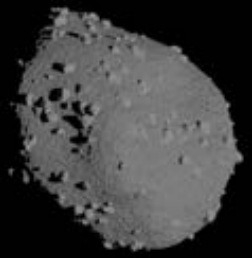
Shadow-Resilient Pose Estimation of Tumbling Bodies

MSc Thesis

Arion Zimmermann

Why tumbling bodies?

- Asteroids do not generally have a uniform inertia matrix
- **Dzhanibekov effect**: Unstable rotation around second principal axis
- Chaotic motion impossible to predict in the long term according to Euler equations



M. Brozović, L. A. Benner, J. G. McMichael, *et al.*, "Goldstone and arecibo radar observations of (99942) apophis in 2012–2013," *Icarus*, vol. 300, pp. 115–128, 2018

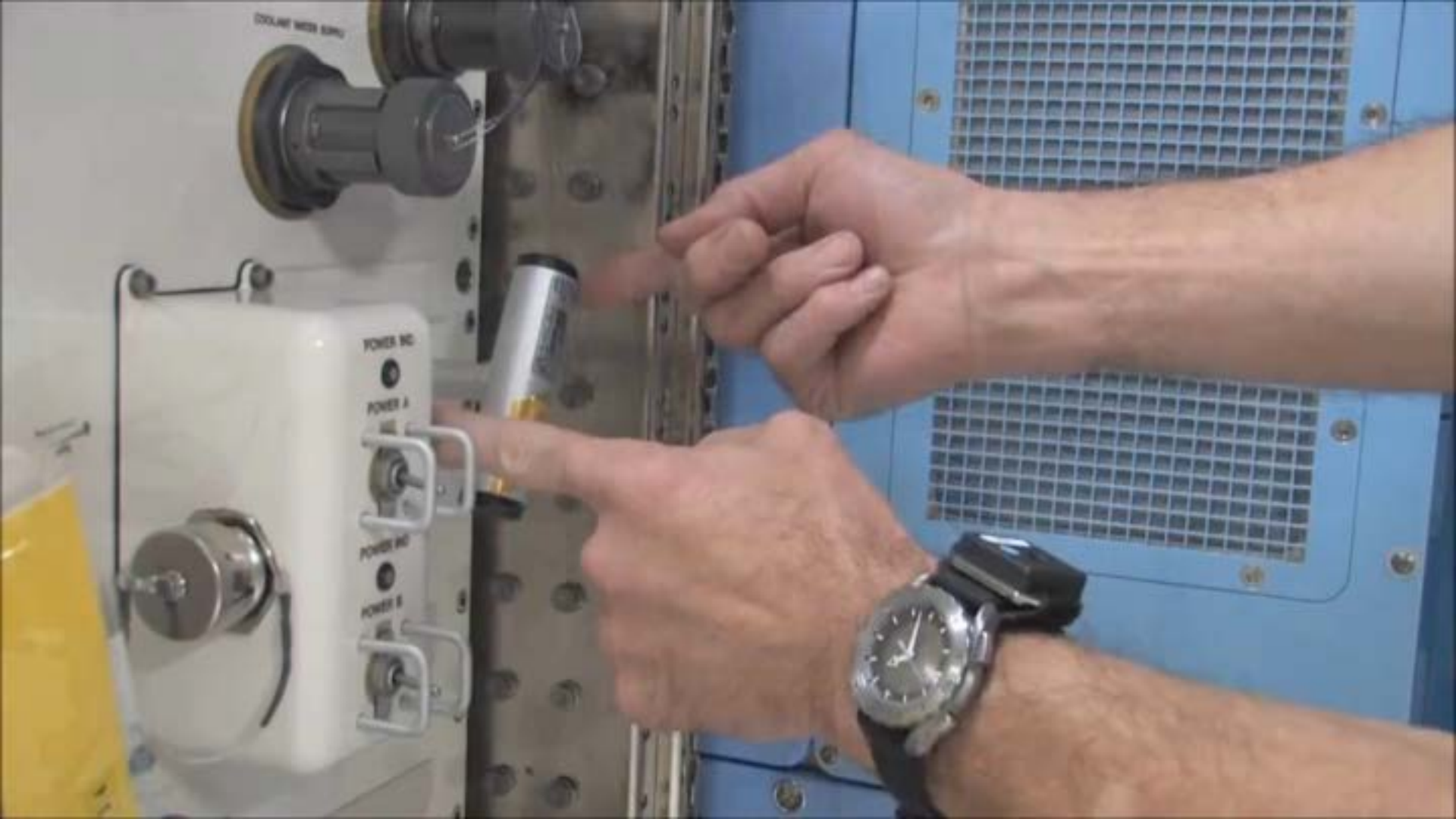
Why tumbling bodies?

- Asteroids do not generally have a uniform inertia matrix
- Dzhanibekov effect: Unstable rotation around second principal axis
- Chaotic motion impossible to predict in the long term according to Euler equations

$$I_x\dot{\omega}_x + (I_z - I_y)\omega_y\omega_z = 0$$
$$I_y\dot{\omega}_y + (I_x - I_z)\omega_z\omega_x = 0$$
$$I_z\dot{\omega}_z + (I_y - I_x)\omega_x\omega_y = 0$$

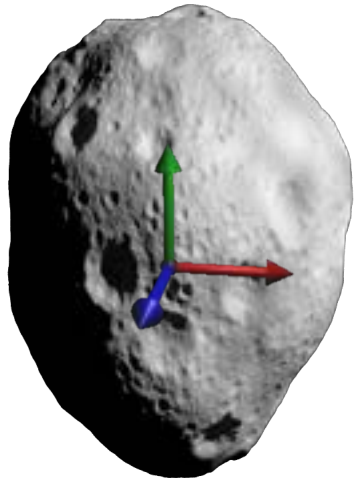
$$I_x < I_y < I_z, \quad \omega_y \gg \omega_x, \omega_z$$

$$\ddot{\omega}_x = \lambda\omega_x, \quad \ddot{\omega}_z = \lambda\omega_z, \quad \lambda > 0$$


POWER
POWER

Why pose estimation?

- Future space missions may require orbit synchronization to a tumbling asteroid
- Standard model-based filters are not sufficient because of the chaotic process
- Instantaneous pose estimation is therefore required



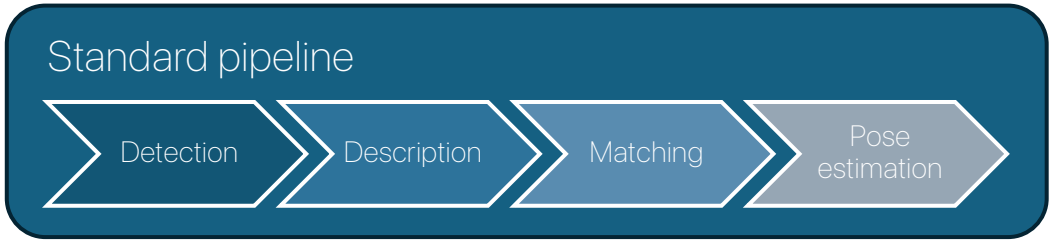
EPFL Caltech

Challenges

- The lack of atmosphere make shadows extremely dark
- Asteroids are textureless and do not diffract light
- Rigid bodies can tumble very quickly

Standard approach

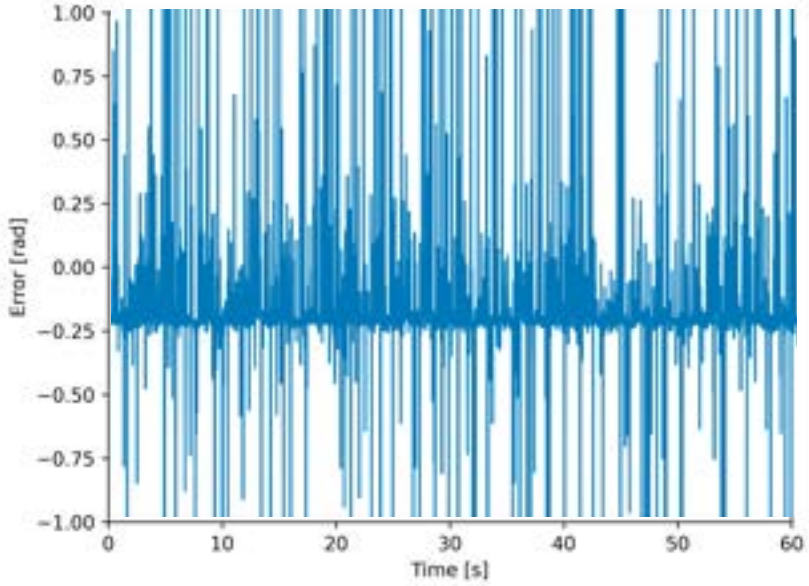
- Detect a sparse set of salient keypoints (contours and edges)
- Describe the keypoints by analyzing their neighborhood and assign a high-dimensional descriptor
- Find correspondences between feature descriptors in successive frames
- Estimate the pose by solving the 6DoF projection equation



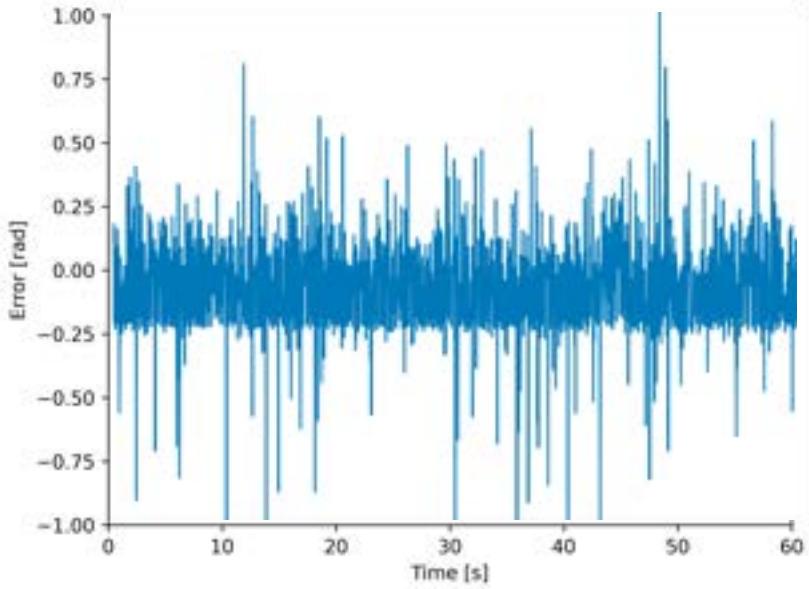
EPFL

Caltech

Why not SIFT?

