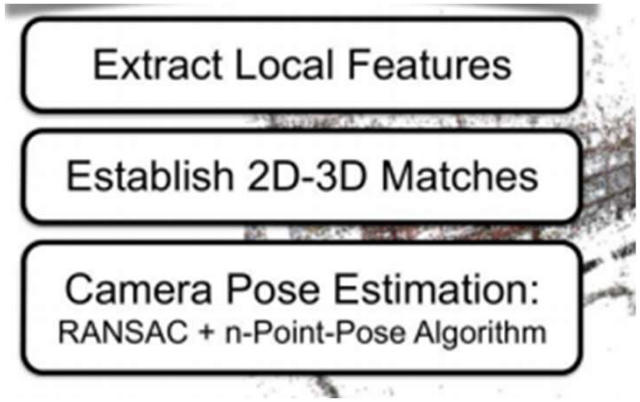
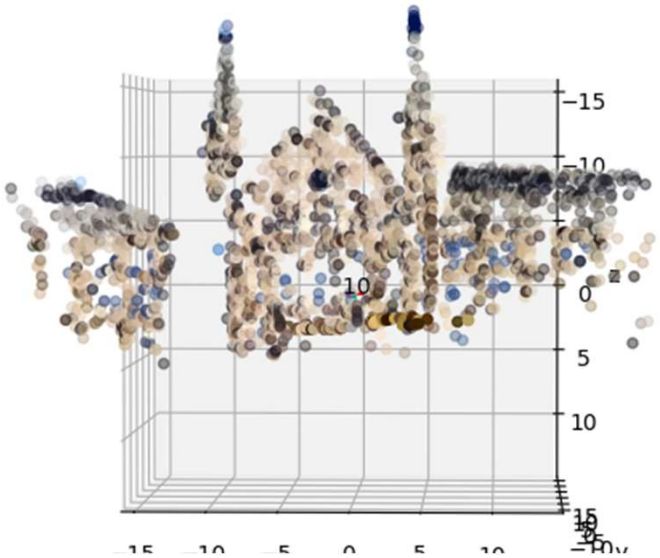
# Robotic Vision

## Project 2

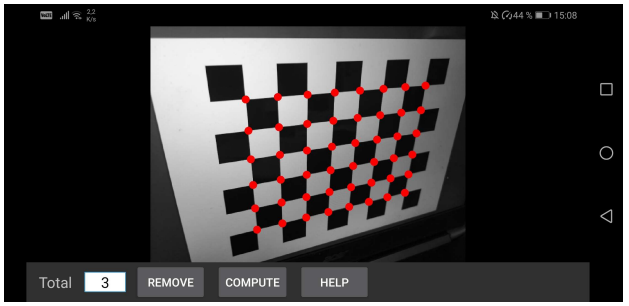
Nadia Garson Wangberg , Ludvig Løken Sundøen

# Structure-based visual localization




# Camera calibration + extras

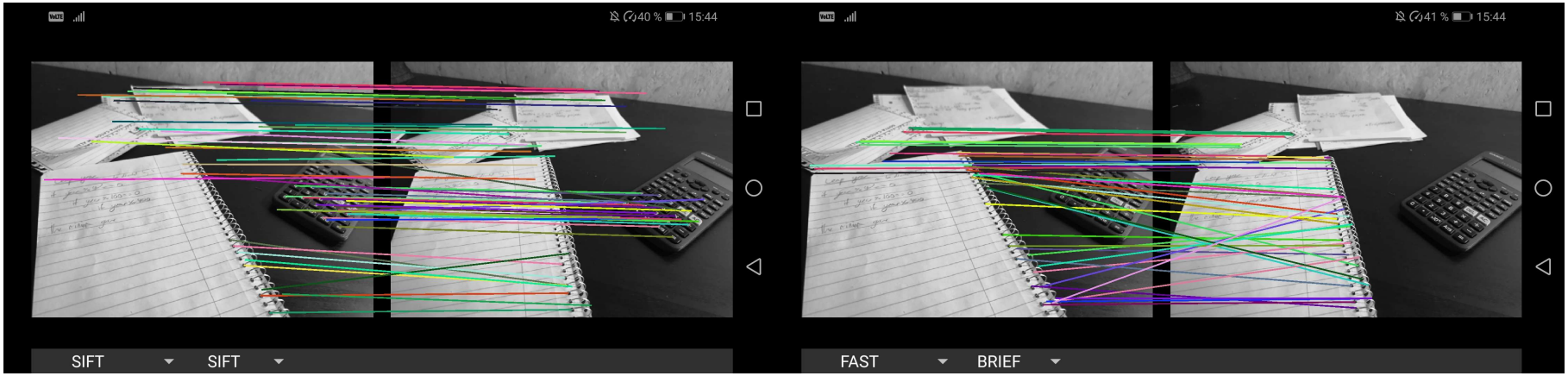



## BoofCV

- Open source real-time computer vision app
- Could specify resolution and which camera
- Also used to compare feature detectors + descriptors

