http://users.ics.forth.gr/~lourakis/sba/PRCV_colloq.pdf
lecture by Lourakis “Bundle adjustment gone public”

- Bundle Adjustment (BA) is a key ingredient of SaM, almost always used as its last step
  - • It is an optimization problem over the 3D structure and viewing parameters (camera pose, intrinsic calibration, & radial distortion parameters), which are simultaneously refined for minimizing reprojection error
  - very large nonlinear least squares problem, typically solved with the Levenberg-Marquardt (LM) algorithm
  - Std LM involves the repetitive solution of linear systems, each with O(N^3) time and O(N^2) storage complexity, resp.
    - • Example: for 54 cameras and 5207 3D points, N = 15945. ==> N^3 = 1e12
  - Sparse LM is a better solution.
  - Example:
    - M images
    - N features
    - **x_i_j** = measured feature “i” on image “j”
    - **a_j** = vector of parameters for camera “j”
    - **b_i** = vectors of parameters for point “i”
    - Q(aj , bi) = the predicted projection of point i on image j, <== needs to be in camera ref frame
    - d(., .) the Euclidean distance between image points
    - vij = 1 iff point i is visible in image j
    - minimize reprojection error over a_j, b_i: min_aj, bi (summation_i=1_to_N(summation_j=1_to_M((v_i_j * d(Q(aj,bi), x_i_j))^2)))
      - ==> total number of parameters is M*(camera parameters) + N*(point parameters)
    - let **P** = parameter vector of camera then point parameters = [P_C P_P]
    - let **X_hat** = [(x_hat_1_1)^T x_hat_1_2)^T … x_hat_1_M)^T x_hat_2_1)^T … x_hat_N_M)^T]
    - where ***x_hat_i_j = Q(a_j, b_i)*** is the projection onto camera plane
    - let error **eps** = [(eps_1_1)^T eps_1_2)^T … eps_1_M)^T eps_2_1)^T … eps_N_M)^T]
    - where ***eps = x_i_j - x_hat_i_j***

http://users.ics.forth.gr/~lourakis/sba/PRCV_colloq.pdf

Bundle Adjustment (BA) becomes

min (summation_i=1_to_N(summation_j=1_to_M((eps_i_j)^2))) over P

Jacobian J = d(X_hat) / d(P) which has a block structure because of P being [camera parameters   point parameters]

J = [A | B] where A = d(X_hat) / d(a) and B = A = d(X_hat) / d(b)

The LM updating vector delta = [(delta(a))^T (delta(b))^T

The normal equations:

| A^T * A   A^T * B | * | delta(a)) | = | A^T * eps |
| B^T * A   B^T * B | | delta(b)) | | B^T * eps |

The lhs matrix above is sparse due to A and B being sparse:

∂x̂ij ∂a_k = 0, ∀ j != k and

∂x̂ij ∂b_k = 0, ∀ i != k

http://users.ics.forth.gr/~lourakis/sba/PRCV_colloq.pdf

(example cont.) M images = 3, N features = 4

$\mathbf{J} = \frac{\partial \hat{\mathbf{X}}}{\partial \mathbf{P}}$ has a block structure $[\,\mathbf{A}|\mathbf{B}\,]$,

Let $\mathbf{A}_{ij} = \frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{a}_j}$ and $\mathbf{B}_{ij} = \frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{b}_i}$

- The Jacobian J in block form:

$$
\frac{\partial \hat{\mathbf{X}}}{\partial \mathbf{P}} =
\begin{array}{c}
 \\
\mathbf{x}_{11}\\ \mathbf{x}_{12}\\ \mathbf{x}_{13}\\ \mathbf{x}_{21}\\ \mathbf{x}_{22}\\ \mathbf{x}_{23}\\ \mathbf{x}_{31}\\ \mathbf{x}_{32}\\ \mathbf{x}_{33}\\ \mathbf{x}_{41}\\ \mathbf{x}_{42}\\ \mathbf{x}_{43}
\end{array}
\begin{array}{c}
\begin{array}{ccccccc}
\mathbf{a}_1{}^T & \mathbf{a}_2{}^T & \mathbf{a}_3{}^T & \mathbf{b}_1{}^T & \mathbf{b}_2{}^T & \mathbf{b}_3{}^T & \mathbf{b}_4{}^T
\end{array}\\
\begin{pmatrix}
\mathbf{A}_{11} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{11} & \mathbf{0} & \mathbf{0} & \mathbf{0}\\
\mathbf{0} & \mathbf{A}_{12} & \mathbf{0} & \mathbf{B}_{12} & \mathbf{0} & \mathbf{0} & \mathbf{0}\\
\mathbf{0} & \mathbf{0} & \mathbf{A}_{13} & \mathbf{B}_{13} & \mathbf{0} & \mathbf{0} & \mathbf{0}\\
\mathbf{A}_{21} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{21} & \mathbf{0} & \mathbf{0}\\
\mathbf{0} & \mathbf{A}_{22} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{22} & \mathbf{0} & \mathbf{0}\\
\mathbf{0} & \mathbf{0} & \mathbf{A}_{23} & \mathbf{0} & \mathbf{B}_{23} & \mathbf{0} & \mathbf{0}\\
\mathbf{A}_{31} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{31} & \mathbf{0}\\
\mathbf{0} & \mathbf{A}_{32} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{32} & \mathbf{0}\\
\mathbf{0} & \mathbf{0} & \mathbf{A}_{33} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{33} & \mathbf{0}\\
\mathbf{A}_{41} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{41}\\
\mathbf{0} & \mathbf{A}_{42} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{42}\\
\mathbf{0} & \mathbf{0} & \mathbf{A}_{43} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{43}
\end{pmatrix}
\end{array}
\quad (1)
$$

This is the so-called *primary structure* of BA

- Approximate Hessian in block form:

$$
\mathbf{J}^T\mathbf{J} =
\begin{array}{c}
 \\
\mathbf{a}_1\\ \mathbf{a}_2\\ \mathbf{a}_3\\ \mathbf{b}_1\\ \mathbf{b}_2\\ \mathbf{b}_3\\ \mathbf{b}_4
\end{array}
\begin{array}{c}
\begin{array}{ccccccc}
\mathbf{a}_1{}^T & \mathbf{a}_2{}^T & \mathbf{a}_3{}^T & \mathbf{b}_1{}^T & \mathbf{b}_2{}^T & \mathbf{b}_3{}^T & \mathbf{b}_4{}^T
\end{array}\\
\begin{pmatrix}
\mathbf{U}_1 & 0 & 0 & \mathbf{W}_{11} & \mathbf{W}_{21} & \mathbf{W}_{31} & \mathbf{W}_{41}\\
0 & \mathbf{U}_2 & 0 & \mathbf{W}_{12} & \mathbf{W}_{22} & \mathbf{W}_{32} & \mathbf{W}_{42}\\
0 & 0 & \mathbf{U}_3 & \mathbf{W}_{13} & \mathbf{W}_{23} & \mathbf{W}_{33} & \mathbf{W}_{43}\\
\mathbf{W}_{11}^T & \mathbf{W}_{12}^T & \mathbf{W}_{13}^T & \mathbf{V}_1 & 0 & 0 & 0\\
\mathbf{W}_{21}^T & \mathbf{W}_{22}^T & \mathbf{W}_{23}^T & 0 & \mathbf{V}_2 & 0 & 0\\
\mathbf{W}_{31}^T & \mathbf{W}_{32}^T & \mathbf{W}_{33}^T & 0 & 0 & \mathbf{V}_3 & 0\\
\mathbf{W}_{41}^T & \mathbf{W}_{42}^T & \mathbf{W}_{43}^T & 0 & 0 & 0 & \mathbf{V}_4
\end{pmatrix}
\end{array}
\equiv \quad (2)
$$

$$
\equiv \begin{pmatrix} \mathbf{U} & \mathbf{W} \\ \mathbf{W}^T & \mathbf{V} \end{pmatrix},
$$

$\mathbf{U}_j \equiv \sum_{i=1}^{4} \mathbf{A}_{ij}^T \mathbf{A}_{ij}$, **for 1 image, summing over all features**

$\mathbf{V}_i \equiv \sum_{j=1}^{3} \mathbf{B}_{ij}^T \mathbf{B}_{ij}$, **for 1 feature, summing over all images**

$\mathbf{W}_{ij} = \mathbf{A}_{ij}^T \mathbf{B}_{ij}$

<u>(example cont.) M images = 3, N features = 4</u>

Bundle Adjustment Revisited

Yu Chen1, Yisong Chen1, Guoping Wang1
1 Peking University, Department of Computer Science and Technology,
Graphics and Interactive Lab, Beijing, China

For convenience, we use (18) to show how to solve the bundle adjustment problem. set $\hat{u}_{ij} = \pi(C_j, X_i)$, and we order the parameter x into camera block c and structure block p:

$$x = [c,\ p] \tag{20}$$

it's easily to realize that:

$$J_{ij} = \frac{\partial r_{ij}}{\partial x_k} = \frac{\partial \hat{u}_{ij}}{\partial x_k}, \frac{\partial \hat{u}_{ij}}{\partial c_k} \underline{= 0, \forall j \neq k}, \frac{\partial \hat{u}_{ij}}{\partial p_k} \underline{= 0, \forall i \neq k} \tag{21}$$

Consider now, that we have $m = 3$ cameras and $n = 4$ 3D points. Set $A_{ij} = \frac{\partial \hat{u_{ij}}}{\partial c_j}, B_{ij} = \frac{\partial \hat{u_{ij}}}{\partial p_i}$, we can obtain the Jacobi:

k dimension is row number w.r.t. the block of i where i is the feature number

$$J = \frac{\partial \hat{u}}{\partial x} = \begin{array}{r} \left.\begin{array}{c} \\ k = 1 \\ \\ \end{array}\right\{ \\ \\ \\ \\ \\ \\ \\ \\ \\ \left.\begin{array}{c} \\ k = 4 \\ \\ \end{array}\right\{ \end{array} \begin{bmatrix} A_{11} & 0 & 0 & B_{11} & 0 & 0 & 0 \\ 0 & A_{12} & 0 & B_{12} & 0 & 0 & 0 \\ 0 & 0 & A_{13} & B_{13} & 0 & 0 & 0 \\ A_{21} & 0 & 0 & 0 & B_{21} & 0 & 0 \\ 0 & A_{22} & 0 & 0 & B_{22} & 0 & 0 \\ 0 & 0 & A_{23} & 0 & B_{23} & 0 & 0 \\ A_{31} & 0 & 0 & 0 & 0 & B_{31} & 0 \\ 0 & A_{32} & 0 & 0 & 0 & B_{32} & 0 \\ 0 & 0 & A_{33} & 0 & 0 & B_{33} & 0 \\ A_{41} & 0 & 0 & 0 & 0 & 0 & B_{41} \\ 0 & A_{42} & 0 & 0 & 0 & 0 & B_{42} \\ 0 & 0 & A_{43} & 0 & 0 & 0 & B_{43} \end{bmatrix} \tag{22}$$

http://users.ics.forth.gr/~lourakis/sba/PRCV_colloq.pdf

<u>(example cont.) M images = 3, N features = 4</u>

NOTE: eps_a = A^T * eps, eps_b = B^T * eps

- The augmented normal equations $(\mathbf{J}^T\mathbf{J} + \mu\mathbf{I})\delta_\mathbf{p} = \mathbf{J}^T\epsilon$ take the form

where μ > 0

(3) $$\begin{pmatrix} \mathbf{U}^* & \mathbf{W} \\ \mathbf{W}^T & \mathbf{V}^* \end{pmatrix} \begin{pmatrix} \delta_\mathbf{a} \\ \delta_\mathbf{b} \end{pmatrix} = \begin{pmatrix} \epsilon_\mathbf{a} \\ \epsilon_\mathbf{b} \end{pmatrix}$$

- Performing block Gaussian elimination in the lhs matrix, $\delta_\mathbf{a}$ is determined with Cholesky from $\mathbf{V}^*$'s Schur complement:

(4) $$(\mathbf{U}^* - \mathbf{W}\,\mathbf{V}^{*-1}\,\mathbf{W}^T)\,\delta_\mathbf{a} = \epsilon_\mathbf{a} - \mathbf{W}\,\mathbf{V}^{*-1}\,\epsilon_\mathbf{b}$$

Schur complement: multiply 1st matrix on res by

| I   0 |
| (((-V*)^-1)W^T   I |

resulting in:

| (U*) - W(((V*)^-1)W^T   W | * | delta_a | = | eps_a |
| 0   (V*) | | delta_b | | eps_b |

$$\mathbf{V}^{*-1} = \begin{pmatrix} \mathbf{V}_1^{*-1} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{V}_2^{*-1} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}$$

note (V*) is invertible and only the block diagonals are populated, so each V_i is inverted.