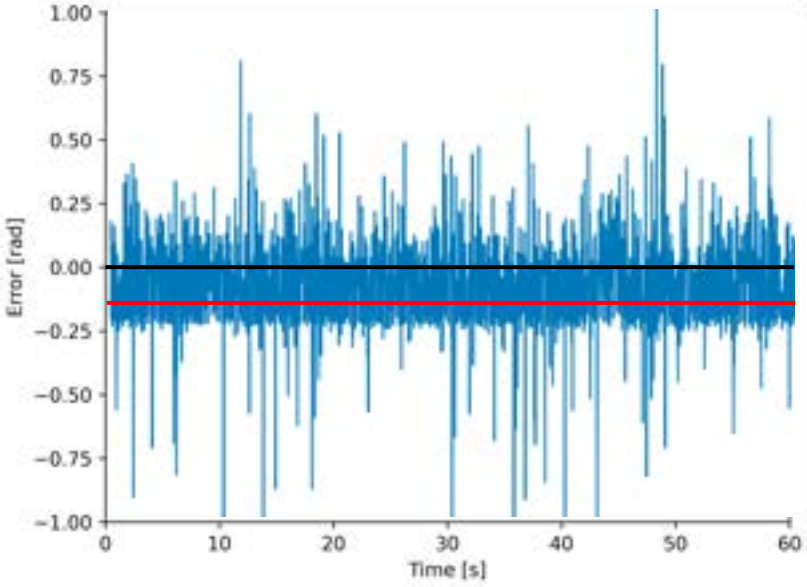

1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
Error [rad]
0
10
20
30
40
50
60
Time [s]

Why not Superpoint?

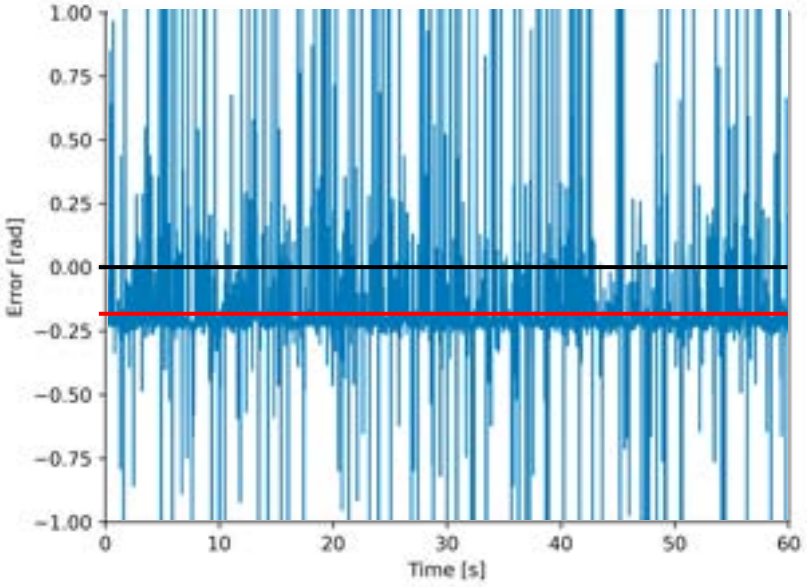


Estimation bias

Superpoint



SIFT



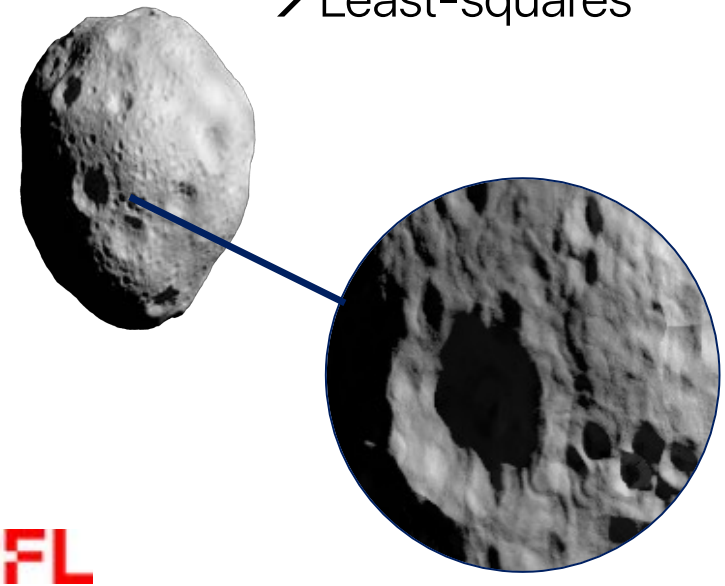
EPFL

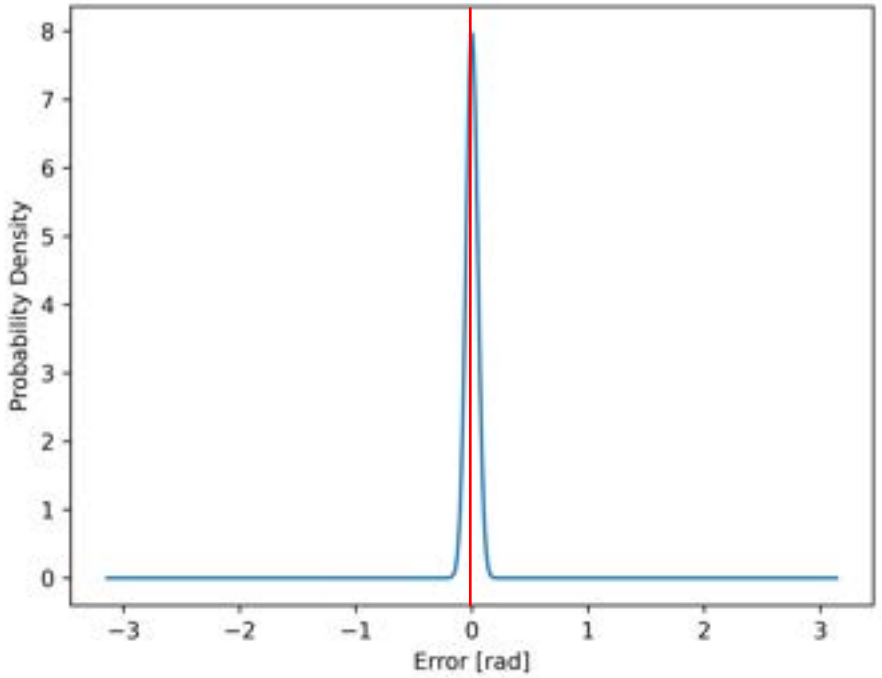
Caltech

Effect of occlusion on pose estimation

- Gaussian error?

→ Least-squares





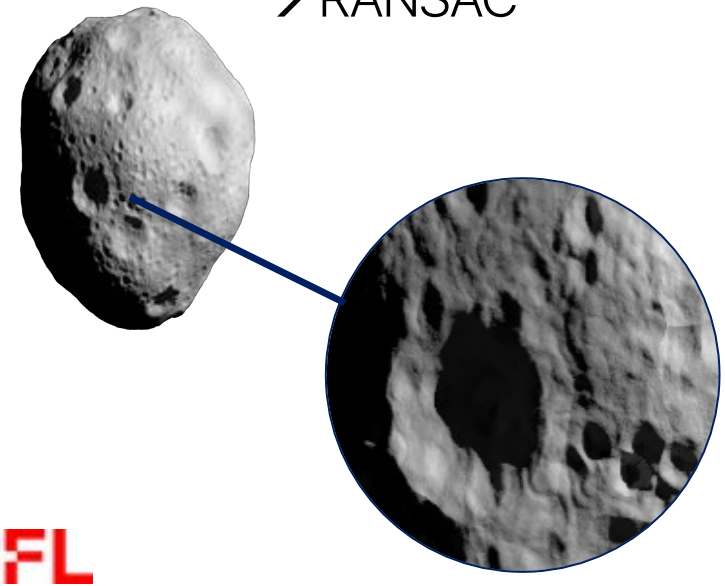
EPFL

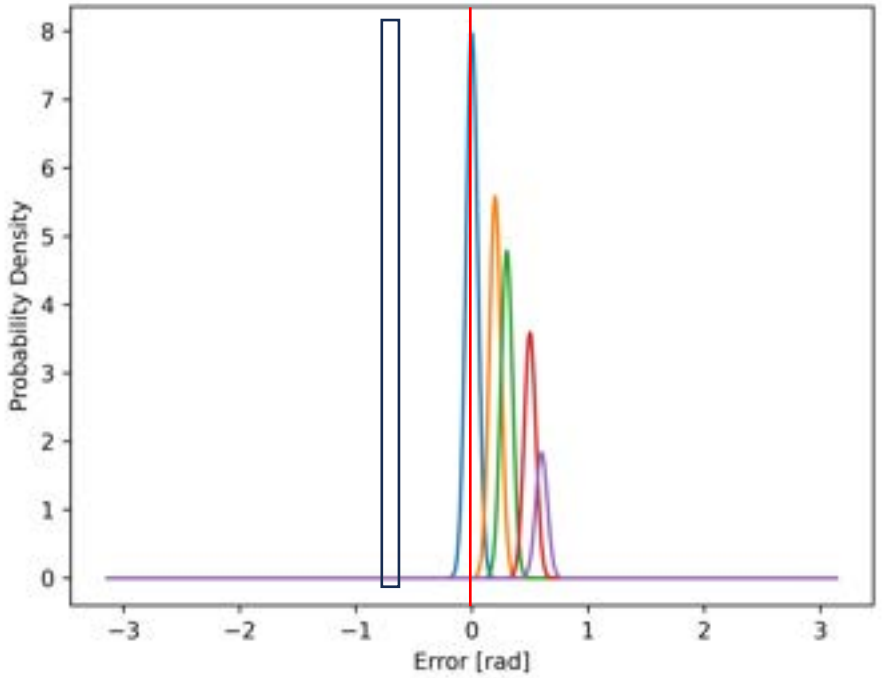
Caltech

Effect of occlusion on pose estimation

- Multimodal Gaussian error?

