Vision-based Navigation supported by Convolutional Neural Networks for Lunar and Planetary Landing Missions

Pedro Maia Pinheiro
pedro.maia.pinheiro@tecnico.ulisboa.pt
Instituto Superior Técnico, Universidade de Lisboa, Portugal
July 2021

Abstract

Vision-based navigation systems are a prominent technology in the space industry. This method is based on using camera images as the primary navigation system to estimate spacecraft relative position and attitude, namely for rendezvous and proximity operations, such as lunar or planetary landing missions. In the last few decades, conventional image processing techniques are being replaced by Convolutional Neural Networks in a vast number of tasks and domains, since these Artificial Intelligence methods are outperforming on most benchmarks. Inspired by these promising advances, in this Thesis, we investigate Deep Learning alternatives to classical image processing algorithms, which may be applicable to image-based navigation in the scope of planetary landing missions. Thus, we propose a complete framework to evaluate any image feature detector for the task of motion states estimation during a planetary landing mission, using both representative and simulation datasets, as well as the most recent Perseverance Rover's landing video from NASA. To accomplish this goal, a solution based on homography relation is designed. A qualitative and quantitative evaluation of the whole pipeline is presented, comparing both classical feature detectors and one based on Deep Learning architectures. From our promising results for applying machine learning to this problem, we introduce an alternative to classical Computer Vision algorithms. Furthermore, we discuss some possible future work to improve the results.

1. Introduction

Navigation systems like GPS are only available on Earth, which means that spacecraft exploring other bodies in space need to estimate their position by different methods. Sensors, algorithms and onboard computing are able to substitute human visual navigation and even outperform it to enable safe landings in space. Cameras are replacing conventional sensors for Entry, Descent and Landing missions as they are versatile and lightweight, allowing absolute and relative navigation, hazard detection and avoidance, to provide precision and reach safe locations, on the Moon, Mars, and beyond, making vision-based navigation the most promising technology for lunar and planetary landings.

In recent years, CNNs are replacing classical image processing methods in a vast number of tasks and domains. Space industry is also starting to rely on these networks to solve problems, such as target detection and identification or pose estimation of space targets. Our purpose is to investigate the use of CNNs to image-based navigation in the scope of planetary landing missions.

Vision-based relative navigation uses a camera to identify surface features and compare their locations along the frame sequence in order to figure out the relative position and attitude with respect to the ones from previous time instants. For planetary landing missions, we rely in the assumption that the surface seen by the camera is planar, since terrain relative navigation (TRN) starts a few kilometers from the ground, and the problem can be tackled by homographies, that give exact or almost exact image-to-image transformations.

The first step is to detect and describe keypoints on two different images of the same scene. Then, we find correspondences between the points on both images and estimate an homography matrix that represents the transformation between the reference and current image. Finally, that relation is converted into euclidean coordinates and decomposed into relative rotation and translation between the cameras. If we choose a fixed reference frame, we can estimate attitude and translation of the spacecraft along the landing trajectory.

Our focus goes to the detection and description step, where we perform experiments using both classical and deep learning algorithms to compare the navigation estimates and find the advantages of moving into the machine learning approach. Our contributions include:

1. We propose a framework for evaluating feature detectors for the task of motion states estimation during a planetary landing mission, applicable to any video sequence dataset, assuming that ground truth positions and attitudes of the spacecraft+camera system are known.

2. We ran several experiments to compare classical feature detectors and one relying on machine learning with the purpose of getting the most accurate estimates of position and attitude on a landing trajectory.

3. We have shown that it is possible to use deep learning feature detectors, namely SuperPoint, to accurately perform an image-based navigation perception system in landing missions, and that it actually outperforms classical methods such as Harris Corners or SIFT, specifically when estimating relative pose between highly spaced frames.

4. We confirmed the possibility of using simulation datasets to evaluate the vision-based system performance and we have done experiments using the most recent and representative dataset of our problem, the Perseverance Rover's Descent and Touchdown on Mars from NASA.

2. Background

We use homographies to recover spacecraft poses during landing missions. Traditional image processing techniques have been used during the past years to solve this problem in a sequential process of detecting important points in images taken in successive instants of time, find a relation between these points in different images and then use these relations to compute the transformation between different viewpoints.

2.1. Feature Detection and Description

The first step is to identify repeatable and distinguishable keypoints in each image of the sequence and then, to find a way of reliably describing them. In the past years, engineers have proposed several handcrafted algorithms to detect and describe those points based on heuristics. One of the first and most popular attempts is called **Harris Corners** and was done in 1988 in [14]. Later, J. Shi and C. Tomasi proposed a small modification to the Harris Corner detector in [15]. Both only return corner locations, with no descriptors, and they fail when there are large scale changes. Then, **SIFT** was presented in [10] and [11], and later a speeded-up version, called **SURF**, in [6]. These two got good results, but they were patented for several years. Thinking in real-time applications, some feature detectors are not fast enough. So, a new algorithm called **FAST** was proposed in [3] and later revisited in [4], to detect feature locations. Likewise, there is a growing need for local descriptors that are fast to compute, fast to match, and memory efficient. Hence, **BRIEF** descriptor was proposed in [12]. From the "OpenCV Labs", came out **ORB** in [5], which is a fusion of FAST detector and BRIEF descriptor with some modifications on both. A lot more efforts were done and a lot more hand-engineered detectors and descriptors have been made during the years.

