Research Project 1: Final Report

A final report written as part of an undergraduate Research project under the supervision of Prof. Pini Gurfil.

Guy Sassy

Department of Aerospace Engineering

Technion – Israel Institute of Technology

1. Introduction and overview

Lunar landing is a challenging objective. This research is aimed at establishing a new method for accomplishing this goal by vision aided navigation. Vision aided navigation (VAN) can use a camera or a set of cameras in order to estimate the position and orientation (pose) of the spacecraft. Once the pose of the vehicle is known with a sufficiently low uncertainty, the actions needed in order to accomplish the landing sequence can be achieved (e.g. Belief Space Planning (BSP) in the Model Predictive Control (MPC) framework). VAN algorithms can vary from one camera set (Mono) algorithms to multiple cameras algorithms, where the information acquired from these implementations is different as well. Furthermore, the algorithms can vary from pose estimation of the vehicle to pose estimation of the vehicle and the environment (SLAM) which can contribute to a scenario where the environment is unknown a priori and loop closures are necessary.

2. Camera geometry

Camera geometry is vital for understanding the constraints and errors which the observation model includes. To model the camera, a pinhole camera model is used, which defines the object observed, the film and the barrier between them, where the pinhole is placed, as seen in Fig. 1.

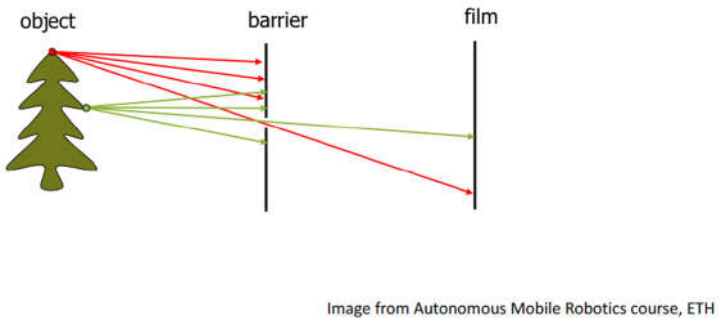

Figure 1: Pinhole camera model

This model yields in a specific projection called a perspective projection. Perspective projection preserves straight lines, which helps distinguishing between affine geometry and Euclidean geometry. For convenience, the image plane is in front of the camera center such that the image preserves the same orientation (i.e. not inverted).

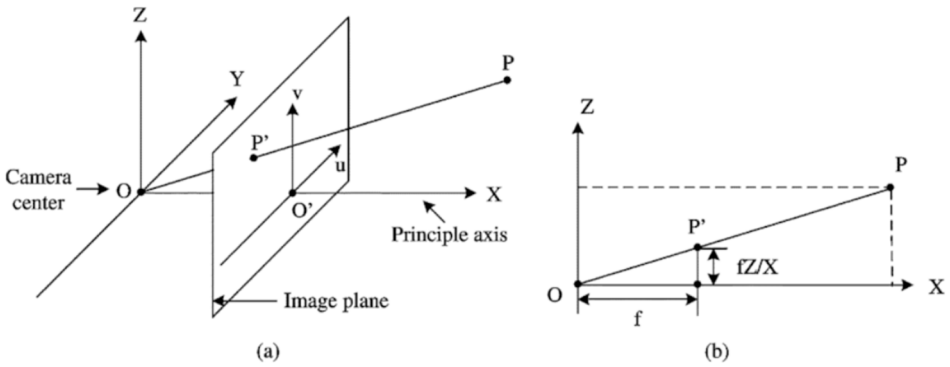

Figure 2: illustration of the pinhole camera model. (Taken from researchgate.net)

An illustration of the model is visualized in Figure 2(a), which denotes the principal point O' as the point of intersection between the image plane and the optical axis (noted as principle axis). Furthermore, a point captured by the camera with the global coordinates $P = (X, Y, Z)$ is projected into the image plane with image coordinates $P' = (u, v)$. In Figure 2(b), the focal length f is the distance on the principle axis between the camera and the camera plane. The distance of P' from the principle axis in the Z coordinate is $z = f \cdot Z/X$ and using the same for the Y coordinate yields $y = f \cdot Y/X$. This implies that the camera is observing the relative angles of an object and not the relative distance. This conclusion is vital, as the information gathered from an observation is not enough for three-dimensional pose extraction.

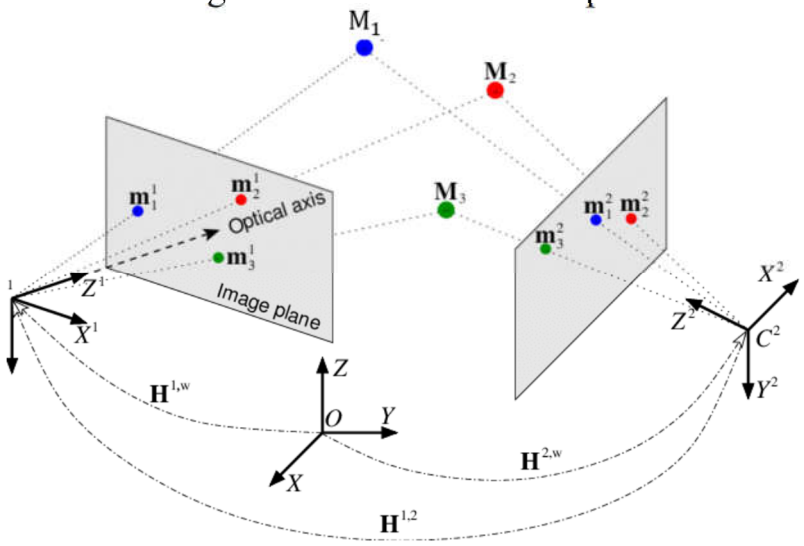

Figure 3: Observations acquired in two different poses. (Taken from researchgate.net)

Figure 3 depicts two observations C^j of the same landmarks M_i where $H^{1,2}$ is the Euclidean transformation matrix between the poses.

