Symmetric Least Squares Matching — SYM-LSM

Wolfgang Förstner

July 19, 2020

Contents

1 Summary

The report describes the basics of symmetric least squares matching (SYM-LSM) which is useful for high precision image matching and realized in MATLAB.

The principle of SYM-LSM is to minimize the weighted sum of the squares of the residuals of the intensity of two images g and h and this way obtains statistically optimal estimates for the parametrized geometric and radiometric distortions between two overlapping images. The estimated geometric transformation may be used in the context of relative image orientation for refining the coordinates of corresponding keypoints or for tracking keypoints within a video sequence. We assume images are grey-level images.

The main properties of the method are the following:

1. In contrast to classical approaches, exchanging the two signals leads to identical results, i.e., mutually inverse geometric and radiometric transformations.

2. LSM can be realized such that – for sets of simulated data – the three statistical tests on the correctness of the implementation generally do not fire:

   (a) The estimated variance factor does not differ from 1 too much. This indicates that model and (simulated) data are consistent.

   (b) The empirical covariance matrix derived from samples does not significantly differ from the theoretical covariance matrix, which is the Cramer-Rao bound. This suggests, that the theoretical covariance matrix can be used as reliable uncertainty indicator.

   (c) The estimates show no significant bias.

   Therefore we can rely on the estimated parameters and their covariance matrix in real cases if convergence is achieved and the variance factor does not differ from 1 too much. If $\widehat{\sigma}_0$ is not close to 1, this indicates, the assumed model does not fit to the observations. The cause of this effect cannot be given: it may be, that the scene is not flat, the windows are too large, the estimated noise variance deviates from the noise variance in the windows, the shadow situation in both images is different, and so on.

3. Individual variances for all observations can be taken into account, especially, position or signal depending variances. Covariances are neglected. The signal dependent noise variacances can be derived from given images without needing any control parameters.

4. Interpolation of the observed signals is necessary for reconstructing the unknown signal. This causes some – generally negligible – approximations in the method.

5. For affine geometric transformations the similarity transformation derivable from corresponding Lowe-keypoints can be used as approximate transformation.

6. From the estimated parameters and their covariance matrix one can derive refined point correspondences of affine correspondences. The affine parameters on an average have a relative accuracy a few percent, and the estimated shifts/parallaxes a standard deviation below 0.1 pixels.

   The affine correspondences can be used for estimating the relative pose of two calibrated or partially calibrated cameras, i.e., with or without focal length based on pairs of affine matches.

The MATLAB software provided consists of a core routine

`LSM_62_sym_main.m`

and several demo routines

- `demo_LSM_small.m` for showing the use of the main routine,
- `demo_LSM_medium.m` for showing the use of keypoints and the noise varaicne estimation,
- `demo_LSM_simulated.m` for checking the implementation with simulated data, and
- `demo_LSM_image_pairs.m` for manually providing keypoints in real data.

On Notation. Signals are two-dimensional function. The function names taken from the middle of the alphabet, e.g., f, g, and h. Coordinate names are taken from the end of the alphabet: so e.g., $f(\boldsymbol{x})$, $g(\boldsymbol{y})$, and $h(\boldsymbol{z})$. Sometimes, we use the convention $\boldsymbol{x} = [x, y]^\mathsf{T}$. Functions depend on coordinates $\boldsymbol{x}_i$, where the index range is a set of integers $\in \mathbb{Z}$ If a coordinate $\boldsymbol{x}$ has no index, the context tells whether it is a real or an integer. If two coordinates, say $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$ have the same index they refer to corresponding points. Homogeneous coordinates and matrices are boldface upright, e.g., coordinates $\mathbf{x}$ or transformation A. The dimension depends on the context. Stochastical variables are underscored, e.g., the discrete noise function is $\underline{m}(\boldsymbol{x}_i)$. Sets and names are written with calligraphic letters, e.g., the set of all pixels in the first image is $\mathcal{G} = \{(x, y, g)_i, i = 1, ..., I\}$. We denote the two images g and h as left and right image or first and second image, depending on the context.

2 Image Matching

This section describes the details of the model underlying the matching approach and the method for estimating the unknown parameters and their uncertainty.

The pipeline for using the matching procedure SYM-LSM is shown in Fig. 1.

2.1 Model

Let the two square image windows $g(\boldsymbol{y})$ and $h(\boldsymbol{z})$ be given, see Fig. 2. The coordinates refer to the centre of the windows. We assume, both windows are noisy observations of

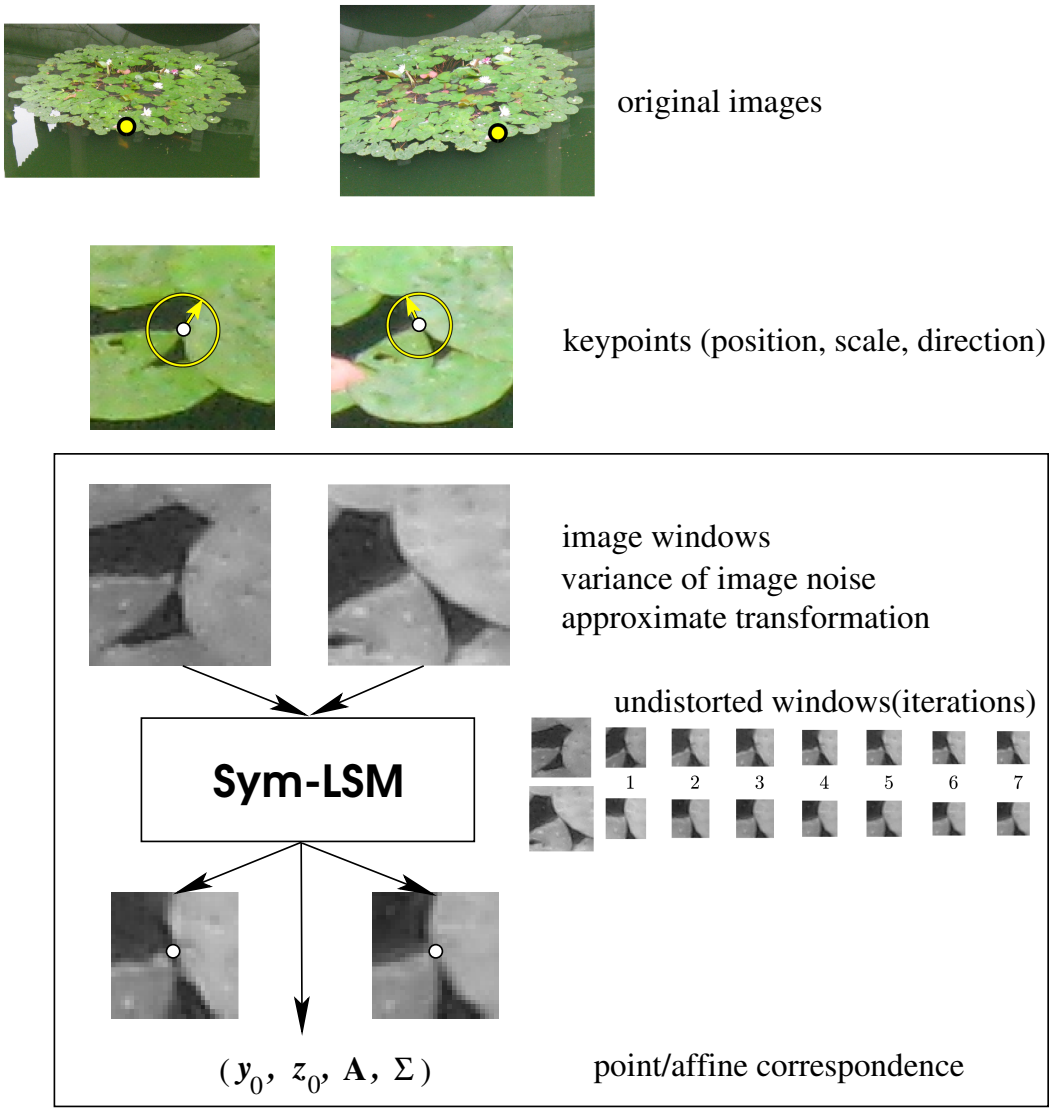

Figure 1: Pipeline for image matching. Starting from two images and measured keypoints with position, scale and direction, two corresponding windows are determined as input to the matching procedure SYM-LSM. Given approximate values and variances for the image noise it iteratively mutually undistorts the two image windows. After convergence it provides best estimates for the geometric and radiometric transformation between the two image windows, which allow to derive a point or an affine correspondence together with its covariance matrix

an unknown true underlying signal $f(\boldsymbol{x})$, with individual geometric distortion, brightness, and contrast. We want to determine the geometric distortion $\boldsymbol{z} = \mathcal{A}(\boldsymbol{y})$ and the radiometric distortion $h = \mathcal{R}(g) = pg + q$. Classical matching methods, assume the geometric and radiometric distortion of one of the two windows is zero, e.g., assuming the reference image is identical to the first image $g(\boldsymbol{y}) = f(\boldsymbol{x})$, with $\boldsymbol{y} = \boldsymbol{x}$.

We break this asymmetry by placing the unknown signal $f(\boldsymbol{x})$ in the middle between

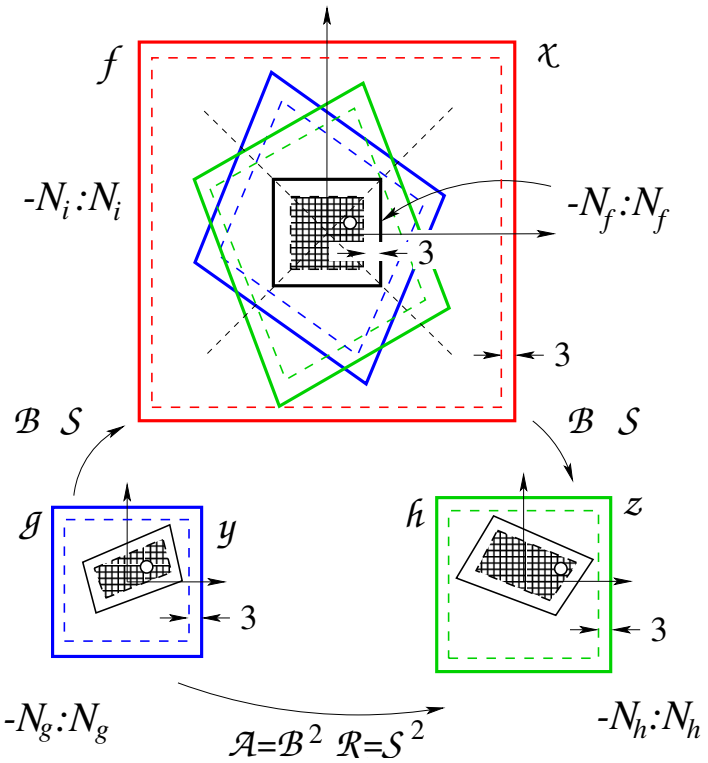

Figure 2: Relations between two given square image patches $g(\boldsymbol{y})$ (blue) and $h(\boldsymbol{h})$ (green) and the mean patch $f(\boldsymbol{x})$ (which is the black within the red region). The two image patches $\mathcal{g}$ and $\mathcal{h}$ are related by geometric and radiometric affinities $\mathcal{B}$ and $\mathcal{S}$, respectively. The correspondence is established by the patch f. Geometrically and radiometrically it lies in the middle between g and h. Only a region in the overlap of the two patches g and h mapped to f can be used. We choose the maximum square (black). The observations are all pixels in g and h which map into the black square of the reference image f. We assume the reference image f is a restored version of the weighted mean of the two projected images g and h. The patches g and h may have different sizes. The size of the unknown signal (black, textured) depends on the sizes of g and h, the approximate affine transformation $\mathcal{A} = \mathcal{B}^2$ and a border to allow bi-cubic interpolation, and is adapted in each iteration. The large image in the $\mathcal{x}$-frame is used for generating artificial images. The radiometric transformation $\mathcal{R} = \mathcal{S}^2$ is splitted in the same way (not shown here). The size of the windows is given by the ranges of the pixels, e.g., for image f the range is $-N_i : N_i := -N_i, ..., N_i$

the observed signals between g and h:

$$g(\boldsymbol{y}) \xrightarrow{\mathcal{B},\mathcal{S}} f(\boldsymbol{x}) \xrightarrow{\mathcal{B},\mathcal{S}} h(\boldsymbol{z}) \quad \text{such that} \quad \mathcal{A} = \mathcal{B}^2 \,, \mathcal{R} = \mathcal{S}^2 \,. \tag{1}$$

The geometric and the radiometric transformations $\mathcal{A}$ and $\mathcal{R}$ are split into the sequence of two identical geometric and radiometric transformations $\mathcal{B}$ and $\mathcal{S}$, with the signal $f(\boldsymbol{x})$ having the same distortion w.r.t. $g(\boldsymbol{y})$ as the signal $h(\boldsymbol{z})$ has w.r.t. $f(\boldsymbol{x})$. Therefore the complete geometric transformations $\mathcal{A}$ and $\mathcal{R}$ result from two times applying $\mathcal{B}$ and $\mathcal{S}$ to $g(\boldsymbol{y})$: hence, we have $\boldsymbol{z} = \mathcal{B}(\mathcal{B}(\boldsymbol{y})) = \mathcal{B}^2(\boldsymbol{y})$ and $h = \mathcal{S}(\mathcal{S}(\boldsymbol{g})) = \mathcal{S}^2(g)$.

