Tutorial:
Multiple View Geometry

(in conjunction with ICPR2000)

Time : 9.00-13.20

Anders Heyden

Centre for Mathematical Sciences

Lund University

SWEDEN

How to Contact me:

Address:

Anders Heyden
Centre for Mathematical Sciences
Lund University
Box 118
SE-221 00 Lund
SWEDEN

e-mail:

heyden@maths.lth.se

Contents:

1. Introduction
2. Tensor Calculus
3. Modeling cameras
4. Projective geometry
5. Multiple view tensors
6. Linear estimation of tensors
7. Tensorial transfer
8. Factorization and bundle adjustment
9. Flexible calibration
0. Conclusions

Coffee-break: 10.30-11.00

1. Introduction

Motivation

Reconstructing the three-dimensional world from a number of its two-dimensional perspective images.

Applications:

- Building autonomous vehicles
- Robotics
- Making CAD-models of old buildings where blueprints are missing
- Making precise 3D measurements of big industrial parts
- Analyzing car-crash tests in 3D

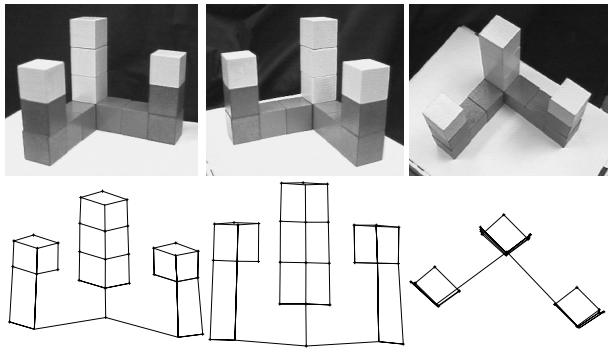
Why using the tensorial description?

- Minimal parameterization without internal Gauge freedoms
- Simple transfer formulas
- Canonical description of multiple view geometry that is
  (i) compact and
  (ii) elegant
- Both point and line features can be used
- A complete characterization of the geometry

Goals

- Theoretical understanding of MVG
- Knowledge of how the tensors can be used to solve computer vision problems
- Understanding of how the tensors can be used in some different applications:
  (i) structure and motion
  (ii) view synthesis
  (iii) the correspondence problem (RANSAC)
- Ability to implement and use the basic techniques for estimating tensor components and reconstruct a scene

Example of a Reconstruction



2. Tensor Calculus

Contents:

2.1 Introduction and Motivation

2.2 Cartesian tensors
- Definitions
- Coordinate Transformations
- Vectors
- Tensors
- Tensor Algebra
- Tensor Analysis

2.3 Affine tensors

2.4 General tensors

2.5 Conclusions

Suggested reading

. Spain, *Tensor Calculus*, University Mathematical Texts, Oliver and oyd, Edingburgh, 1953 (125 pages).

. G. Jaeger, *Cartesian Tensors in Engineering Science*, Pergamon Press, xford, 1966 (116 pages).

. Jeffreys, *Cartesian Tensors*, Cambridge University Press, 1931 3 pages).

. O. Myklestad, *Cartesian Tensors - the Mathematical Language of ngineering*, Van Nostrand, Princeton, 1967 (141 pages).

. Temple, *Cartesian Tensors*, Methuen's Monographs on Physical ubjects, London, 1960 (92 pages).

2.1 Introduction and Motivation

- A natural way to represent physical entities that are independent of the chosen coordinate system
- Gives a more compact notation
- Makes sure that formulas are independent of the chosen coordinate system

Applications:

- Mechanics, structural mechanics
- Electrodynamics, Maxwells equations
- Mathematical physics, theory of relativity

xample 1. Consider a particle moving freely in a Cartesian coordinate ystem subjected to the forces F_x, F_y and F_z in the three coordinate irections. Let (u_x, u_y, u_z) denote the position of the particle. Newtons rst law gives:

$$m\frac{d^2u_x}{dt^2} = F_x, \quad m\frac{d^2u_y}{dt^2} = F_y, \quad m\frac{d^2u_z}{dt^2} = F_z \ ,$$

hich may be written

$$m\frac{d^2u_i}{dt^2} = F_i \quad (i = x, y, z) \ ,$$

r even

$$m\frac{d^2u_i}{dt^2} = F_i \ ,$$

here $(i = x, y, z)$ are omitted. ■

Example 2. Consider a state of tension in a continuum. The conditions for equilibrium gives

$$\frac{\partial\sigma_{11}}{\partial x_1} + \frac{\partial\sigma_{21}}{\partial x_2} + \frac{\partial\sigma_{31}}{\partial x_3} = 0$$
$$\frac{\partial\sigma_{12}}{\partial x_1} + \frac{\partial\sigma_{22}}{\partial x_2} + \frac{\partial\sigma_{32}}{\partial x_3} = 0$$
$$\frac{\partial\sigma_{13}}{\partial x_1} + \frac{\partial\sigma_{23}}{\partial x_2} + \frac{\partial\sigma_{33}}{\partial x_3} = 0$$

hese equations can be written

$$\sum_{i=1}^{3} \frac{\partial \sigma_{ij}}{\partial x_i} = 0 \quad (j = 1,2,3) \ .$$

aking the values for i and j for granted gives

$$\sum \frac{\partial \sigma_{ij}}{\partial x_i} = 0 \ .$$

urthermore, introducing the convention that summation always takes lace when an index that appears twice

$$\frac{\partial \sigma_{ij}}{\partial x_i} = 0 \ .$$

inally, introcuding the index , i for $\frac{\partial}{\partial x_i}$ we arrive at

$$\sigma_{ij,i} = 0 \ .$$

■

Different Physical Entities

a) Scalars: Defined by **one** number.
Extensive scalars: properties of a body as a whole
Examples: mass, volume, energy
Intensive scalars: local properties that may vary from point to point
Examples: temperature, density, energy-density

b) Vectors: Defined by **three** numbers (in a 3D-space).
Free vectors: Pairs of forces defining a moment
Line dependent: Forces on rigid bodies
Point dependent: Forces on deformable bodies

When the vector varies from point to point we have a **vector field**, whith **field-lines** and we can calculate **divergence** and **rotation**.

he **scalar product** between two vectors is defined acording to

$$\mathbf{u}.\mathbf{v} = \sum_{i=1}^{3} u_i v_i = u_i v_i \ .$$

he **vector product** between two vectors is denoted by $\mathbf{u} \times \mathbf{v}$.

) Second order tensors: Defined by **nine** numbers (in a 3D-space).
xamples:
) The stress tensor

$$\begin{bmatrix} \sigma_x & \tau_{xy} & \tau_{xz} \\ \tau_{yz} & \sigma_y & \tau_{yz} \\ \tau_{zx} & \tau_{zy} & \sigma_z \end{bmatrix} = \begin{bmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{xz} \\ \sigma_{yz} & \sigma_{yy} & \sigma_{yz} \\ \sigma_{zx} & \sigma_{zy} & \sigma_{zz} \end{bmatrix} = \begin{bmatrix} \sigma_{11} & \sigma_{12} & \sigma_{13} \\ \sigma_{21} & \sigma_{22} & \sigma_{23} \\ \sigma_{31} & \sigma_{32} & \sigma_{33} \end{bmatrix}$$

r more compactly

$$\sigma_{ij}$$

(ii) The tension-tensor

$$\begin{bmatrix} \epsilon_x & \frac{1}{2}\gamma_{xy} & \frac{1}{2}\gamma_{xz} \\ \frac{1}{2}\gamma_{yz} & \epsilon_y & \gamma_{yz} \\ \frac{1}{2}\gamma_{zx} & \frac{1}{2}\gamma_{zy} & \epsilon_z \end{bmatrix} = \begin{bmatrix} \epsilon_{xx} & \epsilon_{xy} & \epsilon_{xz} \\ \epsilon_{yz} & \epsilon_{yy} & \epsilon_{yz} \\ \epsilon_{zx} & \epsilon_{zy} & \epsilon_{zz} \end{bmatrix} = \begin{bmatrix} \epsilon_{11} & \epsilon_{12} & \epsilon_{13} \\ \epsilon_{21} & \epsilon_{22} & \epsilon_{23} \\ \epsilon_{31} & \epsilon_{32} & \epsilon_{33} \end{bmatrix} \rightarrow \epsilon_{ij}$$

(iii) The moments of inertia of a rigid body

$$\begin{bmatrix} J_x & D_{xy} & D_{xz} \\ D_{yz} & J_y & D_{yz} \\ D_{zx} & D_{zy} & J_z \end{bmatrix} = \begin{bmatrix} J_{xx} & J_{xy} & J_{xz} \\ J_{yz} & J_{yy} & J_{yz} \\ J_{zx} & J_{zy} & J_{zz} \end{bmatrix} = \begin{bmatrix} J_{11} & J_{12} & J_{13} \\ J_{21} & J_{22} & J_{23} \\ J_{31} & J_{32} & J_{33} \end{bmatrix} \rightarrow J_{ij}$$

Note the number of parameters used:
Scalars (zeroth order tensors) $3^0 = 1$
Vectors (first order tensors) $3^1 = 3$
Second order tensors $3^2 = 9$

.2 Cartesian Tensors

- All coordinate transformations are Euclidean (rigid), i.e. they can be described as a translation and a rotation.
- All indeces are written as sub-indices and each index corresponds to a coordinate direction.

his implies that

- The metric is the same in every point.
- The coordinate system is the same in every point.

2.2.1 Definitions

(i) *The index rule*: When an index appears once in a formula, the formula is valid for every value of that index.
(ii) *The summation rule*: When an index appears twice in a term, it is assumed that a summation is taken place over all possible values of that index.
(iii) *The maximim rule*: An index can not be used more than twice in the same term.

Example 3. $a_i + 2b_i = 0$ means $a_1 + 2b_1 = 0, a_2 + 2b_2 = 0$ and $a_3 + 2b_3 = 0$. ■

Example 4. $\sigma_{ij} = \sigma_{ji}$ means $\sigma_{12} = \sigma_{21}, \sigma_{13} = \sigma_{31}$ and $\sigma_{23} = \sigma_{32}$. ■

Example 5. $\epsilon_{ii} = 0$ means $\epsilon_{11} + \epsilon_{22} + \epsilon_{33} = 0$.
$a_k b_k = 0$ means $a_1 b_1 + a_2 b_2 + a_3 b_3 = 0$. ■

Example 6. $A_{rs} b_s = c_r$ means $A_{11} b_1 + A_{12} b_2 + A_{13} b_3 = c_1$, etc. ■

Consequences

) An index that appears once in every term can be changed to another ne (that is not already present in the expression).
) An index that appears twice in a term can be changed to another one hat is not already present in the term).

he following exprssions are not valid:
$A_i A_i C_i = 0$
$A_{jj} B_{jj} = 0$
$A_{ij} + C_{jk} = 0$

he following expressions are equivalent:
$f_i \alpha_{ij} = 0$ and $f_i \alpha_{ik} = 0$
$A_i B_i = 0$ and $A_j B_j = 0$
$T_{ii} + U_{jj} = S$ and $T_{ii} + U_{ii} = S$

Contraction

Definition 1. The number of free indices in a tensor is called the **degree** or **order** of the tensor, e.g. A_{ijk} denotes a third order tensor. ■

Definition 2. When a free index is changed to another free index, this index appears twice and an implicit summation is taking place. This process is called **contraction**, e.g. the fourth order tensor A_{ijkl} may be contracted to the second order tensor A_{ijjl}. ■

Example 7. $A_{ij} = B_{ik} C_{kj}$ can be contracted, giving $A_{ii} = B_{ik} C_{ki}$. ■

Example 8. $\frac{\partial A_i}{\partial x_k} = 0$ can be contracted, giving $\frac{\partial A_i}{\partial x_i} = 0$. ■

The Kronecker Delta

Definition 3. The special symbol δ_{ij} defined according to

$$\delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

s called **the Kronecker delta**. ■

Observe that $\delta_{ii} =$ the dimension of the space, i.e. $\delta_{ii} = 3$ in 3D-space.

Example 9. $\delta_{ij}x_j = x_i$

$\delta_{ij}A_{ik} = A_{jk}$

$\delta_{ij}\delta_{ik} = \delta_{jk}$

$\delta_{ij}\delta_{ij} = 3$

$\delta_{ij}\delta_{jk}\delta_{kl} = \delta_{il}$ ■

The Levi-Civita Epsilon

Definition 4. The special symbol ϵ_{ijk} defined according to (for 3D-spaces)

$$\epsilon_{ijk} = \begin{cases} 1 & \text{when i,j,k is a even permutation of 1,2,3} \\ -1 & \text{when i,j,k is an odd permutation of 1,2,3} \\ 0 & \text{when at least two indices are equal} \end{cases}$$

is called **the Levi-Cevita epsilon** or **the permutation symbol**. ■

A similar definition applies for other spaces.

Observe that $\epsilon_{ii} = 0$.

Example 10. $\epsilon_{ij} = -\epsilon_{ji}, \quad \epsilon_{ij}\epsilon_{ik} = \delta_{jk}, \quad \epsilon_{ij}\epsilon_{ij} = 2$

$\epsilon_{ijk}\epsilon_{pqk} = \delta_{ip}\delta_{jq} - \delta_{iq}\delta_{jp}, \quad \epsilon_{ijk}\epsilon_{pjk} = 2\delta_{ip}, \quad \epsilon_{ijk}\epsilon_{ijk} = 6$ ■

Note that $\mathbf{u} \times \mathbf{v} = \epsilon_{ijk}u_j v_k$.

.2.2 Coordinate Transformations

Observe that a translation of the coordinate system does not change the components of a vector. Thus we will only consider rotations.

Consider two different (positively oriented) cartesian coordinate systems $Ox_1x_2x_3$ and $Ox'_1x'_2x'_3$.

Define the **directional cosines** as

$$a_{ij} = \cos(Ox'_i, Ox_j) \ .$$

The coordinate transformation can now be written as

$$\begin{aligned} x'_1 &= a_{11}x_1 + a_{12}x_2 + a_{13}x_3 \\ x'_2 &= a_{21}x_1 + a_{22}x_2 + a_{23}x_3 \\ x'_3 &= a_{31}x_1 + a_{32}x_2 + a_{33}x_3 \end{aligned}$$

r

$$x'_i = a_{ij}x_j \ .$$

The Inverse Transformation

Similarly the inverse transformation is given by

$$x_i = a_{ji}x'_j \ ,$$

where $a_{ji} = \cos(Ox'_j, Ox_i)$.

