Transformaciones en imágenes

Gastón Araguás

Centro de Investigación en Informática para la Ingeniería
Universidad Tecnológica Nacional, F.R.C.
http://ciii.frc.utn.edu.ar
Córdoba, Argentina

26 de octubre de 2013

Coordenadas homogeneas

Puntos y rectas

Un punto $(x, y) \in \mathbb{R}^2$ puede representarse por el vector $\boldsymbol{x}$

$$\boldsymbol{x} = \begin{bmatrix} x \\ y \end{bmatrix} = [x, y]^T$$

Una recta en el plano $ax + by + c = 0$ puede representarse por el vector $\boldsymbol{l}$

$$\boldsymbol{l} = \begin{bmatrix} a \\ b \\ c \end{bmatrix} = [a, b, c]^T$$

pero $\boldsymbol{l} = [ka, kb, kc]^T$, con $k \neq 0$, representa la misma recta.

- Una recta tienen 2 DOF, dado por los cocientes de sus componentes $\{a : b : c\}$.
- Estos vectores equivalentes se conocen como vectores homogeneos.
- El conjunto de vectores homogeneos en $\mathbb{R}^3$, menos el vector $[0, 0, 0]^T$, forman el espacio proyectivo $\mathbb{P}^2$.

Coordenadas homogeneas

Puntos y rectas

Un punto (x, y) pertenece a la recta $\boldsymbol{l}$ si $ax + by + c = 0$, o bien,

$$[x, y, 1][a, b, c]^T = 0$$

El vector $[x, y, 1]^T$ es la representación del punto (x, y) en coordenadas homogeneas.

Punto en coordenadas homogeneas

Un punto en el plano de coordenadas finitas $(\frac{x_1}{x_3}, \frac{x_2}{x_3})$, (con $x_3 \neq 0$,) se representa en coordenadas homogeneas por el vector $\boldsymbol{x} \in \mathbb{P}^2$ dado por

$$\boldsymbol{x} = [x_1, x_2, x_3]^T$$

Nota: Un punto también tiene dos DOF, dados por $\{x_1 : x_2 : x_3\}$

Coordenadas homogeneas

Rectas y puntos

Por lo anterior . . .

Punto en una recta

Sean el punto $\boldsymbol{x} \in \mathbb{P}^2$ y la recta $\boldsymbol{l} \in \mathbb{P}^2$, entonces $\boldsymbol{x}$ está en $\boldsymbol{l}$ si

$$\boldsymbol{x}^T \boldsymbol{l} = \boldsymbol{l}^T \boldsymbol{x} = 0$$

Observación: El producto escalar entre $\boldsymbol{x}$ y $\boldsymbol{l}$ también es nulo, ya que

$$\boldsymbol{x}.\boldsymbol{l} = ax + by + c = 0$$

por lo que los vectores son perpendiculares.

Coordenadas homogeneas

Rectas y puntos

Intersección de dos rectas

Sean $\boldsymbol{l}_1$ y $\boldsymbol{l}_2$ dos rectas en $\mathbb{P}^2$. Definiendo $\boldsymbol{x} = \boldsymbol{l}_1 \times \boldsymbol{l}_2$, que es un vector perpendicular a ambas rectas, tal que el producto escalar

$$\boldsymbol{l}_1.(\boldsymbol{l}_1 \times \boldsymbol{l}_2) = \boldsymbol{l}_2.(\boldsymbol{l}_1 \times \boldsymbol{l}_2) = 0$$

es decir

$$\boldsymbol{l}_1^T \boldsymbol{x} = \boldsymbol{l}_2^T \boldsymbol{x} = 0$$

luego, si $\boldsymbol{x}$ representa a un punto, este es el punto de intersección de $\boldsymbol{l}_1$ y $\boldsymbol{l}_2$.

Coordenadas homogeneas

Rectas y puntos

Recta por dos puntos

Sean $\boldsymbol{x}_1$ y $\boldsymbol{x}_2$ dos puntos en $\mathbb{P}^2$. Definiendo $\boldsymbol{l} = \boldsymbol{x}_1 \times \boldsymbol{x}_2$ se tiene

$$\boldsymbol{l}^T \boldsymbol{x}_1 = \boldsymbol{l}^T \boldsymbol{x}_2 = 0$$

luego, el vector $\boldsymbol{x}_1 \times \boldsymbol{x}_2$ representa a la recta $\boldsymbol{l}$ que pasa por los puntos $\boldsymbol{x}_1$ y $\boldsymbol{x}_2$.

Dualidad

Debido a la representación dual de rectas y puntos, los enunciados tiene siempre su forma dual.

En este caso la recta que pasa por dos puntos es dual al anterior, que puede leerse como “el punto que pasa por dos rectas”.

Coordenadas homogeneas

Rectas paralelas - Punto en el infinito

Que pasa con las paralelas?

Intersección de rectas paralelas

Sean $\boldsymbol{l} = [a, b, c]^T$ y $\boldsymbol{l}' = [a, b, c']^T$ dos rectas paralelas en $\mathbb{P}^2$, la intersección de las rectas $\boldsymbol{l}$ y $\boldsymbol{l}'$ será

$$\boldsymbol{x} = \boldsymbol{l} \times \boldsymbol{l}' = [b, -a, 0]^T$$

con $\boldsymbol{x}$ un punto ideal o punto en el infinito.

Punto en el infinito

Un punto en $\mathbb{P}^2$ de coordenadas

$$\boldsymbol{x} = [x_1, x_2, 0]^T$$

no representa ningún punto finito en el plano, se le llama punto ideal o punto en el infinito.

Coordenadas homogeneas

Recta en el infinito

Todo punto ideal $\boldsymbol{x} = [x_1, x_2, 0]^T$ pertenece a la recta $\boldsymbol{l}_\infty = [0, 0, 1]^T$, ya que

$$\boldsymbol{x}^T \boldsymbol{l}_\infty = 0$$

$\boldsymbol{l}_\infty$ se llama recta en el infinito.

