# A "Loop and Zhang" Reader for Stereo Rectification

**Avinash Kak**

**Purdue University**

November 15, 2022

5:45pm

## An RVL Tutorial Presentation

**Originally presented in Fall 2020. Corrections and code included in Fall 2022.**

# CONTENTS


| | *Section Title* | *Page* |
|---|---|---|
| **1** | **What is a "Reader"?** | 3 |
| **2** | **Why the Loop and Zhang Algorithm Merits a Reader** | 4 |
| **3** | **Epipolar Geometry** | 6 |
| **4** | **The Need for Stereo Rectification** | 10 |
| **5** | **Properties of Rectifying Homographies** | 14 |
| **6** | **The Loop and Zhang Algorithm – In a Nutshell** | 25 |
| **7** | **Computing the Purely Projective Components of $H$ and $H'$** | 31 |
| **8** | **Computing the Similarity Components of $H$ and $H'$** | 44 |
| **9** | **Computing the Shearing Components of $H$ and $H'$** | 51 |
| **10** | **A C++ Implementation of the Algorithm by Álvarez and García** | 58 |

Back to TOC

# 1: What is a "Reader"?

- One of the less commonly known meanings of the word "reader" is that it is a book (in these days a PDF document) that is meant to serve as an instructional aid in learning from the published literature on a subject.

- The purpose of this document is to serve as a "Reader" for the stereo rectification algorithm by Loop and Zhang.

- This Reader is meant to serve as a handout for Lecture 24 of my class on Computer Vision at Purdue University. Here is a link to the course website so that you can see for yourself where this lecture belongs in an overall organization of the course:

`https://engineering.purdue.edu/kak/ComputerVision`

Back to TOC

# 2: Why the Loop and Zhang Algorithm Merits a Reader

- When I first sat down to look over the paper by Loop and Zhang, I gave myself an hour to understand the algorithm and to write notes on it so that I could explain it well in my class. But, as it turned out, it took considerably longer and the end-result is the document you are looking at.

- **The Loop and Zhang algorithm for stereo rectification is one of the most beautiful algorithms I have encountered in my years of research. The algorithm has more "moving parts" than is commonly the case with computer vision algorithms. And there is a certain harmony to how the parts meld together in the final solution.**

- Most people develop an imagistic understanding of an algorithm especially when geometry is involved. The Loop and Zhang algorithm provides a rich tableau of such imagery. It has elements that send image points off to infinity where they can be dealt with more easily than at the finite. In addition, much of the reasoning power in Loop and Zhang is based on giving separate geometrical interpretations to the different rows of the main transformation matrices — which is rather unusual in and

of itself. Operations involving matrix rows are therefore evocative of mental imagery of moving parts in a complex system that must work together in order to achieve the desired goal. A good thing about such imagery is that it makes you think more clearly about future possibilities associated with this class of algorithms.

- **What I personally find magical about stereo rectification with Loop and Zhang type of algorithms is that all they need for rectifying a given pair of stereo images of a scene is the $3 \times 3$ Fundamental Matrix for the images. That is, you do NOT need any calibration information for the cameras — at least that's the case in principle. Therefore, as far as theory is concerned, there is sufficient geometrical structure encoded in the Fundamental Matrix to alter the two images through a combination of nonlinear and linear transformations for their rectification. The main challenge then becomes one of teasing out of the Fundamental Matrix the information needed to construct these transformations.**

- Here is a link to the paper by Loop and Zhang:

  Charles Loop and Zhengyou Zhang, "Computing rectifying homographies for stereo vision" CVPR 1999.

  https://ieeexplore.ieee.org/abstract/document/786928

  http://www.charlesloop.com/tr99-21.pdf

Back to TOC

# 3: Epipolar Geometry

- As explained in the Lecture 23 scroll, the epipolar geometry is the geometry of stereo imaging with two pin-hole cameras. Typically, the principal axes (the same thing as the pointing angles) of the two cameras intersect somewhere in the scene of interest. This is done so that the viewpoint coverage of the scene and the quality of the focus are more or less the same in both images. We will use the notation $I$ and $I'$ to refer to the two images of a stereo pair.

- As shown in Figure 1, for any given world point $\mathbf{X}$, the plane passing through $\mathbf{X}$ and the two camera centers $C$ and $C'$ is known as the *epipolar plane*. This plane intersects the two image planes in two lines, typically denoted $l$ and $l'$, that are known as the *epipolar lines*. It is implied by the geometry of pin-hole imaging that all the epipolar lines in each image will pass through a distinguished point in that image that is known as the *epipole*. Figure 1 shows the epipoles $\mathbf{e}$ and $\mathbf{e}'$ in the two images. The epipole in each image is the projection of the camera center for the other camera. That is, $\mathbf{e}$ is the projection in the image $I$ of the camera center $C'$. Similarly, $\mathbf{e}'$ is the projection in $I'$ of the camera center $C$. A unique straight line passes through all four points, $C$, $\mathbf{e}$, $\mathbf{e}'$, and $C'$. The portion of

this line from $C$ to $C'$ is called the *baseline* for a given stereo rig.

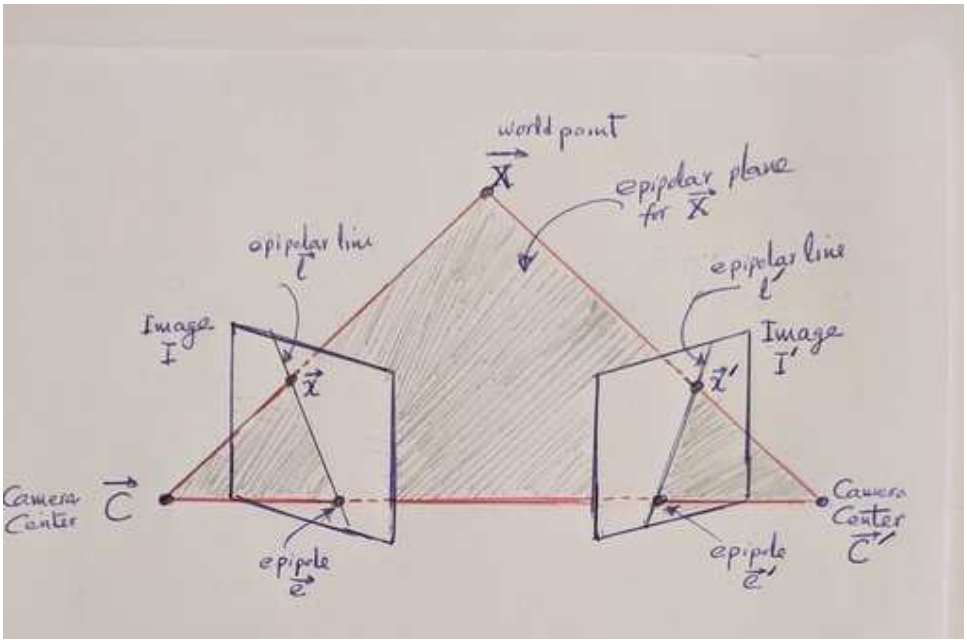



Figure 1: *Epipolar Geometry*

- As mentioned in Lecture 23 scroll, the epipolar geometry for a given stereo rig is represented algebraically by the *Fundamental Matrix* – a $3 \times 3$ matrix of rank 2. Denoting this matrix by $F$, the two corresponding pixels $\mathbf{x}$ and $\mathbf{x}'$ that are the camera projections of the same world point $\mathbf{X}$, are related to each other by the following constraint based on $F$:

$$\mathbf{x}'^T F \mathbf{x} \;=\; 0 \tag{1}$$

It is important to remember that it is only a necessary constraint on the pair of pixels $(\mathbf{x}, \mathbf{x}')$ to be each other's corresponding pixels, but not a sufficient constraint. That is, if

$(\mathbf{x}, \mathbf{x}')$ are truly each other's corresponding pixels, they will indeed satisfy the constraint shown above. However, for, say, a given pixel $\mathbf{x}$ in image $I$, there will be many pixels $\mathbf{x}'$ in image $I'$ that will satisfy the relationship. In general, for a given pixel $\mathbf{x}$ in $I$, all the pixels in $I'$ that will satisfy the above relationship will form the epipolar line $l'$ in $I'$ that is given by

$$l' \;=\; F\mathbf{x} \tag{2}$$

- Along the same lines, for a given pixel $\mathbf{x}'$ in $I'$, all the pixels in $I$ that satisfy the constraint in Eq. (1) will fall on the line $l$ given by

$$l \;=\; F^T\mathbf{x}' \tag{3}$$

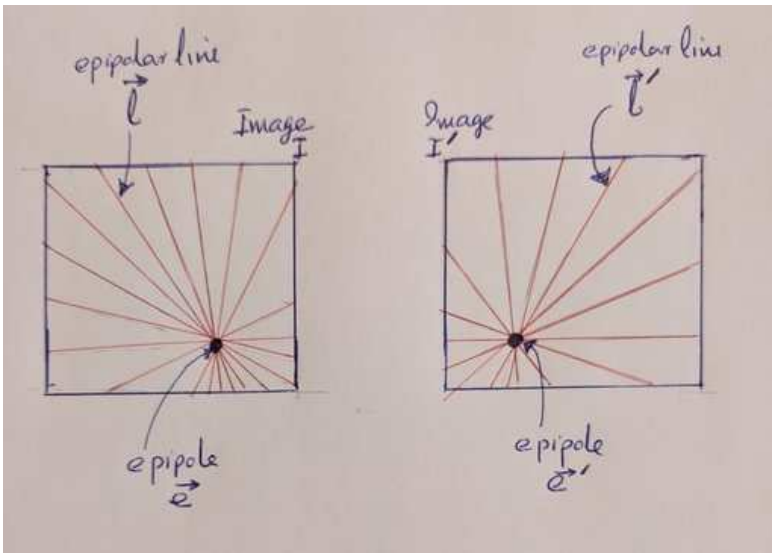



Figure 2: *Epipolar Lines*

- For every one of the pixels on the line $l$ in $I$, its matching pixel will be on the line $l'$ in $I'$ and vice versa. We can therefore to the lines $l$ and $l'$ in the two images as being in epipolar correspondence.

- Figure 2 emphasizes the fact that all epipolar lines in the two images must pass through the respective epipoles the images,

Back to TOC

# 4: The Need for Image Rectification

- A most important goal of stereo imaging is 3D scene reconstruction. That frequently requires that we scan one of the two images of a stair pair – we will refer to that image by $I$ and the other image by $I'$ — pixel by pixel and, for *each* pixel $\mathbf{x} \in I$, we find its corresponding pixel $\mathbf{x}' \in I'$. From the pixel correspondences $(\mathbf{x}, \mathbf{x}')$ thus constructed we compute the coordinates of the world points $\mathbf{X}$ according to the method described in the Lecture 24 scroll. The world points defined by the 3D coordinates thus estimated would then constitute a point-cloud as obtained from the $(I, I')$ stereo pair. A point cloud thus created would normally be converted to a DSM (Digital Surface Map). If you are able to record stereo images from multiple directions, you could either try to aggregate all the pairwise point clouds together and then create a fuller DSM or first create pair-wise DSMs and then fuse them together subsequently.

- The scene reconstruction process outlined above would not work unless you have a good approach in place for constructing a dense set of pixel correspondences represented by $(\mathbf{x}, \mathbf{x}')$.

- Based on the ideas presented in the previous section, in order to

find the corresponding pixel $\mathbf{x}'$ in $I'$ for a given pixel $\mathbf{x}$ in $I$, you will have to search along the epipolar line given by $l' = F\mathbf{x}$ in $I'$. So for every pixel $\mathbf{x}$ in $I$, you would construct a line like the ones shown in Figure 2 and search along such a line for the best matching pixel. There are several shortcomings associated with this approach to searching for the corresponding pixels: (1) Since several pixels in $I$ will have the same $l'$ associated with them, you would be wasting computational effort. (2) Searching along the epipolar lines of the sort shown in Figure 2 makes it more difficult to invoke heuristics to help out with the search for correspondences.

- The difficulties of solving the correspondence problem mentioned above can be alleviated if we first rectify the two images. Figure 3 provides a visual explanation of the idea of stereo rectification.

- As illustrated pictorially in Figure 3, image rectification means that you apply appropriate homographies to the original image pair $(I, I')$ in order to obtain the image pair $(\hat{I}, \hat{I}')$ shown in the figure. We refer to the images $\hat{I}$ and $\hat{I}'$ as the rectified images.