Recently, several approaches were proposed to tackle feature detection and description using CNNs. In 2016, a novel deep network, called **LIFT**, was introduced in [9]. Later, **SuperPoint** [2] comes from a self-supervised framework for training interest point detectors and descriptors. **LF-Net** article [16] proposed a novel deep architecture and a training strategy to learn a local feature pipeline from scratch, using images without the need for human supervision. **Key.Net** [1] uses a combination of handcrafted and learned CNN features to produce a keypoint detector. **D2-Net** [13] is a close approach to SuperPoint as it also shares a deep representation between detection and description. **R2D2** [8] introduced the idea of reliability apart from repeatability. The problem of efficiently and accurately detect and describe interest points using CNNs for higher level computer vision applications is still an open field of research these days.

2.2. Feature Tracking

Features can be tracked along a video sequence using one of two methods. **Optical flow** methods rely on the minimization of the brightness constancy of an object between consecutive frames, so they are more suited to image sequences than image pairs from different views and, due to the assumptions made, the main problems are large motions, occlusion, strong illumination changes and changes of the appearance of the objects.

Feature matching uses feature descriptors to match features with one another by a nearest neighbor search in the feature space. Feature matching algorithms are far better if there is a perspective difference between the images, or the frames, or when the transformations are large, e.g., for a wide interval between frames. They are scale and rotation invariant and are robust to changes in illumination, as those caused by shadow or different contrast.

Both methods produce a list of correspondences between points on both images.

2.3. Projective Homography Estimation

Let $p_i = (u_i, v_i, 1)$ be the vector containing the homogeneous coordinates of a point in the first image and let $p'_i = (u'_i, v'_i, 1)$ be the homogeneous coordinates of the same point in the second image. The projective homography matrix transforms p_i into p'_i, up to a scale factor and can be solved using the Direct Linear Transform algorithm, as explained in chapter 4.1 in [17], requiring a minimum of 4 correspondences between points on both images.

In general, there are many more than four corresponding points in a pair of images, some of them being outliers that will degrade the solution if taken into account using Least Squares. RANSAC algorithm is applied to reject these outliers in order to get a more accuracte estimate of the homography.

2.4. Pinhole Camera Model

The information from the 3D world is projected into a 2D plane when a camera acquires an image. The simplest way of representing this transformation is the pinhole model. A point in the world $P = (X, Y, Z)$ is mapped into the projection plane to a point $p = (x, y)$, which is the intersection of the projection plane and the projection line that contains the point P and the camera centre. Similar triangles allow to calculate the coordinates of the projected point,

$$p = (x, y) = \left(f\frac{X}{Z}, f\frac{Y}{Z}\right) \tag{1}$$

Homogeneous coordinates can simplify the transformations between points.

$$\bar{p} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \equiv \begin{bmatrix} fX \\ fY \\ Z \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2}$$

The same way, can be written in a concise form as,

$$\bar{p} = K[I|0]\bar{P} \tag{3}$$

where K is called the *camera calibration matrix* or **intrinsics matrix** and can be written as

$$K = \begin{bmatrix} \alpha_x & s & u_0 \\ 0 & \alpha_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4}$$

where $\alpha_x = fm_x$ and $\alpha_y = fm_y$ represent the focal length of the camera in pixel dimensions. f represents the focal length in meters, and m_x and m_y the number of pixels per unit distance in both directions. Similarly, the principal point needs to be converted to pixels, using $u_0 = \alpha_x o_x$ and $v_0 = \alpha_y o_y$, where (o_x, o_y) are the coordinates of the principal point in the image frame. s represents the *skew* factor

2.5. Euclidean Homography

World points can be represented with respect to different frames.

$$P_2 = RP_1 + T \tag{5}$$

where P_1 and P_2 represent the coordinates of a point P in the world according to frame 1 and frame 2, respectively. $R = R_1^2$ is the rotation matrix from frame 1 to frame 2 and $T = t_{2\to 1}^2$ is the translation vector of the frame 2 to frame 1 represented in the frame 2.

When all points lie on a plane, we have another constraint,

$$\frac{1}{d}N^T P_1 = 1 \quad \text{for points} P_1 \text{in the plane} \tag{6}$$

Substituting Eq. 6 in Eq. 5, the transformation becomes,

$$P_2 = RP_1 + T\frac{1}{d}N^T P_1 = HP_1 \tag{7}$$

where

$$H = R + \frac{1}{d}TN^T, \quad H \in \mathbb{R}^{3x3} \tag{8}$$

H is known as *Euclidean Homography* or *planar homography matrix* that allows to compute the relative pose between the cameras. Points in the world represented in camera frame can be converted up to a scale into the image frame using the camera matrix,

$$\bar{p}_1 = \alpha_1 K_1 \bar{P}_1 \quad \text{and} \quad \bar{p}_2 = \alpha_2 K_2 \bar{P}_2 \tag{9}$$

Assuming the same camera in both frames, $K = K_1 = K_2$, we can derive the relation between the homogeneous pixel coordinates of common points in both images.

$$\bar{p}_1 = \gamma K \left(R + \frac{1}{d}TN^T\right) K^{-1}\bar{p}_2 \tag{10}$$

where γ is the scale factor. Now, we can relate the projective homography with the euclidean homography,

$$H_{proj} = \gamma K H_{euc} K^{-1} \tag{11}$$

3. Implementation

We evaluate the performance of several image processing algorithms and we aim at proposing a more accurate monocular vision-based relative navigation system. Our implementations and evaluations use two classical algorithms, Harris Corners and SIFT, as reference, and SuperPoint as the machine learning alternative in test.

