POSE ESTIMATION OF UNCOOPERATIVE SPACECRAFT USING MONOCULAR VISION AND DEEP LEARNING

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF AERONAUTICS AND ASTRONAUTICS
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Sumant Sharma
August 2019

© 2019 by Sumant Sharma. All Rights Reserved.

Re-distributed by Stanford University under license with the author.

This work is licensed under a Creative Commons Attribution-Noncommercial 3.0 United States License.
http://creativecommons.org/licenses/by-nc/3.0/us/

This dissertation is online at: http://purl.stanford.edu/yx323yk4488

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Simone D'Amico, Primary Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Stephen Rock

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Mac Schwager

Approved for the Stanford University Committee on Graduate Studies.

Patricia J. Gumport, Vice Provost for Graduate Education

This signature page was generated electronically upon submission of this dissertation in electronic format. An original signed hard copy of the signature page is on file in University Archives.

Abstract

This dissertation addresses the design and validation of new pose estimation algorithms for spaceborne vision-based navigation in close proximity to known uncooperative spacecraft. The onboard estimation of the pose, i.e., relative position and attitude, is a key enabling technology for future on-orbit servicing and debris removal missions. The use of vision-based sensors for pose estimation is particularly attractive due to their low volumetric and power requirements, particularly in comparison to sensors such as LiDAR. Past demonstrations of this technology have relied on various combinations of cooperative use of fiducial markers on the target spacecraft, frequent inputs from the ground control stations, and post-processing of the images on-ground. This research overcomes these limitations by developing novel pose estimation methods that take as input a single two-dimensional image and a simple three-dimensional (3D) wireframe model of the target. These methods estimate the pose without requiring any a-priori pose information or long initialization phases. The first pose estimation method relies on a novel feature detection algorithm based on the filtering of the weak image gradients to identify the edges of the target spacecraft in the image, even in the presence of the Earth in the background. As compared with state-of-the-art feature detection-based methods, this method is shown to be more accurate and computationally faster through experiments on flight imagery. The second pose estimation method leverages modern learning-based algorithms by using a convolutional neural network architecture to classify the pose of the target spacecraft in the image. As compared to the feature detection-based methods, this method is shown to be more robust and two orders of magnitude computationally faster during inference using experiments on synthetic imagery. The third pose estimation method, the Spacecraft Pose Network (SPN), combines the higher accuracy

potential of feature detection-based methods with the higher robustness and computational efficiency of learning-based methods. The SPN method achieves this by integrating a convolutional neural network with a Gauss-Newton algorithm. The use of a convolutional neural network allows SPN to implicitly perform feature detection without the need for intensive manual tuning of hyperparameters. The use of a Gauss-Newton algorithm allows the direct application of the underlying physics of the pose estimation problem, i.e., the perspective equations for quantifying the uncertainty in the estimated pose. In contrast to current learning-based methods, the SPN method can be trained using solely synthetic images of a target spacecraft and is shown to generalize its performance on flight imagery of the same target spacecraft. This research also demonstrates that the SPN method can be used for target-in-target pose estimation to handle terminal stages of a docking scenario where only a partial view of the target spacecraft is available. A unique contribution of this research is the generation of the Spacecraft Pose Estimation Dataset (SPEED), which is used to train and evaluate the performance of pose estimation methods. SPEED consists of synthetic images created by fusing OpenGL-based renderings of a spacecraft 3D model with actual meteorological images of the Earth. SPEED also consists of actual camera images created using a seven degrees-of-freedom robotic arm, which positions and orients a vision-based sensor with respect to a full-scale mock-up of a spacecraft. SPEED is being used to host an international competition on pose estimation in collaboration with the European Space Agency. Infinite Orbits is adopting the pose estimation methods developed during this research for onboard deployment during their commercial on-orbit servicing missions.

Acknowledgment

I want to express my immense gratitude to my advisor and mentor, Prof. Simone D'Amico. This dissertation would not have been possible without his continuous support, patience, and critique.

Besides my advisor, I would like to thank the rest of my dissertation reading committee: Prof. Stephen Rock and Prof. Mac Schwager. In particular, conversations with Prof. Rock helped me be better at abstract thinking and not missing the forest for the trees. I want to thank Prof. Schwager for teaching me an excellent framework to evaluate research papers through his multi-robot control, communication, and sensing course. The framework has undoubtedly helped me be a better reader and researcher.

I want to thank my fellow lab-mates at the Space Rendezvous Lab for stimulating discussions regarding spacecraft formation flying research and the latest internet memes. Most importantly, I thank them for being my family away from home. In particular, I would like to thank Josh Sullivan, Duncan Eddy, Connor Beierle, Adam Koenig, Tommaso Guffanti, Lukas Steindorf, Ibrahim Makhadmi, Matthew Willis, Vince Giralo, Michelle Chernick, Corinne Lippe, Nathan Stacey, and Jeff Park. Special thanks to Jacopo Ventura for being an excellent collaborator, co-author, host, and friend.

Finally, I would like to acknowledge the financial support of King Abdulaziz City for Science and Technology (KACST) Center of Excellence for research in Aeronautics & Astronautics (CEAA) at Stanford University and NASA Ames Research Center for sponsoring this work.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Close range proximity operations between spacecraft has a rich history dating back to the Apollo program [3]. Since then examples of close-range proximity operations include the assembly and re-supply of the International Space Station and five on-orbit repair missions of the Hubble Space Telescope [4]. The close-range navigation in all these missions was made possible by the presence of onboard crew, cooperation, and active inter-spacecraft communication. Current and future generations of space robotics missions such as the RemoveDEBRIS mission by Surrey Space Centre [5], the Phoenix program by DARPA [6], and the Restore-L mission by NASA [7] also require close range navigation. These missions include technology demonstrators for applications such as repair, refuel, and de-orbiting of end-of-life and nonfunctional satellites. The main challenge with performing close-range navigation in actual on-orbit servicing and debris removal missions is that the target spacecraft may be uncooperative. Here, "uncooperative" implies that the target spacecraft may not be equipped with an active communication link or identifiable markers such as light-emitting diodes or corner cube reflectors for distance and attitude estimation. Further, ground-based estimates of the motion of the target spacecraft may be affected by significant uncertainties, and frequent inputs from the ground-station may not be possible for these missions. Therefore, both the relative position and orientation of the target spacecraft must be estimated onboard the servicer spacecraft using the onboard sensor suite.

The onboard estimation of the pose, i.e., the relative position and attitude, can be achieved relying only on sensors such as monocular vision or infrared cameras, stereo cameras, and Light Detection And Ranging (LiDAR). A selection of the sensor must be made considering the resources available onboard the servicer spacecraft in terms of power, mass, and computation. Also, the particular mission scenario and sustained future costs must be taken into account. This dissertation considers the use of monocular vision-based cameras. In contrast to LiDAR and stereo camera sensors, monocular sensors can ensure rapid pose determination with lower power and mass requirements [8]. Also, monocular sensors have lower hardware complexity and cost as compared to LiDAR and stereo sensors while having a more extensive operational range. However, the benefit of lower hardware complexity trades off with increased algorithmic complexity since a monocular sensor cannot provide direct measurements of the relative range. Moreover, monocular sensors can be less robust to variable illumination conditions typical of the space environment. For example, images captured by spaceborne monocular cameras exhibit high luminance contrast and low signal-to-noise ratio.

Overall this research is motivated by the scientific, commercial, and social benefits of performing on-orbit servicing and de-orbiting of inoperational spacecraft. In particular, the onboard and autonomous estimation of the pose of the target spacecraft is a critical technology to enable these missions. While using monocular sensors for pose estimation has its benefits, they are challenged by the aforementioned problems that need to be addressed using innovative algorithms. Therefore, this research is intended to explore state-of-the-art monocular vision-based pose estimation algorithms, propose new methods, and validate these methods using high-fidelity imagery.

1.2 Problem Statement

The primary objective of this research is to develop and validate methods that can estimate the pose of a target spacecraft using a single grayscale monocular image and a three-dimensional (3D) wireframe model of the target spacecraft. Note that the use of a single image is in contrast to methods relying on batches or sequences of multiple images, e.g., 3D reconstruction or Simultaneous Localization and Mapping (SLAM)

algorithms. Instead, this research focuses on the problem of pose estimation from a single image to avoid the reliance on favorable relative motion and long initialization phases that those algorithms typically require. This research aims to develop methods that can initialize a navigation filter to handle pose tracking in subsequent images. Similarly, the solution of the single image-based pose estimation can also be used as pseudo-measurements of the pose inside the navigation filter.

1.3 State-of-the-art and Limitations

1.3.1 Spaceborne Close-Proximity Relative Navigation

Several navigation systems for close-proximity operations employing spaceborne monocular cameras have been proposed to enable rapid pose estimation and tracking in close range [9, 10, 11, 12, 13, 14, 15, 16]. Here "close-range" typically implies an inter-spacecraft range of tens of meters to a few centimeters. Typically, these systems employ an image processing subsystem that identifies the visible target's features in the monocular image followed by a dedicated pose solver. The estimated pose is then used within a navigation filter. This routine is executed in closed-loop for pose tracking during the rendezvous maneuver. In general, the pose solver is an iterative algorithm that minimizes a specific fit error between the features detected in the image and the corresponding features of a known 3D model. All of the aforementioned navigation systems also require a-priori information of the pose to kick-start the pose estimation and tracking. In the presence of a poor initialization, these systems may converge to a local minimum, resulting in incorrect pose solutions. Existing work on pose initialization for spaceborne applications aims to directly apply terrestrial robotic navigation algorithms to the space imagery [17, 18, 19]. However, these techniques require the presence of known fiducial markers on the target spacecraft and/or rely on hand-tuned image processing parameters that are not robust to the rapidly varying illumination conditions typically present in spaceborne imagery.

1.3.2 Feature-Based Monocular Pose Estimation

All prior demonstrations of close-range pose estimation for spaceborne applications have typically utilized image processing based on hand-engineered features [1, 9, 17, 20, 12, 21, 22, 23, 24] and an a-priori knowledge of the pose [25, 26, 27, 28]. The problem of determining the pose of an object utilizing a monocular image and the object's 3D model has been extensively addressed in the research field of computer vision, typically for terrestrial applications. One of the first solutions to this problem is due to Dhome [29], who proposed a closed-form solution to solve for the pose given the correspondences between the edges detected in the image and the line segments of the 3D model. To solve for the correct correspondences, it follows an exhaustive predictive and verification step by matching all possible sets of 3D model line segments with three 2D image edges. To avoid an exhaustive search for the feature correspondences, authors have utilized soft-assign [30, 31]. However, its accuracy depends on the manual tuning of parameters, and it is slower than the Random Sample Consensus (RANSAC) method [32]. Following the work of Dhome, numerous algorithms for finding the object's pose from the image and model feature correspondences have been proposed [33, 34, 35, 36, 37]. While these algorithms do not require a-priori pose information, they produce a correct pose solution only if several correct feature correspondences are provided. Notably, this aspect limits the real-time application of these algorithms to pose initialization.

In regards to solving the problem of pose initialization for spacecraft applications, Kanani [19] and Petit [38] presented a pose initialization architecture based on a dedicated offline learning approach where a hierarchical model view graph using prototype views of the 3D model is built. The view graph is then explored during the online phase to find the prototype view whose contour corresponds the most with the detected contour in the image. This approach is inspired by template matching approaches where the input monocular image is searched for specific features and/or image sections which can be matched to an assigned template [39]. However, in the absence of an extensive database of pre-computed renderings of the target, these approaches tend to be very limiting as small changes in the target orientation and position may significantly affect its appearance. Grompone [11] proposed an initialization scheme that employs the Harris algorithm [40] for feature detection together

with a linear eight-point algorithm to solve for the pose. However, the scheme only provides relative distance information based on background subtraction and Gaussian blob detection algorithms. Tests on actual images of the Soyuz and Orbital Express spacecraft showed that the methodology is capable of correctly determining the Region of Interest (ROI) in the image. Finally, D'Amico [1] proposed to solve the pose initialization problem through the perceptual organization of the edges detected in the image using the Sobel and Hough algorithms [41]. Preliminary tests on actual images from the PRISMA mission [42] showed the capability to determine the precise relative position and attitude of the target vehicle without a-priori state information. However, tests also demonstrated two critical limitations of this technique: first, the initial pose is dependent on a computationally expensive iterative view-space discretization not suitable for real-time execution; second, the image processing lacks robustness to illumination and background conditions.

A key strength for many of the methods mentioned above is their use of the perspective transformation between the scene and the image to hypothesize and test feature correspondences detected in the 2D image and the known 3D model of target spacecraft. However, the formulation of specific features is not scalable to spacecraft of different structural and physical properties as well as not robust to the dynamic illumination conditions of space. Secondly, the a-priori knowledge of the pose of the target is not always available due to mission operations constraints nor desirable when full autonomy is required.

1.3.3 Deep Learning-Based Monocular Pose Estimation

Recent advancements in pose estimation techniques for terrestrial applications have relied on deep learning algorithms. Broadly, these algorithms bypass the classical image processing based pipeline and instead attempt to learn the non-linear transformation between the two-dimensional input image space and the six-dimensional output pose space in an end-to-end fashion. The deep learning-based methods either discretize the pose space and solve the resulting classification problem [43, 44, 45, 46] or directly regress the relative pose from the input image [47, 48, 49]. Render-for-CNN [44] demonstrated the use of rendered images to train a convolutional neural network for viewpoint estimation on actual camera images. The viewpoint estimation

problem is cast as classifying the camera rotation parameters into fine-grained bins. PoseCNN [49] used separate branches of a convolutional neural network to predict semantic labels, object centers in the 2D image, and object rotation using direct regression. However, the approach is not accurate enough without further refinement using the iterative closest point method [50]. Classification-based approaches rely on the fine discretization of the pose space into a large number of pose labels to achieve reasonable pose accuracy. On the other hand, direct regression-based approaches require a careful choice of parameters to avoid unpredictable behavior while learning the transformation between the input two-dimensional pixel information and the output regression parameters describing the six-dimensional pose space.

Recently proposed methods for terrestrial computer vision tasks such as object detection using deep learning have shown significant improvements in accuracy and robustness as compared to feature-based methods. However, the applicability of these methods to spaceborne pose estimation is not trivial. Firstly, unlike terrestrial applications, spaceborne navigation cameras are challenged with quickly varying illumination conditions and capture imagery with low signal-to-noise ratio and high contrast. Secondly, the end-to-end non-linear mapping between the inputs and the desired outputs make these algorithms opaque as compared to feature-based methods. This end-to-end mapping reduces the explainability of the output of these methods and makes the detection of failure cases challenging. Lastly, these methods require a large amount of training data containing actual imagery of the target objects annotated with precise pose data. Gathering a large amount of space imagery is extremely expensive and difficult to obtain. More recently, authors have proposed the use of synthetic imagery for deep learning-based object detection and pose estimation [51, 52, 53]. However, the proposed methods still require mixing in some real data with the synthetic data at the time of training and therefore are dependent on precise pose annotation. Further, the annotation of the pose for these massive datasets is generally performed by hand. More recently, automatic annotation methods using motion capture cameras or independent 3D scene reconstruction have been proposed. However, these independent sources must also be calibrated and can limit the distribution of camera and object poses sampled in the dataset.

1.4 Research Objectives

The primary objective of this research is to develop methods for pose estimation using a single monocular image and the 3D wireframe model of a target spacecraft. In particular, the primary objective can be broken down into the following research objectives.

1. To identify the gaps in the current state-of-the-art monocular vision-based pose estimation for spaceborne applications.

2. To develop a feature-based pose estimation method that is computationally faster and more robust than the current state-of-the-art in feature-based pose estimation.

3. To determine the feasibility of applying state-of-the-art deep learning methods for monocular vision-based pose estimation.

4. To generate large-scale image datasets that can be used to train deep learning-based methods for pose estimation.

5. To develop a pose estimation method that exploits the benefits of both feature-based and deep learning-based pose estimation methods to meet the aforementioned primary objective.

1.5 Contributions

The primary contribution of this research is the development of the Spacecraft Pose Network (SPN) method for pose estimation. This method combines several advancements in conventional feature-based and modern learning-based algorithms in order to provide pose solutions with high accuracy and robustness. The SPN method and the advancements that this research made along the way form four main contributions to the state-of-the-art in monocular pose estimation. These four contributions are articulated below along with references to publications where these were first introduced.

1.5.1 Comparative Assessment of Perspective-n-Point Solvers

A *systematic review contribution* through the functional and performance characterization of pose initialization techniques.

1. Reviewed state-of-the-art solutions of the Perspective-n-Point (PnP) problem.
2. Analyzed the performance sensitivity of the PnP solvers to the number of input features, sensor noise, number of outliers, and inter-spacecraft separation using Monte-Carlo simulations.
3. Generated recommendations on the relative advantages and disadvantages of the PnP solvers in the form of a decision matrix.

Documented in: Sharma S., DAmico S.; *Comparative Assessment of Techniques for Initial Pose Estimation using Monocular Vision*; Acta Astronautica (2016).

1.5.2 Weak Gradient Elimination and Feature Grouping for Pose Initialization

An *algorithmic contribution* through the development and validation of the Sharma-Ventura-D'Amico (SVD) method for pose initialization.

1. Developed the Weak Gradient Elimination (WGE), a new image processing technique for edge detection that ensures robustness to image features in the background.

2. Developed a new feature description technique based on feature grouping to dramatically reduce the search space of the 2D-3D feature correspondence problem.

3. Characterized the performance of the SVD method and compared it with state-of-the-art methods.

Documented in: Sharma S., Ventura, J., DAmico S.; *Robust Model-Based Monocular Pose Initialization for Noncooperative Spacecraft Rendezvous*; Journal of Spacecraft and Rockets (2018).

1.5.3 Generation of High Fidelity Spacecraft Imagery for Pose Estimation

An *experimental contribution* through the development of two complementary sources of high-fidelity monocular imagery that allow for training, validation, and verification of pose estimation algorithms.

1. Developed an augmented-reality image generation pipeline that fused an OpenGL based camera emulator with Earth imagery captured by meteorological satellites.

2. Calibrated and gathered imagery from the Testbed for Rendezvous and Optical Navigation (TRON) facility at the Space Rendezvous Laboratory.

3. Publicly released the resulting Spacecraft Pose Estimation Dataset (SPEED), the first large-scale image dataset that allows for training and testing of pose estimation methods.

4. Organized an international competition on spacecraft pose estimation in collaboration with the European Space Agency.

Documented in: Sharma S., D'Amico S.; *Pose Estimation for Non-Cooperative Rendezvous Using Neural Networks*; 2019 AAS/AIAA Astrodynamics Specialist Conference, Ka'anapali, Maui, HI, January 13-17 (2019).

1.5.4 Spacecraft Pose Network

An *algorithmic contribution* through the development, training, and validation of the Spacecraft Pose Network (SPN), a new deep learning-based method to perform pose estimation using a single monocular image.

1. Developed the SPN method that uses a convolutional neural network for bounding box detection and relative attitude estimation using a hybrid classification and regression strategy. The method then uses a Gauss-Newton algorithm based on the perspective equations for relative position estimation.

2. Developed an algorithm to quantify the uncertainty in the pose estimates of the SPN method.

3. Generalized the SPN method to rendezvous and docking scenarios specifically to handle inter-spacecraft separations where only a portion of the spacecraft is in the field of view.

4. Characterized and compared the performance of the SPN method versus other state-of-the-art methods using SPEED and flight imagery.

Documented in: Sharma S., D'Amico S.; *Pose Estimation for Non-Cooperative Rendezvous Using Neural Networks*; 2019 AAS/AIAA Astrodynamics Specialist Conference, Ka'anapali, Maui, HI, January 13-17 (2019).

1.6 Outline

This dissertation is divided into six chapters that cover distinct aspects of the design and validation of pose estimation methods.

Figure 1.1 shows a schematic outline of the main chapters in this dissertation. After this introduction, Chapter 2 presents the mathematical preliminaries underlying the problem of pose estimation using a single image. Next, Chapter 3 presents the development and validation of the Sharma-Ventura-D'Amico (SVD) method for pose estimation. Chapter 3 concludes with a performance comparison of this novel feature-based method against other state-of-the-art feature-based pose estimation methods.

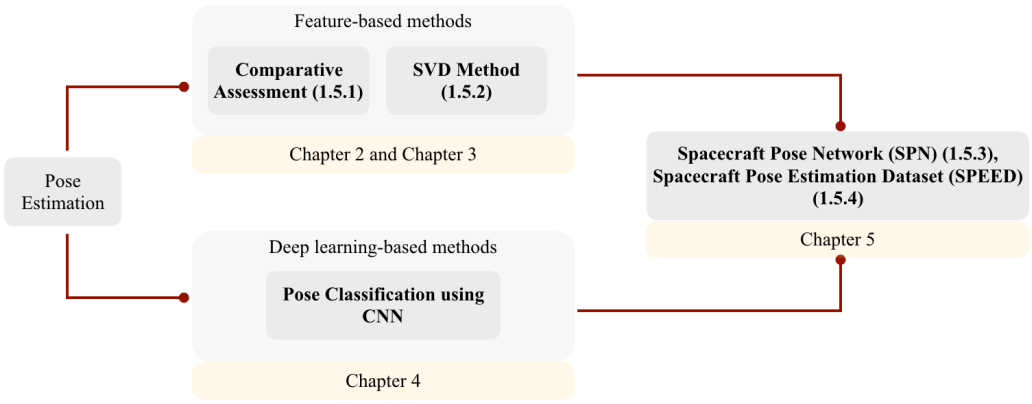


Figure 1.1: An outline of the main chapters in this dissertation. The section numbers in the parentheses point to the sections where a brief discussion of each of the contributions can be found.

Chapter 4 presents the use of a state-of-the-art convolutional neural network (CNN) to perform pose estimation using a classification methodology to explore the feasibility of using learning-based methods for pose estimation of spacecraft. Next, Chapter 5 presents the main contribution of this research with the integration of the ideas of Chapter 3 and Chapter 4 to develop the Spacecraft Pose Network (SPN), a novel pose estimation method. This chapter also details the generation of the first large-scale image dataset that can be used for training and testing pose estimation methods, the Spacecraft Pose Estimation Dataset (SPEED). Chapter 5 concludes with a performance characterization of the SPN method and comparison with the SVD method. Finally, Chapter 6 summarizes the results of this research and provides recommendations for further study.

Chapter 2

Mathematical Preliminaries

This chapter introduces some mathematical preliminaries underlying the problem of pose estimation using a single image. Particularly vocabulary and mathematical notation will be introduced throughout this chapter that will be used in the rest of the dissertation. Moreover, the mathematical details of the solutions of perspective equations and the training of convolutional neural networks are presented. These details are essential to understanding the Spacecraft Pose Network (SPN), which is the primary contribution of this research and will be introduced in Chapter 5. This chapter is organized as follows. Section 2.1 discusses the Perspective-n-Point (PnP) problem, its state-of-the-art solutions, and a comparative assessment of these solutions. Section 2.2 discusses the mathematical definition of convolutional neural networks and how these networks are trained.

2.1 Perspective-n-Point problem

An integral component of the feature-based pose estimation methods introduced in Section 1.3.2 is the Perspective-n-Point (PnP) problem. Recall that these pose estimation methods leverage the knowledge of the 3D model of the target spacecraft, the known intrinsics of the camera, and 2D features detected in the captured image using image processing techniques. The PnP problem connects these known quantities with the unknown relative pose between the target spacecraft and the camera using the underlying perspective geometry. Therefore, the performance of such pose

estimation methods hinges on the solution of the PnP problem. Typically, a PnP solver would be called multiple times to estimate the pose and would be subject to a wide variety of input from the image processing subsystem and the 3D model. Hence, a PnP solver should not only be fast and efficient but more importantly be reliable and robust to overcome challenges unique to monocular vision-based navigation in space. In the remainder of this section, a formal problem statement of the PnP problem is presented before reviewing state-of-the-art PnP solvers. A framework for the assessment of these solvers is then presented where a discussion of the simulation input generation, performance criteria, and test cases is detailed. Finally, relevant results from the assessment are presented with a discussion on the applicability of these solvers in the monocular vision-based navigation systems.

2.1.1 Review of Solution Methods

Let $\mathbf{q}_i^{\mathrm{B}} = [x_i \ y_i \ z_i]^T$, where $i = 1, 2, ..., n$, be n 3D model points in the body-fixed reference frame B. Let $\mathbf{p}_i = [u_i \ v_i \ 1]^T$, where $i = 1, 2, ..., n$, be the corresponding n image points in the image-fixed reference frame P. The PnP problem aims to retrieve the direction cosine matrix from frame B to the camera-fixed reference frame C, $\mathbf{R}^{\mathrm{BC}}$, and the translation vector from the origin of frame B to the origin of frame C, $\mathbf{t}^{\mathrm{C}}$.

$$\begin{aligned} \mathbf{r}^{\mathrm{C}} &= \left[x^{\mathrm{C}} \ y^{\mathrm{C}} \ z^{\mathrm{C}}\right]^T = \mathbf{R}^{\mathrm{BC}}\,\mathbf{q}^{\mathrm{B}} + \mathbf{t}^{\mathrm{C}} \\ \mathbf{p} &= \left[\tfrac{x^{\mathrm{C}}}{z^{\mathrm{C}}} f_x + C_x, \ \tfrac{y_{\mathrm{C}}}{z^{\mathrm{C}}} f_y + C_y\right], \end{aligned} \tag{2.1}$$

where $\mathbf{r}^{\mathrm{C}}$ represents the points of the target spacecraft expressed in camera frame C according to the current pose, $\mathbf{t}^{\mathrm{C}}$ and $\mathbf{R}^{\mathrm{BC}}$. Here f_x and f_y denote the focal lengths of the camera, (C_x, C_y) denotes the principal point of the image. Without loss of generality, it is assumed that the direction C_3 is pointed along the bore-sight of the camera and that the directions C_1 and C_2 are aligned with the directions P_1 and P_2 of the image frame, P, respectively.

Re-writing Equation 2.1 using homogeneous coordinates and representing camera parameters as a matrix K, a PnP solver estimates the 3×4 matrix P whose first

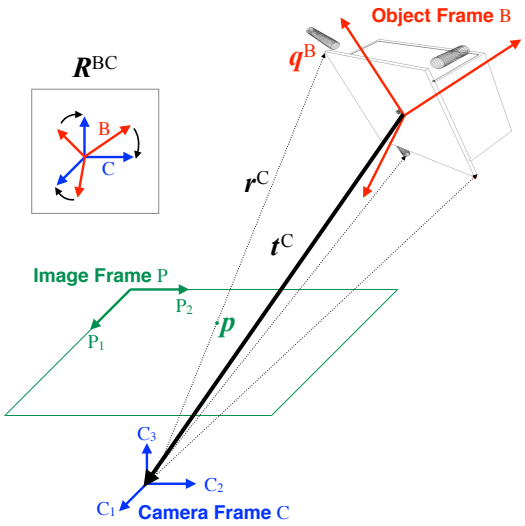


Figure 2.1: Geometric representation of the Perspective-n-Point problem.

three columns represent $\mathbf{R}^{\mathrm{BC}}$ and the fourth column represents $\mathbf{t}^{\mathrm{C}}$, i.e.,

$$\begin{bmatrix} w_i u_i \\ w_i v_i \\ w_i \end{bmatrix} = \begin{bmatrix} \\ K \\ \\ \end{bmatrix} \begin{bmatrix} \\ P \\ \\ \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ z_i \\ 1 \end{bmatrix}. \tag{2.2}$$

Without loss of generality, Equation (2.2) can be expanded to obtain Equation (2.3), assuming square pixels in the image sensor, zero skewness, and zero distortion, i.e.,

$$\begin{bmatrix} w_i u_i \\ w_i v_i \\ w_i \end{bmatrix} = \begin{bmatrix} f_x & 0 & 0 \\ 0 & f_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R}^{\mathrm{BC}}(1,1) & \mathbf{R}^{\mathrm{BC}}(1,2) & \mathbf{R}^{\mathrm{BC}}(1,3) & \mathbf{t}^{\mathrm{C}}(1) \\ \mathbf{R}^{\mathrm{BC}}(2,1) & \mathbf{R}^{\mathrm{BC}}(2,2) & \mathbf{R}^{\mathrm{BC}}(2,3) & \mathbf{t}^{\mathrm{C}}(2) \\ \mathbf{R}^{\mathrm{BC}}(3,1) & \mathbf{R}^{\mathrm{BC}}(3,2) & \mathbf{R}^{\mathrm{BC}}(3,3) & \mathbf{t}^{\mathrm{C}}(3) \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ z_i \\ 1 \end{bmatrix}. \tag{2.3}$$

Equation (2.3) has six unknown coefficients as there are six degrees of freedom in P. Three degrees are required to describe the relative attitude, and three are required to describe the relative position. Hence, three image points, i.e., $n = 3$, will provide six measurements that can be used to solve six equations in six unknowns. However,

there will be four possible solutions as the conversion to non-homogeneous coordinates is non-linear. In fact, it has been shown that a unique solution with a general configuration of points is only possible with $n = 6$ [54]. Note that even when a unique solution is theoretically available for the PnP problem, there is no guarantee that a unique solution exists in the case of symmetrical spacecraft geometries. For example, a PnP solver can provide six different solutions for a cube with six identical sides.

The PnP problem has been systematically investigated in literature and some of the most commonly used PnP solvers are PosIt and Coplanar PosIt [37], EPnP [55], OPnP [56], and Lu-Hager-Mjolsness method [57]. Note that these solvers assume that the correspondence between model points and observed image points has already been established. State-of-the-art PnP solvers usually take either an iterative minimization approach or a multi-stage analytical approach. The multi-stage analytical approaches estimate coordinates of some or all points in the camera frame and then solve the problem in 3D space using linear forms of the Equation 2.1. This is done to make computation fast for a large number of measurements at the expense of accuracy for a small number of measurements. Other solvers that utilize non-linear constraints posed by Equation 2.1 tend to be relatively more computationally expensive. The iterative minimization approaches minimize an error function defined in the image or object space while taking non-linear constraints of Equation 2.1 into account. Four of these state-of-the-art solvers, namely PosIt, Coplanar PosIt, EPnP, and the Newton-Raphson Method are discussed in more detail in the following sections to provide analytical reasoning behind their performance in the subsequent comparative assessment. PosIt, Coplanar PosIt, and the Newton-Raphson Method take an iterative minimization approach while EPnP takes a multi-stage analytical approach. Note that the assessment of the Lu-Hager-Mjolsness method [57] is excluded as it is also an iterative minimization approach, which is shown to be approximately four times slower than the Newton-Raphson Method [58].

PosIt and Coplanar PosIt

PosIt [37] requires a minimum of four non-coplanar 3D model points and corresponding 2D image points for pose calculation. It approximates true perspective projection with a Scaled Orthographic Projection (SOP) to form a coarse estimate of the pose

which is then iteratively improved until convergence to a solution. The algorithm for PosIt scales both sides of Equation (2.3) by $1/\mathbf{t}^{\mathrm{C}}(3)$. Then, expanding the first two rows yields the following relationship, where $s = f/\mathbf{t}^{\mathrm{C}}(3)$.

$$x_i \; s\mathbf{R}^{\mathrm{BC}}(1,1) + y_i \; s\mathbf{R}^{\mathrm{BC}}(1,2) + z_i \; s\mathbf{R}^{\mathrm{BC}}(1,3) + s\mathbf{t}^{\mathrm{C}}(1) = u_i \; \frac{w_i}{\mathbf{t}^{\mathrm{C}}(3)} \tag{2.4}$$

$$x_i \; s\mathbf{R}^{\mathrm{BC}}(2,1) + y_i \; s\mathbf{R}^{\mathrm{BC}}(2,2) + z_i \; s\mathbf{R}^{\mathrm{BC}}(2,3) + s\mathbf{t}^{\mathrm{C}}(2) = v_i \; \frac{w_i}{\mathbf{t}^{\mathrm{C}}(3)} \tag{2.5}$$

PosIt initializes with the SOP assumption, i.e., $\frac{w_i}{\mathbf{t}^{\mathrm{C}}(3)} = 1$, so that Equation (2.4) and Equation (2.5) can be simplified to linear equations with only eight unknowns, i.e., $s\mathbf{R}^{\mathrm{BC}}(1,1)$, $s\mathbf{R}^{\mathrm{BC}}(1,2)$, $s\mathbf{R}^{\mathrm{BC}}(1,3)$, $s\mathbf{t}^{\mathrm{C}}(1)$, $s\mathbf{R}^{\mathrm{BC}}(2,1)$, $s\mathbf{R}^{\mathrm{BC}}(2,2)$, $s\mathbf{R}^{\mathrm{BC}}(2,3)$, and $s\mathbf{t}^{\mathrm{C}}(2)$. With $n = 4$, these eight unknowns can be solved using the eight linear equations formed by writing Equation (2.4) and Equation (2.5) for $i = 1, 2, 3$, and 4. Then using the resulting estimate of the pose matrix, the value of $\frac{w_i}{\mathbf{t}^{\mathrm{C}}(3)}$ can be updated at the end of each iteration. For each iteration, the reprojected model points are calculated by using the current estimate of the pose matrix in Equation (2.3). At the end of each iteration, an improvement score, ΔE, is calculated by measuring the Euclidean distance between the reprojected model points of the current and previous iterations.

$$\Delta E = \sum_{i=1}^{n} f\sqrt{\Delta u_i^2 + \Delta v_i^2} \tag{2.6}$$

Iterations are stopped if either ΔE falls below 0.1 or 1000 iterations are reached. At low focal lengths or when the target spacecraft is close to the camera, SOP is not a valid approximation of true perspective projection and leads to inaccurate solutions. However, PosIt's "performance" (relative to other solvers) is expected to improve as SOP begins to closely approximate true perspective projection when the space resident object is far from the camera.

Coplanar PosIt [59] is a variation of PosIt which exclusively addresses the case of coplanar 3D model points and corresponding 2D image points. It acknowledges that an SOP approximation leads to two possible solutions of $\mathbf{R}^{\mathrm{BC}}$ due to an additional degree of freedom. This is addressed by keeping track of the solution with the lower reprojection error for a given iteration. The reprojection error is the average Euclidean distance between the measured image points, p_i, and the reprojected model points

obtained by using the estimate pose matrix in Equation (2.2). Similar to PosIt, the solutions are iteratively refined until ΔE of one of the branches falls below 0.1. A solution branch is discarded if it estimates any of the 3D model points to be behind the camera. For a given iteration, if the two possible solutions lie close to each other in the solution space, only refining the solution with the lower reprojection error does not guarantee convergence to a solution with the eventual lower reprojection error. This is a significant shortcoming of Coplanar PosIt and can only be remedied by sacrificing computational efficiency and tracking all possible solution branches.

EPnP

Applicable to both coplanar and non-coplanar 3D model points, EPnP [36] attempts to find a closed-form pose solution and requires a minimum of four corresponding model and image points. The main idea is to express the 3D model points as a weighted sum of four non-coplanar virtual control points, c_j, where $j = 1, 2, 3$, and 4, in the object frame B (see Fig. 2.1). Using a matrix inversion, the homogeneous barycentric coordinates, α_{ij}, can be computed from the n 3D model points, q_i, assuming arbitrary coordinates of the control points.

$$q_i = \sum_{j=1}^{4} \alpha_{ij} c_j \tag{2.7}$$

Using Equation (2.7), Equation (2.3) can be re-written in terms of the 12 unknown control point coordinates in the camera frame, $[\hat{\alpha_j},\ \hat{\beta_j},\ \hat{\gamma_j}]^T$, where $j = 1, 2, 3$, and 4.

$$\begin{bmatrix} w_i u_i \\ w_i v_i \\ w_i \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \sum_{j=1}^{4} \alpha_{ij} \begin{bmatrix} \hat{\alpha_j} \\ \hat{\beta_j} \\ \hat{\gamma_j} \end{bmatrix} \tag{2.8}$$

By substituting the values of w_i from the third row into the first two rows of Equation (2.8), two linear equations are formed for each corresponding pair of a 3D model

point and an image point.

$$\sum_{j=1}^{4} \alpha_{ij} f \hat{\alpha_j} - \alpha_{ij} u_i \hat{\gamma_j} = 0 \tag{2.9}$$

$$\sum_{j=1}^{4} f \alpha_{ij} \hat{\beta}_j - \alpha_{ij} v_i \hat{\gamma_j} = 0 \tag{2.10}$$

Hence, the $2n$ linear equations that result from writing Equation (2.9) and Equation (2.10) for $i = 1, 2, ..., n$, can be used to estimate the 12 unknowns in the form of a 12×1 vector, $\hat{x}$, by solving a linear system of the form $M\hat{x} = 0$. Note that M is a $2n \times 12$ matrix formed by arranging the coefficients of the $2n$ linear equations. When the image points are perfect data from a true perspective projection of the model points, the solution is calculated as a 12×1 vector, $\hat{x}$, in the null-space of the 12×12 matrix, $M^T M$. However, the matrix may have as many as four linearly dependent columns due to measurement noise and/or the focal length of the camera. There could be four possible solutions expressed as a linear combination of the eigenvectors of this matrix. Out of the four possible solutions, the one with the lowest reprojection error is selected. Typically, this is inaccurate when the focal length is large, and the matrix is poorly conditioned. In this scenario, even a small amount of measurement noise could lead to highly inaccurate estimates of the solution.

Newton-Raphson Method

The Newton-Raphson Method (NRM) [1] iteratively solves the true perspective equations, given by Equation (2.1), for an estimate of the pose. It requires an input guess of the pose and a minimum of three corresponding 3D model points and 2D image points. For each iteration, the reprojection error is linearized about the current pose estimate, $\vec{x} = [\mathbf{t}^{\mathrm{C}} \ \varphi_{BC}]^T$. Note that $\vec{x}$ is a 6×1 vector containing the three components of the translation vector, $\mathbf{t}^{\mathrm{C}}$ (see Fig. 2.1), and the three Euler angles, φ_{BC}, representing the rotation from frame B to frame C.

$$E_{RE,i} = \frac{\partial p_i}{\partial r_i} \frac{\partial r_i}{\partial \mathbf{t}^{\mathrm{C}}} \Delta \mathbf{t}^{\mathrm{C}} + \frac{\partial p_i}{\partial r_i} \frac{\partial r_i}{\partial \varphi_{BC}} \Delta \varphi_{BC} \tag{2.11}$$

The derivatives of the perspective equations with respect to $\mathbf{t}^{\mathrm{C}}$ and φBC are evaluated in the form of a $2n\times$ 6 Jacobian matrix, $J = [J_1, J_2, ..., J_n]^T$.

$$J_i = \begin{bmatrix} \frac{\partial p_i}{\partial r_i}\frac{\partial r_i}{\partial \mathbf{t}^{\mathrm{C}}} & \frac{\partial p_i}{\partial r_i}\frac{\partial r_i}{\partial \varphi_{BC}} \end{bmatrix} \tag{2.12}$$

For each iteration, the least squares solution by using the reprojection error from the previous iteration, E_{RE}, and the Jacobian matrix provides an update to the pose estimate, $\Delta\vec{x}$. Note that E_{RE} is a $2n\times1$ vector formed by writing Equation (2.11) for $i = 1, 2, ..., n$.

$$\Delta\vec{x} = (J^T J)^{-1} J^T E_{RE} \tag{2.13}$$

The pose estimate is refined until either $|\Delta\vec{x}|$ falls below 10^{-10} or 50 iterations are reached. A simulation with a synthetic 3D model and 2D image points was conducted to examine robustness to initialization with coarse pose estimates. For a fixed set of six 3D model points, the initial guess for $\mathbf{R}^{\mathrm{BC}}$ was varied 10^4 times to generate 10^4 corresponding sets of six image points. Each time the initial guess for $\mathbf{R}^{\mathrm{BC}}$ was generated from random values of the three Euler angles drawn from the uniform distributions of $[-180°, 180°]$, $[-180°, 180°]$, and $[0°, 180°]$. After computing a pose estimate for each corresponding set of six 3D model points and six image points, it was observed that given the true value of $\mathbf{t}^{\mathrm{C}}$, NRM typically converged to the correct solution when the initial guess for $\mathbf{R}^{\mathrm{BC}}$ was within 60° of the true value of $\mathbf{R}^{\mathrm{BC}}$.

2.1.2 Framework for Comparative Assessment

For pose initialization, PnP solvers will be called multiple times to find the correct correspondence hypothesis. Let S be the number of all possible correspondence hypotheses and equivalently, possible pose solutions. Given p points in the image plane and q points in the 3D model,

$$S = \binom{p}{n}\binom{q}{n} n!n!. \tag{2.14}$$

Therefore, to reduce the number of possible pose hypotheses, a pose initialization method is expected to use as small of a value for n as possible. Hence, small values of n are used in these simulations as inputs to the PnP solvers. Monte-Carlo simulations

are used to rigorously inspect the performance of the PnP solvers in the context of pose initialization. It is imperative to define performance criteria, test cases, and generate simulation input data that exhaustively represent scenarios of space-borne applications. The simulations made use of vectorized implementations of all PnP solvers and were carried out on a 2.4 GHz Intel Core i5 processor. The profiling results have been scaled appropriately to assess the potential implementation of these solvers on a state-of-the-art spaceborne microprocessor running at clock-rates of 30-300 MHz. Coplanar PosIt only accepts coplanar 3D model points while PosIt only accepts non-coplanar 3D model points. Therefore, for each simulation, the input's coplanarity is checked to select the more suitable of the two solvers. Results for this combination of solvers are labeled as PosIt+.

Simulation Input Generation

The three inputs of the PnP solvers are n 3D model points extracted from a wire-frame model, camera's intrinsic parameters, and n 2D image points extracted from a virtual image of the wire-frame model. For NRM, an input guess of the pose is also required. A randomly selected attitude within $\pm$ 60° of the true attitude and a random position guess with a magnitude within $\pm$ 30% of the magnitude of the true position are selected.

The 3D model needs to carry a high number of features to reduce ambiguities associated with the symmetry of spacecraft but also be simplified enough to boost the efficiency of the search algorithms during feature correspondence and pose estimation. With this consideration, a wire-frame model derived from a high fidelity CAD model of the Tango spacecraft from the PRISMA mission is used in this chapter (see Fig. 2.2). The model is input in MATLAB as a stereolithographic file from which information about surfaces and edges is generated. Duplicate edges, as well as edges on flat surfaces, are removed while the remaining edges are discretized as 3D points. Based on an empirically determined probability distribution, n 3D points are selected at random for each simulation and used as an input to the PnP solvers (see Fig. 2.3). Figure 2.2 shows the results of the 3D model reduction on a publicly available CAD model of the Rosetta spacecraft [60] to show the broad applicability of this method. However, note that the Rosetta spacecraft's 3D model was not used in the comparative

assessment of the PnP solvers. The model reduction procedure can potentially be used for pre-launch validation of pose estimation techniques when a CAD model of the target spacecraft is available, or during orbit when the 3D model of the target space resident object can be generated using Structure-from-Motion (SfM) techniques.