→ RANSAC





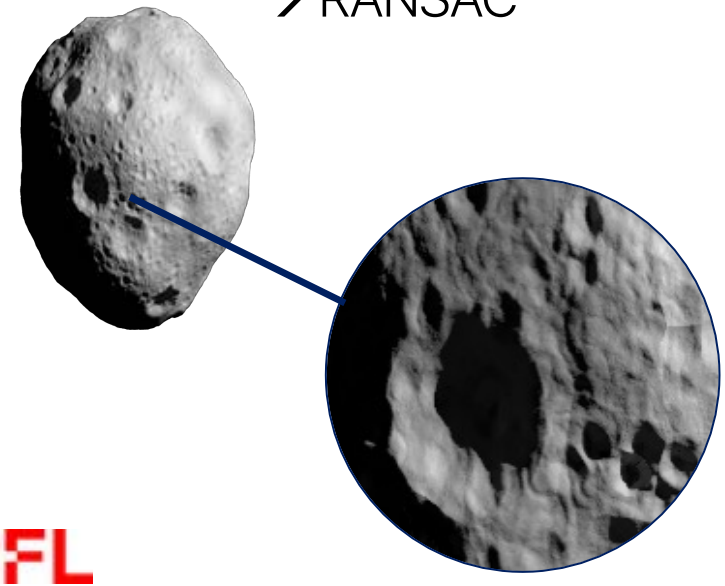
EPFL

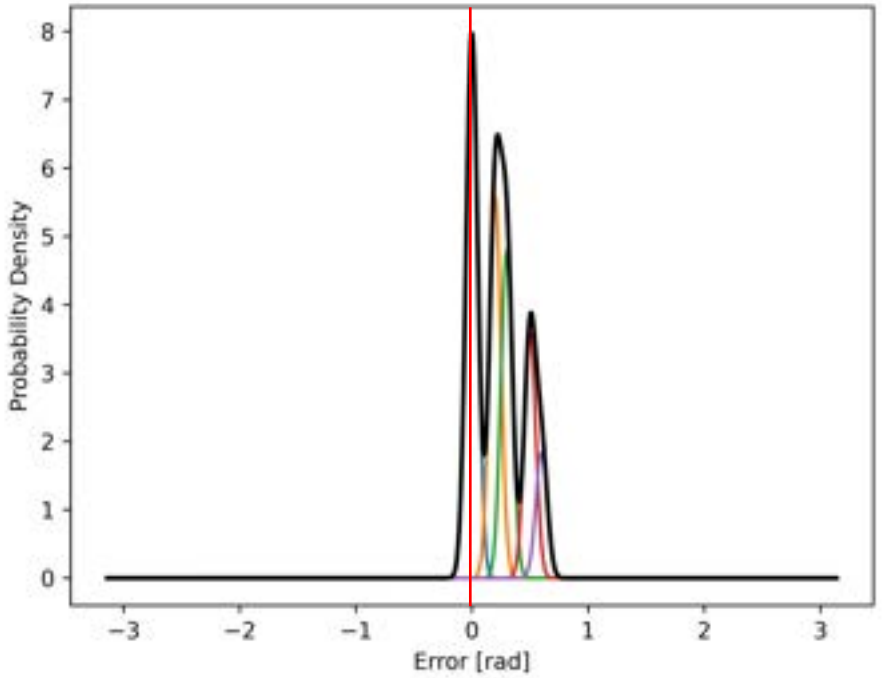
Caltech

Effect of occlusion on pose estimation

- Multimodal Gaussian error?

→ RANSAC





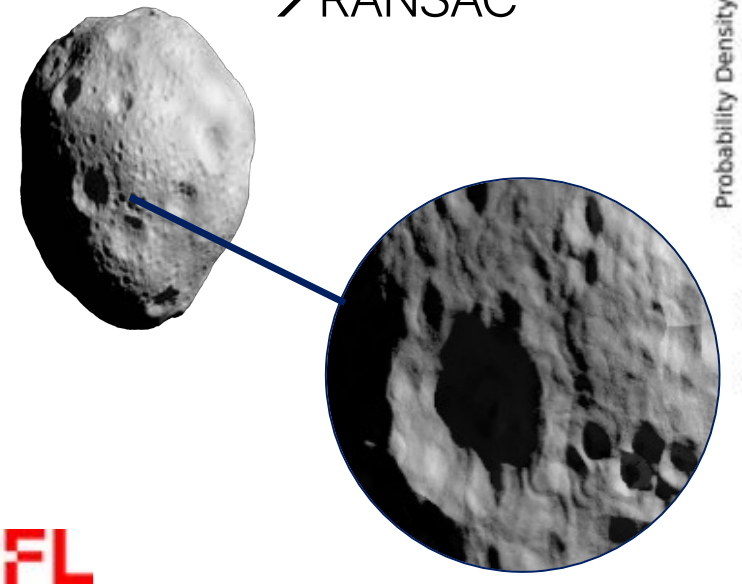
EPFL

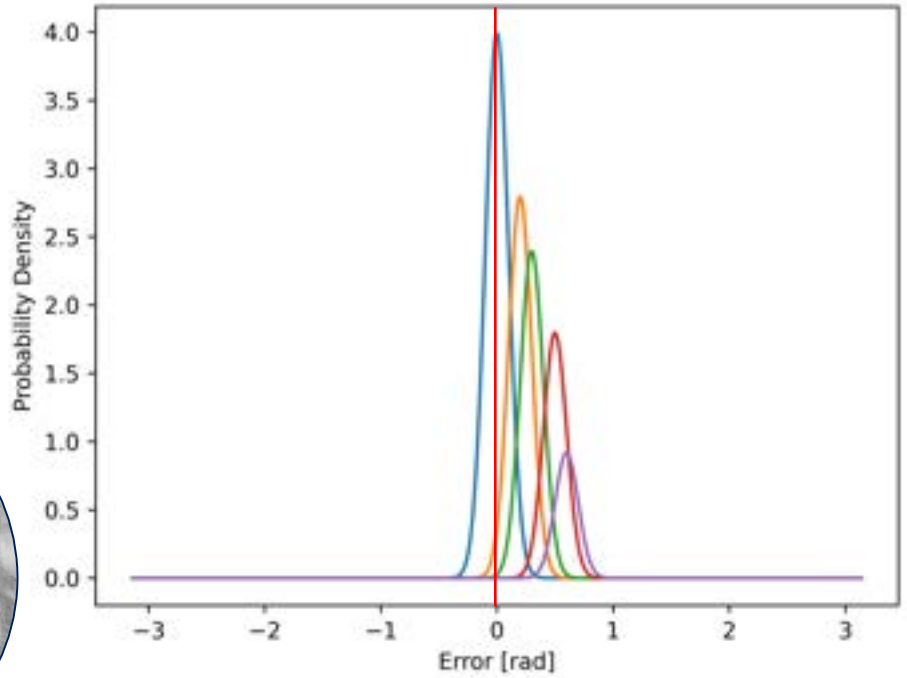
Caltech

Effect of occlusion on pose estimation

- Multimodal Gaussian error?

→ RANSAC

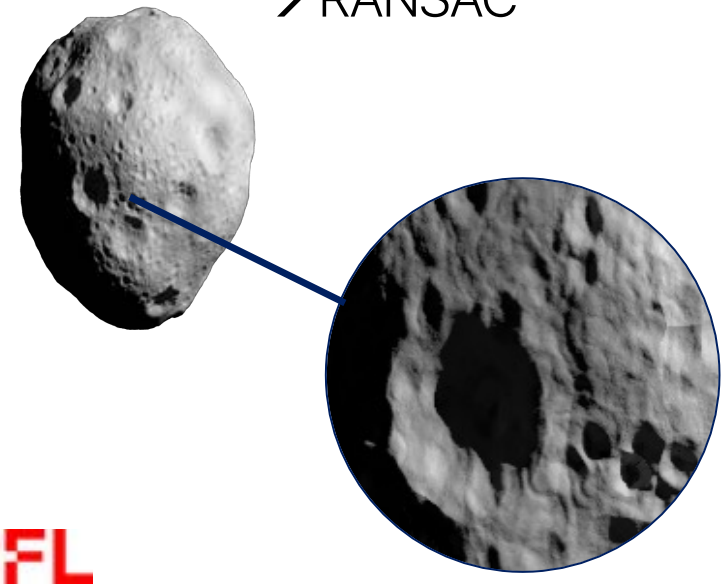


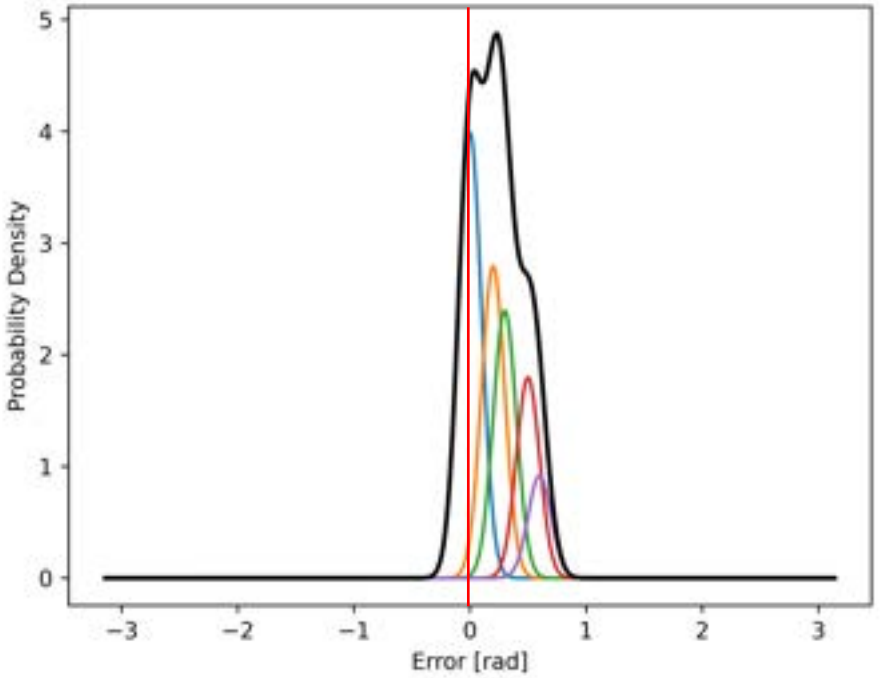


Effect of occlusion on pose estimation

- Multimodal Gaussian error?