Assuming affinities for the geometric and the radiometric distortion, we have the

following generative model. The geometric and the radiometric models for the two images are

$$\boldsymbol{y} \mapsto \boldsymbol{x}: \quad \boldsymbol{x} = B\boldsymbol{y} + \boldsymbol{b} \quad \text{and} \quad \boldsymbol{x} \mapsto \boldsymbol{z}: \quad \boldsymbol{z} = B\boldsymbol{x} + \boldsymbol{b} \tag{2}$$

$$g \mapsto f: \quad f = sg + t \quad \text{and} \quad f \mapsto h: \quad h = sf + t \tag{3}$$

with

$$B = \begin{bmatrix} b_1 & b_3 \\ b_2 & b_4 \end{bmatrix} \quad \text{and} \quad \boldsymbol{b} = \begin{bmatrix} b_5 \\ b_6 \end{bmatrix} \tag{4}$$

In the following we collect the eight unknown parameters of the affinities in the vector

$$\boldsymbol{\theta} = \begin{bmatrix} \boldsymbol{\theta}_G \\ \boldsymbol{\theta}_I \end{bmatrix} \quad \text{with} \quad \boldsymbol{\theta}_G = [b_1, b_2, b_3, b_4, b_5, b_6]^\mathsf{T} \quad \text{and} \quad \boldsymbol{\theta}_I = [s, t]^\mathsf{T}. \tag{5}$$

Hence, we have the relation of the geometric transformations $\mathcal{B}$ to the compound transformation $\mathcal{A}$ with the six parameters in $\boldsymbol{a}$

$$\mathsf{A} = \begin{bmatrix} A & \boldsymbol{c} \\ \boldsymbol{0}^\mathsf{T} & 1 \end{bmatrix} = \mathsf{B}^2 \quad \text{with} \quad A = \begin{bmatrix} a_1 & a_3 \\ a_2 & a_4 \end{bmatrix} \quad \text{and} \quad \boldsymbol{c} = \begin{bmatrix} a_5 \\ a_6 \end{bmatrix} \tag{6}$$

This model is rigorous only, if

1. the scene surface is planar in a differentiable region, and
2. the intensity differences result from brightness or contrast changes only.

Hence, the geometric model is adequate, if the surface is smooth and the window size is not too large. Usually we have window sizes between 15×15 and 100×100 pixels, but larger windows may be fine in special cases. The radiometric model is adequate if there are no occlusions or local illumination differences, e.g., caused by shadows, when the images are not taken simultaneously.

We now assume the intensities g and h are noisy and distorted versions of an underlying true signal f with standard deviations $\sigma_n(g)$ and $\sigma_m(h)$ depending on g and h. Integrating the geometry and intensity transformation we arrive at the following model, which is generative, i.e., allows to simulate observed images. Using the radiometric transformations (3) we first obtain for all pixels j and k in the first and the second image and the corresponding pixels in the reference image f

$$\underline{g}(\boldsymbol{y}_j) = 1/s \cdot (f(\boldsymbol{x}_j) - t) + \underline{n}(\boldsymbol{y}_j), \quad j = 1, \ldots, J \tag{7}$$

$$\underline{h}(\boldsymbol{z}_k) = (s\, f(\boldsymbol{x}_k)) + t) + \underline{m}(\boldsymbol{z}_k), \quad k = 1, \ldots, K \tag{8}$$

With the geometric transformations (2) we thus explicitly have

$$\underline{g}(\boldsymbol{y}_j) = 1/s \cdot \left(f\left(B\boldsymbol{y}_j + \boldsymbol{b}\right) - t\right) + \underline{n}(\boldsymbol{y}_j), \quad j = 1, \ldots, J \tag{9}$$

$$\underline{h}(\boldsymbol{z}_k) = \left(s\, f\left(B^{-1}(\boldsymbol{z}_k - \boldsymbol{b})\right) + t\right) + \underline{m}(\boldsymbol{z}_k), \quad k = 1, \ldots, K. \tag{10}$$

The reference image f is assumed to be a smooth function. We represent it using a regular grid with bi-cubic interpolation, which is necessary, since the coordinates $\boldsymbol{x}_j = B\boldsymbol{y}_j + \boldsymbol{b}$ and $\boldsymbol{x}_k = B^{-1}(\boldsymbol{z}_k - \boldsymbol{b})$ are real valued and generally do not fall on the grid.

2.2 The estimation

The task is to estimate the eight parameters $\boldsymbol{\theta} = (\boldsymbol{\theta}_G, \boldsymbol{\theta}_I)$ for the geometric and the radiometric transformation and the unknown true signal f from the observed values $g(\boldsymbol{y}_j)$ and $h(\boldsymbol{z}_k)$.

The explicit modeling in (9) and (10) allows us to write the problem as nonlinear Gauss-Markov model with the residuals

$$n_j(\boldsymbol{\theta}, f) = g_j - 1/s \cdot \left(f\left(\mathsf{B}\boldsymbol{y}_j + \boldsymbol{b}\right) - t\right) \quad , \qquad \mathbb{D}(\underline{n}_j) = \sigma_{n_j}^2 \, , \quad j = 1, ..., J \tag{11}$$

$$m_k(\boldsymbol{\theta}, f) = h_k - \left(s\, f\left(\mathsf{B}^{-1}(\boldsymbol{z}_k - \boldsymbol{b})\right) + t\right) \, , \qquad \mathbb{D}(\underline{m}_k) = \sigma_{m_k}^2 \, , \quad k = 1, ..., K \tag{12}$$

for all pixels $g_j := g(\boldsymbol{y}_j)$ of $\boldsymbol{g}$ and all pixels $h_k := h(\boldsymbol{z}_k)$ of $\boldsymbol{h}$ falling into the common region in $\boldsymbol{f}$. Maximum likelihood (ML) estimates result from minimizing the weighted sum of the residuals

$$\Omega(\boldsymbol{\theta}, f) \;\; = \;\; \sum_j w_j n_j^2(\boldsymbol{\theta}, f) + \sum_k w_k m_k^2(\boldsymbol{\theta}, f) \tag{13}$$

w.r.t. the unknown distortion parameters $\boldsymbol{\theta}$ and the unknown signal f, using proper weights

$$w_j = \frac{1}{\sigma_{n_j}^2} \quad \text{and} \quad w_k = \frac{1}{\sigma_{n_k}^2} \tag{14}$$

The statistical properties of the noise need to be specified, e.g., assuming the variance to be signal dependent, thus e.g.,

$$\sigma_n^2(\boldsymbol{y}) = V_g(\tilde{g}(\boldsymbol{y})) \qquad \sigma_m^2(\boldsymbol{z}) = V_h(\tilde{h}(\boldsymbol{z})) \, . \tag{15}$$

We prior to the optimization estimate the signal dependent variance functions of the two images, see Förstner (2000). In addition we take the effect of bi-linear interpolation into account, see (18).

Since, due to the size of f, the number of unknowns is comparably large, say, in the range between 200 and 10000. Therefore we solve this problem by alternatively fixing one group of the parameters and solving for the other:

$$\widehat{\boldsymbol{\theta}} \mid \widehat{f} \;\; = \;\; \text{argmin}_\theta \Omega(\boldsymbol{\theta}, \widehat{f}) \, , \tag{16}$$

$$\widehat{f} \mid \widehat{\boldsymbol{\theta}} \;\; = \;\; \text{argmin}_f \Omega(\widehat{\boldsymbol{\theta}}, f) \, , \tag{17}$$

in a block Gauss-Seidel fashion.

Especially, the estimated unknown function f is the weighted mean of the functions g and h transformed into the coordinate system $\boldsymbol{x}$ of f, which can be calculated pixel wise as a weighted sum of the two image windows warped into f:

$$\widehat{f}_i \mid \widehat{\boldsymbol{\theta}} = \frac{w_{f_i}^{(g)} \; f_i^{(g)} \; + \; w_{f_i}^{(h)} \; f_i^{(h)}}{w_{f_i}^{(g)} \; + \; w_{f_i}^{(h)}} \, , \tag{18}$$

with the warped image windows

$$f_i^{(g)} := f_i^{(g)}(\boldsymbol{x}_i) = s \cdot g(\boldsymbol{y}_i) + t \quad \text{and} \quad f_i^{(h)} := f_i^{(h)}(\boldsymbol{x}_i) = 1/s \cdot (h(\boldsymbol{z}_i) - t) \tag{19}$$

from (3) and

$$\boldsymbol{y}_i = B^{-1}(\boldsymbol{x}_i - \boldsymbol{b})) \quad \text{and} \quad \boldsymbol{z}_i = B\boldsymbol{x}_i + \boldsymbol{b} \tag{20}$$

from (2). The weights are

$$w_{f_i}^{(g)} = \frac{1}{s^2 \cdot V_g(g(\boldsymbol{y}_i))} \quad \text{and} \quad w_{f_i}^{(h)} = \frac{s^2}{V_g(h(\boldsymbol{z}_i))}\,. \tag{21}$$

Bi-cubic interpolation is used to transfer $g(\boldsymbol{y}_i)$ and $h(\boldsymbol{z}_i)$ to $f(\boldsymbol{x}_i)$, see (19). Observe, the individual pixels f_i of the estimated function function generally do not lie in the middle of corresponding pixels $f_i^{(g)}$ and $f_i^{(h)}$.

Bi-cubic interpolation induces additional errors, which we interprete as additional noise of the observations. Hence, the variances of the observations are assumed to have two components, one from the imaging process and one, the variances $\sigma_{\delta_g}^2$ and $\sigma_{\delta_h}^2$, from the interpolation process:

$$\sigma_n^2(\boldsymbol{y}) = V_g(\tilde{g}(\boldsymbol{y})) + 1/s \cdot \sigma_{\delta_g}^2 \qquad \sigma_m^2(\boldsymbol{z}) = V_h(\tilde{h}(\boldsymbol{z})) + s\ \sigma_{\delta_h}^2\,. \tag{22}$$

For details for the interpolation error see the Appendix, Eq. (129).

As a result of the ML-estimation we obtain: (1) the parameters $\widehat{\boldsymbol{\theta}}$, (2) their $\Sigma_{\widehat{\theta}\widehat{\theta}}$, and (3) the variance factor ,

$$\widehat{\sigma}_0^2 = \frac{\Omega(\widehat{\boldsymbol{\theta}}, \widehat{f})}{R}\,, \tag{23}$$

where R is the redundancy of the system, i.e., the efficient number of observations[1] $K_g + K_h$ minus the number of unknown parameters $8 + K_f$, where we take the approximation $K_f = \sqrt{K_g K_h}$ for the number of parameters of the grid defining f:

$$R = K_g + K_h - (8 + \sqrt{K_g K_h})\,. \tag{24}$$

If the model holds, the variance factor is Fisher distributed with $F(R, \infty)$, thus should be close to 1. Therefore, it is reasonable to multiply the covariance matrix $\Sigma_{\widehat{\theta}\widehat{\theta}}$ with the variance factor to arrive at a realistic characterization

$$\widehat{\Sigma}_{\widehat{\theta}\widehat{\theta}} = \widehat{\sigma}_0^2\ \Sigma_{\widehat{\theta}\widehat{\theta}} \tag{25}$$

of the uncertainty of the estimated parameters.