- Let's now focus on what makes the rectified pair $(\hat{I}, \hat{I}')$ so special with regard to solving the stereo correspondence problem:

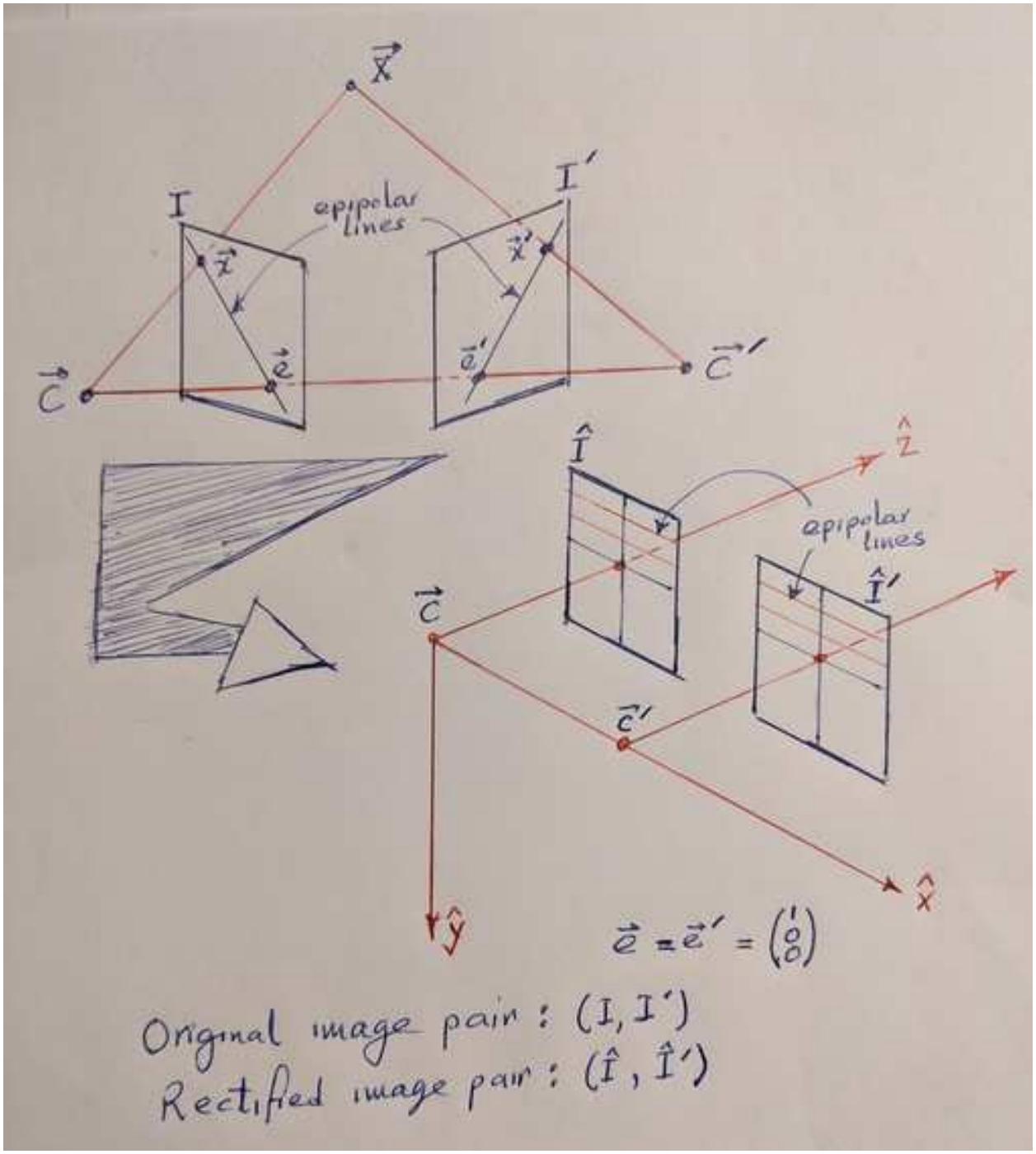



Figure 3: *Stereo Rectification*

- Without question, the most important property of the rectified images that the epipolar lines, which originally formed a star shaped pattern as shown in Figure 2, are now parallel.

- The second most important property of the rectified images is that the corresponding pair of epipolar lines are characterized by the same row index in the two images.

- For another property — although it is less a property and more a convention — we assume that after rectification that the world coordinate frame becomes congruent with the the $\hat{I}$ image, meaning that the world origin is at the camera center for the rectified image $\hat{I}$, and that the world $\hat{X}$ axis is along the rows in the images as shown.

- The goal of stereo rectification is to come up with the homographies $H$ and $H'$ for $I$ and $I'$, respectively, so that

$$H : I \quad \rightarrow \quad \hat{I} \tag{4}$$

$$H' : I' \quad \rightarrow \quad \hat{I}' \tag{5}$$

  with the condition that the rectified images $\hat{I}$ and $\hat{I}'$ would look like as shown in Figure 3.

Back to TOC

# 5: Properties of Rectifying Homographies

As mentioned in the previous section, we use $H$ and $H'$ to represent the rectifying $3 \times 3$ homographies for the original images $I$ and $I'$ In other words, $H$ is meant for the image $I$ and $H'$ for $I'$.

**Property 1:** Interpreting the three rows of $H$ and $H'$ as homogeneous representations of lines in the respective image planes, we can show that the lines corresponding to the second and the third rows of the homographies contain the epipole in the image planes. That is, the epipole $\mathbf{e}$ is on the two lines represented by the second and the third rows of $H$. Similarly, the epipole $\mathbf{e}'$ is on the two lines represented by the second and the third rows of $\mathbf{e}'$.

*Proof:* Regardless of where the epipoles $\mathbf{e}$ and $\mathbf{e}'$ are actually located in their respective image planes, in order to transform the original images into the images in their target configuration shown in Figure 3, the following must be true:

$$H\mathbf{e} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \tag{6}$$

$$H'\mathbf{e}' = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \tag{7}$$

We will now express the homographies $H$ and $H'$ in terms of their row vectors:

$$H = \begin{bmatrix} \mathbf{u}^T \\ \mathbf{v}^T \\ \mathbf{w}^T \end{bmatrix} = \begin{bmatrix} u_a & u_b & u_c \\ v_a & v_b & v_c \\ w_a & w_b & w_c \end{bmatrix} \tag{8}$$

$$H' = \begin{bmatrix} \mathbf{u'}^T \\ \mathbf{v'}^T \\ \mathbf{w'}^T \end{bmatrix} = \begin{bmatrix} u'_a & u'_b & u'_c \\ v'_a & v'_b & v'_c \\ w'_a & w'_b & w'_c \end{bmatrix} \tag{9}$$

The symbols $\mathbf{u}$, $\mathbf{v}$ and $\mathbf{w}$ stand for the vectors that represent the rows of the $3 \times 3$ homography $H$. Similarly, $\mathbf{u'}$, $\mathbf{v'}$ and $\mathbf{w'}$ stand for the vectors of the homography $H'$. Obviously, then, we can write

$$H\mathbf{e} = \begin{bmatrix} \mathbf{u}^T\mathbf{e} \\ \mathbf{v}^T\mathbf{e} \\ \mathbf{w}^T\mathbf{e} \end{bmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \tag{10}$$

$$H'\mathbf{e'} = \begin{bmatrix} \mathbf{u'}^T\mathbf{e'} \\ \mathbf{v'}^T\mathbf{e'} \\ \mathbf{w'}^T\mathbf{e'} \end{bmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \tag{11}$$

Now recall that the condition for a point $\mathbf{x}$ to be on a line $l$ is that $l^T\mathbf{x} = 0$. Therefore, the following two equations that are implied by Eq. (10)

$$\mathbf{v}^T\mathbf{e} = 0 \tag{12}$$

$$\mathbf{w}^T\mathbf{e} = 0 \tag{13}$$

tell us that if we choose to interpret the 3-vectors $\mathbf{v}$ and $\mathbf{w}$ as lines in the plane of the image $I$, both these two lines pass through image epipole $\mathbf{e}$. Similarly, the following two equations implied by Eq. (11)

$$\mathbf{v'}^T\mathbf{e'} = 0 \tag{14}$$

$$\mathbf{w'}^T\mathbf{e'} = 0 \tag{15}$$

that the lines $\mathbf{v'}$ and $\mathbf{w'}$ in the plane of the image $I'$ pass through the epipole $\mathbf{e'}$.

So we have proved that the second and the third rows of the homography $H$, when interpreted as lines in the plane of the image $I$, pass through the epipole $\mathbf{e}$ in $I$. Likewise, we have proved that the second and the third rows of the homography $H'$, when interpreted as lines in the plane of the image $I'$, pass through the epipole $\mathbf{e'}$.

**Property 2:** All three rows of the homographies $H$ and $H'$ are linearly independent. This follows from the fact that by definition a homography must be non-singular. As a consequence of this independence, the row vectors $\mathbf{v}$ and $\mathbf{w}$ mentioned in the previous property represent two different lines in the plane of the image $I$. That implies that the lines $\mathbf{v}$ and $\mathbf{w}$ must intersect at the epipole $\mathbf{e}$ in the image $I$. Similar facts hold in the image $I'$. That is, the lines $\mathbf{v'}$ and $\mathbf{w'}$ from the homography $H'$ intersect at the epipole $\mathbf{e'}$ in $I'$.

**Property 3:** We are free to choose whatever we wish for the first rows of the homographies $H$ and $H'$ for these transformations to provide the "basic" image rectification action: *sending the epipoles to infinity along the world X-axis and aligning the epipolar lines row-wise in the two images.* However, as implied by the arguments in the following proof, while the second and the third rows of the two homographics will send epipoles to infinity (a *necessary* condition that must be satisfied) along the world X axis, it is *not* sufficient to achieve a distortion free rectification of the two stereo images. What's shown below is a proof of the assertion that the relationship between the pre- and the post-rectification forms of the fundamental matrix are independent of the first rows of the rectifying homographies.

*Proof:* As shown on page 4 of my Lecture 23 scroll, when the same camera is used for both images, the Fundamental Matrix for the rectified images $(\hat{I}, \hat{I}')$ is given by:

$$\hat{F} = [\hat{\mathbf{e}}']_\times \tag{16}$$

$$= \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}_\times \tag{17}$$

$$= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \tag{18}$$

where $[1\ 0\ 0]^T$ represents a point at infinity on the world X-axis

and the notation $[.]_\times$ means the cross-product representation of a 3-vector as a $3 \times 3$ matrix. The algebraic constraint satisfied by the corresponding pixels $(\hat{\mathbf{x}}, \hat{\mathbf{x}'})$ in the rectified images vis-a-vis the Fundamental Matrix $\hat{F}$ is:

$$\hat{\mathbf{x}'}^T \hat{F} \hat{\mathbf{x}} = 0 \tag{19}$$

However,

$$\hat{\mathbf{x}} = H\mathbf{x} \tag{20}$$

$$\hat{\mathbf{x}'} = H'\mathbf{x}' \tag{21}$$

Substituting these in Eq. (19) yields:

$$\mathbf{x}'^T H'^T \hat{F} H \mathbf{x} = 0 \tag{22}$$

which implies that the Fundamental Matrix, $F$, for the original image pair $(I, I')$ is related to the post-rectification Fundamental Matrix $\hat{F}$ by

$$F = H'^T \hat{F} H \tag{23}$$

$$= H'^T \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} H \tag{24}$$

$$= \begin{bmatrix} u'_a & u'_b & u'_c \\ v'_a & v'_b & v'_c \\ w'_a & w'_b & w'_c \end{bmatrix}^T \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} u_a & u_b & u_c \\ v_a & v_b & v_c \\ w_a & w_b & w_c \end{bmatrix} \tag{25}$$

$$= \begin{bmatrix} u'_a & v'_a & w'_a \\ u'_b & v'_b & w'_b \\ u'_c & v'_c & w'_c \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} u_a & u_b & u_c \\ v_a & v_b & v_c \\ w_a & w_b & w_c \end{bmatrix} \quad (26)$$

$$= \begin{bmatrix} u'_a & v'_a & w'_a \\ u'_b & v'_b & w'_b \\ u'_c & v'_c & w'_c \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 \\ -w_a & -w_b & -w_c \\ v_a & v_b & v_c \end{bmatrix} \quad (27)$$

$$= \begin{bmatrix} -v'_a w_a + w'_a v_a & -v'_a w_b + w'_a v_b & -v'_a w_c + w'_a v_c \\ -v'_b w_a + w'_b v_a & -v'_b w_b + w'_b v_b & -v'_b w_c + w'_b v_c \\ -v'_c w_a + w'_c v_a & -v'_c w_b + w'_c v_b & -v'_c w_c + w'_c v_c \end{bmatrix} \quad (28)$$

As you can see, the first-row elements of the rectifying homographies $H$ and $H'$ do not enter the above final matrix shown for $F$. So, regardless of how we set the first rows of $H$ and $H'$, the target Fundamental Matrix will have the desired form shown in Eq. (18) as long as we get the other two rows of the rectifying homographies right. If we wanted to, we could even set the first rows to the line at infinity (as long as we are giving line interpretations to the different rows of the homographies). If we did that, the first rows of both $H$ and $H'$ would be set to [0 0 1]. But, as you will see, this freedom in setting the first rows of the two homographies is illusory on account of our need to control the distortion in the target images $(\hat{I}, \hat{I}')$.

