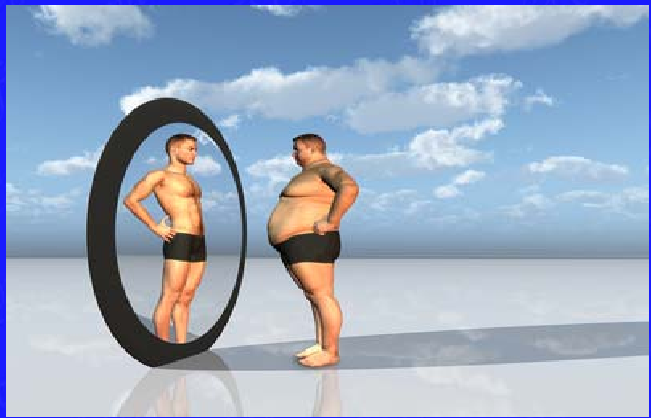
Deformación de imágenes (warping)

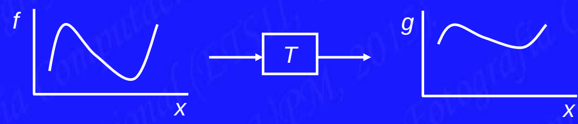


Algunas transparencias inspiradas en Derek Hoiem, Alyosha Efros + Steve Seitz

Transformaciones de rango (previo)

El valor del píxel puede cambiar pero sigue estando asociado a la misma posición (x,y) original.

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

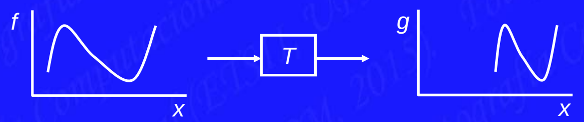


Original $im(\bar{x})$

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

Transformaciones de dominio (hoy)

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



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

Original $im(\bar{x})$

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

Temas a tratar

Transformaciones de coordenadas:

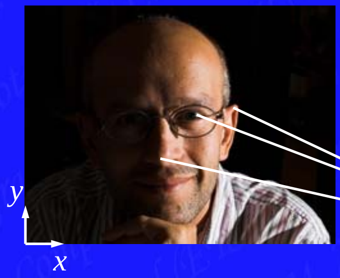
- Transformaciones globales
- Puntos de control: determinar transformación
- Transformaciones locales: mallas, triangulación
  Interpolación

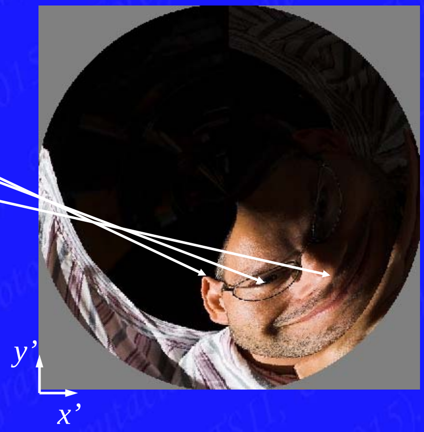
Aplicaciones:

- Creación de panoramas
- Registro de imágenes previo a su combinación.
- Morphing

T(x) puede ser cualquier cosa

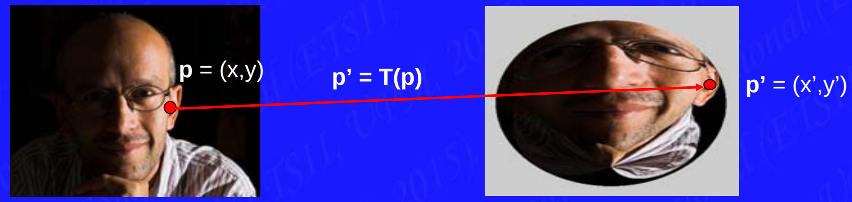
T(x) define una relación entre las coordenadas (x,y) y (x',y').





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

Transformaciones globales paramétricas



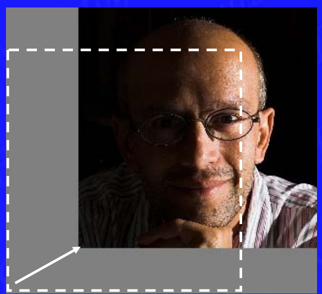
La transformación T es un cambio de coordenadas: p' = T(p)

- Global: la misma T para transformar todos los puntos
- Paramétrica: descrita por unos pocos parámetros.

Transformaciones lineales: T se representa como una matriz

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

Ejemplos transformaciones globales



Desplazamiento

Escalado

Cambio de relación de aspecto o factor de forma = escalado desigual

Ejemplos transformaciones globales

Transformación afín

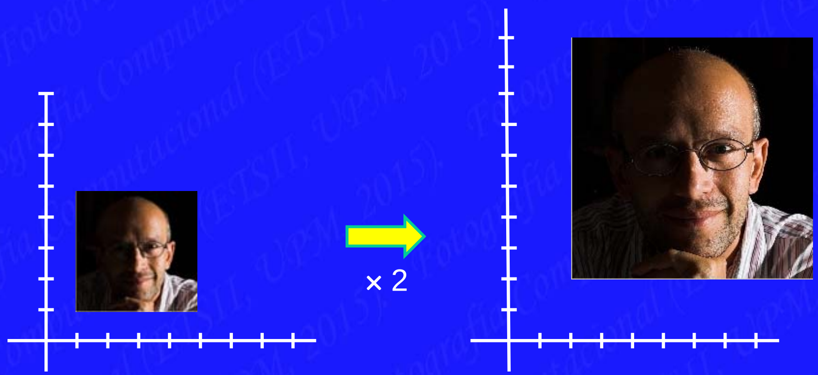
Rotation

Perspectiva

Escalado

Multiplicar cada coordinada por un número (distinto?)