separate delta_b: 0*delta_a + (V*) * delta_b = eps_b ==> delta_b = eps_b * ((V*)^-1)
solving for delta_a (typically M images << N features) after substitute delta_b:
((U*) - W(((V*)^-1)W^T) * delta_a + W * delta_b = eps_a
((U*) - W(((V*)^-1)W^T) * delta_a = eps_a - W((V*)^-1)eps_b

NOTE: (U*) - W(((V*)^-1)W^T is called the reduced camera matrix (because delta_a is camera parameters)

http://users.ics.forth.gr/~lourakis/sba/PRCV_colloq.pdf

RCM (reduced camera matrix) is sparse because not all features appear in all cameras.
this is known as ***secondary structure*.**

For very large datasets, RCM tends to be in one of two classes:
(1) ○ visual mapping: extended areas are traversed, limited image overlap (sparse RCM)
(2) ○ centered-object: a large number of overlapping images taken in a small area (dense RCM)

Solving for delta_a in the equation containing RCM. several ways:
(1) Store as dense, decompose with **ordinary linear algebra** ○[M. Lourakis, A. Argyros: SBA: A Software Package For Generic Sparse Bundle Adjustment. ACM Trans. Math. Softw. 36(1): (2009) ○ C. Engels, H. Stewenius, D. Nister: Bundle Adjustment Rules. Photogrammetric Computer Vision (PCV), 2006.
(2) • Store as sparse, factorize with **sparse direct solvers** ○ K. Konolige: Sparse Sparse Bundle Adjustment. BMVC 2010: 1-11
(3) Store as sparse, use **conjugate gradient methods** memory efficient, iterative, precoditioners necessary! ○ S. Agarwal, N. Snavely, S.M. Seitz, R. Szeliski: Bundle Adjustment in the Large. ECCV (2) 2010: 29-42 ○ M. Byrod, K. Astrom: Conjugate Gradient Bundle Adjustment. ECCV (2) 2010: 114-127
(4) • Avoid storing altogether ○ C. Wu, S. Agarwal, B. Curless, S.M. Seitz: Multicore Bundle Adjustment. CVPR 2011: 30 57-3064 ○ M. Lourakis: Sparse Non-linear Least Squares Optimization for Geometric Vision. ECCV (2) 2010: 43-56

Engels, Stewenius, Nister 2006, “Bundle Adjustment Rules”

m images (= video frames from same calibrated camera)
? features
each feature **x** has **M** dimensions
n iterations of bundle adjustment over the last m video frames

Engels “RCM” is formed from a jacobian which places point parameters before camera parameters,
so is different than that in the Lourakis notes.

$$J_f = \left[\begin{array}{cc} J_P & J_C \end{array} \right], \tag{15}$$

$$H = \left[\begin{array}{cc} J_P^\top J_P & J_P^\top J_C \\ J_C^\top J_P & J_C^\top J_C \end{array} \right], \tag{16}$$

$$\left[\begin{array}{cc} H_{PP} & H_{PC} \\ H_{PC}^\top & H_{CC} \end{array} \right] \left[\begin{array}{c} dP \\ dC \end{array} \right] = \left[\begin{array}{c} b_P \\ b_C \end{array} \right], \tag{17}$$

where we have defined $H_{PP} = J_P^\top J_P$, $H_{PC} = J_P^\top J_C$, $H_{CC} = J_C^\top J_C$, $b_P = -J_P^\top f$,$b_C = -J_C^\top f$ to simpify the notation, and dP and dC represent the update of the point parameters and the camera parameters, respectively. Note that the matrices H_{PP} and H_{CC} are block-diagonal, where the blocks correspond to

of as multiplying by

$$\left[\begin{array}{cc} I & 0 \\ -H_{PC}^\top & I \end{array} \right] \tag{20}$$

from the left on both sides, resulting in the smaller equation system (from the lower part)

$$\underbrace{\left(H_{CC} - H_{PC}^\top H_{PP}^{-1} H_{PC}\right)}_{A} dC = \underbrace{b_C - H_{PC}^\top H_{PP}^{-1} b_P}_{B} \tag{21}$$

for the camera parameter update dC. For very large systems,

We use straightforward Cholesky factorization.

J_P is [2*n*m X 3*m] J_C is [2n*m X 9*n] J is [2nm X (3m + 9n)]

Engels, Stewenius, Nister 2006, "Bundle Adjustment Rules"

blocks: J_P_i is [2*n*m X 3] J_C_i is [2n*m X 9]

M images = 3, j

N features = 4, i

sparse blocks along diagonal: J_P_i is [3 X 3] J_C_i is [9 X 9]

their "RCM" is formed from a jacobian which places point parameters before camera parameters, so is different than that in the Lourakis notes.

Lourakis

Let $\mathbf{A}_{ij} = \frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{a}_j}$ and $\mathbf{B}_{ij} = \frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{b}_i}$

2X9 **C** 2X3 **P**

$$\mathbf{J} = \frac{\partial \hat{\mathbf{X}}}{\partial \mathbf{P}} = \left[\frac{\partial \hat{\mathbf{X}}}{\partial \mathbf{a}} \; \frac{\partial \hat{\mathbf{X}}}{\partial \mathbf{b}} \right] = \left[\; J_C \quad J_P \; \right]$$

- The Jacobian J in block form:

$$\frac{\partial \hat{\mathbf{X}}}{\partial \mathbf{P}} = \begin{array}{c} \\ \mathbf{x}_{11} \\ \mathbf{x}_{12} \\ \mathbf{x}_{13} \\ \mathbf{x}_{21} \\ \mathbf{x}_{22} \\ \mathbf{x}_{23} \\ \mathbf{x}_{31} \\ \mathbf{x}_{32} \\ \mathbf{x}_{33} \\ \mathbf{x}_{41} \\ \mathbf{x}_{42} \\ \mathbf{x}_{43} \end{array} \begin{array}{c} \begin{array}{ccccccc} \mathbf{a}_1^T & \mathbf{a}_2^T & \mathbf{a}_3^T & \mathbf{b}_1^T & \mathbf{b}_2^T & \mathbf{b}_3^T & \mathbf{b}_4^T \end{array} \\ \begin{pmatrix} \mathbf{A}_{11} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{11} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_{12} & \mathbf{0} & \mathbf{B}_{12} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{A}_{13} & \mathbf{B}_{13} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{A}_{21} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{21} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_{22} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{22} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{A}_{23} & \mathbf{0} & \mathbf{B}_{23} & \mathbf{0} & \mathbf{0} \\ \mathbf{A}_{31} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{31} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_{32} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{32} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{A}_{33} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{33} & \mathbf{0} \\ \mathbf{A}_{41} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{41} \\ \mathbf{0} & \mathbf{A}_{42} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{42} \\ \mathbf{0} & \mathbf{0} & \mathbf{A}_{43} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{43} \end{pmatrix} \end{array} \quad (1)$$

Engels

Let $\mathbf{A}_{ij} = \frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{a}_j}$ and $\mathbf{B}_{ij} = \frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{b}_i}$

2X9 **C** 2X3 **P**

$$J_f = \left[\; J_P \quad J_C \; \right],$$

- The Jacobian J in block form:

$$\frac{\partial \hat{\mathbf{X}}}{\partial \mathbf{P}} = \begin{array}{c} \\ \mathbf{x}_{11} \\ \mathbf{x}_{12} \\ \mathbf{x}_{13} \\ \mathbf{x}_{21} \\ \mathbf{x}_{22} \\ \mathbf{x}_{23} \\ \mathbf{x}_{31} \\ \mathbf{x}_{32} \\ \mathbf{x}_{33} \\ \mathbf{x}_{41} \\ \mathbf{x}_{42} \\ \mathbf{x}_{43} \end{array} \begin{array}{c} \begin{array}{ccccccc} \mathbf{b}_1^T & \mathbf{b}_2^T & \mathbf{b}_3^T & \mathbf{b}_4^T & \mathbf{a}_1^T & \mathbf{a}_2^T & \mathbf{a}_3^T \end{array} \\ \begin{pmatrix} \mathbf{B}_{11} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{A}_{11} & \mathbf{0} & \mathbf{0} \\ \mathbf{B}_{12} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{A}_{12} & \mathbf{0} \\ \mathbf{B}_{13} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{A}_{13} \\ \mathbf{0} & \mathbf{B}_{21} & \mathbf{0} & \mathbf{0} & \mathbf{A}_{21} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{B}_{22} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{A}_{22} & \mathbf{0} \\ \mathbf{0} & \mathbf{B}_{23} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{A}_{23} \\ \mathbf{0} & \mathbf{0} & \mathbf{B}_{31} & \mathbf{0} & \mathbf{A}_{31} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{B}_{32} & \mathbf{0} & \mathbf{0} & \mathbf{A}_{32} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{B}_{33} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{A}_{33} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{41} & \mathbf{A}_{41} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{42} & \mathbf{0} & \mathbf{A}_{42} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{43} & \mathbf{0} & \mathbf{0} & \mathbf{A}_{43} \end{pmatrix} \end{array} \quad (1)$$

Engels, Stewenius, Nister 2006, “Bundle Adjustment Rules”

J is [2nm X (3n + 9m)]

J^T * J is [(3n+9m) X (3n+9m)]

M images = 3, j
N features = 4, i

their “RCM” is formed from a jacobian which places point parameters before camera parameters, so is different than that in the Lourakis notes.

Lourakis

Let $\mathbf{A}_{ij} = \frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{a}_j}$ and $\mathbf{B}_{ij} = \frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{b}_i}$

C **P**

$$\mathbf{J} = \frac{\partial \hat{\mathbf{X}}}{\partial \mathbf{P}} = \left[\frac{\partial \hat{\mathbf{X}}}{\partial \mathbf{a}} \; \frac{\partial \hat{\mathbf{X}}}{\partial \mathbf{b}} \right] = [\; J_C \quad J_P \;]$$

J^T * J is [(3*n + 9*m)] X (3*n + 9*m)]

$$\mathbf{J}^T\mathbf{J} =$$

$$\begin{array}{c|ccccccc}
 & \mathbf{a}_1{}^T & \mathbf{a}_2{}^T & \mathbf{a}_3{}^T & \mathbf{b}_1{}^T & \mathbf{b}_2{}^T & \mathbf{b}_3{}^T & \mathbf{b}_4{}^T \\
\hline
\mathbf{a}_1 & \mathbf{U}_1 & 0 & 0 & \mathbf{W}_{11} & \mathbf{W}_{21} & \mathbf{W}_{31} & \mathbf{W}_{41} \\
\mathbf{a}_2 & 0 & \mathbf{U}_2 & 0 & \mathbf{W}_{12} & \mathbf{W}_{22} & \mathbf{W}_{32} & \mathbf{W}_{42} \\
\mathbf{a}_3 & 0 & 0 & \mathbf{U}_3 & \mathbf{W}_{13} & \mathbf{W}_{23} & \mathbf{W}_{33} & \mathbf{W}_{43} \\
\mathbf{b}_1 & \mathbf{W}_{11}^T & \mathbf{W}_{12}^T & \mathbf{W}_{13}^T & \mathbf{V}_1 & 0 & 0 & 0 \\
\mathbf{b}_2 & \mathbf{W}_{21}^T & \mathbf{W}_{22}^T & \mathbf{W}_{23}^T & 0 & \mathbf{V}_2 & 0 & 0 \\
\mathbf{b}_3 & \mathbf{W}_{31}^T & \mathbf{W}_{32}^T & \mathbf{W}_{33}^T & 0 & 0 & \mathbf{V}_3 & 0 \\
\mathbf{b}_4 & \mathbf{W}_{41}^T & \mathbf{W}_{42}^T & \mathbf{W}_{43}^T & 0 & 0 & 0 & \mathbf{V}_4
\end{array}$$

Engels

Let $\mathbf{A}_{ij} = \frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{a}_j}$ and $\mathbf{B}_{ij} = \frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{b}_i}$

2X9 **C** 2X3 **P**

$$\mathbf{J} = \frac{\partial \hat{\mathbf{X}}}{\partial \mathbf{P}} = \left[\frac{\partial \hat{\mathbf{X}}}{\partial \mathbf{b}} \; \frac{\partial \hat{\mathbf{X}}}{\partial \mathbf{a}} \right] = [\; J_P \quad J_C \;],$$