3.1. Software Tools

A software pipeline was implemented in *Python* language. Since we are working with images and many matrix calculations are needed, the implementation makes use of Python libraries that are optimized for real-time image processing applications such as OpenCV[1] and Numpy[2], and

[1] https://opencv.org/
[2] https://numpy.org/

that are optimized for neural networks computations in both CPU and GPU, such as PyTorch[3].

3.2. Evaluation Datasets

3.2.1 Spin.Works UAV Landing Mission Dataset

Spin.Works ran several experiments of landing missions using UAVs in representative terrain surfaces such as a quarry. From them resulted a video that was highly used during our performance evaluations, which was acquired by a camera attached to the UAV looking downwards during the landing phase in a quarry. Spin.Works used a Structure-from-Motion (SfM) software, which uses the first 249 frames to compute the orthophotograph of the terrain surface, the position and attitude of the camera/aircraft, an estimation of the model of the camera used and the point cloud of the terrain. Hence, we get the motion states we use as reference.

3.2.2 Spin.Works Moon Landing Mission Simulation

Getting real datasets with the ideal representation of the problem is a difficult task. Spin.Works generated some of these simulations using PANGU[4], which is a proprietary software from STAR-Dundee[5] often used by ESA. Hence, from simulated Moon landing trajectories, the software produced a collection of synthetic images reproducing the environment, forming a video sequence in a completely controllable environment where the camera parameters, the motion states and the 3D of the surface are perfectly known.

3.2.3 Perseverance Rover's Descent and Touchdown on Mars (Official NASA Video)

NASA posted online many images and videos from the Mars landing mission, including the video[6] from Perseverance rover's descent and touchdown on February 18, 2021, seen by a down-looking camera. We have extracted the frames from the YouTube video, at 10 Hz, and then resized each frame to 512×512 dimension, forming a set of 1310 images of the surface from Mars during descent. NASA did not publish navigation data for us to compare our estimates. However, at Spin.Works, they used, once again, the SfM software to get position and attitude of the camera at each frame from the frame sequence. Then, we have rotated data so that the last frame is vertical to the terrain surface. Finally, since backshell separation occurs at about 2.1 kilometers and that frame is visible on the sequence, we scaled the position, resulting in a trajectory starting at 11 kilometers from the surface of Mars, which is consistent with the EDL diagram published by NASA.

[3]https://pytorch.org/

[4]https://pangu.software/

[5]https://www.star-dundee.com/

[6]Video URL: https://www.youtube.com/watch?v=4czjS9h4Fpg

3.3. Feature Detection Algorithms

3.3.1 Harris Corners

Harris detector was implemented using the function *goodFeaturesToTrack* from OpenCV with the flag *useHarrisDetector* set to True. This function returns a list of coordinates for the features identified. For each of these features we computed a descriptor as the 15×15 patch of intensity values around the pixel location and flattened to a vector of dimension 225. The similarity measure used to compare descriptors was *normalized cross-correlation*.

3.3.2 SIFT

SIFT algorithm was also implemented using OpenCV. In this case, the function returns the keypoints' locations and also the SIFT descriptors. The default parameters were used, we also chose the maximum number of features to be similar to the other algorithms during performance evaluations. The similarity metric adopted was the *L2 norm*.

3.3.3 SuperPoint

SuperPoint was implemented and trained from scratch using PyTorch framework, following the procedure described in the paper [2], but with slight differences. The descriptor loss was slightly modified from the original paper inspired by the paper [7] (section 3.2 paragraph Matching layer) and the *Tensorflow* implementation in the repository https://github.com/rpautrat/SuperPoint. The descriptor loss is now computed on L2 normalized descriptors and, after computing the distance matrix between the descriptors of both images, we have applied ReLu activation to eliminate obvious non matches (similarity of the descriptor below 0). We have implemented the L2-norm as a similarity measure for the descriptor matching. Operations were implemented with the possibility of using GPU acceleration due to parallelization of most computations.

SuperPoint corner locations are defined with pixel coordinates, while other detectors like SIFT have sub-pixel precision. So, we have implemented a method to refine corner locations from SuperPoint. After getting the corner locations, we apply the function *cornerSubPix* from OpenCV that iterates to find the sub-pixel accurate location of corners.

3.4. Homography Estimation & Motion Reconstruction

3.4.1 Ground Truth Homographies

Our datasets do not have explicit ground truth homographies between the frames along the video sequences for us to use as reference. Nevertheless, it is possible to compute reference homographies using the camera model and the motion parameters.

Our approach is to compute the transformations between the ground surface plane and the image plane, projecting the camera versors, calculated using navigation states, into the ground surface and getting the 4 corners of the surface region seen by the camera, in meters. The ground surface plane remain constant along the frame sequence. Therefore, it is possible to calculate inter-frame homographies, by the formula,

$$H_i^j = H_j H_i^{-1} \tag{12}$$

where H_i^j represents the homography between frames i and j, and H_i and H_j are the transformations between the world surface and the image plane, respectively.

3.4.2 Test Set Homographies

The procedure to compute the test set of projective homographies is independent of the algorithm chosen for the feature detection, description and matching. The homography matrix is computed from the correspondences between features from different frames using the DLT algorithm. However, not all correspondences are correctly identified and we have implemented the RANSAC algorithm to reject outliers and compute the estimate only with correspondences that are considered inliers using a threshold on a geometric distance between estimated points and original points.