Intersección con l_∞

La intersección de las rectas paralelas $\boldsymbol{l}$ y $\boldsymbol{l}'$ con $\boldsymbol{l}_\infty$ es en el punto ideal $\boldsymbol{x} = [b, -a, 0]^T$

$$[b, -a, 0]^T \boldsymbol{l} = [b, -a, 0]^T \boldsymbol{l}' = [b, -a, 0]^T \boldsymbol{l}_\infty = 0$$

- Notar que el vector $[b, -a]^T \in \mathbb{R}^2$ representa la dirección de la recta $\boldsymbol{l} = [a, b, c]^T$. Por lo tanto

$$\boldsymbol{l}_\infty : \{\text{Conjunto de direcciones de rectas de } \mathbb{R}^2\}$$

Coordenadas homogeneas

Cónicas

Una cónica en $\mathbb{R}^2$ es de la forma: $ax^2 + bxy + cy^2 + dx + ey + f = 0$, en coordenadas homogeneas

$$x \to \frac{x_1}{x_3}; y \to \frac{x_2}{x_3} \Rightarrow$$

$$ax_1^2 + bx_1x_2 + cx_2^2 + dx_1x_3 + ex_2x_3 + fx_3^2 = 0$$

En forma de producto matricial

Cónica en coordenadas homogeneas

Sea el punto $\boldsymbol{x} \in \mathbb{P}^2$, $\boldsymbol{x}$ está en la cónica C si

$$\boldsymbol{x}^T C \boldsymbol{x} = 0 \quad \text{con} \quad C = \begin{bmatrix} a & b/2 & d/2 \\ b/2 & c & e/2 \\ d/2 & e/2 & f \end{bmatrix}$$

una matriz en $\mathbb{P}^2$ que representa la cónica C

Nota: Una cónica tiene 5 DOF, $\{a : b : c : d : e : f\}$.

Coordenadas homogeneas

Rectas y cónicas

Recta tangente a la cónica

Sea la recta $\boldsymbol{l} \in \mathbb{P}^2$ definida como $\boldsymbol{l} = C\boldsymbol{x}$, si el punto $\boldsymbol{x}$ pertenece a la cónica, pertenece también a $\boldsymbol{l}$

$$\boldsymbol{x}^T\boldsymbol{l} = \boldsymbol{x}^T C\boldsymbol{x} = 0$$

y además $\boldsymbol{l}$ corta a la cónica C en ese solo punto.

Si otro punto $\boldsymbol{y}$ de $\boldsymbol{l}$ pertenece también a C

$$\boldsymbol{y}^T\boldsymbol{l} = \boldsymbol{y}^T C\boldsymbol{x} = \boldsymbol{y}^T C\boldsymbol{y} = 0$$

pero entonces todo punto de forma $(\boldsymbol{x} + \alpha\boldsymbol{y})$ que pertenece a $\boldsymbol{l}$ también pertenece a C

$$(\boldsymbol{x} + \alpha\boldsymbol{y})^T C(\boldsymbol{x} + \alpha\boldsymbol{y}) = 0$$

y toda la recta pretenece a C. Se dice que C es una cónica degenerada.

Coordenadas homogeneas

Cónica dual

Una cónica definida como $\boldsymbol{x}^T C \boldsymbol{x} = 0$ se llama cónica de puntos. También puede definirse en términos de las rectas tangentes cuando C es invertible.

Cónica de rectas

Sea $\boldsymbol{l} = C\boldsymbol{x}$, luego $\boldsymbol{x} = C^{-1}\boldsymbol{l}$, entonces

$$\begin{aligned} \boldsymbol{x}^T C \boldsymbol{x} = 0 \Rightarrow (C^{-1}\boldsymbol{l})^T C (C^{-1}\boldsymbol{l}) &= 0 \\ (\boldsymbol{l}^T C^{-T}) C (C^{-1}\boldsymbol{l}) &= 0 \\ \boldsymbol{l}^T C^{-1} \boldsymbol{l} &= 0 \end{aligned}$$

donde por su simetría $C^{-T} = C^{-1}$.

$\boldsymbol{l}^T C^{-1} \boldsymbol{l} = 0$ se llama cónica de rectas.

Transformaciones proyectivas

Definición

Definición

Una proyectividad es un mapa invertible h de $\mathbb{P}^2$ en $\mathbb{P}^2$ que lleva un conjunto de puntos $\{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n\}$ pertenecientes a una recta a otro conjunto $\{h(\boldsymbol{x}_1), \cdots, h(\boldsymbol{x}_n)\}$ pertenecientes también a una recta.

es decir mapea rectas en rectas.

- Proyectividad, transformación proyectiva, homografía, colinealidad, son sinónimos.
- Las homografías forman un grupo, ya que su inversa también es una homografía, como también la composición de homografías es otra homografía.

Transformaciones proyectivas

Definición

En términos algebraicos

Definición

Un mapa lineal $h : \mathbb{P}^2 \to \mathbb{P}^2$ es una proyectividad si y sólo si existe una matriz H 3×3 invertible tal que

$$h(\boldsymbol{x}) = H\boldsymbol{x} \qquad \forall \boldsymbol{x} \in \mathbb{P}^2$$

Prueba 2: Sean $\boldsymbol{x}_i, i = 1, \cdots, n$ puntos en $\boldsymbol{l}$ tal que $\boldsymbol{l}^T \boldsymbol{x}_i = 0$. Sea H una matriz 3×3 invertible, se verifica que

$$\boldsymbol{l}^T H^{-1} H \boldsymbol{x}_i = (H^{-T} \boldsymbol{l})^T H \boldsymbol{x}_i = 0$$

todos los puntos $H\boldsymbol{x}_i$ pertenecen a la recta $\boldsymbol{l}' = H^{-T}\boldsymbol{l}$.

