Pose estimation from point and line correspondences

Giorgio Panin

October 17, 2008

1 Problem formulation

Estimate (in a LSE sense) the pose of an object from N correspondences between known object points (3D or 2D) $\mathbf{X}_i$ and their noisy projections on the 2D image plane, $\mathbf{x}_i$.

The pose is represented by a (3×3) or (4×4) homogeneous transformation matrix T, belonging to a given *sub-group* G of object-space transformations. G is a subset of all possible transforms, closed under matrix product:

$$T_1, T_2 \in G \Rightarrow (T_1 \cdot T_2) \in G$$

The intrinsic camera matrix K ((3×3) or (3×4), respectively) is supposed to be known in advance.

Moreover, a reference (constant) transform matrix $\bar{T}$ may be present, which is pre-multiplied by T (e.g. a fixed displacement in articulated structures, with Denavit-Hartenberg parameters), and this matrix may also not belong to the same transformation group G.

Overall, this is a projective geometry problem, that can be formulated in basically two ways: using homogeneous or non-homogeneous coordinates. They raise two different LSE error measures to be optimized: the *algebraic* and *geometric* error, respectively.

The latter is the real target of our estimation (Maximum-Likelihood solution), but usually non-linear, while the former gives redundant and sub-optimal equations, but usually linear. Therefore, we can use the algebraic solution as a starting-point to minimize the geometric error.

1.1 Geometric error

We have the following problem:
Given N exact model points $\mathbf{X}$ and corresponding noisy image points $\mathbf{x}$ in homogeneous coordinates

$$\begin{aligned}&X = (X, Y, 1) \text{ or } X = (X, Y, Z, 1)\\&x = (x, y, 1)\end{aligned}$$

find the optimal transformation T^* belonging to a group G, such that

$$T^* = \min_{T \in G} \sum_{i=1}^{N} \left\| \pi(K \cdot \bar{T} \cdot T \cdot X_i) - \pi(x_i) \right\|^2$$

where π is the nonlinear *projection* operator, from homogeneous to non-homogeneous coordinates

$$\pi(x, y, w) = \left[\begin{array}{cc} x/w & y/w \end{array} \right]^T$$

and $\bar{T}$ is a constant matrix, not necessarily belonging to the same group G.

Solution: nonlinear LSE optimization (Gauss-Newton), starting from an initial guess T_0, close enough to T^* in order to ensure convergence.

All of the groups we will consider hereafter are smooth manifolds with a *Lie group* structure, and the local tangent space at T is a *Lie algebra*, which maps to the whole group through the *exponential mapping*. Therefore, Gauss-Newton optimization is straightforwardly performed with the Lie generators G_i and the compositional update (see e.g. [1][2][3]).

1.2 Algebraic error

In homogeneous coordinates, we look for T^* in G that satifies the equations:

$$T^* \in G : \forall i, \exists \lambda_i : \left(K \cdot \bar{T} \cdot T \cdot \mathbf{X}_i \right) = \lambda_i \mathbf{x}_i$$

As we can see, the homogeneous formulation carry the projective ambiguity as coefficients (λ_i) which account for the augmented number of equations (3 instead of 2) per point. We remove λ_i by writing the problem as a cross-product

$$T^* \in G : \mathbf{x}_i \times (K \cdot \bar{T} \cdot T \cdot \mathbf{X}_i) = 0, \forall i$$

which provides a redundant set of homogeneous equations in T. That means, one component of each cross product (usually the third) can be discarded from the equations above, which then will be $2N$ again.

For noisy data, the problem above can be cast into an LSE form

$$T^* = \min_{T \in G} \sum_{i=1}^{n} \left\| x_i \times (K \cdot \bar{T} \cdot T \cdot X_i) \right\|^2$$

and this problem can be usually put into a linear form $\min_{\mathbf{p}} \|A\mathbf{p}\|$ or $\min_{\mathbf{p}} \|A\mathbf{p} - \mathbf{b}\|$, where $\mathbf{p}$ is a vector parametrizing the transformation T, and A is a column-rank deficient matrix with ∞^1 solutions in $\mathbf{p}$.

By imposing a constraint on $\mathbf{p}$ such as a unit-norm condition $\|\mathbf{p}\| = 1$, the globally optimal solution can be found in one step, via the SVD algorithm. The resulting algorithm is called DLT (Direct Linear Transform).

By resuming, we solve the original problem in two steps:

1. Estimate a T_0 matrix (hopefully close enough to T^*), by minimizing the algebraic error (DLT)

2. Starting from T_0, minimize the geometric error with the Gauss-Newton (or Levenberg-Marquardt) method, in order to obtain T^*. For this purpose, we use compositional updates ΔT and Lie algebra derivatives

In some cases (especially 2D-2D or 3D-3D problems), the geometric error is linear and can be solved at once, without need for T_0. And also some nonlinear cases (e.g. the absolute orientation problem) can still be solved in one step for the geometric error.

However, most 3D-2D cases have an inherent nonlinearity due to the projection $\pi()$, therefore the two-step procedure cannot be generally avoided. In that case, altough step 2 has a common formulation for all poses (apart from different Jacobians, obtained through the respective Lie generators) step 1, instead, must be solved differently for each class.

2 2D-2D transforms: T = (3x3) matrix

In a 2D-2D problem, both $\mathbf{X}$ and $\mathbf{x}$ are given by 3 homogeneous coordinates, of which the third is usually set to 1.

Moreover, we suppose all (3x3) K matrices to have the simple form:

$$K_{2D} = \begin{bmatrix} 1 & 0 & r_x/2 \\ 0 & 1 & r_y/2 \\ 0 & 0 & 1 \end{bmatrix}$$

where (r_x, r_y) are the horizontal and vertical image resolution, respectively. Therefore, all image data points $\mathbf{x}_i$ can be pre-processed in order to remove both K and $\bar{T}$:

$$\bar{\mathbf{x}}_i = \bar{T}^{-1} K^{-1} \mathbf{x}_i = \bar{T}^{-1} \left(\mathbf{x}_i - [r_x/2, r_y/2, 0]^T \right)$$

and the two LSE errors are re-formulated as

- *Algebraic*: $T^* = \min\limits_{T \in G} \sum_{i=1}^{N} \|\bar{\mathbf{x}}_i \times (H \cdot \mathbf{X}_i)\|^2$
- *Geometric*: $T^* = \min\limits_{T \in G} \sum_{i=1}^{N} \|\pi(H \cdot \mathbf{X}_i) - \pi(\bar{\mathbf{x}}_i)\|^2$

NOTE: If the last row of T is $[0, 0, 1]$, then the geometric error

$$T^* = \min_{T \in G} \sum_{i=1}^{N} \|T \cdot \mathbf{X}_i - \bar{\mathbf{x}}_i\|^2$$

has a linear form, and the problem can be directly solved in non-homogeneous coordinates.