The projection equations of point M_1 are given by

$$\begin{pmatrix} \tilde{u}_1 \\ \tilde{v}_1 \\ \tilde{w}_1 \end{pmatrix} = \underbrace{\begin{bmatrix} f_x & s & u_0 \\ & f_y & v_0 \\ & & 1 \end{bmatrix}}_{K} \underbrace{[R_1 \quad \boldsymbol{t}_1]}_{H^{1,w}} \begin{pmatrix} X_1 \\ Y_1 \\ Z_1 \\ 1 \end{pmatrix} \quad (1)$$

$$\begin{pmatrix} \tilde{u}_2 \\ \tilde{v}_2 \\ \tilde{w}_2 \end{pmatrix} = \underbrace{\begin{bmatrix} f_x & s & u_0 \\ & f_y & v_0 \\ & & 1 \end{bmatrix}}_{K} \underbrace{[R_2 \quad \boldsymbol{t}_2]}_{H^{2,w}} \begin{pmatrix} X_1 \\ Y_1 \\ Z_1 \\ 1 \end{pmatrix} \quad (2)$$

where R_i is the rotation matrix and t_i is the translation from world frame to the i camera frame, $\begin{pmatrix} X_1 \\ Y_1 \\ Z_1 \end{pmatrix}$ are the point M_1 coordinates in world frame and the matrix K is the calibration matrix of the camera formed by the focal lengths f_x, f_y, the principal point coordinates in the image plane (u_0, v_0), and the skew s, which can be taken as 0. Note that the left hand side vector contains the *augmented* coordinates of the pixel, such that the original coordinates can be recovered by $u_i = \tilde{u}_i/\tilde{w}_i$, $v_i = \tilde{v}_i/\tilde{w}_i$. The main error source in this model is the reprojection error. The reprojection error is caused by an error in the pixel coordinates, as the resolution is finite and there is noise in the measurements. Note that this error will be embedded in every algorithm chosen, as the model will remain the same. The reprojection error can be written mathematically as

$$e = z - \pi(\boldsymbol{x}, \boldsymbol{l}) \quad (3)$$

where z is the observation (u, v) and $\pi(x, l)$ is the prediction of the measurement according to the pose of the agent x and the landmark l. The error can be visualized as shown in Figure 4.

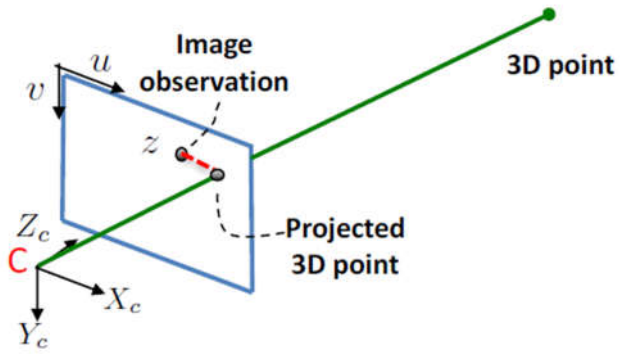

Figure 4: The reprojection error

3. Proposed algorithms

In order to understand which algorithm to implement, research in the VAN field is required. The algorithms will be introduced and compared by the information acquired and their relevancy to the implementation. Note that for all the introduced algorithms, a proper camera calibration is needed in order to calculate the pose of the agent.

3.1. Localization via known map

Localization of the agent can be done using a given map by using Eq. (1), where the vector of the landmark position is given. Therefore, (1) can be written as

$$K^{-1}\lambda \cdot \begin{pmatrix} u_1 \\ v_1 \\ 1 \end{pmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{pmatrix} X_1 \\ Y_1 \\ Z_1 \\ 1 \end{pmatrix} \quad (4)$$

The matrix needed to be calculated has 6 parameters but Eq. (4) has 2 constraints. Hence, by sampling more than 3 points the pose can be extracted. This algorithm is applicable as a map of the moon is available. The problem is to acquire the information needed in order to recognize the features sampled, as recognizing the features is essential in order to apply the given map coordinates into the algorithm. For instance, such recognitions can be done by machine learning algorithms and can be proposed as a future research.

3.2. Image based motion estimation

Motion estimation can be calculated using only one image by the multiple view geometry technique. Consider the scenario in Figure 3, where the two frames are obtained from the same camera. Matching the features obtained by both poses yields the information of corresponding 3D points of the environment, but not the points themselves, as the map is not given.

We introduce the multiple view geometry by using the following geometric definitions: **Baseline -** a line connecting the two poses to the camera centers. **Epipole -** point of intersection between the image plane and the baseline. **Epipolar Plane** - a plane that contains the baseline. **Epipolar line –** a line of intersection between the epipolar plane and the image plane. Figure 5 depicts these geometric definitions, where π is the Epipole plane and the lines $le, l'e'$ are the Epipole lines.

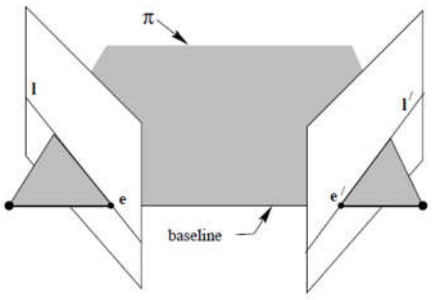

Figure 5: Multiple view geometry visualization. Taken from Ref [3].

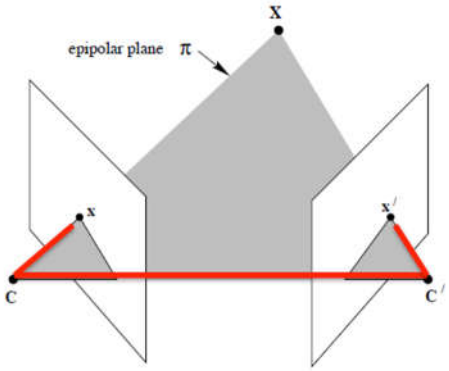

Figure 6: Epipolar constraint. Taken from Ref [3].

Constraining the baseline and the lines of sights of both poses to be on the epipolar plane as seen in Figure 6 is called the *epipolar constraint* and can be written as

$$\hat{\boldsymbol{x}}' \overbrace{[\boldsymbol{t}]_{\times} R}^{E} \hat{\boldsymbol{x}} = 0 = \boldsymbol{x}' \overbrace{(K^T)^{-1} E K^{-1}}^{F} \boldsymbol{x} \quad (5)$$

where $[t]_{\times}$ is the matrix cross-product equivalent, R is the rotation matrix and $\hat{x}, \hat{x}'$ are the lines of sight of the involved poses. Note that the line of sight can be written as $\hat{x} = K^{-1}x$ where x is the feature coordinates in the image plane. Therefore, by acquiring matched features the motion calculation can be done. The algorithm inputs are two images with n image correspondences (matched features) and the calibration matrix of the camera. The algorithm outputs are the motion between the two images, obtained by extracting the fundamental matrix F. The extraction of F requires another constraint to be introduced. This constraint is named the *singularity constraint* and can be written as:

$$det(F) = \mathbf{0} \quad (6)$$

Justifying Eq. (6) is done by looking at the epipole point. The epipole is in the left and right null space of F and therefore F is singular.

