Lab 1 – Camera geometry with Eigen

25.01.2018

Part 1: Introduction to Eigen

Eigen 3

- C++ library for linear algebra
  - http://eigen.tuxfamily.org/

- “Template library” – “Header only”
  - Multi platform, no linking!

- Good documentation!
  - https://eigen.tuxfamily.org/dox/
  - https://eigen.tuxfamily.org/dox/group__TutorialMatrixClass.html
  - https://eigen.tuxfamily.org/dox/AsciiQuickReference.txt

Get to know Eigen

- Create a few vectors and matrices
  - $\boldsymbol{t} = \begin{bmatrix} 1.0 \\ 0.0 \\ 3.0 \end{bmatrix}$
  - $A = \begin{bmatrix} 1.0 & 0.0 & 3.0 \\ 4.0 & 5.0 & 6.0 \\ 7.0 & 8.0 & 9.0 \end{bmatrix}$
  - $I = \begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$
  - $T = \begin{bmatrix} A & \boldsymbol{t} \\ 0 & 1 \end{bmatrix}$
  - $B = A^T$

Get to know Eigen

- Play with coefficients
  - $t = \begin{bmatrix} 1.0 \\ 0.0 \\ 3.0 \end{bmatrix}$
  - $A = \begin{bmatrix} 1.0 & 0.0 & 3.0 \\ 4.0 & 5.0 & 6.0 \\ 7.0 & 8.0 & 9.0 \end{bmatrix}$

Set to 2.0

Get to know Eigen

- Block operations
  - Create a vector from a row in $A = \begin{bmatrix} 1.0 & 2.0 & 3.0 \\ 4.0 & 5.0 & 6.0 \\ 7.0 & 8.0 & 9.0 \end{bmatrix}$
  - Create a vector from a column in $A = \begin{bmatrix} 1.0 & 2.0 & 3.0 \\ 4.0 & 5.0 & 6.0 \\ 7.0 & 8.0 & 9.0 \end{bmatrix}$
  - Create a matrix from the middle 2x2 sum matrix in T
  - What happens with A and T if these vectors/matrices are altered?

Get to know Eigen

- Matrix and vector arithmetic
  - Add two vectors/matrices
  - Multiply to matrices
  - Take the dot product between two vectors
  - Take elementwise multiplication between two matrices
- Homogenous representation
  - Try homogenous() and hnormalize()

Get to know Eigen

- Reductions
  - Take the sum of all elements in a matrix

  - Compute the minimum value in a matrix
    - Find its position

  - Create a vector that is the minimum of each column in a matrix

  - Find the L1- and L2 norm of a vector

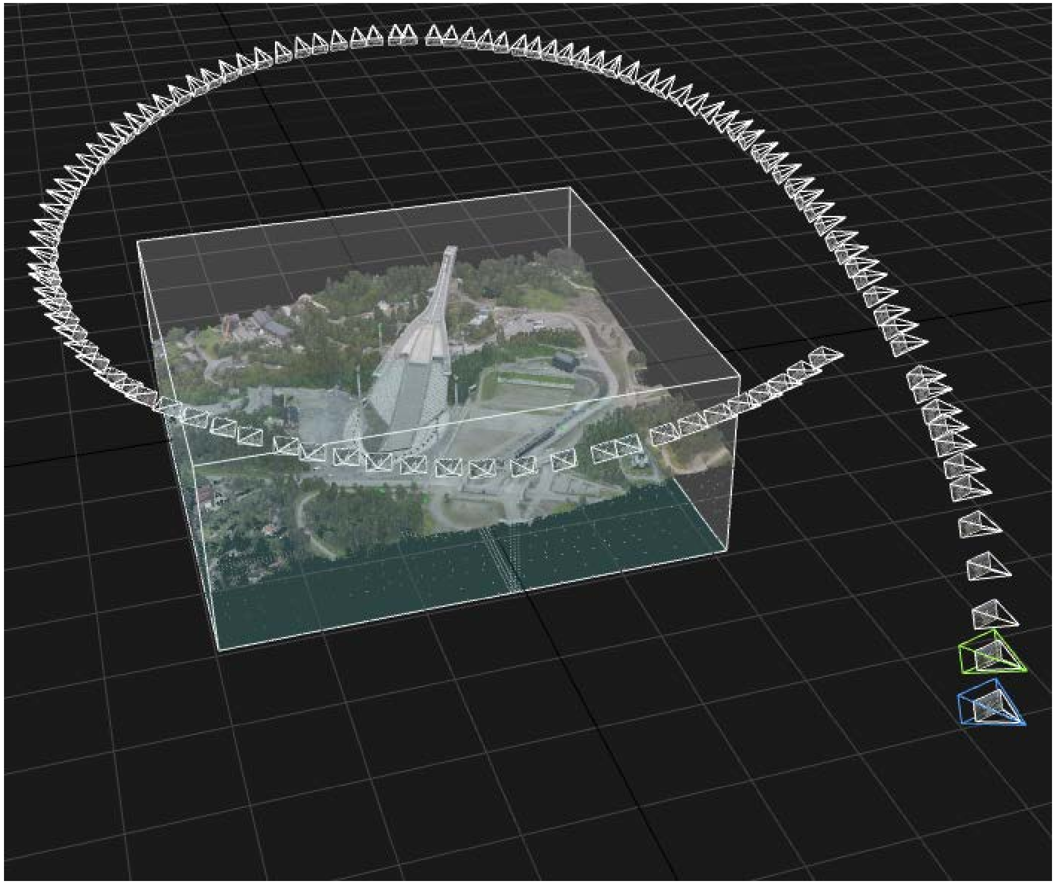
  - Find the number of elements that is greater than a given value