**Property 4:** With $H$ and $H'$ expressed as shown in Eqs. (8) and (9), the inverses of these homographies can be expressed as

$$H^{-1} = [\mathbf{v}\times\mathbf{w} \quad \mathbf{w}\times\mathbf{u} \quad \mathbf{u}\times\mathbf{v}] \tag{29}$$

$$H'^{-1} = [\mathbf{v}'\times\mathbf{w}' \quad \mathbf{w}'\times\mathbf{u}' \quad \mathbf{u}'\times\mathbf{v}'] \tag{30}$$

*Proof:* The proof that follows is for $H^{-1}$. That for $H'^{-1}$ has similar steps.

$$HH^{-1} = \begin{bmatrix} \mathbf{u}^T \\ \mathbf{v}^T \\ \mathbf{w}^T \end{bmatrix} [\mathbf{v}\times\mathbf{w} \quad \mathbf{w}\times\mathbf{u} \quad \mathbf{u}\times\mathbf{v}] \tag{31}$$

$$= \begin{bmatrix} \mathbf{u}^T(\mathbf{v}\times\mathbf{w}) & \mathbf{u}^T(\mathbf{w}\times\mathbf{u}) & \mathbf{u}^T(\mathbf{u}\times\mathbf{v}) \\ \mathbf{v}^T(\mathbf{v}\times\mathbf{w}) & \mathbf{v}^T(\mathbf{w}\times\mathbf{u}) & \mathbf{v}^T(\mathbf{u}\times\mathbf{v}) \\ \mathbf{w}^T(\mathbf{v}\times\mathbf{w}) & \mathbf{w}^T(\mathbf{w}\times\mathbf{u}) & \mathbf{w}^T(\mathbf{u}\times\mathbf{v}) \end{bmatrix} \tag{32}$$

$$= \begin{bmatrix} \mathbf{u}\cdot(\mathbf{v}\times\mathbf{w}) & \mathbf{u}\cdot(\mathbf{w}\times\mathbf{u}) & \mathbf{u}\cdot(\mathbf{u}\times\mathbf{v}) \\ \mathbf{v}\cdot(\mathbf{v}\times\mathbf{w}) & \mathbf{v}\cdot(\mathbf{w}\times\mathbf{u}) & \mathbf{v}\cdot(\mathbf{u}\times\mathbf{v}) \\ \mathbf{w}\cdot(\mathbf{v}\times\mathbf{w}) & \mathbf{w}\cdot(\mathbf{w}\times\mathbf{u}) & \mathbf{w}\cdot(\mathbf{u}\times\mathbf{v}) \end{bmatrix} \tag{33}$$

$$= \begin{bmatrix} \mathbf{u}\cdot(\mathbf{v}\times\mathbf{w}) & 0 & 0 \\ 0 & \mathbf{v}\cdot(\mathbf{w}\times\mathbf{u}) & 0 \\ 0 & 0 & \mathbf{w}\cdot(\mathbf{u}\times\mathbf{v}) \end{bmatrix} \tag{34}$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{35}$$

where the last equality follows from the fact that the value returned by a scalar triple product is unchanged under a circular shift of its three operands and fact that we are dealing with homogeneous objects.

**Property 5:** Continuing with our interpretation of the rows of $H$ and $H'$ as lines in the original images $I$ and $I'$, respectively, we now claim the following property for the lines $\mathbf{v}$ and $\mathbf{w}$ in $I$ vis-a-vis the lines $\mathbf{v}'$ and $\mathbf{w}'$ in $I'$: The pair of lines $(\mathbf{v}, \mathbf{v}')$ are in epipolar correspondence, as are the pair of lines $(\mathbf{w}, \mathbf{w}')$. This property points to a strong linkage between the homography $H$ for the image $I$ and the homography $H'$ for the image $I'$.

*Proof:* The starting point for this proof is the relationship between $F$ and $\hat{F}$ shown in Eq. (23). It follows from that relationship that

$$FH^{-1} \;=\; H'^{T}\hat{F} \tag{36}$$

In what follows, I'll first simplify the left-hand-side expression $FH^{-1}$ and then I'll do the same for the right-hand-side expression $H'^{T}\hat{F}$.

Substituting for $H^{-1}$ from Eq. (29), we can write

$$FH^{-1} \;=\; F[\mathbf{v}\times\mathbf{w} \quad \mathbf{w}\times\mathbf{u} \quad \mathbf{u}\times\mathbf{v}] \tag{37}$$

It was stated earlier in Property 1 that both the lines $\mathbf{v}$ and $\mathbf{w}$ contain the epipole $\mathbf{e}$. We also know that $\mathbf{v}$ and $\mathbf{w}$ are independent vectors, as stated in Property 2. Therefore, the epipole $\mathbf{e}$ in $I$ is the only point that is common to the lines $\mathbf{v}$ and $\mathbf{w}$. That is, the lines $\mathbf{v}$ and $\mathbf{w}$ in $I$ must intersect at $\mathbf{e}$. That implies

$$\mathbf{v}\times\mathbf{w} \;=\; \mathbf{e} \tag{38}$$

We will now define two putative points in the plane of the $I$ image:

$$\mathbf{y} = \mathbf{v} \times \mathbf{u} \tag{39}$$

$$\mathbf{z} = \mathbf{w} \times \mathbf{u} \tag{40}$$

The vector $\mathbf{y}$ represents a point at the intersection of the lines $\mathbf{v}$ and $\mathbf{u}$ in the plane of the image $I$. By the same token, $\mathbf{z}$ is a point at the intersection of the lines $\mathbf{w}$ and $\mathbf{u}$, also in $I$. What will be important to us going forward is that the point $\mathbf{y}$ is on line $\mathbf{v}$ and the point $\mathbf{z}$ is on line $\mathbf{w}$ in the image $I$. Substituting these in Eq. (37), we get

$$FH^{-1} = F[\mathbf{e} \quad \mathbf{z} \quad -\mathbf{y}] \tag{41}$$

From Lecture 23 scroll, the epipole $\mathbf{e}$ is the right null-vector of $F$:

$$F\mathbf{e} = 0 \tag{42}$$

Therefore, the result in Eq. (41) further simplifies to

$$FH^{-1} = [\mathbf{0} \quad F\mathbf{z} \quad -F\mathbf{y}] \tag{43}$$

That's it for the simplification of the left-hand side of Eq. (36).

Let's now work on the right-hand side of Eq. (36). Using the form shown in Eq. (9) for $H'$, we can write the following:

$$H'\hat{F} = [\mathbf{u}' \quad \mathbf{v}' \quad \mathbf{w}'] \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \tag{44}$$

$$= [\mathbf{0} \quad \mathbf{w}' \quad -\mathbf{v}'] \tag{45}$$

Substituting the RHS's in Eqs. (43) and (45) in Eq. (36), we get

$$[\mathbf{0} \quad F\mathbf{z} \quad -F\mathbf{y}] = [\mathbf{0} \quad \mathbf{w}' \quad -\mathbf{v}'] \tag{46}$$

Recall from the comment in red just after Eq. (40), $\mathbf{z}$ is a point on line $\mathbf{w}$ in image $I$, and $\mathbf{y}$ is a point on line $\mathbf{v}$ in the same image. What the above equation tells is that $\mathbf{w}'$ is the epipolar line in image $I'$ for the point $\mathbf{z}$ in image $I$. Similarly, $\mathbf{v}'$ is the epipolar line in image $I'$ for the point $\mathbf{y}$ in $I$. [Recall that when the fundamental matrix multiplies a point in the left image, you get the epipolar line in the right image for that point.]

So we conclude that the last row $\mathbf{w}'$ of the rectifying homography $H'$ for image $I'$ is the epipolar line in $I'$ for the point $\mathbf{z}$ in image $I$. And that the second row $\mathbf{v}'$ of the same rectifying homography for image $I'$ is the epipolar line in $I'$ for the point $\mathbf{y}$ in image $I$.

**Note that by Property 3 we are free to choose any vectors for the line u in $I$. Therefore, since z is at the intersection of the lines u and w in $I$ (see Eq. (39)), we can consider z to be an arbitrary point on line w. Along the same lines, through Eq. (40), we can consider y to be an arbitrary point on line v in $I$.**

Since for *every* position of point $\mathbf{z}$ on line $\mathbf{w}$ in $I$ we end up with the same epipolar line $\mathbf{w}'$ in $I'$, it must be the case that the lines $(\mathbf{w}, \mathbf{w}')$ in the two images are in epipolar correspondence with each other.

Along the same lines as above, since for *every* position of point $\mathbf{y}$ on line $\mathbf{v}$ in $I$ we end up with the same epipolar line $\mathbf{v}'$ in $I'$, it must be the case that the lines $(\mathbf{v}, \mathbf{v}')$ in the two images are in epipolar correspondence with each other.

To summarize, we can claim that the pair of lines $(\mathbf{w}, \mathbf{w}')$ in the two images are in epipolar correspondence with each other, as are the pair of lines $(\mathbf{v}, \mathbf{v}')$.

********

That ends the presentation of the section on The Properties of the Rectifying Homographies.

In the rest of this Reader, I'll first summarize the key steps of the Loop and Zhang algorithm in the section that follows. In subsequent sections, I'll invoke the properties you now understand for a more detailed presentation of the algorithm.

Back to TOC

# 6: The Loop and Zhang Algorithm – In a Nutshell

- The Loop and Zhang algorithm decomposes the rectifying homographies $H$ and $H'$ as follows:

$$H = H_{sh} H_{sim} H_p \tag{47}$$

$$H' = H'_{sh} H'_{sim} H'_p \tag{48}$$

  where

  – $H_p$ and $H'_p$ are purely projective homographies whose purpose is to send the epipoles $\mathbf{e}$ and $\mathbf{e}'$ to infinity in the respective image planes. An important question in specifying $H_p$ and $H'_p$ is the choice of the direction in which the epipoles should go to infinity. **We want to choose that direction which causes minimal distortion to the images.** As you would expect, pure projectivity can be highly distorting in general. One manifestation of projective distortion is that the scene lines that are parallel in the original image would appear to converge or diverge after the image is transformed with a projective homography.

  – $H_{sim}$ and $H'_{sim}$ are similarity homographies. A similarity homography can only rotate, translate, and uniformly scale

an image. Apart from the scale change, it cannot distort the image. The task assigned to the two similarity homographies is to rotate the epipoles, that are at infinity after the application of the projective homographies, so they are on the world-X axis.

– $H_{sh}$ and $H'_{sh}$ are shearing homographies. To appreciate the need for these homographies, note that after the application of $H_{sim}H_p$ to $I$ and the application of $H'_{sim}H'_p$ to $I'$ we have a pair images with epipolar lines that are parallel and that are oriented along the world-X axis. **Nonetheless, it remains that, in general, a nonlinear distortion of the two images caused by projective homography CANNOT be undone by any affine homography — for the simple reason that similarity transformations are a strict subgroup of the projective transformations in the hierarchy of transformations. Therefore, in general, it would be theoretically impossible for the nonlinear distortion of the images caused by $H_p$ and $H'_p$ to be undone by any choices for $H_{sim}$ and $H'_{sim}$. Consequently, the best we can hope to do is to REDUCE as much as possible this distortion by introducing additional degrees of freedom into the overall rectification transformation.** As you will see later in this report, that is what is accomplished by the shearing transforms $H_{sh}$ and $H'_{sh}$.

