

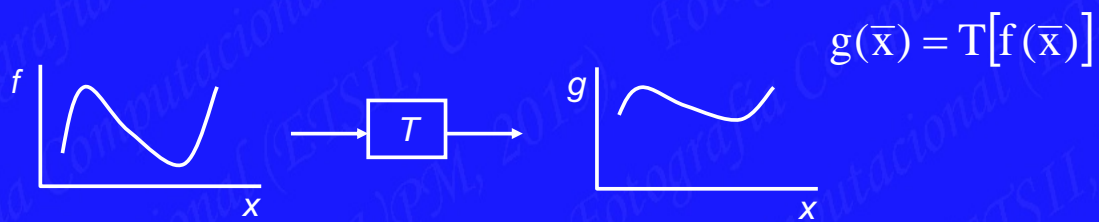
Deformación de imágenes (warping)



Algunas transparencias inspiradas en Derek Hoiem, Alyosha Efros + Steve Seitz

Transformaciones de rango (previo)

Cambiamos el valor de los píxeles pero éstos siguen estando asociados a la misma posición (x,y) original.



Original $im(\bar{x})$

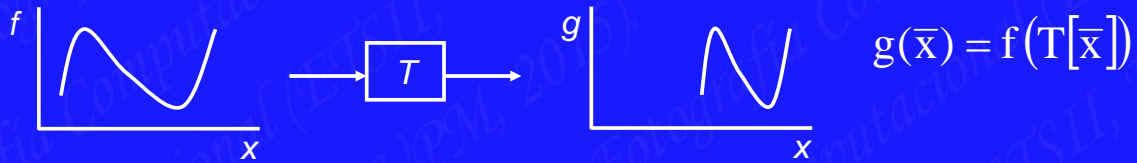


$im'(\bar{x}) = T[im(\bar{x})]$



Transformaciones de dominio (hoy)

Transformaciones de dominio: el valor del píxel no cambia pero cambiamos su posición de (x,y) a una nueva (x',y') .



Original $\text{im}(\bar{x})$



$\text{im}'(\bar{x}) = \text{im}(T[\bar{x}])$



Temas a tratar

Transformaciones de coordenadas:

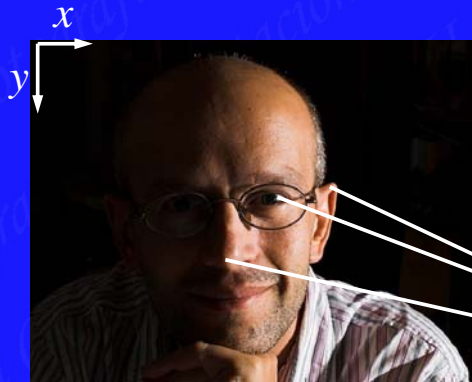
- Aplicación de la transformación: interpolación.
- Transformaciones globales y locales
- Determinar transformación usando puntos de control.

Aplicaciones:

- Registro de imágenes previo a su combinación.
- Mosaicos de fotos.
- Morphing

$T(\underline{x})$ puede ser cualquier cosa

$T(\underline{x})$ define una relación entre las coordenadas $T(x,y) = (x',y')$.



$$x' = y \cdot \cos(x)$$

$$y' = y \cdot \sin(x)$$



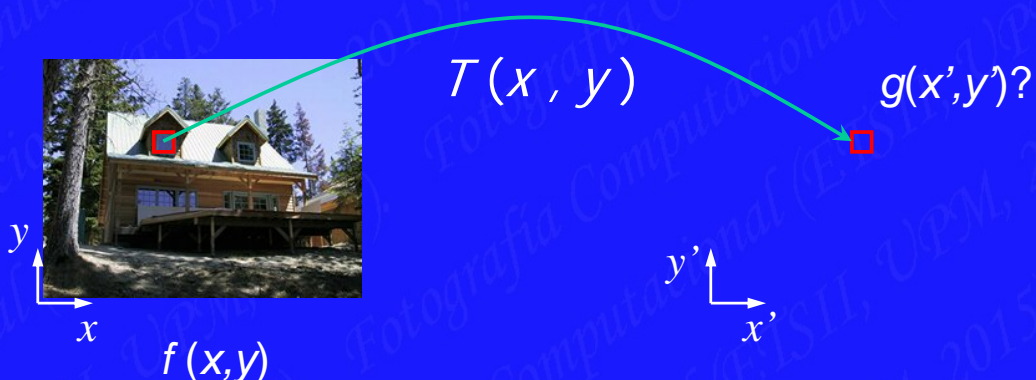
Implementación: cuestiones prácticas

Dada una transformación de coordenadas $(x',y')=T(x,y)$:

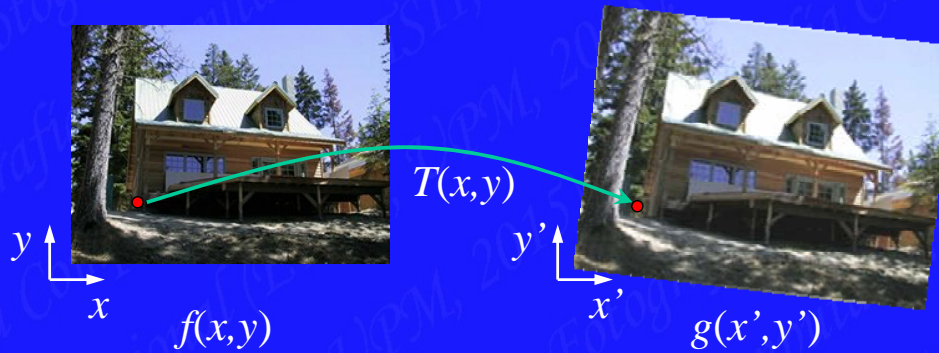
$$T(x,y) = \begin{cases} x' = 2x - 3y + 1 \\ y' = x + y + 10 \end{cases} \quad \begin{pmatrix} x = 5 \\ y = 1 \end{pmatrix} \rightarrow \begin{pmatrix} x' = 8 \\ y' = 16 \end{pmatrix}$$

Conocemos la imagen de partida $f(x,y)$.

¿Cómo calculamos la imagen final: $g(x',y') = f(T(x,y))$?



Warping directo (forward)

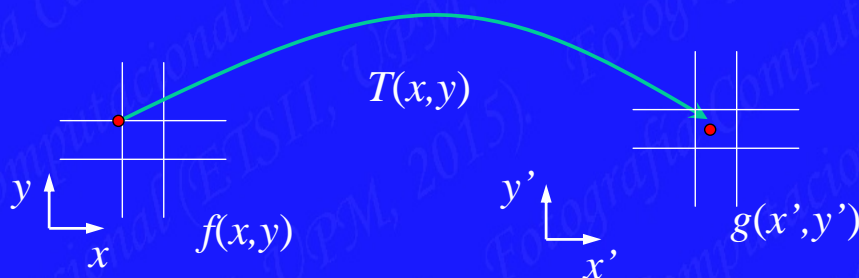


- Bucle barriendo la imagen de partida en (x, y) :
 - Calcula la nueva posición $(x', y') = T(x, y)$
 - Coloca el valor del pixel $f(x, y)$ en la nueva posición:

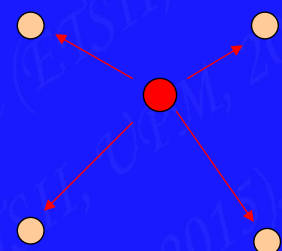
$$g(x', y') = f(x, y)$$

Warping directo

Problema: es muy posible que (x', y') no sean enteros.



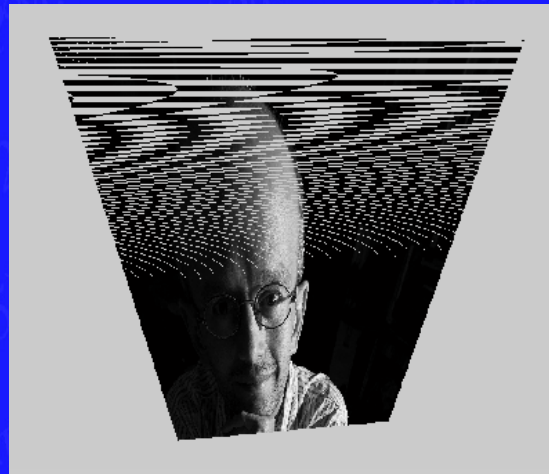
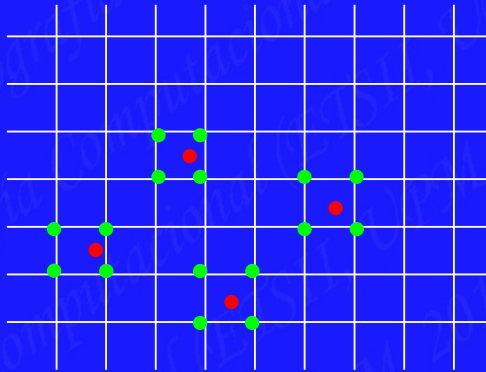
¿Qué hacemos cuando un un píxel cae entre líneas?
Esparcir su "color" entre sus vecinos (splatting)



¿Problemas de warping directo?

Factor multiplicativo debido a las diferentes "áreas" de las imágenes origen y destino.

Posibilidad de agujeros en la imagen de destino



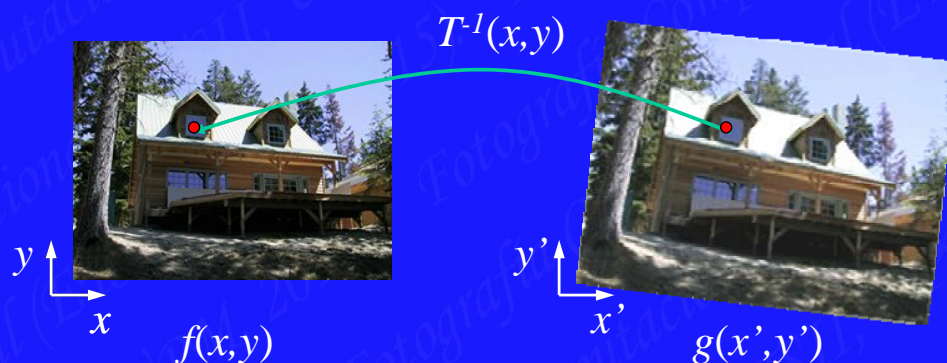
Warping inverso

Barremos la imagen destino en sus coordenadas (x', y') .

Aplicamos T **inversa** obteniendo coordenadas de origen (x, y)

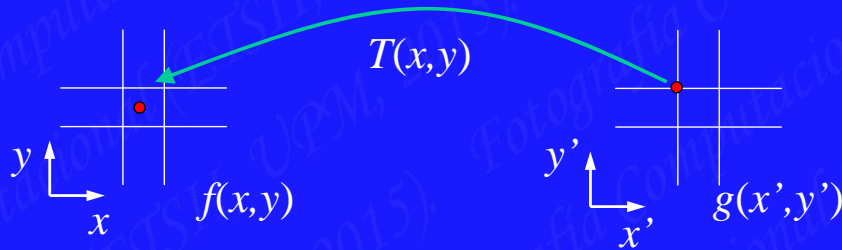
Usamos $f(x,y)$ para rellenar el píxel $g(x',y')$

Bucle en (x',y') : calculo $(x,y) = T^{-1}(x',y') \Rightarrow g(x',y') = f(x,y)$



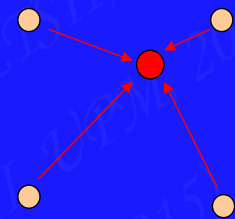
Warping inverso

- Evitamos el problema de "agujeros" en imagen destino.
- Sigue el problema de que las coordenadas obtenidas sobre la imagen original $(x, y) = T^{-1}(x', y')$ no serán valores enteros.