Part 2 – Camera geometry

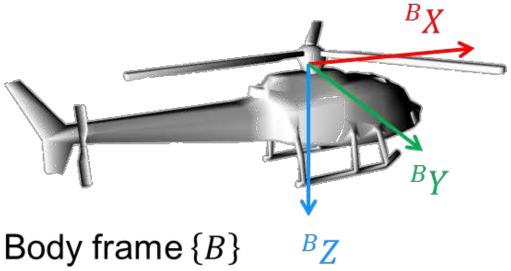
Holmenkollen dataset

- 110 images taken from helicopter

- For each image
  - Intrinsic calibration
  - Helicopter pose in geographical coordinates
  - Camera pose relative to helicopter

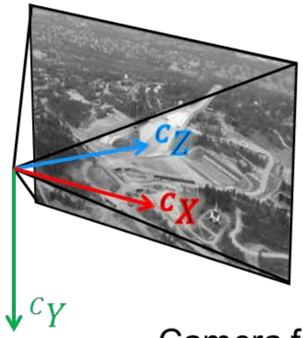


Coordinate systems



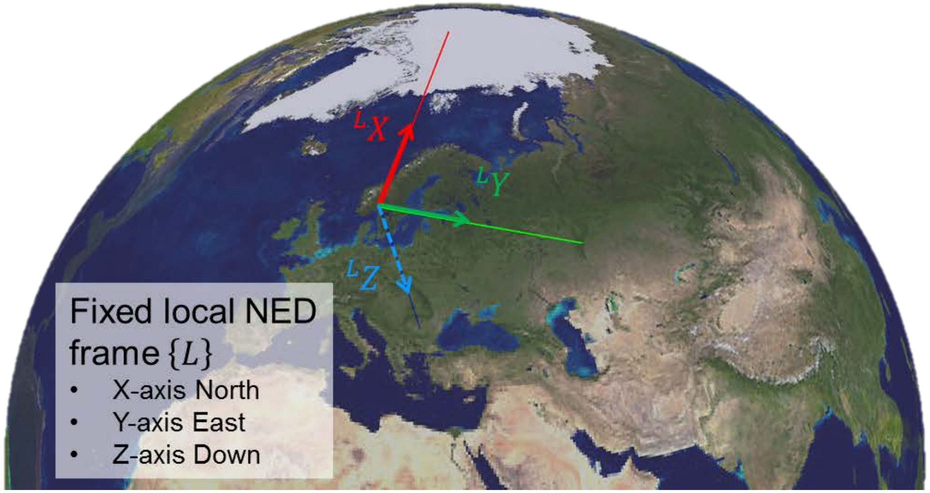
Body frame $\{B\}$

- X-axis forward
- Y-axis to the right
- Z-axis down



Camera frame $\{C\}$

- X-axis to the right
- Y-axis down
- Z-axis forward



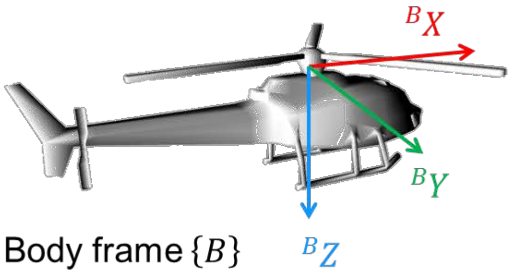
Fixed local NED frame $\{L\}$

- X-axis North
- Y-axis East
- Z-axis Down

Coordinate systems

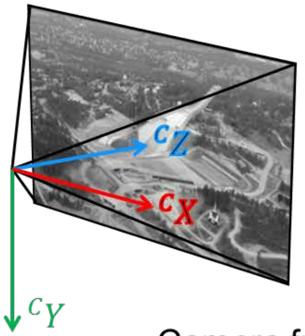
We represent the orientation using euler angles as