Escalado uniforme = mismo factor en todos los ejes

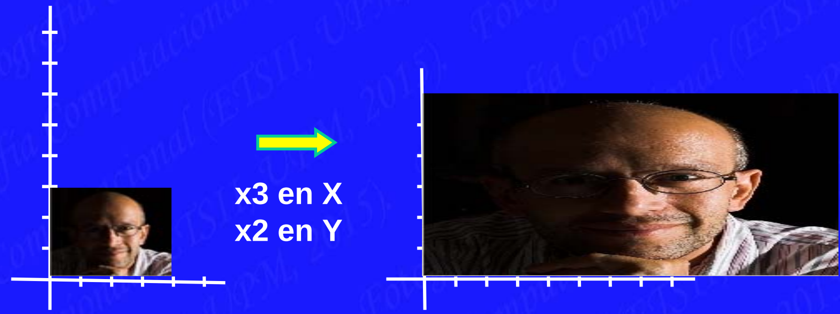


Escalado no uniforme

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

$$x' = ax$$
$$y' = by$$

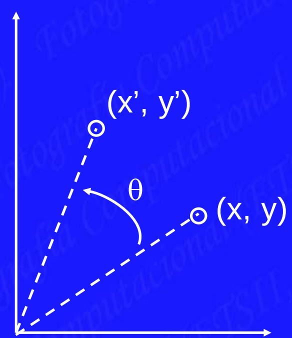
En forma matricial: $$\begin{bmatrix} x' \\ y' \end{bmatrix} = \underbrace{\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}}\begin{bmatrix} x \\ y \end{bmatrix}$$

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



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

En forma matricial: $$\begin{pmatrix} x' \\ y' \end{pmatrix} = \overbrace{\begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}}^{R} \cdot \begin{pmatrix} x \\ y \end{pmatrix}$$

Rotación 2D

Aunque sin(θ), cos(θ) sean funciones no lineales, las coordenadas (x',y') son combinación lineal de (x,y).

Para deshacer la transformación basta rotar por (- θ):

$$R^{-1} = \begin{bmatrix} \cos(-\theta) & -\sin(-\theta) \\ \sin(-\theta) & \cos(-\theta) \end{bmatrix} = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix} = R^T$$

Transformaciones lineales como matrices

2D identidad

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

2D Escalado

$$x' = s_x \cdot x \quad y' = s_y \cdot y \qquad \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

2D Rotación alrededor del (0,0)

$$x' = \cos\theta \cdot x - \sin\theta \cdot y \quad y' = \sin\theta \cdot x + \cos\theta \cdot y \qquad \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Otras Transformaciones con matrices 2x2

Cizalladura

$$x' = x + c_x \cdot y \quad y' = c_y \cdot x + y \qquad \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & c_x \\ c_y & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Reflejo sobre un eje o sobre el origen

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

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.
- Las relaciones entre puntos de una recta se mantienen.
- Su combinación sigue siendo una transformación lineal

¿Qué pasa con la translación?

No va a funcionar con nuestro esquema p' = M x p

Una translación desplaza el origen.

Eso es algo que una transformación lineal (matriz 2x2) no puede hacer.

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

Coordenadas homogéneas

La translación es una operación muy importante y la representación de transformaciones usando matrices

x' = Mx

nos resulta muy conveniente.

- Objetivo: representar translaciones con matrices.

$$x' = x + t_x$$
$$y' = y + t_y$$

ANTONIO TABERNERO GALÁN, 2015 FOTOGRAFÍA COMPUTACIONAL, ETSII, UPM

Coordenadas homogéneas

Representamos coordenadas 2D con vectores 3D:

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

Al terminar de operar (aplicar matrices) a las nuevas coordenadas volvemos a coordenadas 2D:

$$\begin{pmatrix} X \\ Y \\ W \end{pmatrix} \longrightarrow \begin{cases} x' = \dfrac{X}{W} \\ y' = \dfrac{Y}{W} \end{cases}$$

Vuelta a coordenadas 2D

ANTONIO TABERNERO GALÁN, 2015 FOTOGRAFÍA COMPUTACIONAL, ETSII, UPM

Translación como matriz

¿Cómo representar T como una matriz?

$$x' = x + t_x$$
$$y' = y + t_y$$

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

Salida Entrada

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

ANTONIO TABERNERO GALÁN, 2015 FOTOGRAFÍA COMPUTACIONAL, ETSII, UPM

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 ?

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \left(\begin{bmatrix} 1 & 0 & tx \\ 0 & 1 & ty \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} sx & 0 & 0 \\ 0 & sy & 0 \\ 0 & 0 & 1 \end{bmatrix} \right) \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

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

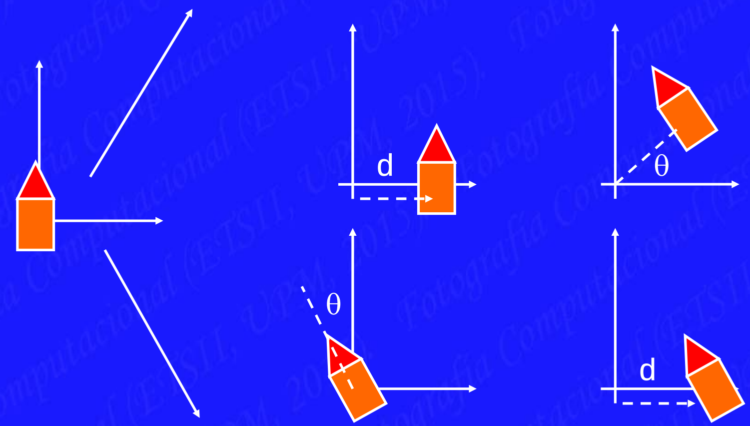
$$p' = T \cdot R \cdot S \cdot p = Q \cdot p$$

Operador compuesto $Q = T \cdot R \cdot S$ producto de matrices

¿Importa el orden de las operaciones?

