

2013

Machine-Vision-Based Pose Estimation System Using Sensor Fusion for Autonomous Satellite Grappling

Andres Felipe Velasquez Escandon
West Virginia University

Follow this and additional works at: <https://researchrepository.wvu.edu/etd>

Recommended Citation

Velasquez Escandon, Andres Felipe, "Machine-Vision-Based Pose Estimation System Using Sensor Fusion for Autonomous Satellite Grappling" (2013). *Graduate Theses, Dissertations, and Problem Reports*. 577. <https://researchrepository.wvu.edu/etd/577>

This Dissertation is protected by copyright and/or related rights. It has been brought to you by the The Research Repository @ WVU with permission from the rights-holder(s). You are free to use this Dissertation in any way that is permitted by the copyright and related rights legislation that applies to your use. For other uses you must obtain permission from the rights-holder(s) directly, unless additional rights are indicated by a Creative Commons license in the record and/ or on the work itself. This Dissertation has been accepted for inclusion in WVU Graduate Theses, Dissertations, and Problem Reports collection by an authorized administrator of The Research Repository @ WVU. For more information, please contact researchrepository@mail.wvu.edu.

**Machine-Vision-Based Pose Estimation System Using Sensor Fusion for Autonomous
Satellite Grappling**

Andrés Felipe Velásquez Escandón

**Dissertation submitted
to the Benjamin M. Statler College of Engineering and Mineral Resources
at West Virginia University**

in partial fulfillment of the requirements for the degree of

**Doctor of Philosophy in
Aerospace Engineering**

**Marcello R. Napolitano, Ph.D., Chair
Giacomo Marani, Ph.D.
Jacky Prucz, Ph.D.
Powsiri Klinkhachorn, Ph.D.
Thomas Evans, Ph.D.
Xin Li, Ph.D.**

Department of Mechanical and Aerospace Engineering

**Morgantown, West Virginia
2013**

**Keywords: Pose Estimation; Machine Vision; Sensor Fusion;
Satellite Servicing; Satellite Capture
Copyright 2013 Andres Felipe Velasquez**

ABSTRACT

Machine-Vision-Based Pose Estimation System Using Sensor Fusion for Autonomous Satellite Grappling

Andrés Felipe Velásquez Escandón

When capturing a non-cooperative satellite during an on-orbit satellite servicing mission, the position and orientation (pose) of the satellite with respect to the servicing vessel is required in order to guide the robotic arm of the vessel towards the satellite. The main objective of this research is the development of a machine vision-based pose estimation system for capturing a non-cooperative satellite. The proposed system finds the satellite pose using three types of natural geometric features: circles, lines and points, and it merges data from two monocular cameras and three different algorithms (one for each type of geometric feature) to increase the robustness of the pose estimation. It is assumed that the satellite has an interface ring (which is used to attach a satellite to the launch vehicle) and that the cameras are mounted on the robot end effector which contains the capture tool to grapple the satellite. The three algorithms are based on a feature extraction and detection scheme to provide the detected geometric features on the camera images that belong to the satellite, which its geometry is assumed to be known. Since the projection of a circle on the image plane is an ellipse, an ellipse detection system is used to find the 3D-coordinates of the center of the interface ring and its normal vector using its corresponding detected ellipse on the image plane. The sensor and data fusion is performed in two steps. In the first step, a pose solver system finds pose using the conjugate gradient method to optimize a cost function and to reduce the re-projection error of the detected features, which reduces the pose estimation error. In the second step, an extended Kalman filter merges data from the pose solver and the ellipse detection system, and gives the final estimated pose. The inputs of the pose estimation system are the camera images and the outputs are the position and orientation of the satellite with respect to the end-effector where the cameras are mounted. Virtual and real simulations using a full-scale realistic satellite-mockup and a 7DOF robotic manipulator were performed to evaluate the system performance. Two different lighting conditions and three scenarios each with a different set of features were used. Tracking of the satellite was performed successfully. The total translation error is between 25 mm and 50 mm and the total rotation error is between 2 deg and 3 deg when the target is at 0.7 m from the end effector.

ACKNOWLEDGEMENTS

I would like to thank my research advisor Dr. Marcello Napolitano for his support and guidance and the opportunity to work in this project.

I would also like to thank the staff of the West Virginia Robotic Technology Center: Dr. Thomas Evans for his support in the experimental tests; Patrick Lewis for all the help and support in the experimental tests and in the development and application on the algorithms and Dr. Giacomo Marani for his help, bright ideas, and for providing so many helpful algorithms.

I would like to thank my committee members Dr. Jacky Prucz, Dr. Powsiri Klinkhachorn and Dr. Xin Li for their support and useful feedback.

I would like to thank Dr. Eugene Cilento and all the professors and students associated with the West Virginia Robotic Technology Center for their support and feedback provided during the development of this project.

Finally, I would like to thank my mother, family and friends for their support during all these years.

Table of Contents

1 INTRODUCTION	1
2 ON ORBIT SATELLITE SERVICING	6
2.1 Satellite Capture Missions	6
2.2 Grasping Point.....	7
2.3 Vision Sensors.....	7
2.4 Markers and Features on the Satellite	7
2.5 Simulation of the Satellite Motion	8
2.6 Simulation of the Lightning Conditions.....	8
3 MACHINE VISION-BASED POSE ESTIMATION SYSTEMS	9
3.1 Classification of Machine Vision based pose estimation systems.	9
3.2 CAD Model Based Methods	11
3.3 Pose Estimation System based on Points	12
3.4 Pose Estimation System based on Lines	13
3.5 Pose Estimation System based on Circles.....	13
3.6 Pose Estimation System based on Several Features	13
3.7 Features Extraction Methods.....	14
3.8 Feature Matching Methods	14
3.9 Data Fusion	15
3.10 Motion Model	15
4 RELATED WORK AND CONTRIBUTIONS	16
4.1 Related Work	16
4.2 Contributions.....	17
5 GENERAL DESCRIPTION OF THE PROPOSED POSE ESTIMATION SYSTEM.....	18

5.1 System Components.....	18
5.2 Relative Pose.....	20
5.3 Algorithm General Description.....	21
5.3.1 Image Acquisition.....	23
5.3.2 Feature Extraction.....	23
5.3.3 Detection.....	24
5.3.4 Circle Pose Calculation and EDS.....	26
5.3.5 Pose Solver.....	26
5.3.6 Extended Kalman Filter.....	28
6 FEATURE EXTRACTION SYSTEM.....	29
6.1 Ellipse Extraction.....	29
6.2 Line Extraction.....	32
6.3 Point Extraction.....	34
7 FEATURE DETECTION SYSTEM.....	36
7.1 Ellipse-Curve and Ellipse-Point detection.....	36
7.1.1 Ellipse-Curve Detection.....	36
7.1.2 Ellipse-Points Detection.....	37
7.2 Line Detector.....	40
7.3 Point Detector.....	43
8 CIRCLE POSE CALCULATOR.....	46
8.1 Coefficients of the Ellipse Equation in Terms of the Geometric Parameters.....	47
8.2 Center and Normal of the Circle.....	48
9 POSE SOLVER.....	54
9.1 Ellipse Cost Function.....	56
9.2 Line Cost Function.....	56

9.3 Point Cost Function.....	57
9.4 Total Cost Function.....	57
9.5 Conjugate Gradients Method	58
9.6 Pose Solver Example.....	60
10 EXTENDED KALMAN FILTER	65
10.1 Motion Model and State Vector.....	65
10.2 Observation Process	69
10.2.1 Circle Center	69
10.2.2 Circle Normal.....	70
10.2.3 Pose Vector	70
10.2.4 Output Vector.....	71
10.3 Extended Kalman Filter Equations	71
11 METHODOLOGY FOR THE EVALUATION OF THE PROPOSED SYSTEM.....	74
11.1 Testing Scenarios	75
11.2 Virtual Testing	78
11.2.1 Virtual Simulator.....	78
11.2.2 Software Configuration.....	82
11.2.3 Testing Procedure.....	83
11.3 Experimental Testing	87
11.3.1 Experimental Setup.....	88
11.3.2 Hardware and Software Configuration	92
11.3.3 Testing Procedure.....	93
12 TESTING RESULTS.....	97
12.1 Virtual Testing	97
12.1.1 Effects of Camera Parameters in the Pose Estimation Error.....	97

12.1.2 Effect of Relative Pose in the Pose Estimation Error	102
12.1.3 Dynamic 6DOF Tests.....	103
12.1.4 Approach Tests.....	106
12.2 Experimental Testing	112
12.2.1 Tests for Static Pose Error.....	112
12.2.2 Effect of Initial Conditions.....	116
12.2.3 Dynamic 6DOF Tests.....	117
12.2.4 Approach Tests.....	130
12.2.5 Processing Time	135
13 DISCUSSION	136
13.1 Virtual Tests.....	136
13.1.1 Effects of Camera Parameters in the Pose Estimation Error.....	136
13.1.2 Effects of Relative Pose in the Pose Estimation Error	136
13.1.3 Dynamic 6DOF Tests.....	136
13.1.4 Approach Tests.....	137
13.2 Experimental Testing	138
13.2.1 Tests for Static Pose Error.....	138
13.2.2 Effect of Initial Conditions.....	139
13.2.3 Dynamic 6DOF Tests.....	139
13.2.4 Approach Tests.....	139
13.2.5 Processing Time	139
13.3 General Discussion.....	140
13.3.1 Effects of the light conditions	140
13.3.2 Effects of the scenarios	140
13.3.3 Virtual images vs. Experimental images.....	142

14 CONCLUSIONS.....	144
15 REFERENCES.....	145

Nomenclature

deg: Degrees

EKF: Extended Kalman filter

FEE: Fast ellipse extractor

GSFC: Goddard Space Flight Center

MLI: Multi-layer insulation

ROI: Region of interest

Rot.: Rotation

Trans: translation

w.r.t.: With respect to

WVRTC: West Virginia Robotic Technology Center

1 INTRODUCTION

The interest in the development of technical capabilities for on-orbit satellite servicing has been increasing recently due to its potential of increasing the life cycle of a satellite considerably at lower cost than launching a new satellite [1]. The life of a satellite can prematurely end due to mechanical failures and on-orbit anomalies, such as dead batteries, faulty sensors, undeployed solar panels, etc. Additionally, several fully functional satellites are retired due to exhausted fuel resources.

There are three main technical difficulties that need to be considered for an on-orbit satellite servicing mission and that define the main components of the mission. They are described in the following paragraphs.

Unmanned Mission. For economical reasons the mission has to be unmanned and robots have to perform the different activities of the servicing process: approach, capture, accessing the fuel tank valve, refueling and sealing the valve, etc..

Non-cooperative Satellite. Satellites that were not designed with servicing capabilities are known as non-cooperative satellites, and current satellite belong to this category. They do not have grasping or docking ports, sensors or visual clues to assist the approach and grasping. Additionally it is not easy to access, unseal and reseal the fuel valve and battery compartments.

Autonomous mission. Due to the time delay in communications between Low Earth Orbit (LEO) and ground a human will not be able to operate a robotic arm accurately and capture the satellite. The robots are required to capture the target satellite autonomously to avoid collisions or chasing the satellite without grasping it.

There have been space missions developed to demonstrate the technology required for on-orbit servicing, JAXA's ETS VII, in 1999, performed the first autonomous capture of a satellite using a robotic arm [2], [3]. In 2007 DARPA's OrbitalExpress used a cooperative satellite to test the approach, capture, docking and servicing operations [4]. In both mission the target satellite was developed specifically for the mission. Although cooperative satellites were used in these missions, the results provided insights and useful details and requirements towards the development of an actual servicing mission. Current efforts related to the development of the on-orbit servicing include the NASA's Robotic Refueling Mission (RRM) [1], which has performed several servicing activities including refueling on the International Space Station. Current developments towards the servicing of non-cooperative satellites include the DLR's DEOS [5] and MDA's

Space Infrastructure Servicing (SIS). In addition to space missions, there are several ground testing facilities created to develop the satellite servicing concept [6], [5], they use robotic arms to simulate the motion of the servicer and the target satellite, and simulate the light conditions on space, which allows them to evaluate the performance of the pose estimation, navigation and control systems along with the tools involved in the capture and servicing process.

During a typical conceptual servicing mission [6] the servicing vessel approaches the target satellite using the vessel propulsion system. The vessel is equipped with several robotics arms which are used to capture and hold the satellite and perform the different servicing tasks. When the satellite is within the working range of the robotic arm (a few meters from the vessel), the close proximity operations begin and the robotic arm is deployed to capture and secure the satellite (see Figure 1). After this step the servicing operations can be performed, which may include refueling, battery replacement, etc.

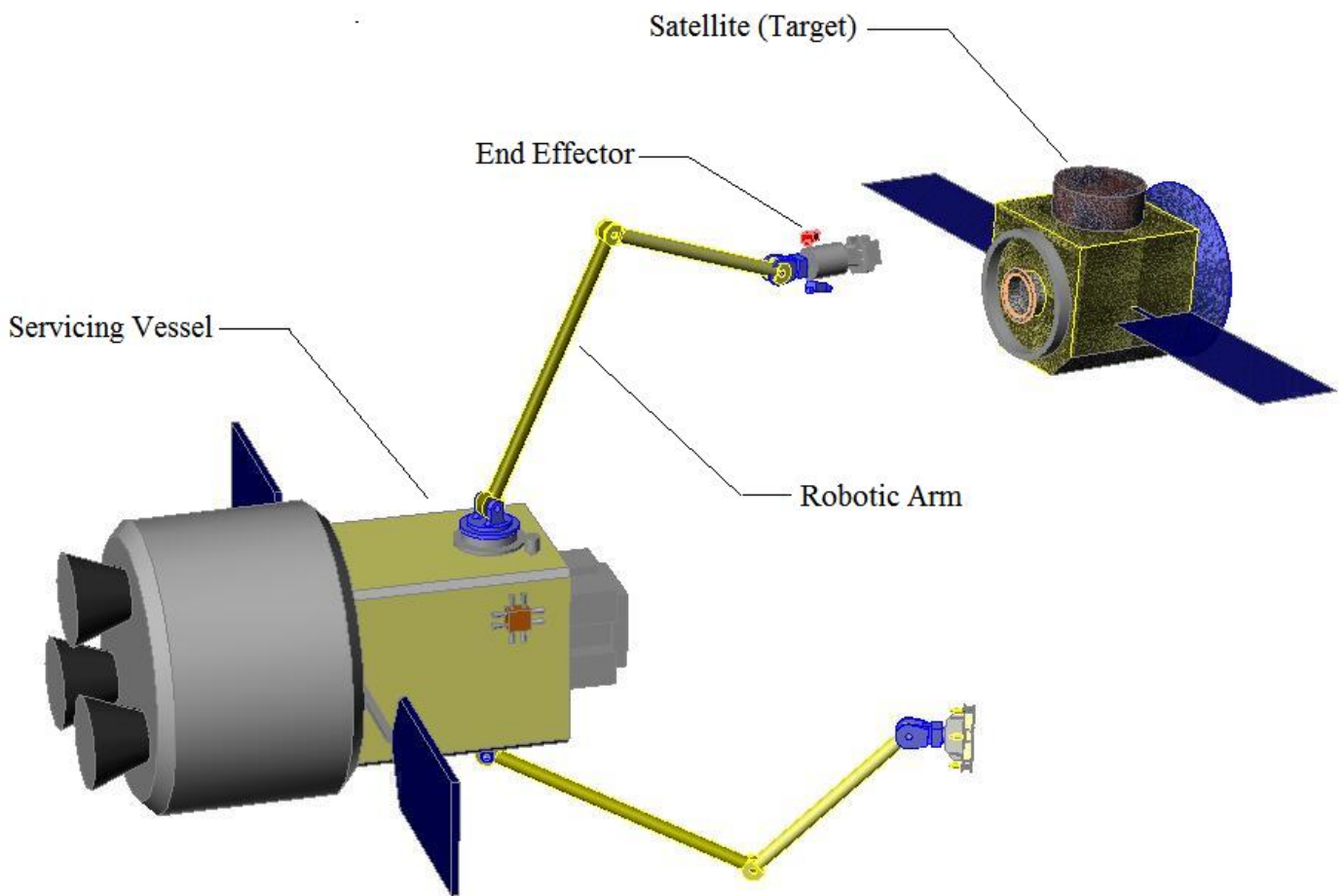


Figure 1. Capture phase of an on-orbit servicing mission

The current robotic manipulators used in the industry for assembly, welding and painting are capable of perform the majority of activities associated with the servicing mission. Work on the use of non-cooperative targets has already been addressed by several authors [6]. Similarly, work on autonomous robotic operations has been performed in the field of underwater robotics for example.

Although the previous technical issues have been individually addressed still there is work to do when considering the entire on-orbit servicing mission and when adding the restrictions of the space environment. One important system that is required for a successful mission, specifically during the capture phase is the pose estimation system; it gives the position and orientation of the satellite to the robot controller, so it can compute the trajectory to grasp the satellite. The lack of docking aids of a non-cooperative satellite makes difficult the pose estimation problem. Machine vision based pose estimation system has been proposed to perform this task [2], [7], [8], [9], [3].

To determine the pose of a non-cooperative satellite using machine vision systems, the pose estimation system needs to be able to use satellite natural geometric features as clues/markers for the pose estimation.

Although artificial satellites have common geometric features (thrusters, interface ring, straight edges, etc.) they are not standardized and were not designed for servicing activities. Additionally, satellites are usually covered with MLI (Multi-Layer Insulation) which is a highly reflective material with textures (see Figure 2), that complicates the recognition of the features by the machine vision system.

The interface ring has been proposed as a possible location for grasping the satellite [1], [3] since it is easily accessible and has the appropriate structural properties. The function of an interface ring is to attach the satellite to the launching vehicle, and they are a fairly common feature in several satellites.

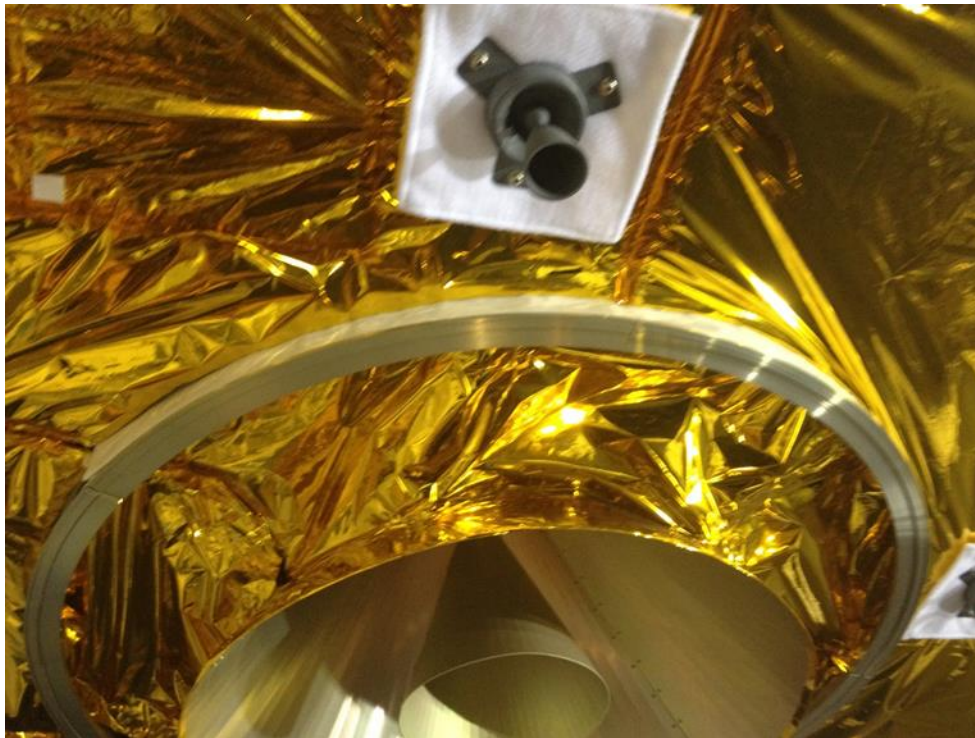


Figure 2. Satellite Multi-Layer Insulation (MLI)

The current research focuses on the development and evaluation of a machine vision based pose estimation system. The proposed system estimates the pose using three particular geometric features: corners/points, rectilinear edges and circles; and using data from two monocular cameras which are mounted on the robot end-effector. It is assumed that the satellite has an interface ring, and it is the target of one of the cameras. The system is intended for autonomous on-orbit satellite servicing missions where it will provide the position and orientation of a non-cooperative satellite with respect to the end-effector of a robotic manipulator during the capture phase of the mission, i.e. in the close proximity operations. The capture tool is mounted on the end effector.

Data from the two cameras is processed by three different algorithms, each one using a particular type of geometric features (circles, lines, points). The three algorithms (known as ellipse, line and point detector) are based on a feature extraction and detection scheme, to provide the geometric features on the camera images that correspond to the features of interest in the satellite. These features are known as the detected features and they are the output of each one of the three algorithms. It is assumed that the geometry of the features of interest in the satellite is known.

The outcome of each algorithm is merged using a system called the pose solver system which finds pose using the conjugate gradient method to optimize a cost function and to reduce the re-projection error of the detected features. Additionally, since the projection of a circle on the image plane is an ellipse, a system

known as EDS for ellipse detection system is used to find the 3D-coordinates of the center of the interface ring and its normal vector using its corresponding detected ellipse on the image plane of one of the cameras. The pose obtained by the pose solver system and by EDS is given to an extended Kalman filter to obtain the final estimated pose. The pose is given in terms of a homogeneous transformation matrix.

The evaluation of the pose estimation system was performed using virtual simulations and experimental testing. The virtual simulator was implemented in C++ and uses a CAD model of an actual weather satellite, in this case the pose estimation system uses the synthetic images generated by the virtual simulator. The virtual simulator was used to generate different types of motions between the target and the satellite, to test the system under controlled conditions, and to evaluate the effect of external disturbances such as lens distortion, image noise, camera calibration errors, etc.

The experimental testing was performed at the West Virginia Robotic Technology Center (WVRTC), using a full scale satellite mockup of the same weather satellite used in the virtual simulator, the mockup has highly realistic details including the interface ring and the MLI wrap. The cameras are mounted on the end-effector of a 7DOF robotic manipulator, which was used to generate the relative motion between the satellite and the cameras. The estimated pose is compared against the ground truth, obtained using a laser tracker system, which has a submillimeter accuracy. Testing was conducted using three different scenarios, each one with a different set of geometric features, and using two different light conditions. Different types of maneuvers were performed to simulate the relative motion between the satellite and the end-effector, these include translations, rotations and approaches. The relative distance target-satellite was maintained between 1.5m and 0.7m.

2 ON ORBIT SATELLITE SERVICING

This section describes common aspects of missions and on-ground testing related to autonomous satellite capture using machine vision based pose estimation systems.

2.1 Satellite Capture Missions

Two space missions that successfully performed autonomous satellite capture are the Engineering Test Satellite No. 7 or ETS-VII from JAXA in 1997 and the Orbital Express from NASA in 2007. Both missions performed tests on satellite capture.

The ETS-VII mission used two satellites (chaser and target). The chaser satellite had a 6-DOF robotic arm with a hand-eye camera. The target satellite had visual markers and a handle to facilitate the grasping operation of the robotic arm.

The Orbital Express mission had a similar configuration: a target satellite called NEXTSat and chaser satellite called ASTRO with a 6-DOF robotic arm. The NEXTSat was captured using two techniques: direct capture, where a capture mechanism (a gripper directly attached to the satellite body) is used to mate the two satellites and free-flying capture where a robotic arm is used to capture the satellite. There are three main phases in a satellite capture mission [6] :

Long-range rendezvous. The chaser satellite approaches the target satellite so the distance between them varies from a few kilometers to a few hundred meters. The chaser satellite is guided by navigational systems (GPS, radar, star tracker, etc.). The vision system is not used in this phase

Short-Range Rendezvous. The distance between satellites changes from a few hundred meters to a few meters. The satellite has to be guided by on-board sensors that directly measure the relative distance and attitude of the target (radar, lidar, and machine-vision).

Capture. The distance between satellites is of the order of meters, so the robotic arm of the chaser satellite can approach the target satellite and grapple it at a predefined point. Guidance is based on machine vision systems and the relative attitude between satellites is such that the grapping point is in the working zone of the robotic arm and the features used by the vision system are in the field of view (FOV) of the sensors, manipulation of the robotic arm may be required in order to have the features in the sensor FOV. Fly-around inspections can be performed before attempting capture.

This chapter focuses on the machine vision systems used for pose estimation during the capture phase. During this phase the vision system has to provide the relative pose of the two satellites with a predefined accuracy and sampling rate.

Besides actual space mission several ground test have been performed by several researchers to evaluate vision systems (as early as 1989, [10]). Correct simulation of the on-orbit conditions is required to perform this task. Some important aspects related to simulation and testing of vision based pose estimation systems for satellite capture are considered below.

2.2 Grasping Point

The spot to grasp and hold the target satellite must have enough structural strength to support the forces applied during contact with the robotic arm, and during the servicing operations. On the ETS_VII a handle was used. The NEXTSat from the Orbital express mission had a grapple fixture. In both cases cooperative satellites were used. On non-cooperative satellites two spots may satisfy the above requirement: the interface ring and the apogee thruster [11], [12]. In this work the interface ring is used.

2.3 Vision Sensors

The sensors can be catalogued as 2D or 3D sensor. Monocular cameras (vision or infrared cameras) can be considered as 2D sensor. Stereo cameras, LIDAR and LADAR systems are 3D sensors.

Monocular cameras are frequently used to extract features on the target surface. 3D sensors find 3D point-clouds belonging to the target surface.

Sensors can be mounted on the robotic arm end-effector. Reference [13] incorporates an additional supervising camera that is mounted in manipulator base. This camera has the function of monitoring the lightning conditions and the motion of the robotic arm.

Several types of sensors can be used simultaneously with a data fusion system that gives a better estimation of the pose.

2.4 Markers and Features on the Satellite

When a satellite has feature specifically design for machine vision system, such as reflectors, lights or fiducial markers, it is referred as a cooperative target. A non-cooperative satellite does not have these devices and a machine vision system has to use the target natural features in order to determine its pose. If a hand eye camera is used the features should be close to the grasping point, to avoid them to be out of sensor FOV as the end-effector approaches the grasping point, an alternative is to use an additional camera mounted in different location. Reference [3] uses the interface ring of the target satellite as a feature to track.

2.5 Simulation of the Satellite Motion

Due to several external disturbances (gravitation torque, solar wind, Earth magnetic field) a satellite with a non-operating attitude control system will have random motions (rotations, wobbling). It is important to characterize and simulate this motion in order to test a vision system, and evaluate if the sampling rate and accuracy of the vision system are appropriated.

Reference [6] uses a robotic arm to simulate the target satellite motion, it consists of a rotation imposed to an elliptic translation. A similar configuration is used by [14] to test an algorithm to estimate the dynamics of a satellite mockup based on a vision system.

2.6 Simulation of the Lightning Conditions

There are three main light sources on orbit that can affect the performance of the vision system: the sun, the Earth albedo (diffuse light) and the reflection of the satellite components. The intensity of these sources varies with time, and the frequency of this variation depends on the specific orbit of the satellite.

Proper simulation of these conditions is required in order to correctly test the vision system, especially when using feature extraction based techniques. Since satellites are covered with MLI (Multi-Layer Insulation) which is a highly specular reflective material, the satellite mockup used for simulation should be covered with MLI, and include similar external features (form and material) such as thrusters, antennas, etc.

The testing setup used in Reference [15] uses an Earth albedo reflector and a moving light source to simulate the sunlight and its direction changes. Reference [3] reports variation of the illumination factor on-orbit from zero to several hundreds of thousands lux.

3 MACHINE VISION-BASED POSE ESTIMATION SYSTEMS

This section shows the classification, trends, description and main characteristics of existing algorithms for machine vision based pose estimation systems.

3.1 Classification of Machine Vision based pose estimation systems.

Pose estimation is not mandatory to control the position of a robotic arm with respect to a given target. Image based visual servoing (IMVS) [16] can also be used to accomplish this task. In IMVS the robot is controlled according to how an object or a feature appears in the camera image, with no pose being computed. The robot controller is in closed loop with the vision system using a signal that is related to the difference between what should be observed vs. what is being observed, a control law tries to minimize that difference. On the contrary in object based visual servoing (OBVS) the robot controller requires the pose of the target to be able to approach it.

Machine vision based pose estimation systems can be classified according to how the pose is computed, and two categories are established: feature based systems and CAD model based systems. In feature based systems pose is estimated using correspondences between image features and object features. In CAD model based systems pose is determined by minimizing error between an image and the projection of a CAD model of the object in the image plane. Figure 3 and Figure 4 show the basic general scheme of these approaches, which are described below.

The CAD model based method generates a projection of the model according to an assumed pose creating a virtual image, and then it extracts the features (edges, points, lines, ellipses, etc.) of that virtual image. Additionally features from the real camera image are also extracted. Features of both real and virtual images are compared and matched. If the matching error is larger than a threshold the assumed pose is modified and the process repeated, if not the assumed pose is declared as the estimated pose. The matching error is usually computed as the sum of the distances between the respective virtual and real features.

The feature based approach extract features from the camera image, and then it relates those features with 3D features that are defined in terms of their internal geometry, as an example: the coordinates of a set of points, the diameter and center location of a circle. Once the features have been matched the pose is calculated using analytic, probabilistic or optimization methods.

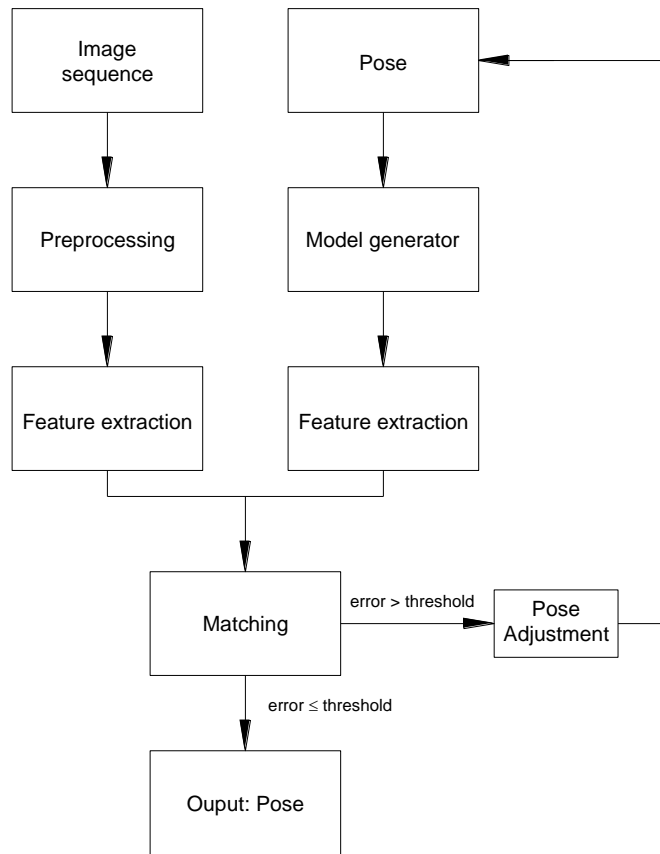


Figure 3. Typical Model-based approach (from [17])

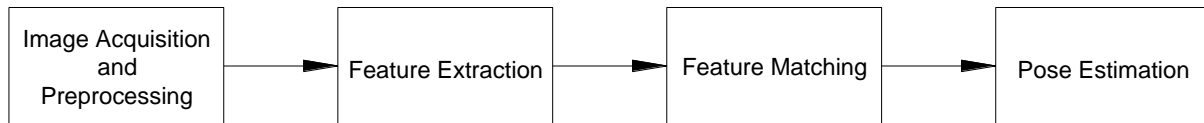


Figure 4. Typical feature-based approach

Table 1 classifies pose estimation systems of several references that are considered relevant for this research. It can be seen that systems based on point features are the widely used. Table 2, from Reference [17], shows a comparison of the two approaches according to five criteria, it can be concluded from that table that the CAD model based approach is more robust but less accurate, and requires more information about the target and computation power than the feature based approach.

Table 1. Machine vision based pose estimation systems classification

CAD Model Based	Edges/Pixels/Gradient (images)	[18], [19], [20], [21], [22]*	
	Feature Extraction	Points	[23], [24], [25]*
		Lines	[7]*
		Multiple Features	[26], [27], [28]
Feature Based	Points	[29], [30], [31], [32], [33], [34], [35], [36], [37], [38], [39], [40], [41], [42], [43], [44], [45], [46], [47], [48], [49], [50], [51], [52], [53], [54], [55], [56]*, [8]*, [57]*, [58], [59]	
	Lines	[60], [61], [62], [63], [64], [65], [66], [67], [9]*	
	Circles	[68], [69], [70], [71], [72], [73], [74]	
	Multiple Features	[75], [76], [77], [78], [79]	

*satellite capture application

Table 2. Comparison of feature based and CAD model based approaches. (From [17])

Comparison Items	Pose Estimation Approaches	
	<i>Feature based</i>	<i>Model based</i>
Accuracy	average	worst
Complexity	least	most
Request for input	average	lowest
Anti-noise	average	best
Cost time	least	most

According to Reference [17] there are three trends in the development of new pose estimation systems: the use of higher quality images, fusion of different approaches (CAD model base and Feature based) and integration of the several process in the algorithm, since some of them use the same information, for example matching, feature extraction and pose calculation all of them required the knowledge of the target geometry and are related to the previous time pose.

3.2 CAD Model Based Methods

This section is an overview of the CAD model based systems; it shows the main aspect of the references presented in Table 1.

Reference [18] uses edges and the shading of the image to match the CAD model with the image and a non-linear least squares method and data fusion to estimate pose. Reference [19] work on a pixel level, using pixel attributes to match the image and the CAD model, attribute can be colors, or local texture features. It defines and minimizes an illumination-invariant distance between the image and the projected 3D model to find pose. Reference [20] uses edges to match the model and Lie group algebra to describe the edge motion in geometric terms, and uses a least square method to estimate the target motion. Reference [22] also uses

edges and is based on the Lie group technique. Reference [21] uses edges and neural networks to estimate the pose, which requires the use of training images. Reference [23] uses an off-line learning process to define feature point that will be used to solve the matching problem during the run time. Reference [24] uses points and genetic algorithms to solve the matching/pose problem. Reference [25] uses feature points with high intensity gradients and redefines the point matching problem as multi-class classification problem; it matches the extracted points during run-time with points extracted during a training phase; the training phase is performed using random views of the CAD model. Reference [7] uses a CAD model of the target made of lines, extracts image edges and match them with the projection of the CAD model using the previous time pose, then it uses the Moving Edge Algorithm which creates sample points in the projected lines and after that it searches at each sample point along the direction orthogonal to the projected line until it finds a correspondent point on the image; the new pose is defined as the one that minimizes the distance between the correspondent point and the projected lines. Reference [26] uses lines and points and a non-linear optimization method to find pose, it uses RANSAC for 2D-3D point correspondence and the Moving Edge algorithm for line matching, additionally it uses an M-estimator to improve accuracy and robustness in the pose. The method of Reference [27] extracts segments from the image and classifies them as lines or elliptical curves, then merges them into lines and ellipses respectively; it then groups extracted features and relates them to 3D objects (circles and 3D corners), pose is estimated using points and ellipses are just used for evaluating matching errors.

3.3 Pose Estimation System based on Points

Pose estimation systems that use points can be classified in two types: systems that assumed the matching or correspondence problem is solved, [37]- [54], and system that deal with it [58] [59]. The correspondence problem is the correct identification of the object features and their corresponding projected features in the image plane. When the points are correctly matched, several methods can be used to determine pose: analytical, probabilistic [41] [29] (particle filter) and optimization methods.

Reference [38] discusses different types of probabilistic methods. In Reference [54] a least median of squares estimator and an M-estimator are used to handle outliers and increase robustness. Reference [41] also uses least median of squares to estimate pose.

The minimum number of points to determine pose is three, when more points are used the solution can be more robust and estimation methods are used. Reference [50] uses 3 points and Reference [51] uses 4 points to compute pose.

3.4 Pose Estimation System based on Lines

The perspective projection of a 3D line segment in the image plane is a 2D line segment (except when the segment is orthogonal to the image plane). Some systems assumed the matching problem solved and others show methods to correctly identify the 2D-3D pair of lines. Analytical [63] [64] [65], probabilistic [66] (Particle filter based) and optimization methods [62] are used. A minimum number of three a lines are required to determined pose, more lines increase robustness and estimation methods can be used.

3.5 Pose Estimation System based on Circles

The perspective projection of a 3D circular feature is an ellipse (it will be a circle if the image and the plane of the circular feature are parallel). Even in the case that there is just one elliptical feature in the image, when using an ellipse extractor there will be more than one extracted ellipse due to false detections and noise in the extractor. According to this the correspondence problem for circular features also has to be considered.

Once the correct match is found method analytical methods [69] [70] can be used to find the 3D pose (center location and normal vector) of the circular feature, these methods require the knowledge of the ellipse geometric properties (major and minor axes, center and orientation angle). When just one ellipse is used to compute pose there are two possible solutions for this problem and additional information is required, Reference [71] deals with this problem.

3.6 Pose Estimation System based on Several Features

Some systems combine different types of features to improve their robustness and flexibility, Reference [26] uses lines and points with an M-estimator, Reference [27] uses lines and ellipses while Reference [28] can use points, lines and ellipses. Reference [75] uses a probabilistic method with points, lines and linear contours of cylinders. Reference [76] uses lines and points with a probabilistic method to find pose, then it uses the Kanade-Lucas-Tomasi tracker between two image frames to predict boundary locations and uses lines for improving the estimation. Reference [77] uses a general approach to find pose using the minimal number of lines and points required to define pose: three points, two points and one line, one point and two lines and three lines. Reference [78] uses lines and circles on a plane (a Hockey rink) and pose is defined and calculated in terms of the homography. Reference [79] shows how to calculate pose using rectangular, circular and triangular features.

There are some basic functions that are common for several pose estimation systems; these are described in the following sections.

3.7 Features Extraction Methods

Edges, points, lines and ellipses are common image features that are used by several pose estimation systems. Edges can be extracted using Canny [76] [8] and Sobel [21] edge detection.

Several methods to extract points on an image are used: Harris [33] [23] [26] [36] , Susan [36], minimum eigenvalues [80]. Reference [36] compares the performance of the Harris and Susan corner detector.

Line extraction is usually performed using the Hough transform [76] [8].

Since the projection of 3D circular feature in the image plane is an ellipse, ellipse extraction is required. It can be performed using the method described in [81], or by using the binary edge map and performing ellipse fitting, as in done in References [27] [74]. Reference [68] extracts blobs and uses their mean and covariance matrix to check if they are ellipses or not.

3.8 Feature Matching Methods

A pose estimation system usually has two operating modes: initialization and tracking. During initialization the object pose and its projection in the image are completely unknown, heavy computations methods, not practical for real time, are used in order to solve the matching problem and determine pose. Another option is user initialization [28] [36], where the user manually selects the correct corresponding features in the image or defines the initial transformation matrix, sometimes just an approximated value is enough.

When the image acquisition and processing is performed with a satisfactory frame rate (around 5Hz) or when the target moves at a moderate speed with respect to the camera, the motion of the features in the image plane are of the order of a few pixels; this can help to solve the matching problem during the tracking mode, since the features remain close between two consecutive frames. In this mode a different, faster, algorithm is used, or the same used for initialization providing a faster convergence rate. Optical flow [17] is widely used for tracking small motions such as inter-frame motions.

The pose estimation problem and matching problem are closely related each other, usually they need to be solved simultaneously, since the solution of one requires the solution of the other, and sometimes it is difficult to differentiate if an algorithm is used for matching or for pose estimation.

RANSAC (Random Sample Consensus) [82] [9] [26] [76] is a popular method used to solve the matching problem; it is a probabilistic method based on random hypothesis, sometimes taken as the standard robust

estimator in computer vision. Genetic algorithms have also been used [24] [46], Reference [46] compares its results to RANSAC. Reference [59] uses nonlinear mean shift clustering and also compares its result to RANSAC. Reference [56] uses SoftPOSIT which is an optimization method based on an iterative pose and iterative correspondence assignment approach.

In the case of small number of circular features the matching problem can be simplified by just checking the size of the ellipse and its location, provided low noise in the ellipse extraction process.

3.9 Data Fusion

Data fusion can be used to integrate information from several cameras [55] [8], several features (see Table 1 for references) or several algorithms [83]. Reference [8] uses two cameras to see different parts of the same feature on a satellite; due to the size of the feature one camera is not capable of having the entire feature on view at close distance. Data and sensor fusion is usually performed using a Kalman filtering [29] [36].

One main problem when using several cameras is the calibration. Errors in the extrinsic calibration of the cameras can lead to bias errors in the pose estimation system. References [22] [55] [84] deal with the camera calibration problem.

3.10 Motion Model

Kalman filter approaches usually require a camera-target motion model. A popular model is the constant velocity model [32] [60] [66], which assumes that the motion of the target with respect to the camera occurs with a constant angular and translational velocity. This model is usually described using Euler angles [32] or quaternions [60]. The advantage of using quaternions is that they describe the motion unambiguously. The constant velocity motion model using quaternions will be used in this research.

4 RELATED WORK AND CONTRIBUTIONS

4.1 Related Work

This section show work related to machine vision based pose estimation systems intended to be used for satellite capture. In reference [85] a virtual simulation is used to evaluate a pose estimation system intended for satellite servicing; the system uses a monocular camera and tracks the interface ring and one edge of the satellite. It reports pose estimation errors of 5 cm and 2 deg when the distance target/service vehicle is 5m; tests using real images were not performed. The virtual simulation uses only a basic satellite model (no end-effector model) with no realistic details that make difficult the performance of a vision system, such as occlusion of the interface ring by other components (end-effector, heat shield) or the MLI and its reflections. Additionally the system is not evaluated at close proximities to the target where some of the used features can be totally or partially out of the camera view, as an example the interface ring will appear as an elliptic arc in the camera view.

Reference [7] proposes a model based pose estimation system using a monocular camera. It find pose using the moving edge technique to align the edges of a CAD model with the camera image. The evaluation of the system was performed using synthetic images and real images from a 1/50 scale model of a satellite. The camera is mounted on a 6DOF robotic manipulator to simulate motions, ground truth is obtained using the robot position. Test were performed at distances larger than 124m (scaled up distance) from the satellite. The authors report translation errors of less than 10cm (0.08%) and rotation errors of less than 5deg. This system uses a complete view of the satellite, including its solar panels, which can be easy to detect by a machine vision system, but during close proximity operations these features can be out of the camera view. Additionally the satellite scale model lacks the realistic details mentioned above which affect negatively the performance of the machine vision system.