$$R = R_z(z\_rot)R_y(y\_rot)R_x(x\_rot)$$



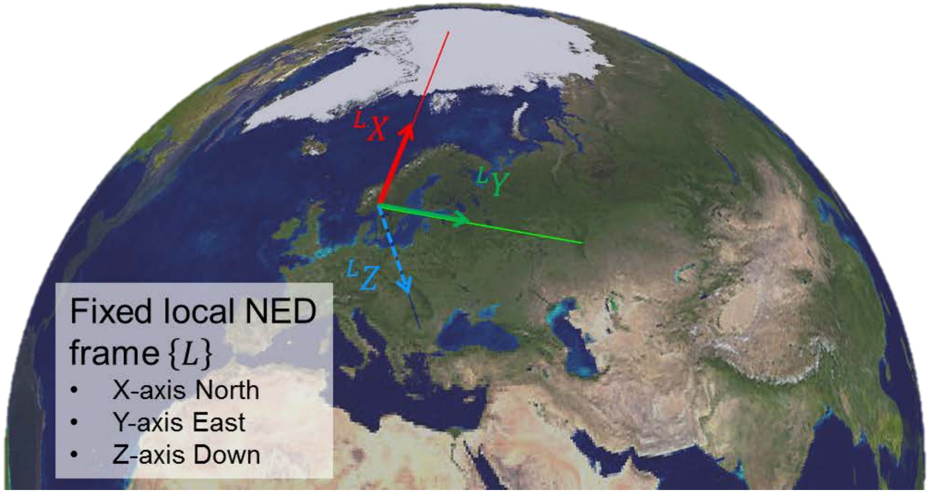
Body frame $\{B\}$

- X-axis forward
- Y-axis to the right
- Z-axis down



Camera frame $\{C\}$

- X-axis to the right
- Y-axis down
- Z-axis forward

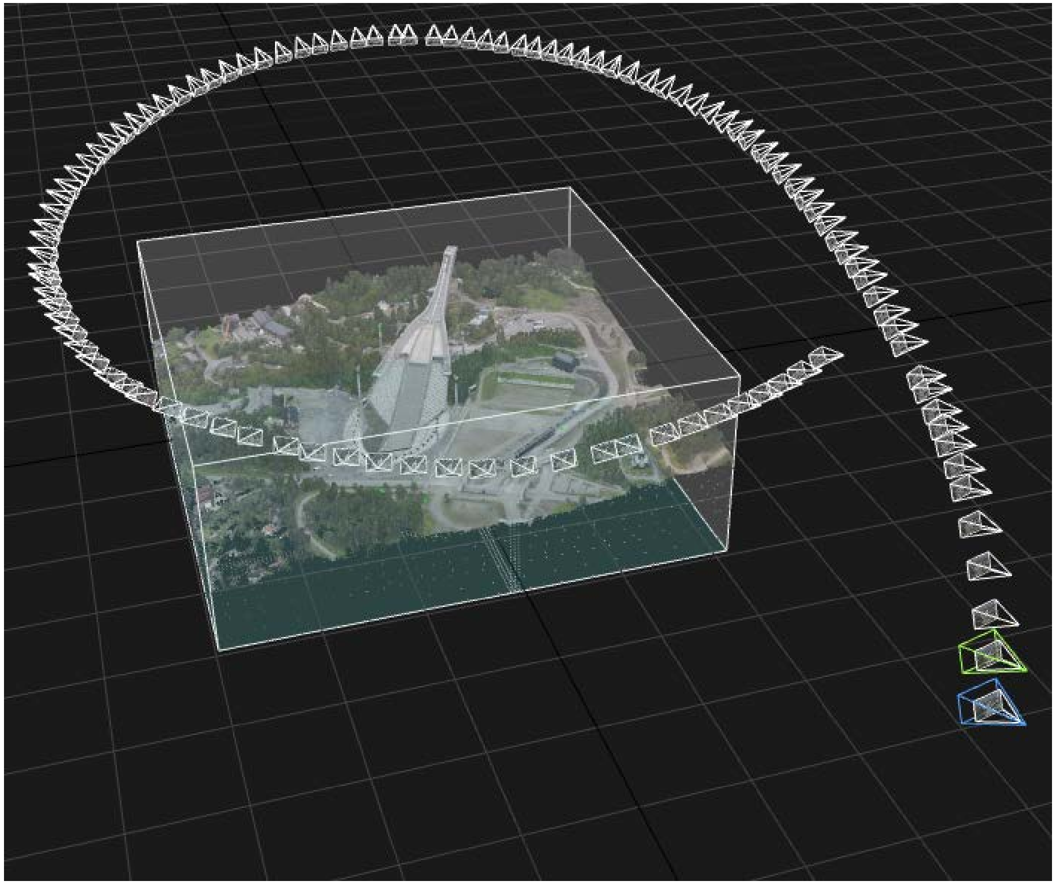


Fixed local NED frame $\{L\}$

- X-axis North
- Y-axis East