- In order to separate the distortion inducing transformations and

what can only be approximate attempts at reducing the distortions, it is useful to rewrite the homography decompositions shown in Eqs. (47) and (48) as follows:

$$H = H_a H_p \tag{49}$$

$$H' = {H'}_a {H'}_p \tag{50}$$

which shows that fundamentally any rectification homography must be a product of an affine part and a projective part, with the latter needed for sending the epipoles to infinity and the former needed primarily for mitigating the distortions caused by the latter.

- Since $H_p$ and ${H'}_p$ are pure projective, we can express them as follows:

$$H_p = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ w_a & w_b & 1 \end{bmatrix} \tag{51}$$

$${H'}_p = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ {w'}_a & {w'}_b & 1 \end{bmatrix} \tag{52}$$

- By inspection, we can write the following for the inverses of the projective homographies:

$$H_p{}^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -w_a & -w_b & 1 \end{bmatrix} \tag{53}$$

$$H'_p{}^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -w'_a & -w'_b & 1 \end{bmatrix} \tag{54}$$

- The inverses shown above allow us to write the following for the affine parts of the rectification homographies as shown in Eqs. (49) and (50):

$$H_a = H H_p{}^{-1} \tag{55}$$

$$H'_a = H' H'_p{}^{-1} \tag{56}$$

- Going forward, we will use the following forms for $H$ and $H'$ in which we have set the the elements $w_c$ and $w'_c$ in the last rows to 1.

$$H = \begin{bmatrix} \mathbf{u}^T \\ \mathbf{v}^T \\ \mathbf{w}^T \end{bmatrix} = \begin{bmatrix} u_a & u_b & u_c \\ v_a & v_b & v_c \\ w_a & w_b & 1 \end{bmatrix} \tag{57}$$

$$H' = \begin{bmatrix} \mathbf{u'}^T \\ \mathbf{v'}^T \\ \mathbf{w'}^T \end{bmatrix} = \begin{bmatrix} u'_a & u'_b & u'_c \\ v'_a & v'_b & v'_c \\ w'_a & w'_b & 1 \end{bmatrix} \tag{58}$$

- Regarding explicitly setting $w_c$ and $w'_c$ to 1 should not, in general, cause a problem for us since $H$ and $H'$ are homogeneous. **However, as described in my Lecture 11 scroll, setting, say, $w_c$ to 1 will prevent sending the origin in the image $I$ to infinity.** That could become an issue for us if the epipole in $I$ were located at the origin of the coordinate frame used for $I$. Typically, a corner of the sensor array is used as the image origin and the likelihood of the epipole coinciding with such an origin are rather slim. The two cameras would need to be angled sharply toward each other for that to be the case — not a typical camera configuration for stereo imaging.

- Using the forms for $H$ and $H'$ shown in Eqs. (55) and (56) and using for inverse projectivity the forms shown in Eqs. (53) and (54), we get the following result for the affine components of the rectification homographies:

$$H_a = \begin{bmatrix} u_a - u_c w_a & u_b - u_c w_b & u_c \\ v_a - v_c w_a & v_b - v_c w_b & v_c \\ 0 & 0 & 1 \end{bmatrix} \tag{59}$$

$$H'_a = \begin{bmatrix} u'_a - u'_c w'_a & u'_b - u'_c w'_b & u'_c \\ v'_a - v'_c w'_a & v'_b - v'_c w'_b & v'_c \\ 0 & 0 & 1 \end{bmatrix} \tag{60}$$

- The rest of this Reader discusses how to make the best possible choices for the individual transformations in the decompositions

of the rectification homographies as shown on the right hand side in Eqs. (47) and (48).

- Our first order of business is obviously to send the epipoles $\mathbf{e}$ in $I$ and $\mathbf{e}'$ in $I'$ to infinity. That's what we will start with in the next section.

Back to TOC

# 7: Computing the Purely Projective Components of $H$ and $H'$

- As you know well by this time, the purpose of the projective components $H_p$ of $H$ and ${H'}_p$ of $H'$ is send the epipoles in the respective planes to infinity. **The question then is: In what direction should the epipoles be sent to infinity?**

- One might think that the answer to the question posed above should be obvious from the lower part of Figure 3 that shows the epipolar lines in the rectified images to be aligned with the world-X axis, which requires that both the epipoles be at infinity on the world-X axis.

- If you use Figure 3 to directly set $H_p$ and ${H'}_p$, that is likely to distort the images $I$ and $I'$ to such an extent that, in general, it may not be possible "fix" the images sufficiently with subsequent affine homographies for a good solution to the stereo correspondence problem.

- **The approach taken in Loop and Zhang is to consider all possible directions for sending the epipoles to infinity and then use the direction that results in the least distortion to**

**the images.**

- In the discussion that follows, we will focus on the image $I$ for finding the best direction in which to send its epipole $\mathbf{e}$ to infinity. Because the homographies for $I$ and $I'$ are coupled (See Property 5 in Section 5), that discussion will also throw up the best projective homography for the other image $I'$.

- For finding the best direction for $I$ in which to send its epipole to infinity, we define a **epipolar line selector vector z** [As to why we call **z** the epipolar line selector vector will soon be clear.] and parameterize it by

$$\mathbf{z} = \begin{bmatrix} \lambda \\ \mu \\ 0 \end{bmatrix} \tag{61}$$

  The point $\mathbf{z}$, being an Ideal Point since it is on the line $l_\infty$ in $I$, serves as a direction vector in the plane of the image.

- IMPORTANT: The definition of $\mathbf{z}$ above is not to be confused with the definition of the same symbol in Eq. (40). There we defined $\mathbf{z}$ as the intersection of the two lines in $I$, one standing for the first row of the rectifying homography $H$ and the second for the third row. On the other hand, here, while the point $\mathbf{z}$ will continue to be on the line standing for the third row of $H$ as shown below, its precise definition will be different. Note that as shown by Eqs. (51) and (57), the third rows of $H$ and $H_p$ are identical.

- We now set the last row of $H_p$, as given by the row vector $\mathbf{w}$, to the line passing through the epipole $\mathbf{e}$ and the ideal point given by our epipolar line selector $\mathbf{z}$:

$$\mathbf{w} \;=\; \mathbf{e} \times \mathbf{z} \tag{62}$$

  That is, $\mathbf{w}$ is to be thought of as a line that passes through the epipole $\mathbf{e}$ and the ideal point $\mathbf{z}$.

- On the basis of Property 5 in Section 5, with the third row of $H_p$ set as above, the third row of $H'$ must be given by

$$\mathbf{w}' \;=\; F\mathbf{z} \tag{63}$$

  since by Eq. (62) the point $\mathbf{z}$ is on line $\mathbf{w}$ and since the lines $(\mathbf{w}, \mathbf{w}')$ must be in epipolar correspondence.

- For each choice for the epipolar-line selector vector $\mathbf{z}$ in the $I$ image, we will end up with different values for $(\mathbf{w}, \mathbf{w}')$ for the last rows of the projective homographies $H_p$ and ${H'}_p$. Stated equivalently, by choosing different values for the direction parameters $\lambda$ and $\mu$ in Eq. (61), **we steer the epipolar line w around the epipole w in image $I$.**

- **We now say that the best choice for z in $I$ is one that causes the least projective distortion to the image $I$ — with the expectation that that the resulting w′ will also imply the least distortion for the image $I'$.**

- That begs the question: How to measure the distortion caused by the different choices for the selector vector $\mathbf{z}$. That issue is addressed in what follows.

- To explain how the projective distortion can be measured, let's say that we have chosen a set of pixels $\{\mathbf{p}_1, \mathbf{p}_1, \ldots \mathbf{p}_n\}$ for monitoring the projective distortion as $I$ is subject to the homography $H_p$. Focusing on just one of these pixels for a moment, let the location of the pixel $\mathbf{p}_i$ be given by

$$\mathbf{p}_i = \begin{bmatrix} p_{i,u} \\ p_{i,v} \\ 1 \end{bmatrix} \tag{64}$$

- When $I$ undergoes the transformation given by $H_p$ as defined in Eq. (51), the point $\mathbf{p}_i$ will move to

$$H_p \mathbf{p}_i = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ w_a & w_b & 1 \end{bmatrix} \begin{bmatrix} p_{i,u} \\ p_{i,v} \\ 1 \end{bmatrix} \tag{65}$$

$$= \begin{bmatrix} p_{i,u} \\ p_{i,v} \\ w_i \end{bmatrix} \tag{66}$$

where the last coordinate $w_i$ is given by

$$w_i = \mathbf{w}^T \mathbf{p}_i \tag{67}$$

- What that means is that the physical coordinates of the pixel $\mathbf{p}_i$ after it goes through the projective transformation $H_p$ will be

$$\left( \frac{p_{i,u}}{w_i}, \frac{p_{i,v}}{w_i} \right) \tag{68}$$

As you would expect with a projective effect, on account of the value of the normalizer $w_i$ shown in Eq. (67), the extent of movement of a pixel depends on where the pixel is located in the image. **The greater this variability in pixel movement across the image $I$, the greater the distortion of the image.** The projective effect depends on value of the vector $\mathbf{w}$, whose value is set by the epipolar-line selector $\mathbf{z}$ through Eq. (62). **Therefore, our goal must be to use that value for the selector $\mathbf{z}$ which gives rise the smallest variability in the projectivity-induced pixel movements.**

- The variability in the pixel movements can be estimated from the movements at all the pixels in the set $\{\mathbf{p}_1, \mathbf{p}_1, \ldots \mathbf{p}_n\}$ that was defined earlier and comparing those movements with a hypothetical pixel whose coordinates are at the average of the pixels. For our set of pixels, the average pixel will have the coordinates:

$$\mathbf{p}_c \quad = \quad \sum_{i=1}^{n} \mathbf{p}_i \tag{69}$$

- It follows from Eq. (67) that the normalizing weight for the hypothetical pixel $\mathbf{p}_c$ will be given by

$$w_c \quad = \quad \mathbf{w}^T \mathbf{p}_c \tag{70}$$

- Now we can write the following as a measure of how non-uniform the normalizing weights are across the image $I$:

$$\sum_{i=1}^{n} \left( \frac{w_i - w_c}{w_c} \right)^2 \tag{71}$$

Substituting in the above measure the expression for $w_i$ from Eq. (67) and for $w_c$ from Eq. (70), we can write for our distortion measure:

$$\sum_{i=1}^{n} \left( \frac{\mathbf{w}^T(\mathbf{p}_i - \mathbf{p}_c)}{\mathbf{w}^T \mathbf{p}_c} \right)^2 \tag{72}$$

which can be expressed more compactly as shown below

$$\textbf{projectivity induced distortion in image } I \quad = \quad \frac{\mathbf{w}^T P P^T \mathbf{w}}{\mathbf{w}^T \mathbf{p}_c \mathbf{p}_c{}^T \mathbf{w}} \tag{73}$$

where $P$ is the $3 \times n$ matrix:

$$P \quad = \quad \begin{bmatrix} p_{1,u} - p_{c,u} & p_{2,u} - p_{c,u} & \ldots & p_{n,u} - p_{c,u} \\ p_{1,v} - p_{c,v} & p_{2,v} - p_{c,v} & \ldots & p_{n,v} - p_{c,v} \\ 0 & 0 & \ldots & 0 \end{bmatrix} \tag{74}$$

The reason for the last row to be all zeros is that the third coordinates for both $\mathbf{p}_i$ and $\mathbf{p}_c$ are the same — they are both set to 1.