GNFIR (Goddard Natural Feature Image Recognition) [86] is a machine vision base pose estimation system that have been considered for satellite servicing missions of non-cooperative satellites. It is part of the Argon system which is a set of sensors (two cameras, a flash lidar and a flight computer) designed for rendezvous and docking of spacecraft. Argon was developed in the Satellite Servicing Capabilities Office (SSCO) at the Goddard Space Flight Center (GSFC). GNFIR is a model based pose estimation system that extracts image

features and match them with a 3D model of the target defined as a set edges, it is based on the algorithm described in Reference [20]. GNfir was evaluated in [86] using robotic manipulators to perform approaches to a satellite mockup, the target-camera relative distance during testing was varied between 10m and 2m. The reported translation errors are 29cm in range, 1.5cm in transverse translation, and rotation errors of 5.3deg in roll and 9.1deg in transverse attitude when tracking a non-static target with rotation rates lower than 0.5deg/sec. Tests were performed at two different light conditions. No close proximity tests were reported.

The NRL's FREN (Front End Robotic Enabling Neat-Term Demonstrator) is a DARPA project developed to study the capture of non-cooperative satellites for on-orbit servicing. Testing was performed using a robotic manipulator and a satellite mockup. It uses a Lidar system to obtain 6DOF pose and a machine vision system with three cameras on the robot end effector, which also contains the capture tool. The machine vision system is used to correct the pose error of the lidar system and to guide the robot during final approach. The machine vision only operates when the capture tool is within 20cm to 1.5cm from the target and is designed to track the interface ring or small features such as bolt holes. The accuracy of the system is ± 2 cm noise (2σ) and bias at 20cm, and ± 0.25 cm noise and bias at 1.5cm.

4.2 Contributions

The principal contribution of this research is the development of a machine vision based pose estimation system intended to be used at close proximities of a non-cooperative satellite, where close proximities means relative distances of less than 2m between the target and the end-effector where the cameras are located. The system uses different types of geometric features and data from several cameras. Realistic conditions that appear at close proximities are simulated, such as occlusions, partial views of the target and reflections from the satellite's MLI.

Another important contribution is the study of the effect of the system parameters in the pose error. Particularly parameters such as the lens distortion and camera calibration errors are considered.

5 GENERAL DESCRIPTION OF THE PROPOSED POSE ESTIMATION SYSTEM

This chapter presents the proposed pose estimation system, its main components, the definition of relative pose and the algorithm flowchart.

5.1 System Components

The main components of the pose estimation system are the end-effector, the two cameras and the capture tool as presented in Figure 5. The end-effector and the cameras are represented by their corresponding reference frames.

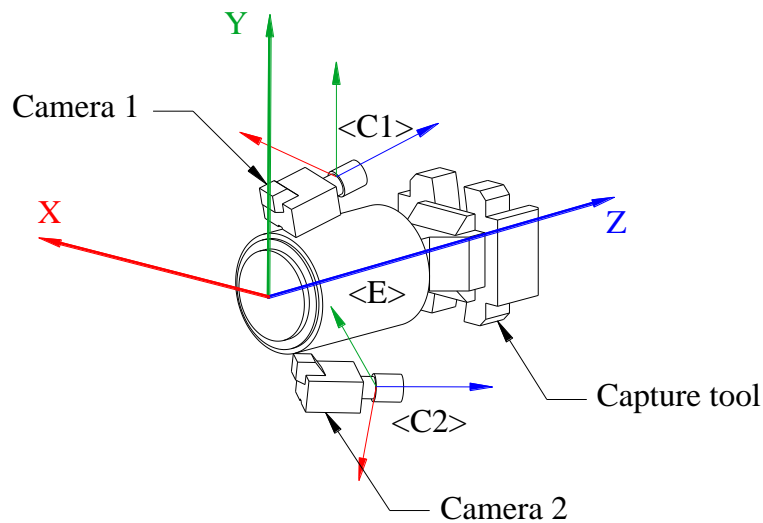


Figure 5. Hardware components of the pose estimation system

The target satellite is also represented by a reference frame as can be seen in Figure 6. A simplified view of the end-effector and the satellite that shows only their corresponding reference frames is presented in Figure 7.

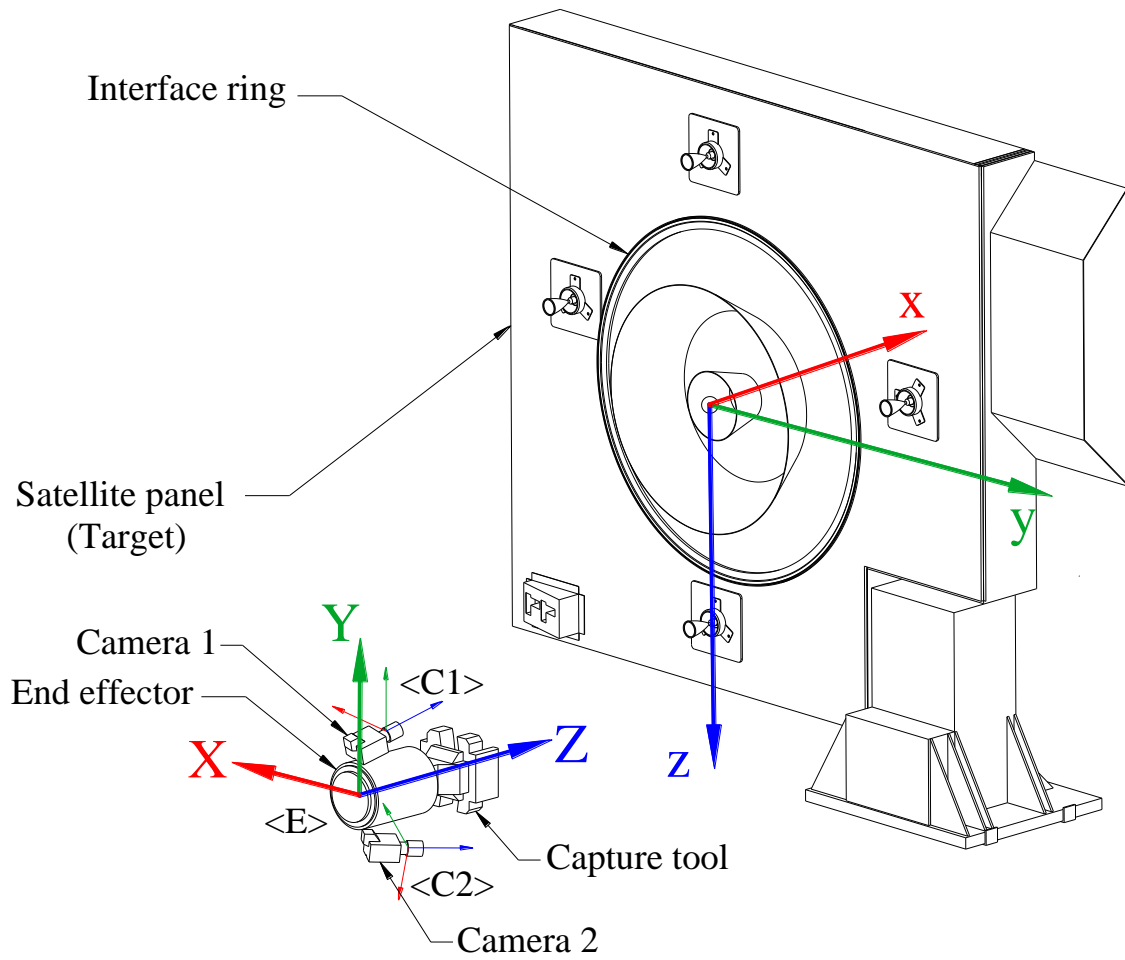


Figure 6. End-effector and satellite.

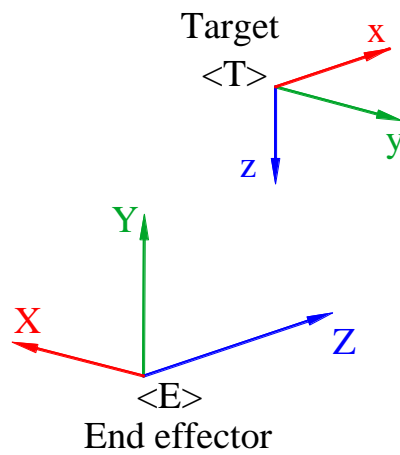


Figure 7. Reference frames of the end-effector and the satellite.

The task of the pose estimation system is to find the pose of the satellite reference frame w.r.t. the end-effector reference frame. In this research it is assumed that the satellite (and the satellite mockup used for the experimental testing) has an interface ring (see Figure 6) and that it is being tracked by camera-1. Camera-2 tracks other types of features.

5.2 Relative Pose

The relative 6DOF pose between the two references frames (end-effector and target satellite) is defined using a translation vector,

$$t_{T/E} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad (1)$$

and a rotation matrix,

$${}^E R_{T/E} = R_{T/E} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \quad (2)$$

where the subscript T/E means “target w.r.t end-effector”. The superscript E in ${}^E R_{T/E}$ indicates that the rotation matrix is expressed in the end-effector frame and is usually avoided, except in cases such as ${}^T R_{T/E}$ which means “rotation matrix of the end-effector w.r.t to the target expressed in the target frame” or when a third frame is involved.

The rotation matrix and translation vector can be combined in a homogeneous transformation matrix:

$${}^E T_{T/E} = T_{T/E} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & x \\ r_{21} & r_{22} & r_{23} & y \\ r_{31} & r_{32} & r_{33} & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3)$$

The rotation matrix can be expressed in terms of the unit quaternion,

$$q = q_0 + \vec{q} = q_0 + q_1 \vec{I} + q_2 \vec{J} + q_3 \vec{K} \quad (4)$$

and the transformation matrix between the frames can be expressed as:

$$T_{T/E} = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1 q_2 - q_0 q_3) & 2(q_0 q_2 + q_1 q_3) & x \\ 2(q_0 q_3 + q_1 q_2) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2 q_3 - q_0 q_1) & y \\ 2(q_1 q_3 - q_0 q_2) & 2(q_0 q_1 + q_2 q_3) & q_0^2 - q_1^2 - q_2^2 + q_3^2 & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (5)$$

To summarize, there are four references frames involved in the pose estimations problem (see Figure 6):

- E : End effector frame, it is the main frame, the output of the pose estimation is defined in this frame.

- $C1$: Camera-1 frame. It is a standard camera frame (z -axis aligned with the focal axis). The position of this frame with respect to the end-effector frame is constant and is determined experimentally using a camera extrinsic calibration algorithm.
- $C2$: Camera-2 frame. Similar to camera-1 frame.
- T : Target Frame, the objective of the pose estimation system is to determine the pose of this frame with respect to the end-effector frame.

5.3 Algorithm General Description

The main algorithm flowchart is present in Figure 8. It has the following main subsystems: image acquisition (2), feature extraction (2), feature detection (2), pose solver, circle pose calculation and an extended Kalman filter. They are described in the following sections.

The pose estimation system uses the target features to estimate pose. It is assumed that the geometry of these features are known. At each iteration (i.e. each captured image) the system needs to identify the projected features on each camera image corresponding to the features of interest in the target or features to track, as shown in Figure 9. It is assumed that Camera-1 is tracking a circle, or its corresponding projected feature which is an ellipse. The feature identification is achieved by using the feature extraction system and the feature detection system. They give the detected features which are an estimation of the projected image features. Once the features have been identified, the circle pose calculator system uses the detected ellipse to find the 3D-position of the circle center and its normal vector. At the same time the Pose Solver finds pose using the conjugate gradient method to optimize a cost function and to reduce the re-projection error of the detected features, which reduces the pose estimation error. Finally, an extended Kalman filter merges data from the pose solver and the ellipse detection system to give the final estimated pose. From Figure 8 it can be seen that the ellipse detection system or EDS [87], [88] is immersed in the flowchart. EDS is a standalone pose estimation system for circles, it was developed at the WVRTC and it uses some of the components of the proposed pose estimation system.

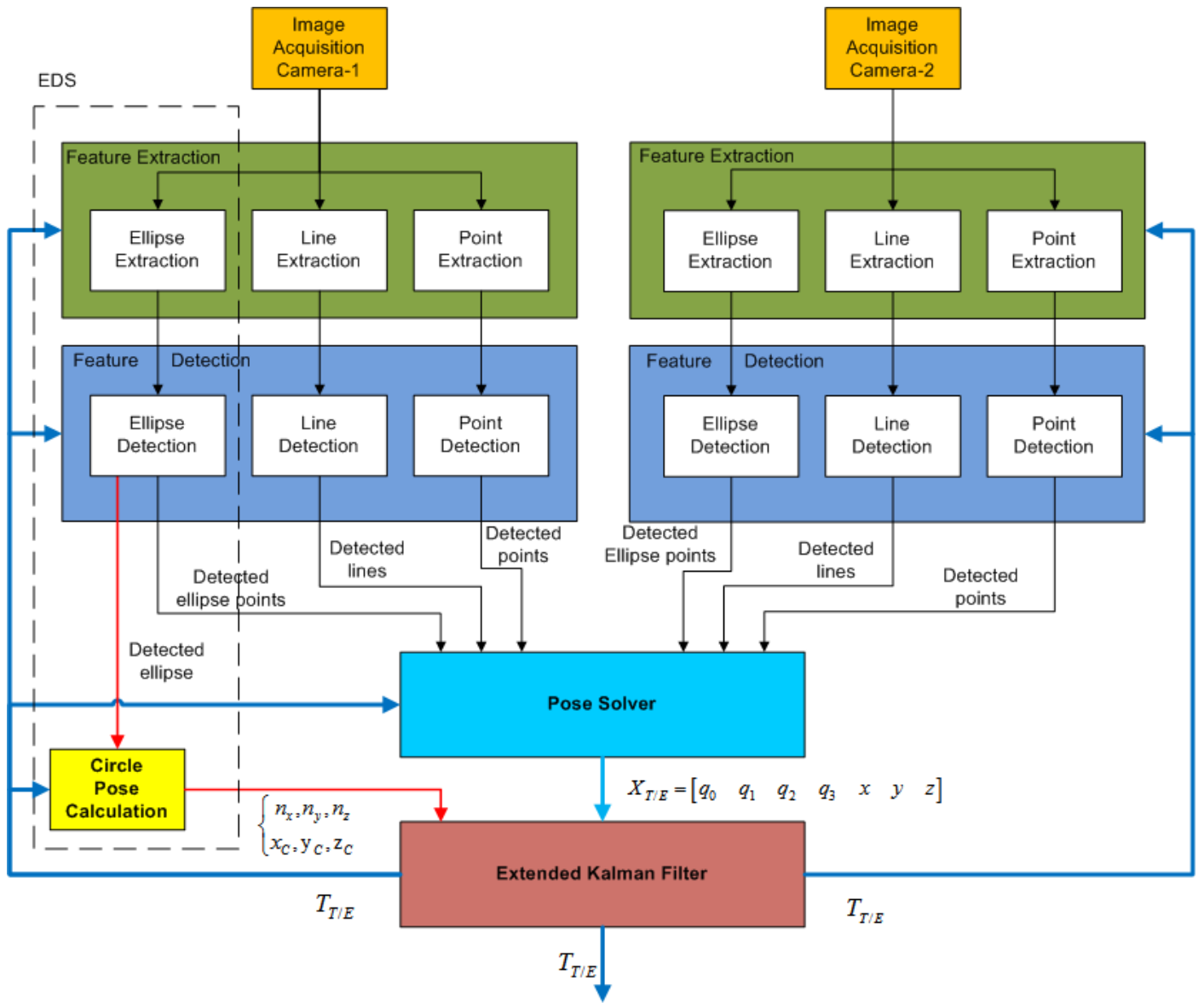


Figure 8. Main algorithm flowchart

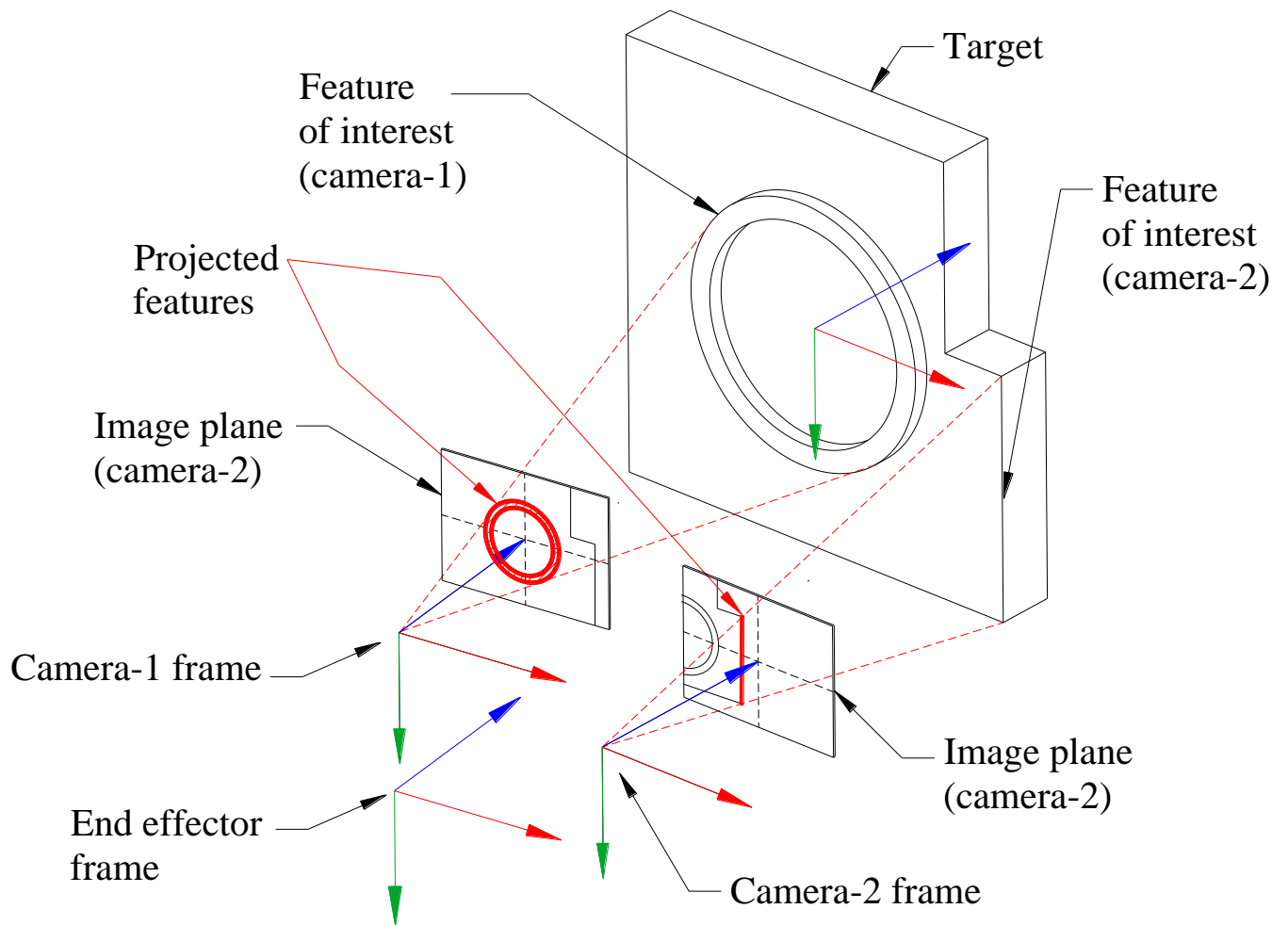


Figure 9. Target features and projected features

5.3.1 Image Acquisition

In this step the images from the cameras are acquired and converted to OpenCV matrices. Each image has three channels, each channel representing the image components in the RGB color space.

5.3.2 Feature Extraction

The inputs of the feature extraction system are the acquired color images. A particular channel of the image is selected according to the feature of interest, this is performed in order to increase the contrast between the features of interest and the background, as an example: when detecting the interface ring the blue channel is used since it has a better contrast between the ring and the MLI. The single channel images are then processed by a canny edge detector (from OpenCV) to find the edge binary map.

The Feature Extraction subsystem uses the edge binary map to extract the geometric features on the image that are of the same type of a particular feature of interest. There are three different types of feature of interest: points/corners, line segments and ellipses (projection of circles in the image plane). If the particular of interest is a circle, then the feature extraction system will extract all the ellipses on the image.

As can be seen from Figure 8 there are two modules of the feature extraction system, one for each camera. The system can work with at least one circular feature for camera-1¹ and then just one feature extraction module is used, or it can work with the three different type of feature for each camera.

Point/corner Extraction is used when the features to track are corners or very small objects like bolt heads or holes. The Minimum Eigenvalue Method (from OpenCV) is used; it gives the coordinates, in pixels, of the extracted corners.

Line Extraction is used when straight edges are tracked, the Hough transform is used to perform line extraction (from OpenCV), which gives the coordinates, in pixels, of the two extreme points of each of the extracted line segments.

When tracking a circular feature, since its projection on the image plane is an ellipse, the Fast Ellipse Extractor algorithm [81] from the LTI library is used, it gives the geometric parameters of the extracted ellipses: coordinates of the center, the length of the major semi-axis and minor semi-axis in the image plane, all in pixels, and the angle of rotation of the ellipse, in radians. Additionally points that belong to ellipses (ellipse points) are also extracted.

As can be seen from the flowchart of Figure 8 the feature extraction system uses the EKF's estimated pose, which is used to limit the extraction to a particular region of interest (ROI) in the image and improve the extraction process.

The outputs of the feature extraction system are sets of extracted features, each set is associated with a particular feature of interest in the target.

5.3.3 Detection

The sets of extracted features may contain the true features (the ones that are being tracked and correspond to real feature in the satellite) and false features that can be due to noise, occlusion, shadows, reflections or real features that are not used for tracking. The function of the detection system is to solve the correspondence problem and separate (if present) the true features from the false features and inform when the track of a

¹ only 5DOF pose will be obtained

particular feature is lost. The inputs of the Detection system are the predicted pose from the Extended Kalman filter, the geometry of the target features and the extracted features.

The Detection system (see Figure 10) uses the pin-hole model to project the geometry of a target feature in the image plane using the predicted transformation matrix given by the EKF, then the obtained projected feature, known as the reference feature, is compared with the set of corresponding extracted features, the best match is declared as the detected features, and they are the output of the detection system. The comparison is performed using a cost function, where the cost represent the distance of the extracted feature to the reference feature. Features with a cost larger than a threshold are ignored. The threshold is dynamic and is computed using a moving average filter over the cost of the last 10 detected features. In the case of points and lines several extracted features can be declared as detected features, and an additional step is performed where they are averaged to find a unique detected feature.

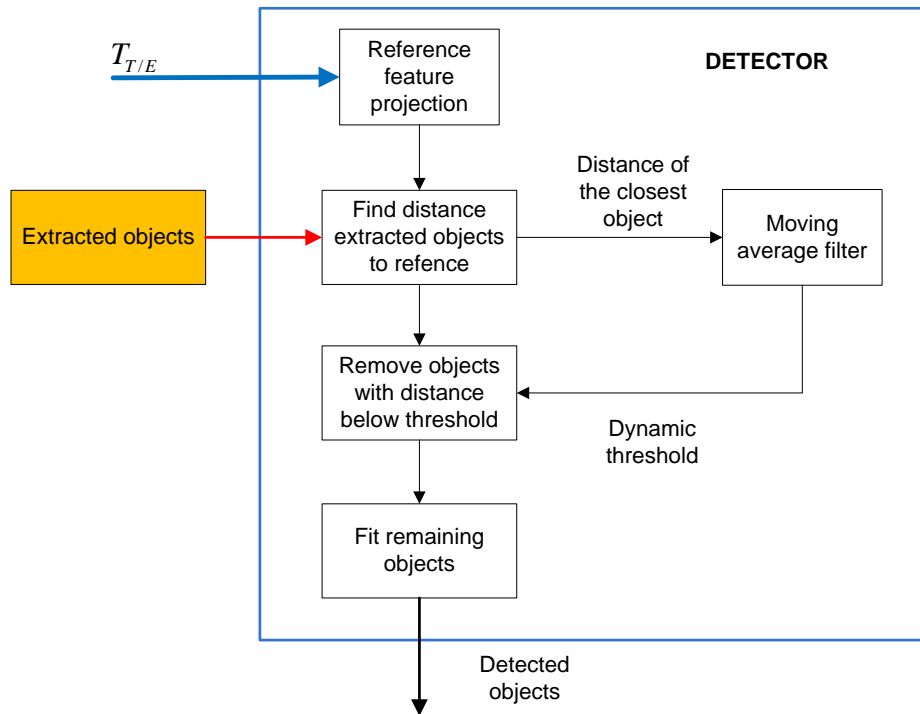


Figure 10. Detector flowchart

As an example the projection of a circular feature of the target in the image plane is an ellipse. The ellipse extractor will give several ellipses. To compare the extracted vs. the reference ellipses the following cost function, J , can be used,

$$J_i = (a_{ref} - a_i)^2 + (b_{ref} - b_i)^2 + (x_{ref} - x_i)^2 + (y_{ref} - y_i)^2 \tag{6}$$

Where a and b are the length of the ellipse major and minor semi-axis respectively and (x, y) are the coordinates of the ellipse center. The subscripts “*ref*” and “*i*” refers to the geometric parameters of the reference ellipse and extracted ellipses, respectively. The extracted ellipse with the smallest value of the cost function is then declared as the detected ellipse. An upper limit for the minimum value of the cost function can be used to declare loss of track of the circular feature.

The detected ellipse is given to the circle pose calculation system to find the 3D-position of the circle center and its normal vector. The detection system also gives the detected ellipse points, which are points that are close to the reference ellipse. The set of detected ellipse points in addition to the detected lines and corners are given to the pose solver.

5.3.4 Circle Pose Calculation and EDS

The circle pose calculation system finds the pose of the circle in the target using the ellipse corresponding to its projection on the image plane. Having the geometric parameters of the detected ellipse (which is an estimation of the actual projected ellipse), the coefficients (A, B, C, D, E and F) of the quadratic equation of the detected ellipse:

$$Ax^2 + Bxy + Cy^2 + Dxz + Eyz + Fz^2 = 0 \tag{7}$$

are found analytically. Using these coefficients and the method described in [69] the 3D-coordinates of the circle center and its normal vector can be found analytically. It is important to note that the method proposed in [69] has two possible solutions, but the duality is solved by comparing the two solutions with the reference solution given by the EKF’s estimated transformation matrix.

5.3.5 Pose Solver

The pose solver uses all the detected features from each camera to find pose. It is achieved by minimizing a cost function that related the distance between the detected features and the reference features. The detected features are given by the feature detection system. The concept of the pose solve is presented in Figure 11 and described below.

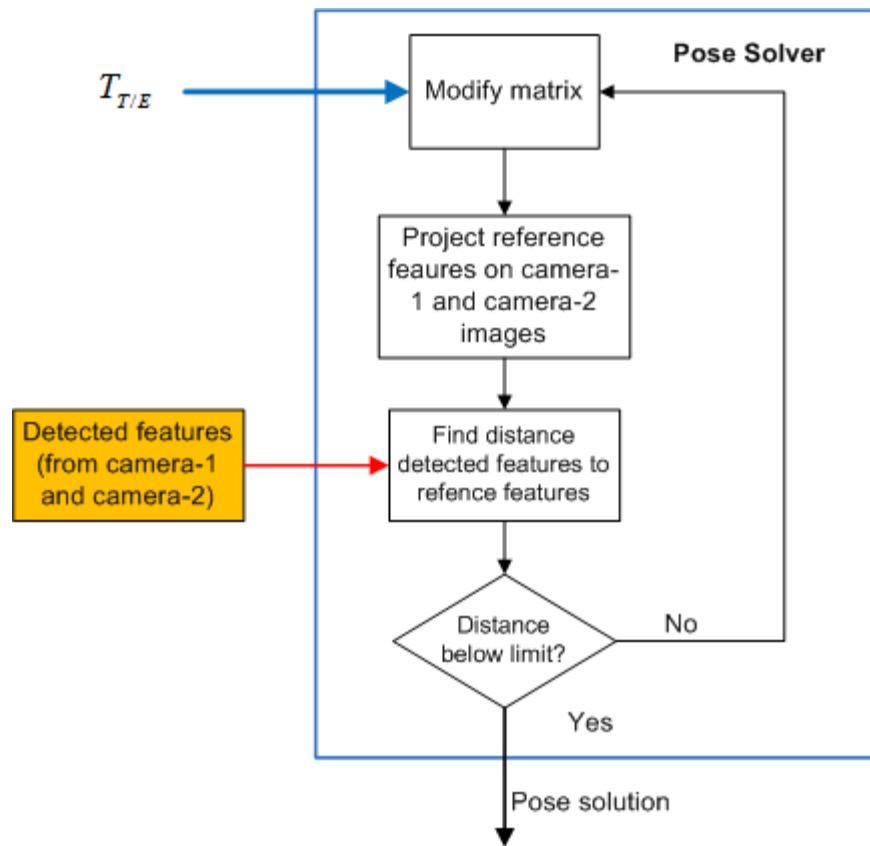


Figure 11. Pose solver concept

Using an assumed transformation matrix between the target and the end-effector, the reference features are obtained using the pin-hole model to project the target features in the corresponding camera image. The assumed transformation matrix is a small modification of the EKF's estimated transformation matrix at the previous time. The reference features are compared with the detected features to find the cost corresponding to the particular assumed transformation matrix. Several assumed matrices are used in an iterative process until the maximum allowed number of iterations are reached or until the cost is lower than a specified value. In the latter case the pose solution is found.

The previously described process is only the concept, since no criteria is established to modify the transformation matrix, and there are infinity ways to modify the matrix. To avoid this problem the minimization is performed using the conjugated gradient method, it minimizes the cost function using the gradient of the cost function. This method requires that the cost function has to be expressed in terms of the modified transformation matrix and obtain its gradient, this is described in Chapter 9.

5.3.6 Extended Kalman Filter

An extended Kalman filter (EKF) is used to obtain the estimated pose using the information given by the pose solver and the ellipse detection system. An extended Kalman filter is used due to the non-linearities of the system. Chapter 10 describes the EKF in more detail.

The following chapters describe the main components of the pose estimation system, the description is focused on the particular application of this research, i.e. estimating the pose of a satellite that has an interface ring.

6 FEATURE EXTRACTION SYSTEM

The feature extraction system has three components: the ellipse extraction system, the line extraction system and the point extraction system. The following sections describe each component.

6.1 Ellipse Extraction

The input of this subsystem is the color image of the target, which in this case is a satellite mockup with the interface ring. The task of the system is to identify all the ellipses that are on the image. To achieve that purpose the Fast Ellipse Extractor (FEE) algorithm described in [81] is used. The input of the algorithm is the binary edge map of the image and the outputs are the geometric parameters of the extracted ellipses (see Figure 12), which are the coordinates of the center (x, y) in the image plane, the length of the major semi-axis (a) , the length of the minor semi-axis (b) , all in pixels, and the angle of rotation of the ellipse, in radians.

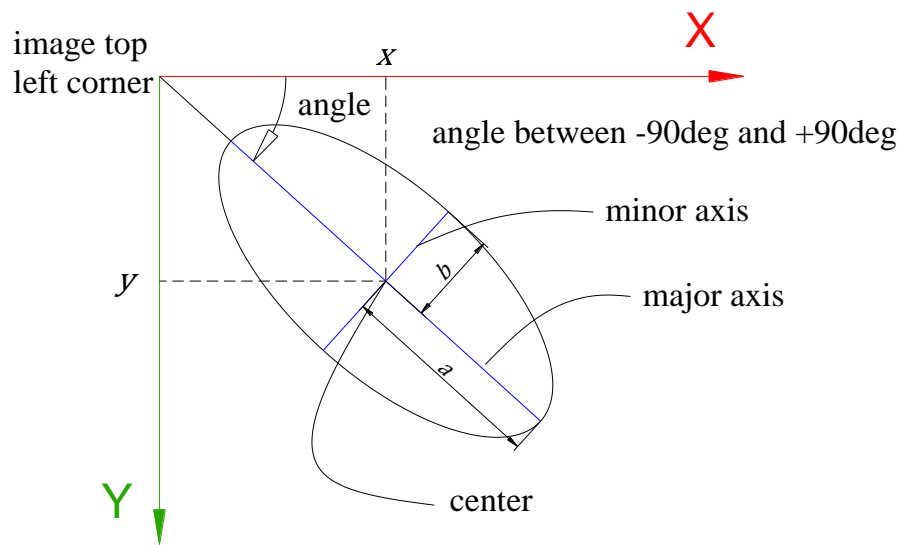


Figure 12. Ellipse geometric parameters

To obtain a clean edge map that the ellipse extractor can use, some standard image processing techniques are used, as can be seen in Figure 13. Using the color image (step 1 of Figure 13) the blue channel is selected (step 2), since it provides a better contrast between the interface ring and the MLI background. Next the canny edge detector is applied over the blue channel to find the binary edge map (step 3), which is the input of the FEE algorithm. The image in step 4 shows the extracted ellipses, and as can be seen not all of them correspond to actual circular features in the scene. The image corresponding to step 5 shows the extracted ellipse points.

The FEE algorithm extracts line segments from the edge map, they are formed by at least two adjacent pixels, and next, adjacent segments are merged to create lines. The adjacent lines are then combined to create circular arcs. These arcs are represented by the coordinates of their center and their initial and final points. Extended arcs, which are elliptical arcs, are created by grouping arcs. Finally ellipses are formed by at least one extended arc. At each step several thresholds are defined to decide if an object is qualified to be part of a higher level object.

Besides the ellipse geometric parameters, any information related to the lower level objects can be accessed; of particular interest are the coordinates of arc endpoints, which are referred as the extracted points and are also an output of the Ellipse Extraction subsystem, they can be seen in step number 5 of Figure 13.

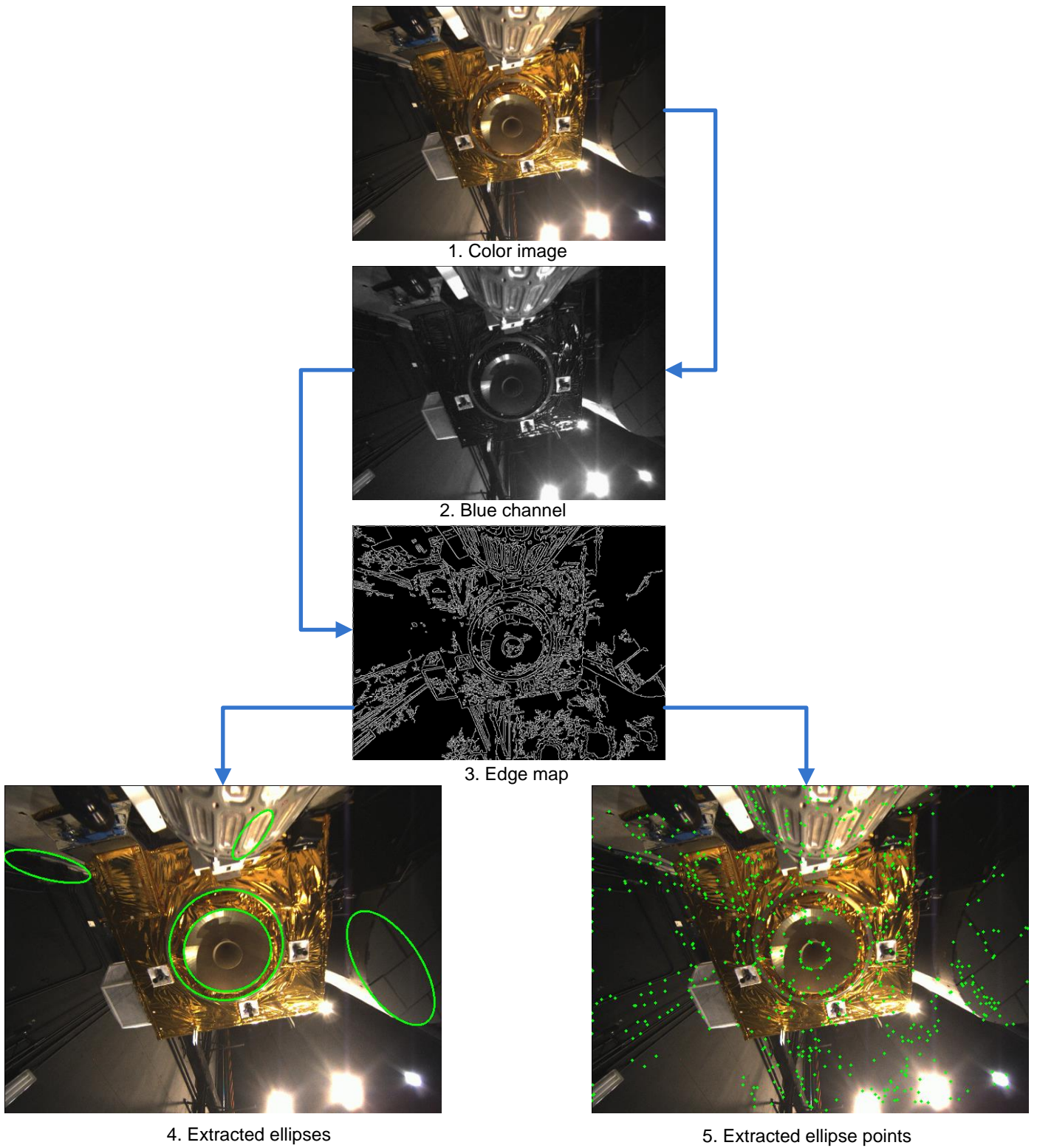


Figure 13. Image processing operations in the ellipse extraction system

6.2 Line Extraction

The input of the line extraction subsystem is the color image of the target acquired by the camera. The system has to extract all the lines that are on the image. This is achieved using the Hough transform algorithm from the OpenCV library. The algorithm provides the coordinated of extreme points of the extracted lines. The coordinates of each point are expressed in pixels w.r.t the image top left corner. The input of the OpenCV Hough transform is the edge map of the image.

In order to obtain an adequate edge map and to limit the line extraction to a particular region, the image processing techniques presented in Figure 14 are applied, where it is assumed that the borders of the white squared pad corresponding to the thruster base are the features of interest. Using the color image (step 1 of Figure 14) the blue channel is selected (step 2), since it provides a better contrast between the white pad and the MLI background. After that, an image thresholding is performed (step 3) where the pixels of the image with an intensity lower than a threshold value are given a value of zero (black) otherwise they are given a value of 255 (white). Next a mask is generated (step 4) using the previous time pose estimated by the EFK. The mask is applied over the thresholded image (step 5) to find the region of interest. Next the canny edge detector is applied over the ROI (step 6) and finally the OpenCV Hough transform is applied (step 7) to extract the lines on the image. As it can be seen there are several lines for each side of the pad. All these lines are the output of the line extraction system.

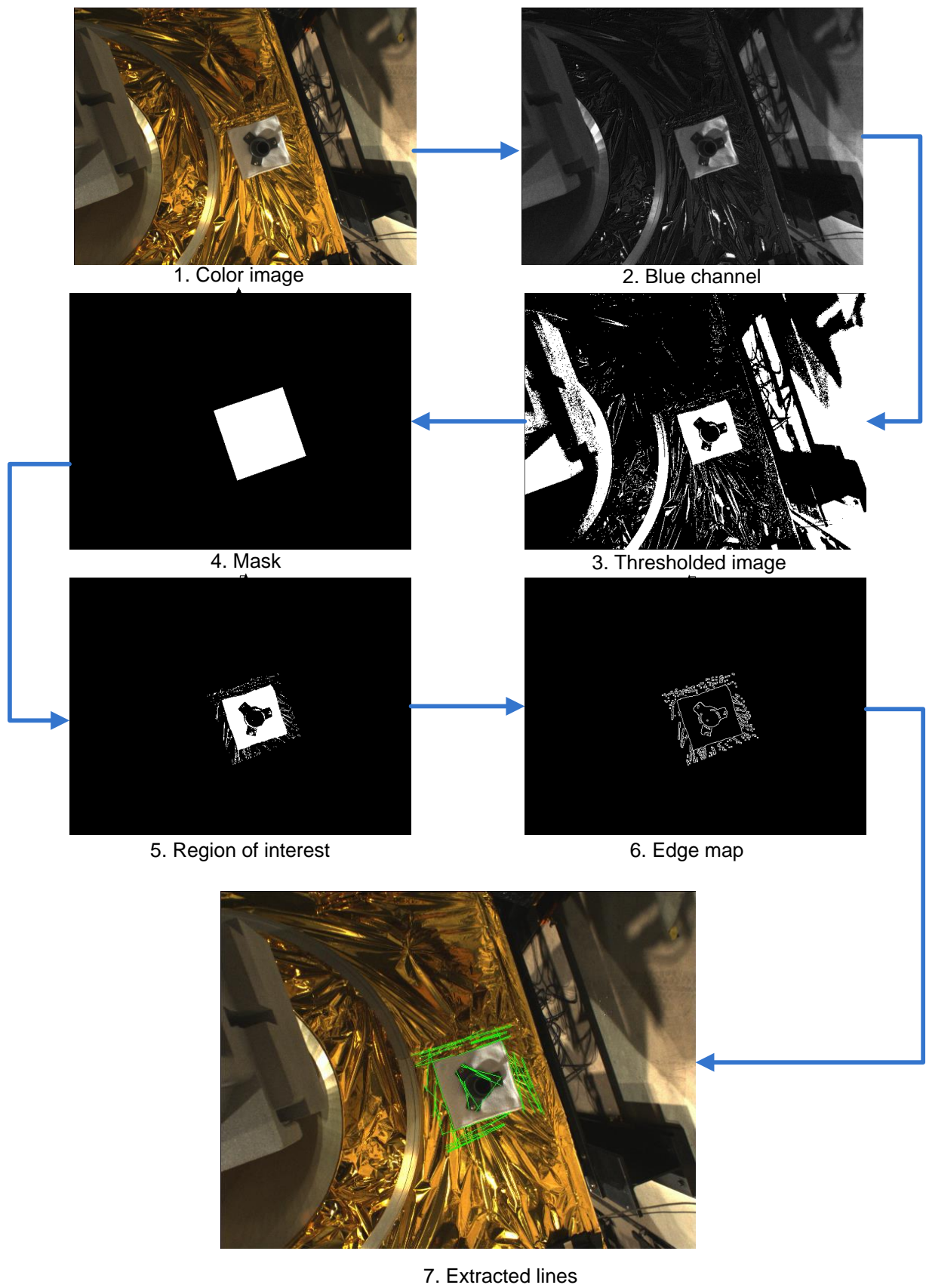


Figure 14. Image processing operations in the line extraction system

6.3 Point Extraction

The input of the point extraction subsystem is the color image of the target acquired by the camera. The system extracts all the isolated points and corners that are on the image. This is achieved by using the OpenCV function `goodFeaturesToTrack`. The function's input is a binary or gray-scale image (not an edge map) and it provides the coordinates the extracted pints in pixels and w.r.t the image top left corner.