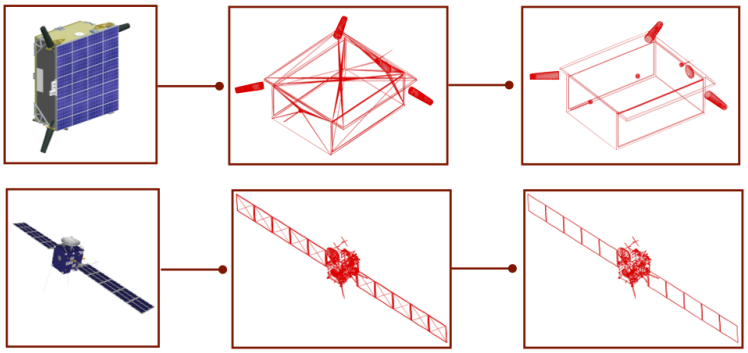


Figure 2.2: Typical model reduction steps to obtain simplified wire-frame model from CAD model of Tango spacecraft (top). Output of the same steps for Rosetta spacecraft (bottom).

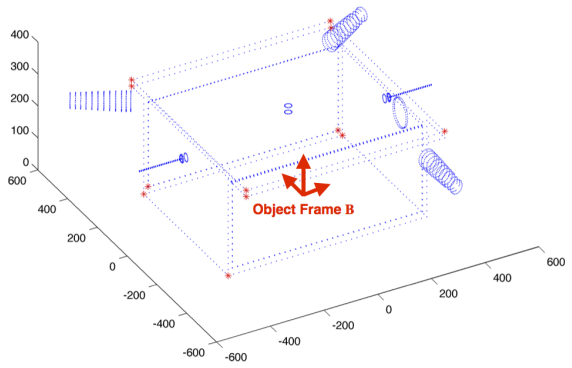


Figure 2.3: Sample simulation input of 3D model points (shown in red). The discretized 3D model (shown in blue) has been plotted as a reference.

A virtual pinhole camera model is adopted to model the close-range vision camera embarked on the servicer spacecraft of the PRISMA mission [1]. The focal length is $20{,}187{\cdot}10^{-6}$ m for both axes of the image sensor which produces images 752 px $\times$ 580

px in size. It is assumed that the images are produced with zero skewness and zero distortion. Origin of the camera frame C coincides with the origin of the image frame P, with the optical axis aligned with the z-axis. The internal calibration matrix of the camera is as follows

$$K = \begin{bmatrix} 2347 & 0 & 0 \\ 0 & 2432 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.15}$$

Typically, the camera will be calibrated on-ground but it can also be calibrated on-board by treating the intrinsic parameters as five additional unknowns of pose estimation.

Simulation input for n image points is obtained through true perspective projection (Eq. (2.1)) of the selected n 3D model points using the virtual camera (see Fig. 2.4).

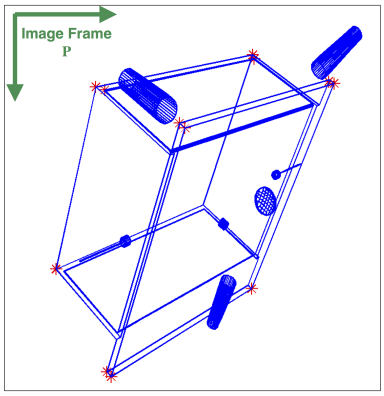


Figure 2.4: Sample simulation input of 2D image points (shown in red). The 3D model (shown in blue) has been plotted as a reference.

Performance Criteria

Four criteria are used to give a quantitative definition of the accuracy, speed, and robustness of the PnP solvers. These criteria are computational runtime of the solver, image plane error, translation error, and rotation error in the pose estimate output by the solver in comparison to the ground truth used to generate simulation input.

1. Computational Runtime: MATLAB command *tic* is used to start a stopwatch timer when a PnP solver is called whereas the command *toc* is used to stop the timer. Thus, the time elapsed on the timer is reported as the computational runtime to estimate the efficiency of the solvers relative to each other. This runtime is then used to provide a coarse first-order approximation of the runtime on a spaceborne microprocessor with a clock rate of 30 MHz. Since the pose estimation methods developed in this dissertation are intended to be implemented onboard a spacecraft microprocessor for real-time performance, the PnP solvers need to consume minimal computational resources.

2. Image Plane Error: Also referred to as reprojection error, it is defined as the mean Euclidean distance in pixels between input 2D image points p_i and the corresponding virtual 2D image points p_i' constructed from the 3D model points and the pose estimate from the PnP solver,

$$E_{2D} = \frac{1}{n}\sum_{i=1}^{n} |p_i - p_i'|. \tag{2.16}$$

3. Translation Error: As a measure of accuracy in the estimation of relative position, the translation error is defined as the percentage difference between the magnitudes of the true translation vector and the estimated translation vector,

$$E_T = \frac{|\vec{T}_{true}| - |\vec{T}_{est}|}{|\vec{T}_{true}|} \cdot 100. \tag{2.17}$$

4. Rotation Error: Unit quaternion representation of the true rotation matrix, q_{true}, and the estimated rotation matrix, q_{est}, is used to compute the rotation error. Quaternion algebra is used to compute a unit quaternion, q_{diff}, which represents the relative rotation between q_{true} and q_{est}. The rotation error is expressed as an equivalent angle and reported in degrees. Note that $q_{diff}(4)$ represents the scalar component of the unit quaternion,

$$E_R = 2\cos^{-1}(q_{diff}(4)). \tag{2.18}$$

Test Cases

Four test cases were developed to represent the range of possible input to PnP solvers. The main idea is to decouple the effects of the Earth's shadow, inter-spacecraft distance, background interaction, and measurement noise on the performance of the PnP solvers. Different levels of the Earth's shadow on the target spacecraft as well as the spacecraft orientation relative to the sun will vary the number of features detected through image processing. This effect is simulated by varying the number of feature correspondences, n, being passed as an input to the PnP solvers. Measurement noise due to the image sensor characteristics is simulated by adding random Gaussian noise with zero mean and a standard deviation, σ_P, to the simulation input of the image points being passed into the PnP solvers. Background interaction may lead to features being detected that are not present on the target spacecraft. This effect is simulated by changing the percentage of outlier image points, $\tilde{p}$. An outlier image point in the simulations is defined as an image point with a measurement noise greater than $5\sigma_P$.

It is required that the values of n, σ_P, and $\tilde{p}$ are representative of images taken during spaceborne applications of monocular vision to simulate these effects. A series of images captured by visual navigation cameras of the Orbital Express [61] and PRISMA [1] missions were referred to make these values as representative as possible. Edge detection followed by Hough transform [41] is performed to simulate the output of a state-of-the-art object detection subsystem. Typical results are shown in Figure 2.6 with intermediate steps shown in Figure 2.5. These results are used to empirically determine the range of n, σ_P, and $\tilde{p}$ used in the following test cases.

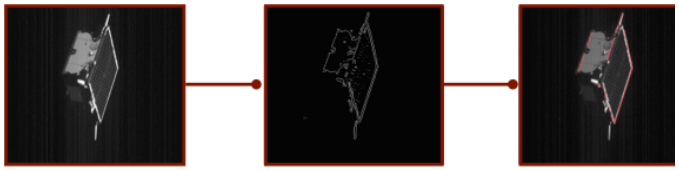


Figure 2.5: Intermediate steps of object detection subsystem based on edges: preprocessing (left), feature extraction (middle), feature description (right).

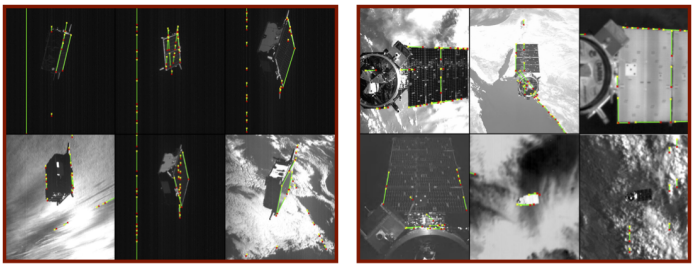


Figure 2.6: Typical output of an object detection subsystem overlaid on actual space imagery from the PRISMA mission (left) and Orbital Express mission (right).

1. Test Case 1: Number of feature correspondences. As the first test case, simulations with a varying number of feature correspondences from a minimum of three to a maximum of twelve correspondences were performed. For each value of n, 10^4 simulations are run, each with a different input set of 3D model points. The true pose is kept constant for each simulation in this test case to negate the effect of the geometry of the target spacecraft. Also, note that feature correspondences are perfect, i.e., the input set of the 3D model points is correctly mapped to the corresponding input set of the 2D image points.

2. Test Case 2: Pixel location noise. The 2D image points can be modeled as a sum of the true pixel locations and the pixel location noise. The pixel location noise can be modeled as a Gaussian distribution in two dimensions with a standard deviation of σ_P, which is shown to be proportional to image intensity noise σ_I for state-of-the-art object detection algorithms [62]. Since σ_I can be estimated through a principal component analysis of homogeneous grayscale image patches [63], the expected values of σ_P in a spaceborne application can also be estimated. Hence, as the second test case, 10^4 simulations were performed with varying values of pixel location noise. Noise is characterized as a Gaussian distribution with a mean of zero and a standard deviation of σ_P. This noise is added to the 2D image points generated from the true perspective projection of six 3D model points. The true pose is kept constant and feature correspondences are perfect for all simulations in this test case. For a fair comparison, the number of iterations for PosIt and NRM were limited so that the computational runtime is comparable to the closed-form solver EPnP.

3. Test Case 3: Outliers. Feature correspondence of input 3D model points and 2D image points is prone to contain outliers due to errors in the object detection subsystem or simply due to the geometry of the space resident. An object detection subsystem based on edges can output partial edges due to illumination conditions and/or false edges due to background interaction (see Fig. 2.6. Such an output from the object detection subsystem can result in an incorrect feature correspondence. Moreover, a probabilistic approach to feature correspondence is prone to errors even when object detection is perfect; for example, when the geometry of the space resident contains indistinguishable or symmetric surfaces. A robust PnP solver should negate the effects of incorrect feature correspondences (outliers) present in its input and produce an accurate pose estimate. Hence, as the third test case, 10^4 simulations with a varying number of outliers were performed. Outliers are 2D image points with a pixel location error of 10 px. This is in addition to the pixel location noise characterized by Gaussian distribution of a mean of zero and a standard deviation of 2 px. This noise is added to the 2D image points generated from the true perspective projection of twelve 3D model points. As in Test Case 2, the true pose is kept constant for all simulations and iterations for NRM and PosIt are limited.

4. Test Case 4: Distance along Optical Axis. With fixed camera characteristics, increasing the separation decreases the spread in the distribution of the 2D features on the image plane, making it difficult for the object detection subsystem to resolve features. Hence, as the fourth test case, 10^4 simulations with varying relative separation of the space resident object and the camera in the direction of the optical axis of the camera were performed. Pixel location noise is added to the 2D image points generated from the true perspective projection of six 3D model points. Noise is characterized by a Gaussian distribution with a mean of zero and a standard deviation of 2 px. The relative attitude is kept constant and feature correspondences are perfect for all simulations in this test case. As in Test Case 2, the number of iterations for NRM and PosIt are limited.

2.1.3 Results & Discussion

For each test case, mean and variance of the performance criteria for all simulations is represented as markers/lines and bars/shaded regions, respectively (see Fig. 2.7 and Fig. 2.8). Variance is reported as the interquartile range, i.e., the difference between the upper and lower quartile values of performance criteria.

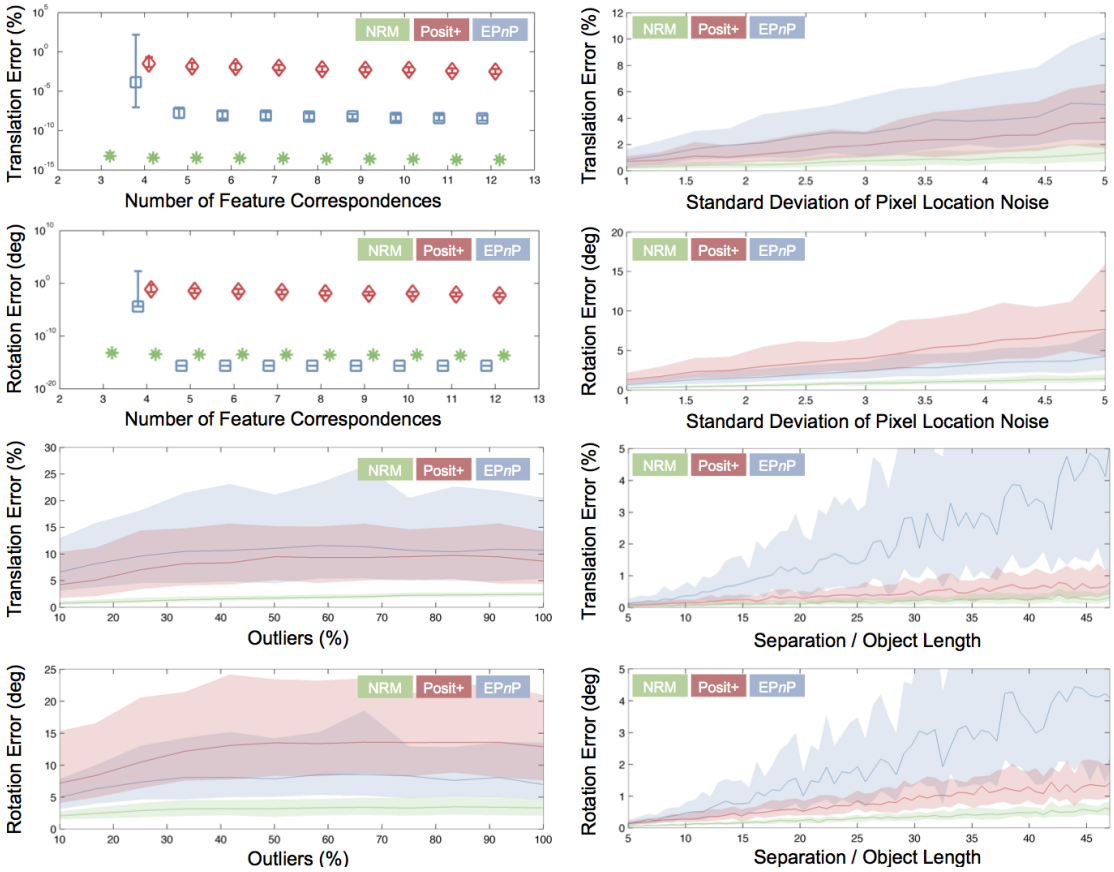


Figure 2.7: Results for translation error and rotation error from simulations of all test cases: number of feature correspondences (top left), pixel location noise (top right), outliers (bottom left), and distance along optical axis (bottom right).

Test Case 1: Number of Feature Correspondences

NRM is the most computationally expensive solver due to its use of least squares to invert the Jacobian matrix at each iteration. Typically, least-squares in NRM

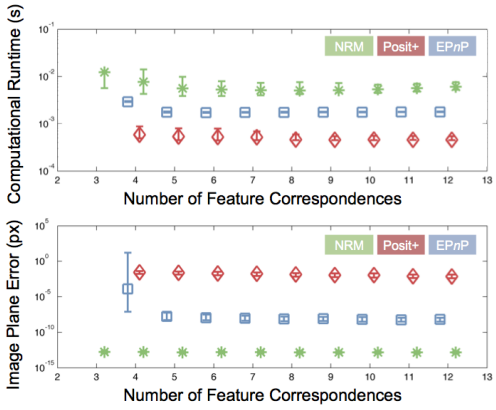


Figure 2.8: Results for computational runtime and image plane error from simulations of test case 1.

is an operation of $O(c^2n)$ complexity where c is a constant equal to 6, the number of unknowns for the PnP problem. This is highly expensive in comparison to an $O(c^3n)$ overall complexity of EPnP and $O(24n)$ overall complexity for PosIt. Note that for $3 \leq n \leq 6$, NRM requires fewer and fewer iterations to converge as n is increased. This leads to an overall decrease in computational runtime even though complexity per iteration increases. For this test case, PosIt has the highest errors in comparison to other solvers but it improves in accuracy for increasing values n. Even though EPnP provides a pose estimate for $n = 4$, it has the largest variance in the translation error and the rotation error at this value of n.

Test Case 2: Pixel Location Noise

With an increase in pixel location noise in the input 2D image points, all solvers exhibit a decrease in accuracy. NRM has the best performance due to its use of least squares, which is intentionally well suited to handle Gaussian noise. PosIt and EPnP have approximately the same variance in rotation error as well as translation error and exhibit a linear increase in errors with an increase in noise. However, unlike EPnP, which provides a closed-form solution, errors of PosIt can be reduced at the expense of computational runtime if more iterations are allowed. As with the first test case, PosIt's runtime is the lowest and NRM's runtime is the highest.

Test Case 3: Outliers

Recall that an outlier image point was earlier defined as an image point with a random pixel location error of greater than $5\sigma_P$, where σ_P is fixed at 2 px to characterize the measurement noise. All solvers exhibited a linear increase in errors when outliers made up less than 40% of the input 2D image points. For higher percentages, errors hold approximately constant values as the performance degrades to levels where additional outliers have no significant effect on accuracy. However, the slope of the increase in errors is proportional to the pixel location error used to generate the outliers. Comparatively, NRM has the lowest errors while PosIt and EPnP have similar levels of error. All solvers have sub-optimal performance in the context of initial pose estimation where feature correspondences are unknown, and input of 2D image points can contain high pixel location errors.

Test Case 4: Distance along Optical Axis

As the distance along the optical axis is increased, EPnP is the worst affected. One of the intermediate steps in EPnP is the calculation of a basis of the null space of a matrix containing all 2D image points. With increasing distance along the optical axis and apparent shrinkage of the image, this matrix tends to become sparse and null space estimation becomes increasingly challenging. However, as the distance is increased, SOP tends to be a better approximation of true perspective projection. Hence, PosIt, which is based on the SOP approximation, has lower rotation and translation error than EPnP. However, NRM has the lowest translation and rotation error as even at large separations as least-squares is well suited to handle noisy input of 2D image points.

Decision Matrix

To conclude the comparative assessment, a qualitative decision matrix is constructed (see Fig. 2.9). The decision matrix indicates PnP solvers on the rows and the test cases on the columns. Each cell is color-coded to represent the solvers' weighted sum of relative performance as measured by the four performance criteria. Recall that these performance criteria were earlier defined as the computational runtime,

the image plane error, the translation error, and the rotation error. Equal weights are given to the difference of each performance criteria from the mean value of the respective performance criteria, for all test cases and all PnP solvers. The terms "superior", "par", and "inferior" are used to represent this difference of performance criteria. The use of PosIt+ makes the PosIt solver applicable to both coplanar and

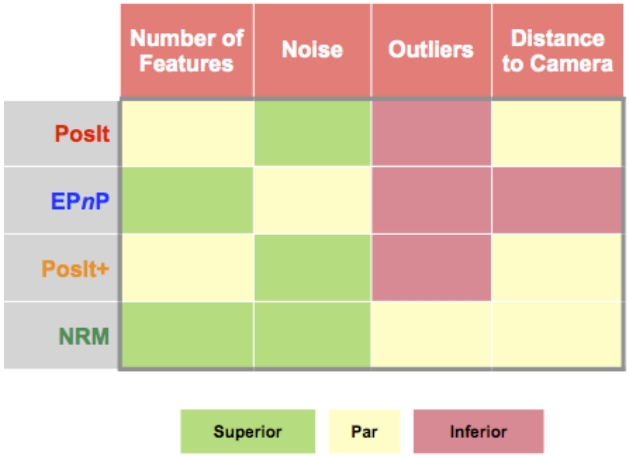


Figure 2.9: Comparative assessment results for simulations from all test cases as a qualitative decision matrix.

non-coplanar points without having a significant impact on computational runtime. EPnP has the best performance when pixel locations are free from noise or outliers. NRM has the lowest errors across all test cases. However, it requires an input guess of the pose and it also has an order of magnitude higher computational time across all test cases. In the presence of feature outliers, all solvers have sub-optimal levels of rotation and translation errors.

The results from the comparative assessment of the PnP solvers indicates that the strength of each PnP solver lies in different regimes and calls for a strategy to exploit the identified synergies. Accuracy of the PnP solvers is acceptable only when feature correspondence is perfect and is sub-optimal in the presence of feature outliers. Hence, an iterative statistical approach will be necessary to achieve pose convergence in real-world applications where multiple feature correspondence hypotheses need to be validated. For example, since EPnP has the lowest runtime, it can be used during

the first few iterations of pose estimation when a large number of correspondence hypotheses need to be validated. However, in later iterations, when the search space for correct feature correspondence has been reduced to a few ambiguous hypotheses, NRM can be used due to its better accuracy in the presence of outliers. If the pose estimate is still ambiguous, the method could acquire and process subsequent images and validate each ambiguous pose estimate by exploiting principles of relative orbit dynamics and kinematics.

2.2 Convolutional Neural Networks

An integral component of the pose estimation methods introduced in Section 1.3.3 are Convolutional Neural Networks (CNNs). The CNNs are an example of artificial neural networks that are specialized to process image data, i.e., two- or three-dimensional grids of image intensity values [64]. This section describes the fundamental building blocks of CNNs: the convolution operation and the pooling operations. The section also describes how CNNs are trained using the backpropagation and gradient descent techniques.

2.2.1 Convolution

The underlying mathematical operation of a convolutional layer is the convolution. Convolution is a linear operation that acts on two functions of a real-valued argument. It is typically denoted with an asterisk between the input, $x(t)$, and a weighting function or kernel, $w(t)$,

$$y(t) = (x * w)(t) \tag{2.19}$$

Typically, the kernel $w(t)$ is a probability density function in order to keep the output, $y(t)$, a valid weighted average of the input, $x(t)$. Mathematically, the output or the activation map is computed as

$$y(t) = \int x(a)w(t-a)dt \tag{2.20}$$

For image processing and deep learning applications, the input, $x(t)$, and the kernel, $w(t)$ are typically multi-dimensional. Therefore, the convolution operation is carried out in multiple axes simultaneously, i.e.,

$$Y_{i,j} = (X * W)_{i,j} = \sum_m \sum_n X_{m,n} K_{i-m,j-n}. \tag{2.21}$$

Figure 2.10 visually represents the convolution operation between the input image X and the kernel, W. Note that the figure only shows two steps of the convolution process as the kernel is "slid" across the entire input image.

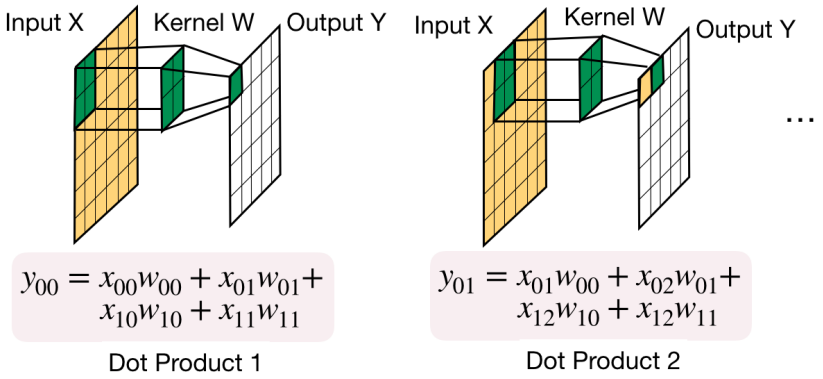


Figure 2.10: Representation of the convolution operation between the input image, X, and the kernel, W.

A convolutional layer is a collection of learnable kernels where each kernel is described by width, height, and depth. The depth for each kernel is fixed such that it spans the entire depth of the input volume. For example, let's assume a kernel of size $w_1 \times h_1 \times d$. Given an input image of size $w_2 \times h_2 \times d$, the kernel is convolved across the width and height of the input image volume to compute the dot product between the entries of the kernel and the input volume at all positions. The resulting output of the convolution, $y(t)$, is a two-dimensional activation map that gives the responses of this particular kernel at every spatial position of the input volume. This process is repeated for each constituent kernel of the convolutional layer. The output size of the convolutional layer is typically governed by the input length and width; the kernel length, width, and depth; and the kernel stride. The choice of these hyperparameters (among others) governs the volumetric representation of the input image as it passes

through various convolutional layers. Figure 2.11 shows an example of an input image passing through a convolutional neural network, where the application of each subsequent convolutional layer modifies the size of the volume. In this manner, a high-dimensional volume, such as an image, can be related to a low-dimensional vector describing the image, such as the six-dimensional relative pose between the camera and the target spacecraft.

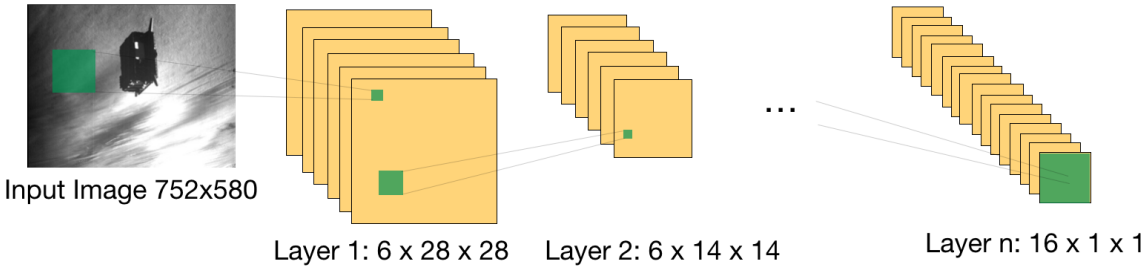


Figure 2.11: An example of a convolutional neural network acting on an input image volume of 752×580 and converting it into a vector of size $16 \times 1 \times 1$ using a series of convolutional layers.

For example, the first convolutional layer of AlexNet, a popular CNN architecture contains 96 kernels that are each $11 \times 11 \times 3$ in size [65]. Each of these kernels is convolved with an input image of size $227 \times 227 \times 3$ to produce an activation map of size $55 \times 55 \times 96$. The parameters of each kernel can be determined using supervised learning and the techniques mentioned in the subsequent sections, such that their resulting activation maps contain features such as edges, blobs, intricate patterns, etc. For reference, Figure 2.12 shows the 96 kernels of AlexNet when their parameters were learned using the ImageNet dataset [65].

2.2.2 Activation and Pooling functions

A typical convolutional layer consists of the linear convolution operation, a non-linear activation function, and a dimensionality reducing or pooling function. The non-linear activation functions are inserted to model the non-linear relationship between the input image and the desired output (i.e., the relative pose). The non-linearities introduced via the activation function are important since in the absence of the activation function a series of convolutions would be simply reduced to a linear regression

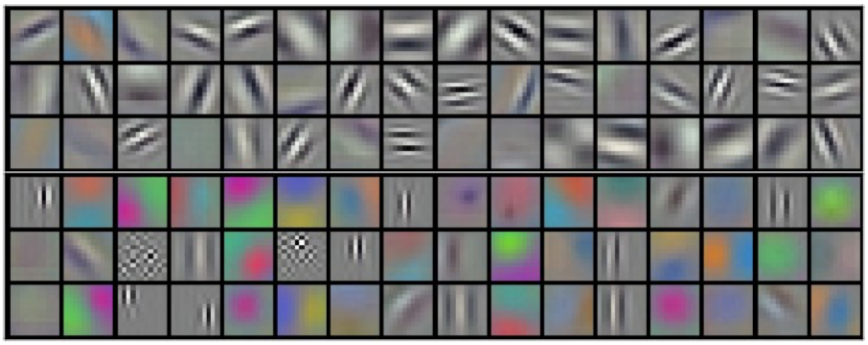


Figure 2.12: Representation of the 96 kernels constituting the first convolutional layer of the AlexNet architecture when trained using the ImageNet dataset.

problem relating the image with the relative pose. Typically, functions such as sigmoid $f(x) = 1/(1 + e^{-x})$, hyperbolic tangent, rectified linear unit $f(x) = max(0, x)$, and leaky rectified linear unit $f(x) = \mathbb{1}(x < 0)(ax) + \mathbb{1}(x >= 0)(x)$ are used as the activation function.

After the activation function, a pooling operation such as the max-pooling operation is used for dimensionality reduction [66]. The max-pooling operation is typically performed using a square grid of 2 or 3 pixels wide and outputs the maximum value within the square grid of the input that is being considered. Pooling has an effect of making the output of the convolutional layer to be invariant to small translations of the input. The dimensionality reduction offered by pooling operations progressively reduces the spatial size of the intermediate representation of the input image, thus reducing the number of parameters and computation required for a CNN.

2.2.3 Gradient-based Learning

At the heart of machine learning and deep learning-based pose estimation techniques lies the concept of gradient-based learning. Gradient-based learning is a numerical approach to approximate the non-linear function, $Y_i \approx F(X_i, W)$ that relates the patterns contained in the input image, X_i, and the desired output, Y_i. The "learning" occurs via the minimization of a loss function, L, that measures the difference between the modeled output, $\tilde{Y}_i$, and the desired output Y_i,

$$E_i = L(Y_i, \tilde{Y}_i). \tag{2.22}$$

The above equation represents the loss for a single image, X_i contained in the training dataset. To minimize the loss over the entire training dataset, the gradient of the loss function with respect to the learnable parameters, W, is computed.

Efficient and accurate computation of the gradient can be done either analytically or numerically via finite differences. In practice, the backpropagation algorithm [67] is the most widely used algorithm to compute the gradient values. Note that the backpropagation algorithm is also known as "automatic differentiation" or the "adjoint method" outside the field of artificial neural networks. Its use in training artificial neural networks to solve complex perception tasks was not widely adopted until it was realized that the presence of local minima in the minimization of loss functions was not a significant problem in practice [68]. The details of the backpropagation algorithm can be found in the original paper as well as several textbooks. Therefore, these are not included here for brevity [67, 69].

The resulting gradient values are then used to adjust the values of W in an iterative fashion

$$W_k = W_{k-1} - \epsilon \frac{\partial E(W_{k-1})}{\partial W_{k-1}}, \tag{2.23}$$

where ϵ can be a constant and scalar learning rate and treated as a hyper-parameter. The specific flavor of gradient learning that involves updating the learnable parameters for a single image at a time, as shown above, is called stochastic gradient descent. Updates to W can be performed for every mini-batch of n images in the training dataset to avoid significant variance in the parameter values. Theoretically, gradient descent does not guarantee global minimization of the loss function. Therefore in practice, both stochastic and mini-batch gradient descent can be problematic since variances in hyperparameter ϵ can lead to slow convergence or even divergence. State-of-the-art learning algorithms suggest more sophisticated methods to adjust the learning rate, such as momentum update [70] and Adaptive Moment Estimation (ADAM). Momentum update helps accelerate the stochastic gradient descent by using a component of the previous update of the learnable parameters to compute the

current update,

$$v_k = \gamma v_{k-1} + \epsilon \frac{\partial E(W_{k-1})}{\partial W_{k-1}}, \tag{2.24}$$

$$W_k = W_{k-1} - v_k, \tag{2.25}$$

where γ is another hyperparameter. Instead, the ADAM method adaptively computes the learning rate using an exponentially decaying average of the past values of the gradients. Specifically, the first and second moments of the gradient for each learnable parameter, g_i, are computed as

$$m_k = \beta_1 m_{k-1} + (1 - \beta_1) g_i \tag{2.26}$$

$$v_k = \beta_2 v_{k-1} + (1 - \beta_2) g_i^2 \tag{2.27}$$

These moments are normalized as

$$\tilde{m}_k = \frac{m_k}{1 - \beta_1^k}, \tag{2.28}$$

$$\tilde{v}_k = \frac{v_k}{1 - \beta_2^k}, \tag{2.29}$$

and then used to compute the parameter update

$$W_k = W_{k-1} - \frac{\eta}{\sqrt{\tilde{v}_k} + \epsilon} \tilde{m}_k. \tag{2.30}$$

Here both β_1 and β_2 are hyperparameters. A comparison of these and other variants of gradient learning methods can be found in the literature [71, 72].

Chapter 3

Feature-based pose estimation

This chapter presents the design and validation of the Sharma-Ventura-D'Amico (SVD) method for robust monocular vision-based pose initialization based on feature detection. In the context of this dissertation, the SVD method is an example of a feature-based pose estimation method. With respect to the state-of-the-art techniques, the proposed method is: (1) more robust to the background in the images; (2) requires no a-priori knowledge of the target spacecraft's attitude and position; and (3) more computationally efficient. To ensure robustness to the background, this chapter introduces the Weak Gradient Elimination (WGE) technique. WGE detects and eliminates the regions with a weak gradient in the image and highlights the regions of the strongest gradients. In this way, only the strongest edges in the image are used for feature extraction using the Hough transform [41]. The extracted features are then organized into geometrically complex groups to dramatically reduce the search space of the feature correspondence problem. Possible hypotheses for the feature correspondence are used to efficiently solve the Perspective-n-Point (PnP) problem. The method is validated through flight imagery collected during the PRISMA mission [42] at about 700 [km] altitude and 10 [m] inter-spacecraft separation. The validation experiments presented in this chapter allow a direct comparison of the SVD method against other feature-based methods. This is especially important since the performance envelope of feature-based methods must be characterized before diving into the main contribution of this research, the Spacecraft Pose Network (SPN), in Chapter 5.

This chapter is organized as follows: Section 3.1 introduces the pose initialization

problem and an overview of the proposed method. Section 3.2 describes the technical design details of the image processing subsystem. Section 3.3 describes the technical design details of the pose determination subsystem. Section 3.4 presents four tests that compare the proposed SVD method against the state-of-the-art techniques for monocular vision-based pose initialization. These tests utilize synthetic imagery as well as imagery collected during the PRISMA mission. Section 3.5 elaborates on the conclusions with a discussion on the applicability of this method in a monocular vision-based navigation system for on-orbit servicing and formation flying missions.

3.1 Architecture

The schematic representation of the noncooperative pose initialization problem from a 2D image and a 3D model is shown in Figure 3.1. Note that the following description is the same as the description of the PnP problem in Section 2.1.1 except for one important distinction. The problem of noncooperative pose initialization does not assume known correspondences between the 3D model points and the 2D image points while these are assumed to be known in the PnP problem. In particular, the pose estimation problem consists of determining the position of the target's center of mass, $\mathbf{t}^{\mathrm{C}}$, and the orientation of the vehicle's principal axes, $\mathbf{R}^{\mathrm{BC}}$, with respect to the camera frame C. It is assumed that the 3D model of the target is defined in the body-fixed reference frame, B and it is aligned with the target's principal axes with its origin at the center of mass. The orientation is defined by the direction cosine matrix $\mathbf{R}^{\mathrm{BC}}$ from the reference frame B to C.

Let $\mathbf{q}^{\mathrm{B}}$ be points of the 3D model expressed in reference frame B. By employing the standard pinhole camera model, the corresponding points $\mathbf{p} = [u,\ v]^T$ in the rectified image can be obtained using the 3D-2D true perspective projection equation

$$\mathbf{r}^{\mathrm{C}} = \left[x^{\mathrm{C}}\ y^{\mathrm{C}}\ z^{\mathrm{C}}\right]^T = \mathbf{R}^{\mathrm{BC}}\,\mathbf{q}^{\mathrm{B}} + \mathbf{t}^{\mathrm{C}} \tag{3.1}$$

$$\mathbf{p} = \left[\frac{x^{\mathrm{C}}}{z^{\mathrm{C}}} f_x + C_x,\ \frac{y_{\mathrm{C}}}{z^{\mathrm{C}}} f_y + C_y\right], \tag{3.2}$$

where $\mathbf{r}^{\mathrm{C}}$ represents the points of the target spacecraft expressed in camera frame C according to the current pose, $\mathbf{t}^{\mathrm{C}}$ and $\mathbf{R}^{\mathrm{BC}}$. Here f_x and f_y denote the focal lengths

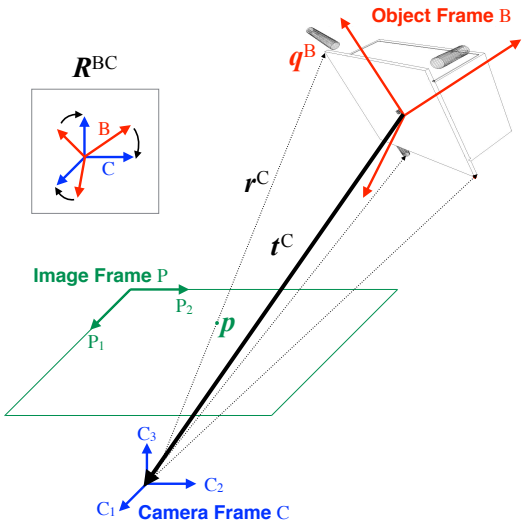


Figure 3.1: Schematic representation of the pose estimation problem using monocular image.

of the camera, (C_x, C_y) denotes the principal point of the image. Without loss of generality, it is assumed that the direction C_3 is pointed along the bore-sight of the camera and that the directions C_1 and C_2 are aligned with the directions P_1 and P_2 of the image frame, P, respectively. The unknown coefficients in the 3D-2D true perspective equations are the three components of the relative position $\mathbf{t}^C$ and the three parameters that define the rotation matrix $\mathbf{R}^{BC}$ of the relative orientation. At least three image points and corresponding model points are required to solve this system of equations. However, at least six correspondences between image and model points are required to obtain a unique solution with a general configuration of points [54].

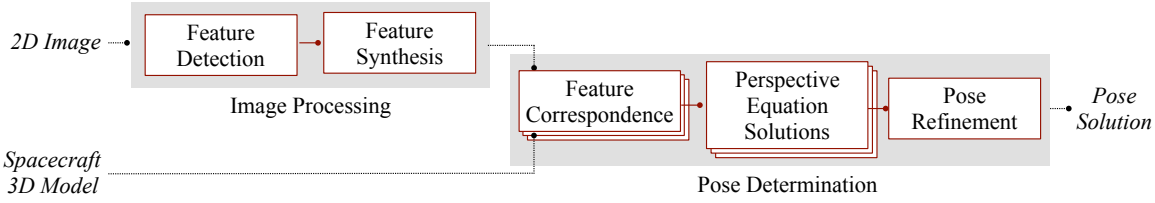


Figure 3.2: Proposed pose initialization method with inputs of a single 2D image and a 3D model and an output of the pose solution.

Figure 3.2 illustrates the architecture for the SVD method. The key features of the SVD method are: (1) the fusion of the WGE technique with the Sobel edge detector and the Hough algorithm for feature detection; (2) the use of feature synthesis to reduce the search space for feature correspondences between the image and the 3D model; and (3) the combination of the EPnP solver and the Newton Raphson Method for pose determination. The SVD method consists of two subsystems:

1. Image processing: This subsystem accepts a single 2D image as its input. It distinguishes the target spacecraft from the background, detects the target spacecraft's edge features, and then synthesizes them into geometric groups.

2. Pose determination: This subsystem accepts the feature groups detected in the image processing subsystem and the 3D model as its input. It pairs the 2D and 3D geometric groups to create multiple correspondence hypotheses. For each hypothesis, the end-points of the line segments forming the geometric groups are used in solving Equation 3.1 and Equation 3.2. Of the multiple resulting pose solutions, the top five are iteratively refined, and the best is selected using reprojection error as the confidence metric.

With respect to the state-of-the-art image processing, the key innovation of this subsystem is the use of the WGE technique to distinguish the target spacecraft from the background efficiently. The same technique is used to boost the output of off-the-shelf edge detection techniques to provide a robust identification of the small as well as large edges of the spacecraft. Secondly, the critical innovation in the pose determination subsystem is the use of geometrically complex feature groups detected in the image to solve the feature correspondence problem. This use of feature groups is used to overcome the challenge of pose ambiguity while still being more computationally efficient than the state-of-the-art methods such as pose-space grid search or RANSAC. Lastly, the pose solution output is accompanied by its reprojection error, which denotes the quality of its fit with the detected image features. Therefore, if the reprojection error is calculated to be higher than a threshold, the solution can be classified as a "low confidence solution". In case none or a low number of features are detected in the image, a coarse relative position solution is output using the ROI detected by the WGE technique. In this manner, the SVD method can be used

iteratively on a sequence of images until a "high confidence solution" is available.

Since the above method requires solving the 3D-2D perspective equations (Eqs. 3.1-3.2) for multiple 3D-2D feature correspondences, it is important to take the results from Section 2.1 into account. In particular, the relative strengths and weaknesses of the PnP solvers highlighted in that section are used to make design choices for the two subsystems of the SVD method. The subsequent sections delve into the details of the image processing and pose determination subsystems of the SVD method.

3.2 Image Processing

The goal of the image processing subsystem is to extract the most significant features of the target spacecraft in the input image. Subsequently, the extracted features are organized in geometrically complex groups to perform an efficient feature correspondences analysis for pose determination.

Pose initialization is assumed to be executed at the beginning of the close-range procedures, ideally before the pose tracking that leads to the capture of the target spacecraft. Since the distance from the target is still considerable (30 meters in the case of the PRISMA mission [42]), small details in the target's surface are not visible from the acquired image and only the most significant features such as the end-points of the body edges, antennae, and solar panels can be distinguished. For this reason, the image processing subsystem solely focuses on extracting edges or straight line segments. Furthermore, as opposed to features based on color gradients, textures, and optical flow, edges are less sensitive to illumination changes [73].

Robust image processing is one of the main challenges in vision-based pose estimation. Illumination conditions in space may even vary during a single orbit and, therefore, may cause inaccurate and unreliable features detection. This is mainly due to the high-contrast illumination and low Signal-to-Noise Ratio (SNR). Moreover, the potential presence of the Earth in the image background affects the image processing due to the presence of additional features of the planet's surface and the high reflectivity of oceans and clouds. State-of-the-art techniques for straight-line segment detection such as the Canny Edge detector [74] followed by the Hough transform [41] may be biased if applied directly to the image as these algorithms are gradient-based

and do not distinguish the foreground from the background. These methods also require the definition of numerous hyper-parameters, which are difficult to tune for broad applicability as the imaged scene and the illumination conditions are continually changing throughout the orbit. Finally, current methods for image segmentation [39] are computationally expensive and therefore not suitable for onboard spacecraft applications.