Producto de Matrices no es conmutativo

Desplazamiento + Giro Q = R·T

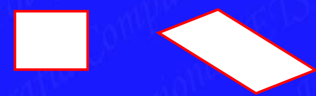


Giro + Desplazamiento Q = T·R

Transformaciones afines

Combinación de Transformaciones lineales + Desplazamiento:

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



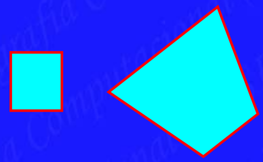
Propiedades:

- 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 vamos a empezar a usar la tercera coordenada w?

Transformaciones proyectivas

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



Combinación de transformaciones afines + proyecciones.
También llamadas homografías

- Origen no tiene por qué mantenerse (translaciones)
- 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?

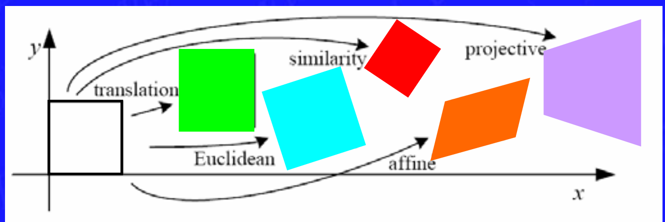
$$\begin{bmatrix} X \\ Y \\ w \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \longrightarrow \begin{cases} x' = \dfrac{X}{w} \\ y' = \dfrac{Y}{w} \end{cases}$$

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

Tras reducir a coordenadas 2D el resultado final es el mismo

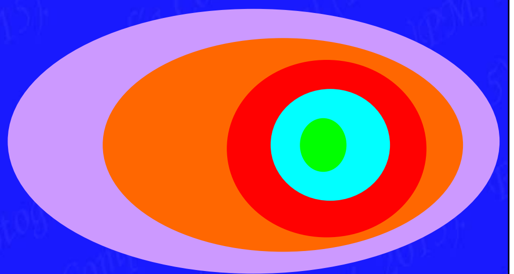
ANTONIO TABERNERO GALÁN, 2015 FOTOGRAFÍA COMPUTACIONAL, ETSII, UPM

Transformaciones 2D de imágenes



Cada una de las familias engloba a las anteriores

Sus inversas están dentro de la familia respectiva.



ANTONIO TABERNERO GALÁN, 2015 FOTOGRAFÍA COMPUTACIONAL, ETSII, UPM

Transformaciones 2D de imágenes

Name	Matrix	# D.O.F.	Preserves:	Icon
translation	$\begin{bmatrix} I \mid t \end{bmatrix}_{2\times 3}$	2	orientation $+\cdots$	
rigid (Euclidean)	$\begin{bmatrix} R \mid t \end{bmatrix}_{2\times 3}$	3	lengths $+\cdots$	
similarity	$\begin{bmatrix} sR \mid t \end{bmatrix}_{2\times 3}$	4	angles $+\cdots$	
affine	$\begin{bmatrix} A \end{bmatrix}_{2\times 3}$	6	parallelism $+\cdots$	
projective	$\begin{bmatrix} \tilde{H} \end{bmatrix}_{3\times 3}$	8	straight lines	

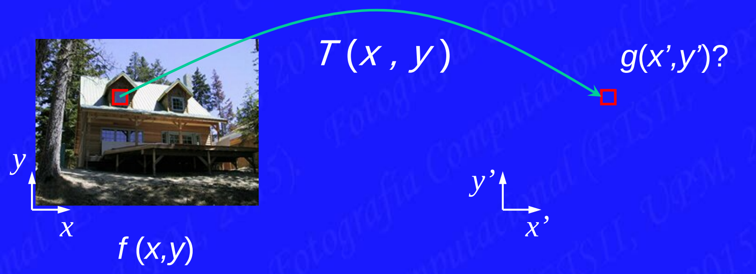
ANTONIO TABERNERO GALÁN, 2015 FOTOGRAFÍA COMPUTACIONAL, ETSII, UPM

Cuestiones prácticas implementación

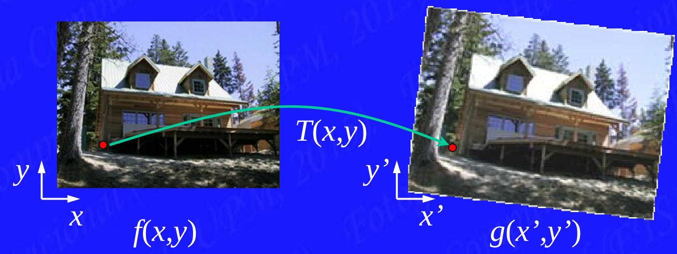
Me dan una transformación de coordenadas (x’,y’) = T(x,y)

Conocida la imagen de pártida f(x,y).

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



Warping directo (forward, hacia delante)

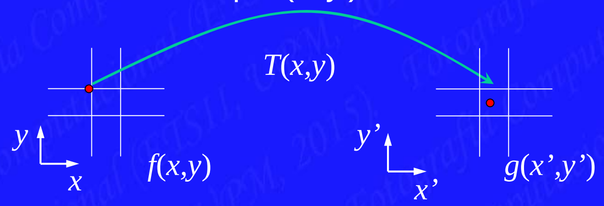


- Barre imagen de partida en (x,y)
- Para cada posición (x,y) calcula la posición (x’, y’)=T(x,y) y coloca el pixel f(x,y) en dicha posición de destino:

Bucle en (x,y): calculo (x’,y’) = T(x,y) => g(x’,y’) = f(x,y)

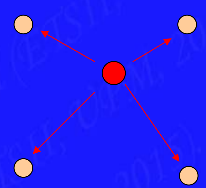
Warping directo

Problema: lo normal es que (x’,y’) no sean enteros.