The covariance matrix $\widehat{\Sigma}_{\widehat{\psi}\widehat{\psi}}$ of the sought affinities finally are derived by variance propagation from

$$\mathsf{A}(\boldsymbol{\psi}_{1..6}) = \begin{bmatrix} \psi_1 & \psi_3 & \psi_5 \\ \psi_2 & \psi_4 & \psi_6 \\ 0 & 0 & 1 \end{bmatrix} = \mathsf{B}(\boldsymbol{\theta}_{1..6})^2 = \begin{bmatrix} \theta_1^2 + \theta_2\theta_3 & \theta_1\theta_3 + \theta_3\theta_4 & \theta_5 + \theta_1\theta_5 + \theta_3\theta_6 \\ \theta_1\theta_2 + \theta_2\theta_4 & \theta_4^2 + \theta_2\theta_3 & \theta_6 + \theta_2\theta_5 + \theta_4\theta_6 \\ 0 & 0 & 1 \end{bmatrix} \tag{26}$$

and

$$\mathsf{R}(\boldsymbol{\psi}_{7,8}) = \begin{bmatrix} \psi_7 & \psi_8 \\ 0 & 1 \end{bmatrix} = \mathsf{S}(\boldsymbol{\theta}_{7,8})^2 = \begin{bmatrix} \theta_7^2 & \theta_8 + \theta_7\theta_8 \\ 0 & 1 \end{bmatrix} \tag{27}$$

[1] We set $K_g := J$ and $K_h := K$.

with the Jacobian

$$J_{\psi\theta} = \begin{bmatrix} 2\,\theta_1 & \theta_3 & \theta_2 & 0 & 0 & 0 & 0 & 0 \\ \theta_2 & \theta_1+\theta_4 & 0 & \theta_2 & 0 & 0 & 0 & 0 \\ \theta_3 & 0 & \theta_1+\theta_4 & \theta_3 & 0 & 0 & 0 & 0 \\ 0 & \theta_3 & \theta_2 & 2\,\theta_4 & 0 & 0 & 0 & 0 \\ \theta_5 & 0 & \theta_6 & 0 & \theta_1+1 & \theta_3 & 0 & 0 \\ 0 & \theta_5 & 0 & \theta_6 & \theta_2 & \theta_4+1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2\,\theta_7 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \theta_8 & \theta_7+1 \end{bmatrix}, \tag{28}$$

hence

$$\widehat{\Sigma}_{\widehat{\psi}\widehat{\psi}} = \widehat{\sigma}_0^2 \; J_{\psi\theta} \; \Sigma_{\widehat{\theta}\widehat{\theta}} \; J_{\psi\theta}^{\mathsf{T}} \, . \tag{29}$$

2.3 Refined correspondences

The result of the LSM can be used to provide refined correspondences.

2.3.1 Point and affine correspondences

Point correspondences follow from (26). For some arbitrary point $\boldsymbol{y}_0$ in the left image we obtain its correspondent coordinates in the right image from

$$\boldsymbol{z} = \widehat{A}\,\boldsymbol{y}_0 + \widehat{\boldsymbol{c}} \tag{30}$$

with

$$\widehat{A} = \begin{bmatrix} \widehat{\psi}_1 & \widehat{\psi}_3 \\ \widehat{\psi}_2 & \widehat{\psi}_4 \end{bmatrix} \quad \text{and} \quad \widehat{\boldsymbol{c}} = \begin{bmatrix} \widehat{\psi}_5 \\ \widehat{\psi}_6 \end{bmatrix} . \tag{31}$$

Thus we have the corresponding point pair

$$\boxed{\{\boldsymbol{y}_0, \boldsymbol{z}\} \quad \text{with} \quad \boldsymbol{z} = \widehat{A}\,\boldsymbol{y}_0 + \widehat{\boldsymbol{c}}\,.} \tag{32}$$

If the centre point in the left image is taken, we have $\boldsymbol{y}_{00} = \boldsymbol{0}$ and the corresponding point pair is

$$\boxed{\{\boldsymbol{y}_{00}, \boldsymbol{z}\} \quad \text{with} \quad \boldsymbol{y}_{00} = \boldsymbol{0} \quad \text{and} \quad \boldsymbol{z} = \widehat{\boldsymbol{c}}\,.} \tag{33}$$

Affine correspondences are defined as a pair $\{\boldsymbol{y}_0, \boldsymbol{z}_0\}$ of corresponding keypoints together with the affine transform $\widehat{\mathsf{A}}$ in (26) which – as seen above, see (30) – can be used to refine the position of $\boldsymbol{z}_0$ as in (30) together with the local distortion:

$$\boxed{\left\{\boldsymbol{y}_0, \boldsymbol{z}_0, \widehat{\mathsf{A}}\right\} .} \tag{34}$$

2.3.2 Uncertain point correspondences

The covariance matrix of the estimated parameters can be used to derive a pair of corresponding points together with their uncertainty. In case we choose the centre $\boldsymbol{y}_{00} = \mathbf{0}$ of the left window as point to be matched we have the following uncertain correspondence

$$\boxed{\{\boldsymbol{y}_{00}, \boldsymbol{z}, \Sigma_{zz}\}} \tag{35}$$

with

$$\boldsymbol{y}_{00} = \mathbf{0}\,, \quad \boldsymbol{z} = \widehat{\boldsymbol{c}} = \begin{bmatrix} \widehat{\psi}_5 \\ \widehat{\psi}_6 \end{bmatrix} \quad \text{and} \quad \Sigma_{zz} = \Sigma_{\widehat{c}\widehat{c}} = \begin{bmatrix} \sigma^2_{\widehat{\psi}_5} & \sigma_{\widehat{\psi}_5}\sigma_{\widehat{\psi}_6} \\ \sigma_{\widehat{\psi}_6}\sigma_{\widehat{\psi}_5} & \sigma^2_{\widehat{\psi}_6} \end{bmatrix} \tag{36}$$

Observe, the point in the left image is assumed to be certain, while the point in the right image carries all uncertainty. The covariance matrix $\Sigma_{\widehat{a}\widehat{a}}$ therefore also provides the uncertainty of the parallaxes $\boldsymbol{p} = \boldsymbol{z} - \boldsymbol{y}_{00} = \boldsymbol{c}$:

$$\Sigma_{pp} = \Sigma_{\widehat{c}\widehat{c}}\,. \tag{37}$$

3 Realization of symmetric least squares matching

In this section we derive the Jacobians in detail.

3.1 Jacobians of g

We start with the differential for

$$g_j = 1/\theta_7 \cdot (f(\boldsymbol{x}_j) - \theta_8) \quad \text{with} \quad \boldsymbol{x}_j = B\boldsymbol{y}_j + \boldsymbol{b} \tag{38}$$

w.r.t. to the unknown parameters $\boldsymbol{\theta}$. We explicitly have

$$g_j = 1/\theta_7 \cdot (f(\theta_1 x_j + \theta_3 y_j + \theta_5, \theta_2 x_j + \theta_4 y_j + \theta_6) - \theta_8)\ . \tag{39}$$

Assuming approximate values $\boldsymbol{\theta}^a$ for the parameters, we have

$$g_j = g_j^a + \mathrm{d}g_j = g_j^a + \left.\frac{\partial g_j}{\partial \boldsymbol{\theta}}\right|_{\theta^a} \mathrm{d}\boldsymbol{\theta}\,, \tag{40}$$

with

$$g_j^a = 1/\theta_7^a \cdot \left(f(B^a\boldsymbol{y}_j + \boldsymbol{b}^a) - \theta_8^a\right)\ , \tag{41}$$

and

$$\frac{\partial g_j}{\partial \boldsymbol{\theta}} = 1/\theta_7 \cdot [f_{x,j}x_j \mid f_{y,j}x_j \mid f_{x,j}y_j \mid f_{y,j}y_j \mid f_{x,j} \mid f_{y,j} \mid -(f_j - \theta_8)/\theta_7 \mid -1]\ , \tag{42}$$

evaluated at the approximate values. This is due to $\mathrm{d}(1/x) = -1/x^2\,\mathrm{d}x$, which yields

$$\begin{aligned}
\mathrm{d}g_j &= +1/\theta_7 \cdot f_{x,j} \cdot x_j \cdot \mathrm{d}\theta_1 && (43)\\
&\quad +1/\theta_7 \cdot f_{y,j} \cdot x_j \cdot \mathrm{d}\theta_2 && (44)\\
&\quad +1/\theta_7 \cdot f_{x,j} \cdot y_j \cdot \mathrm{d}\theta_3 && (45)\\
&\quad +1/\theta_7 \cdot f_{y,j} \cdot y_j \cdot \mathrm{d}\theta_4 && (46)\\
&\quad +1/\theta_7 \cdot f_{x,j} \cdot \mathrm{d}\theta_5 && (47)\\
&\quad +1/\theta_7 \cdot f_{y,j} \cdot \mathrm{d}\theta_6 && (48)\\
&\quad -1/\theta_7^2 \cdot (f_j - \theta_8) \cdot \mathrm{d}\theta_7 && (49)\\
&\quad -1/\theta_7 \cdot\ \mathrm{d}\theta_8\,. && (50)
\end{aligned}$$

3.2 Jacobians of h

For the derivatives of

$$h_k = \theta_7 \underbrace{f(\boldsymbol{x}_k)}_{f_k} + \theta_8 \quad \text{with} \quad \boldsymbol{x}_k = B^{-1}(\boldsymbol{z}_k - \boldsymbol{b}) \tag{51}$$

w.r.t. $\boldsymbol{\theta}$ we need some preparation. The Jacobian of the inverse is

$$\mathrm{d}B^{-1} = -B^{-1}(\mathrm{d}B)B^{-1} \quad \text{with} \quad B^{-1} = \frac{1}{D}\begin{bmatrix} a_4 & -a_2 \\ -a_3 & a_1 \end{bmatrix} \quad \text{and} \quad D := |B|\,. \tag{52}$$

We in addition use the substitution

$$\boldsymbol{x}'_k = B^{-1}(\boldsymbol{z}_k - \boldsymbol{b})\,. \tag{53}$$

These are the coordinates of the point in f corresponding to $\boldsymbol{z}_k$. Hence, we have – only referring to the first four parameters

$$\begin{aligned}
\mathrm{d}(B^{-1}(\boldsymbol{z}_k - \boldsymbol{b})) &= -B^{-1}(\mathrm{d}B)B^{-1}(\boldsymbol{z}_k - \boldsymbol{b}) && (54)\\
&= -B^{-1}(\mathrm{d}B)\boldsymbol{x}'_k && (55)\\
&= -(\boldsymbol{x}'^{\mathsf{T}}_k \otimes B^{-1})\mathrm{d}\boldsymbol{\theta}\,. && (56)
\end{aligned}$$

We finally have

$$\begin{aligned}
\frac{\partial f(\boldsymbol{z}'_k)}{\partial \boldsymbol{\theta}} &= \frac{\partial f(\boldsymbol{z}'_k)}{\partial x'_k}\frac{\partial x'_k}{\partial \boldsymbol{\theta}} + \frac{\partial f(\boldsymbol{z}'_k)}{\partial y'_k}\frac{\partial y'_k}{\partial \boldsymbol{\theta}} && (57)\\
&= - \quad \nabla^{\mathsf{T}} f\ (\boldsymbol{x}'^{\mathsf{T}}_k \otimes B^{-1}) && (58)\\
&= -(1 \otimes \nabla^{\mathsf{T}} f)(\boldsymbol{x}'^{\mathsf{T}}_k \otimes B^{-1}) && (59)\\
&= -\boldsymbol{x}'^{\mathsf{T}}_k \otimes \nabla^{\mathsf{T}} f B^{-1} && (60)
\end{aligned}$$

Therefore with

$$\nabla\alpha = \begin{bmatrix} \alpha_x \\ \alpha_y \end{bmatrix} = B^{-\mathsf{T}}\nabla f \tag{61}$$

we obtain

$$\frac{\partial f(\boldsymbol{z}'_k)}{\partial \boldsymbol{\theta}} = -\boldsymbol{x}'^{\mathsf{T}}_k \otimes \nabla^{\mathsf{T}} \alpha_k = -[x'_k\alpha_{x,k} \quad x'_k\alpha_{y,k} \quad y'_k\alpha_{x,k} \quad y'_k\alpha_{y,k}]\,. \tag{62}$$

