

CS 4495/7495 Computer Vision

## Assignment 2: Multiview Geometry (Part3 updated)

3rd October 2006

### Introduction

There are three parts in this assignment. The first part gives you the idea of stereo triangulation. The second part gives you hands-on experience with the use of epipolar constraints to recover the fundamental matrix in a 2-view system. The third part is to compute the trifocal tensor for a 3-view system. Each section also has a piece at the end titled 'Deliverables' that contains the thing you should turn in related to that section. The first and second part are due on 10/03 and the third part is due on 10/10.

### 1 Stereo triangulation

Given a calibrated stereo rig and two matching image points  $p$  and  $p'$  (3-vector homogeneous coordinate), we could reconstruct the 3D point  $P$  by intersecting the two rays. With an algebraic approach, suppose the  $3 \times 4$  projection matrices for two cameras are denoted by  $M$  and  $M'$ , we could write the constraints  $z p = MP$  and  $z' p' = M'P$  as:

$$\begin{cases} p \times MP = 0 \\ p' \times M'P = 0 \end{cases} \Leftrightarrow \begin{pmatrix} [p_x]M \\ [p'_x]M' \end{pmatrix} P = 0,$$

where  $[p_x] = \begin{bmatrix} 0 & -p_z & p_y \\ p_z & 0 & -p_x \\ -p_y & p_x & 0 \end{bmatrix}$ . This whole equation could be written as  $AP = 0$ , which

is an over-constrained system of four independent linear equations. To solve  $P$ , we could use SVD on  $A$  to perform a matrix factorization such that  $A = UDV^T$ . In which, the diagonal matrix  $D = \text{diag}\{\delta_1, \delta_2, \delta_3, \dots, \delta_n\}, \delta_1 \geq \delta_2 \geq \delta_3 \geq \dots \geq \delta_n$ .  $P$  will be the last column of  $V$  corresponding to the smallest eigenvalue  $\delta_n$ . Matlab code is provided in <http://swiki.cc.gatech.edu:8080/cs4495-f104/59>.

## 1.1 Deliverables

1. Please download the package here [http://www-static.cc.gatech.edu/classes/AY2007/cs4495\\_fall/Material](http://www-static.cc.gatech.edu/classes/AY2007/cs4495_fall/Material). There are two images, Book1.jpg and Book2.jpg, taken from two calibrated cameras with the projection matrix  $M_1, M_2$  (available in cam\_book1.txt and cam\_book2.txt). We have manually established a pair of matching feature points in the images. The pixel coordinate of the feature point in Book1.jpg is (130, 220, 1) and the coordinate for corresponding feature in Book2.jpg is (246, 211, 1). Solve the 3D point.
2. Verify it by projecting it to the third image Book3.jpg taken from the camera with projection matrix  $M_3$  available in cam\_book3.txt. Label the point in the image. Is it in the right place?

## 2 The Eight-point algorithm

### 2.1 Main Idea

In class we discussed the linear 8-point algorithm to recover the fundamental matrix given corresponding points  $(p_i, p'_i)$  in two views, where  $i \in 1..N$  and  $N \geq 8$ . In this assignment you are asked to implement and use the 8-point algorithm for manually picked correspondences. The rest of this document describes the process in greater detail.

There are two images, mantle1.jpg and mantle2.jpg, available from the class website. We have manually established eight features in the images that correspond to each other.

The pixel coordinates of the features in mantle1.jpg are

(330, 620), (304, 414), (760, 526), (506, 568), (706, 512), (36, 298), (868, 188), (916, 410)

and the corresponding pixel coordinates in mantle2.jpg are

(204, 558), (168, 384), (642, 574), (378, 552), (612, 548), (160, 248), (900, 232), (870, 492)

The fundamental matrix has rank 2 and 7 degrees of freedom, and hence, can be recovered from only seven correspondences. However, the algorithm is much simpler if we have eight correspondences. If  $(u, v)$  and  $(u', v')$  are two corresponding points in the two images, and  $F$  is the fundamental matrix, the epipolar constraint is

$$(u, v, 1) \begin{pmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{pmatrix} \begin{pmatrix} u' \\ v' \\ 1 \end{pmatrix} = 0 \quad (1)$$

where we have written out the individual elements of the fundamental matrix.