Transformaciones proyectivas

Puntos

Transformación de puntos

Una homografía mapea el punto $\boldsymbol{x} \rightarrow \boldsymbol{x}'$ mediante

$$\boldsymbol{x}' = H\boldsymbol{x}$$

- La proyección no cambia si se multiplica H por cualquier escalar distinto de cero, H está definida hasta un factor de escala.
- Por lo tanto H tiene 8 DOF.
- H es una matriz homogenea.

Transformaciones proyectivas

Rectas

Transformación de rectas

De la prueba dada en la definición de homografía

$$\boldsymbol{l}^T H^{-1} H\boldsymbol{x} = (H^{-T}\boldsymbol{l})^T H\boldsymbol{x} = 0$$
$$(\boldsymbol{l}')^T \boldsymbol{x}' = 0$$

y la recta $\boldsymbol{l}$ es mapeada a $\boldsymbol{l}' = H^{-T}\boldsymbol{l}$.

Transformaciones proyectivas

Cónicas

Transformación de cónicas

Un punto en una cónica satisface $\boldsymbol{x}^T C \boldsymbol{x} = 0$, de la transformación $\boldsymbol{x} = H^{-1}\boldsymbol{x}'$, luego

$$\begin{aligned}(H^{-1}\boldsymbol{x}')^T C (H^{-1}\boldsymbol{x}') &= 0\\ \boldsymbol{x}'^T H^{-T} C H^{-1}\boldsymbol{x}' &= 0\\ \boldsymbol{x}'^T C' \boldsymbol{x}' &= 0\end{aligned}$$

con $C' = H^{-T} C H^{-1}$.

Jerarquía de transformaciones

Isometría

Isometría

Una isometría es una transformación que preserva distancia Euclidea.

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

con $\varepsilon = \pm 1$.

Si $\varepsilon = 1$, se llama transformación Euclidea, que además

- preserva orientación
- transformación del cuerpo rígido.

Jerarquía de transformaciones

Isometría

Transformación Euclidea

$$\boldsymbol{x}' = H_E \boldsymbol{x} = \begin{bmatrix} R & \boldsymbol{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \boldsymbol{x}$$

con R una matriz de rotación de 2×2 ($RR^T = R^T R = I$), $\mathbf{0}^T = (0, 0)^T$ y $\boldsymbol{t}$ un vector de traslación.

- H_E tiene 3 DOF, uno de la rotación y dos de la traslación.

Invariantes

Cantidades que se preservan en una transformación. Longitudes, ángulos y áreas son invariantes de H_E.

Jerarquía de transformaciones

Similaridad

Similaridad

Una similaridad es una isometría con escalado isotrópico.

$$\boldsymbol{x}' = H_S \boldsymbol{x} = \begin{bmatrix} sR & \boldsymbol{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \boldsymbol{x}$$

- H_S tiene 4 DOF, uno de la rotación, dos de la traslación y el escalado s.
- Esta transformación queda definida mediante un par de puntos correspondientes $\boldsymbol{x}' \leftrightarrow \boldsymbol{x}$.

Invariantes

Ángulos entre rectas, paralelismo, relación entre áreas son invariantes de H_S.

Jerarquía de transformaciones

Transformación afín

Transformación afín

Es una transformación no singular (invertible) seguida de una traslación

$$\boldsymbol{x}' = H_A \boldsymbol{x} = \begin{bmatrix} A & \boldsymbol{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \boldsymbol{x}$$

con A una matriz no singular.

- H_A tiene 6 DOF y puede ser recuperada con 3 puntos correspondientes $\boldsymbol{x}' \leftrightarrow \boldsymbol{x}$.

Jerarquía de transformaciones

Transformación afín

Que hace A?

Descomposición SVD

Puede verse mas claramente la acción de A a partir de su descomposición SVD

$$A = UDV = (UV)V^{-1}DV$$

$$A = R(\theta)R(-\varphi)DR(\varphi); \quad \text{con } D = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

λ_1 y λ_2 son los valores singulares de A.

- Si $\det A > 0$, la transformación preserva dirección.
- El escalado es no isotrópico pero actúa en direcciones otrogonales.

Jerarquía de transformaciones

Transformación afín

Invariantes

Paralelismo: Dos rectas paralelas intersectan a $\boldsymbol{l}_\infty$ en el punto $[x_1, x_2, 0]^T$, luego de la transformación

$$\boldsymbol{x}' = \begin{bmatrix} A & \boldsymbol{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \boldsymbol{x} = [x'_1, x'_2, 0]$$

Razón entre longitudes de segmentos paralelos: El escalado es común a rectas de igual dirección.
Razón entre áreas: Las áreas son todas escaladas una cantidad $\lambda_1\lambda_2$.

Jerarquía de transformaciones

Transformación proyectiva

Transformación proyectiva

Es una transformación general invertible de forma

$$\boldsymbol{x}' = H_P \boldsymbol{x} = \begin{bmatrix} A & \boldsymbol{t} \\ \boldsymbol{v}^T & v \end{bmatrix} \boldsymbol{x}$$

con $\boldsymbol{v}^T = [v_1, v_2]$ y v un escalar.

- H_P tiene 8 DOF (los 9 elementos menos la escala) y puede computarse con 4 puntos correspondientes (3 deben ser no colineales).
- Mapea puntos ideales en puntos finitos

$$\begin{bmatrix} A & \boldsymbol{t} \\ \boldsymbol{v}^T & v \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ 0 \end{bmatrix} = \begin{bmatrix} A \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \\ v_1 x_1 + v_2 x_2 \end{bmatrix}$$

Jerarquía de transformaciones

Transformación proyectiva

Descomposición

Cualquier H puede obtenerse componiendo las transformaciones anteriores

$$H = H_P H_A H_S$$

donde H_S tiene 4 DOF, H_A tiene 2 DOF mas y H_P 2 DOF mas, para hacer un total de 8 DOF.