Combining these equations gives

$$x_k = a_{ik}a_{ij}x_j = \delta_{kj}x_j \quad \Rightarrow \quad a_{ik}a_{ij} = \delta_{kj}$$

and similarly

$$a_{ik}a_{jk} = \delta_{ij} \ ,$$

which tells us that the coordinate axes are orthonormal.

Taking the determinant gives

$$\det(a_{ik}a_{jk}) = \det(a_{ik})\det(a_{jk}) = 1 \quad \Rightarrow \quad \det(a_{ik}) = \pm 1 \ ,$$

where the minus sign is impossible (why?).

.2.3 Vectors

ince translations does not change the components of the vector we may ssume that it starts at O.

Consider a vector $\mathbf{v}$ with components v_i in the coordinate system Ox_i and omponents v_i' in the coordinate system Ox_i'. It follows in the same way s for points:

$$v_i' = a_{ij}v_j \quad v_i = a_{ji}v_i'$$

Formal Definition

Definition 5. A triplet (v_1, v_2, v_3) are the components of a **vector** iff they transform as above. ■

The sum of two vectors is defined as: $a_i = u_i + v_i$
The difference of two vectors is defined as: $b_i = u_i - v_i$
The product of a vector and a scalar: $c_i = mu_i$
The scalar product: $d = u_i v_i$
The vector product: $e_i = \epsilon_{ijk} u_j v_k$
The length of a vector: $|v_i| = \sqrt{v_i v_i}$
The angle between two vectors: $\cos\theta = \frac{u_i v_i}{\sqrt{u_j u_j v_k v_k}}$
Orthogonal vectors: $u_i v_i = 0$

Exercise: Show that a_i, b_i, etc. are vectors according to the definition above, e.g.

$$d' = u_i' v_i' = a_{ij} u_j a_{ik} v_k = \delta_{ik} u_j v_k = u_j v_j = d$$

.2.4 Tensors

Example 11. Consider a state of stress in a continuous media:

ssume that the tensions are given by σ_{ij} in the coordinate system $Ox_1x_2x_3$ and by σ'_{ij} in the coordinate system $Ox'_1x'_2x'_3$.

quilibrium gives:

$$\sigma_{ij} = \sigma_{ji} \quad \text{and} \quad \sigma'_{ij} = \sigma'_{ji}$$

Surface	Area			
ABC	S	σ'_{11}	σ'_{12}	σ'_{13}
OCA	$a_{11}S$	$-\sigma_{11}$	$-\sigma_{12}$	$-\sigma_{13}$
OAB	$a_{12}S$	$-\sigma_{21}$	$-\sigma_{22}$	$-\sigma_{23}$
OBC	$a_{13}S$	$-\sigma_{31}$	$-\sigma_{32}$	$-\sigma_{33}$

Consider equilibrium for the tetrahedron OABC in the Ox'_1-direction:

$$\sigma'_{11}S - \sigma_{11}a_{11}Sa_{11} - \sigma_{12}a_{11}Sa_{12} - \sigma_{13}a_{11}Sa_{13} - \sigma_{21}a_{12}Sa_{11} - \sigma_{22}a_{12}Sa_{12} - \sigma_{23}a_{12}Sa_{13} - \sigma_{31}a_{13}Sa_{11} - \sigma_{32}a_{13}Sa_{12} - \sigma_{33}a_{13}Sa_{13} = 0$$

or equivalently

$$\sigma'_{11} = a_{1k}a_{1l}\sigma_{kl} \ .$$

In the same way we obtain the equations

$$\sigma'_{12} = a_{1k}a_{2l}\sigma_{kl} \quad \sigma'_{13} = a_{1k}a_{3l}\sigma_{kl} \ ,$$

which can be written as

$$\sigma'_{1j} = a_{1k}a_{jl}\sigma_{kl} \ .$$

Consider similar tetrahedrons oriented in the Ox'_2 and Ox'_3-directions:

$$\sigma'_{2j} = a_{2k}a_{jl}\sigma_{kl} \quad \sigma'_{3j} = a_{3k}a_{jl}\sigma_{kl} \ .$$

These three equations can be written

$$\sigma'_{ij} = a_{ik}a_{jl}\sigma_{kl} \ .$$

■

Formal Definition

In 3D-space we make the following definitions:

Definition 6. A collection of nine numbers t_{ij} is called **a second order tensor** if it transforms according to

$$t'_{ij} = a_{ik}a_{jl}t_{kl} \ .$$

Definition 7. A collection of 3^m numbers $t_{i_1\ldots i_m}$ is called **an m:th order tensor** if it transforms according to

$$t'_{i_1\ldots i_m} = a_{i_1j_1}\ldots a_{i_mj_m}t_{j_1\ldots j_m} \ .$$

Similar definitions apply for other dimensions than 3.

Observe that:
Scalar = a zeroth order tensor
Vector = a first order tensor

Example 12 (cont'd). Consider the previous example and a triangular surface with normal vector $\mathbf{n}$ and directional cosines $a_{(n)i}$.

Then equilibrium gives

$$\sigma_j^{(n)} = a_{(n)i}\sigma_{ij} \ .$$

Thus σ_{ij} can be seen as a transformation from $a_{(n)i}$ to $\sigma_j^{(n)}$, i.e. it can be used to transform directional vectors to normal tensions.

This fact can be used to define tensors, e.g. an $m+1$:th order tensor is a transformation from $a_{(n)i}$ to an m:th order tensor according to

$$t^{(n)}_{i_1\ldots i_m} = a_{(n)i_{m+1}}s_{i_1\ldots i_{m+1}} \ .$$

■

Other Examples

Example 13. The tensions in a continuous medium transforms according to

$$\epsilon'_{ij} = a_{ik}a_{jl}\epsilon_{kl} \ .$$

They thus form a second order tensor. ■

Example 14. The moments of inertia of a rigid body, defined by

$$I_{ij} = \int x_i x_j dm$$

transforms according to

$$I'_{ij} = a_{ik}a_{jl}I_{kl} \ .$$

They thus form a second order tensor. ■

Invariants

Example 15. Consider again a state of stress in a homogeneous media.

A surface with only normal tension obeys

$$a_{(n)i}\sigma_{ij} = \sigma a_{(n)j} = \sigma\delta_{ij}a_{(n)i}$$

for some σ, which gives

$$(\sigma_{ij} - \sigma\delta_{ij})a_{(n)i} = 0 \quad \Rightarrow \quad \det(\sigma_{ij} - \sigma\delta_{ij}) = 0 \ .$$

This is the well-known characteristic equation for σ_{ij}, considered as a matrix. The three real roots are called the **normal stresses** and the three corresponding surfaces are perpendicular, called **normal directions**. ■

Example 16. For the tensor obtained from the moments of inertia we obtain in the same way the **principal directions** and the **principal moments**. ■

Definition of Invariants

For every second order tensor t_{ij} we can calculate the eigenvalues and eigendirections from

$$\det(t_{ij} - t\delta_{ij}) = 0 \ ,$$

which can be written as

$$t^3 - I_1 t^2 + I_2 t - I_3 \ ,$$

where

$$I_1 = t_{11} + t_{12} + t_{13} \quad \text{and} \quad I_3 = \det(t_{ij}) \ .$$

It follows that I_1, I_2 and I_3 are invariant to coordinate transformations.

Definition 8. I_1, I_2 and I_3, defined above, are called the **main invariants** to the tensor t_{ij}. ■

Special Tensors

It can be shown that δ_{ij} is a second order tensor with $\delta'_{ij} = \delta_{ij}$.
It is usually called **the unit tensor**.

It can also be shown that ϵ_{ijk} is a third order tensor with $\epsilon'_{ijk} = \epsilon_{ijk}$.
It is usually called **the permutation tensor**.

Hint: Use

$$\epsilon_{ijk}u_i v_j w_k = \begin{bmatrix} u_1 & u_2 & u_3 \\ v_1 & v_2 & v_3 \\ w_1 & w_2 & w_3 \end{bmatrix}$$

and insert a_{ij} for u, v and w.

sotropic Tensors

)efinition 9. A tensor that has the same components in every coordinate ystem is called an **isotropic tensor**. ■

:xample 17. Every scalar is isotropic.
A first order tensor (vector) can not be isotropic.
$_{ij}$ is an isotropic second order tensor.
$_{ijk}$ is an isotropic third order tensor. ■

t can be shown that all isotropic tensors of order two can be written $a\delta_{ij}$.

All isotropic tensors of order three can be written $b\epsilon_{ijk}$.

t can also be shown that every isotropic tensor of fourth order can be vritten as

$$t_{ijkl} = c\delta_{ij}\delta_{kl} + d\delta_{ik}\delta_{jl} + e\delta_{il}\delta_{jk} \ .$$

2.2.5 Tensor Algebra

From two tensors of the same order we can form the sum and the difference:

$$s_{ij} = r_{ij} + t_{ij} \quad d_{ij} = r_{ij} - t_{ij}$$

They are both tensors and the operations are commutative and associative.

From two tensors we can form the **outer product**:

$$p_{ijlmn} = r_{ij}t_{lmn}$$

which is a tensor with degree equal to the sum of the degrees of the factors and the operation is associative and distributive, but in general not commutative.

One tensor can be **contracted** by setting two indices equal:

$$t_{ijjl}$$

which is a second order tensors obtained from a fourth order tensor.

The dyadic multiplication of two vectors gives: $t_{ij} = u_i v_j$ Contracting ives the inner product: $t_{ii} = u_i v_i$

The following result, called **the quiotent rule**, is very useful:
An entity that gives a tensor when taking the inner (or outer) product with an arbitrary tensor is itself a tensor.

A tensor is called **symmetric** with respect to two indices if it remains the ame when interchanging the indices.

A tensor is called **totally symmetric** if it remains the same when nterchanging any pair of indices.

Exercise: Show that if a tensor is symmetric in one coordinate system, hen it is symmetric in every coordinate system.

A tensor is called **skew-symmetric** with respect to two indices if it changes sign when interchanging the indices.

A tensor is called **totally skew-symmetric** if it changes sign when interchanging any pair of indices.

Observe that contracting over two skew-symmetric indices gives the zero tensor.

Observe that the inner product of a symmetric and an skew-symmetric tensor (over the appropriate indices) gives the zero tensor.

Every skew-symmetric degree two tensor can be written as $t_{ij} = \epsilon_{ijk}v_k$ for some vector v_k.

Every skew-symmetric degree three tensor can be written as $t_{ij} = t\epsilon_{ijk}$ for some vector t.

.2.6 Tensor Analysis

he following equations are basic:

$$\frac{\partial x_i}{\partial x_j} = \delta_{ij} \quad \frac{\partial x'_i}{\partial x_j} = a_{ij} \quad \frac{\partial x_i}{\partial x'_j} = a_{ji}$$

efinition 10. The **gradient** to the scalar field $s(x_1, x_2, x_3)$ is defined ccording to

$$\operatorname{grad} s = \frac{\partial s}{\partial x_i} \ .$$

ne can show that

$$\frac{\partial s}{\partial x'_i} = a_{ij} \frac{\partial s}{\partial x_j} \ ,$$

hich implies that the gradient is a first order tensor, written as

$$\frac{\partial s}{\partial x_i} = s_{,i} \ .$$

Vector Fields

A vector-field $v_i(x_1, x_2, x_3)$ has the gradient $\partial v_i / \partial x_j$, which can be shown to be a second order tensor, written

$$v_{i,j} = \frac{\partial v_i}{\partial x_j} \ .$$

The gradient of an n:th order tensor is a $(n+1)$:th order tensor:

$$t_{i_1 \cdots i_n, i_{n+1}} = \frac{\partial t_{i_1 \cdots i_n}}{\partial x_{i_{n+1}}} \ .$$

Definition 11. The **divergence** of the vector field $v_i(x_1, x_2, x_3)$ is

$$\operatorname{div} \mathbf{v} = \delta_{ij} v_{i,j} = v_{i,i}$$

and the **rotation** is

$$\operatorname{rot} \mathbf{v} = \epsilon_{ijk} v_{k,j} \ .$$

Observe that div is a scalar and $\operatorname{rot} \mathbf{v}$ is a vector.

ome Theorems

heorem 1.

$$\operatorname{rot} \operatorname{grad} s = 0 \quad \textit{or} \quad \epsilon_{ijk} s_{,kj} = 0$$
$$\operatorname{div} \operatorname{rot} \mathbf{v} = 0 \quad \textit{or} \quad \epsilon_{ijk} v_{k,ji} = 0$$

roof. Follows directly from the fact that ϵ_{ijk} is skew-symmetric and $s_{,kj}$ nd $v_{k,ji}$ are symmetric. ■

heorem 2 (Gauss).

$$\iiint_V t_{i_1 \cdots i_m, i_{m+1}} dV = \iint_S t_{i_1 \cdots i_m} n_{m+1} dS$$

xample 18.

$$\iiint_V s_{,i} dV = \iint_S s n_i dS$$
$$\iiint_V v_{i,i} dV = \iint_S v_i n_i dS$$

2.3 Affine Tensors

- All coordinate transformations are Affine, i.e. they can be described as a translation and a linear transformation.
- Some indeces are written as sub-indices, some indices as super-indices and each index corresponds to a coordinate direction.

This implies that

- The metric is the same in every point, but can changed due to a change of coordinates.
- The coordinate system is the same in every point.

New: There are two different types of indices: **covariant** (below) and **contravariant** (upper) that transforms differently.

New rule: A repeated index must appear once as a covariant and once as a contravariant index.

Transformation properties

Given a basis e. Introduce a new basis $\hat{e}$ according to $\hat{e} = S^T e$, or in tensor notation:

$$\hat{e}_j = S^i_j e_i \ ,$$

where S^i_j means S^{row}_{col}.

The covariant indices transform according to

$$\hat{l}_j = S^i_j l_i$$

and the contravariant according to

$$\hat{x}^j = (S^{-1})^j_i x^i \quad \text{or} \quad x^i = S^i_j \hat{x}^j \ .$$

The covariant indices co-varies with the base and the contravariant indices contra-varies with the base.

A tensor is said to be of type (k, l) if it has k contravariant indices and l covariant indices.