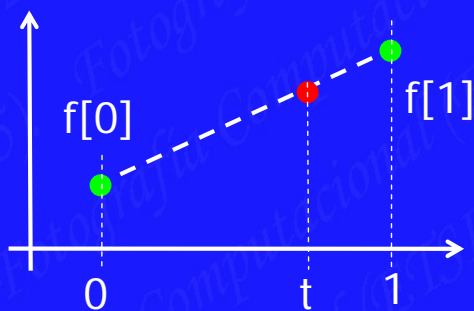


¿Qué hacer si un píxel viene de "entre" varios?
Interpolar a partir de sus vecinos:

MATLAB: `im2 = interp2(im, x, y, metodo);`
`metodo = 'nearest', 'bilinear', 'spline'`



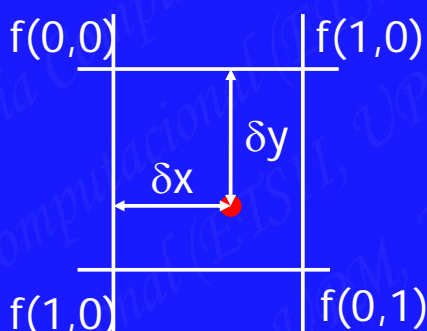
Interpolación bilineal



Interpolación lineal: caso 1D

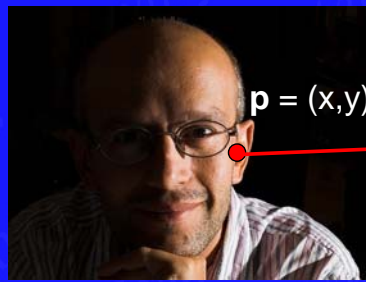
$$f(t) = (1-t) \cdot f[0] + t \cdot f[1]$$

Caso 2D: interpolación bilineal



$$f(x, y) = (1-\delta x)(1-\delta y) f(0,0) + (1-\delta x)(\delta y) f(1,0) + (\delta x)(1-\delta y) f(1,1) + (\delta x)(\delta y) f(0,1)$$

Transformaciones globales paramétricas



$$p' = T(p)$$

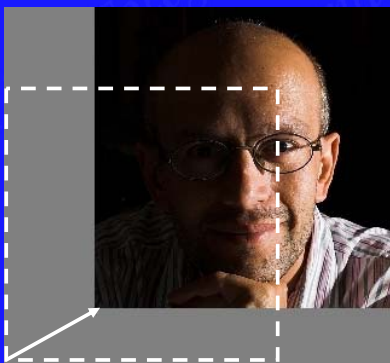


- La misma T se usa para transformar todos los puntos y la transformación se describe con unos pocos parámetros.

Caso especial: Transformaciones lineales, donde T consiste en la aplicación de una matriz M sobre las coordenadas.

$$\bar{p}' = \begin{bmatrix} x' \\ y' \end{bmatrix} = \mathbf{M} \cdot \begin{bmatrix} x \\ y \end{bmatrix} = \mathbf{M} \cdot \bar{p}$$

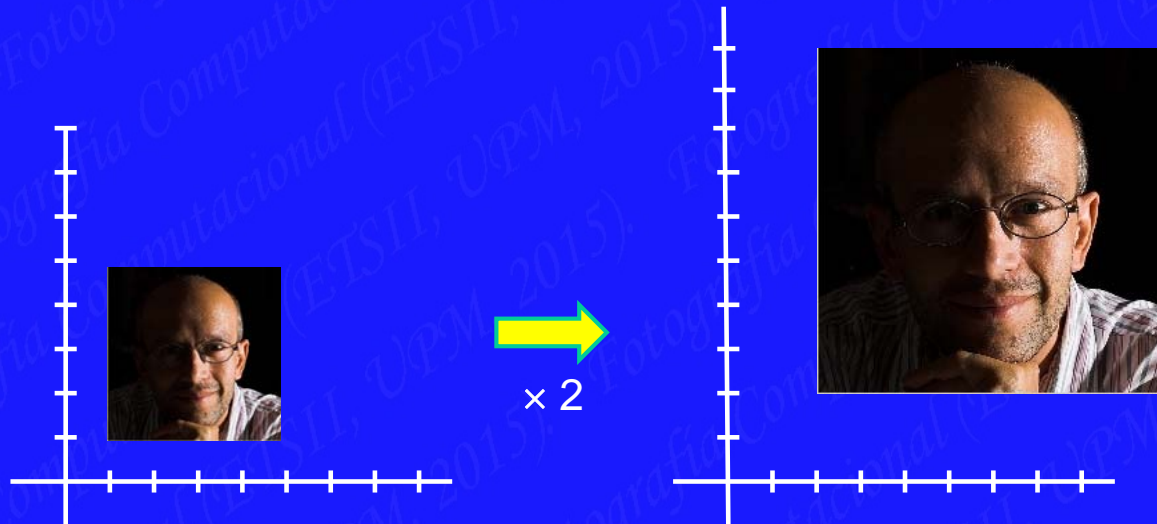
Ejemplos transformaciones lineales



Escalado

Multiplicar cada coordenada por un número (distinto?)

Escalado uniforme = mismo factor en todos los ejes

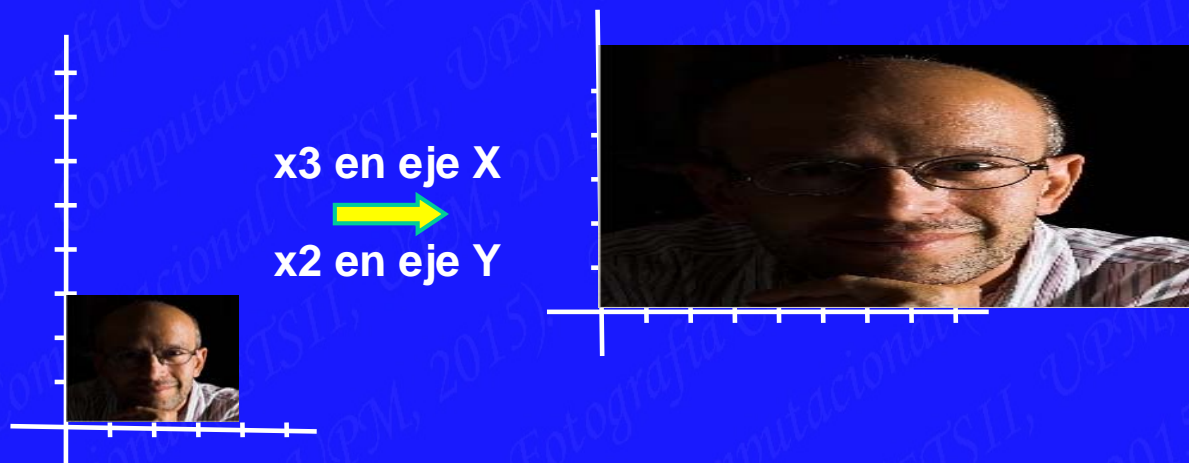


Escalado no uniforme

Diferentes escalas en cada eje.

Supone cambiar la relación de aspecto o factor de forma.

Relación de aspecto = cociente ancho/alto de la imagen



Matrices de transformación de escala

$$x' = ax$$

$$y' = by$$

En forma matricial:

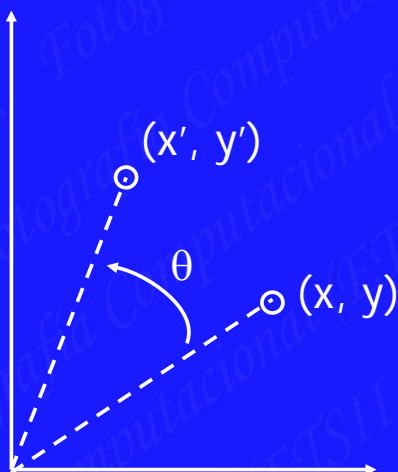
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \underbrace{\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}} \begin{bmatrix} x \\ y \end{bmatrix}$$

Matriz de escalado S (diagonal)

La matriz inversa deshace operación:

$$S^{-1} = \begin{bmatrix} a^{-1} & 0 \\ 0 & b^{-1} \end{bmatrix}$$

Rotación 2D alrededor del origen



$$x' = x \cos(\theta) - y \sin(\theta)$$

$$y' = x \sin(\theta) + y \cos(\theta)$$



R

En forma matricial:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \overbrace{\begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}} \cdot \begin{pmatrix} x \\ y \end{pmatrix}$$

Otras Transformaciones con matrices 2x2

Reflejo sobre uno de los ejes o sobre el origen

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \quad \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$



ANTONIO TABERNERO GALÁN, 2019

FOTOGRAFÍA COMPUTACIONAL, ETSII, UPM

Otras Transformaciones con matrices 2x2

Cizalladura $x' = x + c_x \cdot y$
 $y' = c_y \cdot x + y$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & c_x \\ c_y & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$



$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & -0.1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$



ANTONIO TABERNERO GALÁN, 2019

FOTOGRAFÍA COMPUTACIONAL, ETSII, UPM

Caso General

Todas las transformaciones lineales son combinación de:
Escalados, Rotaciones, Cizalladuras y Reflejos.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} 1 & c_x \\ c_y & 1 \end{bmatrix} \begin{bmatrix} r_x & 0 \\ 0 & -r_x \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Propiedades:

El origen (0,0) se queda donde estaba.

Rectas se mantienen como rectas.

Las líneas paralelas lo siguen siendo.

Su combinación sigue siendo una transformación lineal

¿Y qué pasa con las translaciones?

No van a funcionar con nuestro esquema $p' = M \cdot p$

Una translación desplaza el origen.

La transformación lineal (matriz 2x2) no puede hacerlo

$$\begin{aligned} x' &= x + t_x \\ y' &= y + t_y \end{aligned} \quad \begin{pmatrix} x' \\ y' \end{pmatrix} \neq M \begin{pmatrix} x \\ y \end{pmatrix}$$

Coordenadas homogéneas

La translación es una operación muy importante y la representación de transformaciones usando matrices nos resulta muy conveniente.

Objetivo:

¿ cómo representar translaciones con matrices ?

$$x' = x + t_x$$

$$y' = y + t_y$$

Coordenadas homogéneas

- Representamos coordenadas 2D con vectores 3D:

$$\begin{array}{l} \text{Coordenadas} \\ \text{2D de partida} \end{array} \begin{pmatrix} x \\ y \end{pmatrix} \longrightarrow \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \begin{array}{l} \text{Coordenadas} \\ \text{homogéneas} \\ \text{(aumentadas)} \end{array}$$

- Operamos con ellas (aplicando matrices transformación 3x3)
- Al terminar volvemos a 2D (dividiendo por 3ª componente):

$$\begin{pmatrix} X \\ Y \\ W \end{pmatrix} \longrightarrow \begin{pmatrix} X/W \\ Y/W \\ 1 \end{pmatrix} \longrightarrow \begin{cases} x' = \frac{X}{W} \\ y' = \frac{Y}{W} \end{cases} \begin{array}{l} \text{Vuelta a} \\ \text{coordenadas 2D} \end{array}$$

Traducción como matriz

¿Cómo representar T como una matriz? $x' = x + t_x$

$$y' = y + t_y$$

$$\mathbf{T} = \begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix}$$

Entrada Salida

↓ ↓

$$\begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} x + t_x \\ y + t_y \\ 1 \end{pmatrix} \Rightarrow \begin{cases} x' = (x + t_x)/1 \\ y' = (y + t_y)/1 \end{cases} \Rightarrow \begin{cases} x' = x + t_x \\ y' = y + t_y \end{cases}$$

Operaciones anteriores con matrices 3x3

Desplazamiento

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Escala

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Rotación

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Genérica

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & 0 \\ c & d & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Composición de operadores

¿Aplicar Escalado + Giro + Desplazamiento ?