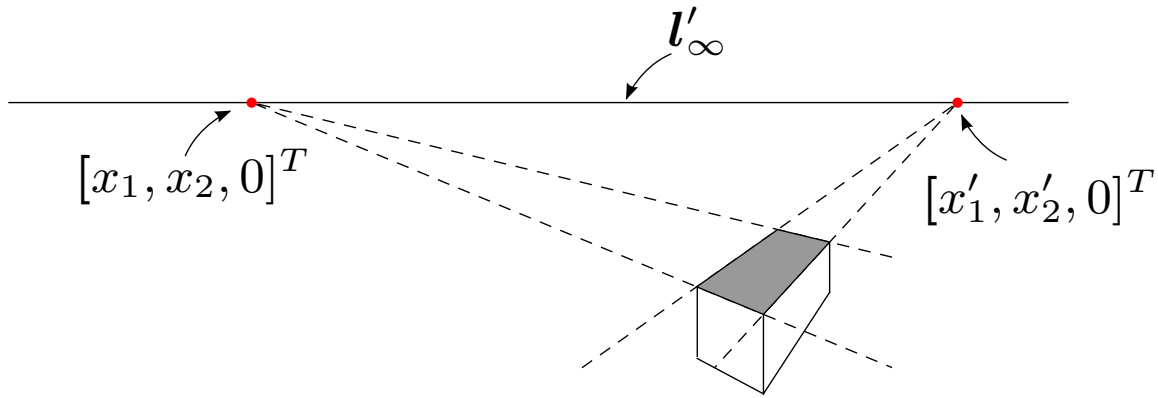
Conociendo donde fue mapeado algún invariante de un espacio, pueden recuperarse las propiedades correspondientes.

Estimación de H

Rectificación

Ejemplo: conociendo donde fue mapeada $\boldsymbol{l}_\infty$ (dos DOF) luego de una transformación proyectiva, se pueden recuperar las propiedades afín.

Sea $\boldsymbol{l}'_\infty = [l_1, l_2, l_3]^T$ la imágen de $\boldsymbol{l}_\infty = [0, 0, 1]^T$.



Luego si $l_3 \neq 0$ se puede elegir H_{PA} tal que $H_{PA}^{-T}\boldsymbol{l}'_\infty = \boldsymbol{l}_\infty$

$$H_{PA}^{-T} = \begin{bmatrix} 1 & 0 & -\frac{l_1}{l_3} \\ 0 & 1 & -\frac{l_2}{l_3} \\ 0 & 0 & \frac{1}{l_3} \end{bmatrix} \quad \Rightarrow \quad H_{PA} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ l_1 & l_2 & l_3 \end{bmatrix}$$

Aplicando H_{PA} a todos los puntos se realiza una “rectificación afín”

Estimación de H

Rectificación

Si se conocen 4 pares de puntos correspondientes, la rectificación puede ser completa (8 DOF).

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \Rightarrow \begin{array}{l} y'_1 = \frac{y_1}{y_3} = \frac{h_{11}x_1+h_{12}x_2+h_{13}x_3}{h_{31}x_1+h_{32}x_2+h_{33}x_3} \\ \\ y'_2 = \frac{y_2}{y_3} = \frac{h_{21}x_1+h_{22}x_2+h_{23}x_3}{h_{31}x_1+h_{32}x_2+h_{33}x_3} \end{array}$$

y cada punto genera dos ecuaciones

$$\begin{aligned} y'_1(h_{31}x_1 + h_{32}x_2 + h_{33}x_3) &= h_{11}x_1 + h_{12}x_2 + h_{13}x_3 \\ y'_2(h_{31}x_1 + h_{32}x_2 + h_{33}x_3) &= h_{21}x_1 + h_{22}x_2 + h_{23}x_3 \end{aligned}$$

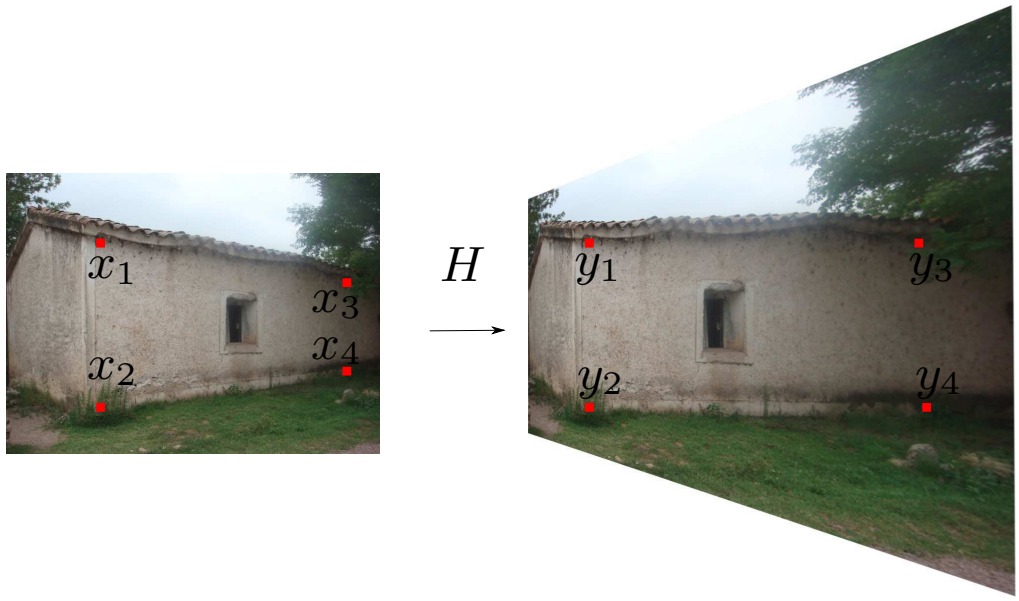
Usando notación matricial

$$\begin{bmatrix} x_1 & x_2 & x_3 & 0 & 0 & 0 & -x_1y'_1 & -x_2y'_1 & -x_3y'_1 \\ 0 & 0 & 0 & x_1 & x_2 & x_3 & -x_1y'_2 & -x_2y'_2 & -x_3y'_2 \end{bmatrix} [\boldsymbol{h}] = 0$$

