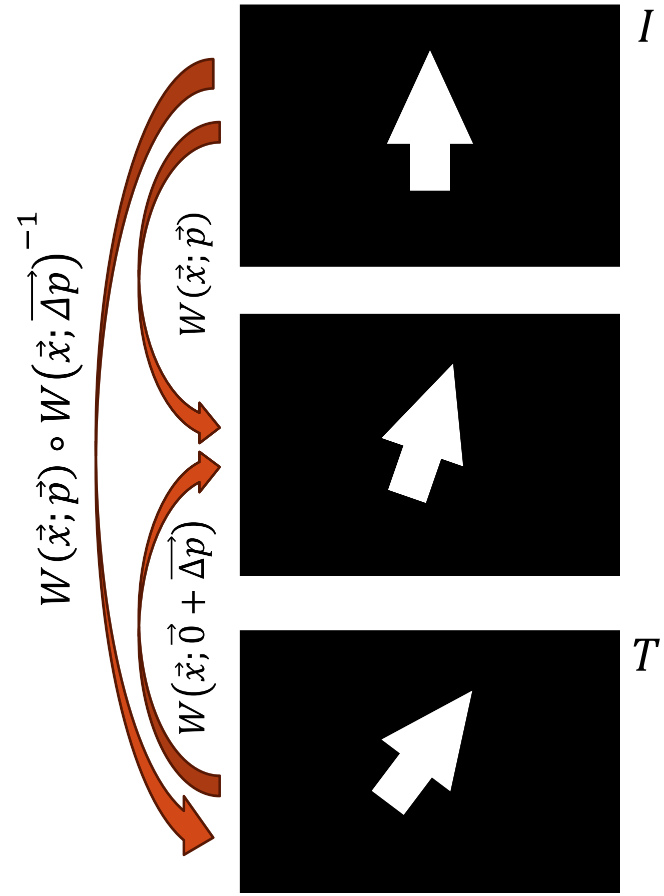
Computer Vision Lectures: http://www.cs.cmu.edu/~16385/

ALGORITMO BM – ALINEAMIENTO POR COMPOSICIÓN INVERSA

1. Transf. la imagen $I(W(\vec{x};\vec{p}))$
2. Calcular el error
$$E(\vec{x}) = [T(\vec{x}) - I(W(\vec{x};\vec{p}))]$$
3. Calcular el gradiente
$$\nabla T\left(W(\vec{x};\vec{0})\right)$$
4. Evaluar el Jacobiano $\frac{\partial W(\vec{x};\vec{0})}{\partial \vec{p}}$
5. Calcular el Hessiano
$$H = \sum_{\vec{x}} \left[\nabla T \frac{\partial W}{\partial \vec{p}}\right]^T \left[\nabla I \frac{\partial W}{\partial \vec{p}}\right]$$

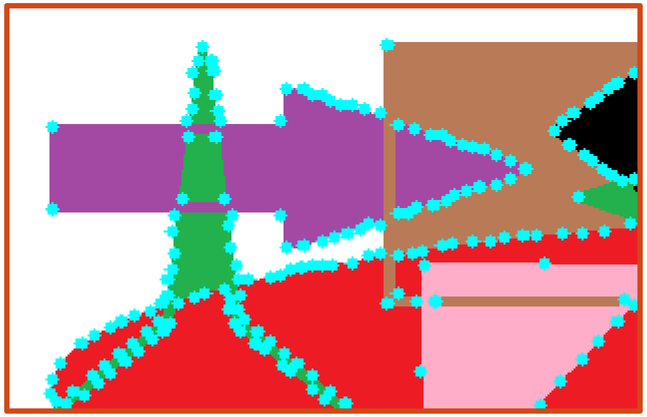
Precalculado

6. Calcular la solución aprox.
$$\overrightarrow{\Delta p} = H^{-1} \sum_{\vec{x}} \left[\nabla I \frac{\partial W}{\partial \vec{p}}\right]^T E(\vec{x})$$
7. Actualizar los parámetros $W(\vec{x};\vec{p}) \leftarrow W(\vec{x};\vec{p}) \circ \boldsymbol{W}(\vec{\boldsymbol{x}};\overrightarrow{\boldsymbol{\Delta p}})^{-1}$