¿Qué hacemos cuando un un píxel cae entre líneas?

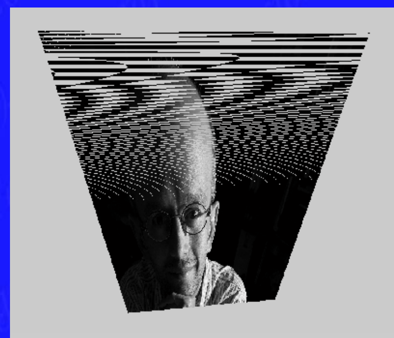
Esparcir su “color” entre sus vecinos (splatting)



¿Problemas de warping directo?

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

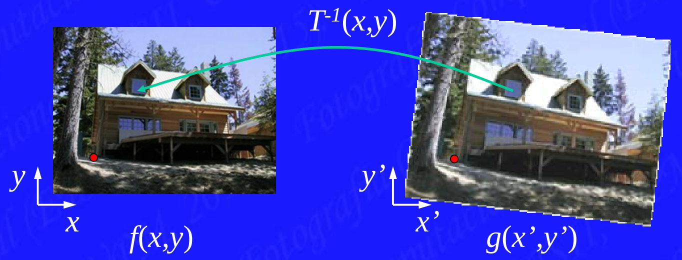
Agujeros en la imagen de destino



Warping inverso

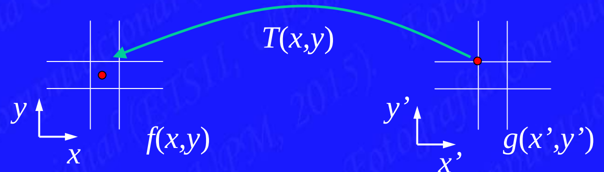
Barre imagen destino en sus coordenadas (x',y').
Aplicamos T inversa para obtener coordenadas de origen (x,y)
Usamos f(x,y) para rellenar el píxel g(x',y')

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

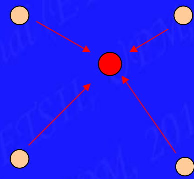


Warping inverso

Mismo problema: (x , y)= T^{-1}(x',y') no serán enteros

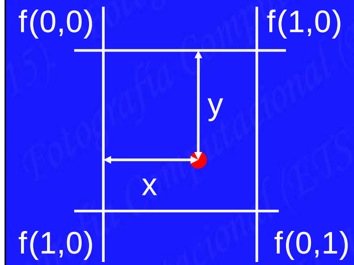


¿Qué hacer si un píxel viene de "entre" varios?
Interpolar a partir de sus vecinos:

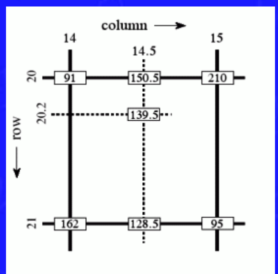


MATLAB: im2 = interp2(im, x_i, y_i, metodo);
metodo = 'nearest', 'bilinear', 'spline'

Interpolación bilineal



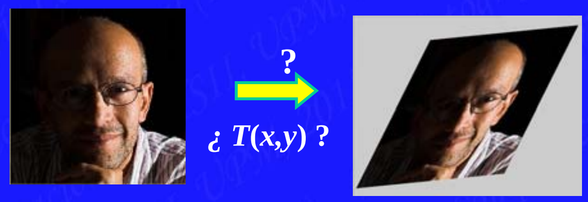
$$
\begin{aligned}
f(x,y) &= (1-x)(1-y) \quad f(0,0) \\
&+ \quad (1-x)(y) \quad f(1,0) \\
&+ \quad (x)(1-y) \quad f(1,0) \\
&+ \quad (x)(y) \quad f(1,1)
\end{aligned}
$$



Recuperando una transformación

Otro problema: Imagen1 → T ? → Imagen 2

f (x,y)



g(x',y')

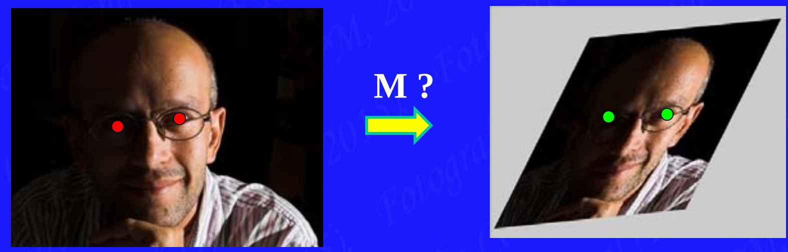
Conocidas f(x,y) y g(x',y') hallar transformación T con T(f)=g

Similar al que problema que resolvimos en 3D

Aplicaciones en registro de imágenes.

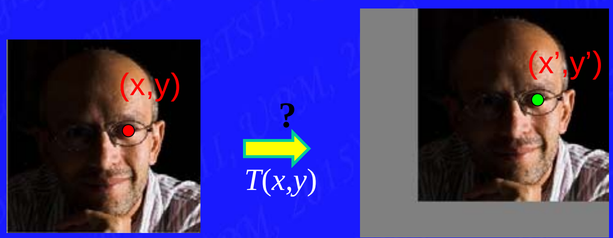
Puntos de Control

- 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



¿Cómo hallarlos? ¿Cuántos necesitaremos?