Estimación de H

Rectificación

Ejemplo:



H mapea $\begin{matrix} \boldsymbol{x}_1 = [35, 80, 1]^T \\ \boldsymbol{x}_2 = [35, 16, 1]^T \\ \boldsymbol{x}_3 = [131, 65, 1]^T \\ \boldsymbol{x}_4 = [131, 30, 1]^T \end{matrix}$ en $\begin{matrix} \boldsymbol{y}_1 = [35, 80, 1]^T \\ \boldsymbol{y}_2 = [35, 16, 1]^T \\ \boldsymbol{y}_3 = [153, 80, 1]^T \\ \boldsymbol{y}_4 = [153, 16, 1]^T \end{matrix}$ es decir $\boldsymbol{y}_i = H\boldsymbol{x}_i$

Estimación de H

Algorítmo DLT

Otra forma (numéricamente más estable)

Algorítmo DLT

$\boldsymbol{x}'$ y $H\boldsymbol{x}$ son vectores con la misma dirección, por ende

$$\boldsymbol{x}' \times H\boldsymbol{x} = 0$$

explícitamente

$$\begin{bmatrix} x'_1 \\ x'_2 \\ x'_3 \end{bmatrix} \times \begin{bmatrix} \boldsymbol{h}1^T\boldsymbol{x} \\ \boldsymbol{h}2^T\boldsymbol{x} \\ \boldsymbol{h}3^T\boldsymbol{x} \end{bmatrix} = \begin{bmatrix} x'_2\boldsymbol{h}_3^T\boldsymbol{x} - x'_3\boldsymbol{h}_2^T\boldsymbol{x} \\ x'_3\boldsymbol{h}_1^T\boldsymbol{x} - x'_1\boldsymbol{h}_3^T\boldsymbol{x} \\ x'_1\boldsymbol{h}_2^T\boldsymbol{x} - x'_2\boldsymbol{h}_1^T\boldsymbol{x} \end{bmatrix} = 0$$

donde $\boldsymbol{h}i^T$ son los vectores fila de H.

Estimación de H

Algorítmo DLT

Algorítmo DLT, continuación

Cambiando $\boldsymbol{h}i^T\boldsymbol{x} = \boldsymbol{x}^T\boldsymbol{h}i$ y completando

$$\begin{bmatrix} \mathbf{0}^T & -x'_3\boldsymbol{x}^T & x'_2\boldsymbol{x}^T \\ x'_3\boldsymbol{x}^T & \mathbf{0}^T & -x'_1\boldsymbol{x}^T \\ -x'_2\boldsymbol{x}^T & x'_1\boldsymbol{x}^T & \mathbf{0}^T \end{bmatrix} \begin{bmatrix} \boldsymbol{h}1 \\ \boldsymbol{h}2 \\ \boldsymbol{h}3 \end{bmatrix} = \tilde{A}\boldsymbol{h} = 0$$

con $\tilde{A}$ una matriz 3×9 obtenida de cada correspondencia.
La matriz $\tilde{A}$ es de rango 2, sólo dos de sus filas son útiles para el computo de H, generalmente se usan las dos primeras.

$$\begin{bmatrix} \mathbf{0}^T & -x'_3\boldsymbol{x}^T & x'_2\boldsymbol{x}^T \\ x'_3\boldsymbol{x}^T & \mathbf{0}^T & -x'_1\boldsymbol{x}^T \end{bmatrix} \begin{bmatrix} \boldsymbol{h}1 \\ \boldsymbol{h}2 \\ \boldsymbol{h}3 \end{bmatrix} = \hat{A}\boldsymbol{h} = 0$$

con $\hat{A}$ una matriz de 2×9 por cada par correspondiente. Con 4 pares se resuelve hasta un factor de escala.

Estimación de H

Estimación con ruido

Sean $x_i' \leftrightarrow x_i$ un conjunto de correspondencia con ruido Gaussiano,

Algorítmo DLT con ruido

Considerando $n > 4$ pares de correspondencias se llega al sistema sobredeterminado que se desea resolver

$$A\boldsymbol{h} = \varepsilon \tag{1}$$

con A una matriz de $2n \times 9$

- Si los puntos correspondientes son exactos, ε en (1) será nulo, sino se busca $\boldsymbol{h}$ que mejor aproxima a 0.
- Minimizar (1) usando mínimos cuadrados: encontrar $\boldsymbol{h}$ que minimice $||A\boldsymbol{h}||$ sujeto a $||\boldsymbol{h}|| = 1$.

Estimación de H

Algorítmo DLT

Mínimos cuadrados usando SVD

Sea $A = UDV^T$ se debe minimizar $||UDV^T\boldsymbol{h}||$. Como U y V^T preservan norma

$$||UDV^T\boldsymbol{h}|| = ||DV^T\boldsymbol{h}||$$

y

$$||\boldsymbol{h}|| = ||V^T\boldsymbol{h}|| = 1$$

definiendo $\boldsymbol{k} = V^T\boldsymbol{h}$, el problema es ahora minimizar $||D\boldsymbol{k}||$ sujeto a $||\boldsymbol{k}|| = 1$

Luego, como D es una matriz diagonal (con sus elementos ordenados de mayor a menor), la solución será $\boldsymbol{k} = [0, 0, \cdots, 1]^T$.

Finalmente,

$$\boldsymbol{h} = V\boldsymbol{k}$$

es la última columna de V, el vector singular correspondiente al menor valor singular de A.

Estimación de H

Distancia algebraica

Distancia algebraica

El algorítmo DLT minimiza distancia algebraica

$$\sum_i d_{alg}(\boldsymbol{x}'_i, H\boldsymbol{x}_i)^2 = ||\boldsymbol{\varepsilon}||^2 = ||A\boldsymbol{h}||^2$$

Tiene las ventajas

- es lineal (en h)
- tiene solución única
- es barato

pero

- se minimiza un parámetro que no siempre tiene sentido geométrico o estadístico
- requiere normalización (o acondicionamiento numérico), no es invariante a similaridades.