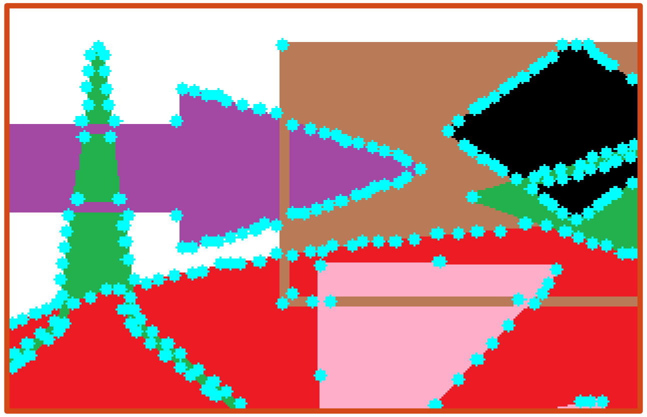


Computer Vision Lectures: http://www.cs.cmu.edu/~16385/

REGISTRO DE IMÁGENES USANDO PUNTOS DE INTERÉS



I_i

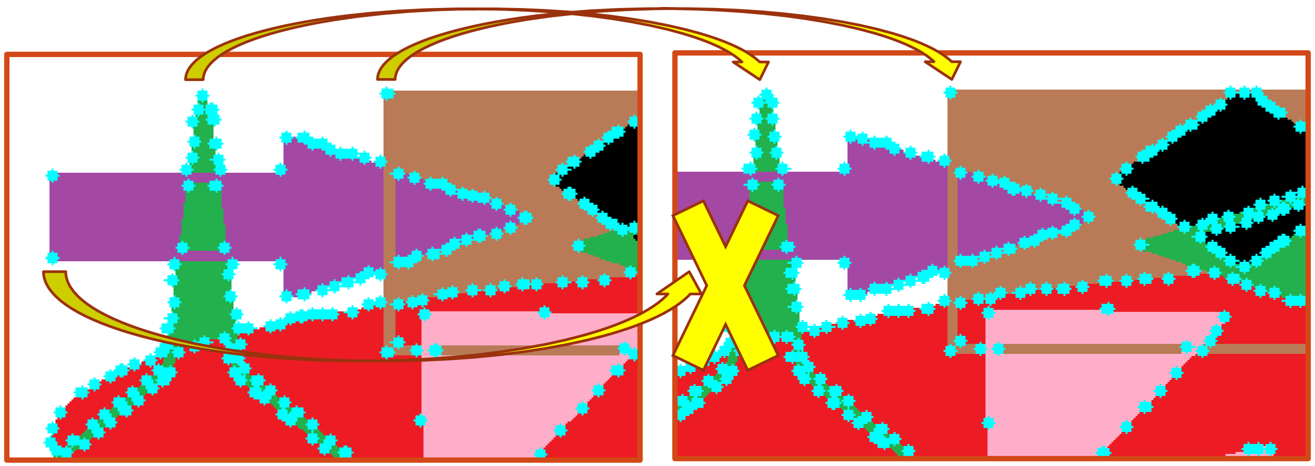


I_j

Detección de características en las imágenes

- goodFeaturesToTrack()
- SIFT()
- SURF(), etc.

REGISTRO DE IMÁGENES USANDO PUNTOS DE INTERÉS (C1)

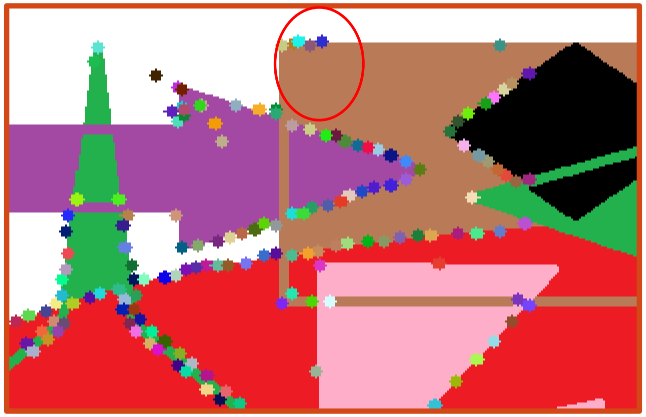


I_i I_j

Cálculo de correspondencias entre las características

REGISTRO DE IMÁGENES USANDO PUNTOS DE INTERÉS (C2)

I_i



I_j

Cálculo de correspondencias entre las características

- Flujo óptico (Lucas-Kanade Piramidal)