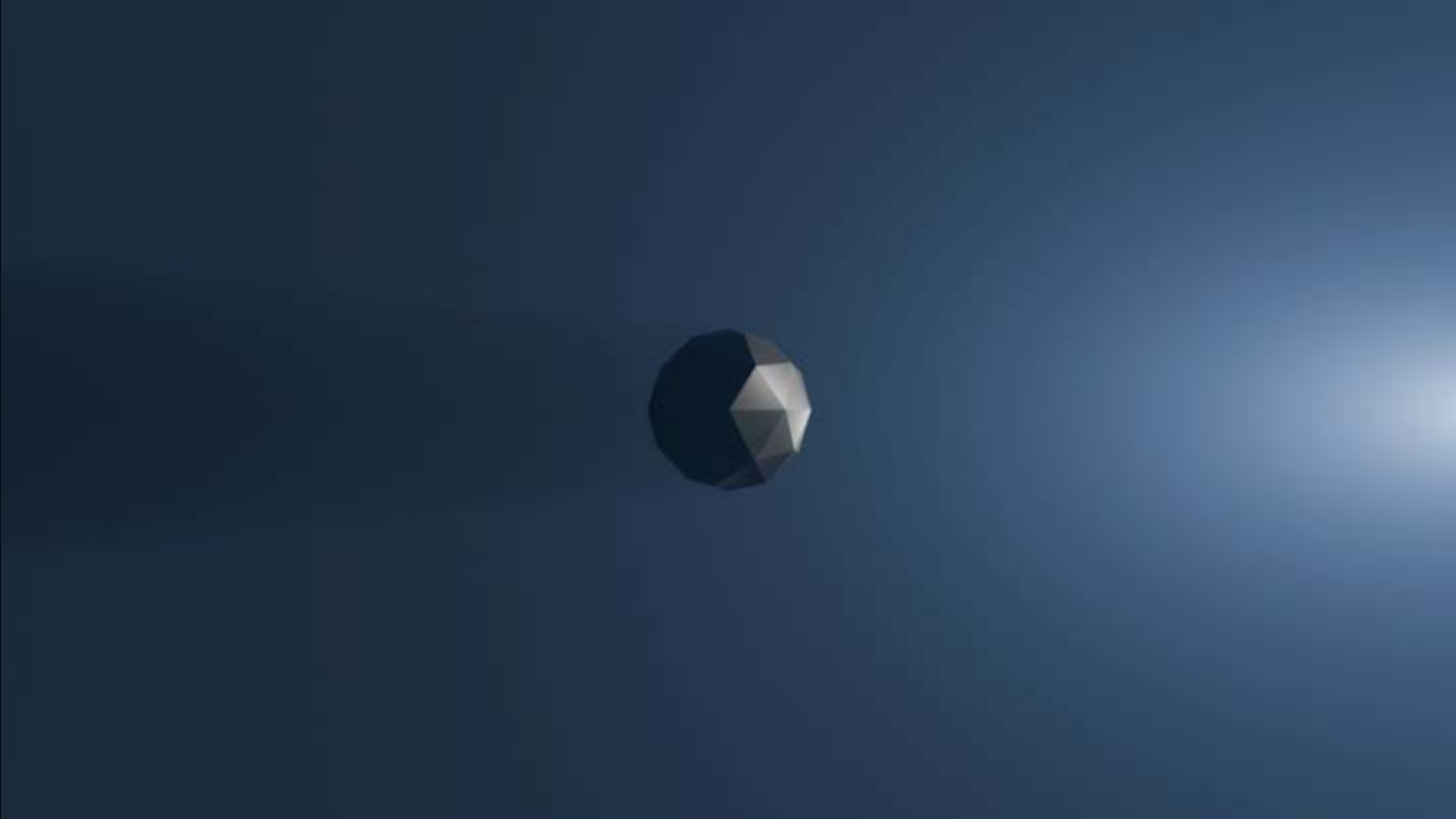
→ RANSAC





EPFL

Caltech

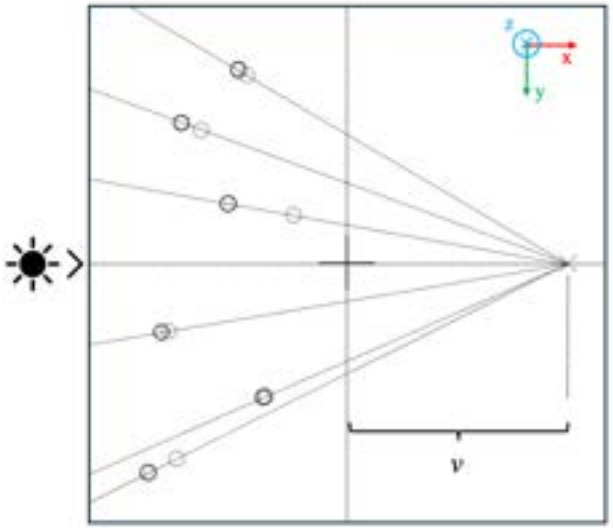


$$v = \begin{bmatrix} f_x \tan(\phi) \\ 0 \end{bmatrix}$$

Vanishing Point

COFFEE detector

- The sun-phase angle is known thanks to the onboard sun-tracker
- All shadows are cast towards a vanishing point
- Scan through the rays of the image, find edges and encode features as keypoints, shadow size as descriptors



EPFL

Caltech

COFFEE detector

- The sun-phase angle is known thanks to the onboard sun-tracker
- All shadows are cast towards a vanishing point
- Scan through the rays of the image, find edges and encode features as keypoints, shadow size as descriptors

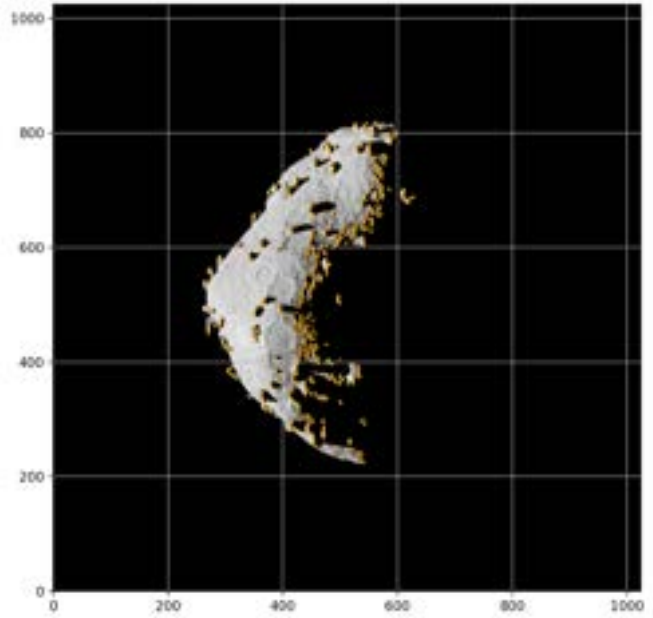
Algorithm 1 COFFEE detector

```
Input: input dense image, φ sun phase angle
Output: output, sparse image
rectified ← Rectify(input, φ)
thresholded ← Thresholding(rectified)
filtered ← EdgeFilter(thresholded)
compressed, crows, cols ← CSR(filtered)
for each row i do
    for each column j do
        start_idx ← crows[row_idx]
        end_idx ← crows[row_idx + 1]
        index ← start_idx + col_idx
        if compressed [index] < 0 then
            values[index] ← arctan(cols[index + 1] - cols[index])
        end if
    end for
end for
return sparse_representation
```

EPFL Caltech

COFFEE detector

- The sun-phase angle is known thanks to the onboard sun-tracker
- All shadows are cast towards a vanishing point
- Scan through the rays of the image, find edges and encode features as keypoints, shadow size as descriptors

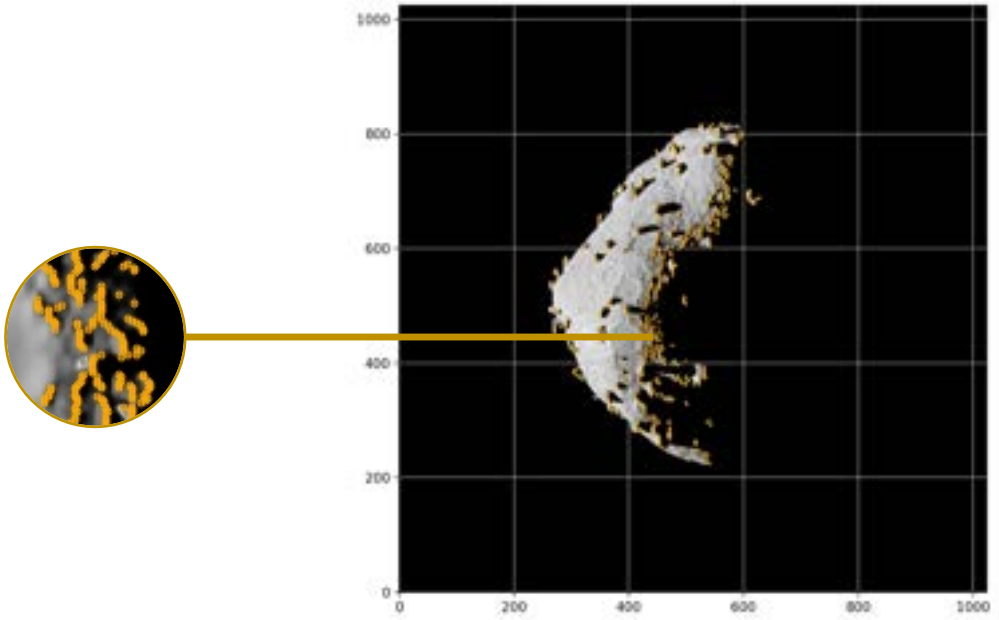


EPFL

Caltech

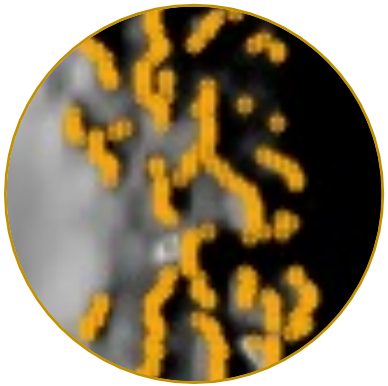
COFFEE detector

- Shadow-size encoding as feature descriptor is not enough
- Exploit the remaining structure of the keypoints through a neural network



COFFEE descriptor

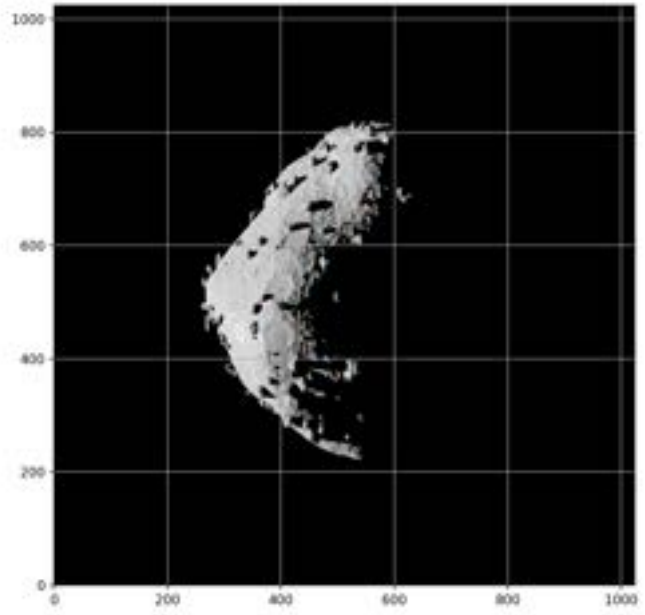
- Use Sparse Submanifold CNNs to extract additional structural information
- Exploit the remaining structure of the keypoints through a neural network
- Inference complexity is reduced from $O(mns^2)$ to $O(mns)$ for each CNN layer



B. Graham, M. Engelcke, and L. v. d. Maaten, “3d semantic segmentation with submanifold sparse convolutional networks,” *CVPR*, 2018, pp. 9224–9232

COFFEE descriptor

- Several NN-architectures were tested (VggNet, U-Net, Inception)
- ResNet-Bottleneck layers happened to be the best candidates
- FP256 feature descriptors are assigned after 17 SCNN layers



K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” CVPR 2016, pp. 770–778.