Estimación de H

Distancia geométrica

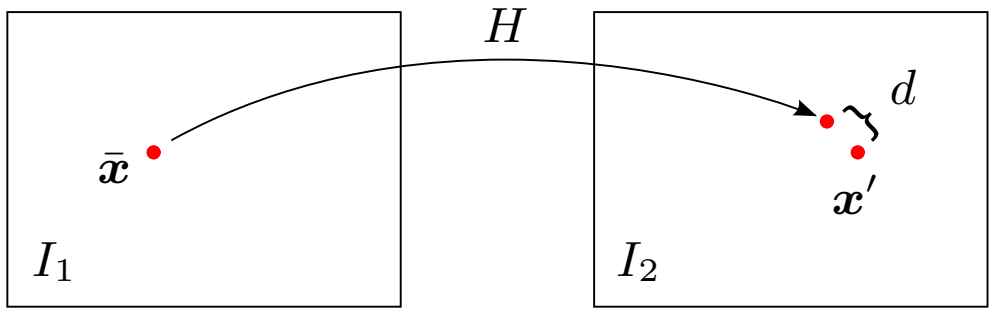
Se basa en la distancia Euclidea $d(\boldsymbol{x}, \boldsymbol{y})$ entre puntos medidos y proyectados.

Error de transferencia

Supone ruido en una sola de las imágenes

$$\sum_i d(\boldsymbol{x}'_i, H\bar{\boldsymbol{x}}_i)^2$$

con $\bar{\boldsymbol{x}}$ el punto medido sin ruido sobre la imagen 1.



Estimación de H

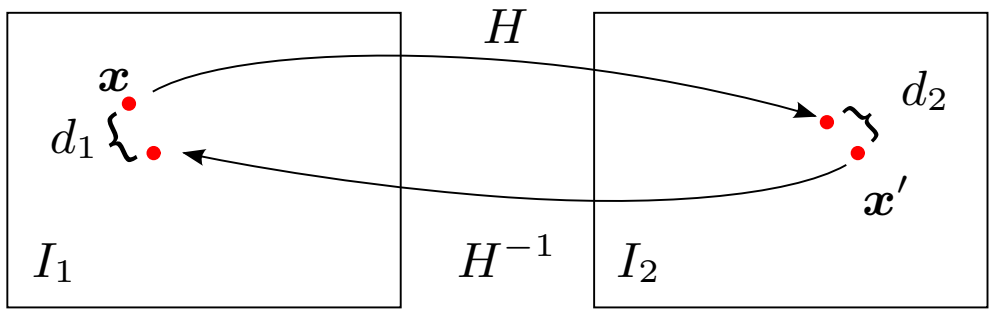
Distancia geométrica

Error de transferencia simétrico

Asume ruido en ambas imágenes

$$\sum_i d_1(\boldsymbol{x}_i, H^{-1}\boldsymbol{x}'_i)^2 + \sum_i d_2(\boldsymbol{x}'_i, H\boldsymbol{x}_i)^2$$

es la suma el error de transferencia en las dos imágenes.



Estimación de H

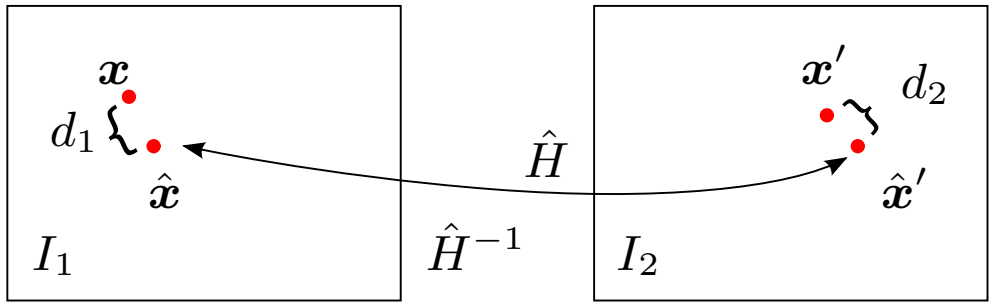
Distancia geométrica

Error de reproyección

Asume ruido en ambas imágenes

$$\sum_i d_1(\boldsymbol{x}_i, \hat{\boldsymbol{x}}'_i)^2 + \sum_i d_2(\boldsymbol{x}'_i, \hat{\boldsymbol{x}}_i)^2$$

es la distancia a un punto intermedio $\hat{\boldsymbol{x}}'_i = \hat{H}\hat{\boldsymbol{x}}_i$.



En un espacio afín $d_{alg}(\boldsymbol{x}, \boldsymbol{x}') = d(\boldsymbol{x}, \boldsymbol{x}')$, por ende puede usarse DLT para minimizar la distancia geométrica.

Estimación de H

Funcional estadístico - MLE

Probabilidad de $\{x'_i\}$ dada H

Asumiendo ruido gaussiano independiente en la medición de cada coordenada de x' (una imagen), y que existe H tal que $x' = H\bar{x} + \rho$

$$Pr(x'|H) = \left(\frac{1}{2\pi\sigma}\right) e^{-d(x', H\bar{x})^2/(2\sigma^2)}$$

luego, por la independencia en las mediciones, la probabilidad del conjunto es

$$Pr(\{x'_i\}|H) = \prod_i \left(\frac{1}{2\pi\sigma}\right) e^{-d(x'_i, H\bar{x}_i)^2/(2\sigma^2)}$$

Se conoce como función de similitud o “likelihood”.