Finally, the differential of h is

$$\begin{aligned}
\mathrm{d}h_k &= -\theta_7\alpha_{x,k}x'_k\,\mathrm{d}\theta_1 && (63)\\
&\quad -\theta_7\alpha_{y,k}x'_k\,\mathrm{d}\theta_2 && (64)\\
&\quad -\theta_7\alpha_{x,k}y'_k\,\mathrm{d}\theta_3 && (65)\\
&\quad -\theta_7\alpha_{y,k}y'_k\,\mathrm{d}\theta_4 && (66)\\
&\quad -\theta_7\alpha_{x,k}\,\mathrm{d}\theta_5 && (67)\\
&\quad -\theta_7\alpha_{y,k}\,\mathrm{d}\theta_6 && (68)\\
&\quad +f_k\,\mathrm{d}\theta_7 && (69)\\
&\quad +1\,\mathrm{d}\theta_8 && (70)
\end{aligned}$$

Hene we have the compact form

$$h_k = h^a_k + \mathrm{d}h_k = h^a_k + \frac{\partial h_k}{\partial \boldsymbol{\theta}}\,\mathrm{d}\boldsymbol{\theta} \tag{71}$$

with

$$\frac{\partial h_k}{\partial \boldsymbol{\theta}} = -\theta_7\left[\alpha_{x,k}x_k \mid \alpha_{y,k}x_k \mid \alpha_{x,k}y_k \mid \alpha_{y,k}y_k \mid \alpha_{x,k} \mid \alpha_{y,k} \mid -f_k\theta_7^{-1} \mid -1\right]\,, \tag{72}$$

evaluated at the approximate values $\boldsymbol{\theta}^a$ and

$$h^a_k = \theta^a_7 f\left((B^a)^{-1}(\boldsymbol{z}_k - \boldsymbol{b}^a)\right) + \theta^a_8\,. \tag{73}$$

3.3 Jacobian $\boldsymbol{X}$ and the linearized observations $\Delta\boldsymbol{y}$

With the abbreviations

$$\boldsymbol{x}^{\mathsf{T}}_{g_j\theta} = \frac{\partial g_j}{\partial \boldsymbol{\theta}} \quad \text{and} \quad \boldsymbol{x}^{\mathsf{T}}_{h_k\theta} = \frac{\partial h_k}{\partial \boldsymbol{\theta}} \tag{74}$$

the design matrix $\mathsf{X} = \partial\boldsymbol{y}/\partial\boldsymbol{\theta}$ is given by

$$\mathsf{X} = \frac{\partial \boldsymbol{y}}{\partial \boldsymbol{\theta}} = \begin{bmatrix} \mathsf{X}_g \\ \mathsf{X}_h \end{bmatrix} \quad \text{with} \quad \mathsf{X}_g = \begin{bmatrix} \boldsymbol{x}^{\mathsf{T}}_{g_1\theta} \\ \cdots \\ \boldsymbol{x}^{\mathsf{T}}_{g_j\theta} \\ \cdots \\ \boldsymbol{x}^{\mathsf{T}}_{g_J\theta} \end{bmatrix} \quad \text{and} \quad \mathsf{X}_h = \begin{bmatrix} \boldsymbol{x}^{\mathsf{T}}_{h_1\theta} \\ \cdots \\ \boldsymbol{x}^{\mathsf{T}}_{h_k\theta} \\ \cdots \\ \boldsymbol{x}^{\mathsf{T}}_{h_K\theta} \end{bmatrix} \tag{75}$$

Similarly, we have the residuals or the negative linearized observation

$$\boldsymbol{v} = -\Delta\boldsymbol{y} = -\begin{bmatrix} \Delta\boldsymbol{y}_g \\ \Delta\boldsymbol{y}_h \end{bmatrix} = \begin{bmatrix} [g^a_j - g_j] \\ [h^a_k - h_k] \end{bmatrix}\,. \tag{76}$$

This leads to the normal equation system

$$\mathsf{N}\widehat{\Delta\boldsymbol{\theta}} = \boldsymbol{n} \quad \text{with} \quad \mathsf{N} = \mathsf{X}^{\mathsf{T}}\mathsf{W}\mathsf{X} \quad \text{and} \quad \boldsymbol{n} = \mathsf{X}^{\mathsf{T}}\mathsf{W}\Delta\boldsymbol{y}\,, \tag{77}$$

and the updates in the ν-th iteration

$$\widehat{\boldsymbol{\theta}}^{(\nu+1)} = \widehat{\boldsymbol{\theta}}^{(\nu)} + \widehat{\Delta\boldsymbol{\theta}}^{(\nu)}\,. \tag{78}$$

3.4 The accuracy potential of LSM

The potential of the refinement of the affinitiy using LSM, namely the expected precision of the affinity for ideal cases, can be easily derived, see Barath et al. (2020b). This is based on the part of the normal equation matrix N related to the 6 parameters of the geometric affinity: $\mathsf{N} = \sigma_n^{-2} \sum_{ij} \nabla f_\theta(i,j) \nabla f_\theta^\mathsf{T}(i,j)$, where the sum is over all pixels in an $N \times N$ window. If we assume the distortion is zero and f is known, then the gradient is $\nabla f = [xf_x, yf_x, xf_y, yf_y, f_x, f_y]$, see (43) ff. Observe, the 2×2 matrix referring to the translation parameters is proportional to the structure tensor of the patch. We now assume that the gradients in the window have the same variance $\sigma_{f'}^2$ and are mutually uncorrelated. Then the normal equation matrix will be diagonal leading to the covariance matrix

$$\Sigma_{\alpha\alpha} = \begin{bmatrix} \sigma_a^2 I_4 & 0 \\ 0 & \sigma_p^2 I_3 \end{bmatrix} \quad \text{with} \quad \sigma_a = \frac{\sqrt{12}}{N^2} \frac{\sigma_n}{\sigma_{f'}} \quad \text{and} \quad \sigma_p = \frac{1}{N} \frac{\sigma_n}{\sigma_{f'}} . \tag{79}$$

Hence, the standard deviations of the estimated affinity $\widehat{A}$ and shift $\widehat{\boldsymbol{c}}$ are below 1% and 0.1 pixels, except for very small scales. Moreover, for the window size $M \times M$, the standard deviations decrease with on average with M^2 and M, respectively, see Barath et al. (2020a)

3.5 The algorithm

The algorithm `LSM_62_sym_main.m` is given below.

The input, the output and the algorithmic steps are the following:

- The two image windows must be square and have values in the range $[0, 255]$. They have sizes $M_g \times M_g$ and $M_h \times M_h$, where the widths M_g and M_h are odd numbers. Their centres are at $[N_g + 1, N_g + 1]$ and $[N_h + 1, N_h + 1]$ with

$$N_g = \frac{M_g - 1}{2} \quad \text{and} \quad N_h = \frac{M_h - 1}{2} . \tag{80}$$

  The centres of the image windows – with coordinates $\boldsymbol{x}_0$ and $\boldsymbol{y}_0$ in the image – are assumed to be given by some keypoint detector, possibly with rounded coordinates. The affinity refers to a coordinate systems S_y and S_z parallel to the image coordinate system located at these centres. The coordinate systems are right handed, with x =rows and y =columns – in contrast to the MATLAB-image convention.

- The geometric affinity must not change the sign or the cheirality of the coordinate system, i.e., must not contain a mirroring or an exchange of the two axes. This is checked internally. In case the affinity between the two images is not sign preserving, one of the images needs to be mirrored before calling the matching routine.

- The variance functions $V_g = V_g(g)$ and $V_h = V_h(h)$ provide the variances as a function of the intensities. These function may be derived by some ‚blindÂ´ noise

Algorithm 1: Symmetric Least Squares Matching.
$[\widehat{\boldsymbol{\theta}}, \Sigma_{\widehat{\theta}\widehat{\theta}}, \widehat{\sigma}_0^2, R] = \{\texttt{Sym\_LSM}\}(g, h, V_g, V_h, \mathsf{A}^a, \mathsf{R}^a, \sigma_s, \max_\nu)$
Input: observed image patches $\{g, h\}$, must be square with odd width
variance functions V_g, V_h
approximate transformations $\mathsf{A}^a, \mathsf{R}^a$
smoothing parameter σ_s
maximum number of iterations $\max_\nu$.
Output: estimated parameters $\{\widehat{\boldsymbol{\theta}}, \Sigma_{\widehat{\theta}\widehat{\theta}}\}$;
variance factor $\widehat{\sigma}_0^2$;
redundancy R.

```
1  Initial approximate transformations/parameters: B^a = (A^a)^{1/2}, S^a = (R^a)^{1/2};
2  set iterations ν = 0, approximate parameters θ̂^0 := θ̂^a;
3  repeat
4  |  iteration ν := ν + 1;
5  |  find overlap N_f and observing pixels 𝒴, 𝒵 with weights w;
6  |  determine estimate f̂ of true signal, possibly smoothed with G(σ_s);
7  |  determine derivatives f̂_x and f̂_y;
8  |  foreach i = {g, h} do
9  |  |  warp [f̂, f̂_x, f̂_y] into i;
10 |  |  forall p ∈ i do
11 |  |  |  determine f̂(p), f̂_x(p), f̂_x(p);
12 |  |  |  determine Δy and x_iθ;
13 |  |  end
14 |  end
15 |  build system NΔθ = n with N = X^T W X, n = X^T W Δy;
16 |  determine estimates Δ̂θ and new approximate values θ̂^{ν+1};
17 |  determine covariance matrix Σ_θ̂θ̂ = N^{-1};
18 |  determine the redundancy R and the variance factor σ̂_0^2;
19 until max_u(||Δ̂θ_u||/σ_θ̂u) < 0.1 or ν = max_ν;
20 derive transformations Â and R̂ and parameters ψ̂ with Σ_ψ̂ψ̂
```

estimation procedure, e.g. `noise_standard_deviation_estimation.m`, which yields the noise standard deviation as a function of the intensity without requiring ground truth. Interpolation errors can be taken into account in this function.

- The approximate affinities for the geometric and the radiometric transformation are to be given as 3×3 matrix and 1×2 vector.

line 1 Internally half of the the approximate transformations is used.

line 2 The eight parameters in $\boldsymbol{\theta}$ refer to the six geometric parameters column wise and the two radiometric parameters.

line 5 `LSM_62_sym_par_find_observation_positions.m`: The overlap of the two images yields a square image f with size $M_f \times M_f$ (odd) with centre at $[N_f{+}1, N_f{+}1]$ being the origin of the coordinate system S_x, where the geometric coordinate transformations refer to. Again, the half size is $N_f = (M_f - 1)/2$. Only those pixels in g and h which fall into the common overlap after transformation into the coordinate system of f are used. Their coordinates are stored in $\mathcal{Y}$(`yi`) and $\mathcal{Z}$(`zi`). In order to obtain stable results and avoid oscillations, the overlap is not changed if the parameters change less than 50% of their standard deviations.

line 6 `LSM_62_sym_par_estimate_f.m`: The estimated signal $\widehat{f}$ in a first instance is the weighted average of g and h, transformed into the coordinate system S_x. It may be smoothed with a Gaussian having smoothing kernel σ_s.

7–18 `LSM_62_sym_par_estimate_parameters.m`.

lines 7 The derivatives $\widehat{f_x}$ and $\widehat{f_y}$ are realized by convolutions with Scharr's improved Sobel operator, see Jähne et al. (1999, Vol. II, p. 223)

$$d_x = \frac{1}{32} \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} [3 \;\; 10 \;\; 3] \quad \text{and} \quad d_y = d_x^{\mathsf{T}} \,. \tag{81}$$

(see also https://en.wikipedia.org/wiki/Sobel_operator, Alternative operators).

line 9 The three images $[\widehat{f}, \widehat{f_x}, \widehat{f_y}]$ are simultaneously warped into the images g and h to directly access the values in the coordinate systems S_y and S_z.

line 11 All function values and their derivatives are interpolated at the observing pixels, stored in $\mathcal{Y}$ and $\mathcal{Z}$.

line 12 The linearized observation $\widehat{\Delta y}$ refers to the original pixels in g and h and the interpolated pixels in f. This allows to determine the Jacobian X for all pixels stored in $\mathcal{Y}$ and $\mathcal{Z}$.

line 16 The parameters are corrected additively.

line 18 The redundancy needs an explanation. We have $K_g = |\mathcal{Y}|$ and $K_h = |\mathcal{Z}$ observations in g and h. The unknowns are the eight parameters $\boldsymbol{\theta}$ and the unknown function f. We assume, that the function f can be represented by the geometric mean of K_g and K_h parameters, which reflects the fact that if the affinity is a pure scaling, say by s, the numbers K_g and K_h differ by s^2. Therefore we determine the redundancy by

$$R = (K_g + K_h) - (8 + \sqrt{K_g K_h}) \,. \tag{82}$$

line 20 The full transformations are determined by squaring the estimated transformations together with their covariance matrix.