Feature matching

- No large-scale contextual information is added during the description
- Add keypoint location encoding and apply attention mechanism
- Extremely efficient off-the-shelf architecture LightGlue

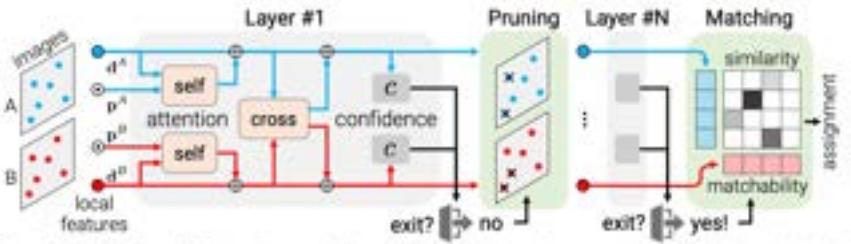
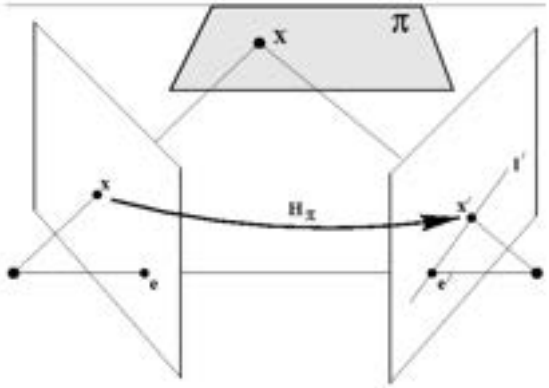


Figure 3. **The LightGlue architecture.** Given a pair of input local features (d, p), each layer augments the visual descriptors (•,•) with context based on self- and cross-attention units with positional encoding ⊙. A confidence classifier c helps decide whether to stop the inference. If few points are confident, the inference proceeds to the next layer but we prune points that are confidently unmatchable. Once a confident state if reached, LightGlue predicts an assignment between points based on their pairwise similarity and unary matchability.

P. Lindenberger, P.-E. Sarlin, and M. Pollefeys, "Lightglue: Local feature matching at light speed," ICCV 2023, pp. 17581–17592.

Pose estimation

- Using the 5-point algorithm to estimate the essential matrix
- Find an outlier-free set with RANSAC
- Extract the pose from the essential matrix

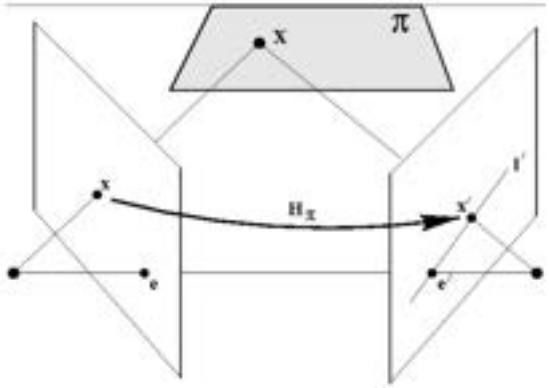


$$x'^{\top} E x = 0$$
$$det(E) = 0$$
$$2EE^{\top}E - tr(EE^{\top})E = 0$$

R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Second. Cambridge University Press

Pose estimation

- Using the 5-point algorithm to estimate the essential matrix
- Find an outlier-free set with RANSAC
- Extract the pose from the essential matrix



$$E = t_\times R \quad \text{where} \quad t_\times = \begin{pmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{pmatrix}$$

R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Second. Cambridge University Press

Pipeline summary

- Shadow rays encoding
- Sparse ResNet CNNs
- LightGlue matcher
- Pose with 5-points algorithm and RANSAC

Dataset generation

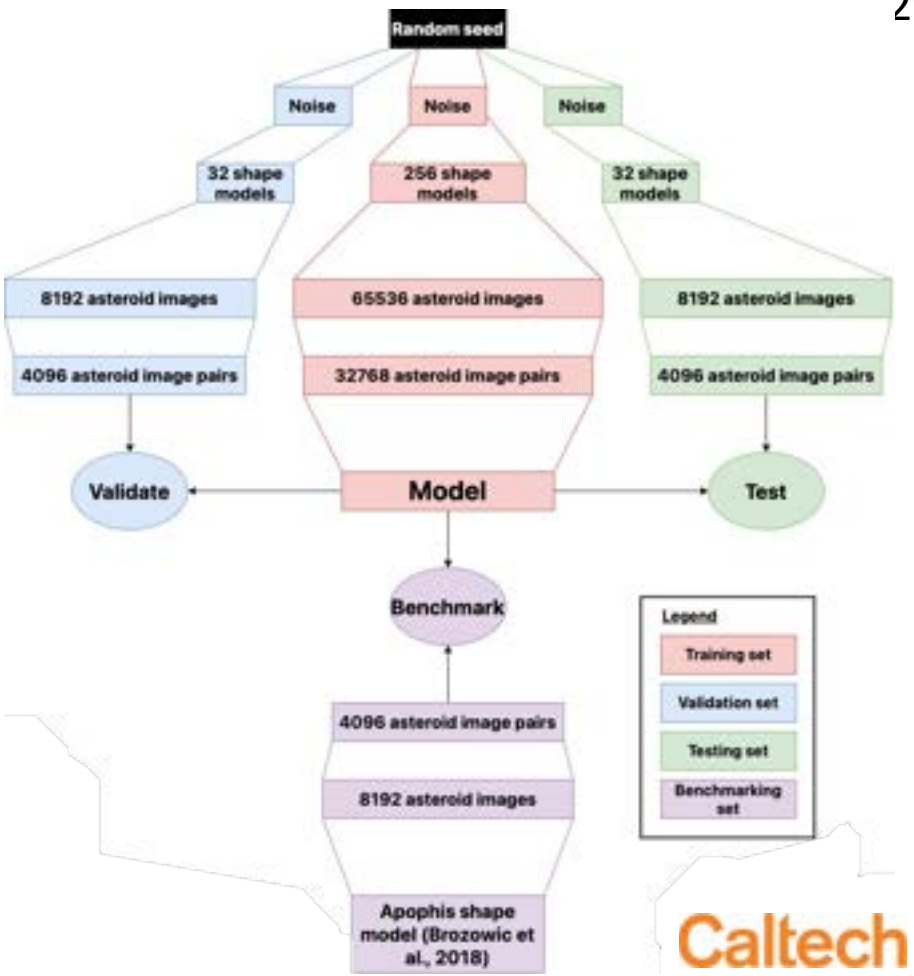
- Very few missions have flown and maintained a stable orbit around an asteroid
- Real dataset would not be representative of the space of possible asteroids
- Synthesis is necessary

Mission	Year	Camera resolution
OSIRIS-Rex	2016	1024x1024
Hayabusa2	2014	1024x1024
Rosetta	2004	1024x1024
Hayabusa	2003	1024x1000
NEAR Shoemaker	1996	537x244

Table 1: Asteroid hovering missions

Dataset generation

- **Four datasets**: train, validate, test, benchmark
- Train, validate, test are generated procedurally from noise
- Benchmark is an "enhanced model" of the Apophis asteroid