Estimación de H

Funcional estadístico - MLE

Maximum Likelihood Estimation

Tomando $log()$ se obtiene "log-likelihood"

$$log\left(Pr(\{\boldsymbol{x}'_i\}|H)\right) = -\frac{1}{2\sigma^2}\sum_i d(\boldsymbol{x}'_i, H\bar{\boldsymbol{x}}_i)^2 + const.$$

y la transformación $\hat{H}$ de máxima similitud con la verdadera H será la que maximice "log-likelihood", o la que minimice

$$\sum_i d(\boldsymbol{x}'_i, H\bar{\boldsymbol{x}}_i)^2$$

que es exactamente igual a minimizar el error de transferencia.

Estimación de H

Funcional estadístico - MLE

Maximum Likelihood Estimation

Si se asume error en ambas imágenes y una transformación verdadera tal que $\{\bar{\boldsymbol{x}}'_i = H\bar{\boldsymbol{x}}_i\}$

$$Pr(\{\boldsymbol{x}_i, \boldsymbol{x}'_i\}|H, \{\bar{\boldsymbol{x}}_i\}) = \prod_i \left(\frac{1}{2\pi\sigma}\right) e^{-\left(d(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_i)^2 + d(\boldsymbol{x}'_i, H\bar{\boldsymbol{x}}_i)^2\right)/(2\sigma^2)}$$

El conjunto $\{\bar{\boldsymbol{x}}_i\}$ de puntos sin error es reemplazado por un conjunto de puntos "corregidos" $\{\hat{\boldsymbol{x}}_i\}$ como en el cálculo del error de reproyección $\hat{\boldsymbol{x}}'_i = \hat{H}\hat{\boldsymbol{x}}_i$.

$$\sum_i d(\boldsymbol{x}_i, \hat{\boldsymbol{x}}_i)^2 + d(\boldsymbol{x}'_i, \hat{\boldsymbol{x}}'_i)^2$$

Esta minimización es igual al error de reproyección.

Estimación de H

Estimación iterativa

Para minimizar estos funcionales geométricos se requieren técnicas iterativas, por ej. usando Levenberg-Marquardt

Métodos iterativos

- Requiere inicialización
- Son más lentos que los métodos cerrados
- Pueden no converger o quedarse en un mínimo local
- Decidir cuando parar puede ser problemático

Estimación robusta

RANSAC

Concepto

Todo punto con un error mayor al que se espera por ruido es “outlier”.

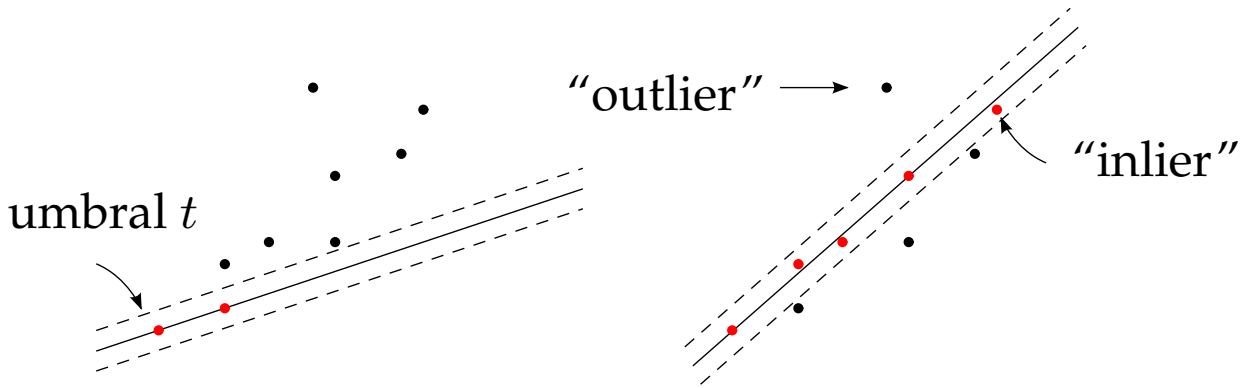
Proceso

- Elegir al azar un conjunto mínimo de datos que describen el modelo.
- Contar cuantos de los puntos restantes “consensúan” con este modelo (umbral t). Estos puntos son “inliers”.
- Si se encuentran un número suficiente de puntos que consensúan (umbral T) se reestima el modelo con todos los “inliers” y se termina.
- Sino se elige otro conjunto y se vuelve a repetir lo anterior.
- Luego de una cantidad determinada (umbral N) de pruebas se elige el modelo con mas consenso y se reestima con todos los “inliers”

Estimación robusta

RANSAC

Ejemplo de RANSAC para una recta



modelo de bajo consenso

modelo de alto consenso

Umbrales

t: depende del nivel de ruido que se espera en los datos
T: se elije considerando la cantidad de inliers que se espera en los datos
N: depende del porcentaje de outliers que se espera en los datos

Estimación robusta

Algorítmo

Finalmente...

1. Usando RANSAC eliminar outliers
2. Computar una $\hat{H}$ aproximada usando por ejemplo DLT
3. Minimizar el error usando Levenberg-Marquardt

El algoritmo completo puede repetirse hasta que los inliers converjan.

Imágenes con Python

Presentación

Wellcome Python!!

Un cacho de cultura

“Python es un lenguaje de programación interpretado cuya filosofía hace hincapié en una sintaxis muy limpia y que favorezca un código legible.”

Wikipedia.

. . .

- Apareció en 1991, creado por Guido van Rossum (quién aún dirige el proyecto)
- Es código abierto (licencia compatible con GPL)
- Es multiplataforma

Imágenes con Python

Presentación

Wellcome Python!!

Un cacho de cultura

“Python es un lenguaje de programación interpretado cuya filosofía hace hincapié en una sintaxis muy limpia y que favorezca un código legible.”

Wikipedia.

. . .

- Apareció en 1991, creado por Guido van Rossum (quién aún dirige el proyecto)
- Es código abierto (licencia compatible con GPL)
- Es multiplataforma

Imágenes con Python

Presentación

Wellcome Python!!