2.1 Additional symbols:

$I_n = (n \times n)$ Identity matrix
$\mathbf{t}_n = (n \times 1)$ Translation vector
$R_n = (n \times n)$ Rotation matrix: $R_n^T R_n = I_n$
$R_{x,y,z} = (3 \times 3)$ Single-axis rotation matrices
$s =$ Uniform scale factor
$D_n = diag(s_1, ..., s_n)$: Non-uniform scale matrix
$A_n = (n \times n)$ Linear transformation
$\mathbf{v}_n = (n \times 1)$ Perspective distortion vector

2.2 Pose2DTranslation (2 dof, min. 1 point)

The simplest case is a pure translation

$$T = \begin{bmatrix} I_2 & \mathbf{t}_2 \\ 0 & 1 \end{bmatrix}$$

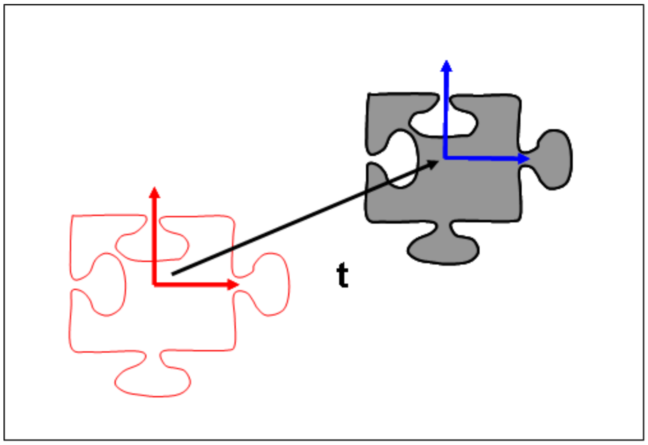

Figure 1: Pure translation.

The geometric error is

$$T^* = \min_{(t_x,t_y)} \sum_{i=1}^{N} \|A_i\mathbf{t} - \mathbf{b}_i\|^2$$

with

$$A_i = I_2$$
$$\mathbf{b}_i = \bar{\mathbf{x}}_i - \mathbf{X}_i$$

that is, $T^* = \min_{\mathbf{t}} \|A\mathbf{t} - \mathbf{b}\|^2$ with $A = \begin{bmatrix} I_2 \\ \cdots \\ I_2 \end{bmatrix}$, $\mathbf{b} = \begin{bmatrix} \mathbf{b}_1 \\ \cdots \\ \mathbf{b}_n \end{bmatrix}$

This is a linear LSE, solved by $\mathbf{t}^* = A^+\mathbf{b}$; in this case, the LSE solution corresponds to the displacement of point centroids:

$$\mathbf{t}^* = \frac{1}{N}\sum_{i=1}^{N}(\bar{\mathbf{x}}_i - \mathbf{X}_i) = \mu_{\bar{\mathbf{x}}} - \mu_{\mathbf{X}}$$

2.3 Pose2D1ScaleTranslation (3 dof, min. 2 point)

Next, we add a uniform scale factor

$$T = \begin{bmatrix} sI_2 & \mathbf{t}_2 \\ 0 & 1 \end{bmatrix}$$

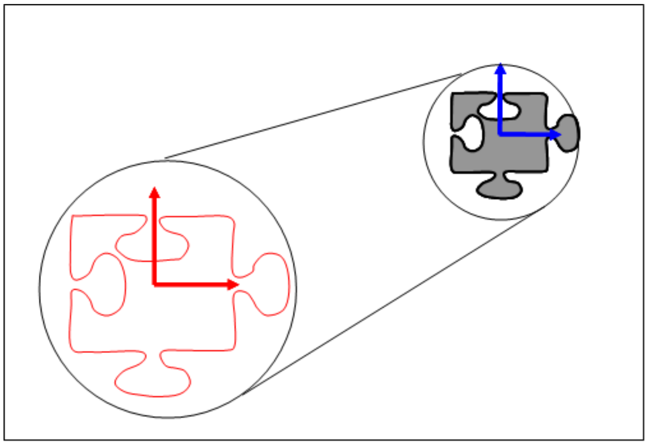

Figure 2: Translation and uniform scale.

The geometric error results

$$\sum_{i=1}^{N}\|p(T\mathbf{X}_i) - p(\bar{\mathbf{x}}_i)\|^2 = \sum_{i=1}^{N}\left\| \begin{array}{c} sX_i + t_x - \bar{x}_i \\ sY_i + t_y - \bar{y}_i \end{array} \right\|^2 = \left\| A \cdot \begin{bmatrix} s \\ t_x \\ t_y \end{bmatrix} - \mathbf{b} \right\|^2$$

with

$$A = \begin{bmatrix} X_1 & 1 & 0 \\ Y_1 & 0 & 1 \\ & \cdots & \\ X_N & 1 & 0 \\ Y_N & 0 & 1 \end{bmatrix}, \mathbf{b} = \begin{bmatrix} \bar{x}_1 \\ \bar{y}_1 \\ \cdots \\ \bar{x}_N \\ \bar{y}_N \end{bmatrix}$$

This is again a linear LSE problem in (s, t_x, t_y).

2.4 Pose2D2ScalesTranslation (4 dof, min. 2 points)

A non-uniform scale with translation is given by

$$T = \begin{bmatrix} D_2 & \mathbf{t}_2 \\ 0 & 1 \end{bmatrix}$$

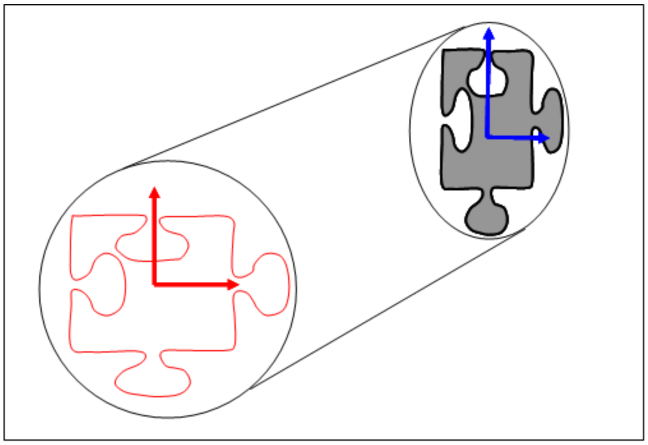

Figure 3: Translation and non-uniform scale.

This is similar to the previous problem, but with 2 scales (s_1, s_2, t_x, t_y); the geometric error minimization gives:

$$T = \min_{(s_1,s_2,t_x,t_y)} \left\| \begin{bmatrix} X_1 & 0 & 1 & 0 \\ 0 & Y_1 & 0 & 1 \\ & \cdots & \cdots & \\ X_N & 0 & 1 & 0 \\ 0 & Y_N & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} s_1 \\ s_2 \\ t_x \\ t_y \end{bmatrix} - \begin{bmatrix} \bar{x}_1 \\ \bar{y}_1 \\ \cdots \\ \bar{x}_N \\ \bar{y}_N \end{bmatrix} \right\|^2$$

2.5 Pose2D1ScaleRotoTranslation (4 dof, min. 2 points)

Now we consider transformations involving rotations.

These are in principle nonlinear problems (because of the rotation matrix R) but fortunately, due to the nature of the problem, the LSE geometric error can still be globally optimized in one step, by using the SVD decomposition.

