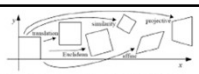
Transformaciones 2D y 3D

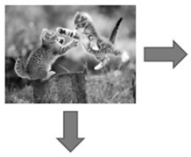
Transformaciones 2D



- Es posible modificar las propiedades de primitivas geométricas (punto, línea, conos, planos) mediante operaciones lineales o no lineales.
  - Utilizando coordenadas homogéneas las transformaciones son operaciones matriciales…Multiplicaciones

Transformaciones 2D

- Translación de puntos



$(x, y) =$ Puntos iniciales
$(x', y') =$ Puntos transformados
$(t_x, t_y) =$ parametros de translación

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

Transformaciones 2D

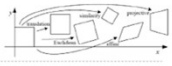
- Translación de puntos – Aspectos prácticos
- dst=cv.warpPerspective(src, M, dsize[, dst[, flags[, borderMode[, borderValue]]]])
  - *Applies a perspective transformation to an image.*

Parameters

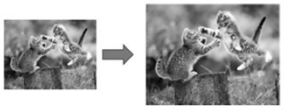
src	input image.
dst	output image that has the size dsize and the same type as src .
M	3×3 transformation matrix.
dsize	size of the output image.
flags	combination of interpolation methods (INTER_LINEAR or INTER_NEAREST) and the optional flag WARP_INVERSE_MAP, that sets M as the inverse transformation (**dst** $\rightarrow$ **src**).
borderMode	pixel extrapolation method (BORDER_CONSTANT or BORDER_REPLICATE).
borderValue	value used in case of a constant border; by default, it equals 0.

- **[warpPerspective()]**
  https://docs.opencv.org/3.4/da/d54/group__imgproc__transform.html#gaf73673a7e8e18ec6963e3774e6a94b87
- Ver Notebook → EX1Trans

Transformaciones 2D



- Escalamiento de puntos



$(x, y) =$ Puntos iniciales
$(x', y') =$ Puntos transformados
$(S_x, S_y) =$ parametros de escalado

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- Si Sx=Sy ….escalamiento uniforme
- Si Sx= 1/Sy o Sy= 1/Sx …se mantiene la relación de aspecto

Transformaciones 2D

- Escalamiento de puntos – Aspectos prácticos
- dst=cv.warpPerspective(src, M, dsize[, dst[, flags[, borderMode[, borderValue]]]])
  - *Applies a perspective transformation to an image.*

Parameters

src	input image.
dst	output image that has the size dsize and the same type as src .
M	3×3 transformation matrix.
dsize	size of the output image.
flags	combination of interpolation methods (INTER_LINEAR or INTER_NEAREST) and the optional flag WARP_INVERSE_MAP, that sets M as the inverse transformation (**dst** $\rightarrow$ **src**).
borderMode	pixel extrapolation method (BORDER_CONSTANT or BORDER_REPLICATE).
borderValue	value used in case of a constant border; by default, it equals 0.

- **[warpPerspective()]**
  https://docs.opencv.org/3.4/da/d54/group__imgproc__transform.html#gaf73673a7e8e18ec6963e3774e6a94b87
- **[resize()]**
  https://docs.opencv.org/3.4/da/d54/group__imgproc__transform.html#ga47a974309e9102f5f08231edc7e7529d
- Ver Notebook → EX2Esc

Transformaciones 2D

- Rotación de puntos

$(x, y) =$ Puntos iniciales

$(x', y') =$ Puntos transformados

- Matriz de rotación → Propiedades geométricas
- Matriz de rotación → Matriz ortonormal

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

$$RR^T = I \quad det(R) = 1$$

Transformaciones 2D

- Rotación de puntos – Aspectos prácticos
- dst=cv.warpPerspective(src, M, dsize[, dst[, flags[, borderMode[, borderValue]]]])
  - *Applies a perspective transformation to an image.*

Parameters

src	input image.
dst	output image that has the size dsize and the same type as src .
M	3 × 3 transformation matrix.
dsize	size of the output image.
flags	combination of interpolation methods (INTER_LINEAR or INTER_NEAREST) and the optional flag WARP_INVERSE_MAP, that sets M as the inverse transformation (**dst** → **src**).
borderMode	pixel extrapolation method (BORDER_CONSTANT or BORDER_REPLICATE).
borderValue	value used in case of a constant border; by default, it equals 0.