Figure 3.3 shows the hybrid image processing subsystem proposed to detect the edges corresponding to the target spacecraft even in the presence of the Earth in the background. With respect to the off-the-shelf feature detection techniques, the key innovation of this subsystem is the introduction of the WGE technique and its fusion with the state-of-the-art edge detection techniques to provide an efficient and robust identification of the actual edges of the spacecraft. As demonstrated in Section 3.4, the WGE technique identifies a more accurate and robust ROI in the image as compared to the state-of-the-art techniques such as Maximally Stable Regions (MSER) [75]. The ROI detection makes the subsystem robust to the background as well as allows for an automatic selection of hyper-parameters required for the Hough transform. Hence, the subsystem not only finds straight line segments corresponding to the large features of the spacecraft, but it also detects straight line segments corresponding to smaller features such as antennae. Even though the current image processing subsystem relies on detecting straight line segments, it can be easily extended to include cylindrical, spherical, and circular features through the formulation of separate Hough transforms. The edges corresponding to these features can then be classified as additional feature groups within the proposed method. In the following sections, the proposed feature detection and feature synthesis procedures are described in detail.

3.2.1 Feature Detection

The feature detection procedure aims at identifying a robust set of line segments in the rectified input image corresponding to the most significant edges of the target spacecraft. Some line segments correspond to the spacecraft's large features such as the contour of the bus and solar panels, whereas others correspond to small features such as antennae. In particular, small features are essential for resolving geometry related ambiguity in pose determination. This aspect will be discussed in more detail

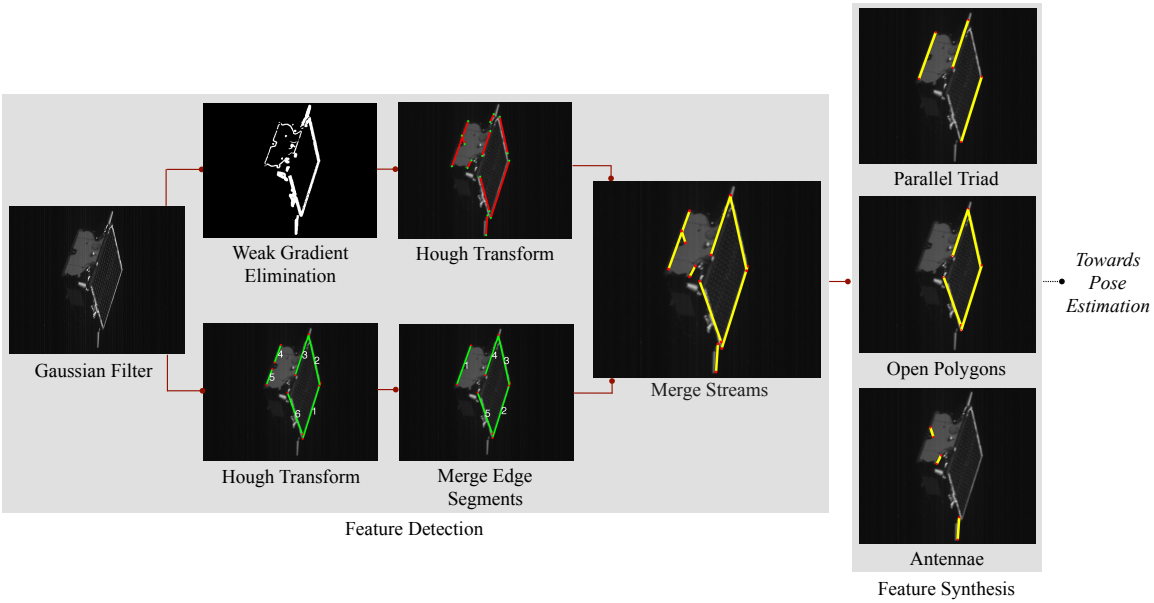


Figure 3.3: Main steps of the image processing subsystem with a single 2D image as the input and feature groups as the output.

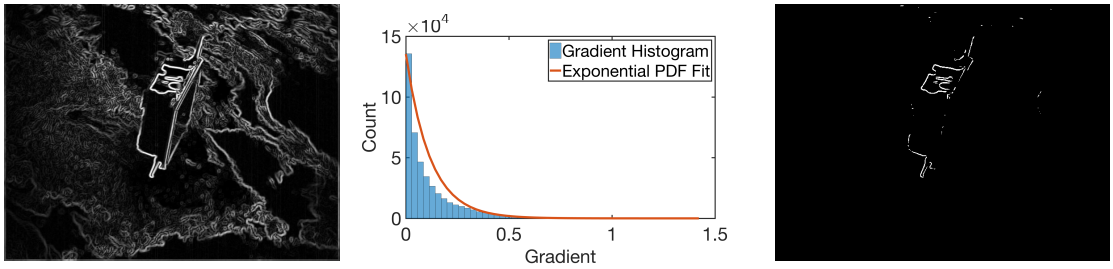
in Section 3.3.

Figure 3.3 shows the intermediate steps of the feature detection procedure. The input raw image is assumed to be rectified, i.e., corrected for lens distortion [76]. The Gaussian filter is applied to the input image to attenuate the image noise. The filtered image is then subjected to two parallel streams. In the first stream, the WGE technique is applied to distinguish the spacecraft from the background. A Hough transform is then applied to find smaller edges that could correspond to features such as the antennae. In the second stream, the Sobel algorithm is used for edge detection followed by another Hough transform to extract long edges that could correspond to features such as the solar panels. The line segments obtained from these two streams are finally merged while duplicates are discarded.

Weak Gradient Elimination

The Weak Gradient Elimination (WGE) technique was conceived to distinguish the target spacecraft in the foreground from the background. The size of the resulting ROI enables the automated selection of hyper-parameters required for the detection

of both small and large features of the spacecraft. State-of-the-art algorithms based on the Hough transform rely on a single set of manually tuned hyper-parameters and therefore tend to either detect long line segments only or fuse the short line segments with one of the proximal longer line segments. Moreover, if the image has the planet in the background, line segments belonging to clouds or coastlines bias the output. The WGE technique solves both of these problems.



(a) (b) (c)

Figure 3.4: Steps of weak gradient elimination, (a) Gradient detection of the original image, (b) Histogram of the normalized gradient values and the exponential PDF approximation, and (c) Output filtered image gradients.

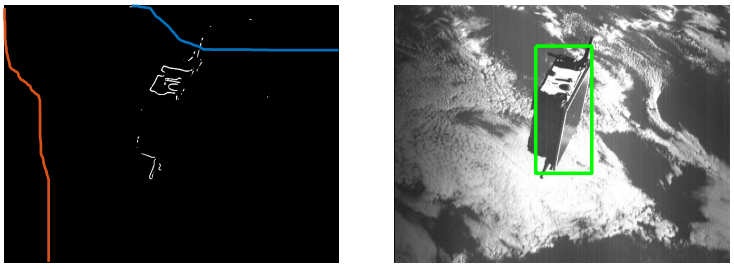
The first step of this technique is the calculation of the image gradient $\mathbf{G}(\mathbf{u}, \mathbf{v})$ at all pixel locations by using the Prewitt operator [77]. In particular, two 3×3 kernels are convolved with the original image to calculate approximations of the horizontal and vertical derivatives. Let the input image be $\mathbf{A}$, the approximation of the horizontal derivative at each point in the image be $\mathbf{G_x}$, and the approximation of the vertical derivative at each point in the image be $\mathbf{G_y}$. Then,

$$\mathbf{G_x} = \begin{bmatrix} -1 & 0 & +1 \\ -1 & 0 & +1 \\ -1 & 0 & +1 \end{bmatrix} * \mathbf{A} \qquad \mathbf{G_y} = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ +1 & +1 & +1 \end{bmatrix} * \mathbf{A} \tag{3.3}$$

where $*$ denotes the 2-dimensional convolution operation. At each point in the image, $\mathbf{A}(u, v)$, the resulting gradient approximations can be combined to give the gradient magnitude, using

$$\mathbf{G}(u, v) = \sqrt{\mathbf{G}_x{}^2(u, v) + \mathbf{G}_y{}^2(u, v)} \tag{3.4}$$

Figure 3.4(a) shows the gradient of a test image that has the Earth as the background. Due to the presence of the planet, it is difficult to distinguish the spacecraft's true edges from the background clearly. To detect the spacecraft, the histogram of the gradient is obtained by sorting $\mathbf{G}(\mathbf{u}, \mathbf{v})$ in 100 uniformly spaced bins. As can be observed in Figure 3.4(b), most of the gradient intensities are weak and correspond to the features in the background or on the spacecraft's surface. The obtained histogram can be approximated by an exponential probability distribution function (PDF) as in Figure 3.4(b). The weak gradient pixel locations are then classified by thresholding the PDF fit to the gradient histogram. More precisely, the bin corresponding to the cumulative distribution of 0.99 is found by calculating the area under the curve of the PDF. All pixel locations corresponding to bins below this are classified as "weak" and their gradient value is set to zero. Figure 3.4(c) shows the result of this technique, the pixel locations corresponding to the background and reflective surfaces have been eliminated leaving behind pixel locations corresponding to the most prominent features of the spacecraft.



(a) (b)

Figure 3.5: ROI detection process, (a) Cumulative population of strong gradients along the two image axes and (b) Output ROI.

Algorithms relying on the Hough transform require the tuning of hyperparameters such as the expected minimum length of line segments and the expected maximum gap between two points to be considered in the same line segment. This makes these algorithms unsuitable for broad applicability as these parameters need to be varied for different imaging scenarios [78]. To overcome this challenge, the WGE technique adaptively computes these hyper-parameters through the detection of a rectangular ROI around the target spacecraft in the image. The limits of the rectangular ROI are

determined independently in each of the two axes of the image. To build the ROI, the Cumulative Distribution Function (CDF) of the filtered image gradient obtained from the WGE is determined along the two image axes, as seen in Figure 3.5. Assuming the filtered image gradient is normally distributed, the coordinates of the ROI are determined by axes positions corresponding to a CDF of 0.025 and 0.975 (therefore, only the central 95% of the normal distribution is considered).

The Hough transform is applied to the binary image of the filtered gradient to extract line segments corresponding to small spacecraft features. The required hyper-parameters of the Hough transform, namely, the expected minimum length of the line segments, $l_{\text{min,Hough}}$, and the maximum gap between two points to be considered in the same line segment, λ_{Hough}, can be calculated as scalar multiples of the diagonal length of the ROI, l_{ROI}.

$$l_{\text{min,Hough}} = \kappa_1 * l_{\text{ROI}} \qquad \lambda_{\text{Hough}} = \kappa_2 * l_{\text{ROI}} \tag{3.5}$$

The output of short line segments are stored and later merged with the line segments belonging to large features. The scalars κ_1 and κ_2 can be empirically estimated in a simulation on-ground before the mission or can be estimated onboard using prior knowledge of the inter-spacecraft range based on an angles-only navigation phase [79].

An innovative feature of the WGE technique is its ability to provide a coarse relative position solution even prior to the pose determination subsystem.

Figure 3.6 shows that the knowledge of the diagonal characteristic length, L_C, of the spacecraft 3D model and the diagonal length of the detected ROI, l_{ROI}, can be used to obtain the position of the target spacecraft relative to the camera frame, $\mathbf{t}^{\text{C}}$. In particular, the range to the target spacecraft from the origin of the camera frame is

$$||\mathbf{t}^{\text{C}}||_2 = \frac{\left(\frac{f_x+f_y}{2}\right) L_C}{l_{\text{ROI}}} \tag{3.6}$$

where f_x and f_y denote the focal lengths of the camera. Azimuth and elevation angles, (α, β), from the origin of the camera frame, C to the origin of the body-fixed coordinate system, B can be derived using the principal point of the image, (C_x, C_y)

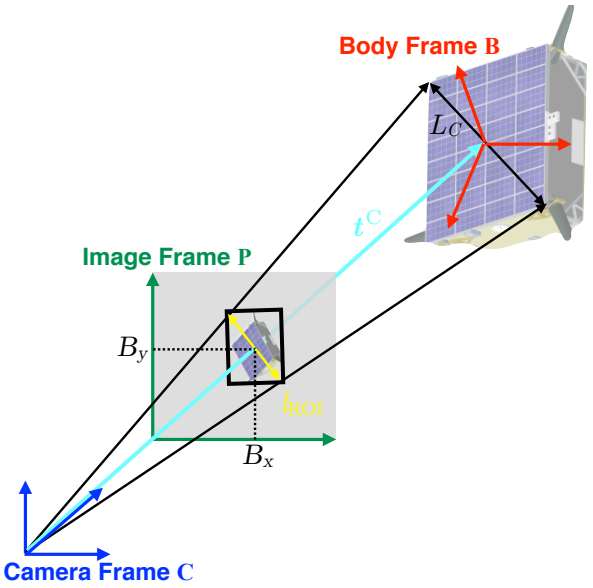


Figure 3.6: Calculation of a coarse relative position solution using the WGE technique.

and the center of the ROI, (B_x, B_y)

$$\alpha = \tan^{-1}\left(\frac{B_x - C_x}{f_x}\right) \tag{3.7}$$

$$\beta = \tan^{-1}\left(\frac{B_y - C_y}{f_y}\right). \tag{3.8}$$

Finally, the coarse relative position solution is given by

$$\mathbf{t}^{\mathrm{C}} = \begin{bmatrix} C_\alpha & 0 & -S_\alpha \\ 0 & 1 & 0 \\ S_\alpha & 0 & C_\alpha \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & C_\beta & S_\beta \\ 0 & -S_\beta & C_\beta \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ ||\mathbf{t}^{\mathrm{C}}||_2 \end{bmatrix}. \tag{3.9}$$

Sobel & Hough

The second stream of the feature detection procedure consists of the application of the Sobel operator and the Hough transform technique (S&H) [41] to the rectified image. The objective is to efficiently extract line segments corresponding to the silhouette of the large components of the spacecraft. Any line segment whose mid-point lies

outside the ROI detected from the WGE technique is rejected. The Hough transform hyper-parameters are calculated as

$$l_{\text{min,Hough}} = \kappa_3 * l_{\text{ROI}} \qquad \lambda_{\text{Hough}} = \kappa_4 * l_{\text{ROI}}. \tag{3.10}$$

Note that instead of manually tuning the Hough Transform hyper-parameters for each image separately, the hyper-parameters are adaptively computed based on the scalar multiples κ_3 and κ_4. Due to this formulation of the Hough hyper-parameters, the proposed straight line segment extraction is mostly robust to the inter-spacecraft separation. The hypothesis is that as the target spacecraft gets closer to the camera, the expected lengths of its edges in the image plane are expected to grow proportionally to the size of the detected bounding box. Hence, the determination of parameters κ_3 and κ_4 needs to occur just once, for example, in an offline phase before the mission.

Merging Edges

The output line segments from the Hough transform often correspond to multiple and truncated edges. In Figure 3.3, Edge #4 and Edge #5 obtained from the Hough transform correspond to the same true line segment. To resolve this issue, similar line segments are merged into a single line segment. For example, consider two line segments, l_1 and l_2, expressed in the polar form as show in Figure 3.7a.

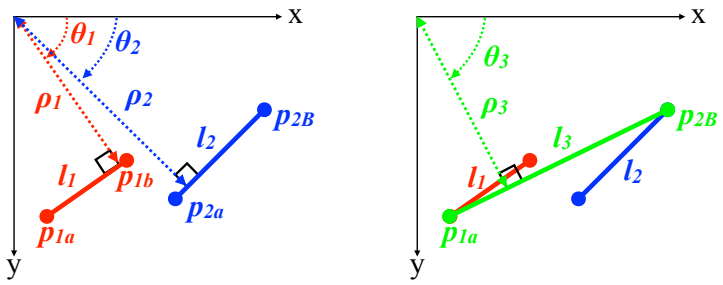
$$l_1: \quad \rho_1 = x\cos\theta_1 + y\sin\theta_1 \tag{3.11}$$

$$l_2: \quad \rho_2 = x\cos\theta_2 + y\sin\theta_2 \tag{3.12}$$

The condition of similarity is that $|\theta_1 - \theta_2| < \theta_{\text{thresh}}$ and $|\rho_1 - \rho_2| < \rho_{\text{thresh}}$. Furthermore, the Euclidean distance between the farthest pair of end-points of the two line segments must be less than d_{thresh}. The parameter d_{thresh} is adaptively computed for each image as half of the mean length of the detected edge segments while the parameters θ_{thresh} and ρ_{thresh} are set equal to the resolution of θ and ρ in the Hough space. If the similarity condition is verified, the two line segments are replaced with l_3, the line segment defined by the farthest pair of end-points of the line segments, as

shown in Figure 3.7b.

$$l_3 : \quad \rho_3 \quad = x \cos\theta_3 + y \sin\theta_3 \tag{3.13}$$



(a) (b)

Figure 3.7: Merging of two truncated edges, (a) original edges and (b) output merged edge.

Merging Streams

The final step of the feature detection procedure is to merge the line segments detected with the WGE and S&H techniques. It is made sure that unique line segments are output from each of the two streams before merging. This uniqueness check resolves the issue of detecting repeated edges as encountered in previous work [1]. Pairs of close and similar line segments are detected in the output of both the streams separately and only the longer line segment from the pair is preserved. Using the example of the pair of line segments l_1 and l_2 from Equation 3.11 and Equation 3.12, the longer line segment is retained if $|\theta_1 - \theta_2| < \tilde{\theta}_{\text{thresh}}$, $|\rho_1 - \rho_2| < \tilde{\rho}_{\text{thresh}}$, and the Euclidean distance between the midpoints is less than half of the length of the longer line segment. The parameters $\tilde{\theta}_{\text{thresh}}$ and $\tilde{\rho}_{\text{thresh}}$ can be tuned to a certain threshold or expressed as functions of the size of the ROI.

The results of the two streams are combined taking into account the cases where short line segments detected from the WGE technique overlap or intersect with the large line segments output from the S&H technique. Pairs of line segments from the

two streams are compared and checked whether they intersect. If they do intersect, then assuming the shorter line segment is divided into two portions with lengths L_1 and L_2 (where $L_1 < L_2$), only the longer line segment is preserved if $L_1/L_2 > 0.25$.

3.2.2 Feature Synthesis

The objective of feature synthesis is to organize the extracted line segments into higher-level features to reduce the search space of the correspondence problem, i.e., matching features of the model to the features in the image. Given n points in the image and m points in the 3D model, the number of possible correspondences and equivalently, the number of hypothetical pose solutions is given by $\binom{m}{6}\binom{n}{6}6!6!$. Note that at least six corresponding model and image points are needed to guarantee a unique solution to the PnP problem [54]. However, more constraints can be generated using the knowledge of the 3D model and by organizing the points into higher-level groups. For example, if four of these image points belong to a polygonal feature such as a solar panel, then a unique solution can be found using just four correspondences. This drastically reduces the number of possible correspondences to $8m_{\text{planar}}n_{\text{planar}}$, where m_{planar} and n_{planar} are the number of four-sided polygonal features of the 3D model and the image, respectively. Therefore, the key idea is to solve the correspondence problem using just a small number of higher-level feature groups instead of using a large number of feature points.

In this implementation of feature synthesis, the detected line segments are organized into five groups, namely, parallel pair, proximal pair, open polygonal triad, parallel triad, and closed polygonal tetrad. These groups are built using relations that are preserved over a wide range of camera viewpoints [80], thereby adding robustness to the proposed method. Moreover, these groups can be easily and quickly found from the detected line segments by examining just a few geometric constraints, as observed in Figure 3.8. For example, two segments comprise a proximity pair if they satisfy the condition

$$d_{12} \leq d_{\max} \tag{3.14}$$

where d_{12} is the shortest distance between endpoints and $d_{\max}$ is a threshold value in

pixels. Similarly, two segments compose a parallel pair if they satisfy the condition

$$\theta_{12} = \|\theta_1 - \theta_2\| \leq \theta_{\max} \tag{3.15}$$

where θ_1 and θ_2 are the line segment slopes and $\theta_{\max}$ is a threshold value in degrees. If two groups of proximity pairs share a line segment, they are categorized as an open polygonal triad if they satisfy the condition

$$(P_{1a} - P_{3a}) \cdot (P_{1b} - P_{3b}) > 0. \tag{3.16}$$

This ensures that the end-points of the non-common line segments lie on the same side with respect to the shared line segment. If two open polygonal triad are found with two shared line segments, then they are categorized as a closed polygonal tetrad. Similarly, if two parallel pairs are found with a shared line segment, they are categorized as a parallel triad. Finally, any line segments detected through the WGE technique feature detection stream that were not detected from the S&H feature detection stream are classified as antennae if the length of the segment is less than one-third of l_{ROI}. This particular threshold can be easily tuned prior to the mission as the 3D model of the target spacecraft is assumed to be available. As for the 3D model of the target spacecraft, the same groups can be pre-computed using the same conditions prior to the mission.

3.3 Pose Determination

Position and attitude of the target spacecraft are determined by solving the 3D-2D perspective projection equation for $\mathbf{R}^{\text{BC}}$ and $\mathbf{t}^{\text{C}}$. Since this requires corresponding image and 3D model points, the detected feature groups must be first matched with the corresponding feature groups of the model. Notably, several possible combinations between image and correlated model features must be considered since the exact correspondences are unknown. The 3D-2D true perspective projection equation is then solved for each combination of feature correspondences using the EPnP method [36]. The best pose candidates are then refined through a Newton-Raphson algorithm to output a single pose solution. Recall that the performance of the EPnP

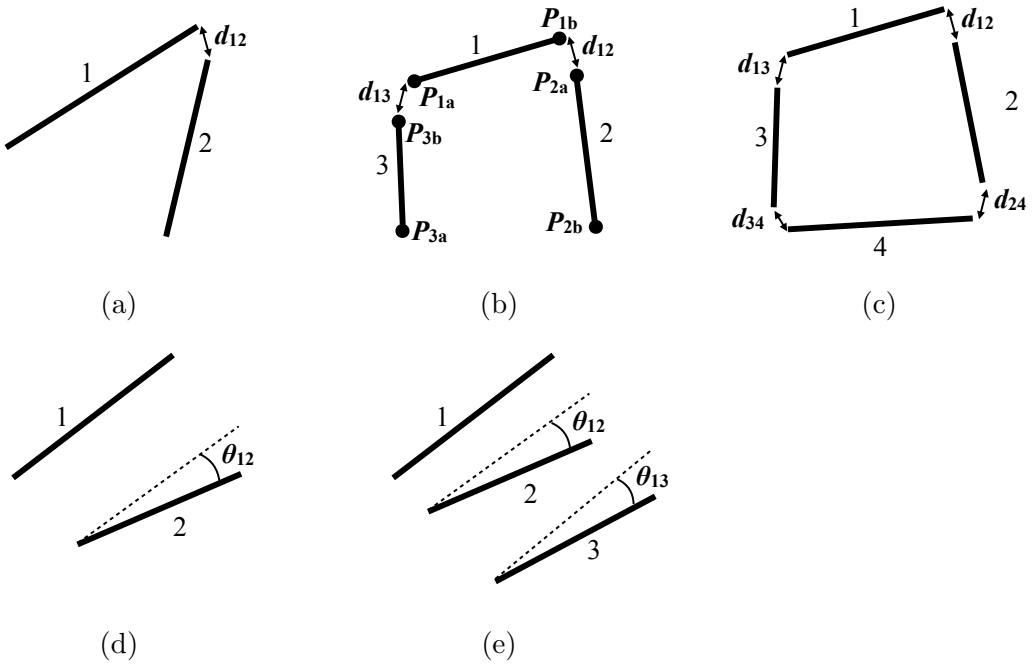


Figure 3.8: Synthesis of detected line segments into higher level features, (a) Proximal pair, (b) Open polygonal triad, (c) Closed polygonal tetrad, (d) Parallel pair, and (e) Parallel triad.

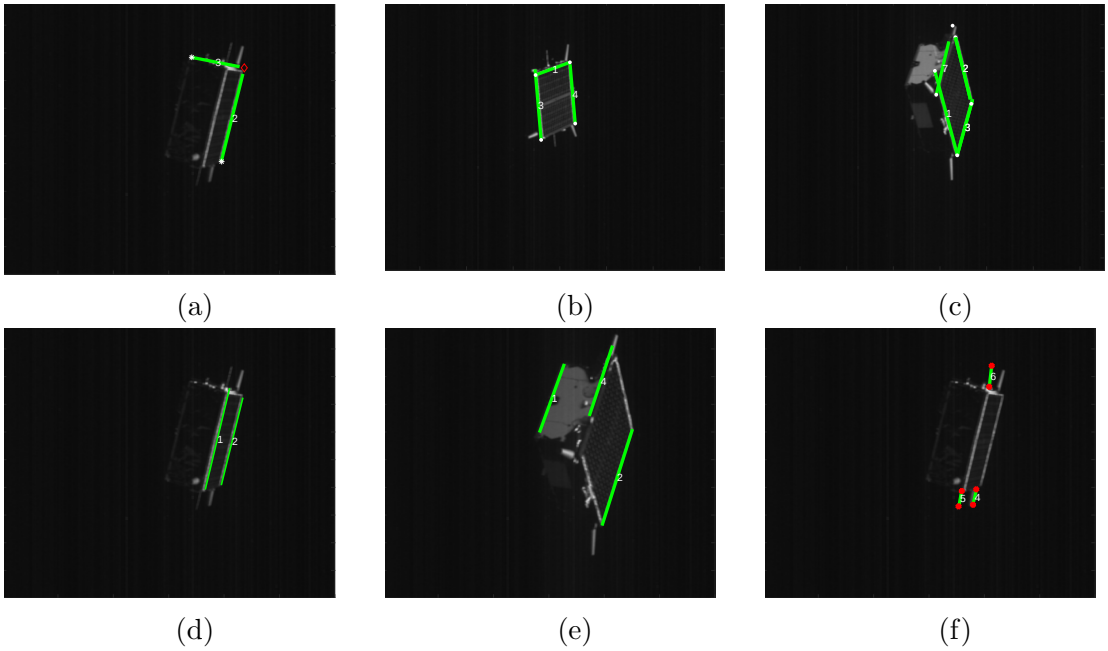


Figure 3.9: Intermediate results from synthesis of detected line segments into higher level features, (a) Proximal pair, (b) Open polygonal triad, (c) Closed polygonal tetrad, (d) Parallel pair, (e) Parallel triad, and (f) Antennae.

and Newton-Raphson algorithm along with other state-of-the-art PnP solvers were characterized in Section 2.1. That characterization concluded that the EPnP was the fastest PnP solver while the Newton-Raphson method was the most accurate. Figure 3.10 presents the pose determination subsystem while the following subsections discuss its main steps in detail.

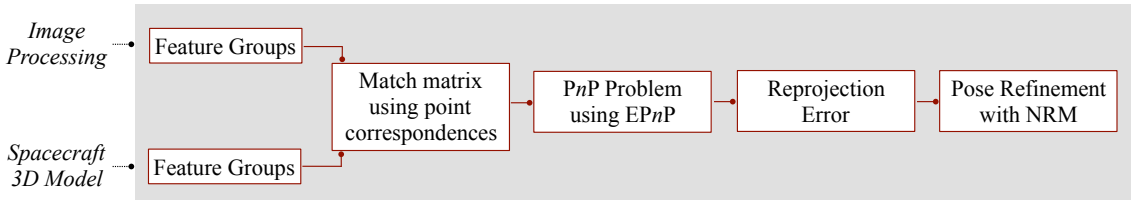


Figure 3.10: Main steps of the pose determination subsystem with feature groups from the image and the 3D model as input and a single pose solution as the output.

3.3.1 Feature Correspondence

The correspondences between the image points and the 3D model points are obtained by pairing each feature group detected in the image with each analogous group of the 3D model of the target spacecraft. For each matching feature group pair, the endpoints of the line segments in the image are hypothesized to correspond with the endpoints of the 3D model's lines through simple combinations. For instance, a closed polygon identified in the image is coupled with every closed polygon of the model. This provides eight different combinations of the point correspondences between closed polygon detected in the image and the closed polygon of the 3D model. This approach is applied to all the feature groups considered in the feature synthesis in Section 3.2.2.

The point correspondences between the image and the 3D model are stored in the so-called match matrix, which is then input to the EPnP method [36]. The rows of the matrix represent the different hypotheses for feature correspondence while the columns store the corresponding 3D model and 2D image points. EPnP requires at least six point correspondences to guarantee a unique pose solution from Eqs. 3.1-3.2 [23] while a single feature group typically provides less than six (see Table 3.1). Therefore, the point correspondences from the feature groups are combined with the

point correspondences provided by the antennae feature group to form at least six point correspondences. Note that the decision to combine point correspondences from complex feature groups with point correspondences from the antennae feature group is a design choice since the Tango spacecraft has five prominent antennae visible from most viewing angles. For other spacecraft where either the antennae are not present or not prominent, point correspondences from other complex feature groups may be used to form at least six point correspondences. The feature groups are ranked according to their geometric complexity as follows (in descending order): closed polygonal tetrad, open polygonal tetrad, parallel triad, proximal pair, and parallel pair. To build the match matrix, only the most geometrically complex feature group detected in that image is considered. Table 3.1 shows the number of rows in the match matrix for a typical scenario where at least three antennae are detected. Building the match matrix in this manner has two main advantages: first, the probability of an accidental detection of the higher geometric complexity group is less than that of the lower geometric complexity group [80]; and second, the higher the geometric complexity of the feature group, the lower is the number of possible correspondences between the image and the 3D model features.

Table 3.1: The expected number of rows in the match matrix (Column 5) based on the most geometrically complex feature group detected in the image (Column 1).

Feature Group	Number of points per feature group	Number of feature groups in image	Number of feature groups in 3D model	Number of rows in match matrix
Closed Polygonal Tetrad	4	ϕ_a	ϕ'_a	$8\phi_a\phi'_a\phi_f\phi'_f$
Open Polygonal Tetrad	4	ϕ_b	ϕ'_b	$8\phi_b\phi'_b\phi_f\phi'_f$
Parallel Triad	6	ϕ_c	ϕ'_c	$24\phi_c\phi'_c$
Parallel Pair	4	ϕ_d	ϕ'_d	$8\phi_d\phi'_d\phi_f\phi'_f$
Proximal Pair	3	ϕ_e	ϕ'_e	$2\phi_e\phi'_e\phi_f\phi'_f$
Antenna	1	ϕ_f	ϕ'_f	-

Unlike most previous algorithms to hypothesize feature correspondence, not all possible feature matches are treated identically. Instead, a small set of matches are hypothesized and then verified. This is in stark contrast to view-based approaches

where image features were compared with pre-computed 2D views of the object to determine the object pose [81, 82, 83, 84]. These approaches tried to deal with the full geometric search space by clustering the views. None of these are practical for the constrained space-hardened hardware due to their extremely large geometric search space.

The EPnP algorithm is applied to each combination of feature correspondences defined in the match matrix. The reprojection error is the Euclidean distance between image features and corresponding model points projected onto the image, which is mathematically expressed as

$$E_{2\mathrm{D}} = \frac{1}{n}\sum_{i=1}^{n}\sqrt{\left[u_i - \left(\frac{{x_i}^{\mathrm{C}}}{{z_i}^{\mathrm{C}}}f_x + C_x\right)\right]^2 + \left[v_i - \left(\frac{{y_i}^{\mathrm{C}}}{{z_i}^{\mathrm{C}}}f_y + C_y\right)\right]^2}. \tag{3.17}$$

Notably ${\mathbf{r}_i}^{\mathrm{C}} = \left[{x_i}^{\mathrm{C}}, {y_i}^{\mathrm{C}}, {z_i}^{\mathrm{C}}\right]$ is computed from the model feature point ${\mathbf{p}_i}^{\mathrm{B}}$ using Equation 3.1.

3.3.2 Pose Refinement

The best five pose solutions in terms of their solution error (Equation 3.17) are used for pose refinement. The first step of this procedure is to solve the 3D-2D perspective projection equation using the Newton-Raphson Method (NRM) [1] using each pose solution as the first guess. For each feature correspondence, the following fit error between detected image feature and projected model point is defined:

$$\mathbf{E}_i = \left[u_i - \left(\frac{{x_i}^{\mathrm{C}}}{{z_i}^{\mathrm{C}}}f_x + C_x\right), v_i - \left(\frac{{y_i}^{\mathrm{C}}}{{z_i}^{\mathrm{C}}}f_y + C_y\right)\right] \qquad i = 1, 2, ..., n \tag{3.18}$$

where ${\mathbf{r}_i}^{\mathrm{C}} = \left[{x_i}^{\mathrm{C}}, {y_i}^{\mathrm{C}}, {z_i}^{\mathrm{C}}\right]$ is obtained from ${\mathbf{p}_i}^{\mathrm{B}}$ using Equation 3.1. The fit error in Equation 3.18 has 6 unknown parameters being $\mathbf{x} = \left[\mathbf{t}^{\mathrm{C}}, \theta_{\mathrm{BC}}\right]$, where θ_{BC} is the Euler angles sequence that defines the rotation matrix $\mathbf{R}^{\mathrm{BC}}$. Since each feature correspondence provides two conditions, at least three matches between detected corners and projected features are required to solve the system of equations defined by the fit errors. Let $n \geq 3$ be the number of correspondences between image and model

features. The system of $2\times n$ equations that must be solved for $\mathbf{x}$ is given by

$$\mathbf{E}_{\mathrm{s}} = \begin{bmatrix} \mathbf{E}_1 \\ \dots \\ \mathbf{E}_n \end{bmatrix}. \tag{3.19}$$

NRM solves this system of equations by iteratively updating the solution as

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \left(\mathbf{J}^T\mathbf{J}\right)^{-1}\mathbf{J}^T\mathbf{E}_{\mathrm{s}}\left(\mathbf{x}_k\right), \tag{3.20}$$

where $\mathbf{E}_{\mathrm{s}}\left(\mathbf{x}_k\right)$ is evaluated using Equation 3.19 at $\mathbf{x}_k$ and $\mathbf{J}$ is the Jacobian of the system:

$$\mathbf{J} = \frac{\partial \mathbf{E}_{\mathrm{s}}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial \mathbf{q}_1}{\partial {\mathbf{r}_1}^{\mathrm{C}}}\frac{\partial {\mathbf{r}_1}^{\mathrm{C}}}{\partial \mathbf{t}^{\mathrm{C}}} & \frac{\partial \mathbf{q}_1}{\partial {\mathbf{r}_1}^{\mathrm{C}}}\frac{\partial {\mathbf{r}_1}^{\mathrm{C}}}{\partial \theta_{\mathrm{BC}}} \\ \dots & \dots \\ \frac{\partial \mathbf{q}_n}{\partial {\mathbf{r}_n}^{\mathrm{C}}}\frac{\partial {\mathbf{r}_n}^{\mathrm{C}}}{\partial \mathbf{t}^{\mathrm{C}}} & \frac{\partial \mathbf{q}_n}{\partial {\mathbf{r}_n}^{\mathrm{C}}}\frac{\partial {\mathbf{r}_n}^{\mathrm{C}}}{\partial \theta_{\mathrm{BC}}} \end{bmatrix}. \tag{3.21}$$

The partial derivatives in Equation 3.21 are obtained from Eqs. 3.1-3.2 and are given by:

$$\frac{\partial \mathbf{q}}{\partial \mathbf{r}^{\mathrm{C}}} = \begin{bmatrix} \frac{f_x}{z^{\mathrm{C}}} & 0 & -\frac{f_x x^{\mathrm{C}}}{(z^{\mathrm{C}})^2} \\ 0 & \frac{f_y y^{\mathrm{C}}}{z^{\mathrm{C}}} & -\frac{f_y}{(z^{\mathrm{C}})^2} \end{bmatrix} \tag{3.22}$$

$$\frac{\partial \mathbf{r}^{\mathrm{C}}}{\partial \mathbf{t}^{\mathrm{C}}} = \mathbf{R}^{\mathrm{BC}}\mathbf{p}^{\mathrm{B}} \tag{3.23}$$

$$\frac{\partial \mathbf{r}^{\mathrm{C}}}{\partial \mathbf{t}^{\mathrm{C}}} = \frac{\partial \mathbf{R}^{\mathrm{BC}}}{\partial \theta_{\mathrm{BC}}}\mathbf{p}^{\mathrm{B}}. \tag{3.24}$$

The iterative routine stops when either the improvement of the solution achieves the tolerance or the number of iterations reaches the maximum number. By applying the NRM to each of the selected best solutions generated by EPnP, new pose solutions are obtained. This set of solution candidates is used to project the 3D model using the Painter's algorithm. The nearest neighbor search is employed to match the endpoints of the line segments detected in the image with the endpoints of the projected model's line segments. The output pose solution is the one that minimizes the reprojection error (see eq. 3.17). Since the reprojection error is a measure of how well the pose solution "fits" the detected image features, it can be used as a confidence metric. A "high confidence pose" will generally have a low reprojection error compared to

a "low confidence pose". This allows the proposed method to be run on successive images until a threshold reprojection error is met.

3.4 Validation

The SVD method and its constituent subsystems are independently tested on the imagery collected during the PRISMA mission [42] to evaluate their performance and quantitatively compare their strengths and weaknesses against the state-of-the-art methods. Figure 3.11 illustrates the 3D wireframe model used in this validation effort. This model is derived by reducing a high fidelity Computer-Aided Design (CAD) model of the Tango spacecraft [1] to only contain a low number of features. The set of features present in this model were selected to reduce possible pose ambiguities (by maintaining geometric asymmetry) and to reduce the number of feature correspondence hypotheses during pose determination. It consists of a polygon representing the solar panel (560 × 750 [mm]), a convex polyhedron representing the spacecraft body (560 × 550 × 300 [mm]). Five additional segments (204 [mm]) represent the radio-frequency antennae. The origin of the body frame is located at the center of the bottom face of the spacecraft body. The model is input in MATLAB as a stereo-lithographic file from which information about surfaces and edges is generated. Notably, the same feature synthesis groups introduced in Section 3.2.2 can be extracted from the 3D model using the same condition checks and functions. Specifically, 16 proximal pairs, 18 parallel pairs, 12 parallel triads, six closed polygonal tetrads, six open polygonal tetrads, and five antennae were detected from the 3D model. In the tests that utilized datasets containing PRISMA imagery, flight dynamics products from the PRISMA mission [85] have been used for performance evaluation. Specifically, on-ground precise relative orbit determination based on GPS (accurate to about 2 [cm] 3D rms) [85] is used as the "true" relative position and onboard coarse attitude estimate from the sun-sensors and magnetometers (accurate to about 3° 3D rms) [1] is used to calculate the "true" relative attitude. The accuracy in the estimated relative position is evaluated by the following translation error

$$E_T = |\mathbf{t}^{\mathrm{C}}{}_{\mathrm{true}} - \mathbf{t}^{\mathrm{C}}{}_{\mathrm{est}}|, \tag{3.25}$$

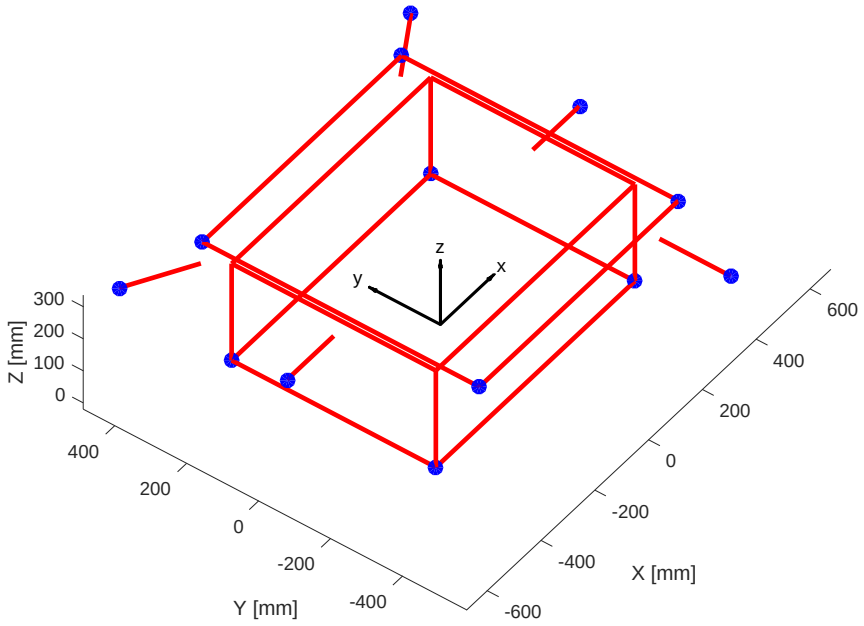


Figure 3.11: 3D wireframe model of the TANGO satellite

which represents the element-wise absolute difference between the position ${\mathbf{t}^{\mathrm{C}}}_{\mathrm{true}}$ of the target obtained from the flight dynamics products and the position ${\mathbf{t}^{\mathrm{C}}}_{\mathrm{est}}$ provided by the pose solution. Similarly, the accuracy of the attitude solution of the target vehicle is evaluated through the Euler angle representation of the rotational error

$$E_R(\mathbf{R}_{\mathrm{diff}}) = (\phi,\ \theta,\ \psi) \tag{3.26}$$

where $\mathbf{R}_{\mathrm{diff}}$ is a direction cosine matrix representing the relative rotation between the true and the estimate value of $\mathbf{R}^{\mathrm{BC}}$

$$\mathbf{R}_{\mathrm{diff}} = \mathbf{R}^{\mathrm{BC}}_{\mathrm{est}} \left(\mathbf{R}^{\mathrm{BC}}_{\mathrm{true}}\right)^T. \tag{3.27}$$

Geometrically, ϕ, θ, and ψ represent the errors in the estimated attitude about the directions C_1, C_2, and C_3, respectively. As shown in Figure 3.1, the direction C_3 is pointed along the bore-sight while C_1 and C_2 are aligned with the image frame, P. Lastly, the reprojection error, E_{2D}, of the final pose estimate is calculated using

Equation 3.17 and is reported in pixels. The reprojection error quantifies how closely the final pose estimate recreates the features detected in the image. Table 3.2 presents the purpose, image datasets, and the methods used during the four tests conducted as part of the validation effort. The following sub-sections present a detailed account of these four tests.

Table 3.2: Description of the tests conducted for the validation and comparison of the SVD method against the state-of-the-art methods.