3.6 Realizing image warpings

Two steps in the algorithm require to warp images:

1. the warping of g and h into f in order to determine the estimate $\widehat{f}$.
2. the warping of f and its derivatives for building the design matrix X and the lienarized observations $\Delta \boldsymbol{y}$.

We provide two realizations. They differ in speed:

1. The first realization[2] perfoms the warping in a loop over all pixels. Due to the interpreter-characteristics of MATLAB this is slow.
2. The second realization[3] performs the warping using the function `imtransform.m`, see the blue and green boxes in Fig. 1. The MATLAB function `imtransform.m` is optimized which leads to lower CPU-times.

A transfer to a language which allows to compile the code, like `C`, preferably uses the first realization, since only the necessary pixels are handled. Furthermore, the set of pixels used for matching could be restricted to those, where the gradient is above a, possibly local, threshold in order to further speed up the algorithm.

4 Checking the implementation

We can check the coherence of the assumed model and the implementation using simulated data. For this purpose, we start from some ideal, true observation, i.e., image windows, and add noise according to the assumed model to all pixels. Varying the noise, we obtain K samples for the observations, and consequently $k = 1, ..., K$ samples for the estimates.[4]

We then can perform three checks, which can be realized as statistical tests:

1. The K estimated variance factors $\widehat{\sigma}^2_{0,k}$ should on an average be 1, since if the observed images are noisy affinly distorted versions of the true image, the expectation of the variance factors is 1. Deviations of the mean variance factor from 1 indicates, that the assumed model may not hold. If the variance factors do not differ from 1 too much, there is no reason to doubt the underlying model. The mean estimated variance factor follows a F-distributed test statistic F.
2. The K samples lead to K estimates $\boldsymbol{\theta}_k$ of the unknown parameters. The variation of these parameters, captured in their empirical covariance matrix should be close to the theoretical covariance matrix derived from estimation process, which is determined by variance propagation. The difference between the empirical and the theoretical covariance matrix is measured by a χ^2-distributed test statistic

[2] in `LSM_62_sym_estimate_f` and `LSM_62_sym_estimate_parameters`

[3] in `LSM_62_sym_warp_estimate_f` and `LSM_62_sym_warp_estimate_parameters`

[4] In this context the number K is not to be confused with the number of pixels in the image h.

X. If this test statistic is not in the rejection region, we have no reason not to use the theoretical covariance matrix as substitute for the empirical covariance matrix.

3. The K estimates $\boldsymbol{\theta}_k$ of the unknown parameters should on an average be close to the given/true parameters specified by the simulation. The bias, i.e., the difference between the mean of the parameters and the true values, leads to a χ^2-distributed test statistic X. If this test statistic is not in the rejection region, we have no reason to assume the estimates are biased.

The interpretation of the three test statistics assumes the model underlying the estimation procedure does not contain any approximations and the software implementation of the estimation model is perfect. Hence, if the tests to not fire, i.e., the test statistics are not in the rejection reason, we can assume both, possible approximations in the model are small and the implementation does not contain (severe) errors. This is the value of these tests.

If, however, the test statistics are in the rejection region, this indicates either the model is valid only approximately or the implementation contains errors.

In our case, there are several small approximations in the estimation process, e.g., the bi-cubic interpolation only leads to approximated interpolated values, since the true underlying signal is not known. This refers to the estimated signal f as well to the first derivatives, which need to be determined at non-integer coordinates.

The three tests are quite sensitive, i.e., increasing the number of samples K increases the probability that the test statistics lie in the rejection region.

The theory for these tests is documented in Förstner and Wrobel (2016, Sect. 4.6.8.1), see App. 8.3.

5 Noise variance estimation

The strength of the estimation procedure is to provide the covariance matrix of the estimated parameters which indicates the uncertainty of the estimated geometric and radiometric transformations. In order to this covariance matrix to be realistic, we need realistic variances for the given observations, i.e., the pixels in the two image windows.

In a first approximation we assume the pixels to be statistically independent and the variance of each pixels is a function of the intensity. This is motivated by the Poisson statistics of the photon counts which electronically are transferred into intensities. Since digital cameras generally aim at yielding nicely looking images, the photon counts of the sensors are further processed. This process usually is not made public by the producer of cameras.

Therefore we assume the variance of the intensities are some arbitrary smooth function of the intensity, and estimate this functions. This is done for each channel of the left and the right image. Since the matching algorithms assumes gray-level images, we take the average variance of the three channels as sufficiently good approximation for the variance function.

The noise variance estimation assumes, that the image contains sufficiently many pixels where the gradient is small, and estimates the noise variance from the gradients in these regions. Therefore noise estimation should be based on large enough windows, possibly larger than those used for matching.

Hence noise variance estimation can be done in two modes:

1. The noise variance functions $\sigma_n^2(g)$ and $\sigma_n^2(h)$ (variables `vg` and `vh`) are determined from one or several images which are characteristic for the matching windows, and used during the matching process.

2. The noise variance function $\sigma_n^2(g)$ and $\sigma_n^2(h)$ are determined from a large enough region, e.g., 400×400 pixels around the matching window, e.g., if the matching windows are specified by a keypoint in a larger image.

The second alternative is realized in the demos `demo_medium.m` and `demo_image_pairs.m`.

The theory for the noise estimation is given in Förstner (2000, Sect. 3), see App. 8.2.

6 Demo routines

We have realized four demo-routines,

1. `demo_LSM_small.m` for showing the use of the main routine when two image windows, approximate values and the variacne functions are given,

2. `demo_LSM_medium.m` for showing the use of the main routine when two image windows together with two corresponding LOWE-keypoints are given,

3. `demo_LSM_simulated.m` using simulated data for checking the correctness of the implementation, and

4. `demo_LSM_image_pairs.m` using real data with interactively identified correspondences.

Generally the control parameters are set in separate files and stored in the struct `par`.

The two files are `simulated_set_parameters.m` and `image_pair_set_parameters.m`.

6.1 Demo `demo_LSM_small.m`

The demo routine `demo_LSM_small.m` shows the most simple form of using of the main routine, here `LSM_62_sym_warp_main.m`. The required input data are loaded from file. The routine assumes the approximate values for the geometric transformation is given. Furthermore, the noise variance functions vg and vh are assumed to be provided, e.g. determined from a representative image using the routine `noise_standard_deviation_estimation.m`. The input images, the noise standard deviations and the change of the estimated image windows are shown in figures. It is best to start with this demo.

6.2 Demo `demo_LSM_medium.m`

The demo routine `demo_LSM_medium.m` shows how the noise variance estimation is integrated into the matching process. Again the required input data are loaded from file. Here these are the two complete images together with two corresponding LOWE-keypoints (coordinates, scale, direction). These are used to define the window size and the approximate values. The noise variance functions `vg` and `vh` are determined automatically from the area around the keypoints. First, the input images with the keypoints are shown. When zooming into the keypoints the centre and the direction vector fixing the scale and the direction can be seen. The start and the end of this arrow can be provided interactively, when setting the parameters `par.readX`=1 in the routine `image_pair_set_parameters.m`, line 34. Further figures show the selected windows, the noise standard deviations and the change of the estimated image windows.

6.3 Demo `demo_LSM_simulated.m`

The demo routine `demo_LSM_simulated.m` is meant to check the correctness of the implementation based on simulated data. It allows to monitor the individual iterations for a single case, or to statistically test, whether the resultant parameters and their covariance matrix are coherent with the theoretical values, the true values of the parameters and the theoretical covariance matrix derived by the estimation procedure (Cramer-Rao bound).

6.3.1 Control parameters

We have the following options, which can be set in the main routine:

- Choosing whether the random sequence is pre-specified or randomly initiated:
  variable `init_rand`
- Choosing the number of samples for checking the covariance matrix and for bias:
  variable `N_samples`
- Choosing between three artificially generated images and taking a window of a given image as reference:
  variable `type_data`

The true transformations and the true images are generated. Within a loop, the true images are perturbed by Gaussian noise and rounded to integers. The `N_samples` are used to test for the correctness of the estimation. In addition we have the following options, which can be set in the routine `simulated_set_parameters.m`:

- Choosing the approximate window size of the overlapping image:
  variable `Nh`
- Choosing whether a test on swapping the two images is to be performed:
  variable `test_symmetry`

- Choosing the geometric and radiometric transformation:
  variables `A_true` and `R_true`
- Choosing the smoothing kernel σ_s:
  variable `sigma_smooth`
- Choosing the maximum number of iterations:
  variable `max_iter`
- Choosing the significance number S of the statistical tests:
  variable `S`

6.3.2 Output

The output is different, when looking into the individual iterations (`N_samples` $= 1$) or when checking the implementation (`N_samples` > 9).

The individual iterations. When analysing the individual iterations for a single sample (`N_samples` $= 1$), the command window shows the following information

- Document of the control parameters
- Per iteration the the number of observations, `N` and the estimated $\widehat{\sigma}_0$ as `sigma_0_est`. If $\widehat{\sigma}_0$ is not close to 1, this indicates, the assumed model does not fit to the observations. The cause of this effect cannot be given: it may be, that the scene is not flat, the windows are too large, the estimated noise variance deviates from the noise variance in the windows, the shadow situation in both images is different, and so on.
- The final result is characterized by the estimated transformations `A_est` and `R_est`.
- A warning is given, if the maximum number of iterations is reached.
- If the symmetry of the solution is tested, the checks $\widehat{\mathsf{A}} \cdot \widehat{\mathsf{A}^{-1}} - I_3$ and $\widehat{\mathsf{R}} \cdot \widehat{\mathsf{R}^{-1}} - I_2$ are provided as `check_symmetry_AAi_I` and `check_symmetry_RRi_I`

In addition the following figures are provided

- the true image $f(\boldsymbol{x})$ (large black box in Fig. 2)
- the true mean, left, and right images $f(\boldsymbol{x})$, $g(\boldsymbol{y})$ and $h(\boldsymbol{z})$, respectively (the small black box,and blue and the green box in Fig. 2)
- the noisy image window pair
- for each iteration, left and the right image warped into the x-coordinate system, i.e., $f(\boldsymbol{y})$ and $f(\boldsymbol{z})$ (the part of the blue and the green parallelograms lying within the small black box in Fig. 2)[5].

[5]The size of the image patches depends on the maximum number of iterations.

Checking the implementation. When checking the implementation, thus N_samples > 9, the command window shows the following informations

- Document of the control parameters
- Monitoring the samples
- The number of cases, where the maximum number of iterations is reached is documented as a warning. These are not used for the following analysis.
- The result of the statistical tests for all 8 parameters and only the 6 geometric parameters:
  1. test whether the mean of the estimated variance factor deviates from 1,
  2. test whether the empirical covariance matrix $\widehat{\Sigma}_{\widehat{\theta}\widehat{\theta}}$ of the parameters, derived from the estimates $\widehat{\boldsymbol{\theta}}$, coincides with the theoretical covariance matrix $\Sigma_{\widehat{\theta}\widehat{\theta}}$ from the inverse normal equation matrix,
  3. test whether the mean of the estimated parameters is identical to the true (simulated) value,

  see Förstner and Wrobel (2016, Chapt. 4.6.8). Test statistics lying in the rejection region are indicated with `*****`. Actually, the non-rejection region is given.
- For each parameter, the theoretical and the empirical standard deviation, their ratio, the mean, the standard deviation, and the maximum bias.
- The average standard deviation of the parameters of the affinity, the translation and the radiometric transformation.
- Information about the CPU time.

In addition there are figures showing a noisy sample image window pair, the histograms of the estimated variance factors (twice) and the number of iterations.

6.4 Demo `demo_LSM_image_pairs.m`

The routine is meant to apply the matching routine to real data. The user may choose to interactively measure the correspondences or read the previously measured data from file in folder `Images`. For each image pair the correspondences are stored in a `mat`-file in the folder `Data/ImageCoordinates`. Color images are converted to black and white images.

The user is asked to identify two corresponding points together with a scale σ and orientation, mimicking the output of the Lowe-detector. The sequence of actions is the following. For each image

- identify an approximate position in the image,
- a blow-up of the surrounding point is provided,

- the first point to be measured is the centre of the window,
- the second point to be measured provides *three times* the dominant scale σ and the direction.

Windows of size $8\sigma \times 8\sigma$ around the measured keypoints are used for matching. The signal dependent noise variances for both images are estimated from a larger neighbourhood ($\leq 200 \times 200$ pixels) and used for defining the weights of the intensities.