3.4.3 Non-Linear Least Squares

Homographies are used to estimate camera positions and attitude in order to reconstruct the motion of the spacecraft. Matrices computed from image processing techniques must be converted to euclidean homographies, as described in section 2.5, by the formula,

$$H_{euc} = \frac{K^{-1} H_{proj} K}{\gamma} \tag{13}$$

where γ is the scale factor. If we notice that the median of the singular values of H_{euc}, then we can compute the scale factor γ as follows:

$$\gamma = med(svd(K^{-1} H_{proj} K)) \tag{14}$$

Pose information is implicit in euclidean homographies and can be recovered defining a model using equation 8. Our goal is to produce an estimate of the vector of parameters, x, which we call *state vector*, that represents the position and orientation of the aircraft at each time frame.

The model is non-linear, which complicates the problem. However, we can assume that given an initial rough condition for our state vector x, it is possible to produce some cost function where our state space is locally convex and, then, where it is possible to converge to a local minimum that matches the optimal estimate of x. This problem can be described as a local optimization problem, and can be solved iteratively, using **non-linear least squares** (NLLS). The linearization of the homography function becomes

$$H_i^f(x) = H_i^f(x_0) + \frac{dH_i^f}{dx}(x_0)[x - x_0] \tag{15}$$

where $H_i^f(x)$ represents the estimate of the euclidean homography at state x, $H_i^f(x_0)$ the result of applying the homography model at initial state vector, x_0, $\frac{dH_i^f}{dx}(x_0)$ denotes the Jacobian of the model at x_0, explicitly defined as $\left[\frac{dH}{d\phi}, \frac{dH}{d\theta}, \frac{dH}{d\psi}, \frac{dH}{dt_x}, \frac{dH}{dt_y}, \frac{dH}{dt_z}\right]$, a 9x6 matrix and, finally x stands for an improved estimate of the actual state vector (rinse repeat).

A key frame must be given, a camera whose navigation motion states are known: position (altitude above ground) and attitude. The rotation matrix $R(\phi, \theta, \psi)$ is replaced by $R(\phi, \theta, \psi)R_0$ in the model, where R_0 is the rotation matrix from the key frame camera to the (ground) local vertical reference frame.

At the end of each convergence (between any two frames), we store the estimated attitude (ϕ, θ, ψ) and translation $t = (t_x, t_y, t_z)$. The translation vector needs to be first transformed back to the reference frame of the key-camera-frame, through: $t = -R(\phi, \theta, \psi)^T \times t$, before storage.

3.5. Evaluation Metrics

The objective of this Thesis is to evaluate the performance of the image processing algortihms to reconstruct the motion of the aircraft in landing missions. Therefore, our evaluations were conducted in three phases. We started by testing the repeatability of the points, then the accuracy of homography estimation and finally we got to a more advanced metric where we compare the motion states directly.

On one hand, we detect points on every frame and compute estimates of the homography relating pairs of successive frames. The transformation from the world plane to the i^{th} frame plane can be estimated as a sequence of frame-to-frame homographies multiplication as follows:

$$H_w^i = H_{i-1}^i H_{i-2}^{i-1} ... H_0^1 H_w^0 \tag{16}$$

We tested different conditions, such as changing the reference frame and the *delta* between frames, i.e., the interval between frames we use to estimate the homographies. On the other hand, we chose the key frame, detect the keypoints and save the descriptors. Then, we perform directly the estimation of the homography between the key frame and the current frame. For each frame i we get the expression

$$H_w^i = H_0^i H_w^0 \tag{17}$$

These experiments allows to evaluate the strength of the features detected by the different algorithms so we can perceive the ones that keep on being detected during longer periods of time. This procedure contributes to the elimination of the cumulative error caused by the composition of successive homographies.

4. Results

4.1. Spin.Works UAV Landing Mission

The Spin.Works UAV landing mission dataset is our first approach to compare algorithms in an environment similar to what is expected in a planetary landing mission.

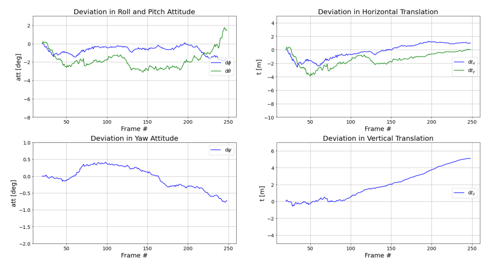
We started by evaluating the performance of the algorithms in the task of estimating the homographies that represent the image transformations between different frame transitions. First, we tested all the transitions between consecutive frames and then, we have successively increased the interval between the frames. Results indicate that SuperPoint performs comparably to the classical methods in the case of intervals of consecutive frames and slightly better when we increase the delta between frames, specially when compared to SIFT. This consequence suggests that SuperPoint could be a powerful alternative for transitions that are more spaced in time. The detected corners remain being correctly matched along longer time sequences.

However, these metrics are insufficient to demonstrate that SuperPoint keypoints and descriptors are advantageous over Harris Corners and SIFT at the high level task of estimating motion states for the vision-based navigation problem. Qualitative results from matchings between highly spaced frames also shown that SuperPoint has a stronger performance when facing high scale changes.