$F \in R^{3\times3}$ has seven degrees of freedom, as a 3x3 homogenous matrix has eight independent ratios (there are nine elements, and the common scaling is not significant), and the constraint $det\ F = 0$ reduces one more. Hence, by using the *normalized 8-point algorithm*, the fundamental matrix can be calculated. Once the fundamental matrix is acquired, the essential matrix E can be inferred as $E = K^T F K$, where the calibration matrix K is given. The translation can be extracted via SVD (e.g. left singular vector corresponds to the least singular value) as

$$\boldsymbol{t}^T \cdot E = \boldsymbol{t}^T \cdot [\boldsymbol{t}]_\times R = \mathbf{0} \quad (7)$$

After the translation is acquired the rotation matrix R can be calculated with 4 given solutions. The solution that will satisfy the required rotation is the one in which the landmark is in front of both cameras, as seen in Figure 7(b).

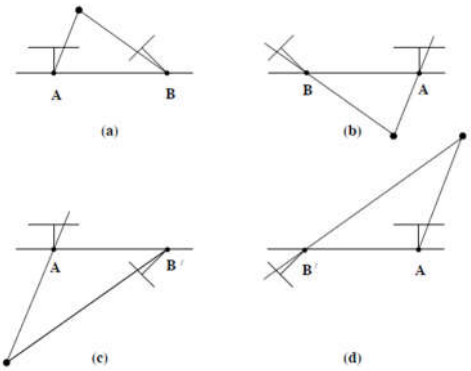

Figure 7: Four possible solutions for pose extraction. Taken from Ref [3].

In conclusion, given two images with 8 matched features and the calibration matrix of the camera, the motion between the two images can be recovered.

The main issue in image-based motion estimation with one camera is that the translation is calculated up to scale. The observations are done by a camera which observes only the angles (line of sight) of the object as noted in Section 2. This lack of information corresponds to the depth loss of the object seen by the camera. Another issue arises when accuracy is vital. Image-based motion estimation is not accurate as a stand-alone algorithm. The algebraic calculation of the fundamental matrix has errors embedded in it which cannot be filtered out once the matrix is acquired. Furthermore, the pose estimate is calculated in an odometry manner which results in dead reckoning effects.

3.3 Visual stereo odometry

Stereo cameras can solve some of the problems introduced in Section 3.2. Since the baseline between the two cameras is known by design, an algorithm which uses triangulation at each time step to infer the landmark position by Eq. (1) can be applied. Once the landmark poses are known, the problem is now solved by the algorithm given the known map in Section 3.1. This process is depicted in Figure 7. The known map at this "inner layer" of the algorithm – where the landmark points are given - is known with some uncertainty, as the reprojection error remains. Furthermore, the baseline between the cameras can reach a maximum length of several meters. This fact alone makes the algorithm impractical at an early stage landing sequence, as initially the height of the vehicle is larger than the baseline by several orders. Thus, the triangulation may not suffice for landmark pose estimation.

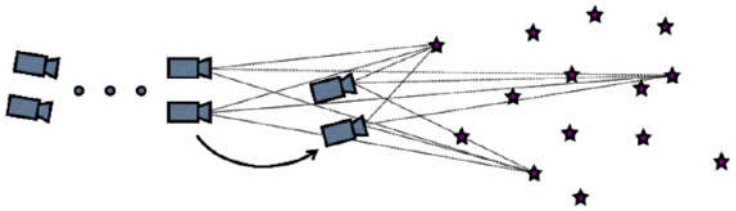

Figure 7: Stereo visual odometry. Image from Ref [1].

Reviewing the proposed algorithms yields the possible implementation of image-based motion estimation in early stages, noting the algorithm needs to be reinforced with distance measurement (e.g. distance from the moon surface). Later stages of the landing can be done by switching to a stereo camera system as the baseline length becomes relevant in lower heights, and, therefore, the precision of a stereo algorithm is high enough.

4. Implementation

This part of the research is focused at implementing the proposed algorithm of a single camera, to estimate the pose of an object from a sequence of photos in video format. The main concept is to extract the frames from the video in a chronological order and to estimate the motion done between the frames. Features extraction is done by Scale Invariant Feature Transform (SIFT) algorithm. SIFT algorithm extracts pixels from a frame which can be identified in other frames as well. This algorithm uses Difference of

Gaussians (DoG) as a tool to find scale space invariant extrema by filtering and blurring the image with different Gaussians as seen in Figure 8, subtracting them to find the different DoGs as seen in Figure 9. The process of filtering and blurring in different scales is made in different octaves which in this implementation is set to 3.

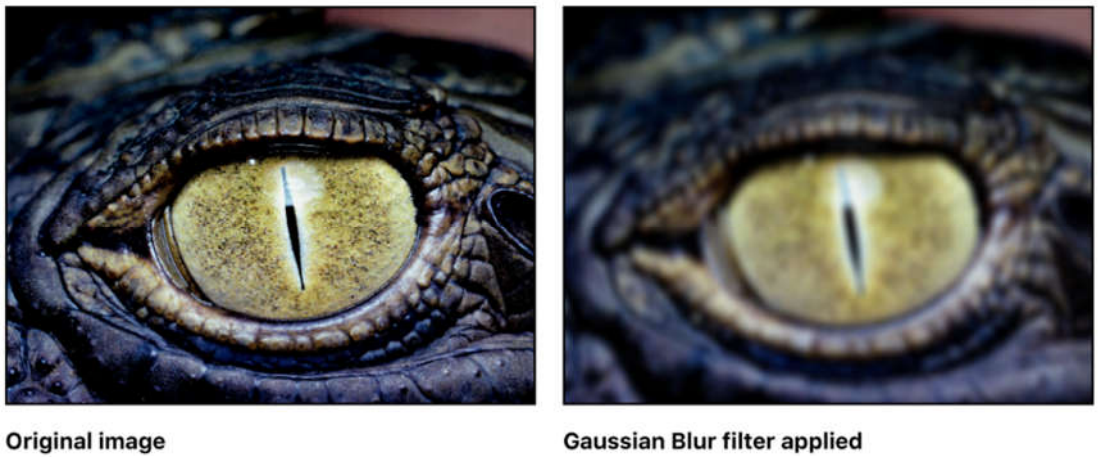

Figure 8: Gaussian Blur filter application. Image taken from apple.com

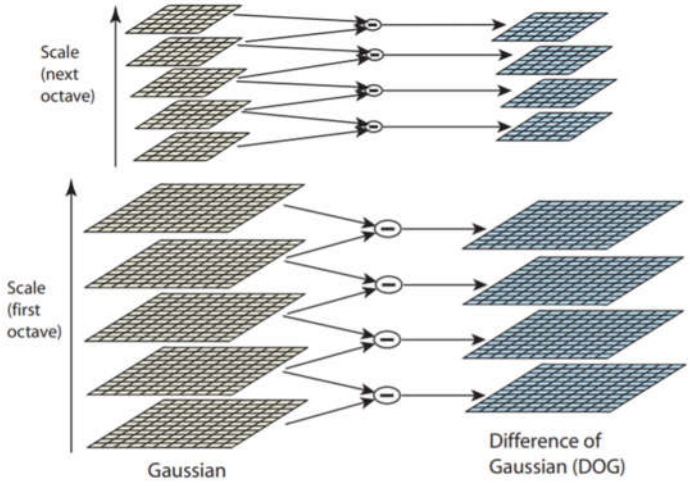

Figure 9: Image's convulsions with different Gaussians (Left), Difference of Gaussian (Right). Image taken from Ref [4].