Contravariant Vectors

Example 19. A type $(1, 0)$ tensor, v^j, is called a **contravariant vector**. It transforms according to

$$v^j = S^j_k \hat{v}^k \ .$$

Example 20. Let v^j denote the components of a vector, $\mathbf{v}$, in $\mathbb{R}^n$ expressed using a basis e_j, i.e.

$$\mathbf{v} = v^j e_j \ .$$

A new basis $\hat{e}$ is chosen according to $\hat{e}_k = S^j_k e_j$. Then $\mathbf{v}$ can be expressed using the new basis $\hat{e}_j$ according to

$$\mathbf{v} = \hat{v}^k \hat{e}_k = \hat{v}^k S^j_k e_j = S^j_k \hat{v}^k e_j \ ,$$

which gives

$$v^j = S^j_k \hat{v}^k \ .$$

Thus the components of a vector transforms as a contravariant tensor. ■

Covariant Vectors

Example 21. A type $(0, 1)$ tensor, v_j, is called a **covariant vector**. It transforms according to

$$\hat{v}_k = S^j_k v_j \ .$$

■

Example 22. Let $\mathbf{l}$ denote a line in the plane given by

$$l_j x^j = 0 \ .$$

Express the (contravariant) vector x^j in a new basis according to $x^j = S^j_k \hat{x}^k$, which gives

$$0 = l_j x^j = l_j S^j_k \hat{x}^k = \hat{l}_k \hat{x}^k \ ,$$

with

$$\hat{l}_k = S^j_k l_j \ .$$

Thus the components of a line transforms covariantly. ■

2.4 General Tensors

- The coordinate transformations are general, i.e. they might be non-linear and different in different points.
- Both contra-variant anc co-variant indices are used.

This implies that

- The metric is in general different at each point.
- The coordinate system is in general different at each point.

New: Calculating derivatives are much more complicated since the coordinate system is changing from point to point.

Coordinate Transformations

Write the coordinates in a general n-dimensional space as $(x^1, \ldots, x^n)$, i.e. super-scripts are used.

Assume a general transformation:

$$\bar{x}^i = \psi^i(x^1, \ldots, x^n) \ ,$$

where ϕ^i are differentiable and independent.

The inverse transformation can be written

$$x^i = \phi^i(\bar{x}^1, \ldots, \bar{x}^n) \ .$$

We have

$$d\bar{x}^i = \frac{\partial \psi^i}{\partial x^r} dx^r = \frac{\partial \bar{x}^i}{\partial x^r} dx^r$$

Contravariant vectors

Definition 12. A vector with components A^i is **contravariant** if it transforms according to

$$\bar{A}^i = \frac{\partial \bar{x}^i}{\partial x^j} A^j \ .$$

Note:

$$\frac{\partial x^k}{\partial \bar{x}^i} \bar{A}^i = \frac{\partial x^k}{\partial \bar{x}^i} \frac{\partial \bar{x}^i}{\partial x^j} A^j = \delta^k_j A^j = A^k \ ,$$

according to the chain rule, giving

$$A^k = \frac{\partial x^k}{\partial \bar{x}^i} \bar{A}^i \ .$$

Example 23. The differential dx^i is a contravariant vector.
The tangent vector dx^i/du to the curve $x^i = x^i(u)$ is also a contravariant vector. ■

Covariant vectors

Definition 13. A vector with components A^i are **contravariant** if it transforms according to

$$\bar{A}_i = \frac{\partial x^j}{\partial \bar{x}^i} A_j \ .$$

Note:

$$\frac{\partial \bar{x}^i}{\partial x^k} \bar{A}_i = \frac{\partial \bar{x}^i}{\partial x^k} \frac{\partial x^j}{\partial \bar{x}^i} A_j = \frac{\partial x^j}{\partial x^k} A_j = A_k \ .$$

Example 24. The gradient

$$\partial f / \partial x^j$$

is a covariant vector, where $f(x^1, \ldots, x^n)$ is a scalar function. ■

Invariants

Definition 14. Any function

$$I(x_1, \ldots, x_n)$$

is called an **invariant** if $I = \bar{I}$, i.e. if it is unchanged under coordinate transformations. ■

Example 25. The scalar product is invariant since

$$\bar{A}^i \bar{B}_i = \frac{\partial \bar{x}^i}{\partial x^j} A^j \frac{\partial x^k}{\partial \bar{x}^i} B_k = \delta^k_j A^j B_k = A^k B_k \ .$$

δ^i_i is also an invariant. ■

Higher Order Tensors

Definition 15. A set of n^{s+p} functions $A^{t_1\cdots t_s}_{q_1\cdots q_p}$ of the n coordinates x^i are the components of a **mixed tensor** of order $s+p$, contravariant of order s and covariant of order p if

$$\bar{A}^{u_1\cdots u_s}_{r_1\cdots r_p} = \frac{\partial \bar{x}^{u_1}}{\partial x^{i_1}} \cdots \frac{\partial \bar{x}^{u_s}}{\partial x^{i_s}} \frac{\partial x^{q_1}}{\partial \bar{x}^{r_1}} \cdots \frac{\partial x^{q_p}}{\partial \bar{x}^{r_p}} A^{t_1\cdots t_s}_{q_1\cdots q_p} \ .$$

Example 26. $A^{ij} = B^i C^j$, where B^i and C^j are contravariant vectors, is a second order contravariant tensor.
$A^i_j = B^i C_j$, where B^i is a contravatiant vector and C_j is contravariant vector, is a second order mixed tensor. ■

Symmetry and skew-symmetry are defined as before, but only indices of the same type may be interchanged.

Tensor Algebra

Addition, subtraction and multiplication are defined as before, when the obvious constraints on the type of indices have been taken into account.

Example 27. A tensor A_{ij} can be written as

$$A_{ij} = \frac{1}{2}(A_{ij} + A_{ji}) + \frac{1}{2}(A_{ij} - A_{ji}) = S_{ij} + D_{ij} \ ,$$

where S_{ij} is symmetric and D_{ij} skew-symmetric. ■

The outer product is also defined as before, e.g.

$$A^{ijl}_{kmnt} = B^{ij}_k C^l_{mnt} \ .$$

A contraction has to be made over one covariant and one contravariant index, e.g.

$$A^{ij}_{lmj} \ .$$

The Fundamental Tensor

When the distance ds between the points x^i and $x^i + dx^i$ is given by

$$ds^2 = g_{ij} dx^i dx^j \ ,$$

where g_{ij} are functions of x^i, we have a **Riemannian space**.

g_{ij} is a symmetric covariant tensor of the second order, with $\det g_{ij} \neq 0$, called the **fundamental tensor**.
The quadratic form $g_{ij}dx^i dx^j$ is called the **metric**.
ds is called the **line element** and is an invariant.

In three-dimensional Euclidean space the line element is given by

$$ds^2 = (dx^1)^2 + (dx^2)^2 + (dx^3)^2$$

and $g_{11} = g_{22} = g_{33} = 1$.

When g_{ij} is not positive definite we use $ds^2 = e g_{ij} dx^i dx^j$, where the **indicator** $e = \pm 1$ makes sure that ds is real.

Applications

Definition 16. The length of the curve $x^i(t), t_1 \leq t \leq t_2$ is defined by

$$s = \int_{t_1}^{t_2} \sqrt{e g_{ij} \frac{dx^i}{dt} \frac{dx^j}{dt}} dt \ .$$

When $g_{ij} \frac{dx^i}{dt} \frac{dx^j}{dt} = 0$ along the curve, it is called **minimal** or **null**. ■

Definition 17. The **magnitude** of a contravariant vector A^i is defined by

$$(A)^2 = e_{(A)} g_{ij} A^i A^j \ .$$

The **magnitude** of a covariant vector B_i is defined by

$$(B)^2 = e_{(A)} g^{ij} B_i B_j \ ,$$

where $g_{ij} g^{ik} = \delta^k_j$. ■

Definition 18. The **associate tensor** to A^j is defined by

$$A_i = g_{ij}A^j$$

and similarly the associate tensor to B_j by

$$B^i = g^{ij}B_j \ .$$

Note that $g^{ij}A_j = g^{ij}g_{ik}A^k = \delta^i_k A^k = A^i$, i.e. the associate to the associate gives back the same vector.

Definition 19. The **angle** between the unit vectors A^i and B^j is defined by

$$\cos\theta = g_{ij}A^iB^j \ .$$

Definition 20. The **principal directions** L_i to the tensor A_{ij} is defined by

$$\det(A_{ij} - \lambda g_{ij})L^i = 0 \ .$$

The **principal values** λ are invariants and L^i are contravariant vectors. ■

Christoffel Symbols

Definition 21. The symbol (not tensor)

$$\Gamma_{ij,m} = \frac{1}{2}(\frac{\partial g_{ik}}{\partial x^j} + \frac{\partial g_{jk}}{\partial x^i} - \frac{\partial g_{ij}}{\partial x^k})$$

is called the **Christoffel symbol of the first kind** and

$$\Gamma^l_{ij} = g^{lk}\Gamma_{ij,k}$$

is called the **Christoffel symbol of the second kind**. ■

We have

$$\Gamma_{ij,m} = g_{lm}\Gamma^l_{ij} \quad \text{and} \quad \frac{\partial g_{ik}}{\partial x^j} = \Gamma_{ij,k} + \Gamma_{kj,i}$$

Some calculations give

$$\frac{\partial g^{mk}}{\partial x^l} = g^{lm}g\frac{\partial g_{lm}}{\partial x^j} \quad \text{and} \quad \Gamma^i_{ij} = \frac{1}{2}\frac{1}{g}\frac{\partial g}{\partial x^j} \ ,$$

where $g = \det(g_{ij})$.

Transformatin Laws

It can be shown that the Cristoffel symbols transforms according to

$$\bar{\Gamma}_{lm,n} = \Gamma_{ij,k}\frac{\partial x^i}{\partial \bar{x}^l}\frac{\partial x^j}{\partial \bar{x}^m}\frac{\partial x^k}{\partial \bar{x}^n} + g_{ij}\frac{\partial x^i}{\partial \bar{x}^n}\frac{\partial^2 x^j}{\partial \bar{x}^l \partial \bar{x}^m}$$

$$\bar{\Gamma}^p_{lm} = \Gamma^s_{ij}\frac{\partial \bar{x}^p}{\partial x^s}\frac{\partial x^i}{\partial \bar{x}^l}\frac{\partial x^j}{\partial \bar{x}^m} + \frac{\partial \bar{x}^p}{\partial \bar{x}^j}\frac{\partial^2 x^j}{\partial \bar{x}^l \partial \bar{x}^m} \ ,$$

which shows that $\Gamma_{lm,n}$ and Γ^p_{lm} are not tensors.

We also have

$$\frac{\partial^2 x^r}{\partial \bar{x}^l \partial \bar{x}^m} = \bar{\Gamma}^p_{lm}\frac{\partial x^r}{\partial \bar{x}^p} - \Gamma^r_{ij}\frac{\partial x^i}{\partial \bar{x}^l}\frac{\partial x^j}{\partial \bar{x}^m} \ ,$$

which will be useful later on.

Covariant Differentiation

Differentiate the transformation law

$$A^k = \bar{A}^i\frac{\partial x^k}{\partial \bar{x}^i}$$

with respect to x^j, which gives

$$\frac{\partial A^k}{\partial x^j} = \frac{\partial \bar{A}^i}{\partial \bar{x}^n}\frac{\partial \bar{x}^n}{\partial x^j}\frac{\partial x^k}{\partial \bar{x}^i} + \bar{A}^i\frac{\partial^2 x^k}{\partial \bar{x}^i \partial \bar{x}^n}\frac{\partial \bar{x}^n}{\partial x^j} \ ,$$

which shows that $\frac{\partial A^k}{\partial x^j}$ is not a tensor.

However, using the last equation on the previous slide,

$$\frac{\partial A^k}{\partial x^j} = \frac{\partial \bar{A}^i}{\partial \bar{x}^n}\frac{\partial \bar{x}^n}{\partial x^j}\frac{\partial x^k}{\partial \bar{x}^i} + \bar{A}^i\frac{\partial \bar{x}^n}{\partial x^j}\left[\bar{\Gamma}^p_{in}\frac{\partial x^k}{\partial \bar{x}^p} - \Gamma^k_{rs}\frac{\partial x^r}{\partial \bar{x}^i}\frac{\partial x^s}{\partial \bar{x}^n}\right]\frac{\partial \bar{x}^n}{\partial x^j} \ ,$$

which gives

Covariant Differentiation (cont'd)

$$\frac{\partial A^k}{\partial x^j} + \Gamma^k_{rj} A^r = \left[\frac{\partial A^i}{\partial x^n} + \bar{\Gamma}^i_{rn}\bar{A}^r\right]\frac{\partial x^n}{\partial \bar{x}^j}\frac{\partial x^k}{\partial \bar{x}^i} \ .$$

Definition 22. The **covariant derivative** of A^k with respect to x^j is defined according to

$$A^k_{,j} = \frac{\partial A^k}{\partial x^j} + \Gamma^k_{rj} A^r \ .$$

The previoius equation shows that $A^k_{,j}$ is a tensor.

In the same way we may define the **covariant derivative** of A_j with respect to x^n as

$$A_{j,n} = \frac{\partial A_j}{\partial x^n} + \Gamma^r_{jn} A_r \ ,$$

which is also a tensor.

Divergence

Definition 23. The contraction $A^j_{,j}$ is called the **divergence** of A^i and denoted $\operatorname{div} A^i$. ■

We have

$$A^j_{,j} = \frac{\partial A^j}{\partial x^j} + \Gamma^j_{rj} A^r = \frac{\partial A^j}{\partial x^j} + A^r \frac{\partial}{\partial x^r}(\log\sqrt{g}) = \frac{1}{\sqrt{g}}\frac{\partial}{\partial x^r}(A^j \log\sqrt{g}) \ ,$$

which is an invariant.

Definition 24. The **divergence** of A_i is defined according to

$$\operatorname{div} A_i = g^{jk} A_{j,k}$$

and denoted $\operatorname{div} A_i$. ■

Covariant Differentiation of Tensors

Similarly

Definition 25. The **covariant derivative** of the tensor A^i_j is defined according to

$$A^m_{l,t} = \frac{\partial A^m_l}{\partial x^i} + \Gamma^m_{rt} A^r_l - \Gamma^r_{lt} A^m_r \ .$$

It can be shown that $A^m_{l,t}$ is a tensor.

Covariant derivatives obeys the usual laws for derivation, e.g. the product rule.