We start by giving here the solution for the general *similarity* transform (uniform scale, rotation and translation), and deduce its sub-cases afterwards.

$$T = \begin{bmatrix} sR_2 & \mathbf{t}_2 \\ 0 & 1 \end{bmatrix}$$

From the [Umeyama] paper: the LSE geometric error is optimized by the $(s, R, \mathbf{t})$ parameters obatined from the following steps:

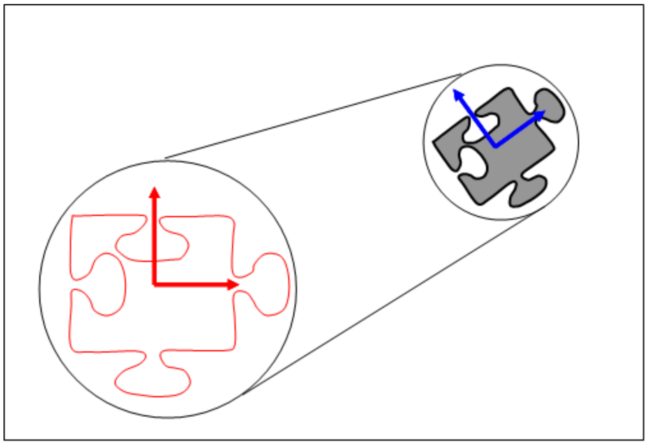

Figure 4: Similarity (rigid roto-translation and uniform scale).

Mean vectors: $\mu_{\bar{\mathbf{x}}} = \frac{1}{n}\sum_{i=1}^{n} \bar{\mathbf{x}}_i, \mu_{\mathbf{X}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{X}_i$

Variance of the norms: $\sigma^2_{\bar{\mathbf{x}}} = \frac{1}{n}\sum_{i=1}^{n} \|\bar{\mathbf{x}}_i - \mu_{\bar{\mathbf{x}}}\|^2, \sigma^2_{\mathbf{X}} = \frac{1}{n}\sum_{i=1}^{n} \|\mathbf{X}_i - \mu_{\mathbf{X}}\|^2$

Cross-covariance matrix (2×2): $\Sigma_{\bar{\mathbf{x}}\mathbf{X}} = \frac{1}{n}\sum_{i=1}^{n}(\bar{\mathbf{x}}_i - \mu_{\bar{\mathbf{x}}})(\mathbf{X}_i - \mu_{\mathbf{X}})^T$

SVD of the cross-covariance: $\Sigma_{\bar{\mathbf{x}}\mathbf{X}} = UDV^T$

Sign correction for $det(R)$: $S = \begin{cases} I & \text{if } \det(\Sigma_{\bar{\mathbf{x}}\mathbf{X}}) \geq 0 \\ diag(1,1,1,...,-1) & \text{if } \det(\Sigma_{\bar{\mathbf{x}}\mathbf{X}}) < 0 \end{cases}$

Rotation reconstruction: $R = USV^T$

Scale reconstruction: $s = \frac{1}{\sigma^2_{\mathbf{X}}} tr(DS)$

Translation vector: $\mathbf{t} = \mu_{\bar{\mathbf{x}}} - sR\mu_{\mathbf{X}}$

2.6 Pose2D1ScaleRotation (2 dof, min. 1 point)

Similar as before, but without translation:

$$T = \begin{bmatrix} sR_2 & 0 \\ 0 & 1 \end{bmatrix}$$

This implies that the mean vectors are $\mu_{\bar{\mathbf{x}}} = \mu_{\mathbf{X}} = 0$. Therefore we have:

Variance of the norms: $\sigma^2_{\bar{\mathbf{x}}} = \frac{1}{n}\sum_{i=1}^{n} \|\bar{\mathbf{x}}_i\|^2, \sigma^2_{\mathbf{X}} = \frac{1}{n}\sum_{i=1}^{n} \|\mathbf{X}_i\|^2$

Cross-covariance matrix (2×2): $\Sigma_{\bar{\mathbf{x}}\mathbf{X}} = \frac{1}{n}\sum_{i=1}^{n} \bar{\mathbf{x}}_i\mathbf{X}_i^T$

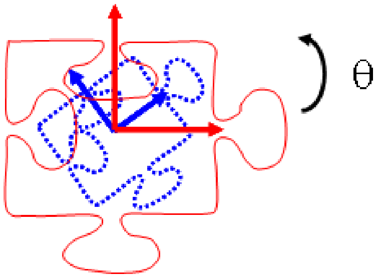

Figure 5: Rotation and uniform scale.

SVD of the cross-covariance: $\Sigma_{\bar{\mathbf{x}}\mathbf{X}} = UDV^T$

Sign correction for $det(R)$: $S = \begin{cases} I & \text{if } \det(\Sigma_{\bar{\mathbf{x}}\mathbf{X}}) \geq 0 \\ diag(1,1,1,...,-1) & \text{if } \det(\Sigma_{\bar{\mathbf{x}}\mathbf{X}}) < 0 \end{cases}$

Rotation reconstruction: $R = USV^T$

Scale reconstruction: $s = \frac{1}{\sigma_{\bar{\mathbf{x}}}^2} tr(DS)$

2.7 Pose2DRotoTranslation (3 dof, min. 2 points)

By removing the scale ($s = 1$), we get the Euclidean transform (rigid roto-translation):

$$H = \begin{bmatrix} R_2 & t_2 \\ 0 & 1 \end{bmatrix}$$

where $\sigma_{\bar{\mathbf{x}}} = \sigma_{\mathbf{X}} = 1$. The algorithm becomes

Mean vectors: $\mu_{\bar{\mathbf{x}}} = \frac{1}{n}\sum_{i=1}^{n} \bar{\mathbf{x}}_i, \mu_{\mathbf{X}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{X}_i$

Cross-covariance matrix (2×2): $\Sigma_{\bar{\mathbf{x}}\mathbf{X}} = \frac{1}{n}\sum_{i=1}^{n}(\bar{\mathbf{x}}_i - \mu_{\bar{\mathbf{x}}})(\mathbf{X}_i - \mu_{\mathbf{X}})^T$

SVD of the cross-covariance: $\Sigma_{\bar{\mathbf{x}}\mathbf{X}} = UDV^T$

Sign correction for $det(R)$: $S = \begin{cases} I & \text{if } \det(\Sigma_{\bar{\mathbf{x}}\mathbf{X}}) \geq 0 \\ diag(1,1,1,...,-1) & \text{if } \det(\Sigma_{\bar{\mathbf{x}}\mathbf{X}}) < 0 \end{cases}$

Rotation reconstruction: $R = USV^T$

Translation vector: $t = \mu_{\bar{\mathbf{x}}} - R\mu_{\mathbf{X}}$

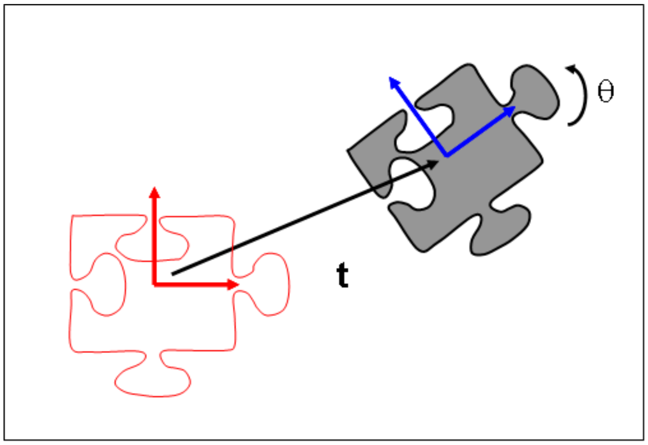

Figure 6: Euclidean transform (rigid roto-translation).