Key point of an image (feature) is a point with large gradients of intensity in specific directions, so it can be recognized as a corner or a line. Calculating these gradients in different scales can filter out properties of the image itself, such as time of the picture being taken (day or night) and its orientation. Filtering out these properties makes these points comparable from different images, as they will be taken from different perspectives and light conditions. In order to compare different features, each feature has a vector of values called descriptor. The descriptor vector is made of 128 values, which are calculated from 16 squares around the pixel with 8 gradient orientations for each square. By calculating descriptor

norms between different images (i.e. Euclidean distance) the comparison is executed. In this implementation an open code called "vlfeat" was used in matlab environment to detect and match the features using 3 levels of Gaussians octaves. The feature matching algorithm output is shown in Figure 10.

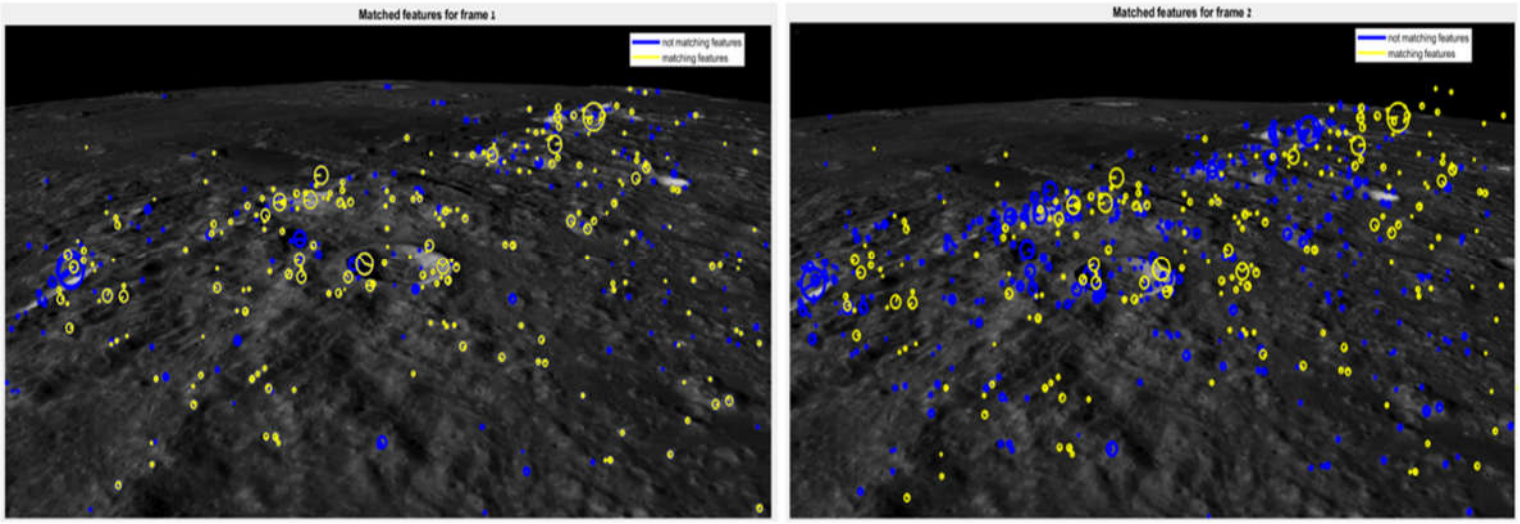

Figure 10: Matched features taken from two frames of the moon surface

Note that the matched features acquired from the algorithm are not completely accurate, as they include outliers. In order to tackle this problem an algorithm called RANSAC (Random Sample Consensus) is used. RANSAC is an algorithm which fits a model to a sample data set and checks which points in the data set fits the model (inliers). The model with the most inliers is the chosen model, and the points that do not fit are assumed to be outliers. In this implementation the RANSAC algorithm works with maximum trials of model finding set to 1000 and maximum distance of the point check to the model of 10. The concept of RANSAC is demonstrated for fitting a line (model) to points (data set) and visualized in Figure 11. The output of the implementation is shown in Figure 12.