- **Next we need a formula like the one shown in Eq. (73) for the other image of the stereo pair — the image $I'$.** Recall what $\mathbf{w}$ is for image $I$ $\mathbf{w}'$ is for image $I'$. **But we must keep in the mind the relationship between w and w′ as they are in epipolar correspondence with each other. See the comment just after Eq. (63).** Shown below is the version of Eq. (73) for image $I'$:

$$\text{projectivity induced distortion in image } I' \;=\; \frac{\mathbf{w}'^T P' P'^T \mathbf{w}'}{\mathbf{w}'^T \mathbf{p}'_c \mathbf{p}'^T_c \mathbf{w}'} \qquad (75)$$

- Combining the distortion measures shown in Eq. (73) and (75), we can write for the overall distortion caused by the projective transformation of the two images:

$$\text{Total projectivity distortion} \;=\; \frac{\mathbf{w}^T P P^T \mathbf{w}}{\mathbf{w}^T \mathbf{p}_c \mathbf{p}_c^T \mathbf{w}} \;+\; \frac{\mathbf{w}'^T P' P'^T \mathbf{w}'}{\mathbf{w}'^T \mathbf{p}'_c \mathbf{p}'^T_c \mathbf{w}'} \qquad (76)$$

- It's time to revisit our main goal in this section: **To find the value for the epipolar-line selector vector z in image $I$ that minimizes the total projective distortion in both the images $I$ and $I'$. That's going to require that we cast the above equation in terms of the selector vector z.** The relationship between $\mathbf{w}$ and $\mathbf{z}$ in the first term shown above is given by Eq. (62) and the relationship between $\mathbf{w}'$ and $\mathbf{z}$ in the second term shown above is given by Eq. (63). Substituting these in the

above form, we get:

$$D_T = \frac{\mathbf{z}^T[\mathbf{e}]_\times^T P P^T [\mathbf{e}]_\times \mathbf{z}}{\mathbf{z}^T[\mathbf{e}]_\times^T \mathbf{p}_c \mathbf{p}_c^T [\mathbf{e}]_\times \mathbf{z}} + \frac{\mathbf{z}^T F^T P' P'^T F \mathbf{z}}{\mathbf{z}^T F^T \mathbf{p'}_c \mathbf{p'}_c^T F \mathbf{z}} \tag{77}$$

where $[\mathbf{e}]_\times$ stands for the cross-product representation of a vector as a $3 \times 3$ matrix of its elements and where $D_T$ stands for the total distortion in the two images.

- In the expression shown above, except for the unknown $\mathbf{z}$, all other items are known. The epipole $\mathbf{e}$ in image $I$ is simply the right null vector of the Fundamental Matrix F that is assumed to be known; the matrix $P$ the actual coordinates of the pixels chosen for monitoring the distortion error in $I$; and the pixel $\mathbf{p}_c$ the point that is the average of all the pixels in $P$. The matrix $P'$ and the pixel $\mathbf{p'}_c$ are for $I'$ what $P$ and $\mathbf{p}_c$ are for $I$. We now define the following four variables to capture all the quantities that are known:

$$A = [\mathbf{e}]_\times^T P P^T [\mathbf{e}]_\times \tag{78}$$

$$B = [\mathbf{e}]_\times^T \mathbf{p}_c \mathbf{p}_c^T [\mathbf{e}]_\times \tag{79}$$

$$A' = F^T P' P'^T F \tag{80}$$

$$B' = F^T \mathbf{p'}_c \mathbf{p'}_c^T F \tag{81}$$

- In terms of the new variables defined above, the projectivity induced distortion in the two images can be expressed as

$$D_T = \frac{\mathbf{z}^T A \mathbf{z}}{\mathbf{z}^T B \mathbf{z}} + \frac{\mathbf{z}^T A' \mathbf{z}}{\mathbf{z}^T B' \mathbf{z}} \tag{82}$$

- For its numerical minimization, the expression for the total distortion shown above lends itself to the following two simplifications:

  **Simplification 1:** We can take advantage of the fact that $\mathbf{z}$ as specified by Eq. (61) is an ideal point and, therefore, has its third coordinate set to 0. This implies that we need to use only the upper-left $2 \times 2$ blocks of the four matrices $A$, $B$, $A'$ and $B'$ for the needed minimization. We will represent these blocks by $A_{2\times 2}$, $B_{2\times 2}$, ${A'}_{2\times 2}$ and ${B'}_{2\times 2}$.

  **Simplification 2:** The bullet for Eq. (64) mentions using a select set of $n$ pixels for monitoring the projective distortion. As it turns out, if you use all the pixels in $I$ and $I'$ for measuring the projective distortion, you end up with a simple formula for the product $PP^T$. Let's say our images are of size $M \times N$ and let's assume that both $M$ and $N$ are even (as they normally are). For the reference pixel $\mathbf{p}_c$, let's use the pixel at the coordinates $M/2, N/2$. (Strictly speaking, the reference pixel can be anywhere in the images. That is, it does NOT have to be at the dead center.) For this case, the matrix $P$ shown in Eq. (74) becomes

$$P = \begin{bmatrix} -\frac{M}{2} & \ldots & \frac{M-2}{2} \\ -\frac{N}{2} & \ldots & \frac{N-2}{2} \\ 0 & \ldots & 0 \end{bmatrix} \tag{83}$$

and, for the upper-left $2 \times 2$ block of the $PP^T$ product:

$$PP^T \Big|_{2\times 2} = \frac{MN}{12} \begin{bmatrix} N^2-1 & 0 \\ 0 & M^2-1 \end{bmatrix} \tag{84}$$

The product $PP^T$ involves the sum of squares like $1^2 + 2^2 + 3^2 + \ldots + k^2$ of the first $k$ integers for some $k$. By Faulhaber's formula, such a sum is given by $\frac{k(k+1)(2k+1)}{6}$. The factor 12 that you see in Eq. (84) arises from the fact that summing the squares over the range $(-\frac{M}{2}, \frac{M-2}{2})$ is the same as summing the squares over the range $(1, M)$ and dividing the result by 2.

Finally, regarding the terms $\mathbf{p}_c\mathbf{p}_c^T$ $\mathbf{p'}_c\mathbf{p'}_c^T$ in the denominators in Eq. (77), in line with the comment that introduces Simplification 2, we have

$$\mathbf{p}_c\mathbf{p}_c^T = \begin{bmatrix} \frac{M}{2} \\ \frac{N}{2} \\ 1 \end{bmatrix} \begin{bmatrix} \frac{M}{2} & \frac{N}{2} & 1 \end{bmatrix} \tag{85}$$

$$= \begin{bmatrix} \frac{M^2}{4} & \frac{MN}{4} & \frac{M}{2} \\ \frac{MN}{4} & \frac{N^2}{4} & \frac{N}{2} \\ \frac{M}{2} & \frac{N}{2} & 1 \end{bmatrix} \tag{86}$$

For its upper-left $2 \times 2$ block that we are interested in, we can write

$$\mathbf{p}_c\mathbf{p}_c^T\Big|_{2\times 2} = \frac{1}{4} \begin{bmatrix} M^2 & MN \\ MN & N^2 \end{bmatrix} \tag{87}$$

- Using Simplification 1 described above and also using Eq. (61), we express the result in Eq. (82) in the following form:

$$f(\lambda,\mu) = \frac{(\lambda\ \mu)^T A_{2\times 2}\begin{pmatrix}\lambda\\ \mu\end{pmatrix}}{(\lambda\ \mu)^T B_{2\times 2}\begin{pmatrix}\lambda\\ \mu\end{pmatrix}} + \frac{(\lambda\ \mu)^T A'_{2\times 2}\begin{pmatrix}\lambda\\ \mu\end{pmatrix}}{(\lambda\ \mu)^T B'_{2\times 2}\begin{pmatrix}\lambda\\ \mu\end{pmatrix}}$$

$$= \frac{A_{11}\lambda^2 + (A_{12}+A_{21})\lambda\mu + A_{22}\mu^2}{B_{11}\lambda^2 + (B_{12}+B_{21})\lambda\mu + B_{22}\mu^2} + \frac{A'_{11}\lambda^2 + (A'_{12}+A'_{21})\lambda\mu + A'_{22}\mu^2}{B'_{11}\lambda^2 + (B'_{12}+B'_{21})\lambda\mu + B'_{22}\mu^2} \tag{88}$$

- At its minimum, the partial derivatives of $f(\lambda,\mu)$ with respect to $\lambda$ and $\mu$ would be zero. If you were to take the partial derivative of the right hand side in Eq. (88) with respect to $\lambda$ and set it to zero, you end up with a polynomial of degree 7 in $\lambda$. That indicates that the function $f(\lambda,\mu)$ is likely to posses a global minimum along with several multiple local minima. This suggests an iterative approach for the needed minimization in which we start with a good guess for the solution. **Loop and Zhang have discovered that a good starting guess for the parameter $\lambda$ is given by:**

$$\lambda_{starting\_guess} = \lambda_{g_1} + \lambda_{g_2} \tag{89}$$

where $\lambda_{g_1}$ is the $\lambda$ value that minimizes just the first term in Eq. (88) and $\lambda_{g_2}$ is the $\lambda$ value that minimizes just the second term in the same equation. What is interesting is that minimizing the individual terms in Eq. (88) is a simpler problem comparing minimizing both at the same time since the former involves only a cubic polynomial in $\lambda$.

- **The rest of this section presents a fast numerical approach for calculating the guess $\mathbf{z}_{g_1} = (\lambda_{g_1}, \mu_{g_1})^T$ for both the parameters at the same time. The same logic would apply for generating the guess $\mathbf{z}_{g_2} = (\lambda_{g_2}, \mu_{g_2})^T$.**

- We first convert the minimization problem

$$\min_{\lambda} \frac{\mathbf{z}^T A \mathbf{z}}{\mathbf{z}^T B \mathbf{z}} \tag{90}$$

into the maximization problem

$$\max_{\lambda} \frac{\mathbf{z}^T B \mathbf{z}}{\mathbf{z}^T A \mathbf{z}} \tag{91}$$

- We take advantage of the fact that the matrix $A$ is symmetric and positive-definite. Any such matrix can be expressed as a product of a nonsingular matrix $\mathcal{M}$ and its transpose:

$$A \quad = \quad \mathcal{M}\mathcal{M}^T \tag{92}$$

- We now introduce a new variable $\mathbf{y}$:

$$\mathbf{y} \quad = \quad \mathcal{M}\mathbf{z} \tag{93}$$

- Substituting $\mathbf{z} = M^{-1}\mathbf{y}$ in Eq. (91), we end up with:

$$\max \frac{\mathbf{y}^T \mathcal{M}^{-T} B \mathcal{M}^{-1} \mathbf{y}}{\mathbf{y}^T \mathbf{y}} \tag{94}$$

- Recall that we started with $\mathbf{z} = (\lambda, \mu, 0)^T$ in Eq. (61) which is invariant to any scale change. In our 2D version of the problem starting with "Simplification 1" and Eq. (88), we use $\mathbf{z} = (\lambda, \mu)^T$, the solution provided by which is again invariant to scale change in the 2D vector $\mathbf{z}$. What that implies is that the solution provided by the maximization in Eq. (94) is only defined up to a scale factor. So we are allowed to set

$$||\mathbf{y}|| \quad = \quad 1 \tag{95}$$

- Subject to the above constraint, the solution to the maximization in Eq. (94) is given by

$$\mathbf{y} \quad = \quad \mathcal{M}^{-T} B \mathcal{M} \tag{96}$$

Back to TOC

# 8: Computing the Similarity Components of $H$ and $H'$

- This section shows how to compute the similarity components of the rectification homographies shown in Eqs. (47) and (48).

- However, before getting down to the task of deriving the forms for $H_{sim}$ and $H'_{sim}$, we need to express the individual elements of the rectification homographies as expressions directly involving the elements of the Fundamental Matrix. **Recall that the Loop and Zhang algorithm is predicated on you having previously estimated the Fundamental Matrix for your stereo images.** Toward that end, we revisit the form of the Fundamental Matrix shown in Eq. (28) under the assumptions that $w_c = w'_c = 1$ (**see a discussion related to this assumption in the bullet that follows Eq. (58)**):

$$F = \begin{bmatrix} w'_a v_a - v'_a w_a & w'_a v_b - v'_a w_b & v_c w'_a - v'_a \\ w'_b v_a - v'_b w_a & w'_b v_b - v'_b w_b & v_c w'_b - v'_b \\ v_a - v'_c w_a & v_b - v'_c w_b & v_c - v'_c \end{bmatrix} \tag{97}$$

- To make explicit the fact that we are already in possession of

the value for the elements of $F$, let's display this matrix as

$$F = \begin{bmatrix} F_{00} & F_{01} & F_{02} \\ F_{10} & F_{11} & F_{12} \\ F_{20} & F_{21} & F_{22} \end{bmatrix} \tag{98}$$

- Comparing the Eq. (97) with (98), we get the following for some of the elements of the two rectification homographies $H$ and $H'$:

$$\begin{aligned} F_{20} &= v_a - v'_c w_a && (99) \\ F_{21} &= v_b - v'_c w_b && (100) \\ F_{22} &= v_c - v'_c && (101) \\ F_{02} &= v_c w'_a - v'_a && (102) \\ F_{12} &= v_c w'_b - v'_b && (103) \end{aligned}$$

- The above equalities yield the following expressions for the middle row of the $H$ homography and the first two element of the middle row of the homography $H'$:

$$\begin{aligned} v_a &= F_{20} + v'_c w_a && (104) \\ v_b &= F_{21} + v'_c w_b && (105) \\ v_c &= F_{22} + v'_c && (106) \\ v'_a &= -F_{02} + v_c w'_a && (107) \\ v'_b &= -F_{12} + v_c w'_b && (108) \end{aligned}$$

We will find these useful when we derive the final results for the the similarity homographies $H_{sim}$ and $H'_{sim}$.