2.8 Pose2DRotation (1 dof, min. 1 point)

Finally, if both scale and translation are removed, we obtain the *absolute orientation* problem:

$$T = \begin{bmatrix} R_2 & 0 \\ 0 & 1 \end{bmatrix}$$

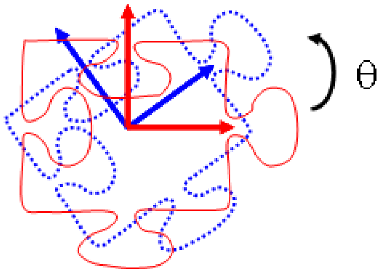

Figure 7: Pure rotation.

which is solved by

Cross-covariance matrix (2×2): $\Sigma_{\bar{\mathbf{x}}\mathbf{X}} = \frac{1}{n}\sum_{i=1}^{n} \bar{\mathbf{x}}_i \mathbf{X}_i^T$

SVD of the cross-covariance: $\Sigma_{\bar{\mathbf{x}}\mathbf{X}} = UDV^T$

Sign correction for $det(R)$: $S = \begin{cases} I & \text{if } \det(\Sigma_{\bar{\mathbf{x}}\mathbf{X}}) \geq 0 \\ diag(1,1,1,...,-1) & \text{if } \det(\Sigma_{\bar{\mathbf{x}}\mathbf{X}}) < 0 \end{cases}$

Rotation reconstruction: $R = USV^T$

2.9 Pose2D2ScalesRotoTranslation (5 dof, min. 3 points)

The problems with non-uniform scale and rotation cannot be solved like the uniform scale cases. Since the number of degrees of freedom is close to that of an affinity (6 vs. 5 dof), we prefer to solve first for an affinity, and then upgrade to the non-uniform scale similarity, by removing 1 degree of freedom.

We consider here two sub-cases: with and without translation. The first one is given by

$$T = \begin{bmatrix} R_2 D_2 & t_2 \\ 0 & 1 \end{bmatrix}$$

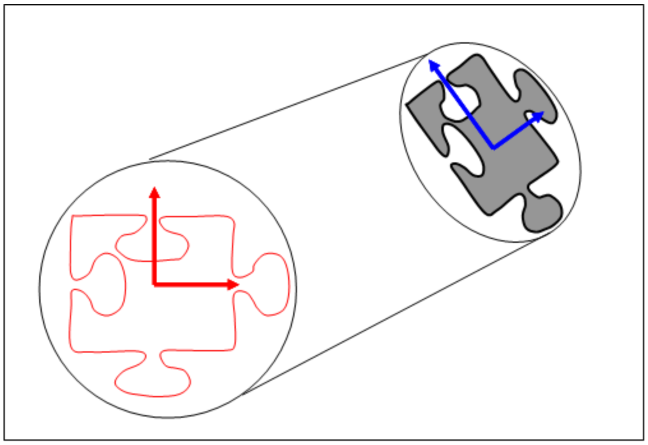

Figure 8: Roto-translation with non-uniform scale.

and the complete procedure is the following:

1. solve for an affine transform (see below), and find $A, \mathbf{t}$

2. using the SVD, compute $A = R(\theta)R(-\phi)DR(\phi)$ with $D = diag(s_1, s_2)$, and $s_1 > s_2$

3. set $R = R(\theta)$ and remove ϕ: if $\phi \approx 0$, then keep the order of (s_1, s_2) in D; if $\phi \approx 90\,\text{deg}$, then swap the scales

4. Finally, since this solution is generally *not* the optimal LSE, it is recommended to run a Gauss-Newton optimization in the geometric error.

2.10 Pose2D2ScalesRotation (3 dof, min. 2 point)

Next, we consider the rotation with non-uniform scale:

$$H = \begin{bmatrix} R_2 D_2 & 0 \\ 0 & 1 \end{bmatrix}$$

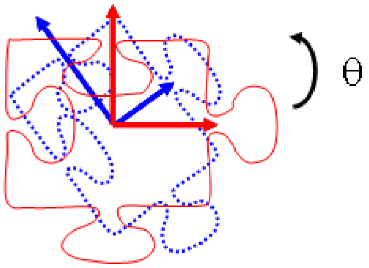

Figure 9: Rotation with non-uniform scale.

This is a special case of the previous one, where $\mathbf{t} = 0$. We can solve it as a purely linear (affine without translation) transform with 4 dof, then upgrading A to the non-uniform scale and rotation matrix as in the previous Section.

2.11 Pose2DAffine (6 or 4 dof, respectively 3 or 2 points)

$$T = \begin{bmatrix} A_2 & t_2 \\ 0 & 1 \end{bmatrix}$$

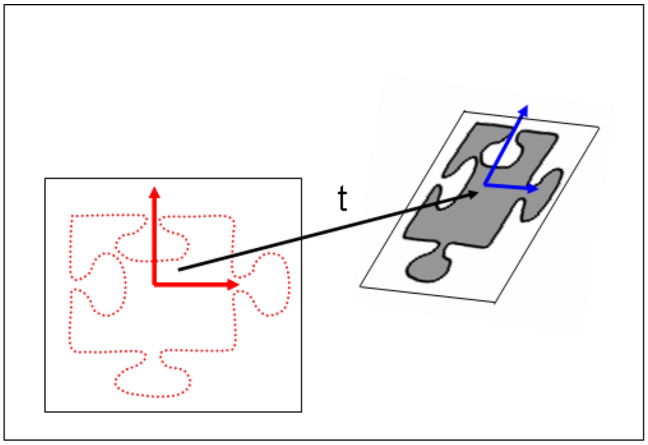

Figure 10: Affine transform (linear+constant).

The affine case is again a linear LSE problem in the geometric error

$$[A^*, t^*] = \min_{(A,t)} \sum_{i=1}^{N} \|A\mathbf{X}_i + t - \bar{\mathbf{x}}_i\|^2 = \min \|M_i\mathbf{a} - \bar{\mathbf{x}}_i\|^2$$

with $\mathbf{a}$ the 6 stacked parameters

$$\mathbf{a} = \left[\begin{array}{cccccc} A_{11} & A_{12} & A_{21} & A_{22} & t_x & t_y \end{array} \right]^T$$

and M_i a coefficient matrix function of $\mathbf{X}_i$

$$M_i = \left[\begin{array}{cccccc} X_i & Y_i & 0 & 0 & 1 & 0 \\ 0 & 0 & X_i & Y_i & 0 & 1 \end{array} \right]$$

By stacking together the M_i matrices and the $\bar{\mathbf{x}}_i$ vectors, and solving for $\mathbf{a}^*$, we get the LSE affine parameters.