J^T * J is [(3*n + 9*m)] X (3*n + 9*m)]

$$\mathbf{J}^T\mathbf{J} =$$

$$\begin{array}{c|ccccccc}
 & \mathbf{b}_1{}^T & \mathbf{b}_2{}^T & \mathbf{b}_3{}^T & \mathbf{b}_4{}^T & \mathbf{a}_1{}^T & \mathbf{a}_2{}^T & \mathbf{a}_3{}^T \\
\hline
\mathbf{b}_1 & \mathbf{V}_1 & 0 & 0 & 0 & \mathbf{W}_{11}^T & \mathbf{W}_{12}^T & \mathbf{W}_{13}^T \\
\mathbf{b}_2 & 0 & \mathbf{V}_2 & 0 & 0 & \mathbf{W}_{21}^T & \mathbf{W}_{22}^T & \mathbf{W}_{23}^T \\
\mathbf{b}_3 & 0 & 0 & \mathbf{V}_3 & 0 & \mathbf{W}_{31}^T & \mathbf{W}_{32}^T & \mathbf{W}_{33}^T \\
\mathbf{b}_4 & 0 & 0 & 0 & \mathbf{V}_4 & \mathbf{W}_{41}^T & \mathbf{W}_{42}^T & \mathbf{W}_{43}^T \\
\mathbf{a}_1 & \mathbf{W}_{11} & \mathbf{W}_{21} & \mathbf{W}_{31} & \mathbf{W}_{41} & \mathbf{U}_1 & 0 & 0 \\
\mathbf{a}_2 & \mathbf{W}_{12} & \mathbf{W}_{22} & \mathbf{W}_{32} & \mathbf{W}_{42} & 0 & \mathbf{U}_2 & 0 \\
\mathbf{a}_3 & \mathbf{W}_{13} & \mathbf{W}_{23} & \mathbf{W}_{33} & \mathbf{W}_{43} & 0 & 0 & \mathbf{U}_3
\end{array}$$

where

$\mathbf{U}_j \equiv \sum_{i=1}^{4} \mathbf{A}_{ij}^T \mathbf{A}_{ij}$, <u>**for 1 image, sum over features**</u> 9X9

$\mathbf{V}_i \equiv \sum_{j=1}^{3} \mathbf{B}_{ij}^T \mathbf{B}_{ij}$, <u>**for 1 feature, sum over images**</u> 3X3 $\equiv \sum_{j=1}^{3} \left(\frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{b}_i}\right)^T \frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{b}_i}$

$\mathbf{W}_{ij} = \mathbf{A}_{ij}^T \mathbf{B}_{ij}$ 9X3

Engels, Stewenius, Nister 2006, "Bundle Adjustment Rules"

Note that $\mathbf{V}^{*-1} = \begin{pmatrix} \mathbf{V}_1^{*-1} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{V}_2^{*-1} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}$

$\delta_\mathbf{b}$ is dP is [3*n_features X 1]
$\epsilon_\mathbf{b}$ is b_P is [3*n_features X 1]
$\delta_\mathbf{a}$ is dC is [9*m_images X 1]
$\epsilon_\mathbf{a}$ is b_C is [9*m_images X 1]

V* is [3*n X 3*n]

W is [9*m X 3*n]

U* is [9*m X 9*m]

Lourakis

The augmented normal equations $(\mathbf{J}^T\mathbf{J} + \mu\mathbf{I})\delta_\mathbf{p} = \mathbf{J}^T\epsilon$ take the form

(3) $$\begin{pmatrix} \mathbf{U}^* & \mathbf{W} \\ \mathbf{W}^T & \mathbf{V}^* \end{pmatrix} \begin{pmatrix} \delta_\mathbf{a} \\ \delta_\mathbf{b} \end{pmatrix} = \begin{pmatrix} \epsilon_\mathbf{a} \\ \epsilon_\mathbf{b} \end{pmatrix}$$

$$\begin{bmatrix} U - WV^{-1}W^T & 0 \end{bmatrix} \begin{bmatrix} \delta_\mathbf{a} \\ \delta_\mathbf{b} \end{bmatrix} = \begin{bmatrix} I & -WV^{-1} \end{bmatrix} \begin{bmatrix} \epsilon_\mathbf{a} \\ \epsilon_\mathbf{b} \end{bmatrix}$$

(solve delta a first because typically m_images << n_features)

determine $\delta_\mathbf{a}$ with Cholesky **(or other method)**

$$(\mathbf{U}^* - \mathbf{W}\ \mathbf{V}^{*-1}\ \mathbf{W}^T)\ \delta_\mathbf{a} = \epsilon_\mathbf{a} - \mathbf{W}\ \mathbf{V}^{*-1}\ \epsilon_\mathbf{b}$$

$\delta_\mathbf{b}$ can be computed by back substitution into

$$\mathbf{V}^*\ \delta_\mathbf{b} = \epsilon_\mathbf{b} - \mathbf{W}^T\ \delta_\mathbf{a}$$

$$\delta_\mathbf{b} = \mathbf{V}^{*-1}\ \epsilon_\mathbf{b}\ -\ \mathbf{V}^{*-1}\ \mathbf{W}^T\ \delta_\mathbf{a}$$

Engels

$$\begin{pmatrix} \mathbf{V}^* & \mathbf{W}^T \\ \mathbf{W} & \mathbf{U}^* \end{pmatrix} \begin{pmatrix} \delta_\mathbf{b} \\ \delta_\mathbf{a} \end{pmatrix} = \begin{pmatrix} \epsilon_\mathbf{b} \\ \epsilon_\mathbf{a} \end{pmatrix}$$

$$\begin{bmatrix} H_{PP} & H_{PC} \\ H_{PC}^\top & H_{CC} \end{bmatrix} \begin{bmatrix} dP \\ dC \end{bmatrix} = \begin{bmatrix} b_P \\ b_C \end{bmatrix},$$

$$\begin{aligned} H_{PP} &= J_P^\top J_P \\ H_{PC} &= J_P^\top J_C, \\ H_{CC} &= J_C^\top J_C, \\ b_P &= -J_P^\top f, \\ b_C &= -J_C^\top f \end{aligned}$$

$$(\mathbf{U}^* - \mathbf{W}\ \mathbf{V}^{*-1}\ \mathbf{W}^T)\ \delta_\mathbf{a} = \epsilon_\mathbf{a} - \mathbf{W}\ \mathbf{V}^{*-1}\ \epsilon_\mathbf{b}$$

$$\underbrace{(H_{CC} - H_{PC}^\top H_{PP}^{-1} H_{PC})}_{A} dC = \underbrace{b_C - H_{PC}^\top H_{PP}^{-1} b_P}_{B}$$

$\delta_\mathbf{b}$ can be computed by back substitution into

$$\mathbf{V}^*\ \delta_\mathbf{b} = \epsilon_\mathbf{b} - \mathbf{W}^T\ \delta_\mathbf{a}$$

$$H_{PP}\ dP = b_P - H_{PC}\ dC$$

$$dP = H_{PP}^{-1} b_P - H_{PP}^{-1} H_{PC} dC.$$

Qu

$$\begin{pmatrix} \boldsymbol{B} & \boldsymbol{E} \\ \boldsymbol{E}^T & \boldsymbol{C} \end{pmatrix} \begin{pmatrix} \boldsymbol{p}_c \\ \boldsymbol{p}_p \end{pmatrix} = -\begin{pmatrix} \boldsymbol{g}_c \\ \boldsymbol{g}_p \end{pmatrix} \quad (3.61)$$

see eqn (3.70) too

$$-\boldsymbol{g}^k = -\boldsymbol{J}^{k^T} \boldsymbol{F}^k$$

where k is a feature block summed over all images?

$$(\boldsymbol{B} - \boldsymbol{E}\boldsymbol{C}^{-1}\boldsymbol{E}^T)\boldsymbol{p}_c = -\boldsymbol{g}_c + \boldsymbol{E}\boldsymbol{C}^{-1}\boldsymbol{g}_p$$

$$\boldsymbol{p}_p = \boldsymbol{C}^{-1}(-\boldsymbol{g}_p - \boldsymbol{E}^T\boldsymbol{p}_c)$$

Engels, et al 2006

M images = 3, j

N features = 4, i

$$H_{PP} = J_P^\top J_P \equiv \mathbf{V}^*$$
$$H_{PC} = J_P^\top J_C, \equiv \mathbf{W}^T$$
$$H_{CC} = J_C^\top J_C, \equiv \mathbf{U}^*$$
$$b_P = -J_P^\top f, \equiv \epsilon_{\mathbf{b}}$$
$$b_C = -J_C^\top f \equiv \epsilon_{\mathbf{a}}$$

$\mathbf{A}_{ij} = \frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{a}_j}$ and $\mathbf{B}_{ij} = \frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{b}_i}$

$\boxed{2X9}$ **C** $\boxed{2X3}$ **P**

Note that $\mathbf{V}^{*-1} = \begin{pmatrix} \mathbf{V}_1^{*-1} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{V}_2^{*-1} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}$

$\mathbf{U}_j \equiv \sum_{i=1}^4 \mathbf{A}_{ij}^T \mathbf{A}_{ij},$ **[9X9], for 1 image, sum over features**

$\mathbf{V}_i \equiv \sum_{j=1}^3 \mathbf{B}_{ij}^T \mathbf{B}_{ij},$ **[3X3], for 1 feature, sum over images**

$\mathbf{W}_{ij} = \mathbf{A}_{ij}^T \mathbf{B}_{ij}$ **[9X3]**

V* is [3*n X 3*n]

W is [9*m X 3*n]

U* is [9*m X 9*m]

$$\begin{pmatrix} \mathbf{V}^* & \mathbf{W}^T \\ \mathbf{W} & \mathbf{U}^* \end{pmatrix} \begin{pmatrix} \delta_{\mathbf{b}} \\ \delta_{\mathbf{a}} \end{pmatrix} = \begin{pmatrix} \epsilon_{\mathbf{b}} \\ \epsilon_{\mathbf{a}} \end{pmatrix}$$

JP^T = [3n X 2mn]

B11^T	**B12^T**	**B13^T**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
0	**0**	**0**	**B21^T**	**B22^T**	**B23^T**	**0**	**0**	**0**	**0**	**0**	**0**
0	**0**	**0**	**0**	**0**	**0**	**B31^T**	**B32^T**	**B33^T**	**0**	**0**	**0**
0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**B41^T**	**B42^T**	**B43^T**

bP = -JP^T * F = [3n X 1]

-B11T*F11-B12T*F12-B13T*F13
-B21T*F21-B22T*F22-B23T*F23
-B31T*F31-B32T*F32-B33T*F33
-B41T*F41-B42T*F42-B43T*F43

where each row is [3X2]*[2X1] = [3X1]

F11
F12
F13
F21
F22
F23
F31
F32
F33
F41
F42
F43

JC^T = [9m X 2mn]

A11T	**0**	**0**	**A21T**	**0**	**0**	**A31T**	**0**	**0**	**A41T**	**0**	**0**
0	**A12T**	**0**	**0**	**A22T**	**0**	**0**	**A32T**	**0**	**0**	**A42T**	**0**
0	**0**	**A13T**	**0**	**0**	**A23T**	**0**	**0**	**A33T**	**0**	**0**	**A43T**

bC = -JC^T * F = [9m X 1]

-A11T*F11- A21T*F21-A31T*F31-A41T*F41
-A12T*F12- A22T*F22-A32T*F32-A42T*F42
-A13T*F13- A23T*F23-A33T*F33-A43T*F43

where each row is [9X2][2X1] = [9X1]

F11
F12
F13
F21
F22
F23
F31
F32
F33
F41
F42
F43

[2mn X 1]

$$\mathbf{V}^* \qquad \delta_{\mathbf{b}}$$

$$\text{n blocks}\left\{\underbrace{\begin{matrix} \underset{[3X3]}{\mathbf{V}_1} & 0 & 0 & 0 \\ 0 & \underset{[3X3]}{\mathbf{V}_2} & 0 & 0 \\ 0 & 0 & \underset{[3X3]}{\mathbf{V}_3} & 0 \\ 0 & 0 & 0 & \underset{[3X3]}{\mathbf{V}_4} \end{matrix}}_{\text{n blocks}}\right. \quad \begin{matrix} [x0,y0,z0]^T \\ [x1,y1,z1]^T \\ [x2,y2,z2]^T \\ [x3,y3,z3]^T \end{matrix} \quad \textbf{[3*nX1]}$$

$$\mathbf{U}^* \qquad \delta_{\mathbf{a}}$$

$$\text{m blocks}\left\{\underbrace{\begin{matrix} \underset{[9X9]}{\mathbf{U}_1} & 0 & 0 \\ 0 & \underset{[9X9]}{\mathbf{U}_2} & 0 \\ 0 & 0 & \underset{[9X9]}{\mathbf{U}_3} \end{matrix}}_{\text{m blocks}}\right. \quad \begin{matrix} [9X1]^T \\ [9X1]^T \\ [9X1]^T \end{matrix} \quad \textbf{[9*mX1]}$$

