

Tensors in Multiview Geometry

Frank Dellaert

21st October 2002

Lecture Notes for Georgia Tech course CS8803D, "Multiview Geometry in Computer Vision"

1 Tensors for Projective Geometry

For our purposes, simply think of tensors as multi-dimensional matrices, with two types of indices: *covariant* indices are indicated as subscripts, *contravariant* indices are indicated as superscripts. Following Triggs, we will adopt the convention that 2D projective spaces are denoted uppercase A,B,C, etc, and 3D space as lowercase a, b, c... The number and position of the indices determines the *valency* of the tensor.

2D Projective Space

The following presents the story in P^2 :

- A point is a contravariant tensor x^A .
- A line is a covariant tensor l_A .
- Point and line coincide (dot product): $l_A x^A = \sum_{A=1}^3 l_A x^A = 0$.
- Join of two points (cross product):

$$l_A = x^{A_1} x^{A_2} \epsilon_{A_1 A_2 A}$$

where ϵ is the permutation tensor, defined as

$$\epsilon_{ijk} \stackrel{\Delta}{=} 1 \text{ if } ijk \text{ is an even permutation}$$

$$\epsilon_{ijk} \stackrel{\Delta}{=} -1 \text{ if } ijk \text{ is an odd permutation}$$

$$\epsilon_{ijk} \stackrel{\Delta}{=} 0 \text{ if } ijk \text{ is not a permutation}$$

Note that the familiar skew-symmetric matrix is a covariant tensor with valency two:

$$x^{A_1} \epsilon_{A_1 A_2 A} = [x^{A_1}]_{\times A_2 A}$$

- Join of two lines:

$$x^A = l_{A_1} l_{A_2} \epsilon^{A_1 A_2 A}$$

- Homography: $x^A = H_B^A x^B$

3D Projective Space

This is analogous to 2D, but with lowercase indices, and we talk about planes π_a instead of lines l_A .

- A 3D point is a contravariant tensor x^a .
- It's dual, a *plane* is a covariant tensor π_a .
- Point and plane coincide (dot product): $\pi_a x^a = 0$.
- Join of three points (In 3D, we will use i, j, k instead of the more cumbersome a_1, a_2, a_3):

$$\pi_l = x^i x^j x^k \epsilon_{ijkl}$$

- Join of three planes:

$$x^l = \pi_i \pi_j \pi_k \epsilon^{ijkl}$$

- Homography: $x^a = H_b^a x^b$

2 The Projective Camera as a mixed 3D to 2D Tensor

The familiar 3×4 projective camera matrix can now be seen to be a mixed valency 2 tensor, with a contravariant index A denoting the 2D image, and a covariant index a for the 3D space:

$$P_a^A$$

and projection and back-projection operators can be implemented using simple contractions:

- 3D point projected to 2D: $x^A = P_a^A x^a$
- Backprojecting a 2D line to a plane: $\pi_a = P_a^A l_A$

3 Multiview Tensors: Overview

Two Views: The Fundamental Matrix

The *fundamental matrix* F encapsulates all projective properties between two views. Two cameras have 22 independent parameters between them, and projective properties are invariant to a 3D projective transformation with 15 degrees of freedom (DOF), which leaves exactly 7 DOF for the fundamental matrix. The two constraints on the 9 entries of F are (a) it is only defined up to a scale, and (b) F has rank 2.

The fundamental matrix has two covariant indices:

$$F_{AB}$$

with the following familiar relations:

- Epipolar line in view 2: $l_B = F_{AB} x^A$
- Epipolar line in view 1: $l_A = F_{AB} x^B$
- Correspondence between two points: $F_{AB} x^A x^B = 0$

Three Views: The Trifocal Tensor

Just like F encapsulates all projective properties between two views, the *trifocal tensor* T does this for three views. Three cameras have 33 independent parameters between them, which leaves $33 - 15 = 18$ DOF for the trifocal tensor. Since a $3 \times 3 \times 3$ tensor has 27 entries, this means there are 9 constraints on T : one scale ambiguity and 8 (non-trivial) algebraic constraints.

