# 逆透视变换的一种方法

朱葛峻

2015年12月

## 1 写这篇文章的原因

前段时间有人问了我这篇帖子（[分享] 好久没有人发技术贴了，来个透视变换与其逆变换的矩阵公式，打打鸡血），文中的方法很有道理，但却很复杂。于是我想把我的方法写出来。

## 2 数学模型

小孔摄像机数学模型：

$$\boldsymbol{s}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

针对以上模型，我们可以归纳为：

$$\boldsymbol{s \cdot m' = A \cdot [R|t] \cdot M'}$$

其中s是倍乘因子，$m'$指图像坐标系，$M'$指世界坐标系，$[R|t]$指线性变换矩阵，A指相机内参数矩阵。

## 3 对于模型的一点解释

$\boldsymbol{s \cdot m'}$这个就是我们通常程序中对图像的贮存，即获得的二维数组，当然也有些同学用一维数组表示。也许以后有同学能设计出一个数据结构呢。

A内参数矩阵的意义是，将相机坐标系转换为像平面坐标系，可以这么想像，将一幅本来存在于芯片镜面上的图像转换为我们脑海里的数组。其实就是转换坐标系，图像一开始在芯片以毫米英寸为单位，现在转换为以一个像素为单位了。其实就像可以描述一个人为一米八，也可以描述为为九头身。

$\boldsymbol{[R|t]}$就是我们所说的线性变换矩阵。关于这里的知识可以访问百度文库里的这篇演示文档：图像处理之逆透视变换。

M'就是世界坐标系了。

## 4 应用

这个公式源自张正友先生的论文，opencv里也有引用。这里讲下在飞思卡尔智能车的应用下的方式。

首先，一般智能车选择的是单目摄像头且不存在测距模块，无法获得一副图像的深度图，那么自然而然，Z=0。

$$\boldsymbol{s}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix}$$

$$let \quad \boldsymbol{H} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix}$$

$$and \quad \boldsymbol{H_{34}} = \begin{bmatrix} H_{11} & H_{12} & Don'tCare & H_{13} \\ H_{21} & H_{22} & Don'tCare & H_{23} \\ H_{31} & H_{32} & Don'tCare & 1 \end{bmatrix}$$

又因为$H_{33}$可以归纳为s的一部分，即可得到最终我们需要的一个表达式。

$$\begin{bmatrix} \overline{x} \\ \overline{y} \\ \overline{s} \end{bmatrix} = \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

## 5 演算

直到这里，这些都依旧是难以使用的东西。

$$x = \frac{\overline{x}}{\overline{s}} \tag{1}$$

$$y = \frac{\overline{y}}{\overline{s}} \tag{2}$$

然后就可以得到

$$x = \frac{XH_{11} + YH_{12} + H_{13}}{XH_{31} + YH_{32} + 1} \tag{3}$$

$$y = \frac{XH_{21} + YH_{22} + H_{23}}{XH_{31} + YH_{32} + 1} \tag{4}$$

$$xXH_{31} + xYH_{32} + x - XH_{11} - YH_{12} - H_{13} = 0 \tag{5}$$

$$yXH_{31} + yYH_{32} + y - XH_{21} - YH_{22} - H_{23} = 0 \tag{6}$$

$$\begin{bmatrix} X & Y & 1 & 0 & 0 & 0 & -xX & -xY \\ 0 & 0 & 0 & X & Y & 1 & -yX & -yY \end{bmatrix} \begin{bmatrix} H_{11} \\ H_{12} \\ H_{13} \\ H_{21} \\ H_{22} \\ H_{23} \\ H_{31} \\ H_{32} \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\rightarrow A \cdot X = B$$

学过线性代数的我们知道，要有唯一解，需要保证增广矩阵为满秩，即$rank(A|B) = 8$。那么就需要四组不相关的图像坐标系的点。由于世界坐标系的点在标定的时候，我们是知道的。那么我们可以选择一个矩形的四个顶点，将之代入上式即可。

## 6 代码实现

具体代码实现在我以前的帖子里就有，这里我也不再赘述。逆透视变换的代码。总之就一句话，保证全局有解再去追求迭代的快速。