Natural Construction of Homography Transformation

Dejan Milosavljevic

dmilos@gmail.com

TOC

Description

Computing the plane to plane homography without fancy thing such as inversion of big matrix or eugen vectors values.
In here it will be present method in step-by-step manner where every step has some geometric meaning.

Nomenclature

2D Euclid vector

Ordered pair of two real numbers.
$\mathbf{p} = (x, y) = [x,y]^T; x,y \in \mathbb{R}$

2D Homography vector

Ordered triplet of three real numbers.
$\mathbf{p} = (x, y, z) = [x,y,z]^T \; x, y, z \in \mathbb{R}$

2D Euclid matrix

2 by 2 table of real number

$$\begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix}$$

2D Homography matrix

$$\left[\begin{array}{cc|c} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{22} \\ \hline h_{23} & h_{32} & h_{33} \end{array}\right]$$

Problem

What we have

- $s_0, s_x, s_y, s_z \in \mathbb{R}^2$
- $t_0, t_x, t_y, t_z \in \mathbb{R}^2$
- o=(0,0)
- det(s_i, s_j, s_k) $\neq 0$; $i \neq j \neq k$; i,j, k $\in$ { 0, x, y, z }, this condition can be relaxed to only det(s_0, s_x, s_y) $\neq 0$;
- det(t_i, t_j, t_k) $\neq 0$; $i \neq j \neq k$; i,j, k $\in$ { 0, x, y, z }
- Nice to have:
  | det(t_i, t_j, t_k) | $\leq$ det(t_0, t_x, t_y)
  | det(s_i, s_j, s_k) | $\leq$ det(s_0, s_x, s_y)

What we want

- **H** is homography matrix.
- t_i = **H** (s_i); i $\in$ { 0, x, y, z }

Existing Solution

TODO

TODO

TODO

TODO

$h_{33} = 1$

For simplicity in here $x_1 = s_{0_x}$, $y_1 = s_{0_y}$, ..., $x'_1 = t_{0_x}$, $y'_1 = t_{0_y}$, ...

And we have to solve :

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1 x'_1 & -y_1 x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1 y'_1 & -y_1 y'_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x_2 x'_2 & -y_2 x'_2 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -x_2 y'_2 & -y_2 y'_2 \\ x_3 & y_3 & 1 & 0 & 0 & 0 & -x_3 x'_3 & -y_3 x'_3 \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \end{bmatrix} = \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ x_3 \end{bmatrix}$$

$$
\begin{array}{cccccccccc}
0 & 0 & 0 & x_3 & y_3 & 1 & -x_3\,y'_3 & -y_3\,y'_3 & h_{23} & y_3 \\
x_4 & y_4 & 1 & 0 & 0 & 0 & -x_4\,x'_4 & -y_4\,x'_4 & h_{31} & y_4 \\
0 & 0 & 0 & x_4 & y_4 & 1 & -x_4\,y'_4 & -y_4\,y'_4 & h_{32} & x_4
\end{array}
$$

Main disadvantaged here is that h_{33} can be near zero and possibly produce numerical instabilities.

Bricks

... or what we need to assemble to get our matrix.

Translation

Move for some vector.

(x+X,y+Y)=**P**(x,y)

Homography matrix:

$$
\left[\begin{array}{cc|c}
1 & 0 & X \\
0 & 1 & Y \\
\hline
0 & 0 & 1
\end{array}\right]
$$

Linear

If **M** is linear function then:
$\mathbf{M}(\alpha\,\mathbf{x} + \beta\,\mathbf{y}) = \alpha\,\mathbf{M}(\mathbf{x}) + \beta * \mathbf{M}(\mathbf{y})$

Represented using homography matrix:

$$
\left[\begin{array}{cc|c}
a_{11} & a_{12} & 0 \\
a_{12} & a_{22} & 0 \\
\hline
0 & 0 & 1
\end{array}\right]
$$

Affine

Represented using homography matrix:

$$
\left[\begin{array}{cc|c}
a_{11} & a_{12} & a_{13} \\
a_{21} & a_{22} & a_{23} \\
\hline
0 & 0 & 1
\end{array}\right]
$$

Where: $\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$ is linear part.

$[\,a_{13}, a_{23}]^T$ is translation part.

