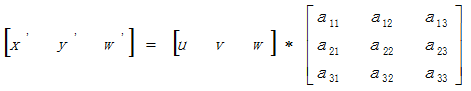
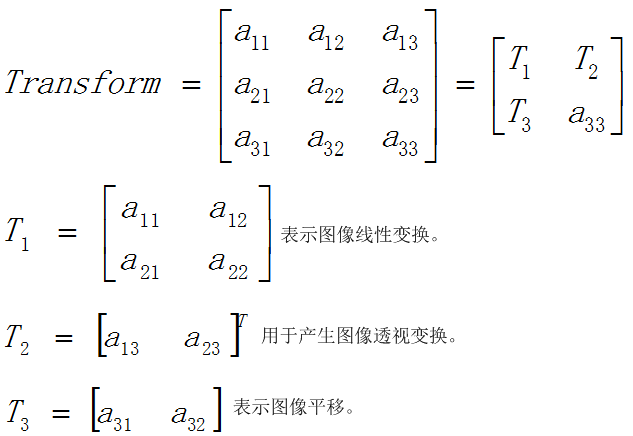
[**图像几何变换之透视变换**](https://www.cnblogs.com/liekkas0626/p/5262942.html).

1.基本原理

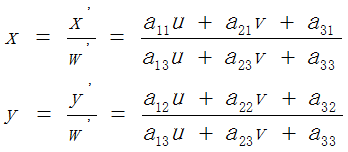
  透视变换（Perspective Transformation）的本质是将图像投影到一个新的视平面，其通用变换公式为：



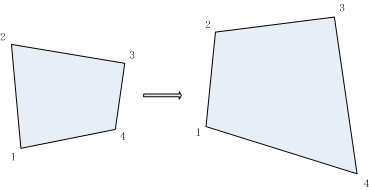
  （u，v）为原始图像像素坐标，（x=x’/w’，y=y’/w’）为变换之后的图像像素坐标。透视变换矩阵图解如下：



  仿射变换（Affine Transformation）可以理解为透视变换的特殊形式。透视变换的数学表达式为：



  所以，给定透视变换对应的四对像素点坐标，即可求得透视变换矩阵；反之，给定透视变换矩阵，即可对图像或像素点坐标完成透视变换，如下图所示：



Mat getPerspectiveTransform(const Point2f\* src, const Point2f\* dst)  
// Calculate a perspective transform from four pairs of the corresponding points.  
// src – Coordinates of quadrangle vertices in the source image.  
// dst – Coordinates of the corresponding quadrangle vertices in the destination image.

void warpPerspective(InputArray src, OutputArray dst, InputArray M, Size dsize,  int flags=INTER\_LINEAR, int borderMode=BORDER\_CONSTANT, const Scalar& borderValue=Scalar())  
// Apply a perspective transform to an image.  
// src – Source image.  
// dst – Destination image that has the size dsize and the same type as src.  
// M – 3\*3 transformation matrix.  
// dsize – Size of the destination image.  
// flags – Combination of interpolation methods and the optional flag WARP\_INVERSE\_MAP that means that M is the inverse transformation src).◊(dst  
// borderMode – Pixel extrapolation method. When borderMode=BORDER\_TRANSPARENT, it means that the pixels in the destination image that corresponds to the “outliers” in the source image are not modified by the function.  
// borderValue – Value used in case of a constant border. By default, it is 0.

3.程序

#include <iostream>  
#include "highgui.h"  
#include "opencv2/imgproc/imgproc.hpp"  
int main()  
{  
   // get original image.  
   cv::Mat originalImage = cv::imread("road.png");  
      
   // perspective image.  
   cv::Mat perspectiveImage;  
      
   // perspective transform  
   cv::Point2f objectivePoints[4], imagePoints[4];  
  
   // original image points.  
   imagePoints[0].x = 10.0; imagePoints[0].y = 457.0;  
   imagePoints[1].x = 395.0; imagePoints[1].y = 291.0;  
   imagePoints[2].x = 624.0; imagePoints[2].y = 291.0;  
   imagePoints[3].x = 1000.0; imagePoints[3].y = 457.0;  
  
   // objective points of perspective image.  
   // move up the perspective image : objectivePoints.y - value .  
   // move left the perspective image : objectivePoints.x - value.  
   double moveValueX = 0.0;  
   double moveValueY = 0.0;  
  
   objectivePoints[0].x = 46.0 + moveValueX; objectivePoints[0].y = 920.0 + moveValueY;  
   objectivePoints[1].x = 46.0 + moveValueX; objectivePoints[1].y = 100.0 + moveValueY;  
   objectivePoints[2].x = 600.0 + moveValueX; objectivePoints[2].y = 100.0 + moveValueY;  
   objectivePoints[3].x = 600.0 + moveValueX; objectivePoints[3].y = 920.0 + moveValueY;  
  
   cv::Mat transform = cv::getPerspectiveTransform(objectivePoints, imagePoints);  
  
   // perspective.  
   cv::warpPerspective(originalImage,  
                       perspectiveImage,  
                       transform,  
                       cv::Size(originalImage.rows, originalImage.cols),  
                       cv::INTER\_LINEAR | cv::WARP\_INVERSE\_MAP);  
  
   // cv::imshow("perspective image", perspectiveImage);  
   // cvWaitKey(0);  
  
   cv::imwrite("perspectiveImage.png", perspectiveImage);  
  
   return 0;  
}

透视变换效果：