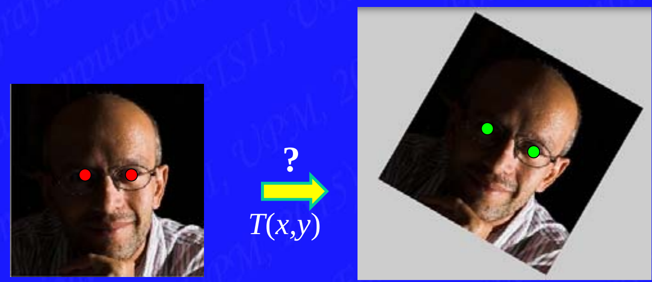
Translación Pura



Grados de libertad (DOF) = 2 = coeficientes a determinar
Necesitamos 2 ecuaciones. Un solo punto nos basta

$$\begin{aligned} \mathbf{x}' &= \mathbf{x} + \mathbf{t}_x \\ \mathbf{y}' &= \mathbf{y} + \mathbf{t}_y \end{aligned} \Rightarrow \begin{aligned} t_x &= x' - x \\ t_y &= y' - y \end{aligned} \Rightarrow \mathbf{M} = \begin{bmatrix} 1 & 0 & x'-x \\ 0 & 1 & y'-y \\ 0 & 0 & 1 \end{bmatrix}$$

Transformación similaridad



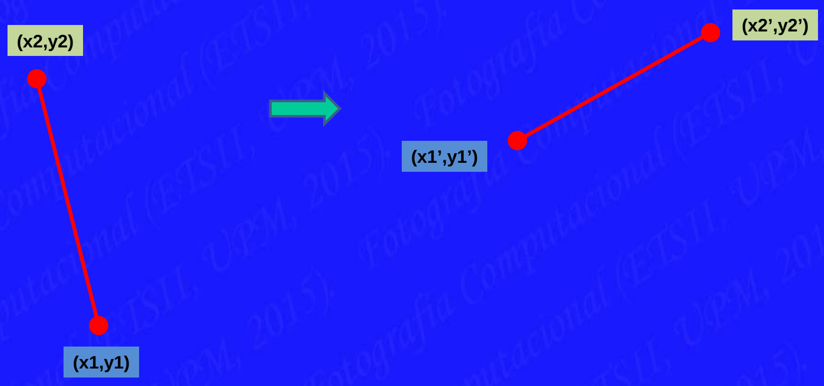
Giros + translación + escalado uniforme

¿Grados de libertad? = Parámetros a determinar: (tx, ty, s, q)

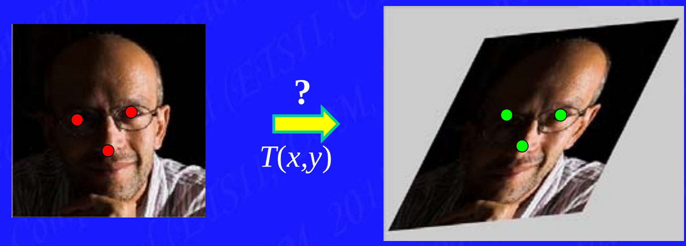
Necesitamos 4 ecuaciones = 2 puntos

T similaridad

Una transformación de similaridad permite convertir cualquier recta de la imagen (1) en cualquier otra en la imagen (2).



Transformación Afín



Grados de libertad = parámetros a determinar ?

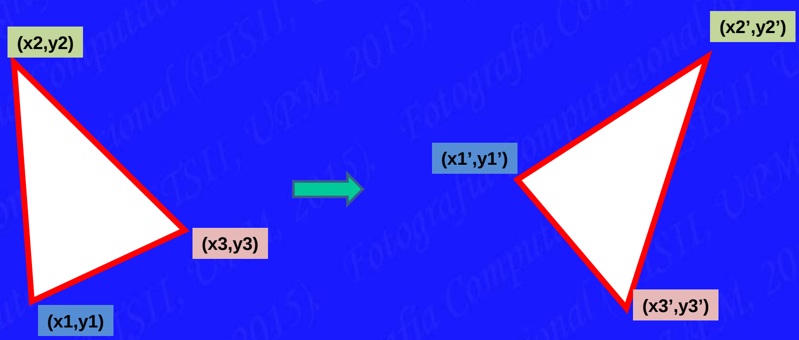
¿Cuántas ecuaciones? ¿Cuántos puntos necesitamos?

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

Transformación Afín

Permite transformar un triángulo en cualquier otro.

¿Cómo obtendremos los parámetros?



Transformación afín

Conocemos tres puntos (xi ,yi) en img1 y los correspondientes tres puntos (xi’,yi’) en img2

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

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

Determinar T afín

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

$$x' = ax + by + c$$
$$y' = dx + ey + f$$

$$\begin{aligned} x_1' &= ax_1 + by_1 + c \\ y_1' &= dx_1 + ey_1 + f \\ x_2' &= ax_2 + by_2 + c \\ y_2' &= dx_2 + ey_2 + f \\ x_3' &= ax_3 + by_3 + c \\ y_3' &= dx_3 + ey_3 + f \end{aligned}$$

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

Separabilidad de T afín

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

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

Ajuste

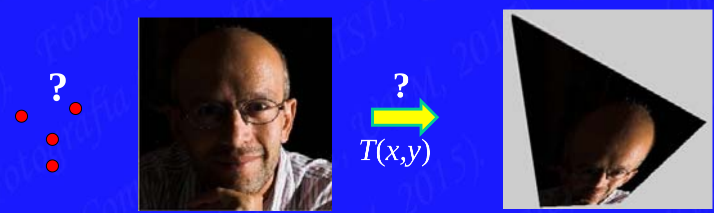
Si hay más puntos de los 3 necesarios -> problema de ajuste