Similar equations can be written for the purely linear case ($\mathbf{t} = 0$)

$$T = \left[\begin{array}{cc} A_2 & 0 \\ 0 & 1 \end{array} \right]$$

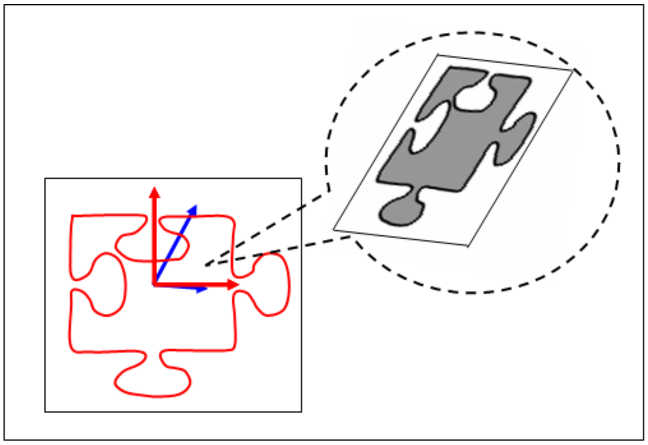

Figure 11: Purely linear transform.

with 4 parameters only

$$\mathbf{a} = \left[\begin{array}{cccc} A_{11} & A_{12} & A_{21} & A_{22} \end{array} \right]^T$$

$$M_i = \left[\begin{array}{cccc} X_i & Y_i & 0 & 0 \\ 0 & 0 & X_i & Y_i \end{array} \right]$$

2.12 Pose2DHomography (8 dof, min. 4 points): the DLT algorithm

In the most general 2D-2D case, the matrix T can be any linear transform in the homogeneous coordinates. This is defined up to a scale factor, that we remove by setting $T(3,3)=1$:

$$T = \begin{bmatrix} A_2 & \mathbf{t}_2 \\ \mathbf{v}_2^T & 1 \end{bmatrix}$$

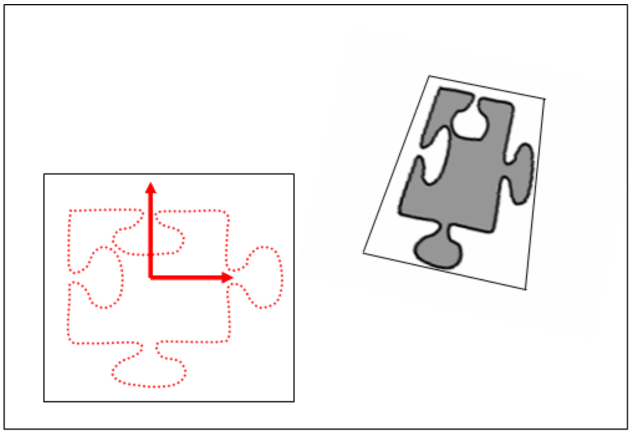

Figure 12: General 2D homography.

By applying the projection operator p, this leads to a non-linear geometric error; therefore, the pose estimation problem must be formulated in two steps as described in the introduction (algebraic and geometric error minimization).

Concerning the first, after pre-processing (removing K and $\bar{T}$ from the data points $\mathbf{x}_i$) we have an algebraic error of the form

$$T = \min_T \sum_{i=1}^{N} \|\bar{\mathbf{x}}_i \times (T\mathbf{X}_i)\|^2$$

with $\mathbf{X}_i, \bar{\mathbf{x}}_i \in \Re^3, T \in \Re^{3\times 3}$. By writing $\mathbf{X}_i = (X_i, Y_i, Z_i), \bar{\mathbf{x}}_i = (\bar{x}_i, \bar{y}_i, \bar{z}_i)$ we take the first two terms of the cross product, and we have, for each point i, two homogeneous equations

$$\begin{bmatrix} 0^T & -\bar{z}_i X_i^T & \bar{y}_i X_i^T \\ \bar{z}_i X_i^T & 0^T & -\bar{x}_i X_i^T \end{bmatrix} \begin{pmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \\ \mathbf{h}_3 \end{pmatrix} = 0$$

where $\mathbf{h}_i$ are the three (transposed) rows of $H = \begin{bmatrix} \mathbf{h}_1^T \\ \mathbf{h}_2^T \\ \mathbf{h}_3^T \end{bmatrix}$.

If A is the $(2n \times 9)$ stacked matrix of all l.h.s. terms above, we get the DLT equation $\min\limits_{T} \|A\mathbf{h}\|$ with rank-deficient A that, as we expect, has ∞^1 solutions in $\mathbf{h}$. By imposing $\|\mathbf{h}\| = 1$, the solution is obtained by the SVD decomposition

$$A = USV^T$$

as the last column of V (corresponding to the minimum singular value in S).

In addition, the normalization technique (removing the centroids of $\mathbf{X}$ and $\mathbf{x}$, followed by isotropic coordinate scaling) ensures a better numerical stability, and therefore we use it.

1. Removing centroids: For both point sets, we compute the mean values $\mu_{\bar{\mathbf{x}}} = \frac{1}{n} \sum_{i=1}^{n} \bar{\mathbf{x}}_i, \mu_{\mathbf{X}} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{X}_i$ and remove them from each point
2. Isotropic scaling: Afterwards, we make sure that the average distance from the mass center (which is 0) is equal to $\sqrt{2}$, i.e. that the "average" point of both sets is $(1, 1, 1)^T$

These two steps ultimately correspond to multiply the two point sets for two matrices $T_{\mathbf{X}}, T_{\bar{\mathbf{x}}}$. Therefore, after estimating the transformation $\tilde{T}^*$ the two normalizations are removed by

$$T^* = T_{\bar{\mathbf{x}}}^{-1} \tilde{T}^* T_{\mathbf{X}}$$

3 3D-2D transforms: T = (4x4), K = (3x4)

Here, the calibration matrix K can be more general; in particular, we consider *pinhole* models, without distortion and skew, and with equal focal lengths:

$$K = \left[\begin{array}{cc} K' & 0 \end{array}\right], K' = \left[\begin{array}{ccc} f & 0 & r_x/2 \\ 0 & f & r_y/2 \\ 0 & 0 & 1 \end{array}\right]$$

Moreover, we have extrinsic transformation matrices the type

$$T = \left[\begin{array}{cc} A_{3\times 3} & \mathbf{t}_3 \\ \mathbf{0}^T & 1 \end{array}\right] \in G$$

$$\bar{T} = \left[\begin{array}{cc} \bar{A}_{3\times 3} & \bar{\mathbf{t}}_3 \\ \mathbf{0}^T & 1 \end{array}\right] \in G'$$

belonging to possibly different groups G, G'. The base estimation problem becomes: find $(A^*, \mathbf{t}^*)$ such that

$$\left[\begin{array}{cc} A^* & \mathbf{t}^* \end{array}\right] \in G : \forall i, \exists \lambda_i : \left(K' \left[\begin{array}{cc} \bar{A}A & \bar{A}\mathbf{t} + \bar{\mathbf{t}} \end{array}\right] \mathbf{X}_i\right) = \lambda_i \mathbf{x}_i$$

We can pre-process the image data $\mathbf{x}_i$

$$\bar{\mathbf{x}}_i = (K')^{-1} \mathbf{x}_i$$

and re-write the equations

$$\left[\begin{array}{cc} \bar{A}A & \bar{A}\mathbf{t} + \bar{\mathbf{t}} \end{array}\right] \mathbf{X}_i = \lambda_i \bar{\mathbf{x}}_i$$

However, unlike the 2D-2D case, in general we cannot remove neither $\bar{A}$ nor $\bar{\mathbf{t}}$, because of the non-square matrix on the left side[1].