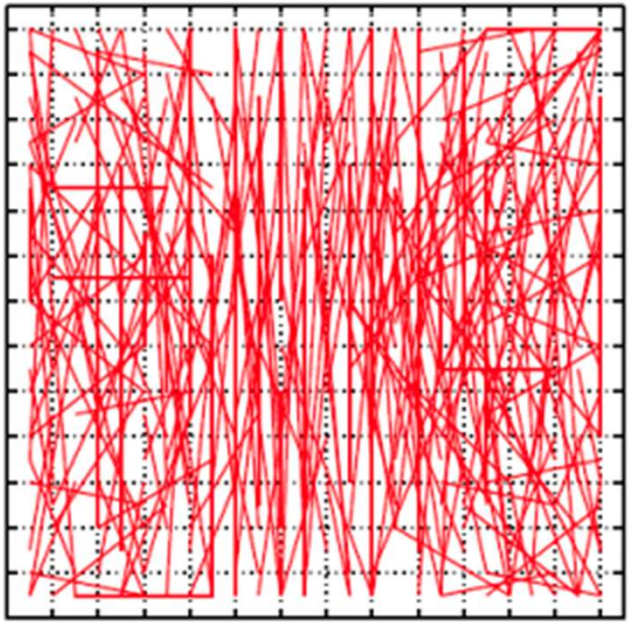
## Challenges – Phone cameras are getting advanced

- Several cameras
- AI features, Automatic calibration

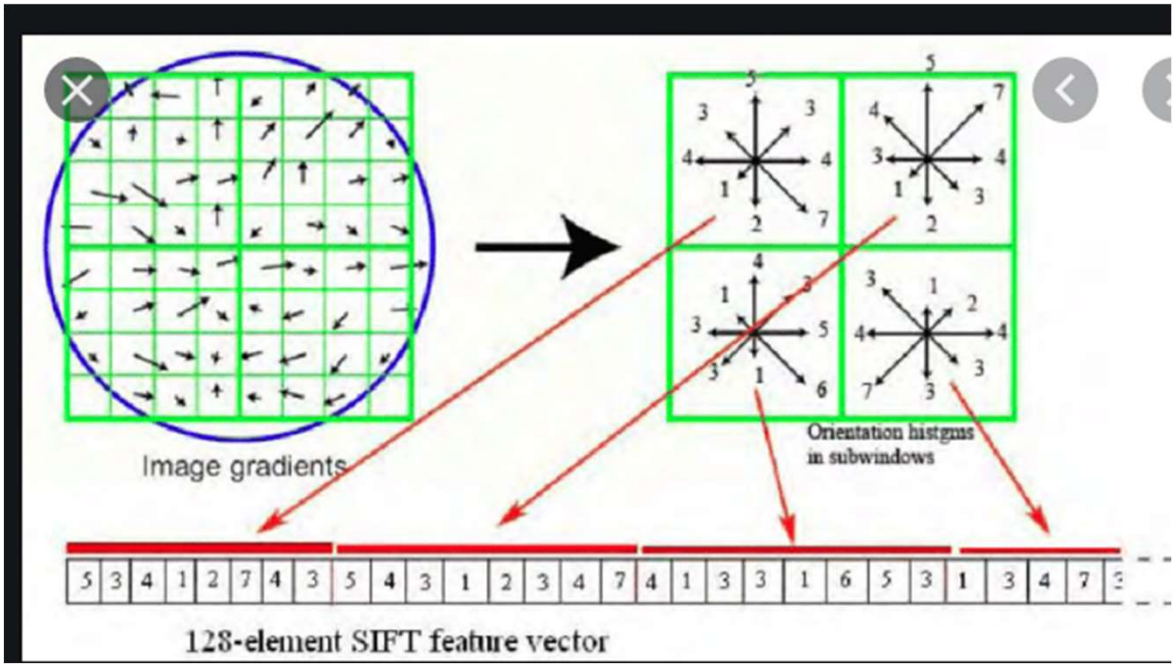


# Extra features – SIFT vs ORB

- ORB: FAST detector + BRIEF descriptor + modifications, efficient
- SIFT:  Less efficient but more accurate, patented