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

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

La resolución en MATLAB es idéntica: coefs = H \ x'

Transformación proyectiva



N puntos (xi,yi) en imagen 1 y N puntos (xi',yi') en imagen 2

¿ Cómo verificar

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

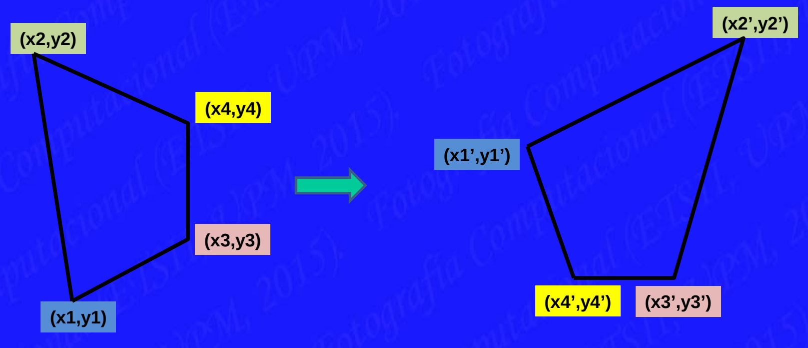
para el conjunto de puntos de control?

Las incógnitas son los elementos de H, h_ij.

Transformación proyectiva

Permite transformar un cuadrilatero en otro.

8 incógnitas => 8 ecuaciones = 4 parejas de puntos.



Solución T proyectiva

Dados {x,y} + {x',y'}, ¿como 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 elementos de P como vector incógnitas

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

Solución T proyectiva

Planteando las ecuaciones a cumplir, el vector h debe verificar

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

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

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

Solución T proyectiva

Las submatrices Z, A, B, C son de tamaño 4 x 3:

$$A = \begin{pmatrix} \begin{pmatrix} \bar{x} \end{pmatrix}, \begin{pmatrix} \bar{y} \end{pmatrix}, \begin{pmatrix} \bar{1} \end{pmatrix} \end{pmatrix} \qquad Z = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

$$b = \begin{pmatrix} \begin{pmatrix} \bar{x}' \end{pmatrix}, \begin{pmatrix} \bar{x}' \end{pmatrix}, \begin{pmatrix} \bar{x}' \end{pmatrix} \end{pmatrix} \Rightarrow B = -A.*b \qquad c = \begin{pmatrix} \begin{pmatrix} \bar{y}' \end{pmatrix}, \begin{pmatrix} \bar{y}' \end{pmatrix}, \begin{pmatrix} \bar{y}' \end{pmatrix} \end{pmatrix} \Rightarrow C = -A.*c$$

¿Solución de S·h=0?

Autovector de la matriz Q=(S'·S) con el menor autovalor

En MATLAB:

```
Q = (S'*S);
[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 P(3,3)=1.
```


Normalización previa

En la matriz S aparecen números de escalas muy diversas.

Eso es malo de cara a resolver (matriz mal condicionada).

Es conveniente preprocesar los datos de forma que:

- Las medias de {x} e {y} sean 0.
- Una vez corregida su media, escalar {x, y} de forma que la media de su módulo sea sqrt(2).
- Repetir para {x',y'} (quitar media + escalado)

Resolver P para las coordenadas modificadas y corregirla de acuerdo a las transformaciones T y T' usadas:

P = T2^-1 x P x T1

Deformaciones locales o globales

Las transformaciones presentadas son 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

Las transformaciones presentadas son 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

Aplicaciones: Mosaicos de fotos

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

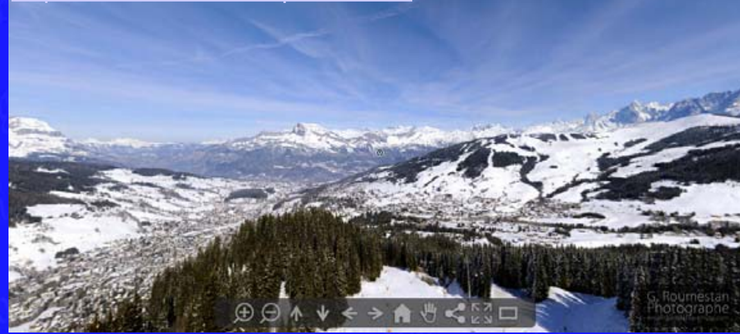
Es equivalente a cambiar el sensor de la cámara por otro mucho más grande, con una densidad mucho mayor de píxeles o con una relación de aspecto diferente.

Resultados?

- Conseguir un wide-angle virtual
- Obtener imágenes con mucha mayor resolución.
- Cambiar relación de aspecto (panoramas)

Ejemplo: Gigapixel photography

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



16 Gigapixels = 215000 x 75000 píxeles

Campo de visión 172º x 64º

Composición de 2321 fotografías con un objetivo de 500 mm.

Haciendo "zoom" sobre la foto

Mosaicos de fotos

- Hallar una transformación T entre los puntos de control de la imagen 2 (xy2) y la imagen 1 (xy1), tal que

$$T(xy2) \cong xy1$$

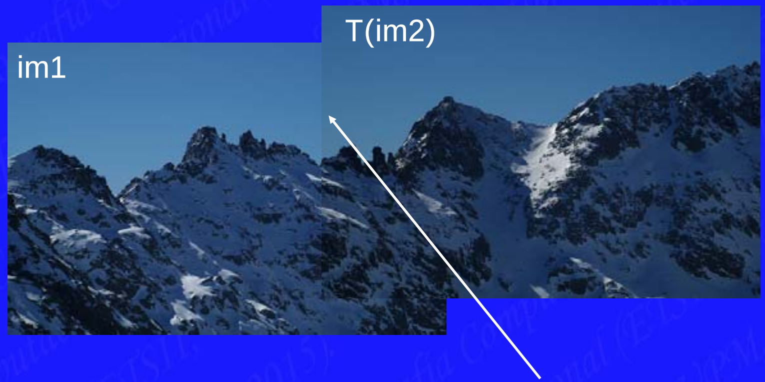
¿ Qué tipo de transformación T usar?