Test name	Test purpose	Image datasets	Methods
Test 1	Extraction of line segment endpoints	PRISMA-142	FAST [86], BRISK [87], Sobel + Hough [41], Canny + Hough [74], Prewitt + Hough [77], WGE
Test 2	Detection of Region-of-Interest (ROI)	PRISMA-142	MSER [75], WGE
Test 3	Solution of perspective equations	PRISMA-5, Synthetic-5	EPnP [36], ASPnL, LPnL-DLT, LPnL-LS, LPnL-ENull [34], Ansar [35], Mirzaei [33]
Test 4	Pose initialization with unknown feature correspondences	PRISMA-25	WGE + RANSAC [54], WGE + SVD

3.4.1 End-point extraction

This test compared the performance of state-of-the-art feature extractors with that of WGE using a dataset of 142 images from the PRISMA mission (referred in Table 3.2 as PRISMA-142). The image processing subsystem is required to output pixel locations of edge endpoints that correspond to line segment endpoints of the 3D model. This can be achieved either through (1) coupling edge detectors such as Sobel [41], Canny [74], and Prewitt [77] with the Hough transform; (2) using corner detectors such as FAST [86]; or (3) using keypoint detectors such as BRISK [87]. True Positive Rate (TPR) and Positive Predictive Value (PPV) were calculated for each image in

PRISMA-142 as

$$\text{TPR} = \frac{\text{Number of true positives}}{\text{Number of true positives} + \text{Number of false positives}} \cdot 100 \tag{3.28}$$

$$\text{PPV} = \frac{\text{Number of true positives}}{\text{Number of true positives} + \text{Number of false negatives}} \cdot 100 \tag{3.29}$$

The ground truth for the edge endpoints were manually annotated on the images. A pixel location output was classified as a true positive if it was within a Euclidean distance of 5 pixels from a true edge endpoint. All output pixel locations outside this range were classified as false positives. Any true edge endpoint that remained undetected, i.e., none of the pixel locations output by the feature extractor were within a Euclidean distance of 5 pixels, was counted as a false negative. Figure 3.12 shows

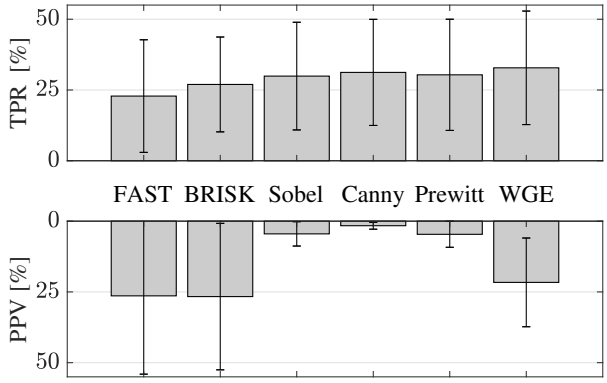


Figure 3.12: Mean and standard deviation of the True Positive Rate (TPR) and Positive Predictive Values (PPV) of the feature extraction algorithms on the PRISMA-142 dataset.

that the WGE achieved a true positive rate of 32.8%, the highest among all feature detectors tested, followed closely by Canny (31.2%) and Sobel (30.4%). However, WGE has a precision of 21.6% compared to 1.6% for Canny and 4.6% for Sobel. Moreover, this performance comes at an order of magnitude less computational time as other edge detectors since the weak gradient elimination process reduces the number of pixel locations considered by the Hough transform during feature extraction. The performance of WGE is, therefore, far superior to the feature extraction methods

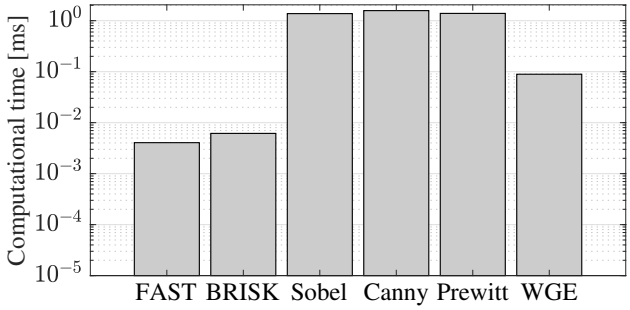


Figure 3.13: Mean and standard deviation of the computational time required by the feature extraction algorithms on the PRISMA-142 dataset.

based on edge detection. The mean PPV value of WGE in this test was lower than BRISK and FAST. However, there were zero images where WGE failed to produce a single true positive compared to 30 for FAST and 11 for BRISK. Both FAST and BRISK are an order of magnitude faster in comparison to WGE, however that is expected since WGE not only detects pixel locations of endpoints but also provides line segment information (i.e., which endpoints fall on the same edge).

3.4.2 Region-of-Interest Detection

This test compared the ROI output of WGE on the PRISMA-142 dataset with that of Maximally Stable Extremal Regions (MSER) [75]. MSER is a blob detector that outputs a list of pixel locations whereas WGE provides a bounding box in the image plane. Therefore, the pixel location list output by MSER is first converted to a list of bounding boxes. Following that, a single (or multiple in some images) ROI is obtained by applying Non-Maximum Suppression (NMS) [88] to the list of bounding boxes. The output of MSER+NMS and WGE is compared against the ROI ground truth obtained through manual annotation of the images. The PPV and TPR for each ROI are calculated by classifying its bounded area and using equations 3.28 and 3.29. Figure 3.14 shows the definitions of true positive, false positive, and true negative regions in the image.

Figure 3.15 shows some of the results obtained using WGE and MSER+NMS on the PRISMA-142 dataset. The mean values of TPR and PPV across all 142 images for WGE were measured as 90.59% and 85.98%, respectively. These values

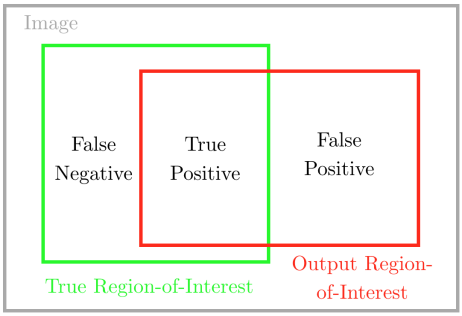


Figure 3.14: Class definitions of the area bounded by the Region-of-Interest (ROI) output by WGE and MSER+NMS

are superior to those measured for MSER + NMS (mean TPR = 89.45% and mean PPV = 81.77%). The mean computational time for MSER + NMS, as measured using `tic-toc` on MATLAB running on a 2.4 GHz Intel Core i5 machine, was 0.4867 seconds while the mean computational time for WGE was 0.0878 seconds. Moreover, as seen in Figure 3.15, MSER+NMS tends to produce multiple ROIs per image and conventionally a machine learning algorithm is required to classify whether the ROI contains the object of interest. However, multiple ROIs per image could be a desirable property if more than one satellite were present in the image. Figure 3.15 also exhibits a failure case for WGE, where the curvature of the Earth is present in the image. The output of WGE is biased since the horizon represents a sharp change in image intensity.

3.4.3 PnP versus PnL

This test compared the output of several perspective equation solvers to select one that is both accurate and computationally fast. In particular, the attitude and translation output of several PnL algorithms (ASPnL [34], LPnL [34], Ansar [35], Mirzaei [33]) is compared against EPnP [36], which is a PnP solver. The test only considers a single PnP solver since EPnP has been shown to have superior performance in comparison to other PnP solvers [23]. This test employs the use of two datasets. The first dataset contained five synthetic images that were generated by projecting the 3D model of the spacecraft (see fig. 3.11) onto the image plane using random poses. For

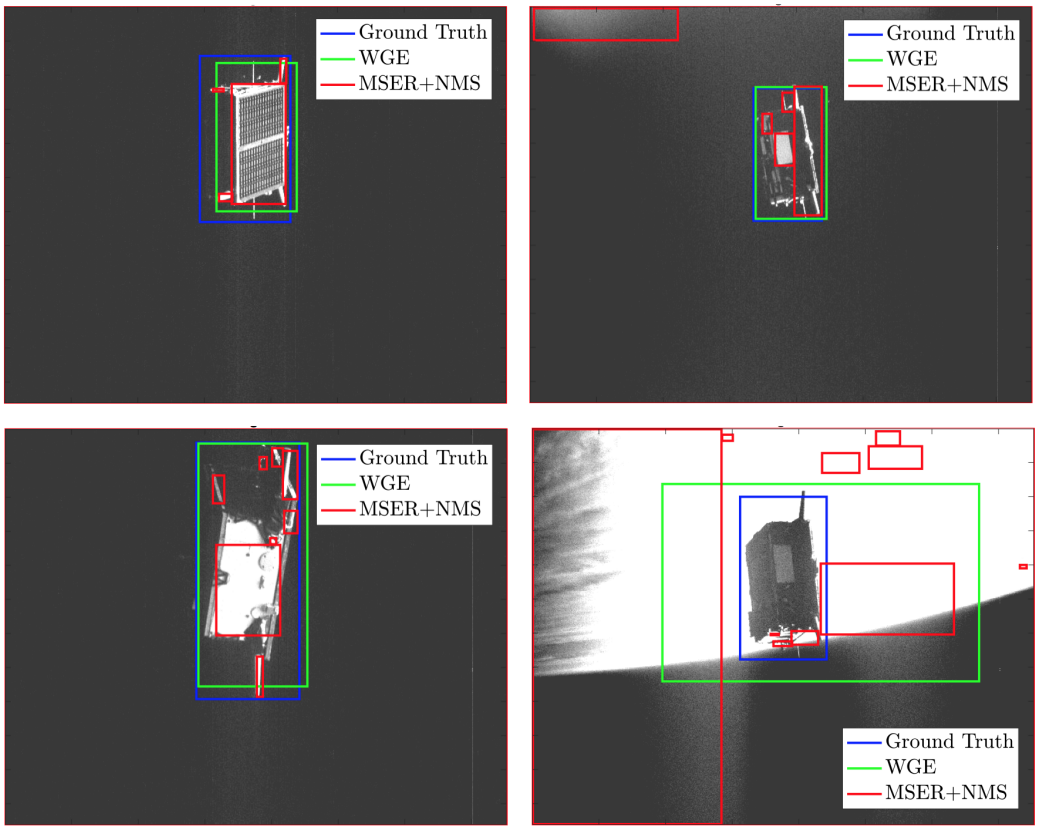


Figure 3.15: Region-of-Interest (ROI) output by WGE and MSER+NMS on a set of four images from the PRISMA-142 dataset.

each image, the PnL solvers were tested on different combinations of line correspondences. To create the combinations, several sizes of combinations were considered, and each solver was tested on all possible combinations of line correspondences of that size. Unique line segment endpoints were selected from the line correspondences as input for EPnP. The second dataset contained five actual images of the TANGO spacecraft acquired during the PRISMA mission [42]. Line correspondences between the image and the 3D model were manually selected and each algorithm was tested on several combinations of correspondences. To compare the output of the pose solvers, a success rate based on the number of "correct" pose outputs is calculated. In particular, the success rate is the fraction of pose outputs that have $||E_T||_2 < 30$ [cm] and $||E_R||_2 < 10$ [deg]. Figure 3.16 shows the success rate of the pose solvers tested

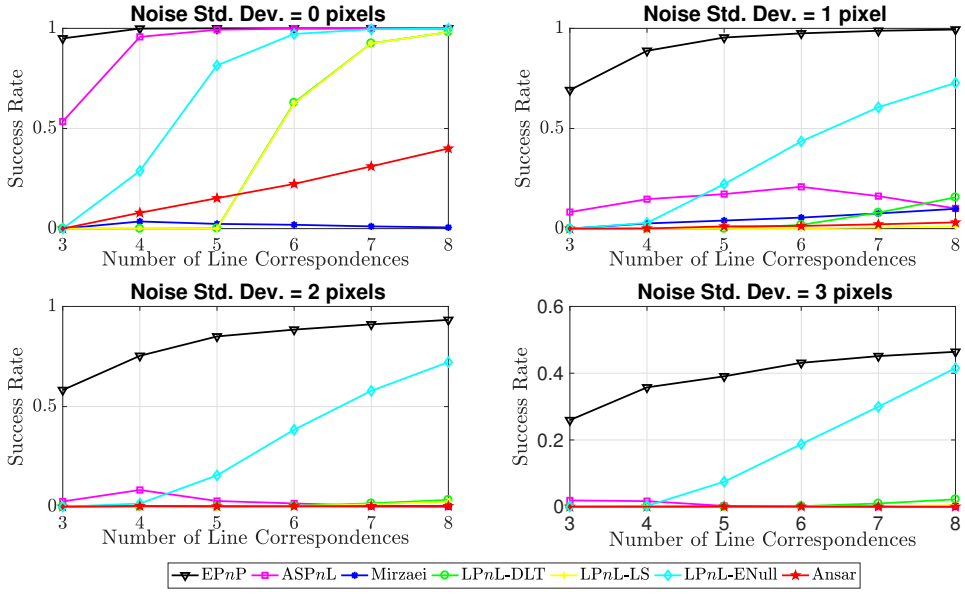


Figure 3.16: Average success rate of the pose solvers as a function of the number of line correspondences. Plots generated using five random poses and different level of noise in the projected image.

on the dataset of synthetic images. Measurement noise due to the image sensor characteristics is simulated by adding Gaussian noise with zero mean and varying levels of the standard deviation to the input of the pose solvers. EPnP was found to have the highest success rate for every size of input correspondences and noise level. The reason for the poor performance of PnL solvers is that these solvers require a large

number of line correspondences to solve the PnL problem and they are susceptible to noise primarily when few correspondences are provided. This result is not surprising since other authors have only demonstrated satisfactory results of PnL solvers with at least ten line correspondences [34]. However, the performance of EPnP also degenerates with increasing levels of measurement noise. Figure 3.17 shows the performance

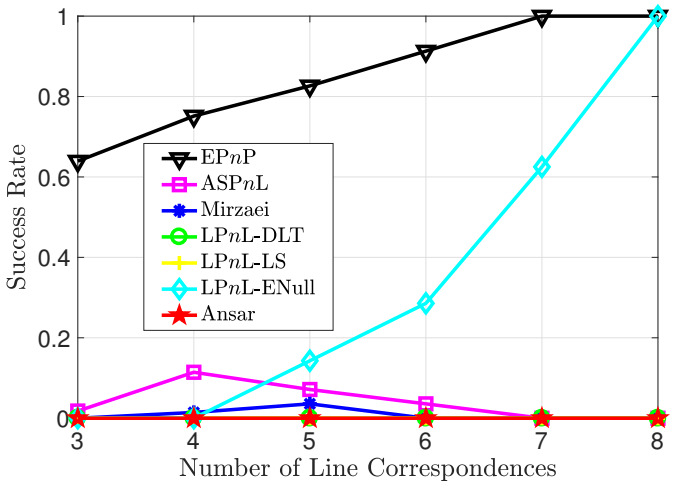


Figure 3.17: Average success rate of the pose solvers as a function of the number of line correspondences. Plots generated using five images from the PRISMA mission.

of the pose solvers tested on the dataset of five real images from the PRISMA mission. Similar to the test cases of the synthetic dataset, EPnP was again found to have the highest average success rate for all sets of feature correspondences. Note that the failure cases of EPnP with three and four feature correspondences are because coplanar point correspondences may lead to multiple pose solutions [89]. In cases where both EPnP and LPnL-Enull produced a correct pose solution, their accuracy (using $||E_T||_2$ and $||E_R||_2$) and computational runtime (using MATLAB commands *tic*, *toc*) were measured for comparison. As summarized in Table 3.3, EPnP offered a superior performance in terms of both pose accuracy and runtime. For these reasons, EPnP was chosen as the pose solver in the proposed pose initialization method.

Table 3.3: Accuracy and Computation Runtime of the pose outputs from EPnP and LPnL-ENull.

Algorithm	$\|\|E_T\|\|_2$ [m]	$\|\|E_R\|\|_2$ [deg]	Runtime [ms]
EPnP	0.23	2.7	3.0
LPnL-ENull	0.30	8.1	3.5

3.4.4 Pose Initialization

This test compared the pose estimate of the proposed pose initialization method based on the use of feature groups with one based on the use of RANSAC [54]. In particular, the test highlights the advantages and disadvantages of handling unknown feature correspondences through feature synthesis as compared to random hypotheses. RANSAC randomly samples three edges from the image and three line segments from the 3D model and provides their endpoints to EPnP for a pose solution. This pose is then verified by projecting the 3D model on the image plane and counting the number of "inlier" edges that agree with it, i.e., the number of detected edges that are closer than a predefined threshold to the corresponding projected edges. The pose solution with the highest number of inlier edges is output after k random hypotheses. The value of k is governed by

$$k = \frac{\log(1-p)}{\log(1-(\frac{w}{mn})^6)} \tag{3.30}$$

where p is the probability of randomly drawing a correct sample of corresponding 3D model and image feature points, w is the number of detected image feature points which correspond to at least one 3D model point, m and n are the total number of 3D model and image feature points, respectively. In case two pose solutions have the same number of inlier edges, the solution that has a lower median Euclidean distance between the inlier edge endpoints and their corresponding projected edge endpoints is chosen. For a consistent comparison of RANSAC with SVD, both methods used point features from the same image processing subsystem. A dataset of 25 images from the PRISMA mission was used in this test, and the pose estimates from the two methods were compared against the PRISMA flight dynamic products to calculate E_R and E_T.

Figures 3.18 and 3.19 visualize the pose solutions computed by SVD and RANSAC,

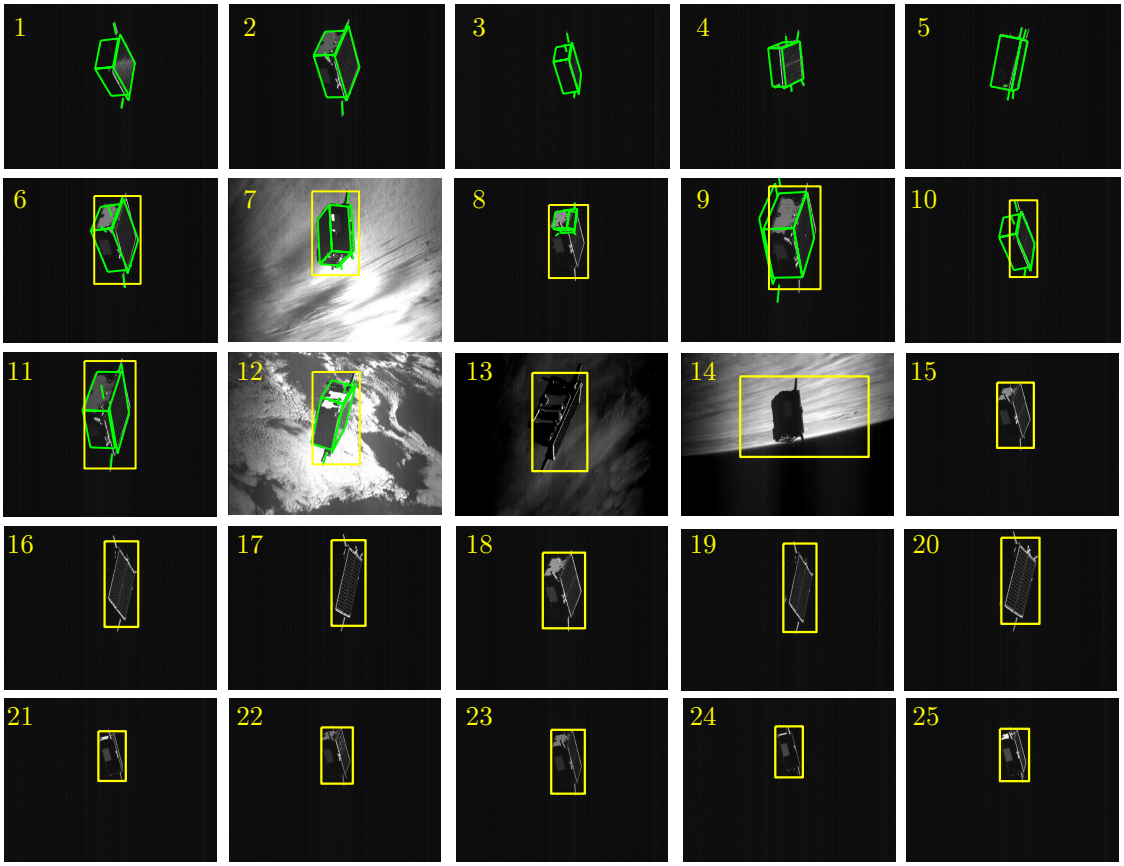


Figure 3.18: Pose initialization results using the Sharma-Ventura-D'Amico (SVD) method. Images 1-5 produced high confidence pose solutions, images 6-12 produced low confidence pose solutions, and images 13-25 produced relative position solutions only (green shows the 3D model projected on the image plane using the pose solution and yellow shows the Region-of-Interest detected by Weak Gradient Elimination).

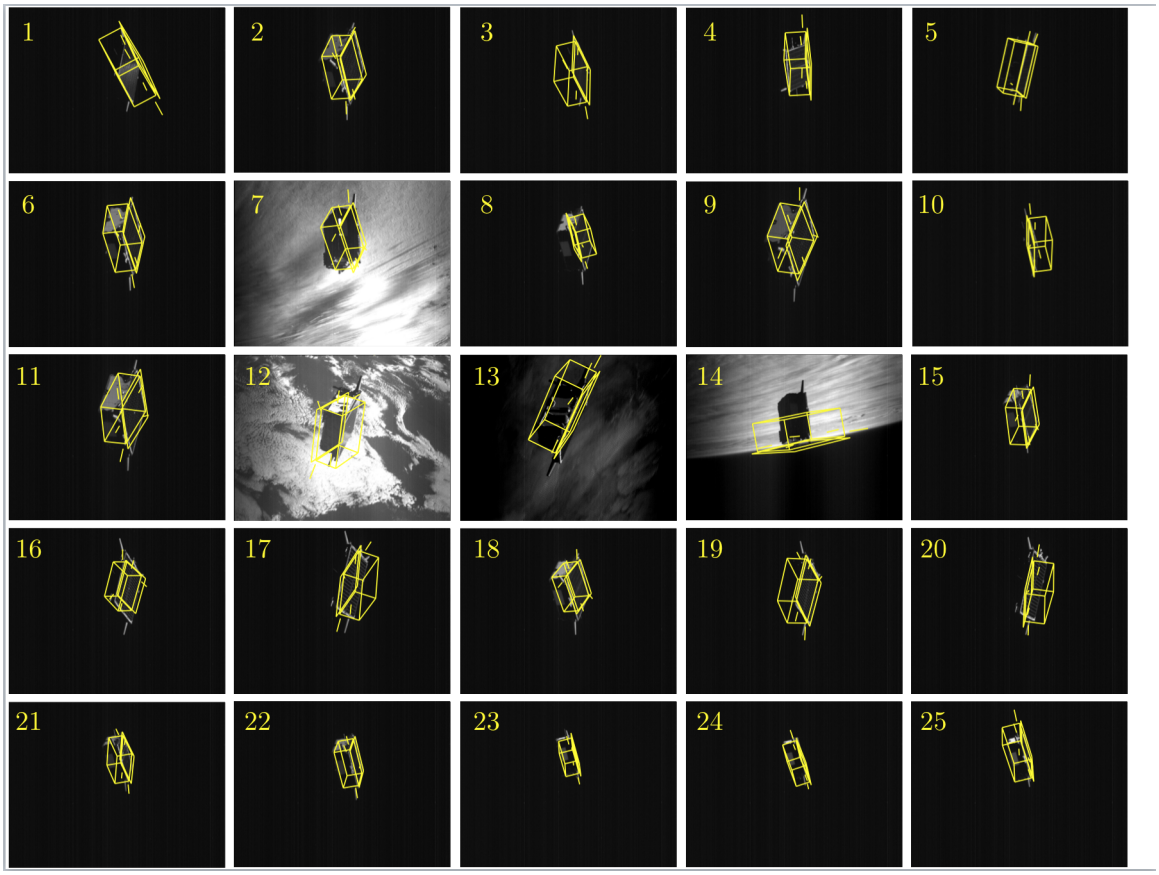
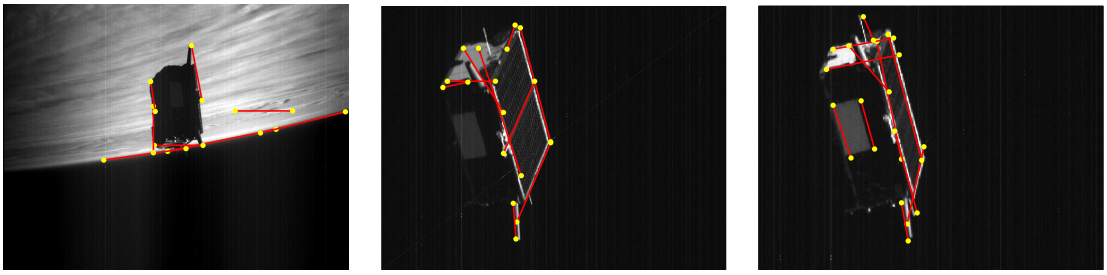


Figure 3.19: Pose initialization results using the RANSAC method (yellow shows the 3D model projected on the image plane using the pose solution).

Table 3.4: Accuracy of the pose solutions provided by SVD and RANSAC on the PRISMA-25 dataset. Values of E_R and E_T are mean values computed across the different images belonging to the particular solution type.

SVD Solution Type	Number of Images	Method	E_R [deg]	E_T [m]
High Confidence	5	SVD	(-0.57, 0.59, -1.37)	(0.14, 0.06, 0.51)
		RANSAC	(29.84, 7.52, -17.87)	(0.46, 0.52, 1.94)
Low Confidence	7	SVD	(-23.56, -0.67, 16.78)	(0.18, 0.005, 0.75)
		RANSAC	(75.66, 2.18, -22.16)	(0.28, 0.35, 1.13)
Relative Position Only	13	SVD	-	(0.07, 0.03, 0.51)
		RANSAC	(-10.04, -2.11, 23.41)	(0.24, 0.48, 1.38)

respectively while Table 3.4 presents the attitude and translation accuracy of the solutions. In particular, Table 3.4 presents the mean and standard deviation of E_R and E_T for the three classes of pose solutions provided by SVD. The method using RANSAC produced a pose solution for all 25 images. However, the accuracy of these solutions is lower as compared to SVD. As seen in Figure 3.19, the pose solutions by RANSAC were successful in aligning the detected edges with the line segments in the 3D model, producing a meter-level relative position accuracy. However, in general, this alignment of edges does not guarantee a correct relative attitude solution as visible in Figure 3.19 and Table 3.4. In contrast, the SVD method provided a high confidence pose solution for five images, a low confidence pose solution in seven images, and only a relative position solution for 13 images. The SVD high confidence pose solutions had decimeter-level relative position accuracy and degree-level relative attitude accuracy. In comparison, the relative attitude accuracy for the SVD low confidence pose solutions is reduced due to the geometric ambiguity resulting from the low number of attitude distinguishing features detected. For example, due to the geometry of the Tango spacecraft, it is impossible to distinguish between two attitude solutions that are mirrors of each other if only a single polygonal tetrad and an antenna are detected. In these cases, the "correct" attitude solution was found to be always part of the set of top five solutions input to the NRM during the pose



(a) (b) (c)

Figure 3.20: Example images where the image processing subsystem output contained spurious edges. Case (a) shows edges detected from the horizon, which did not get eliminated by the WGE technique due to their sharp intensity gradients. Cases (b) and (c) show duplicate edges as well as edges detected from parts of the spacecraft absent in its 3D model.

determination subsystem. However, after NRM the reprojection error of the "correct" solution was almost equal but slightly lower than the output solution. Lastly, SVD did not produce an attitude solution for 13 images since these images resulted in a pose solution with a high reprojection error. A common characteristic of these images is the detection of partial edges, duplicate edges, and detection of edges that are absent in the 3D model. Figure 3.20 presents three examples of these cases. In such cases, the decimeter-level accurate relative position was output using the ROI detected by the WGE technique. For both RANSAC and SVD methods, the highest uncertainty in relative position and attitude solutions was in the C_3 direction which is aligned with the camera bore-sight. The test was run on a 2.4 GHz Intel Core i5-4570T processor and made use of vectorized implementation of the SVD pose initialization method in MATLAB. The MATLAB command *tic* was used to start a stopwatch timer when the image processing begins whereas the command *toc* is used to stop the timer when the pose solution is output. The 12 images in which SVD produced a pose solution, it required 8.2163 [s] on average. The majority of the runtime was spent solving the feature correspondence problem, which is expected since the method does not rely on an a-priori pose information. In comparison, RANSAC required an average of 13.464 [s] for the same set of 12 images where SVD produced a pose solution.

3.5 Summary

This chapter has described an method for robust model-based pose initialization of noncooperative spacecraft to enable autonomous proximity operations. The detailed description of the proposed techniques is accompanied by a thorough validation effort using simulated image measurements as well as actual space imagery from the PRISMA mission to show the functional applicability and accuracy potential. The proposed method improves upon the state-of-the-art by introducing a hybrid approach to image processing by fusing the WGE technique with the Sobel operator followed by Hough transform to detect both small and large features of the target spacecraft. The hyper-parameters of the Hough transform are expressed as scalar multiples of the size of the ROI determined from the image processing subsystem, thereby alleviating the problem of manually tuning them for each image. The scalar multiples can be easily determined using on-ground simulations or onboard using prior knowledge of the inter-spacecraft range from angles-only measurements. Comparisons with independent flight dynamics operational products have shown pose accuracy for images captured during the PRISMA missions at the level of 1.5968° (3D RMS error) and 0.5322 [m] (3D RMS error). Notably, the errors in the translation and attitude in and along the bore-sight direction were the largest. On average, the SVD method required 8.2163 [s] to produce a pose solution when implemented in MATLAB on a 2.4 GHz Intel Core i5-4570T processor. Notably, up to 92% of the runtime is contributed by EPnP calls to determine the correct feature correspondences between the image and the 3D model. Therefore, there would be merit in further developing techniques to detect complex feature groups, for example, separate Hough transforms can be employed for each complex geometric shape.

Future work on feature-based methods can exploit features from subsequent images to estimate the pose if a complex feature group is not detected in the first image. Additionally, the pose determination at close-range can be aided by the estimation of the relative orbit through azimuth and elevation angle measurements made at the far-range. Moreover, once the pose has been determined for the first image, it can be used as an initial guess for the pose in subsequent images. This will lead to a dramatically lower computational runtime for the subsequent images as the expensive step of determining the feature correspondence between the image and the 3D

model does not have to be repeated. The output of the feature detection was processed to merge any partially detected line segments. This has vastly improved the quality of edge detection as compared to previous work, however, further improvements must be made since it is still susceptible to producing spurious edges. This lead to the method producing no relative attitude solution for 13 out of 25 PRISMA images. Hence, methods for image processing utilizing alternative feature types must be explored to make it more robust. The next chapter explores the use of modern learning-based methods to address this issue. Finally, future work will incorporate the use of simulated test imagery and hardware-in-the-loop experiments to train and evaluate the SVD method.

Chapter 4

CNN based pose determination

A critical drawback of the SVD method introduced in the previous chapter is that its solution availability and computational runtime were highly dependent on the success of the image processing subsystem. Moreover, the image processing subsystem required a rules-based design, which was empirically developed for a single target spacecraft geometry. It is therefore evident that the bottleneck for a pose estimation method is its image processing subsystem. This chapter explores the use of a Convolutional Neural Network (CNN) to enable pose determination and overcome the challenges posed by the use of manually defined features. In the context of this dissertation, the pose estimation method introduced in this chapter is an example of a learning-based method. In particular, this chapter aims to answer the following questions:

1. Can a "state-of-the-art" CNN be used to perform pose determination based on an offline training of synthetic images of a target spacecraft?

2. Can such a CNN be trained to be robust to deviations from the distribution of the images used during training?

3. How does such a CNN compare against conventional feature-based pose initialization techniques developed in the previous chapter?

This chapter will introduce a novel pose determination method based on the AlexNet CNN architecture [65] to answer the above questions. The method involves

discretizing the pose space and training the CNN with images corresponding to the resulting pose labels. The method will leverage transfer learning and learning based on synthetic space imagery datasets. The chapter will investigate the relationship between the performance of this method and factors such as the size of the training datasets, sensor noise, and the level of pose-space discretization. Finally, the chapter also compares the performance of this method against state-of-the-art feature-based methods, including the SVD method. This performance comparison is especially important to understand the relative strengths of the learning-based methods and feature-based methods. Further, it will allow their integration into the Spacecraft Pose Network (SPN), which is the main contribution of this research and is presented in the next chapter.

The method consists of an offline training phase and an on-line prediction phase. During the training phase, the method automatically generates several thousand synthetic images of a target spacecraft and uses them to train a CNN. During the prediction phase, the input to the method is a grayscale image of a target satellite taken at $\sim$10 [m] inter-satellite separation. The trained CNN is then used to predict a pose label corresponding to a region in the four-dimensional space. Of these four dimensions, three correspond to the attitude of the camera reference frame with respect to the target's body reference frame and one corresponds to the distance from the origin of the camera reference frame to the origin of the target's body reference frame. Note that this reduces the problem of estimating the full three-dimensional relative position to a unidimensional relative range. Practically, this implies that the architecture requires a sliding-window based approach [90] to detect the region of the image where the target is present and then use the resulting bearing angle information to re-construct the three-dimensional relative position. Since low-level features (e.g., edges, blobs, etc.) for both terrestrial and spaceborne applications can be hypothesized to be similar, the first few layers of the CNN are trained with images from the ImageNet dataset [91] while the fully connected layers of the network are trained with synthetically generated images of the Tango satellite of the PRISMA mission.

This chapter is organized as follows: Section 4.1 describes the framework for the synthetic dataset generation and the CNN architecture; Section 4.2 describes the

various combinations of datasets used for training, validation experiments, and accompanying results; and Section 4.3 presents conclusions from this study and presents directions for further work and development.

4.1 Methods

Formally, the objective is the determination of the attitude and position of the camera frame, C, with respect to the body frame of the target spacecraft, B. In particular, $\mathbf{t_{BC}}$ is the relative position of the origin of the target's body reference frame with respect to the origin of the camera's reference frame. Similarly, $\mathbf{q}(\mathbf{R_{BC}})$ is the quaternion associated with the rotation matrix that aligns the target's body reference frame with the camera's reference frame.

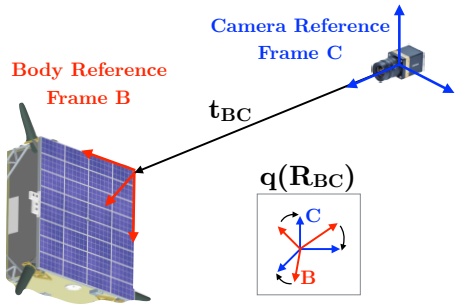


Figure 4.1: Illustration of the pose determination problem.

Training a CNN usually requires enormous labeled image datasets such as ImageNet [91] and Places [92], which contain millions of images. Collecting and labeling such an amount of actual space imagery is extremely difficult. Therefore, this work employs two techniques to overcome this limitation:

- a pipeline for automated generation and labeling of synthetic space imagery.
- transfer learning, which pre-trains the CNN on the large ImageNet dataset.

These two techniques are discussed in detail in the following subsections.

4.1.1 Synthetic Dataset Creation

The automated pipeline for generation and labeling of space imagery is based on discretizing the four-dimensional view-space around a target spacecraft. Three degrees-of-freedom result from the attitude of the target spacecraft relative to the camera and one degree of freedom results from the distance of the camera from the target spacecraft.

Uniformly locating a set of n camera locations around the target spacecraft is akin to solving for a minimum-energy configuration for charged particles on a sphere of radius r. The determination of a stable configuration of particles constrained on a sphere and being acted by an inverse square repelling force is known as the Thomson problem [93]. The solution is a set of $n(n-1)/2$ separations $s_{i,j}$ that minimizes

$$E = \sum_{i=1}^{n-1} \sum_{j=1+1}^{n} \frac{1}{s_{i,j}}. \tag{4.1}$$

Effectively, a locally optimal solution to the problem can be found by iteratively updating the particle positions along the negative gradient of E. A small mesh of camera locations generated in such a manner can be successively subdivided until n camera locations are present on the sphere. Camera locations, thus obtained, account for two of the four degrees-of-freedom in the view-space. The third degree of freedom is the rotation of the camera about the boresight direction, which can be uniformly discretized in $m-1$ intervals from zero to 360°. Finally, the degree of freedom corresponding to the distance of the camera relative to the target can be simulated by generating spheres of varying radii. Hence, the inputs to the pipeline are:

- Sphere radii, $|\mathbf{t_{BC}}|$
- Number of camera locations per sphere, n
- Number of rotations about the camera boresight per camera location, m
- 3D texture model of the target spacecraft along with the reflective properties of each of its surfaces and a coarse knowledge of the location of the illumination sources

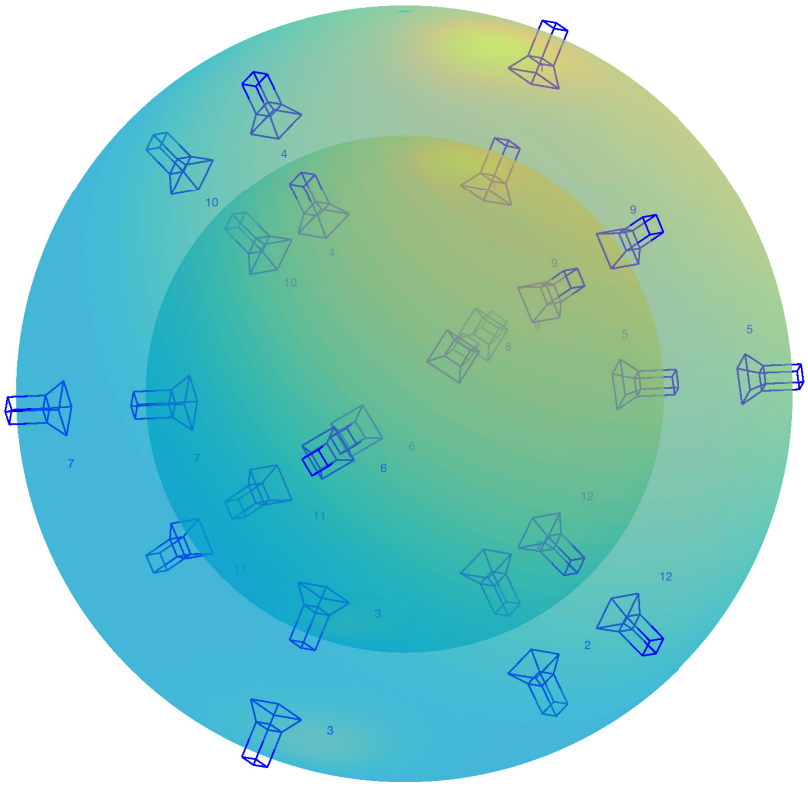


Figure 4.2: Illustration of the pose space discretization using multiple spheres with uniformly distributed camera locations. This scenario shows two spheres with ten camera locations each.

Figure 4.2 shows a mock scenario with $|\mathbf{t_{BC}}| = 2$, $n = 10$, $m = 1$. To create a total of 125,000 images for the purpose of this chapter, the following values were chosen as inputs for the pipeline: $|\mathbf{t_{BC}}| = [8, 9, 10, 11, 12, 13]$ meters, $n = 500$, $m = 50$. For each of these images, three additional copies were produced with varying levels of Zero Mean White Gaussian Noise (ZMWGN). In particular, the variance of the three levels of noise was selected as 0.01, 0.05, and 0.1 (note that image pixel intensity varies from 0 to 1). Typical images taken in spaceborne applications suffer from high levels of noise due to small sensor sizes and high dynamic range imaging. Therefore, it is imperative to create synthetic images that also possess similar noise characteristics. For each of the 125,000 noise-free images, three additional copies were created in which the target satellite was not aligned with the center of the image plane. This simulates cases where the target spacecraft is in one corner of the image plane, possibly with a few of its features outside the viewing cone of the camera. Finally, a dataset of 25 images (referred to as "Imitation-25" in Table 4.3) was rendered to imitate 25 actual images of the Tango spacecraft from the PRISMA mission [42]. Specifically, flight dynamics products from the PRISMA mission consisting of on-ground precise relative orbit determination based on GPS (accurate to about 2 [cm] 3D RMS) [85] is used as the relative position. On-board attitude estimates from the Tango spacecraft (accurate to about 3° 3D RMS) and the Mango spacecraft (accurate to about 0.1° 3D RMS) [1] are used to obtain the relative attitude. Note that the Tango spacecraft employed sun sensors and magnetometers while the Mango spacecraft employed a star tracker. Figure 4.3 shows a montage of the synthetically generated images part of this dataset compared against their real counterparts.

In total, a superset of approximately 500,000 images was created. The images were rendered using C++ language bindings of OpenGL. Although the pipeline was used to generate synthetic images of the Tango spacecraft used in the PRISMA mission [94], it can comfortably accommodate any other spacecraft. The camera field of view was selected to be 31.5 degrees, modeling after the close-range camera flown aboard the Mango spacecraft of the PRISMA mission. Table 4.1 provides the parameters of the pinhole camera model used to generate the images. Note that the generated images were resized to be 227 pixels by 227 pixels to match the input size of the AlexNet architecture [65] as well as to conserve disk space and RAM usage during

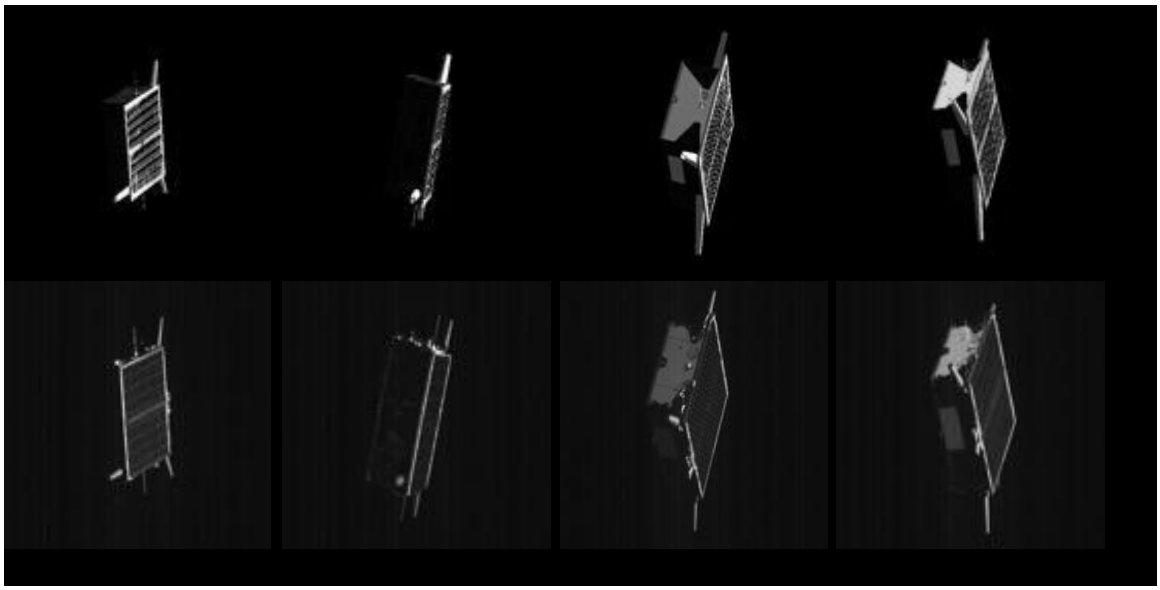


Figure 4.3: Comparison of the synthetically generated images from the Imitation-25 dataset (top row) with actual space imagery (bottom row) from the PRISMA mission. Relative position and orientation of the camera used for image generation were obtained from actual flight data for this dataset.

the training process.