Engels, et al 2006

M images = 3, j

N features = 4, i

$$
\begin{aligned}
H_{PP} &= J_P^\top J_P &\equiv \mathbf{V}^*\\
H_{PC} &= J_P^\top J_C, &\equiv \mathbf{W}^T\\
H_{CC} &= J_C^\top J_C, &\equiv \mathbf{U}^*\\
b_P &= -J_P^\top f, &\equiv \epsilon_\mathbf{b}\\
b_C &= -J_C^\top f &\equiv \epsilon_\mathbf{a}
\end{aligned}
$$

$$(\mathbf{U}^* - \mathbf{W}\ \mathbf{V}^{*-1}\ \mathbf{W}^T)\ \delta_\mathbf{a} = \epsilon_\mathbf{a} - \mathbf{W}\ \mathbf{V}^{*-1}\ \epsilon_\mathbf{b}$$

$$\underbrace{(H_{CC} - H_{PC}^\top H_{PP}^{-1} H_{PC})}_{A}\, dC = \underbrace{b_C - H_{PC}^\top H_{PP}^{-1} b_P}_{B}$$

$$\underset{\boxed{2X9}}{\mathbf{A}_{ij}} = \frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{a}_j} \quad \mathbf{C} \quad \text{and} \quad \underset{\boxed{2X3}}{\mathbf{B}_{ij}} = \frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{b}_i} \quad \mathbf{P}$$

Note that $\mathbf{V}^{*-1} = \begin{pmatrix} \mathbf{V}_1^{*-1} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{V}_2^{*-1} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}$

$$\mathbf{J}^T\mathbf{J} = \begin{array}{c} \\ \mathbf{b}_1 \\ \mathbf{b}_2 \\ \mathbf{b}_3 \\ \mathbf{b}_4 \\ \mathbf{a}_1 \\ \mathbf{a}_2 \\ \mathbf{a}_3 \end{array} \begin{array}{c} \begin{array}{ccccccc} \mathbf{b}_1{}^T & \mathbf{b}_2{}^T & \mathbf{b}_3{}^T & \mathbf{b}_4{}^T & \mathbf{a}_1{}^T & \mathbf{a}_2{}^T & \mathbf{a}_3{}^T \end{array} \\ \begin{pmatrix}
\mathbf{V}_1 & 0 & 0 & 0 & \mathbf{W}_{11}^T & \mathbf{W}_{12}^T & \mathbf{W}_{13}^T \\
0 & \mathbf{V}_2 & 0 & 0 & \mathbf{W}_{21}^T & \mathbf{W}_{22}^T & \mathbf{W}_{23}^T \\
0 & 0 & \mathbf{V}_3 & 0 & \mathbf{W}_{31}^T & \mathbf{W}_{32}^T & \mathbf{W}_{33}^T \\
0 & 0 & 0 & \mathbf{V}_4 & \mathbf{W}_{41}^T & \mathbf{W}_{42}^T & \mathbf{W}_{43}^T \\
\mathbf{W}_{11} & \mathbf{W}_{21} & \mathbf{W}_{31} & \mathbf{W}_{41} & \mathbf{U}_1 & 0 & 0 \\
\mathbf{W}_{12} & \mathbf{W}_{22} & \mathbf{W}_{32} & \mathbf{W}_{42} & 0 & \mathbf{U}_2 & 0 \\
\mathbf{W}_{13} & \mathbf{W}_{23} & \mathbf{W}_{33} & \mathbf{W}_{43} & 0 & 0 & \mathbf{U}_3
\end{pmatrix} \end{array}$$

$\mathbf{U}_j \equiv \sum_{i=1}^{4} \mathbf{A}_{ij}^T \mathbf{A}_{ij},$ **[9X9], for 1 image, sum over features**

$\mathbf{V}_i \equiv \sum_{j=1}^{3} \mathbf{B}_{ij}^T \mathbf{B}_{ij},$ **[3X3], for 1 feature, sum over images**

$\mathbf{W}_{ij} = \mathbf{A}_{ij}^T \mathbf{B}_{ij}$ **[9X3]**

V* is [3*n X 3*n]

W is [9*m X 3*n]

U* is [9*m X 9*m]

$$\begin{pmatrix} \mathbf{V}^* & \mathbf{W}^T \\ \mathbf{W} & \mathbf{U}^* \end{pmatrix} \begin{pmatrix} \delta_\mathbf{b} \\ \delta_\mathbf{a} \end{pmatrix} = \begin{pmatrix} \epsilon_\mathbf{b} \\ \epsilon_\mathbf{a} \end{pmatrix}$$

W*V^-1= [9mX3n] the V's are inverses
W11*V1 W21*V2 W31*V3 W41*V4
W12*V1 W22*V2 W32*V3 W42*V4
W13*V1 W23*V2 W33*V3 W43*V4
where each block is [9X3]*[3X3] = [9X3]

W*V^-1*W^T= [9mX9] the V's are inverses
W11*V1*W11T + W21*V2*W21T + W31*V3*W31T + W41*V4*W41T
W12*V1*W12T + W22*V2*W22T + W32*V3*W32T + W42*V4*W42T
W13*V1*W13T + W23*V2*W23T + W33*V3*W33T + W43*V4*W43T
where each block is [9X3]*[3X9] = [9X9]

$$\text{n blocks}\left\{ \underbrace{\overset{\mathbf{V}^*}{\begin{matrix} \underset{[3X3]}{\mathbf{V}_1} & 0 & 0 & 0 \\ 0 & \underset{[3X3]}{\mathbf{V}_2} & 0 & 0 \\ 0 & 0 & \underset{[3X3]}{\mathbf{V}_3} & 0 \\ 0 & 0 & 0 & \underset{[3X3]}{\mathbf{V}_4} \end{matrix}}}_{\text{n blocks}} \right. \quad \overset{\delta_\mathbf{b}}{\begin{matrix} [x0,y0,z0]^T \\ [x1,y1,z1]^T \\ [x2,y2,z2]^T \\ [x3,y3,z3]^T \end{matrix}} \quad [3^*nX1]$$

$$\text{m blocks}\left\{ \underbrace{\overset{\mathbf{U}^*}{\begin{matrix} \underset{[9X9]}{\mathbf{U}_1} & 0 & 0 \\ 0 & \underset{[9X9]}{\mathbf{U}_2} & 0 \\ 0 & 0 & \underset{[9X9]}{\mathbf{U}_3} \end{matrix}}}_{\text{m blocks}} \right. \quad \overset{\delta_\mathbf{a}}{\begin{matrix} [9X1]^T \\ [9X1]^T \\ [9X1]^T \end{matrix}} \quad [9^*mX1]$$

bC - (W*V^-1)*bP

bC = -JC^T * F = [9m X 1]

-A11T*F11- A21T*F21-A31T*F31-A41T*F41
-A12T*F12- A22T*F22-A32T*F32-A42T*F42
-A13T*F13- A23T*F23-A33T*F33-A43T*F43

where each row is [9X2][2X1] = [9X1]

W*V^-1= [9mX3n] the V's are inverses

W11*V1 W21*V2 W31*V3 W41*V4
W12*V1 W22*V2 W32*V3 W42*V4
W13*V1 W23*V2 W33*V3 W43*V4

where each block is [9X3]*[3X3] = [9X3]

bP = -JP^T * F [3n X 1]

-B11T*F11-B12T*F12-B13T*F13
-B21T*F21-B22T*F22-B23T*F23
-B31T*F31-B32T*F32-B33T*F33
-B41T*F41-B42T*F42-B43T*F43

where each row is [3X2]*[2X1] = [3X1]

Engels, et al 2006

M images = 3, j

N features = 4, i

$$H_{PP} = J_P^T J_P \equiv \mathbf{V}^*$$
$$H_{PC} = J_P^T J_C, \equiv \mathbf{W}^T$$
$$H_{CC} = J_C^T J_C, \equiv \mathbf{U}^*$$
$$b_P = -J_P^T f, \equiv \epsilon_{\mathbf{b}}$$
$$b_C = -J_C^T f \equiv \epsilon_{\mathbf{a}}$$

$$\mathbf{A}_{ij} = \frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{a}_j} \text{ and } \mathbf{B}_{ij} = \frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{b}_i}$$

2X9 **C** 2X3 **P**

Note that $\mathbf{V}^{*-1} = \begin{pmatrix} \mathbf{V}_1^{*-1} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{V}_2^{*-1} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}$

$$(\mathbf{U}^* - \mathbf{W}\,\mathbf{V}^{*-1}\,\mathbf{W}^T)\,\delta_{\mathbf{a}} = \epsilon_{\mathbf{a}} - \mathbf{W}\,\mathbf{V}^{*-1}\,\epsilon_{\mathbf{b}}$$

$$\underbrace{(H_{CC} - H_{PC}^T H_{PP}^{-1} H_{PC})}_{A}\, dC = \underbrace{b_C - H_{PC}^T H_{PP}^{-1} b_P}_{B}$$

$$\mathbf{J}^T\mathbf{J} =$$
$$\begin{array}{c} \\ \mathbf{b}_1 \\ \mathbf{b}_2 \\ \mathbf{b}_3 \\ \mathbf{b}_4 \\ \mathbf{a}_1 \\ \mathbf{a}_2 \\ \mathbf{a}_3 \end{array}
\begin{array}{c} \begin{array}{ccccccc} \mathbf{b}_1{}^T & \mathbf{b}_2{}^T & \mathbf{b}_3{}^T & \mathbf{b}_4{}^T & \mathbf{a}_1{}^T & \mathbf{a}_2{}^T & \mathbf{a}_3{}^T \end{array} \\
\begin{pmatrix}
\mathbf{V}_1 & 0 & 0 & 0 & \mathbf{W}_{11}^T & \mathbf{W}_{12}^T & \mathbf{W}_{13}^T \\
0 & \mathbf{V}_2 & 0 & 0 & \mathbf{W}_{21}^T & \mathbf{W}_{22}^T & \mathbf{W}_{23}^T \\
0 & 0 & \mathbf{V}_3 & 0 & \mathbf{W}_{31}^T & \mathbf{W}_{32}^T & \mathbf{W}_{33}^T \\
0 & 0 & 0 & \mathbf{V}_4 & \mathbf{W}_{41}^T & \mathbf{W}_{42}^T & \mathbf{W}_{43}^T \\
\mathbf{W}_{11} & \mathbf{W}_{21} & \mathbf{W}_{31} & \mathbf{W}_{41} & \mathbf{U}_1 & 0 & 0 \\
\mathbf{W}_{12} & \mathbf{W}_{22} & \mathbf{W}_{32} & \mathbf{W}_{42} & 0 & \mathbf{U}_2 & 0 \\
\mathbf{W}_{13} & \mathbf{W}_{23} & \mathbf{W}_{33} & \mathbf{W}_{43} & 0 & 0 & \mathbf{U}_3
\end{pmatrix} \end{array}$$

$\mathbf{U}_j \equiv \sum_{i=1}^4 \mathbf{A}_{ij}^T \mathbf{A}_{ij},$ **[9X9], for 1 image, sum over features**

$\mathbf{V}_i \equiv \sum_{j=1}^3 \mathbf{B}_{ij}^T \mathbf{B}_{ij},$ **[3X3], for 1 feature, sum over images**

$\mathbf{W}_{ij} = \mathbf{A}_{ij}^T \mathbf{B}_{ij}$ **[9X3]**

V* is [3*n X 3*n]

W is [9*m X 3*n]

U* is [9*m X 9*m]

$$\begin{pmatrix} \mathbf{V}^* & \mathbf{W}^T \\ \mathbf{W} & \mathbf{U}^* \end{pmatrix} \begin{pmatrix} \delta_{\mathbf{b}} \\ \delta_{\mathbf{a}} \end{pmatrix} = \begin{pmatrix} \epsilon_{\mathbf{b}} \\ \epsilon_{\mathbf{a}} \end{pmatrix}$$

W*V^-1= [9mX3n] the V's are inverses

W11*V1 W21*V2 W31*V3 W41*V4
W12*V1 W22*V2 W32*V3 W42*V4
W13*V1 W23*V2 W33*V3 W43*V4
where each block is [9X3]*[3X3] = [9X3]

W*V^-1*W^T= [9mX9m] the V's are inverses

```
| W11*V1   W21*V2  W31*V3  W41*V4 |   *   | W11  W12 W13 |
| W12*V1   W22*V2  W32*V3  W42*V4 |       | W21  W22 W23 |
| W13*V1   W23*V2  W33*V3  W43*V4 |       | W31  W32 W33 |
                                          | W41  W42 W43 |
```
=