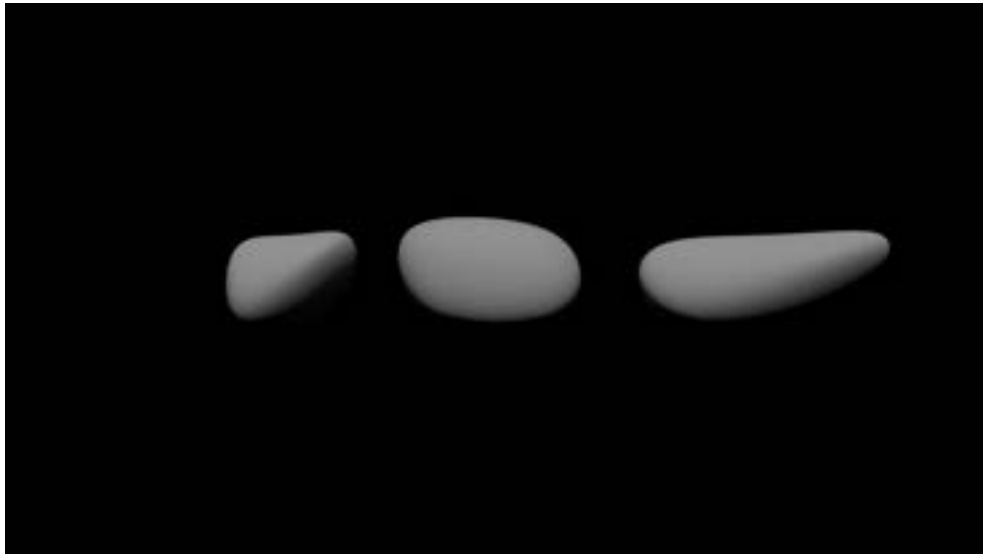


EPFL

Caltech

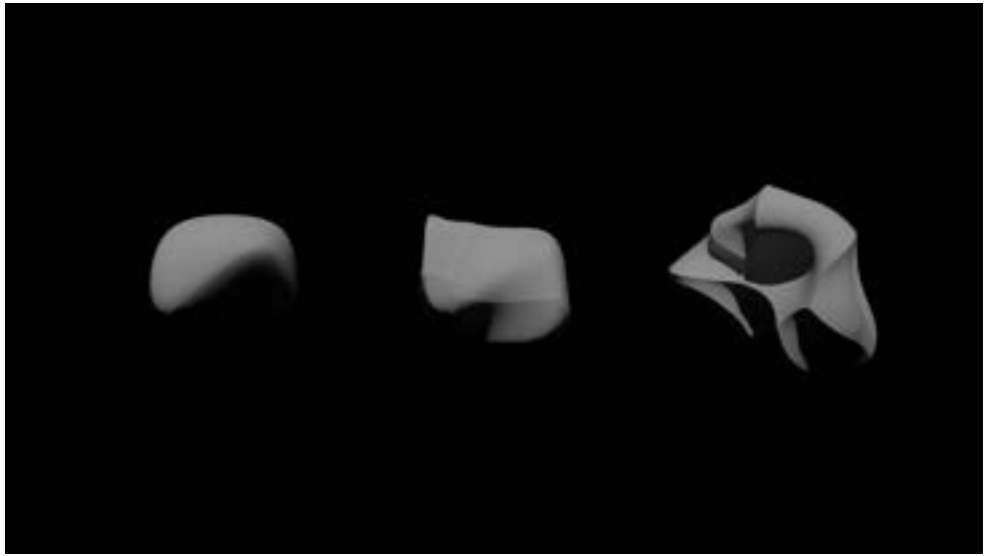
Dataset generation

- Custom size



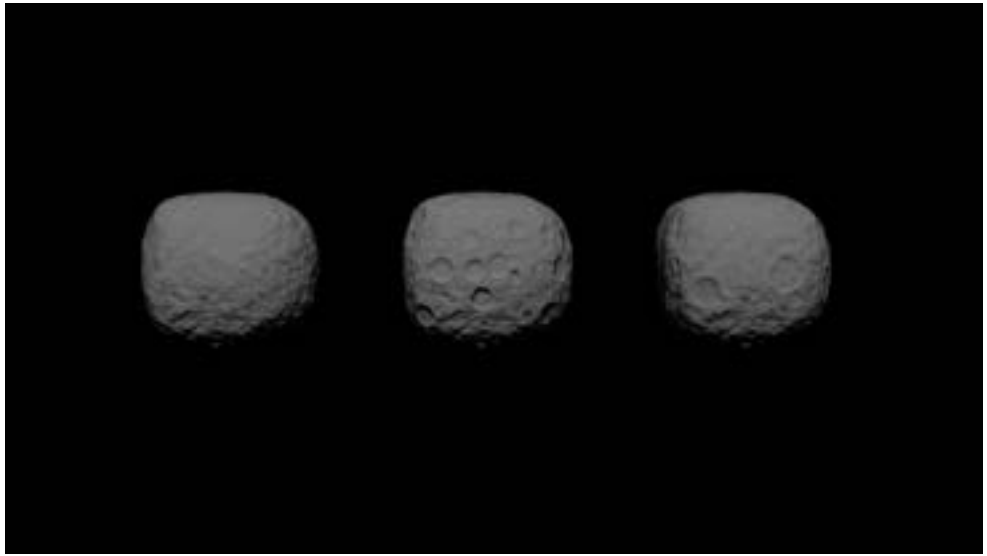
Dataset generation

- Custom size
- Custom deformation



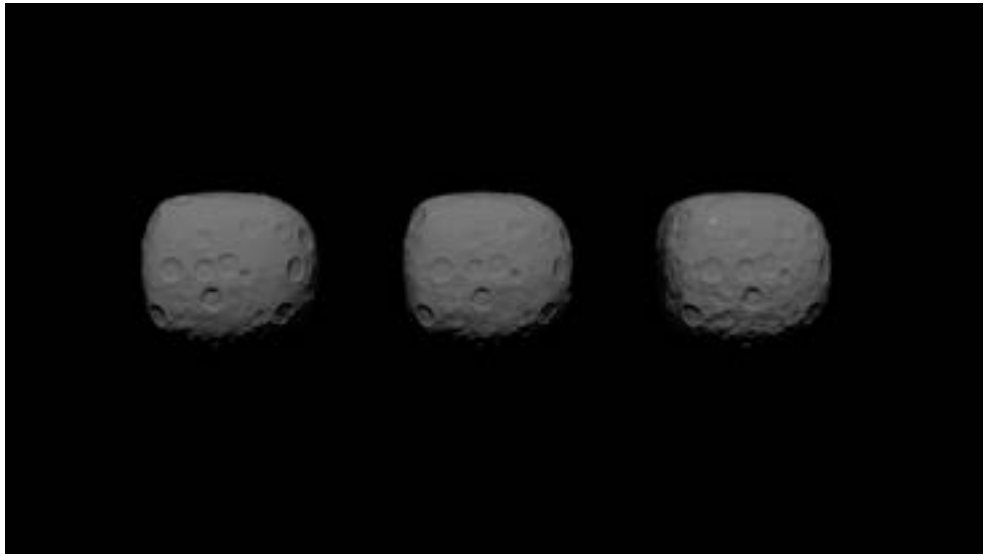
Dataset generation

- Custom size
- Custom deformation
- Custom crater size/distribution



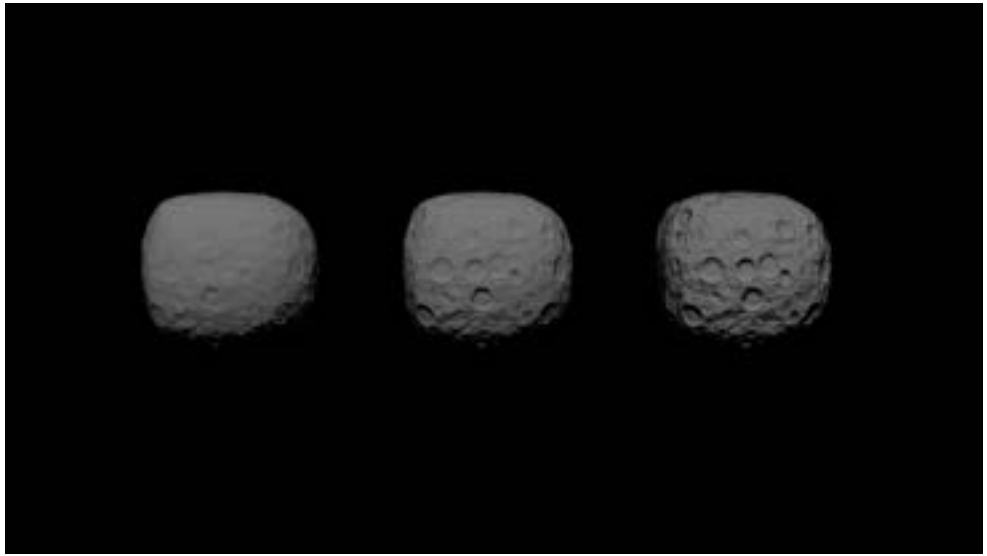
Dataset generation

- Custom size
- Custom deformation
- Custom crater size/distribution
- Custom roughness



Dataset generation

- Custom size
- Custom deformation
- Custom crater size/distribution
- Custom roughness
- Custom depth



Dataset generation

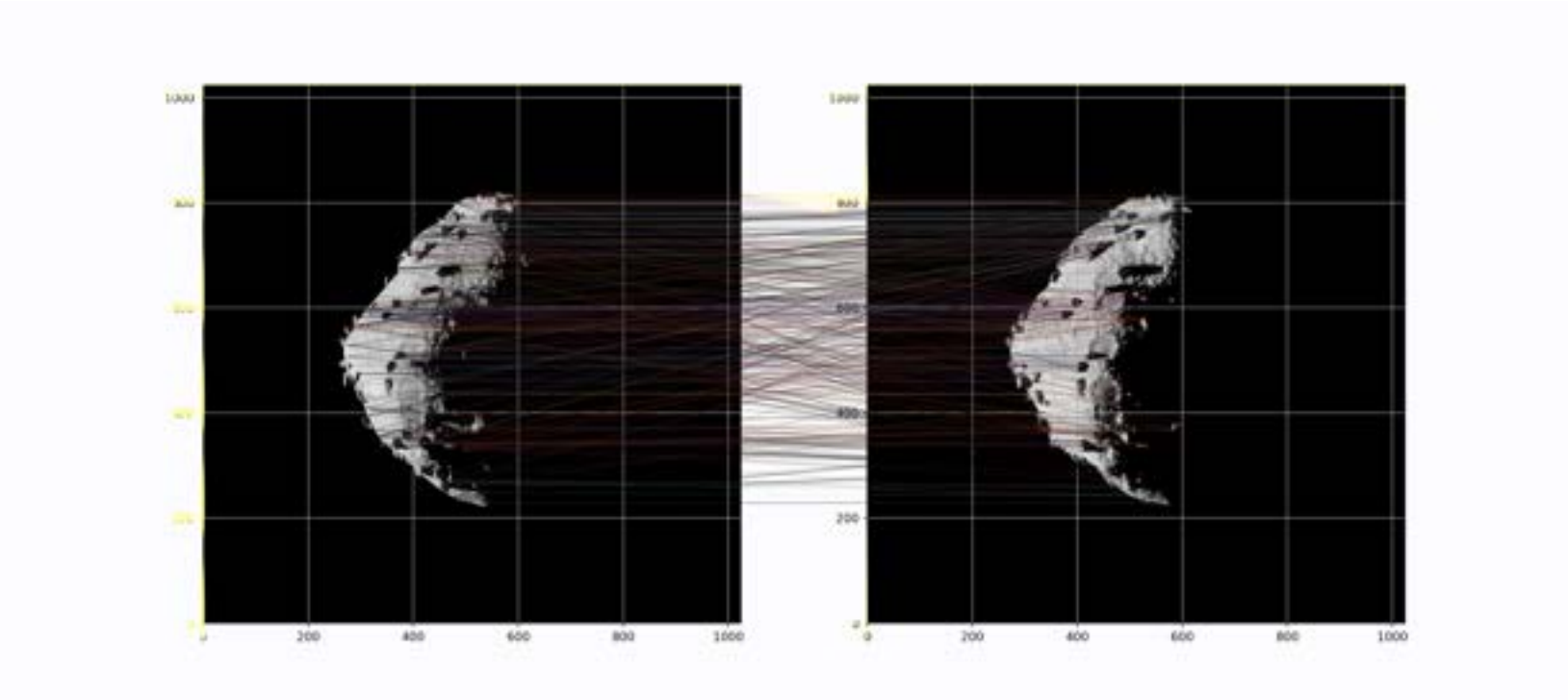
- Custom size
- Custom deformation
- Custom crater size/distribution
- Custom roughness
- Custom depth
- Custom boulder size/count