- Z-axis Down

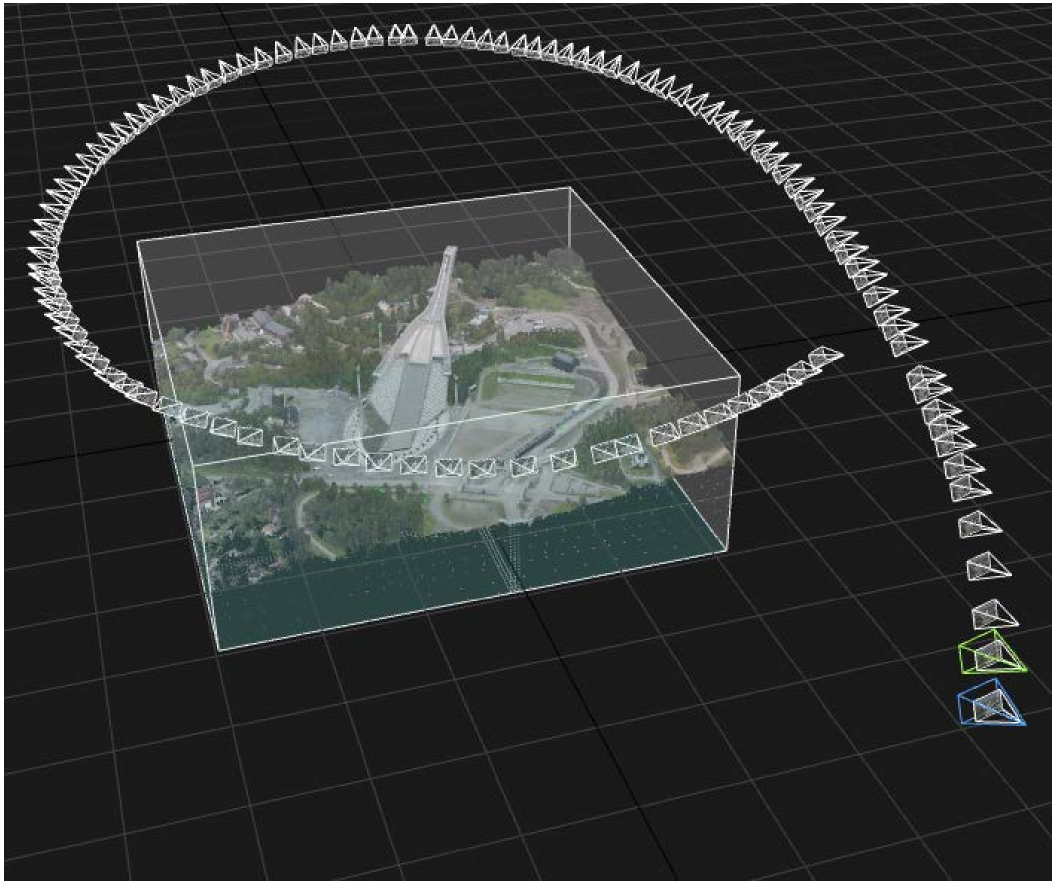
Problem

- Project (real geographical) world points into the images



Solution

- Download lab_1_2
- Follow the steps in lab_1_2.cpp



Extra

- Project other points into the images
  - http://www.norgeskart.no

- Create a virtual camera
  - Project points
  - Visualize in 3D