Figure 15 shows the image processing techniques performed in the point extraction system, where it is assumed that the corners of the white pad corresponding to the AOC thruster are the features of interest. Using the color image (step 1 of Figure 15) the blue channel is selected (step 2), since it provides a better contrast between the white pad and the MLI background. After that, an image thresholding is performed (step 3) where the pixels of the image with an intensity lower than a threshold value are given a value of zero (black) otherwise they are given a value of 255 (white). Next a mask is generated (step 4) using the previous time pose estimated by the EFK. The mask is applied over the thresholded image (step 5) to find the region of interest. Finally the OpenCV `goodFeaturesToTrack` function is applied (step 6) to extract the corners on the image. As it can be seen there are several point around the actual corner of the pad and points that represents other features. All these points are the output of the point extraction system.

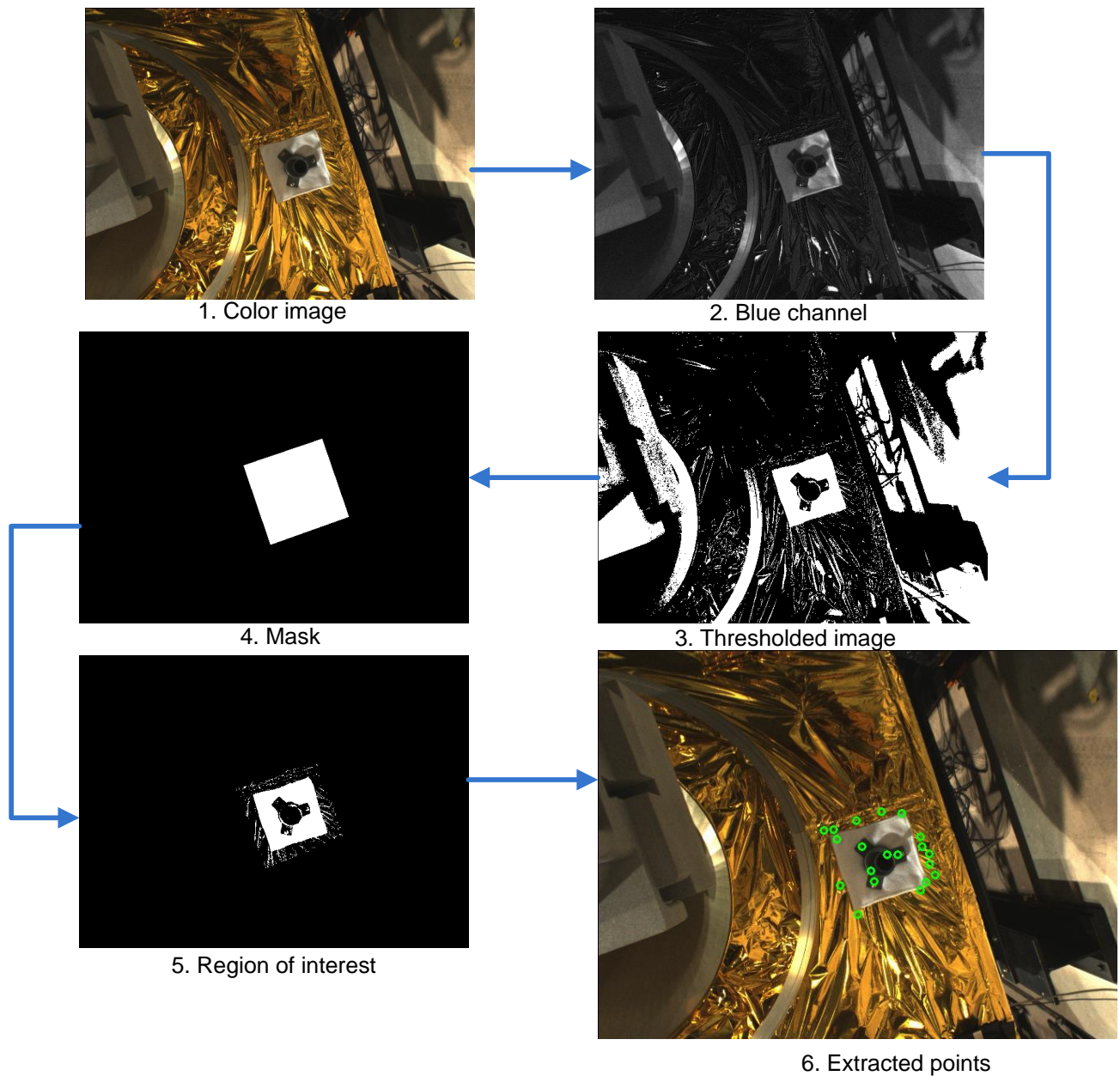


Figure 15. Image processing operations in the point/corner extraction system

As can be noted from the previous images, there are several extracted features, some of them correspond to actual features of interest on the target, and others are actual features on the target but are not features of interest, finally there are features that are noise or wrong result from the extraction system, nevertheless all these features are passed to the detection system which will estimate the extracted features that correspond to the features of interest on the target.

7 FEATURE DETECTION SYSTEM

The feature detection system has three subsystems: the ellipse detection system, the line detection system and the point detection system. The following sections describe each subsystem.

7.1 Ellipse-Curve and Ellipse-Point detection

The ellipse detection system has two subsystems, the ellipse-curve detection and the ellipse-point detection system, which are described in the following sections.

7.1.1 Ellipse-Curve Detection

As depicted in Figure 16, the main function of the ellipse-curve detection system is to find the ellipse from the set of extracted ellipses (green ellipses on Figure 16) that best match the projection of the circular feature of interest, in this case the interface ring, this projected ellipse is known as the reference ellipse (red ellipse) and the best match is declared as the detected ellipse (blue ellipse). The system uses the EKF's estimated transformation matrix at the previous time and the pin-hole model to project the circular feature on the image plane and obtain the reference ellipse. It is assumed that the geometry of the ring is known.

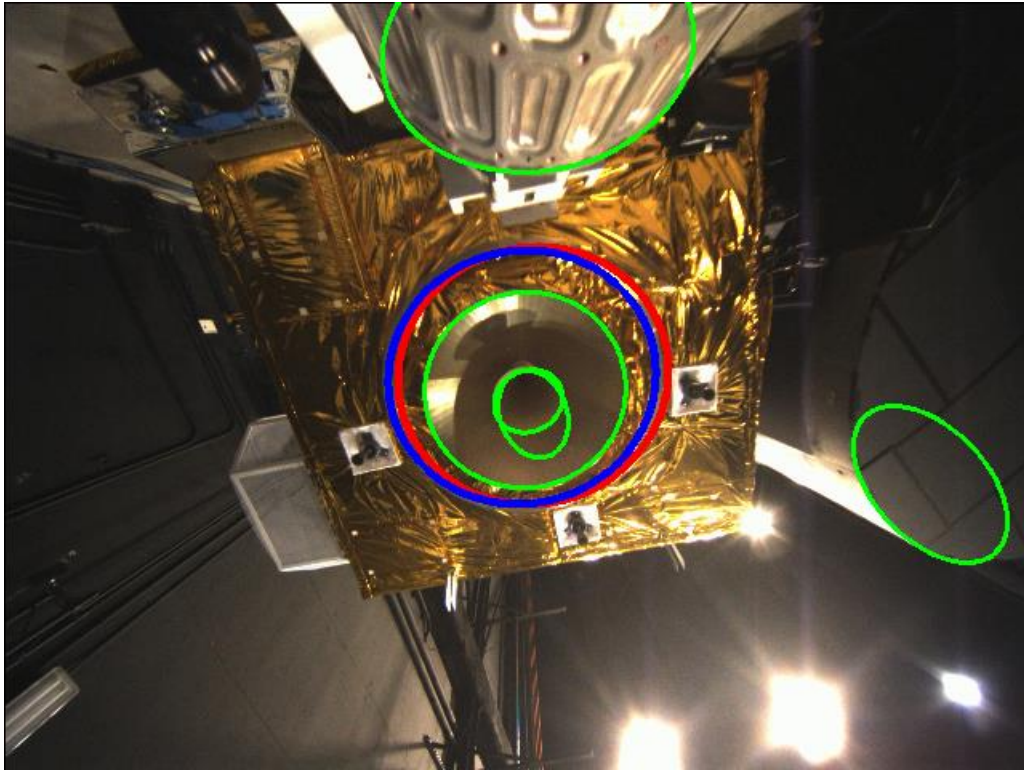


Figure 16. Reference ellipse (red), detected ellipse (blue) and extracted ellipses (green)

To select the best match the detection system uses a cost function in terms of the geometric parameters (a : major semi-axis, b : minor semi-axis, x : center x-coordinate and y : center y-coordinate) of the ellipses, the proposed cost function is:

$$J_{i,E} = w_a(a_{ref}-a_i)^2 + w_b(b_{ref}-b_i)^2 + w_x(x_{ref}-x_{c_i})^2 + w_y(y_{ref}-y_{c_i})^2 \quad (8)$$

where the variables with subscript "ref" indicate the geometric parameters of the reference ellipse, the subscript "i" refers to the geometric parameter of the extracted ellipses, w_a , w_b , w_x and w_y are the associated weights. The ellipse with the smallest value of the cost function ($J_{min,E}$) is then declared as the detected ellipse, and its geometry parameters are the output of the ellipse detection system.

An additional requirement is that the ellipses with a value of the cost function larger than a threshold are ignored. The value of the threshold is dynamic and is computed as the average of the cost of the ten previous best matched ellipses.

7.1.2 Ellipse-Points Detection

In addition to the extracted ellipses the system also uses the ellipse points, which are points that belong to ellipses. Figure 17 and Figure 18 show the features associated with the ellipse-point detection system. The

main function of this system is to detect the points from the set of extracted ellipse-points (green points on Figure 17) that are closer to the reference ellipse (red ellipse), the points that are closer to the reference ellipse are declared as the detected ellipse-points (blue points) and are the output of the system. The reference ellipse is obtained as described in the previous section.

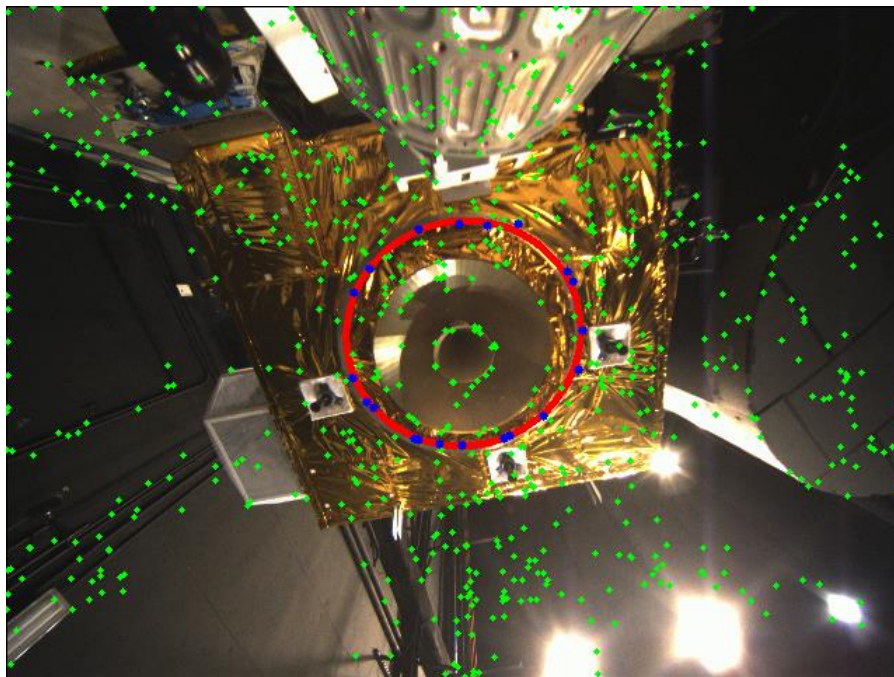


Figure 17. Camera-1 image. Reference ellipse (red), detected ellipse points (blue) and extracted ellipse points (green)

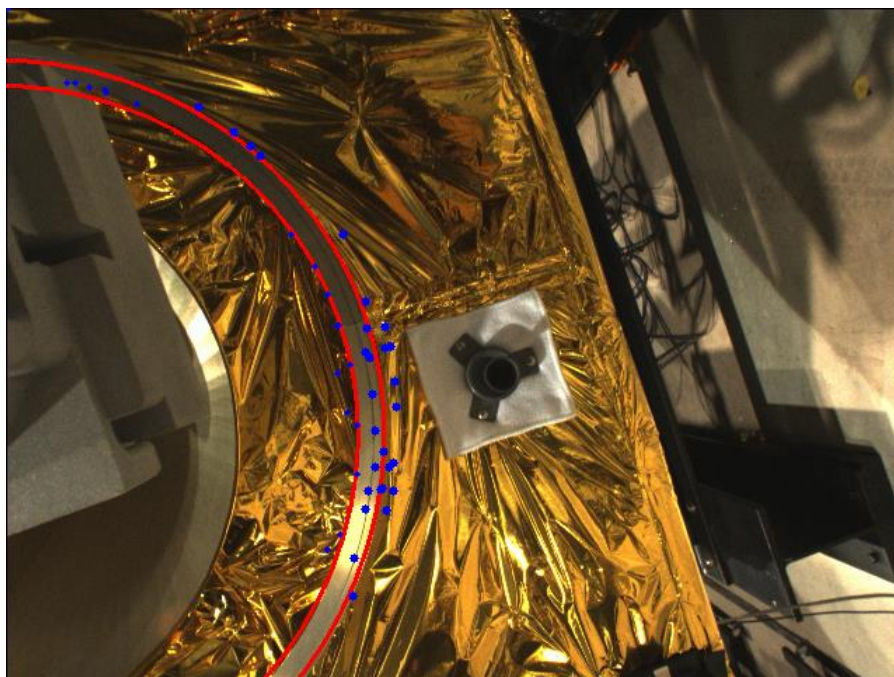


Figure 18. Camera-2 image. Reference ellipse (red), detected ellipse points (blue)

To find the detected ellipse-points the detection system uses a cost function which is defined in terms of the coefficients of the quadratic equation of the reference ellipse. The proposed cost function is:

$$J_{i,EP} = \frac{(Au^2 + Buv + Cv^2 + Du + Ev + F)^2}{0.25(f_x + f_y)^2 AB} \quad (9)$$

where (u_i, v_i) are the coordinate of the extracted points under evaluation, A, B, C, D, E and F are the coefficients of the reference ellipse equation, f_x and f_y are the camera intrinsic parameters corresponding to the focal distance. Clearly if an extracted point belongs to the reference ellipse the value of its cost function is zero. The points with a value of the cost function smaller than a threshold are declared as the detected ellipse-points and as in the previous case the value of the threshold is dynamic and is computed as the average of the cost of the ten previous best matched points.

The coefficients of the reference ellipse equation in terms of transformation matrix can be found analytically, the procedure to obtain them is outline below.

A point $P(x_{P/T}, y_{P/T}, z_{P/T})$ that belongs to a circle in the target satisfies the following equation

$$(y_{P/T} - Y_C)^2 + (z_{P/T} - Z_C)^2 = r^2 \quad (10)$$

where it is assumed that the center is on a plane parallel to the YZ-plane and its center is located at $C(X_C, Y_C, Z_C)$. The circle radius is r . The same point expressed in the camera frame is,

$$P = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}_{P/C} = T_{T/C} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}_{P/T} \quad (11)$$

where $T_{T/C}$ is the transformtin matrix between the target and the camera that is traking the circle. Next, the pin-hole model is used to find the projection (u, v) of the circle point in the camera image plane,

$$\begin{cases} u = f_x \frac{x_{P/C}}{z_{P/C}} + c_x \\ v = f_y \frac{y_{P/C}}{z_{P/C}} + c_y \end{cases} \quad (12)$$

where f_x, c_x, f_y and c_y are the intrinsic parameters of the camera. Substituting equation (11) in Equation (12) an explicit expression for $y_{P/T}$ and $z_{P/T}$ can be obtained, which has the form,

$$\begin{cases} y_{P/T} = x_{P/T}(u, v, x_{P/T}, T_{T/C}) \\ z_{P/T} = y_{P/T}(u, v, x_{P/T}, T_{T/C}) \end{cases} \quad (13)$$

Since the $P(x_{P/T}, y_{P/T}, z_{P/T})$ belong to the target circle, then, if the previous expression is used in Equation (10), an expression for the equation of the projected ellipse can be found,

$$Au^2 + Buv + Cv^2 + Du + Ev + F = 0 \quad (14)$$

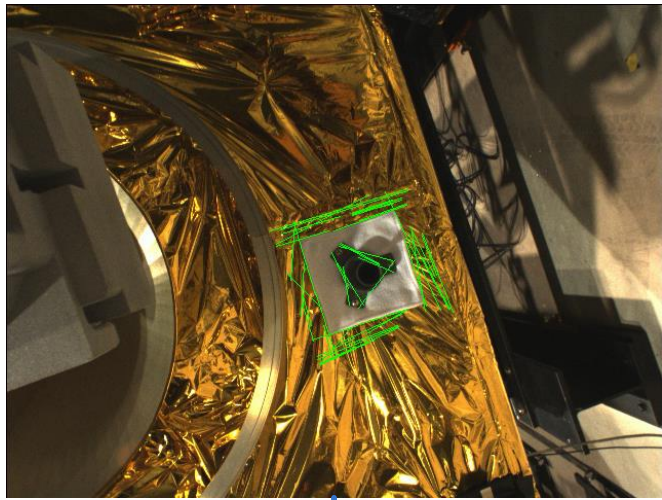
where A, B, C, D, E and F are the coefficients of the ellipse equation in the image plane, these coefficients are function of the transformation matrix ($T_{T/C}$), camera intrinsic parameters and the $x_{P/T} = X_c$ coordinate, the last two elements are constant, while the transformation matrix changes with time (due to relative motion), as an example the coefficient A , can be expressed as,

$$A = A(T_{T/C}) \quad (15)$$

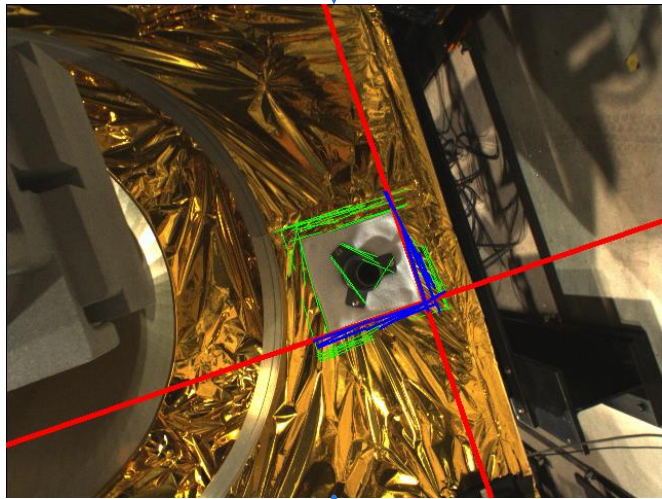
7.2 Line Detector

As presented in Figure 19 the main function of the line detector system is to identify the lines from the set of extracted lines (green points on step 1 of Figure 19) that are closer to reference lines (red lines). In the example of Figure 19 two lines of the borders of the white pad are being used as a feature of interest, and their corresponding corner. The lines that are closer to the reference lines are declared as the detected lines (blue lines in step 2). The reference lines are obtained using the EKF's estimation transformation matrix at the previous time to project the lines of interest (3D-lines of the target) on the image plane, these projections are the reference lines. It is assumed that the geometry of the lines in the target is known.

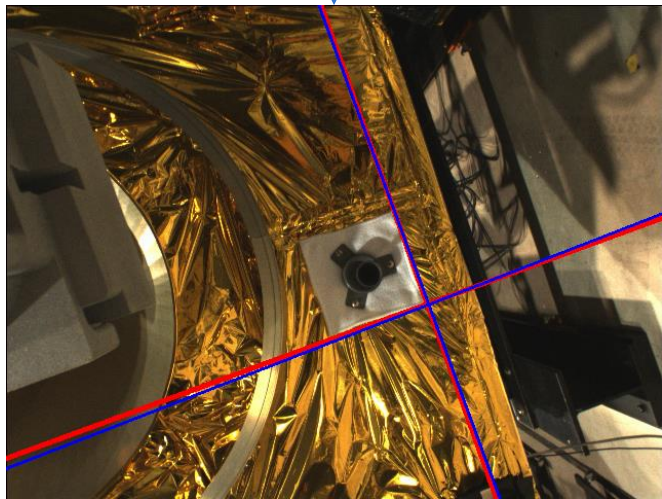
As can be seen in Figure 19 there are several detected lines for each reference line, an additional step is to fit all these lines in one line for each reference line, these lines are declared as the fitted lines and are the output of the line detection system.



1. Extracted lines



2. Detected lines



3. Fitted lines

Figure 19. Images of the operation of the line detector

To find the detected lines the detection system uses a cost function which is defined in terms of the coefficients of the reference line equation. The proposed cost function is:

$$J_{i,L} = \frac{|A_L u_{i1} + B_L v_{i1} + C_L|}{A_L + B_L} + \frac{|A_L u_{i2} + B_L v_{i2} + C_L|}{A_L + B_L} \quad (16)$$

where u_{i1}, v_{i1}, u_{i2} and v_{i2} are the coordinates of the points that define the extracted line under evaluation, A_L , B_L , and C_L are the coefficients of the reference line equation,

$$A_L u + B_L v + C_L = 0 \quad (17)$$

these coefficients are function of the transformation matrix.

This proposed cost functions represents the square of the Euclidean distance between the reference line and the points of the extracted line. Note that if an extracted line match the reference line the value of its cost function is zero. The extracted lines with a value of the cost function smaller than a threshold are declared as the detected lines and as in the previous case the value of the threshold is dynamic and is computed as the average of the cost of the ten previous best matched lines.

The coefficient of the 2D-line equation can be expressed in terms of the transformation matrix using the following procedure. An arbitrary point, P_t , belonging to a 3D-line of the target is describes by,

$$P_t(t) = V_L t + P_L \quad (18)$$

where V_L is a vector parallel to the 3D-line in the target and P_L is a point on that line, and t is the parameter of the equation and can take any real value. The previous two vectors can be expressed in the camera frame using the transformation matrix,

$$V_L = \begin{bmatrix} v_x \\ v_y \\ v_z \\ 0 \end{bmatrix}_{L/C} = T_{T/C} \begin{bmatrix} v_x \\ v_y \\ v_z \\ 0 \end{bmatrix}_{L/T} \quad (19)$$

and,

$$P_L = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}_{L/C} = T_{T/C} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}_{L/T} \quad (20)$$

then the components of the point P_t , w.r.t. the camera frame can be expressed as,

$$\begin{cases} x_{L/C} = v_{xL/C} t + x_{L/C} \\ y_{L/C} = v_{yL/C} t + y_{L/C} \\ z_{L/C} = v_{zL/C} t + z_{L/C} \end{cases} \quad (21)$$

Using the pin-hole it is obtained,

$$\begin{cases} u = f_x \frac{x_{L/C}}{z_{L/C}} + c_x \\ v = f_y \frac{y_{L/C}}{z_{L/C}} + c_y \end{cases} \quad (22)$$

The projection of the 3D-line is also a line in the image plane (plane uv), which has the following equation,

$$A_L u + B_L v + C_L = 0 \quad (23)$$

The expression for the line equation coefficients can be found using three different and arbitrary values for the parameter t of Equation (22). After that, it can be seen that the coefficients are function of the transformation matrix between the target and the particular camera,

$$\begin{cases} A_L = A_L(T_{T/C}) \\ B_L = B_L(T_{T/C}) \\ C_L = C_L(T_{T/C}) \end{cases} \quad (24)$$

7.3 Point Detector

As presented in Figure 20 the main function of the point detector system is to identify the points from the set of extracted points (green points in Figure 20) that are closer to reference points (red points). In the example of Figure 20 one of the corners of the white pad is the feature of interest. The extracted points that are closer to the reference points are declared as the detected points (blue points). The reference point is obtained using the EKF's estimation transformation matrix at the previous time to project the point of interest (3D-point on the target) on the image plane, this projection is the reference point. It is assumed that the geometry of the point in the target is known.

If there are several detected points, an additional step is to average all these points in one point, and this is declared as the fitted point and is the output of the point detection system.

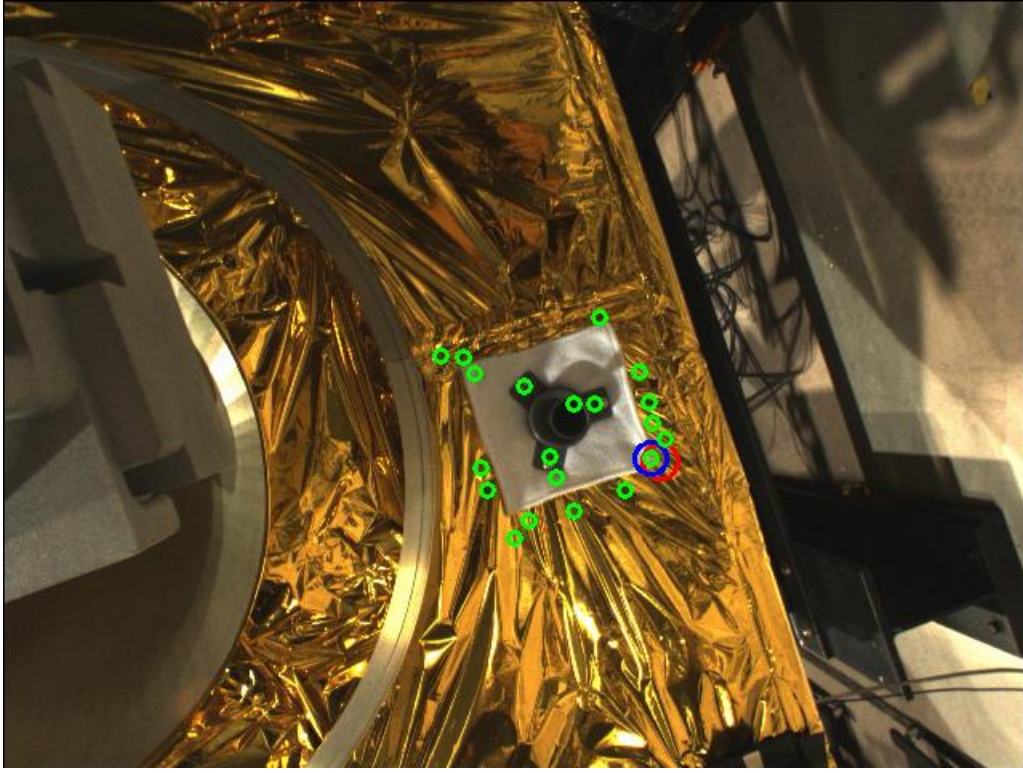


Figure 20. Reference point (red), detected point (blue) and extracted points (green)

To find the detected points the detection system uses a cost function which is defined in terms of the coordinates of the reference point. The proposed cost function is:

$$J_{i,P} = (u_{ref} - u_i)^2 + (v_{ref} - v_i)^2 \quad (25)$$

where u_i and v_i are the coordinates of the extracted point under evaluation, u_{ref} and v_{ref} are the coordinates of the reference point. This cost function represents the square of the Euclidean distance between the reference point and an extracted point. The points with a value of the cost function smaller than a threshold are declared as the detected points and as in the previous case the value of the threshold is dynamic and is computed as the average of the cost of the ten previous best matched points.

The reference point is obtained using the pin-hole model to project a target point, $P(x, y, z)$, in the image plane of the image,

$$P_P = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}_{P/C} = T_{T/C} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}_{P/T} \quad (26)$$

to obtain,

$$\begin{cases} u = f_x \frac{x_{P/C}}{z_{P/C}} + c_x \\ v = f_y \frac{y_{P/C}}{z_{P/C}} + c_y \end{cases} \quad (27)$$

As can be seen from (26) and (27) each coordinate is function of the transformation matrix,

$$\begin{cases} u = u(T_{T/C}) \\ v = v(T_{T/C}) \end{cases} \quad (28)$$

8 CIRCLE POSE CALCULATOR.

The feature detection system gives the geometric parameters of the detected ellipse to the pose circle calculator. As presented in Figure 21 the function of this system is to obtain the 3D-coordinates of the center (\vec{X}_C in Figure 21) and the normal vector \vec{N}_C of the circle which projection correspond to the given detected ellipse.

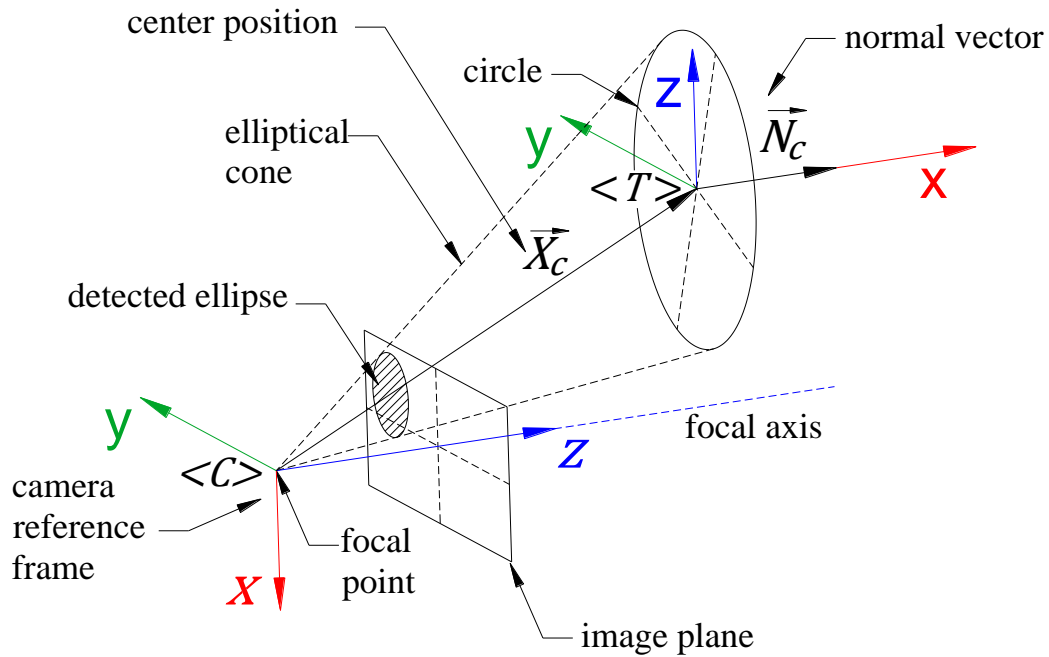


Figure 21. Detected ellipse and circle pose

To achieve this, the system uses the geometric parameters of the detected ellipse to find the coefficient of the ellipse equation. Using the coefficients of the detected ellipse the circle pose (center position and normal vector) is obtained using the method proposed in [69].

The first step in the method is to find the equation of the elliptical cone that pass through the ellipse on the focal plane (see Figure 23) and has its vertex at the focal point. This equation is expressed with respect to the camera frame and it can be obtained using the ellipse equation. In the second step, the equation of the cone is represented with respect to a new reference frame, where the equation of the elliptical cone has the canonical form (see Figure 24). In the third step the plane that intersects the cone forming a circle of the same diameter of the circle on the target (interface ring). Since there are two possible planes that satisfy this condition, they will generate two possible solution for the circle pose, i.e. two normals and two center positions (see Figure

22). In the final step the obtained solutions are transformed back to the original frame. The unique solution is obtained by comparing the two normals with EKF estimation (the reference normal is the first column of the rotation matrix of the target w.r.t. the camera). The normal closer to the reference, along with its corresponding position vector, are declared as the circle pose solution.

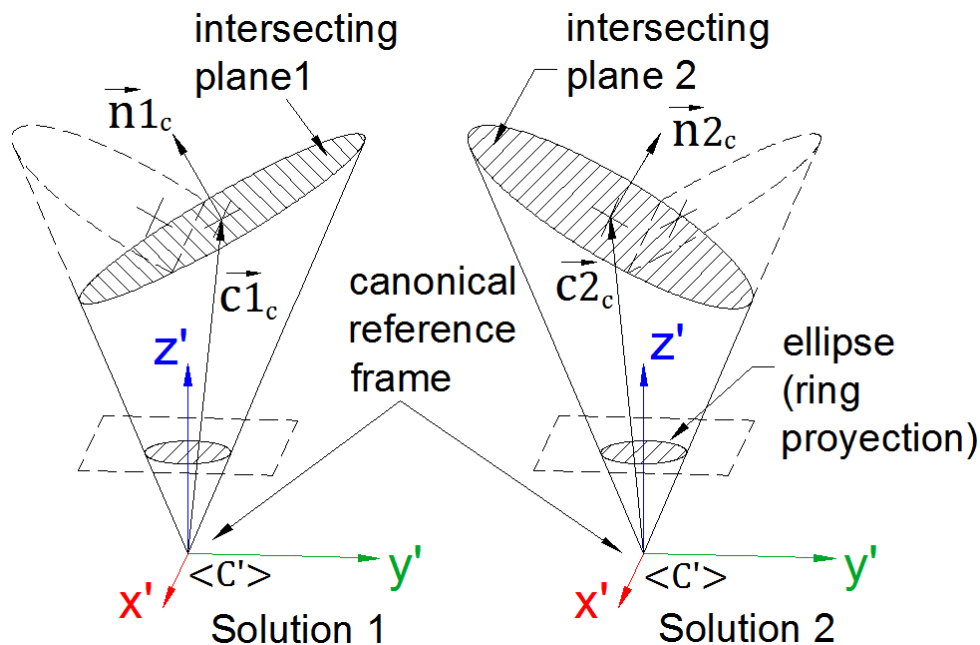


Figure 22. Canonical reference frame and two intersecting planes.

8.1 Coefficients of the Ellipse Equation in Terms of the Geometric Parameters

The coefficients, A, B, C, D, E and F of the ellipse quadratic equation,

$$Au^2 + Buv + Cv^2 + Du + Ev + F = 0 \tag{29}$$

can be obtained using the geometric parameters of the ellipse. The following are the equations that show these relationship. Let a, b, x, y and θ be the geometric parameters of the detected ellipse, as presented in Figure 12: major semi-axis, minor semi-axis, center x-coordinate, center y-coordinate and orientation angle respectively. The coefficients of the ellipse equation can be expressed in terms of these parameter, as follows,

$$A = a^2 + (b^2 - a^2)\cos(\theta)^2 \tag{30}$$

$$B = 2(b^2 - a^2)\cos(\theta)\sin(\theta) \tag{31}$$

$$C = b^2 + (a^2 - b^2)\cos(\theta)^2 \quad (32)$$

$$D = -2a^2x + 2y(a^2 - b^2)\sin(\theta)\cos(\theta) + 2x(a^2 - b^2)\cos(\theta)^2 \quad (33)$$

$$E = -2b^2y + 2x(a^2 - b^2)\sin(\theta)\cos(\theta) - 2y(a^2 - b^2)\cos(\theta)^2 \quad (34)$$

and,

$$F = -a^2b^2 + b^2y^2 + a^2x^2 + (-a^2x^2 + a^2y^2 + b^2x^2 - b^2y^2)\cos(\theta)^2 + (-2a^2xy + 2b^2xy)\cos\theta\sin\theta \quad (35)$$

8.2 Center and Normal of the Circle

Using the coefficients of the ellipse equation, the 3D-position vector and the normal vector of the circle that has the detected ellipse as its projection in the image plane can be found using the method present in [69]. The expressions for these vectors are obtained using the procedure described below. Let the quadratic equation of an ellipse in the camera focal plane ($u'v'$ - plane) be:

$$Au'^2 + Bu'v' + Cv'^2 + Du' + Ev' + F = 0 \quad (36)$$

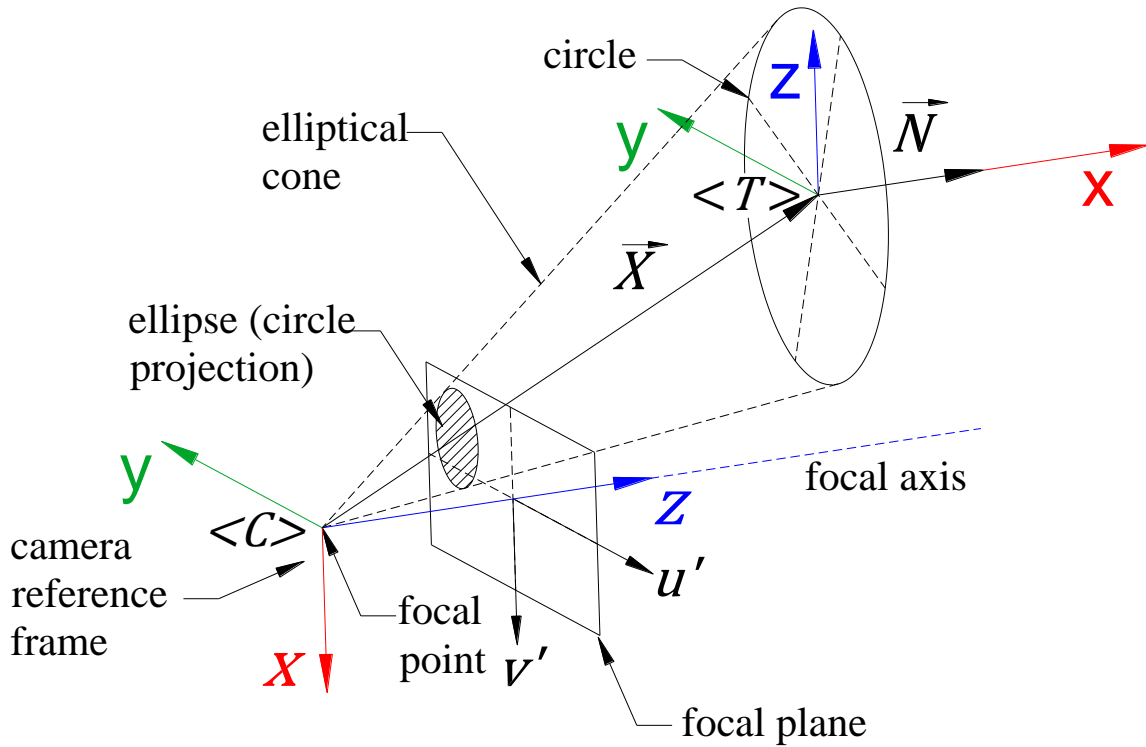


Figure 23. Camera pin-hole model and ring projection.

where (u', v') are the coordinate of the ellipse with respect to the intersection of the focal axis with the image plane, as presented in Figure 23. The equation of the cone with its vertex at the focal point and that passes through the ellipse in the image plane is,

$$A_c x^2 + B_c xy + C_c y^2 + D_c xz + E_c yz + F_c z^2 = 0 \quad (37)$$

At the focal plane, $z = f_c$, thus

$$A_c x^2 + B_c xy + C_c y^2 + D_c f_c x + E_c f_c y + F_c f_c^2 = 0 \quad (38)$$

then the relationship between the cone equation and the ellipse equation can be found,

$$A = f_c^2 a, B = f_c^2 b, C = f_c^2 c, D = f_c d, E = f_c e, F = f \quad (39)$$

The cone equation can be written as

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}^T Q \begin{bmatrix} x \\ y \\ z \end{bmatrix} = 0 \quad (40)$$

where,

$$Q = \begin{bmatrix} A & \frac{B}{2} & \frac{D}{2} \\ \frac{B}{2} & C & \frac{E}{2} \\ \frac{D}{2} & \frac{E}{2} & F \end{bmatrix} \quad (41)$$

To represent the cone equation in a standard canonical form a diagonalizing matrix P for Q is used,

$$P^{-1}QP = D = \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix} \quad (42)$$

then,

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix}^T P^{-1}QP \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = 0 \quad (43)$$

where x', y' and z' represent of the cone in the new frame where the cone has the standard form (see Figure 24),

$$\lambda_1 x'^2 + \lambda_2 y'^2 + \lambda_3 z'^2 = 0 \quad (44)$$

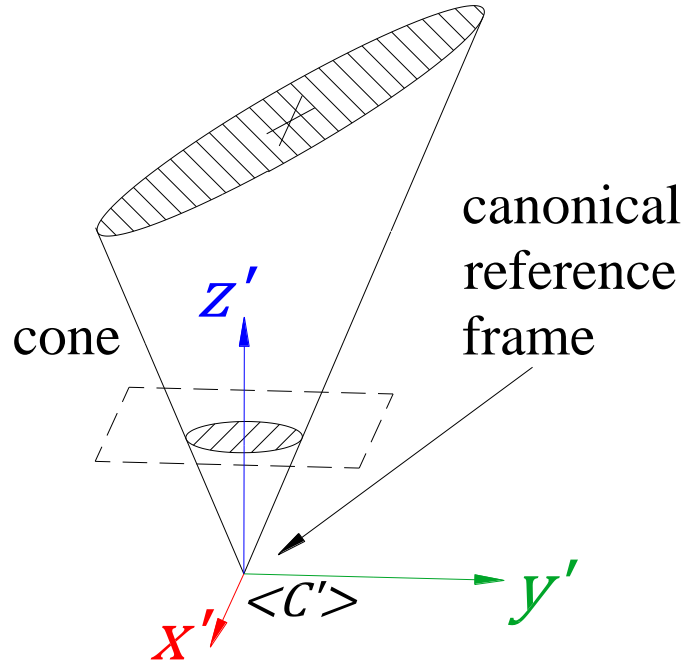


Figure 24. Cone and canonical reference frame

Assuming a cone aligned with the z-axis,

$$|\lambda_1|x'^2 + |\lambda_2|y'^2 - |\lambda_3|z'^2 = 0 \quad (45)$$

The following is the procedure to find P propose in [69]. Let $\vec{e}_1, \vec{e}_2, \vec{e}_3$ be the column vectors of P ,

$$P = [\vec{e}_1, \vec{e}_2, \vec{e}_3] \quad (46)$$

The matrix P can be obtained using the eigenvalues (μ_1, μ_2, μ_3) of the matrix Q and its normalized eigenvectors $(\vec{f}_1, \vec{f}_2, \vec{f}_3)$. The difference between $\lambda_1, \lambda_2, \lambda_3$ and μ_1, μ_2, μ_3 is the order, the difference between the columns of the diagonalizing matrix and the eigenvectors of Q is also the order and the direction by a factor of ± 1 . To match the eigenvalues the following conditions are used.