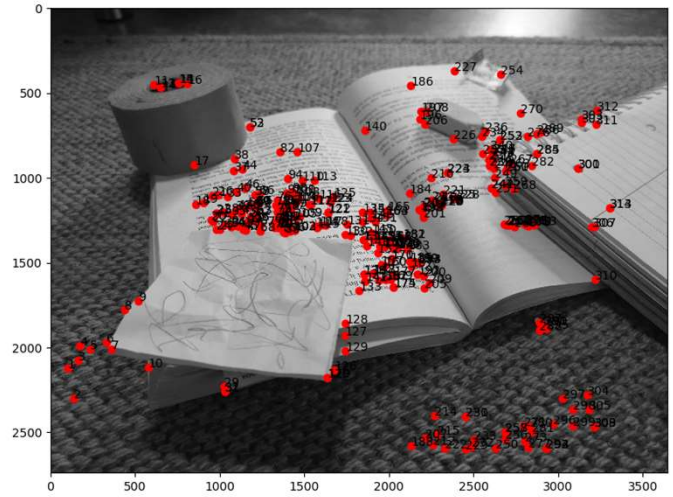
ORB sampling pairs

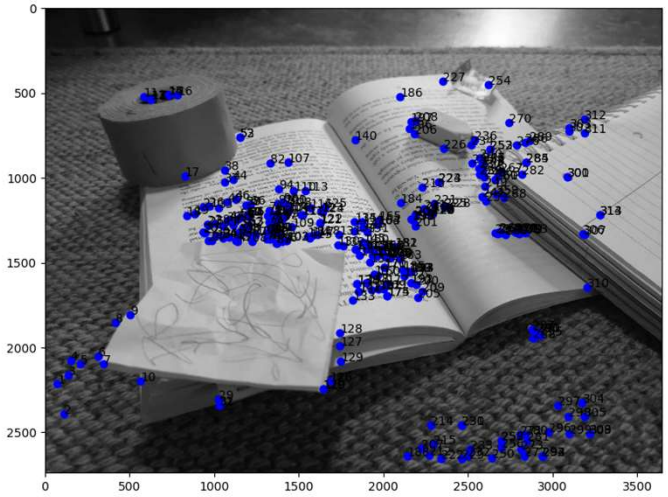


# Extract local features – 2D-2D matches

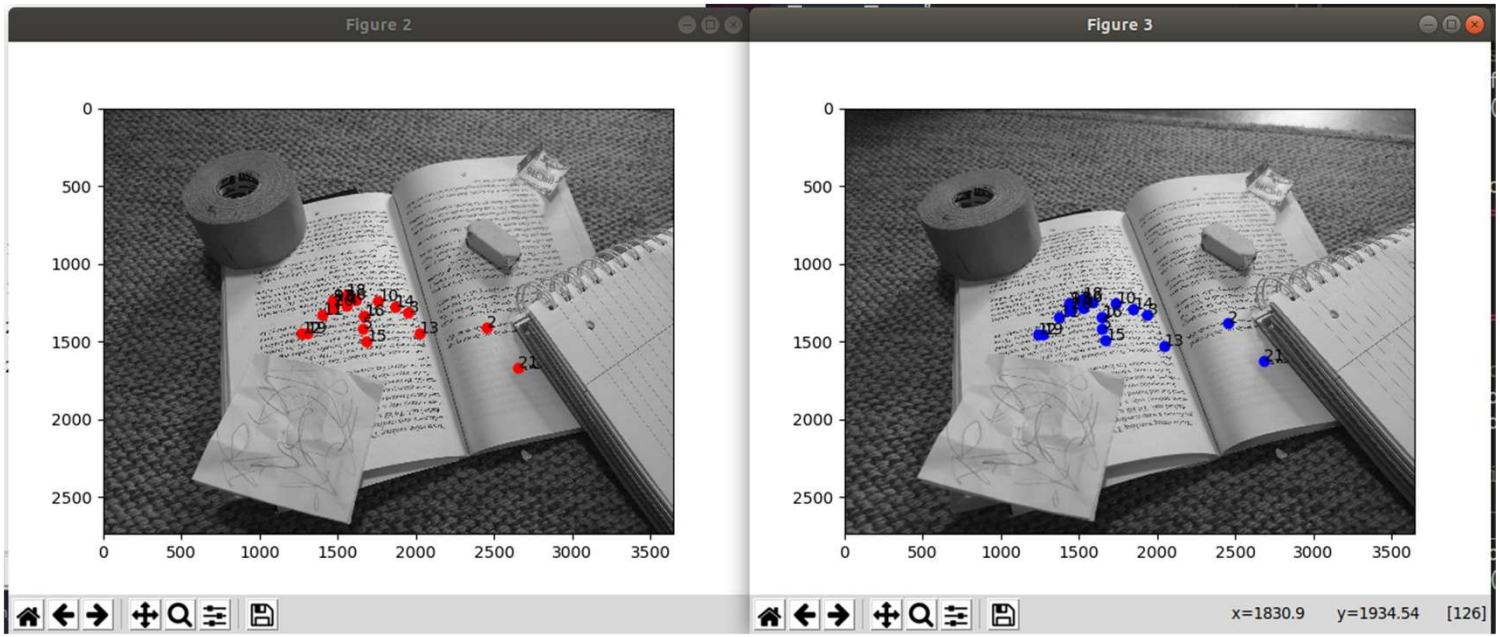
1) Detect features + compute descriptors
2) Flann based feature matcher
3) Lowes ratio test
4) Epipolar constraintment (0 = x_r' E x_l)
5) Save feature matches to .txt file