Dataset generation

Parameter	Value
Number of rocks	1
Roughness	$U(2, 10)$
Detail	4
Scale factor	$U^3(1, 3)$
Deform	$U(1, 10)$
Depth	$U(0.1, 0.5)$
Large rock count	$U(1, 10)$
Medium rock count	$U(10, 100)$
Small rock count	$U(100, 1000)$
Large rock size	$U(0.01, 0.03)$
Medium rock size	$U(0.003, 0.01)$
Small rock size	$U(0.001, 0.003)$

Table 5: Parameter summary for shape model generation

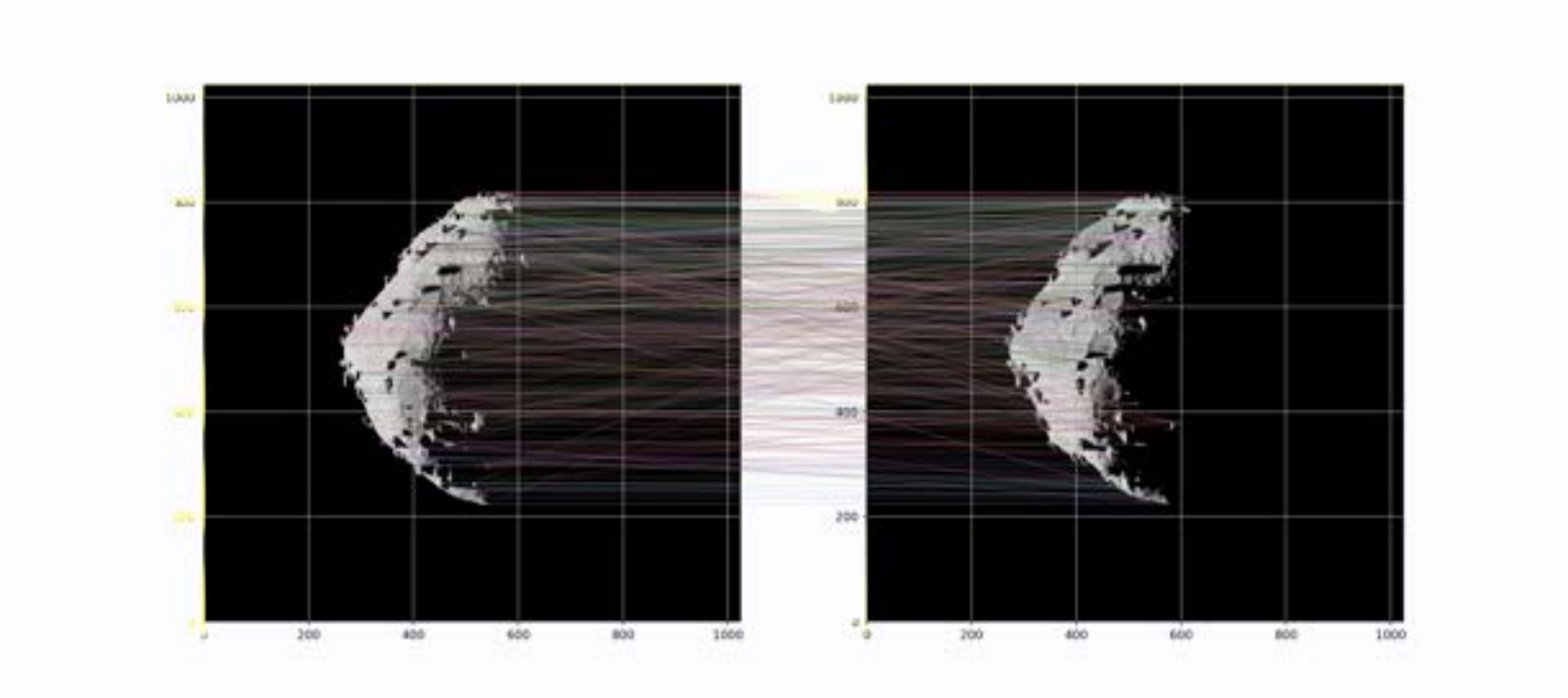
Qualitative comparison: SIFT



EPFL

Caltech

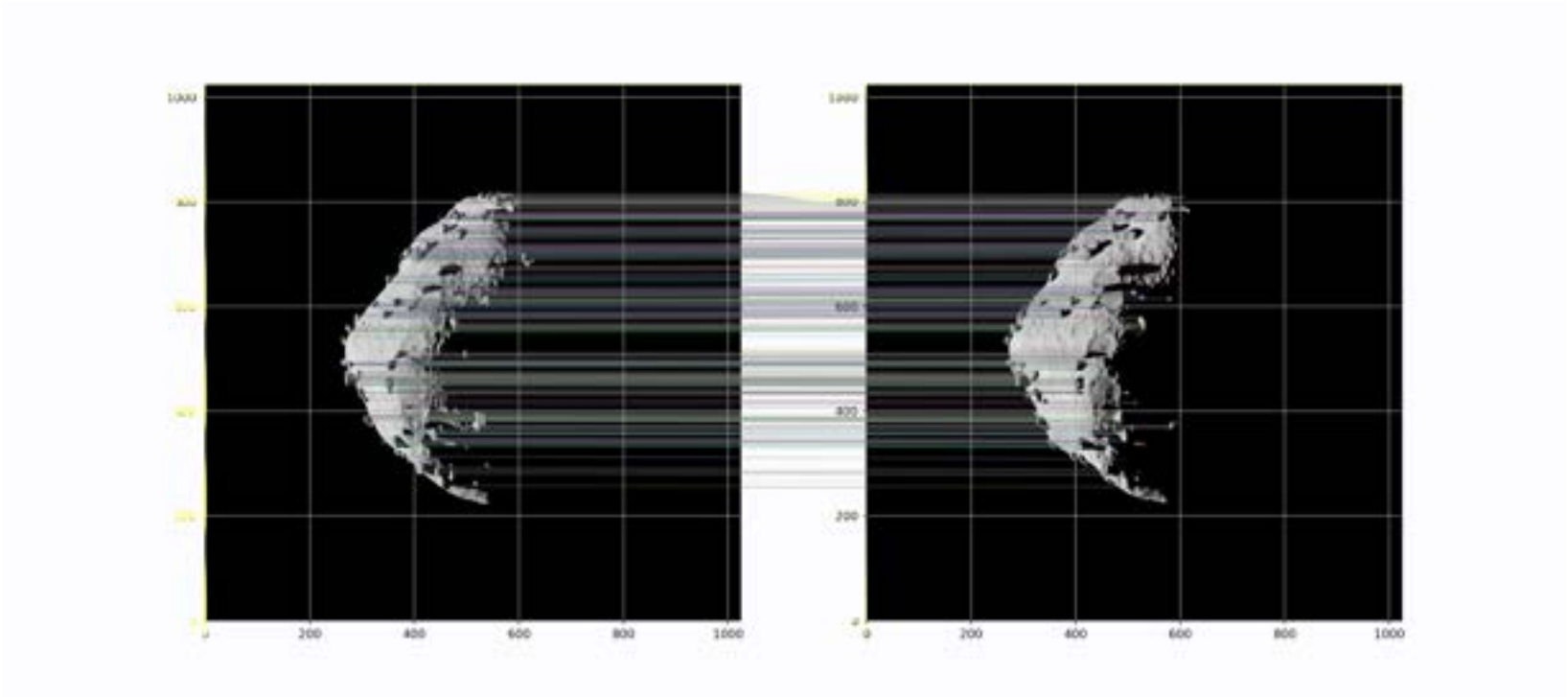
Qualitative comparison: Superpoint



EPFL

Caltech

Qualitative comparison: COFFEE



Quantitative results: Definitions

- Predicted match matrix M:

$$M_{ij} = \begin{cases} 1 & \text{if } f_i^A \text{ is predicted to match } f_j^B \\ 0 & \text{otherwise} \end{cases}$$

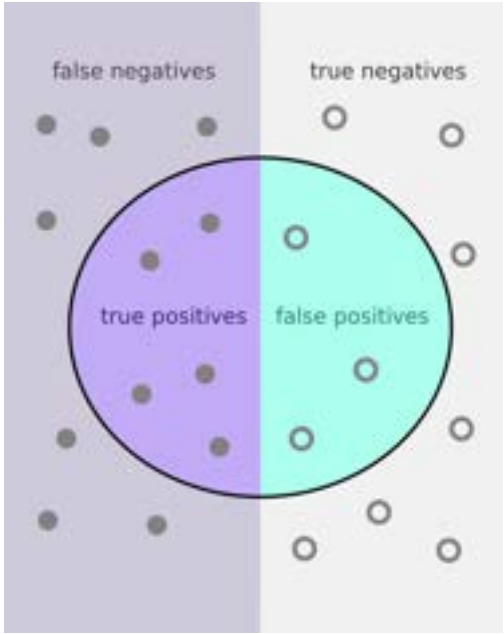
- Ground-truth match matrix G:

$$G_{ij} = \begin{cases} 1 & \text{if } ||P_A^B(c_i^A) - c_j^B||_2 < 1 \\ 0 & \text{otherwise} \end{cases}$$

- N.B. If $0 < i < N_A$ and $0 < j < N_B$, then M is of shape (N_A, N_B) but only contains $\min(N_A, N_B)$ positive elements

Quantitative results: Definitions

- Precision: $$P = \frac{\sum_{i,j} G_{ij} M_{ij}}{\sum_{i,j} M_{ij}}$$
- Recall: $$R = \frac{\sum_{i,j} G_{ij} M_{ij}}{\sum_{i,j} G_{ij}}$$
- F_1-score: $$F_1 = 2 \cdot \frac{P \cdot \mathrm{R}}{\mathrm{P} + \mathrm{R}}$$



Quantitative results

Algorithm	Precision (K=100)	Precision (K=200)	Precision (K=500)
COFFEE (ours)	**82.5%**	**77.5%**	**68.1%**
Superpoint	69.4%	52.9%	30.4%
ContextDesc	47.1%	45.1%	37.3%
Disk	67.5%	57.4%	41.3%
LFNet	16.2%	14.1%	10.7%
R2D2	16.9%	15.0%	10.9%
SIFT	17.4%	14.8%	10.8%
ORB	5.2%	4.3%	3.1%
AKAZE	9.4%	8.0%	5.1%

Table 8: Precision for a given number of features (best in **bold**)