Since the cone opens along the z-axis then λ_1 and λ_2 must have the same sign, and it is different from the sign of λ_3 , then,

$$\lambda_3 = \mu_d \quad (47)$$

where,

$$\mu_d = \begin{cases} \mu_1 & \text{if } \text{sign}(\mu_2) = \text{sign}(\mu_3) \\ \mu_2 & \text{if } \text{sign}(\mu_1) = \text{sign}(\mu_3) \\ \mu_3 & \text{if } \text{sign}(\mu_1) = \text{sign}(\mu_2) \end{cases} \quad (48)$$

and the direction of the eigenvector \vec{e}_3 is along the positive direction of the z-axis, then

$$\vec{e}_3 = \begin{cases} \vec{f}_d & \text{if } \vec{f}_d \cdot [0 \ 0 \ 1]^T > 0 \\ -\vec{f}_d & \text{if } \vec{f}_d \cdot [0 \ 0 \ 1]^T < 0 \end{cases} \quad (49)$$

To find the two remaining eigenvalues (denoted as ω_1 and ω_2 with the corresponding eigenvectors \vec{g}_1 and \vec{g}_2 respectively) reference [69] assumed that,

$$|\lambda_1| > |\lambda_2| \quad (50)$$

then,

$$\lambda_2 = \begin{cases} \omega_1 & \text{if } |\omega_1| < |\omega_2| \\ \omega_2 & \text{if } |\omega_2| < |\omega_1| \end{cases} \quad (51)$$

and,

$$\vec{e}_2 = \begin{cases} \vec{g}_1 & \text{if } |\omega_1| < |\omega_2| \\ \vec{g}_2 & \text{if } |\omega_2| < |\omega_1| \end{cases} \quad (52)$$

The third eigenvector is computed as the cross product of the two identified eigenvectors,

$$\vec{e}_1 = \vec{e}_2 \times \vec{e}_3 \quad (53)$$

Using the obtained eigenvectors the matrix Q can be normalized and the cone can be expressed in the canonical form, and rest of the four steps described above can be performed. The two solutions of the circle center $(x_{c,1}, y_{c,1}, z_{c,1})$ with respect to the camera frame can be obtained (see [69] for details) and they are presented below. The first solution for the circle center is,

$$x_{c,1} = e_{1x}r \sqrt{\frac{|\lambda_3|(|\lambda_1| - |\lambda_2|)}{|\lambda_1|(|\lambda_1| + |\lambda_3|)}} + e_{3x}r \sqrt{\frac{|\lambda_1|(|\lambda_2| + |\lambda_3|)}{|\lambda_3|(|\lambda_1| + |\lambda_3|)}} \quad (54)$$

$$y_{c,1} = e_{1y}r \sqrt{\frac{|\lambda_3|(|\lambda_1| - |\lambda_2|)}{|\lambda_1|(|\lambda_1| + |\lambda_3|)}} + e_{3y}r \sqrt{\frac{|\lambda_1|(|\lambda_2| + |\lambda_3|)}{|\lambda_3|(|\lambda_1| + |\lambda_3|)}} \quad (55)$$

$$z_{c,1} = e_{1z}r \sqrt{\frac{|\lambda_3|(|\lambda_1| - |\lambda_2|)}{|\lambda_1|(|\lambda_1| + |\lambda_3|)}} + e_{3z}r \sqrt{\frac{|\lambda_1|(|\lambda_2| + |\lambda_3|)}{|\lambda_3|(|\lambda_1| + |\lambda_3|)}} \quad (56)$$

The first solution for the circle normal $(n_{x,1}, n_{y,1}, n_{z,1})$ is,

$$n_{x,1} = e_{1x} \sqrt{\frac{|\lambda_1| - |\lambda_2|}{|\lambda_1| + |\lambda_3|}} - e_{3x} \sqrt{\frac{|\lambda_2| + |\lambda_3|}{|\lambda_1| + |\lambda_3|}} \quad (57)$$

$$n_{y,1} = e_{1y} \sqrt{\frac{|\lambda_1| - |\lambda_2|}{|\lambda_1| + |\lambda_3|}} - e_{3y} \sqrt{\frac{|\lambda_2| + |\lambda_3|}{|\lambda_1| + |\lambda_3|}} \quad (58)$$

$$n_{z,1} = e_{1z} \sqrt{\frac{|\lambda_1| - |\lambda_2|}{|\lambda_1| + |\lambda_3|}} - e_{3z} \sqrt{\frac{|\lambda_2| + |\lambda_3|}{|\lambda_1| + |\lambda_3|}} \quad (59)$$

The second solution for the circle center $(x_{C,2}, y_{C,2}, z_{C,2})$ is,

$$x_{C,2} = -e_{1x} r \sqrt{\frac{|\lambda_3|(|\lambda_1| - |\lambda_2|)}{|\lambda_1|(|\lambda_1| + |\lambda_3|)}} + e_{3x} r \sqrt{\frac{|\lambda_1|(|\lambda_2| + |\lambda_3|)}{|\lambda_3|(|\lambda_1| + |\lambda_3|)}} \quad (60)$$

$$y_{C,1} = -e_{1y} r \sqrt{\frac{|\lambda_3|(|\lambda_1| - |\lambda_2|)}{|\lambda_1|(|\lambda_1| + |\lambda_3|)}} + e_{3y} r \sqrt{\frac{|\lambda_1|(|\lambda_2| + |\lambda_3|)}{|\lambda_3|(|\lambda_1| + |\lambda_3|)}} \quad (61)$$

$$z_{C,1} = -e_{1z} r \sqrt{\frac{|\lambda_3|(|\lambda_1| - |\lambda_2|)}{|\lambda_1|(|\lambda_1| + |\lambda_3|)}} + e_{3z} r \sqrt{\frac{|\lambda_1|(|\lambda_2| + |\lambda_3|)}{|\lambda_3|(|\lambda_1| + |\lambda_3|)}} \quad (62)$$

The second solution for the circle normal $(n_{x,2}, n_{y,2}, n_{z,2})$ is,

$$n_{x,2} = -e_{1x} \sqrt{\frac{|\lambda_1| - |\lambda_2|}{|\lambda_1| + |\lambda_3|}} - e_{3x} \sqrt{\frac{|\lambda_2| + |\lambda_3|}{|\lambda_1| + |\lambda_3|}} \quad (63)$$

$$n_{y,2} = -e_{1y} \sqrt{\frac{|\lambda_1| - |\lambda_2|}{|\lambda_1| + |\lambda_3|}} - e_{3y} \sqrt{\frac{|\lambda_2| + |\lambda_3|}{|\lambda_1| + |\lambda_3|}} \quad (64)$$

$$n_{z,2} = -e_{1z} \sqrt{\frac{|\lambda_1| - |\lambda_2|}{|\lambda_1| + |\lambda_3|}} - e_{3z} \sqrt{\frac{|\lambda_2| + |\lambda_3|}{|\lambda_1| + |\lambda_3|}} \quad (65)$$

where,

$$\vec{e}_1 = \begin{bmatrix} e_{1x} \\ e_{1y} \\ e_{1z} \end{bmatrix} \quad (66)$$

$$\vec{e}_2 = \begin{bmatrix} e_{2x} \\ e_{2y} \\ e_{2z} \end{bmatrix} \quad (67)$$

$$\vec{e}_3 = \begin{bmatrix} e_{3x} \\ e_{3y} \\ e_{3z} \end{bmatrix} \quad (68)$$

When the circle is parallel to the image plane the solution are the same and $\lambda_1 = \lambda_2$.

9 POSE SOLVER

The function of the pose solver is to fuse information from the two cameras and from the different detected features that are tracked. This is achieved by minimizing a cost function that represent the distance between the detected features and some reference features corresponding to the projection of the target features when the target is located w.r.t the end-effector according to an assumed transformation matrix. The assumed position that minimizes the cost function has to be determined and is the output of the pose solver system. As can be seen in Figure 25, the assumed transformation matrix ($T_{T/E,k+1}$) is a small variation (ΔT_k) of the EKF estimated transformation matrix at the previous time ($T_{T/E,k}$).

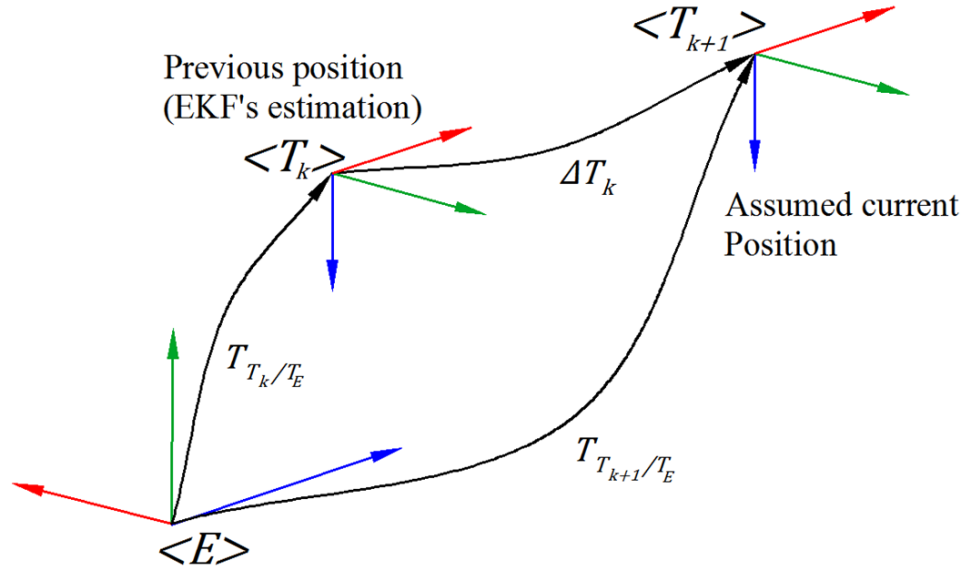


Figure 25. Matrices relating the assumed pose and the estimated pose

The variation matrix ΔT_k , can be expressed in term of the Euler angles and a translation vector as follows,

$$\Delta T_k = \begin{bmatrix} \cos\psi\Delta\cos\Delta\theta & -\sin\Delta\psi\cos\Delta\varphi + \cos\Delta\psi\sin\Delta\theta\sin\Delta\varphi & \sin\Delta\psi\sin\Delta\varphi + \cos\Delta\psi\sin\Delta\theta\cos\Delta\varphi & \Delta x \\ \sin\Delta\psi\cos\Delta\theta & \cos\Delta\psi\cos\Delta\varphi + \sin\Delta\psi\sin\Delta\theta\sin\Delta\varphi & -\sin\Delta\varphi\cos\Delta\psi + \sin\Delta\psi\sin\Delta\theta\cos\Delta\varphi & \Delta y \\ -\sin\Delta\theta & \cos\Delta\theta \sin\Delta\varphi & \cos\Delta\theta\cos\Delta\varphi & \Delta z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (69)$$

where $\Delta\varphi$, $\Delta\psi$ and $\Delta\theta$ are the Euler angles that represent the rotation component of the variation matrix and Δx , Δy and Δz the components of the translation vector. A vector containing these six variables is known as pose solver vector, X_{PS} , i.e.,

$$X_{PS} = [\Delta\varphi \quad \Delta\theta \quad \Delta\psi \quad \Delta x \quad \Delta y \quad \Delta z] \quad (70)$$

The definition of the variation matrix using pose solver vector is implemented in order to facilitate the process of obtaining the gradient required to perform the minimization process. Then the minimization process will find a pose vector that will minimize the cost function. Having the vector, it is possible to obtain the variation matrix using Equation (70).

The pose vector represents the change in the target position with respect to the previous target position, this change can be due to actual motion between the target and the end-effect or due to noise/errors in the estimation process or both. It is assumed that the current unknown pose is close to the previous pose, so the values of the pose vector are small.

The reference features are compared with the detected features to find the cost corresponding to the particular assumed transformation matrix. All the inputs of the pose solver are 2D-points in the image plane representing the detected features of each camera. As an example if Camera-1 is tracking a circle and Camera-2 is tracking two lines a one corner, the inputs of the pose solver system are:

- A set of points belonging to an ellipse (circle of Camera-1)
- Two sets of points, each set with two points (2 lines of Camera-2)
- One point (Corner of Camera-2)

Then the cost function is expressed in terms of these points, which are constant during the iterative process of the minimization routine. They only change when a new image is acquired or equivalently at each execution step of the pose estimation system.

The variation matrix ΔT_k^* , that minimizes the cost function is obtained using the conjugated gradient method, which requires the computation of the gradient of the cost function. This requires that the cost function has to be expressed in terms of the modified transformation matrix and obtain its gradient, the next sections show how this is achieved.

The relation between the matrices of Figure 25 can be expressed as,

$$T_{T/E,k+1} = \Delta T_k T_{T/E,k} \quad (71)$$

this is the assumed position of the target w.r.t. the end-effector. The target can also be expressed with respect to a camera using the following expression,

$$T_{T/C,k+1} = T_{T/C} T_{T/E,k+1} = T_{T/C} \Delta T_k T_{T/E,k} \quad (72)$$

The previous expression is needed since the cost function uses the features projected in the image plane of the cameras and the properties of the features are function of this matrix.

The output of the pose solver system is the quaternion, $[q_0 \ q_1 \ q_2 \ q_3]^*$, and the translation vector, $[x \ y \ z]^*$, corresponding to the transformation matrix $T_{T/E,k+1}^*$, that minimizes the cost function, which is obtained as,

$$T_{T/E,k+1}^* = \Delta T_k^* T_{T/E,k} \quad (73)$$

then,

$$X_{out} = [q_0 \ q_1 \ q_2 \ q_3 \ x \ y \ z]^T \quad (74)$$

is the output of the pose solver system. The next section shows the cost function for each type of feature (ellipses, lines and points) and the definition of each cost function in terms of the pose vector.

9.1 Ellipse Cost Function

The cost function for ellipses is defined in the same way as the cost function of the ellipse-point detection,

$$J_{i,E} = \frac{(Au^2 + Buv + Cv^2 + Du + Ev + F)^2}{0.25(f_x + f_y)^2 AB} \quad (75)$$

where (u, v) are the coordinate of the detected points, A, B, C, D, E and F are the coefficients of the ellipse equation, f_x and f_y are the camera intrinsic parameters corresponding to the focal distance. The detected ellipse points are constant during the minimization processes. The coefficients of the ellipse equation are obtained as the projection of the target circle in the camera image plane using the transformation matrix $T_{T/C,k+1}$, and then these coefficients change during each iteration of the minimization since they are function of the pose vector, as an example,

$$A = A(\Delta\varphi, \Delta\theta, \Delta\psi, \Delta x, \Delta y, \Delta z) \quad (76)$$

This can be obtained using expressions (15), (72) and (69). This also implies that the cost function can be expressed in term of the pose vector,

$$J_{i,E} = J_{i,E}(\Delta\varphi, \Delta\theta, \Delta\psi, \Delta x, \Delta y, \Delta z)$$

9.2 Line Cost Function

The cost function for lines is defined in the same way as the cost function of the line detection,

$$J_{i,L} = \left(\frac{A_L u_{i1} + B_L v_{i1} + C_L}{A_L + B_L} \right)^2 + \left(\frac{A_L u_{i2} + B_L v_{i2} + C_L}{A_L + B_L} \right)^2 \quad (77)$$

where u_{i1}, v_{i1}, u_{i2} and v_{i2} are the coordinate of the two points defining the detected line, A_L, B_L , and C_L are the coefficients of the equation of the projected line, obtained using the modified matrix $T_{T/C,k+1}$. As the ellipses cost function this function is also function of the pose vector. This can be seen from equations (24), (72) and (69).

9.3 Point Cost Function

The cost function for points is defined in the same way as the cost function of the point detection,

$$J_{i,P} = (u_p - u_i)^2 + (v_p - v_i)^2 \quad (78)$$

where u_i and v_i are the coordinates of the detected points, u_p and v_p are the coordinates of the projected points, obtained using the modified matrix $T_{T/C,k+1}$. As the line cost function this function is also function of the pose vector. This can be seen from equations (28), (72) and (69).

All the previous cost functions were define in the same way as it was done in the detection system and they are related to the distance between the detected features (expressed as points) and the projected target features in the image plane, where the projection is performed using the pin-hole model and the assumed transformation matrix. A total cost function is used, it represents the sum of the individual cost functions. As can be noted from the previous equations all the individual cost functions are function of the variation matrix or the pose vector.

9.4 Total Cost Function

The total cost function is the weighted sum of the total number of features. The weights are defined for each type of features of each camera, and since it can be possible to have several features of the same type for one camera, then the weight will affect them equally. The expression for the total cost function is,

$$J = w_{E,1}J_{E,1} + w_{L,1}J_{L,1} + w_{P,1}J_{P,1} + w_{E,2}J_{E,2} + w_{L,2}J_{L,2} + w_{P,2}J_{P,2} \quad (79)$$

where $w_{E,1}, w_{L,1}$ and $w_{P,1}$ are the weight for the cost function of the ellipses, lines and points in camera-1 respectively. The corresponding values for camera-2 are for $w_{E,2}, w_{L,2}$ and $w_{P,2}$. Assuming that there are $N_{E,1}$ ellipse, $N_{L,1}$ lines and $N_{P,1}$ ppoints for camera-1 and being $N_{E,2}, N_{L,2}, N_{P,2}$ the corresponding number of features for camera-2, the individual cost functions are defined as,

$$J_{E,1} = \frac{1}{N_{E,1}} \sum_{i=1}^{N_{E,1}} J_{i,E} \quad (80)$$

this is the total cost function for the ellipses in Camera-1. The term

$$J_{L,1} = \frac{1}{N_{L,1}} \sum_{i=1}^{N_{L,1}} J_{i,L} \quad (81)$$

is the total cost function for the lines in camera-1, and,

$$J_{P,1} = \frac{1}{N_{P,1}} \sum_{i=1}^{N_{P,1}} J_{i,P} \quad (82)$$

is the total cost function for the points in camera-1, $J_{E,2}$, $J_{L,2}$ and $J_{P,2}$ are defined in a similar way.

Equation (79) can be expressed explicitly in terms of the components of the pose vector X_{PS} and then its gradient can be found, as required by the conjugate gradient method, which is used to find the pose that minimizes the total cost function. The weight of the cost function are useful to give more importance to a particular feature, as example if a feature is difficult to detect or is occluded at certain condition a low weight can be used.

As was mentioned before all the individual functions are function of the pose vector, then the minimization process of the total cost function will serve as data fusion system of all the information provided by each camera and each algorithm. The data/sensor occurs since all the values of the cost function during the minimization process are considered together and simultaneously.

9.5 Conjugate Gradients Method

The cost function described above is a nonlinear function of the pose vector, $X_{PS} = [\varphi \ \theta \ \psi \ x \ y \ z]$, its gradient $\bar{\nabla}J$ is,

$$\bar{\nabla}J = \left[\frac{\partial J}{\partial \varphi} \ \frac{\partial J}{\partial \theta} \ \frac{\partial J}{\partial \psi} \ \frac{\partial J}{\partial x} \ \frac{\partial J}{\partial y} \ \frac{\partial J}{\partial z} \right]^T \quad (83)$$

and its hessian $H(J)$ is,

$$H(J) = \begin{bmatrix} \frac{\partial^2 J}{\partial \varphi^2} & \frac{\partial^2 J}{\partial \varphi \partial \theta} & \frac{\partial^2 J}{\partial \varphi \partial \psi} & \frac{\partial^2 J}{\partial \varphi \partial x} & \frac{\partial^2 J}{\partial \varphi \partial y} & \frac{\partial^2 J}{\partial \varphi \partial z} \\ \frac{\partial^2 J}{\partial \varphi \partial \theta} & \frac{\partial^2 J}{\partial \theta^2} & \frac{\partial^2 J}{\partial \theta \partial \psi} & \frac{\partial^2 J}{\partial \theta \partial x} & \frac{\partial^2 J}{\partial \theta \partial y} & \frac{\partial^2 J}{\partial \theta \partial z} \\ \frac{\partial^2 J}{\partial \varphi \partial \psi} & \frac{\partial^2 J}{\partial \theta \partial \psi} & \frac{\partial^2 J}{\partial \psi^2} & \frac{\partial^2 J}{\partial \psi \partial x} & \frac{\partial^2 J}{\partial \psi \partial y} & \frac{\partial^2 J}{\partial \psi \partial z} \\ \frac{\partial^2 J}{\partial \varphi \partial x} & \frac{\partial^2 J}{\partial \theta \partial x} & \frac{\partial^2 J}{\partial \psi \partial x} & \frac{\partial^2 J}{\partial x^2} & \frac{\partial^2 J}{\partial x \partial y} & \frac{\partial^2 J}{\partial x \partial z} \\ \frac{\partial^2 J}{\partial \varphi \partial y} & \frac{\partial^2 J}{\partial \theta \partial y} & \frac{\partial^2 J}{\partial \psi \partial y} & \frac{\partial^2 J}{\partial x \partial y} & \frac{\partial^2 J}{\partial y^2} & \frac{\partial^2 J}{\partial y \partial z} \\ \frac{\partial^2 J}{\partial \varphi \partial z} & \frac{\partial^2 J}{\partial \theta \partial z} & \frac{\partial^2 J}{\partial \psi \partial z} & \frac{\partial^2 J}{\partial x \partial z} & \frac{\partial^2 J}{\partial y \partial z} & \frac{\partial^2 J}{\partial z^2} \end{bmatrix} \quad (84)$$

and they can be obtained since the cost function is known. This makes possible the use of the conjugate gradient optimization method. The outline of the nonlinear conjugate gradients algorithm is presented in Table 3, where,

i : is a counter for the number of iteration

r : represent the residual, equal to $-\vec{\nabla}J$

d : is the search direction, constructed by conjugation of the residuals

α : is the step length in the search direction and is computed using the Newton-Raphson to minimize J (line search)

i_{max} : is the maximum allowed number of iteration in the conjugate gradient method.

ε : is the error tolerance of the conjugate gradient method, it terminates when $|r_i| < \varepsilon|r_0|$

j_{max} : is the maximum allowed number of iteration in the Newton Rapshon method.

γ : is the error tolerance of the Newton Rapshon method.

Having the gradient and the hessian the conjugate gradient method can be performed, it does not require computation of the value of the cost function. The next section presents an example of the conjugate gradient method.

Table 3. Outline of the conjugate gradient method

Initialization	Main loop
$i = 0$ $r = -\nabla J$ $d = r$ $\delta_{new} = r^T r$ $\delta_0 = \delta_{new}$	<p><i>Do while</i> ($i < i_{max}$ and $\delta_{new} > \epsilon \delta_0$)</p> $j = 0$ $\delta_d = d^T d$ $\alpha_0 = -\frac{[\nabla J]^T d}{d^T [H(J)] d}$ <p><i>Do</i></p> $\alpha_{j+1} = \alpha_j - \frac{\left. \frac{d}{d\alpha} J(X + \alpha d) \right _{X_j + \alpha_j d}}{\left. \frac{d^2}{d\alpha^2} J(X + \alpha d) \right _{X_j + \alpha_j d}}$ $X_{j+1} = X_j + \alpha d$ $j = j + 1$ <p><i>while</i> $j < j_{max}$ and $\alpha^2 \delta_d > \gamma^2$</p> $r = -\nabla J$ $\delta_{old} = \delta_{new}$ $\delta_{new} = r^T r$ $\beta = \frac{\delta_{new}}{\delta_{old}}$ $d = r + \beta d$ $\delta_0 = \delta_{new}$ $i = i + 1$

9.6 Pose Solver Example

To observe the performance of the pose solver an example with artificial data is presented. It is assumed that the relative target end-effector location is as in Figure 6, camera-1 and camera-2 are tracking two concentric circles (interface ring inner and outer diameters), additionally camera-2 is tracking two lines and one point. The initial relative position of the target w.r.t end effector is:

$$X_{T/E,previous} = [90^\circ \quad -90^\circ \quad 0 \quad 0 \quad 0.6 \quad 0.75] \quad (85)$$

this position corresponds to the previous time estimation in the pose estimation system, for the purposes of this example it is considered as the actual previous position. It is assumed that the target moved -50mm along its x-axis, 50mm along its y-axis, 50mm along its z-axis and rotated 10deg along its z-axis, then the new position is,

$$X_{T/E,current} = [90^\circ \quad -90^\circ \quad 10^\circ \quad -0.05 \quad 0.55 \quad 0.7] \quad (86)$$

this vector corresponds to the current pose, which is unknown to the pose estimation system and the pose solver will estimate it by minimizing the cost function using the nonlinear conjugate gradient method. Figure 26 shows the features as observed by the camera at the previous and current position.

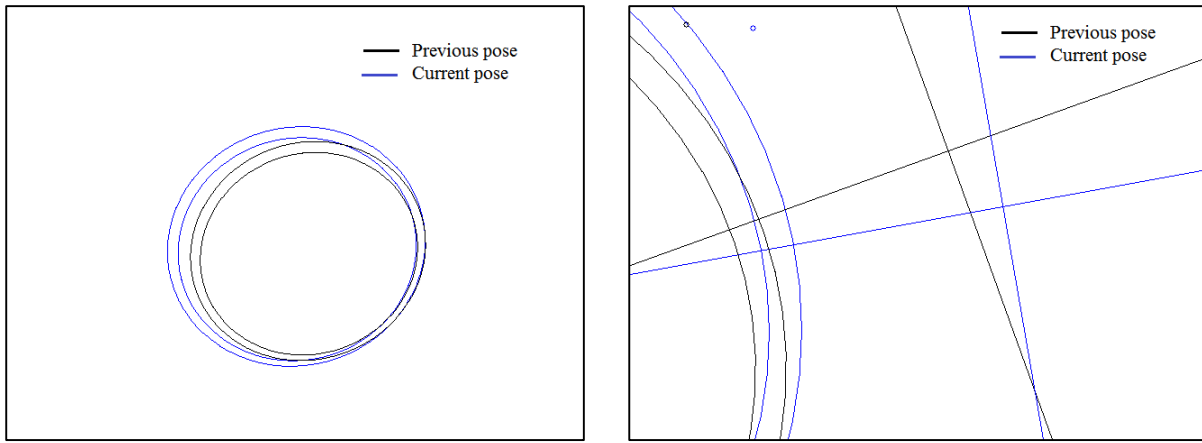


Figure 26. Features in the image plane as seen by the cameras at the previous and current pose. Left: camera-1, right: camera-2. Previous-time view in black, current-time view in blue.

The previous time pose vector, $X_{T/E,previous}$, is used in the initialization of the pose solver in addition with the position of the detected features (ellipses, lines, points) in the image corresponding to the current position, in this example it is assumed that they are the actual features and are detected correctly, i.e. without noise or cases of false detection.

The results for 24 iterations are presented in Figure 27 and Figure 28. Figure 27 shows how the value of the cost function changes with the iterations. Figure 28 show how the pose estimation error (see Chapter 11) changes with the iterations. The case for iteration 0 correspond to the initial position, where the rotation error is equivalent to the assumed motion (10 deg) and the translation error is $\sqrt{50^2 + 50^2 + (-50)^2} = 86.6mm$, as presented in Figure 28. Table 4 shows the projected features in the image plane as seen by the camera after each iteration of the pose solver.

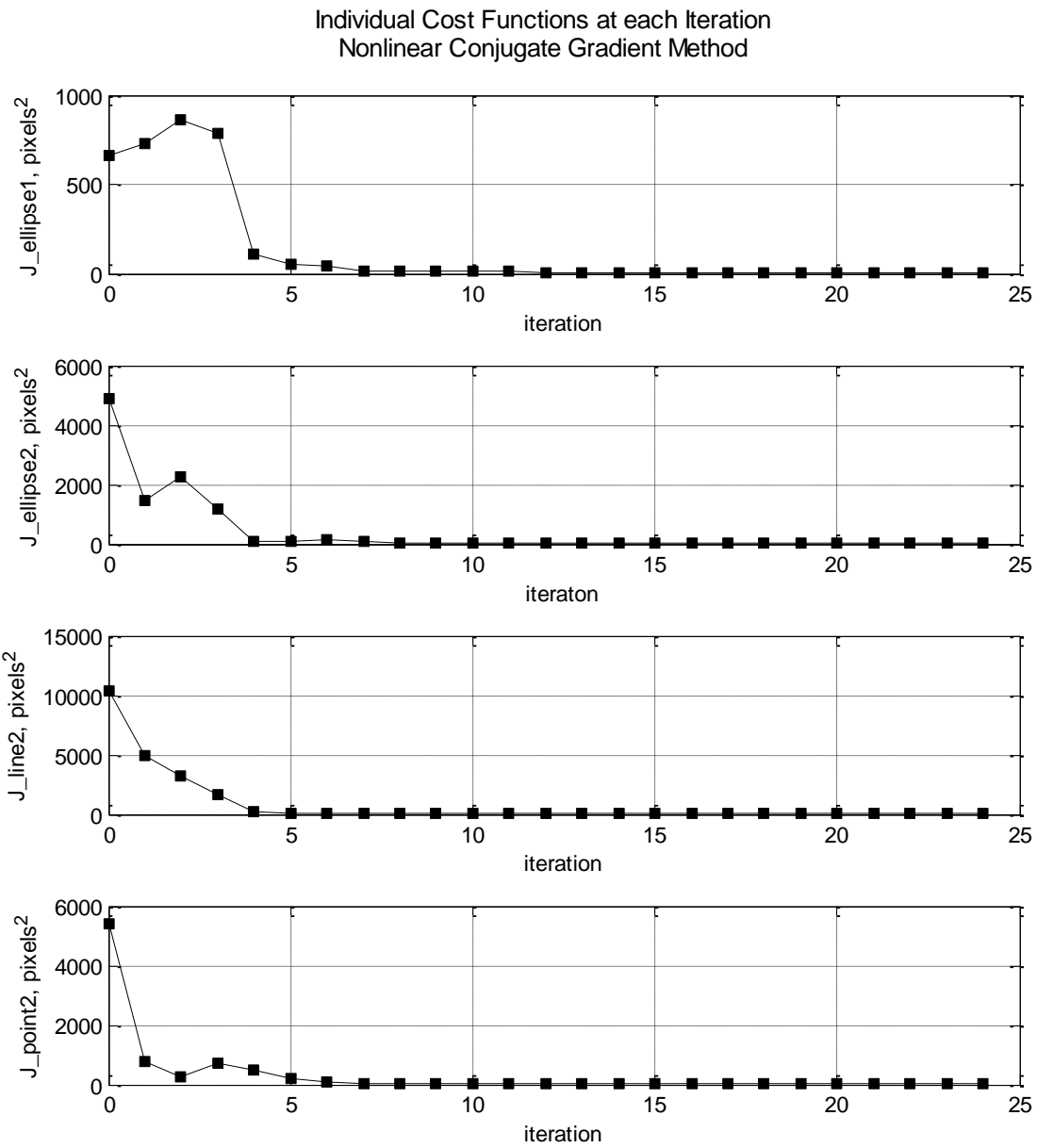


Figure 27. Cost function of each feature as a function of the iteration number

Pose Error at each Iteration
Nonlinear Conjugate Gradient Method

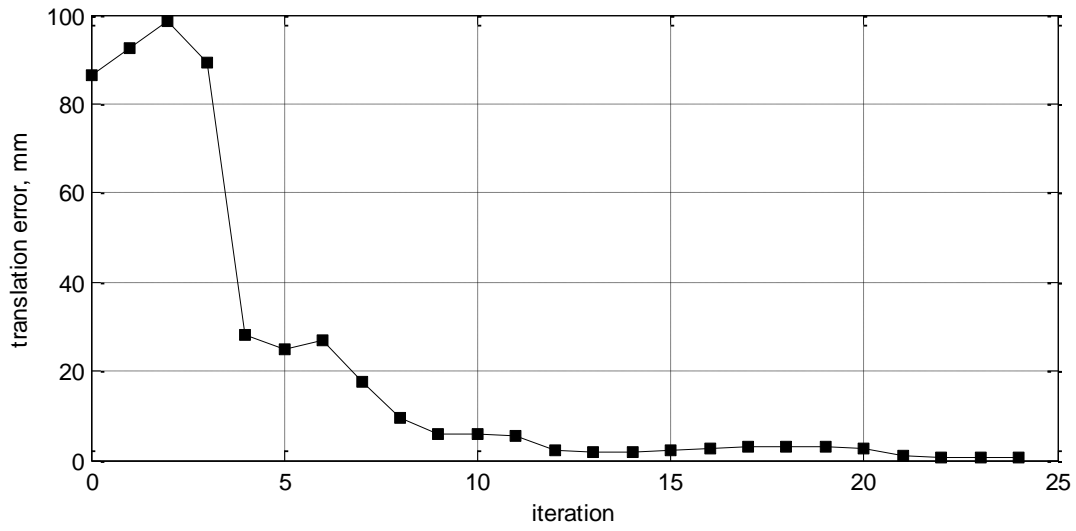
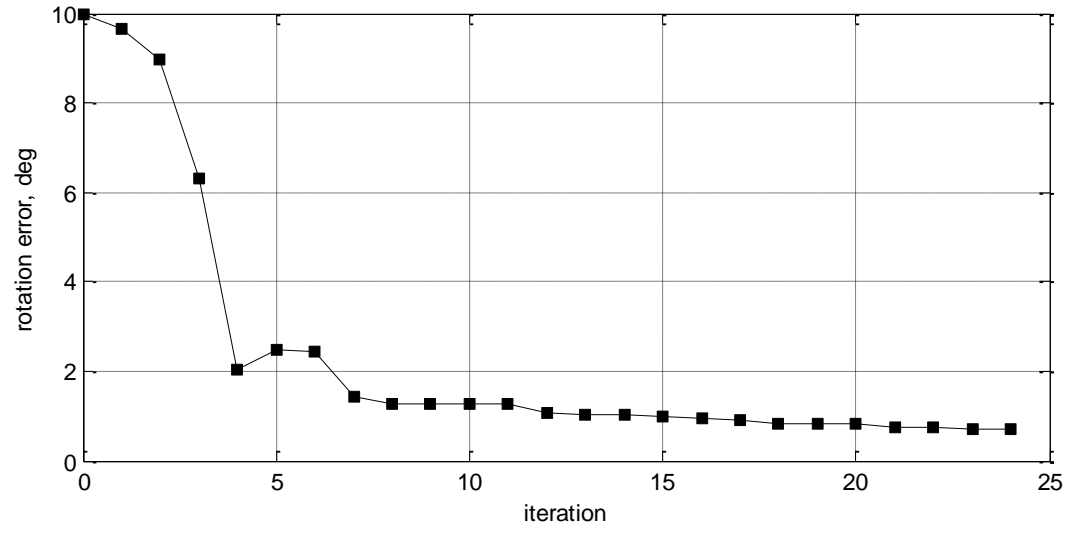
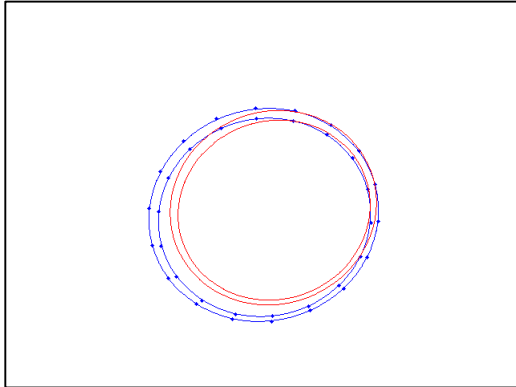
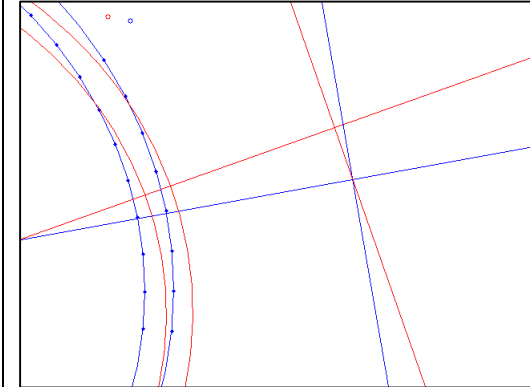
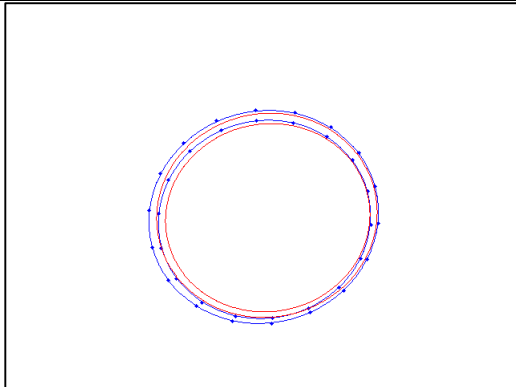
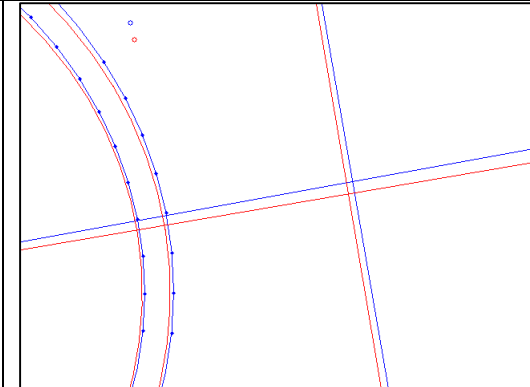
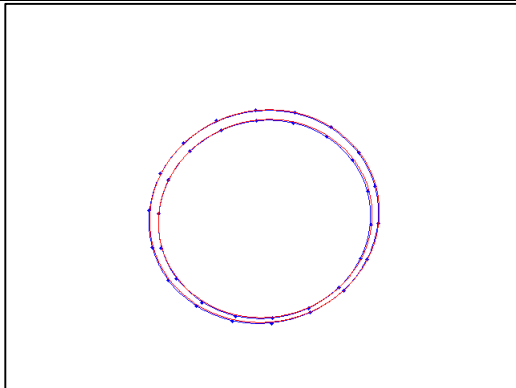
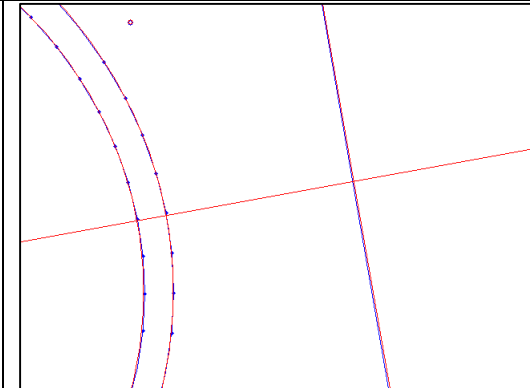


Figure 28. Pose error as a function of the iteration number

Table 4. Projected feature in the image plane for iteration number 1, 4 and 24.

Iteration	Camera 1 view	Camera 2 view
1		
4		
24		

From the previous images it can be observed that after four iterations the post error and the value of the cost function have decreased considerably, although there are some noticeable differences between the detected and the projected features as can be seen in Table 4.

The computer algebra system Maple was used to obtain the optimized C++ code which gives the gradient and hessian of the total cost function.

10 EXTENDED KALMAN FILTER

The extended Kalman filter is used to merge the information from the circle pose calculator and the pose solver systems. A discrete Kalman filter is used. The following sections describe the details of the implementation of the Kalman filter for this particular application.

10.1 Motion Model and State Vector

A motion model between the satellite and the end-effector (and the cameras) is required in order to implement the Kalman filter. In this case a discrete constant velocity model is used and a sampling time of Δt is used. It is assumed that the relative motion of the target w.r.t the end effector is a translation with constant velocity $V_{T/E}$ and a rotation with constant angular velocity $\omega_{T/E}$. The components of these velocities are defined below,

$$V_{T/E,k} = \begin{bmatrix} v_{x,k} \\ v_{y,k} \\ v_{z,k} \end{bmatrix} \quad (87)$$

and,

$${}^T\omega_{T/E,k} = \begin{bmatrix} \omega_{x,k} \\ \omega_{y,k} \\ \omega_{z,k} \end{bmatrix} \quad (88)$$

Note that the angular velocity is expressed in the target frame. The motion model can be expressed as a function f of the state vector X_k and the process noise w_k . The function f relates the change of the state vector between the time $t + \Delta t$ and t , or equivalent between the step $k + 1$ and k , this is expressed as,

$$X_{k+1} = f(X_k, w_k) \quad (89)$$

where,

$$X = [\omega_x \quad \omega_y \quad \omega_z \quad q_0 \quad q_1 \quad q_2 \quad q_3 \quad \dot{x} \quad \dot{y} \quad \dot{z} \quad x \quad y \quad z]^T \quad (90)$$

and, q_0, q_1, q_2, q_3 are the components of the quaternion representing the rotation of the target w.r.t the end-effector and x, y, z are the position of the target origin with respect to the end-effector i.e. the translation vector. Note that the transformation matrix, $T_{T/E}$, of the target wrt the end effector can be expressed in terms of the quaternions and the translation vector, as was expressed in Equation (5).

The process noise w is due to unknown disturbances that differ from the constant velocity model such as motion oscillations, vibrations, or unknown inputs such as applied acceleration in the end-effector. The components of the process noise vector are,

$$w = [w_{\omega_x} \quad w_{\omega_y} \quad w_{\omega_z} \quad w_{q_0} \quad w_{q_1} \quad w_{q_2} \quad w_{q_3} \quad w_{\dot{x}} \quad w_{\dot{y}} \quad w_{\dot{z}} \quad w_x \quad w_y \quad w_z] \quad (91)$$

It is assumed that the process noise is white with zero-mean and covariance Q ,

$$w_k \sim (0, Q) \quad (92)$$

The expression for $f(X_k, w_k)$ can be found using the constant velocity assumption. The following paragraphs show the procedure to obtain f .