Intrinsic Derivation

Definition 26. The **intrinsic derivative** of a tensor A^i_j defined on a curve $x^i = x^i(t)$ is defined according to

$$\frac{\delta A^i_j}{\delta t} = A^i_{j,k}\frac{dx^k}{dt} \ ,$$

i.e. of the same order and type as the original tensor. ■

For an invariant we have

$$\frac{\delta I}{\delta t} = I_{,k}\frac{dx^k}{dt} = \frac{\partial I}{\partial x^k}\frac{dx^k}{dt} = \frac{dI}{dt} \ ,$$

i.e. the intrinsic derivative coincides with the total derivative.

Intrinsic derivatives of higher order are defined in the same way.

Intrinsic derivative is not commutative.

Other Topics

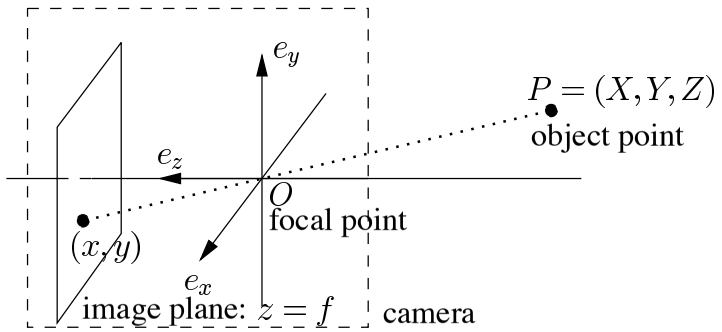
- Geodesics-Parallelism
- Curvature Tensor
- Applications to differential geometry
- Applications to elasticity
- Applicatinos to the theory of relativity

Can be found in
. Spain: *Tensor Calculus*.

2.5 Conclusions

- Motivation
- Cartesian tensors
- Affine tensors
- General tensors

3. Modeling cameras



Using uniform triangles gives

$$\frac{x}{f} = \frac{X}{Z}, \quad \frac{y}{f} = \frac{Y}{Z} \ .$$

These equations can be written

$$Z \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \ ,$$

by using homogeneous coordinates $(x, y, 1)$ instead of (x, y).

Putting $\lambda = Z$ and using homogeneous coordinates also for the object points gives

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \ .$$

ntroducing different coordinate systems

n most cases we can not assume that the same coordinate system can be sed for the camera and for the scene, since the relative orientation is not nown.

hus, we have to use different coordinate systems for the camera and for he scene.

he relation between these coordinate systems is described by a eometric transformation.

here are (at least) three interesting transformation groups:

- Euclidean transformations
- Affine transformation
- Projective transformations

Similarity transformation = Euclidean + global change of scale.)

Euclidean transformations

The relation between the different coordinate systems is described by a translation and a rotation, which can be written as

$$\begin{bmatrix} X_2 \\ Y_2 \\ Z_2 \end{bmatrix} = \begin{bmatrix} R \end{bmatrix} \begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \end{bmatrix} + t \ ,$$

where R denotes a 3×3 **orthogonal** matrix and t a 3×1 translation vector.

Note that, using homogeneous coordinates, we can write

$$\begin{bmatrix} X_2 \\ Y_2 \\ Z_2 \\ 1 \end{bmatrix} = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \\ 1 \end{bmatrix} \ .$$

ffine transformations

he relation between the different coordinate systems is described by a ranslation and a linear transformation, which can be written as

$$\begin{bmatrix} X_2 \\ Y_2 \\ Z_2 \end{bmatrix} = \begin{bmatrix} A \end{bmatrix} \begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \end{bmatrix} + b \ ,$$

here A denotes **any non-singular** 3×3 matrix and b a 3×1 translation ector.

lote that, using homogeneous coordinates, we can write

$$\begin{bmatrix} X_2 \\ Y_2 \\ Z_2 \\ 1 \end{bmatrix} = \begin{bmatrix} A & b \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \\ 1 \end{bmatrix} \ .$$

Projective transformations: Collineations

The relation between the different coordinate systems is described by

$$\lambda \begin{bmatrix} X_2 \\ Y_2 \\ Z_2 \\ 1 \end{bmatrix} = \begin{bmatrix} A & b \\ c & d \end{bmatrix} \begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \\ 1 \end{bmatrix} = \begin{bmatrix} H \end{bmatrix} \begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \\ 1 \end{bmatrix} \ ,$$

where H denotes **any non-singular** 4×4 matrix.

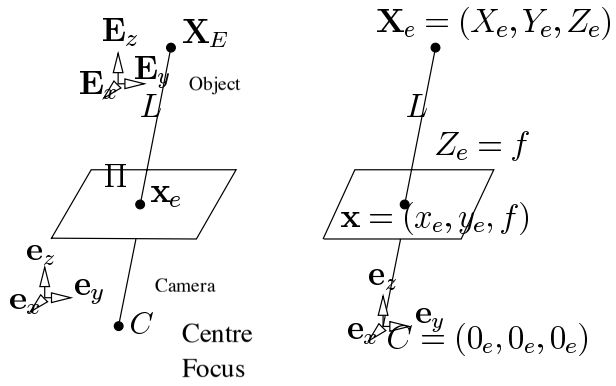
Why is λ needed, but not for Euclidean and Affine transformations?

Sometimes written

$$\mathbf{X}_2 \sim H\mathbf{X}_1 \ ,$$

where $\mathbf{X}_1$ and $\mathbf{X}_2$ denote homogeneous coordinate vectors and $\sim$ denotes equality up to scale.

Notations



: image (camera) coordinate system

: object coordinate system

Change of coordinates

Introduce a Euclidean change of coordinates between the camera coordinate system and the object coordinate system.

$$\mathbf{X}_e = R(\mathbf{X}_E - t) \ ,$$

where R denote a orthogonal matrix.

This gives the camera equation

$$\lambda \begin{bmatrix} x_e \\ y_e \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} [\, R \mid -Rt \,] \begin{bmatrix} X_E \\ Y_E \\ Z_E \\ 1 \end{bmatrix} = K [\, R \mid -Rt \,] \begin{bmatrix} X_E \\ Y_E \\ Z_E \\ 1 \end{bmatrix} \ .$$

Note that t gives the coordinates of the focal point in the E-system.

The camera matrix

ntroduce the notation

$$\mathbf{X} = \begin{bmatrix} X_E \\ Y_E \\ Z_E \\ 1 \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x_e \\ y_e \\ 1 \end{bmatrix}$$

or image coordinates and object coordinates.

Definition 27. A matrix, P, that relates object coordinates to image coordinates according to $\lambda \mathbf{x} = P\mathbf{X}$ is called a **camera matrix**. ■

Observe that $P = K [\, R \mid -Rt \,]$ and that

$$P \begin{bmatrix} t \\ 1 \end{bmatrix} = 0 \ ,$$

e. the focal point is obtained from the nullspace to the camera matrix.

A more detailed model

In the model above, the camera is modeled as

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = K [\, R \mid -Rt \,] \begin{bmatrix} X_E \\ Y_E \\ Z_E \\ 1 \end{bmatrix}, \quad K = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \ .$$

A common modification is to introduce more parameters in K:

$$K = \begin{bmatrix} \gamma f & sf & x_0 \\ 0 & f & y_0 \\ 0 & 0 & 1 \end{bmatrix} \ .$$

Camera parameters

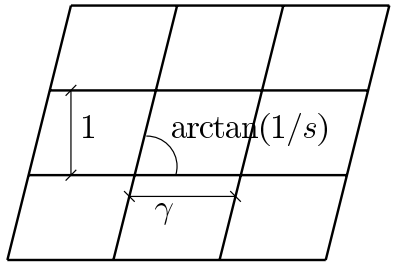
Intrinsic parameters:

f : **focal length** - also called camera constant

a : **aspect ratio** - modeling non-quadratic light-sensitive elements

s : **skew** - modeling non-rectangular light-sensitive elements

(x_0, y_0) : **principal point** - orthogonal projection of the focal point onto the image plane



Number of parameters

Extrinsic parameters:

t : translation (3 parameters)

R : rotation (3 parameters - Euler angles or unit quaternions)

Camera matrix:

In total 5 intrinsic + 6 extrinsic = 11 parameters.

The same as in a general 3×4 matrix, defined up to scale.

Conclusion: When the intrinsic and extrinsic parameters are unknown and possibly varying between the different images in an image sequence, the distinction between intrinsic and extrinsic parameters is not needed.

With i denoting image number we write:

$$\lambda_i \mathbf{x}_i = P_i \mathbf{X} \ .$$

This point of view makes it natural to work in a **projective space** instead of Euclidean.

Different camera models

- K known - **calibrated camera**:

$$\lambda \mathbf{x} = K[R|-Rt]\mathbf{X} \quad \Rightarrow \quad \tilde{\mathbf{x}} \sim [R|-Rt]\mathbf{X} \ ,$$

  where $\mathbf{x} = K\tilde{\mathbf{x}}$.

- K unknown - **uncalibrated camera**:

$$\lambda \mathbf{x} = P\mathbf{X} \quad \Rightarrow \quad \mathbf{x} \sim P\mathbf{X}$$

- Image sequence with K unknown and varying - **uncalibrated image sequence**:

$$\lambda \mathbf{x}_i = P_i\mathbf{X} \quad \Rightarrow \quad \mathbf{x}_i \sim P_i\mathbf{X}$$

Several images of the same object

Observation 1: *From one image, it is only possible to determine in which direction (from the focal point) the object point is located, not how far away, i.e. the object point is constrained to lie on the corresponding line.*

Observation 2: *The object point can be reconstructed from two images with known relative orientation by intersecting the corresponding lines with eachother.*

Problem: *Determine both the relative orientation and the location of the object points from only the location of the image points, given known point correspondences.*

Note: We will deal only with uncalibrated image sequences.

4. Projective geometry

Definition 28. Let V be a vector space of dimension $n+1$. The set of one-dimensional subspaces of V is called the **projective space of dimension** n, denoted by $\mathbb{P}^n$. ■

Assume that the vector space V is $\mathbb{R}^{n+1}$.

Introduce coordinates in $V = \mathbb{R}^{n+1}$: $(x_0, x_1, \ldots, x_n)$

Example 28. A point in $\mathbb{P}^1$ is a one-dimensional linear subspace of $\mathbb{R}^2$, i.e. a line through the origin.

$\mathbb{P}^1$ can be viewed as directions in $\mathbb{R}^2$. ■

Example 29. A point in $\mathbb{P}^2$ is a one-dimensional linear subspace of $\mathbb{R}^3$, i.e. a line through the origin.

$\mathbb{P}^2$ can be viewed as directions in $\mathbb{R}^3$ ■

Homogeneous coordinates

A point in $\mathbb{P}^n$ is represented by the n-tuple $\mathbf{x} = (x_0, x_1, \ldots, x_n) \in V$, where two points $\mathbf{x}$ and $\mathbf{y}$ are considered to be equal if there exists a $\lambda \in \mathbb{R}, \lambda \neq 0$ such that

$$\mathbf{x} = \lambda \mathbf{y} \quad \Leftrightarrow \quad \mathbf{x} \sim \mathbf{y} \ .$$

Definition 29. Any representation of $\mathbf{x} \in \mathbb{P}^n$ of the form

$$\mathbf{x} = (x_0, x_1, \ldots, x_n)$$

will be called **homogeneous coordinates** for $\mathbf{x}$. ■

Example 30. In $\mathbb{P}^1$ the following holds:

$$(1,2) \sim (2,4) \sim (\pi, 2\pi) \sim \ldots$$
$$(3,5) \sim (4.5, 7.5) \sim (300, 500) \sim \ldots$$

■

Geometric entities

All geometric entities (points, lines, planes, etc.) are inherited from the ambient vector space, V.

A **point**, $\mathbf{p}$, is represented by its homogeneous coordinates: $\mathbf{p} : (x_0, \ldots, x_n)$.

A **line**, $\mathbf{l}$ can be represented by two different points on the line: $\mathbf{l} : (\mathbf{p}_1, \mathbf{p}_2)$.

A **plane**, π can be represented by a vector $\mathbf{v} = (v_0, \ldots, v_n)$:

$$\pi : \{ \mathbf{x} = (x_0, \ldots, x_n) \in \mathbb{P}^n \,|\, v_0 x_0 + v_1 x_1 + \ldots + v_n x_n = 0 \}$$

Observe that $\mathbf{v}$ and $\lambda\mathbf{v}, \lambda \in \mathbb{R}$, represents the same plane.

A **conic** in $\mathbb{P}^2$ or a **quadric** in $\mathbb{P}^3$ is defined as

$$\{ \mathbf{x} \in \mathbb{P}^n \,|\, \mathbf{x}^T C \mathbf{x} = 0 \} \ ,$$

where C is a 3×3 or 4×4 matrix.

Intersection

The intersection of geometric entities is defined as usual.

Example 31. The intersection of two lines in $\mathbb{P}^2$, represented by $\mathbf{l}_1$ and $\mathbf{l}_2$ is obtained as the solution to

$$\mathbf{l}_1^T \mathbf{x} = 0, \quad \mathbf{l}_2^T \mathbf{x} = 0 \ ,$$

i.e. a system of two linear equations in three variables. This system has always a solution. ■

Note:

- Two different lines in $\mathbb{P}^2$ always intersect in a point
- Parallelism has no meaning in projective space

In the same way we obtain:

- Two different planes in $\mathbb{P}^3$ always intersect in a line.
- A plane and a line in $\mathbb{P}^3$ always intersect in a point.

The line/plane at infinity

Definition 30. The subspace

$$\mathbb{A}_i = \{(x_0, \ldots, x_n) \in \mathbb{P}^n \,|\, x_i \neq 0\}$$

in $\mathbb{P}^n$ will be called an **affine piece of** $\mathbb{P}^n$. With respect to the affine piece $\mathbb{A}_i$, the plane $H_\infty : x_i = 0$ will be called **the plane at infinity**. ■

In general the affine piece $\mathbb{A}_n$ will be used and the point with homogeneous coordinates

$$(x_0, \ldots, x_{n-1}, x_n) \sim (y_1, \ldots, y_n, 1) \ ,$$

where $y_i = x_{i-1}/x_n$, will be identified with the point $(y_1, \ldots, y_n)$ in n-dimensional affine space $\mathbb{A}^n$.

Projective space = Affine space + Plane at infinity

In $\mathbb{P}^2$ the plane at infinity is usually called **the line at infinity**, denoted l_∞.