W11*V1*W11+W21*V2*W21+W31*V3*W31+W41*V4*W41 W11*V1*W12+W21*V2*W22+W31*V3*W32+W41*V4*W42 W11*V1*W13+W21*V2*W23+W31*V3*W33+W41*V4*W43
W12*V1*W11+W22*V2*W21+W32*V3*W31+W42*V4*W41 W12*V1*W12+W22*V2*W22+W32*V3*W32+W42*V4*W42 W12*V1*W13+W22*V2*W23+W32*V3*W33+W42*V4*W43
W13*V1*W11+W23*V2*W21+W33*V3*W31+W43*V4*W41 W13*V1*W12+W23*V2*W22+W33*V3*W32+W43*V4*W42 W13*V1*W13+W23*V2*W23+W33*V3*W33+W43*V4*W43
where each block is [9X3]*[3X3]*[3X9] = [9X9]

A is [9*mImages X 9*mImages]

U - W*V^-1*W^T= [9mX9m] the V's are inverses
```
| U1   0    0   | - | W11*V1*W11+W21*V2*W21+W31*V3*W31+W41*V4*W41   W11*V1*W12+W21*V2*W22+W31*V3*W32+W41*V4*W42  W11*V1*W13+W21*V2*W23+W31*V3*W33+W41*V4*W43|
| 0    U2   0   |   | W12*V1*W11+W22*V2*W21+W32*V3*W31+W42*V4*W41   W12*V1*W12+W22*V2*W22+W32*V3*W32+W42*V4*W42  W12*V1*W13+W22*V2*W23+W32*V3*W33+W42*V4*W43|
| 0    0    U3  |   | W13*V1*W11+W23*V2*W21+W33*V3*W31+W43*V4*W41   W13*V1*W12+W23*V2*W22+W33*V3*W32+W43*V4*W42  W13*V1*W13+W23*V2*W23+W33*V3*W33+W43*V4*W43|
```
where each block is [9X9]

Engels, et al 2006

M images = 3, j

N features = 4, i

$$\begin{aligned} H_{PP} &= J_P^\top J_P &\equiv \mathbf{V}^* \\ H_{PC} &= J_P^\top J_C, &\equiv \mathbf{W}^T \\ H_{CC} &= J_C^\top J_C, &\equiv \mathbf{U}^* \\ b_P &= -J_P^\top f, &\equiv \epsilon_{\mathbf{b}} \\ b_C &= -J_C^\top f &\equiv \epsilon_{\mathbf{a}} \end{aligned}$$

$$\mathbf{A}_{ij} = \frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{a}_j} \text{ and } \mathbf{B}_{ij} = \frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{b}_i}$$

$\boxed{2X9}$ **C** $\boxed{2X3}$ **P**

Note that $\mathbf{V}^{*-1} = \begin{pmatrix} \mathbf{V}_1^{*-1} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{V}_2^{*-1} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}$

$$(\mathbf{U}^* - \mathbf{W}\ \mathbf{V}^{*-1}\ \mathbf{W}^T)\ \delta_{\mathbf{a}} = \epsilon_{\mathbf{a}} - \mathbf{W}\ \mathbf{V}^{*-1}\ \epsilon_{\mathbf{b}}$$

$$\underbrace{(H_{CC} - H_{PC}^\top H_{PP}^{-1} H_{PC})}_{A}\, dC = \underbrace{b_C - H_{PC}^\top H_{PP}^{-1} b_P}_{B}$$

$$\mathbf{J}^T\mathbf{J} = \begin{array}{c|ccccccc} & \mathbf{b}_1{}^T & \mathbf{b}_2{}^T & \mathbf{b}_3{}^T & \mathbf{b}_4{}^T & \mathbf{a}_1{}^T & \mathbf{a}_2{}^T & \mathbf{a}_3{}^T \\ \hline \mathbf{b}_1 & \mathbf{V}_1 & 0 & 0 & 0 & \mathbf{W}_{11}^T & \mathbf{W}_{12}^T & \mathbf{W}_{13}^T \\ \mathbf{b}_2 & 0 & \mathbf{V}_2 & 0 & 0 & \mathbf{W}_{21}^T & \mathbf{W}_{22}^T & \mathbf{W}_{23}^T \\ \mathbf{b}_3 & 0 & 0 & \mathbf{V}_3 & 0 & \mathbf{W}_{31}^T & \mathbf{W}_{32}^T & \mathbf{W}_{33}^T \\ \mathbf{b}_4 & 0 & 0 & 0 & \mathbf{V}_4 & \mathbf{W}_{41}^T & \mathbf{W}_{42}^T & \mathbf{W}_{43}^T \\ \mathbf{a}_1 & \mathbf{W}_{11} & \mathbf{W}_{21} & \mathbf{W}_{31} & \mathbf{W}_{41} & \mathbf{U}_1 & 0 & 0 \\ \mathbf{a}_2 & \mathbf{W}_{12} & \mathbf{W}_{22} & \mathbf{W}_{32} & \mathbf{W}_{42} & 0 & \mathbf{U}_2 & 0 \\ \mathbf{a}_3 & \mathbf{W}_{13} & \mathbf{W}_{23} & \mathbf{W}_{33} & \mathbf{W}_{43} & 0 & 0 & \mathbf{U}_3 \end{array}$$

$\mathbf{U}_j \equiv \sum_{i=1}^4 \mathbf{A}_{ij}^T \mathbf{A}_{ij},$ **[9X9], for 1 image, sum over features**

$\mathbf{V}_i \equiv \sum_{j=1}^3 \mathbf{B}_{ij}^T \mathbf{B}_{ij},$ **[3X3], for 1 feature, sum over images**

$\mathbf{W}_{ij} = \mathbf{A}_{ij}^T \mathbf{B}_{ij}$ **[9X3]**

$\boxed{\text{V* is [3*n X 3*n]}}$

$\boxed{\text{W is [9*m X 3*n]}}$

$\boxed{\text{U* is [9*m X 9*m]}}$

$$\begin{pmatrix} \mathbf{V}^* & \mathbf{W}^T \\ \mathbf{W} & \mathbf{U}^* \end{pmatrix} \begin{pmatrix} \delta_{\mathbf{b}} \\ \delta_{\mathbf{a}} \end{pmatrix} = \begin{pmatrix} \epsilon_{\mathbf{b}} \\ \epsilon_{\mathbf{a}} \end{pmatrix}$$

```
W*V^-1*W^T=  [9mX9m]  the V's are inverses
------
| W11*V1   W21*V2  W31*V3  W41*V4 |   *   | W11  W12 W13 |
| W12*V1   W22*V2  W32*V3  W42*V4 |       | W21  W22 W23 |
| W13*V1   W23*V2  W33*V3  W43*V4 |       | W31  W32 W33 |
                                          | W41  W42 W43 |
=
W11*V1*W11+W21*V2*W21+W31*V3*W31+W41*V4*W41   W11*V1*W12+W21*V2*W22+W31*V3*W32+W41*V4*W42  W11*V1*W13+W21*V2*W23+W31*V3*W33+W41*V4*W43
W12*V1*W11+W22*V2*W21+W32*V3*W31+W42*V4*W41   W12*V1*W12+W22*V2*W22+W32*V3*W32+W42*V4*W42  W12*V1*W13+W22*V2*W23+W32*V3*W33+W42*V4*W43
W13*V1*W11+W23*V2*W21+W33*V3*W31+W43*V4*W41   W13*V1*W12+W23*V2*W22+W33*V3*W32+W43*V4*W42  W13*V1*W13+W23*V2*W23+W33*V3*W33+W43*V4*W43
where each block is [9X3]*[3X3]*[3X9] = [9X9]

from Engels, set into matrix A:
    j = 1:mImages
      calc tPC = HPC^T * invHPP
      j2 = 1:mImages
        Subtract tPC * HPC2 (which is HPC^T * invHPP * HPC2) from block (c, c2)
                        W^T  * inv(V) * W2^T
    Looks correct:
    i=1,j=1,j2=1: block(1,1)-= (W11^T * inv(V1) * W11^T)
    i=1,j=1,j2=2: block(1,2)-= (W11^T * inv(V1) * W12^T)
    i=1,j=1,j2=3: block(1,3)-= (W11^T * inv(V1) * W13^T)
    i=1,j=2,j2=1: block(2,1)-= (W12^T * inv(V1) * W11^T) <=== j2 indexes are not >= j, but instead start at first index, else would never fill in block(2,1), (3,1),(3,2)...
    i=1,j=2,j2=2: block(2,2)-= (W12^T * inv(V1) * W12^T)
    i=1,j=2,j2=3: block(2,3)-= (W12^T * inv(V1) * W13^T)
    ...
    i=2,j=1,j2=1: block(1,1)-= (W11^T * inv(V1) * W21^T)
```

Engels, et al 2006

M images = 3, j
N features = 4, i

$$
\begin{aligned}
H_{PP} &= J_P^T J_P && \equiv \mathbf{V}^* \\
H_{PC} &= J_P^T J_C, && \equiv \mathbf{W}^T \\
H_{CC} &= J_C^T J_C, && \equiv \mathbf{U}^* \\
b_P &= -J_P^T f, && \equiv \epsilon_{\mathbf{b}} \\
b_C &= -J_C^T f && \equiv \epsilon_{\mathbf{a}}
\end{aligned}
$$

$$\underset{\boxed{2X9}}{\mathbf{A}_{ij}} = \frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{a}_j} \text{ and } \underset{\boxed{2X3}}{\mathbf{B}_{ij}} = \frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{b}_i}$$

C **P**

Note that $\mathbf{V}^{*-1} = \begin{pmatrix} \mathbf{V}_1^{*-1} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{V}_2^{*-1} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}$

$$(\mathbf{U}^* - \mathbf{W}\,\mathbf{V}^{*-1}\,\mathbf{W}^T)\,\delta_{\mathbf{a}} = \epsilon_{\mathbf{a}} - \mathbf{W}\,\mathbf{V}^{*-1}\,\epsilon_{\mathbf{b}}$$

$$\underbrace{(H_{CC} - H_{PC}^T H_{PP}^{-1} H_{PC})}_{A}\, dC = \underbrace{b_C - H_{PC}^T H_{PP}^{-1} b_P}_{B}$$

$$dP = H_{PP}^{-1} b_P - H_{PP}^{-1} H_{PC}\, dC.$$

$$
\mathbf{J}^T\mathbf{J} =
\begin{array}{c}
 \\ \mathbf{b}_1 \\ \mathbf{b}_2 \\ \mathbf{b}_3 \\ \mathbf{b}_4 \\ \mathbf{a}_1 \\ \mathbf{a}_2 \\ \mathbf{a}_3
\end{array}
\begin{array}{c}
\begin{array}{ccccccc} \mathbf{b}_1{}^T & \mathbf{b}_2{}^T & \mathbf{b}_3{}^T & \mathbf{b}_4{}^T & \mathbf{a}_1{}^T & \mathbf{a}_2{}^T & \mathbf{a}_3{}^T \end{array} \\
\begin{pmatrix}
\mathbf{V}_1 & 0 & 0 & 0 & \mathbf{W}_{11}^T & \mathbf{W}_{12}^T & \mathbf{W}_{13}^T \\
0 & \mathbf{V}_2 & 0 & 0 & \mathbf{W}_{21}^T & \mathbf{W}_{22}^T & \mathbf{W}_{23}^T \\
0 & 0 & \mathbf{V}_3 & 0 & \mathbf{W}_{31}^T & \mathbf{W}_{32}^T & \mathbf{W}_{33}^T \\
0 & 0 & 0 & \mathbf{V}_4 & \mathbf{W}_{41}^T & \mathbf{W}_{42}^T & \mathbf{W}_{43}^T \\
\mathbf{W}_{11} & \mathbf{W}_{21} & \mathbf{W}_{31} & \mathbf{W}_{41} & \mathbf{U}_1 & 0 & 0 \\
\mathbf{W}_{12} & \mathbf{W}_{22} & \mathbf{W}_{32} & \mathbf{W}_{42} & 0 & \mathbf{U}_2 & 0 \\
\mathbf{W}_{13} & \mathbf{W}_{23} & \mathbf{W}_{33} & \mathbf{W}_{43} & 0 & 0 & \mathbf{U}_3
\end{pmatrix}
\end{array}
$$

$\mathbf{U}_j \equiv \sum_{i=1}^{4} \mathbf{A}_{ij}^T \mathbf{A}_{ij},$ **[9X9], for 1 image, sum over features**

$\mathbf{V}_i \equiv \sum_{j=1}^{3} \mathbf{B}_{ij}^T \mathbf{B}_{ij},$ **[3X3], for 1 feature, sum over images**

$\mathbf{W}_{ij} = \mathbf{A}_{ij}^T \mathbf{B}_{ij}$ **[9X3]**

V* is [3*n X 3*n]

W is [9*m X 3*n]