- **We are now all set to calculate the similarity homographies $H_{sim}$ and $H'_{sim}$.** We start with the fact that after we have specified the projective components as shown in Eqs. (51) and (52), the remaining affine components (which are products of shearing and similarity homographies) are given by Eqs. (59) and (60).

- As stated in Section 6, the main job of the similarity components $H_{sim}$ and $H'_{sim}$ of $H_a$ and $H'_a$, respectively, is to rotate the epipoles at infinity so they are along the world-X axis. Recall that the principal responsibility of the projective homographies we covered in the last section was to send the epipoles to infinity. Subsequently, this epipole must be rotated so that it is the ideal point on the world-X axis. A similarity homography is ideally set up for this rotation because the action of such homographies is limited to rotation, translation, and uniform scale change.

- Our task, therefore, boils down **to carving out** similarity homographies with the desired behavior from the affine homographies shown in Eqs. (59) and (60).

- Let's say the best choice for the selector vector $\mathbf{z}$ in the previous section sent the epipole $\mathbf{e}$ in image $I$ to $\hat{\mathbf{e}}$ and the epipole $\mathbf{e}'$ in image $I'$ to $\hat{\mathbf{e}}'$. We want the overall action of the affine homographies $H_a$ and $H'_a$ shown in Eqs. (59) and (60) to

accomplish the following

$$\begin{bmatrix} u_a - u_c w_a & u_b - u_c w_b & u_c \\ v_a - v_c w_a & v_b - v_c w_b & v_c \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{e}_u \\ \hat{e}_v \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{109}$$

$$\begin{bmatrix} u'_a - u'_c w'_a & u'_b - u'_c w'_b & u'_c \\ v'_a - v'_c w'_a & v'_b - v'_c w'_b & v'_c \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{e'}_u \\ \hat{e'}_v \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{110}$$

Note that we set the third coordinates of the transformed epipoles $\hat{\mathbf{e}}$ and $\hat{\mathbf{e}'}$ to 0 since, on the basis of the discussion in the previous section, those epipoles are guaranteed to be at infinity.

- Looking at the left hand side in Eq. (109), in the three inner products of the rows of the $3 \times 3$ matrix with the column vector in the matrix-vector product shown there, the last one is trivially satisfied. The same thing applies to Eq. (110). And, for carving out $H_{sim}$ and $H'_{sim}$ from $H_a$ and $H'_a$ we are not so interested in the first inner-product either because, according to Property 3 in Section 5, the first rows of the rectification homographies are not so important from the standpoint of transforming the epipoles in the manner desired for rectification. That leaves only the following inner products as the most important:

$$\begin{bmatrix} v_a - v_c w_a & v_b - v_c w_b & v_c \end{bmatrix} \begin{bmatrix} \hat{e}_u \\ \hat{e}_v \\ 0 \end{bmatrix} = 0$$

$$\begin{bmatrix} v'_a - v'_c w'_a & v'_b - v'_c w'_b & v'_c \end{bmatrix} \begin{bmatrix} \hat{e}'_u \\ \hat{e}'_v \\ 0 \end{bmatrix} = 0 \qquad (111)$$

- This suggests using the row "$v_a - v_c w_a \quad v_b - v_c w_b \quad v_c$" of $H_a$ as a seed for generating a $3 \times 3$ similarity homography $H_{sim}$ and the row "$v'_a - v'_c w'_a \quad v'_b - v'_c w'_b \quad v'_c$" of $H'_a$ as the seed for specifying the similarity homography $H'_{sim}$. As it turns out, these seeds are sufficient for the task at hand on account of the fact that **similarity homographies are affine and and their upper-left $2 \times 2$ block must be orthogonal.** We can use the orthogonality property to set the first row from the seed and the affine property to set the last row as shown below

$$H_{sim} = \begin{bmatrix} v_b - v_c w_b & v_c w_a - v_a & 0 \\ v_a - v_c w_a & v_b - v_c w_b & v_c \\ 0 & 0 & 1 \end{bmatrix} \qquad (112)$$

$$H'_{sim} = \begin{bmatrix} v'_b - v'_c w'_b & v'_c w'_a - v'_a & 0 \\ v'_a - v'_c w'_a & v'_b - v'_c w'_b & v'_c \\ 0 & 0 & 1 \end{bmatrix} \qquad (113)$$

Obviously, only the first two elements of the first rows of $H_{sim}$ and $H'_{sim}$ are specified by the orthogonality of the upper-left $2 \times 2$ blocks of the matrices. So, at this point in time, we do not

have sufficient information to set the last element of the first row in these homographies. So by appealing to Property 1 in Section 5 we arbitrarily set that element to 0. Substituting from Eq. (104) through (106) in Eq. (112), we can write $H_{sim}$ in the following form:

$$H_{sim} = \begin{bmatrix} F_{21} + v'_c w_b - v_c w_b & (F_{22} + v'_c) w_a - F_{20} - v'_c w_a & 0 \\ F_{20} + v'_c w_a - v_c w_a & F_{21} + v'_c w_b - v_c w_b & v_c \\ 0 & & 1 \end{bmatrix}$$

$$= \begin{bmatrix} F_{21} - w_b F_{22} & w_a F_{22} - F_{20} & 0 \\ F_{20} - w_a F_{22} & F_{21} - w_b F_{22} & F_{22} + v'_c \\ 0 & 0 & 1 \end{bmatrix} \tag{114}$$

Note that, from Eq. (106) we have $v_c - v'_c = F_{22}$. This identity was used in simplifying the first element of the first row and all of the elements in the second row. At this point, the only unknown in the similarity homography $H_{sim}$ shown above is $v'_c$.

- Along the same lines, substituting from Eqs. (106) through (108) in Eq. (113) gives us

$$H'_{sim} = \begin{bmatrix} v_c w'_b - F_{12} - v'_c w'_b & v'_c w'_a + F_{02} - v_c w'_a & 0 \\ v_c w'_a - F_{02} - v'_c w'_a & v_c w'_b - F_{12} - v'_c w'_b & 0 \\ 0 & & 1 \end{bmatrix}$$

$$= \begin{bmatrix} F_{12} - w'_b F_{22} & w'_a F_{22} - F_{02} & 0 \\ F_{02} - w'_a F_{22} & F_{12} - w'_b F_{22} & v'_c \\ 0 & 0 & 1 \end{bmatrix} \tag{115}$$

As is the case with the result shown for $H_{sim}$ in Eq. (114), at this point, everything except for the element $v'_c$ is known in the result for $H'_{sim}$ above.

- As already mentioned, in the similarity homographies in Eqs. (114) and (115), the only quantity that has yet to be specified is the element $v'_c$ in the second row of the homography $H'$. The value of this element controls the vertical translation (meaning the translation along the world-Y direction in Figure 3 between the two images after rectification. In order to bring the rectified images into row-wise correspondence for the epipolar lines we set $v'_c$ so that the smallest value of the $v$ coordinate for the pixels is 0 in both images.

Back to TOC

# 9: Computing the Shearing Components of $H$ and $H'$

- If the goal of stereo rectification were only to make the epipolar lines row-wise parallel and congruent in the two images (and doing so by moving the epipoles to infinity along the world-X direction), all we would need to do would be to apply to the images the homographies presented in the previous two sections. The projective homography of Section 7 takes care of sending the epipoles to infinity and the similarity homography of Section 8 takes care of rotating the epipoles at infinity in order to line them up with the world-X axis while at the same aligning the two images row-wise.

- Unfortunately, the story of image rectification does not end there. The reason for that is fundamental: In the hierarchy of transformations, projective transformations are at the root of the hierarchy and the other types of transformations are subgroups of the group of projective transformations. **What that implies is that, in general, it would be theoretically impossible for the nonlinear distortion (meaning the distortion in which the different pixels in the original images move by different amounts) caused by projective imaging to be completely undone by *any* non-projective transformation.** In

particular, any similarity transformation of the sort presented in the previous section *cannot be expected* to undo the nonlinear distortion caused by the projective homography of Section 7.

- **Fortunately, we have not yet pinned down all of the degrees of freedom available in the rectification homographies $H$ and $H'$.** The third row in both these homographies took care of the projective transformations of Section 7 and the second row the requirements of the similarity transformations in Section 8. Yes, the upper-left $2 \times 2$ block orthogonality requirements of the similarity transformation did generate values for the first row in $H$ and $H'$, but those values were in service of the constraints created by the second row in these mappings. **It would therefore be accurate to say that the first rows in $H$ and $H'$ have not yet been fully specified.**

- The goal of this section to pin down the first rows in $H$ and $H'$ homographies by doing the best that is possible with regard to the mitigation of the projective distortion caused by the component $H_p$ of $H$ and the component $H_{p'}$ of $H'$. If this distortion were to be left unaddressed, that would make it harder to carry out dense stereo matching that is the next step after rectification in a scene reconstruction pipeline. In the presence of distortion, it would be more difficult to find the matching pixels in the other image as a reference image is scanned pixel by pixel.

- The distortion mitigation in Loop and Zhang is carried out by specifying shearing homographies for the two images as follows:

$$H_{sh} = \begin{bmatrix} s_a & s_b & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{116}$$

$$H'_{sh} = \begin{bmatrix} s'_a & s'_b & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{117}$$

- As to why these transforms are referred to as shearing transforms and also to see that such affine transformations can be used to cause nonlinear movements of pixels (meaning that the movements at different pixels are different), just try applying a homography like $\begin{bmatrix} 1 & 2 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ to a square pattern whose four corners are at the points $[(1,1),(1,2),(2,1),(2,2)]$. However, unlike projective mappings, the nonlinear affects achieved by an affine homography are of a special kind: the parallel lines stay parallel.

- In case you are wondering why we set to zero the last element in the first rows in $H_{sh}$ and $H'_{sh}$, that element merely causes a translation of the output pattern for a given input pattern. We are not interested in such translations.