The constant velocity model implies that,

$$\begin{cases} v_{x,k+1} = v_{x,k} \\ v_{y,k+1} = v_{y,k} \\ v_{z,k+1} = v_{z,k} \end{cases} \quad (93)$$

and for the angular velocity,

$$\begin{cases} \omega_{x,k+1} = \omega_{x,k} \\ \omega_{y,k+1} = \omega_{y,k} \\ \omega_{z,k+1} = \omega_{z,k} \end{cases} \quad (94)$$

Using the x-component of the translation velocity and the definition of velocity,

$$v_{x,k} = \frac{dx}{dt} \Rightarrow \int_{x_k}^{x_{k+1}} dx = \int_t^{t+\Delta t} v_{x,k} dt \Rightarrow x_{k+1} = v_{x,k} \Delta t + x_k \quad (95)$$

doing the same for the y and z components of the velocity, it is obtained,

$$\begin{cases} x_{k+1} = v_{x,k} \Delta t + x_k \\ y_{k+1} = v_{y,k} \Delta t + y_k \\ z_{k+1} = v_{z,k} \Delta t + z_k \end{cases} \quad (96)$$

To obtain the expression for the quaternion, the concept of the transition quaternion is used. A transition quaternion, \mathbf{p} , can be used to related two different quaternions, which in this case are \mathbf{q}_{k+1} and \mathbf{q}_k ,

$$\mathbf{q}_{k+1} = \mathbf{q}_k \mathbf{p} \quad (97)$$

The component of the transition quaternion are,

$$\mathbf{p} = \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \end{bmatrix} \quad (98)$$

which can be expressed in terms of the rotation angle $\theta/2$ around the unit vector of component (u_x, u_y, u_z) between the two quaternions,

$$\mathbf{p} = \begin{bmatrix} \cos\theta \\ u_x \sin\left(\frac{\theta}{2}\right) \\ u_y \sin\left(\frac{\theta}{2}\right) \\ u_z \sin\left(\frac{\theta}{2}\right) \end{bmatrix} \quad (99)$$

Assuming that the rotation angle $\theta/2$ is equal to $\omega\Delta t/2$, with $\omega = {}_T\omega_{T/E,k}$

$$\mathbf{p} = \begin{bmatrix} \cos\theta \\ u_x \sin\left(\frac{\theta}{2}\right) \\ u_y \sin\left(\frac{\theta}{2}\right) \\ u_z \sin\left(\frac{\theta}{2}\right) \end{bmatrix} = \begin{bmatrix} \cos(\omega\Delta t) \\ u_x \sin\left(\frac{\omega\Delta t}{2}\right) \\ u_y \sin\left(\frac{\omega\Delta t}{2}\right) \\ u_z \sin\left(\frac{\omega\Delta t}{2}\right) \end{bmatrix} \quad (100)$$

and assuming an small rotation between the two quaternions (i.e. $\omega\Delta t$ is small),

$$\mathbf{p} = \begin{bmatrix} 1 \\ u_x \frac{\omega\Delta t}{2} \\ u_y \frac{\omega\Delta t}{2} \\ u_z \frac{\omega\Delta t}{2} \end{bmatrix} = \begin{bmatrix} 1 \\ \omega_x \frac{\Delta t}{2} \\ \omega_y \frac{\Delta t}{2} \\ \omega_z \frac{\Delta t}{2} \end{bmatrix} \quad (101)$$

The transition quaternion can be expressed in the expanded matrix form as,

$$\vec{p} = \begin{bmatrix} 1 & -\omega_x \frac{\Delta t}{2} & -\omega_y \frac{\Delta t}{2} & -\omega_z \frac{\Delta t}{2} \\ \omega_x \frac{\Delta t}{2} & 1 & -\omega_z \frac{\Delta t}{2} & \omega_y \frac{\Delta t}{2} \\ \omega_y \frac{\Delta t}{2} & \omega_z \frac{\Delta t}{2} & 1 & -\omega_x \frac{\Delta t}{2} \\ \omega_z \frac{\Delta t}{2} & -\omega_y \frac{\Delta t}{2} & \omega_x \frac{\Delta t}{2} & 1 \end{bmatrix} \quad (102)$$

And the quaternion \mathbf{q}_k as,

$$\vec{q}_k = \begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & -q_3 & q_2 \\ q_2 & q_3 & q_0 & -q_1 \\ q_3 & -q_2 & q_1 & q_0 \end{bmatrix} \quad (103)$$

Then using equations (101) and (103) in (97) it is obtained,

$$\vec{q}_{k+1} = \begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & -q_3 & q_2 \\ q_2 & q_3 & q_0 & -q_1 \\ q_3 & -q_2 & q_1 & q_0 \end{bmatrix} \begin{bmatrix} 1 \\ \omega_x \frac{\Delta t}{2} \\ \omega_y \frac{\Delta t}{2} \\ \omega_z \frac{\Delta t}{2} \end{bmatrix} \quad (104)$$

performing the multiplication it is obtained,

$$\vec{q}_{k+1} = \begin{bmatrix} q_{0,k+1} \\ q_{1,k+1} \\ q_{2,k+1} \\ q_{3,k+1} \end{bmatrix} = \begin{bmatrix} q_{0,k} + \frac{\Delta t}{2} (-\omega_x q_1 - \omega_y q_2 - \omega_z q_3) \\ q_{1,k} + \frac{\Delta t}{2} (\omega_x q_0 - \omega_y q_3 + \omega_z q_2) \\ q_{2,k} + \frac{\Delta t}{2} (\omega_x q_3 + \omega_y q_0 - \omega_z q_1) \\ q_{3,k} + \frac{\Delta t}{2} (-\omega_x q_2 + \omega_y q_1 + \omega_z q_0) \end{bmatrix} \quad (105)$$

Note that the angular velocities are expressed in the target frame. Now all the components of X_{k+1} in terms of have X_k been found, and they expressed below,

$$\begin{bmatrix} \omega_{x,k+1} \\ \omega_{y,k+1} \\ \omega_{z,k+1} \\ q_{0,k+1} \\ q_{1,k+1} \\ q_{2,k+1} \\ q_{3,k+1} \\ v_{x,k+1} \\ v_{y,k+1} \\ v_{z,k+1} \\ x_{k+1} \\ y_{k+1} \\ z_{k+1} \end{bmatrix} = \begin{bmatrix} \omega_{x,k} \\ \omega_{y,k} \\ \omega_{z,k} \\ q_{0,k} + \frac{\Delta t}{2} (-\omega_x q_1 - \omega_y q_2 - \omega_z q_3) \\ q_{1,k} + \frac{\Delta t}{2} (\omega_x q_0 - \omega_y q_3 + \omega_z q_2) \\ q_{2,k} + \frac{\Delta t}{2} (\omega_x q_3 + \omega_y q_0 - \omega_z q_1) \\ q_{3,k} + \frac{\Delta t}{2} (-\omega_x q_2 + \omega_y q_1 + \omega_z q_0) \\ v_{x,k} \\ v_{y,k} \\ v_{z,k} \\ x_k \\ y_k \\ z_k \end{bmatrix} \quad (106)$$

The previous equation does not include noise, introducing the noise as linear addition to the state vector, the expression for function f is finally obtained,

$$X_{k+1} = f(X_k, w(t)) = \begin{bmatrix} \omega_{x,k} + w_{\omega x} \\ \omega_{y,k} + w_{\omega y} \\ \omega_{z,k} + w_{\omega z} \\ q_{0,k} + \frac{\Delta t}{2}(-\omega_x q_1 - \omega_y q_2 - \omega_z q_3) + w_{q0} \\ q_{1,k} + \frac{\Delta t}{2}(\omega_x q_0 - \omega_y q_3 + \omega_z q_2) + w_{q1} \\ q_{2,k} + \frac{\Delta t}{2}(\omega_x q_3 + \omega_y q_0 - \omega_z q_1) + w_{q2} \\ q_{3,k} + \frac{\Delta t}{2}(-\omega_x q_2 + \omega_y q_1 + \omega_z q_0) + w_{q3} \\ v_{x,k} + w_{vx} \\ v_{y,k} + w_{vy} \\ v_{z,k} + w_{vz} \\ x_k + w_x \\ y_k + w_y \\ z_k + w_z \end{bmatrix} \quad (107)$$

Notice that the quaternions have to be normalized after each time this expression is used since it adds noise to the quaternions.

10.2 Observation Process

The state variables are observable indirectly through the sensors (cameras) and their algorithms. The data given by the pose solver is a full 6DOF pose vector (translation and rotation), while the data given by the circle pose calculator represents 5DOF pose only (circle center and normal). The output vector Y of the Kalman filter is a random vector which include the mentioned measurements, which can be defined in terms of the state vector, this is expresses as:

$$Y_k = h_k(X(t), v_k) \quad (108)$$

where v is the measurement noise vector. The following sections show the equations of the outputs in terms of the state variables, where it is specifically assumed that Camera-1 is tracking the interface ring.

10.2.1 Circle Center

The position of the circle center, P , in the target frame is given by,

$$X_{P/T} = [x_c \quad y_c \quad z_c \quad 1]^T \quad (109)$$

Using the transformation matrix $T_{T/E}$, the position of the same point with respect to the end-effector frame is,

$$X_{P/E} = T_{T/E}X_{P/T} \quad (110)$$

and then the same point can be expressed in the camera-1 frames as,

$$X_{P/C1} = T_{E/C1}X_{P/E} = T_{E/C1}T_{T/E}X_{P/T} \quad (111)$$

where $T_{E/C1}$ is the transformation matrix of the end-effector w.r.t the Camera-1, this is matrix is constant since the cameras are fixed w.r.t the end effector.

10.2.2 Circle Normal

Since the normal vector, N , of the target circle is a free vector, it is expressed in the target frame as,

$$N_{N/E} = [n_x \quad n_y \quad n_z \quad 0]^T \quad (112)$$

Using the transformation matrix $T_{T/E}$, the normal vector can be expressed w.r.t the end-effector frame as,

$$N_{N/E} = T_{T/E}N_{N/T} \quad (113)$$

and then the same vector can be expressed in the camera-1 frames as,

$$N_{N/C1} = T_{E/C1}N_{N/E} = T_{E/C1}T_{T/E}N_{N/T} \quad (114)$$

Since the transformation matrix of the target w.r.t. the end effector can be expressed in terms of the quaternions and the translation vector used in the state vector, then these two measurements (circle center and normal vector) can be expressed in terms of the state vector, specifically in terms of the quaternion and the translation vector.

10.2.3 Pose Vector

The output of the pose solver system is a vector of seven components corresponding to the quaternion $[q_0 \quad q_1 \quad q_2 \quad q_3]^T$ and a translation vector $[x \quad y \quad z]^T$, corresponding to the pose of the target w.r.t the end-effector,

$$X_{PS} = [q_0 \quad q_1 \quad q_2 \quad q_3 \quad x \quad y \quad z] \quad (115)$$

As can be seen it is directly expressed in terms of the state vector components.

In addition to the previous defined measurements the angular and translational velocities are included in the output vector, although they are not directly measured, they are assumed to be zero, this is to avoid that the estimated velocities make the target drift away from the camera view, which affects negatively the performance of the feature detector system.

10.2.4 Output Vector

Collecting all the previous defined outputs, the observation vector, Y , is,

$$Y_k = h_k(X_k, v_k) = \begin{bmatrix} n_x \\ n_y \\ n_z \\ x_C \\ y_C \\ z_C \\ \omega_{x,m} \\ \omega_{y,m} \\ \omega_{z,m} \\ q_{0,m} \\ q_{1,m} \\ q_{2,m} \\ q_{3,m} \\ v_{x,m} \\ v_{x,m} \\ v_{x,m} \\ x_m \\ y_m \\ z_m \end{bmatrix} + v_k = \begin{bmatrix} n_x + v_{nx} \\ n_y + v_{ny} \\ n_z + v_{nz} \\ x_C + v_{xc} \\ y_C + v_{yc} \\ z_C + v_{zc} \\ \omega_{x,m} + v_{\omega x} \\ \omega_{y,m} + v_{\omega y} \\ \omega_{z,m} + v_{\omega z} \\ q_{0,m} + v_{q0} \\ q_{1,m} + v_{q1} \\ q_{2,m} + v_{q2} \\ q_{3,m} + v_{q3} \\ v_{x,m} + v_{vx} \\ v_{x,m} + v_{vy} \\ v_{x,m} + v_{vz} \\ x_m + v_{xm} \\ y_m + v_{ym} \\ z_m + v_{zm} \end{bmatrix} \quad (116)$$

where, v_k is the measurement noise which is assumed to be a discrete time white noise with zero-mean and covariance R_k ,

$$v_k \sim (0, R_k) \quad (117)$$

10.3 Extended Kalman Filter Equations

As mentioned before an extended Kalman filter is used due to the non-linearity of the motion model and the observation process. The discrete time version of the extended Kalman filter is used since it facilitates the software implementation. The following are the Extended Kalman Filter equations [89].

The system equations are defined as,

$$\begin{cases} X_{k+1} = f_k(X_k, w_k) \\ Y_k = h_k(X_k, v_k) \\ w_k \sim (0, Q) \\ v_k \sim (0, R) \end{cases} \quad (118)$$

The system matrices are computed as,

$$\begin{cases} F_k = \frac{\partial f_k}{\partial x} \Big|_{\hat{x}_k^+} \\ L_k = \frac{\partial f_k}{\partial w} \Big|_{\hat{x}_k^+} \\ H_k = \frac{\partial h_k}{\partial x} \Big|_{\hat{x}_k^-} \\ M_k = \frac{\partial h_k}{\partial v} \Big|_{\hat{x}_k^-} \end{cases} \quad (119)$$

The above matrices are evaluated at the current stated estimate. The system state and its covariance matrix are initialized ($k = 0$) using the following expressions,

$$\hat{X}_0^+ = E[X_0] \quad (120)$$

and,

$$P_0^+ = E[(X_0 - \hat{X}_0^+)(X_0 - \hat{X}_0^+)] \quad (121)$$

Since the initial pose is unknown a value is assumed, which is close to the actual value. After initialization the following steps are performed for the subsequent operation of the Kalman filter and the pose estimation system, where $k = 1, 2, \dots$

Step 1: The state estimate and the covariance matrix are integrated from time $(k)^+$ to time $(k+1)^-$ using the following equations:

$$P_{k+1}^- = F_k P_k^+ F_k^T + L_k Q L_k^T \quad (122)$$

and,

$$\hat{X}_{k+1}^- = f_k(\hat{X}_k^+, 0) \quad (123)$$

The initial values for this integration are $\hat{X} = \hat{X}_k^+$ and $\hat{P} = \hat{P}_k^+$ and after the integration the values for $\hat{X} = \hat{X}_{k+1}^-$ and $\hat{P} = \hat{P}_{k+1}^-$ will be obtained.

Step 2: The measurements Y_k are used at time k to improve the estimated values,

$$K_k = P_{k+1}^- H_k^T (H_k P_{k+1}^- H_k^T + M_k R M_k^T)^{-1} \quad (124)$$

$$\hat{X}_k^+ = \hat{X}_{k+1}^- + K_k (Y_k - h_k(\hat{X}_{k+1}^-, 0)) \quad (125)$$

$$P_k^+ = (I - K_k H_k) P_k^- (I - K_k H_k)^T + K_k M R M K_k^T \quad (126)$$

In the previous expression the superscript “-“ indicates the a priori estimate and “+” indicates the a posteriori. The a priori estimate is found using all the measurements before time k, thus \hat{X}_k^- indicates the estimate of X_k before using the measurements at time k, and \hat{X}_k^+ after using it.

11 METHODOLOGY FOR THE EVALUATION OF THE PROPOSED SYSTEM

Virtual simulations and experimental testing were used to evaluate the performance of the pose estimation system. There are two types of test for each case: static and dynamic. In the static tests there is no relative motion between the end effector and the target (satellite). In the dynamic tests different types of motions are generated between the two objects. In each case the relative pose is estimated continuously during the test. The different test are described in this chapter.

One important parameter in the evaluation is the pose estimation error, it represents the difference between the estimated pose and the actual pose given by the ground truth. It has two components, the translation error and the rotation error.

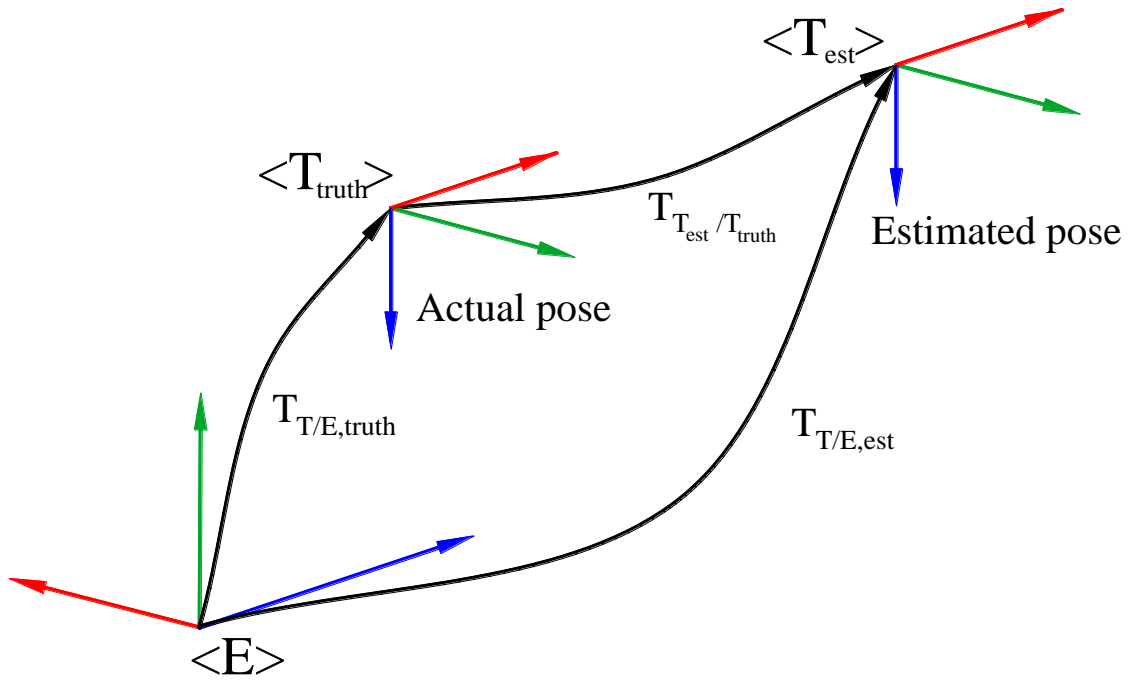


Figure 29. Estimated pose and ground truth

The translation error is defined as,

$$Trans_{err} = \sqrt{(x_{tru} - x_{est})^2 + (y_{tru} - y_{est})^2 + (z_{tru} - z_{est})^2} \quad (127)$$

where x_{est} , y_{est} and z_{est} are the components of the estimated translation vector and x_{tru} , y_{tru} and z_{tru} are the components of the translation vector of the ground truth.

To compute the rotation error, Rot_{err} , for a full 6DOF pose, first the rotation matrix R_{err} is computed

$$R_{err} = R_{tru} R_{est}^{-1} \quad (128)$$

where R_{est} is the estimated rotation matrix, R_{tru} is the rotation matrix of the ground truth. Then, R_{err} is converted to Euler angles (φ_{err} , θ_{err} and ψ_{err}) and the rotation error is,

$$Rot_{err} = \sqrt{\varphi_{err}^2 + \theta_{err}^2 + \psi_{err}^2} \quad (129)$$

In the case of a 5DOF pose (center position of the interface ring and normal vector to ring plane), the rotation error is determined as,

$$Rot_{err} = \text{asin}(\|R_{tru}(1:3,1) \times R_{est}(1:3,1)\|) \quad (130)$$

where $R_{tru}(1:3,1)$ and $R_{est}(1:3,1)$ correspond to the normal vector of the interface ring expressed in the end-effector frame (ground truth and estimated respectively), the 5DOF rotation error is the magnitude of the angle between the ground truth and estimated normal vectors.

In the virtual simulations and experimental tests three different scenarios were used. Each scenario corresponds to a different set of geometric features. The number of features used in each scenario is low, with a maximum of four features, in order to test the system under non-optimal conditions. The use of more features tends to favor the performance of the system, and in real application the number of available features can be limited. The three scenario are defined in the next section.

11.1 Testing Scenarios

The first scenario is presented in Figure 30, where both cameras are tracking the interface ring and only one type of geometric feature is used: ellipses. In this case only 5DOF pose can be obtained.

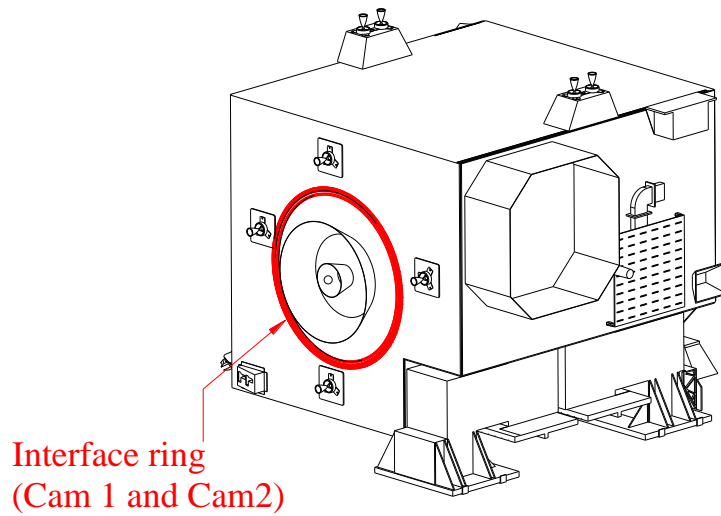


Figure 30. Testing Scenario 1

The second scenario can be seen in Figure 31. In this case camera-1 is tracking the interface ring and camera-2 is tracking one border of the satellite panel, thus only two types of features are used: ellipse and lines. A full 6DOF pose can be obtained.

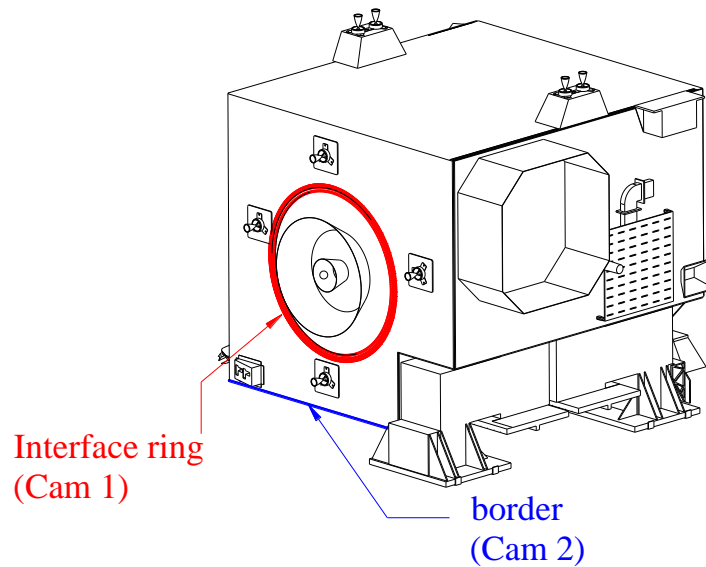


Figure 31. Testing Scenario 2

The third scenario is presented in Figure 32, where camera-1 is tracking the interface ring and camera-2 is tracking two borders of the pad of the AOC thruster and the corner formed by those two borders. In this case all the three types of geometric feature that the system is able to handle are used.

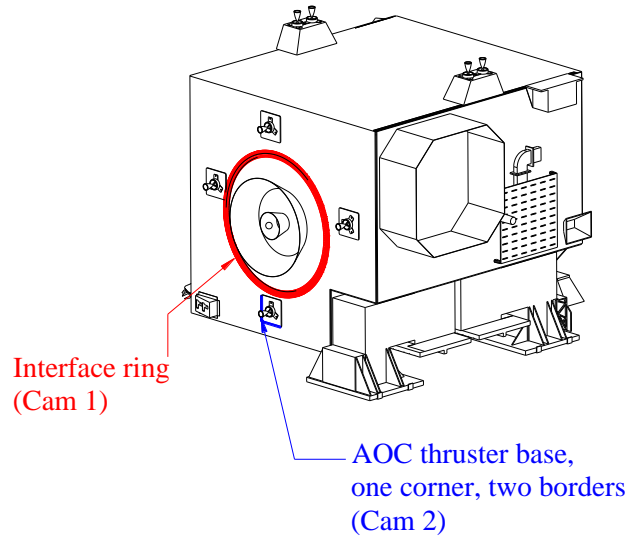


Figure 32. Testing Scenario 3

The next sections describe the virtual simulations and the experimental testing that are used to evaluate the system performance under the scenarios previously described.

Each one of the performed test began with a similar initial position (see Figure 33) of the target w.r.t. the end effector, the main difference was the z -component ($Z_{T/E}$) of the relative position. The results of the tests are reported with its corresponding value of $Z_{T/E}$. The Euler angles and the translation vector that define the initial position w.r.t the end effector are presented in Table 5.

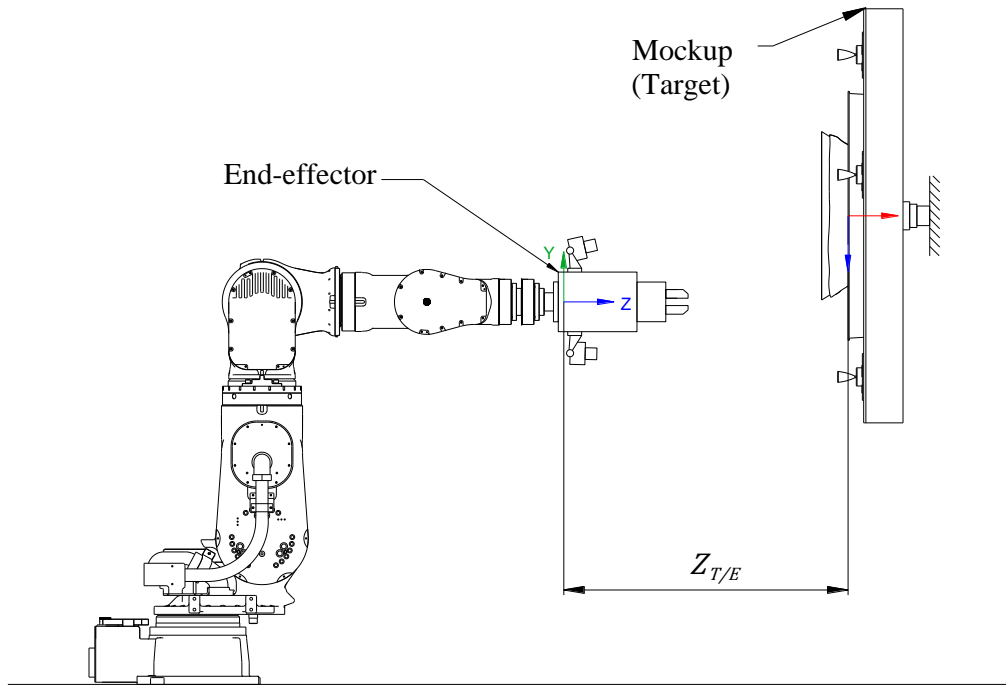


Figure 33. Initial nominal position

Table 5. Initial nominal position

$\varphi_{T/E}$	$\theta_{T/E}$	$\psi_{T/E}$	$X_{T/E}$	$Y_{T/E}$	$Z_{T/E}$
86.5deg	-87.2deg	4.7deg	0.00m	0.47m	variable

11.2 Virtual Testing

The objective of the virtual testing is to find the effect of the system parameters in the pose estimation error under controlled conditions. As an example, the virtual simulator can generate an image without distortion or with a known amount of distortion, which makes possible the evaluation of the effect of the image distortion in the pose error. Additionally, the image noise in the virtual simulation can be eliminated, and the system can be tested under ideal conditions, i.e. in the absence of effects such as unfavorable light conditions, reflections, occlusions, detection of wrong features, etc. The virtual simulator is described in the next section.

11.2.1 Virtual Simulator

The virtual simulator generates synthetic images of the target and its features corresponding to the views of the system cameras, i.e. using the intrinsic and extrinsic parameters of the actual cameras. In the simulator, a virtual model of the satellite and the end effector are used, as can be seen in Figure 34, the same satellite was

used in the experimental test. The cameras are located with respect to the end-effector reproducing the same conditions used in the experimental test and the views are generated using the virtual camera with the same intrinsic parameters as the cameras used in the experimental testing. The pose between the target and the end effector in the virtual simulator can be controlled easily, and different types of motions can be generated, such as rotations along each axis, translations, approaches, precession, etc.

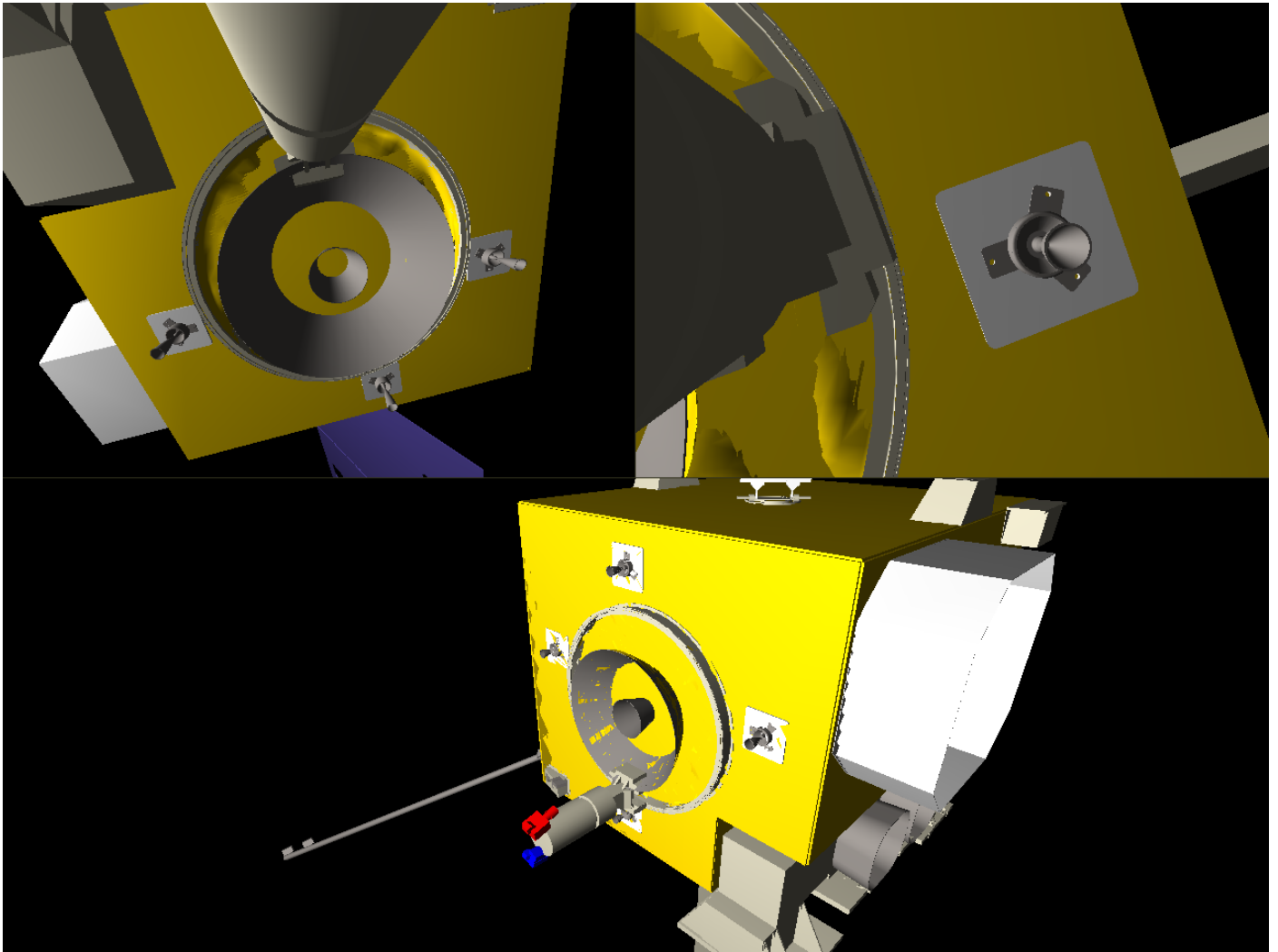
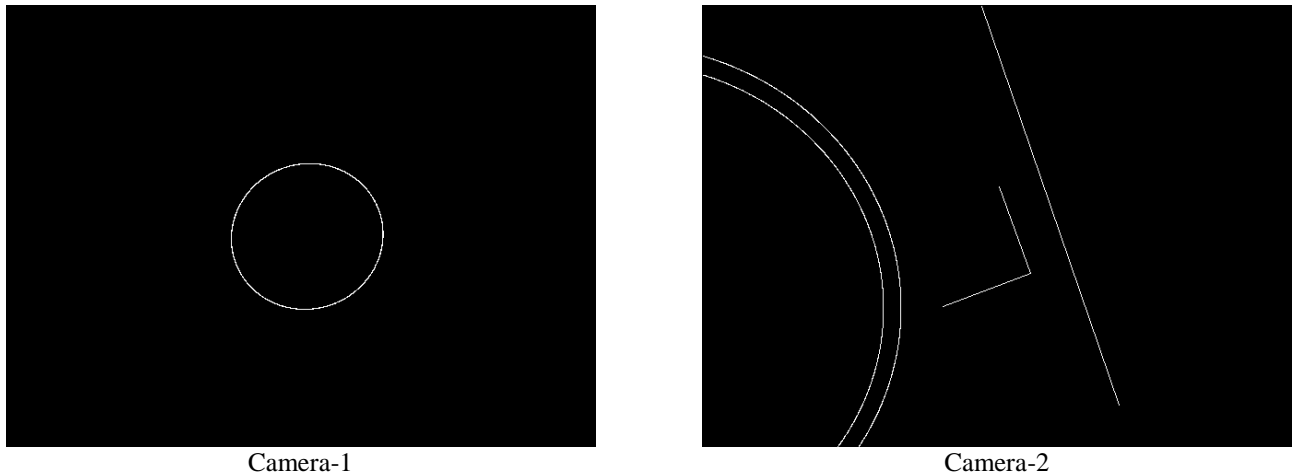


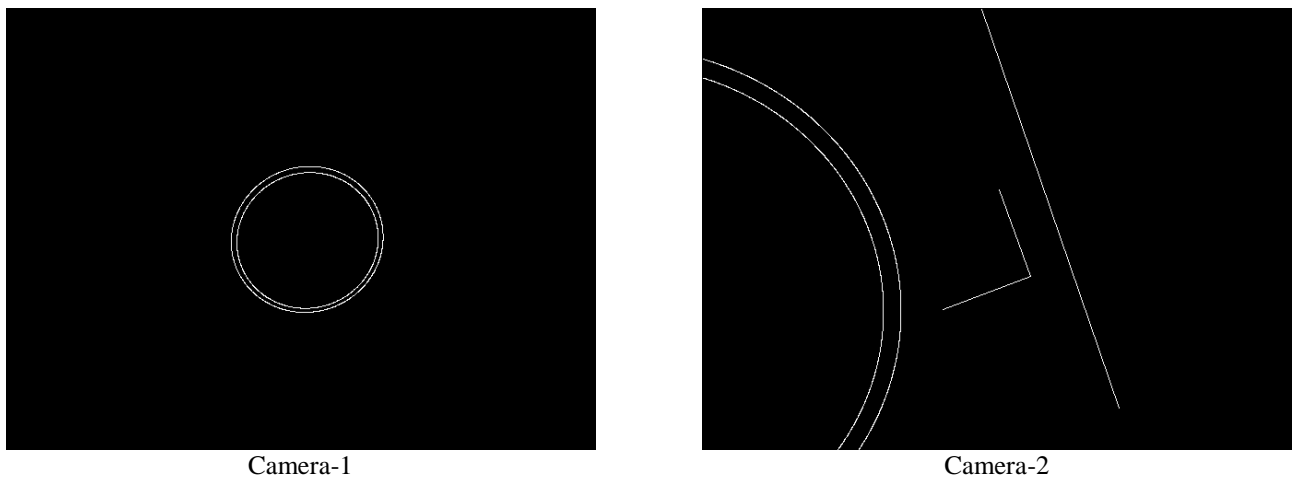
Figure 34. CAD model mode of the virtual simulator. Top left: camera-1 view, top right: camera-2 view, bottom: external view

The generated synthetic images can be used by the pose estimation system to estimate the pose of the target. Since the relative position of the target satellite and the end-effector are known, then they can be used as the ground truth data to compute the pose estimation error.

The virtual simulator can generate two types of images: solid images using a CAD-model (as in Figure 34) and binary images using a wireframe model as in Figure 35. The CAD model of the satellite was provided by NASA's GSFC. Two different wireframe models are used, in wireframe mode-1 (Figure 35) the interface ring is observed in camera-1 as a single circle, using only the outer diameter. In wireframe model-2 (Figure 36) the inner and outer diameters of the interface ring are used in the camera-1 view.

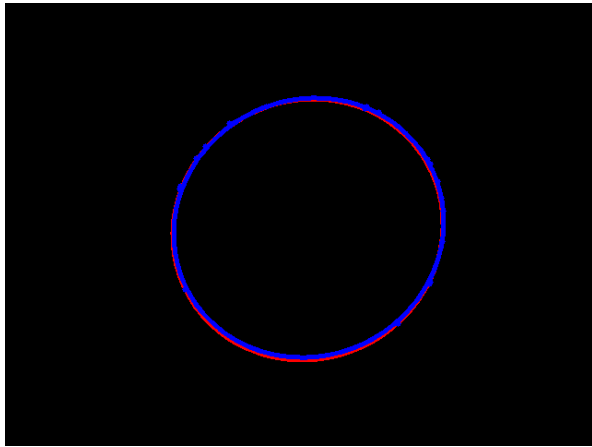


Camera-1 Camera-2
Figure 35. Camera views of the wireframe model 1 of the virtual simulator

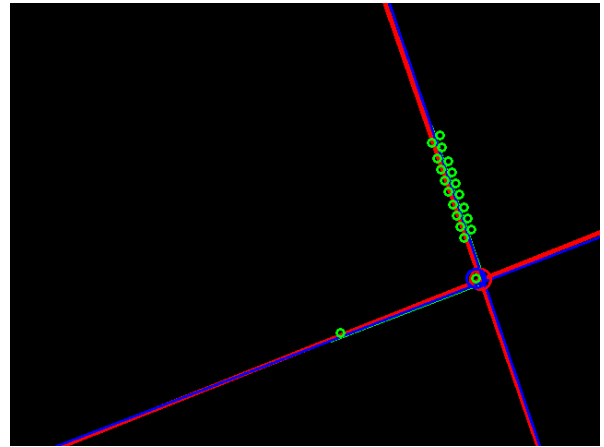


Camera-1 Camera-2
Figure 36. Camera views of the wireframe model 2 of the virtual simulator

Figure 37 show the detected and reference features by the pose estimation system when using the virtual simulator with wireframe model-1. Figure 38 show the edge map (from canny edge detector) of the images generated using the CAD model images. Figure 39 shows the extracted, detected and reference features of the pose estimation system when using the images generated by the virtual simulator when using the CAD model.

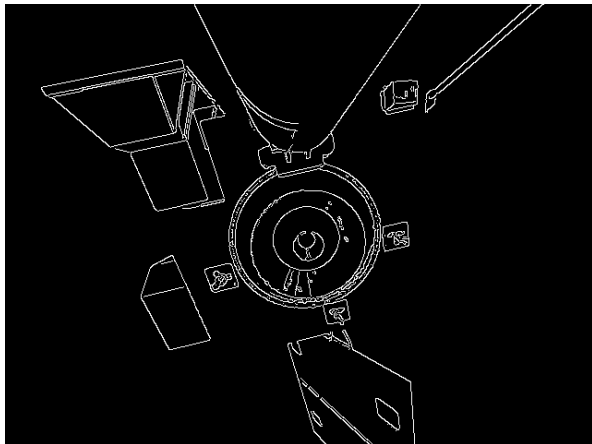


Camera-1

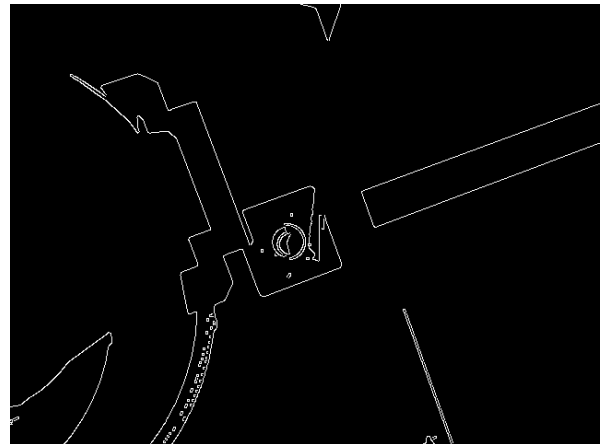


Camera-2

Figure 37. Reference features (red), detected features (blue) and extracted features (green) obtained using the pose estimation system with the images of the virtual simulator in the wireframe model 1 mode.

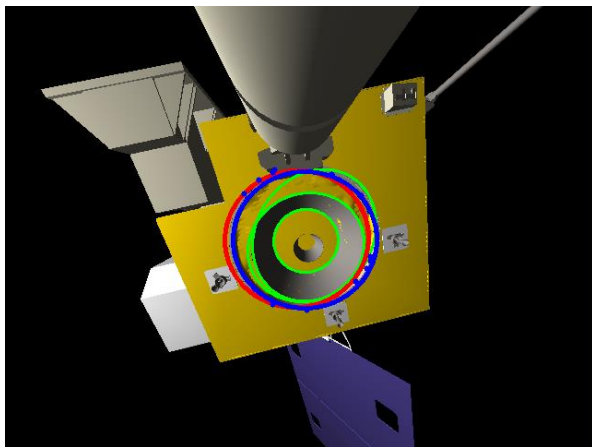


Camera-1

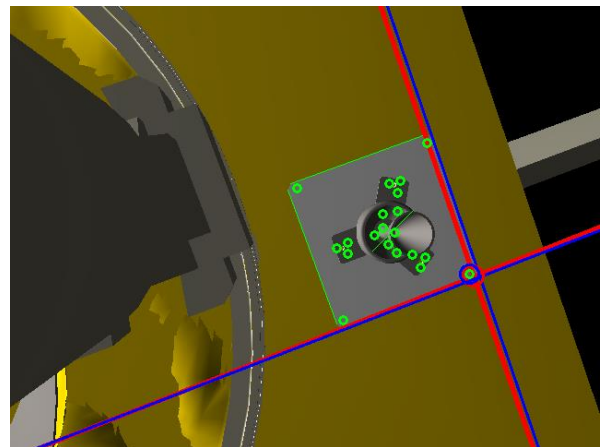


Camera-2

Figure 38. Edge map of the camera views of the virtual simulator in CAD model mode



Camera-1



Camera-2

Figure 39 . Reference features (red), detected features (blue) and extracted features (green) obtained using the pose estimation system with the images of the virtual simulator in the CAD model mode.

11.2.2 Software Configuration

The virtual simulator is implemented in C++ using OpenGL. The implementation of the virtual simulation when coupled with the pose estimation system is presented in Figure 40. The OpenGL frames are converted to OpenCV images and given to the pose estimation system. The resolution of the generated images is 640×480p. A data logger system saves all the relevant variables, such as the estimated pose and the ground truth. As can be seen there are two threads that run simultaneously, they run the feature extraction and detection systems for each camera. Figure 41 shows the implementation of the individual feature extraction and feature detection subsystems in each thread, i.e. showing the ellipse, line and point extraction and detection subsystems. The system is running on a Linux desktop with two quad-core Xeon processors and 4GB of Ram.