After the generation of images, each image must be assigned a pose label that best approximates the true pose of the camera relative to the target spacecraft while capturing the image. This approximate pose label will be used to train the CNN with the expectation that the CNN will learn the visual features associated with the cluster of images belonging to each pose label. More importantly, it is expected that the CNN will learn the correlation between these learned features and the approximate pose label associated with those images. Since the CNN solves a classification problem to determine a pose that exists in a continuous domain, it is essential to clarify the distinction between classification and pose estimation accuracies. It is expected that the level of discretization of the pose space used during training drives the accuracy of the on-line pose estimation. Thus, the choice of the number of pose labels used during training depends on the required pose estimation accuracy. For this chapter, four levels of discretization were used resulting in 6, 18, 648, and 3000 pose labels. The pose labels were generated using the same procedure as described above for the

Table 4.1: Parameters of the camera model used to capture the synthetic images.

Parameter	Description	Value
N_u	Number of horizontal pixels	752
N_v	Number of vertical pixels	580
f_x	Horizontal focal length	0.0199 m
f_y	Vertical focal length	0.0199 m
du	Horizontal pixel length	$8.6 \cdot 10^{-6}$ m
dv	Vertical pixel length	$8.3 \cdot 10^{-6}$ m

Table 4.2: Summary of discretization levels used to generate the pose labels.

# Pose Labels	$\lvert\mathbf{t_{BC}}\rvert$ [m]	**n**	**m**
6	{3}	6	1
18	{3, 5, 9}	6	1
648	{8,9,10,11}	162	1
3000	{8,9,10,11,12}	300	10

image generation. The input values $|\mathbf{t_{BC}}|$, n, m used for each of these pose labels is presented in Table 4.2. Each generated image is then assigned to a pose label for each of the four levels of discretization using a simple search algorithm. First, all pose labels associated with the same camera distance relative to the target as the image are selected. Then, for each possible pose label, an axis-angle parametrization of the attitude change required to match the camera attitude associated with the image can be calculated. Finally, the pose label that minimizes this angular change is selected as that image's pose label. Figure 4.4 shows a visualization of the resulting distribution of 10242 camera locations into 162 pose labels.

For this chapter, this algorithm allowed the composition of the superset of 500,000 images into ten datasets. Table 4.3 presents the details for each of these datasets. Each dataset was further divided into a training, validation, and test set, which represented 60%, 20%, and 20% images of the dataset, respectively. Figure 4.5 shows a montage of images associated with four different pose labels of the Clean-648 training

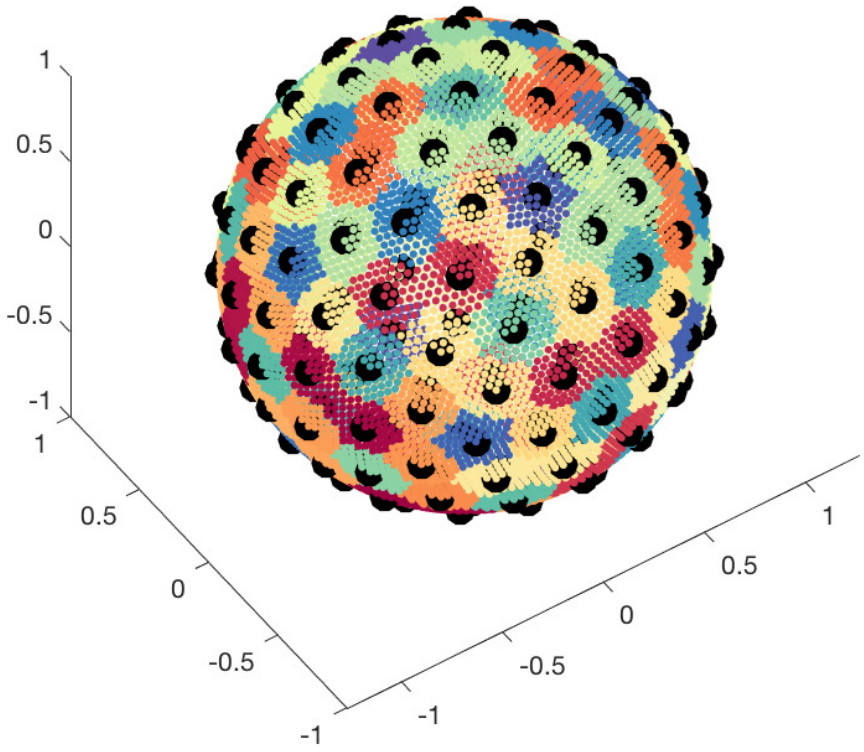


Figure 4.4: Visualization of a uniform distribution of 10242 camera locations (small colored markers) and 162 pose labels (large black markers) around a unit sphere. Camera locations associated to the same pose label are denoted by the same colored marker.

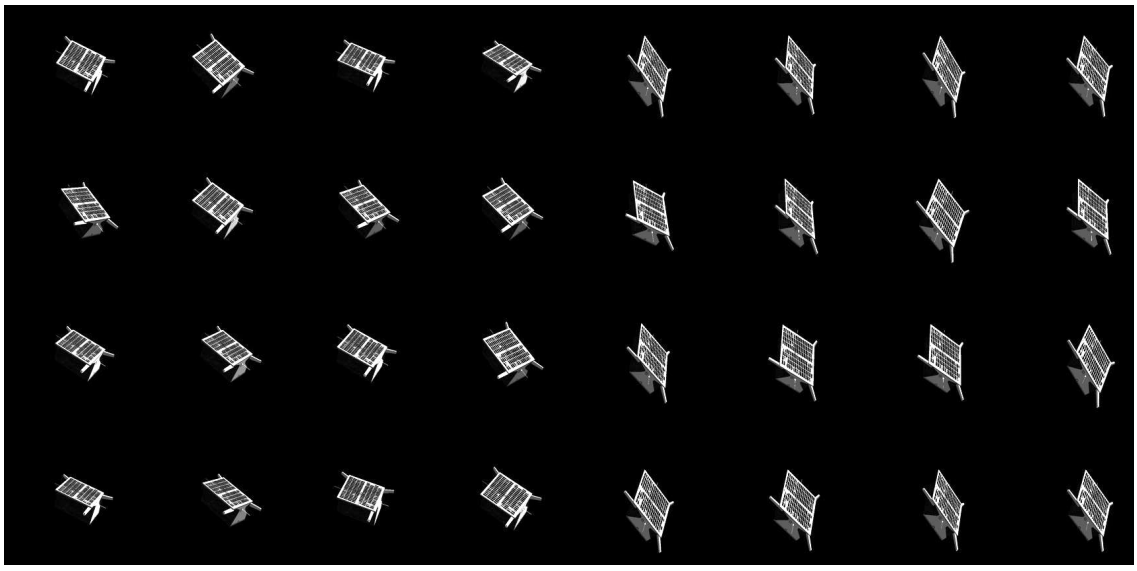
Table 4.3: Description of the ten datasets created from the synthesized images.

Dataset	Description	# Pose Labels	# Images
Clean	No noise, centered target	6	1601
Clean-18	No noise, centered target	18	4803
Clean-648	No noise, centered target	648	40968
Clean-3k	No noise, centered target	3000	125000
Noisy-3k	ZMGWN with variance of 0.01, centered target	3000	125000
Gaussian-1	ZMGWN with variance of 0.01, centered target	6	1601
Gaussian-5	ZMGWN with variance of 0.05, centered target	6	1601
Gaussian-10	ZMGWN with variance of 0.1, centered target	6	1601
Centered-1	No noise, $t = [0.2, 0.0, 3.0]$ meters	6	1601
Centered-2	No noise, $t = [0.0, 0.2, 3.0]$ meters	6	1601
Centered-3	No noise, $t = [0.2, 0.2, 3.0]$ meters	6	1601
Imitation-25	No noise, PRISMA flight data	3000	25

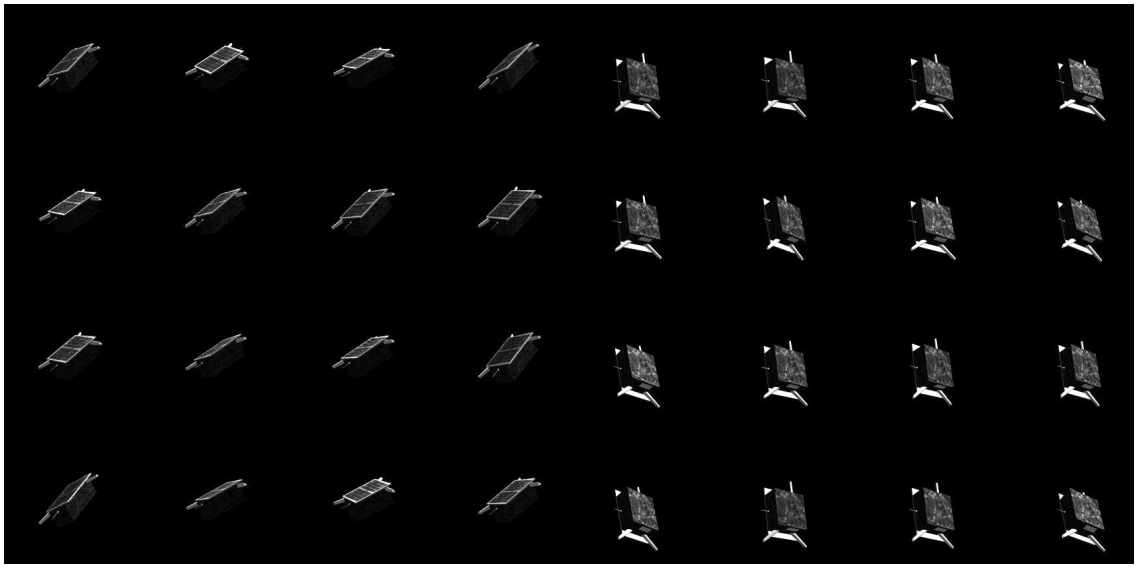
dataset.

4.1.2 Convolutional Neural Network

The CNN used in this work adopts the structure of the AlexNet architecture [65]. AlexNet was chosen over networks such as VGG-16 [95] and Inception [96] due to the relatively lower number of operations required for inference [97]. Moreover, since the number of images in the synthetically generated datasets is not as high as typical datasets used to train these networks, the proposed method relies on transfer learning. The hypothesis is that low-level features detected by the first few layers of a CNN are the same across the terrestrial and spaceborne domains. Therefore, only the parameters in the last few layers need to be determined to adapt the network for space



(a) (b)



(c) (d)

Figure 4.5: Montage of a few images from four different pose labels of the Clean-648 dataset.

imagery. The AlexNet architecture was used to train eight networks with varying sizes and compositions of the training set. The description of the eight networks is presented in Table 4.4.

Table 4.4: Description of the eight networks trained for this work. Note that the columns represent the number of training images used from the particular dataset.

Network	# Pose Labels	Clean	Gaussian-1	Clean-18	Clean-648	Clean-3k	Noisy-3k
net1	6	873	0	0	0	0	0
net2	6	728	0	0	0	0	0
net3	6	582	0	0	0	0	0
net4	6	436	0	0	0	0	0
net5	6	873	436	0	0	0	0
net6	18	0	0	2619	0	0	0
net7	648	0	0	0	24581	0	0
net8	3000	0	0	0	0	75000	0
net9	3000	0	0	0	0	75000	75000

The AlexNet architecture is shown in Figure 4.6, it contains eight layers with weights, the first five are convolutional layers and the remaining three are fully-connected layers. Note that except for net9 (see Table 4.4), all networks involved training the three fully-connected layers. For net9, the fifth convolutional layer was also trained. The output of the last fully-connected layer is used in an x-way softmax loss function which produces a distribution over the class labels (where x is the number of pose labels in the dataset used to train the network). The formula to compute the softmax loss function is presented below in equations 4.2 and 4.3. The network maximizes the multinomial logistic regression objective, which is equivalent to maximizing the average across training cases of the log-probability of the correct label under the prediction distribution.

$$\text{Loss}_{\text{softmax}} = \sum_{i=1}^{x} L_i \tag{4.2}$$

where

$$L_i = -\log\left(\frac{e^{f_{y_i}}}{\sum_j e^{f_j}}\right) \tag{4.3}$$

In Equation 4.3, the notation f_j refers to the j-th element of the vector of values output by the last fully-connected layer, f.

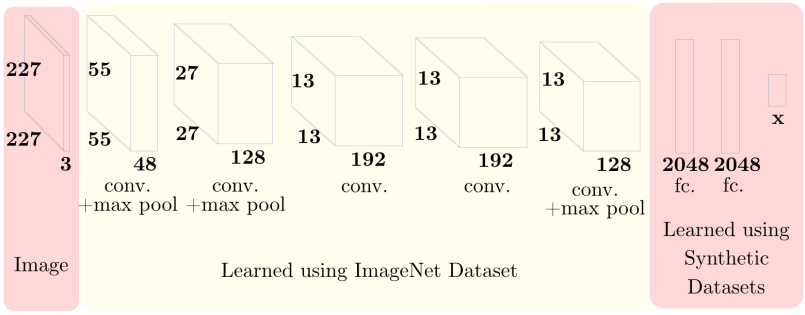


Figure 4.6: An illustration of the AlexNet architecture, which was used as the baseline for all eight networks in this chapter. The network's input is 154587-dimensional, and the number of neurons in the network's remaining layers is given by 145200–93312–32448–32448–21632–2048–2048–x. The last layer contains as many neurons as the number of pose labels in the dataset used to train the particular network.

The "dropout" technique [98] was used while training the fully connected layers. This technique consists of setting to zero the output of each hidden neuron with a probability of 0.5. The "dropped" neurons do not contribute to the forward pass and do not participate in back-propagation. The output of the remaining neurons is scaled by a factor of 2 such that the expected sum remains unchanged. This technique reduces the possibility of co-adaptations of neurons, i.e., neurons cannot rely on the presence of other particular neurons but instead learn more robust features.

4.2 Experiments

This section presents the results of two types of experiments. The first type of experiments was carried to understand the training process of a CNN on small sets of images and evaluating the network's test accuracy as a factor of:

- Number of images used in training

- Amount of ZMGWN in test images
- Amount of displacement of the target from the center of the image plane

The second set of experiments were carried to understand the feasibility of training CNN's for a realistic on-orbit servicing mission and evaluating its performance on imagery imitating actual space imagery from the PRISMA mission. Both of these experiments and their results are discussed in the following subsections.

4.2.1 Type 1

The Type 1 experiments involved comparing the performance of net1, net2, net3, net4, net5, and net6 on the Clean-6, Clean-18, Gaussian-1, Gaussian-5, Gaussian-10, Centered-1, Centered-2, and Centered-3 datasets. The comparison is based on the accuracy of the predictions and the F-Measure (FM) of the classifications. The results presented here are based on testing the networks on the test set of the datasets described in Table 4.3. In particular, the accuracy is defined as the percentage of the test images that were correctly classified by the network. FM of the classifications is based on precision and recall. These metrics are based on the number of false positives (FP), false negatives (FN), and true positives (TP) over several samples. To compute these values, each class is treated as a binary classification problem, defining a positive sample when it belongs to that class, and negative otherwise.

$$\text{precision} = \frac{TP}{(TP + FP)} \tag{4.4}$$

$$\text{recall} = \frac{TP}{(TP + FN)} \tag{4.5}$$

These are then used to calculate the F-Measure:

$$FM = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{4.6}$$

There are several trends as seen in Figure 4.7, which shows the classification accuracy of five separately trained networks on six different datasets. Firstly, all networks are more or less equally capable in classifying images in the Clean-6 dataset

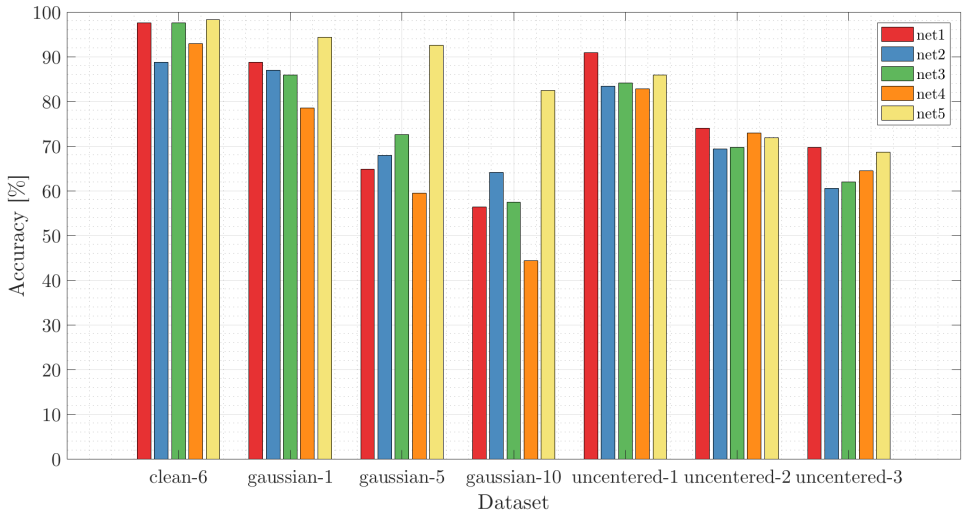


Figure 4.7: Classification accuracy [%] for seven datasets using five separate networks.

where images are free from ZMGWN and the target is centered in the image plane (all networks trained in this work were trained with centered targets). This shows that the networks have a vast number of parameters and can easily over-fit the features seen in the images. Secondly, as the ZMGWN is added, all networks show a decline in the classification accuracy. Notably, "net5" fares quite well as compared to the other networks as it used some noisy images during training. This implies that as long as sensor noise is known and can be modeled beforehand, the CNN can be made to be more robust to noise through the augmentation of the training data with noise. Thirdly, the classification accuracy of the networks correlates with the number of training images used during the training since "net1", which was trained with more images than "net2", "net3", and "net4" has higher accuracy for Clean-6, Uncentered-1, Uncentered-2, and Uncentered-3 datasets.

Lastly, "net6" was trained and tested using the Clean-18 dataset without data augmentation. The network produced a classification accuracy of 99.4%, which was significantly higher than any other networks trained on smaller datasets with data augmentation. The test set images of Clean-18 were embedded according to their features from the penultimate fully connected layer to visualize how the network had learned to separate the 18 different classes. The t-Distributed Stochastic Neighbor

Embedding (t-SNE) technique was used for dimensionality reduction [99]. This technique represents images as nodes in a graph and arranges the nodes in two dimensions in a manner that respects the high-dimensional L2 distances between their features. In other words, t-SNE arranges images that have similar features nearby in a 2D embedding. This can be visualized in Figure 4.8 for the Clean-18 test set images, where images from three different inter-satellite ranges are represented by different marker types. It can be easily seen that the network has learned to differentiate images from separate ranges. Moreover, for each range, certain classes are learned to be closer as compared to the others. This is to be expected since for example, two pose labels visually do look similar (in fact, they are close to being horizontal mirrors of each other).

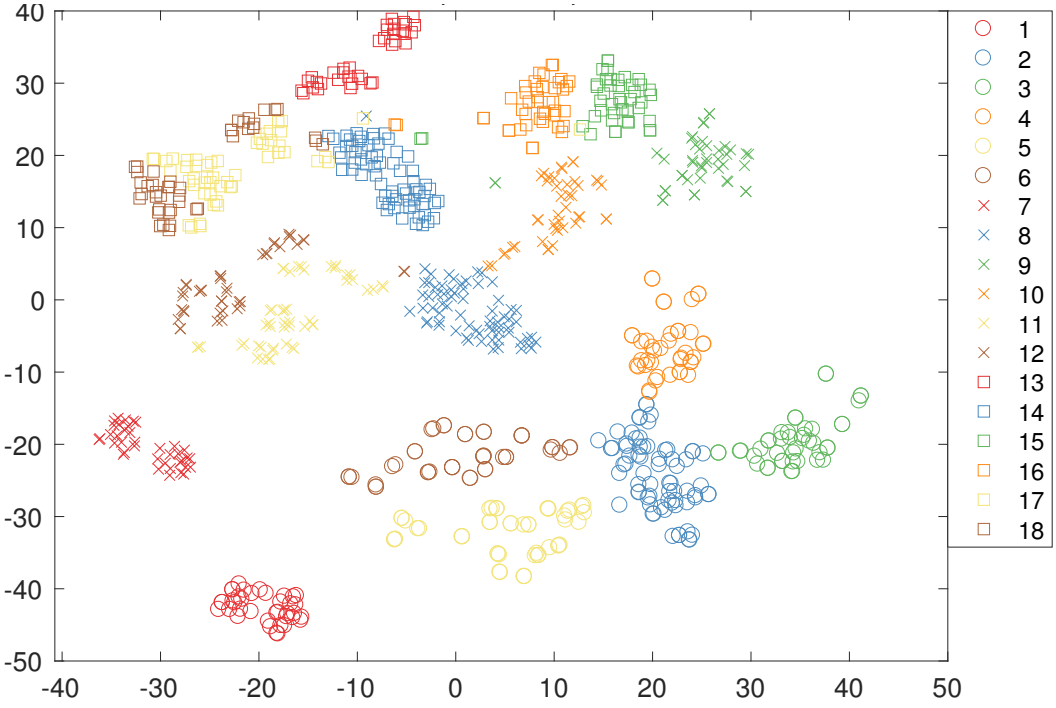


Figure 4.8: The t-Distributed Stochastic Neighbor Embedding (t-SNE)representation of the Clean-18 test set images.

Table 4.5: Performance of the net7 and net8 networks on test sets of the Clean-648 and Clean-3k datasets, respectively.

Metric	net7	net8
Mean E_R (deg)	22.91	11.94
Mean E_T (m)	0.53	0.12
Mean Classification Accuracy (%)	83.3	35

4.2.2 Type 2

The Type 2 experiments involved evaluating the performance of net7, net8, and net9 on the Clean-648, Clean-3k, and Imitation-25 datasets. The goal of these experiments was to stress-test all critical aspects of this method, from conceiving pose labels, to training and testing. Unlike the Type 1 experiments, these experiments were run with a high number of pose labels and large training datasets to achieve a higher pose estimation accuracy. In particular, the pose estimation accuracy is defined by two metrics: E_R and E_T, which are differences between true and estimated values of relative attitude and position, respectively. In particular,

$$E_T = \|\mathbf{t}_{\mathbf{BC},\text{est}} - \mathbf{t}_{\mathbf{BC},\text{true}}\| \tag{4.7}$$

$$E_R = 2\cos^{-1}(z_s), \text{where} \tag{4.8}$$

$$\mathbf{z} = [z_s \ \mathbf{z_v}] = \mathbf{q}_{\text{true}} * \text{conj}(\mathbf{q}_{\text{est}}).$$

Here $\mathbf{q}_{\text{true}}$ and $\mathbf{q}_{\text{est}}$ are true and estimated values of the quaternion associated with the rotation matrix that aligns the target's body reference frame and the camera's reference frame. Table 4.5 shows the test set accuracy for net7 and net8 on the Clean-648 and Clean-3k datasets, respectively. Note that net7 has a much higher classification accuracy as compared to net8 since it only needs to pick the correct pose label out of a set of 648 pose labels compared to 3000 for net7. Also, net7 was trained on the Clean-648 dataset, which contained approximately 45 images per pose label as compared to 25 images per pose label for the Clean-3k dataset used for net8. However, due to the larger number of classes in Clean-3k dataset, net8 produced higher pose estimation accuracy compared to net7. Since the output of the

Table 4.6: Performance of net7, net8, and net9 networks compared against conventional pose determination methods using the Imitation-25 dataset.

Method	Mean E_R (deg)	Mean E_T (m)	Solution Availability (%)	Mean runtime (s)
net7	82.18	1.79	100	0.014
net8	24.72	0.54	100	0.016
net9	17.12	0.44	100	0.016
SVD	38.99	1.46	50	8.22
EPnP + RANSAC	140.71	1.45	100	13.46
net8 (high conf.)	14.67	0.61	48	0.016
net9 (high conf.)	16.48	0.43	86	0.016
SVD (high conf.)	2.76	0.51	20	8.93

fully connected layer of the net8 is used in a 3000-way softmax (648-way softmax for net7), the values can be interpreted as the probability of the image being associated to each pose label. Further, this allows the setting up of a confidence metric to classify the pose solutions. For example, a pose solution can be classified as "high confidence" if the ratio of the highest and the next-highest probability values is greater than 2. Figures 4.9 and 4.10 present a few of these high and low confidence pose solutions provided by net8 on the Imitation-25 dataset. Note that the Imitation-25 dataset was generated using the PRISMA flight dynamics products, independent of the datasets used in training and validating these networks.

Table 4.6 shows the pose estimation accuracy of the high confidence solutions provided by net8 and net9 alongside all solutions provided by net7, net8, and net9 on the Imitation-25 dataset. Table 4.6 also shows the pose estimation accuracy of two other methods, namely, the Sharma-Ventura-D'Amico (SVD) architecture [9] and an architecture based on the EPnP [36] and RANSAC algorithms [54]. These two methods rely on the conventional method of hypothesizing and verifying poses based on the extraction of edge features from the image. Predictably, net8 and net9 have much higher accuracy compared to net7 due to the finer discretization of the pose space in the Clean-3k and Noisy-3k datasets as compared to the Clean-648 dataset. Further, net9 has a higher accuracy as compared to net8. Note that net9 was trained on twice the number of images as compared to net8, with half of the

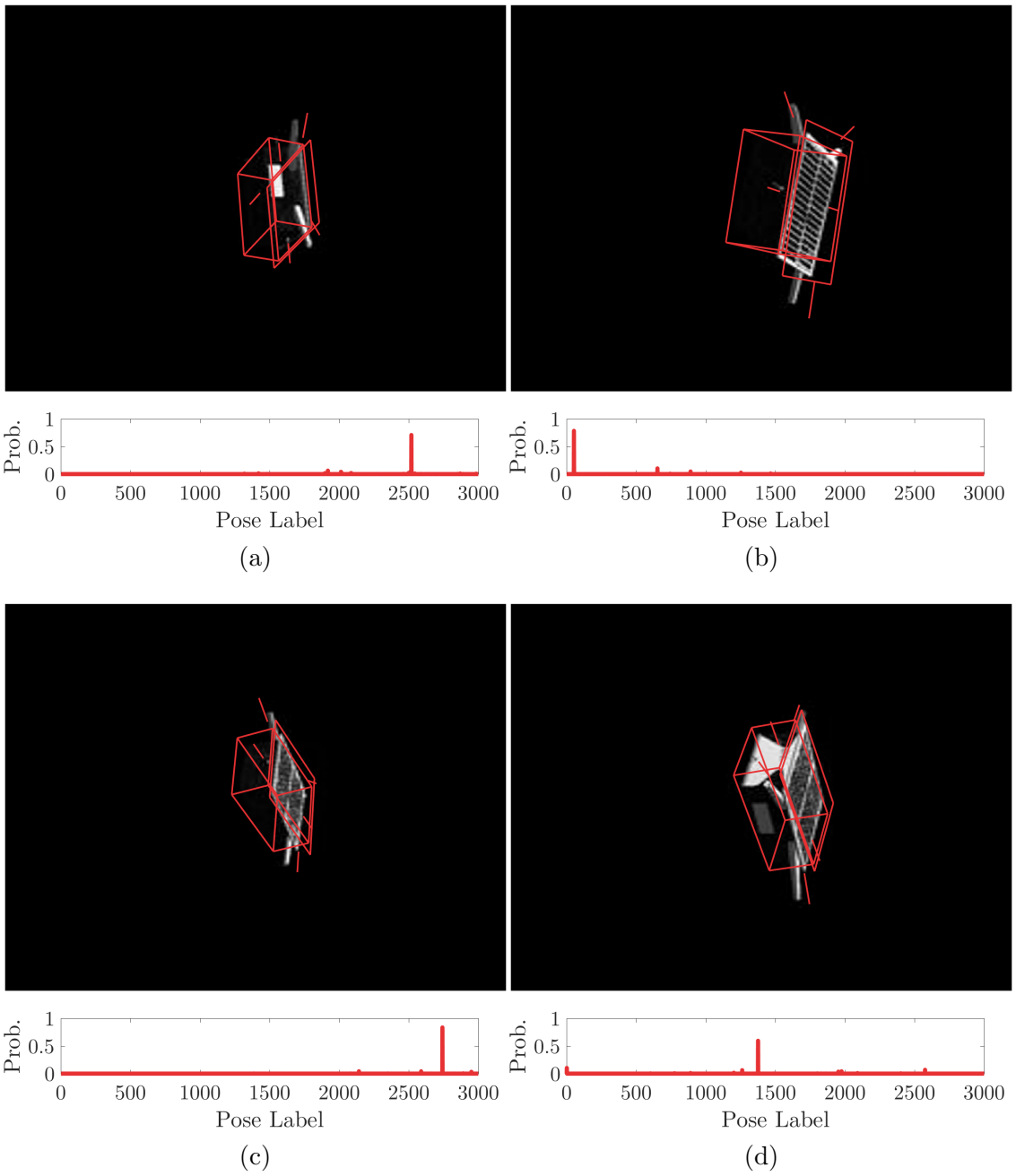


Figure 4.9: Montage of a few images from the high confidence pose solutions produced by net8 on the Imitation-25 dataset.

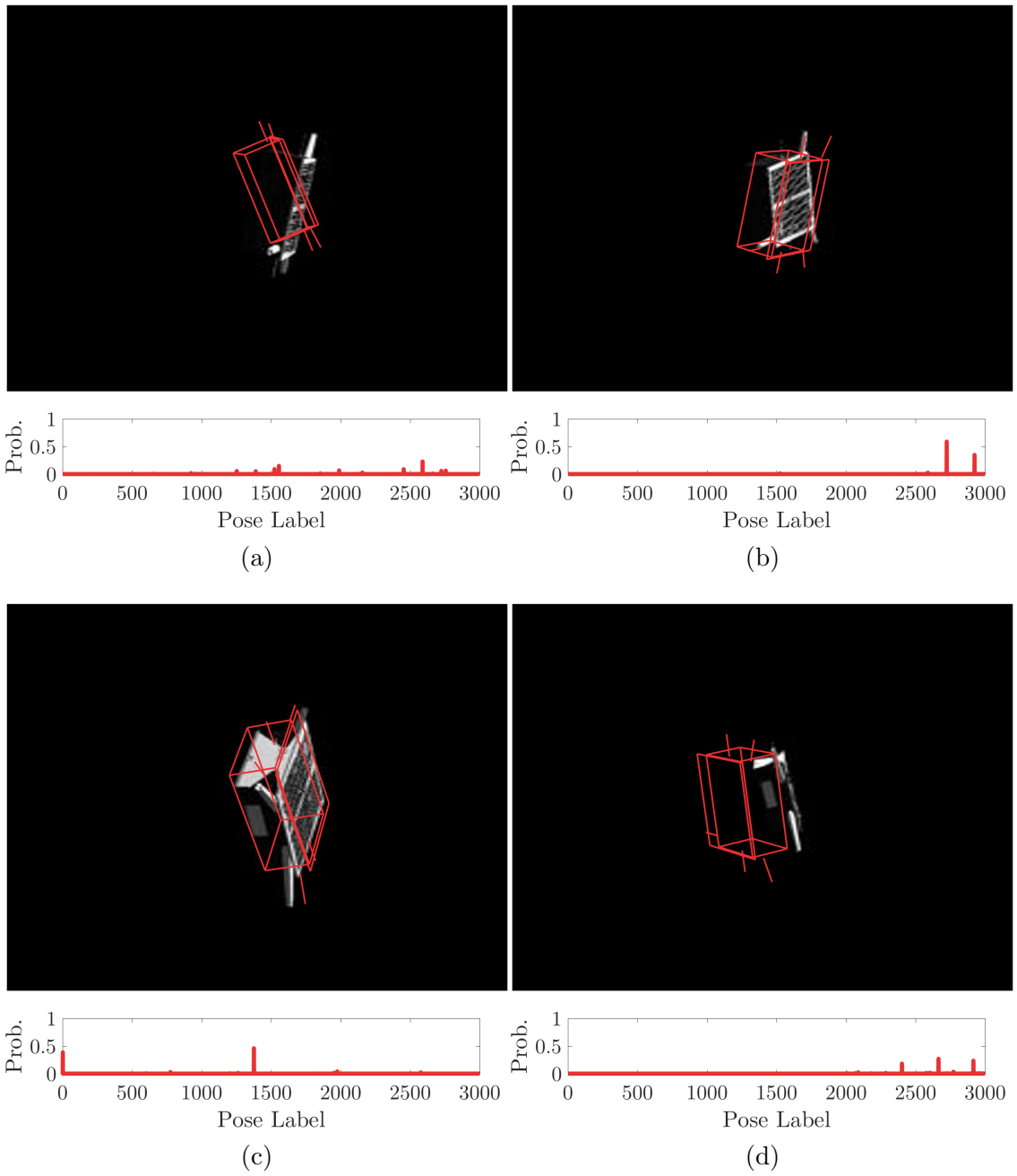


Figure 4.10: Montage of a few images from the low confidence pose solutions produced by net8 on the Imitation-25 dataset.

images containing ZMGWN. This also shows a plateauing effect that adding more images into the training process has on the pose estimation accuracy. Interestingly, net9 is also more accurate than the SVD and EPnP + RANSAC methods. However, when considering the high confidence solutions of each method, the SVD method produces the most accurate relative attitude while net9 produces the most accurate relative position. The relative attitude accuracy of net9 is limited by the level of discretization in the pose space. The mean angular separation between the closest pairs of quaternions used in training net9 is approximately 15 degrees. Also, note that the high confidence solutions of SVD are only available on 20% of the images of the Imitation-25 dataset whereas net9 provides a high confidence solution on 86% of the images. This suggests the potential fusion of net9 (or similar CNN based approaches) and the SVD method (or similar feature-based approaches) to provide high confidence and highly accurate relative pose estimates. Finally, the mean computational runtime for each method is also presented. The CNN-based approaches dominate the feature-based approaches in terms of computational runtime performance. Note that net8 and net9 have slightly higher runtimes than net7. This is likely due to both these networks containing more learnable parameters in their last fully connected layers to account for a finer discretization of the pose space. Each method was run on an Intel® Core™ i5-8400 CPU with a base frequency of 3.60GHz. While these computational runtimes will be when these methods are embedded in a flight-grade CPU, the presented runtimes are only used to make conclusions for the relative performance of the methods.

4.3 Summary

In this chapter, a framework for pose determination using convolutional neural networks was successfully set up and exhaustively tested against different datasets. Some interesting conclusions can be drawn from these experiments, which will be used as the building blocks for the development of the SPN method in the next chapter. Firstly, the size of the training set is shown to correlate with classification accuracy positively, and the level of discretization of the pose space is positively correlated with the pose estimation accuracy. This warrants the generation of more massive datasets that captures more diversity in the pose space. Secondly, as compared to networks

trained on noise-free images, the classification accuracy of the network trained with images containing small amounts of Gaussian white noise had a better performance on test images containing high amounts of Gaussian white noise. This proves that as long as the sensor noise could be modeled beforehand or removed using pre-processing techniques, CNN has excellent potential for pose determination using actual space imagery. Thirdly, all networks were trained using the transfer learning approach, which only required training of the last few layers. This proves that several low-level features of spaceborne imagery are also present in terrestrial objects, and there is no need to train a network entirely from scratch for spaceborne applications. Lastly, the network trained using 150e3 images associated with 3000 pose labels showed the highest pose estimation accuracy on the Imitation-25 dataset. Its accuracy was better than methods based on classical feature detection algorithms. As compared to the best-performing feature detection based algorithm, the network provided high confidence solutions for four times as many images. Therefore, the network has the potential to be used in conjunction with the current state-of-the-art pose determination algorithms.

However, there are several caveats in the presented work and significant potential for future development and enhancements. Firstly, the networks need to be tested not just using synthetic imagery but also actual space imagery. To train the CNN for this task, the current image rendering pipeline needs to be improved to have finer control of solar illumination, spacecraft reflectance properties, and rendering of Earth in the background. Further, a larger dataset containing real images is required for a comprehensive comparative assessment of the CNN-based methods with the conventional feature-based methods. Secondly, the pose estimation problem was cast into a classification problem in this chapter. This simplification theoretically limits the accuracy of the current CNN architecture due to the discretization of the pose space. Lastly, the current CNN-based method needs to be augmented to predict a relative range instead of a relative position.

Chapter 5

Spacecraft Pose Network (SPN)

A critical drawback of the CNN-based pose determination method introduced in the previous chapter is that its pose estimation accuracy was limited by the level of discretization of the pose space used during training. It also lacked the prediction of a relative position, opting instead to predict a relative range to simplify the problem. Secondly, the images generated by the synthetic image generation pipeline could not render the Earth in the background. This meant that the CNN-based pose determination method could not be trained to be robust to the background. Lastly, the method lacked a quantification of the uncertainty in its pose prediction. This is important since the goal of this dissertation is to provide a pose prediction subsystem that can be used to initialize or fuse with an onboard relative navigation filter. To overcome the above challenges, this chapter explores the augmentation of the image generation pipeline and the fusion of a Gauss-Newton algorithm with a deep convolutional neural network. In particular, this chapter aims to answer the following questions:

1. Can the accuracy of a CNN-based pose determination method be made independent of the pose space discretization used during training?

2. Can the synthetic image generation pipeline that was developed in the previous chapter be augmented to render high fidelity textures of the Earth in the background?

3. Can the uncertainty in the pose output of a CNN-based pose determination be quantified?

To answer the above questions, this chapter will introduce the Spacecraft Pose Network (SPN) method, a new pose estimation method that integrates the strengths of learning-based methods with those of feature-based methods. In particular, the SPN method uses a deep convolutional neural network to leverage the critical strength of learning-based methods, i.e., their inherent feature extraction capabilities that do not require manual hand-tuning or definition of particular features. The SPN method uses a Gauss-Newton algorithm to leverage the critical strength of feature-based methods, i.e., their direct use of the underlying physics of the pose estimation problem described by the perspective equations. This chapter will also introduce the Spacecraft Pose Estimation Dataset (SPEED), which fuses software-based synthetic imagery with actual camera imagery for training, validation, and testing of pose determination methods.

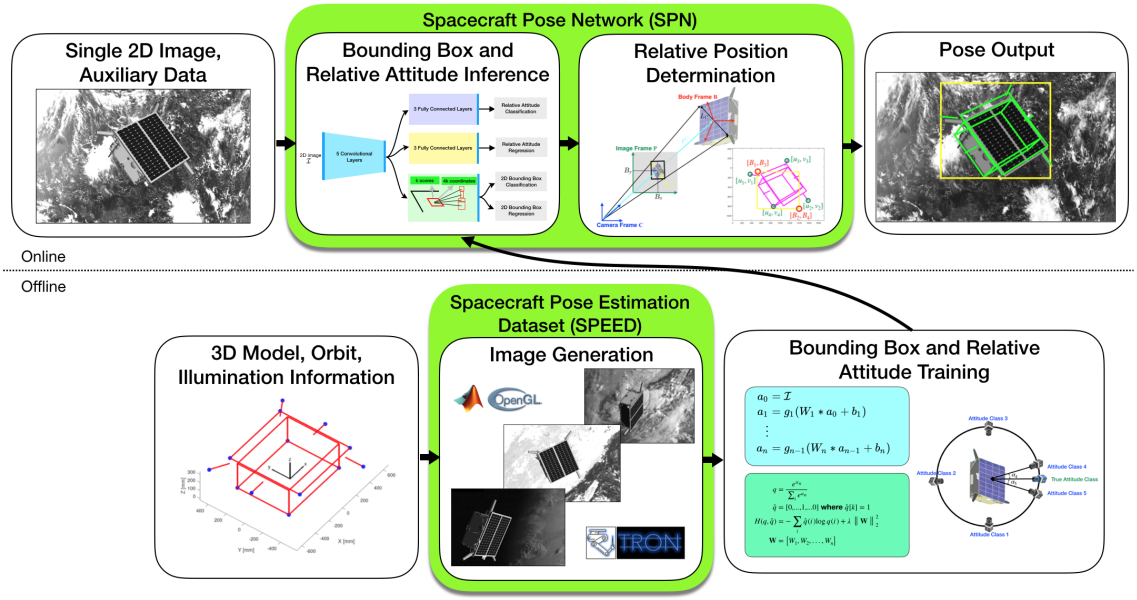


Figure 5.1: Modules of the proposed SPN method, which takes as input a 2D image and a trained convolutional neural network for relative attitude and position determination.

As shown in Figure 5.1, the SPN method uses a convolutional neural network to estimate the relative position and relative attitude in a decoupled fashion from a single grayscale image. One branch of the convolutional neural network is used to detect a 2D bounding box around the target spacecraft in the input image. The other two

branches are used to estimate the relative attitude using a hybrid discrete-continuous method. The relative attitude and the 2D bounding box are then combined with geometrical constraints of the perspective transformation to estimate the relative position using the Gauss-Newton algorithm. In contrast to current deep learning-based techniques, the relative attitude accuracy provided by the SPN method is not limited by the level of discretization of the pose space and it explicitly uses the geometrical knowledge of the perspective transformation in the estimation of relative position. As compared with techniques comparing image features with features of a known target spacecraft model onboard, the SPN method provides a pose estimate from a single image without requiring a long initialization phase or favorable relative translation motion. Further, due to the decoupling of the relative attitude and position estimation and the use of transfer learning, the SPN method can be used to infer the pose of the target spacecraft through training on a relatively small number of synthetic images of the same spacecraft.

SPEED consists of high-fidelity imagery involving close-range operations around a tumbling spacecraft. The dataset contains 15,300 images from two sources: a purely software-based Augmented Reality (AR) source that fuses synthetic and actual space imagery, and a purely reality-based source that uses an actual camera sensor to capture images of a mock-up spacecraft under high-fidelity illumination conditions. The convolutional neural network of the SPN method is trained only on a portion of these AR images while it is tested on the remaining AR images as well as the actual camera images of SPEED. SPEED was made publicly available to the community to enable a fair comparison of the state-of-the-art pose estimation techniques through a competition on pose estimation for spaceborne applications organized in collaboration with the European Space Agency[100].

This chapter is organized as follows: Section 5.1 describes the framework for the SPN method; Section 5.2 describes the SPEED image generation; Section 5.3 describes the experiments conducted to validate the performance of SPN method; and Section 5.4 presents conclusions from this chapter.

5.1 Architecture

Formally, the problem statement for this work is the estimation of the attitude and position of the camera frame, C, with respect to the body frame of the target spacecraft, B. As shown in Figure 5.2, $\mathbf{t_{BC}}$ is the relative position of the origin of the target's body reference frame w.r.t. the origin of the camera's reference frame. Similarly, $\mathbf{q_{BC}}$ is the quaternion associated with the rotation that aligns the target's body reference frame with the camera's reference frame.

