CSCE 643 Multi-View Geometry CV Homework I

Yukun Zeng
Department of Computer Science and Engineering
Texas A&M University
College Station, TX 77840
Email: yzeng@tamu.edu

I. Four point rectification

As we know, the pictures taken by a camera is actually projections of Euclidean prototype of real-world scenarios. Assuming we have a point in the Euclidean space whose homogenized coordinate is (x, y, z), and its counterpart in projective space, or in the picture, is (x', y', z). According to the definition of projective transformation, the planar projective transformation can be represented by a non-singular 3×3 matrix:

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} \tag{1}$$

more briefly as $\mathbf{x}' = \mathbf{H}\mathbf{x}$, this can be further transformed into (note that by default we assume $h_{33} = 1$):

$$\begin{aligned} x' &= h_{11}x + h_{12}y + h_{13}z \\ y' &= h_{21}x + h_{22}y + h_{23}z \\ z' &= h_{31}x + h_{32}y + z \end{aligned} \tag{2}$$

However, this is up scale, and if we go down scale and set $z' = 1, z = 1$, it turns out that it can be transformed through equation 3–5:

$$\begin{aligned} x' &= \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + 1} \\ y' &= \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + 1} \end{aligned} \tag{3}$$

$$\begin{aligned} x'(h_{31}x + h_{32}y + 1) &= h_{11}x + h_{12}y + h_{13} \\ y'(h_{31}x + h_{32}y + 1) &= h_{21}x + h_{22}y + h_{23} \end{aligned} \tag{4}$$

$$\begin{aligned} x' &= h_{11}x + h_{12}y + h_{13} - h_{31}x'x - h_{32}xy \\ y' &= h_{21}x + h_{22}y + h_{23} - h_{31}xy' - h_{32}yy' \end{aligned} \tag{5}$$

which could also be represented by matrix forms:

$$\begin{bmatrix} x & y & 1 & 0 & 0 & 0 & -xx' & -yx' & -x' \\ 0 & 0 & 0 & x & y & 1 & -xy' & -yy' & -y' \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} = 0 \tag{6}$$

Now, say we have four points whose coordinate in Euclidean space is $(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4)$, and their corresponding coordinates in projective space is $(x'_1, y'_1), (x'_2, y'_2), (x'_3, y'_3), (x'_4, y'_4)$, let:

$$\begin{aligned} p_i &= \begin{pmatrix} x_i & y_i & 1 & 0 & 0 & 0 & -x_i x'_i & -y_i x'_i & -x'_i \end{pmatrix} \\ p'_i &= \begin{pmatrix} 0 & 0 & 0 & x_i & y_i & 1 & -x_i y'_i & -y_i y'_i & -y'_i \end{pmatrix} \end{aligned} \tag{7}$$

we can then easily scale equation 6 for our current point set:

$$\begin{bmatrix} p_1 \\ p'_1 \\ p_2 \\ p'_2 \\ p_3 \\ p'_3 \\ p_4 \\ p'_4 \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ 1 \end{bmatrix} = 0 \tag{8}$$

Then, we solve the above simultaneous linear equations thereby getting the homograpy $\mathbf{H}$ and apply the homography on the original picture to rectify it, hereby the question 1 is solved and a review of solution steps is listed as follows:

1) Pick up 4 apex points of a rectangle in the picture and get their coordinates.
2) Acquire the coordinates of those points picked in the actual Euclidean space (as we just want to rectify the image here we just arbitrarily select 4 coordinates that can form a rectangle and ignores things about scales and actual location).
3) Do reverse homography to get the homography matrix using those coordinates we get.
4) Apply the homography we get to all points in the picture space and get the rectified image.