Given 8 such correspondences, we can recover the matrix  $F$  from the homogeneous system

$$\begin{pmatrix} u_1 u'_1 & u_1 v'_1 & u_1 & v_1 u'_1 & v_1 v'_1 & v_1 & u'_1 & v'_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_8 u'_8 & u_8 v'_8 & u_8 & v_8 u'_8 & v_8 v'_8 & v_8 & u'_8 & v'_8 & 1 \end{pmatrix} \begin{pmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \\ F_{33} \end{pmatrix} = AF = 0$$

The least-squares solution for  $F$  is the eigenvector corresponding to the smallest eigenvalue of  $A$ , that is, the last column of  $V$  in the SVD,  $A = UDV^T$ . This is the uncorrected fundamental matrix  $\tilde{F}$ , as the rank 2 constraint is not enforced using this procedure.

### 2.1.1 Deliverables

You should implement and run the eight-point algorithm using the provided test images to obtain the fundamental matrix  $\tilde{F}$ . Plot 15-20 epipolar lines in `mantle2.jpg`, obtained by varying the pixel location  $x$  in the equation  $l = \tilde{F}^T x$ , in `mantle1.jpg`. Your deliverables are the code for the eight-point algorithm, the uncorrected fundamental matrix  $\tilde{F}$ , and the images respectively with your chosen features and the epipolar lines drawn on it.

## 2.2 Rank Correction

To obtain the correct rank 2 fundamental matrix, we again use SVD on  $\tilde{F}$  to perform a matrix factorization (as in assignment 1). Let the diagonal matrix obtained from the SVD be written as  $D = \text{diag}(r, s, t)$ . Then the correct rank 2 fundamental matrix  $\hat{F}$  is given by  $\hat{F} = U \times \text{diag}(r, s, 0) \times V^T$ .

### 2.2.1 Deliverables

The deliverables are exactly the same, but now computing (showing) the rank-corrected fundamental matrix  $\hat{F}$ .

## 2.3 Normalization of Measurements

The eight-point algorithm is sensitive to scaling and translation in the measurements. Hence, normalization of the input measurements is required. The normalization is performed to center the measurements on the origin and make the mean distance of the measurements from the origin to

be  $\sqrt{2}$ . Note that the normalization is performed separately for measurements from each image as follows

- Center the measurements at origin by pre-multiplying each measurement in homogeneous coordinates by  $T = \begin{pmatrix} 1 & 0 & -m_x \\ 0 & 1 & -m_y \\ 0 & 0 & 1 \end{pmatrix}$  where  $(m_x, m_y)$  is the mean of the measurements.
- Scale the modified measurements to have mean length  $\sqrt{2}$  by pre-multiplying with the matrix  $S = \begin{pmatrix} \frac{\sqrt{2}}{d} & 0 & 0 \\ 0 & \frac{\sqrt{2}}{d} & 0 \\ 0 & 0 & 1 \end{pmatrix}$ , where  $d$  is the average length of the measurements after the above step.

After the normalization, the eight-point algorithm can be run on the measurements. After obtaining the fundamental matrix, we have to de-normalize it. This can be done by the step  $F'' = T_1^T S_1 F' S_2 T_2$ , where  $T_1$  and  $S_1$  are the normalization matrices for the features from the first image,  $T_2$  and  $S_2$  are the corresponding matrices for the second image, and  $F''$  is the true fundamental matrix obtained after the de-normalization.

### 2.3.1 Deliverables

Obtain the fundamental matrix using the provided test images, but this time with the normalization implemented. You should plot the epipolar lines in mantle2.jpg as before.