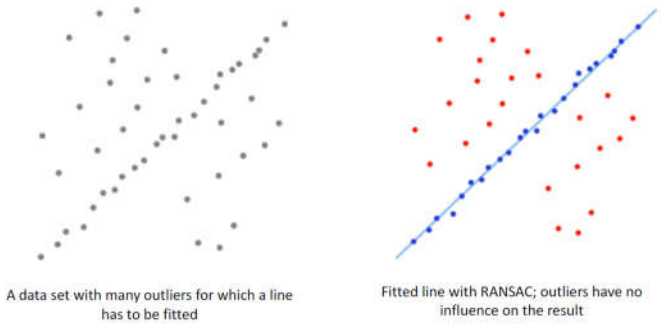

Figure 11: RANSAC fitted line to a data set. Taken from Wikipedia

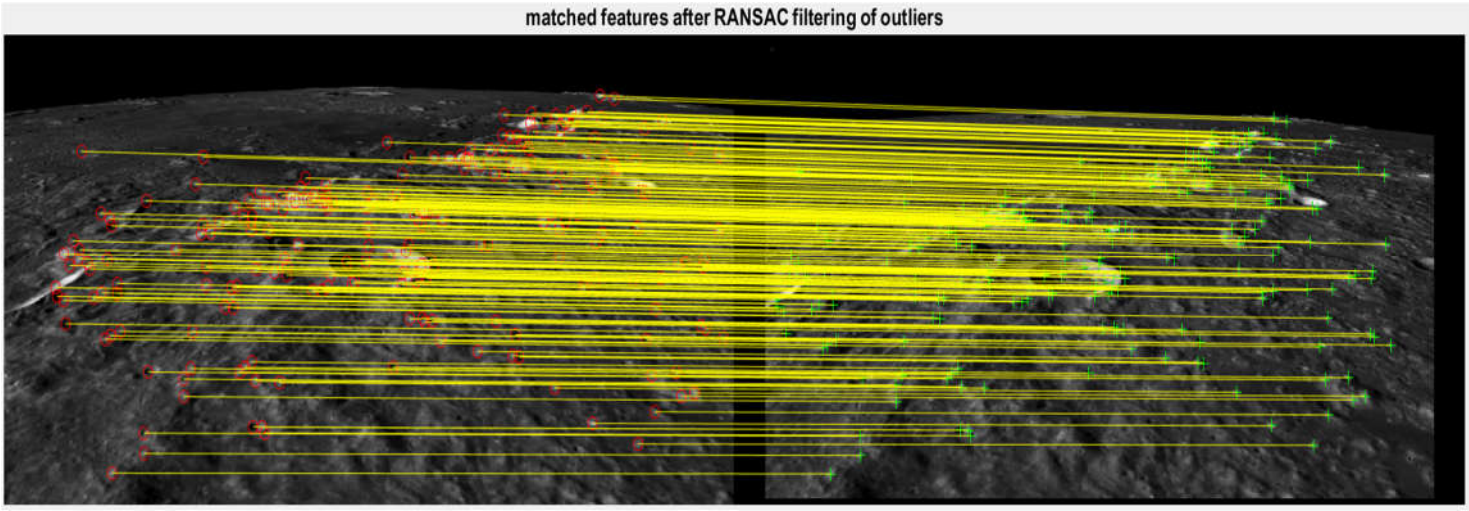

Figure 12: Matched feature of the frames from figure 9 after RANSAC filtering.

The matched features shown in Figure 12 are consistent in their direction, which implies that the pose calculation can be done with good precision.

The next step is to use the features that are left after the RANSAC filtering to calculate the fundamental matrix from the epipolar constraint as discussed in Section 3.2 and shown in Eqs. (5) and (6). In order to calculate the fundamental matrix the calibration matrix is needed. In order to test the algorithm in different scenarios my phone's camera was calibrated using the calibration toolbox from Caltech university and a chess board. This procedure is made by the algorithm discussed in Section 3.1 with different inputs and outputs. In this part, the algorithm is used to output the calibration matrix K given the 3D points and pixels as shown in Eq. (4). The procedure is demonstrated in Figures 13,14,15.

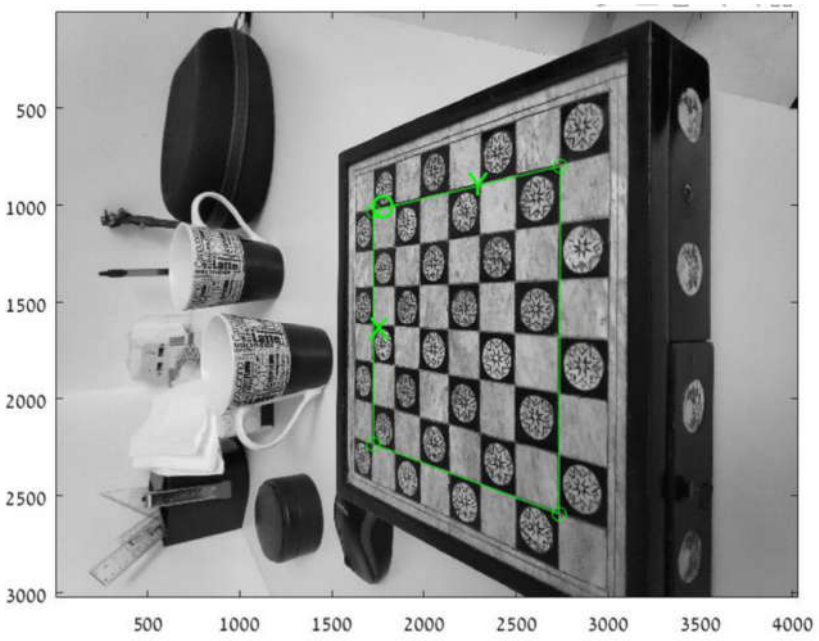

Figure 13: Chess board photo as seen in the calibration toolbox. The framework is marked with green line.

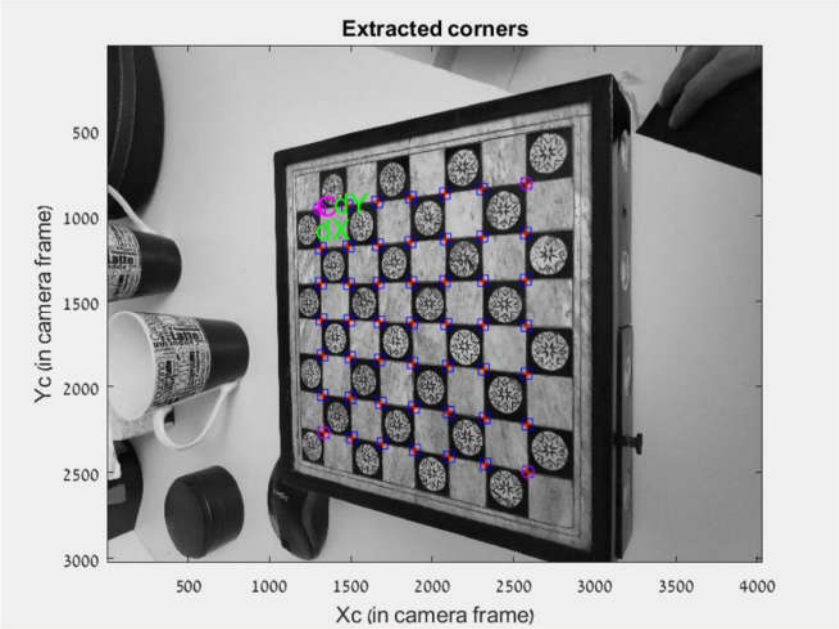

Figure 14: Corners are being marked by hand with [dx,dy] increments.