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\mathbf{p}' = \mathbf{T}(t_x, t_y) \mathbf{R}(\theta) \mathbf{S}(s_x, s_y) \mathbf{p}$$

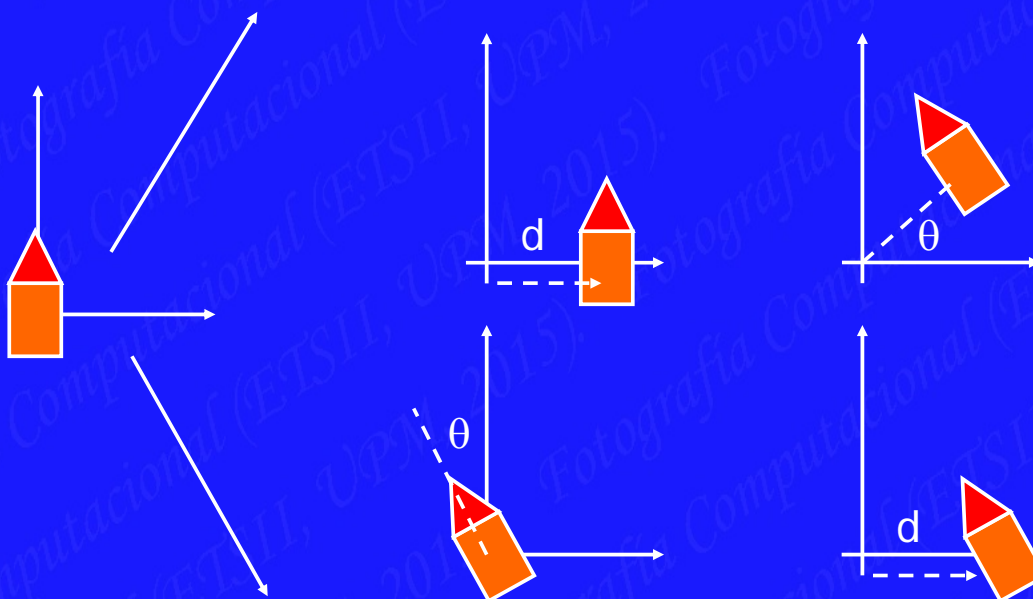
$$\mathbf{p}' = \mathbf{T} \cdot \mathbf{R} \cdot \mathbf{S} \cdot \mathbf{p} = \mathbf{Q} \cdot \mathbf{p}$$

Operador compuesto $\mathbf{Q} = \mathbf{T} \cdot \mathbf{R} \cdot \mathbf{S}$, producto de matrices

¿Importa el orden de las operaciones?

El producto de matrices no es conmutativo

Desplazamiento seguido de un giro: $\mathbf{Q} = \mathbf{R} \cdot \mathbf{T}$



Giro seguido de un desplazamiento $\mathbf{Q} = \mathbf{T} \cdot \mathbf{R}$

Transformaciones afines

Combinación de Transformaciones lineales + Desplazamiento:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & e \\ c & d & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$



Propiedades:

El origen no tiene por qué conservarse

Las rectas siguen siendo rectas

Las rectas paralelas siguen siendo paralelas.

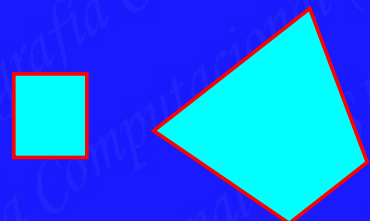
La composición de operadores afines sigue siendo afín

¿Cuándo saldrá algo distinto de 1 en la tercera coordenada w?

Transformaciones proyectivas

¡¡ Por fin !! \longrightarrow

$$\begin{bmatrix} x' \\ y' \\ w \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$



Combinación de transformaciones afines + proyecciones.

También llamadas homografías

Origen no tiene por qué mantenerse (traslaciones)

Rectas siguen siendo rectas, pero las paralelas no se mantienen.

Combinación de transformaciones sigue siendo proyección.

La matriz de proyección tiene 8 grados de libertad.

Transformaciones proyectivas (8 DOF)

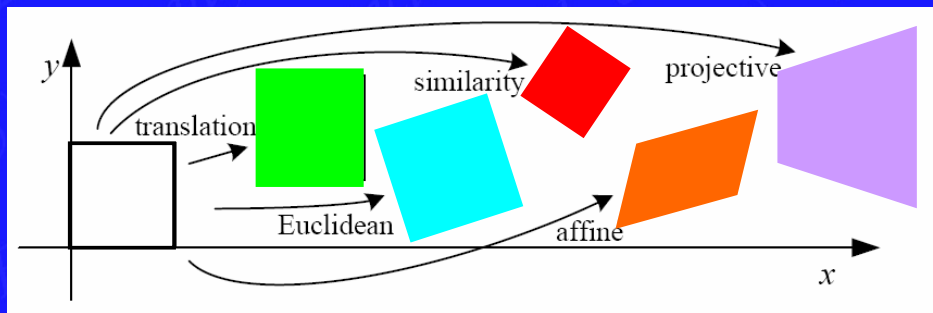
¿Por qué hemos desperdiciado un parámetro?

$$\begin{bmatrix} X \\ Y \\ w \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \longrightarrow \begin{cases} x' = \frac{X}{w} \\ y' = \frac{Y}{w} \end{cases}$$

$$\begin{bmatrix} k \cdot X \\ k \cdot Y \\ k \cdot w \end{bmatrix} = \begin{bmatrix} ka & kb & kc \\ kd & ke & kf \\ kg & kh & k \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \longrightarrow \begin{cases} x' = \frac{kX}{kw} = \frac{X}{w} \\ y' = \frac{kY}{kw} = \frac{Y}{w} \end{cases}$$

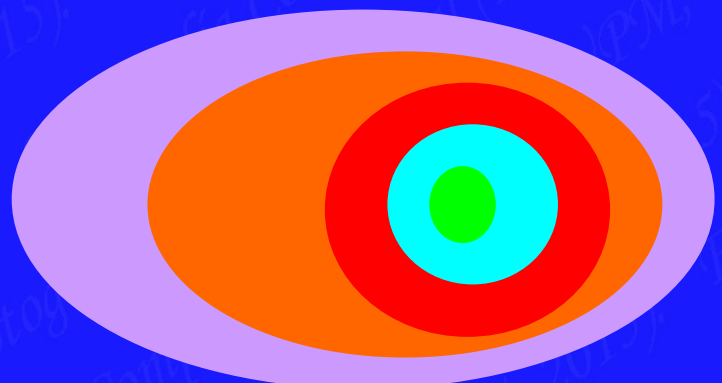
Las coordenadas finales 2D (que es lo que nos interesa) son las mismas en ambos casos: H y $k \cdot H$ son equivalentes.

Transformaciones 2D de imágenes



Cada una de las familias engloba a las anteriores

Sus inversas están dentro de la familia respectiva.



Registro de Imágenes

Muchas veces no nos dan explícitamente la transformación T .

Datos: 2 imágenes, $img1$ / $img2$

Problema: deducir la transformación que existe entre ellas.

$img1$



¿ $T(x,y)$?



$img2$

Conocidas $f(x,y)$ y $g(x',y')$ hallar transformación T tal que

$$g(\bar{x}) = f(T[\bar{x}])$$

Similar a los problemas que resolvimos en el paso de 3D a 2D

Puntos de Control

- Usaremos **puntos de control** para hallar los parámetros de la transformación T = los coeficientes de la matriz M
- Los puntos de control son una serie de puntos $(x,y) + (x',y')$ emparejados en ambas imágenes



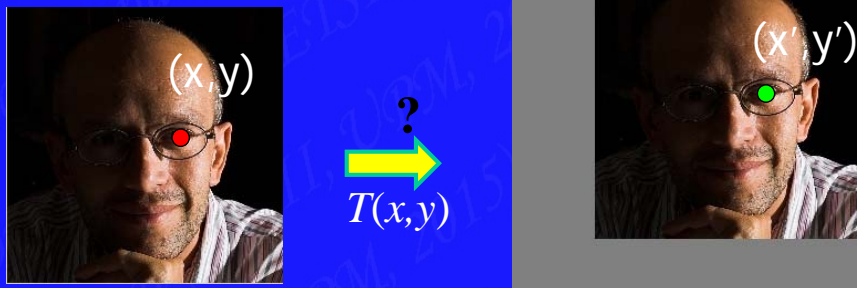
M ?



¿Cómo hallarlos?

¿Cuántos necesitamos?

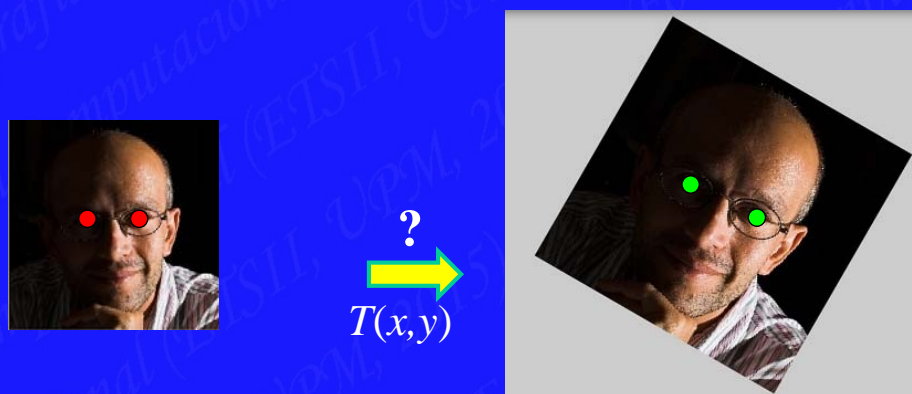
Traducción Pura



Grados de libertad (DOF) = 2 coeficientes a determinar
Nos basta con 1 solo punto = 2 ecuaciones.

$$\begin{aligned} x' &= x + t_x \\ y' &= y + t_y \end{aligned} \quad \Rightarrow \quad \begin{aligned} t_x &= x' - x \\ t_y &= y' - y \end{aligned} \quad \Rightarrow \quad \mathbf{M} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

Transformación de similitud



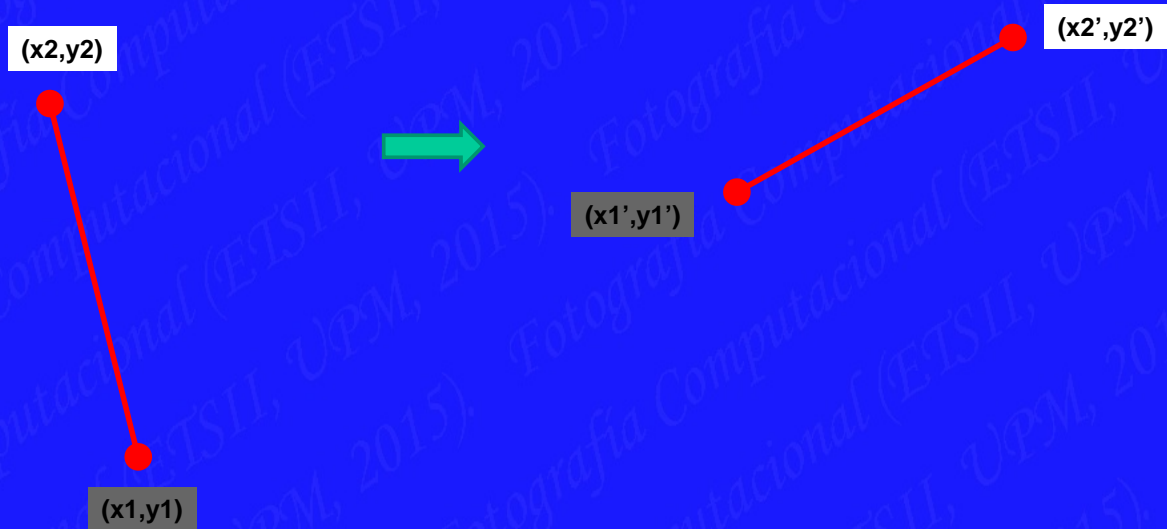
Giro + translación + escalado uniforme

¿Grados de libertad? = Parámetros a determinar: (t_x, t_y, s, θ)

Necesitamos 4 ecuaciones = 2 puntos de control

T similaridad

Una transformación de similaridad permite convertir cualquier segmento de img_1 en cualquier otro segmento en img_2 .