- **In the rest of this section, we will focus on calculating the**

**shearing homography for just the image $I$. Calculating the same for $I'$ proceeds in exactly the same fashion.**

- In Loop and Zhang, the specification of the shearing homographies is based on compensating for the distorting effects of the combination $H_{sim}H_p$ on the original image $I$. This distortion is estimated by measuring the consequences of $H_{sim}H_p$ on the four points $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$ and $\mathbf{d}$ at the midpoints of the four edges of $I$ as shown in Figure 4. Toward that end, we define two vectors by

$$\begin{aligned} \mathbf{x} &= \mathbf{b} - \mathbf{d} \\ \mathbf{y} &= \mathbf{a} - \mathbf{c} \end{aligned} \tag{118}$$

- If the image $I$ is of size $M \times N$ with even values for $M$ and $N$, we can define the four points $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$ and $\mathbf{d}$ by

$$\mathbf{a} = \begin{bmatrix} \frac{M}{2} - 1 \\ 0 \\ 1 \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} 0 \\ \frac{N}{2} - 1 \\ 1 \end{bmatrix} \quad \mathbf{c} = \begin{bmatrix} \frac{M}{2} - 1 \\ N - 1 \\ 1 \end{bmatrix} \quad \mathbf{d} = \begin{bmatrix} M - 1 \\ \frac{N}{2} - 1 \\ 1 \end{bmatrix} \tag{119}$$

- Let $\hat{\mathbf{a}}$, $\hat{\mathbf{b}}$, $\hat{\mathbf{c}}$ and $\hat{\mathbf{d}}$ denote the locations of the four pixels after they are subject to the combined effect of $H_{sim}H_p$. And let $\hat{\mathbf{x}}$ and $\hat{\mathbf{y}}$ be the vectors that correspond to $\mathbf{x}$ and $\mathbf{y}$ defined in Eq. (118). Obviously,

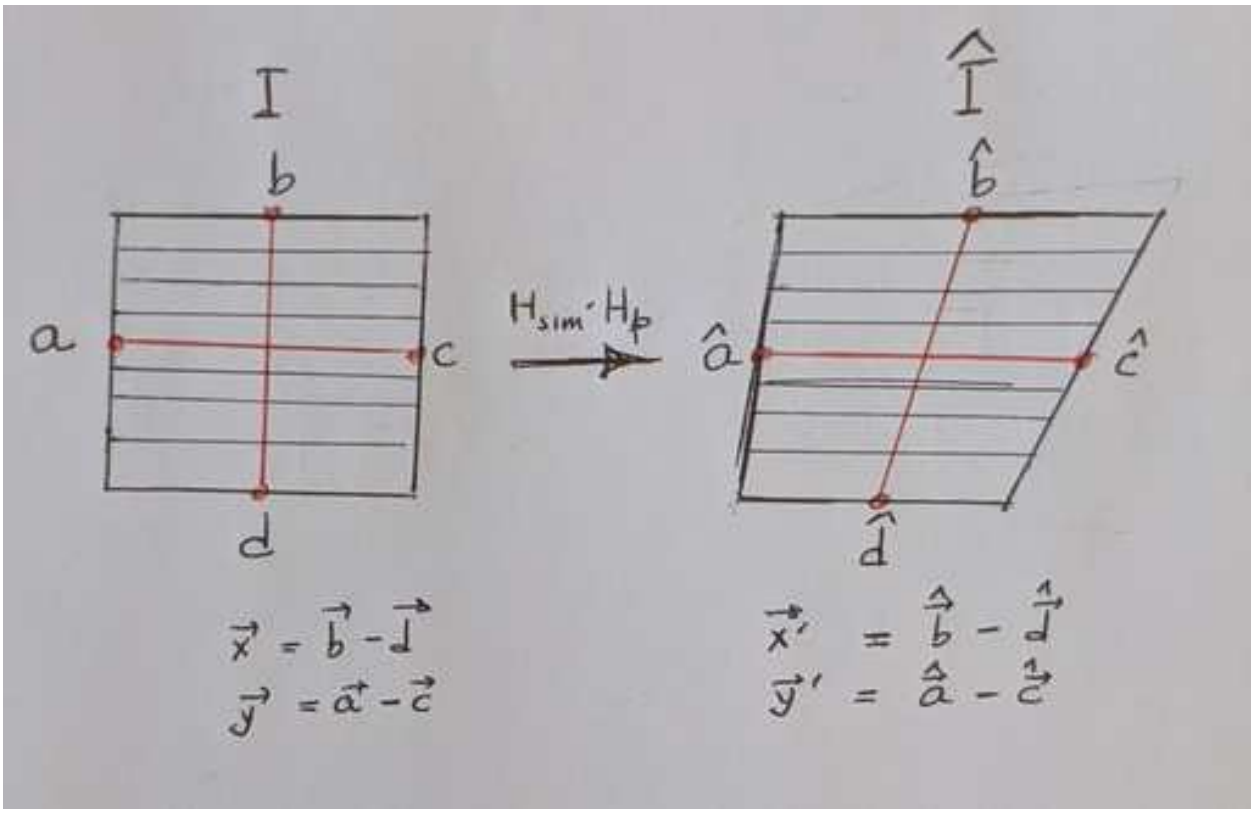



Figure 4: *Defining the "a-c" and "b-d" vectors for the Shearing Homography*

$$\begin{aligned} \hat{\mathbf{x}} &= \hat{\mathbf{b}} - \hat{\mathbf{d}} \\ \hat{\mathbf{y}} &= \hat{\mathbf{a}} - \hat{\mathbf{c}} \end{aligned} \tag{120}$$

- Assume that the shearing homography $H_{sh}$ when applied to the output of $H_{sim}H_p$ takes the points $(\hat{\mathbf{a}}, \hat{\mathbf{b}}, \hat{\mathbf{c}}, \hat{\mathbf{d}})$ to the points $(\hat{\hat{\mathbf{a}}}, \hat{\hat{\mathbf{b}}}, \hat{\hat{\mathbf{c}}}, \hat{\hat{\mathbf{d}}})$. The vectors $\hat{\mathbf{x}}$ and $\hat{\mathbf{y}}$ will therefore transform into $\hat{\hat{\mathbf{x}}}$ and $\hat{\hat{\mathbf{y}}}$.

$$\begin{aligned} \hat{\hat{\mathbf{x}}} &= H_{sh}\hat{\mathbf{x}} \\ \hat{\hat{\mathbf{y}}} &= H_{sh}\hat{\mathbf{y}} \end{aligned} \tag{121}$$

- The extent of the correction provided by $H_{sh}$ can now be measured by the degree to which the output vectors $\hat{\hat{\mathbf{x}}}$ and $\hat{\hat{\mathbf{y}}}$ are perpendicular and the degree to which the ratio of their magnitudes corresponds to the aspect ratio of the original image $I$. That is, we want to set the elements of $H_{sh}$ so that

$$(H_{sh}\hat{\mathbf{x}})^T(H_{sh}\hat{\mathbf{y}}) = 0 \tag{122}$$

$$\frac{(H_{sh}\hat{\mathbf{x}})^T(H_{sh}\hat{\mathbf{x}})}{(H_{sh}\hat{\mathbf{y}})^T(H_{sh}\hat{\mathbf{y}})} = \frac{M^2}{N^2} \tag{123}$$

- Eqs. (122) and (123) give us two quadratic forms that can be solved for the two unknowns in the definition of $H_{sh}$ in Eq. (116). The solution up to a sign is given by

$$s_a = \frac{M^2 x_v^2 + N^2 y_v^2}{MN(x_v y_u - x_u y_v)} \tag{124}$$

$$s_b = \frac{M^2 x_u x_v + N^2 y_u y_v}{MN(x_u y_v - x_v y_u)} \tag{125}$$

where $x_u$ and $x_v$ represent the horizontal and the vertical coordinates of the vector $\hat{x}$ and $y_u$ and $y_v$ the same for the vector $\hat{y}$.

- Loop and Zhang say that the combined transform $H = H_{sh} H_{sim} H_p$ for the image $I$ and $H' = H'_{sh} H'_{sim} H'_p$ for image $I'$ will rectify the images with minimal distortion. However, the two images thus transformed may NOT be of appropriate size and may also have undergone undesirable translational shifts along the "vertical" direction. That is, while the images will be row-wise aligned, they may both appear displaced vertically. Note that any translations along the world-X are of no consequence. They say that it may therefore be necessary to apply additional uniform scaling and vertical translation to the images.

Back to TOC

# 10: A C++ Implementation of the Algorithm by Álvarez and García

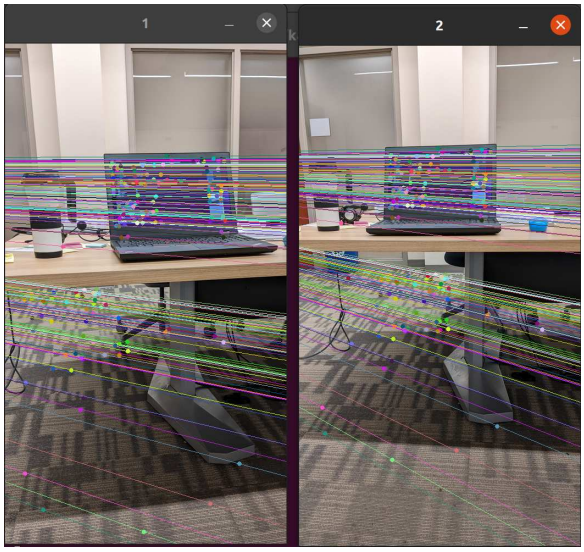
- GitHub has a great C++ implementation of the Loop and Zhang algorithm by Antonio Álvarez and Alejandro García:

`https://github.com/agarciamontoro/image-rectification`

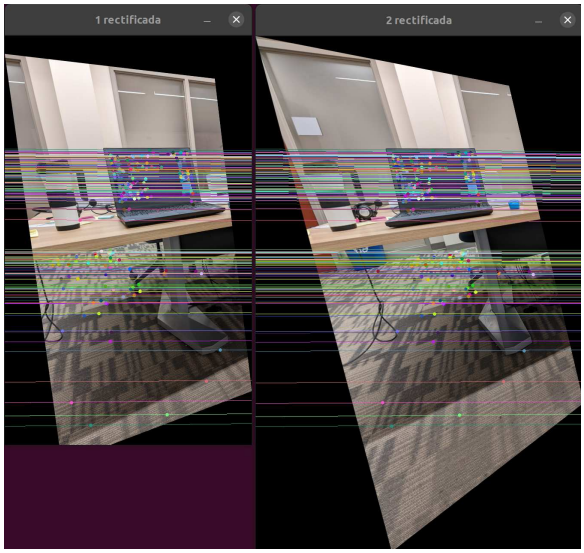
- In order to get the code to run with the latest version of OpenCV, Fangda Li had to make changes to the file `util.cpp` in the `src` directory of the repository.

- Shown in Figure 5 are the results obtained with the algorithm on a pair of images I recorded with my cellphone camera in my work area in RVL. **The lines superimposed on the images are the epipolar lines.** After you have estimated the fundamental matrix $F$, for a given keypoint $\mathbf{x}$ in, say, the left image, its epipolar line in the right image would be given by $l = F \cdot \mathbf{x}$. It is these lines that are drawn superimposed on the images.

- Note that it is entirely possible that the algorithm would fail to rectify a pair of stereo images if they were taken from two




Original Images

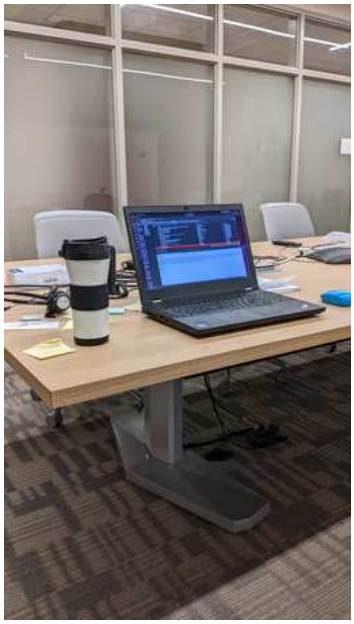



Rectified Images

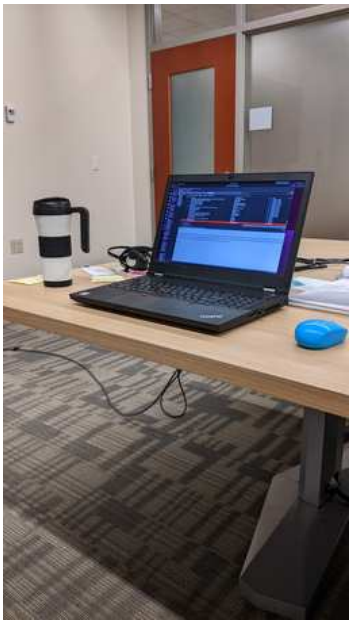
Figure 5: *The pre- and post-rectification results for two stereo images recorded with my cellphone in my RVL work area.*

viewpoints that are much too converging. In such cases, the background in the two images may be so different that the algorithm does not find a sufficient number of matchable keypoints for estimating the Fundamental Matrix $F$. When that happens, the algorithm is likely to fail with an error report that may look like "Floating exception (core dumped)". Shown in Figure 6 is a pair of images of roughly the same scene in my RVL work area but with the optic axes of the cameras subtending an angle of approximately 45°. In this case the algorithm failed.

- In the rest of this section, I'll present my annotated version of the file `main.cpp` that you will find in the `src` directory of the GitHub repository. My annotations will show the connections



Left Image



Right Image

Figure 6: *The two images for which the algorithm fails.*

between the equations in this Reader and the statements in the code.