Un cacho de cultura

"Python es un lenguaje de programación interpretado cuya filosofía hace hincapié en una sintaxis muy limpia y que favorezca un código legible."

Wikipedia.

. . .

- Apareció en 1991, creado por Guido van Rossum (quién aún dirige el proyecto)
- Es código abierto (licencia compatible con GPL)
- Es multiplataforma

Imágenes con Python

Presentación

Algunas características

- Las variables se definen de forma dinámica
- Tiene bucles `for` y `while`, tiene `if`, clases (`class name(class):`) y funciones (`def name(param):`), como todo el mundo
- Existen `listas = []`, `tuplas = ()`, `diccionarios = {}` y `conjuntos = set()` de datos
- Puede agregar funcionalidades importando módulos (`import`)
- Para todo lo demás, existe `ipython`

Imágenes con Python

Manipulación de imágenes (`rotate.py`)

```
# coding: utf-8
from scipy import ndimage

im = ndimage.imread('romaa.jpg')

print 'Variable tipo: ', type(im)
print 'Tamano: ',         im.shape
print 'Tipo de dato: ',   im.dtype

rot_im = ndimage.rotate(im, 45, reshape=False)

import pylab as pl
pl.figure()
pl.imshow( pl.concatenate((im, rot_im), axis=1 ))
pl.show()
```

Imágenes con Python

Entrada de datos (`select_image_points.py`)

```
import pylab as pl
from scipy import ndimage
import homography

im = ndimage.imread("romaa_gris.jpg")

pl.figure()
pl.gray()
pl.imshow(im)
pl.axis('equal')
pl.axis('off')

print 'Seleccionar cuatro puntos'
pts = pl.ginput(4, show_clicks=True)

pts = homography.make_homog( pl.array(pts).T )
print 'Puntos seleccionados en coordenadas homogeneas: \n', pts
```

Imágenes con Python

Funciones (`homography.py`)

```
import pylab as pl

def normalize(points):
 """Normiliza un conjunto de puntos en coordenadas homogeneas,
 haciendo 1 la ultima fila"""

 for row in points:
   row /= points[-1]
 return points

def make_homog(points):
  """Convierte un conjunto de puntos (de un array de dim*n) a
  coordenadas homogeneas"""

  return pl.vstack( (points, pl.ones( (1, points.shape[1]))))
```

Imágenes con Python

Miscelaneas (`contour.py`)

```
import pylab as pl
from scipy import misc

im = misc.lena()

pl.figure()
pl.gray()
pl.subplot(121)
pl.imshow(im)
pl.axis('image')
pl.axis('off')

pl.subplot(122)
pl.contour( im,  origin='image')
pl.axis('image')
pl.axis('off')
```

Imágenes con Python

Transformación afín (`image_affine_warping.py`)

```
import pylab as pl
from scipy import ndimage

im = ndimage.imread("romaa_gris.jpg")
pl.figure()
pl.gray()
pl.imshow(im)

H = pl.array([[1.4, 0.05,-300],
              [0.5, 1.5, -500],
              [0,   0,     1]])
A = H[:2,:2]
t = ( H[0,2], H[1,2] )

warped_im = ndimage.affine_transform(im, A, t)

pl.figure()
pl.gray()
pl.imshow(warped_im)

pl.show()
```

Imágenes con Python

Transformación afín (`affine_warp_image_to_image.py`)

```
from scipy import ndimage
from scipy import misc
import pylab as pl
import warp

im = ndimage.imread("romaa_gris.jpg")

pl.figure(), pl.gray()
pl.imshow(im)
pl.axis('equal'), pl.axis('off')

print 'Seleccionar 3 puntos'
pts = pl.ginput(3, show_clicks=True)
# cambiar x por y para poner pts en modo fila=y , columna=x
pts_h = pl.array([pl.array([p[1], p[0], 1]) for p in pts]).T

in_im = warp.image_in_image_affine(misc.lena(), im, pts_h)

pl.imshow(in_im)
pl.axis('equal'), pl.axis('off')

pl.show()
```

Imágenes con Python

Propuesta: Escribir la funcion `image_in_image_affine()`.

```
"""Pasos para escribir image_in_image_affine(im1, im2, tp):
 armar la matriz A tal que h7=h8=0
 cada par de correspondencias debe formar 2 filas
 [ -fp[0], -fp[1], -1, 0, 0, 0, tp[0] ]
 [ 0, 0, 0, -fp[0], -fp[1], -1, tp[1] ]
 donde tp y fp son los puntos correspondientes"""

"""obtener el vector h usando SVD"""
U,S,V = pylab.linalg.svd(A)
"""h es la ultima columna de V"""

"""construir H, agregando h7=h8=0"""
H = h.reshape(3,3)

""" de H obtener A y t para llamar a """
warped_im = ndimage.affine_transform( im1, A, t, size)
"""retorna la imagen transformada rellenando con 0.
 Para transformar busca en im1 cada punto de warped_im1
 warped(i,j) = im1(A(i,j)+b(i,j))"""

"""para incrustar warped_im1 en im2"""
mascara = (warped_im1 > 0)*1
return (1-mascara)*im2 + (mascara)*warped_im1
```

Imágenes con Python

Acondicionamiento numérico para DLT

```
# condicionamiento de puntos por razones numericas
# -- from points --
m = pl.mean(fp[:2], axis=1)
maxstd = max(pl.std(fp[:2], axis=1)) + 1e-9
C1 = pl.diag([1/maxstd, 1/maxstd, 1])
C1[0][2] = -m[0]/maxstd
C1[1][2] = -m[1]/maxstd
fp = pl.dot(C1,fp)

# -- to points --
m = pl.mean(tp[:2], axis=1)
maxstd = max(pl.std(tp[:2], axis=1)) + 1e-9
C2 = pl.diag([1/maxstd, 1/maxstd, 1])
C2[0][2] = -m[0]/maxstd
C2[1][2] = -m[1]/maxstd
tp = pl.dot(C2,tp)
```