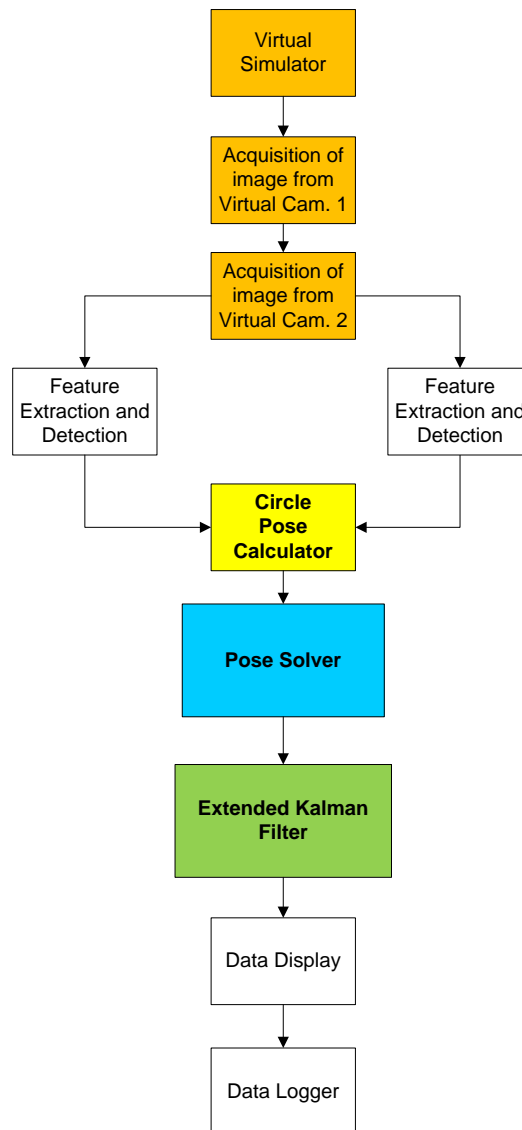


Figure 40. Flowchart of the implement code of the virtual simulator and the pose estimation system

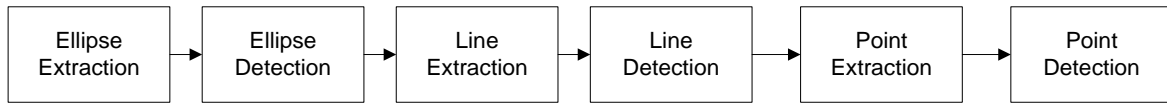


Figure 41. Flowchart of the implement code of feature extraction and detection system

11.2.3 Testing Procedure

Two types of virtual tests were performed: static tests and dynamic tests. In the static tests the relative position between the end effector and the target was held constant. The wireframe model-1 and Scenrio-3 were used for the static tests. For these test the distance between the target and the end effector was 0.8 meters. There are two types of static tests.

11.2.3.1 Effects of Camera Parameters

The objective of these tests is to observe how the pose error is affected by the error in the extrinsic and intrinsic parameters of the cameras. This is important since the pin-hole model is used in the pose estimation system and the estimated pose is function of the camera intrinsic parameters (c_x, c_y, f_x, f_y) and image distortion. The estimated pose is also function of the camera extrinsic parameters since the pose is specified w.r.t the end effector. In addition, the camera parameters are obtained experimentally and are subjected to measurement errors. The nominal values of the camera parameters are presented in Table 6, the extrinsic parameters correspond to the Euler angles and the translation vector of the camera frames w.r.t. the end effector frame.

Table 6. Intrinsic and extrinsic parameters of the cameras

Parameter		Camera 1	Camera 2
Intrinsic Parameters	f_x	162.157	472.332
	f_y	162.003	472.732
	c_x	321.272	312.882
	c_y	240.417	242.175
Extrinsic Parameters	$\varphi_{C/E}$	-0.2169 rad	-0.0177 rad
	$\theta_{C/E}$	0.0763 rad	0.0870 rad
	$\psi_{C/E}$	0.2168 rad	-1.2183 rad
	$X_{C/E}$	-0.0573 m	0.0580 m
	$Y_{C/E}$	0.1835 m	-0.1589 m
	$Z_{C/E}$	0.0692 m	0.1392 m

To test the effect of the parameters, each parameter is modified one at a time, w.r.t. the nominal values of each camera, by an additive known value, which represents the parameter error. These “erroneous” value is given to the subsystems of the pose estimation system to estimate the pose. The images are generated using

the correct values of the camera parameters. All the parameter for each camera are modified, one at a time and the pose estimation error is compute. As an example, if the parameter $f_x = 300$ of camera-1 is being tested then a value of $f_x = 300 + \Delta f_x = 300 + 10 = 310$ is given to the pose estimation system (it does not know that the true value of 300) while the virtual image for camera-1 is generated using the value of 300. For each parameter of each camera five different values are used.

The effect of the camera radial distortion is evaluated using an image with a known amount of distortion. The distorted image is generated using the following radial distortion transformation,

$$\begin{cases} x_u = x_d(1 + K(x_d^2 + y_d^2)) \\ y_u = y_d(1 + K(x_d^2 + y_d^2)) \end{cases} \quad (131)$$

where x_d and y_d are the pixel coordinates in the distorted image (w.r.t. the center of the image), x_u and y_u are the coordinated of the corresponding pixel in the undistorted image. K is the distortion coefficient, which is computed as,

$$K = \frac{d}{(0.5 + d)^3 V_{pix}^2} \quad (132)$$

where V_{pix} is the image vertical resolution in pixels and d is the amount of distortion, as an example $d = 0.03$ for a 3% distortion. Figure 42 shows a representation of the terms in the previous equation.

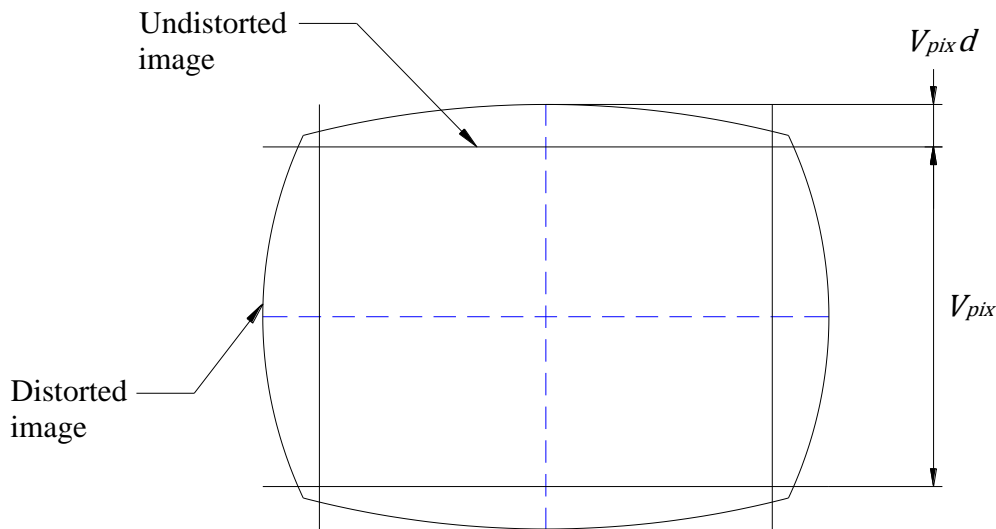


Figure 42. Image barrel distortion

11.2.3.2 Effect of Relative Pose

These tests show the effect of the relative pose in the pose estimation error. In this case different relative poses between the target and the end effector are used to find the pose error. The poses are generated by

rotating and translating the target along each of the axis of its reference frame starting from the nominal position, as presented in Figure 43 and in Figure 44. For each axis five different translations and five different rotations are performed. Motions along each axis are performed individually one at a time. These are static tests and the tests start with the system at the altered position, i.e. there is no motion during the tests.

11.2.3.3 Dynamic 6DOF Tests

There are two type of dynamic tests, the 6DOF motions tests and the approach tests. The 6DOF tests are performed using the virtual simulator with the CAD-model and starting from an initial position where the target is at 1.2m from the end effector. Scenario number three is used for this test. During these tests rotations and translations of the target are performed along each of the axis of its reference frame, as presented in Figure 43 and in Figure 44. Table 7 shows the values of the rotation angle and the displacement for each motion. As an example, when performing a typical maneuver such as a translation along the x-axis of the target, the system starts at the initial nominal position-2, it stays there for about 20 seconds, then it moves 20cm along the x-axis, it waits 20 seconds at the new position, then it returns to the original position and waits there for 20 seconds, finalizing the test. This is done for each axis with rotation and translations, one at a time.

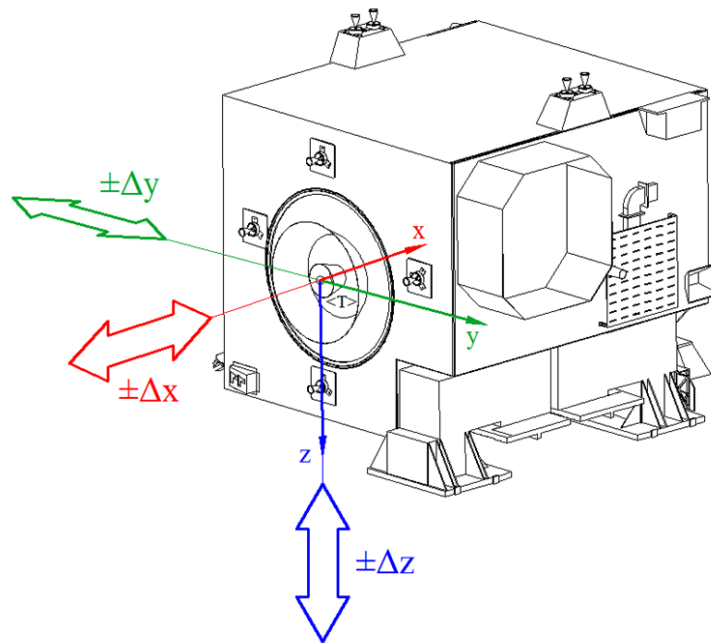


Figure 43. Target translation movements during virtual testing.

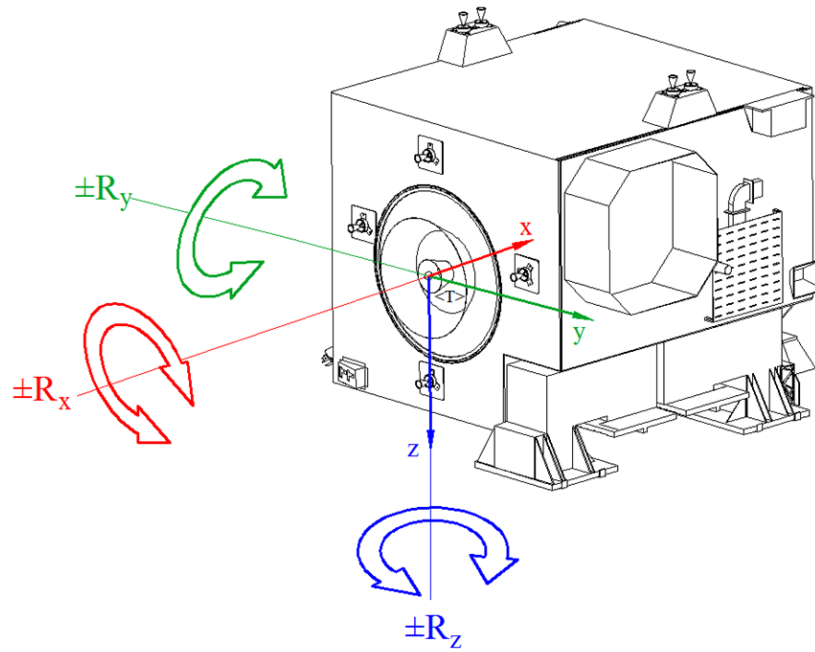


Figure 44. Target rotation movements during virtual testing.

Table 7. Target motions for virtual testing

Motion	Value
x-translation	0.2m
y-translation	0.2m
z-translation	0.2m
x-rotation	-20deg
y-rotation	-20deg
z-rotation	-20deg

11.2.3.4 Approach Tests

The second type of dynamic tests are the approach tests, in this case the system starts from an initial position where the target is at 1.6m from the end effector, and it approaches the target until they are at 0.7m from each other. These tests are performed with the CAD model, the wireframe model-1 and the wireframe model-2 and using the three scenarios. A test using a modified scenario-3 is performed, where the base of the heat shield of the apogee thruster is used instead of the interface ring. Additionally a non-calibrated camera is used, where it is assumed that virtual camera-1 has an error in an extrinsic parameter of 5 deg, i.e. there is a rotation error of 5 deg in the position of the camera with respect to the end-effector. The virtual image is generated with the correct camera position but the pose estimation system uses the “erroneous” camera position.

During all the tests all the data generated by of the pose estimation system including its internal parameter and variables is logged. Table 8 summarizes the virtual tests.

Table 8. Virtual tests

Test classification		Test variables	Scenario	Model	Target-End effector distance	# runs
Static Tests	Effect of parameters	Camera-1 parameters: $c_{x1}, c_{y1}, f_{x1}, f_{y1}, K_1$	3	Wireframe 1	0.8m	25 (5 per parameter)
		Camera-2 parameters: $c_{x2}, c_{y2}, f_{x2}, f_{y2}, K_2$	3	Wireframe 1	0.8m	25 (5 per parameter)
	Effect of parameters	Camera-1 position changes: rot-x, rot-y, rot-z trans-x, trans-y, trans-z	3	Wireframe 1	0.8m	30 (5 per parameter)
		Camera-2 position changes: rot-x, rot-y, rot-z trans-x, trans-y, trans-z	3	Wireframe 1	0.8m	30 (5 per parameter)
	Effect of position	Target position changes: rot-x, rot-y, rot-z trans-x, trans-y, trans-z	3	Wireframe 1	0.8m	30 (5 per position)
Dynamic tests	6DOF motions	Target motions: rot-x, rot-y, rot-z trans-x, trans-y, trans-z	3	Wireframe 1	1.2m	6 (1 per motion)
	Approach tests	Calibrated cam	1,2,3	CAD - model	1.6 - 0.7m	3 (1 per scenario)
		Calibrated cam	3	Wireframe 1	1.6 - 0.7m	1
		Calibrated cam	3	Wireframe 2	1.6 - 0.7m	1
		Calibrated	Modified 3: heat shield base instead of ring	CAD-model	1.6 - 0.7m	1
		Uncalibrated cam	3	CAD - model	1.6 - 0.7m	1
		Uncalibrated cam	3	Wireframe 2	1.6 - 0.7m	1
				Total runs	154	

11.3 Experimental Testing

Experimental tests using a realistic satellite mockup were performed. The satellite mockup was provided by NASA’s GSFC and the tests were conducted at the WVRCT facilities. The satellite is based on an existing weather satellite. The objective of the experimental testing is to evaluate the pose estimation system under realistic conditions. These conditions make difficult the performance of the system due to image noise, image

distortion, calibration errors in the cameras, light reflections, low contrast between the satellite components and the features of interest; and textures in the surface of the satellite components that creates false edges in the image. The following section describes the experimental setup and the different types of test that were performed.

11.3.1 Experimental Setup

A schematic view of the experimental setup is presented in Figure 45, and the actual view of the setup at the WRTC facilities is presented in Figure 46. The main components of the setup are described below.

- **Satellite mockup.** It replicates the front panel of an actual weather satellite, where the interface ring is located. It is covered with actual satellite MLI. It is important to use this component as real as possible since it create several reflections and textures that complicate the performance of the machine vision base pose estimation system. The diameter of the interface ring is 1m approximately and it is a machined component made of an aluminum alloy. It has an apogee thruster and four AOC thruster, each thruster has its own heat shield. It is important to replicate these elements since they can create occlusions over the features of interest.
- **End effector.** The end effector contains the two cameras and the capture tool and it is attached to a robotic manipulator. It is important to include all the component of the end effector since they can create occlusions of the target in the camera views.
- **Robotic manipulator.** This 7DOF manipulator is used to simulate the relative motions between the target and the end effector since the target is fixed during the tests. It is a Motoman SIA50D. The readings from the robot joints are used to calculate the relative motion.
- **Camera-1.** This camera is tracking the interface ring, it has a wide angle lens to maximize the view of the ring at close distances. The properties of the camera are presented in Table 9. It is mounted on the end-effector as presented in Figure 47.
- **Camera-2.** This camera has a regular angle lens. It tracks the features on the mockup that were defined on each testing scenario. Its properties are presented in Table 9. It is mounted on the end-effector as presented in Figure 47.
- **Laser tracker.** A Leica laser tracker system (Model AT901) is used to obtain the ground truth. It tracks a device installed on the mockup to find 6DOF pose with a submillimeter accuracy.
- **Spot light.** The spot light is used to test the system at low light conditions.

The intrinsic parameters of the cameras were obtained using the OpenCV camera calibration functions and a checkerboard. The extrinsic calibration (camera position and orientation w.r.t the end effector-frame) was performed using known robot motions and a pose estimation system based on concentric circles. Camera intrinsic and extrinsic parameters were presented in Table 6.

Table 9. Camera specifications

Camera-1 and Camera-2	
Model	Sony XCD-SX90CR
Max resolution	1280 by 960
Max frame rate	30 fps
Digital Interface	IEEE 1394b-2002 x 2
Sensor size	1/3"
Lens of Camera-1	
Model	Theia MY125M
HFOV for 1/3" sensor	125deg
Distortion	less than 3% barrel distortion
Lens of Camera-2	
Model	Kowa LM3NCM
HFOV for 1/3" sensor	70deg
Distortion	0.4%

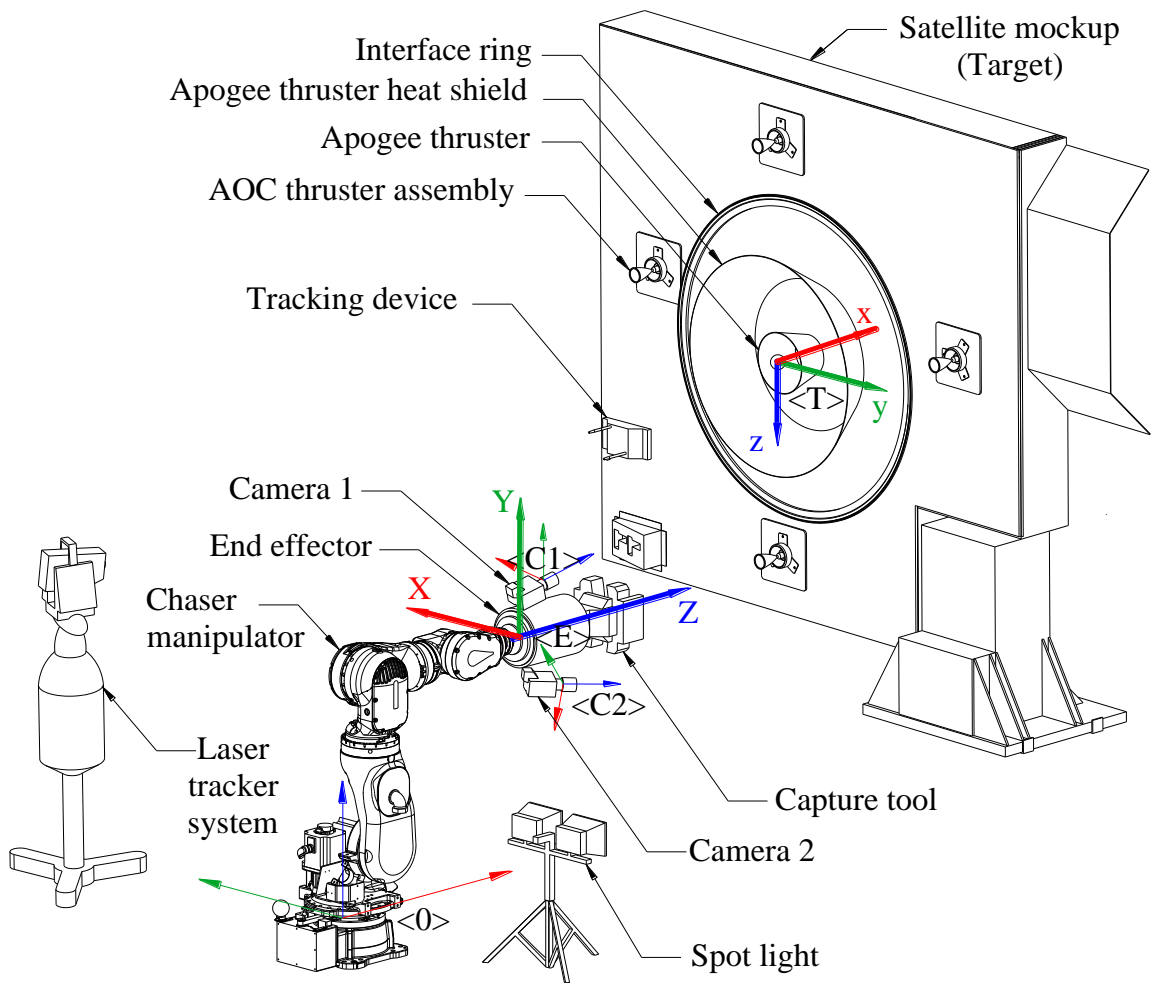


Figure 45. Schematic view of the experimental setup

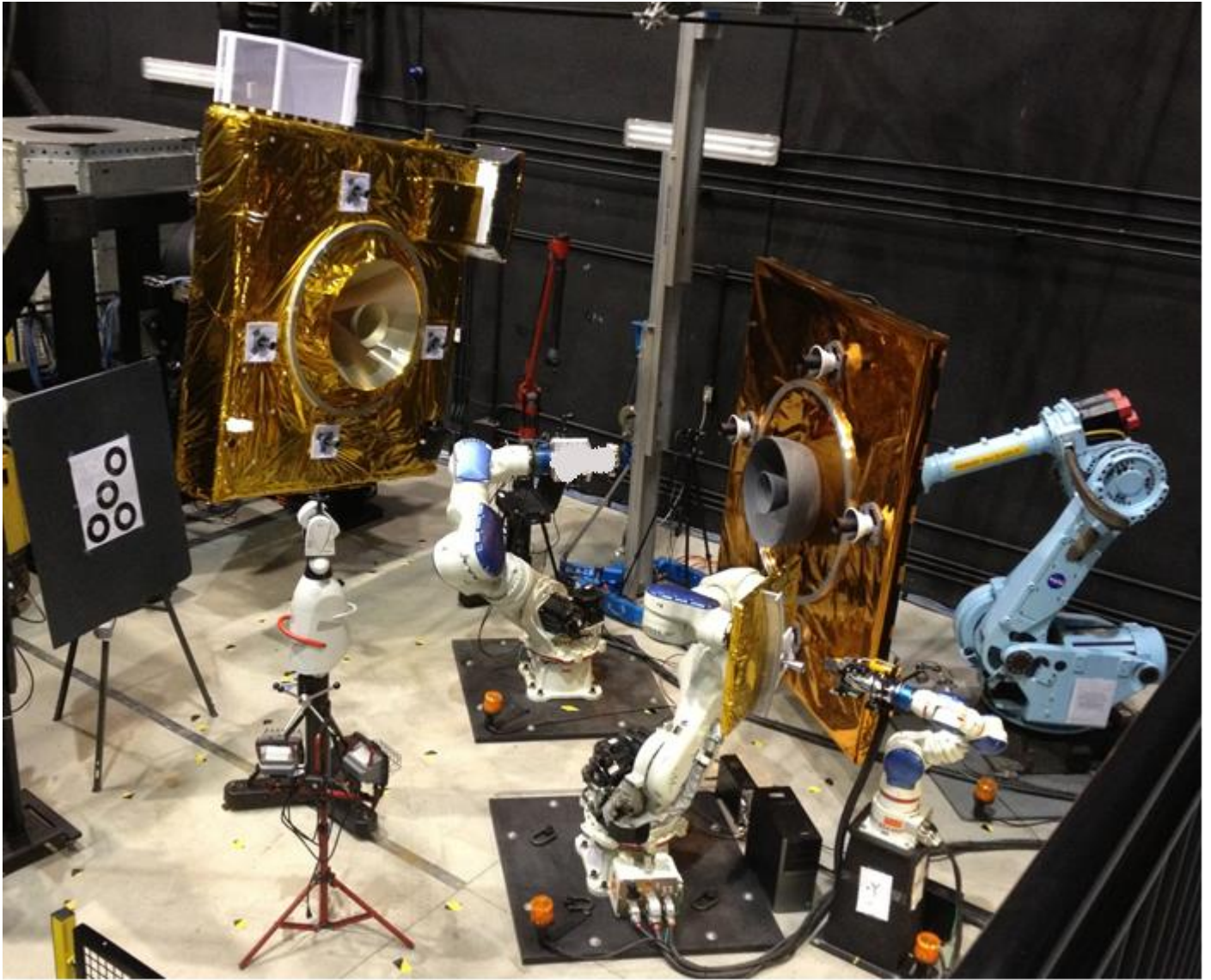


Figure 46. Experimental setup at the WVRTC facility

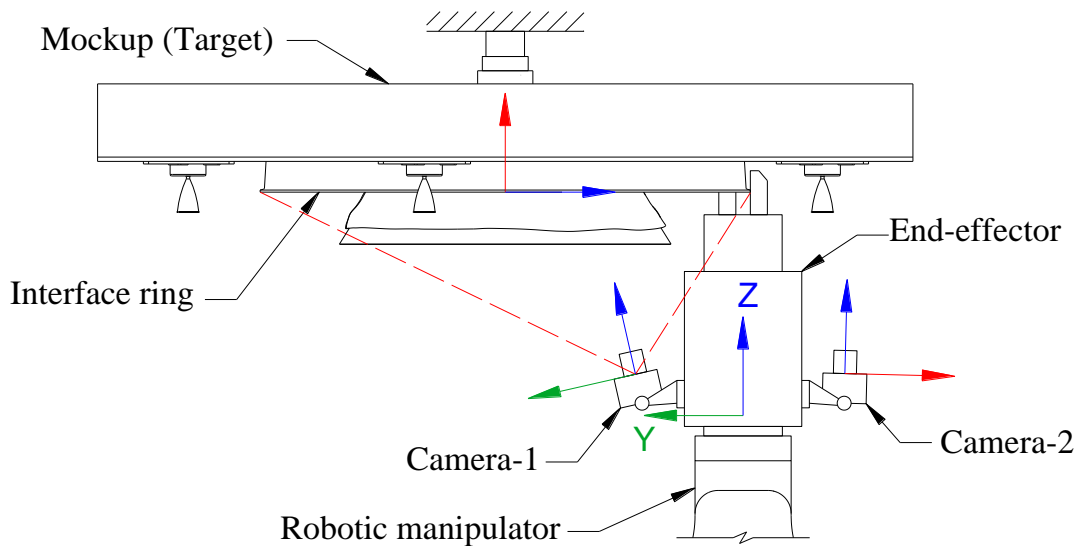


Figure 47. Location of cameras on the end-effector (schematic view)

11.3.2 Hardware and Software Configuration

During the experimental testing video from the two cameras, the readings from the laser tracking system and the robot position is recorded. The system is evaluated in an offline process using all the gathered data. The flowchart of this version of the pose estimation system that reads videos is presented in Figure 48. It was implemented in C++, the only difference with the implementation used during the virtual test is that this version takes the images from a video, while the former uses images generated by the virtual simulator. Although the video is captured at 1280×960p they are resized to 640×480p. A data logger system saves all the relevant variables, such as the estimated pose and the ground truth, which is computed using information from the laser tracker and the robot joints. As can be seen there are two threads that run simultaneously, they run the feature extraction and detection systems for each camera. Figure 41 shows the implementation of the feature extraction and feature detection subsystems in each thread, i.e. showing the ellipse, line and point extraction and detection subsystems. The system was implemented on a Linux desktop with two quad-core Xeon 2.8GHz processors and 4GB of RAM.

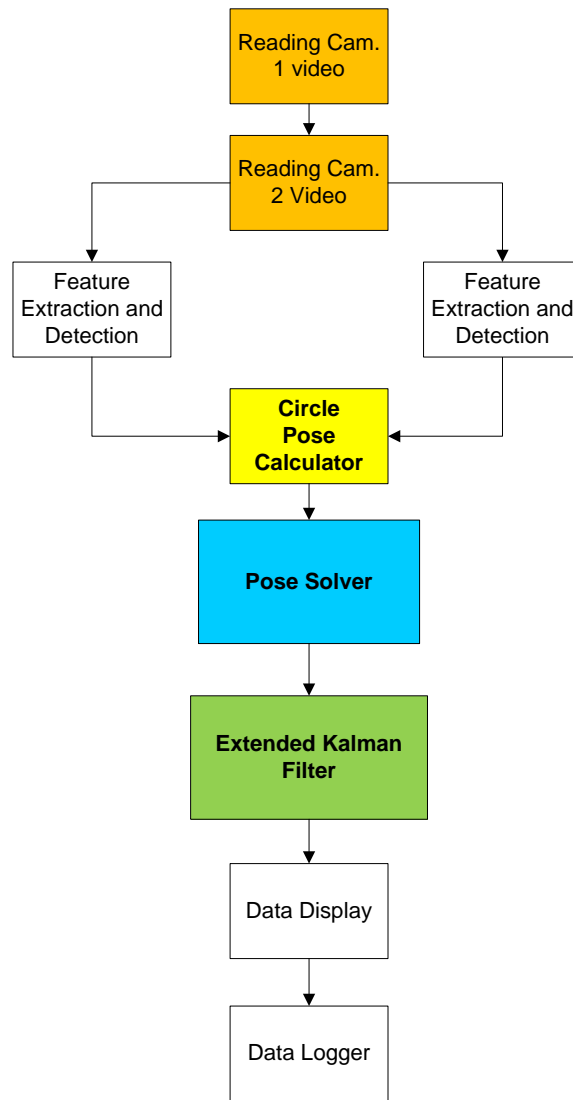


Figure 48. Flowchart of the implement code of the virtual simulator and the pose estimation system

11.3.3 Testing Procedure

Two types of experimental test are performed: static tests and dynamic tests. In the static tests the relative position between the end effector and the target is held constant. Tests were performed under two different light conditions: full light and low light and using the three scenarios previously defined. The details of each type of test are described in the following sections.

11.3.3.1 Tests for Static Pose Error

The first type of static tests, known as test for static pose error, are performed with the target at two different positions from the end-effector: 1.5m and 1.2m. The tests are performed at two different lighting conditions

and for the three scenarios. The lens iris and focus were in the same position during all the tests. During the tests data was collected for about 20 seconds.

11.3.3.2 Effect of initial conditions

The second type of static tests are intended to determine the effect of the initial conditions. In this case the initial condition that the EKF requires for initialization is modified by a small perturbation to make it different from the actual pose of the target. This test is performed for scenario-2 under low light condition and with the target at 1.2m from the end effector. Six different perturbations are used for the initial position of the target, each one corresponding to translations and rotations along each axis of the target frame. A value of 40cm was used for the translation perturbation and a value of 30deg for the rotation (20 deg for rot-z). As an example, for testing perturbation in the x-component, the target initial position is translated along its x-axis an amount of 20cm, and this modified position is given to the EKF, although the actual target is always at the same nominal position.

11.3.3.3 Dynamic 6DOF Tests

The first type of dynamic tests are the 6DOF tests, the initial position of the target during these tests was 1.2m, from the end effector and the three scenarios were used. During these tests rotations and translations of the end effector were performed along each axis of its reference frame. The target was held fixed. Figure 43 and in Figure 44 illustrate the motions performed. Table 10 shows the values of the rotation angle and the displacement for each motion. As an example, when performing a typical maneuver such as a translation along the x-axis of the end-effector, the system starts at the initial nominal position, it stays there for about 20 seconds, then it moves 20cm along the x-axis, it waits 20 seconds at the new position, then it returns to the original position and waits there for 20 seconds, finalizing the test. This is done for each axis with rotation and translations, one at a time.

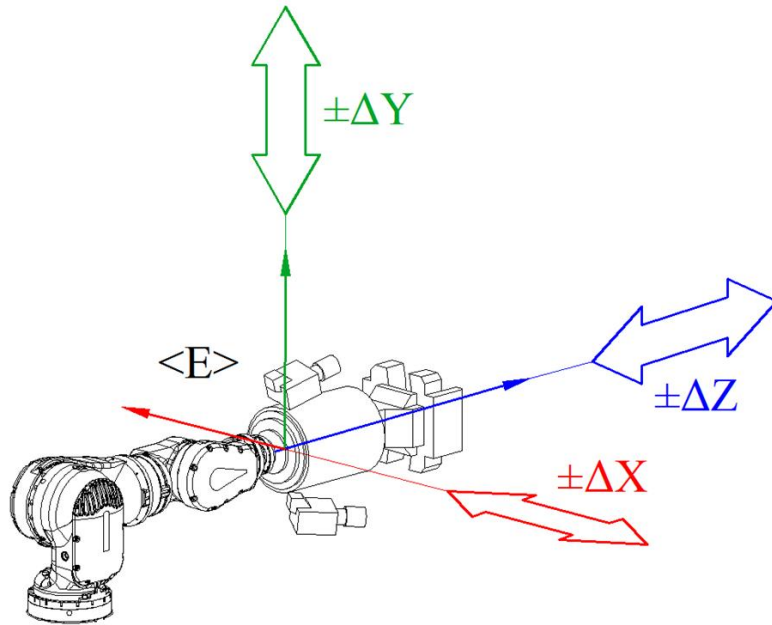


Figure 49. End-effector translation movements during experimental testing.

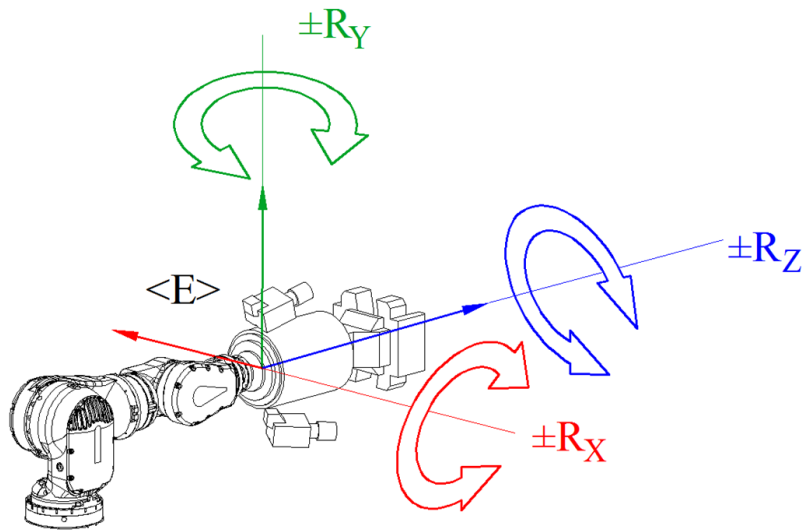


Figure 50. Target rotation movements during experimental testing.

Table 10. End effector motions for experimental testing

Motion	Value
X-translation	0.2m
Y-translation	0.15m
Z-translation	0.2m
X-rotation	20deg
Y-rotation	20deg
Z-rotation	20deg

11.3.3.4 Approach Tests

The second type of dynamic tests are the approach tests, in this case the system starts from a position where the target is at 1.5m from the end effector, and it approaches the target until they are at 0.7m from each other. These tests are performed at two lightning conditions and using the three scenarios.

During all the test all the data generated by the pose estimation system including its internal parameter and variables is logged. Table 11 summaries the experimental tests.

Table 11. Experimental tests

Test classification		Test Variables	Scenario	Light conditions	Target-End effector distance	# runs
Static Tests	Static Pose error		1,2,3	Full and Low	1.5m and 0.7m	12
	Effect of initial conditions	Target initial condition perturbation: rot-x, rot-y, rot-z trans-x, trans-y, trans-z	2	Low	1.2m	6 (1 per perturbation)
Dynamic tests	6DOF motions	Target motions: rot-x, rot-y, rot-z trans-x, trans-y, trans-z	1,2,3	Full and Low	1.2m	36
	Approach tests		1,2,3	Full and Low	1.5 - 0.7m	6
				Total runs		60

12 TESTING RESULTS

The results of the test described in the previous chapter are presented in the following sections. Screenshots of the camera views are also presented.

12.1 Virtual Testing

The results of the virtual tests are presented in the following sections.

12.1.1 Effects of Camera Parameters in the Pose Estimation Error

The rotation and translation error as a function of the camera parameter error for these tests are presented in Figure 51 through Figure 54. Each point in the figures represents an average of the data taken during the particular test.

Variation of the Translation Error due to Errors in the Parameters of Camera-1
 Scenario 3 - Wireframe model 1 - $Z_{T/E} = 0.8m$

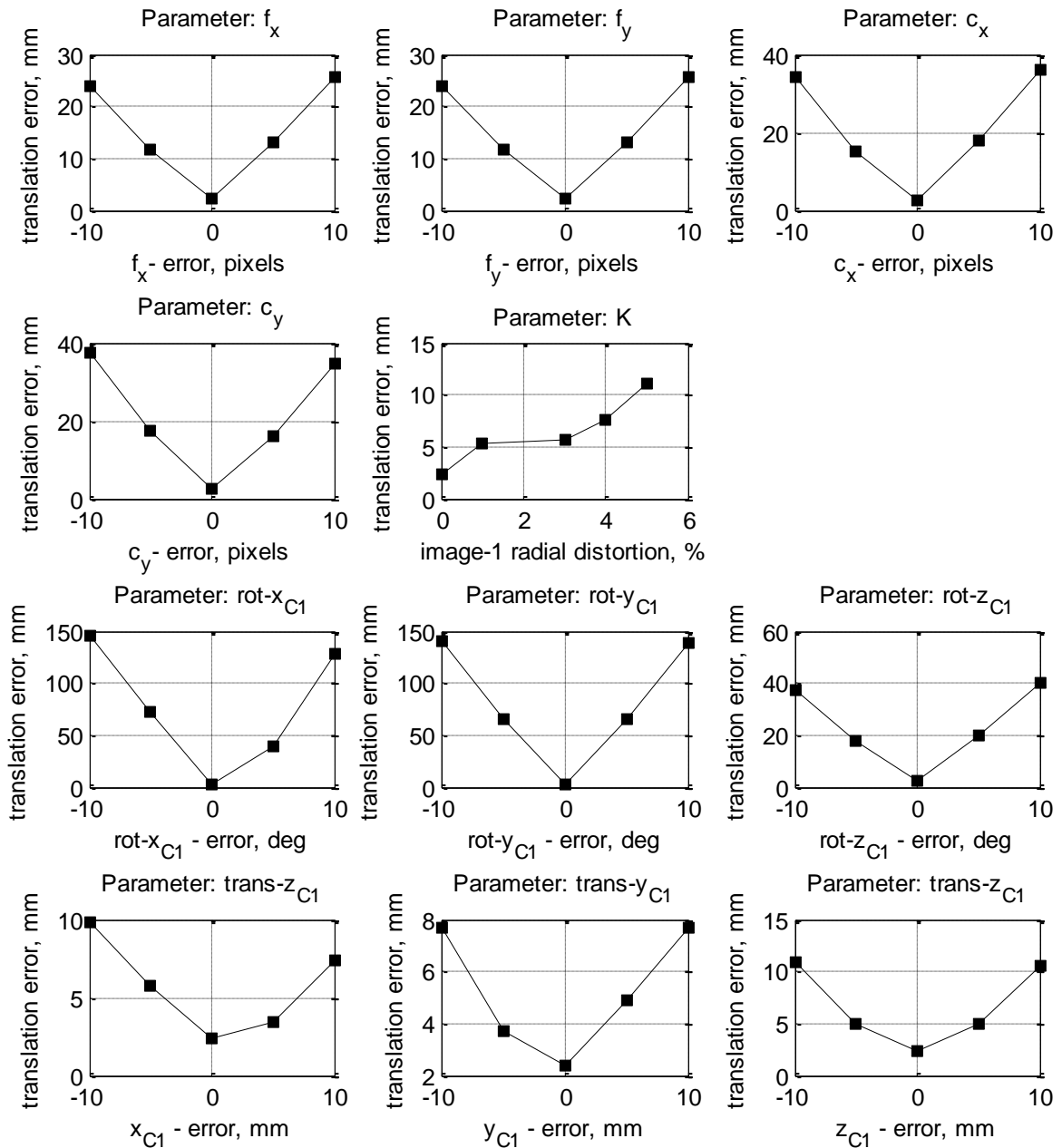


Figure 51. Effect of the camera-1 parameters in the translation error

Variation of the Rotation Error due to Errors in the Parameters of Camera-1
 Scenario 3 - Wireframe model 1 - $Z_{T/E} = 0.8m$

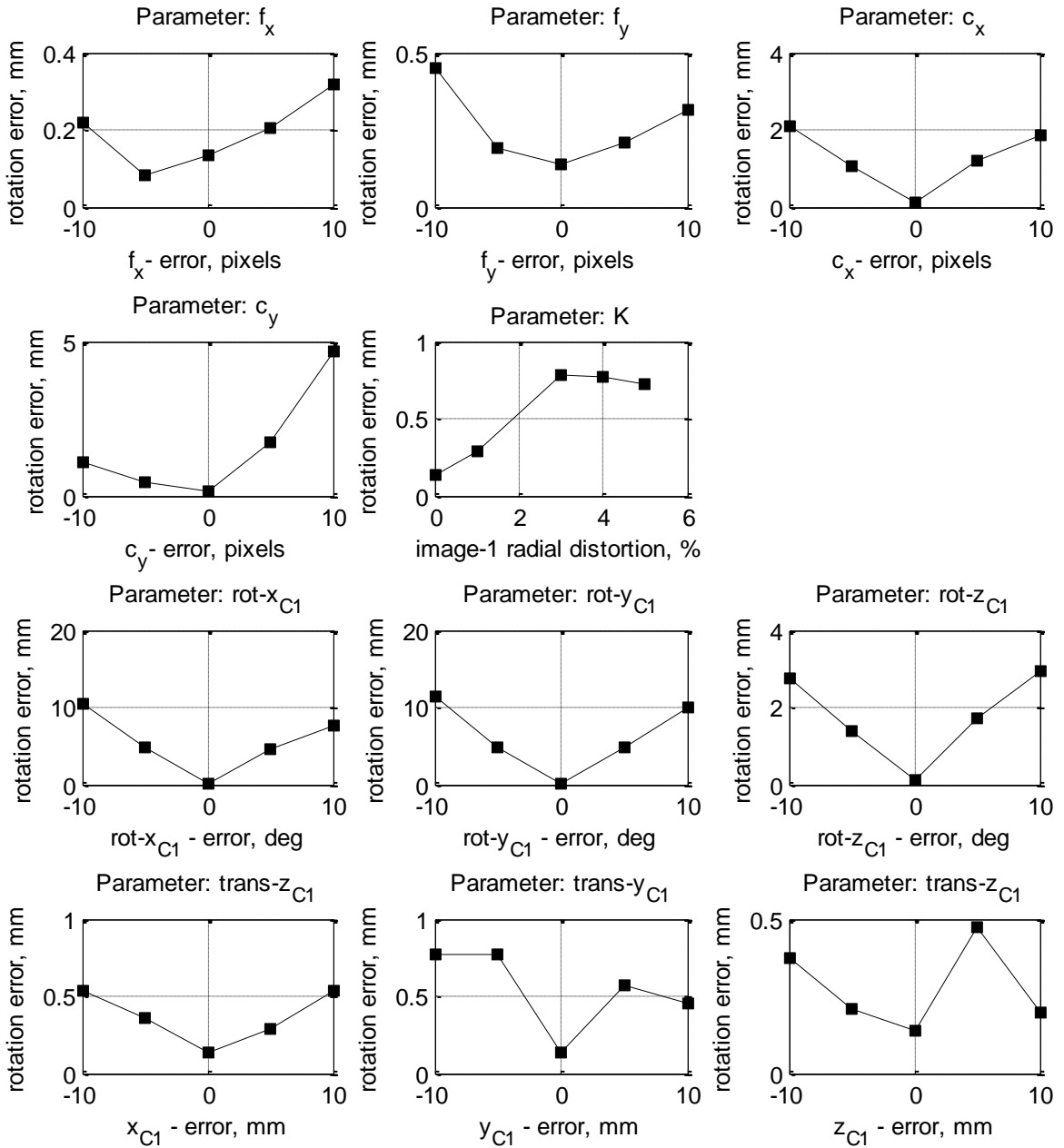


Figure 52. Effect of the camera-1 parameters in the rotation error

Variation of the Translation Error due to Errors in the Parameters of Camera-2
 Scenario 3 - Wireframe model 1 - $Z_{T/E} = 0.8m$