Affine structure in $\mathbb{P}^n$

Definition 31. When a plane at infinity has been chosen, two lines are said to be **parallel** if they intersect at a point at infinity. ■

Example 32. The lines $y_1 = 0$ and $y_1 = 1$ in $\mathbb{A}^2$, is represented in $\mathbb{P}^2$ in homogeneous coordinates as

$$x_0 = 0 \text{ and } x_0 = x_2 \ ,$$

which have the common solution $x_0 = x_2 = 0, x_1 = t$, i.e. the point

$$(0, 1, 0)$$

in $\mathbb{P}^2$ representing the direction $(0, 1)$. ■

Affine transformations in $\mathbb{P}^n$

Definition 32. When a plane at infinity has been chosen, the subgroup of projective transformations preserving the plane at infinity will be called **affine transformations**. ■

Assume that the plane at infinity is given by $x_n = 0$. Then the affine transformations can be written as collineations described by the matrix

$$H = \begin{bmatrix} A & b \\ \mathbf{0} & 1 \end{bmatrix} \ ,$$

where A denotes a nonsingular $(n \times n)$ matrix and b an n-vector.

The corresponding affine transformation is

$$x \mapsto Ax + b \ .$$

Euclidean structure and the Absolute Conic

Definition 33. The conic, Ω, described by

$$x_0^2 + x_1^2 + x_2^2 = 0 \text{ and } x_3 = 0$$

(on the plane at infinity) in $\mathbb{P}^3$ will be called the **absolute conic**. ■

Theorem 3. *The subgroup of projective transformations that preserves the absolute conic can be written as collineation of the type*

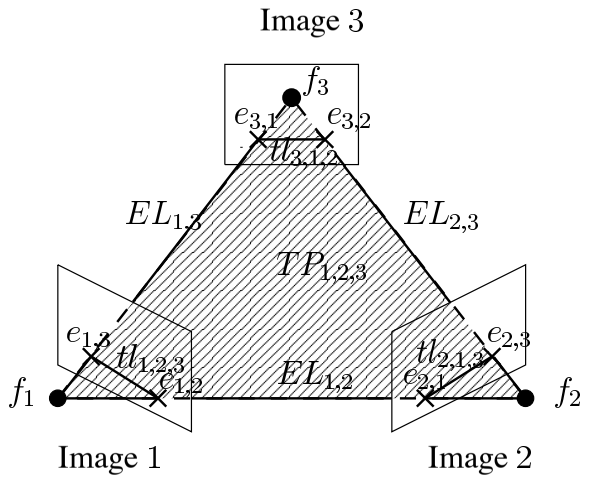
$$H = \begin{bmatrix} cR & t \\ \mathbf{0} & 1 \end{bmatrix} \ ,$$

where R denotes a orthogonal (3×3) matrix, t a 3-vector and $c \in \mathbb{R}$, i.e. a similarity transformation.

The absolute conic induces a Euclidean structure in $\mathbb{P}^3$

5. Multiple View Tensors

Goal: *To understand, computationally and theoretically, the geometry for multiple projective transformations.*



The epipoles

Definition 34. The **epipole** is the projection of the focal point of one camera in another image. Sometimes we say the epipole from camera i in image j. ■

Example 33. Let

$$P_1 = [\, A_1 \mid b_1 \,] \quad P_2 = [\, A_2 \mid b_2 \,] \ .$$

Then the c_1 is given by

$$P_1 \begin{bmatrix} c_1 \\ 1 \end{bmatrix} = [\, A_1 \mid b_1 \,] \begin{bmatrix} c_1 \\ 1 \end{bmatrix} = A_1 c_1 + b_1 = 0 \ ,$$

i.e. $c_1 = -A_1^{-1} b_1$ and the epipole in the second camera is

$$P_2 \begin{bmatrix} c_1 \\ 1 \end{bmatrix} = [\, A_2 \mid b_2 \,] \begin{bmatrix} c_1 \\ 1 \end{bmatrix} = A_2 c_1 + b_2 = -A_2 A_1^{-1} b_1 + b_2 \ .$$

A fundamental limitation

Theorem 4. *Given an uncalibrated image sequence with corresponding points, then it is only possible to reconstruct the object up to an unknown projective transformation.*

Proof. Assume that $\mathbf{X}_j$ is a reconstruction of n points in m images, with camera matrices P_i according to

$$\mathbf{x}_{ij} \sim P_i \mathbf{X}_j, \ i = 1, \ldots m, \ j = 1, \ldots n \ .$$

Then $H \mathbf{X}_j$ is also a reconstruction, with camera matrices $P_i H^{-1}$, for every non-singular 4×4 matrix H, since

$$\mathbf{x}_{ij} \sim P_i \mathbf{X}_j \sim P_i H^{-1} H \mathbf{X}_j \sim (P_i H^{-1}) (H \mathbf{X}_j) \ .$$

The transformation

$$\mathbf{X} \mapsto H \mathbf{X}$$

corresponds to all projective transformations of the object. ■

One consequence

Assume that we have calculated two camera matrices, representing the two-view geometry:

$$P_1 = [\, A_1 \mid b_1 \,] \quad P_2 = [\, A_2 \mid b_2 \,] \ .$$

Then we can multiply by

$$H = \begin{bmatrix} A_1^{-1} & -A_1^{-1} b_1 \\ 0 & 1 \end{bmatrix}$$

from the right and obtain

$$\bar{P}_1 = P_1 H = [\, I \mid 0 \,] \quad \bar{P}_2 = P_2 H = [\, A_2 A_1^{-1} \mid b_2 - A_2 A_1^{-1} b_1 \,] \ .$$

We may always assume that the first camera matrix is $[\, I \mid 0 \,]$

Relation to the epipole

Observe that $\bar{P}_2 = [\, A_{12} \mid e \,]$, where e denotes the epipole in the second image.

Observe also that we may multiply again with

$$\bar{H} = \begin{bmatrix} I & 0 \\ v^T & 1 \end{bmatrix}$$

without changing $\bar{P}_1$, but

$$\bar{H}\bar{P}_2 = [\, A_{12} + ev^T \mid e \,] \ ,$$

i.e. the last column of the second camera matrix still represents the epipole.

A first attempt for the two-view case

Consider a fixed point, $\mathbf{X}$, in 2 views:

$$\lambda_1 \mathbf{x}_1 = P_1\mathbf{X} = [\, A_1 \mid b_1 \,]\mathbf{X}, \quad \lambda_2 \mathbf{x}_2 = P_2\mathbf{X} = [\, A_2 \mid b_2 \,]\mathbf{X} \ .$$

Use the first camera equation to solve for X, Y, Z

$$\lambda_1 \mathbf{x}_1 = P_1\mathbf{X} = [\, A_1 \mid b_1 \,]\mathbf{X} = A_1 \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + b_1 \quad \Rightarrow \quad \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = A_1^{-1}(\lambda_1 \mathbf{x}_1 - b_1)$$

and insert into the second one

$$\lambda_2 \mathbf{x}_2 = A_2 A_1^{-1}(\lambda_1 \mathbf{x}_1 - b_1) + b_2 \ ,$$

that is $\mathbf{x}_2$, $A_2A_1^{-1}\mathbf{x}_1$ and $t = A_2A_1^{-1}b_1 + b_2$ are lin. dep.

A detour

If the vectors a, b and c are linearly dependent, then $b \times c$ is orthogonal to a and thus $a.(b \times c) = 0$, where . denotes scalar product and $\times$ vector (cross) product. Written in matrix notation: $a^T T_c b = 0$ where T_c denotes the anti-symmetric matrix that fulfils $T_c(x) = c \times x$, i.e.

$$T_c = \begin{bmatrix} 0 & -c_3 & c_2 \\ c_3 & 0 & -c_1 \\ -c_2 & c_1 & 0 \end{bmatrix} \ .$$

Example 34. Let $c = (1, 2, 3)$ then

$$T_c x = \begin{bmatrix} 0 & -3 & 2 \\ 3 & 0 & -1 \\ -2 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -3x_3 + 2x_1 \\ 3x_1 - x_3 \\ -2x_1 + x_2 \end{bmatrix} = c \times x \ .$$

The fundamental matrix

From

$\mathbf{x}_2$, $A_2A_1^{-1}\mathbf{x}_1$ and $t = A_2A_1^{-1}b_1 + b_2$ are lin. dep.

we have

$$\mathbf{x}_1^T A_1^{-T} A_2^T T_t \mathbf{x}_2 = \mathbf{x}_1^T F \mathbf{x}_2 = 0 \ ,$$

with

$$F = A_1^{-T} A_2^T T_t \ ,$$

that is a **bilinear constraint in the image coordinates**, defined by **the fundamental matrix**, F.

Question: How can this procedure be generalized to more than two images?

Answer: Not easily.

Connection to epipoles

Observe that since

$$F = A_1^{-T} A_2^T T_t = A_{12}^T T_e \ ,$$

we immediately obtain

$$Fe = 0 \ .$$

Theorem 5. *The epipole in the second image is obtain as the right nullspace to the fundamental matrix and the epipole in the left image is obtained as the left nullspace to the fundamental matrix.*

The statement about the epipole in the left image follows from symmetry.

Theorem 6. *The fundamental matrix is singular, i.e.* $\det F = 0$.

An interesting observation

From the previous considerations we have the following pair

$$F = A_{12}^T T_e \quad \Leftrightarrow \quad P_1 = [\, I \mid 0 \,], \quad P_2 = [\, A_{12} \mid e \,] \ .$$

Observe that

$$F = A_{12}^T T_e = (A_{12} + ev^T)^T T_e$$

for every vector v, since

$$(A_{12} + ev)^T T_e(x) = A_{12}^T (e \times x) + ve^T (e \times x) = A_{12}^T T_e x \ ,$$

since $e^T(e \times x) = e.(e \times x) = 0$.

Observe that this corresponds to the transformation

$$\bar{H}\bar{P}_2 = [\, A_{12} + ev^T \mid e \,] \ .$$

There are three free parameters in the choice of the second camera matrix when the first is $P_1 = [\, I \mid 0 \,]$.

Matrix formulation of camera equations

Consider one object point and its m images: $(\lambda_i \mathbf{x}_i = P_i \mathbf{X}, i = 1 \ldots m)$

$$\underbrace{\begin{bmatrix} P_1 & \mathbf{x}_1 & 0 & 0 & \ldots & 0 \\ P_2 & 0 & \mathbf{x}_2 & 0 & \ldots & 0 \\ P_3 & 0 & 0 & \mathbf{x}_3 & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ P_m & 0 & 0 & 0 & \ldots & \mathbf{x}_m \end{bmatrix}}_{M} \begin{bmatrix} \mathbf{X} \\ -\lambda_1 \\ -\lambda_2 \\ -\lambda_3 \\ \vdots \\ -\lambda_m \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \ .$$

which gives

$$\operatorname{rank} M < m + 4$$

Laplace expansions

The rank condition implies that all $(m+4) \times (m+4)$ minors of M are equal to 0.

These can be written (using Laplace expansions) as sums of products of determinants of **four rows** taken from the first four columns of M and of image coordinates.

There are 3 different categories of such minors depending on the number of rows taken from each image, since one row has to be taken from each image and then the remaining 4 rows can be distributed freely.

Note: Use (x^1, x^2) instead of (x, y) for image coordinates and let $x^3 = 1$. (or regard (x^1, x^2, x^3) as homogeneous coordinates in a projective space)

A brief review on Laplace expansions

Given a quadratic $n \times n$ matrix

$$M = \begin{bmatrix} A & B \end{bmatrix} \ ,$$

where A denotes an $n \times k$ matrix and B and $n \times (n-k)$ matrix, then

$$\det M = \epsilon(I) \sum_{I} \det(A)_I \det(B)_{I'} \ ,$$

where

I denotes all subsets of k indices out of n,
I' the complementary index set,
$\epsilon(I)$ the permutation of I and
$(A)_I$ the submatrix obtained from the rows given by I.

An example

Example 35.

$$\det \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 5 \\ 3 & 4 & 5 & 6 \\ 4 & 5 & 6 & 7 \end{bmatrix} =$$

$$= \det \begin{bmatrix} 1 & 2 \\ 2 & 3 \end{bmatrix} \det \begin{bmatrix} 5 & 6 \\ 6 & 7 \end{bmatrix} - \det \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \det \begin{bmatrix} 4 & 5 \\ 6 & 7 \end{bmatrix} + \det \begin{bmatrix} 1 & 2 \\ 4 & 5 \end{bmatrix} \det \begin{bmatrix} 4 & 5 \\ 5 & 6 \end{bmatrix}$$

$$- \det \begin{bmatrix} 2 & 3 \\ 3 & 4 \end{bmatrix} \det \begin{bmatrix} 3 & 4 \\ 6 & 7 \end{bmatrix} + \det \begin{bmatrix} 2 & 3 \\ 4 & 5 \end{bmatrix} \det \begin{bmatrix} 3 & 4 \\ 5 & 6 \end{bmatrix} - \det \begin{bmatrix} 3 & 4 \\ 4 & 5 \end{bmatrix} \det \begin{bmatrix} 3 & 4 \\ 4 & 5 \end{bmatrix} =$$

$$= (-1)*(-1) - (-2)*(-2) + (-3)*(-1) - (-1)*(-3) + (-2)*(-2) - (-1)*(-1) =$$

$$= 1 - 4 + 3 - 3 + 4 - 1 = 0$$

The three different types

1. Take the 2 remaining rows from one camera matrix and the 2 remaining rows from another camera matrix, gives 2-view constraints.
2. Take the 2 remaining rows from one camera matrix, 1 row from another and 1 row from a third camera matrix, gives 3-view constraints.
3. Take 1 row from each of four different camera matrices, gives 4-view constraints.

Observe that the determinant can be factorized as a product of the 2-, 3- or 4-view constraint and image coordinates in the other images.