6.5 Error messages and convergence

The following error messages may occur:

- `Geometric affinity is not positive definite`. The approximate affinity must be represented by a positive definite 2×2 matrix A. No mirroring is allowed.
- `Overlap is too small`. The overlap of the two images in each iteration must lead to a square window of at least 9×9.

In both cases the output parameter `Red` of the main-routine is negative.

Convergence is guaranteed if the number `N_iter` of used iterations is smaller than the maximum number `max_iter` of iterations. If the number of used iterations is identical to the maximum number of iterations, and the maximum relative change `max_ratio`$= \max_u(||\widehat{\Delta\theta}_u^{(\nu)}||/\sigma_{\widehat{\theta}})$ of the parameters in the last iteration is smaller than 1, then convergence can be assumed. Generally, there is no guarantee, that the global optimum is reached.

7 Timing

The time mainly depends on the size of the images, i.e., the average number N of pixels in the two images. On an Lenovo X220 with Matlab 2018 we have found the following approximate relation between the number of pixels and the time per iteration, depending on whether the warping function of MATLAB is used or the design matrix is built up using loops on the individual pixels:

$$t_{\text{warping}}\ [\text{ms}] = (0.0047\ N + 12)\ [\text{ms}] \quad \text{and} \quad t_{\text{loop}}\ [\text{ms}] = (0.050\ N + 6.2)\ [\text{ms}]\,. \quad (83)$$

Hence, when using the warping function of MATLAB, the computing time takes below 0.05 milliseconds/pixel. With usually 3 iterations, windows of 40×40 can be matched in less than 0.1 seconds.

8 Appendix

8.1 Bi-cubic interpolation

Bi-cubic interpolation is required for estimating the true underlying function f from the observed images g and h. This interpolation induces errors, which we take into account

when specifying the variances of the observed images. Therefore, we analyse the effect of this interpolation onto a signal and derive the relations between the original and the interpolated signal as a basis for variance propagation.

8.1.1 1D cubic interpolation

Compact representation. Here we follow Shu (2013). For each value x in the interval $[i, i+1]$ the function is a cubic polynomial which satisfies the following conditions

1. The function at i has the values $f(i) = b_i$
2. The function at $i+1$ has the value $f(i+1) = b_{i+1}$.
3. The derivative at i is $f'(i) = (b_{i+1} - b_{i-1})/2$.
4. The derivative at $i+1$ is $f'(i) = (b_{i+2} - b_i)/2$.

Hence, we need the four neighbouring values collected in

$$\boldsymbol{b} = \begin{bmatrix} b_{i-1} \\ b_i \\ b_{i+1} \\ b_{i+2} \end{bmatrix} . \tag{84}$$

We use the substitution

$$\boldsymbol{u}(x) = \begin{bmatrix} 1 \\ (x-i) \\ (x-i)^2 \\ (x-i)^3 \end{bmatrix} \quad \text{with} \quad i = \lfloor x \rfloor \, . \tag{85}$$

Then with

$$M_0 = \frac{1}{2} \begin{bmatrix} 0 & 2 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 2 & -5 & 4 & -1 \\ -1 & 3 & -3 & 1 \end{bmatrix} . \tag{86}$$

the interpolated value is

$$f(x) = \boldsymbol{u}^\mathsf{T}(x) M_0 \boldsymbol{b} \, , \tag{87}$$

as above.

Proof: We use

$$\boldsymbol{a} = \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} . \tag{88}$$

Then cubic function in the i-th interval $[i, i+1]$ can be written as

$$f^{(i)}(u) = a_0 + a_1(u-i) + a_2(u-i)^2 + a_3(u-i)^3 = \boldsymbol{u}^\mathsf{T} \boldsymbol{a} \tag{89}$$

The derivative is

$$f_u^{(i)}(u) = a_1 + 2a_2(u-i) + 3a_3(u-i)^2 = \boldsymbol{u} \begin{bmatrix} a_1 \\ 2a_2 \\ 3a_3^2 \\ 0 \end{bmatrix} = \boldsymbol{u}^\mathsf{T} D \boldsymbol{a} \quad \text{with} \quad D = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 3 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{90}$$

The four conditions then can be written as

$$\begin{bmatrix} f^{(i)}(i) \\ f^{(i)}(i+1) \\ f_u^{(i)}(i) \\ f_u^{(i)}(i+1) \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -1/2 & 0 & 1/2 & 0 \\ 0 & -1/2 & 0 & 1/2 \end{bmatrix}}_{U} \begin{bmatrix} b_{i-1} \\ b_i \\ b_{i+1} \\ b_{i+2} \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & 3 \end{bmatrix}}_{V} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} \tag{91}$$

or compactly

$$\boldsymbol{c} = U\boldsymbol{b} = V\boldsymbol{a}\,. \tag{92}$$

Therefore

$$\boldsymbol{a} = V^{-1}U = M_0\boldsymbol{b}\,, \tag{93}$$

which holds since

$$U = VM_0 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -1/2 & 0 & 1/2 & 0 \\ 0 & -1/2 & 0 & 1/2 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & 3 \end{bmatrix} \begin{bmatrix} 0 & 2 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 2 & -5 & 4 & -1 \\ -1 & 3 & -3 & 1 \end{bmatrix}. \tag{94}$$

◇

Remark: This definition of interpolating cubic splines does not minimize the total curvature of the interpolating function, thus differs from the classical definition. In contrast to the classical definition, the function values $f(u)$ only depend on four neighbouring points b_i linearly, not on all values of the profile. ◇

Since, with $\boldsymbol{a} = M_0\boldsymbol{b}$ the first derivative of the polynomial from (90) we obtain the compact expression for the derivative

$$\boxed{f'(x) = \boldsymbol{u}^\mathsf{T}(x) M_1 \boldsymbol{b}} \tag{95}$$

with

$$M_1 = DM_0 = \frac{1}{2} \begin{bmatrix} -1 & 0 & 1 & 0 \\ 4 & -10 & 8 & -2 \\ -3 & 9 & -9 & 3 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{96}$$

Variance of cubic interpolation. We now give the variances of cubic interpolation. This is the uncertainty of the interpolated values assuming the given values are uncertain and bi-cubic interpolation is the correct model.

Cubic interpolation at $r \in [0,1]$ requires the vales of g at $[-1,0,1,2]$. Specifically we obtain the interpolated value

$$\begin{aligned} g(r) &= \frac{1}{2}\left[\left(-r+2\,r^2-r^3\right)g_{-1}\right. && (97)\\ &\quad +\left(2-5\,r^2+3\,r^3\right)g_0 && (98)\\ &\quad +\left(r+4\,r^2-3\,r^3\right)g_1 && (99)\\ &\quad \left.+\left(-r^2+r^3\right)g_2\right]\,, && (100) \end{aligned}$$

see (86). Assuming homogeneous noise variance, we obtain the variance

$$\sigma_{\overline{n}}^2(r) = q_{\text{cubic}}^2(r)\sigma_n^2 \tag{101}$$

with

$$\text{with}\quad q_{\text{cubic}}^2(r) = \frac{1}{2}\left[2-9\,r^2+8\,r^3+21\,r^4-30\,r^5+10\,r^6\right]\,. \tag{102}$$

It is symmetric w.r.t. $r=1/2$. It reaches its its maximum $\sigma_x^2(0)=\sigma_n^2$ at $r=0$ and $r=1$ and its minimum at $r=1/2$ min

$$\sigma_{\overline{n}}^2(r=1/2) = \frac{41}{64}\sigma_n^2 \approx 0.641\sigma_n^2\,. \tag{103}$$

We may derive an individual variance as a function of the remainder $r = x - \lfloor x \rfloor$. Furtheron, we also can use the average variance, which is

$$\overline{\sigma_{\overline{n}}^2} = \int_{r=0}^{1}\sigma_x^2 \mathrm{d}r = \frac{57}{70}\sigma_n^2 \approx 0.814\sigma_n^2\,. \tag{104}$$

Using this mean value, we have an error of approximately 13% in the variance.

8.1.2 2D cubic interpolation

We thus obtain *bi-cubic interpolation* using the substitution

$$\boldsymbol{v}(y) = \begin{bmatrix} 1 \\ v \\ v^2 \\ v^3 \end{bmatrix} \quad \text{with} \quad v = y - \lfloor y \rfloor \tag{105}$$

and the collection of the 4×4 neighbouring values in the cell $[i,i+1]\times[j,j+1]$

$$B = \begin{bmatrix} b_{i-1,j-1} & b_{i-1,j} & b_{i-1,j+1} & b_{i-1,j+2} \\ b_{i,j-1} & b_{i,j} & b_{i,j+1} & b_{i,j+2} \\ b_{i+1,j-1} & b_{i+1,j} & b_{i+1,j+1} & b_{i+1,j+2} \\ b_{i+2,j-1} & b_{i+2,j} & b_{i+2,j+1} & b_{i+2,j+2} \end{bmatrix}\,, \tag{106}$$

We obtain

$$f(x,y) = \boldsymbol{u}^{\mathsf{T}}(x)M_0 B M_0^{\mathsf{T}}\boldsymbol{v}(y)\,. \tag{107}$$

The partial derivatives then are

$$\boxed{f_x(x,y) = \boldsymbol{u}^{\mathsf{T}}(x) M_1 B M_0^{\mathsf{T}} \boldsymbol{v}(y) \quad \text{and} \quad f_y(x,y) = \boldsymbol{u}^{\mathsf{T}}(x) M_0 B M_1^{\mathsf{T}} \boldsymbol{v}(y)} \tag{108}$$

The variance of bi-cubic interpolated values is

$$\sigma_{\overline{n}}(x,y) = q^2_{\text{bi-cubic}}(r,s)\sigma_n^2 \quad \text{with} \quad q^2_{\text{bi-cubic}}(r,s) = q^2_{\text{cubic}}(r) q^2_{\text{cubic}}(s)\,. \tag{109}$$

The average variance is

$$\overline{\sigma^2_{\overline{n}}} = \int_{r=0}^{1}\int_{s=0}^{1} \sigma^2_{\overline{n}}(x,y)\mathrm{d}x\mathrm{d}y = \frac{57^2}{70}\,\sigma_n^2 \approx 0.663\sigma_n^2\,. \tag{110}$$

Interpolation error. We want to determine the interpolation error of a function $f(x,y)$. Here we assume the data are fixed, i.e., not contaminated by random errors, and the interpolation leads to erroneous results, since the interpolation rule may be different. Since the true interpolation rule is unknown, we perform two bi-cubic interpolations, and compare the result with the original function.

We do this in three steps, see Fig. 3:

1. Interpolating the function f at the grid at $[i+1/2, j+1/2]$:

$$g(x,y) = f_B(x+1/2, y+1/2)\,. \tag{111}$$

2. Interpolating the function g at the grid at $[i-1/2, j-1/2]$:

$$h(x,y) = g_B(x-1/2, y-1/2)\,. \tag{112}$$

3. Determining the error induced by the two interpolations

$$\sigma^2 = \mathbb{D}(h(x,y) - f(x,y))\,. \tag{113}$$

We start from (107) using

$$u = u(x) - i \quad \text{and} \quad v = v(x) - j\,, \tag{114}$$

with the special choice for $x = +1/2$

$$u_+ = x - \lfloor x \rfloor = +1/2 - 0 = 1/2 \quad \text{and} \quad v_+ = 1/2\,. \tag{115}$$

Hence we have

$$\boldsymbol{u} = \begin{bmatrix} 1 \\ 1/2 \\ 1/4 \\ 1/8 \end{bmatrix} \quad \text{and} \quad \boldsymbol{v} = \begin{bmatrix} 1 \\ 1/2 \\ 1/4 \\ 1/8 \end{bmatrix} \tag{116}$$

We refer to the 49 values of $F(1:7, 1:7)$, see Fig. 3 This allows to derive $G(1:4, 1:4)$ via

$$g(i,j) = \boldsymbol{u}^{\mathsf{T}} M_0 F(i:i+3, j:j+3) M_0^{\mathsf{T}} \boldsymbol{v}\,. \tag{117}$$

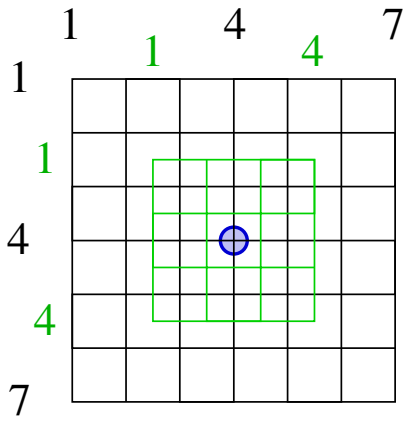

Figure 3: Interpolation error. The black 7×7 grid of f is used to interpolate the green 4×4 grid of g. This is used to obtain the interpolated value h at the position of $f(4,4)$. The difference is an indication for double the interpolation error.