# SIFT

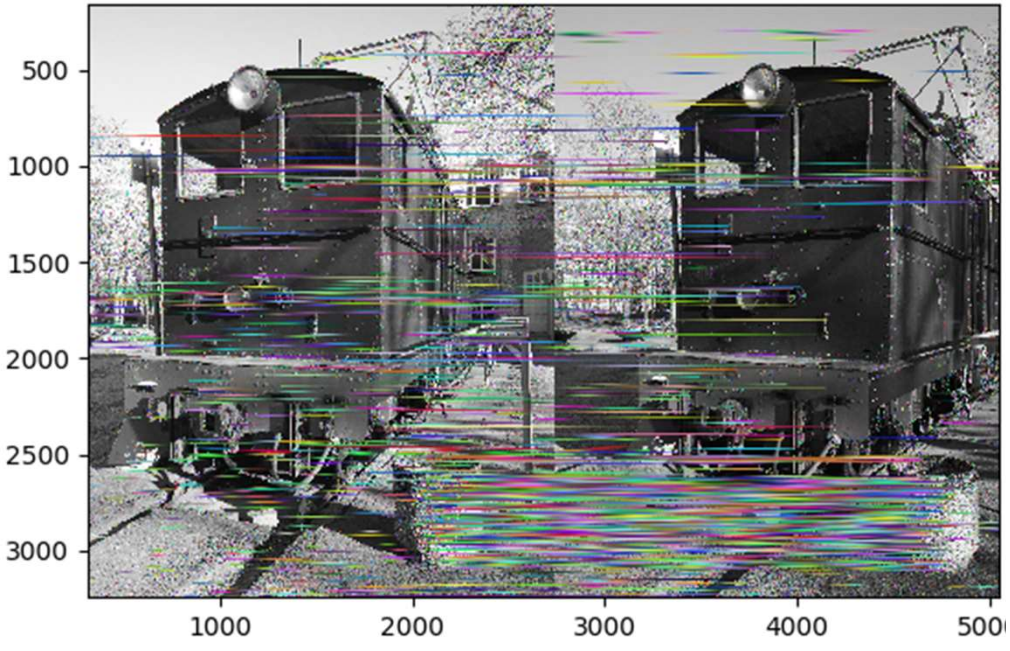




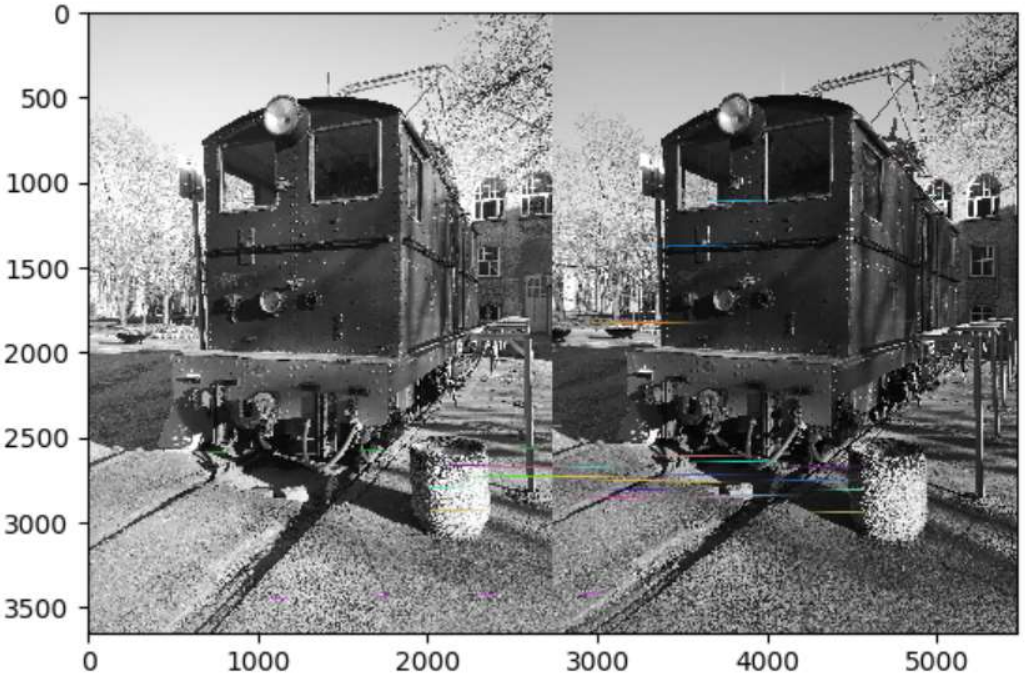
# ORB



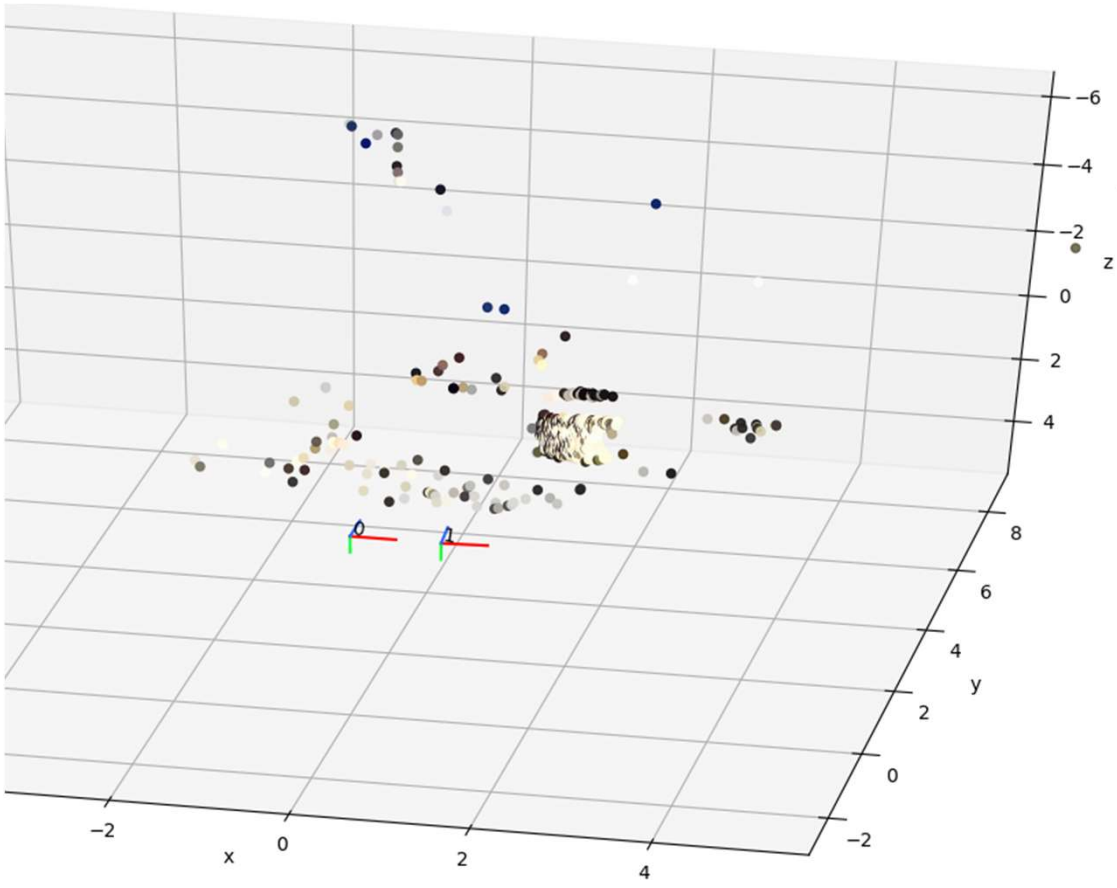
**SIFT**


500
1000
1500
2000
2500
3000
1000
2000
3000
4000

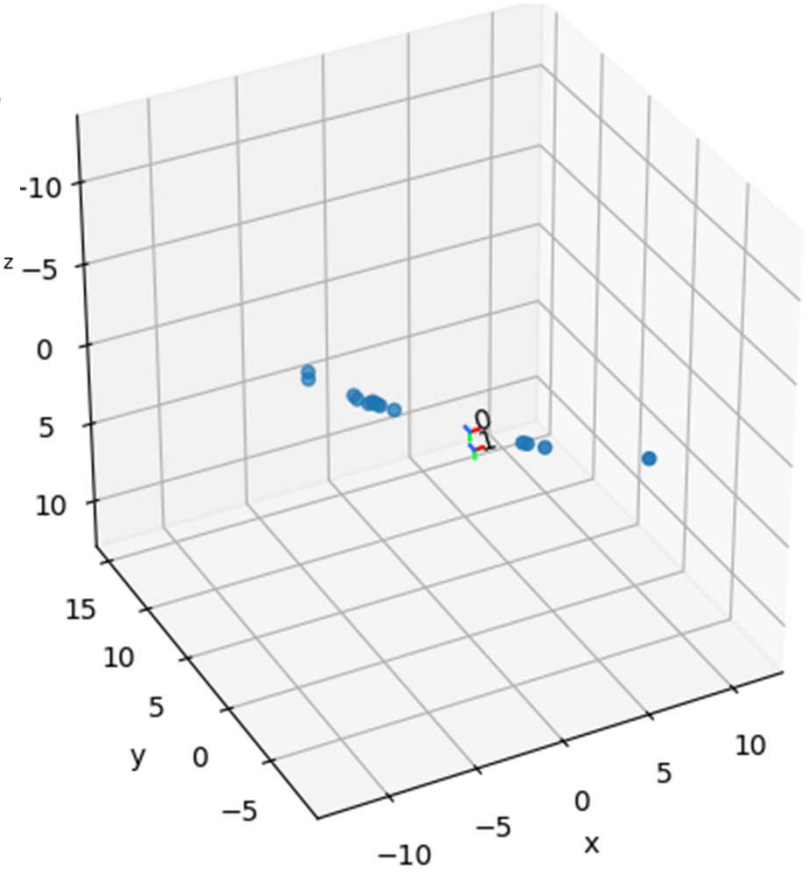
**ORB**


0
500
1000
1500
2000
2500
3000
3500
0
1000
2000
3000
4000
5000

**SIFT**




**ORB**



# Establish 2D-3D matches

- Normalized 8-point algorithm
  - Normalize point correspondances
  - 8-point algorithm (x1' F x2 = 0)
  - Find the closest fundamental matrix - Frobenius norm
- Motion from essential: *E = K2' F K1  U,S,V' = svd(E)*
- Choose solution: *(R1,T1), (R1,T2), (R2,T1), (R2,T2) -> [R | T]*
- Find projection matricies (P1,P2) - *P=K[R | T], X = inv(P)x*
- Output : pose (R,T) and pointcloud (X)

# Camera pose estimating

Solve PnP with RANSAC

- Finds an object pose from 3D-2D point correspondences.

```
(success,rotation_vector, translation_vector,_) = cv2.solvePnPRansac(X, uv, K, dist_coeffs,cv2.SOLVEPNP_UPNP)
```

$$s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & \gamma & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}.$$