Solución

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s \cdot \cos \theta & -s \cdot \sin \theta & t_x \\ s \cdot \sin \theta & s \cdot \cos \theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$x' = ax - by + t_x$$

$$y' = bx + ay + t_y$$

$$a = s \cos \theta \quad b = s \sin \theta$$

$$x_1' = ax_1 - by_1 + tx$$

$$x_2' = ax_2 - by_2 + tx$$

$$y_1' = bx_1 + ay_1 + ty$$

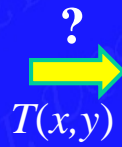
$$y_2' = bx_2 + ay_2 + ty$$

$$\begin{pmatrix} x_1' \\ x_2' \\ y_1' \\ y_2' \end{pmatrix} = \begin{pmatrix} x_1 & -y_1 & 1 & 0 \\ x_2 & -y_2 & 1 & 0 \\ y_1 & x_1 & 0 & 1 \\ y_2 & x_2 & 0 & 1 \end{pmatrix} \begin{pmatrix} a \\ b \\ t_x \\ t_y \end{pmatrix}$$

$$s^2 = a^2 + b^2$$

$$\theta = \tan^{-1}(b, a)$$

Transformación Afín

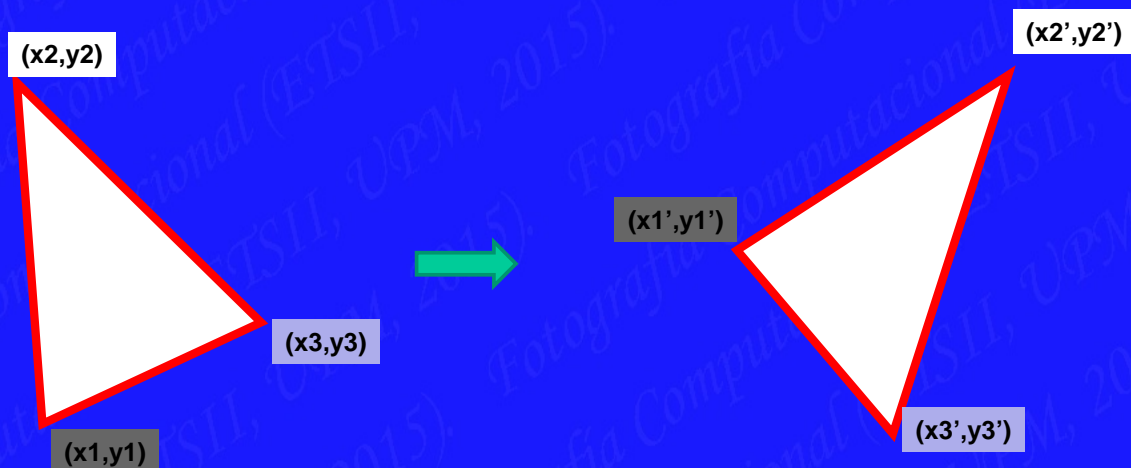


Grados de libertad = ¿ parámetros a determinar ?
¿Cuántas ecuaciones? ¿Cuántos puntos necesitamos?

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & e \\ c & d & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Transformación Afín (3 ptos)

Permite transformar un triángulo en cualquier otro.



Transformación afín

Conocemos tres puntos (x_i, y_i) en img1 y los correspondientes tres puntos (x'_i, y'_i) en img2

Determinar sistema lineal a resolver, con incógnitas:
(a, b, c, d, e, f)

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Sistema lineal a resolver

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$x' = ax + by + c$$

$$y' = dx + ey + f$$

$$\begin{array}{l} x'_1 = ax_1 + by_1 + c \\ y'_1 = dx_1 + ey_1 + f \\ x'_2 = ax_2 + by_2 + c \\ y'_2 = dx_2 + ey_2 + f \\ x'_3 = ax_3 + by_3 + c \\ y'_3 = dx_3 + ey_3 + f \end{array} \quad \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x \\ 0 \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} y \\ 0 \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ x \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ y \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \\ e \\ f \end{pmatrix}$$

Separabilidad de T afín

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x & y & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 \\ x & y & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \\ e \\ f \end{pmatrix}$$

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x & y & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \end{pmatrix} = H \cdot \begin{pmatrix} a \\ b \\ c \end{pmatrix} \quad \begin{pmatrix} y' \end{pmatrix} = H \cdot \begin{pmatrix} d \\ e \\ f \end{pmatrix}$$

Ajuste a una Transformación afín

Si hay más puntos de los 3 necesarios → problema de ajuste

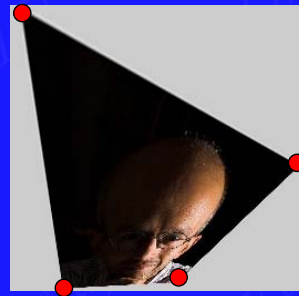
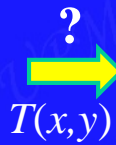
Ahora la matriz H es más alta que ancha, al tener los vectores x, y, x', y' más de 3 componentes:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x & y & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \end{pmatrix} = H \cdot \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$

La resolución en MATLAB es idéntica: $\text{coefs} = H \setminus x' = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$

Transformación proyectiva

4 pts



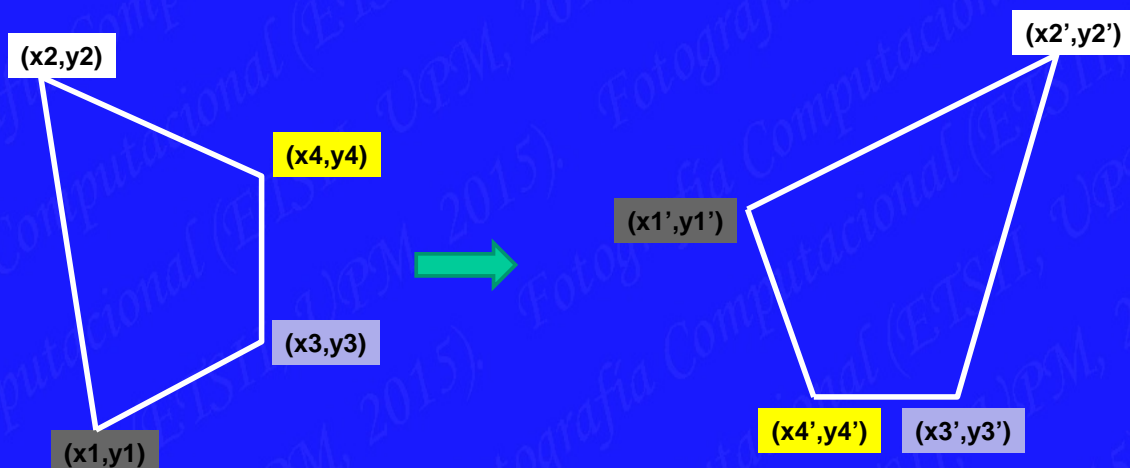
Transformación con 8 parámetros:

Las 8 incógnitas son los elementos de $P = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix}$

Necesitamos 4 puntos (x_i, y_i) en img1 y otros 4 (x'_i, y'_i) en img2

Transformación proyectiva (4 puntos)

8 incógnitas \rightarrow 8 ecuaciones = 4 parejas de puntos.
Permite transformar cualquier cuadrilátero en otro.



Solución T proyectiva

Dados $\{x,y\} + \{x',y'\}$, hacer que se verifique:

$$\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Reorganizar la matriz P como un vector de incógnitas

$$P = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \Rightarrow \bar{h} = \begin{pmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ \dots \\ h_{32} \\ 1 \end{pmatrix}$$

y plantear un sistema lineal para resolver \bar{h} .

Solución T proyectiva

Planteando las ecuaciones a cumplir, el vector h debe verificar

$$M \cdot \bar{h} = \bar{0}$$

donde M es una matriz 8 x 9 construida a partir de los datos de los puntos de control: $\{x,y\} + \{x',y'\}$:

$$M = \begin{pmatrix} \begin{pmatrix} A \\ Z \end{pmatrix} & \begin{pmatrix} Z \\ A \end{pmatrix} & \begin{pmatrix} B \\ C \end{pmatrix} \end{pmatrix}$$

Solución T proyectiva

Las submatrices Z, A, B, C (4 x 3) se construyen:

$$Z = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \quad A = \begin{pmatrix} \left(\begin{smallmatrix} \bar{x} \\ \bar{y} \\ \bar{1} \end{smallmatrix} \right), \left(\begin{smallmatrix} \bar{x}' \\ \bar{y}' \\ \bar{1} \end{smallmatrix} \right), \left(\begin{smallmatrix} \bar{x}'' \\ \bar{y}'' \\ \bar{1} \end{smallmatrix} \right) \end{pmatrix}$$

$$b = \begin{pmatrix} \left(\begin{smallmatrix} \bar{x}' \\ \bar{y}' \\ \bar{1} \end{smallmatrix} \right), \left(\begin{smallmatrix} \bar{x}'' \\ \bar{y}'' \\ \bar{1} \end{smallmatrix} \right), \left(\begin{smallmatrix} \bar{x}''' \\ \bar{y}''' \\ \bar{1} \end{smallmatrix} \right) \end{pmatrix} \Rightarrow B = -A.*b \quad c = \begin{pmatrix} \left(\begin{smallmatrix} \bar{y}' \\ \bar{y}'' \\ \bar{y}''' \end{smallmatrix} \right), \left(\begin{smallmatrix} \bar{x}' \\ \bar{x}'' \\ \bar{x}''' \end{smallmatrix} \right), \left(\begin{smallmatrix} \bar{1}' \\ \bar{1}'' \\ \bar{1}''' \end{smallmatrix} \right) \end{pmatrix} \Rightarrow C = -A.*c$$

¿Solución de $M \cdot h = 0$?

Autovector de la matriz $Q = (M' \cdot M)$ con el menor autovalor

En MATLAB:

```
Q = (M'*M);  
[V D] = eig(Q); % V = autovectores de Q (columnas)  
d = diag(D) ; % d = diagonal de D = autovalores.  
[m, idx]=min(d); % Busca autovalor mínimo.  
h = V(:,idx); % Autovector correspondiente  
  
P = reshape(h,3,3)'; % Reorganiza h como matriz P 3x3  
P = P/P(3,3); % Normaliza para que P(3,3)=1.
```

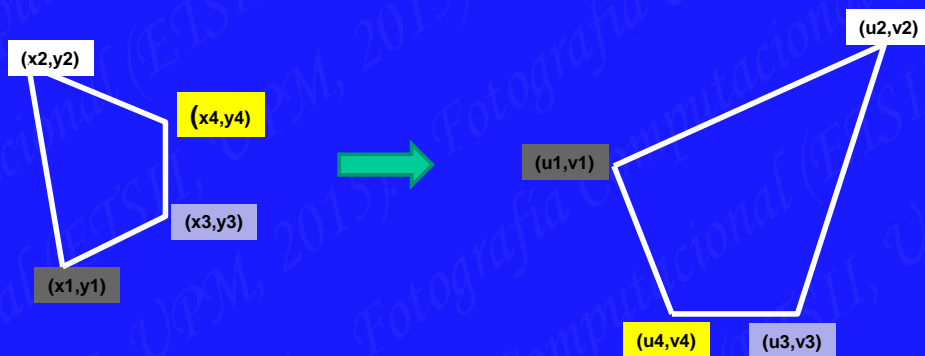
Transformaciones no lineales

No estamos limitados a transformaciones lineales (matrices)
Ejemplo sencillo de una transformación (no lineal):

$$\begin{aligned}u &= A + B \cdot x + C \cdot y + D \cdot x \cdot y \\v &= a + b \cdot x + c \cdot y + d \cdot x \cdot y\end{aligned}$$