Distancia Euclidiana entre los descriptores de las características:

$$d\left(c_r^i, c_s^j\right) = \left\| c_r^i - c_s^j \right\|_2$$

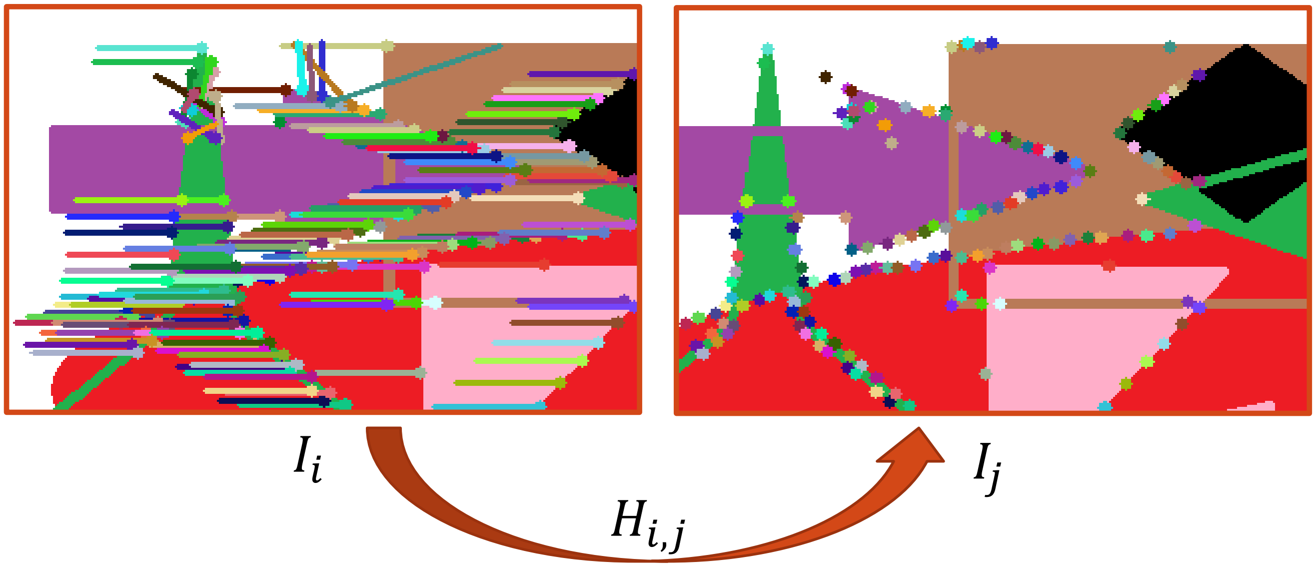
a) Devolver todos los vectores con $d < threshold$

b) Nearest Neighbor (NN): Vector con d más pequeña

c) Nearest Neighbor Distance Ratio (NNDR): Si $NNDR < threshold$

$NNDR = \frac{d_1}{d_2}$, con d_1 el más pequeño y d_2 el segundo más pequeño

REGISTRO DE IMÁGENES USANDO PUNTOS DE INTERÉS (C3)



Estimación de una Homografía $H_{i,j}$ a partir de las correspondencias

- $H_{i,j} \in \mathbb{R}^{3\times 3}$
- Sea $\boldsymbol{x} = (x_1, x_2)^T$ la posición de un pixel en coordenadas cartesianas y $\widehat{\boldsymbol{x}} = (\boldsymbol{x}^T, 1)^T$ su correspondiente punto en coordenadas homogéneas, entonces
$$\widehat{\boldsymbol{x}}^j = H_{i,j}\widehat{\boldsymbol{x}}^i$$
- Necesitamos por lo menos 4 correspondencias
- Podemos usar la técnica DLT (Direct Linear Transform)
- RANSAC (RANdom SAmple Consensus)

Hartley, A., Zisserman, A.: Multiple view geometry in computer vision (2 ed.). Cambridge University Press, Cambridge (2006)

CALCULAR LA HOMOGRAFÍA

- Tenemos que $\widehat{\boldsymbol{x}}^j = H_{i,j}\widehat{\boldsymbol{x}}^i$

$$\begin{pmatrix} x_1^j \\ x_2^j \\ 1 \end{pmatrix} = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} \begin{pmatrix} x_1^i \\ x_2^i \\ 1 \end{pmatrix}$$

- Entonces

$$x_1^j = \frac{h_{11}x_1^i + h_{12}x_2^i + h_{13}}{h_{31}x_1^i + h_{32}x_2^i + h_{33}},\ x_2^j = \frac{h_{21}x_1^i + h_{22}x_2^i + h_{23}}{h_{31}x_1^i + h_{32}x_2^i + h_{33}}$$