For three views A, B, and C, there are actually three different trifocal tensors, depending on which view is taken to be the "special" view:

$$T_A^{BC} \quad T_B^{AC} \quad T_C^{AB}$$

The following three view multi-linear constraints can be stated:

3.0.1 line-line-line

Two lines l_B and l_C corresponding to the same 3D line, define a third line in the first view:

$$l_A = T_A^{BC} l_B l_C$$

We can turn this into three homogeneous equations (of which only 2 are linearly independent) using the “cross-product” trick:

$$l_A l_B l_C T_A^{BC} \epsilon^{AA_1 A_2} = 0^{A_2}$$

3.0.2 point-line-line

If instead of a line l_A we measure a point x^A corresponding to a point on the 3D line defined by l_B and l_C , we have

$$l_A x^A = (T_A^{BC} l_B l_C) x^A = x^A l_B l_C T_A^{BC} = 0$$

3.0.3 point-line-point

Any line l_B in the second view defines a homography between points in view 1 and 3:

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

Thus, if we measure two points in views 1 and 3 that correspond, and lie on the 3D line of which l_B is the projection, we have:

$$x_C = H_A^C x^A = l_B T_A^{BC} x^A$$

or, using the cross product trick:

$$x^{C_1} x^{C_2} \epsilon_{C_1 C_2 C} = x^{C_1} (l_B T_A^{BC} x^A) \epsilon_{C_1 C_2 C} = x^A l_B x^{C_1} (T_A^{BC} \epsilon_{C_1 C_2 C}) = 0_{C_2}$$

3.0.4 point-point-line

Similar to above, but with views 2 and 3 switched.

3.0.5 point-point-point

Three corresponding points satisfy

$$x^A (x^B \epsilon_{BB_1 B_2}) (x^C \epsilon_{CC_1 C_2}) T_A^{B_1 C_1} = 0_{B_2 C_2}$$

yielding 9 equations, 4 of which are linearly independent. Explanation will follow.

Four Views: The Quadrifocal Tensor

The *quadrifocal tensor* Q captures the projective structure of 4 views in one object. Four views is the maximum number for which such an object exists: for 5 views or more such multi-linear constraints cannot be conveniently captured using a single tensor. Four cameras have 44 independent parameters between them, which leaves $44 - 15 = 29$ DOF for the quadrifocal tensor. Since a $3 \times 3 \times 3 \times 3$ tensor has 81 entries, this means there are $81 - 29 = 52$ constraints on Q .

Unlike for three views, there is only one tensor Q , with 4 contravariant indices:

$$Q^{ABCD}$$

4 Multiview Tensors: Derivation

Above the multiview tensors, together with the relationships they capture between points and lines in different views, were introduced without justification. Here we will see where they come from, and why they have the valency that they have. It is actually more intuitive to start this process with the quadrifocal tensor.

Four Views: The Quadrifocal Tensor

Take three arbitrary lines l_A , l_B , and l_C in the first three views. They all back-project to a plane in 3D, say $\pi_i = P_i^A l$, $\pi_j = P_j^B l_B$, and $\pi_k = P_k^C l_C$, and we know that three planes join in a point:

$$x^l = \pi_i \pi_j \pi_k \varepsilon^{ijkl} = l_A l_B l_C P_i^A P_j^B P_k^C \varepsilon^{ijkl}$$

Any other line l_D in the fourth view, corresponding to a 3D line that intersects this same point x^l , will contain the projection $P_l^D x^l$, giving the constraint:

$$l_D (P_l^D x^l) = l_D P_l^D (l_A l_B l_C P_i^A P_j^B P_k^C \varepsilon^{ijkl}) = l_A l_B l_C l_D (P_i^A P_j^B P_k^C P_l^D \varepsilon^{ijkl}) = 0$$