- **[warpPerspective()]** https://docs.opencv.org/3.4/da/d54/group__imgproc__transform.html#gaf73673a7e8e18ec6963e3774e6a94b87
- **[getRotationMatrix2D()]** https://docs.opencv.org/3.4/da/d54/group__imgproc__transform.html#gafbbc470ce83812914a70abfb604f4326
- Ver Notebook → EX3Rot

Transformaciones 2D

- Transformaciones inversas: Translación, escalamiento y rotación
  - Dado que son operaciones matriciales y son coordenadas homogeneas….se obtiene mediante la inversa

Transformación	Matriz	Raciocinio
Translación	$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -t_x \\ 0 & 1 & -t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}$	Si un punto se traslada Tx,Ty el efecto inverso es –Tx,-Ty
Escalamiento	$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 1/S_x & 0 & 0 \\ 0 & 1/S_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}$	Si algo se escala por un factor de S el efecto inverso es 1/S
Rotación	$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}$	Debido a las propiedades del SIN y COS en la rotación por un ángulo negativo

Transformaciones 2D

- Transformada de similitud
  - Es la composición de un escalamiento uniforme, rotación y translación

$(x, y) =$ Puntos iniciales

$(x', y') =$ Puntos transformados

$(t_x, t_y) =$ parametros de translación

$(a, b) =$ parametros de similitud

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a & -b & t_x \\ b & a & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

-Esta transformación preserva los ángulos entre líneas formadas por el conjunto de puntos

Transformaciones 2D

- Transformación de similitud – Aspectos prácticos
- dst=cv.warpPerspective(src, M, dsize[, dst[, flags[, borderMode[, borderValue]]]])
  - *Applies a perspective transformation to an image.*

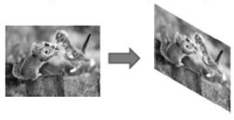
Parameters

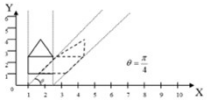
src	input image.
dst	output image that has the size dsize and the same type as src .
M	3 × 3 transformation matrix.
dsize	size of the output image.
flags	combination of interpolation methods (INTER_LINEAR or INTER_NEAREST) and the optional flag WARP_INVERSE_MAP, that sets M as the inverse transformation (**dst** → **src**).
borderMode	pixel extrapolation method (BORDER_CONSTANT or BORDER_REPLICATE)
borderValue	value used in case of a constant border; by default, it equals 0.

- **[warpPerspective()]** https://docs.opencv.org/3.4/da/d54/group__imgproc__transform.html#gaf73673a7e8e18ec6963e3774e6a94b87
- **[getRotationMatrix2D()]** https://docs.opencv.org/3.4/da/d54/group__imgproc__transform.html#gafbbc470ce83812914a70abfb604f4326
- Ver Notebook → EX4Sim

Transformaciones 2D

- Transformada de "*Shear*"
  - Se preservan líneas paralelas pero no los ángulos entre las líneas





Si $b = 0$ y $a = \frac{1}{1/\tan\theta} =$ Skew

-*Skew* solo modifica una coordenada → de cuadrado/rectángulo a paralelogramo

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

Transformaciones 2D

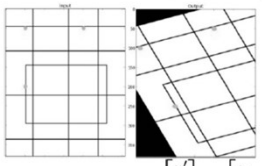
- Transformada afín
  - Las líneas paralelas permanecen paralelas
  - Es la composición de un escalamiento uniforme, *shear*, rotación y translación

- Esta transformación tiene 6 parámetros libres

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Transformaciones 2D

- Transformada afín
  - Las líneas paralelas permanecen paralelas
  - Es la composición de un escalamiento uniforme, *shear*, rotación y translación



- Esta transformación tiene 6 parámetros libres

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Transformaciones 2D

- Transformación afín – Aspectos prácticos
- dst=cv.warpPerspective(src, M, dsize[, dst[, flags[, borderMode[, borderValue]]]])
  - *Applies a perspective transformation to an image.*

Parameters

src	input image.
dst	output image that has the size dsize and the same type as src .
M	3 × 3 transformation matrix.
dsize	size of the output image.
flags	combination of interpolation methods (INTER_LINEAR or INTER_NEAREST) and the optional flag WARP_INVERSE_MAP, that sets M as the inverse transformation (dst ← src).
borderMode	pixel extrapolation method (BORDER_CONSTANT or BORDER_REPLICATE).
borderValue	value used in case of a constant border; by default, it equals 0.