The two-view constraint

Considering minors obtained by taking 3 rows from one image, and 3 rows from another image:

$$\det \begin{bmatrix} P_1 & \mathbf{x}_1 & 0 \\ P_2 & 0 & \mathbf{x}_2 \end{bmatrix} = \det \begin{bmatrix} P_1^1 & x_1^1 & 0 \\ P_1^2 & x_1^2 & 0 \\ P_1^3 & x_1^3 & 0 \\ P_2^1 & 0 & x_2^1 \\ P_2^2 & 0 & x_2^2 \\ P_2^3 & 0 & x_2^3 \end{bmatrix} = 0 \ ,$$

which gives a **bilinear constraint**:

$$\sum_{i,j=1}^{3} F_{ij} \mathbf{x}_1^i \mathbf{x}_2^j = 0 \ .$$

The bifocal tensor

The bifocal tensor F_{ij} is defined by

$$F_{ij} = \sum_{i',i'',j',j''=1}^{3} \epsilon_{ii'i''}\epsilon_{jj'j''} \det \begin{bmatrix} P_1^{i'} \\ P_1^{i''} \\ P_2^{j'} \\ P_2^{j''} \end{bmatrix} .$$

The indices tell us which row to exclude from the camera matrices respectively

The bifocal tensor is covariant in both indices

An example

Assume that

$$P_1 = \begin{bmatrix} 1 & 1 & 2 & 2 \\ 2 & 1 & 2 & 3 \\ 1 & 2 & 1 & 3 \end{bmatrix} \quad P_2 = \begin{bmatrix} 1 & 2 & 1 & 2 \\ 2 & 1 & 2 & 1 \\ 1 & 1 & 1 & 2 \end{bmatrix} .$$

Then

$$F_{21} = -\det \begin{bmatrix} 1 & 1 & 2 & 2 \\ 1 & 2 & 1 & 3 \\ 2 & 1 & 2 & 1 \\ 1 & 1 & 1 & 2 \end{bmatrix} = 2, \quad \text{etc.} \quad \Rightarrow \quad F = \begin{bmatrix} 0 & 0 & 0 \\ 2 & -1 & 3 \\ 2 & -1 & -6 \end{bmatrix} .$$

The three-view constraint

Considering minors obtained by taking 3 rows from one image, 2 rows from another image and 2 rows from a third image, e.g.

$$\det \begin{bmatrix} P_1^1 & x_1^1 & 0 & 0 \\ P_1^2 & x_1^2 & 0 & 0 \\ P_1^3 & x_1^3 & 0 & 0 \\ P_2^1 & 0 & x_2^1 & 0 \\ P_2^2 & 0 & x_2^2 & 0 \\ P_3^1 & 0 & 0 & x_3^1 \\ P_3^3 & 0 & 0 & x_3^3 \end{bmatrix} = 0 ,$$

gives the **trilinear constraints**:

$$\sum_{i,j,j',k,k'=1}^{3} T_i^{jk} \mathbf{x}_1^i \epsilon_{jj'j''} \mathbf{x}_2^{j'} \epsilon_{kk'k''} \mathbf{x}_3^{k'} = 0 .$$

The trilinear constraints

Note that there are in total 9 constraints indexed by j'' and k'' in

$$\sum_{i,j,j',k,k'=1}^{3} T_i^{jk} \mathbf{x}_1^i \epsilon_{jj'j''} \mathbf{x}_2^{j'} \epsilon_{kk'k''} \mathbf{x}_3^{k'} = 0 .$$

Observe that the order of the images are important, since the first image is treated differently. If the images are permuted another set of coefficients are obtained.

The trifocal tensor

In the trifocal tensor T_i^{jk} is defined by

$$T_i^{jk} = \sum_{i',i''=1}^{3} \epsilon_{ii'i''} \det \begin{bmatrix} P_1^{i'} \\ P_1^{i''} \\ P_2^{j} \\ P_3^{k} \end{bmatrix} .$$

The lower index tells us which row to exclude from the first camera matrix and the upper indices tell us which rows to include from the second and third camera matrices respectively.

The trifocal tensor is covariant in one index and contravariant in the other two indices.

Note: When an index tells which row to *exclude* that index becomes *covariant* and when an index tells which row to *include* that index becomes *contravariant*.

Example

Let

$$P_1 = \begin{bmatrix} P_1^1 \\ P_1^2 \\ P_1^3 \end{bmatrix} \quad P_2 = \begin{bmatrix} P_2^1 \\ P_2^2 \\ P_2^3 \end{bmatrix} \quad P_3 = \begin{bmatrix} P_3^1 \\ P_3^2 \\ P_3^3 \end{bmatrix} ,$$

where P_1^1 etc. denote the rows of the camera matrices.

Then

$$T_1^{23} = \det \begin{bmatrix} P_1^2 \\ P_1^3 \\ P_2^2 \\ P_3^3 \end{bmatrix}$$

The four-view constraint

Considering minors obtained by taking 2 rows from each one of 4 different images gives the **quadrilinear constraints**:

$$\sum_{i,i',j,j',k,k',l,l'=1}^{3} Q^{ijkl} \epsilon_{ii'i''} \mathbf{x}_{i'}^1 \epsilon_{jj'j''} \mathbf{x}_{j'}^2 \epsilon_{kk'k''} \mathbf{x}_{k'}^3 \epsilon_{ll'l''} \mathbf{x}_{l'}^4 = 0 .$$

Note that there are in total 81 constraints indexed by i'', j'', k'' and l''.

The quadrifocal tensor

The quadrifocal tensor Q^{ijkl} is defined by

$$Q^{ijkl} = \det \begin{bmatrix} P_1^{i} \\ P_2^{j} \\ P_3^{k} \\ P_4^{l} \end{bmatrix} .$$

Again, the upper indices tell us which rows to include from each camera matrix respectively and they become contravariant indices.

Example

Let

$$P_1 = \begin{bmatrix} P_1^1 \\ P_1^2 \\ P_1^3 \end{bmatrix} \quad P_2 = \begin{bmatrix} P_2^1 \\ P_2^2 \\ P_2^3 \end{bmatrix} \quad P_3 = \begin{bmatrix} P_3^1 \\ P_3^2 \\ P_3^3 \end{bmatrix} \quad P_4 = \begin{bmatrix} P_4^1 \\ P_4^2 \\ P_4^3 \end{bmatrix} .$$

Then

$$Q^{1213} = \det \begin{bmatrix} P_1^1 \\ P_2^2 \\ P_3^1 \\ P_3^3 \end{bmatrix}$$

Geometric Interpretation

The multilinear constraint can be interpreted as follows:

- bilinear constraint : intersection of two lines
- trilinear constraint : intersection of two planes and one line
- quadrilinear constraint : intersection of three planes

The monofocal tensor

All types of minors of the first four rows of M has been used apart from 3 rows from one image and 1 row from another.

For completeness, we introduce the **monofocal tensor**:

$$e^j = \det \begin{bmatrix} P_1^1 \\ P_1^2 \\ P_1^3 \\ P_2^j \end{bmatrix} .$$

The j:th component of the **epipole** from camera 1 in image 2, i.e. the projection of the centre of camera 1 in camera 2.

Contravariant of degree 1.

6. Linear estimation of tensors

The multilinear constraints can be used to estimate the tensor components from corresponding feature points in the images.

The multilinear constraints can also be used find correspondences using so called RANSAC (RANdom SAmpling Conscencus) techniques.

Estimating the bifocal tensor

Each point correspondence gives one linear constraint on the components of the bifocal tensor according to the bilinear constraint:

$$F_{ij}\mathbf{x}_1^i\mathbf{x}_2^j = 0 \ ,$$

written explicitly:

$$\begin{aligned} &F_{11}\mathbf{x}_1^1\mathbf{x}_2^1 + F_{12}\mathbf{x}_1^1\mathbf{x}_2^2 + F_{13}\mathbf{x}_1^1\mathbf{x}_2^3 + \\ &F_{21}\mathbf{x}_1^2\mathbf{x}_2^1 + F_{22}\mathbf{x}_1^2\mathbf{x}_2^2 + F_{23}\mathbf{x}_1^2\mathbf{x}_2^3 + \\ &F_{31}\mathbf{x}_1^3\mathbf{x}_2^1 + F_{32}\mathbf{x}_1^3\mathbf{x}_2^2 + F_{33}\mathbf{x}_1^3\mathbf{x}_2^3 = 0 \ , \end{aligned}$$

i.e. a linear constraint on the tensor components.

Each pair of corresponding points give one linear constraint on the nine homogeneous tensor components.

The eight-point algorithm

Introduce

$$\tilde{F} = [\, F_{11}\, F_{12}\, \ldots\, F_{33}]^T \ .$$

Each pair of corresponding points gives a constraint on $\tilde{F}$:

$$c^T\tilde{F} = 0 \ .$$

Given at least eight corresponding points, stack the constraint in a big matrix as

$$M = \begin{bmatrix} c_1^T \\ \vdots \\ c_8^T \end{bmatrix} \ ,$$

which gives the equation

$$M\tilde{F} = 0 \ .$$

The singular value decomposition

Theorem 7. *Given an arbitrary $m \times n$ ($m < n$) matrix M, it has a unique factorization as*

$$M = U\Sigma V^T \ ,$$

where U and V are orthogonal matrices and

$$\Sigma = \begin{bmatrix} \mathrm{diag}(\sigma_1, \ldots, \sigma_m) & 0 \end{bmatrix} \ ,$$

with $\sigma_1 > \sigma_2 > \ldots > \sigma_m$.

The numbers σ_i are called the **singular values of** M.

Theorem 8. *The minimization problem*

$$\min_{||x||=1} ||Mx||$$

has the solution $x =$ the last column of V in the singular value decomposition of M and the minimum value is equal to σ_m.

The eight-point algorithm (contd.)

Solve

$$M\tilde{F} = 0$$

by solving the minimization problem

$$\min_{||x||=1} ||Mx||$$

by using the singular value decomposition of M.

Note: This procedure works even if we have more than eight points.

Note: It is advantageous to scale the image coordinates, e.g. using $(x, y, 100)$ instead of $(x, y, 1)$, when $x, y \approx 100$.

Reconstructing the scene

When the fundamental matrix has been obtained, factorize as

$$F = A_{12}^T T_e \ ,$$

which can be done by finding e as the right nullspace to F (by SVD) and then solving a linear system of equations for A_{12}, which is under-constrained.

One solution is

$$A_{12} = \begin{bmatrix} 0 & 0 & 0 \\ F_{13} & F_{23} & F_{33} \\ -F_{12} & -F_{22} & -F_{32} \end{bmatrix} \ ,$$

which can be seen from the definition of the tensor components.

Reconstructing the scene (contd.)

Then the camera matrices are obtained as before:

$$P_1 = [\, I \,|\, 0 \,], \quad P_2 = [\, A_{12} \,|\, e \,] \ .$$

Finally the reconstruction is obtained from

$$\begin{bmatrix} P_1 & \mathbf{x}_1 & 0 \\ P_2 & 0 & \mathbf{x}_2 \end{bmatrix} = \begin{bmatrix} X \\ -\lambda_1 \\ -\lambda_2 \end{bmatrix} = 0 \ ,$$

using SVD to find the nullspace.

RANSAC

The epipolar constraint

$$\mathbf{x}_1^T F \mathbf{x}_2 = 0$$

can be used when the correspondence problem is treated, since it constrains the matches.

The following algorithm (RANSAC) has been proven successful:

Given a number of extracted points in two images:

1. Choose at random 8 points from each image and calculate the essential matrix.
2. For each point in the first image find a corresponding point (if possible) on the corresponding epipolar line.
3. Repeat from 1 until a sufficient number of correspondences fulfilling the epipolar constraint has been obtained.

It is advisable to attach a score to each match, calculated by correlation and use these scores to get a measure of the quality of each match.

Estimating the trifocal tensor

Each point correspondence gives 9 linear constraint on the components of the trifocal tensor according to the trilinear constraints:

$$T_i^{jk} \mathbf{x}_1^i \epsilon_{jj'j''} \mathbf{x}_2^{j'} \epsilon_{kk'k''} \mathbf{x}_3^{k'} = 0 \ .$$

However, there are only 4 linearly independent constraints for each point triplet.

Since we have 27 homogeneous components of the trifocal tensor, we need at least 7 corresponding points to estimate the components linearly.

The seven-point algorithm

We can proceed in the same way as in the eight-point algorithm.

Introduce the 27-vector

$$\tilde{T} = [T_1^{11} \dots T_3^{33}]^T \ .$$

Each triplet of corresponding points gives 9 constraints on $\tilde{T}$:

$$c_i^T \tilde{T} = 0, \quad i = 1, \dots, 9 \ .$$

Given at least seven corresponding points, stack the constraint in a big matrix as

$$M = \begin{bmatrix} c_1^T \\ \vdots \end{bmatrix} \ ,$$

which gives the equation

$$M\tilde{T} = 0 \ .$$

Estimating the quadrifocal tensor

Each point correspondence gives 81 linear constraint on the components of the quadrifocal tensor according to the quadrilinear constraints:

$$Q^{ijkl}\epsilon_{ii'i''}\mathbf{x}_1^{i'}\epsilon_{jj'j''}\mathbf{x}_2^{j'}\epsilon_{kk'k''}\mathbf{x}_3^{k'}\epsilon_{ll'l''}\mathbf{x}_4^{l'} = 0 \ .$$

However, there are only 16 linearly independent constraint for each point quadruplet.

Moreover, given two different point quadruplets, there are in total only 31 linearly independent constraint because:

$$Q^{ijkl}\epsilon_{ii'i''}\epsilon_{jj'j''}\epsilon_{kk'k''}\epsilon_{ll'l''}\mathbf{x}^{i'}\mathbf{x}^{j'}\mathbf{x}^{k'}\mathbf{x}^{l'}\hat{\mathbf{x}}^{i''}\hat{\mathbf{x}}^{j''}\hat{\mathbf{x}}^{k''}\hat{\mathbf{x}}^{l''} = 0 \ .$$

Since we have 81 homogeneous components of the quadrifocal tensor and $16n - n(n-1)/2$ linearly independent constraints for n corresponding points, we need at least 6 corresponding points to estimate the components linearly.

The six-point algorithm

We can proceed in the same way as in the eight-point algorithm.

Introduce the 81-vector

$$\tilde{Q} = [Q^{1111} \dots Q^{3333}]^T \ .$$

Each quadruplet of corresponding points gives 81 constraints on $\tilde{Q}$:

$$c_i^T \tilde{Q} = 0, \quad i = 1, \dots, 81 \ .$$

Given at least six corresponding points, stack the constraint in a big matrix as

$$M = \begin{bmatrix} c_1^T \\ \vdots \end{bmatrix} \ ,$$

which gives the equation

$$M\tilde{Q} = 0 \ .$$

7. Tensorial Transfer

There are a number of intersecting properties of the multiple view tensors:

- Tensorial transfer
- Calculation of one tensor from another
- Constraints on the tensor components

We will only deal with tensorial transfer.