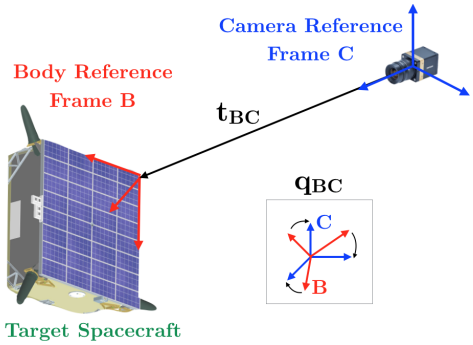


Figure 5.2: Definition of the reference frames, relative position, and relative attitude.

As shown in Figure 5.3, the SPN method uses three separate branches of a convolutional neural network to estimate $\mathbf{t_{BC}}$ and $\mathbf{q_{BC}}$. Branch 1, which is based on the region proposal network[101], is used to detect a 2D bounding box in the image around the target spacecraft. The output of the convolutional layers corresponding to the detected 2D bounding box is used as an input to Branches 2 and 3 to obtain an estimate of the relative attitude, $\tilde{\mathbf{q}}_{\mathbf{BC}}$. Finally, the 2D bounding box and $\tilde{\mathbf{q}}_{\mathbf{BC}}$ are input to the Gauss-Newton algorithm to estimate the relative position as $\tilde{\mathbf{t}}_{\mathbf{BC}}$. The estimation of the relative position and relative attitude are discussed in detail in the following subsections.

5.1.1 Relative Position Estimation

Bounding Box Detection

The relative position estimation begins with the detection of a 2D bounding box in the image around the target spacecraft using the region proposal network [101]. The

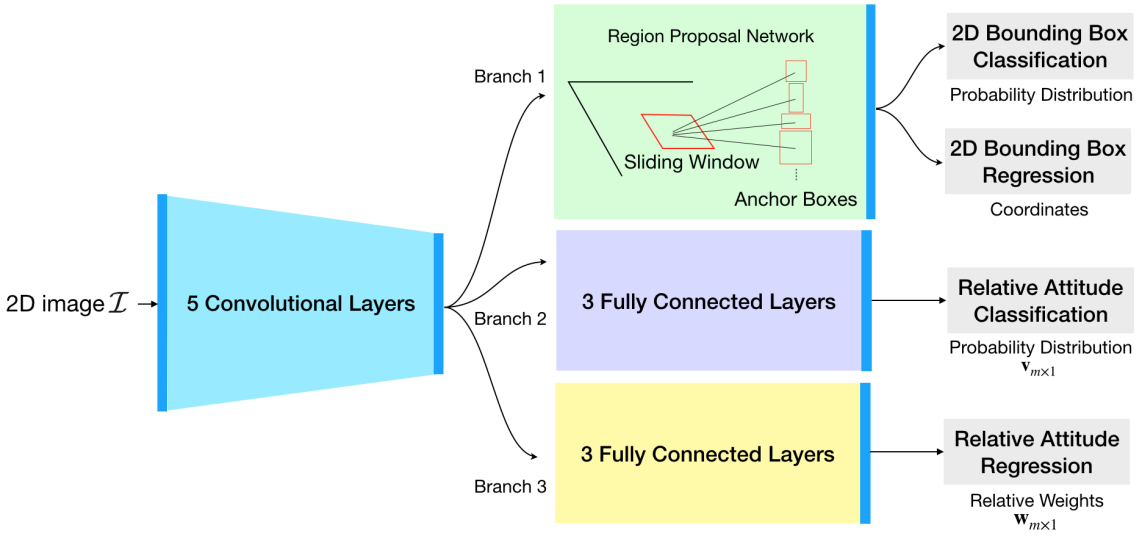


Figure 5.3: Illustration of the convolutional neural network used in the SPN method. Branch 1 uses the region proposal network to detect a 2D bounding box around the target spacecraft while Branches 2 and 3 of the network are used in a hybrid classification-regression fashion to obtain the relative position.

detected 2D bounding box is not only important to effectively remove the background from the image before relative attitude estimation but makes the SPN method extensible to perform pose estimation for multiple objects or spacecraft components from the same image. The region proposal network used in the SPN method is an "off-the-shelf" object detection algorithm that takes the output of the five convolutional layers as an input. This network uses a sliding window approach on the output of the convolutional layers to produce region proposals based on predefined anchor boxes. Each anchor box is centered at the sliding window in question and is associated with a scale and aspect ratio. Typically, three scales and three aspect ratios are used, yielding $k = 9$ anchor boxes at each sliding position. Therefore, for each sliding window location, the region proposal network outputs k probabilities (or scores) of whether the target spacecraft is present and regresses to $4k$ coordinates of the corresponding k 2D bounding boxes. Finally, the 2D bounding box associated with the highest probability is provided as the output. The reader can find further implementation details in the original paper [101]. Even though the SPN method uses the region

proposal network in its current implementation, it can be easily swapped with another state-of-the-art object detection algorithm [102] based on specific computation runtime, storage, and accuracy requirements.

Gauss-Newton Algorithm

The resulting 2D bounding box estimated by the region proposal network and the relative attitude estimated by Branches 2 and 3 of the SPN method are combined with geometrical constraints to estimate the relative position using the Gauss-Newton algorithm. The SPN method uses the fact that the perspective projection of the 3D wireframe model of the target spacecraft must fit tightly within the detected 2D bounding box.

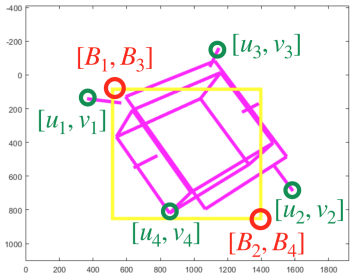


Figure 5.4: Schematic of the projection of the 3D wireframe model of the target (pink) and the estimated bounding box (yellow) in the image plane.

As shown in Figure 5.4, the 2D bounding box can be parametrized by its top-left coordinates, (B_1, B_3), and its bottom-right coordinates, (B_2, B_4), defined in the image plane. Given the estimated relative attitude parametrized as a rotation matrix, $R(\tilde{\mathbf{q}}_{\mathbf{BC}})$, the camera focal lengths, (f_x, f_y), and the camera principal points, (c_x, c_y), the 3D points defined in the body frame of the target spacecraft, $\mathbf{X}_i$, can be projected into the image plane using the perspective equation,

$$\begin{bmatrix} u_i w_i \\ v_i w_i \\ w_i \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R(\tilde{\mathbf{q}}_{\mathbf{BC}}) & \mathbf{t}_{\mathbf{BC}} \end{bmatrix} \mathbf{X}_i. \tag{5.1}$$

The constraint that the perspective projection of the 3D wireframe model fits tightly within the 2D bounding box requires that the four extremal projected points, (u_i, v_i),

be as close as possible to the four edges of the 2D bounding box. Specifically, the four extremal projected points are the "left-most", "right-most", "top-most", and "bottom-most" points, respectively. For example, (u_1, v_1) is the "left-most" projected point and it is constrained to be as close as possible to B_1, the coordinate representing the left edge of the 2D bounding box. Mathematically, the SPN method solves the following minimization problem subject to the constraint posed by Equation 5.1

$$\underset{\mathbf{t_{BC}}}{\text{minimize}} \quad \sum_{i=1}^{2}(u_i(\mathbf{t_{BC}}) - B_i)^2 + \sum_{i=3}^{4}(v_i(\mathbf{t_{BC}}) - B_i)^2. \tag{5.2}$$

The minimization problem is solved using the Gauss-Newton algorithm, which requires an initial guess of $\mathbf{t_{BC}}$. The SPN method uses the detected 2D bounding box and the characteristic length of the 3D wireframe model to provide this initial guess.

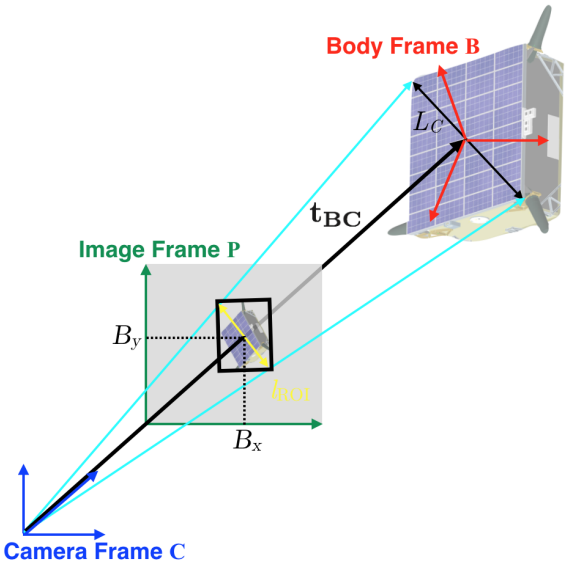


Figure 5.5: Calculation of the relative position using the 2D bounding box.

Figure 5.5 shows that the knowledge of the diagonal characteristic length, L_C, of the spacecraft 3D model and the diagonal length of the detected 2D bounding box, l_{ROI}, can be used to obtain $\mathbf{t_{BC}}$. In particular, a coarse estimate of the distance to

the target spacecraft from the origin of the camera frame is

$$||\mathbf{t_{BC}}||_2 = \frac{\left(\frac{f_x+f_y}{2}\right) L_C}{l_{\text{ROI}}}. \tag{5.3}$$

Assuming that the origin of the target spacecraft body frame, B, lies on the ray projected from the origin of the camera frame, C, towards the center of the detected 2D bounding box, (B_x, B_y), azimuth and elevation angles, (α, β), can be derived using (c_x, c_y) and (B_x, B_y)

$$\alpha = \tan^{-1}\left(\frac{B_x - c_x}{f_x}\right) \tag{5.4}$$

$$\beta = \tan^{-1}\left(\frac{B_y - c_y}{f_y}\right). \tag{5.5}$$

Finally, the coarse relative position used as the initial guess in the Gauss-Newton algorithm is

$$\mathbf{t_{BC}} = \begin{bmatrix} \cos(\alpha) & 0 & -\sin(\alpha) \\ 0 & 1 & 0 \\ \sin(\alpha) & 0 & \cos(\alpha) \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\beta) & \sin(\beta) \\ 0 & -\sin(\beta) & \cos(\beta) \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ ||\mathbf{t_{BC}}||_2 \end{bmatrix}. \tag{5.6}$$

5.1.2 Relative Attitude Determination

The layers of a convolutional neural network transform the raw pixel information, $\mathcal{I}$, to gradually more abstract representations using predefined non-linear functions that contain unknown coefficients. The output of the network is then constrained to minimize a loss function that represents a discrepancy or error between the output of the network and the expected output for a set of training samples (supervised learning). For example, the network used in the SPN method can be represented by

$$\begin{aligned} a_0 &= \mathcal{I} \\ a_1 &= g_1(W_1 * a_0 + b_1) \\ &\vdots \\ a_n &= g_{n-1}(W_n * a_{n-1} + b_n) \end{aligned} \tag{5.7}$$

where W_i are the unknown weights for the i-th layer, b_i are the unknown biases for the i-th layer, and g_i are the known non-linear functions for the i-th layer. For relative attitude estimation, the network's output is expected to be values defined on a continuous non-Euclidean space, prohibiting the direct use of a typical L2 loss function. To handle this problem, other authors have proposed several loss functions [49, 103, 104] that directly regress to a relative attitude estimate; however, these have limited accuracy and require further refinement [105]. Instead, the SPN method uses two branches of fully-connected layers that share the output of five preceding convolutional layers as their inputs. The features output by the final convolutional layer associated with the detected 2D bounding box are selected using the RoI pooling layer[106] to add robustness against the background in the images. Using these features as input, Branch 2 performs a classification task to find the closest predefined attitude classes that describe the input image. Branch 3 uses these features as input to perform a regression task to find the relative weights of the attitude classes identified in Branch 2. For example, Figure 5.6 shows a simplified example where one dimension of the relative attitude has been discretized. For this case, Branch 2 is expected to find attitude classes 4 and 5 as the two adjacent classes that best describe the input image while Branch 3 is expected to find relative weights as a function of the angular differences α_4 and α_5 associated with the corresponding attitude classes. The sizes of the five convolutional layers and both sets of the three fully connected layers have been adopted from the Zeiler and Fergus model architecture [107].

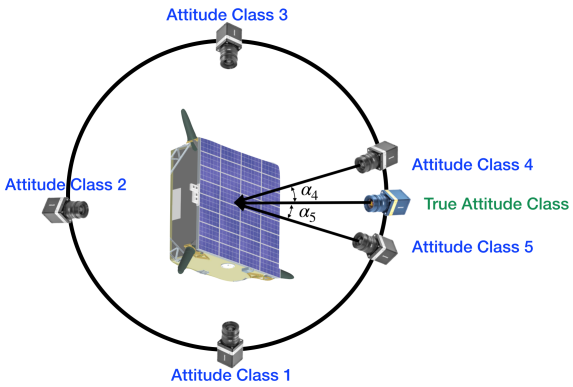


Figure 5.6: Visualization of the relative attitude discretization in a single dimension. Here attitude classes 4 and 5 are adjacent classes closest to the ground truth attitude.

The attitude classes are parametrized as m unit quaternions representing uniformly distributed random rotations in the $SO(3)$ space. Algorithm 1 shows the subgroup algorithm [108] used to obtain these random rotations. The algorithm amounts to multiplying a uniformly distributed element from the subgroup of planar rotations with a uniformly distributed coset represented by the rotations pointing the z-axis in different directions.

Algorithm 1: Computes m uniformly distributed random rotations parametrized as unit quaternions.

Input : Integer m
Output: Matrix quats

$x_0 \leftarrow \text{rand}(m)$ `// get m uniformly distributed float values` $\in$ `[0,1]`
$x_1 \leftarrow \text{rand}(m)$
$x_2 \leftarrow \text{rand}(m)$
$\theta_1 = 2 * \pi * x1$
$\theta_2 = 2 * \pi * x2$
$s_1 = \sin(\theta_1)$
$s_2 = \sin(\theta_2)$
$c_1 = \cos(\theta_1)$
$c_2 = \cos(\theta_2)$
$r_1 = \sqrt{1 - x_0}$
$r_2 = \sqrt{x_0}$
quats $= [s_1 .* r_1, c_1 .* r_1, s_2 .* r_2, c_2 .* r_2]$

To determine which of the m predefined attitude classes are the closest to the given image, Branch 2 is tasked to output an $m \times 1$ vector $\mathbf{v}$, which represents a probability distribution. Each entry $\mathbf{v}_j$ is the probability whether the attitude class in question is one of the n closest attitude classes to the relative attitude of the given image. Note that both m and n are hyper-parameters. Closeness is defined by the angular difference between the unit quaternion representing the attitude class, $\mathbf{q}_i$, and the unit quaternion representing the ground-truth value of the relative attitude, $\mathbf{q_{BC}}$. To ensure that $\mathbf{v}$ is a valid probability distribution, the output of the final layer

of the Branch 2 is passed through the softmax function

$$\sigma(\mathbf{v})_j = \frac{e^{\mathbf{v}_j}}{\sum_{k=1}^{m} e^{\mathbf{v}_j}} \quad \forall j \in [1, m] \tag{5.8}$$

Then the loss function for Branch 2, L_{class}, representing the difference between the branch output and the expected probability distribution, $\tilde{\mathbf{v}}$, can be written as

$$L_{\text{class}} = -\sum_{j=1}^{m} \tilde{\mathbf{v}}_j \log(\sigma(\mathbf{v})_j) + \lambda ||\mathbf{W_1}||_2^2, \tag{5.9}$$

where $\mathbf{W_1}$ represents the weights of the final three layers of Branch 2 and λ is a scalar representing the strength of the L2 regularization. The L2 regularization aims to penalize the existence of large weights and prevent overfitting to the training examples. Note that the entries of $\tilde{\mathbf{v}}$ corresponding to the n closest attitude classes is set to $1/n$ while the rest of the entries are set to zero.

To estimate the relative attitude for the given image, Branch 3 is tasked to output an $m \times 1$ vector, $\mathbf{w}$, containing weights for the n closest attitude classes identified in Branch 2. The weights for the remaining $m - n$ attitude classes are output but not used during either training or inference. The weights of the n closest attitude classes are passed through the softmax function such that their sum adds to unity. Let Ω be the vector of indices of the n largest values in $\sigma(\mathbf{v})$, then the loss function for the second branch can be written as

$$L_{\text{reg}} = -\sum_{j=1}^{m} \tilde{\mathbf{w}}_j \log\left(\frac{e^{\mathbf{w}_j}}{\sum_j e^{\mathbf{w}_j}}\right) + \lambda ||\mathbf{W_2}||_2^2 \quad \forall j \in \Omega, \tag{5.10}$$

where $\mathbf{W_2}$ represents the weights of the final three layers of Branch 3 and $\tilde{\mathbf{w}}$ is the ground truth vector of weights for the given image. The entries of $\tilde{\mathbf{w}}$ are set based on the angular difference between the quaternion of the attitude class in question and the ground truth quaternion of the given image. Specifically,

$$\tilde{\mathbf{w}}_j = \frac{1 - \alpha_j/\pi^2}{n - \sum_j \alpha_j/\pi^2}, \forall j \in \Omega, \tag{5.11}$$

where α_j is the angular difference between the unit quaternion representing the attitude class in question and the unit quaternion representing the ground truth of the relative attitude of the given image. Hence, the total loss function for relative attitude estimation can be written as:

$$L_{\text{total}} = L_{\text{class}} + \mu L_{\text{reg}} \tag{5.12}$$

The classification loss, L_{class}, is designed to force Branch 2's coefficients to be correlated with macroscopic changes in the relative attitude without penalizing classification predictions of attitude classes that have a small angular difference between them. Instead, the regularization loss, L_{reg} is designed to force the Branch 3's coefficients to be correlated with the microscopic differences between adjacent attitude classes.

During inference, the estimate of the relative attitude, $\tilde{\mathbf{q}}_{\mathbf{BC}}$, is computed using an average of the unit quaternions of the n closest attitude classes, $\mathbf{Q}$, weighted by their corresponding relative weights, $\mathbf{\Gamma}$. This is akin to minimizing a weighted sum of the squared Frobenius norms of the differences between the rotation matrix representation of $\mathbf{Q}_i$ and $\mathbf{q}$

$$\tilde{\mathbf{q}}_{\mathbf{BC}} = \arg\min_{\mathbf{q}\in\mathbb{S}^3} \sum_i \mathbf{\Gamma}_i ||R(\mathbf{q}) - R(\mathbf{Q}_i)||_F^2 \tag{5.13}$$

where $\mathbb{S}^3$ denotes the unit 3-sphere[109]. The pseudo-code used to compute the weighted average is presented below as Algorithm 2.

5.1.3 Target-In-Target Pose Estimation

The SPN method is easily generalizable to estimate the pose of multiple target geometries. For example, this fact can be leveraged to perform pose estimation with respect to the docking port, once it comes into view, during proximity operations in an on-orbit servicing scenario. Figure 5.7 shows examples of two images captured during a simulated negative R bar approach trajectory, which is a typical relative trajectory executed during the terminal stages of a rendezvous and docking mission [110]. As compared to conventional feature-based methods, the SPN method can estimate the

Algorithm 2: Computes a unit quaternion that is an average of the input unit quaternions $\mathbf{Q}$ weighted by the input weights $\mathbf{\Gamma}$.

Input : Matrix $\mathbf{Q}$, Vector $\mathbf{\Gamma}$
Output: Vector q

```
A ← zeros(4,4)
M ← length(Q,1)
foreach i in range(M) do
    q = Q_i
    w_i = Γ_i
    A = w_i .* (q * q') + A
    alpha = alpha + w_i
end
A = (1.0 / alpha) * A
// Get the eigenvector corresponding to the largest eigenvalue
q← eigs(A,1)
```

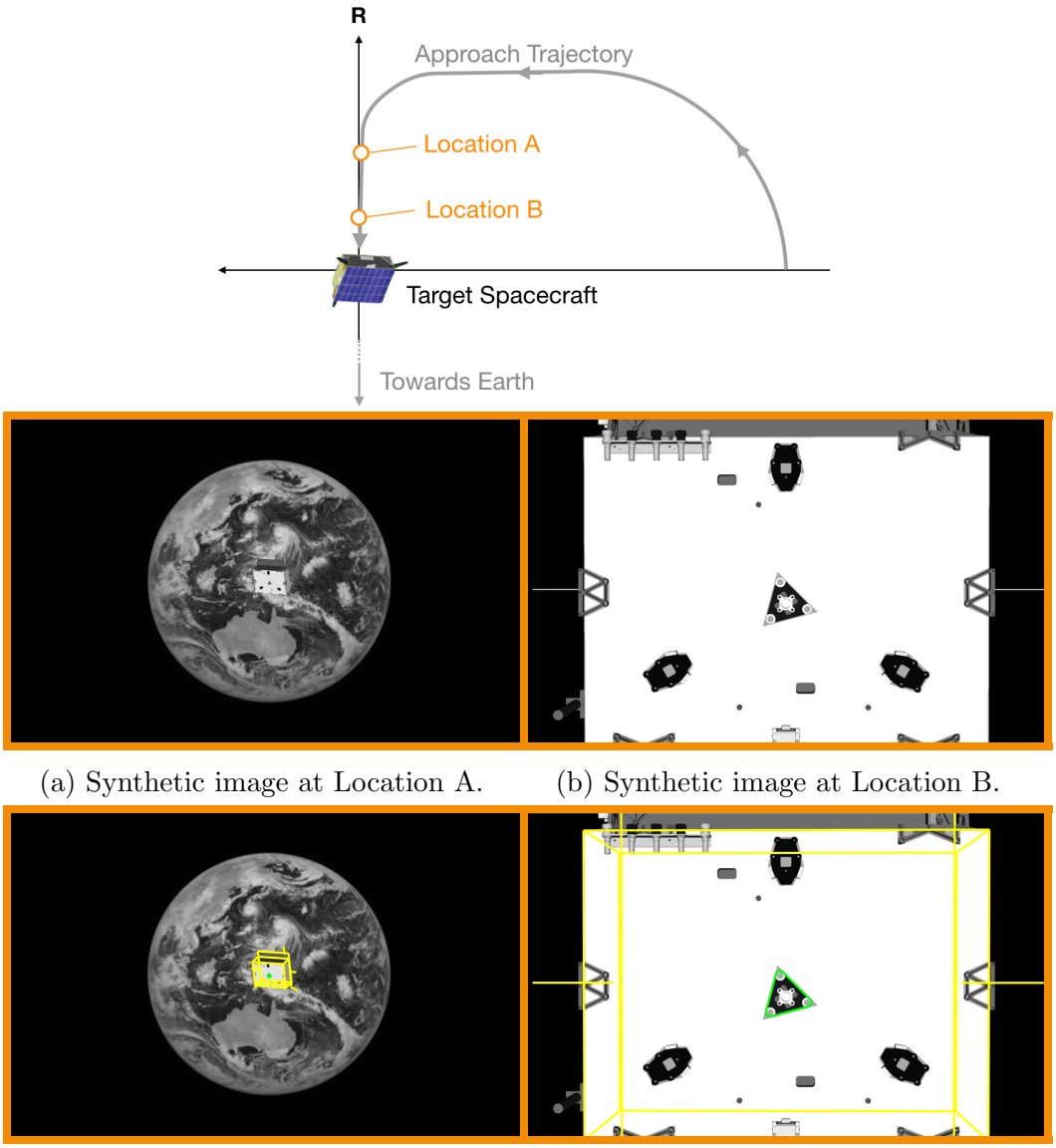
pose for both the example images and does not require the entire spacecraft to be in the camera's field-of-view.

Given the wireframe model for each of the targets, the SPN method can be trained to estimate the pose for each. The region proposal network is trained to estimate the 2D bounding box around each of the targets. The image regions inside these respective 2D bounding boxes are sequentially fed through Branch 2 and Branch 3 to estimate the relative attitude of each of the targets with respect to the camera. To train branches 2 and 3 for each of the multiple targets simultaneously, the total loss function is modified as

$$L_{\text{total,multi}} = \sum_i \kappa_i (L_{\text{class}} + \mu L_{reg})_i, \tag{5.14}$$

$$\sum_i \kappa_i = 1 \tag{5.15}$$

where κ_i is a scalar weight for each of the multiple targets. This scalar weight is set for each of the images during training based on the visibility of the individual targets. In particular, the weights are set to the pixel size of the individual target in a particular image. The weight is set to zero if the target is occluded. The weights for



(a) Synthetic image at Location A. (b) Synthetic image at Location B.

(c) Location A ground truth pose. (d) Location B ground truth pose.

Figure 5.7: An example of two images captured during a simulated negative R bar approach trajectory. The Tango spacecraft is at 18 meters and 1.5 meters away at locations A and B, respectively. The SPN method can be simultaneously trained to estimate the pose of the entire spacecraft (yellow) and the apogee motor (green).

each image are normalized such that they sum to unity. Recall that the output of the region proposal network is passed through the non-maximum suppression algorithm that outputs a set of bounding boxes along with the probability of the target object(s) being present in the corresponding bounding box. At inference, the bounding box with the highest probability is selected and the associated target object is used for relative attitude and relative position estimation. As before, the Gauss-Newton algorithm is run to estimate the relative position for the selected target geometry.

5.1.4 Uncertainty Quantification

In addition to estimating the relative pose of the target spacecraft, the SPN method is also capable of estimating the uncertainty associated with the estimated pose. Uncertainty estimation is a vital capability since it allows the use of the pose estimated by the SPN method to be used as pseudo-measurements in a navigation filter. Unlike state-of-the-art methods such as the probabilistic PoseNet [111], the SPN method can compute this uncertainty without requiring multiple Monte-Carlo samplings of its convolutional neural network. In contrast, the uncertainty in the relative attitude estimate is computed by using the output of Branch 2 and the definition of the predefined attitude labels while the uncertainty in the relative position is computed using the Jacobian matrix of the Gauss-Newton algorithm.

Recall that the output of the final layer of Branch 2 is passed through the softmax function to output a probability distribution vector over the predefined attitude classes, $\sigma(\mathbf{v})$. By leveraging the knowledge of the relative attitude represented by each of the predefined attitude classes, $\sigma(\mathbf{v})$ can be expressed as a distribution over the relative attitude space. In particular, $\sigma(\mathbf{v})$ is first sorted according to the angular difference between the estimated relative attitude, $\tilde{\mathbf{q}}_{\mathbf{BC}}$, and the unit quaternion representation of the j-th predefined attitude class, $\tilde{\mathbf{q}}_j$

$$\alpha_j = 2\cos^{-1}\left(|\langle \tilde{\mathbf{q}}_{\mathbf{BC}}, \mathbf{q}_j \rangle|\right) \tag{5.16}$$

Similarly, the vector of pre-defined attitude class is sorted to obtain $\tilde{\mathbf{q}}_{\text{class,sorted}}$. Then, a half-normal distribution is fit to the resulting sorted probability distribution, $\sigma(\mathbf{v})_s$. The standard deviation of the strictly-positive half-normal distribution, σ_{dist}, is then

used to estimate the standard deviation in the relative attitude as the difference between $\tilde{\mathbf{q}}_{\mathbf{BC}}$ and $\tilde{\mathbf{q}}_k$.

$$\tilde{\mathbf{q}}_{\text{std}} = \tilde{\mathbf{q}}_{\mathbf{BC}} \times (\tilde{\mathbf{q}}_k)^{-1} \tag{5.17}$$

Here $\tilde{\mathbf{q}}_k$ is the unit quaternion associated with the predefined attitude class closest to σ_{dist} and can be found by indexing into $\tilde{\mathbf{q}}_{\text{class,sorted}}$,

$$\tilde{\mathbf{q}}_k = \tilde{\mathbf{q}}_{\text{class,sorted}}[\texttt{round}(\sigma_{\text{dist}})]. \tag{5.18}$$

To calculate the uncertainty in the relative position, $\tilde{\mathbf{q}}_k$ and the estimated relative position, $\tilde{\mathbf{t}}_{\mathbf{BC}}$, are used to re-project the wireframe model on the image plane. This allows the calculation of the extremal points, as explained in Section 5.1.1. The Jacobian matrix, $\mathbf{J}$, and the residual values of the extremal points, $\mathbf{r}$, of the minimization problem setup in Equation 5.2 are then used to compute the uncertainty in the relative position,

$$\tilde{\mathbf{t}}_{\text{std}} = \left| \left(\mathbf{J}^T \mathbf{J} \right) \mathbf{J}^T \mathbf{r} \right|. \tag{5.19}$$

5.2 Spacecraft Pose Estimation Dataset

Training a convolutional neural network usually requires extremely large labeled image datasets such as ImageNet[91] and Places[92], which contain millions of images. Collecting and labeling such an amount of actual space imagery is extremely difficult. Therefore, this chapter introduces the Spacecraft Pose Estimation Dataset (SPEED), a dataset of images that enables not only training and validation of the SPN method but also benchmarking various state-of-the-art monocular vision-based pose estimation techniques. Table 5.1 provides an overview of the SPEED training, validation, and test datasets generated or used in this chapter. Table 5.1 also provides an overview of the PRISMA-25 and Apogee motor datasets that were used to demonstrate the robustness of the SPN method to actual camera imagery and imagery where the spacecraft is partially occluded. Any datasets that used OpenGL as a "source" will be referred to as "synthetic" datasets in the rest of this chapter even though some of those datasets include actual meteorological Earth imagery in the background. On the other hand, SPEED real test and PRISMA-25 will both be

Table 5.1: Overview of the datasets generated for training, validating, and testing the SPN method.

Dataset Name	Source	Camera	No. of Images	Purpose	Ground Truth Source
SPEED training	OpenGL + Himawari-8	PointGrey	12000	Training	MATLAB
SPEED/PRISMA training	OpenGL + Himawari-8	PRISMA	12000	Training	MATLAB
Apogee motor training	OpenGL + Himawari-8	PointGrey	5000	Training	MATLAB
SPEED synthetic test	OpenGL + Himawari-8	PointGrey	3000	Validation	MATLAB
Apogee motor test	OpenGL + Himawari-8	PointGrey	72	Test	MATLAB
Imitation-25	OpenGL	PRISMA	25	Test	Flight dynamics products
SPEED real test	TRON	PointGrey	900	Test	Vicon Motion Tracking
PRISMA-25	PRISMA	PRISMA	25	Test	Flight dynamics products

referred to as "real" or "actual camera" datasets. Regardless, all datasets contain grayscale imagery along with corresponding ground truth pose information for the target spacecraft.

The SPEED images and the corresponding ground truth pose information is generated using two key complementary sources at the Space Rendezvous Laboratory (SLAB) of Stanford University[1]. The first source produced 15,000 augmented reality images based on the Optical Stimulator (OS) camera emulator software [43, 2] and actual images of the Earth captured by the Himawari-8 geostationary meteorological satellite [112]. The second source produced 900 actual camera images of a 1:1 mock-up of the Tango spacecraft using the Testbed for Rendezvous and Optical Navigation (TRON). Table 5.2 provides the camera model used for both these sources. Figure 5.8

[1] `www.slab.stanford.edu`

Table 5.2: Parameters of the PointGrey and PRISMA camera.

Parameter	Description	PointGrey Camera	PRISMA Camera
N_u	Number of horizontal pixels	1920	752
N_v	Number of vertical pixels	1200	580
f_x	Horizontal focal length	0.0176 m	0.0199 m
f_y	Vertical focal length	0.0176 m	0.0193 m
du	Horizontal pixel length	$5.86 \cdot 10^{-6}$ m	$8.6 \cdot 10^{-6}$ m
dv	Vertical pixel length	$5.86 \cdot 10^{-6}$ m	$8.3 \cdot 10^{-6}$ m
$HFOV$	Horizontal Field of View	35.452 deg	18.384 deg
$VFOV$	Vertical Field of View	22.595 deg	14.228 deg

shows an example comparison between imagery that can be generated using these two sources and the imagery available from the PRISMA mission.

The SPEED training-set consists of 12000 synthetic images from the first source while the remainder 3000 synthetic images and the 300 actual camera images from the second source are available as the validation- and test-sets. The motivation for excluding the actual camera images from the training-set is to evaluate the robustness and the domain adaptation capabilities of the pose estimation techniques. Moreover, for an on-orbit servicing mission, access to close-range flight imagery of the target spacecraft is of course inaccessible. Therefore, constraining the training and validation of the pose estimation methods to synthetic imagery simulates a realistic mission scenario.

5.2.1 Synthetic Images

The first source renders synthetic images of the Tango spacecraft using MATLAB and C++ language bindings of OpenGL. A set of relative attitudes and relative positions are selected to create a diverse set of views of the target spacecraft. Unit quaternion parametrization of uniformly random rotations in the $SO(3)$ space is selected using Algorithm 1. Figure 5.14 shows the distribution of $\mathbf{q_{BC}}$ in the SPEED images, parametrized as Euler angles. The relative positions are obtained by separately

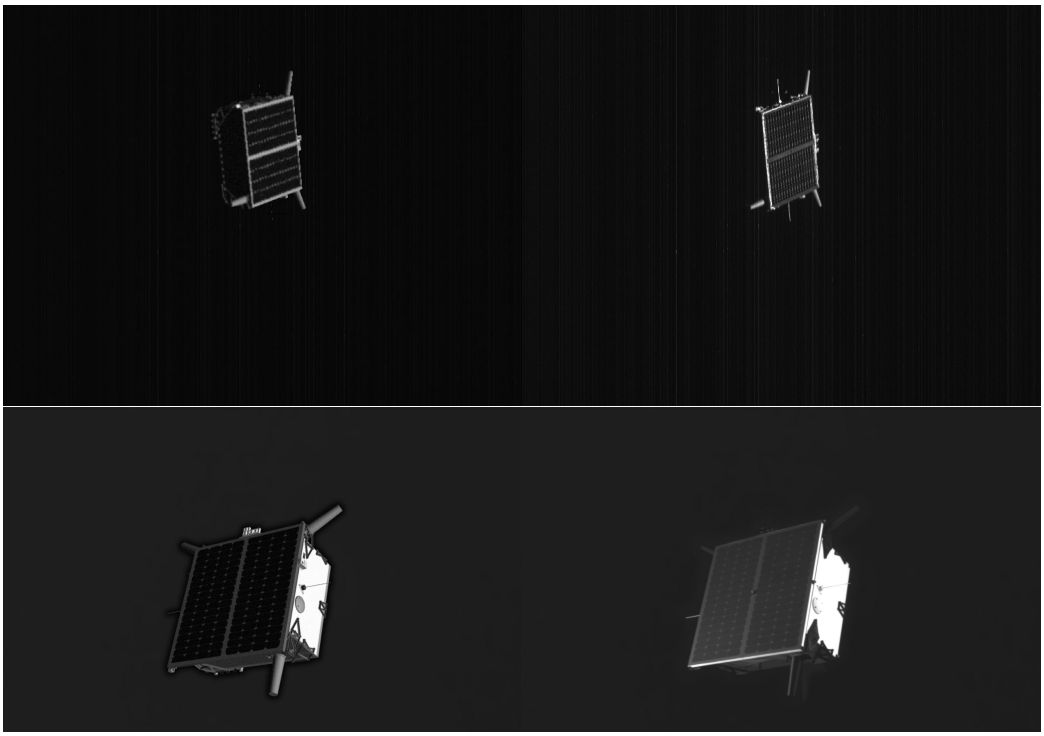


Figure 5.8: A comparison between example images that can be captured using the Optical Stimulator camera emulator (top-left, bottom-left), the TRON facility (bottom-right), and available from the PRISMA mission (top-right).

selecting the relative distance and the bearing angles (defined in Equations 5.4 and 5.5). The bearing angles are uncorrelated random values selected from a multivariate normal distribution, $D_1 \sim \mathcal{N}(\mu = [N_u/2,\ N_v/2]\text{px},\ \sigma = [5N_u/2,\ 5N_v/2]\text{px})$. The relative distance is randomly selected from a standard normal distribution, $D_2 \sim \mathcal{N}(\mu = 3\text{m}, \sigma = 10\text{m})$. Any relative distance values below 3 meters and above 50 meters are rejected. Figure 5.9 shows the distribution of $\mathbf{t_{BC}}$ in the SPEED images.

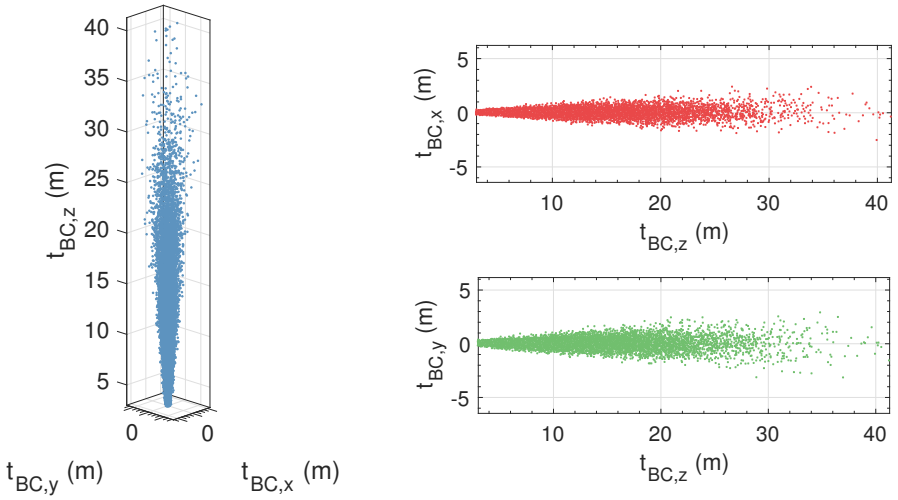


Figure 5.9: The distribution of the relative position, $\mathbf{t_{BC}}$, in the SPEED images.

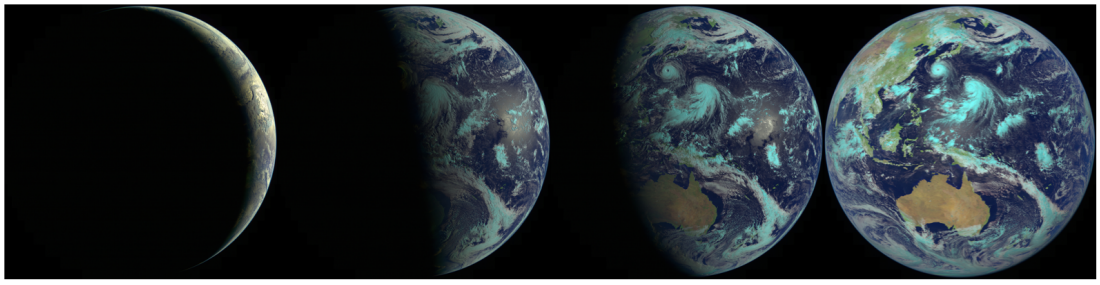


Figure 5.10: A montage of four of the 72 full-disk Earth images used to generate the background for half of the SPEED synthetic images.

As compared to the previous work of Beierle [2], the SPEED synthetic imagery makes four improvements. Firstly, in contrast to the previous work, SPEED synthetic

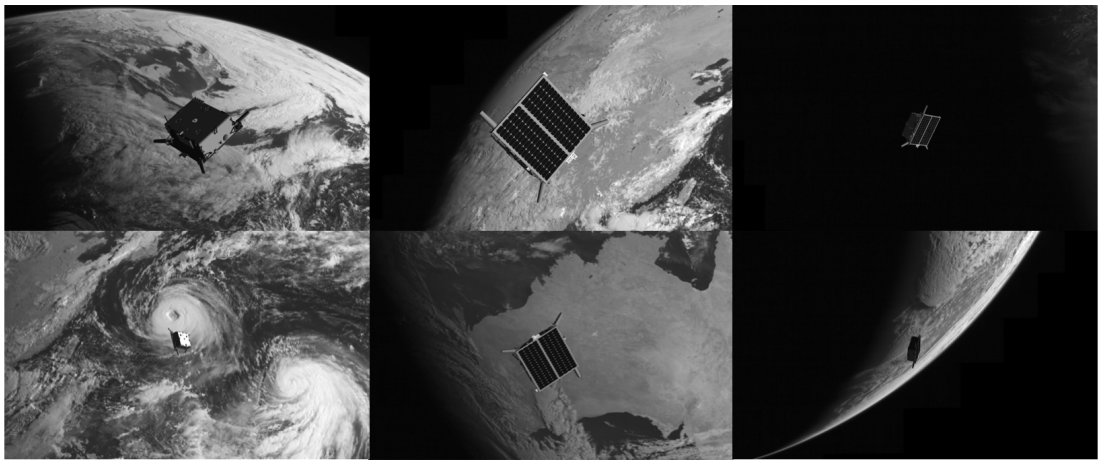


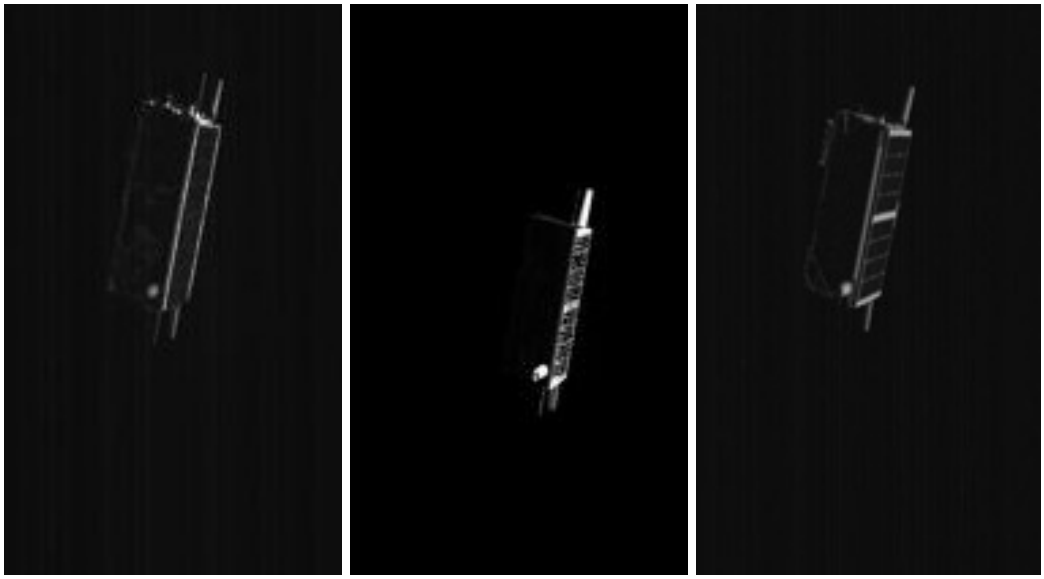
Figure 5.11: A montage of six synthetic images from the SPEED training-set.

imagery contains Earth in the background using high-resolution imagery captured using the Himawari-8 meteorological satellite. Secondly, the SPEED synthetic imagery uses a calibrated pinhole camera model for rendering. This results in a better 2D alignment of the SPEED imagery with actual flight imagery captured during the PRISMA mission. Thirdly, the low contrast background produced due to the Earth's albedo is adjusted to match the histogram of the flight imagery. Lastly, surfaces such as the solar panels and multi-layer insulation are modeled in higher fidelity through the use of 2D texture models obtained from OHB Sweden[2]. Figure 5.12 shows a visual comparison between actual flight imagery [1], synthetic imagery generated using the Optical Stimulator camera emulator software of Beierle and D'Amico [2], and SPEED synthetic imagery, which builds upon the Optical Stimulator camera emulator software.