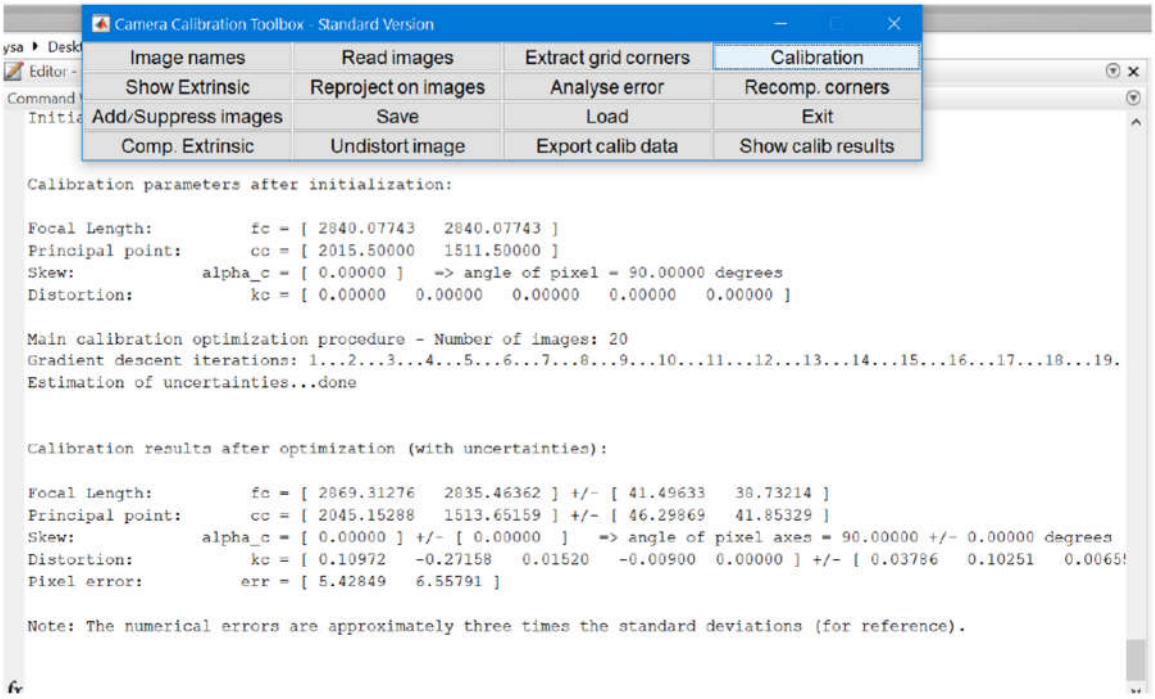

Figure 15: The final step of calibration matrix extraction.

The calibration matrix is used to calculate the essential matrix E from the fundamental matrix F as seen in Eq. (5). The fundamental matrix is achieved first, as the camera's sample is the pixel coordinates of the feature. Once the essential matrix is calculated, the translation is calculated as shown in Eq. (7) and the rotation needs to be calculated with respect to Figure 7(b). In this implementation the computer vision toolbox is used to calculate the rotation and translation given the fundamental matrix, the calibration matrix and the matched features (after RANSAC filter) as input.

The output is the matrix R as the rotation matrix and the vector t as the translation matrix. The transformation matrix T is

$$T_1^2 = \begin{bmatrix} R_1^2 & \boldsymbol{t}_{2\to1}^2 \\ 0^T & 1 \end{bmatrix} \quad (8)$$

The inner layer of the algorithm is now complete with the result of the transformation matrix T.

The outer layer of the algorithm achieves the inner layer output only when an indicator of the sample precision is valid. By default, the outer layer works with a constant increment of 0.3 seconds => 10 frames (30 Hz). The default increment can change due to bad precision of the current sample which is noted by the inner layer. The precision quality of the sample is determined inside the inner layer and a notation variable is sent as an additional output to the outer layer (i.e. the indicator). In case the precision is not good enough, the outer layer does not include the sample, but instead it increases the increment between the frames while the first frame is fixed, to acquire a new sample. The first frame must be fixed as the estimation of the pose is done in an odometry matter, and therefore, must be relative to the last sample. If the increment passes a threshold the process is aborted due to low chances of obtaining enough matching features (40 frames in this implementation).

5. Experiments

Experiments were done in order to test the algorithm in three different scenarios: Circular motion around an object with noticeable texture, straight line motion in a corridor with repeating patterns in the environment, and random walk which includes turns and straight line motions. The experiments were done in a constant height and with my phone's camera. Note that the orientation of the body (phone) is marked with a cartesian frame colored by red, green and blue.

5.1 Results

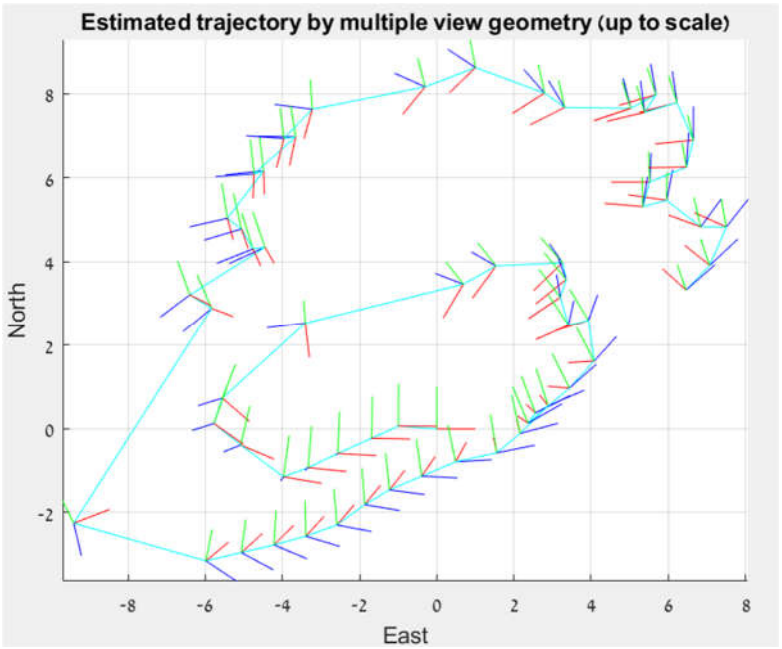

Figure 16: Circular motion estimation. view from above.

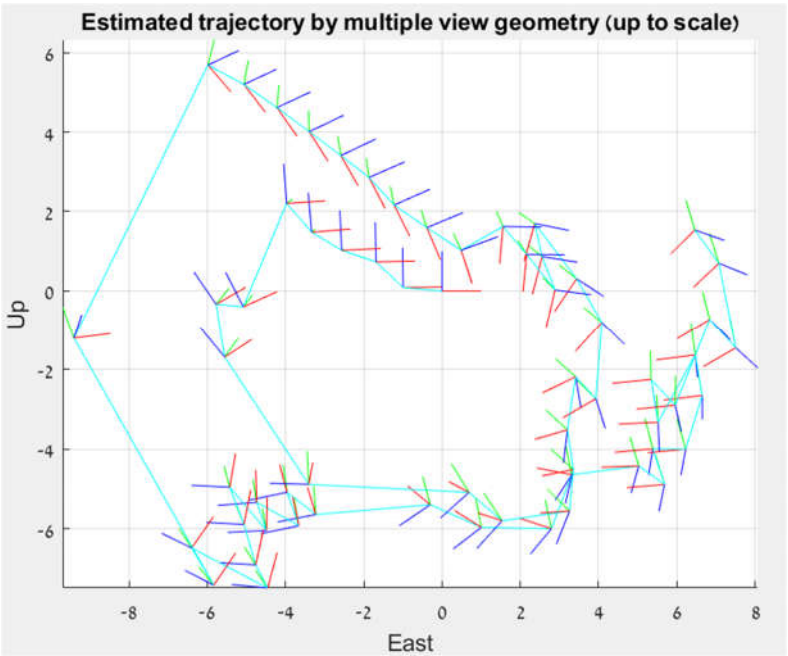

Figure 17: Circular motion estimation. up - east view.