U* is [9*m X 9*m]

$$\begin{pmatrix} \mathbf{V}^* & \mathbf{W}^T \\ \mathbf{W} & \mathbf{U}^* \end{pmatrix} \begin{pmatrix} \delta_{\mathbf{b}} \\ \delta_{\mathbf{a}} \end{pmatrix} = \begin{pmatrix} \epsilon_{\mathbf{b}} \\ \epsilon_{\mathbf{a}} \end{pmatrix}$$

```
tP = hPPI^−1 ∗ bPI is V^−1∗bPI // [3X3][3X1] = [3X1]
```

V^-1= [3nX3n]

V1^-1	0	0	0
0	V2^-1	0	0
0	0	V3^-1	0
0	0	0	V4^-1

where each block is [3X3]

bP = JP^T * F [3n X 1]

-B11T*F11-B12T*F12-B13T*F13
-B21T*F21-B22T*F22-B23T*F23
-B31T*F31-B32T*F32-B33T*F33
-B41T*F41-B42T*F42-B43T*F43

where each row is [3X2]*[2X1]=[3X1]

=

$H_{PP}^{-1} b_P$ `[3n X 1]`

-(V1^-1)*(B11T*F11-B12T*F12-B13T*F13)
-(V2^-1)*(B21T*F21-B22T*F22-B23T*F23)
-(V3^-1)*(B31T*F31-B32T*F32-B33T*F33)
-(V4^-1)*(B41T*F41-B42T*F42-B43T*F43)

where each row is [3X1]

```
tPC = hPCIJT ∗ invHPPI is W∗V^−1// [9X3][3X3]=[9X3]
```

$H_{PC}^T H_{PP}^{-1}$ [9mX3n]

W*V^-1= [9mX3n]

W11*V1^-1	W21*V2^-1	W31*V3^-1	W41*V4^-1
W12*V1^-1	W22*V2^-1	W32*V3^-1	W42*V4^-1
W13*V1^-1	W23*V2^-1	W33*V3^-1	W43*V4^-1

where each block is [9X3]*[3X3] = [9X3]

$$\mathbf{V}^*$$

$$\text{n blocks}\left\{\underbrace{\begin{matrix} \underset{[3X3]}{\mathbf{V}_1} & 0 & 0 & 0 \\ 0 & \underset{[3X3]}{\mathbf{V}_2} & 0 & 0 \\ 0 & 0 & \underset{[3X3]}{\mathbf{V}_3} & 0 \\ 0 & 0 & 0 & \underset{[3X3]}{\mathbf{V}_4} \end{matrix}}_{\text{n blocks}}\right.$$

$\delta_{\mathbf{b}}$ [3*nX1]

```
[x0,y0,z0]ᵀ
[x1,y1,z1]ᵀ
[x2,y2,z2]ᵀ
[x3,y3,z3]ᵀ
```

$$\mathbf{U}^*$$

$$\text{m blocks}\left\{\underbrace{\begin{matrix} \underset{[9X9]}{\mathbf{U}_1} & 0 & 0 \\ 0 & \underset{[9X9]}{\mathbf{U}_2} & 0 \\ 0 & 0 & \underset{[9X9]}{\mathbf{U}_3} \end{matrix}}_{\text{m blocks}}\right.$$

$\delta_{\mathbf{a}}$ [9*mX1]

```
[9X1]ᵀ
[9X1]ᵀ
[9X1]ᵀ
```

Engels, et al 2006

M images = 3, j
N features = 4, i

$$H_{PP} = J_P^\top J_P \equiv \mathbf{V}^*$$
$$H_{PC} = J_P^\top J_C, \equiv \mathbf{W}^T$$
$$H_{CC} = J_C^\top J_C, \equiv \mathbf{U}^*$$
$$b_P = -J_P^\top f, \equiv \epsilon_\mathbf{b}$$
$$b_C = -J_C^\top f \equiv \epsilon_\mathbf{a}$$

$$(\mathbf{U}^* - \mathbf{W}\,\mathbf{V}^{*-1}\,\mathbf{W}^T)\,\delta_\mathbf{a} = \epsilon_\mathbf{a} - \mathbf{W}\,\mathbf{V}^{*-1}\,\epsilon_\mathbf{b}$$

$$\underbrace{(H_{CC} - H_{PC}^\top H_{PP}^{-1} H_{PC})}_{A}\, dC = \underbrace{b_C - H_{PC}^\top H_{PP}^{-1} b_P}_{B}$$

$$\underset{\boxed{2X9}}{\mathbf{A}_{ij}} = \frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{a}_j} \text{ and } \underset{\boxed{2X3}}{\mathbf{B}_{ij}} = \frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{b}_i}$$

C **P**

Note that $\mathbf{V}^{*-1} = \begin{pmatrix} \mathbf{V}_1^{*-1} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{V}_2^{*-1} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}$

$$\mathbf{J}^T\mathbf{J} =$$

$$\begin{array}{c|ccccccc} & \mathbf{b}_1^T & \mathbf{b}_2^T & \mathbf{b}_3^T & \mathbf{b}_4^T & \mathbf{a}_1^T & \mathbf{a}_2^T & \mathbf{a}_3^T \\ \hline \mathbf{b}_1 & \mathbf{V}_1 & 0 & 0 & 0 & \mathbf{W}_{11}^T & \mathbf{W}_{12}^T & \mathbf{W}_{13}^T \\ \mathbf{b}_2 & 0 & \mathbf{V}_2 & 0 & 0 & \mathbf{W}_{21}^T & \mathbf{W}_{22}^T & \mathbf{W}_{23}^T \\ \mathbf{b}_3 & 0 & 0 & \mathbf{V}_3 & 0 & \mathbf{W}_{31}^T & \mathbf{W}_{32}^T & \mathbf{W}_{33}^T \\ \mathbf{b}_4 & 0 & 0 & 0 & \mathbf{V}_4 & \mathbf{W}_{41}^T & \mathbf{W}_{42}^T & \mathbf{W}_{43}^T \\ \mathbf{a}_1 & \mathbf{W}_{11} & \mathbf{W}_{21} & \mathbf{W}_{31} & \mathbf{W}_{41} & \mathbf{U}_1 & 0 & 0 \\ \mathbf{a}_2 & \mathbf{W}_{12} & \mathbf{W}_{22} & \mathbf{W}_{32} & \mathbf{W}_{42} & 0 & \mathbf{U}_2 & 0 \\ \mathbf{a}_3 & \mathbf{W}_{13} & \mathbf{W}_{23} & \mathbf{W}_{33} & \mathbf{W}_{43} & 0 & 0 & \mathbf{U}_3 \end{array}$$

$\mathbf{U}_j \equiv \sum_{i=1}^4 \mathbf{A}_{ij}^T \mathbf{A}_{ij},$ **[9X9], for 1 image, sum over features**

$\mathbf{V}_i \equiv \sum_{j=1}^3 \mathbf{B}_{ij}^T \mathbf{B}_{ij},$ **[3X3], for 1 feature, sum over images**

$\mathbf{W}_{ij} = \mathbf{A}_{ij}^T \mathbf{B}_{ij}$ **[9X3]**

V* is [3*n X 3*n]

W is [9*m X 3*n]

U* is [9*m X 9*m]

$$\begin{pmatrix} \mathbf{V}^* & \mathbf{W}^T \\ \mathbf{W} & \mathbf{U}^* \end{pmatrix} \begin{pmatrix} \delta_\mathbf{b} \\ \delta_\mathbf{a} \end{pmatrix} = \begin{pmatrix} \epsilon_\mathbf{b} \\ \epsilon_\mathbf{a} \end{pmatrix}$$

1 Initialize λ.

2 **Compute cost function** at initial camera and point configuration.

3 Clear the left hand side matrix A and right hand side vector B.

4 For each track p **(p is feature i of N)**

{

Clear a variable H_{pp} ($\mathbf{V}^*$) to represent block p of H_{PP} ($\mathbf{V}^*$) (in our case a symmetric 3×3 matrix) and a variable b_p ($\epsilon_\mathbf{b}$) to represent part p of b_P ($\epsilon_\mathbf{b}$) (in our case a 3-vector).

(Compute derivatives) For each camera c on track p **(c is image j of M)**

{ Compute error vector f of reprojection in camera c of point p and its Jacobians J_p ($\mathbf{B}$) and J_c ($\mathbf{A}$) with respect to the point parameters (in our case a 2×3 matrix) and the camera parameters (in our case a 2×6 matrix), respectively.

Add $J_p^\top J_p$ ($\mathbf{B}^T\mathbf{B}$) to the upper triangular part of H_{pp} ($\mathbf{V}^*$).

Subtract $J_p^\top f$ ($\mathbf{B}^T$) from b_p ($\epsilon_\mathbf{b}$).

If camera c is free

{

Add $J_c^\top J_c$ ($\mathbf{A}^T\mathbf{A}$) (optionally with an augmented diagonal) to upper triangular part of block (c, c) of left hand side matrix A (in our case a 6×6 matrix).

Compute block (p, c) of H_{PC} as $H_{pc} = J_p^\top J_c$ (in our case a 3×6 matrix) and store it until track is done.

Subtract $J_c^\top f$ from part c of right hand side vector B (related to b_C).

}

} **// end c loop**

Augment diagonal of H_{pp}, which is now accumulated and ready. Invert H_{pp}, taking advantage of the fact that it is a symmetric matrix.

Compute $H_{pp}^{-1} b_p$ and store it in a variable t_p.

(Outer product of track) For each free camera c on track p

{

Subtract $H_{pc}^\top t_p = H_{pc}^\top H_{pp}^{-1} b_p$ from part c of right hand side vector B.

Compute the matrix $H_{pc}^\top H_{pp}^{-1}$ and store it in a variable T_{pc}

For each free camera $c2 \geq c$ on track p

{

Subtract $T_{pc} H_{pc2} = H_{pc}^\top H_{pp}^{-1} H_{pc2}$ from block $(c, c2)$ of left hand side matrix A.

}

}

} **// end p loop**

$$\overbrace{\mathbf{V}^*}^{}\qquad \delta_\mathbf{b}$$

$$\text{n blocks}\left\{\begin{matrix} \underset{[3X3]}{\mathbf{V}_1} & 0 & 0 & 0 \\ 0 & \underset{[3X3]}{\mathbf{V}_2} & 0 & 0 \\ 0 & 0 & \underset{[3X3]}{\mathbf{V}_3} & 0 \\ 0 & 0 & 0 & \underset{[3X3]}{\mathbf{V}_4} \end{matrix}\right. \qquad \begin{matrix} [x0,y0,z0]^T \\ [x1,y1,z1]^T \\ [x2,y2,z2]^T \\ [x3,y3,z3]^T \end{matrix}\ [3*nX1]$$

n blocks

$$\mathbf{U}^* \qquad \delta_\mathbf{a}$$

$$\text{m blocks}\left\{\begin{matrix} \underset{[9X9]}{\mathbf{U}_1} & 0 & 0 \\ 0 & \underset{[9X9]}{\mathbf{U}_2} & 0 \\ 0 & 0 & \underset{[9X9]}{\mathbf{U}_3} \end{matrix}\right. \qquad \begin{matrix} [9X1]^T \\ [9X1]^T \\ [9X1]^T \end{matrix}\ [9*mX1]$$

m blocks

Engels, et al 2006

Note that $\mathbf{V}^{*-1} = \begin{pmatrix} \mathbf{V}_1^{*-1} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{V}_2^{*-1} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}$

$$H_{PP} = J_P^\top J_P \equiv \mathbf{V}^*$$
$$H_{PC} = J_P^\top J_C, \equiv \mathbf{W}^T$$
$$H_{CC} = J_C^\top J_C, \equiv \mathbf{U}^*$$
$$b_P = -J_P^\top f, \equiv \epsilon_\mathbf{b}$$
$$b_C = -J_C^\top f \equiv \epsilon_\mathbf{a}$$

$\mathbf{V}_i \equiv \sum_{j=1}^{3} \mathbf{B}_{ij}^T \mathbf{B}_{ij},$ for 1 feature, sum over images

$\mathbf{U}_j \equiv \sum_{i=1}^{4} \mathbf{A}_{ij}^T \mathbf{A}_{ij},$ for 1 image, sum over features

$\delta_\mathbf{b}$ is dP is [3*n_features X 1]

$\epsilon_\mathbf{b}$ is b_P is [3*n_features X 1]

$\delta_\mathbf{a}$ is dC is [9*m_images X 1]

$\epsilon_\mathbf{a}$ is b_C is [9*m_images X 1]

$$(\mathbf{U}^* - \mathbf{W}\ \mathbf{V}^{*-1}\ \mathbf{W}^T)\ \delta_\mathbf{a} = \epsilon_\mathbf{a} - \mathbf{W}\ \mathbf{V}^{*-1}\ \epsilon_\mathbf{b}$$

$$\underbrace{(H_{CC} - H_{PC}^\top H_{PP}^{-1} H_{PC})}_{A}\, dC = \underbrace{b_C - H_{PC}^\top H_{PP}^{-1} b_P}_{B}$$

$$\mathbf{V}^*\ \delta_\mathbf{b} = \epsilon_\mathbf{b} - \mathbf{W}^T\ \delta_\mathbf{a}$$

$$dP = H_{PP}^{-1} b_P - H_{PP}^{-1} H_{PC} dC.$$

= tP - tPC^T * dC

[3nX1] [3nX9m] [9mX1]

Engels

5 (Optional) Fix gauge by freezing appropriate coordinates and thereby reducing the linear system with a few dimensions.