Bajo una serie de supuestos razonables, la relación entre los puntos de control será una homografía (transformación proyectiva).

A veces puede bastarnos con una transformación afín.

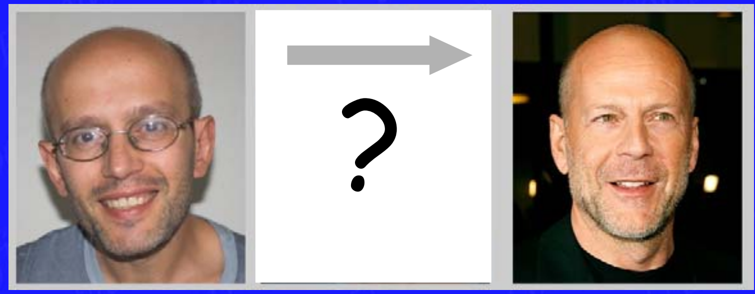
Mosaicos de fotos

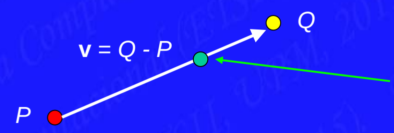
- Aplicar relación encontrada a toda la imagen 2: T(imagen2)



- Fundir im1 + T(im2) para equilibrar cambios de luz, etc.

Transición entre imágenes: morphing





Cualquier punto en la línea entre P y Q:

$X = P + t \cdot v = P + t \cdot (Q - P) = (1-t) \cdot P + t \cdot Q$

con t en [0,1]

P y Q pueden ser cualquier cosa:

puntos en el espacio 2D, colores, imágenes

Aplicación: morphing

PROBLEMA: El objetivo es hallar una “media de dos objetos”, no una media de dos imágenes de los objetos:

im = 0.5*im1 + 0.5*im2

... sino una imagen del objeto promedio.

im1 -> im1’ 0.5im1’ + 0.5im2’ im2’ <- im2

Aplicación: morphing

Habrá que deformar las imágenes para que se solapen.

Pero eso es lo que acabamos de ver

Caso básico: imágenes alineadas



Transicionar desde im1 (t=0) hasta im2 (t=1)

$$im(t) = (1-t)*Image1 + t*image2$$

Trivial si las imágenes está alineadas (cámara fija)

¿Y si no lo están?

Imágenes no alineadas


Solución:

- Alinear imágenes
- Fundido

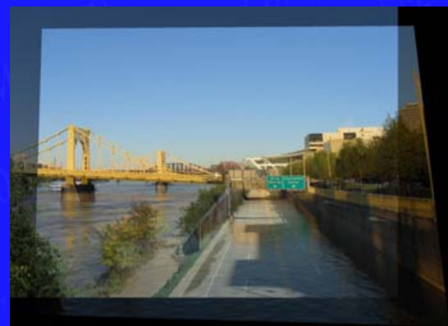
Solución: Alinear imágenes + fundir


T proyectiva

T translación

Una transformación global es adecuada en este caso

Solución: Alinear imágenes + fundir

$t \cong 0$

$t \cong 1$

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

¿Transformación global inadecuada?

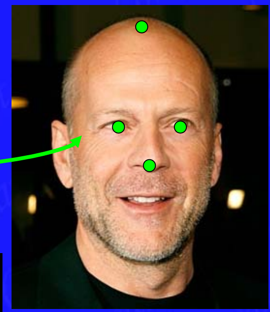
Marcar ojos, ajustar escala horizontal -> sin problemas
Usar T global de escalado que amplia en un pequeño factor.

Marcar ojos y orejas, ajustar escala -> imposible

Ampliar para separar ojos, reducir para juntar orejas.

Warping global no me favorece

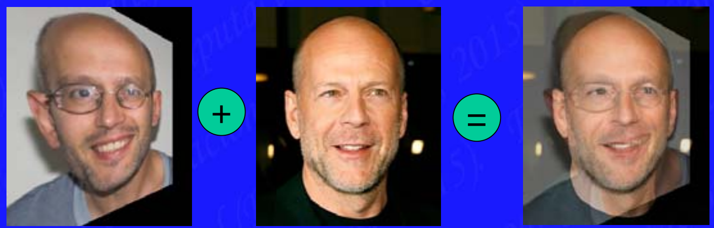
Calculo T para los 4 puntos de control dados.



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 (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)

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)->(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:

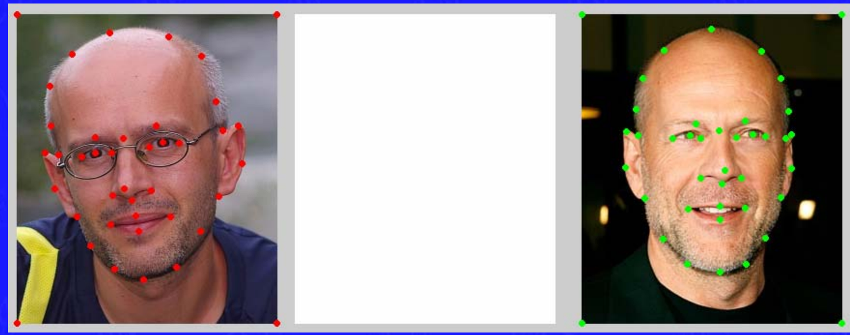
- Sencillas de calcular
- No dan problemas de inestabilidad

¿Cómo especificar warping a trozos?