→ términos no lineal

8 parámetros → determinada con 4 parejas de puntos



Transformaciones no lineales

Determinación de los parámetros (A,B,C,D) + (a,b,c,d)

$$u_1 = A + Bx_1 + Cy_1 + D \cdot x_1y_1$$

$$v_1 = a + bx_1 + cy_1 + d \cdot x_1y_1$$

$$u_2 = A + Bx_2 + Cy_2 + D \cdot x_2y_2$$

$$v_2 = a + bx_2 + cy_2 + d \cdot x_2y_2$$

$$u_3 = A + Bx_3 + Cy_3 + D \cdot x_3y_3$$

$$v_3 = a + bx_3 + cy_3 + d \cdot x_3y_3$$

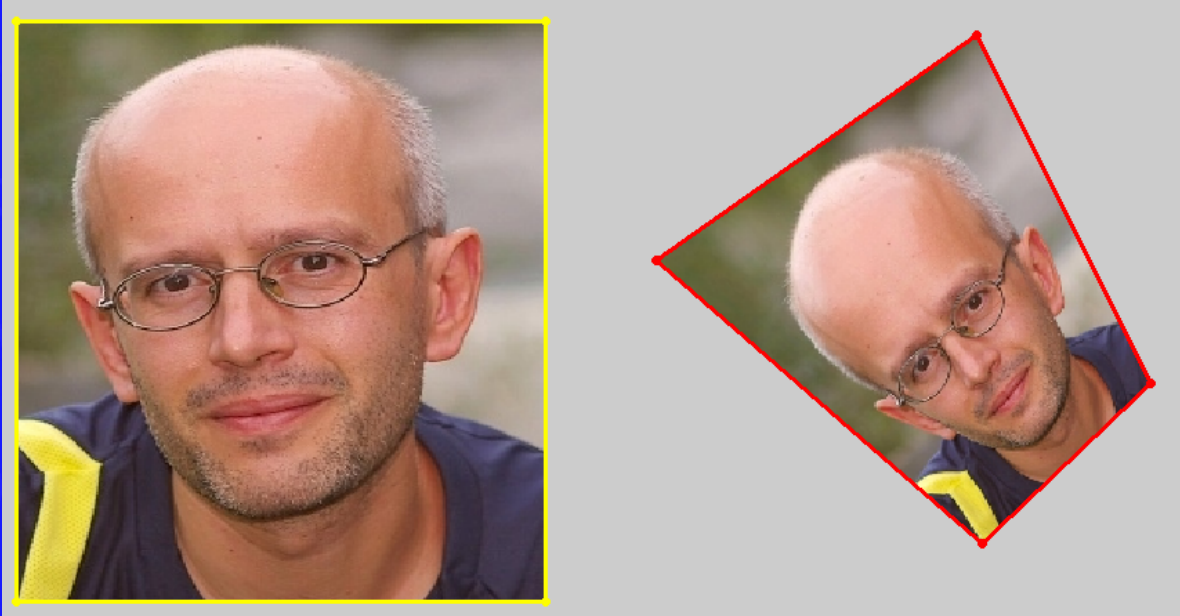
$$u_4 = A + Bx_4 + Cy_4 + D \cdot x_4y_4$$

$$v_4 = a + bx_4 + cy_4 + d \cdot x_4y_4$$

$$\begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{pmatrix} = \begin{pmatrix} 1 & x_1 & y_1 & x_1y_1 \\ 1 & x_2 & y_2 & x_2y_2 \\ 1 & x_3 & y_3 & x_3y_3 \\ 1 & x_4 & y_4 & x_4y_4 \end{pmatrix} \begin{pmatrix} A \\ B \\ C \\ D \end{pmatrix}$$

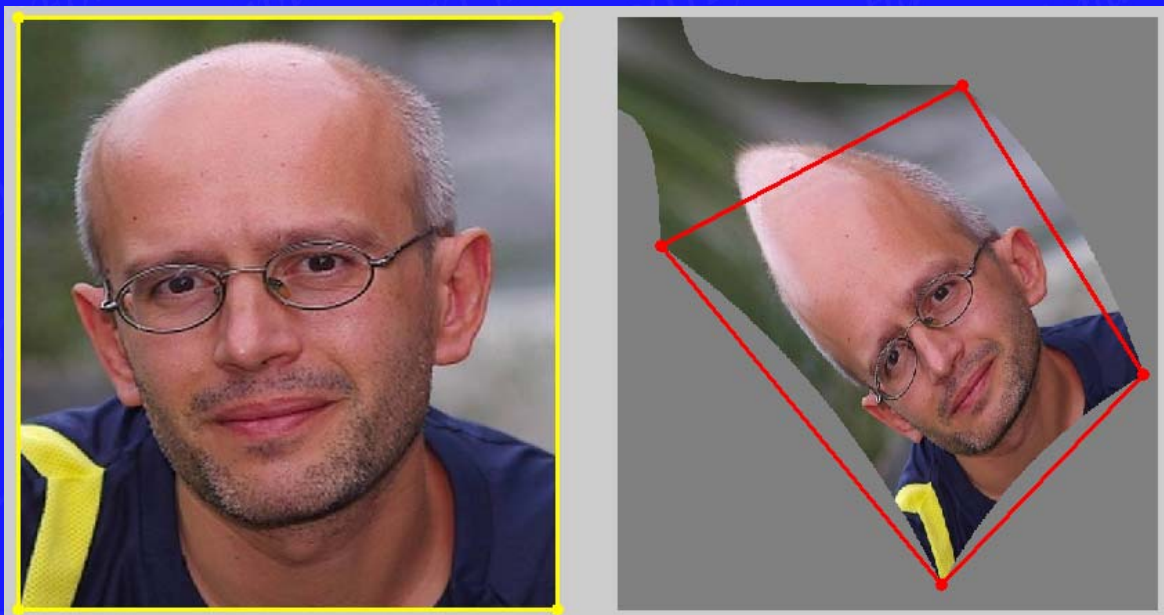
$$\begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{pmatrix} = \begin{pmatrix} 1 & x_1 & y_1 & x_1y_1 \\ 1 & x_2 & y_2 & x_2y_2 \\ 1 & x_3 & y_3 & x_3y_3 \\ 1 & x_4 & y_4 & x_4y_4 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix}$$

¿Qué diferencia hay con una proyectiva?



Transformación proyectiva (lineal)

¿Qué diferencia hay con una proyectiva?



Transformación no lineal

Aplicaciones Transformaciones Geométricas

ANTONIO TABERNERO GALÁN, 2019

FOTOGRAFÍA COMPUTACIONAL, ETSII, UPM

"Mini-mundos"

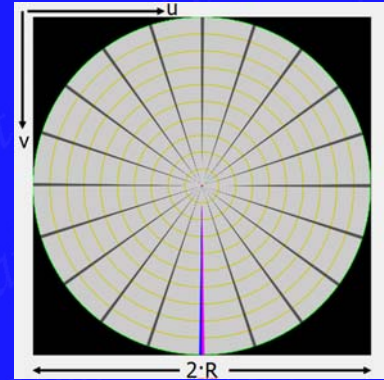
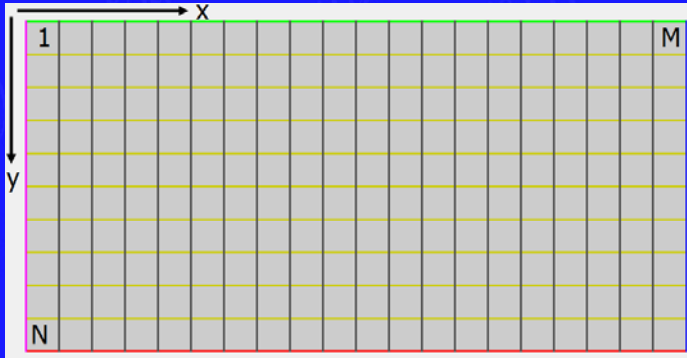
Presentación de un panorama de 360° usando una transformación no lineal polares \leftrightarrow rectangulares



ANTONIO TABERNERO GALÁN, 2019

FOTOGRAFÍA COMPUTACIONAL, ETSII, UPM

Mini-mundos



$$x = 1 + \theta \cdot (M - 1)$$

$$y = N - r \cdot (N - 1)$$

$$r = \frac{\sqrt{(u - R)^2 + (v - R)^2}}{N}$$

$$\theta = \text{atan2}(u - R, v - R)$$

$$\theta = \text{rem}(\theta, 2\pi) / (2\pi)$$

"Tiny planets"



Insertar una imagen dentro de otra



ANTONIO TABERNERO GALÁN, 2019

FOTOGRAFÍA COMPUTACIONAL, ETSII, UPM

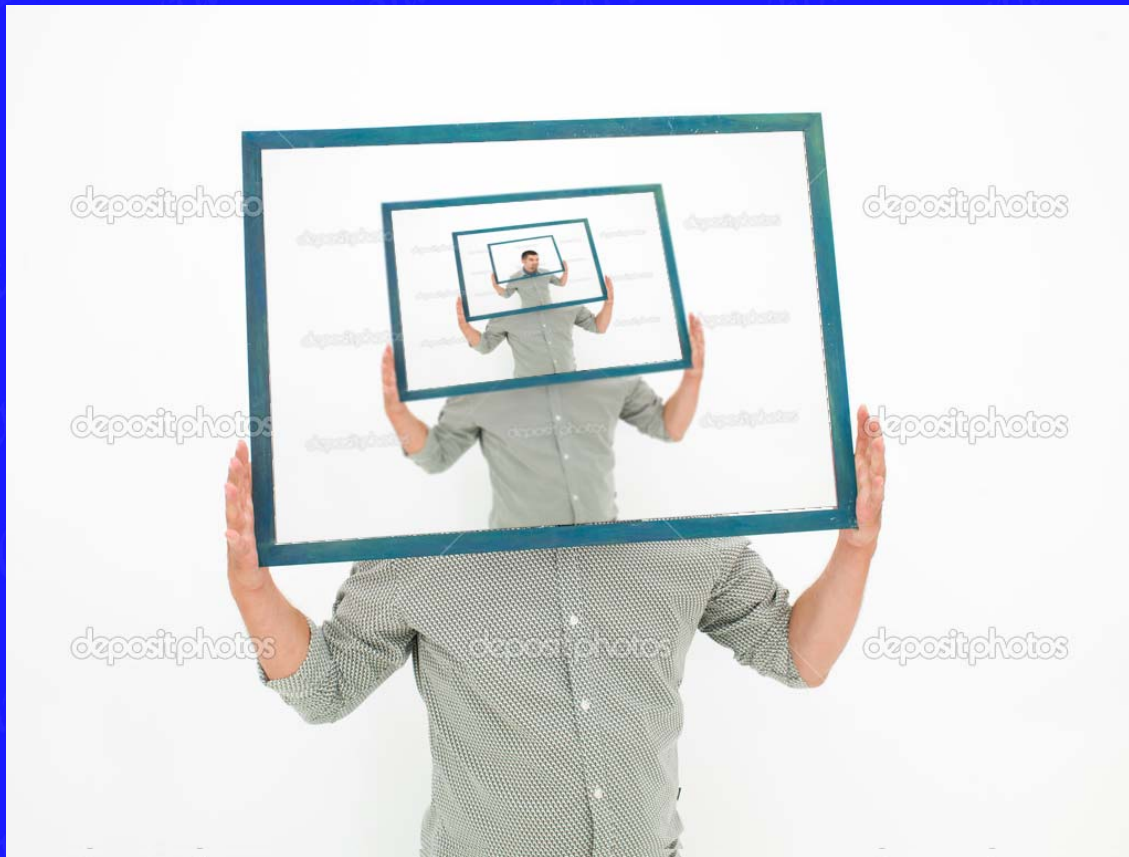
Insertar una imagen dentro de otra



ANTONIO TABERNERO GALÁN, 2019

FOTOGRAFÍA COMPUTACIONAL, ETSII, UPM

Fotos "recursivas"



ANTONIO TABERNERO GALÁN, 2019

FOTOGRAFÍA COMPUTACIONAL, ETSII, UPM

Lo mismo pero con películas

Frame# 390



ANTONIO TABERNERO GALÁN, 2019

FOTOGRAFÍA COMPUTACIONAL, ETSII, UPM

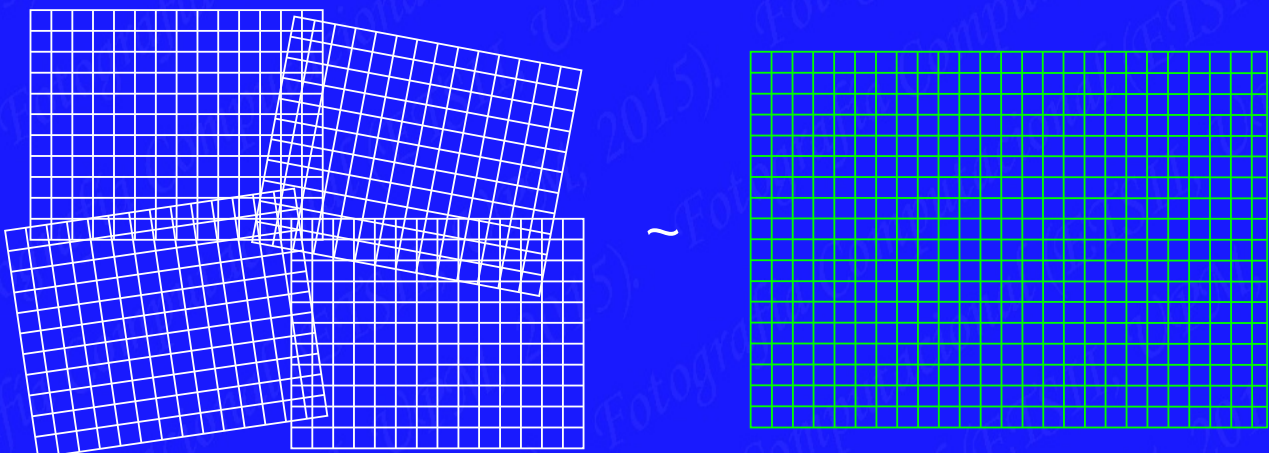
Mosaicos de fotos

- Tomar varias fotos de distintas partes de una escena
- Combinarlas en una única imagen.