Earth in Background

The azimuth and elevation angles for the solar illumination are specifically chosen to match the solar illumination in the 72 actual images of the Earth captured by the Himawari-8 geostationary meteorological satellite. Figure 5.10 shows a montage of these images. Each of the 72 images provided a $100 \cdot 10^6$ pixels resolution disk-view of the Earth and were taken 10 minutes apart from each other over 12 hours. Each

[2] `http://www.ohb-sweden.se`



(a) Flight imagery (b) Beierle and D'Amico (c) SPEED

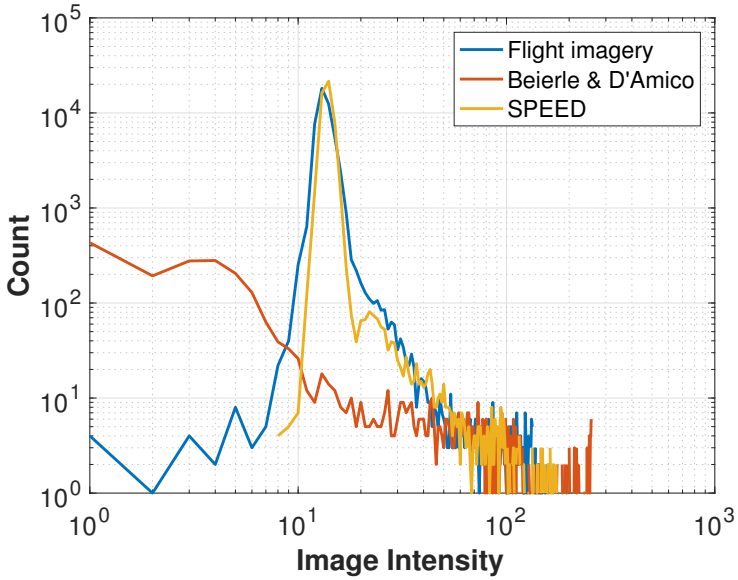


Figure 5.12: Cropped versions of actual flight imagery captured during the PRISMA mission [1] (top-left), synthetic imagery generated using previous work by Beierle and D'Amico [2] (top-middle), SPEED synthetic imagery (top-right), and histogram comparison of the three images (bottom).

image is converted to grayscale, cropped and inserted in as the background for half of the synthetic images of the Tango spacecraft. The location of the crop is selected at random from a uniform distribution spanning the Earth's image. The size of the crop is selected to match the scale of the Earth when viewed through a camera (described in Table 5.2) located at an altitude of 700 km and pointed in the nadir direction. Gaussian blurring and white noise are added to all images to emulate the depth of field and shot noise, respectively. Figure 5.11 shows a montage of the resulting augmented reality images.

5.2.2 Real Images

The second source of the SPEED images is the TRON facility at SLAB. As shown in Figure 5.13, it consists of a seven degrees-of-freedom robotic arm, which positions and orients a vision-based sensor with respect to a target object or scene. Custom illumination devices simulate Earth albedo and sunlight to high fidelity to emulate the illumination conditions present in space. TRON provides images of a 1:1 mock-up model of the Tango spacecraft using a Point Grey Grasshopper 3 camera with a Xenoplan 1.9/17 mm lens. Note that this is the same camera as the one used in the first source. Calibrated motion capture cameras report the positions and attitudes of the camera and the Tango spacecraft, which are then used to calculate the "ground truth" pose of Tango with respect to the camera.

Figure 5.16 shows an example of a typical raw image captured in the TRON facility while Figure 5.15 shows a montage of the SPEED actual camera images. Notice that in addition to the mock-up Tango spacecraft this image also contains background artifacts such as the motion-tracking cameras, the illumination devices, and the structure supporting the mock-up Tango spacecraft. The ground truth relative pose is used to minimize the effect of such artifacts on the training and testing of pose estimation algorithms. In particular, a binary mask is created by projecting a high fidelity 3D model of the Tango spacecraft to the image plane using the ground truth relative pose. This binary mask is used to remove the artifacts present in the original image. This process is carried out in an automated fashion once the "ground truth" pose associated with each image has been computed.

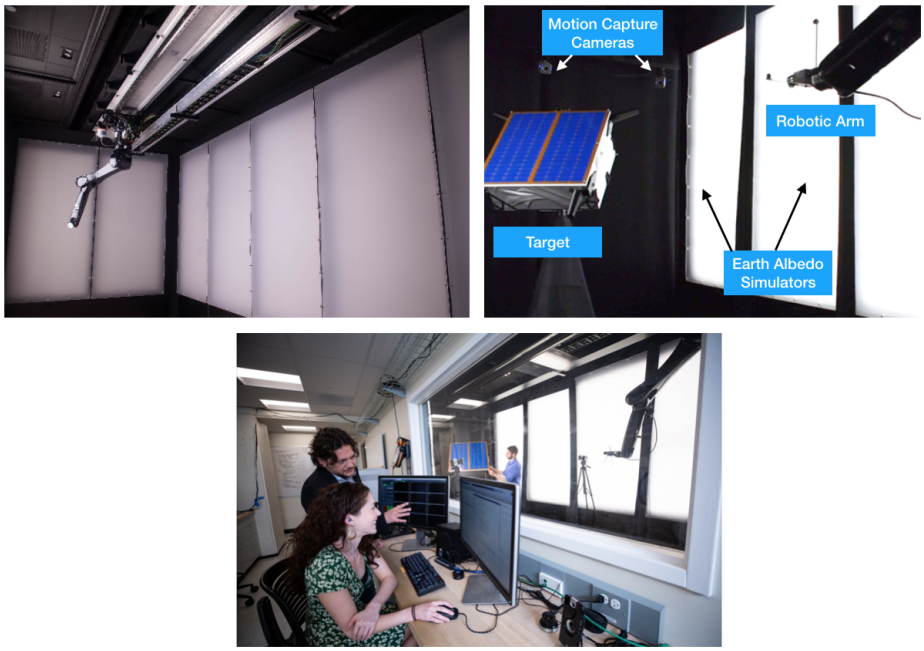


Figure 5.13: The Testbed for Rendezvous and Optical Navigation (TRON) facility at the Space Rendezvous Laboratory (SLAB) of Stanford University.

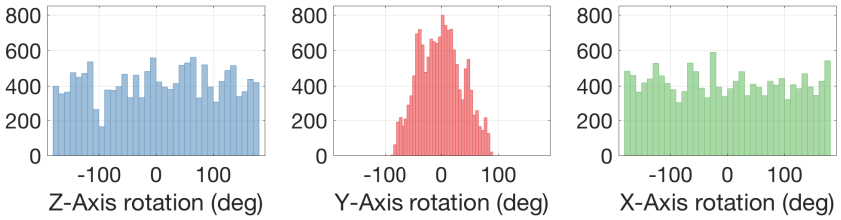


Figure 5.14: The distribution of the relative attitude in the SPEED images. For purposes of visualization, the relative attitude is parametrized as Euler angles.

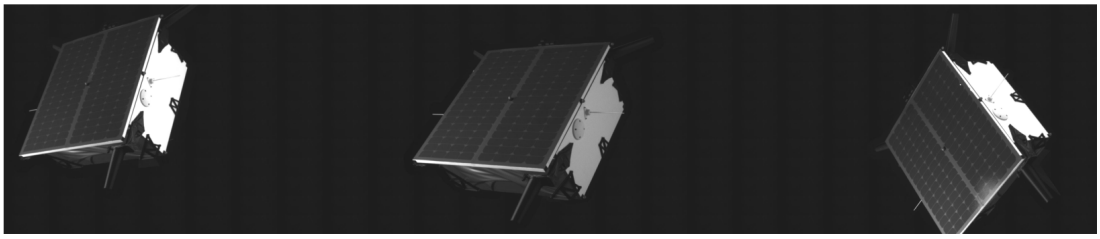


Figure 5.15: A montage of three actual camera images from the SPEED real test-set.

Figure 5.16: An example of a raw image captured in the TRON facility (left), the corresponding image mask generated using the calibrated ground truth pose (middle), and the resulting processed image after the application of the mask (right).

Relative Pose Ground Truth Calibration

Motion tracking cameras and user annotation of a few calibration images are used to calculate the ground truth values of the relative pose associated with the images captured in the TRON facility. The TRON facility consists of 10 Vicon Vero v1.3x motion tracking cameras [113] that individually track Infrared (IR) reflective markers placed in the volume. The cameras feed this information to Vicon Tracker proprietary software, which uses this information to provide up to sub-millimeter level position and sub-degree level attitude information for the camera reference frame, C', and the mock-up Tango reference frame, B'. Both of these reference frames are user-defined in the Vicon Tracker software by grouping the individual IR markers. The origin and alignment of these user-defined reference frames can differ from the "true" reference frames, B and C as defined in Figure 5.2 due to systematic errors. For example, Figure 5.17 shows the reprojection of the Tango wire-frame onto an image captured with TRON based on the uncalibrated ground truth pose information generated using the Vicon software.

In particular, the Vicon Tracker software provides the relative position of the camera, ${}^{\mathbf{V}}\mathbf{t}_{\mathbf{VC'}}$, the relative position of the mock-up Tango, ${}^{\mathbf{V}}\mathbf{t}_{\mathbf{VB'}}$, the rotation matrix that aligns the Vicon base reference frame and the camera reference frame, $\mathbf{R}_{\mathbf{VC'}}$, and the rotation matrix that aligns the Vicon base reference frame and the mock-up Tango reference frame, $\mathbf{R}_{\mathbf{VB'}}$. Note that the superscript represents the reference frame in which the vector is expressed.

The objective of the calibration is to estimate the position and attitude biases between the reference frames as measured by Vicon and the "true" reference frames.

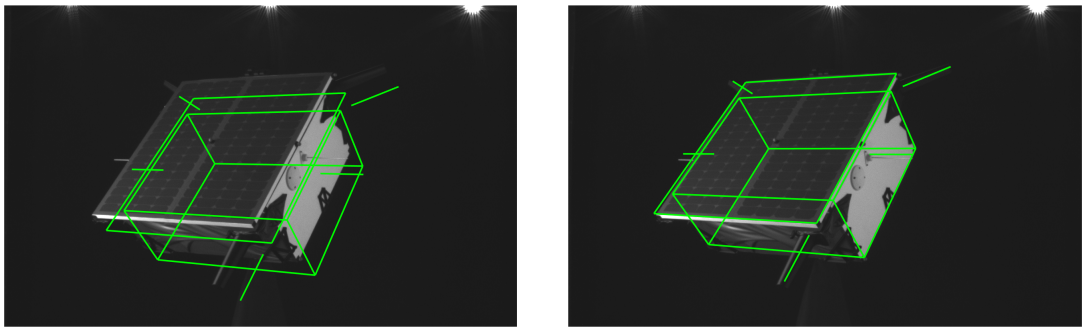


Figure 5.17: The relative pose ground truth projected (in green) on an image captured in the TRON facility before (left) and after (right) calibration.

In particular, ${}^{\mathbf{C}}\mathbf{t}_{\mathbf{B'B}} - {}^{\mathbf{C}}\mathbf{t}_{\mathbf{C'C}}$, $\mathbf{R}_{\mathbf{C'C}}$, and $\mathbf{R}_{\mathbf{B'B}}$ are estimated. Once these constant biases are estimated, the calibrated ground truth for the relative pose of the "true" Tango frame w.r.t. the "true" camera frame can be computed as

$$ {}^{\mathbf{C}}\mathbf{t}_{\mathbf{BC}} = \mathbf{R}_{\mathbf{VC}} \left({}^{\mathbf{V}}\mathbf{t}_{\mathbf{VC'}} - {}^{\mathbf{V}}\mathbf{t}_{\mathbf{VB'}} \right) - \mathbf{r}_{\mathbf{E}} \tag{5.20} $$

$$ \mathbf{R}_{\mathbf{BC}} = \mathbf{R}_{\mathbf{B'B}}\, \mathbf{R}_{\mathbf{VB'}}\, {\mathbf{R}_{\mathbf{VC'}}}^{T}\, {\mathbf{R}_{\mathbf{C'C}}}^{T} \tag{5.21} $$

where $\mathbf{r}_{\mathbf{E}}$ is a shorthand for ${}^{\mathbf{C}}\mathbf{t}_{\mathbf{B'B}} - {}^{\mathbf{C}}\mathbf{t}_{\mathbf{C'C}}$ and $\mathbf{R}_{\mathbf{VC}}$ is equal to $\mathbf{R}_{\mathbf{C'C}}\mathbf{R}_{\mathbf{VC'}}$.

To estimate the three quantities above, the robotic arm is commanded to position the camera at ten different positions. In theory, only three different positions are required to constrain the calibration fully. The camera is commanded to capture an image of the mock-up Tango for each of these positions. Once all the images are captured, the user selects a minimum of six pixel positions in the image frame and corresponding 3D points in the true Tango body reference frame. For each set of six image and model points, the EPnP solver [36] is used to estimate the relative pose between the true Tango body and camera reference frame denoted by ${}^{\mathbf{C}}\mathbf{t}_{\mathbf{BC},\mathrm{pnp}}$ and $\mathbf{R}_{\mathbf{BC},\mathrm{pnp}}$. Following that, the values of $\mathbf{R}_{\mathbf{VC}}$ and $\mathbf{r}_{\mathbf{E}}$ are computed by solving a system of overdetermined linear equations

$$ \mathbf{Y} = \mathbf{A}\mathbf{X} \tag{5.22} $$

with

$$\mathbf{A} = \begin{bmatrix} -{}^{\mathbf{V}}\mathbf{t}_{\mathbf{VC'},1} + {}^{\mathbf{V}}\mathbf{t}_{\mathbf{VB'},1} & 0_{1\times3} & 0_{1\times3} & \\ 0_{1\times3} & -{}^{\mathbf{V}}\mathbf{t}_{\mathbf{VC'},1} + {}^{\mathbf{V}}\mathbf{t}_{\mathbf{VB'},1} & 0_{1\times3} & \mathbf{I_{3\times3}} \\ 0_{1\times3} & 0_{1\times3} & -{}^{\mathbf{V}}\mathbf{t}_{\mathbf{VC'},1} + {}^{\mathbf{V}}\mathbf{t}_{\mathbf{VB'},1} & \\ & \vdots & & \vdots \\ -{}^{\mathbf{V}}\mathbf{t}_{\mathbf{VC'},\mathrm{N}} + {}^{\mathbf{V}}\mathbf{t}_{\mathbf{VB'},\mathrm{N}} & 0_{1\times3} & 0_{1\times3} & \\ 0_{1\times3} & -{}^{\mathbf{V}}\mathbf{t}_{\mathbf{VC'},\mathrm{N}} + {}^{\mathbf{V}}\mathbf{t}_{\mathbf{VB'},\mathrm{N}} & 0_{1\times3} & \mathbf{I_{3\times3}} \\ 0_{1\times3} & 0_{1\times3} & -{}^{\mathbf{V}}\mathbf{t}_{\mathbf{VC'},\mathrm{N}} + {}^{\mathbf{V}}\mathbf{t}_{\mathbf{VB'},\mathrm{N}} & \end{bmatrix}, \tag{5.23}$$

$$\mathbf{Y} = \begin{bmatrix} -{}^{\mathbf{C}}\mathbf{t}_{\mathbf{BC},\mathrm{pnp},1} & \dots & -{}^{\mathbf{C}}\mathbf{t}_{\mathbf{BC},\mathrm{pnp,i}} & \dots & -{}^{\mathbf{C}}\mathbf{t}_{\mathbf{BC},\mathrm{pnp,N}} \end{bmatrix}^T, \tag{5.24}$$

$$\mathbf{X} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{33} & \mathbf{r_E} \end{bmatrix}^T, \text{ and} \tag{5.25}$$

$$\mathbf{R_{VC}} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \tag{5.26}$$

Note that the size of $\mathbf{A}$ is $3N \times 12$ where N is the number of images used in the calibration. After performing this least-squares calculation, an estimate of $\mathbf{R_{VC}}$ is obtained. This rotation matrix is not guaranteed to be orthonormal, which is a required property of any rotation matrix. Therefore, by taking the singular value decomposition of the matrix estimated using the least-squares is projected onto the closest orthonormal matrix. Values of $\mathbf{R_{C'C}}$ and $\mathbf{R_{B'B}}$ can now be estimated using

$$\mathbf{R_{C'C}} = \mathbf{R_{VC}}{\mathbf{R_{VC'}}}^T \tag{5.27}$$

$$\mathbf{R_{B'B}} = {\mathbf{R_{BC,pnp}}}^T\mathbf{R_{C'C}}\mathbf{R_{VC'}}{\mathbf{R_{VB'}}}^T \tag{5.28}$$

5.3 Experiments

This section details the experiments conducted using the SPN method to compare its performance with the CNN-based and feature-based methods developed in Chapters 3 and 4. First, specifics of the training of the SPN method are presented; for example,

the hyperparameters, the dataset splits for training, validation, and testing. Next, a thorough discussion of the relevant results is presented. Specifically, the results subsection describes the performance differential of the SPN method on synthetic images versus real images, target-in-target pose estimation, uncertainty quantification, and performance comparison against the CNN-based and SVD methods.

5.3.1 Training Overview

Table 5.3 provides an overview of the three different versions of the SPN method that were trained using the datasets introduced in Table 5.1. For each of the three networks, the region proposal network, the convolutional layers, and the fully connected layers were pre-trained on the relatively larger ImageNet dataset [91]. Following that, the region proposal network and the fully connected layers were trained using 80% of the available training dataset while the remained 20% was used for validation. The two sets of fully connected layers were trained jointly while the region proposal network was trained separately. Both training routines were carried out using stochastic gradient descent in batches of 16 images. The initial learning-rate was set to 0.003 and set to decay exponentially at a rate of 0.95 every 1000 steps. For attitude determination, the hyper-parameters m and n were set to 1000 and 5, respectively. During training, each image was resized to 224 $\times$ 224 pixels to match the input size of the Zeiler and Fergus model architecture [107].

Table 5.3: Networks trained for performance characterization of the SPN method.

Network Name	Training Dataset	Validation Dataset	Test Dataset
SPN-1	SPEED training	SPEED training	SPEED synthetic test, SPEED real test
SPN-2	SPEED training + Apogee motor training	SPEED training + Apogee motor training	Apogee motor test
SPN-3	SPEED/PRISMA training	SPEED/PRISMA training	Imitation-25, PRISMA-25

For quantitative analysis of the performance, five separate metrics are reported. To measure the accuracy of the 2D bounding box detection as compared with the ground truth 2D bounding box, the Intersection-Over-Union (IoU) metric is reported as

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}. \tag{5.29}$$

To measure the accuracy of the estimated relative position, $\tilde{\mathbf{t}}_{\mathbf{BC}}$ and the ground truth relative position, $\mathbf{t}_{\mathbf{BC}}$, the translation error for each image is calculated as

$$E_T = |\mathbf{t}_{\mathbf{BC}} - \tilde{\mathbf{t}}_{\mathbf{BC}}| \tag{5.30}$$

To measure the accuracy of the estimated relative attitude, $\tilde{\mathbf{q}}_{\mathbf{BC}}$ and the ground truth relative attitude, $\mathbf{q}_{\mathbf{BC}}$, the attitude error for each image is calculated as

$$E_R = 2\cos^{-1}\left|\langle \mathbf{q}_{\mathbf{BC}}, \tilde{\mathbf{q}}_{\mathbf{BC}} \rangle\right|. \tag{5.31}$$

Two additional measures of pose accuracy are presented due to two reasons. Firstly, some of the datasets presented in Table 5.1 were created using different camera models. Secondly, the distribution of the interspacecraft separation for the images is different across some of the datasets. Therefore, to assess the performance across datasets with different field of view and distributions of relative distance, $E_{R,\text{norm}}$ and $E_{T,\text{norm}}$ are reported as the normalized values of rotation error and translation error,

$$E_{T,\text{norm}} = \frac{E_T}{|\mathbf{t}_{\mathbf{BC}}|_2 \quad HFOV}, \tag{5.32}$$

$$E_{R,\text{norm}} = \frac{E_R}{|\mathbf{t}_{\mathbf{BC}}|_2 \quad HFOV}. \tag{5.33}$$

5.3.2 Results

The following subsections detail the results conducted with each of the three networks introduced in Table 5.3.

Table 5.4: Performance of the SPN method on the SPEED synthetic test set, SPEED real test set, and PRISMA-25.

Metric	SPEED synthetic test set	SPEED real test set	PRISMA-25	Imitation-25
Mean IoU (-)	0.858	0.860	0.796	0.817
Median IoU (-)	0.891	0.864	0.823	0.875
Mean E_R (deg)	8.425	18.188	14.724	5.749
Median E_R (deg)	7.0689	13.208	10.368	5.448
Mean $E_{R,\text{norm}}$ (m^{-1})	0.029	0.091	0.069	0.028
Median $E_{R,\text{norm}}$ (m^{-1})	0.020	0.124	0.035	0.027
Mean E_T (m)	[0.055 0.046 0.780]	[0.036 0.015 0.189]	[0.037 0.026 0.834]	[0.028 0.029 0.385]
Median E_T (m)	[0.024 0.021 0.496]	[0.029 0.013 0.191]	[0.021 0.020 0.801]	[0.017 0.026 0.291]
Mean $E_{T,\text{norm}}$ (-)	[0.009 0.007 0.084]	[0.008 0.006 0.082]	[0.004 0.004 0.108]	[0.005 0.004 0.053]
Median $E_{T,\text{norm}}$ (-)	[0.002 0.002 0.044]	[0.002 0.002 0.044]	[0.002 0.003 0.115]	[0.002 0.003 0.040]

Comparison between synthetic and real images

Table 5.4 lists the mean and median values of the five performance metrics for SPN-1 on the SPEED synthetic and real test-sets as well as for SPN-3 when tested on the PRISMA-21 dataset. The mean IoU values are similar across the SPEED synthetic and real-test sets and lower as compared to the Imitation-25 and PRISMA-25 datasets. This suggests that the 25 pose configurations comprising the Imitation-25 and PRISMA-25 datasets are a challenging subdomain of the 16,000 pose configurations sampled in the SPEED datasets. The E_R values of the actual camera images are higher than that of the synthetic images. The same trend can be seen once the E_R values are normalized for different camera models and relative distances to obtain $E_{R,\text{norm}}$. This gap in performance of the relative attitude is a sign of overfitting resulting from the constraint that the SPN-1 and SPN-3 were trained only on synthetic images. A similar performance gap can be seen in the mean E_T and $E_{T,\text{norm}}$ values when comparing the performance on PRISMA-25 and Imitation-25. However, in contrast to the relative attitude performance gap, the gap in relative position error is much smaller. Also, note that the E_T values for the SPEED real test-set are lower than that of the synthetic test-set since the relative distances for the actual camera images are comparatively much lower than that of the synthetic images. However, once the relative position errors are normalized using the relative distance to obtain the $E_{T,\text{norm}}$ metric, it is evident that the relative position error performance for SPN-1 is consistent across both synthetic and actual camera images.

Figure 5.18 shows the qualitative results of the 2D bounding box detection on the SPEED synthetic and real test-sets while Figure 5.19 shows qualitative results of the 2D bounding box detection and pose estimation for PRISMA-25 and Imitation-25. The region proposal network used in the SPN method was successful in detecting the 2D bounding box in images regardless of whether the Earth is visible in the background. In general, the bounding box detection worked marginally better on actual camera images as compared to synthetic images since all actual camera images had a black background. The bounding box detection was successful even in cases with sharp shadows as long as one side of the spacecraft was illuminated.

Figure 5.20a and Figure 5.20b show the cumulative distribution of the relative attitude accuracy for the SPN method across four different datasets: SPEED synthetic

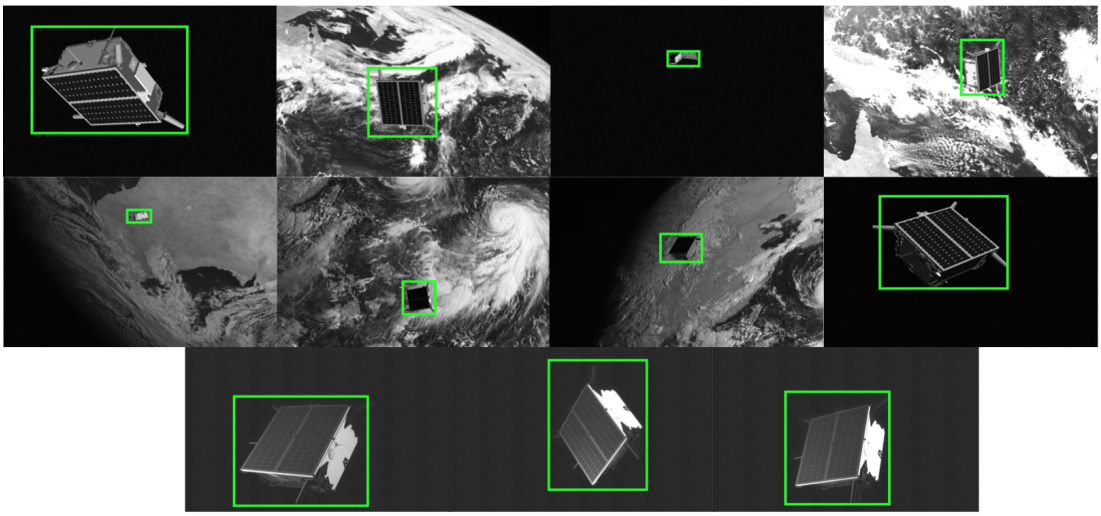


Figure 5.18: A montage of a few images with the 2D bounding box detections produced by the SPN method on the SPEED synthetic test-set.

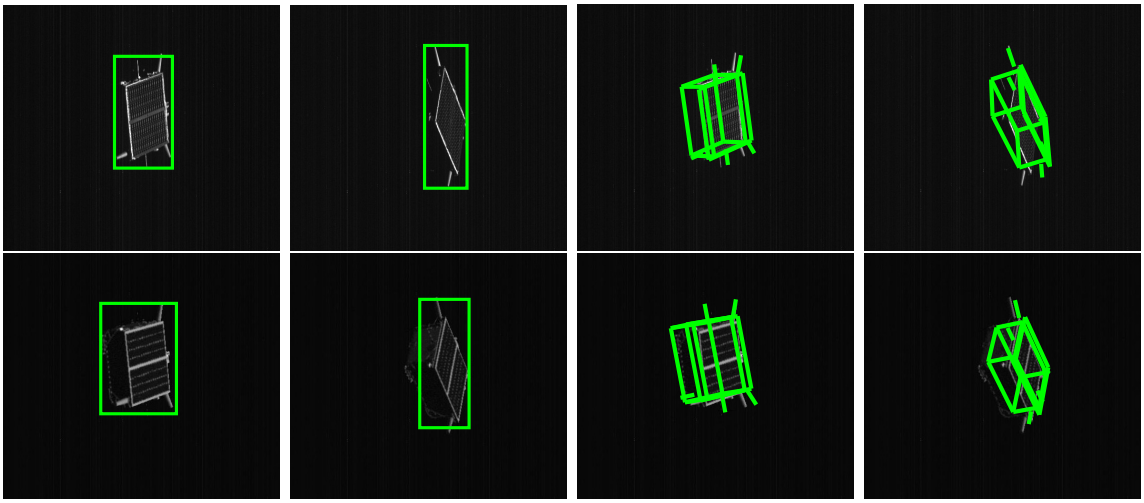
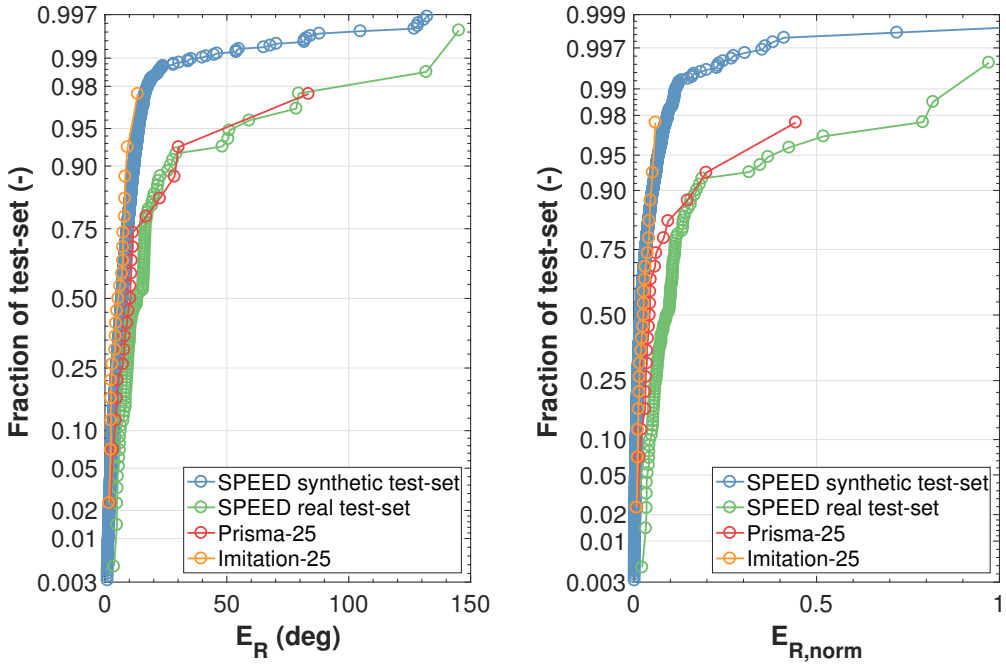


Figure 5.19: Estimated 2D bounding boxes and estimated pose for two images in PRISMA-25 (top) and the corresponding images in Imitation-25 (bottom).

test, SPEED real test, PRISMA-25, and Imitation-25. Figure 5.20a contains the trendlines for E_R and highlights the performance gap between the synthetic and real images when comparing the curves of the SPEED real test-set versus SPEED synthetic test-set and PRISMA-25 versus Imitation-25. Note that this performance gap is evident even in $E_{R,\text{norm}}$, which normalizes the attitude accuracy for different camera models and relative distance distributions used in the four datasets. It is noteworthy that the trendlines for PRISMA-25 are coincident with the Imitation-25 dataset for low values of relative attitude error while these are coincident with the SPEED real test-set for high relative attitude error. This suggests that the TRON facility and the OS camera emulator software can be used in a complementary manner to simulate the performance of pose estimation methods on actual flight imagery. Similar figures for $E_{T,\text{norm}}$ also show the same trend and have been omitted for brevity.



(a) (b)

Figure 5.20: Cumulative distribution of relative attitude accurcacy of the SPN method for the SPEED synthetic test-set, SPEED real test-set, PRISMA-25, and Imitation-25.

The bounding box detection tends to fail in cases where the target spacecraft was

in eclipse or when the target spacecraft was closer than ∼5 meters and not using the 3D model of the apogee motor. Figure 5.21 shows a few examples from both these failure modes. At relative distances less than ∼5 meters, the Tango spacecraft

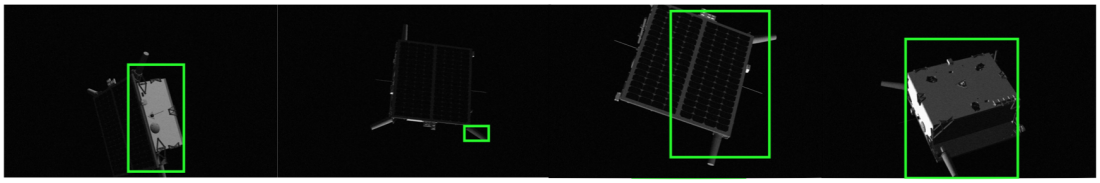


Figure 5.21: A montage of a few images from the SPEED synthetic test-set with inaccurate 2D bounding box detections.

starts getting clipped by the image boundaries leading to unpredictable behavior of the detection. In a docking scenario, this could potentially be resolved by performing pose estimation relative to the docking mechanism or another smaller fixture, instead of the entire spacecraft. Note that the SPN method can be extended to detect multiple 2D bounding boxes corresponding to the fixtures of interest and performing attitude estimation and position determination for each. The results for SPN-2 will elaborate further on this aspect of the SPN method.

To further examine such trends of pose accuracy versus relative distance, three performance metrics are plotted. In particular, the images from the SPEED synthetic test-set were grouped into batches of 100 each according to their ground truth relative distance, $||\mathbf{t_{BC}}||_2$. The mean value of the performance metric was then plotted against the mean ground truth relative distance for each batch.

Figure 5.22 shows the mean IoU values of the SPN method for the SPEED synthetic test-set plotted against the mean relative distance. The bounding box detection has the highest accuracy at ranges of ∼7 meters to ∼20 meters while there is a sharp drop-off in bounding box detection at relatively closer distances of less than ∼5 meters. In contrast, there is a gradual drop-off in performance at relatively farther distances as the target spacecraft starts occupying too few pixels in the image plane to allow for an accurate 2D bounding box detection. Degradation in performance at larger separations is also contributed to by the fact that the SPEED training-set contains more images at lower relative distances.

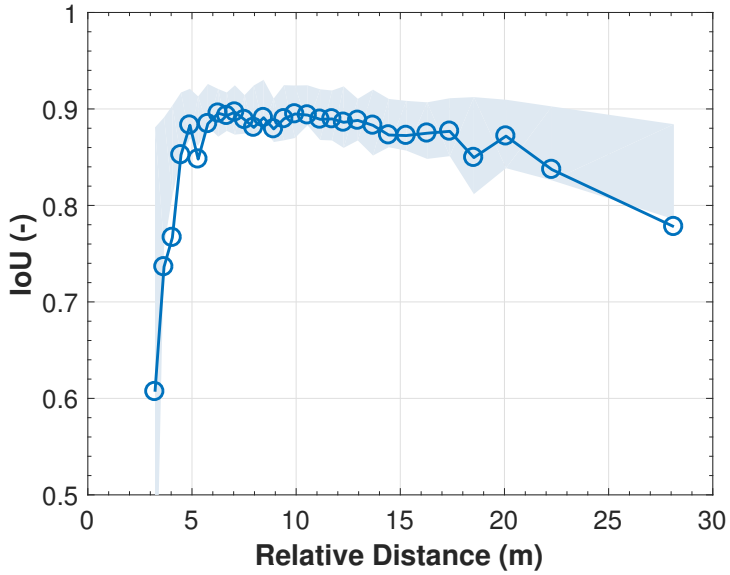


Figure 5.22: Mean IoU plotted against mean relative distance for the SPEED synthetic test-set. The shaded region shows the 25 and 75 percentile values.

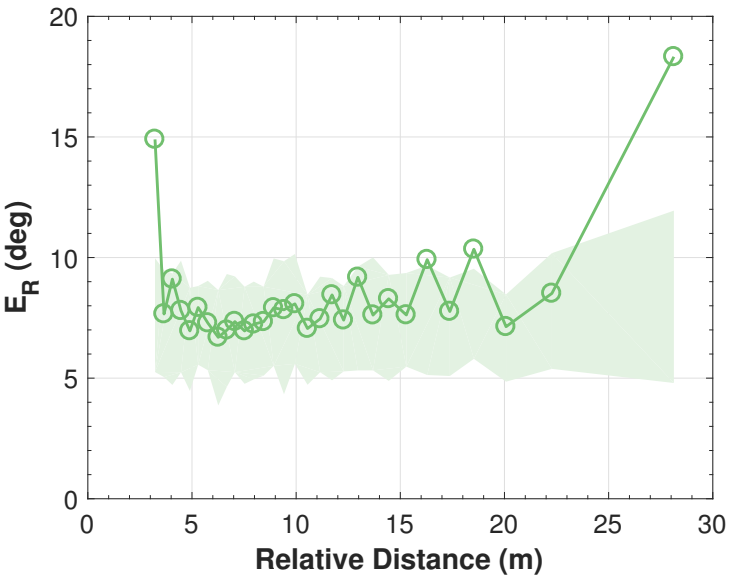


Figure 5.23: Mean E_R plotted against mean relative distance for the SPEED synthetic test-set. The shaded region shows the 25 and 75 percentile values.

Figure 5.23 shows the mean E_R values of the SPN method for the SPEED synthetic test-set plotted against the mean relative distance. The mean E_R values are between 5 and 10 degrees for most of the relative distances. The range of the relative distances where attitude estimation has the highest accuracy is similar to the bounding box detection. This makes sense since both the region proposal network and the two sets of fully connected layers share the output of shared convolutional layers in the SPN method. Unlike the bounding box detection, the attitude estimation has a sharp drop-off in performance at both high and low relative distances. This is due to the presence of more outliers in the attitude estimation as compared with bounding box detection at high relative distances.

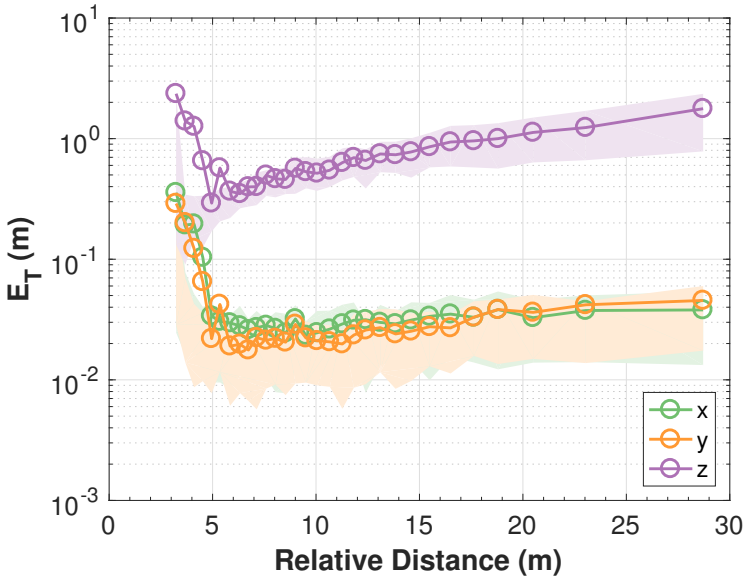


Figure 5.24: Mean E_T plotted against mean relative distance for the SPEED synthetic test-set. The shaded region shows the 25 and 75 percentile values.

The estimated relative attitude value for each image in the SPEED synthetic test-set was then combined with the corresponding 2D bounding box detection to produce estimated values of the relative position. Figure 5.24 shows the three components of the mean E_T values of the SPN method plotted against the mean relative distance. Note that the z-axis is aligned with the camera boresight direction while the x-axis and y-axis (lateral directions) are aligned with the image plane axes. The errors in

the lateral directions were an order of magnitude lower than the camera boresight direction, which implies that the bounding box detection was fairly successful at estimating the center of the bounding box as compared with the size of the bounding box. Trends in performance degradation of E_T mirror those observed for E_R and IoU.

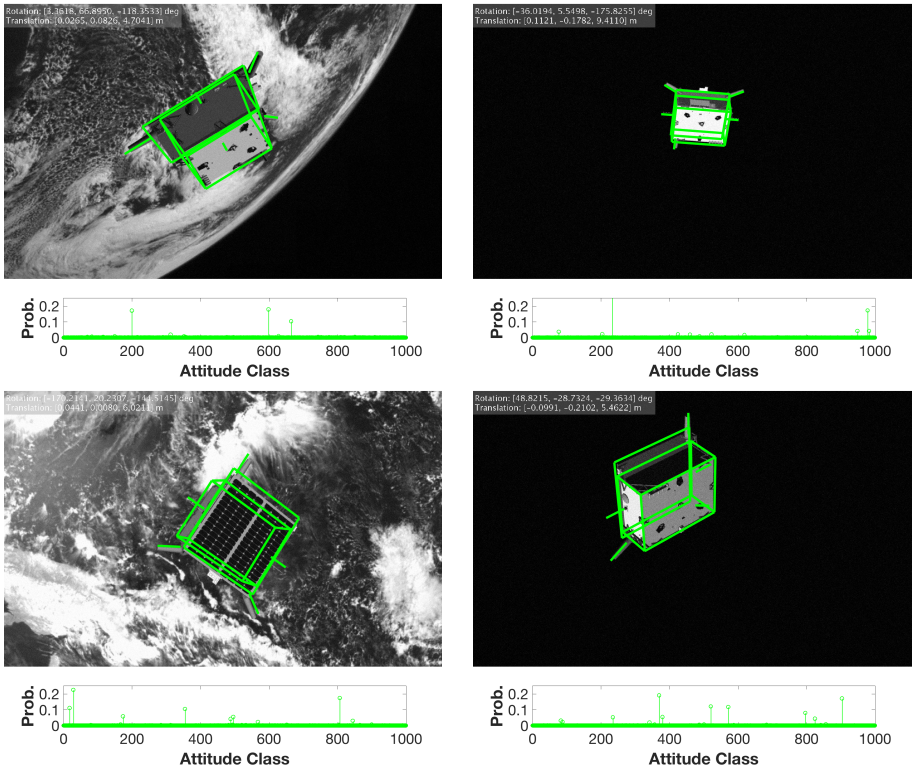


Figure 5.25: A montage of a few images with the pose solutions produced by the SPN method on the on the SPEED synthetic test-set.

Figure 5.25 and Figure 5.26 show some qualitative results of the pose estimated by the SPN method on the SPEED real and synthetic test-sets, respectively. The probability distribution output from the Branch 2 is also plotted for the respective images. In the future, the probability distribution allows the setting up of a confidence metric to reject outliers in addition to being used by a navigation filter to accumulate information from sequential images to provide a more accurate estimate at a high rate. Predictably, the peaks in the probability distribution are lower for the SPEED real test-set as compared with the SPEED synthetic test-set since the convolutional neural network has overfitted the synthetic images in the SPEED training-set to some

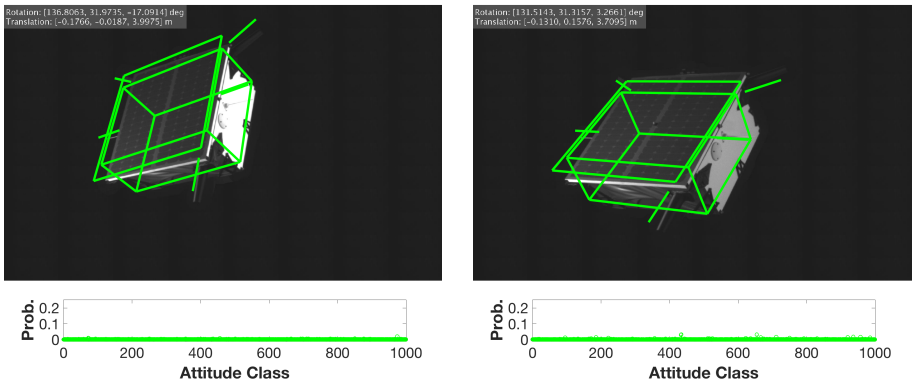


Figure 5.26: A montage of a few images with the pose solutions produced by the SPN method on the on the SPEED real test-set.

extent. This could be addressed by a stronger L2 regularization parameter as well as augmenting the training-set with images containing randomized textures of the target spacecraft similar to work in the area of domain adaptation[114, 115] and domain randomization[116, 117].