- **[warpPerspective()]**
  https://docs.opencv.org/3.4/da/d54/group__imgproc__transform.html#gaf73673a7e8e18ec6963e3774e6a94b87
- **[getRotationMatrix2D()]**
  https://docs.opencv.org/3.4/da/d54/group__imgproc__transform.html#gafbbc470ce83812914a70abfb604f4326
- Ver Notebook → EX5Afin

Transformaciones 2D

- Transformada proyectivas o homografías
  - Las líneas permanecen como líneas
  - Se utilizan todas las componentes de la matriz 3x3

- Esta transformación tiene 8 parámetros libres

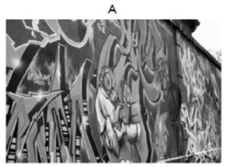
$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ a_7 & a_8 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$x' = u/w$$
$$y' = v/w$$

Transformaciones 2D

- Transformada proyectivas o homografías
  - Una homografía puede usarse para transformar imagen A en la B

A



B

$$x' = u/w \qquad \begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ a_7 & a_8 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$
$$y' = v/w$$

Transformaciones 2D

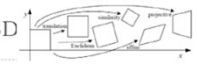
- Transformación de homografía – Aspectos prácticos
- dst=cv.warpPerspective(src, M, dsize[, dst[, flags[, borderMode[, borderValue]]]])
  - *Applies a perspective transformation to an image.*

Parameters

src	input image.
dst	output image that has the size dsize and the same type as src .
M	3 × 3 transformation matrix.
dsize	size of the output image.
flags	combination of interpolation methods (INTER_LINEAR or INTER_NEAREST) and the optional flag WARP_INVERSE_MAP, that sets M as the inverse transformation (dst ← src).
borderMode	pixel extrapolation method (BORDER_CONSTANT or BORDER_REPLICATE).
borderValue	value used in case of a constant border; by default, it equals 0.

- **[warpPerspective()]**
  https://docs.opencv.org/3.4/da/d54/group__imgproc__transform.html#gaf73673a7e8e18ec6963e3774e6a94b87
- **[getRotationMatrix2D()]**
  https://docs.opencv.org/3.4/da/d54/group__imgproc__transform.html#gafbbc470ce83812914a70abfb604f4326
- Ver Notebook → EX6Hom

Transformaciones 2D y 3D



- Transformadas 2D
  - La preservación de propiedades se reduce al bajar en la lista. I.e. La transformada afín preserva las líneas paralalias y las líneas.

Transformation	Matrix	# DoF	Preserves	Icon
translation	$[\ \boldsymbol{I}\ \vert\ \boldsymbol{t}\ ]_{2\times3}$	2	orientation	
rigid (Euclidean)	$[\ \boldsymbol{R}\ \vert\ \boldsymbol{t}\ ]_{2\times3}$	3	lengths	
similarity	$[\ s\boldsymbol{R}\ \vert\ \boldsymbol{t}\ ]_{2\times3}$	4	angles	
affine	$[\ \boldsymbol{A}\ ]_{2\times3}$	6	parallelism	
projective	$\left[\ \tilde{\boldsymbol{H}}\ \right]_{3\times3}$	8	straight lines	

Transformaciones 2D y 3D

- Transformadas 3D
  - La preservación de propiedades se reduce al bajar en la lista. I.e. La transformada afín preserva las líneas paralalias y las líneas.
  - Existen diferentes métodos de parametrización de rotaciones en 3D!!!

Transformation	Matrix	# DoF	Preserves	Icon
translation	$[\ \boldsymbol{I}\ \vert\ \boldsymbol{t}\ ]_{3\times4}$	3	orientation	
rigid (Euclidean)	$[\ \boldsymbol{R}\ \vert\ \boldsymbol{t}\ ]_{3\times4}$	6	lengths	
similarity	$[\ s\boldsymbol{R}\ \vert\ \boldsymbol{t}\ ]_{3\times4}$	7	angles	
affine	$[\ \boldsymbol{A}\ ]_{3\times4}$	12	parallelism	
projective	$\left[\ \tilde{\boldsymbol{H}}\ \right]_{4\times4}$	15	straight lines	