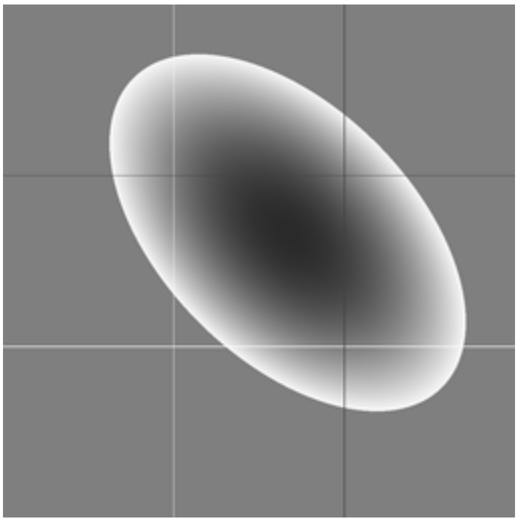
Simple Homography

(0,0) = **P**(0,0);
(1,0) = **P**(1,0);
(0,1) = **P**(0,1);
(X,Y) = **P**(1,1);

This can be represented by homography matrix:

$$\left[\begin{array}{cc|c} X & 0 & 0 \\ 0 & Y & 0 \\ \hline 1-Y & 1-X & X+Y-1 \end{array}\right]$$

$0 \neq (1-Y)^2 + (1-X)^2 + (X+Y-1)^2$
Always $\neq 0$.
Always exists.



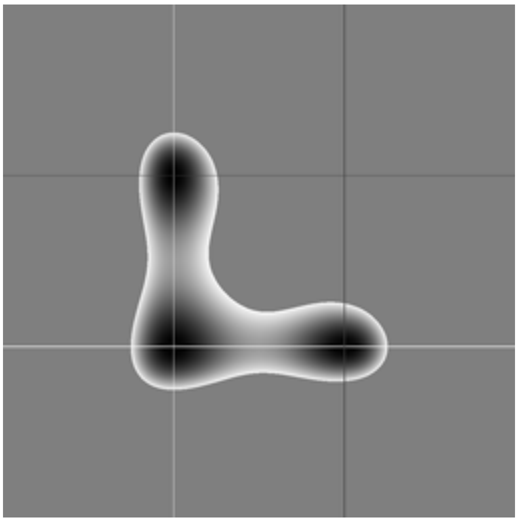
... and vice versa

(0,0) = **P**(0,0);
(1,0) = **P**(1,0);
(0,1) = **P**(0,1);
(1,1) = **P**(X,Y);

Homography matrix:

$$\left[\begin{array}{cc|c} Y * (X + Y - 1) & 0 & 0 \\ 0 & X * (X + Y - 1) & 0 \\ \hline Y*(Y - 1) & X*(X - 1) & X*Y \end{array}\right]$$

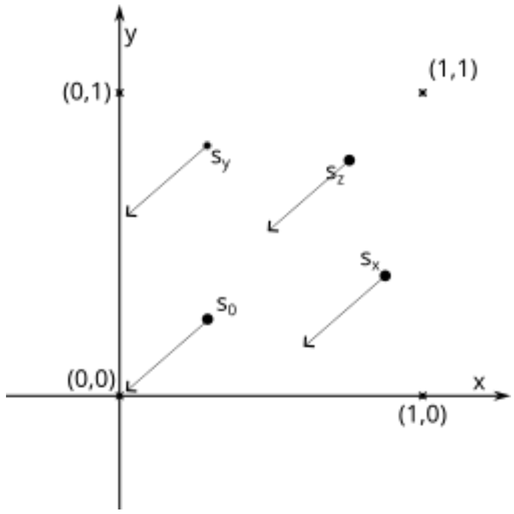
$0 \neq (Y*(Y - 1))^2 + (X*(X - 1))^2 + (X*Y)^2$
$(X,Y) \neq (0,0)$
$(X,Y) \neq (0,1)$
$(X,Y) \neq (0,0)$



Building

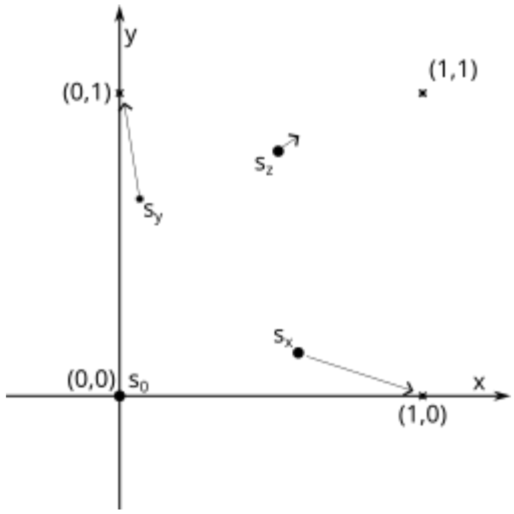
Elements

- **$\mathbf{T_s}$**: translation matrix, translate from s_0 to *o=(0,0)*



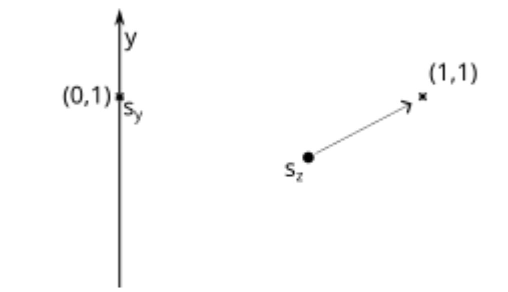
- **$\mathbf{M_s}$**:

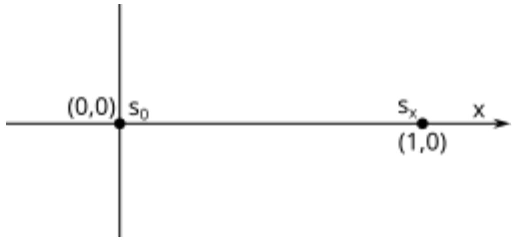
  $(1, 0) = \mathbf{M_s}(\mathbf{T_s} (s_x))$,
  $(0, 1) = \mathbf{M_s}(\mathbf{T_s} (s_y))$



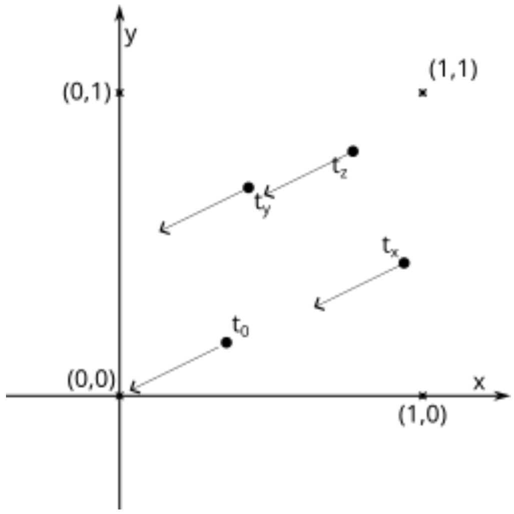
- **$\mathbf{P_s}$**:

  $(0,0) = \mathbf{P_s}((0,0))$,
  $(1,0) = \mathbf{P_s}((1,0))$,
  $(0,1) = \mathbf{P_s}((0,1))$,
  $(1,1) = \mathbf{P_s}(\mathbf{M_s}\ \mathbf{T_s}(s_z))$





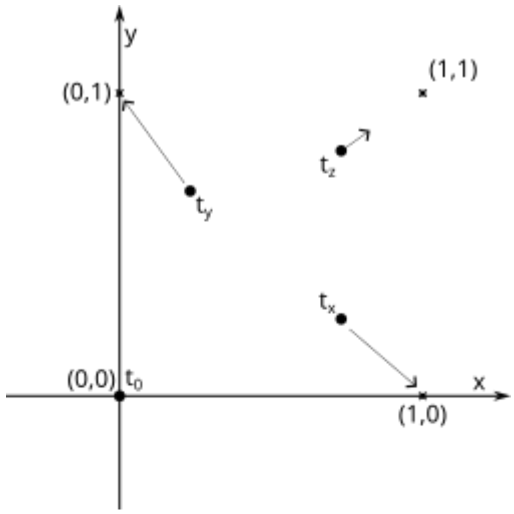
T_t: translation matrix, translate from t_0 to *o*



-

M_t:

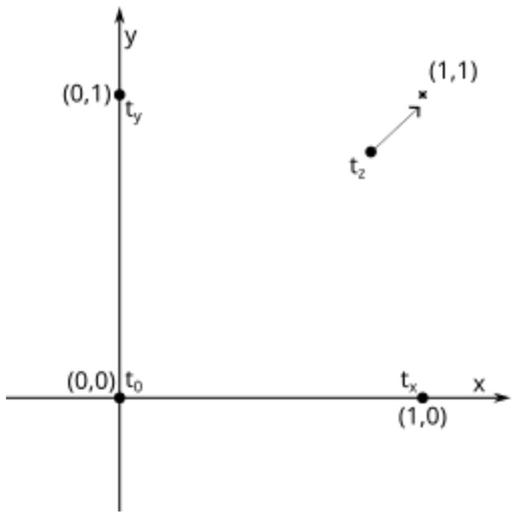
(1, 0) = **M_t**(**T_t** (t_x)),
(0, 1) = **M_t**(**T_t** (t_y))



-

P_t:

(0,0) = **P_t**((0,0)),
(1,0) = **P_t**((1,0)),
(0,1) = **P_t**((0,1)),
(1,1) = **P_t**(**M_t** **T_t**(t_z))
M_t **T_t**(t_z) $\notin$ { (0,0), (1,0), (0,1) }



-

Assembling

$$\mathbf{H} = (\mathbf{P_t}\ \mathbf{M_t}\ \mathbf{T_t})^{-1}\ \mathbf{P_s}\ \mathbf{M_s}\ \mathbf{T_s} = \mathbf{T_t}^{-1}\ \mathbf{M_t}^{-1}\ \mathbf{P_t}^{-1}\ \mathbf{P_s}\ \mathbf{M_s}\ \mathbf{T_s}$$

Miscellaneous

Source code

github.com/dmilos/math/linar/homography/construct2.hpp

Comment

- This can be easily extend to higher dimensions
- With appropriate effort it can be make in close forme.