- Por cada correspondencia tenemos:

$$h_{11}x_1^i + h_{12}x_2^i + h_{13} - x_1^j\left(h_{31}x_1^i + h_{32}x_2^i + h_{33}\right) = 0$$

$$h_{21}x_1^i + h_{22}x_2^i + h_{23} - x_2^j\left(h_{31}x_1^i + h_{32}x_2^i + h_{33}\right) = 0$$

CALCULAR LA HOMOGRAFÍA (C1)

- Por DLT:

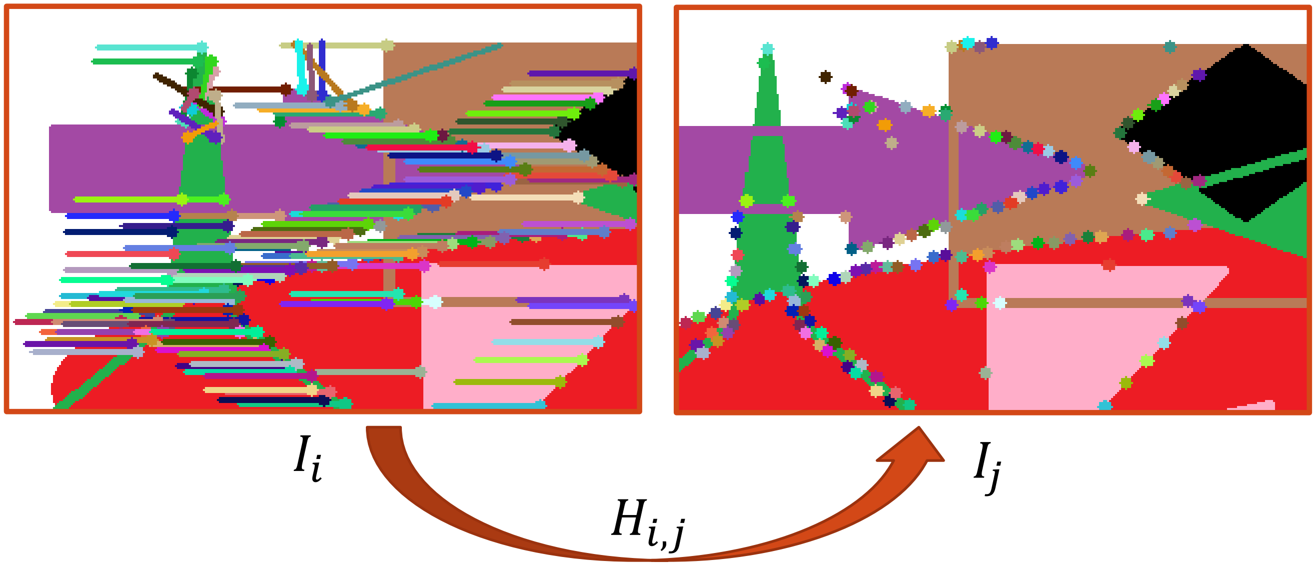
$$\begin{pmatrix} x_1^i & x_2^i & 1 & 0 & 0 & 0 & -x_1^j x_1^i & -x_1^j x_2^i & -x_1^j \\ 0 & 0 & 0 & x_1^i & x_2^i & 1 & -x_2^j x_1^i & -x_2^j x_2^i & -x_2^j \\ \vdots & & & & \vdots & & & & \vdots \end{pmatrix} \begin{pmatrix} h_{11} \\ h_{12} \\ \vdots \\ h_{33} \end{pmatrix} = \mathbf{0}$$

$$\boldsymbol{A}_{2M\times 9}\boldsymbol{h}_9 = \mathbf{0}$$

- Resolver el sistema aplicando SVD
  - >> [U, S, V] = svd(A);
  - >> h = V(:,último); %Si los valores propios estan ordenados de mayor a menor

Hartley, A., Zisserman, A.: Multiple view geometry in computer vision (2 ed.). Cambridge University Press, Cambridge (2006)

CALCULAR LA HOMOGRAFÍA (C2)



Para este par de imágenes se sabe que solo hay un corrimiento de 50 pixeles hacia la derecha, entonces:

$$H_{i,j} = \begin{pmatrix} 1 & 0 & 50 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Resultado computacional:

$$H_{i,j} = \begin{pmatrix} 0.999954 & 0.000025 & 50.001379 \\ 0.000030 & 0.999853 & 0.003057 \\ -0.000000 & 0.000000 & 1.000000 \end{pmatrix}$$

GRACIAS POR SU ATENCIÓN

Francisco J. Hernandez-Lopez

fcoj23@cimat.mx

WebPage:

www.cimat.mx/~fcoj23