Similarly, starting from the interpolated signal g, we can derive the backshifted value

$$h = \boldsymbol{u}^{\mathsf{T}} M_0 G M_0^{\mathsf{T}} \boldsymbol{v} \,. \tag{118}$$

The difference

$$\boxed{\delta_2(F) = h(F) - f(4,4)} \tag{119}$$

is a function of the 49 values of F and represents twice the interpolation error.

We now determine the standard deviation of the interpolation. First, the Jacobian $J_2 = \partial \delta_2 / \partial F$ is independent of f and given by

$$J_2 = \frac{1}{2^{16}} \begin{bmatrix} 1 & -18 & 63 & 164 & 63 & -18 & 1 \\ -18 & 324 & -1134 & -2952 & -1134 & 324 & -18 \\ 63 & -1134 & 3969 & 10332 & 3969 & -1134 & 63 \\ 164 & -2952 & 10332 & -38640 & 10332 & -2952 & 164 \\ 63 & -1134 & 3969 & 10332 & 3969 & -1134 & 63 \\ -18 & 324 & -1134 & -2952 & -1134 & 324 & -18 \\ 1 & -18 & 63 & 164 & 63 & -18 & 1 \end{bmatrix} \tag{120}$$

We now assume that the signal values f are correlated, depending on the underlying power-spectrum. We assume three different cases

1. Gaussian power spectrum. Then the covariance function is

$$C(d) = \exp(-(d/d_0)^2/2) \,. \tag{121}$$

2. Laplacian power spectrum. Then the covariance function is

$$C(d) = \frac{1}{1 + (d/d_0)^2} \,. \tag{122}$$

3. The powers spectrum follows the power law, e.g., in the form $P(u) \propto f^{-2}$, or $P(u) = 1/(1+u^2)/\pi$. Then the covariance function has the form

$$C(d) = \exp(-d/d_0)\,. \tag{123}$$

The value d_0 controls the smoothness of the signal, larger d_0 leads to smoother functions. Via variance propagation we obtain $\mathbb{V}(\delta_2(f))$. Since this difference results from two interpolations we report

$$\boxed{\sigma_\delta = \sqrt{\frac{\mathbb{V}(\delta_2)}{2}}} \tag{124}$$

in the table. We observe:

d_0	P(Gauss)	P(Lapl)	P(Power)
1.0000	0.1855	0.2888	0.3499
1.5000	0.0643	0.1860	0.2947
2.0000	0.0249	0.1180	0.2583
2.5000	0.0112	0.0754	0.2323
3.0000	0.0056	0.0491	0.2128
3.5000	0.0031	0.0327	0.1974
4.0000	0.0019	0.0222	0.1849

Table 1: Standard deviation σ_δ of bi-cubic interpolation error for different types of correlations and correlation widths d_0 error

1. the larger d_0, i.e., the smoother the signal, the smaller the interpolation error is.

2. As to be expected, signals with Laplacian power spectrum are rougher than those with Gaussian, and smoother than those with the power spectrum following the power law.

These results refer to the shift $[1/2, 1/2]$ and depend on the assumed stochastical model for the signal. For the use in LSM, we do not want a dependency on the stochastical model for the unknown image function. Since $\boldsymbol{J}_2 = \partial\delta_2/\partial\boldsymbol{F}$, we can use the matrix $\boldsymbol{J}_2$ as filter for deriving δ_2 from f:

$$\boxed{\delta_2 = \boldsymbol{J}_2 * f\,.} \tag{125}$$

Obviously, the matrix $\boldsymbol{J}_2$ represents a highpass filter. Hence, we can determine the individual interpolation errors due to forward and backward shifting with interpolation from an arbitrary signal. This allows us to derive the variance of the maximal interpolation error from

$$\boxed{\widehat{\sigma}^2_{\delta,\max} = \tfrac{1}{|\mathcal{R}|}\textstyle\sum_{r\in\mathcal{R}} \delta_r^2\,.} \tag{126}$$

Since we are interested in the average interpolation error within the region of one pixel, we determine the mean of the expected variance $\overline{\sigma^2_\delta}$ for all forward and backward shifts

$[x,y], x,y \in [0,1]$. Hence we can determine the interpolation variance for a specific window f from

$$\widehat{\overline{\sigma_\delta^2}} = k\, \widehat{\sigma}^2_{\delta,\max} \quad \text{with} \quad k = \frac{\overline{\sigma_\delta^2}}{\sigma^2_{\delta,\max}}\,. \tag{127}$$

As an example we have the variances of the interpolation error for $d_0 = 1$ and Gaussian covariance function at the grid points $[i,j]/8$, $i,j \in \{0,...,8\}$ of the unit interval:

$$[\sigma_{ij}^2] = \frac{1}{1000}\begin{bmatrix} 0 & 9 & 93 & 247 & 327 & 247 & 93 & 9 & 0 \\ 9 & 22 & 111 & 267 & 349 & 267 & 111 & 22 & 9 \\ 93 & 111 & 208 & 368 & 449 & 368 & 208 & 111 & 93 \\ 247 & 267 & 368 & 527 & 608 & 527 & 368 & 267 & 247 \\ 327 & 349 & 449 & 608 & 688 & 608 & 449 & 349 & 327 \\ 247 & 267 & 368 & 527 & 608 & 527 & 368 & 267 & 247 \\ 93 & 111 & 208 & 368 & 449 & 368 & 208 & 111 & 93 \\ 9 & 22 & 111 & 267 & 349 & 267 & 111 & 22 & 9 \\ 0 & 9 & 93 & 247 & 327 & 247 & 93 & 9 & 0 \end{bmatrix} \tag{128}$$

As to be expected, the interpolation error is zero in the corners of the square, thus for integer coordinates.

Interestingly, the factor k does not vary much for different stochastical models of f, as Tab. 2 shows.

d_0	P(Gauss)	P(Lapl)	P(Power)
1.0000	0.3482	0.3710	0.3803
1.5000	0.3425	0.3608	0.3793
2.0000	0.3432	0.3536	0.3789
2.5000	0.3446	0.3490	0.3787
3.0000	0.3459	0.3462	0.3786
3.5000	0.3468	0.3445	0.3785
4.0000	0.3476	0.3437	0.3785

Table 2: Ratio k of mean and maximal bi-cubic interpolation error variance for different stochastical models for a signal.

Therefore, we can use the following variances of the model error of the given data, e.g., for g in graylevels $[0,...,255]$

$$\boxed{\sigma^2_{n'_j} = \sigma^2_{n_j} + k\, \widehat{\mathbb{V}}(J_2 * g) \quad \text{with} \quad k = 0.38\,.} \tag{129}$$

depending on the empirical noise variance $\sigma^2_{n_j}$ of the intensities of g_i and the interpolation error derived from the image window.

8.2 Noise Variance Estimation

The following section is taken from Förstner (2000)

The noise variance needs to be estimated from images. There are three possible methods to obtain such estimates:

1. *Repeated images*: Taking multiple images of the same scene without changing any parameters yields repeated images. This allows to estimate the noise variance for each individual pixel independently. This certainly is the optimal method in case no model for the noise characteristics is available and can be used as a reference.

   The method is the only one which can handle the case where there is no model for the noise characteristics.

   The disadvantage of this method is the need to have repeated images, which, e. g. in image sequences is difficult to achieve.

2. *Images of homogeneous regions:* Images of homogeneous regions, thus regions with piecewise constant or linear signal, allows to estimate the noise variance from one image alone.

   The disadvantage is the requirement for the segmentation of the images into homogeneous regions. Moreover, it is very difficult to guarantee the constancy or linearity of the true intensity image within the homogeneous regions. Small deviations from deficiencies in the illumination already jeopardize this method.

   The method is only applicable in case the noise only depends on the signal.

3. *Images with little texture:* Images with a small percentage of textured regions allow to derive the noise variance from the local gradients or curvature. For the larger part of the image they can be assumed to have approximately zero mean. Thus presuming a small percentage of textured regions assumes the expectation of the gradient or the curvature in the homogeneous regions to be negligible compared to the noise.

   Also this method is only applicable in case the noise characteristics is only depending on the signal.

We want to describe this method in more detail. We first discuss the method for intensity images. The generalization to range images is straight forward.

8.2.1 Estimation of a constant noise variance in intensity Images

The idea is to analyze the histogram of the gradient magnitude of the image in the area where there are no edges and no texture. The procedure given here is similar to that proposed in Förstner (1991).

We now need to specify the model for the ideal image f. We assume that a significant portion $\mathcal{H}$ of the image area $\mathcal{I}$ is homogeneous, thus shows locally constant intensity, thus $\mu_f = const.$. Adopting notions from statistical testing $H_0 = (r, c) \in \mathcal{H}$ is the null-hypothesis, i. e. the hypothesis a pixel belongs to a homogeneous region. Thus

$$\mathsf{E}(\nabla \underline{f} | H_0) = \mathbf{0} \tag{130}$$

The other area $\mathcal{I} - \mathcal{H}$ covers edges and textured areas with significantly larger gradients.

Then, as to be shown, the histogram of the homogeneoity measure $h = |\nabla g|$ shows exponential behavior in its left part representing the noise in the image and arbitrary behavior in the right part representing the edges:

We assume the intensities to be Gaussian distributed with fixed mean and random noise. Assuming the simple gradient kernels

$$\left(\frac{\partial}{\partial r}\right)_0 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix} \qquad \left(\frac{\partial}{\partial c}\right)_0 = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & -1 \\ 0 & 0 & 0 \end{bmatrix} \tag{131}$$

neglecting the scaling factores $1/2$, we obtain the gradient

$$\nabla g = \begin{bmatrix} g_r \\ g_c \end{bmatrix} = \begin{bmatrix} g_{r+1,c} - g_{r-1,c} \\ g_{r,c+1} - g_{r,c-1} \end{bmatrix} \tag{132}$$

which is Gaussian distributed with covariance matrix

$$\mathsf{D}\left(\begin{matrix} \underline{g}_r \\ \underline{g}_c \end{matrix} \middle| H_0 \right) = \sigma_{n'}^2 \boldsymbol{I} \tag{133}$$

Here we use the convention

$$\sigma_{n'}^2 = \sigma_{n_r}^2 = \sigma_{n_c}^2 \tag{134}$$

which in general is given by

$$\sigma_{n'}^2 = \int_{x,y} G_{x;s}^2(x,y) dx\, dy = \frac{1}{8\pi s^4}\sigma_n^2, \quad \text{or} \quad \sigma_{n'}^2 = \sum_{r,c} \partial_r^2(r,c)\, \sigma_n^2 \tag{135}$$

see Förstner (2000, Eq. (15)). In our case eq. (131) leads to

$$\sigma_{n'}^2 = 2\sigma_n^2 \tag{136}$$

The squared gradient magnitude measures the homogeneity h

$$h_\nabla(r,c) = |\nabla g(r,c)|^2 = g_r^2(r,c) + g_c^2(r,c) \tag{137}$$

It is the sum of two squares of Gaussian variables.

In case the mean $\mu_g = \mu_f$ of $\underline{g}$ is constant in a small region, thus the model eq. (130) holds, the squared gradient magnitude is χ_2^2 or exponentially distributed with density function (neglecting the index ∇ for simplicity)

$$p(h|H_0) = \frac{1}{\mu_h}\, \mathrm{e}^{-\dfrac{h}{\mu_h}} \tag{138}$$

and mean

$$\mathsf{E}(\underline{h}|H_0) = \mu_h = 4\sigma_n^2 \tag{139}$$

Therefore we are able to estimate the parameter μ_h from the empirical density function in the following way:

1. Set the iteration index $\nu = 0$. Specify an approximate value $\sigma_n^{(0)}$ for the noise standard deviation. Use $\mu_h^{(0)} = 4\sigma_n^{2(0)}$ as approximate value for the gradient magnitude.