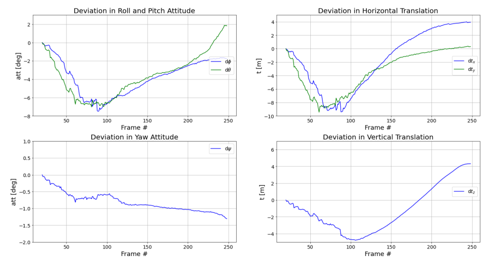
We tested our algorithm using the reference set of homographies, so that we could verify that the motion recovery algorithm is correctly implemented and that it does not contribute with errors to the final solution. Therefore, the estimate deviations from the reference can be used as an evaluation metric of the quality of the homographies produced by the different methods.

4.1.1 Reference frame: 20, Delta between frames: 1

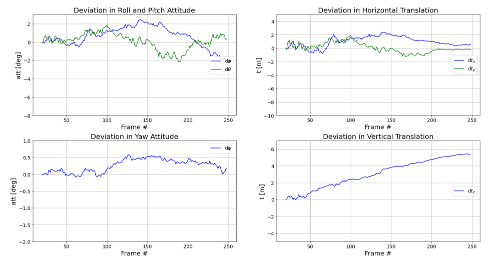
We defined one reference frame from the beginning of the sequence, e.g. frame 20, and the smallest interval possible, delta between frames equal to 1. The deviations between the estimated translations and attitudes by the three methods and the reference values are plotted in Figure 1. We can visually notice that SIFT exhibits the worst performance among the three, since estimates are much deviated from the reference values. Harris Corners and SuperPoint perform similarly, although SuperPoint seems to be slightly more accurate, namely in pitch and y-translation.



(a) Harris Corners



(b) SIFT



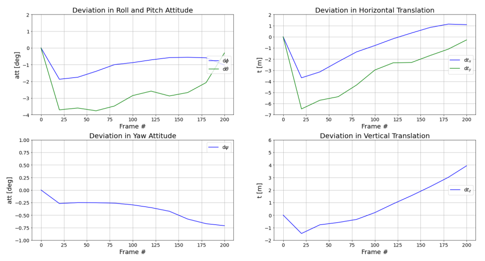
(c) SuperPoint

Figure 1. UAV landing pose recovery deviations, using homographies calculated from image processing techniques with different features detectors. Frame 20 is used as reference and delta between frames is equal to 1. Inter-frame homography matrices are multiplied to get the transformation of the current frame with respect to the reference frame.

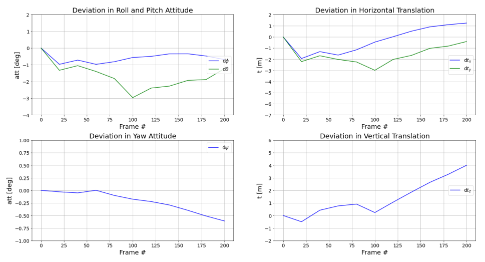
4.1.2 Reference frame: 1, Delta between frames: 20

We performed several experiments changing the reference frame and increasing the delta between frames. In the current condition the frame 1 is the key frame and the delta is equal to 20. Results are presented in plots from figure 2. It is trivial to see that, using SuperPoint, the estimates are clearly better. Both Harris and SIFT fail badly on the final transitions of 20 frames, making estimates completely impractical. To maintain the readability of the plots, we have cut the last estimates from both classical algorithms and that is why the x-axis is shorter than in SuperPoint where the estimate remain accurate until the end of the sequence, since the deviations stay much lower than using Harris or SIFT.

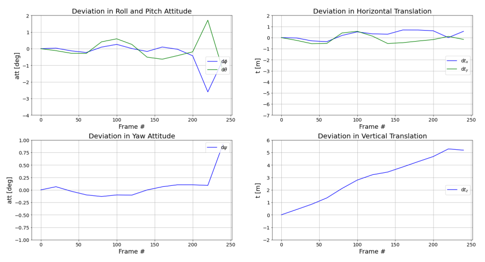
Besides that, we can compare the accuracy of the predictions in the first frames' transitions and the difference is visually noticeable.



(a) Harris Corners



(b) SIFT



(c) SuperPoint

Figure 2. UAV landing pose recovery deviations, using homographies calculated from image processing techniques with different features detectors. Frame 1 is used as reference and delta between frames is equal to 20. Inter-frame homography matrices are multiplied to get the transformation of the current frame with respect to the reference frame. Last estimates cut from both classical algorithms, as Harris and SIFT fail badly on the final transitions.

This method of relative navigation, similar to dead reckoning, is subject to cumulative errors. The composition of the homographies accumulates errors that are introduced in individual inter-frame estimations. For long periods of time, cumulative errors can lead to obsolete information about position and attitude of the aircraft. For that reason, it is important to increase the delta between frames, which reduces the number of estimations made and, consequently, the accumulation of errors. It can be used as an additional source of information to the perception system, like another sensor to introduce into the navigation filters to help rectify the estimates of pose state variables.

4.1.3 Limit for delta between frames

At second pipeline of our experiment, we aim to evaluate the longest period of frames in which the same features remain being correctly detected and matched, producing accurate estimates of pose states. Here, we do not rely on composition of inter-frame homographies. Instead, we perform directly the homography estimation between the reference frame features and the ones from the current frame. We perform experiments using different reference frames and count the number of frames until when deviations start exploding and the estimation become very unstable. Table 1 shows results for the three algorithms for different key frames. The numbers represent the period of frames starting from each reference, where estimations are accurate and stable.