Moreover, as already mentioned in the introduction, because of the dimensionality loss (from 3D to 2D) the projection operator $\pi()$ always provides a nonlinearity, no matter of the form for $A, \mathbf{t}$. Therefore, the linear approach for the algebraic error (DLT) cannot be avoided, in order to provide an initial guess T_0 for the geometric error optimization.

Finally, we need the DLT approach for all transform classes, which may have much less than the maximum number of parameters (12); therefore, we assume that each $T \in G$ can be parametrized (or at least approximated) by a vector $\mathbf{q}$

$$\left[\begin{array}{cc} A(\mathbf{q}) & \mathbf{t}(\mathbf{q}) \end{array}\right]$$

with $dim(\mathbf{q}) = d_q \leq 12$. In what follows, we will call this method *generalized, projective DLT* (GP-DLT).

[1] The only exception is given by $\bar{\mathbf{t}} = \mathbf{0}$, in which case $\bar{A}$ can also be removed from the data points $\bar{\mathbf{x}}_i = \left(K'\bar{A}\right)^{-1} \mathbf{x}_i$ and the problem becomes a purely projective one $\left[\begin{array}{cc} A & \mathbf{t} \end{array}\right] \mathbf{X}_i = P\mathbf{X}_i = \lambda_i \bar{\mathbf{x}}_i$

3.1 Algebraic error for 3D-2D projections: the GP-DLT approach

We re-write the algebraic error in terms of the reduced parameters $\mathbf{q}$ as follows:

$$\mathbf{q}^* = \min_{\mathbf{q}\in\Re^{d_q}} \sum_{i=1}^{n} \left\| \mathbf{x}_i \times \left(\left[\begin{array}{cc} \bar{A}A(\mathbf{q}) & \bar{A}\mathbf{t}(\mathbf{q}) + \bar{\mathbf{t}} \end{array} \right] \cdot \mathbf{X}_i \right) \right\|^2$$

that can be written as

$$\mathbf{q}^* = \min_{\mathbf{q}\in\Re^{d_q}} \sum_{i=1}^{n} \left\| F_i(\mathbf{q}) + \mathbf{f}_i \right\|^2$$

with

$$F_i(\mathbf{q}) = [\mathbf{x}_i]_\times \bar{A} \left[\begin{array}{cc} A(\mathbf{q}) & \mathbf{t}(\mathbf{q}) \end{array} \right] \mathbf{X}_i$$

$$\mathbf{f}_i = [\mathbf{x}_i]_\times \left[\begin{array}{cc} 0_{3\times 3} & \bar{\mathbf{t}} \end{array} \right] \mathbf{X}_i$$

and

$$[\mathbf{x}_i]_\times = \left[\begin{array}{ccc} 0 & -z_i & y_i \\ z_i & 0 & -x_i \\ -y_i & x_i & 0 \end{array} \right]$$

the cross-product matrix.

If we further impose $A(\mathbf{q})$ and $\mathbf{t}(\mathbf{q})$ to be linear in $(\mathbf{q})$, then we can show that

$$F_i(\mathbf{q}) = F_i \cdot \mathbf{q}$$

and the algebraic error becomes a linear LSE problem. This condition seems to be quite restrictive, since most parametrization for transformation groups are nonlinear (particularly if a rotation matrix is involved); nevertheless, we can always find a parametrization $\mathbf{q}_l$ in a linear group G_l that includes G, where $dim(G_l)$ is higher than $dim(G)$, but as close as possible to it.

Afterwards, the so obtained transform T_l can be upgraded to the actual T by a Procrustes analysis ($\min_{T\in G} \|T_l - T\|_F$) or by simpler means, such as *clamping* the affine parameters ϕ for upgrading to a similarity, etc. (similarly to the 2D-2D cases). In any case, the result of this procedure is needed only as a starting point for the geometric error optimization.

3.2 Example of linear constraints for 3D pose parameters

We mention here a few examples of linear constraints that we can impose to $\mathbf{p}$.

- Pure translation in 3D

$$P = \left[\begin{array}{cc} I_3 & \mathbf{t}_3 \end{array} \right]$$

This corresponds to impose

$$Q = \left[\begin{array}{c} 0_{3\times 3} \\ 0_{3\times 3} \\ 0_{3\times 3} \\ I_{3\times 3} \end{array} \right]$$

$$\mathbf{p}_0 = \left[\begin{array}{cccccccccccc} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{array} \right]^T$$

$$\mathbf{q} = \left[\begin{array}{ccc} t_x & t_y & t_z \end{array} \right]^T$$

- "Almost" pure rotation around z (without the non-linear constraint $c^2 + s^2 = 1$)

  $$P = \left[\begin{array}{cccc} c & -s & 0 & 0 \\ s & c & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array} \right]$$

  This corresponds to

  $$Q = \left[\begin{array}{cc} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & -1 \\ 1 & 0 \\ 0 & 0 \\ \cdots & \cdots \\ 0 & 0 \end{array} \right]$$

  $$\mathbf{p}_0 = \left[\begin{array}{cccccccccccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{array} \right]^T$$

  $$\mathbf{q} = \left[\begin{array}{cc} c & s \end{array} \right]^T$$

4 Line correspondences

In some cases, the image measurement consists of line segments, that have to be matched to corresponding model segments.

For example, when performing a hand detection task (Fig. 13), the Hough transform provides very well-aligned line segments on the fingers, associated to the corresponding model lines.

Segment correspondences in principle provide 2 point correspondences (i.e. 4 measurement data). However, as we can see from the fingers in Fig. 13, the end-points of the detected segments are not as well localized as the *line* itself (direction and distance from the origin), therefore the most reliable matching can be obtained by considering pure *line* correspondences.

This unfortunately provides less equations (2 instead of 4) for each feature, but at least assures to use only the most reliable information source for pose estimation[2].

As long as pure lines are concerned, a simpler way to describe correspondences consists of replacing them by 2 *point-to-line* correspondences (i.e. a segment-to-line correspondence), where the two model points can be arbitrarily chosen onto the respective line (in 2D or 3D space).

[2] Alternatively, the end-points information can be still included, but with a lower *weight* in the LSE optimization process

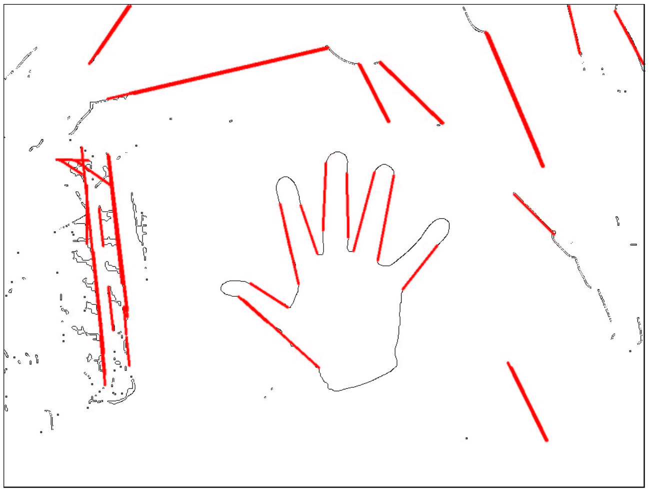

Figure 13: Line detection with the Hough transform for planar hand detection.