- Definir unas zonas {Z1, Z2, ..., Zn} y {Z1’, Z2’, ...Zn’ } en las imágenes de origen y destino.
- Asignar una transformación T(k) a cada zona
- Se debe verificar que:
  - Zk = T(k) [Zk’]
  - Las zonas Z1,Z2,... cubran la imagen de partida.
  - Las zonas Z1’, Z2’ deben recubrir la imagen final.
  - Los puntos de control vayan a donde queremos.

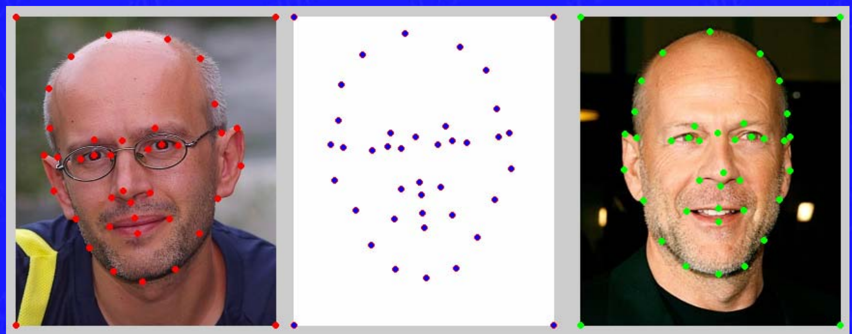
Parece complicado ¿o tal vez no?

Morphing



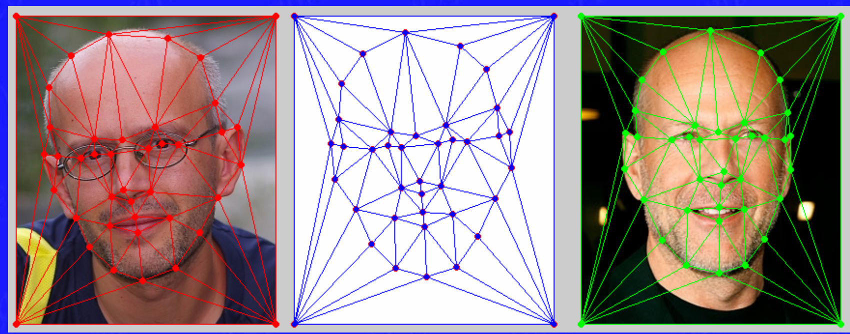
1) Marcamos (muchos) puntos en las dos imágenes, en rasgos comunes.

Morphing



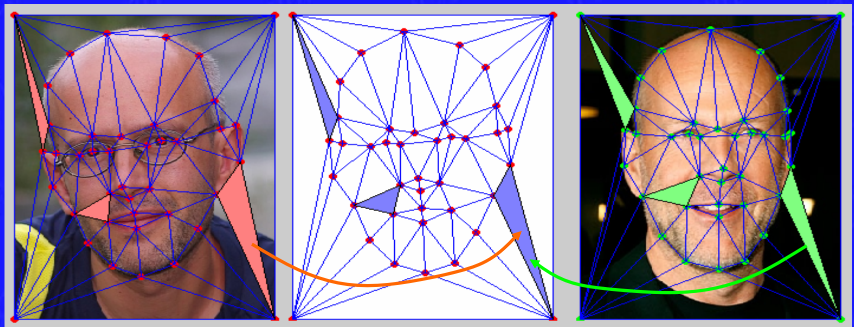
1) Marcamos (muchos) puntos en las dos imágenes, en rasgos comunes.
2) Promediamos los puntos de referencia en una cara “media”

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 una malla a partir de los puntos (por ejemplo, usando el algoritmo de Delaunay)

Morphing



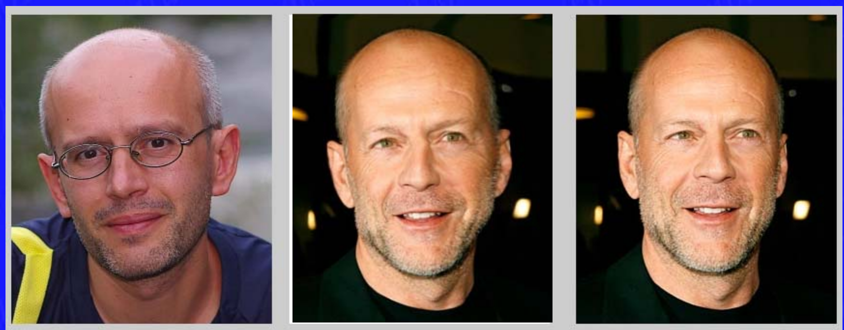
1) Marcamos (muchos) puntos en las dos imágenes, en rasgos comunes.
2) Promediamos los puntos de referencia en una cara “media”.
3) Hacemos una triangulación para crear malla a partir de los puntos.
4) Para todos los triángulos hallamos las transformaciones que nos llevan a la malla media desde ambas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

¿Qué tipo de T usar?

Morphing

4) Deformamos la primera imagen a la posición “media”.

Morphing



4) Deformamos la primera imagen a la posición “media”.
5) Repetimos para la segunda imagen

Morphing

4) Deformamos la primera imagen a la posición “media”.
5) Repetimos para la segunda imagen
6) Mezclamos (0.5 img1 + 0.5 img2) ambas imágenes deformadas.

Secuencia de Morphing

t	img
0	
1	

La transformaciones T1 y T2 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.

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

Secuencia de morphing