Reference Frame	1	45	90	150	200
Harris Corners	70	53	44	29	17
SIFT	70	55	15	**44**	24
SuperPoint	**100**	**84**	**61**	42	**26**

Table 1. Number of frames until degradation of motion states estimates for UAV landing video sequence, from different reference frames

Once again our conviction is confirmed. SuperPoint actually shows better results for longevity of detected features and pose estimation between long periods between frames is significantly more stable and accurate using SuperPoint than classical methods, which indicates that the learned detector and descriptors are more robust and invariant to scale, translation and rotation changes in environments similar to lunar or planetary surface.

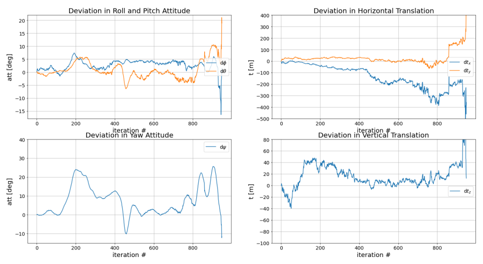
4.2. Spin.Works Moon Landing Mission Simulation

The main advantage of the lunar landing simulation is that we can easily vary the conditions, such as the trajectory, the landing site or the camera model and we have a completely controllable environment where we known for sure the motion states.

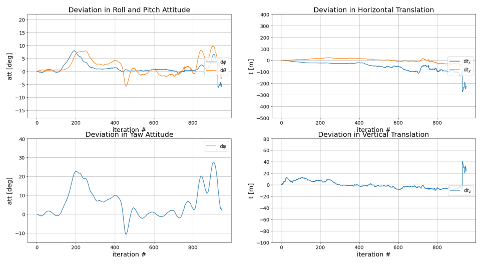
4.2.1 Reference frame: 1, Delta between frames: 1

Once again, we started by testing the methods with the simplest scenario of using the first frame as reference and composing the inter-frame homographies (calculated with a delta between frames equal to 1) to obtain the relation between the current and reference frames. We also the deviations of the estimated translations and attitude from the real values defined by the simulated trajectory of an EDL mission to the Moon, Figure 3.

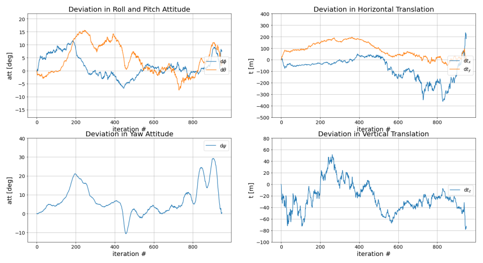
Results show that all three methods are able to retrieve motion states fairly well. However, SIFT exhibits the best



(a) Harris Corners



(b) SIFT



(c) SuperPoint

Figure 3. Moon landing simulation pose recovery deviations from reference states, using homographies calculated from image processing techniques with different features detectors. Frame 1 is used as reference and delta between frames is equal to 1. Inter-frame homography matrices are multiplied to get the transformation of the current frame with respect to the reference frame.

performance among all, as the estimates are far closer to the references, namely the translation ones. SIFT detections have sub-pixel precision, while both Harris Corners and SuperPoint keypoints are defined at integer coordinate locations. Since this dataset has a larger number of frames, there are a lot more transitions to estimate, which may result result in more cumulative errors as we go further from the reference. Besides that, translations are a lot higher than the ones from the UAV dataset. Sub-pixel precision may reduce the errors, which gives SIFT a great advantage and supports the results.

To tackle this problem and make SuperPoint competitive to SIFT, we tested to introduce the function *cornerSubPix* from OpenCV to the corners detected by SuperPoint in order to refine corner locations and allow sub-pixel precision. Results are presented in figure 4. As we expected, estimates are far better now and they can reproduce well both attitude and translation until the end of the trajectory, demonstrating less accumulated errors at the end than SIFT, namely in the x-axis from horizontal translation.

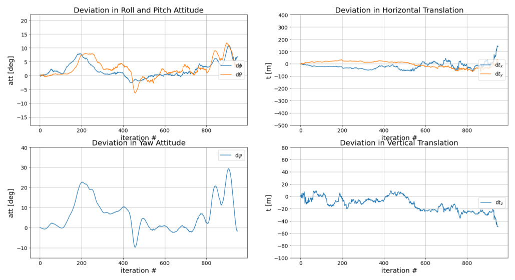


Figure 4. Moon landing simulation pose recovery deviations from reference states, using homographies calculated from SuperPoint-cv.cornerSubPix detections. Frame 1 is used as reference and delta between frames is equal to 1. Results should be compared to Figure 3

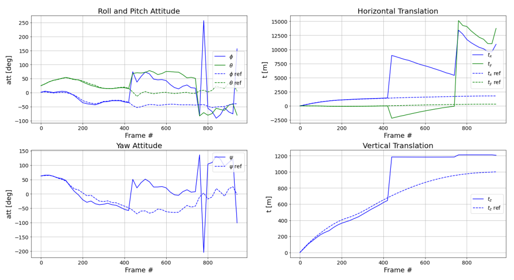
4.2.2 Reference frame: 1, Delta between frames: 20

Next, our experiments concern larger intervals between frames. We have increased delta and significant differences arise for intervals of 20 or 30 frames. Results are represented in figure 5 for delta equal to. Once again, we prove our conviction that SuperPoint brings enormous advantages when we increase the period between the frames, as its detections and descriptors are stronger and more invariant than those using classical methods. Estimates using SuperPoint remain valuable until the end of the sequence. The same does not happen with Harris Corners or SIFT. SIFT performs well until frame 750. From then on, it clearly fails to estimate the homography, which is reflected by the big deviation in the final transitions. Harris Corners fails a lot earlier in the sequence. Figure 5 shows an enormous deviation after frame 400 and another at about frame 750. The plot was cut to maintain readability, but deviation ”steps” keep on happening and getting bigger until the end of the video. We can confirm that SuperPoint is a lot more stable, as estimations did not explode in any condition from these experiments.