## 3 Triple Delight !

### 3.1 Warm up: Tensor representation for Fundamental Matrix

The fundamental matrix with tensor notation has two covariant indices:  $F_{AB}$ . Correspondence between two points in two views  $A$  and  $B$  are

$$F_{AB} x^A x^B = 0$$

We'd like to rewrite the equation above as  $(g_1, g_2, \dots, g_9) \begin{pmatrix} F_1 \\ F_2 \\ \vdots \\ F_9 \end{pmatrix} = 0$ , where  $(g_1, g_2, \dots, g_9)$  are the coefficients. Given 8 correspondences, there will be  $G_{8 \times 9} F = 0$ . Then we could use svd again to solve the tensor. The pseudocode given below is to solve  $F_{AB}$ :

```

For  $i = 1$  to  $N$  ( $N$  is number of correspondences eg.  $N = 8$ )
  For  $j = 1$  to 3
    For  $k = 1$  to 3
       $G(i, (j - 1) \times 3 + k) = x^j x^k$ 
    end
  end
end
svd( $G$ )

```

In each correspondence (there are 8 in total),  $x^j$  is the  $j^{\text{th}}$  element of the vector  $x^A = (u, v, 1)$  in view  $A$  and  $x^k$  is the  $k^{\text{th}}$  element of the vector  $x^B = (u', v', 1)$  in view  $B$ .

### 3.1.1 Deliverables

Given the same 8-point correspondences provided in part2, write a matlab program using the pseudocode above to solve the 9 elements of tensor  $F_{AB}$  and reshape it to  $3 \times 3$  matrix.

## 3.2 Trifocal tensor

Use 3 test images to form a 3-view system, and compute the trifocal tensor for this system using point-line-point correspondence.

### 3.2.1 Approach to solve the tensor using point-line-point correspondence.

Any line  $l_B$  in the second view defines a homography between points in view  $A$  and view  $C$ :

$$H_A^C = l_B T_A^{BC}$$

Thus, if we measure two points  $q^A$  and  $p^C$  in view  $A$  and  $C$  that correspond, and the corresponding 3D point lies on the 3D plane of which the line  $l_B$  is the projection in view  $B$ , we should have:

$$p^C \equiv r^C = H_A^C q^A = l_B T_A^{BC} q^A \quad (2)$$

where  $r^C$  is the predicted projection of  $q^A$  in view  $C$  via the homography  $H_A^C$ . Note  $\equiv$  denotes equivalence up to a scale, as always. We can make this into a set of three equalities using the by now familiar cross-product trick:

$$p \times r = 0$$

or, in tensor-land,

$$p^{C_1} r^{C_2} \varepsilon_{C_1 C_2 C} = 0_C$$

Note that in the above equations (there are three of them!) we renamed some of the  $C$  indices to avoid naming conflicts. Substituting in (2), we obtain

$$p^{C_1} l_B T_A^{BC_2} q^A \varepsilon_{C_1 C_2 C} = 0_C$$

This gives us three equations, only two of which are linearly independent. So we need at least 13 point-line-point correspondences to solve the tensor. Why ?

In more detail, Notice that  $C$  index ranges from 1 to 3, the three equations are:

$$p^2 l_B T_A^{B3} q^A \epsilon_{231} + p^3 l_B T_A^{B2} q^A \epsilon_{321} = 0_1$$

$$p^1 l_B T_A^{B3} q^A \epsilon_{132} + p^3 l_B T_A^{B1} q^A \epsilon_{312} = 0_2$$

$$p^1 l_B T_A^{B2} q^A \epsilon_{123} + p^2 l_B T_A^{B1} q^A \epsilon_{213} = 0_3$$

Where  $p^i$  is the  $i^{th}$  element of point  $p^{C1} = (u, v, 1)^T$ ,  $T_A^{B1}$  denotes the first 9 elements of the tensor,  $T_A^{B2}$  denotes the 9<sup>th</sup> to 18<sup>th</sup> elements of the tensor, and  $T_A^{B3}$  denotes the 18<sup>th</sup> to 27<sup>th</sup> elements of the tensor.  $\epsilon_{123}$ ,  $\epsilon_{231}$ , and  $\epsilon_{312}$  have the value of 1.  $\epsilon_{321}$ ,  $\epsilon_{132}$ , and  $\epsilon_{213}$  have the value of  $-1$ . We have

to rewrite the three equations above in the form of  $G_{3 \times 27} \begin{pmatrix} t_1 \\ t_2 \\ \vdots \\ t_{27} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$ , where  $G_{3 \times 27}$  is the

coefficient matrix and  $\begin{pmatrix} t_1 \\ t_2 \\ \vdots \\ t_{27} \end{pmatrix}$  represents all the 27 elements of the tensor. The pseudocode to get