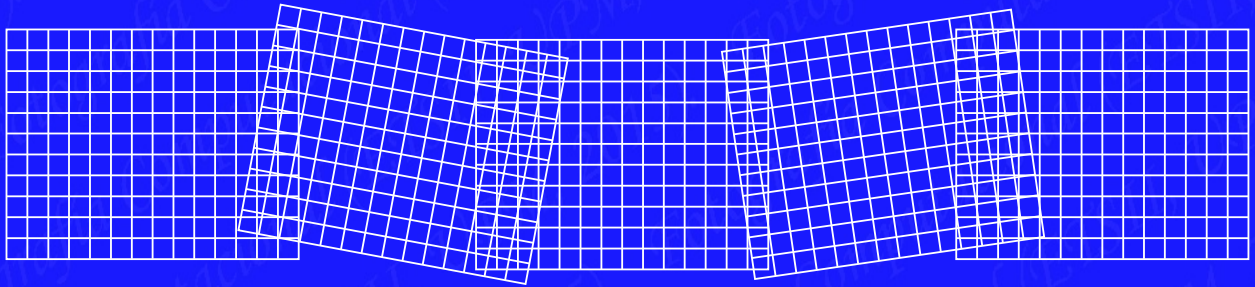


Es equivalente a cambiar el sensor de la cámara por otro mucho más grande, con una densidad mucho mayor de píxeles o con una relación de aspecto diferente.

Aumentar tamaño del sensor (gran angular virtual)



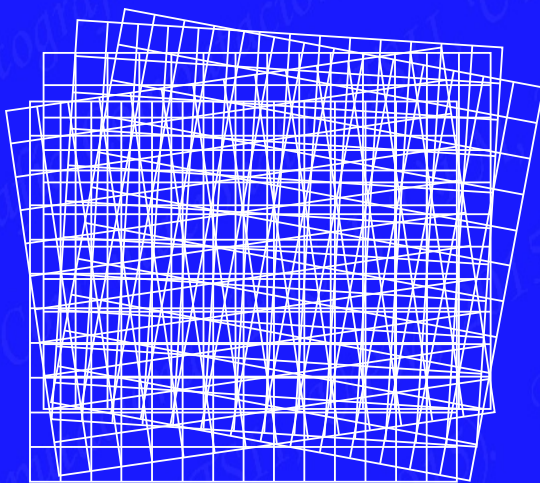
Cambiar relación aspecto del sensor (panoramas)



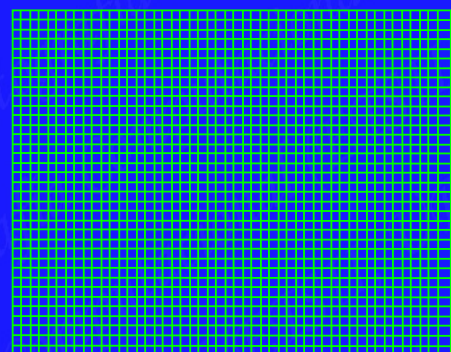
~



Aumentar resolución del sensor (superresolución)

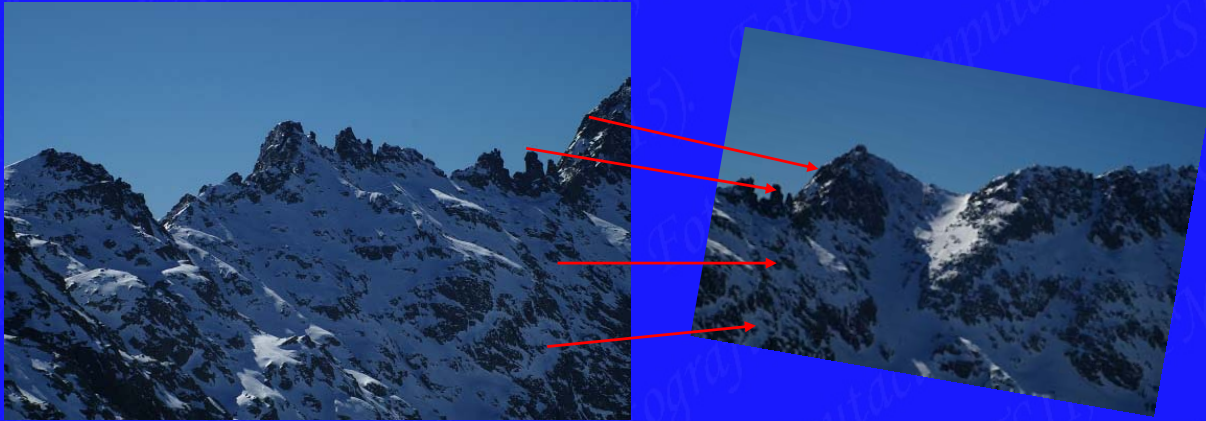


~



Registro de imágenes

- Determinar puntos comunes de control en las 2 fotos



(x_1, y_1)

(x_2, y_2)

...

(x'_1, y'_1)

(x'_2, y'_2)

...

Registro de imágenes

- Hallar una transformación T entre los puntos de control de la imagen 2 (xy_2) y los de la imagen 1 (xy_1) , tal que

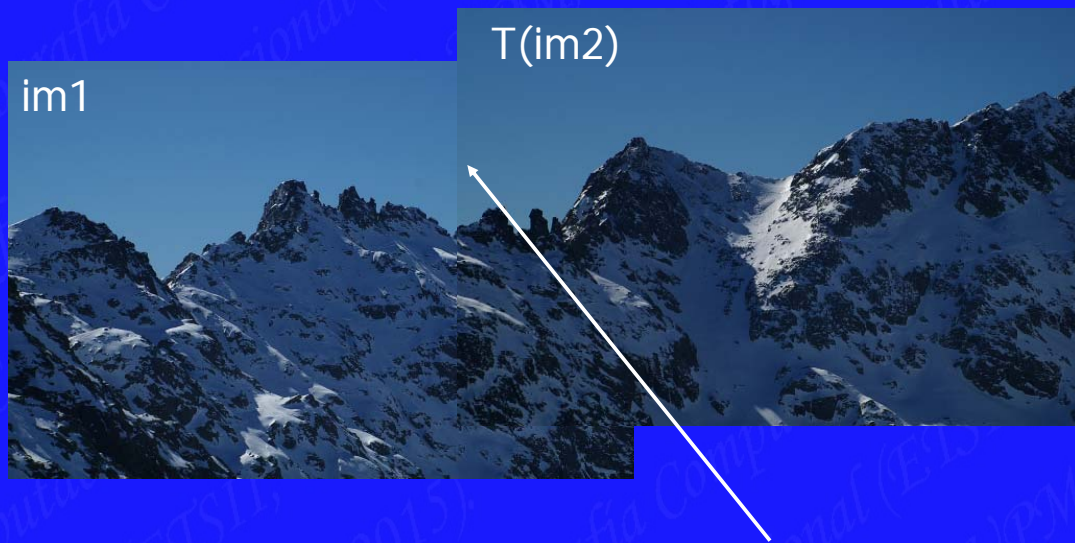
$$T(xy_2) \cong xy_1$$

¿ Qué tipo de transformación T debemos usar?

- Con unos supuestos razonables, la relación entre puntos de control será una homografía (transformación proyectiva).
- A veces puede bastarnos con una transformación afín.

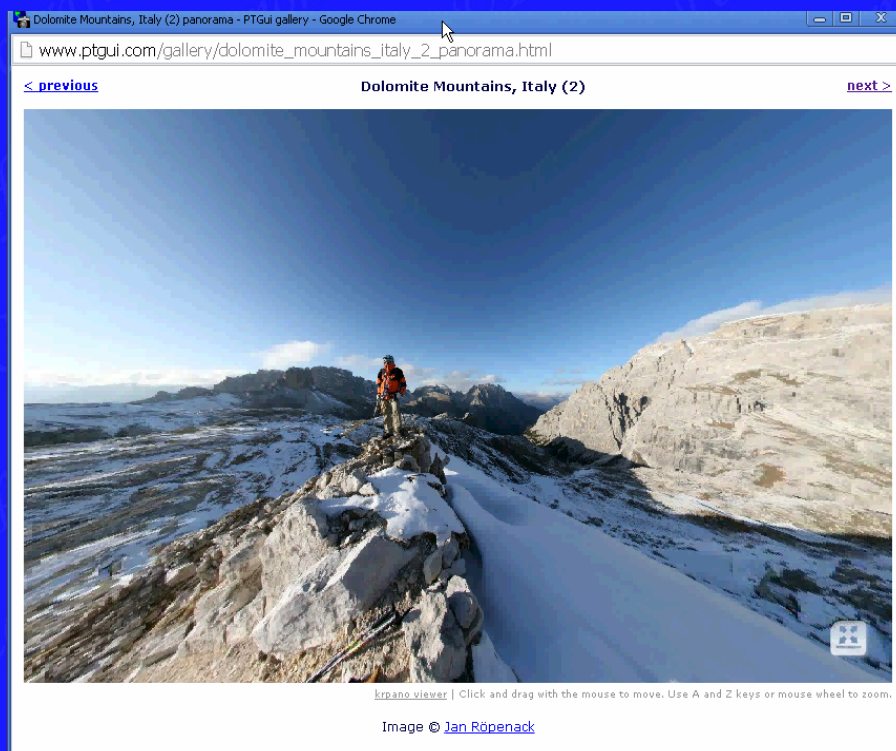
Crear el mosaico

- Aplicar relación encontrada a toda la imagen 2: $T(im2)$



- A veces será necesario retocar $im1 + T(im2)$ para equilibrar cambios de luz, balance de blancos, etc.