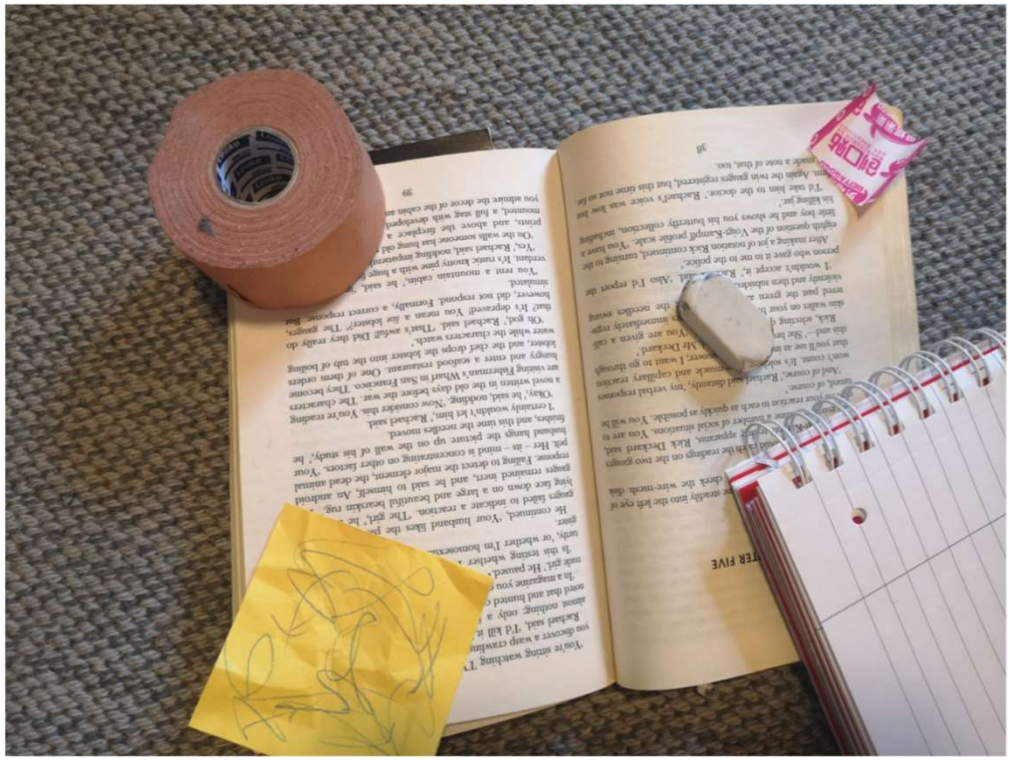
EPnP

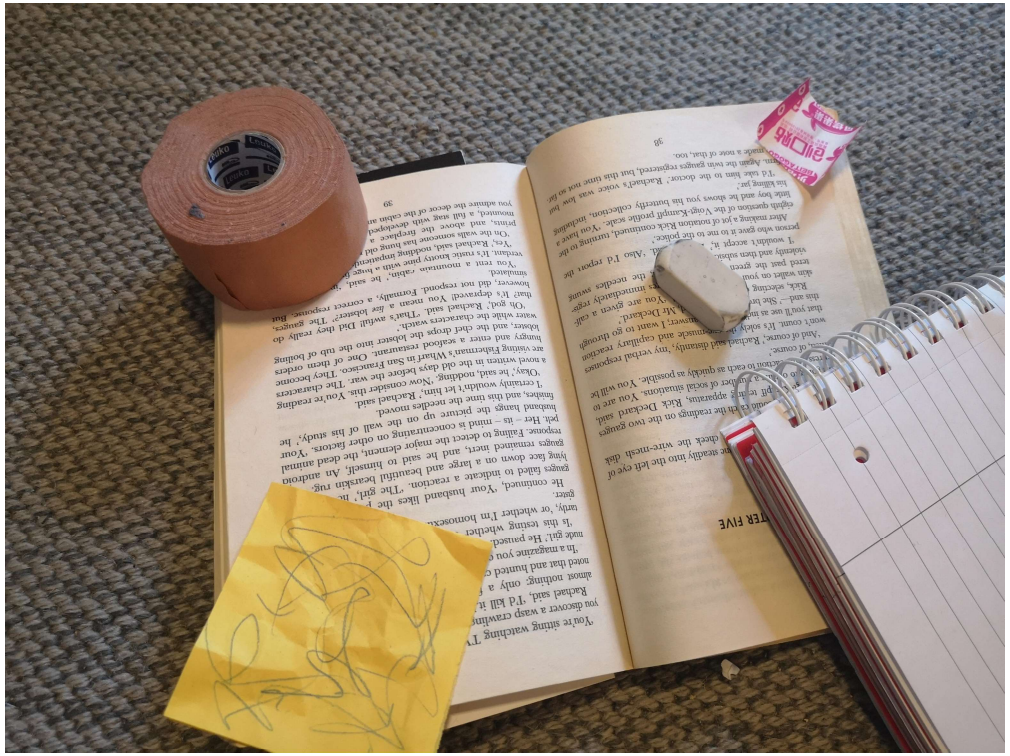
- Linear
- 3D points expressed as a weighted sum of four