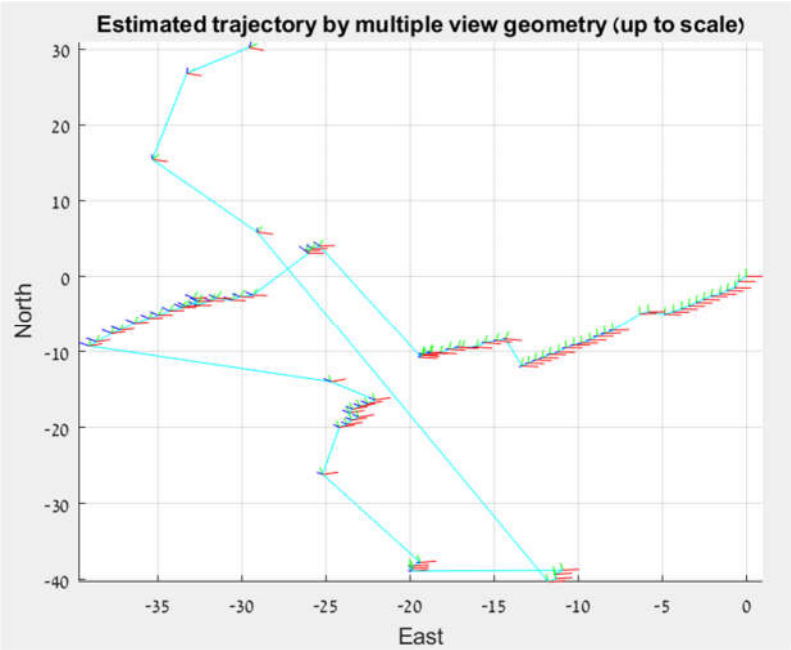

Figure 18: Corridor walk. view from above.

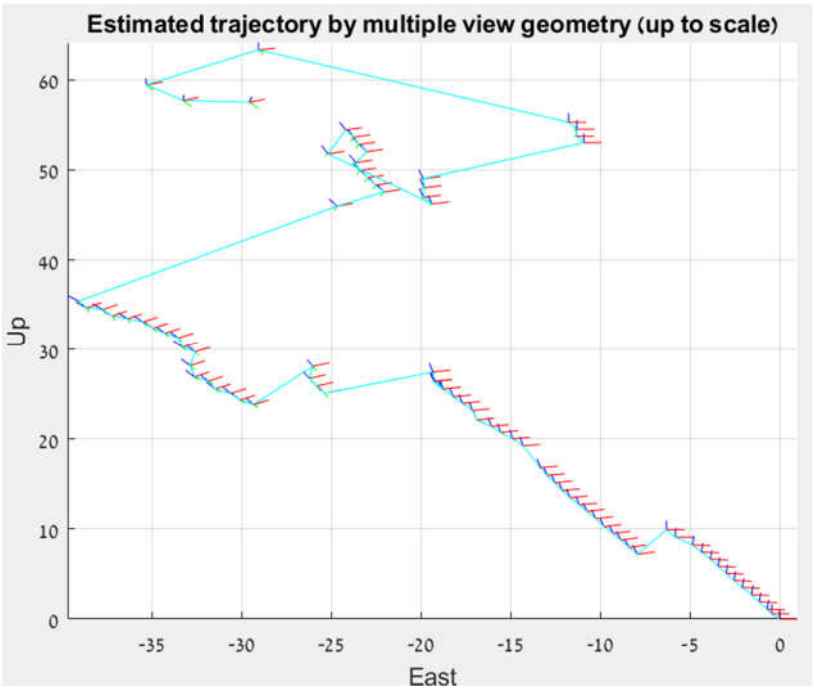

Figure 19: Corridor walk. up - east view.

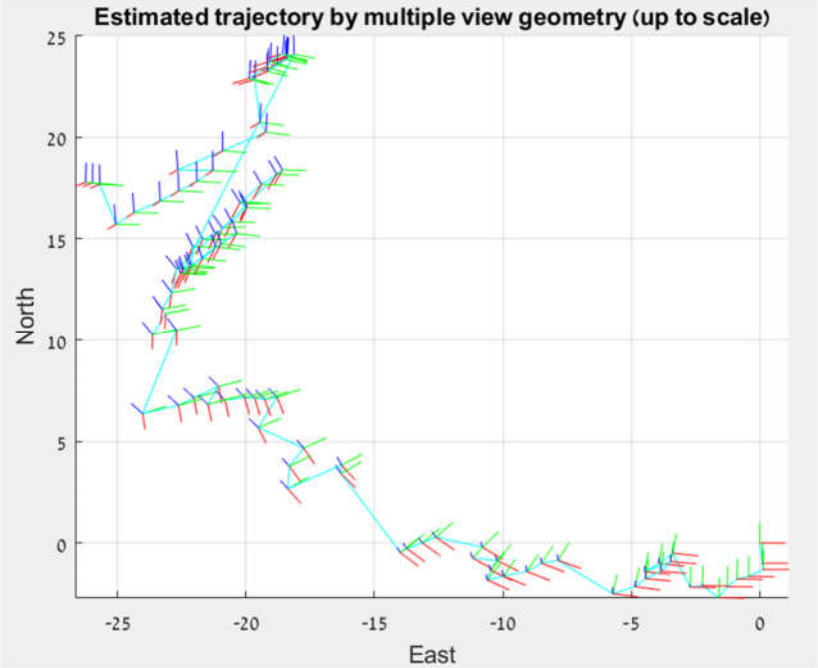

Figure 20: Random walk. view from above.