Target-in-target pose estimation and uncertainty quantification

To demonstrate two key features of the SPN method, namely, target-in-target pose estimation and uncertainty quantification, SPN-1 and SPN-2 were tested on the SPEED synthetic and Apogee test datasets, respectively. Figure 5.27 shows the mean values of E_R and E_T for the Apogee motor test set for SPN-2. Recall that SPN-2 was trained to estimate the pose using the 3D models of Tango and the apogee motor attached to Tango, simultaneously. Figure 5.27 compares the performance of SPN-2 when it used both these 3D models at inference versus when it only used the Tango 3D model. Predictably the translation and attitude errors rise at a low inter-spacecraft distance when SPN-2 only used the Tango 3D model. At inter-spacecraft distances smaller than 5 meters, the Tango spacecraft can appear occluded in the image plane. This adversely affects the detection of the 2D bounding box around Tango, which in turn deteriorates the relative position and attitude estimation. Therefore, when SPN-2 is allowed to use the 3D model of the apogee motor attached to Tango, the rotation and translation errors are comparatively lower. Figure 5.28 shows the re-projections of the estimated pose for SPN-2 and SPN-2 (Tango only) for an image captured at

an inter-spacecraft distance of 1.5 meters.

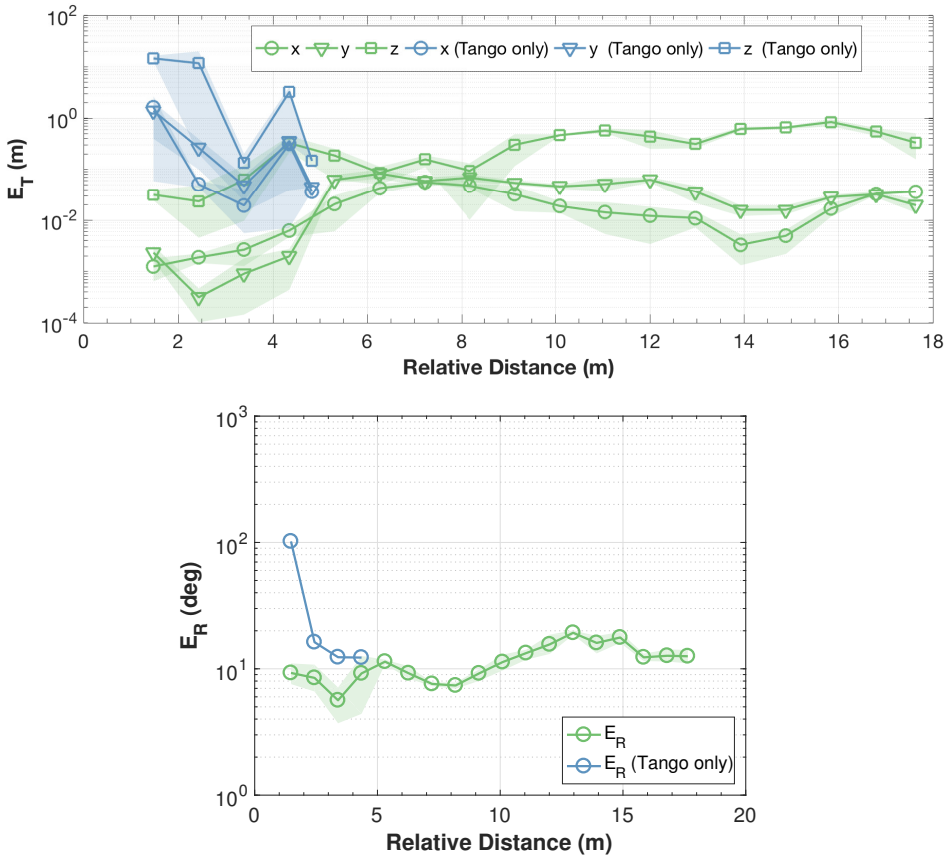
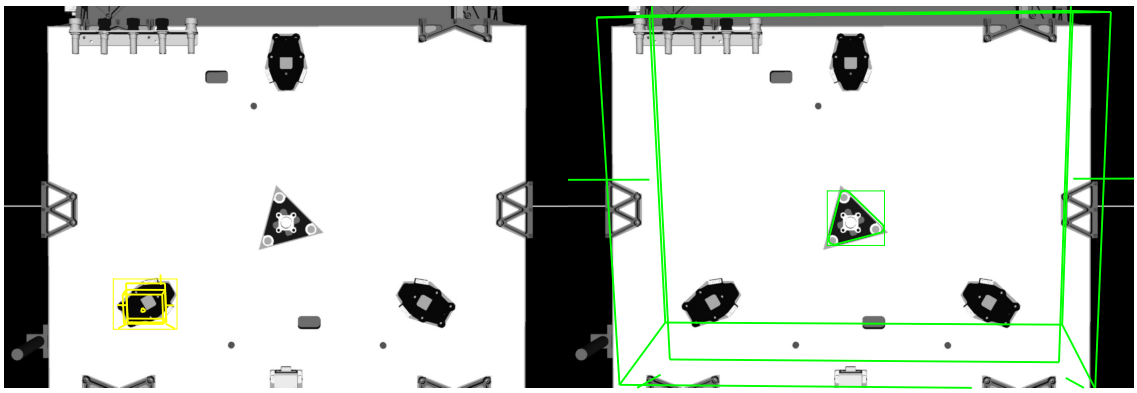


Figure 5.27: Mean E_T and E_R for SPN-2 (in green) and SPN-2 when using Tango 3D model only (in blue) plotted against mean relative distance for the Apogee test set. The shaded region shows the 25 and 75 percentile values.

Figure 5.29 shows the estimated uncertainty (1σ) for E_R and E_T for SPN-1 when tested on the synthetic test set. Note that the x-axes for these plots are image indices sorted according to their values of E_R and $E_{T,z}$. Just looking at the error curves, it is evident that majority of the errors in relative position are below 0.25 m for $E_{T,x}$ and $E_{T,y}$; below 0.5 m for $E_{T,z}$; and below 10 degrees for E_R. The uncertainty curves for the corresponding errors capture these majority inlier cases as well as the high error values for the outlier cases.



(a) (b)

Figure 5.28: Comparison of the pose estimation results between SPN-2 (right) and SPN-2 when using Tango 3D model only (left) using an image from the Apogee motor test set.

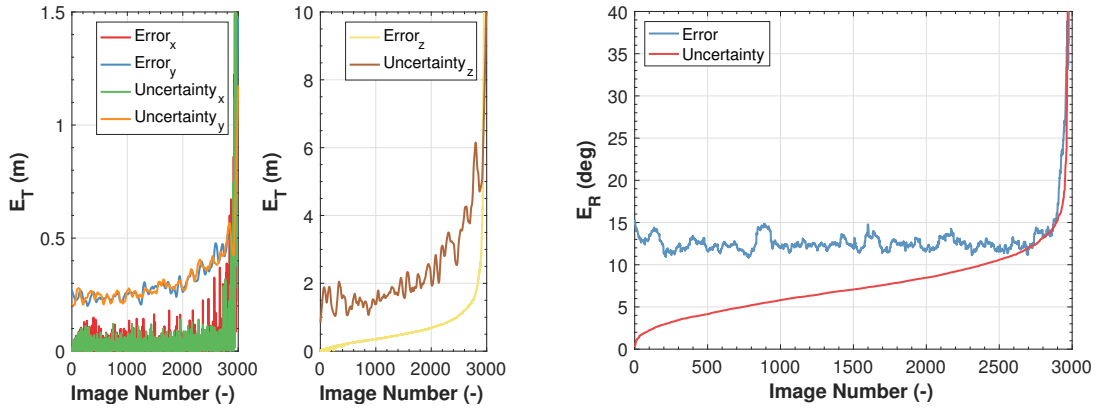


Figure 5.29: Sorted values of the E_T (left) and E_R (right) along with the corresponding estimated uncertainty (1σ) for SPN-1 when tested on the SPEED synthetic test set.

Comparison with CNN-based and SVD methods

The network SPN-3 was tested on the Imitation-25 dataset to compare the performance of the SPN method against the CNN-based developed in Chapter 4 and the SVD method developed in Chapter 3. Table 5.5 presents the results for all three methods. Note that the results for the CNN-based method in the table are the results obtained using the best performing network, "net9", introduced in Chapter 4. As compared to the feature-based methods, it clear that the deep learning-based methods dominate in the computational runtime criteria. Moreover, SPN-3 is superior to the other methods for relative position accuracy in the lateral directions, i.e., the first two elements of E_T. While SPN-3 has a lower level of relative position accuracy along the boresight direction as compared to net9, it has a higher level of relative attitude accuracy. Recall that net9 was explicitly trained to classify the relative position between 8 and 12 meters by training the CNN on 150,000 images. In comparison, the SPN method only used 12,000 images for training and provided similar levels of relative position accuracy between 1 and 18 meters, as evidenced by the results of SPN-2. In comparison to the SVD method, SPN-3 provides superior levels of relative attitude error and relative position error across the board. As compared to the SVD high confidence solutions, SPN-3 provides inferior levels of relative attitude accuracy and relative position accuracy along the camera boresight direction. However, unlike the SVD high confidence solutions, which are only available for 20% of the images, the relative attitude solution from the SPN method is available for every image.

Table 5.5: Performance of the SPN method, CNN-based (net9), and SVD methods on the Imitation-25 dataset.

Method	Mean E_R (deg)	Mean E_T (m)	Solution Availability (%)	Mean runtime (s)
SVD	38.99	[0.13 0.12 1.46]	50	8.22
SVD (high conf.)	2.76	[0.14 0.06 0.51]	20	8.93
net9	17.12	[- - 0.44]	100	0.016
net9 (high conf.)	16.48	[- - 0.43]	86	0.016
SPN-3	5.749	[0.03 0.03 0.83]	100	0.055

5.4 Summary

This chapter introduced the SPN method for estimating the relative pose of a target spacecraft using a single grayscale image without requiring a-priori pose information. The SPN method makes the novel use of a hybrid classification and regression technique to estimate the relative attitude. The SPN method then leverages the geometric knowledge of the perspective transformation and the 2D bounding box detected from a state-of-the-art object detector to estimate the relative position using a Gauss-Newton algorithm. In this manner, the SPN method exploits the synergies of deep learning and "conventional" feature-based methods. This chapter also introduced SPEED, a publicly available dataset to allow for the training and validation of monocular pose estimation methods for spaceborne applications. The SPEED training-set consists of 12000 augmented reality images created by fusing synthetic images of a target spacecraft with meteorological images of the Earth. Apart from the 3000 images of the Tango spacecraft in the SPEED synthetic test-set, 900 actual camera images of the same spacecraft are provided in the SPEED real test-set. The subsequent application of the SPN method on the SPEED synthetic test-set produces degree-level mean relative attitude errors and centimeter-level mean relative position errors exceeding the performance of conventional feature-based methods used in previous work. The relative attitude accuracy was slightly lower in the actual camera images as compared to corresponding synthetic imagery. This signals that the SPN method does suffer from slight overfitting due to being constrained to use only synthetic images for training. However, the performance in terms of relative position accuracy carried over to the actual camera images without any performance gap as compared to the synthetic images.

Chapter 6

Conclusions

This chapter succinctly reviews the contributions of this research and presents some possible directions for future work. The primary objective of this research was to develop methods to estimate the pose, i.e., the relative position and attitude, of a target spacecraft using a single monocular image. This problem statement assumed that apart from a simple three-dimensional (3D) wireframe model of the target spacecraft, no other a-priori information is available. In particular, it was assumed that no prior information about the pose or relative pose dynamics is available to aid in pose estimation. The above assumptions allowed the research to identify gaps and progress the current state-of-the-art monocular vision-based pose estimation methods.

6.1 Review of Contributions

The primary contribution of this research to the state-of-the-art is the development of the Spacecraft Pose Network (SPN) method for pose estimation. This method combines several advancements in conventional feature-based and modern learning-based algorithms in order to provide pose solutions with high accuracy and robustness. The SPN method and the advancements that this research made along the way form four main contributions to the state-of-the-art in monocular pose estimation. Two contributions are in the form of algorithms, namely the Sharma-Ventura-D'Amico (SVD) method and the SPN method. The third contribution is in the form of a systematic review of state-of-the-art solvers of the Perspective-n-Point (PnP) problem.

The fourth contribution is in the form of the Spacecraft Pose Estimation Dataset (SPEED). The four contributions are discussed below.

6.1.1 Comparative assessment of PnP solvers

As part of Chapter 2, this dissertation addresses the dearth of comparative functional and performance characterization of PnP solvers in the literature. The chapter starts by systematically reviewing the state-of-the-art solutions of the perspective equations, i.e., the Perspective-n-Point (PnP) solvers. These solvers form the backbone of all "conventional" feature-based pose estimation methods. The review was followed by analyzing the performance sensitivity of the solvers to the image properties characteristic of actual flight imagery taken during previous missions. These properties included the number of two-dimensional (2D) input features, image sensor noise, number of 2D feature outliers, and inter-spacecraft separation. The results of the Monte-Carlo simulations were used to provide recommendations on the relative advantages and disadvantages of the PnP solvers in the form of a decision matrix. These recommendations informed the design and development of a new feature-based pose estimation method in Chapter 3.

6.1.2 SVD Method

As shown in Chapter 3, the Sharma-Ventura-D'Amico (SVD) method pushes the state-of-the-art in feature-based monocular pose estimation in two directions. Firstly, the SVD method introduces a hybrid approach to image processing by fusing the novel Weak Gradient Elimination (WGE) technique with the Sobel operator followed by Hough transform to detect both small and large features of the target spacecraft. Secondly, the SVD method introduces feature grouping for solving the feature correspondence problem. By grouping edge features into more complex feature types such as polygons, the SVD method reduces the search space for finding the correspondences between the features of the 3D wireframe model and the 2D image. Experiments using images captured during the close-range navigation phase of the PRISMA mission were conducted on a 2.4 GHz Intel Core i5-4570T processor. These

experiments showed that the WGE technique output of edge endpoints had the highest true positive rate as compared to state-of-the-art edge detectors and key-point detectors. The experiments also demonstrated that the SVD method required 8.22 seconds to produce a pose solution as compared to 13.46 seconds for a state-of-the-art random consensus method.

6.1.3 SPEED

Due to the absence of public datasets relevant for space-borne applications, there is a dearth of common benchmarks to compare existing and new pose estimation methods. The Spacecraft Pose Estimation Dataset (SPEED) is the first large-scale publicly available dataset for training, validation, and testing of pose estimation methods for spaceborne applications. SPEED contains 15000 augmented-reality and 900 actual camera images of the Tango spacecraft. These images and the associated ground-truth pose information was a result of the development of two complementary sources of high fidelity monocular imagery. The first source uses synthetic images of the Tango spacecraft and meteorological images of the Earth to produce high-fidelity augmented-reality images. The generation of synthetic images using OpenGL and publicly available high-resolution images of the Earth provides a platform to generate such images rapidly. The addition of Gaussian blurring, white noise, and contrast normalization is used to emulate the effects of the depth of field, shot noise, and sensor saturation, respectively. The second source provides actual camera images captured using the Testbed for Rendezvous and Optical Navigation (TRON) facility at Stanford University. TRON provides images of a 1:1 mock-up model of the Tango spacecraft of the PRISMA mission using an actual Point Grey Grasshopper 3 camera with a Xenoplan 1.9/17mm lens. Motion capture cameras were calibrated using manual annotation of a few images to provide degree-level and cm-level ground truth pose information for each image. Due to the inclusion of two complementary sources of images, SPEED also provides a unique opportunity to evaluate the transferability of pose estimation algorithms from synthetic images to actual camera images. SPEED was made publicly available to the community through an international pose estimation challenge, organized in collaboration with the Advanced Concepts Team of the

European Space Agency. SPEED was made publicly available[1] to the community through an international pose estimation challenge, organized in collaboration with the Advanced Concepts Team of the European Space Agency. The dataset images and associated pose information can be downloaded from the challenge website.

6.1.4 SPN Method

The SPN method is the main contribution of this research and pushes the state-of-the-art in monocular pose estimation in multiple ways.

1. The method uses a hybrid approach of classification followed by regression to infer the relative attitude using a Convolutional Neural Network (CNN). The relative position is estimated by combining the output of the CNN with a Gauss-Newton algorithm. When tested on synthetic imagery captured by simulating the close-range navigation imagery captured during the PRISMA mission, the SPN method produced lower mean relative attitude and relative position errors as compared to state-of-the-art feature-based and deep learning-based methods.

2. Uncertainty in the pose output can be estimated using the probability distribution output of the CNN and the Jacobian matrix of the Gauss-Newton algorithm. In contrast to state-of-the-art deep learning-based methods, this allows for the detection of outlier pose outputs and easier integration of the SPN method into a navigation filter.

3. The method can be extended to perform target-in-target pose estimation by simultaneously training the method with multiple target geometries. This allows for pose estimation when only a partial view of the target spacecraft is available due to the small inter-spacecraft range and the limited field-of-view of the camera. Hence, the pose of the target spacecraft can be continuously estimated during terminal stages of the docking maneuver without the need for switching to a different sensor.

4. In contrast to state-of-the-art methods that use both actual camera and synthetic imagery at the time of training, the SPN method can be trained solely

[1] `https://kelvins.esa.int/satellite-pose-estimation-challenge/`

on synthetic images. As compared with testing on synthetic images, the SPN method shows only a slight degradation in relative pose accuracy when tested on actual camera imagery captured during the close-range navigation phase of the PRISMA mission.

6.2 Directions for Future Work

Further work is required in a few different directions. Most notably, the pose estimation methods proposed in this dissertation need to be embedded in flight-grade hardware to profile its computational runtime and memory usage. Secondly, the performance of these methods on sequential images needs to be validated. In particular, each of the methods can be used to initialize or integrate with a navigation filter. Additional directions for research can be explored that relate to the assumption of the accuracy and availability of the 3D model of the target spacecraft before or during pose estimation. Finally, further branches of pose estimation research could be explored that deal with any combination of the following: multiple images, multiple cameras, multiple sensor types, knowledge of relative dynamics. More specific directions for future work for the SVD and SPN pose estimation methods are listed below.

- SVD Method

    1. Notably, up to 92% of the runtime was contributed by EPnP calls to determine the correct feature correspondences between the image and the 3-D model. Therefore, there would be merit in further developing techniques to detect complex feature groups; for example, separate Hough transforms could be employed for each complex geometric shape.
    2. Exploit feature detection from multiple images to estimate the pose if a complex feature group is not detected in the first image.
    3. Aiding pose estimation at close-range by the estimation of the relative orbit through azimuth and elevation angle measurements made at the far-range.
    4. Output of the feature detection is processed to merge any partially detected edges, and this has vastly improved the quality of edge detection as

compared to previous work. Future work in merging edges with alternative features types can be explored to make it more robust.

- SPN Method

  1. Data augmentation techniques and stronger regularization during training are required to bridge the performance gap of the SPN method between synthetic and real imagery.
  2. Tighter coupling with PnP methods can be explored to improve the accuracy further. For example, image features that correspond to known 3D features can be done before using the PnP solvers to estimate the pose.
  3. The use of domain and texture randomization can be used to generate additional synthetic imagery. The use of this imagery during training could allow a CNN to be more robust to mismatches in texture and reflection properties of the target spacecraft in the synthetic imagery.

Bibliography

[1] Simone D'Amico, Mathias Benn, and John L. Jørgensen. Pose estimation of an uncooperative spacecraft from actual space imagery. *International Journal of Space Science and Engineering*, 2(2):171, 2014.

[2] Connor Beierle and Simone D'Amico. Variable Magnification Optical Stimulator for Training and Validation of Spaceborne Vision-Based Navigation. *Journal of Spacecraft and Rockets*, 56(4):1060–1072, 2018.

[3] Robert D. Langley. Apollo Experience Report - The Docking System. Technical report, NASA, Houston, Texas, 1972.

[4] Douglas Zimpfer, Peter Kachmar, and Seamus Tuohy. Autonomous Rendezvous, Capture and In-Space Assembly:Past, Present and Future. (February):1–12, 2013.

[5] Jason L. Forshaw, Guglielmo S. Aglietti, Nimal Navarathinam, Haval Kadhem, Thierry Salmon, Aurélien Pisseloup, Eric Joffre, Thomas Chabot, Ingo Retat, Robert Axthelm, Simon Barraclough, Andrew Ratcliffe, Cesar Bernal, François Chaumette, Alexandre Pollini, and Willem H. Steyn. RemoveDEBRIS: An in-orbit active debris removal demonstration mission. *Acta Astronautica*, 127(2016):448–463, 2016.

[6] Brook Sullivan, David Barnhart, Lisa Hill, Paul Oppenheimer, Bryan L. Benedict, Gerrit Van Ommering, Laurie Chappell, John Ratti, and Peter Will. DARPA Phoenix Payload Orbital Delivery System (PODs): FedEx to GEO. *AIAA SPACE 2013 Conference and Exposition*, pages 1–14, 2013.

[7] Benjamin B. Reed, Robert C. Smith, Bo J. Naasz, Joseph F. Pellegrino, and Charles E. Bacon. The Restore-L Servicing Mission. In *AIAA Space Forum*, pages 1–8, Long Beach, CA, 2016.

[8] Jacopo Ventura. *Autonomous Proximity Operations for Noncooperative Space Target*. PhD thesis, Technical University of Munich, 2016.

[9] Sumant Sharma, Jacopo Ventura, and Simone D'Amico. Robust Model-Based Monocular Pose Initialization for Noncooperative Spacecraft Rendezvous. *Journal of Spacecraft and Rockets*, 55(6):1414–1429, 2018.

[10] Heike Benninghoff, Toralf Boge, and Tristan Tzschichholz. Hardware-in-the-loop rendezvous simulation involving an autonomous guidance, navigation and control system. *Advances in the Astronautical Sciences*, 145:953–972, 2012.

[11] Alessio A. Grompone. *Vision-Based 3D Motion Estimation for On-Orbit Proximity Satellite Tracking and Navigation*. PhD thesis, Naval Postgraduate School, 2015.

[12] Bo J. Naasz, John Van Eepoel, Steven Z. Queen, C. Michael Southward, and Joel Hannah. Flight results from the HST SM4 Relative Navigation Sensor system. In *33rd Annual AAS Guidance and Control Conference*, Breckenridge, CO, USA, 2010.

[13] Chiun-Hong Chien and Kenneth Baker. Pose Estimation For Servicing of Orbital Replacement Units in a Cluttered Environment. In *Proceedings of the 2004 IEEE International Conference on Robotics & Automation*, number April, pages 5141–5146, New Orleans, LA, USA, 2004.

[14] Son-Goo Kim, John L. Crassidis, Yang Cheng, Adam M. Fosbury, and John L. Junkins. Kalman Filtering for Relative Spacecraft Attitude and Position Estimation. *Journal of Guidance, Control, and Dynamics*, 37(5):1706–1711, 2014.

[15] John Valasek, Kiran Gunnam, Jennifer Kimmett, Monish D Tandale, John L Junkins, and Declan Hughes. Vision-based Sensor and Navigation System for Autonomous Air Refueling. *Journal of Guidance Control and Dynamics*, 28(5):979–989, 2005.

[16] Adam Fosbury and John Crassidis. Relative Navigation of Air Vehicles. *Journal of Guidance, Control, and Dynamics*, 31(4):824–834, 2008.

[17] Alex Cropp and Phil Palmer. Pose Estimation and Relative Orbit Determination of a Nearby Target Microsatellite using Passive Imagery. *5th Cranfield Conference on Dynamics and Control of Systems and Structures in Space 2002*, pages 389–395, 2002.

[18] Mathias Benn. *Vision Based Navigation Sensors for Spacecraft Rendezvous and Docking*. PhD thesis, Danish Technical University, 2010.

[19] Keyvan Kanani, Antoine Petit, Eric Marchand, Thomas Chabot, and Bernard Gerber. Vision Based Navigation for Debris Removal Missions. *63rd International Astronautical Congress*, pages 1–8, 2012.

[20] Chang Liu and Weiduo Hu. Relative pose estimation for cylinder-shaped spacecrafts using single image. *IEEE Transactions on Aerospace and Electronic Systems*, 50(4):3036–3056, 2014.

[21] Vincenzo Capuano, Giovanni Cuciniello, Vincenzo Pesce, Roberto Opromolla, Salvatore Sarno, Michelle Lavagna, Michele Grassi, Federico Corraro, Giuseppe Capuano, Paolo Tabacco, Francesco Meta, Maria Libera Battagliere, and Tuozzi Alberto. VINAG: A highly integrated system for autonomous on-board absolute and relative spacecraft navigation. In *The 4S Symposium 2018*, number 1, 2018.

[22] J.M. Kelsey, J. Byrne, M. Cosgrove, S. Seereeram, and R.K. Mehra. Vision-based relative pose estimation for autonomous rendezvous and docking. *2006 IEEE Aerospace Conference*, 2006.

[23] Sumant Sharma and Simone D'Amico. Comparative assessment of techniques for initial pose estimation using monocular vision. *Acta Astronautica*, 123:435–445, 2015.

[24] Paolo Lunghi, Luca Losi, Vincenzo Pesce, and Michéle Lavagna. Ground testing of vision-based GNC systems by means of a new experimental facility. In *69th International Astronautical Congress (IAC)*, pages 1–15, Bremen, Germany, 2018. IAF.

[25] Antoine Petit, Eric Marchand, and Keyvan Kanani. Vision-based space autonomous rendezvous: A case study. *IEEE International Conference on Intelligent Robots and Systems*, pages 619–624, 2011.

[26] Shijie Zhang and Xibin Cao. Closedform solution of monocular visionbased relative pose determination for RVD spacecrafts. *Aircraft Engineering and Aerospace Technology*, 77(3):192–198, 2005.

[27] M. Avilés, D. Mora, M. Canetri, and P. Colmenarejo. A Complete IP-based Navigation Solution for the Approach and Capture of Active Debris. In *67th International Astronautical Congress*, pages 1–8, 2016.

[28] Sumant Sharma and Simone D'Amico. Reduced-Dynamics Pose Estimation for Non-Cooperative Spacecraft Rendezvous using Monocular Vision. In *Proceedings of the 40th Annual AAS Rocky Mountain Section Guidance and Control Conference*, pages 1–25, Breckenridge, CO, 2017.

[29] Michel Dhome, Marc Richetin, Jean Thierry Lapresteé, and Geérard Rives. Determination of the Attitude of 3-D Objects from a Single Perspective View. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(12):1265–1278, 1989.

[30] Steven Gold, Chien Ping Lui, and Anand Rangarajan. New Algorithms for 2D and 3D Point Matching : Pose Estimation and Correspondence. *Pattern Recognition*, 31(8), 1999.

[31] Philip David, Daniel Dementhon, Ramani Duraiswami, and Hanan Samet. SoftPOSIT: Simultaneous pose and correspondence determination. *International Journal of Computer Vision*, 59(3):259–284, 2004.

[32] Mouna Attia, Yosr Slama, and Mohamed Amine Kamoun. On Performance Evaluation of Registration Algorithms for 3D Point Clouds. *Proceedings - Computer Graphics, Imaging and Visualization: New Techniques and Trends, CGiV 2016*, pages 45–50, 2016.

[33] Faraz M Mirzaei and Stergios I Roumeliotis. Globally Optimal Pose Estimation from Line-Plane Correspondences. *Science*, (612):5581–5588, 2010.

[34] Chi Xu, Lilian Zhang, Li Cheng, and Reinhard Koch. Pose Estimation from Line Correspondences: A Complete Analysis and a Series of Solutions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6):1209–1222, 2017.

[35] Adnan Ansar and Kostas Daniilidis. Linear pose estimation from points or lines. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):578–589, 2003.

[36] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. EPnP: An accurate O(n) solution to the PnP problem. *International Journal of Computer Vision*, 81(2):155–166, 2009.

[37] Daniel F. Dementhon and Larry S. Davis. Model-based object pose in 25 lines of code. *International Journal of Computer Vision*, 15(1-2):123–141, 1995.

[38] Antoine Petit, Keyvan Kanani, and Eric Marchand. Vision-based Detection and Tracking for Space Navigation in a Rendezvous Context. *Int. Symp. on Artificial Intelligence, Robotics and Automation in Space, i-SAIRAS˜2012*, 2012.

[39] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing*. Prentice Hall, Upper Saddle River, NJ, USA, 2002.

[40] Chris Harris and Mike Stephens. A Combined Corner and Edge Detector. *Procedings of the Alvey Vision Conference 1988*, pages 147–151, 1988.

[41] R O Duda and P E Hart. Use of the Hough transform to detect lines and cures in pictures. *Communications of the Association Computing Machinery*, 15(1), 1972.

[42] Per Bodin, Ron Noteborn, Robin Larsson, Thomas Karlsson, Simone D'Amico, Jean-Sébastien Ardaens, Michel Delpech, and Jean Claude Berges. The prisma formation flying demonstrator: Overview and conclusions from the nominal mission. *Advances in the Astronautical Sciences*, 144:441–460, 2012.

[43] Sumant Sharma, Connor Beierle, and Simone D'Amico. Pose Estimation for Non-Cooperative Spacecraft Rendezvous Using Convolutional Neural Networks. In *2018 IEEE Aerospace Conference*, pages 1–12, Big Sky, USA, 2018. IEEE.

[44] Hao Su, Charles R. Qi, Yangyan Li, and Leonidas J. Guibas. Render for CNN: Viewpoint estimation in images using CNNs trained with rendered 3D model views. *Proceedings of the IEEE International Conference on Computer Vision*, pages 2686–2694, 2016.

[45] Shubham Tulsiani and Jitendra Malik. Viewpoints and keypoints. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1510–1519, 2015.

[46] Sumant Sharma, Connor Beierle, and Simone D'Amico. Towards Pose Determination for Non-Cooperative Spacecraft using Convolutional Neural Networks. In *Proceedings of the 1st IAA Conference on Space Situational Awareness (ICSSA)*, pages 1–5, 2017.

[47] Siddharth Mahendran, Haider Ali, and Rene Vidal. 3D Pose Regression Using Convolutional Neural Networks. *Proceedings - 2017 IEEE International Conference on Computer Vision Workshops, ICCVW 2017*, pages 2174–2182, 2017.

[48] Alex Kendall, Matthew Grimes, and Roberto Cipolla. PoseNet: A convolutional network for real-time 6-dof camera relocalization. *Proceedings of the IEEE International Conference on Computer Vision*, 2015 Inter:2938–2946, 2015.

[49] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes. 2017.

[50] Paul Besl and Neil McKay. A Method for Registration of 3-D Shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

[51] Georgios Georgakis, Arsalan Mousavian, Alexander C. Berg, and Jana Kosecka. Synthesizing Training Data for Object Detection in Indoor Scenes. 2017.

[52] Debidatta Dwibedi, Ishan Misra, and Martial Hebert. Cut, Paste and Learn: Surprisingly Easy Synthesis for Instance Detection. *Proceedings of the IEEE International Conference on Computer Vision*, 2017-Octob:1310–1319, 2017.

[53] Vincent Lepetit. BB8 : A Scalable , Accurate , Robust to Partial Occlusion Method for Predicting. *Proceedings of the IEEE International Conference on Computer Vision*, pages 3828–3836, 2017.

[54] Martin A. Fischler and Robert C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applicatlons to Image Analysis and Automated Cartography. *Communications of the ACM*, 24(6):381 – 395, 1981.

[55] Vincent Lepetit and Pascal Fua. Monocular Model-Based 3D Tracking of Rigid Objects. *Foundations and Trends in Computer Graphics and Vision*, 1(1):1–89, 2005.

[56] Yinqiang Zheng, Yubin Kuang, Shigeki Sugimoto, Kalle Astrom, and Masatoshi Okutomi. Revisiting the PnP problem: A fast, general and optimal solution. *Proceedings of the IEEE International Conference on Computer Vision*, pages 2344–2351, 2013.

[57] Chien Ping Lu, Gregory D. Hager, and Eric Mjolsness. Fast and globally convergent pose estimation from video images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(6):610–622, 2000.

[58] Giampiero Campa, Marco Mammarella, Marcello R. Napolitano, Mario L. Fravolini, Lorenzo Pollini, and Brian Stolarik. A comparison of pose estimation algorithms for machine vision based aerial refueling for UAVs. *14th Mediterranean Conference on Control and Automation, MED'06*, 2006.

[59] Denis Oberkampf, Daniel F. DeMenthon, and Larry S. Davis. Iterative Pose Estimation Using Coplanar Feature Points. *Computer Vision and Image Understanding*, 63(3):495–511, 1996.

[60] Eyes on the Solar System. NASA/JPL-Caltech. Nasa 3d resources: Rosetta. `https://nasa3d.arc.nasa.gov/detail/eoss-rosetta`. Accessed May 1, 2019.

[61] David A Whelan, E Allen Adler, Samuel B Wilson III, and Gordon M Roesler Jr. Darpa orbital express program: effecting a revolution in space-based systems. In *International {Symposium} on {Optical} {Science} and {Technology}*, pages 48–56. International Society for Optics and Photonics, 2000.

[62] Prithi Tissainayagam and Davied Suter. Assessing the performance of corner detectors for point feature tracking applications. *Image and Vision Computing*, 22(8):663–679, 2004.

[63] Xinhao Liu, Masayuki Tanaka, and Masatoshi Okutomi. Single-image noise level estimation for blind denoising. *IEEE Transactions on Image Processing*, 22(12):5226–5237, 2013.

[64] Yann LeCun, B Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation Applied to Handwritten Zip Code Recognition, 1989.

[65] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. *Advances In Neural Information Processing Systems*, pages 1–9, 2012.

[66] Dominik Scherer, Andreas Müller, and Sven Behnke. Evaluation of pooling operations in convolutional architectures for object recognition. In *International conference on artificial neural networks*, pages 92–101, 2010.

[67] David E. Rumelhart, Richard Durbin, Richard Golden, and Yves Chauvin. Backpropagation: The basic theory. *Backpropagation: Theory, architectures, and applications*, pages 1–34, 1995.

[68] Yann Lecun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[69] Aaron Courville Ian Goodfellow, Yoshua Bengio. Deep Learning. *MIT Press*, 521(7553):785, 2017.

[70] Ning Qian. On the momentum term in gradient descent learning algorithms. *Neural Networks*, 12(1):145–151, 1999.

[71] Sebastian Ruder. An overview of gradient descent optimization algorithms. *ArXiv:1609.04747*, pages 1–14, 2016.

[72] Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of ADAM and beyond. *ArXiv:1904.09237*, pages 1–23, 2019.

[73] a Yilmaz, O Javed, and M Shah. Object tracking. 38(4):13–es, 2006.

[74] John Canny. A Computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, 1986.

[75] J Matas, O Chum, M Urban, and T Pajdla. Robust Wide Baseline Stereo from Maximally Stable Extremal Regions. *British Machine Vision Conference*, pages 384–393, 2002.

[76] Janne Heikkilä and Olli Silvén. A Four-step Camera Calibration Procedure with Implicit Image Correction. *Cvpr*, pages 1106–1112, 1997.

[77] Jms Prewitt. Object enhancement and extraction. *Picture processing and Psychopictorics*, 10(1):15–19, 1970.

[78] O Barinova, V Lempitsky, and P Kohli. On detection of multiple object instances using Hough transform. 34(9):1773–1784, 2010.

[79] Joshua Sullivan, Adam Koenig, and Simone D'Amico. Improved Maneuver-Free Approach To Angles-Only Navigation for Space Rendezvous. *26th AAS/AIAA Space Flight Mechanics Meeting*, pages 1–24, 2016.

[80] David G. Lowe. Three-dimensional object recognition from single two-dimensional images. *Artificial Intelligence*, 31(3):355–395, 1987.

[81] H. Borotschnig, L. Paletta, M. Prantl, and a. Pinz. Appearance-based active object recognition. *Image and Vision Computing*, 18(9):715–727, 2000.

[82] J.H.M. Byne and J.a.D.W. Anderson. A CAD-based computer vision system. *Image and Vision Computing*, 16(8):533–539, 1998.

[83] Christopher M Cyr and Benjamin B Kimia. 3D Object Recognition Using Shape Similiarity-Based Aspect Graph. *Proceedings of the International Conference on Computer Vision*, pages 254–261, 2001.

[84] W. Eric L Grimson and Daniel P. Huttenlocher. On the Verification of Hypothesized Matches in Model-Based Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(12):1201–1213, 1991.

[85] J. S. Ardaens, S. D'Amico, and O. Montenbruck. Final commissioning of the prisma gps navigation system. *Journal of Aerospace Engineering, Sciences and Applications*, 4(3):104–118, 2012.

[86] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 3951 LNCS:430–443, 2006.

[87] Stefan Leutenegger, Margarita Chli, and Roland Y. Siegwart. BRISK: Binary Robust invariant scalable keypoints. *Proceedings of the IEEE International Conference on Computer Vision*, pages 2548–2555, 2011.

[88] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. *Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*, I:886–893, 2005.

[89] Xiao Shan Gao, Xiao Rong Hou, Jianliang Tang, and Hang Fei Cheng. Complete solution classification for the perspective-three-point problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(8):930–943, 2003.

[90] Pierre Sermanet, David Eigen, Xiang Zhang, Michael Mathieu, Rob Fergus, and Yann LeCun. OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks. 2013.

[91] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[92] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning Deep Features for Scene Recognition using Places Database. In *Advances in Neural Information Processing Systems*, pages 487–495, 2014.

[93] M. Bowick, A. Cacciuto, D. R. Nelson, and A. Travesset. Crystalline Order on a Sphere and the Generalized Thomson Problem. *Physical Review Letters*, 89(24):249902, 2002.

[94] Simone D'Amico, Per Bodin, Michel Delpech, and Ron Noteborn. PRISMA. In Marco D'Errico, editor, *Distributed Space Missions for Earth System Monitoring*, pages 599–637. 2013.

[95] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. pages 1–14, 2014.

[96] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alex Alemi. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. 2016.

[97] Alfredo Canziani, Adam Paszke, and Eugenio Culurciello. An Analysis of Deep Neural Network Models for Practical Applications. pages 1–7, 2016.

[98] Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. pages 1–18, 2012.

[99] Laurens van der Maaten. t-SNE. `https://lvdmaaten.github.io/tsne/`, 2017. Accessed Januray 4, 2017.

[100] European Space Agency. Kelvins - ESA's Advanced Concepts Competition Website. `https://kelvins.esa.int`. Accessed Januray 4, 2019.

[101] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *Advances In Neural Information Processing Systems*, pages 91–99, 2015.

[102] Manisha Kaushal, Baljit S. Khehra, and Akashdeep Sharma. Soft Computing based object detection and tracking approaches: State-of-the-Art survey. *Applied Soft Computing Journal*, 70:423–464, 2018.

[103] Kota Hara, Raviteja Vemulapalli, and Rama Chellappa. Designing Deep Convolutional Neural Networks for Continuous Object Orientation Estimation. *ArXiv:1702.01499*, 2017.

[104] Jimmy Wu. *Robotic Object Pose Estimation with Deep Neural Networks*. PhD thesis, Massachusetts Institute of Technology, 2018.

[105] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. DeepIM: Deep Iterative Matching for 6D Pose Estimation. *ArXiv:1804.00175*, 2018.

[106] Ross Girshick. Fast R-CNN. *ArXiv:1504.08083*, apr 2015.

[107] Matthew D Zeiler and Rob Fergus. Visualizing and Understanding Convolutional Networks. In *European Conference On Computer Vision*, pages 818–833, 2014.

[108] Ken Shoemake. Uniform Random Rotations. In *Graphics Gems III (IBM Version)*, pages 124–132. Elsevier, 1992.

[109] F. Landis Markley, Yang Cheng, John Lucas Crassidis, and Yaakov Oshman. Averaging Quaternions. *Journal of Guidance, Control, and Dynamics*, 30(4):1193–1197, jul 2007.

[110] Koji Yamanaka, Kiyomi Yokota, Katsuhiko Yamada, Hiroshi Koyama, Katsumi Tsukahara, Shoji Yoshikawa, and Taichi Nakamura. Guidance and navigation system design of R-bar approach for rendezvous and docking. 2013.

[111] Alex Kendall and Roberto Cipolla. Modelling Uncertainty in Deep Learning for Camera Relocalization. *arXiv preprint arXiv:1509.059909v2*, 2016.

[112] Kotaro Bessho, Kenji Date, Masahiro Hayashi, Akio Ikeda, Takahito Imai, Hidekazu Inoue, Yukihiro Kumagai, Takuya Miyakawa, Hidehiko Murata, Tomoo Ohno, Arata Okuyama, Ryo Oyama, Yukio Sasaki, Yoshio Shimazu, Kazuki Shimoji, Yasuhiko Sumida, Masuo Suzuki, Hidetaka Taniguchi, Hiroaki Tsuchiyama, Daisaku Uesawa, Hironobu Yokota, and Ryo Yoshida. An Introduction to Himawari-8/9- Japan's New-Generation Geostationary Meteorological Satellites. *Journal of the Meteorological Society of Japan*, 94(2):151–183, 2016.

[113] Vicon Motion Systems Ltd. Vicon Vero by VICON. `https://www.vicon.com/products/camera-systems/vero`. Accessed May 24, 2019.

[114] John Blitzer, Mark Dredze, and Fernando Pereira. Biographies, Bollywood, Boom-boxes and Blenders: Domain Adaptation for Sentiment Classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 440–447, Prague, Czech Republic, 2007. Association for Computational Linguistics.

[115] Hal Daumé. Frustratingly Easy Domain Adaptation. *ArXiv:0907.1815v1*, 2009.

[116] Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, Jonas Schneider, Szymon Sidor, Josh Tobin, Peter Welinder, Lilian Weng, and Wojciech Zaremba. Learning Dexterous In-Hand Manipulation. *ArXiv:1808.00177v2*, pages 1–27, 2018.

[117] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. *IEEE International Conference on Intelligent Robots and Systems*, pages 23–30, 2017.