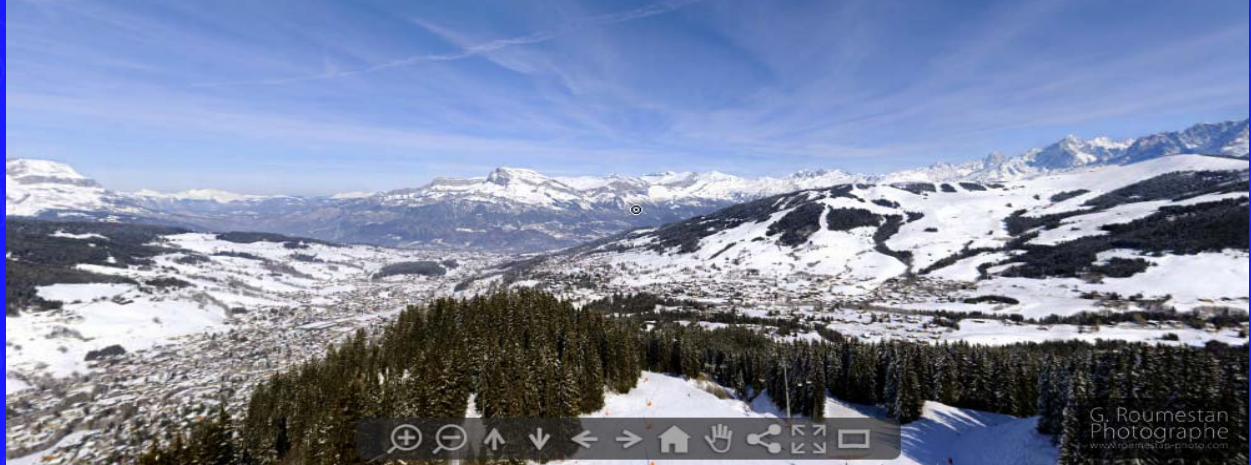
Ejemplo: panoramas esféricos



Más ejemplos en: <http://www.ptgui.com/gallery/>

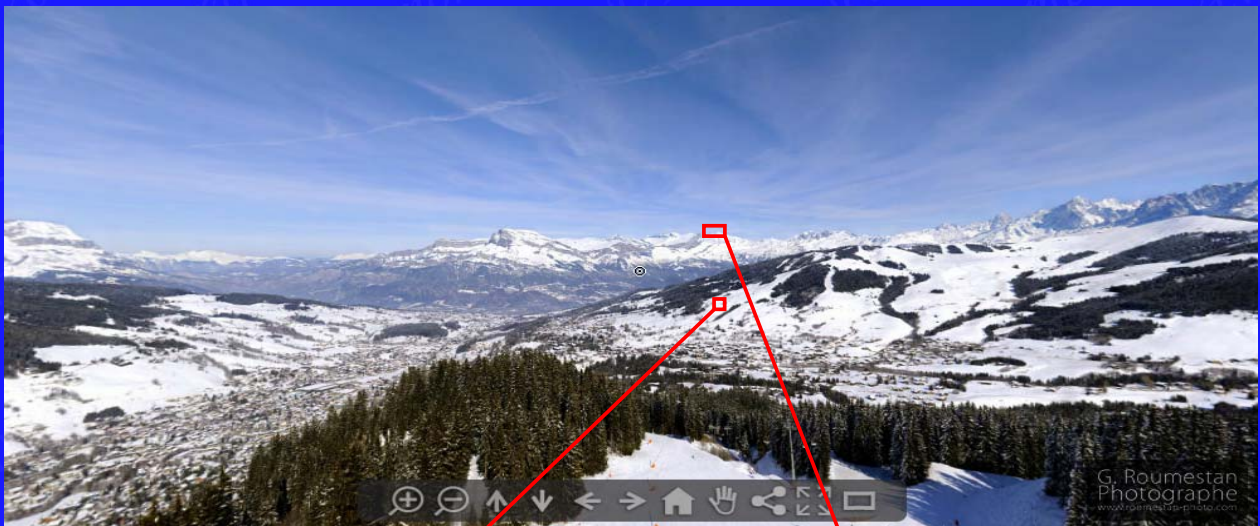
Ejemplo: Gigapixel photography

<http://www.roumestan-photo.com/>



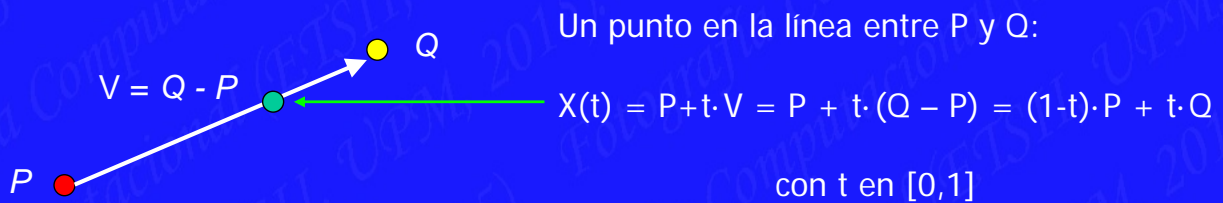
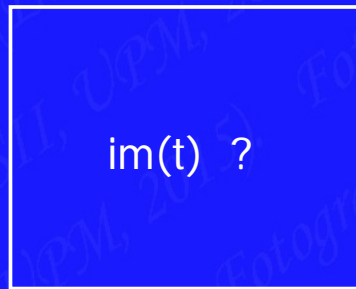
Panorama con un campo de visión 172° x 64°

Haciendo "zoom" sobre la foto



Transición entre dos imágenes

Queremos pasar de im_0 (en $t=0$) a im_1 (en $t=1$).



P y Q pueden ser cualquier cosa: $im(t) = (1-t) \cdot im_0 + t \cdot im_1$

Transición entre dos imágenes



$$im(t) = (1-t) \cdot image1 + t \cdot image2$$

Trivial si las imágenes están alineadas (cámara fija)

¿Y si no lo están?

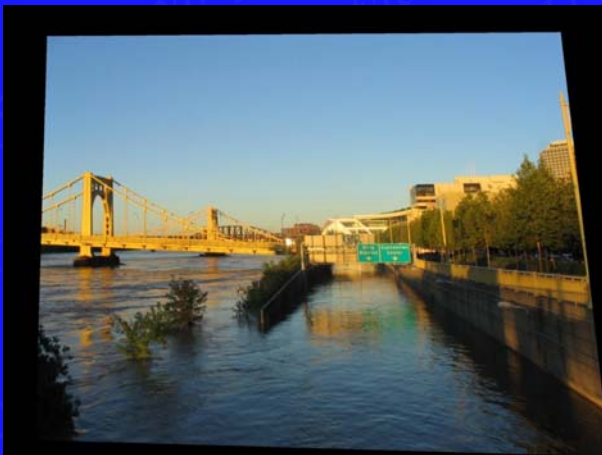
Imágenes no alineadas



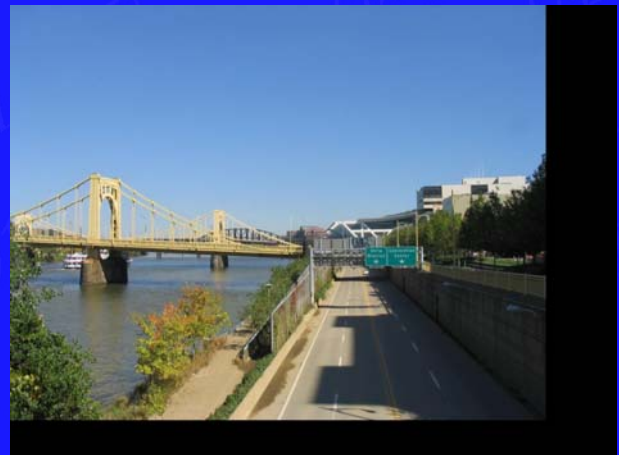
Solución:

- Alinear imágenes: $im1 \rightarrow im1'$ $im2 \rightarrow im2'$
- Fundido: $im1' \cdot (1-t) + im2' \cdot (t)$

Solución: Alinear imágenes + fundir



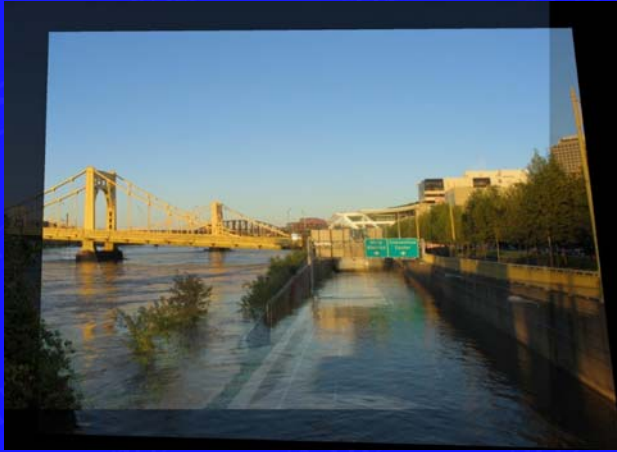
T proyectiva



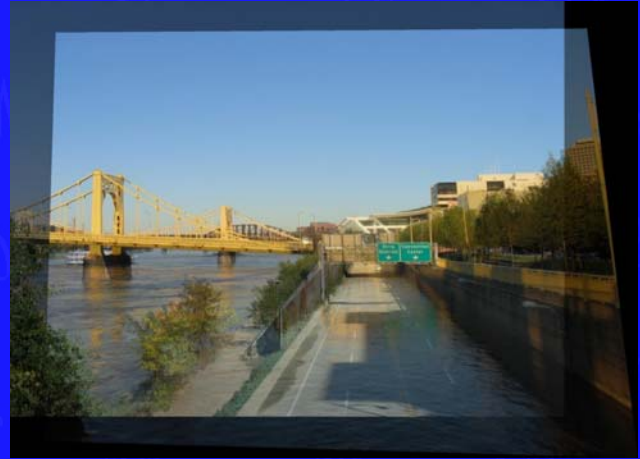
T translación

Una transformación global es adecuada en este caso

Solución: Alinear imágenes + fundir



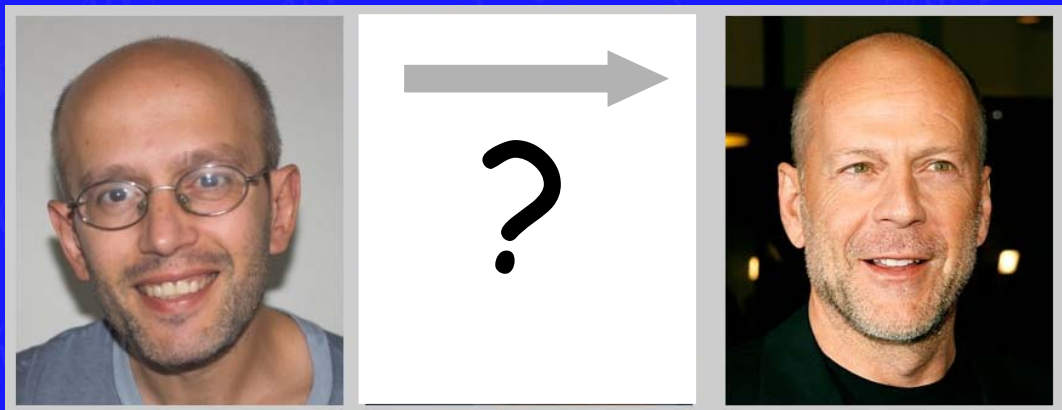
$t \approx 0$



$t \approx 1$

$$im(t) = (1-t) * image1 + t * image2$$

Transición entre imágenes: morphing



Aplicación: morphing



$t=0.5$

PROBLEMA:

Hemos calculado una **media de las imágenes**:

$$im = 0.5 \cdot im_0 + 0.5 \cdot im_1$$

Pero eso no es lo que queríamos.

Aplicación: morphing



Queremos la **media de las caras**, no de las imágenes.
Hay que deformar imágenes para que las caras se solapen

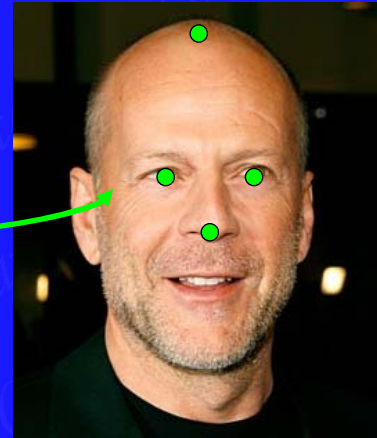
$$im_1 \rightarrow im_1' \quad im_2' \leftarrow im_2$$
$$0.5 \cdot im_1' + 0.5 \cdot im_2'$$

Limitaciones de una T global



Calculo T para los 4 puntos de control dados.