Quantitative results

Algorithm	Error [rad]	Standard deviation [rad]
COFFEE (ours)	**0.028**	**0.021**
Superpoint	0.061	0.042
ContextDesc	0.061	0.049
Disk	0.15	0.14
LFNet	0.11	0.12
R2D2	0.064	0.061
SIFT	0.095	0.077
ORB	0.61	0.69
AKAZE	0.50	0.56

Table 15: Pose estimation bias and std dev. for the best K=500 features (best in **bold**)

Quantitative results

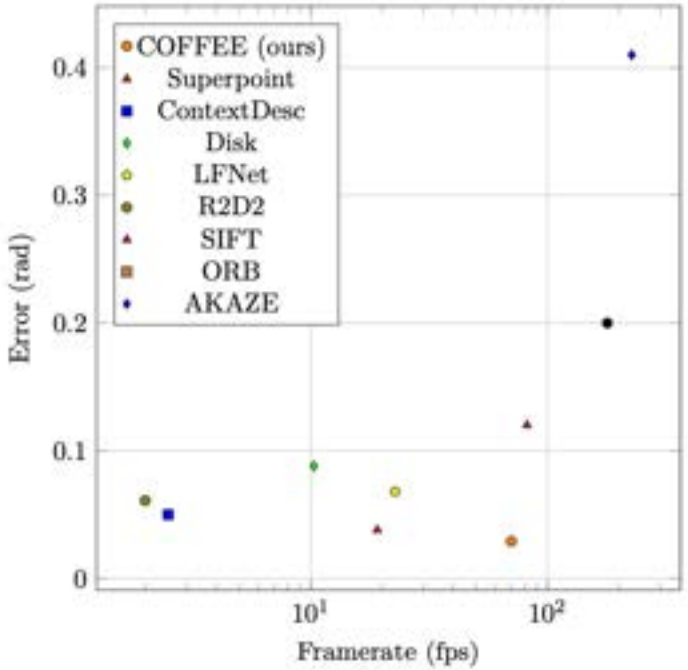
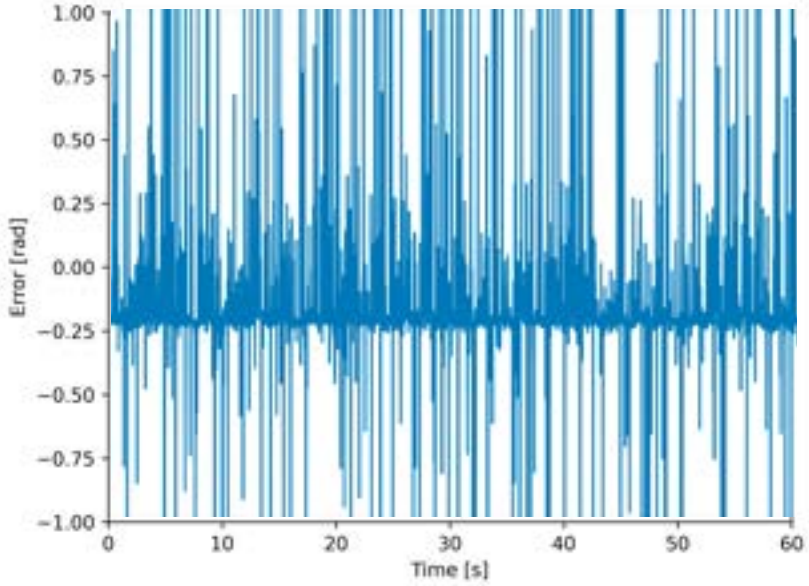


Figure 23: Trade-off between runtime and accuracy

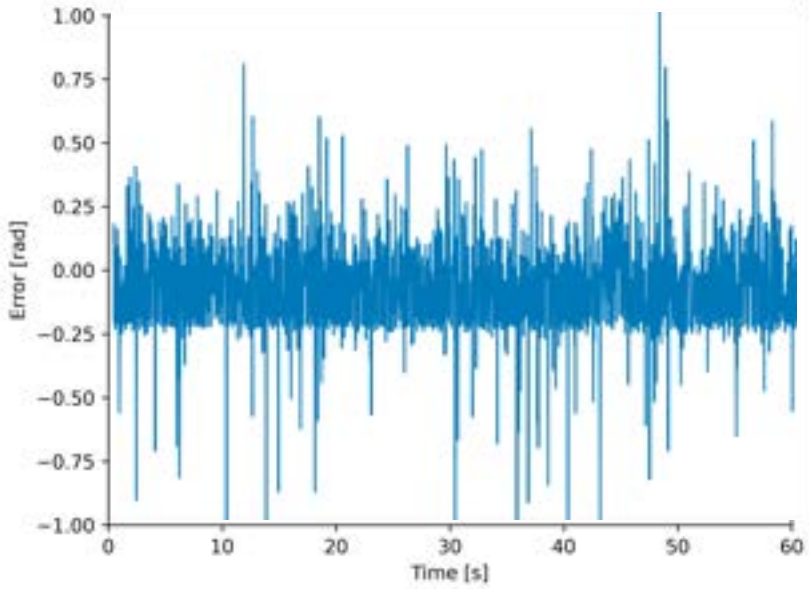
Conclusion

- COFFEE is a data-driven feature detector/descriptor extracting information from the shadow boundaries
- The full pipeline was benchmarked against other state-of-the-art algorithms through renderings of Apophis
- 3.5x faster than the fastest deep-learning algorithm
- 3.5x more accurate than the most accurate classical algorithm

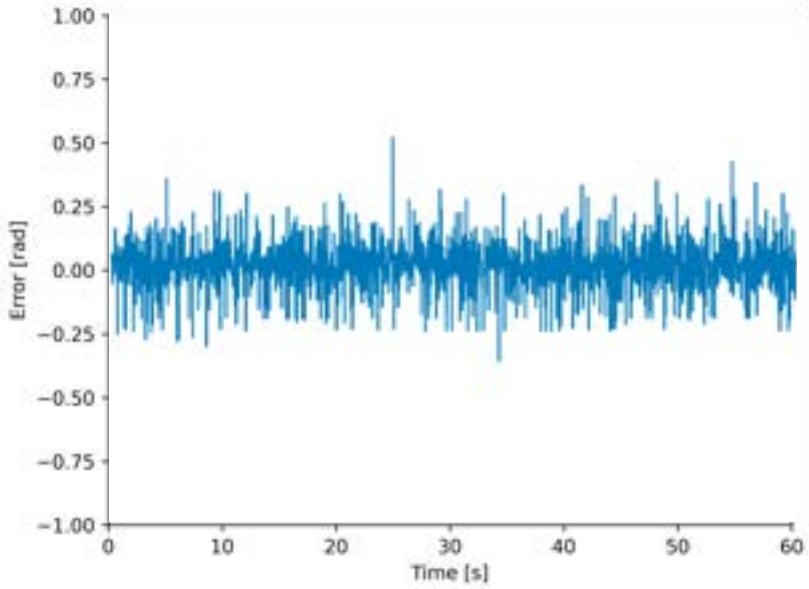
Why not SIFT?


1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
Error [rad]
0
10
20
30
40
50
60
Time [s]

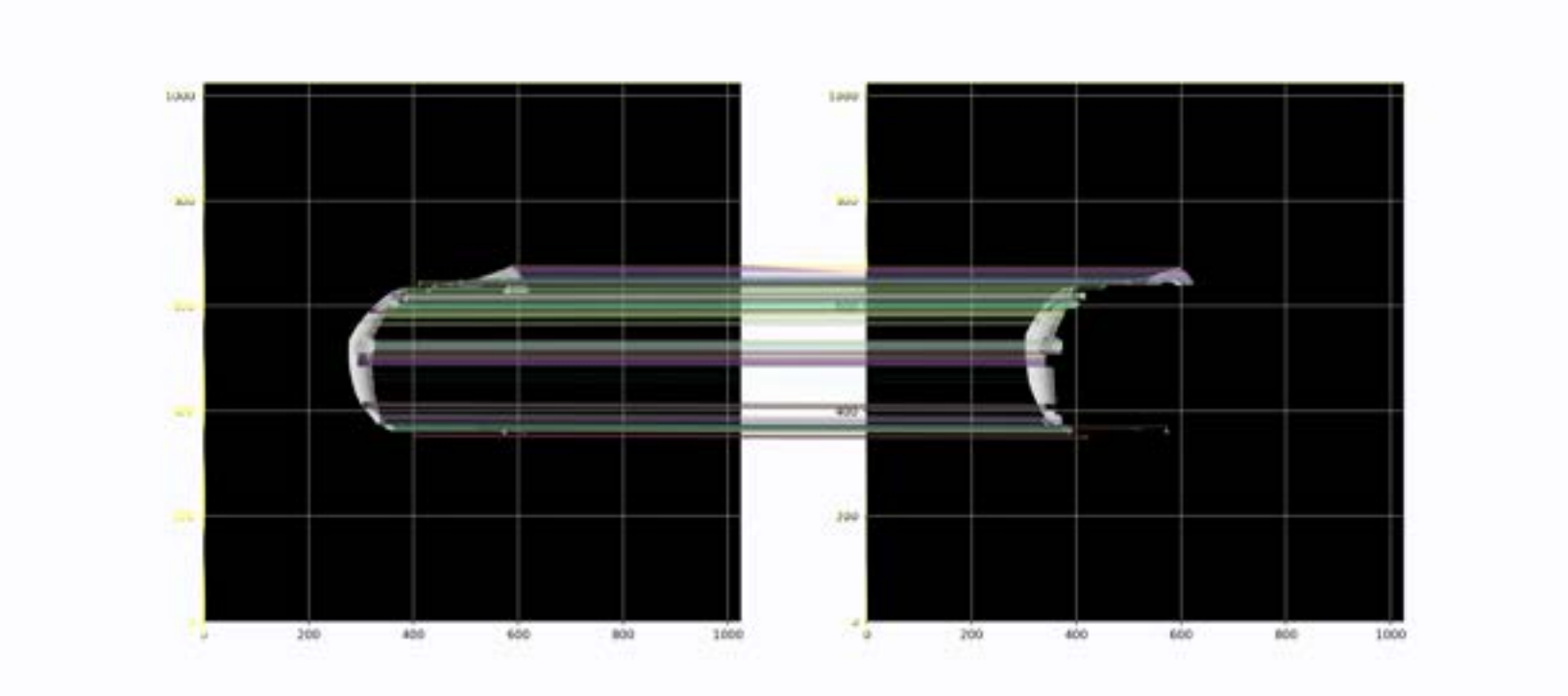
Why not Superpoint?



Why not COFFEE?



Why not COFFEE?



EPFL

Caltech