virtual control points.

With RANSAC

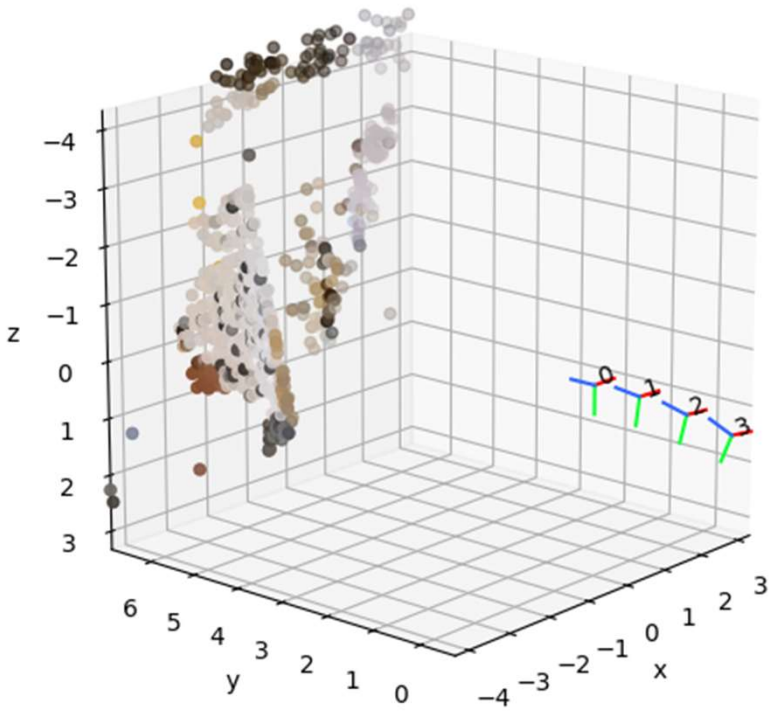
- Pose that minimizes reprojection error
- Resistant to outliers