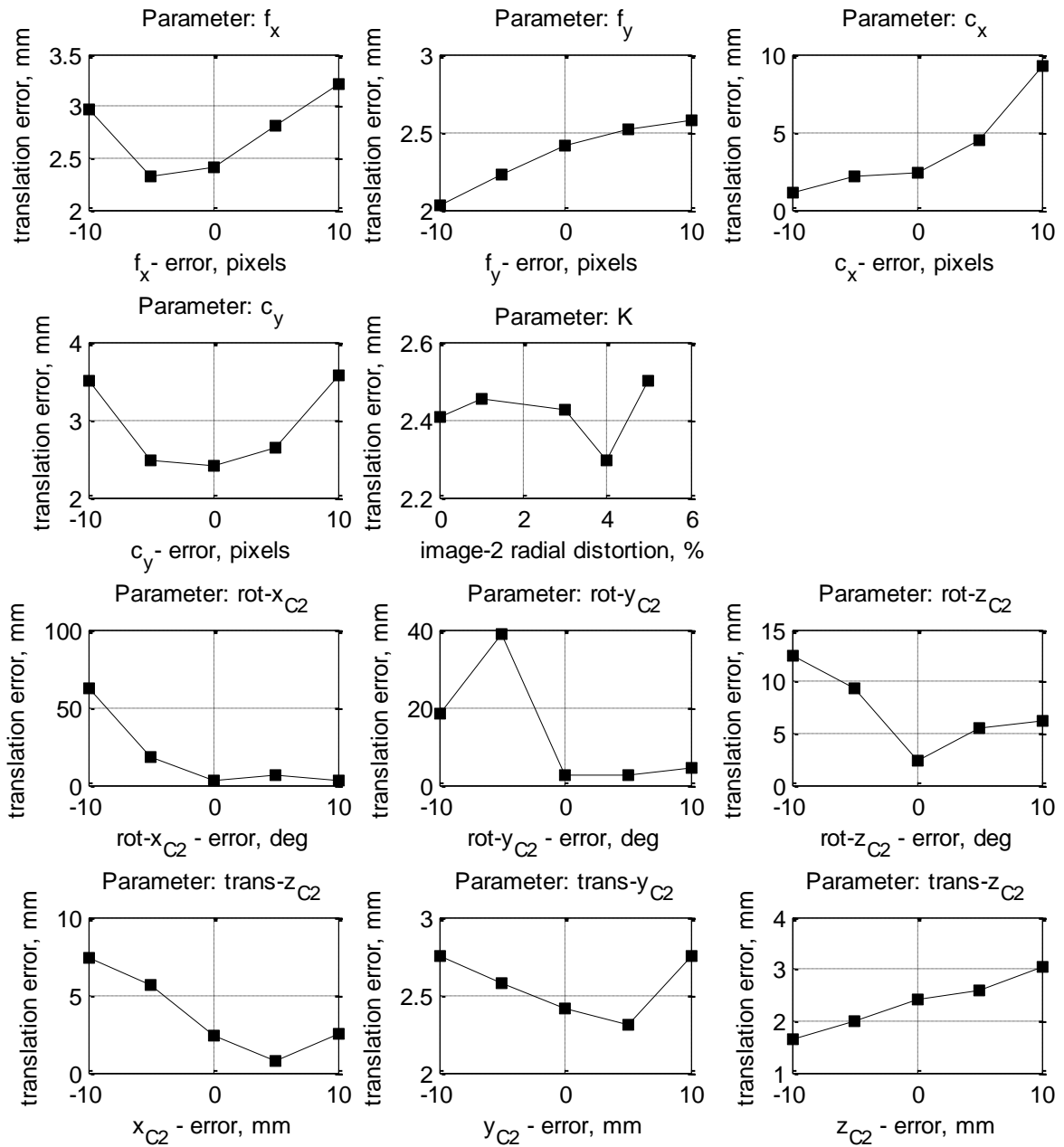


Figure 53. Effect of the camera-2 parameters in the translation error

Variation of the Rotation Error due to Errors in the Parameters of Camera-2
 Scenario 3 - Wireframe model 1 - $Z_{T/E} = 0.8m$

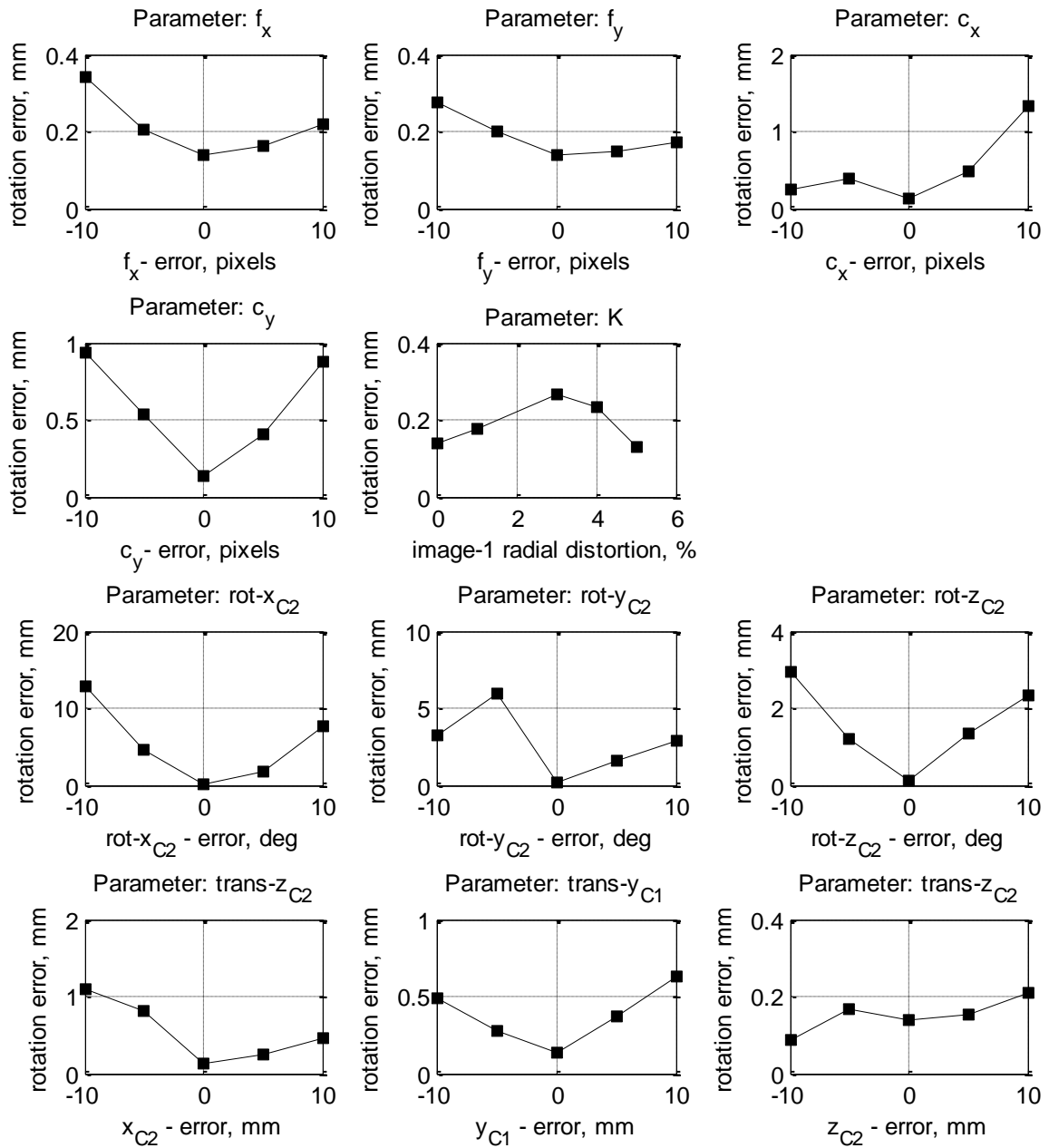


Figure 54. Effect of the camera-2 parameters in the rotation error

12.1.2 Effect of Relative Pose in the Pose Estimation Error

The results for these tests are presented in Figure 55, which shows the rotation and translation error as a function of the target displacement w.r.t the nominal position. Each point in the figures represents an average of the data taken during the particular test.

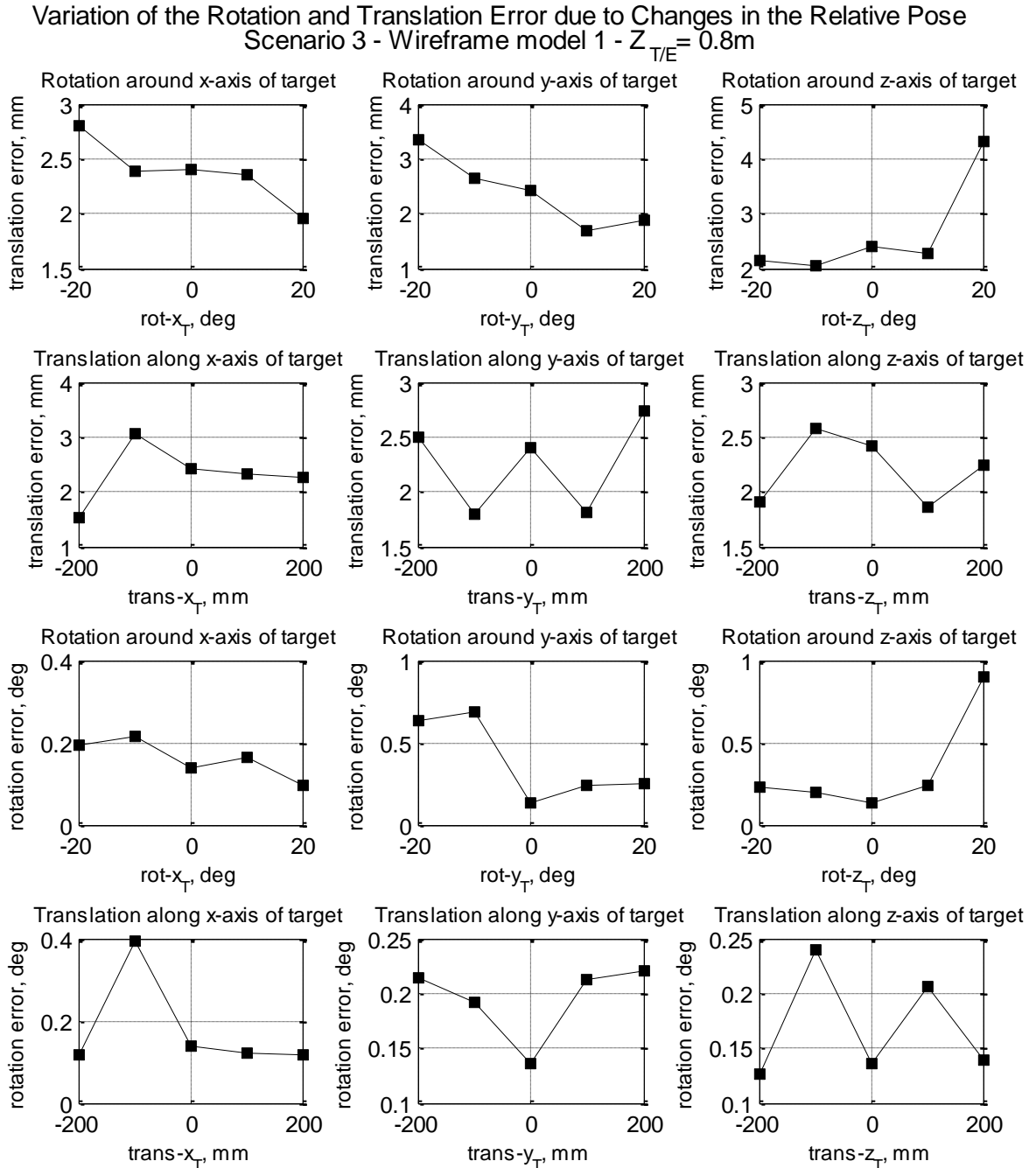


Figure 55. Effect of the relative pose in the system accuracy

12.1.3 Dynamic 6DOF Tests

Figure 56 shows the results for these tests, where the estimated and actual target displacement are shown for each one of the maneuvers. The displacement is expressed in the target frame as it is plotted against the simulation time. The variable corresponding to the performed valued is presented for each case, as an example if the maneuver is a translation along the target x-axis, then the estimated displace along that direction is plotted. Screenshots of the camera views for the initial and extreme positions of each maneuver are presented in Table 12 and in Table 13. The screenshots also show the detected and reference features.

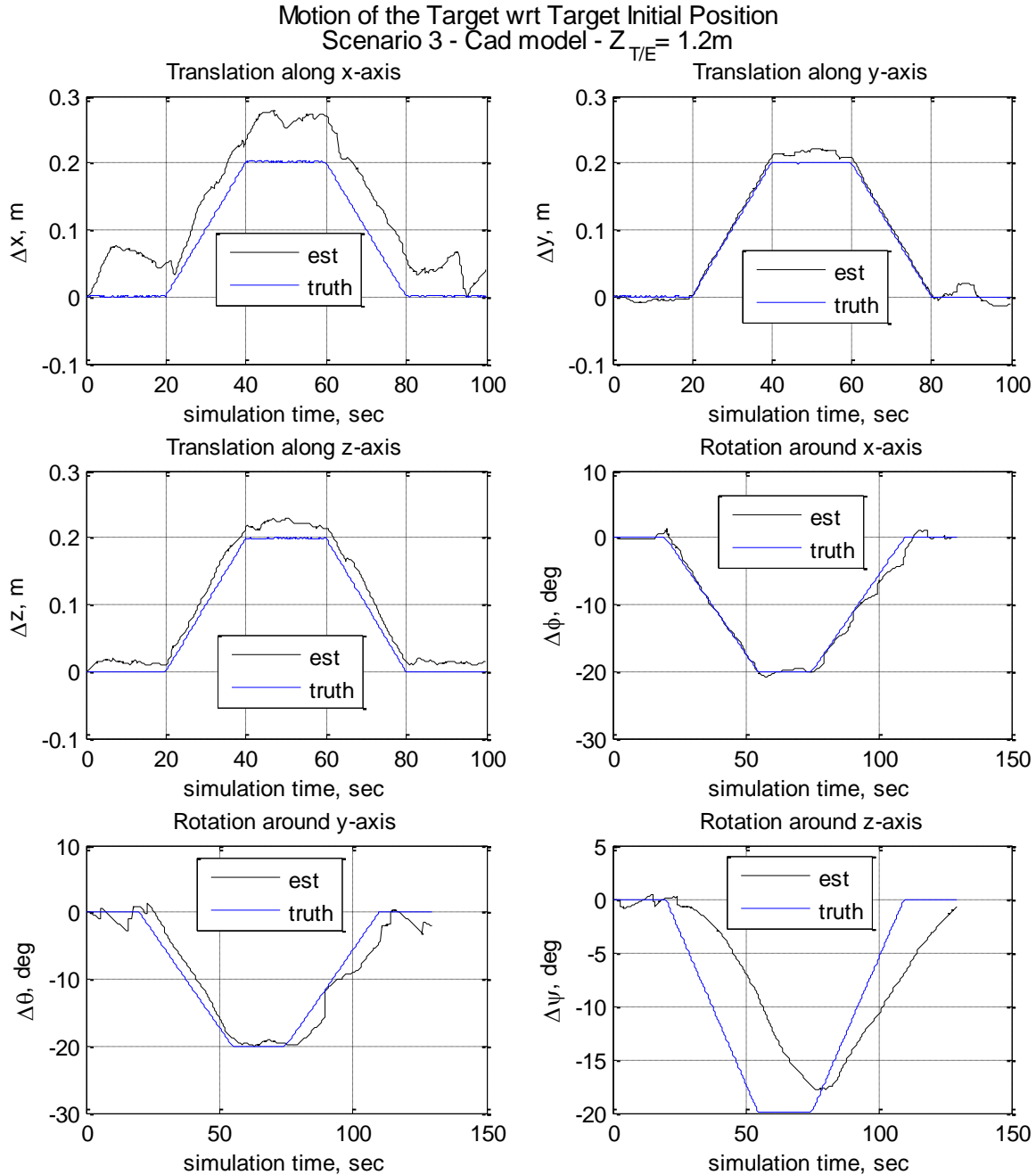
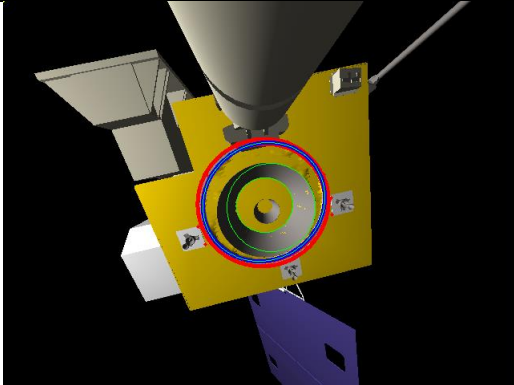
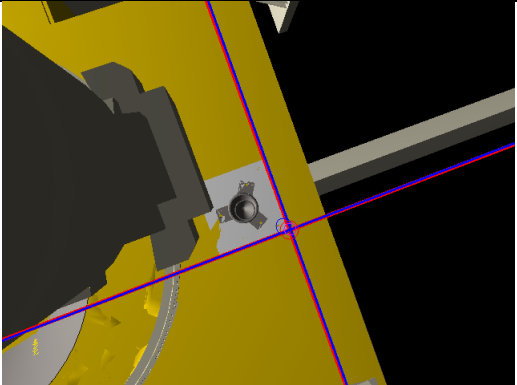
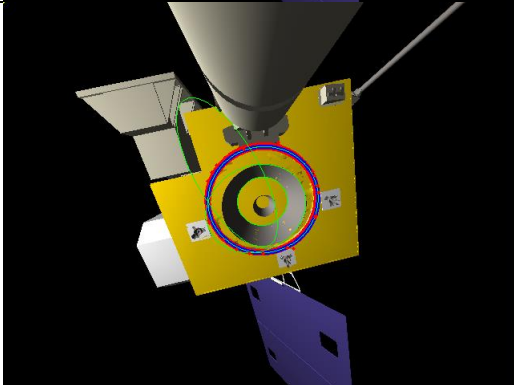
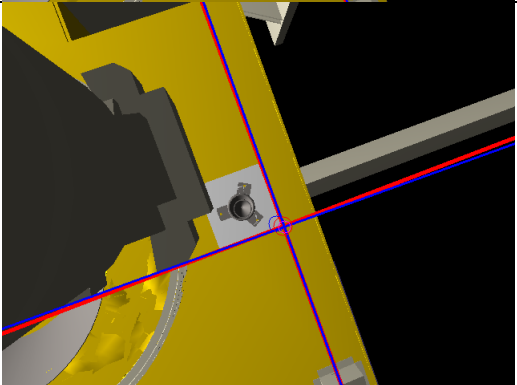
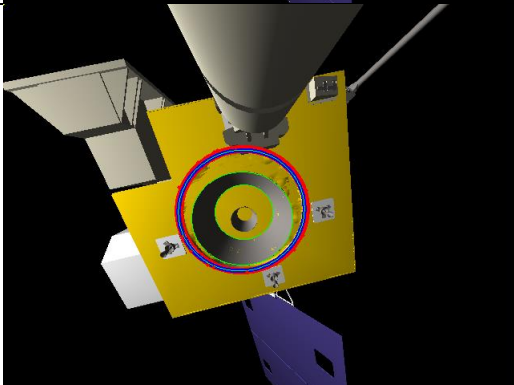
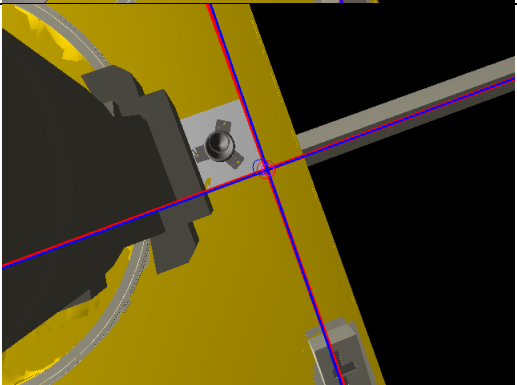
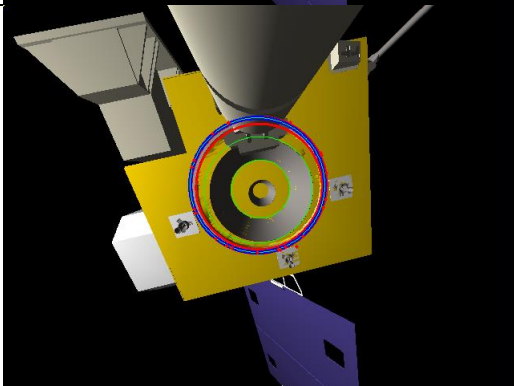
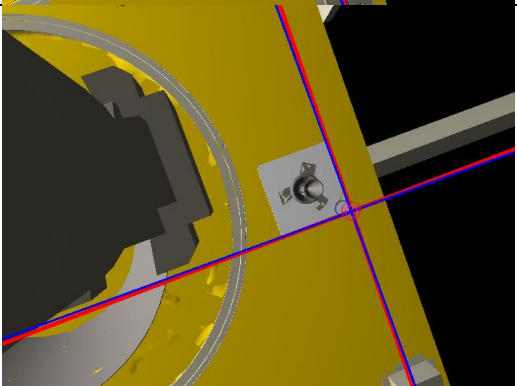


Figure 56. Virtual tracking tests

Table 12. Detected (blue) and reference (red) features at the extreme positions during the rotations of the virtual tracking tests

	Rotations Scenario 3 - $Z_{TE} = 1.2m$	
Motion	Camera 1	Camera 2
Home		
Rot-X		
Rot-Y		
Rot-Z		

Table 13. Detected (blue) and reference (red) features at the extreme positions during the translations of the virtual tracking tests

		Translations Scenario 3 - $Z_{TE} = 1.2m$	
Motion	Camera 1	Camera 2	
Home			
Trans-X			
Trans-Y			
Trans-Z			

12.1.4 Approach Tests

Figure 57 shows the actual values of the z-component of the distance between the target and the end-effector during the approach, which is the same for all the approach tests. Figure 58 shows the rotation and translation error for the tests performed using the CAD model and the three scenarios. Figure 59 shows a comparison between the translation errors of the tests performed using the three scenarios. Figure 60 shows a comparison of the translation error obtained when using the CAD model and the wireframe model-2, these comparison is presented for both the calibrated and the uncalibrated camera. Figure 61 shows the components x, y and z of the translation error when using the CAD model with the calibrated camera. Figure 62 shows the screenshots of the camera views and the detected features when using the base of the heat shield of the apogee thruster. Figure 63 shows a comparison of the translation error when using the CAD model with the interface ring, the CAD model with the heat shield base and the wireframe models 1 and 2.

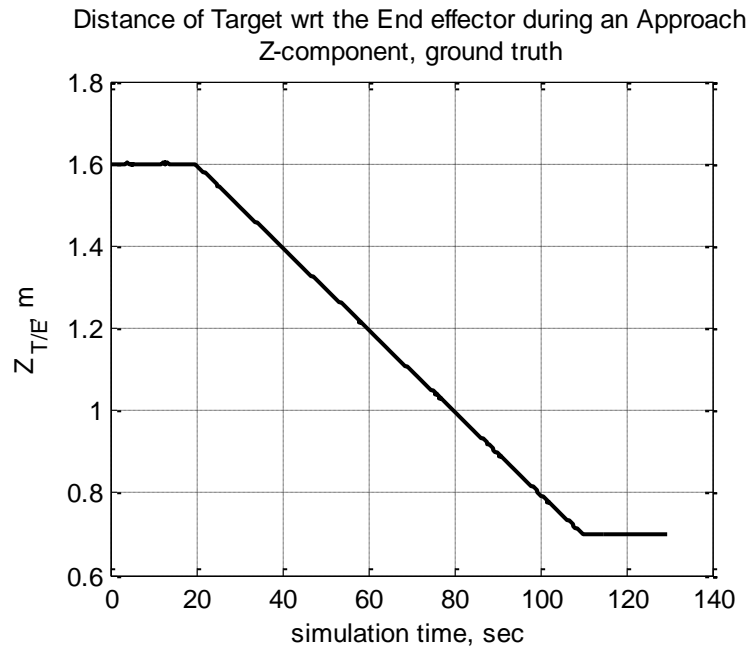


Figure 57. Variation of the z-component of the position of the target w.r.t the end-effector during the approach of the virtual test

Pose Error during an Approach
Scenarios 1, 2 and 3 - CAD model

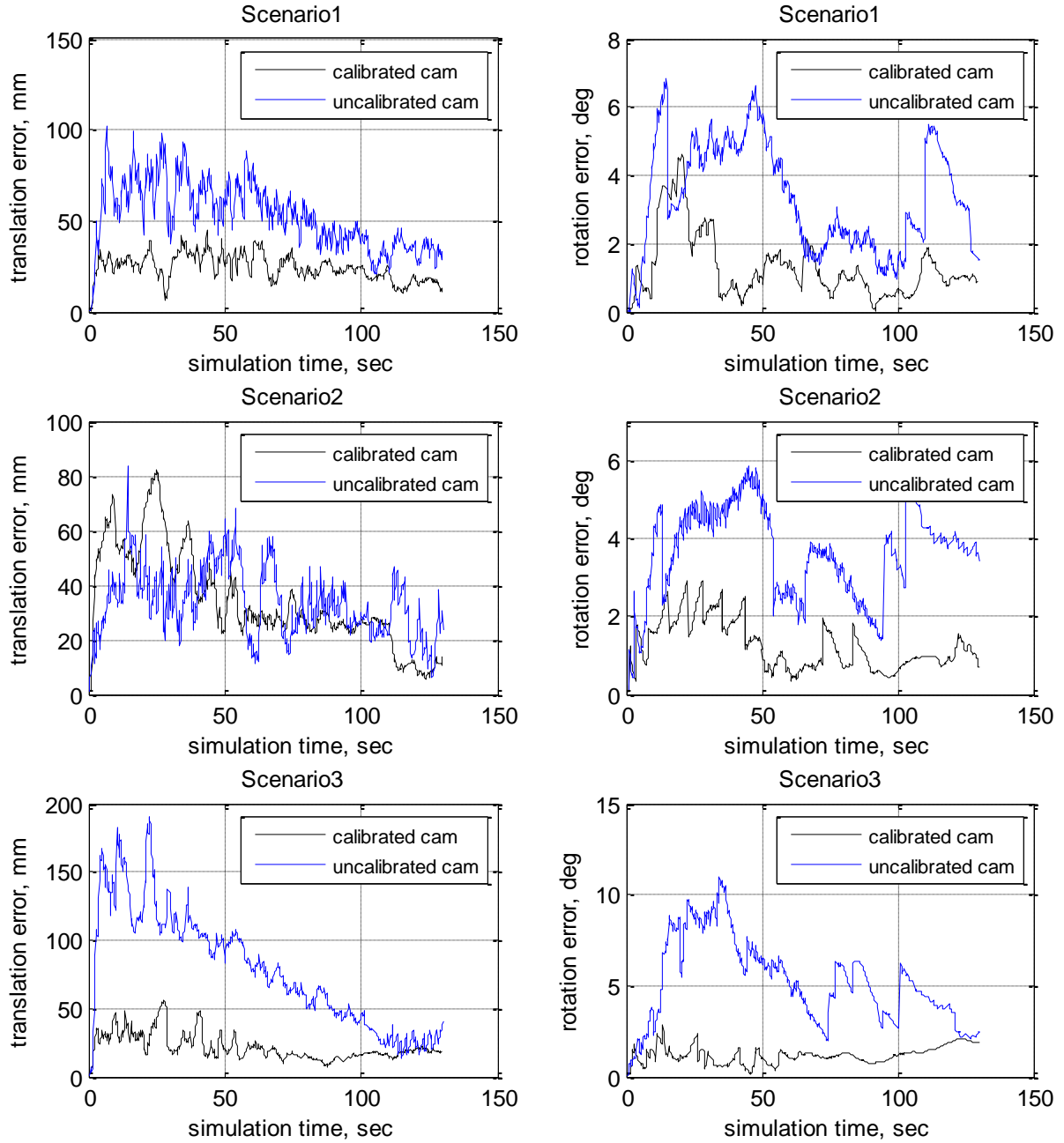


Figure 58. Translation and rotation error during the approach of the virtual test

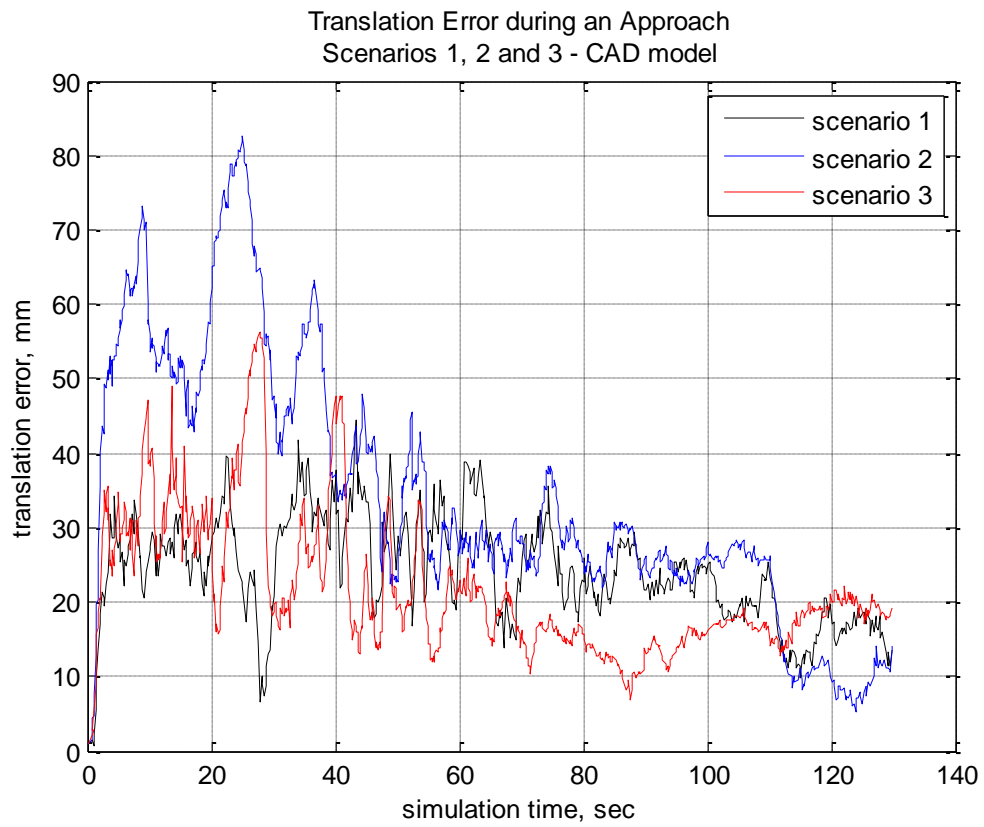


Figure 59. Comparison of the translation error of each scenario during the approach of the virtual test

Translation Error during an Approach. Scenario 3 - CAD model and wireframe model 2

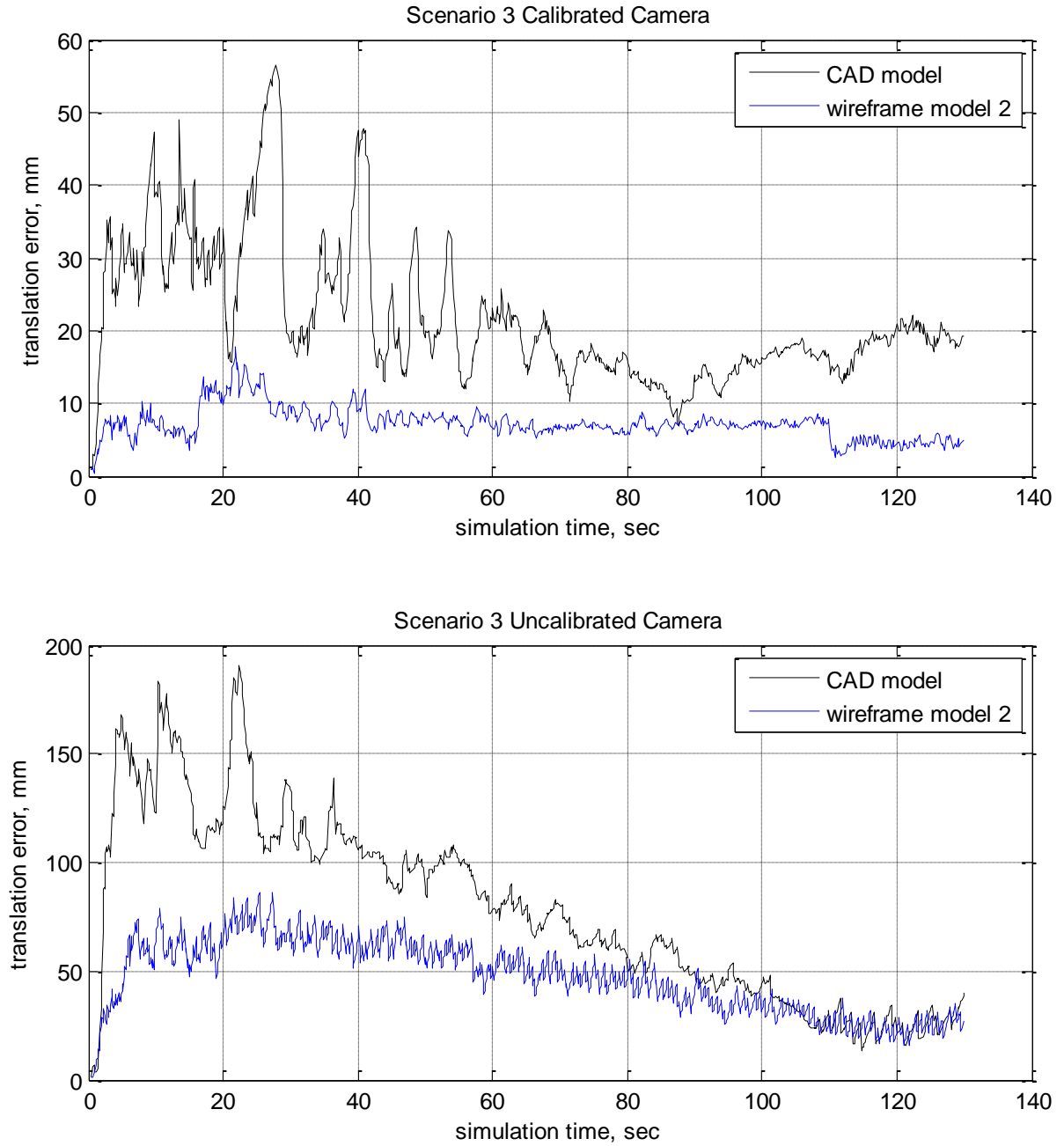


Figure 60. Comparison of the translation error of the CAD model and the wireframe model 2 during the approach of the virtual test

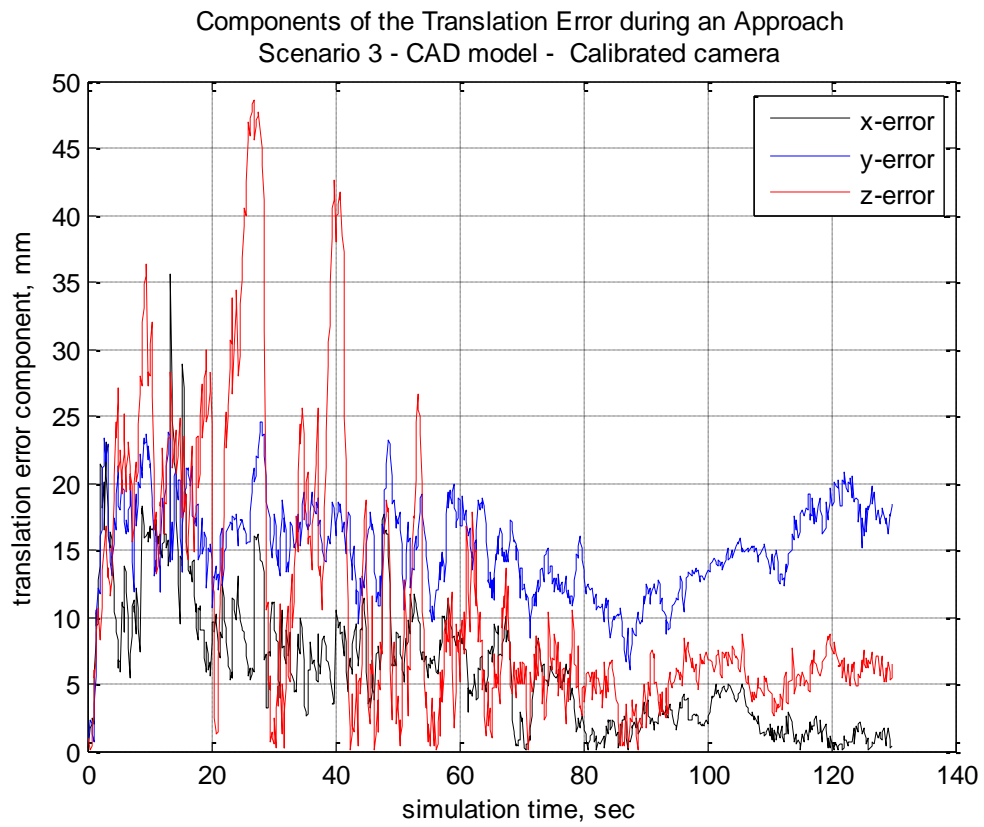
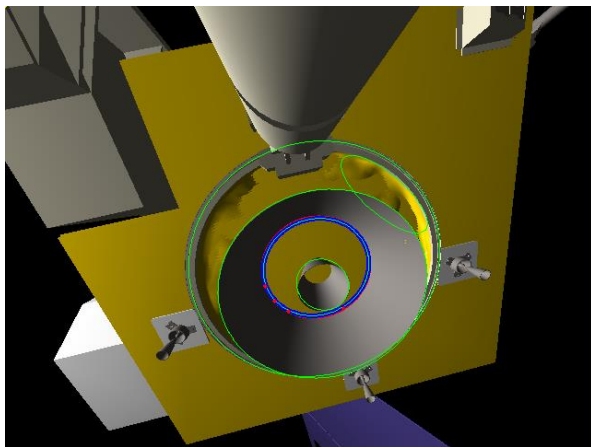
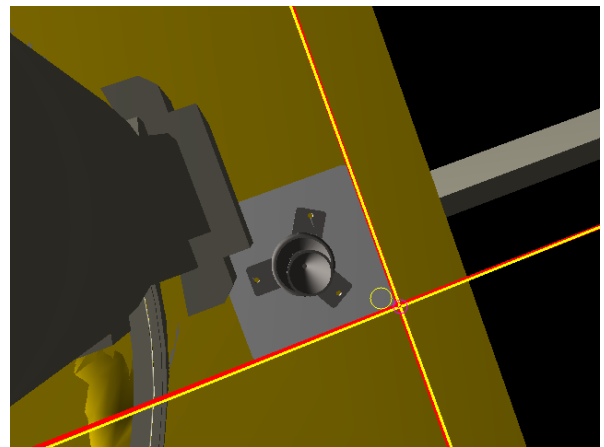


Figure 61. Components of the translation error during the approach of the virtual test



Camera-1



Camera-2

Figure 62. Camera views of the pose estimation system using the base of the heat shield of the apogee thruster

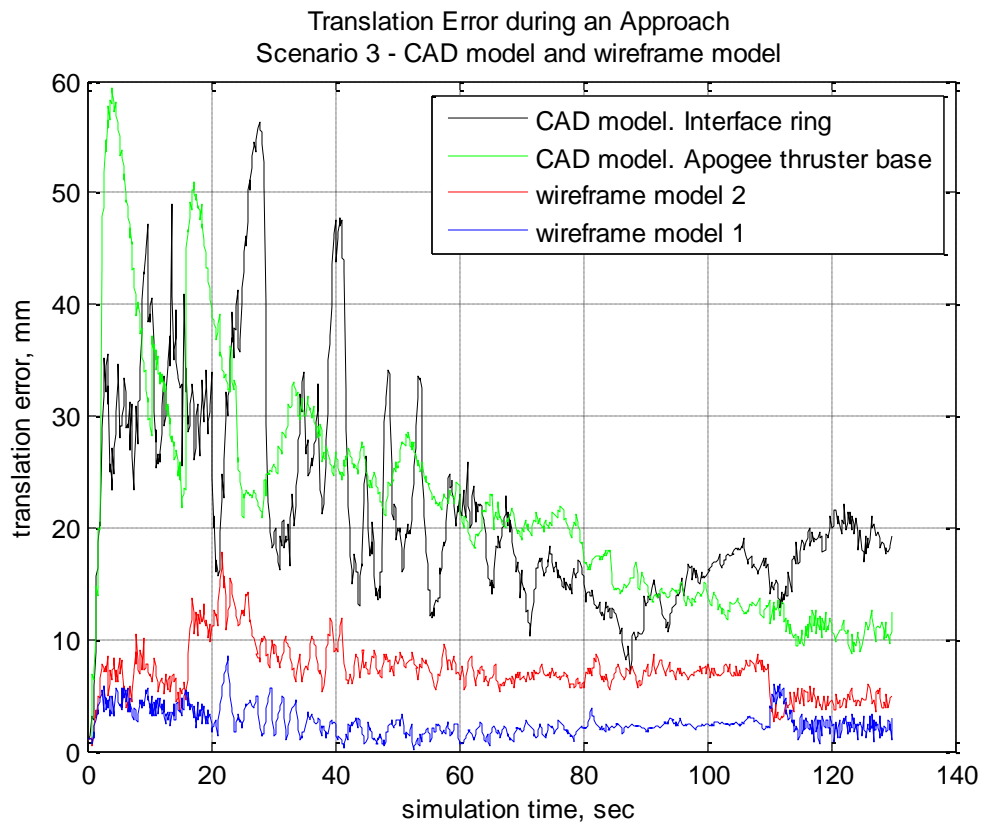


Figure 63. Comparison of the translation error of the CAD model with the interface ring, the CAD model with the heat shield base and the wireframe models 1 and 2 during the approach of the virtual test

12.2 Experimental Testing

This section show the result for the experimental tests that were described in the previous chapter. The discussion of the obtained results is presented in the next chapter.

12.2.1 Tests for Static Pose Error

Figure 64 shows the rotation and translation error for the several conditions used in these tests: scenarios, relative distances and lighting conditions. Each bar in the figures represents and average of the data taken during the particular test. The small bar over the main bar represents the standard deviation of the data. Figure 65 show the individual components (x, y and z) of the translation error. The radial, angular components of the translation error in the target frame (as described in Figure 66) are presented in Figure 67.