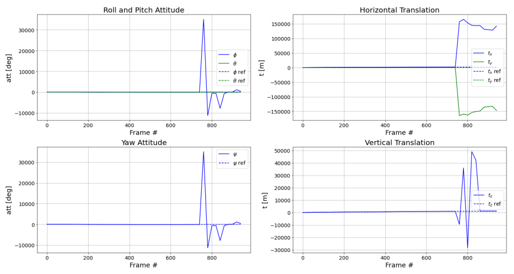
Results from simulation sequences are important because they are much easier to create and to change conditions in order to prepare the algorithms for the environment and dynamics the aircraft will come across when on a real mission. It allows to predict algorithm’s behavior a prior and to parameterize the system to the specific conditions of the mission.

4.3. Perseverance Rover’s Descent and Touchdown on Mars

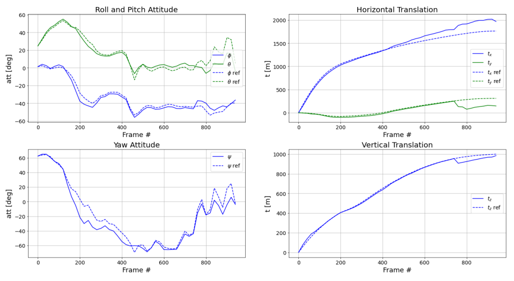
We do not have the actual navigation data from NASA, and the reduced set of information regarding the camera



(a) Harris Corners



(b) SIFT



(c) SuperPoint + cv.cornerSubPix

Figure 5. Moon landing simulation pose recovery, using homographies calculated by image processing techniques, with frame 1 as reference and delta between frames equal to 20

specifications, the reference altitude associated with the key frames and the whole video sequence as a whole, forced us to make several educated guesses during the process of reconstructing the navigation states using SfM methods. Moreover, the quality markers of the SfM results are not so good, indicating that the SfM process found a lot of inconsistencies in the data and, therefore, the quality of the results is questionable. All this uncertainty makes the available navigation results from SfM only valid for qualitative analysis and not quantitative. Still, the novelty of the data justified it being included here.

4.3.1 Reference frame: 50, Delta between frames: 10

The first frames occur while the heat shield is separating, so it appears in the images. Therefore, we have selected frame 50 to be our reference and estimated homographies with a delta equal to 10 until the end of the sequence. Motion states estimated using SuperPoint are represented in

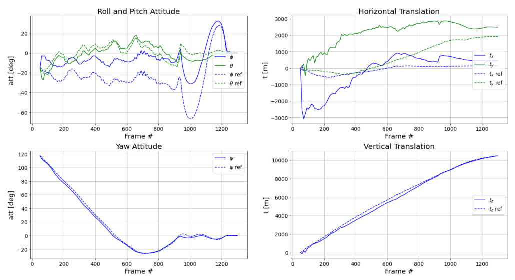
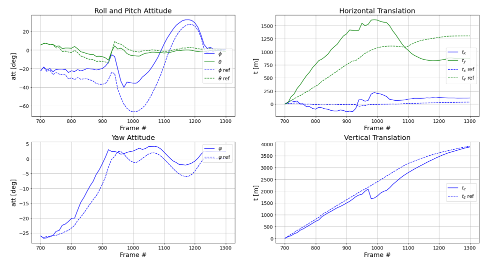


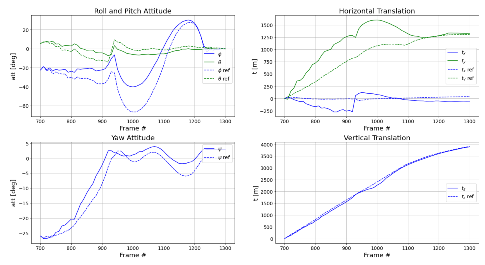
Figure 6. Perseverance landing pose recovery, using homographies calculated by SuperPoint keypoints, with frame 50 as reference and delta between frames equal to 10

figure 6, having a very accurate agreement in attitude and vertical translation, specially in terms of trend following, i.e., neglecting the bias and focusing on the dynamic behavior. Harris Corners is not represented because the results were completely damaged and were considered not significant and SIFT did not detect any point in frame 50 or other frames around, which reinforces our conviction that SuperPoint is a good alternative to classical methods. Unfortunately, the horizontal translation, which is the most important result for GNC given that the remaining states are being well observed by other sensors aboard the spacecraft, show a poor agreement with the reference from SfM.