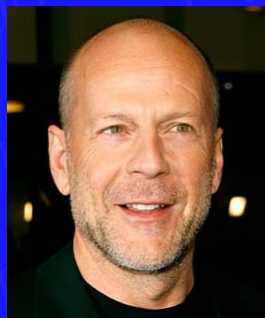
T proyectiva



Aplico T hallada a mi imagen



Resultado



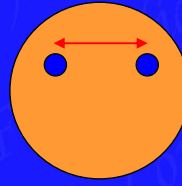
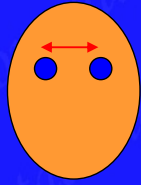
Correcto en los puntos de control (ojos, nariz), pero con fallos en boca, oreja, etc: los puntos que no se usaron para hallar la transformación.

Necesitamos más grados de libertad (más parámetros). ¿Opciones?

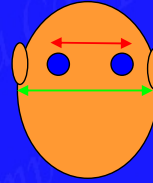
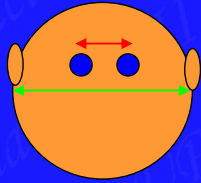
Seguir con el enfoque global, pero aumentando el número de parámetros de la transformación.

No es buena idea (acordaros de lo que pasaba en las interpolaciones con un grado muy alto del polinomio)

¿Transformación global inadecuada?



Marcar ojos, ajustar escala horizontal -> sin problemas
Usar T global de escalado que amplía en un pequeño factor.



Marcar ojos y orejas, ajustar escala -> imposible
Ampliar para separar ojos, reducir para juntar orejas.

Solución: transformaciones locales

No usar la misma T (global) para toda la imagen

En el caso más extremo, podríamos definir una transformación local personalizada para cada punto $(x,y) \rightarrow (x',y')$, pero sería muy costoso en la práctica.

Compromiso: usar funciones "a trozos", dividiendo la imagen en zonas y especificando una $T(k)$ diferente para cada zona.

Cada aplicación $T(k)$ sigue teniendo unos pocos parámetros:

- Sencillas de calcular
- No dan problemas de inestabilidad

Deformaciones locales o globales

- En las transformaciones globales los mismos parámetros se aplican a todas las coordenadas de la imagen.
- Podemos definir T locales (definidas de forma distinta = distintos parámetros para las diferente zonas de la imagen). Una T local permite deformaciones más precisas y localizadas.

Original



T global



Deformaciones locales o globales

- En las transformaciones globales los mismos parámetros se aplican a todas las coordenadas de la imagen.
- Podemos definir T locales (definidas de forma distinta = distintos parámetros para las diferente zonas de la imagen). Una T local permite deformaciones más precisas y localizadas.

Original



T local

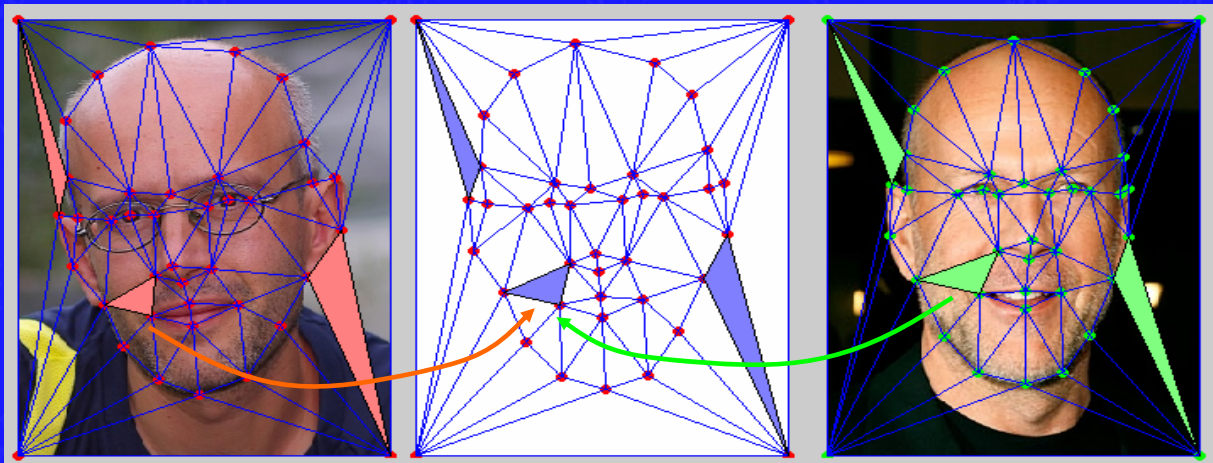


¿Cómo especificar un warping a trozos?

- Definir zonas $\{Z_1, Z_2, \dots, Z_k, \dots\}$ y $\{Z'_1, Z'_2, \dots, Z'_k, \dots\}$ en las imágenes de origen y destino respectivamente.
- Asignar una transformación T_k (distinta) a cada zona
- Se debe verificar que:
 - $Z_k = T_k [Z'_k]$
 - Las zonas Z_1, Z_2, \dots cubran la imagen de partida.
 - Las zonas Z'_1, Z'_2, \dots deben recubrir la imagen final.
 - Los puntos de control vayan a donde queremos.

Parece muy complicado ... ¿o tal vez no?

Morphing



- 1) Marcamos (muchos) puntos en las dos imágenes, en rasgos comunes.
- 2) Promediamos los puntos de referencia en una cara "media".
- 3) Hacemos una triangulación para crear malla a partir de los puntos.
- 4) Para todos los triángulos hallamos las transformaciones que nos llevan desde ambas imágenes a la malla intermedia.



¿Qué tipo de T usar?

Morphing



- 5) Deformamos la primera imagen a la posición "media".
- 6) Repetimos para la segunda imagen
- 7) Mezclamos ($0.5 \text{ img}_1 + 0.5 \text{ img}_2$) ambas imágenes deformadas.

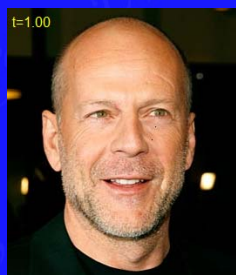
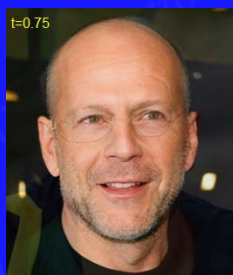
Secuencia de Morphing

t	img
0	
1	

La transformaciones $\{T_1\}$ y $\{T_2\}$ que deforman ambas caras para llevarlas a la malla media donde sus rasgos se superponen también pueden hacerse variar con t .

Podemos hacer evolucionar la malla desde la posición correspondiente a la imagen 1 hasta la imagen 2.

$$\text{img}(t) = (1 - t) \cdot T_1(t)\{\text{img}_1\} + t \cdot T_2(t)\{\text{img}_2\}$$



Cambio de relación de aspecto



Problema: ¿Presentarla en display con relación aspecto 1:1?

Aplicar escalado no uniforme

Necesitamos imagen 500x500 para ajustarse a un display 1:1
Aplicar escalado diferenciado en ambos ejes: $s_y=1$, $s_x=0.5$



Alternativas: recorte o cropping

Alternativas: hacer un crop 500x500 de la imagen original



Problema: perdemos contenido

Alternativas: rellenar con ceros

Escalado uniforme (sin distorsión): $s_y=0.5$, $s_x=0.5$

Imagen final < pantalla disponible \Rightarrow rellenar con ceros



Problema: desaprovechamos "pantalla"
especialmente crítico en dispositivos móviles

Alternativas

Seguir un enfoque de recortar, pero con algunas restricciones:

- sin perder contenido "importante"
- sin que se note.



¿Dónde está el contenido? ¿Cómo recortar sin que se note?

Seam Carving

Definir una "energía" de la imagen, asociada al contenido.

Usar los bordes como indicadores del contenido de la imagen

$$E(i,j) = |dI/dx| + |dI/dy| \quad E(i,j) = |E(i,j) - \text{media}(\text{vecindad})|$$



Energía de una costura

Definimos una línea de recorte vertical como una colección de valores $\{j\}$, uno para cada fila i , con la condición de que:

$$|j(i) - j(i-1)| \leq 1$$

Esto es, no podemos movernos más de una unidad entre fila y fila (esto garantiza continuidad de la imagen recortada).

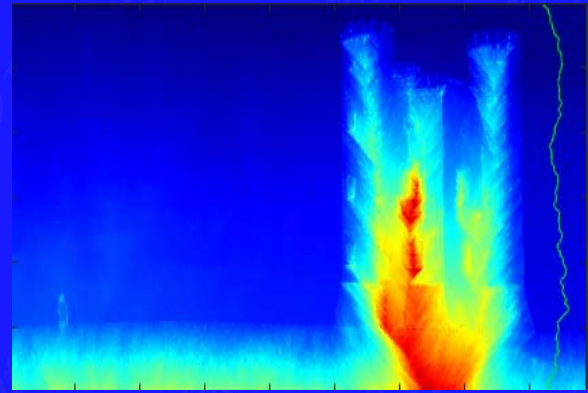
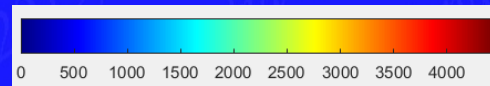
Una línea de recorte óptima será aquella que minimice la función energía (basada en bordes) a lo largo de su recorrido:

$$E(\text{recorte}) = \sum_i E(i, j(i))$$

Energía media acumulada

- 1) Calculamos función de energía $E(i,j)$ para la imagen.
- 2) Calculamos la energía mínima acumulada $M(i,j)$:
 - a) Inicializamos 1ª fila de M : $M(1,:) = E(1,:)$
 - b) Bucle filas $i=2:N$ $M(i,j) = \min\{M(i,j-1:j+1)\} + E(i,j)$
- 3) Al terminar, la última fila de M es la energía mínima de todos los caminos que acaban en cada píxel del ancho de la imagen.
- 4) Píxel con valor mínimo de última fila \rightarrow final de la costura.
- 5) Retrocedemos hallando el camino óptimo.

Energía media acumulada (vertical)

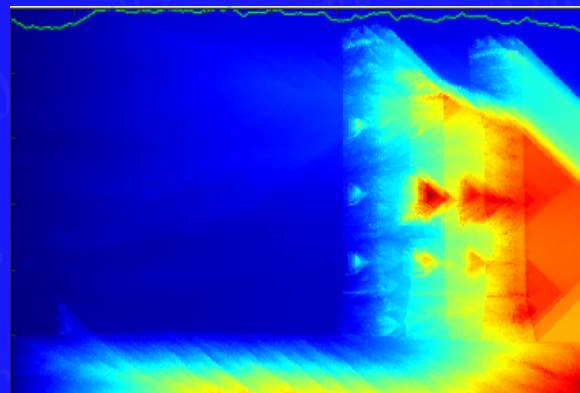
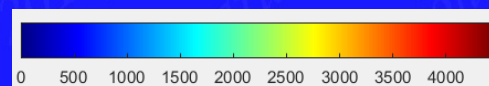


↑
mínimo

ANTONIO TABERNEIRO GALÁN, 2019

FOTOGRAFÍA COMPUTACIONAL, ETSII, UPM

Energía media acumulada (horizontal)



ANTONIO TABERNEIRO GALÁN, 2019

FOTOGRAFÍA COMPUTACIONAL, ETSII, UPM

Seam Carving

- Recortar siguiendo caminos de energía mínima, evitando atravesar bordes de alta energía.
- En el caso anterior: encontrar caminos "verticales" de energía mínima y quitarlos de la imagen hasta llegar a la relación de aspecto deseada:



ANTONIO TABERNERO GALÁN, 2019

FOTOGRAFÍA COMPUTACIONAL, ETSII, UPM

Seam Carving

- Recortar siguiendo caminos de energía mínima, evitando atravesar bordes de alta energía.
- En el caso anterior: encontrar caminos "verticales" de energía mínima y quitarlos de la imagen hasta llegar a la relación de aspecto deseada:



ANTONIO TABERNERO GALÁN, 2019

FOTOGRAFÍA COMPUTACIONAL, ETSII, UPM