Transfer using the bifocal tensor

The bifocal tensor can be used to transfer a point, $\mathbf{x}_2$, in image 2 to the corresponding epipolar line, $\mathbf{l}^1$ in image 1:

$$\mathbf{l}^1_i = F_{ij}\mathbf{x}^j_2 \ .$$

The tensor also encodes a homography between epipolar lines in the first view and epipolar lines in the second view according to

$$\mathbf{l}^1_i = F_{ij}\epsilon_{j''j'}\mathbf{l}^2_{j'}\mathbf{e}^{j''} \ ,$$

since $\epsilon_{j''j'}\mathbf{l}^2_{j'}\mathbf{e}^{j''}$ gives the cross product between the epipole $\mathbf{e}$ and the line , which gives a point on the epipolar line.

Transfer using the trifocal tensor

The trifocal tensor can be used to transfer two corresponding lines, $\mathbf{l}^2$ and $\mathbf{l}^3$ to the corresponding line, $\mathbf{l}^1$ according to

$$\mathbf{l}^1_i = T^{jk}_i\mathbf{l}^2_j\mathbf{l}^3_k$$

There are also other transfer equations such as:

$$\mathbf{x}^j_2 = T^{jk}_i\mathbf{x}^i_1\mathbf{l}^3_k$$

and

$$\mathbf{x}^k_3 = T^{jk}_i\mathbf{x}^i_1\mathbf{l}^2_j$$

Applications to view synthesis

The transfer equation

$$\mathbf{x}^k_3 = T^{jk}_i\mathbf{x}^i_1\mathbf{l}^2_j$$

can be used to transfer corresponding points in the second and third image to the third image by inserting any line through the point in the third image for $\mathbf{l}^2_j$.

Transfer using the quadrifocal tensor

The quadrifocal tensor can be used to transfer three corresponding lines, $\mathbf{l}^2$, $\mathbf{l}^3$ and $\mathbf{l}^4$ to the corresponding point, $\mathbf{x}_1$:

$$\mathbf{x}^i_1 = Q^{ijkl}\mathbf{l}^2_j\mathbf{l}^3_k\mathbf{l}^3_l$$

3. Factorization and bundle adjustment

Motivation

- Multiple view tensors are not always best suited to reconstruction problems
- Especially when many images are available
- Other methods incorporate all points in all images at the same time
- Two methods will be presented:
  - Iterative Factorization: Fast, no initial values needed, non-optimal
  - Bundle Adjustment: Slow, initial values needed, optimal

Factorization Methods

The camera equations

$$\lambda_{i,j}x_{i,j} = P_i X_j, \quad i = 1,\dots,m, \quad j = 1,\dots,n$$

for a fixed image i can be written in matrix form as

$$\mathbf{x}_i\Lambda_i = P_i\mathbf{X} \ ,$$

where

$$\mathbf{x}_i = \begin{bmatrix} x_{i,1} & x_{i,2} & \dots & x_{i,n} \\ y_{i,1} & y_{i,2} & \dots & y_{i,n} \\ 1 & 1 & \dots & 1 \end{bmatrix}, \quad \mathbf{X} = \begin{bmatrix} X_1 & X_2 & \dots & X_n \\ Y_1 & Y_2 & \dots & Y_n \\ Z_1 & Z_2 & \dots & Z_n \\ 1 & 1 & \dots & 1 \end{bmatrix},$$

$$\Lambda_i = \mathrm{diag}(\lambda_{i,1}, \lambda_{i,2}, \dots, \lambda_{i,n}) \ .$$

All equations can be collected for all i as

$$\hat{\mathbf{x}} = \mathbf{P}\mathbf{X} \ ,$$

where

$$\hat{\mathbf{x}} = \begin{bmatrix} \mathbf{x}_1\Lambda_1 \\ \mathbf{x}_2\Lambda_2 \\ \vdots \\ \mathbf{x}_m\Lambda_m \end{bmatrix}, \quad \mathbf{P} = \begin{bmatrix} \mathbf{P}_1 \\ \mathbf{P}_2 \\ \vdots \\ \mathbf{P}_m \end{bmatrix} \ .$$

In these formulas $\mathbf{x}$ are known, but both Λ_i, $\mathbf{P}$ and $\mathbf{X}$ are unknown.

Observe that $\mathbf{PX}$ is a product of a $3m \times 4$ matrix and a $4 \times n$ matrix, i.e. it is a rank 4 matrix.

Factorization

Assume that Λ_i are known (this corresponds to the affine case), then $\hat{\mathbf{x}}$ is known.

Use the singular value decomposition of the left hand side of

$$\hat{\mathbf{x}} = \mathbf{P}\mathbf{X} \quad \Rightarrow \quad \hat{\mathbf{x}} = U\Sigma V \ .$$

In the noise-free case $\Sigma = \mathrm{diag}(\sigma_1, \sigma_2, \sigma_3, \sigma_4, 0, \dots, 0)$ and a reconstruction can be obtained by setting:

$\mathbf{P}$ =the first four columns of $U\Sigma$.

$\mathbf{X}$ =the first four rows of V.

In the case of measurement noise on may replace Σ by $\tilde{\Sigma} = \mathrm{diag}(\sigma_1, \sigma_2, \sigma_3, \sigma_4, 0, \dots, 0)$.

$$\hat{\mathbf{x}} = U\Sigma V \approx U\tilde{\Sigma}V = \mathbf{P}\mathbf{X}$$

terative Factorization

When Λ is unknown the following algorithm can be used:

. Set $\lambda_{i,j} = 1$ (affine approximation).

. Factorize $\hat{\mathbf{x}}$ and obtain an estimate of $\mathbf{P}$ and $\mathbf{X}$. If σ_5 is sufficiently mall then STOP.

. Use $\hat{\mathbf{x}}$, $\mathbf{P}$ and $\mathbf{X}$ to estimate Λ_i from the camera equations (linearly).

$$\mathbf{x}_i \underbrace{\Lambda}_{i} = P_i \mathbf{X}$$

. Goto 2.

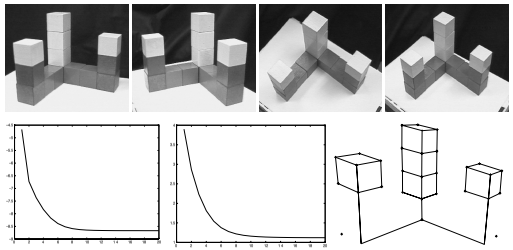
n general the algorithm minimizes the **proximity measure**:

$$\mathcal{P}(\Lambda, \mathbf{P}, \mathbf{X}) = \sigma_5 \ .$$

he performance of the numerics is better when normalized coordinates re used.

Experiment - Real data

- 32 corresponding points
- 4 images



- proximity measure
- estimated standard deviation of noise from reprojected image points
- reconstruction

Bundle Adjustments

Formulation

Consider again the camera equations

$$\lambda_{i,j}\mathbf{x}_{i,j} = P_i\mathbf{X}_j, \quad i = 1,\ldots,m, \quad j = 1,\ldots,n \ . \qquad (1)$$

Assume that we have measurements

$$\tilde{\mathbf{x}}_{i,j} = \mathbf{x}_{i,j} + \mathbf{n}_{i,j} \ ,$$

where $\mathbf{n}_{i,j}$ denote uncorrelated normally distributed noise.

ntroduce parameters for $\lambda_{i,j}$, P_i and $\mathbf{X}_j$ and try to minimize

$$\sum_{i,j} ||\tilde{\mathbf{x}}_{i,j} - \mathbf{x}_{i,j}||$$

ver these parameters when $x_{i,j}$ are calculated from (1).

his can be done using the Gauss-Newton method or generalized least quares.

Optimization problem

Introduce

$$\mathbf{x}_{ij} = \text{pflat}(P_i * \mathbf{X}_j) \ ,$$

i.e. reprojected image points from current estimate of structure ($\mathbf{X}_j$) and motion (P_i) parameters.

The goal is to solve the following optimization problem:

$$\min_{P_i, \mathbf{X}_j} ||\tilde{\mathbf{x}}_{ij} - \mathbf{x}_{ij}|| = f(\mathbf{P}, \mathbf{X}) \ .$$

The algorithm

The bundle of parameters:

$$\mathfrak{m} = \{P_1, \ldots, P_m, \mathbf{X}_1, \ldots, \mathbf{X}_n\} \in \mathcal{M} \ .$$

Local parameterization:

$$\mathcal{M} \times R^N \ni (\mathfrak{m}_0, \Delta\mathbf{x}) \mapsto \mathfrak{m}(\mathfrak{m}_0, \Delta\mathbf{x}) \in \mathcal{M} \ ,$$

where $N = 11m + 3n$ and

$$\Delta\mathbf{x} = [\, \Delta a_1, \ldots, \Delta a_m, \Delta b_1, \ldots, \Delta b_n \,]^T$$

so that Δa_i parametrise changes in camera matrix P_i and Δb_j parametrise changes in reconstructed point X_j.

$$P_i(\mathfrak{m}_0, \Delta\mathbf{x}) =$$

$$\begin{bmatrix} (P_i)_{1,1}+\Delta a_i(1) & (P_i)_{1,2}+\Delta a_i(2) & (P_i)_{1,3}+\Delta a_i(3) & (P_i)_{1,4}+\Delta a_i(4) \\ (P_i)_{2,1}+\Delta a_i(5) & (P_i)_{2,2}+\Delta a_i(6) & (P_i)_{2,3}+\Delta a_i(7) & (P_i)_{2,4}+\Delta a_i(8) \\ (P_i)_{3,1}+\Delta a_i(9) & (P_i)_{3,2}+\Delta a_i(10) & (P_i)_{3,3}+\Delta a_i(11) & (P_i)_{3,4}+\Delta a_i(12) \end{bmatrix}$$

$$\mathbf{X}_j(\mathfrak{m}_0, \Delta\mathbf{x}) = \begin{bmatrix} X_j+\Delta b_j(1) & Y_j+\Delta b_j(2) & Z_j+\Delta b_j(3) & 1 \end{bmatrix}^T$$

Linearization

Residual vector: $\mathbf{Y} = (y_1, y_2, \ldots)$ =all errors in a column vector.

$\mathbf{Y}(\Delta\mathbf{x})$ is a non-linear function of the local parametrisation vector $\Delta\mathbf{x}$.

We would like to minimize: $f = \mathbf{Y}^T\mathbf{Y} = \sum_i y_i^2$.

Linearization of $\mathbf{Y}(\Delta\mathbf{x})$ gives:

$$\mathbf{Y}(\Delta\mathbf{x}) \approx \mathbf{Y}(0) + \frac{\partial \mathbf{Y}}{\partial \Delta\mathbf{x}}(0)\, \Delta\mathbf{x} \ .$$

Solve:

$$\mathbf{Y}(0) + \frac{\partial \mathbf{Y}}{\partial \Delta\mathbf{x}}(0)\, \Delta\mathbf{x} = 0 \ ,$$

giving

$$\Delta\mathbf{x} = -\left(\frac{\partial \mathbf{Y}}{\partial \Delta\mathbf{x}}(0)\right)^+ \mathbf{Y}(0) \ .$$

The error vector

Note that the residual vector looks like:

$$Y = \begin{bmatrix} \tilde{\mathbf{x}}_{1,1} - \mathbf{x}_{1,1} \\ \tilde{\mathbf{x}}_{1,2} - \mathbf{x}_{1,2} \\ \tilde{\mathbf{x}}_{1,3} - \mathbf{x}_{1,3} \\ \vdots \\ \tilde{\mathbf{x}}_{2,1} - \mathbf{x}_{2,1} \\ \tilde{\mathbf{x}}_{2,2} - \mathbf{x}_{2,2} \\ \tilde{\mathbf{x}}_{2,3} - \mathbf{x}_{2,3} \\ \vdots \\ \tilde{\mathbf{x}}_{m,n} - \mathbf{x}_{m,n} \end{bmatrix} \ ,$$

where $\tilde{\mathbf{x}}_{i,j} - \mathbf{x}_{i,j}$ contains the error in both the x-direction and the y-direction.

Algorithm

Let

$$A = \frac{\partial \mathbf{Y}}{\partial \Delta \mathbf{x}}(0)$$

and

$$b = \mathbf{Y}(0) \ .$$

Use Levenberg-Marquart:

$$\Delta \mathbf{x} = -(A^T A + \epsilon I)^{-1} A^T b \ ,$$

where ϵ is a small positive number.

Note: $A^T A$ gives an estimate of the covariance matrix of the estimated parameters.

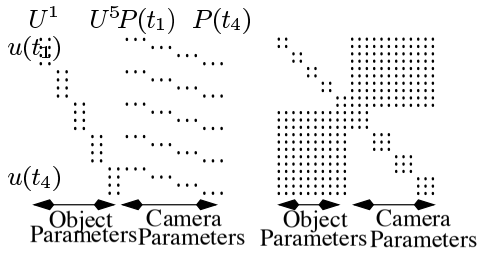
Initial values can be obtained by previously mentioned methods linearly.

Bundle adjustment gives a more accurate reconstruction, but is computationally more expensive.

The structure of the non-zero elements of the normal equations is important.

$$A\Delta \mathbf{x} \approx -b \ .$$

$$\Delta \mathbf{x} = -(A^T A)^{-1} A^T b \ .$$



9. Flexible calibration

Motivation

- Projective reconstruction not very useful
- Knowledge about the camera motion or scene geometry usually not available
- Constraints on the intrinsic parameters arise naturally (zero skew, unit aspect ratio, etc)
- These constraints can be used to obtain a Euclidean reconstruction

The absolute conic

Remember that the **absolute conic**

$$C_0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

is invariant under similarity transformations.

$$C_0 \mapsto H^T C_0 H \sim C_0$$

when

$$H = \left[\begin{array}{c|c} \text{cR} & \text{t} \\ \hline 0 & 1 \end{array}\right] \ ,$$

Projection of absolute conic with calibrated camera

If the intrinsic calibration is known (and corrected for), then the camera matrix P belong to the **manifold of calibrated camera matrices**,

$$\mathcal{M}_C = \{P \sim [\,R \mid t\,] \,|\, R^T R = 1, \det(R) = 1\} \ .$$

It can be shown that the image ω_0 of the absolute conic is always the same

$$\omega_0 \sim \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = P_i C_0 P_i^T \ .$$

Proof:

$$\omega_0 \sim P C_0 P^T = [\,R \,|\, t\,] \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} R^T \\ t^T \end{bmatrix} =$$

$$= [\,R \,|\, t\,] \begin{bmatrix} R^T \\ 0 \end{bmatrix} = R R^T = I$$

If we change object coordinate system with transformation T, then the representation of the absolute conic changes to

$$C = T C_0 T^T$$

Since C_0 has rank 3 and since T is invertible, C has rank 3. Notice also that C is a symmetric matrix.