$G_{3 \times 27}$  is:

```

For j=1 to 3
  For k=1 to 3
    G(1, (j-1) × 3+k) = 0
    G(2, (j-1) × 3+k) = p3 lk qj ε312
    G(3, (j-1) × 3+k) = p2 lk qj ε213
  End
End

```

```

For j=1 to 3
  For k=1 to 3
    G(1, 9+(j-1) × 3+k) = p3 lk qj ε321
    G(2, 9+(j-1) × 3+k) = 0
    G(3, 9+(j-1) × 3+k) = p1 lk qj ε123
  End
End

```

```

For j=1 to 3
  For k=1 to 3
    G(1, 18+(j-1) × 3+k) = p2 lk qj ε231
  End
End

```

$$G(2, 18+(j-1) \times 3+k) = p^1 l_k q^j \epsilon_{132}$$

$$G(3, 18+(j-1) \times 3+k) = 0$$

End  
End

The matrix  $G_{3 \times 27}$  is obtained through one point-line-point correspondence, where  $p^i$  is the  $i^{\text{th}}$  element of point  $p^{C_1} = (u', v', 1)^T$ ,  $q^j$  is the  $j^{\text{th}}$  element of point  $q^A = (u, v, 1)^T$ , and  $l_k$  is the  $k^{\text{th}}$  element of line  $l_B = (u'', v'', 1)^T$ . After getting 13 point-line-point correspondences, what you need to do is stack all the  $G_{3 \times 27}$  matrices (there are 13 of them) to form a  $G_{39 \times 27}$  matrix. We again use SVD on  $G_{39 \times 27}$  to perform a matrix factorization and the tensor will be the last column of  $V$  corresponding to the smallest eigenvalue. According to the pseudocode and index order above, the first 9 elements in the last column correspond to  $[T_1^{11}, T_1^{21}, T_1^{31}, T_2^{11}, T_2^{21}, T_2^{31}, T_3^{11}, T_3^{21}, T_3^{31}]$ , the 10<sup>th</sup> to 18<sup>th</sup> elements correspond  $[T_1^{12}, T_1^{22}, T_1^{32}, T_2^{12}, T_2^{22}, T_2^{32}, T_3^{12}, T_3^{22}, T_3^{32}]$ , and the last 9 elements in the column correspond to  $[T_1^{13}, T_1^{23}, T_1^{33}, T_2^{13}, T_2^{23}, T_2^{33}, T_3^{13}, T_3^{23}, T_3^{33}]$ .

### 3.2.2 Deliverables

View1.jpg, View2.jpg, View3.jpg, View4.jpg are provided online. At least 13 correspondences are required.

1. Use View1.jpg as view  $A$ , View2.jpg as view  $B$ , View4.jpg as view  $C$  to form 3-view system (the order is important). There are two images with 14 marked points and the other with one image line marked. Notice, the 14<sup>th</sup> point is in the bottom of the bulb of the lamp in the image. Use the first 13 points and the image line in View2.jpg to calculate the tensor. Is it a degenerate case? Indicate it from your SVD results.
2. Use View1.jpg as view  $A$ , View3.jpg as view  $B$ , View4.jpg as view  $C$  to form 3-view system. There's one image with multiple lines marked. In this case, we use 14 points. For each point correspondence, you should choose one line from it to be the correspondence. List your 14 correspondences in the form of  $\text{point}(u, v, 1)^T$  in  $A$ ,  $\text{point}(u', v', 1)^T$  in  $C$  and  $\text{line}(u'', v'', 1)^T$  in  $B$ . Solve the tensor  $T_A^{BC}$  and denote it as  $T_{\text{view1}}^{\text{view3view4}}$ .
3. Show one line-line-line correspondence obtained by choosing one 3D line in View3.jpg, then picking one corresponding line in View4.jpg, and showing its computed position in View1.jpg (Hint: The formula is  $l_A = T_A^{BC} l_B l_C$ ).

## 4 How to turn in

Put all your code, images, and computed matrices in a directory named `<name-emailPrefix>`, tar it or zip it, and email it to the TA ([cynthia@cc.gatech.edu](mailto:cynthia@cc.gatech.edu)). You should also hand in during class a hard copy of all the images with lines drawn.