2. Determine all $h(r, c)$

3. Take the mean $m^{(\nu)}$ of all values $h(r, c) < \mu_h^{(\nu)}$. Its expected value is given by

$$\mu_m^{(\nu)} = \frac{\int_{h=0}^{\mu_h^{(\nu)}} hp(h|H_0)dh}{\int_{h=0}^{\mu_h^{(\nu)}} p(h|H_0)dh} = \frac{\mathrm{e}-2}{\mathrm{e}-1}\,\mu_h^{(\nu)} \tag{140}$$

   in case the edges or textured areas do not significantly contribute to this mean. Thus a refined estimate $\mu_h^{(\nu+1)}$ for μ_h is given by:

$$\mu_h^{(\nu+1)} = \frac{\mathrm{e}-1}{\mathrm{e}-2}\,m^{(\nu)} \approx 2.392\,m^{(\nu)} \tag{141}$$

4. Set $\nu = \nu + 1$ and repeat step 3.

Usually, only two iterations are necessary to achieve convergence. A modification would be, to take the median of the values $h(r, c)$ as a robust estimate and compensate for the bias caused 1) by taking the median instead of the mean and 2) by the edge pixels (see Brügelmann and Förstner (1992)).

This procedure can be applied to every channel in a multi channel image, especially in color images or in gradient images of range images.

8.2.2 Estimating a general general noise variance function

In case the noise variance is not constant over the whole image area and can be assumed only to depend on the intensity, we need to parameterize the noise variance function $\sigma_n^2 = s(g)$ in some way.

The easiest possibility is to assume it to be continuous. Then we can partition the range $[0..G]$ of all intensities g into intervals $I_\gamma, \gamma = 1..\Gamma$ and assume the noise variance to be constant in each interval.

Thus we repeat the procedure of subsection 8.2.1 for each intensity interval under the condition $g \in I_\gamma$.

The choice of the intervals obviously requires some discussion, as it may significantly influence the solution. Taking a set of constant intervals may lead to intervals where no intensities belong to, even in case one would restrict to the real range $[g_{\min}, g_{\max}]$. Therefore the intervals should be chosen such that

1. they contain enough intensity values..

   The number should be larger than 100 in order to yield precise enough estimates for the noise variances, which in this case has a relative (internal) accuracy better

than 10 %. The number of intervals should be chosen in dependency of the expected roughness of $s(g)$. For aerial images we have made good experiences with intervals between 1 and 8 grey values on image patches of 300 × 300 pixels (cf. Waegli (1998)).

2. they contain an equal number of intensities. This may easily be achieved by using the histogram of the intensities.

8.3 Checking the Implementation of the Estimation

This section has been taken from Förstner and Wrobel (2016, Sect. 4.6.8)

Before using the implementation of an estimation procedure we need to check whether it yields correct results. This refers to (1) the estimated parameters, (2) their covariance matrix, and (3) the estimated variance factor. The estimated parameters should be unbiased, the covariance matrix should reflect the sensitivity of the estimated parameters w.r.t. random perturbations of the observations, characterized by the stochastical model, especially the covariance matrix of the observations; and the estimated variance factor should not significantly deviate from 1.

If the implementation is correct, small perturbations in the observations following the stochastical model should lead to small perturbations in the variance factor and in the estimated parameters, where they also should follow the predicted covariance matrix. In the case of larger perturbations, effects of the linearization of a nonlinear model will be visible.

Such an evaluation is based on simulated data, since we then have access to the true values. This also has the advantage that no access to the source code is necessary; the check can be based on the output $\{\widehat{\boldsymbol{\theta}}, \Sigma_{\widehat{\theta}\widehat{\theta}}, \widehat{\sigma_0}^2\}$.

Based on given true values $\tilde{\boldsymbol{\theta}}$ for the parameters, a given observational design, represented by the function $\boldsymbol{f}(\boldsymbol{\theta})$ and a stochastical model $\mathbb{D}(\underline{\boldsymbol{y}}) = \Sigma_{yy}$, we can simulate K samples of observations $\boldsymbol{y}_k$ from

$$\underline{\boldsymbol{y}}_k = \boldsymbol{f}(\tilde{\boldsymbol{\theta}}) - \underline{\boldsymbol{v}}_k\,,\ k = 1, ..., K \qquad \underline{\boldsymbol{v}} \sim \mathcal{N}(\mathbf{0}, \Sigma_{yy})\ , \tag{142}$$

leaving the model $\{\boldsymbol{f}(\boldsymbol{\theta}), \Sigma_{yy}\}$ and the true parameters $\tilde{\boldsymbol{\theta}}$ fixed.

The estimation leads to K vectors $\widehat{\boldsymbol{\theta}}_k$ of estimated parameters, to K estimates $\widehat{\sigma}^2_{0k}$ of the variance factor, and – provided the relative accuracy $\sigma_y/\mathbb{E}(y)$ of the observations is below 1% – a sufficiently accurate covariance matrix $\Sigma_{\widehat{\theta}\widehat{\theta}}$. In order to be able to check the validity of the model a sufficiently large number K of samples is necessary, which should be larger than the number of elements of the largest covariance matrix which is to be checked.

In the case of Gaussian noise, the evaluation can be based on well-established statistical tests. If one of these tests fails, there are good reasons to doubt whether the program code is a reliable realization of the envisaged estimation model. However, without further tests there are no clues to the source of the discrepancy; it may be the implementation of the envisaged model or of the simulation. This may require more detailed testing.

We now discuss three tests concerning the noise level, the bias, and the validity of the theoretical covariance matrix. They should be performed on a set of representative estimation tasks before using the estimation procedure in a real application.

8.3.1 Correctness of the Estimated Noise Level

The correctness of the estimated noise level can be reduced to check the validity of the variance factor. The validity of the estimated variance factor can be based on the mean of the K variance factors derived from the K simulations,

$$s^2 = \frac{1}{K}\sum_{k=1}^{K}\widehat{\sigma}_{0k}^2\,. \tag{143}$$

When the implemented model, which is the null hypothesis H_0, holds, the test statistic

$$F = \frac{s^2}{\sigma_0^2}\,, \quad \underline{F}|H_0 \sim F_{KR,\infty} \tag{144}$$

is Fisher distributed with KR and ∞ degrees of freedom, where R is the redundancy of the estimation task. If for a specified significance level S, the test statistic $F > F_{KR,\infty;S}$, then the estimated variance factor indicates deviations from the assumed model – possibly caused by implementation errors. In this case, it might be useful to analyse the histogram in order to find possible sources of the deviations.

Observe, this test does not require the theoretical covariance matrix $\Sigma_{\widehat{\theta}\widehat{\theta}}$ of the estimated parameters.

8.3.2 Correctness of the Covariance Matrix

To make sure we can rely on the theoretical covariance matrix provided by the implemented estimation procedure, we compare it with the empirical covariance matrix of the simulation sample. It is given by

$$\widehat{\Sigma} = \frac{1}{K-1}\sum_{k=1}^{K}(\widehat{\boldsymbol{\theta}}_k - \widehat{\boldsymbol{m}}_{\widehat{\theta}})(\widehat{\boldsymbol{\theta}}_k - \widehat{\boldsymbol{m}}_{\widehat{\theta}})^{\mathsf{T}} \tag{145}$$

with the estimated mean

$$\widehat{\boldsymbol{m}}_{\widehat{\theta}} = \frac{1}{K}\sum_{k=1}^{K}\widehat{\boldsymbol{\theta}}_k\,. \tag{146}$$

When the model holds as implemented and the theoretical precision $\Sigma_{\widehat{\theta}\widehat{\theta}}$ is correct, the test statistic

$$\underline{X}^2 = (K-1)\left[\ln\left(\det\Sigma_{\widehat{\theta}\widehat{\theta}}/\det\underline{\widehat{\Sigma}}\right) - U + \operatorname{tr}\left(\underline{\widehat{\Sigma}}\Sigma_{\widehat{\theta}\widehat{\theta}}^{-1}\right)\right] \sim \chi^2_{U(U+1)/2} \tag{147}$$

is approximately χ^2-distributed with $U(U+1)/2$ degrees of freedom (cf. Koch, 1999, Eq. (2.205)). If for a prespecified significance level S the test statistic X^2 is larger than $\chi^2_{U(U+1)/2,S}$, then there is reason to assume the theoretical covariance matrix, as

it results from the implemented model, does not reflect the covariance matrix of $\widehat{\underline{\boldsymbol{\theta}}}$ sufficiently well. In this case, it might be useful to visualize the covariance matrix in order to identify possible causes for the found deviation.

It is sufficient to take one of them as reference, though the theoretical covariances of the K samples vary slightly, as the variance propagation is performed not using the true mean, but the estimated parameters. However, as the relative size of this variation is a second-order effect, it can be neglected.

8.3.3 Bias in the Estimates

To check the unbiasedness of the estimated parameters we determine their empirical mean.

If the mathematical model holds, the implementation is correct, and higher-order terms during linearization are negligible; the estimated mean of the estimated parameters is Gaussian distributed according to

$$\widehat{\underline{\boldsymbol{m}}}_{\widehat{\theta}} \sim \mathcal{N}\left(\tilde{\boldsymbol{\theta}}, \frac{1}{K}\boldsymbol{\Sigma}_{\widehat{\theta}\widehat{\theta}}\right) . \tag{148}$$

Under these conditions, the test statistic, the Mahalanobis distance,

$$\underline{X} = K(\widehat{\underline{\boldsymbol{m}}}_{\widehat{\theta}} - \tilde{\boldsymbol{\theta}})^{\mathsf{T}}\, \boldsymbol{\Sigma}_{\widehat{\theta}\widehat{\theta}}^{-1}\, (\widehat{\underline{\boldsymbol{m}}}_{\widehat{\theta}} - \tilde{\boldsymbol{\theta}}) \sim \chi^2_U\,, \tag{149}$$

is χ^2-distributed with U degrees of freedom. If $X > \chi^2_{U,S}$ for the test statistic and a prespecified significance level S, we have reasons to reject the hypothesis that the model, including the approximations, actually holds as implemented. In this case it might be useful to visualize the bias in order to find possible causes for the rejection of the model.

If these statistical tests are passed on a set of representative simulation data sets, the statistical tests, when applied to real data, can be used as diagnostic tools for identifying discrepancies between the data and the assumed mathematical model.

References

Barath, D., M. Polic, W. Förstner, T. Sattler, T. Pajdla, and Z. Kukelova (2020a). Making Affine Correspondences Work in Camera Geometry Computation. In *Proc. of ECCV*. 14

Barath, D., M. Polic, W. Förstner, T. Sattler, T. Pajdla, and Z. Kukelova (2020b). Making Affine Correspondences Work in Camera Geometry Computation Supplementaary Material. In *Proc. of ECCV*. 14

Brügelmann, R. and W. Förstner (1992). Noise Estimation for Color Edge Extraction. In W. Förstner and S. Ruwiedel (Eds.), *Robust Computer Vision*, pp. 90–107. Wichmann, Karlsruhe. 33

Förstner, W. (1991). *Statistische Verfahren für die automatische Bildanalyse und ihre Bewertung bei der Objekterkennung und -vermessung*. Ph. D. thesis, Verlag der Bayrischen Akademie der wissenschaften, Reihe C Dissertationen, No. 370. 31

Förstner, W. (2000). Image Preprocessing for Feature Extraction in Digital Intensity, Color and Range Images. In *Geomatic Methods for the Analysis of Data in Earth Sciences*, Volume 95/2000 of *Lecture Notes in Earth Sciences*, pp. 165–189. Springer. 8, 19, 30, 32

Förstner, W. and B. P. Wrobel (2016). *Photogrammetric Computer Vision – Statistics, Geometry, Orientation and Reconstruction*. Springer. 18, 22, 34

Jähne, B., H. Scharr, and S. Körkel (1999). Principles of Filter Design. In B. Jähne, H. Horst Haussecker, and P. Peter Geissler (Eds.), *Handbook of Computer Vision and Applications*, Chapter 10, pp. 209–238. Academic Press. 16

Koch, K.-R. (1999). *Parameter Estimation and Hypothesis Testing in Linear Models* (2nd ed.). Springer. 35

Shu, X. (2013). Bicubic Interpolation. Dept. Electrical and Computer Engineering, McMaster, Canada, https://www.ece.mcmaster.ca/~xwu/3sk3/interpolation.pdf. 24

Waegli, B. (1998). Investigations into the Noise Characteristics of Digitized Aerial Images. In *Intl. Archives of Photogrammetry and Remote Sensing*, Volume 32–2, pp. 341–348. Proc. of ISPRS Comm. II Symposium, Cambridge, UK. 34