4.3.2 Reference frame: 700, Delta between frames: 10



(a) Harris Corners



(b) SuperPoint

Figure 7. Perseverance landing pose recovery, using homographies calculated by image processing techniques, with frame 700 as reference and delta between frames equal to 10

From NASA's plan for the EDL mission, we get that TRN starts at about 4 kilometers above the surface. In a mission similar to this one, our algorithms would work at that phase. So, we decided to perform experiments starting at that point. From the position data we use as ground truth, we found that the rover is at an altitude of 4 kilometers at about frame 700. Only SuperPoint and Harris Corners produced valuable results, that are represented in figure **??** for delta equal to 10. SIFT clearly failed a lot transitions along the sequence, in both conditions, which made results insignificant, that is why we do not present them. SuperPoint and Harris Corners show similar results, but SuperPoint is getting slightly lower deviations, specially in translations.

We did not go further in detail on this dataset because there are a lot of information that can be not 100% tunned. Despite all uncertainties we consider this evaluation very important and promising since this is the most recent and representative dataset for our problem and the satisfactory results points that we are following the right path and this is still an open field for investigation.

5. Conclusions

To accomplish autonomous navigation, the spacecraft needs to know where it is (perception). Specially in space missions, where GPS is not available, cameras proved to be decisive for precision navigation. Concurrently, Artificial Intelligence is becoming hugely popular and recent advances in hardware capabilities allowed DL models to achieve unimaginable performances. CNNs are replacing conventional hand-engineered methods in almost every task, which induced us to do this investigation on combining these domains.

A complete framework to evaluate the performance of any feature detector for the task of estimating position and attitude of an spacecraft along a landing trajectory, using a sequence of images, acquired by a mounted camera, was implemented. Two classical feature detectors (Harris Corners and SIFT) and one using deep learning (SuperPoint) were used to perform evaluations on that framework.

Results have shown that the deep learning detector achieved at least the same navigation performance on all datasets, as the conventional methods, and that it even outperforms them under some conditions. SuperPoint proved to bring enormous advantages in detecting stronger keypoints and descriptors that remain being detected and correctly matched on much more images of the same scene along the video sequence. That is reflected on much better estimates of relative pose between much higher intervals between frames. This quality is of great importance as it can be used as complementary information to the relative pose between successive frames, in order to refine the estimates and reduce cumulative errors that arise from successive small errors from inter-frame predictions.

This problem is still an open field for investigation and the next planetary missions are expected to largely rely on the improvements that could be made by the scientific community until then. The first, and probably most obvious, future work is to implement this solution in hardware. More research could be done to integrate geometric constraints and the camera pose estimation pipeline into the trainable network and, finally, another important aspect to improve is computational efficiency of the deep learning network. The VGG-style backbone has lots of standard 2D convolutional layers that could be converted to most efficient building blocks as the ones proposed by MobileNets.

References

[1] Daniel Ponsa Krystian Mikolajczyk Axel Barroso-Laguna, Edgar Riba. Key.net: Keypoint detection by handcrafted and learned cnn filters. *IEEE International Conference on Computer Vision*, 2019. 2

[2] Tomasz Malisiewicz Andrew Rabinovich Daniel DeTone. Superpoint: Self-supervised interest point detection and description. *CVPR Deep Learning for Visual SLAM Workshop*, 2018. 2, 4

[3] Edward Rosten Tom Drummond. Machine learning for high-speed corner detection. *European Conference on Computer Vision*, 2006. 2

[4] Reid Porter Tom Drummond Edward Rosten. Faster and better: A machine learning approach to corner detection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2010. 2

[5] Kurt Konolige Gary Bradski Ethan Rublee, Vincent Rabaud. Orb: An efficient alternative to sift or surf. *International Conference on Computer Vision*, 2011. 2

[6] Tinne Tuytelaars Luc Van Gool Herbert Bay. Surf: Speeded up robust features. *European Conference on Computer Vision - ECCV 2006*, 2006. 2

[7] Relja Arandjelović Josef Sivic Ignacio Rocco. Convolutional neural network architecture for geometric matching. *CVPR*, 2017. 4

[8] Martin Humenberger Philippe Weinzaepfel Jerome Revaud, Cesar De Souza. R2d2: Reliable and repeatable detector and descriptor. *33rd Conference on Neural Information Processing Systems*, 2019. 2

[9] Vincent Lepetit Pascal Fua1 Kwang Moo Yi, Eduard Trulls. Lift: Learned invariant feature transform. *European Conference on Computer Vision*, 2016. 2

[10] David G. Lowe. Object recognition from local scale-invariant features. *International Journal of Computer Vision*, 1999. 2

[11] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 2004. 2

[12] Christoph Strecha Pascal Fua Michael Calonder, Vincent Lepetit. Brief: Binary robust independent elementary features. *European Conference on Computer Vision*, 2010. 2

[13] Tomas Pajdla Marc Pollefeys Josef Sivic Akihiko Torii Torsten Sattler Mihai Dusmanu, Ignacio Rocco. D2-net: A trainable cnn for joint description and detection of local features. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019. 2

[14] Chris Harris Mike Stephens. A combined corner and edge detector. *Alvey Vision Conference 1988*, 1988. 2

[15] Jianbo Shi Carlo Tomasi. Good features to track. *IEEE Conference on Computer Vision and Pattern Recognition*, 1994. 2

[16] Pascal Fua Kwang Moo Yi Yuki Ono, Eduard Trulls. Lf-net: Learning local features from images. *32nd Conference on Neural Information Processing Systems*, 2018. 2

[17] Richard Hartley Andrew Zisserman. *Multiple View Geometry*. Cambridge University Press, 2^{nd} edition, 2004. 3