II. Affine rectification using parallelism

The key of using parallel lines in projective space to recover affine properties from images is the infinite line. In the affinity space, the infinite line is a fixed line $l_\infty = (0, 0, 1)^T$, however a projective transformation might maps l_∞ from the fixed line at infinity to a finite line l on the space after projection. Then, say we have the infinite line $l = (l_1, l_2, l_3)^T$ in a projective space, where $l_3 \neq 0$, and the homography of this current projection $\mathbf{H}$ can be divided as:

$$\mathbf{H} = \mathbf{H}_A \mathbf{H}_P \tag{9}$$

where $\mathbf{H}_A$ is the affine homography and the last matrix $\mathbf{H}_P$ is the homography for transformation from affine space to current projective space:

$$\mathbf{H}_P = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ l_1 & l_2 & l_3 \end{bmatrix} \tag{10}$$

That is to say, the current projective transformation can be decomposed into two parts, one is the transformation to affine space and the other one is the transformation from affinity to current projective space, and the later one can be directly calculated if the infinite line is given.

Now that we figured out the infinite line could help us back to affinity, we can start to work on the details to calculate the infinite line. We know that in Euclidean space, two parallel line will intersect at an ideal point on infinite line, and if we can get two ideal points then we can easily calculate the infinite line as two points determine a line. Intuitively, we can identify two pairs of parallel lines from the distorted picture and calculate two ideal points through them to form the infinite line and then we can get back to affinity based on our discussion above.

Suppose we have four points p_1, p_2, p_3, p_4 and they form a rectangle similar to what we have in question 1, through those we can simply get two pairs of parallel lines:

$$\begin{aligned} \vec{l_1} &= p_1 \times p_2 \\ \vec{l_2} &= p_3 \times p_4 \\ \vec{m_1} &= p_1 \times p_3 \\ \vec{m_2} &= p_2 \times p_4 \end{aligned} \tag{11}$$

in which we have $l_1 /\!/ l_2$ and $m_1 /\!/ m_2$. Through those pairs of parallel lines, we can further compute two points at the infinite line as follows:

$$\begin{aligned} v_1 &= \vec{l_1} \times \vec{l_2} \\ v_2 &= \vec{m_1} \times \vec{m_2} \end{aligned} \tag{12}$$

And finally we can acquire the line at infinity $\vec{l_\infty} = (l_1, l_2, l_3)$ which can be calculated by:

$$\vec{l_\infty} = v_1 \times v_2 \tag{13}$$

According to our discussion above, now we can form a new matrix same as given in equation 10 based on the infinite line, and the new $\mathbf{H}_P$ can handle the transformation between the picture space and affinity.

References

[1] J. E. Smith, “A study of branch prediction strategies,” in *Proceedings of the 8th annual symposium on Computer Architecture*. IEEE Computer Society Press, 1981, pp. 135–148.

[2] B. Calder, D. Grunwald, M. Jones, D. Lindsay, J. Martin, M. Mozer, and B. Zorn, “Evidence-based static branch prediction using machine learning,” *ACM Transactions on Programming Languages and Systems (TOPLAS)*, vol. 19, no. 1, pp. 188–222, 1997.

[3] J. K. Lee and A. J. Smith, “Branch prediction strategies and branch target buffer design,” in *Instruction-level parallel processors*. IEEE Computer Society Press, 1995, pp. 83–99.

[4] T.-Y. Yeh and Y. N. Patt, “Two-level adaptive training branch prediction,” in *Proceedings of the 24th annual international symposium on Microarchitecture*. ACM, 1991, pp. 51–61.

[5] D. A. Jiménez and C. Lin, “Dynamic branch prediction with perceptrons,” in *High-Performance Computer Architecture, 2001. HPCA. The Seventh International Symposium on*. IEEE, 2001, pp. 197–206.

[6] D. A. Jiménez, “Fast path-based neural branch prediction,” in *Microarchitecture, 2003. MICRO-36. Proceedings. 36th Annual IEEE/ACM International Symposium on*. IEEE, 2003, pp. 243–252.

[7] D. Jiménez, “Idealized piecewise linear branch prediction,” *Journal of Instruction-Level Parallelism*, vol. 7, pp. 1–11, 2005.

[8] A. Seznec, “The l-tage predictor,” *Journal of Instruction Level Parallelism*, vol. 9, 2007.

[9] H.-D. Block, “The perceptron: A model for brain functioning. i,” *Reviews of Modern Physics*, vol. 34, no. 1, p. 123, 1962.

[10] D. A. Jiménez, “Java Fast Path-Based Neural Branch Predictor,” http://hpca23.cse.tamu.edu/taco/public-bp/index.html.