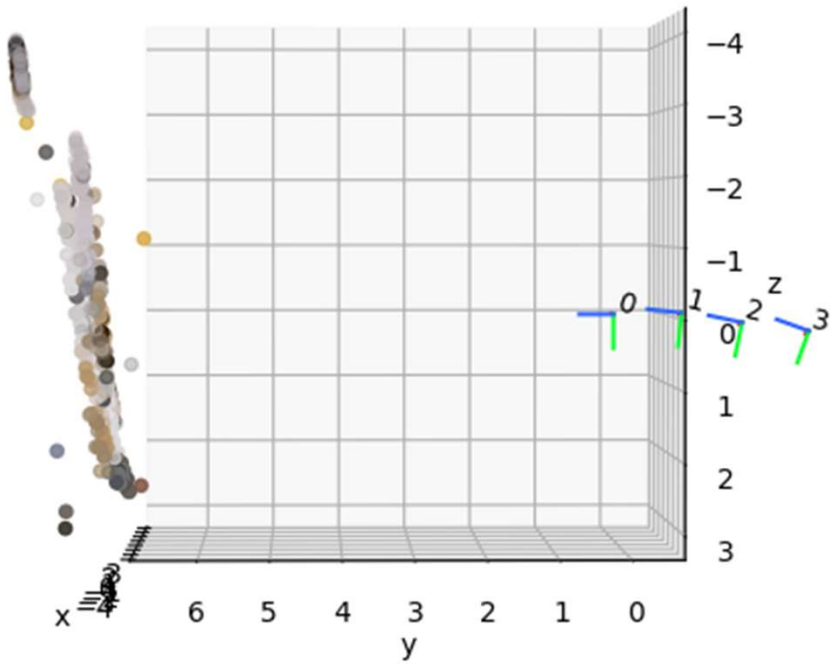
# Book



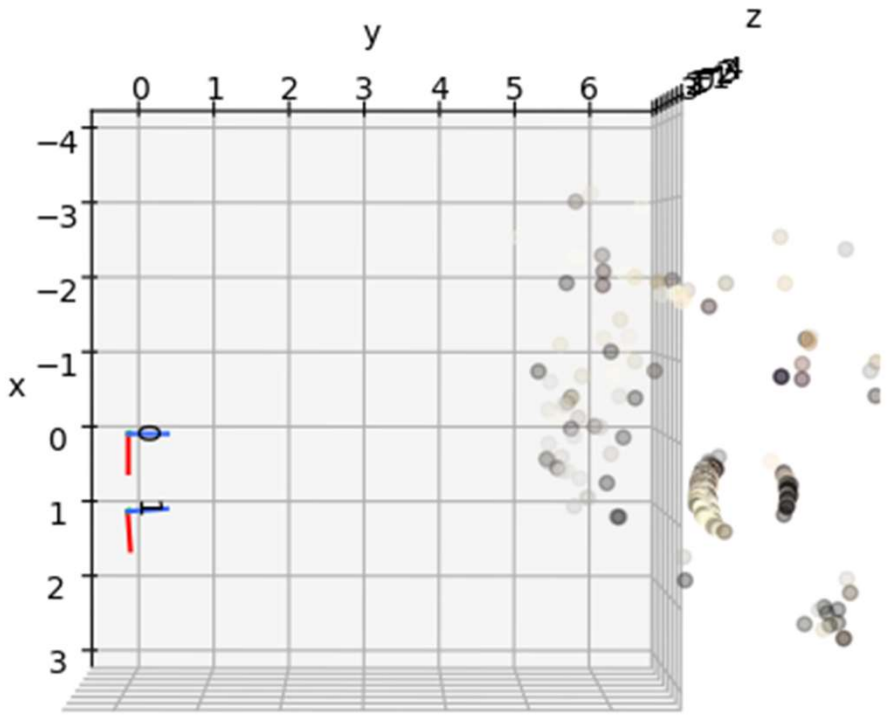


# Book


z
−4
−3
−2
−1
0
1
2
3
y
6
5
4
3
2
1
0
x
−4
−3
−2
−1
0
1
2
3
0
1
2
3


z
−4
−3
−2
−1
0
1
2
3
0
1
2
3
x
y
6
5
4
3
2
1
0

# Ohma



# Extra features

- Colorized point cloud
  - Scatterplot with color parameters
  - Easy to see if point cloud is visualized correctly
  - Using Scatter
- Global 3D point cloud
  - Transforms are confusing