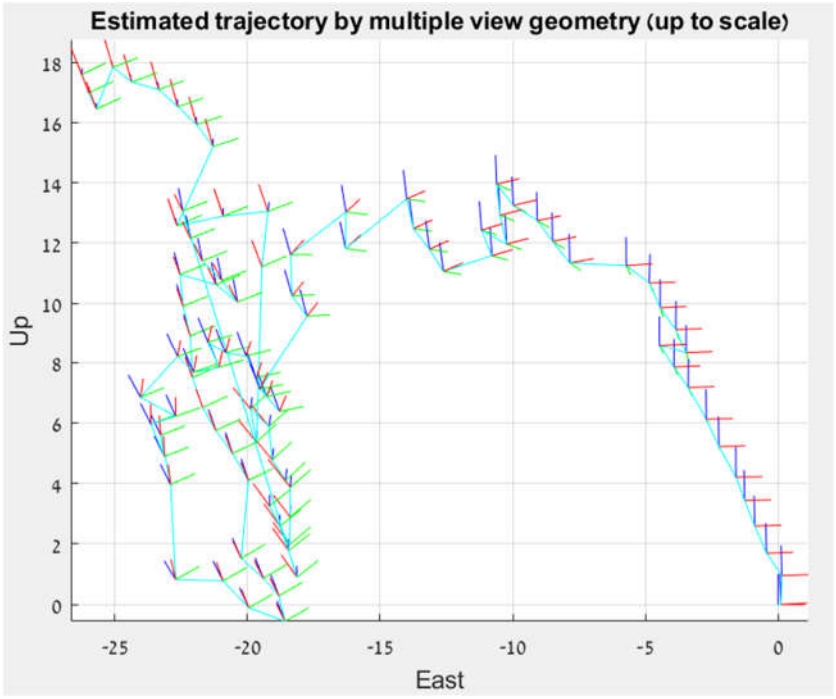

Figure 21: Random walk. up – east view.

5.2 Results analysis

First, the trajectory is estimated up to scale, and therefore, the spatial numbers have no meaning and cannot contribute to the analysis of the results. Nevertheless, trends in the spatial numbers can contribute to the consistency and error analysis. Second, ground truth trajectory of the tests is not available. Therefore, the tests were chosen in a way

that the analysis is still possible by spotting bias errors and random noises.

Looking at the circular motion estimation in Figures 16,17, the estimated trajectory seems noisy but consistent. The movement is estimated as a spiral from the north-east view (Figure 16), which implies to some bias error and a drift due to the odometry estimation. The trajectory from the up-east perspective (Figure 17) is circular but inconsistent and noisy, while the up axis numbers should remain at zero, as all the tests were done in constant height (approximately). Both views have estimation jumps in random directions which are due to noise.

The corridor walk motion estimation is consistent in the south-west direction except for some random jumps (Figure 18), noting that the overall direction after the random jumps, as the estimation is relative. The up-east view looks consistent as well except for some random jumps. The "up" direction seems biased in a constant error that cause drift. This effect of drift in the up direction had no influence on the circular motion and seen in this test for the first time.

The random walk motion estimation is noisy with overall movement in the "west" direction. The noise here seems more dominant, but cannot be analyzed due to the scale being unavailable. In this test, similarly to the corridor walk test, the "up" direction is biased and drifts with time.

In conclusion of the results, the algorithm works in a consistent and systematic operation but includes many errors due to bias and random noise. Bias error might occur due to imperfect calibration of the camera or from numerical errors in the fundamental matrix calculation, as homogeneity cannot be precisely written which makes the calculations being done in a least square sense. The random noise errors might occur due to the reprojection error described in Eq. (3), and due to the outliers filtering done by the RANSAC. These random errors can theoretically be modeled as Gaussians with proper first and second moments and summed with the projection model to define the random process of the observation z as $z = \pi(\boldsymbol{x}, \boldsymbol{l}) + \nu$ where $\nu \sim N(\mu, \Sigma)$.

6. Future Work

The work of this research is a step towards lunar landing via vision aided navigation but there is still much work to be done in order to integrate the system towards accomplishing the goal. The work can be divided into two main subjects. The future research and the future experiments.

6.1. Future research

This report tackled the measurement problem of an early stage landing where the late stage was not discussed. The algorithm proposed is using mono set of cameras because having a stereo set with a baseline in the scales of the vehicle is useless in the early stage of the landing (kilometers above the sampled features). As discussed, the algorithm estimates the trajectory up to scale, and, therefore, more sensors must be introduced for complete trajectory estimation (e.g. IMU, LIDAR). Therefore, integrating these sensors into the system is another topic for further research. Moreover, further research is necessary to tackle the late stage of the landing, as stereo cameras are relevant, but other alternatives might be better in the aspects of implementation and precision.

Note that this report framework is deterministic and should be expanded to the stochastic framework to increase the precision and accuracy drastically.

Moreover, this report offered a solution for the measurement model, but the motion model was not discussed. For instance, in order to design a Kalman filter, having the measurement model is important just as the prediction model, which is modeled by the dynamics of the vehicle. Therefore, future research in the field of the motion of the vehicle, both in the early stage and the late stage of the landing is vital for achieving a complete and accurate navigation model. Expanding the latter, in order to have a complete motion model, the control system must be considered as the inputs (e.g. thrusters) directly influence the motion model through the dynamics of the vehicle. (i.e. matrix B). Moreover, stepping into the control theory must consider the design of the system such that the system is stabilizable and detectable. (i.e. the unstable modes are controllable and observable respectively).

For final stage of the research, autonomous decision making can be implemented by the Model Predictive Control (MPC) framework by planning in the Belief Space Planning (BSP) domain. Optimizing the relevant costs can yield optimized trajectories and the landing can be robust to any unexpected disturbances and events. In my opinion, implementing an autonomous decision making algorithm is a true breakthrough and is not far from being done, given all of the above are completed.

6.2. Future experiments

The experiments done in this report were basic and their goal was to check the consistency and the operation of the algorithm in different basic scenarios. Future experiments should include tests of the robustness of the algorithm, ground truth trajectory for proper comparison and with available scale, a proper error estimation. Moreover, with future research being done as noted above, relevant experiments should take place such as simulations, landing in different zones with different initial conditions, robustness analysis, different algorithms and implementations comparisons in aspects of calculation time, feasibility of the solutions and more.

7. References

[1] Kurt Konolige and Motilal Agrawal. Frame SLAM: From Bundle Adjustment to Real-Time Visual Mapping, IEEE 2008.

[2] Roland Siegwart, Margarita Chli and Martin Rufli. Autonomous mobile robotics course, ETH Zurich, 2017

[3] Richard Hartley and Andrew Zisserman. Multiple View Geometry in computer vision (second edition). Cambridge university, 2004.

[4] David G. Lowe. Distinctive Image features from Scale-Invariant keypoints. International Journal of Computer Vision, 2004.

[5] Vadim Indelman. Vision Aided Navigation course. Technion institute, 2019.

[6] Jean-Yves Bouguet. Camera calibration tool for MATLAB. Caltech university, 2015.