- As you will notice, the Álvarez and García implementation uses the `cv::Mat` class for constructing the containers needed for the arrays (and matrices). For example, in Lines (3) and (4) in the code shown below, the instances constructed from this C++ class are used to directly store the two images in the form of their array representations. In Line (5), the same container type is used to initialize a $3 \times 3$ array for holding the Fundamental Matrix.

- Line (6) declares a 3-vector for holding the epipole of the left

camera.

- Line (7) declares two `vector` containers for holding the epipolar lines in the two images. And Line (8) declares another type of a `vector` container for holding the 2D points matched between the two images for estimating the Fundamental matrix.

- The call to `computeEpiLines()` code in Lines (9) through (12) is for estimating the epipolar geometry for the two images. The implementation code for this function is in the file `util.cpp` in the same `src` directory that has the `main.cpp` file. That call detects the keypoints in the two images, finds the best matching tie-points, and uses them for estimating the Fundamental Matrix.

- About the "`.at()`" syntax you see in Lines (21) through (42), that is used by the container instances constructed from the `cv::Mat` class to return references to the elements stored in the containers. If you are not familiar with the `cv::Mat` class in OpenCV, here is a link to the reference manual:

  `https://docs.opencv.org/4.x/d3/d63/classcv_1_1Mat.html`

- For the code lines beyond Line (12) and up to Line (48), I have indicated in my annotations as to which equation in the Reader is relevant to the line in question.

```
//  This code file is my annotated version of the "main.cpp" in the
//  GitHub repository created by Antonio Álvarez and Alejandro
//  García.  Here is the link to their GitHub page:
//
//      https://github.com/agarciamontoro/image-rectification

//  My goal here is to establish connections between statements in the
//  code shown below and the equation numbers in my "Loop and Zhang"
//  Reader.  The right-most comment, when it is there, will point to
//  the relevant equation in the Reader.
//
//  The first 48 lines of the code are for estimating the rectifying
//  homographies for the two images of a stereo paper.  The rest of
//  the code is for applying the estimated homographies to the images
//  and displaying the epipolar lines before and after rectification.
//
//  The annotation at the right end of each of the first 48 statements
//  starts with a statement number followed by the equation number in
//  the Reader that is relevant to that statement.  I have numbered
//  the statements for convenience in referring to them should there
//  be any questions about my comment that follows.

//  Code Authors:    Antonio Álvarez and Alejandro García
//  Annotations by   Avi Kak  (kak@purdue.edu)
//  Date:            November 15, 2022


#include <opencv2/core/core.hpp>
#include <opencv2/highgui/highgui.hpp>

#include "util.hpp"

#include <iostream>
#include <cctype>

using namespace cv;
using namespace std;

int main(){                                                                     // (1)

    /****************** EPIPOLAR GEOMETRY *************************/
    printf("OpenCV: %s", getBuildInformation().c_str());                        // (2)

    Mat img_1 = imread("../img/perra_7.jpg");                                   // (3)
    Mat img_2 = imread("../img/perra_8.jpg");                                   // (4)

    // Mat img_1 = imread("../img/madera_1.jpg");                               // (3)
    // Mat img_2 = imread("../img/madera_2.jpg");                               // (4)

    // Mat img_1 = imread("../img_avi/left_test.jpg");                          // (3)
    // Mat img_2 = imread("../img_avi/right_test.jpg");                         // (4)

    Mat fund_mat = Mat::zeros(3,3,CV_64F);                                      // (5)

    Vec3d epipole;                                                              // (6)  left camera epipole

    vector<Vec3d> lines_1, lines_2;                                             // (7)  epilines in img 1,2
    vector<Point2d> good_matches_1, good_matches_2;                             // (8)

    // Get epipolar geometry
    computeEpiLines(img_1, img_2,                                               // (9)
                    epipole, fund_mat,                                          // (10)
                    lines_1, lines_2,                                           // (11)
                    good_matches_1, good_matches_2);                            // (12)

    Mat A, B, Ap, Bp;                                                           // (13) Eqs. 78 -- 81

    Mat e_x = crossProductMatrix(epipole);                                      // (14) left camera epipole
                                                                                //   in cross-product rep


    /****************** PROJECTIVE *************************/                  //  Section 7, page 31

    // Get A,B matrix for minimizing z
    obtainAB(img_1, e_x, A, B);                                                 // (15)  Eqs, 78, 79
    obtainAB(img_2, fund_mat, Ap, Bp);                                          // (16)  Eqs. 80, 81
```

```
// Get initial guess for z
Vec3d z = getInitialGuess(A, B, Ap, Bp);                                    // (17)  top of page 42

// Optimizes the z solution
optimizeRoot(A, B, Ap, Bp, z);                                              // (18)  Eqs. 90 -- 96

// Get w
Mat w = e_x * Mat(z);                                                       // (19)  Eq. (62) with sol for z
Mat wp = fund_mat * Mat(z);                                                 // (20)  Eq. (63) with sol for z

w /= w.at<double>(2,0);                                                     // (21)  Eq. (51), norm.: w_2 = 1
wp /= wp.at<double>(2,0);                                                   // (22)  Eq. (52), norm.: w'_2 = 1
                                                                            // In these statements, w is a
                                                                            // 3-ele col vector being stored
                                                                            // as (3,0) cv::Mat matrix object
// Get final H_p and Hp_p matrix for projection
Mat H_p = Mat::eye(3, 3, CV_64F);                                           // (23)  Eq. (51)   initialzation
H_p.at<double>(2,0) = w.at<double>(0,0);                                    // (24)  Eq. (51)
H_p.at<double>(2,1) = w.at<double>(1,0);                                    // (25)  Eq. (51)

Mat Hp_p = Mat::eye(3, 3, CV_64F);                                          // (26)  Eq. (52)   initialization
Hp_p.at<double>(2,0) = wp.at<double>(0,0);                                  // (27)  Eq. (52)
Hp_p.at<double>(2,1) = wp.at<double>(1,0);                                  // (28)  Eq. (52)



/****************** SIMILARITY *************************/                  // Section 8, page 44

// Get the translation term
double vp_c = getTranslationTerm(img_1, img_2, H_p, Hp_p);                  // (29) See last bullet on page 50
                                                                            // vp_c here is the same as v'_c


// Get the H_r and Hp_r matrix directly
Mat H_r = Mat::zeros(3, 3, CV_64F);                                         // (30)  initialization
                                                                            // H_r here is the same as H_sim
                                                                            // Hp_r here is the same as H'_sim
                                                                            // The job of H_r is to rotate the z-
                                                                            // so that it coincides with World-X

H_r.at<double>(0,0) = fund_mat.at<double>(2,1) - w.at<double>(1,0) * fund_mat.at<double>(2,2);  // (31)   Eq. (114)
H_r.at<double>(1,0) = fund_mat.at<double>(2,0) - w.at<double>(0,0) * fund_mat.at<double>(2,2);  // (32)   Eq. (114)

H_r.at<double>(0,1) = w.at<double>(0,0) * fund_mat.at<double>(2,2) - fund_mat.at<double>(2,0);  // (33)   Eq. (114)
H_r.at<double>(1,1) = H_r.at<double>(0,0);                                                      // (33)   Eq. (114)

H_r.at<double>(1,2) = fund_mat.at<double>(2,2) + vp_c;                      // (34)   Eq. (114)
H_r.at<double>(2,2) = 1.0;                                                  // (35)   Eq. (114)


Mat Hp_r = Mat::zeros(3, 3, CV_64F);                                        // (36)  Hp_r is the same as H'_sim

Hp_r.at<double>(0,0) = wp.at<double>(1,0) * fund_mat.at<double>(2,2) - fund_mat.at<double>(1,2); // (37)   // Eq. (115)
Hp_r.at<double>(1,0) = wp.at<double>(0,0) * fund_mat.at<double>(2,2) - fund_mat.at<double>(0,2); // (38)   // Eq. (115)

Hp_r.at<double>(0,1) = fund_mat.at<double>(0,2) - wp.at<double>(0,0) * fund_mat.at<double>(2,2); // (39)   // Eq. (115)
Hp_r.at<double>(1,1) = Hp_r.at<double>(0,0);                                                     // (40)   // Eq. (115)

Hp_r.at<double>(1,2) = vp_c;                                                // (41)   // Eq. (115)
Hp_r.at<double>(2,2) = 1.0;                                                 // (42)   // Eq. (115)



/****************** SHEARING *************************/                   // Section 9, page 51

Mat H_1 = H_r*H_p;                                                          // (43)  combining prev 2 homographies
Mat H_2 = Hp_r*Hp_p;                                                        // (44)  combining prev 2 homographies

Mat H_s, Hp_s;                                                              // (45)  the shear homographies

// Get shearing transforms with the method described on the paper
getShearingTransforms(img_1, img_2, H_1, H_2, H_s, Hp_s);                   // (46)  Eqs. (116) -- (125)
```

```
/****************** RECTIFY IMAGES **********************/

Mat H = H_s * H_r * H_p;                                                    // (47)  Eq. (47)
Mat Hp = Hp_s * Hp_r * Hp_p;                                                // (48)  Eq. (48)


// Get homography image of the corner coordinates from all the images
vector<Point2d> corners_all(4), corners_all_t(4);
double min_x, min_y, max_x, max_y;
min_x = min_y = +INF;
max_x = max_y = -INF;

corners_all[0] = Point2d(0,0);
corners_all[1] = Point2d(img_1.cols,0);
corners_all[2] = Point2d(img_1.cols,img_1.rows);
corners_all[3] = Point2d(0,img_1.rows);

perspectiveTransform(corners_all, corners_all_t, H);

for (int j = 0; j < 4; j++) {
    min_x = min(corners_all_t[j].x, min_x);
    max_x = max(corners_all_t[j].x, max_x);

    min_y = min(corners_all_t[j].y, min_y);
    max_y = max(corners_all_t[j].y, max_y);
}

int img_1_cols = max_x - min_x;
int img_1_rows = max_y - min_y;

// Get homography image of the corner coordinates from all the images
min_x = min_y = +INF;
max_x = max_y = -INF;

corners_all[0] = Point2d(0,0);
corners_all[1] = Point2d(img_2.cols,0);
corners_all[2] = Point2d(img_2.cols,img_2.rows);
corners_all[3] = Point2d(0,img_2.rows);

perspectiveTransform(corners_all, corners_all_t, Hp);

for (int j = 0; j < 4; j++) {
    min_x = min(corners_all_t[j].x, min_x);
    max_x = max(corners_all_t[j].x, max_x);

    min_y = min(corners_all_t[j].y, min_y);
    max_y = max(corners_all_t[j].y, max_y);
}

int img_2_cols = max_x - min_x;
int img_2_rows = max_y - min_y;

// Apply homographies
Mat img_1_dst(img_1_rows, img_1_cols, CV_64F);
Mat img_2_dst(img_2_rows, img_2_cols, CV_64F);

warpPerspective( img_1, img_1_dst, H, img_1_dst.size() );
warpPerspective( img_2, img_2_dst, Hp, img_2_dst.size() );

Vec3d epipole_dst;

vector<Vec3d> lines_1_dst, lines_2_dst;
vector<Point2d> good_matches_1_dst, good_matches_2_dst;

perspectiveTransform(good_matches_1, good_matches_1_dst, H);
perspectiveTransform(good_matches_2, good_matches_2_dst, Hp);

// Get epipolar geometry and draw epilines
computeEpiLines(img_1_dst, img_2_dst, epipole_dst, fund_mat, lines_1_dst, lines_2_dst, good_matches_1_dst, good_matches_2_dst)

drawEpilines(img_1, img_2, lines_1, lines_2, good_matches_1, good_matches_2, 150);
```

```
    drawEpilines(img_1_dst, img_2_dst, lines_1_dst, lines_2_dst, good_matches_1_dst, good_matches_2_dst, 150);

    cout << "\nH = " << H << "\nHp = " << Hp << endl;

    cout << "\nEpipolo antes: " << epipole/epipole[2] << "\nEpipolo después: " << epipole_dst << endl;

    draw(img_1, "1");
    draw(img_1_dst, "1 rectificada");

    char c = 'a';

    draw(img_2, "2");
    draw(img_2_dst, "2 rectificada");

    c = 'a';

    while (tolower(c) != 'q')
      c = waitKey();

    destroyAllWindows();

}
```