6 **(Linear Solving)** Cholesky factor the left hand side matrix B and solve for dC. Add frozen coordinates back in.

7 **(Back-substitution)** For each track p

{

Start with point update for this track $dp = t_p$.

For each camera c on track p

{

Subtract $T_{pc}^\top dc$ from dp (where dc is the update for camera c).

}

Compute updated point.

}

8 **Compute the cost function** for the updated camera and point configuration.

9 If cost function has improved, accept the update step, decrease λ and go to Step 3 (unless converged, in which case quit).

10 Otherwise, increase λ and go to Step 3 (unless exceeded the maximum number of iterations, in which case quit).

```
tP = hPPI^-1 * bPI is V^-1*bPI
```
[3nX1]

-(V1^-1)*(B11T*F11-B12T*F12-B13T*F13)
-(V2^-1)*(B21T*F21-B22T*F22-B23T*F23)
-(V3^-1)*(B31T*F31-B32T*F32-B33T*F33)
-(V4^-1)*(B41T*F41-B42T*F42-B43T*F43)

where each row is [3X1]

```
tPC^T
```
[3nX9m]

V1^-1*W11T	**V1^-1*W12T**	**V1^-1*W13T**
V2^-1*W21T	**V2^-1*W22T**	**V2^-1*W23T**
V3^-1*W31T	**V3^-1*W32T**	**V3^-1*W33T**
V4^-1*W41T	**V4^-1*W42T**	**V4^-1*W43T**

where each block is = [3X9]

```
dC
```
[9mX1]

each block is = [9X1]

Bill **Triggs**, Philip Mclauchlan, Richard Hartley, Andrew Fitzgibbon.
Bundle Adjustment – A Modern Synthesis.
International Workshop on Vision Algorithms,
Sep **2000**, Corfu, Greece. pp.298–372,
10.1007/3-540-44480-7_21 . inria-00548290 **see Appendix B, and page 23…**

$$(\mathbf{U}^* - \mathbf{W}\boxed{\mathbf{V}^{*-1}}\mathbf{W}^T)\,\delta_{\mathbf{a}} = \epsilon_{\mathbf{a}} - \mathbf{W}\,\mathbf{V}^{*-1}\,\epsilon_{\mathbf{b}}$$

$$\underbrace{\underbrace{(H_{CC} - H_{PC}^\top \boxed{H_{PP}^{-1}} H_{PC})}_{A}}_{\overline{\mathsf{D}}}\, dC = \underbrace{b_C - H_{PC}^\top H_{PP}^{-1} b_P}_{B}$$

6.1 The Schur Complement and the Reduced Bundle System

Schur complement: Consider the following block triangular matrix factorization:

$$\mathsf{M} = \begin{pmatrix} \mathsf{A} & \mathsf{B} \\ \mathsf{C} & \mathsf{D} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ \mathsf{C}\,\mathsf{A}^{-1} & 1 \end{pmatrix} \begin{pmatrix} \mathsf{A} & 0 \\ 0 & \overline{\mathsf{D}} \end{pmatrix} \begin{pmatrix} 1 & \mathsf{A}^{-1}\mathsf{B} \\ 0 & 1 \end{pmatrix}, \qquad \overline{\mathsf{D}} \equiv \mathsf{D} - \mathsf{C}\,\mathsf{A}^{-1}\mathsf{B} \tag{16}$$

$$\begin{pmatrix} \mathsf{A} & \mathsf{B} \\ \mathsf{C} & \mathsf{D} \end{pmatrix}^{-1} = \begin{pmatrix} 1 & -\mathsf{A}^{-1}\mathsf{B} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \mathsf{A}^{-1} & 0 \\ 0 & \overline{\mathsf{D}}^{-1} \end{pmatrix} \begin{pmatrix} 1 & 0 \\ -\mathsf{C}\,\mathsf{A}^{-1} & 1 \end{pmatrix} = \begin{pmatrix} \mathsf{A}^{-1} + \mathsf{A}^{-1}\mathsf{B}\,\overline{\mathsf{D}}^{-1}\mathsf{C}\,\mathsf{A}^{-1} & -\mathsf{A}^{-1}\mathsf{B}\,\overline{\mathsf{D}}^{-1} \\ -\overline{\mathsf{D}}^{-1}\mathsf{C}\,\mathsf{A}^{-1} & \overline{\mathsf{D}}^{-1} \end{pmatrix} \tag{17}$$

Here A must be square and invertible, and for (17), the whole matrix must also be square and invertible. $\overline{\mathsf{D}}$ is called the **Schur complement** of A in M. If both A and D are invertible, complementing on D rather than A gives **$\overline{\mathsf{D}}$, the Schur complement, is HPP is V*.**

$$\begin{pmatrix} \mathsf{A} & \mathsf{B} \\ \mathsf{C} & \mathsf{D} \end{pmatrix}^{-1} = \begin{pmatrix} \overline{\mathsf{A}}^{-1} & -\overline{\mathsf{A}}^{-1}\mathsf{B}\,\mathsf{D}^{-1} \\ -\mathsf{D}\,\mathsf{C}\,\overline{\mathsf{A}}^{-1} & \mathsf{D}^{-1} + \mathsf{D}^{-1}\mathsf{C}\,\overline{\mathsf{A}}^{-1}\mathsf{B}\,\mathsf{D}^{-1} \end{pmatrix}, \qquad \overline{\mathsf{A}} = \mathsf{A} - \mathsf{B}\boxed{\mathsf{D}^{-1}}\mathsf{C}$$

Equating upper left blocks gives the **Woodbury formula**:

$$(\mathsf{A} \pm \mathsf{B}\,\mathsf{D}^{-1}\mathsf{C})^{-1} = \mathsf{A}^{-1} \mp \mathsf{A}^{-1}\mathsf{B}\,(\mathsf{D} \pm \mathsf{C}\,\mathsf{A}^{-1}\mathsf{B})^{-1}\,\mathsf{C}\,\mathsf{A}^{-1} \tag{18}$$

This is the usual method of updating the inverse of a nonsingular matrix A after an update (especially a low rank one) $\mathsf{A} \to \mathsf{A} \pm \mathsf{B}\,\mathsf{D}^{-1}\mathsf{C}$. (See §8.1).

Bill **Triggs**, Philip Mclauchlan, Richard Hartley, Andrew Fitzgibbon.
Bundle Adjustment – A Modern Synthesis.
International Workshop on Vision Algorithms,
Sep **2000**, Corfu, Greece. pp.298–372,
10.1007/3-540-44480-7_21 . inria-00548290

see Appendix B, and page 23…

$\mathsf{L} \;=\; \textbf{profile\_cholesky\_decomp}(\mathsf{A})$
  for $i = 1$ **to** n **do**
    for $j = \text{first}(i)$ **to** i **do**

$$a \;=\; \mathsf{A}_{ij} - \sum_{k=\max(\text{first}(i),\text{first}(j))}^{j-1} \mathsf{L}_{ik}\,\mathsf{L}_{jk}$$

$$\mathsf{L}_{ij} \;=\; (j<i)\ ?\ a\,/\,\mathsf{L}_{jj}\ :\ \sqrt{a}$$

$\mathsf{x} \;=\; \textbf{profile\_cholesky\_forward\_subs}(\mathsf{A},\mathsf{b})$
  for $i = \text{first}(\mathsf{b})$ **to** n **do**

$$\mathsf{x}_i \;=\; \left(\mathsf{b}_i - \sum_{k=\max(\text{first}(i),\text{first}(\mathsf{b}))}^{i-1} \mathsf{L}_{ik}\,\mathsf{x}_k\right) /\, \mathsf{L}ii$$

$\mathsf{y} \;=\; \textbf{profile\_cholesky\_back\_subs}(\mathsf{A},\mathsf{x})$
  $\mathsf{y} \;=\; \mathsf{x}$
  for $i = \text{last}(\mathsf{b})$ **to** 1 **step** -1 **do**
    for $k = \max(\text{first}(i), \text{first}(\mathsf{y}))$ **to** i **do**
      $\mathsf{y}_k \;=\; \mathsf{y}_k - \mathsf{y}_i\,\mathsf{L}_{ik}$
    $\mathsf{y}_i \;=\; \mathsf{y}_i\,/\,\mathsf{L}_{ii}$

Figure 10: A complete implementation of profile Cholesky decomposition.

cholesky:
_A_ = L ∗ D ∗ L^T
= L ∗ √D **∗** √D **∗ L^T**
let C = L ∗ √D
then **_A_** = **C ∗ C^T**
aside: L is invertible if none of its diagonal elements are 0.

for _A_=L ∗ L*
(_A_)^-1 = (L*) ∗ (L ∗ L*)^-1

but usually, for A ∗ x= b:
(1) A=L ∗ L*
(2) L ∗ y = b ==> y via forward subst
(3) L* ∗ x = y ==> x via backward subst

http://users.ics.forth.gr/~argyros/mypapers/2004_08_tr340_forth_sba.pdf

The Design and Implementation of a Generic Sparse Bundle Adjustment Software Package Based on the Levenberg-Marquardt Algorithm†

Manolis I.A. Lourakis and Antonis A. Argyros

In all cases, the function pointed to by `proj` is assumed to estimate in `xij` the projection in image `j` of the point `i`. Arguments `aj` and `bi` are respectively the parameters of the `j`-th camera and `i`-th point. In other words, `proj` implements the parameterizing function $\mathbf{Q}()$. Similarly, `projac` is assumed to compute in `Aij` and `Bij` the functions $\frac{\partial \mathbf{Q}(\mathbf{a}_j, \mathbf{b}_i)}{\partial \mathbf{a}_j}$ and $\frac{\partial \mathbf{Q}(\mathbf{a}_j, \mathbf{b}_i)}{\partial \mathbf{b}_i}$, i.e. the jacobians with respect to `aj` and `bi` of the projection of point `i` in image `j`. If `projac` is NULL, the jacobians are

The employed world coordinate frame is taken to be aligned with the initial camera location. All subsequent camera motions are defined relative to the initial location, through the combination of a 3D rotation and a 3D translation. A 3D rotation by an angle θ about a unit vector $\mathbf{u} = (u_1, u_2, u_3)^T$ is represented by the quaternion $\mathbf{R} = (\cos(\frac{\theta}{2}),\ u_1 \sin(\frac{\theta}{2}),\ u_2 \sin(\frac{\theta}{2}),\ u_3 \sin(\frac{\theta}{2}))$ [26]. A 3D translation is defined by a vector $\mathbf{t}$. A 3D point is represented by its Euclidean coordinate vector $\mathbf{M}$. Thus, the parameters of each camera j and point i are $\mathbf{a}_j = (\mathbf{R}_j, \mathbf{t}_j{}^T)^T$ and $\mathbf{b}_i = \mathbf{M}_i$, respectively. With the previous definitions, the predicted projection of point i on image j is

$$\mathbf{Q}(\mathbf{a}_j, \mathbf{b}_i) = \mathbf{K}\ (\mathbf{R}_j\ \mathbf{N}_i\ \mathbf{R}_j^{-1}\ +\ \mathbf{t}_j), \tag{28}$$

where $\mathbf{K}$ is the 3×3 intrinsic camera calibration matrix and $\mathbf{N}_i = (0,\ \mathbf{M}_i{}^T)$ is the vector quaternion corresponding to the 3D point $\mathbf{M}_i$. The expression $\mathbf{R}_j\ \mathbf{N}_i\ \mathbf{R}_j^{-1}$ corresponds to point $\mathbf{M}_i$ rotated by an angle θ_j about unit vector $\mathbf{u}_j$, as specified by the quaternion $\mathbf{R}_j$. Source file `eucsbademo.c` accompanying the `sba` package imhttp://users.ics.forth.gr/~argyros/mypapers/2004_08_tr340_forth_sba.pdf

The Design and Implementation of a Generic Sparse Bundle Adjustment Software Package Based on the Levenberg-Marquardt Algorithm†

Manolis I.A. Lourakis and Antonis A. Argyros

algorithm[3]. This procedure can be embedded into the LM algorithm of section 2 at the point indicated by the rectangular box in Fig. 1, leading to a sparse bundle adjustment algorithm.