The tensor between parentheses in the last expression only depends on the entries of the 4 projection matrices, and is named the *quadrifocal tensor*:

$$Q^{ABCD} \triangleq P_i^A P_j^B P_k^C P_l^D \varepsilon^{ijkl}$$

It is invariant to a projective transformation of 3-space, as it does not involve any 3D indices ! The 4-line correspondence can now conveniently be written as:

$$l_A l_B l_C l_D Q^{ABCD} = 0$$

Two Views: The Fundamental Matrix

A similar story can be spun for the fundamental matrix, but using a trick to represent a point as the join of two lines. Suppose we have two corresponding points x^A and x^B in the respective views. In the first view, take any two lines l_{A_1} and l_{A_2} that join in x^A . Similarly, in the second view, take any two lines l_{B_1} and l_{B_2} . Then, as with the quadrifocal tensor, we can state that all backprojected planes corresponding to those lines meet in a common point:

$$(P_i^{A_1} l_{A_1}) (P_j^{A_2} l_{A_2}) (P_k^{B_1} l_{B_1}) (P_l^{B_2} l_{B_2}) \varepsilon^{ijkl} = l_{A_1} l_{A_2} l_{B_1} l_{B_2} (P_i^{A_1} P_j^{A_2} P_k^{B_1} P_l^{B_2} \varepsilon^{ijkl}) = 0$$

Now, since $x^A = l_{A_1} l_{A_2} \varepsilon^{A_1 A_2 A}$, and hence $l_{A_1} l_{A_2} = x^A \varepsilon_{A_1 A_2 A}$, we substitute this in the above and write:

$$(x^A \varepsilon_{A_1 A_2 A}) (x^B \varepsilon_{B_1 B_2 B}) (P_i^{A_1} P_j^{A_2} P_k^{B_1} P_l^{B_2} \varepsilon^{ijkl}) = x^A x^B (P_i^{A_1} P_j^{A_2} P_k^{B_1} P_l^{B_2} \varepsilon_{A_1 A_2 A B_1 B_2 B}^{ijkl}) = 0$$

Again, we see that the tensor between parentheses only depends on the camera tensors, and is projectively invariant. In addition, in the above we recognize the familiar epipolar correspondence condition, and we recognize the tensor as the fundamental matrix:

$$F_{AB} \triangleq P_i^{A_1} P_j^{A_2} P_k^{B_1} P_l^{B_2} \varepsilon_{A_1 A_2 A B_1 B_2 B}^{ijkl}$$

Three Views: The Trifocal Tensor

By now, the storyline should become familiar. For the trifocal tensor, it is easiest to work from the point-line-line correspondence, where the two lines l_B and l_C correspond to the *same* 3D line. Using the same trick of representing the point x^A using any two lines l_{A_1} and l_{A_2} that join in it, we can again state that all backprojected planes should intersect in one 3D point:

$$(P_i^{A_1} l_{A_1}) (P_j^{A_2} l_{A_2}) (P_k^B l_B) (P_l^C l_C) \varepsilon^{ijkl} = l_{A_1} l_{A_2} l_B l_C (P_i^{A_1} P_j^{A_2} P_k^B P_l^C \varepsilon^{ijkl}) = 0$$

Since since $x^A = l_{A_1} l_{A_2} \varepsilon^{A_1 A_2 A}$, we have

$$(x^A \varepsilon_{A_1 A_2 A}) l_B l_C (P_i^{A_1} P_j^{A_2} P_k^B P_l^C \varepsilon^{ijkl}) = x^A l_B l_C (P_i^{A_1} P_j^{A_2} P_k^B P_l^C \varepsilon_{A_1 A_2 A}^{ijkl}) = 0$$

and we recognize a trifocal tensor (centered on the first view) between the parentheses:

$$T_A^{BC} \triangleq P_i^{A_1} P_j^{A_2} P_k^B P_l^C \varepsilon_{A_1 A_2 A}^{ijkl}$$

Discussion

In all three cases, the geometric object that describes the multiview relationship was derived from an incidence constraint in three-space. In all cases, we found a tensor that does not contain any indices associated with 3D space, and hence is invariant to any projective transformations in 3D. Of course, projective transformations in the views *will* change the tensors. The valency of the 5 different tensors discussed above (one F , three T and one Q) suggests the correspondence constraints between different objects in the different views.