Therefore, we formulate the problem as follows: given a model *segment* $(\mathbf{L}^1, \mathbf{L}^2)$ and a corresponding image *line* $\mathbf{l} = (l_1, l_2, l_3)^T$, find a transformation H such that both points $H\mathbf{L}^1$ and $H\mathbf{L}^2$ lie on l.

When a noisy measurement $\mathbf{l}$ is given, the error can be again formulated in two ways (geometric and algebraic error) which lead to different LSE errors. In order to provide the geometric error, we also assume that the normal direction to the image line $\mathbf{n} = (l_1, l_2)$ is normalized $\|\mathbf{n}\| = 1$; in this way, the third component $l_3 = d$ represents the distance of the line to the origin of image coordinates.

In particular, algebraic errors are defined in homogeneous coordinates, and geometric errors in projected (non-homogeneous) coordinates, through the $\pi()$ function.

- Algebraic errors:

$$\mathbf{l}^T \left(K\bar{T}T \cdot \mathbf{L}^1 \right) = \mathbf{l}^T \left(K\bar{T}T \cdot \mathbf{L}^2 \right) = 0$$

- Geometric errors:

$$\mathbf{n}^T \pi \left(K\bar{T}T \cdot \mathbf{L}^1 \right) + d = \mathbf{n}^T \pi \left(K\bar{T}T \cdot \mathbf{L}^2 \right) + d = 0$$

which, in a least-squares setting, become

- Algebraic LSE:

$$T^* = \min_{T \in G} \sum_{i=1}^{n} \left(\left\| \mathbf{l}_i^T \cdot K\bar{T}T \cdot \mathbf{L}_i^1 \right\|^2 + \left\| \mathbf{l}_i^T \cdot K\bar{T}T \cdot \mathbf{L}_i^2 \right\|^2 \right)$$

- Geometric LSE:

$$T^* = \min_{T \in G} \sum_{i=1}^{n} \left(\left\| \mathbf{n}_i^T \pi \left(K\bar{T}T \cdot \mathbf{L}_i^1 \right) + d_i \right\|^2 + \left\| \mathbf{n}_i^T \pi \left(K\bar{T}T \cdot \mathbf{L}_i^2 \right) + d_i \right\|^2 \right)$$

4.1 2D-2D line correspondences

For most 2D cases (apart from the general homography of Sec. 2.12), the geometric LSE is equivalent to the algebraic one. In fact, if the transformation matrix has the form

$$T = \begin{bmatrix} A & \mathbf{t} \\ 0 & 1 \end{bmatrix}$$

then we have

$$\mathbf{n}^T \pi \left(K\bar{T}T \cdot \mathbf{L} \right) + d = \mathbf{l}^T \left(K\bar{T}T \cdot \mathbf{L} \right) = \mathbf{l}^T K\bar{T} \begin{pmatrix} A\tilde{\mathbf{L}} + \mathbf{t} \\ 1 \end{pmatrix}$$

where $\tilde{\mathbf{L}}$ are the first two (non-homogeneous) coordinates of $\mathbf{L}$. For sake of clarity, in the following we will omit the $\sim$ sign, whenever the context avoids ambiguity of interpretation.

The above equations clearly show how the two terms $K\bar{T}$ can be removed by pre-processing the lines $\mathbf{l}$

$$\bar{\mathbf{l}}^T = \mathbf{l}^T K\bar{T} = (\bar{\mathbf{n}}^T, \bar{d})$$

Furthermore, if the parametrization of the group $A(\mathbf{q}), \mathbf{t}(\mathbf{q})$ is linear in $\mathbf{q}$, then the problem becomes linear, and can be solved in one step via the SVD decomposition.

For example, if we consider the general affine transform (Sec. 2.11), parametrized by

$$\mathbf{q} = \begin{bmatrix} A_{11} & A_{12} & A_{21} & A_{22} & t_x & t_y \end{bmatrix}^T$$

then we have

$$A\mathbf{L} + \mathbf{t} = \begin{bmatrix} L_x & L_y & 0 & 0 & 1 & 0 \\ 0 & 0 & L_x & L_y & 0 & 1 \end{bmatrix} \mathbf{q} = \hat{\mathbf{L}}\mathbf{q}$$

so that the LSE problem becomes

$$\mathbf{q}^* = \min_{\mathbf{q} \in \Re^6} \sum_{i=1}^{n} \left(\left\| \bar{\mathbf{n}}_i^T \hat{\mathbf{L}}_i^1 \cdot \mathbf{q} + \bar{d}_i \right\|^2 + \left\| \bar{\mathbf{n}}_i^T \hat{\mathbf{L}}_i^2 \cdot \mathbf{q} + \bar{d}_i \right\|^2 \right) = \min_{\mathbf{q} \in \Re^6} \sum_{i=1}^{n} \left\| \hat{L}_i \cdot \mathbf{q} + \bar{\mathbf{d}}_i \right\|^2$$

with

$$\hat{L}_i = \begin{bmatrix} \bar{\mathbf{n}}_i^T \hat{\mathbf{L}}_i^1 \\ \bar{\mathbf{n}}_i^T \hat{\mathbf{L}}_i^2 \end{bmatrix}; \ \bar{\mathbf{d}}_i = \begin{bmatrix} \bar{d}_i \\ \bar{d}_i \end{bmatrix}$$

A similar result can be obtained for the similarity case (uniform scale, rotation and translation) with 4 parameters. In order to keep the linearity, we parametrize it as

$$T = \begin{bmatrix} c & -s & t_x \\ s & c & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

with

$$\mathbf{q} = \begin{bmatrix} c & s & t_x & t_y \end{bmatrix}^T$$

so that in this case

$$\hat{\mathbf{L}} = \begin{bmatrix} L_x & -L_y & 1 & 0 \\ L_y & L_x & 0 & 1 \end{bmatrix}$$

and $\hat{L}_i$ is computed from this expression.

However, the pure rotational cases (with $c = \cos(\theta), s = \sin(\theta)$) involve a nonlinearity that, in the point-to-point case, had been dealt with by using the Umeyama approach. In this case, we can simply estimate it as a similarity, and afterwards remove the scale by simply dividing (c, s) by $\sqrt{c^2 + s^2}$.

5 Point and line correspondences

The most general case involves matching points and lines simultaneously. In order to formulate it in an elegant way, we start from the result of the previous Section, and add the point-related terms.

Concerning the geometric error term for a linear pose parametrization $\mathbf{q}$, we have

$$K\bar{T}T(\mathbf{q})\,\mathbf{X} - \bar{\mathbf{x}} = K\bar{T}\hat{\mathbf{X}} \cdot \mathbf{q} - \bar{\mathbf{x}}$$

where the $\hat{\mathbf{X}}$ matrix is defined in the same way as $\hat{\mathbf{L}}$ (for segments). The pre-processing step for points (already been described in the related Section) becomes

$$\bar{\mathbf{x}} = \left(K\bar{T}\right)^{-1}\mathbf{x}$$

that we can see as the “dual” version of the line pre-processing.