If we change image coordinate system with transformation K, then the image of the absolute conic changes to

$$\omega = K \omega_0 K^T = K K^T$$

Correct	Arbitrary
$\hat{P} \in \mathcal{M}_C$	$P = K \hat{P} T^{-1}$
$\hat{C} = C_0$	$C = T \hat{C} T^T$
$\hat{P}_i \hat{C} \hat{P}_i^T = \omega_0$	$P C P^T = \omega$

Additional information

We can distinguish between different projective solutions if additional information is known. Some interesting cases are:

- The intrinsic calibration is known.
- The intrinsic calibration is unknown but constant.
- The intrinsic calibration is unknown but constant and the camera is stationary.
- Only the focal length is unknown and varying.
- Everything but the skew is unknown and varying.

Calibration is equivalent to finding the representation of the absolute conic C and its image ω_i in each image i.

Known intrinsic calibration

If the intrinsic calibration is known (and corrected for), then each camera matrix P_i must belong to the manifold of calibrated camera matrices,

$$\mathcal{M}_C = \{P = [R|r] \,|\, R^T R = 1, \det(R) = 1\} \ .$$

Since we know that the cameras are calibrated it is not possible to change coordinate systems with an arbitrary matrix T.

$$P_i \in \mathcal{M}_C \not\Longrightarrow P_i T \in \mathcal{M}_C$$

It can be shown that if the motion is general enough, then the only ambiguity left is change of Euclidean coordinate system and scale, i.e. a similarity transformation

$$T = \left[\begin{array}{c|c} \text{cR} & \text{t} \\ \hline 0 & 1 \end{array}\right] \ ,$$

Let G_S denote the **group of similarity transformations**

$$G_S = \{T \,|\, T \sim \left[\begin{array}{c|c} \text{cR} & \text{t} \\ \hline 0 & 1 \end{array}\right], R \in SO(3), c > 0\} \ .$$

It can be showed that

$$P_i \in \mathcal{M}_C, T \in G_S \Longrightarrow P_i T \in \mathcal{M}_C$$

This conclusion is that the only transformations that map calibrated camera matrices to calibrated camera matrices are the similarity transformations.

Assume that we have calculated the motion parameters P_i using projective techniques and that the camera matrices are corrected for intrinsic calibration, how can we find a transformation T such that

$$P_i T \in \mathcal{M}_C, \quad \forall i$$

Write T as a block matrix

$$T = \left[\begin{array}{c|c} \text{M} & \text{n} \end{array}\right] \ ,$$

The constraint $P_i T \in \mathcal{M}_C$,

$$P_i \left[\begin{array}{c|c} \text{M} & \text{n} \end{array}\right] \sim \left[\begin{array}{c|c} R_i & t_i \end{array}\right] \ ,$$

implies $P_i M = R_i$ and therefore

$$P_i C P_i^T \sim I = \omega_0, \quad \forall i \ ,$$

with $C = MM^T$.

Observations

The previous equations can be interpreted as the projection of the absolute conic.

Notice also that

$$C = TC_0T^T = [\,M \,|\, n\,] \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} M^T \\ n^T \end{bmatrix} = MM^T$$

nterpretation as absolute conic

he equations

$$P_i C P_i^T \sim I = \omega_0, \quad \forall i \ ,$$

an be interpreted as the projection of the absolute conic.

Correct	Arbitrary
$\hat{P}_i \in \mathcal{M}_C$	$P_i = \hat{P}_i T^{-1}$
$\hat{C} = C_0$	$C = T\hat{C}T^T$
$\hat{P}_i \hat{C} \hat{P}_i^T \sim I \sim \omega_0$	$P_i C P_i^T \sim \omega_0$

y introducing scale factors μ_i the problem becomes **linear** in the nknown parameters,

$$P_i \underbrace{C} P_i^T = \underbrace{\mu_i} \omega_0 \ .$$

olve for C and μ_i using linear algebra methods.

Constant intrinsic parameters Stationary cameras

Choose coordinate system so that the focal point is at the origin.

$$P_i = K[R_i' \,|\, \mathbf{0}], \quad \mathbf{i = 0, \ldots, n} \ .$$

Theorem 9. *Image k and image 0 are related by a 2D projective transformation, represented by the matrix*

$$H_k \sim K R_k K^{-1} \ .$$

A point $\begin{bmatrix} \mathbf{X} \\ 1 \end{bmatrix}$ is projected in image 0 as $\mathbf{x}_0 \sim K R_0' \mathbf{X}$ and in image k as $\mathbf{x}_k \sim K R_k' \mathbf{X}$, therefore

$$\mathbf{x}_k \sim K R_k' (R_0')^{-1} K^{-1} \mathbf{x}_0 \sim K R_k K^{-1} \mathbf{x}_0 \sim H_k \mathbf{x}_0 \ ,$$

with $R_k = R_k'(R_0')^{-1}$.

etting rid of the unknown scale factor

or each projective transformation we get 5 constraints since

$$H_k \sim K R_k K^{-1} \ .$$

he right hand side has determinant one, $\det(K R_k K^{-1}) = 1$.

e can, however, get equality $H_k = K R_k K^{-1}$ if we scale H_k so that et $H_k = 1$.

ow, rewrite the equations

$$H_k = K R_k K^{-1} \quad \Rightarrow \quad H_k K = K R_k$$

ultiply each side with its transpose:

$$H_k K K^T H_k^T = K R_k R_K^T K^T = K K^T$$

nd use the fact that $\omega = K K^T$,

$$H_k \omega H_k^T = \omega$$

Algorithm:

1. Estimate H_k from image point correspondences.

$$\underbrace{\lambda} \mathbf{x}_k = \underbrace{H_k} \mathbf{x}_0$$

The equations are linear in the unknown parameters (λ_k, H_k).

2. Find ω such that

$$H_k \underbrace{\omega} H_k^T = \underbrace{\omega} \qquad (2)$$

The equations are linear in the unknown parameters in ω.

3. Decompose the matrix ω using cholesky factorisation

$$\omega = K K^T$$

Optional: Improve the estimate of K using non-linear optimization.

Constant intrinsic parameters

The absolute conic

Correct	Arbitrary
$\hat{P}_i \in \mathcal{M}_C$	$P_i = K\hat{P}_i T^{-1}$
$\hat{C} = C_0$	$C = T\hat{C}T^T$
$\hat{P}_i\hat{C}\hat{P}_i^T \sim I \sim \omega_0$	$P_i C P_i^T \sim KK^T \sim \omega$

Find C and ω such that $P_i C P_i^T = \omega$ for every i. Non-linear problem.

Guess ω, solve for C using linear methods, improve using non-linar programming.

$$P_i \underbrace{C} P_i^T = \underbrace{\mu_i}\,\underbrace{\omega}$$

Focal length is unknown and varying

Assume known skew=0, aspect ratio=1 and principal point=(0,0):

Correct	Arbitrary
$\hat{P}_i \in \mathcal{M}_C$	$P_i = K_i\hat{P}_i T^{-1}$
$\hat{C} = C_0$	$C = T\hat{C}T^T$
$\hat{P}_i\hat{C}\hat{P}_i^T \sim I \sim \omega_0$	$P_i C P_i^T \sim K_i K_i^T \sim \omega_i$

$$K_i = \begin{bmatrix} f_i & 0 & 0 \\ 0 & f_i & 0 \\ 0 & 0 & 1 \end{bmatrix}, \qquad \omega_i = \begin{bmatrix} f_i^2 & 0 & 0 \\ 0 & f_i^2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Estimating C and f_i from P_i.

$$P_i C P_i^T = P_i \begin{bmatrix} c_1 & c_2 & c_3 & c_4 \\ c_2 & c_5 & c_6 & c_7 \\ c_3 & c_6 & c_8 & c_9 \\ c_4 & c_7 & c_9 & c_{10} \end{bmatrix} P_i^T = \lambda_i\omega_i = \begin{bmatrix} \lambda_i f_i^2 & 0 & 0 \\ 0 & \lambda_i f_i^2 & 0 \\ 0 & 0 & \lambda_i \end{bmatrix}$$

Introduce $a_i = \lambda_i f_i^2$ and $b_i = \lambda_i$.

The unknowns are the parameters of the absolute conic, c_i, and a_i and b_i.
The equation becomes linear in these parameters

$$P_i \underbrace{C} P_i^T = \underbrace{\begin{bmatrix} a_i & 0 & 0 \\ 0 & a_i & 0 \\ 0 & 0 & b_i \end{bmatrix}} \quad .$$

Minimal cases

It can be shown that in theory Euclidean reconstruction is possible even under such weak assumptions that all intrinsic parameters are unknown and varying except one.

Correct	Arbitrary
$\hat{P}_i \in \mathcal{M}_C$	$P_i = K_i\hat{P}_i T^{-1}$
$\hat{C} = C_0$	$C = T\hat{C}T^T$
$\hat{P}_i\hat{C}\hat{P}_i^T \sim I \sim \omega_0$	$P_i C P_i^T \sim K_i K_i^T \sim \omega_i$

$$\omega_i \sim \begin{bmatrix} a_i & b_i & c_i \\ b_i & d_i & e_i \\ c_i & e_i & f_i \end{bmatrix}$$

Zero skew: One interesting case, is to use only that the skew is zero. This puts one constraint on the parameters of ω_i,

$$c_i e_i - b_i f_i = 0 \ .$$

It can be shown that this gives enough information to solve the problem if enough projection matrices are given and if the motion is sufficiently general.

As of today there is no good technique to obtain initial estimates of the intrinsic parameters under these weak assumptions. Usually one assumes p_i approximately known or known except for focal length as in previous exampel.

Non-linear programming has prooved to be useful once such initial estimates are known.

Degenerate motion

Many of these methods fail if the motion of the camera is to restrictive.

For example, auto-calibration under the assumption of constant intrinsic parameters fail if the camera only translates relative to the scene.

These **degenerate cases** is still under intense research.

10. Conclusions

- Definition of multiple view tensors (MVT)
- Properties of MVT
- Usage of MVT for
  - reconstruction problem
  - correspondence problem
  - view synthesis
- Understanding of multiple view geometry

Thank you for your patience

References

[1] O. Faugeras and B. Mourrain. About the correspondence of points between n images. In *IEEE Workshop on Representation of Visual Scenes, MIT, Boston, MA*, pages 37–44. IEEE Computer Society Press, 1995.

[2] O. Faugeras and B. Mourrain. On the geometry and algebra of the point and line correspondences between n images. In *Proc. 5th Int. Conf. on Computer Vision, MIT, Boston, MA*, pages 951–956. IEEE Computer Society Press, 1995. also available in [3].

[3] O. Faugeras and B. Mourrain. On the geometry and algebra of the point and line correspondences between n images. Technical Report 2665, Institut national de rechereche en informatique et en automatique, Okt 1995.

[4] O. Faugeras and T. Papadopoulo. Grassman-cayley algebra for modeling systems of cameras and the algebraic equations of the manifold of trifocal tensors. Technical Report 3225, Institut national de rechereche en informatique et en automatique, july 1997.

[5] O. Faugeras and T. Papadopoulo. Grassman-cayley algebra for modeling systems of cameras and the algebraic equations of the manifold of trifocal tensors. *Transactions of the Royal Society A*, May 1998. also available in [4].

[6] O. Faugeras and T. Papadopoulo. A nonlinear method for estimating the projective geometry of three views. In *Proc. 6th Int. Conf. on Computer Vision, Mumbai, India*, 1998.

[7] R. Hartley. A linear method for reconstruction from points and lines. In *Proc. 5th Int. Conf. on Computer Vision, MIT, Boston, MA*, pages 882–887. IEEE Computer Society Press, 1995.

[8] R. Hartley. Multilinear relationsships between coordinates of corresponding image points and lines. In *Proc. of the Sophus Lie International Workshop on Computer Vision and Applied Geometry, Nordfjordeid, Norway*, 1995. to appear.

[9] R. Hartley. Lines and points in three views and the trifocal tensor. *Int. Journal of Computer Vision*, 22(2):125–140, March 1997.

[10] A. Heyden. Reconstruction from image sequences by means of relative

depths. *Int. Journal of Computer Vision*, 24(2):155–161, September 1997. also in Proc. of the 5th International Conference on Computer Vision, IEEE Computer Society Press, pp. 1058-1063.

[11] A. Heyden. *Multiple View Geometry and Algebra*. Kluwer Verlag, 1999. to appear.

[12] A. Heyden. Tensorial properties of multiple view constraints. *Mathematical Methods in the Applied Scinces*, 1999. to appear.

[13] A. Heyden and K. Åström. Algebraic varieties in multiple view geometry. In B. Buxton and R. Cipolla, editors, *Proc. 4th European Conf. on Computer Vision, Cambridge, UK*, volume 1065 of *Lecture notes in Computer Science*, pages 671–682. Springer-Verlag, 1996.

[14] A. Heyden and K. Åström. Algebraic properties of multilinear constraints. *Mathematical Methods in the Applied Scinces*, 20:1135–1162, 1997.

[15] Q.-T. Luong and T. Vieville. Canonic representations for the geometries of multiple projective views. In J.-O. Eklund, editor, *Proc. 4th European Conf. on Computer Vision, Cambridge, UK*, volume 800 of *Lecture Notes in*

Computer Science, pages 589–599. Springer-Verlag, 1994.

[16] A. Shashua. Trilinearity in visual recognition by alignment. In J.-O. Eklund, editor, *Proc. 4th European Conf. on Computer Vision, Cambridge, UK*, volume 800 of *Lecture Notes in Computer Science*, pages 479–484. Springer-Verlag, 1994.

[17] A. Shashua and M. Werman. On the trilinear tensor of three perspective views and its associated tensor. In *Proc. 5th Int. Conf. on Computer Vision, MIT, Boston, MA*, pages 920–925. IEEE Computer Society Press, 1995.

[18] M. E. Spetsakis and J. Aloimonos. A unified theory of structure from motion. In *Proc. DARPA IU Workshop*, pages 271–283, 1990.

[19] B. Triggs. Matching constraints and the joint image. In *Proc. 5th Int. Conf. on Computer Vision, MIT, Boston, MA*, pages 338–343. IEEE Computer Society Press, 1995.

[20] A. Zisserman. A users guide to the trifocal tensor. Draft, july 1996.