Figure 2: Algorithm for solving the sparse normal equations arising in generic bundle adjustment; see text for details.

Input: The current parameter vector partitioned into m camera parameter vectors $\mathbf{a}_j$ and n 3D point parameter vectors $\mathbf{b}_i$, a function $\mathbf{Q}$ employing the $\mathbf{a}_j$ and $\mathbf{b}_i$ to compute the predicted projections $\hat{\mathbf{x}}_{ij}$ of the i-th point on the j-th image, the observed image point locations $\mathbf{x}_{ij}$ and a damping term μ for LM.

Output: The solution δ to the normal equations involved in LM-based bundle adjustment.

Algorithm:

Compute the derivative matrices $\mathbf{A}_{ij} := \frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{a}_j} = \frac{\partial \mathbf{Q}(\mathbf{a}_j, \mathbf{b}_i)}{\partial \mathbf{a}_j}$, $\mathbf{B}_{ij} := \frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{b}_i} = \frac{\partial \mathbf{Q}(\mathbf{a}_j, \mathbf{b}_i)}{\partial \mathbf{b}_i}$
and the error vectors $\epsilon_{ij} := \mathbf{x}_{ij} - \hat{\mathbf{x}}_{ij}$,
where i and j assume values in $\{1, \ldots, n\}$ and $\{1, \ldots, m\}$ respectively.

Compute the following auxiliary variables:

$$\mathbf{U}_j := \sum_i \mathbf{A}_{ij}{}^T \mathbf{\Sigma}_{\mathbf{x}_{ij}}^{-1} \mathbf{A}_{ij} \quad \mathbf{V}_i := \sum_j \mathbf{B}_{ij}{}^T \mathbf{\Sigma}_{\mathbf{x}_{ij}}^{-1} \mathbf{B}_{ij} \quad \mathbf{W}_{ij} := \mathbf{A}_{ij}{}^T \mathbf{\Sigma}_{\mathbf{x}_{ij}}^{-1} \mathbf{B}_{ij}$$

$$\epsilon_{\mathbf{a}_j} := \sum_i \mathbf{A}_{ij}{}^T \mathbf{\Sigma}_{\mathbf{x}_{ij}}^{-1} \epsilon_{ij} \qquad \epsilon_{\mathbf{b}_i} := \sum_j \mathbf{B}_{ij}{}^T \mathbf{\Sigma}_{\mathbf{x}_{ij}}^{-1} \epsilon_{ij}$$

Augment $\mathbf{U}_j$ and $\mathbf{V}_i$ by adding μ to their diagonals to yield $\mathbf{U}_j^*$ and $\mathbf{V}_i^*$.

Compute $\mathbf{Y}_{ij} := \mathbf{W}_{ij} \mathbf{V}_i^{*-1}$.

Compute $\delta_{\mathbf{a}}$ from $\mathbf{S}\, (\delta_{\mathbf{a}_1}{}^T, \delta_{\mathbf{a}_2}{}^T, \ldots, \delta_{\mathbf{a}_m}{}^T)^T = (\mathbf{e}_1{}^T, \mathbf{e}_2{}^T, \ldots, \mathbf{e}_m{}^T)^T$,
where $\mathbf{S}$ is a matrix consisting of $m \times m$ blocks; block jk is defined by
$\mathbf{S}_{jk} = \delta_{jk} \mathbf{U}_j^* - \sum_i \mathbf{Y}_{ij} \mathbf{W}_{ik}{}^T$, where δ_{jk} is Kronecker's delta
and
$\mathbf{e}_j = \epsilon_{\mathbf{a}_j} - \sum_i \mathbf{Y}_{ij}\, \epsilon_{\mathbf{b}_i}$.

Compute each $\delta_{\mathbf{b}_i}$ from the equation $\delta_{\mathbf{b}_i} = \mathbf{V}_i^{*-1} \left(\epsilon_{\mathbf{b}_i} - \sum_j \mathbf{W}_{ij}{}^T\, \delta_{\mathbf{a}_j}\right)$.

Form δ as $(\delta_{\mathbf{a}}{}^T,\ \delta_{\mathbf{b}}{}^T)^T$.

http://users.ics.forth.gr/~argyros/mypapers/2004_08_tr340_forth_sba.pdf

The Design and Implementation of a Generic Sparse Bundle Adjustment Software Package Based on the Levenberg-Marquardt Algorithm†

Manolis I.A. Lourakis and Antonis A. Argyros

$$\mathbf{p}_{new} := \mathbf{p} + \delta_{\mathbf{p}};$$
$$\rho := (||\epsilon_{\mathbf{p}}||^2 - ||\mathbf{x} - f(\mathbf{p}_{new})||^2)/(\delta_{\mathbf{p}}^T(\mu\delta_{\mathbf{p}} + \mathbf{g}));$$

Figure 1: Levenberg-Marquardt non-linear least squares algorithm; see text and [16, 20] for details. The reason for enclosing a statement in a rectangular box will be explained in section 3.

Input: A vector function $f : \mathcal{R}^m \rightarrow \mathcal{R}^n$ with $n \geq m$, a measurement vector $\mathbf{x} \in \mathcal{R}^n$ and an initial parameters estimate $\mathbf{p}_0 \in \mathcal{R}^m$.
Output: A vector $\mathbf{p}^+ \in \mathcal{R}^m$ minimizing $||\mathbf{x} - f(\mathbf{p})||^2$.
Algorithm:

```
k := 0; ν := 2; p := p_0;
A := J^T J; ε_p := x − f(p); g := J^T ε_p;
stop:=(||g||_∞ ≤ ε_1); μ := τ * max_{i=1,...,m}(A_ii);
while (not stop) and (k < k_max)
    k := k + 1;
    repeat
        [Solve (A + μI)δ_p = g;]
        if (||δ_p|| ≤ ε_2||p||)
            stop:=true;
        else
            p_new := p + δ_p;
            ρ := (||ε_p||^2 − ||x − f(p_new)||^2)/(δ_p^T(μδ_p + g));
            if ρ > 0
                p = p_new;
                A := J^T J; ε_p := x − f(p); g := J^T ε_p;
                stop:=(||g||_∞ ≤ ε_1);
                μ := μ * max(1/3, 1 − (2ρ − 1)^3); ν := 2;
            else
                μ := μ * ν; ν := 2 * ν;
            endif
        endif
    until (ρ > 0) or (stop)
endwhile
```

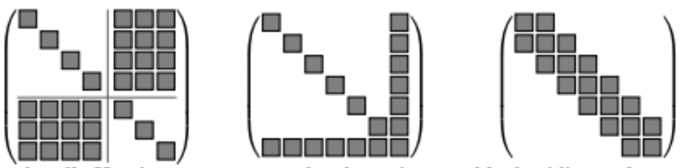
gain ratio

[3n X 1]

factoring an arrowhead Hessian matrix to get a reduced camera system or reduced structure system.

Bill Triggs, Philip Mclauchlan, Richard Hartley, Andrew Fitzgibbon.
Bundle Adjustment – A Modern Synthesis. International Workshop on Vision Algorithms, Sep 2000, Corfu, Greece. pp.298–372, 10.1007/3-540-44480-7_21 . inria-00548290

We seek variable orderings that approximately minimize the total operation count or fill-in over the whole elimination chain. For many problems a suitable ordering can be fixed in advance, typically giving one of a few standard pattern matrices such as band or arrowhead matrices, perhaps with such structure at several levels.


(25)

bundle Hessian | arrowhead matrix | block tridiagonal matrix

The most prominent pattern structure in bundle adjustment is the primary subdivision of the Hessian into structure and camera blocks. To get the reduced camera system (19), we treat the Hessian as an arrowhead matrix with a broad final column containing all of the camera parameters. Arrowhead matrices are trivial to factor or reduce by block 2×2 Schur complementation, *c.f.* (16, 19). For bundle problems with many independent images and only a few features, one can also complement on the image parameter block to get a reduced *structure* system.

http://users.ics.forth.gr/~lourakis/sba/PRCV_colloq.pdf

Bundle adjustment gone public
Manolis Lourakis

- A few interesting practical situations violate its underlying assumption regarding the problem's sparsity pattern, rendering it inapplicable. E.g., fixed but unknown intrinsics shared by <u>all</u> cameras
- Example: Hessians corresponding to BA for motion and structure (left) and BA for motion, structure and shared intrinsics (right)

applying local updates rather than global

How to Avoid Singularity For Euler Angle Set?

Puneet Singla*, Daniele Mortari†, and John L. Junkins‡

Since all rotations are performed about the principal axes of the reference frame, we define $\mathbf{M}_i = \exp(-[\tilde{e}_i]\theta_i)$ as an elementary rotation matrix about the $\mathbf{e}_i$-body axis. Here, $[\tilde{e}_i]$ represents the skew-symmetric cross product matrix given by the following expression:

$$[\tilde{\mathbf{a}}] = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \tag{1}$$

From the expression of $\mathbf{M}_i$, we can construct the following three elementary rotation matrices:

$$\mathbf{M}_1(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{bmatrix} \tag{2}$$

$$\mathbf{M}_2(\theta) = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \tag{3}$$

$$\mathbf{M}_3(\theta) = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4}$$

if i,j, and k, indicate the coordinate axes about which each subsequent rotation is performed, that is, they can be any integer from 1-3, provided that $i \neq j$ and $j \neq k$, are satisfied then the resultant direction cosine matrix can be written as

$$\mathbf{C}_{ijk}(\theta_1, \theta_2, \theta_3) = \mathbf{M}_k(\theta_3)\,\mathbf{M}_j(\theta_2)\,\mathbf{M}_i(\theta_1) \tag{5}$$

applying local updates rather than global

How to Avoid Singularity For Euler Angle Set?

Puneet Singla*, Daniele Mortari†, and John L. Junkins‡

It is a fundamental topological fact that singularities can never be eliminated in any 3-dimensional representation of orientation. But we can avoid this singularity by describing the attitude at a particular instant by the Euler angle set *which is farthest away from singularity*. In this paper, we present an algorithm to switch between different sets of Euler angles to avoid this singularity.

Table 1: Classical Parameterizations of Attitude Rotation Matrix

Paramet-rization	Dimension	Attitude Matrix	Kinematic Equations	Singularities	Constraints
DCM, (C_{ij})	9	$\mathbf{C} = [C_{ij}]$	$\dot{\mathbf{C}} = -[\tilde{\boldsymbol{\omega}}]\mathbf{C}$	None	$\mathbf{C}^T\mathbf{C} = \mathbf{I}$
Euler Angles EA (θ_i)	3	$\mathbf{C} = \begin{bmatrix} \text{transcendental} \\ \text{functions of} \\ \theta_i's \end{bmatrix}$	$\dot{\boldsymbol{\theta}} = \begin{bmatrix} \text{transcendental} \\ \text{functions of} \\ \theta_i's \end{bmatrix} \boldsymbol{\omega}$	$\theta_2 = \pm\frac{\pi}{2}$	None
, Euler-Rodrigues Symmetric Parameters ERSP (q_i)	4	$\mathbf{C} = \begin{bmatrix} \text{algebraic} \\ \text{functions of} \\ q_i's \end{bmatrix}$	$\dot{\mathbf{q}} = \begin{bmatrix} \text{linear} \\ \text{functions of} \\ q_i's \end{bmatrix} \begin{Bmatrix} 0 \\ \boldsymbol{\omega} \end{Bmatrix}$	None	$\mathbf{q}^T\mathbf{q} = 1$
RP (r_i)	3	$\mathbf{C} = \begin{bmatrix} \text{quadratic} \\ \text{functions of} \\ r_i's \end{bmatrix}$	$\dot{\mathbf{r}} = \begin{bmatrix} \text{non-linear} \\ \text{functions of} \\ r_i's \end{bmatrix} \boldsymbol{\omega}$	$\phi = \pm\pi$	None
MRP (σ_i)	3	$\mathbf{C} = \begin{bmatrix} \text{quartic} \\ \text{functions of} \\ \sigma_i's \end{bmatrix}$	$\dot{\boldsymbol{\sigma}} = \begin{bmatrix} \text{non-linear} \\ \text{functions of} \\ \sigma_i's \end{bmatrix} \boldsymbol{\omega}$	$\phi = \pm 2\pi$	None

applying local updates rather than global

How to Avoid Singularity For Euler Angle Set?

Puneet Singla*, Daniele Mortari†, and John L. Junkins‡

second reason comes out from the capability of the Shuster's Method of Sequential Rotations[4] (MSR) to avoid the *un-avoidable* singularity affecting all the minimum attitude parameter. Not only the original QUEST[4] algorithm has taken advantage from the MSR technique, but also some recent attitude determination approaches, like ESOQ2[5] and OLAE[6].

Snavely's GitHub bundler_sfm/lib/sba-1.5/sba_levmar.c