Therefore, for n_l line and n_p point correspondences, we have[3]

$$\mathbf{q}^* = \min_{\mathbf{q}\in\Re^6}\left(\sum_{i=1}^{n_l}\left\|\hat{L}_i \cdot \mathbf{q} + \bar{\mathbf{d}}_i\right\|^2 + \sum_{j=1}^{n_p}\left\|\hat{\mathbf{X}}_j \cdot \mathbf{q} - \bar{\mathbf{x}}_j\right\|^2\right)$$

with $\hat{L}_i, \bar{\mathbf{d}}_i$ defined in the previous Section.

[3] Notice the − sign on the second terms.

A Derivation of the GP-DLT linear equations

In order to derive the linear LSE matrix F_i, we first consider the internal product

$$\bar{A}\left[\begin{array}{cc} A\left(\mathbf{q}\right) & \mathbf{t}\left(\mathbf{q}\right) \end{array}\right]\mathbf{X}_i = W\mathbf{X}_i$$

with $W = \bar{A}\left[\begin{array}{cc} A\left(\mathbf{q}\right) & \mathbf{t}\left(\mathbf{q}\right) \end{array}\right]$ a (3×4) matrix. We express it row-wise

$$W = \left[\begin{array}{c} \mathbf{w}_1^T \\ \mathbf{w}_2^T \\ \mathbf{w}_3^T \end{array}\right]$$

where $\mathbf{w}_j$ is the $j - th$ row (transposed to a column vector). Therefore, we can write it as a matrix-vector product

$$W\mathbf{X}_i = \left[\begin{array}{ccc} \mathbf{X}_i^T & \mathbf{0}^T & \mathbf{0}^T \\ \mathbf{0}^T & \mathbf{X}_i^T & \mathbf{0}^T \\ \mathbf{0}^T & \mathbf{0}^T & \mathbf{X}_i^T \end{array}\right]\left[\begin{array}{c} \mathbf{w}_1 \\ \mathbf{w}_2 \\ \mathbf{w}_3 \end{array}\right]$$

which, after including the cross-product matrix $[\mathbf{x}_i]_\times$, becomes

$$Fi = \hat{X}_i\left[\begin{array}{c} \mathbf{w}_1 \\ \mathbf{w}_2 \\ \mathbf{w}_3 \end{array}\right]$$

and

$$\hat{X}_i = \left[\begin{array}{ccc} \mathbf{0}^T & -\bar{z}_i\mathbf{X}_i^T & \bar{y}_i\mathbf{X}_i^T \\ \bar{z}_i\mathbf{X}_i^T & \mathbf{0}^T & -\bar{x}_i\mathbf{X}_i^T \\ -\bar{y}_i\mathbf{X}_i^T & \bar{x}_i\mathbf{X}_i^T & \mathbf{0}^T \end{array}\right]$$

Next, we consider again the W matrix

$$\begin{array}{l} W = \bar{A}P\left(\mathbf{q}\right) \\ P\left(\mathbf{q}\right) = \left[\begin{array}{cc} A\left(\mathbf{q}\right) & \mathbf{t}\left(\mathbf{q}\right) \end{array}\right] \end{array}$$

each element of the product is given by

$$W_{hk} = \bar{\mathbf{a}}_h^T\mathbf{p}_k$$

$$\bar{A} = \left[\begin{array}{c} \bar{\mathbf{a}}_1^T \\ \bar{\mathbf{a}}_2^T \\ \bar{\mathbf{a}}_3^T \end{array}\right], P = \left[\begin{array}{cccc} \mathbf{p}_1 & \mathbf{p}_2 & \mathbf{p}_3 & \mathbf{p}_4 \end{array}\right]$$

where $\bar{A}$ has been expressed row-wise, and P column-wise. Therefore, we have

$$\mathbf{w}_h = \left[\begin{array}{cccc} \bar{\mathbf{a}}_h^T & \mathbf{0}^T & \mathbf{0}^T & \mathbf{0}^T \\ \mathbf{0}^T & \bar{\mathbf{a}}_h^T & \mathbf{0}^T & \mathbf{0}^T \\ \mathbf{0}^T & \mathbf{0}^T & \bar{\mathbf{a}}_h^T & \mathbf{0}^T \\ \mathbf{0}^T & \mathbf{0}^T & \mathbf{0}^T & \bar{\mathbf{a}}_h^T \end{array}\right]\mathbf{p}\left(\mathbf{q}\right)$$

where the vector $\mathbf{p}$ contains the 12 entries of P (column-wise), which are parametrized by $\mathbf{q}$.

By stacking the matrices M_h row-wise, we have

$$W = \hat{A}\mathbf{p}\left(\mathbf{q}\right)$$

where

$$\hat{A} = \begin{bmatrix} \bar{\mathbf{a}}_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \bar{\mathbf{a}}_2 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \bar{\mathbf{a}}_3 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \bar{\mathbf{a}}_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \bar{\mathbf{a}}_2 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \bar{\mathbf{a}}_3 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \bar{\mathbf{a}}_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \bar{\mathbf{a}}_2 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \bar{\mathbf{a}}_3 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \bar{\mathbf{a}}_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \bar{\mathbf{a}}_2 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \bar{\mathbf{a}}_3 \end{bmatrix}^T$$

Our assumption about the linear dependency on $\mathbf{q}$ can be expressed by

$$\mathbf{p}\left(\mathbf{q}\right) = Q \cdot \mathbf{q} + \mathbf{p}_0$$

with suitable values for the $12 \times d_q$ matrix Q and the vector $\mathbf{p}_0$.

Therefore, in the linear case the l.h.s. becomes

$$\left[\mathbf{x}_i\right]_\times \bar{A}P\left(\mathbf{q}\right)\mathbf{X}_i = \hat{X}_i\hat{A}\left(Q \cdot \mathbf{q} + \mathbf{p}_0\right)$$

where last term of F_i does not depend on $\mathbf{q}$, and therefore can be moved to the right-hand side $\mathbf{f}_i$ of the LSE. The r.h.s. term can be similarly developed:

$$\left[\mathbf{x}_i\right]_\times \begin{bmatrix} 0_{3\times 3} & \bar{\mathbf{t}} \end{bmatrix} \mathbf{X}_i = \hat{X}_i\mathbf{m}$$

with

$$\mathbf{m} = \begin{bmatrix} 0 & 0 & 0 & \bar{t}_1 & 0 & 0 & 0 & \bar{t}_2 & 0 & 0 & 0 & \bar{t}_3 \end{bmatrix}^T$$

so that, finally, the LSE problem becomes

$$\mathbf{q}^* = \min_{\mathbf{q}\in\Re^{d_q}} \sum_{i=1}^{n} \left\|F_i \cdot \mathbf{q} + \mathbf{f}_i\right\|^2$$

where

$$F_i = \hat{X}_i\hat{A}Q$$

$$\mathbf{f}_i = \hat{X}_i\left(\mathbf{m} - \hat{A}\mathbf{p}_0\right)$$

References

[1] T. Drummond and R. Cipolla, "Visual tracking and control using lie algebras," *cvpr*, vol. 02, p. 2652, 1999.

[2] ——, "Real-time visual tracking of complex structures," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 932–946, 2002.

[3] ——, "Real-time tracking of multiple articulated structures in multiple views," in *ECCV '00: Proceedings of the 6th European Conference on Computer Vision-Part II.* London, UK: Springer-Verlag, 2000, pp. 20–36.