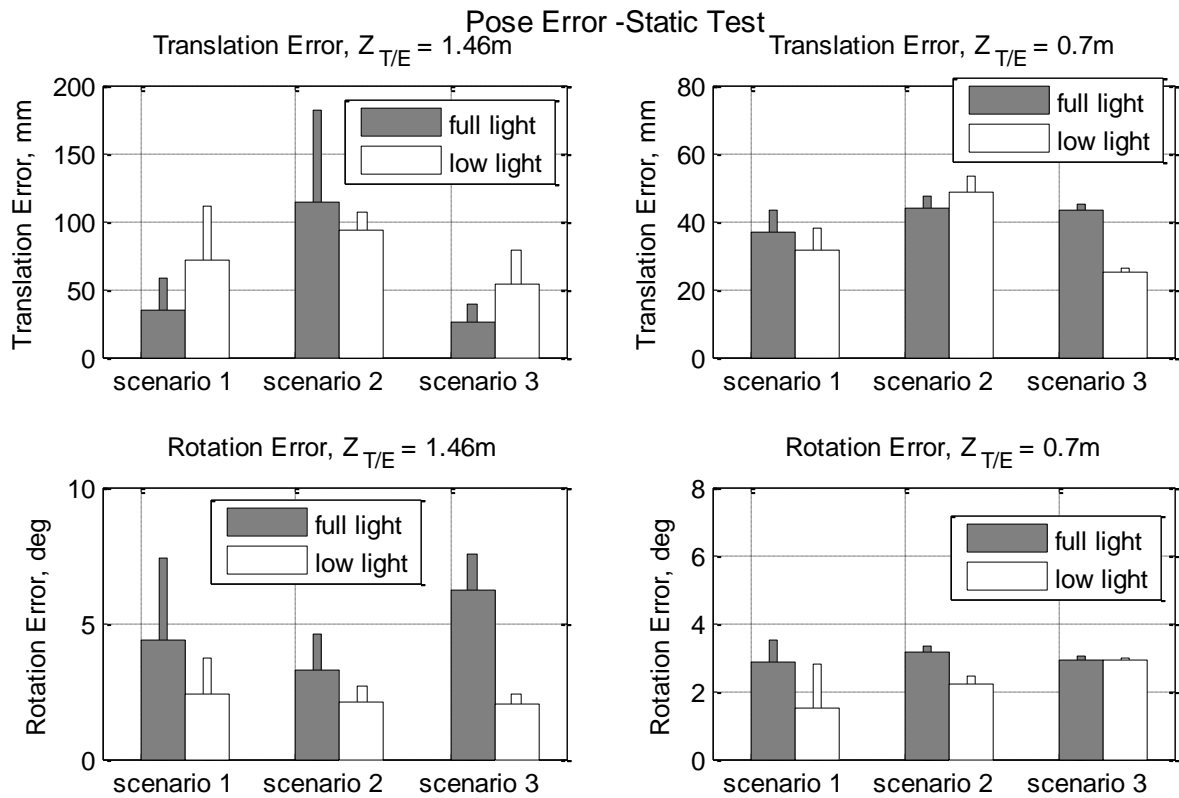


Figure 64. Total pose error during the static experimental tests.

Components of the Translation Error -Static Test

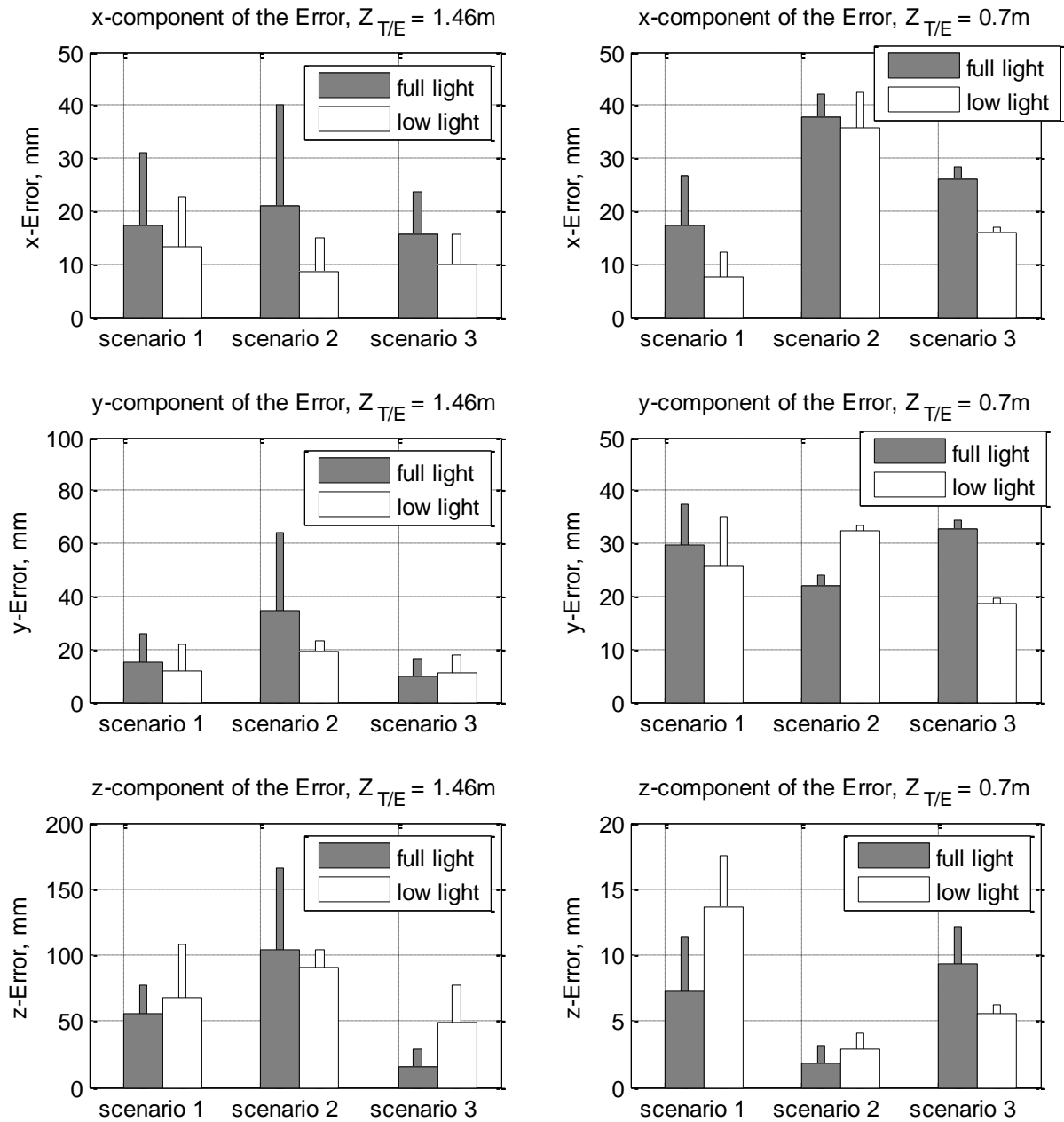


Figure 65. Components of the translation error during the static experimental tests.

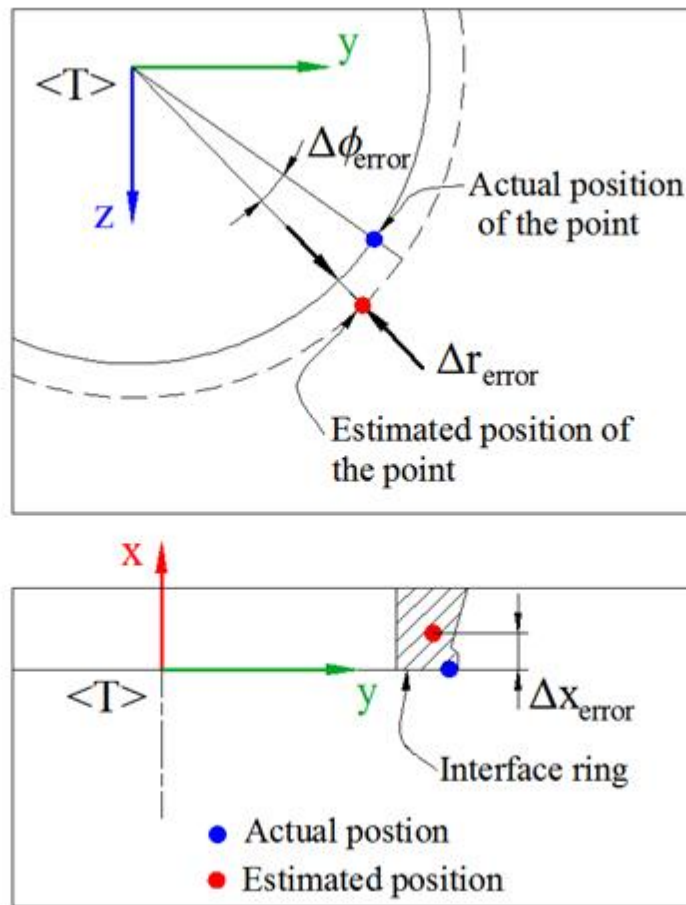


Figure 66. Representation of the radial, angular and axial components of the translation error.

Cylindrical Components of the Translation Error in Target Frame - Static Test

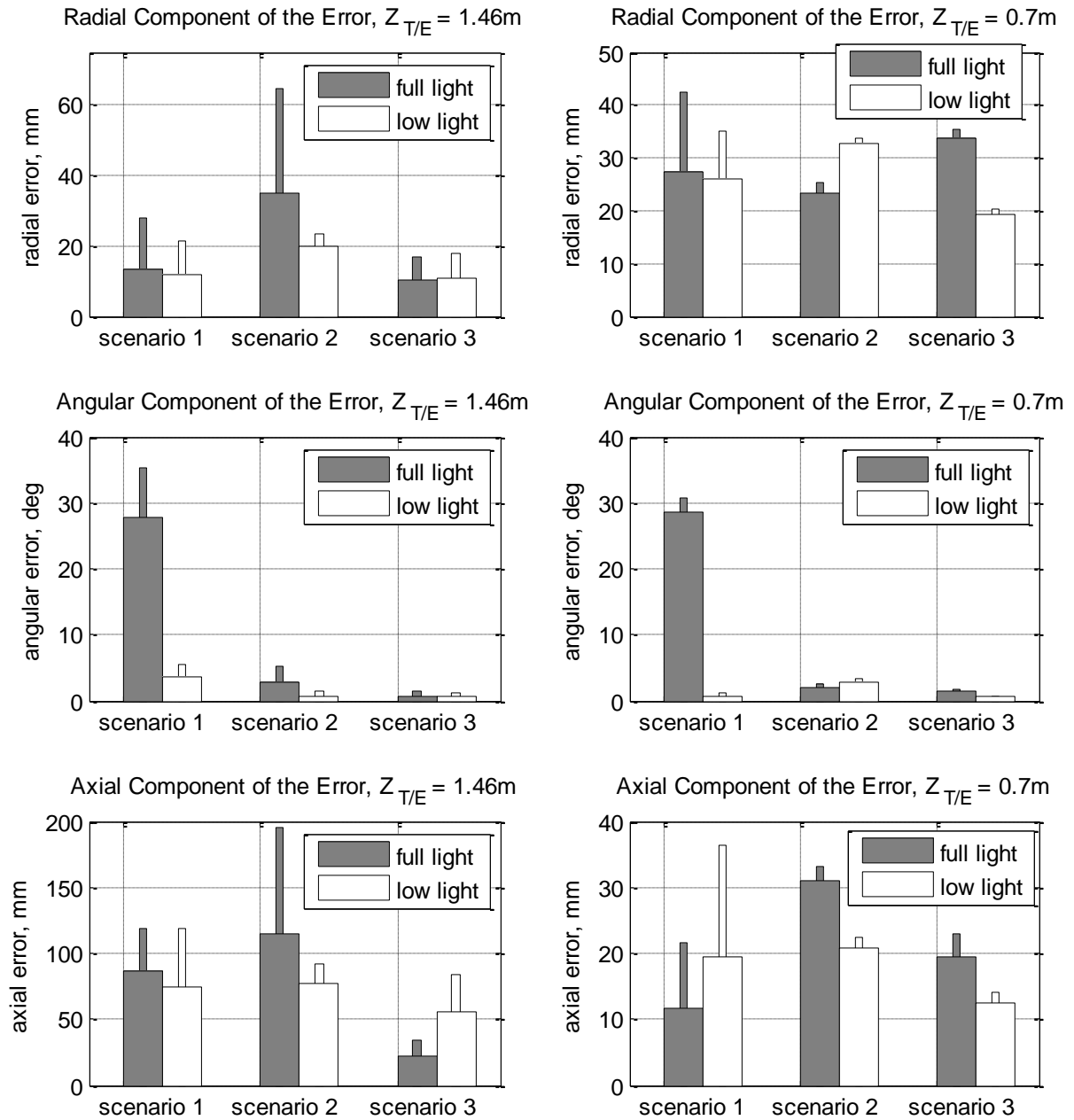


Figure 67. Radial, angular and axial components of the translation error during the static experimental tests.

12.2.2 Effect of Initial Conditions

Figure 68 shows the translation and rotation error for these types of tests. The pose error is plotted against the frame number of the captured video.

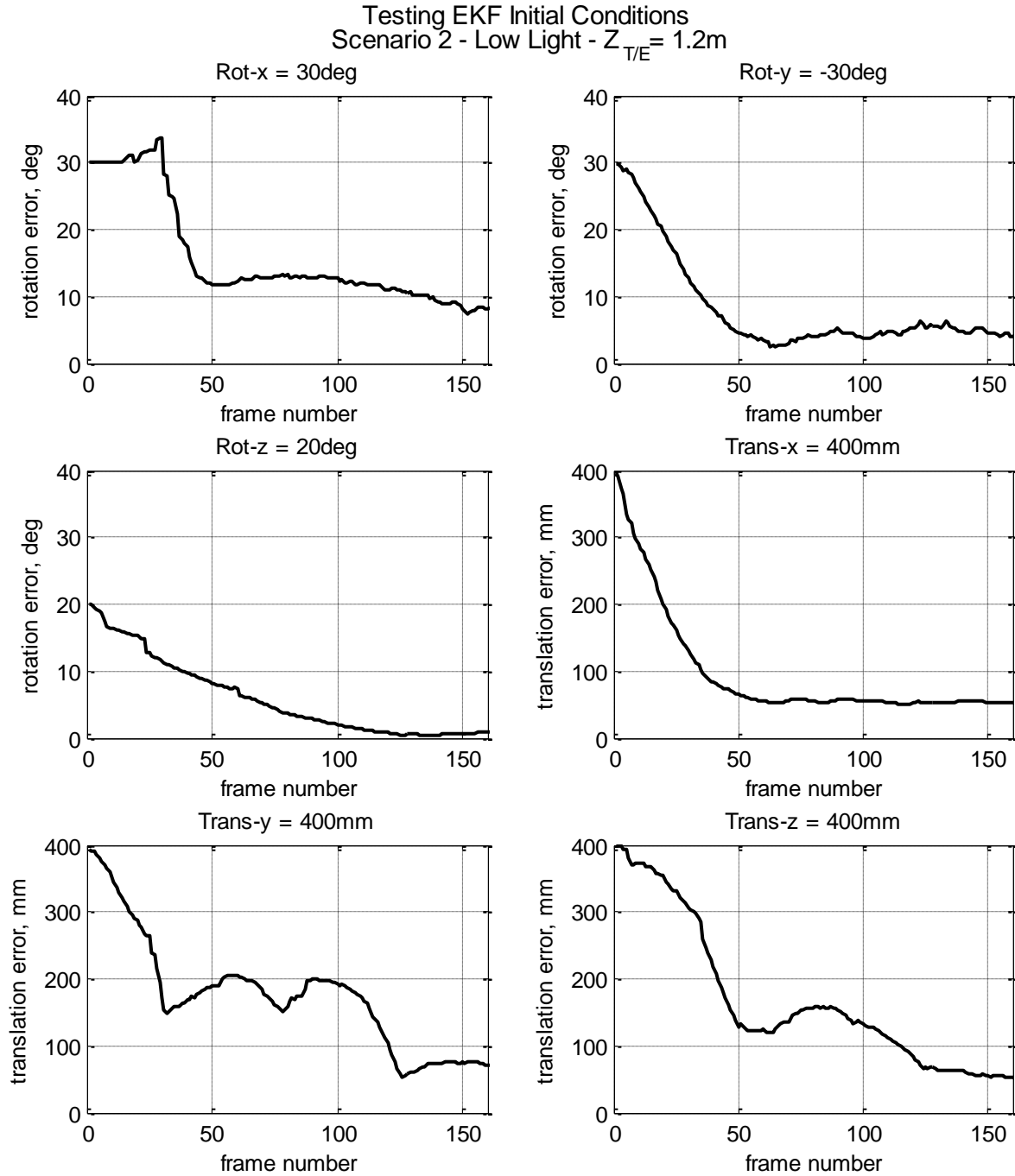


Figure 68. Translation error during the initial condition tests.

12.2.3 Dynamic 6DOF Tests

Figure 71 through Figure 76 show the results for these tests, where the estimated and actual target displacement are shown for each one of the maneuvers. Although the end effector was moved the results are presented as the motion of the target with respect to the target initial position and it is expressed in the target frame, this can be seen in Figure 69. An example is presented in Figure 70, where a rotation of the end effector around its y-axis is performed (left side of the figure). For an observer on the end-effector, it will appear that the target was moved (translated and rotated). The plots shows the components of the target displacement that present a major change when the testing maneuver is performed, as an example when performing a translation along the end effector X-axis, the y-component of the target displacement is plotted (these can be seen from the relative position of the target w.r.t to the end effector). Note that when performing rotations of the end effector target translation are observed from the end effector, as can be seen in the plots.

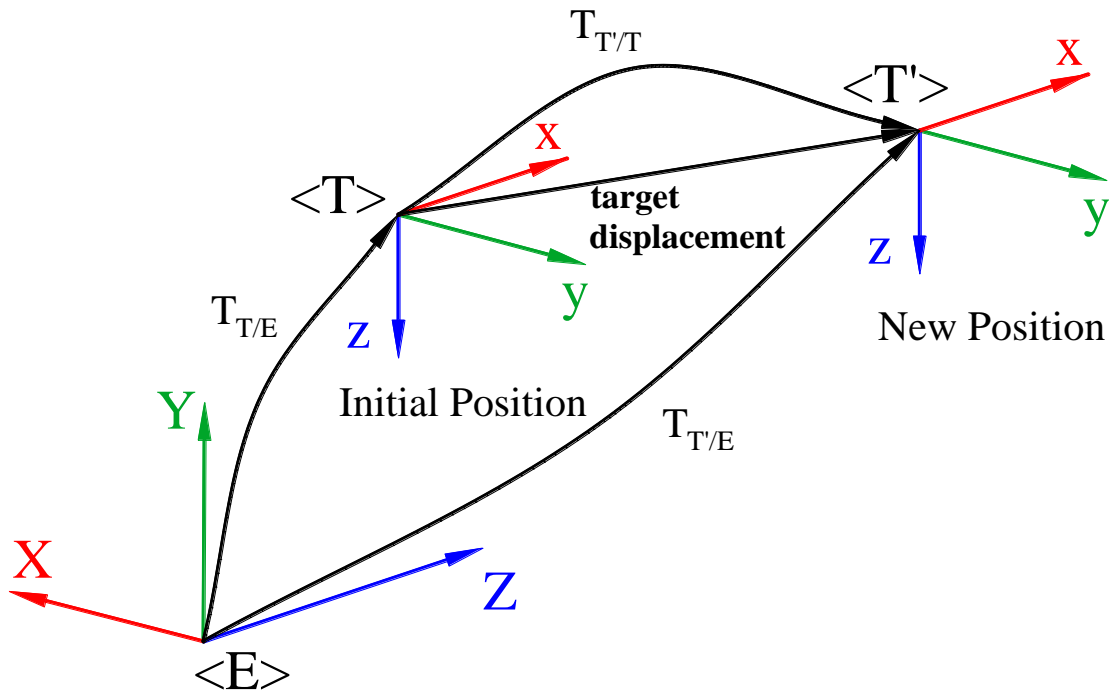


Figure 69. Target displacement w.r.t the initial position

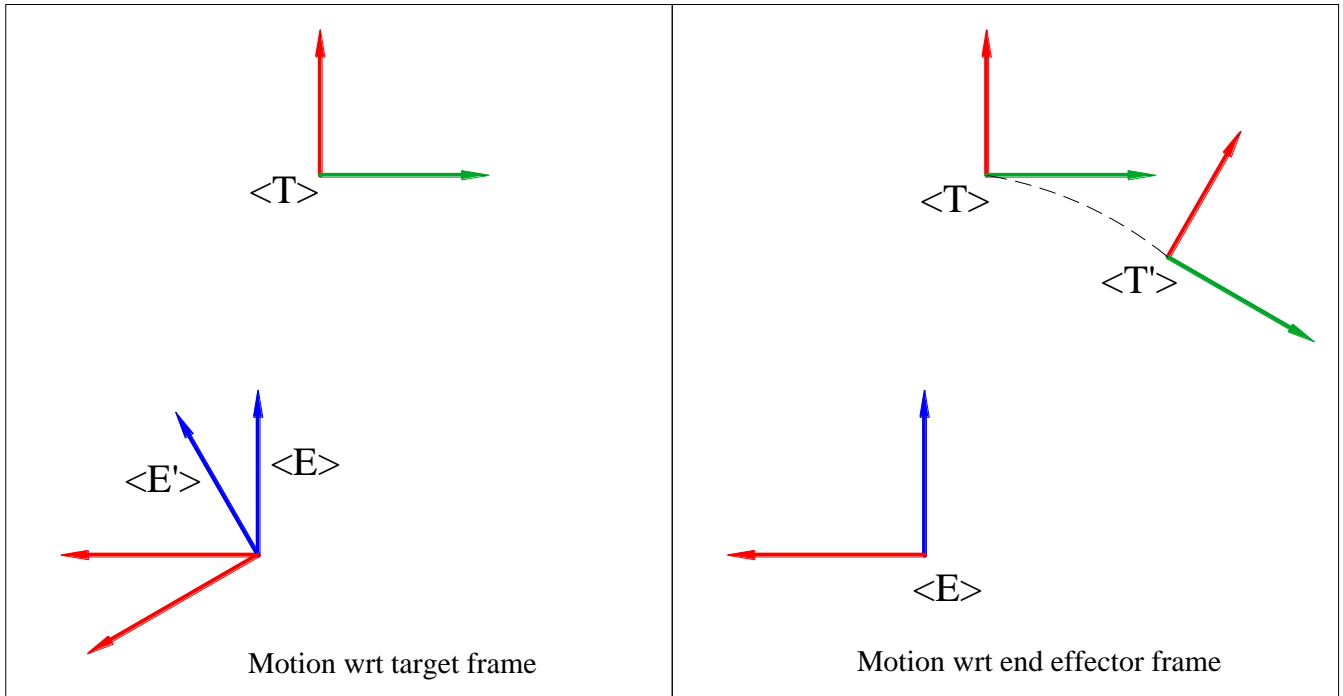


Figure 70. Comparison of the relative motion w.r.t target frame (left) and w.r.t the end-effector frame (right)

Screenshots of the camera views at the initial and extreme positions of each maneuver are presented in Table 14 through Table 17 for scenario 3 at the two lighting conditions. Table 18 shows the screenshot of the camera views at the initial position for scenarios 1 and 2 and for the two lighting conditions. The screenshots also show the detected and reference features.

Motion of the Target wrt Target Initial Position
 Scenario 1 - Full Light - $Z_{T/E} = 1.2\text{m}$

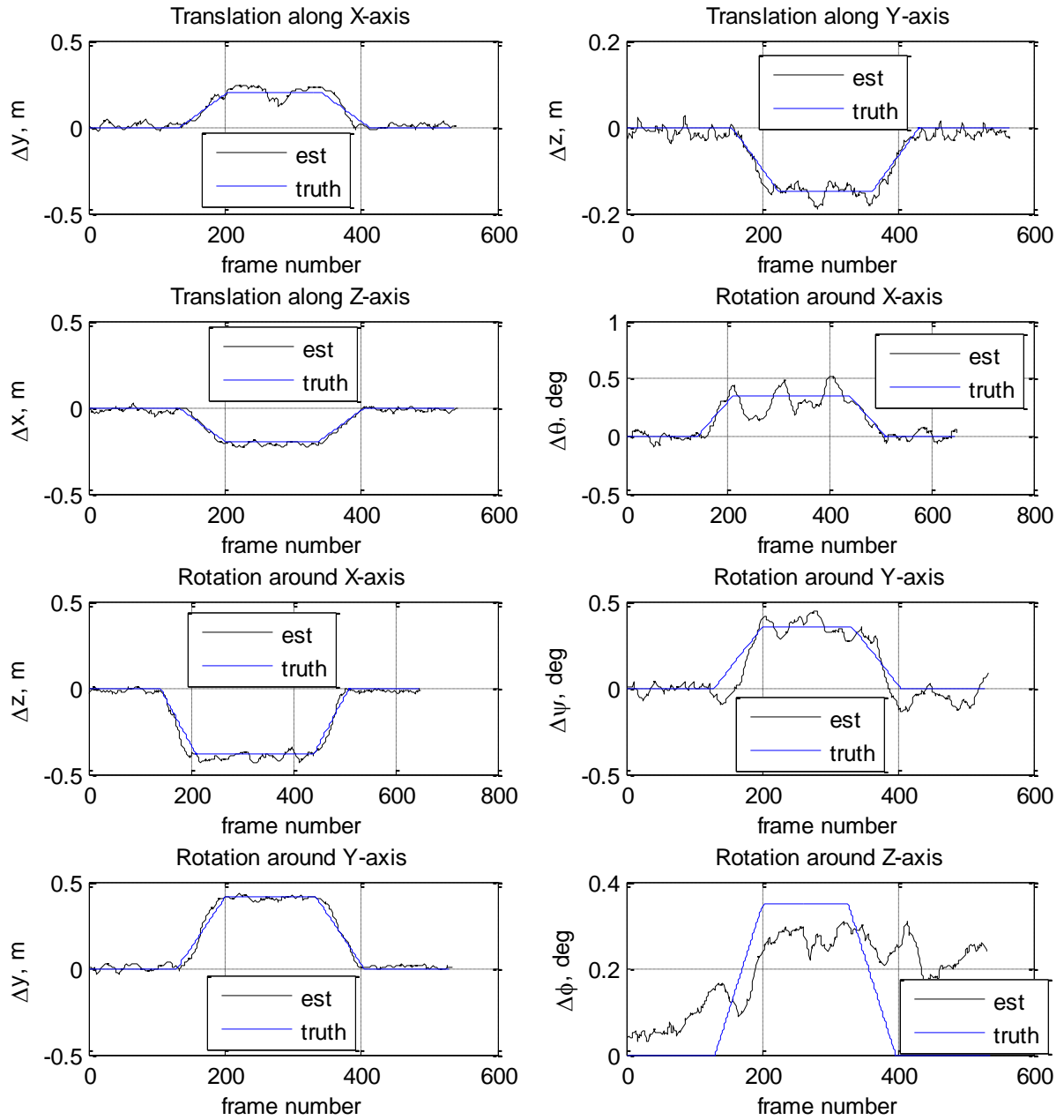


Figure 71. Experimental tracking tests. Scenario 1 – Full light condition.

Motion of the Target wrt Target Initial Position
 Scenario 2 - Full Light - $Z_{T/E} = 1.2\text{m}$

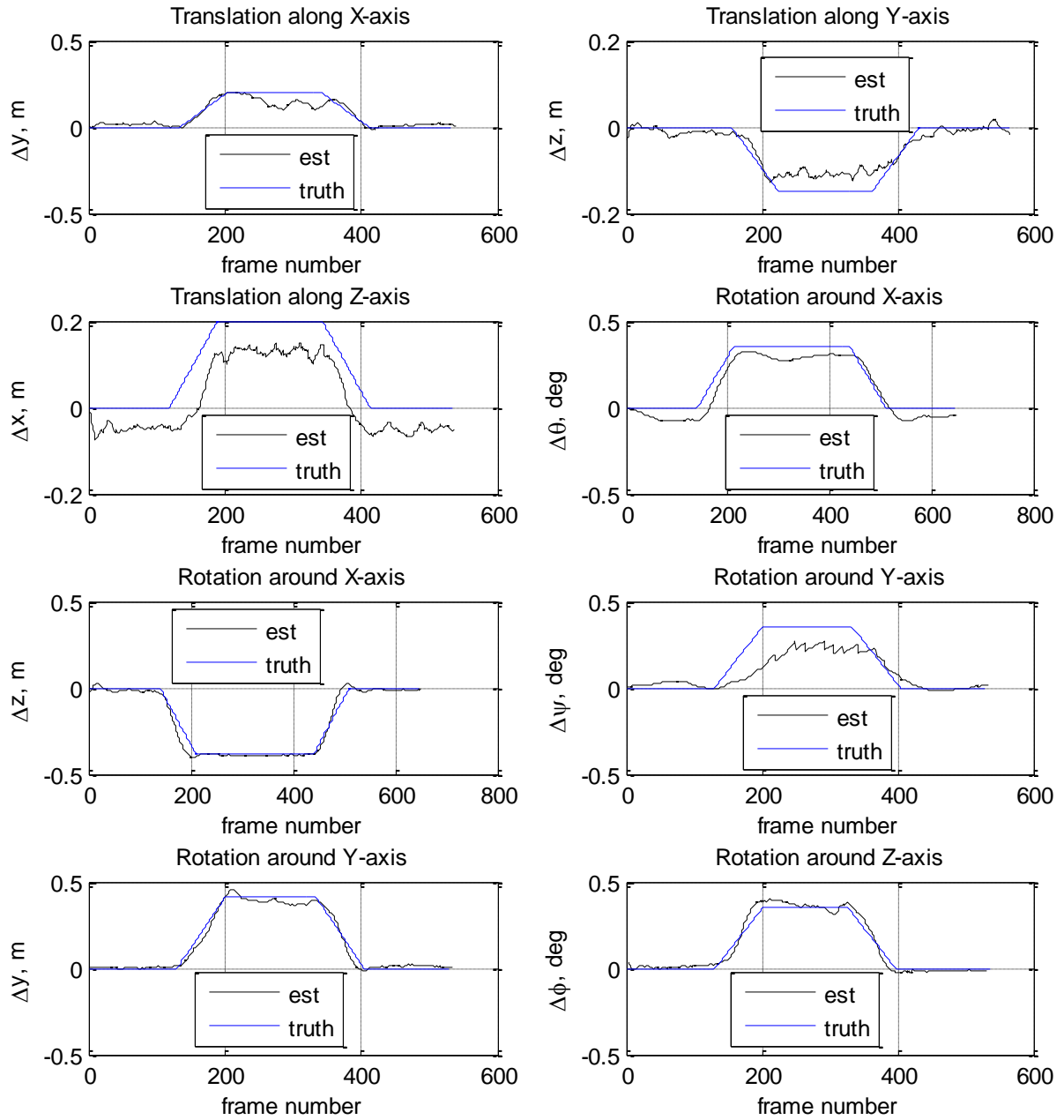


Figure 72. Experimental tracking tests. Scenario 2 – Full light condition.

Motion of the Target wrt Target Initial Position
 Scenario 3 - Full Light - $Z_{T/E} = 1.2\text{m}$

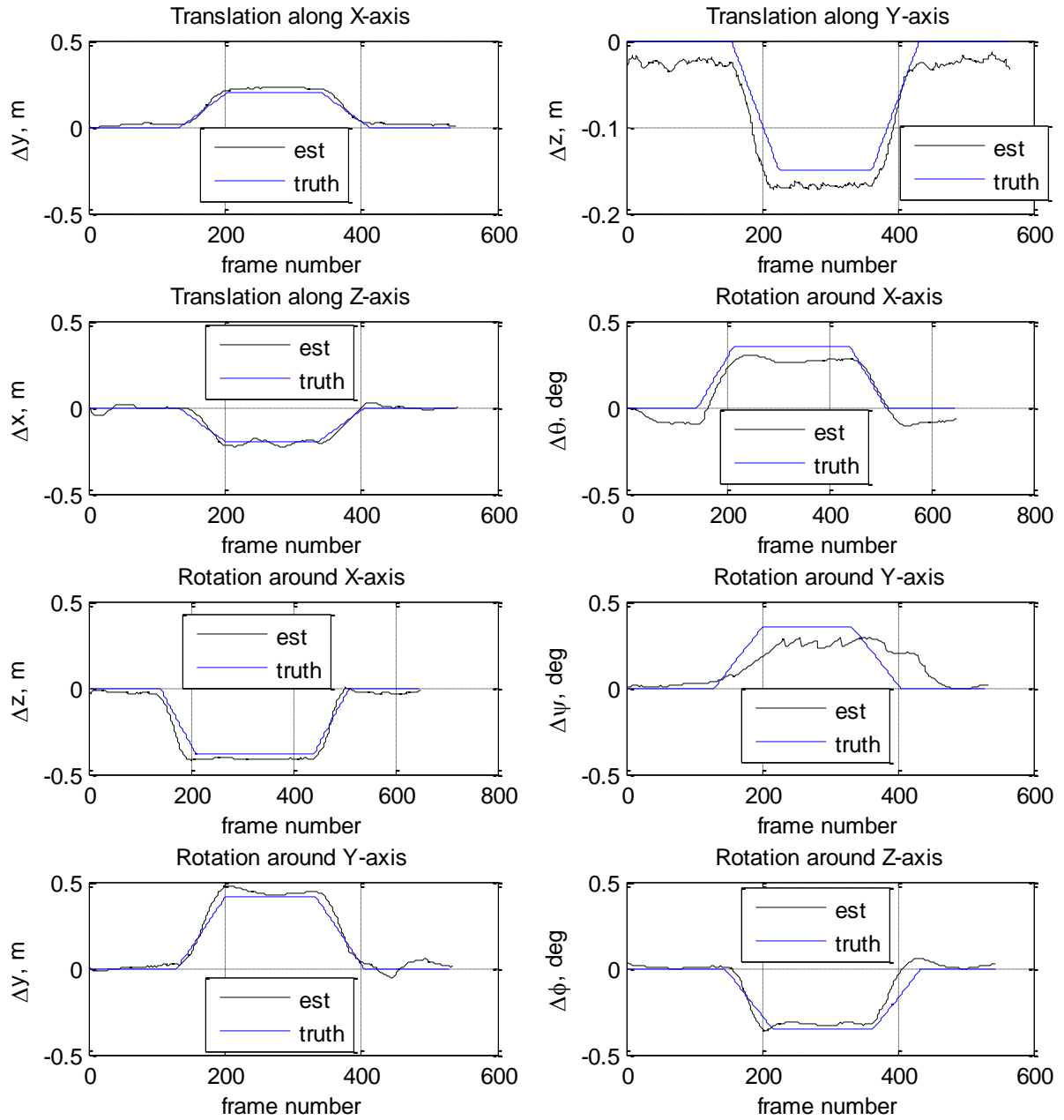


Figure 73. Experimental tracking tests. Scenario 3 – Full light condition.

Motion of the Target wrt Target Initial Position
 Scenario 1 - Low Light - $Z_{T/E} = 1.2\text{m}$

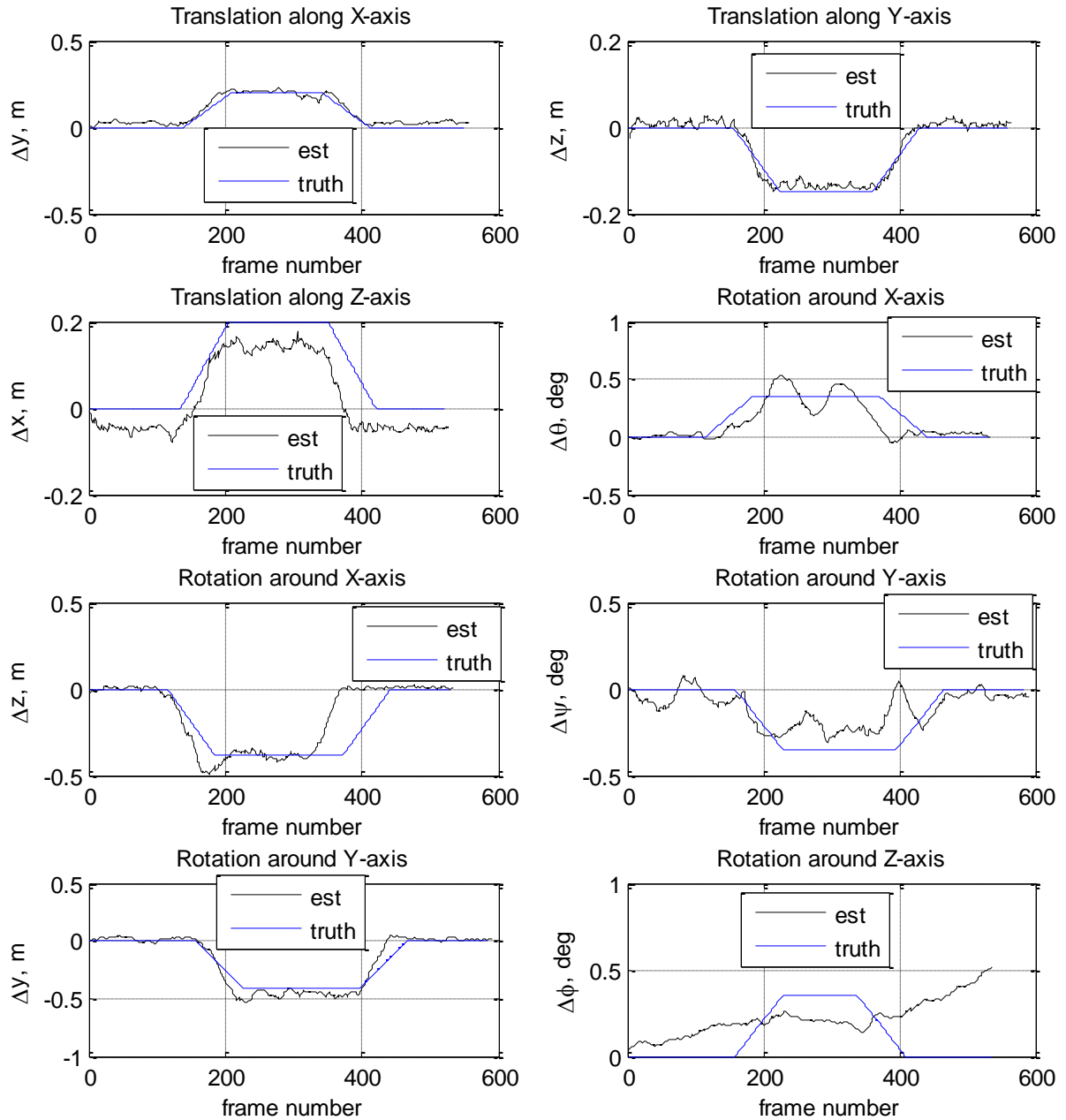


Figure 74. Experimental tracking tests. Scenario 1 – Low light condition.

Motion of the Target wrt Target Initial Position
 Scenario 2 - Low Light - $Z_{T/E} = 1.2\text{m}$

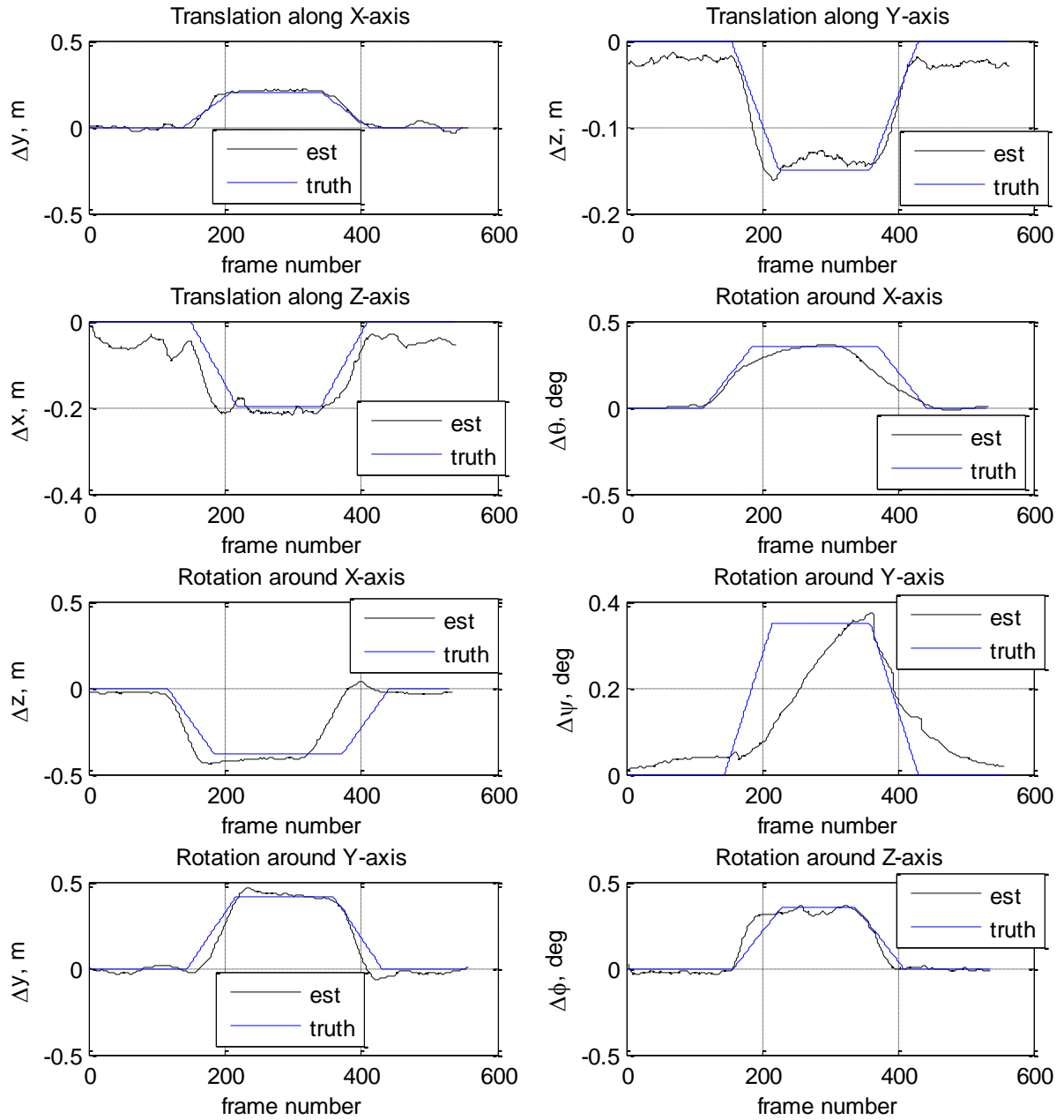


Figure 75. Experimental tracking tests. Scenario 2 – Low light condition.

Motion of the Target wrt Target Initial Position
 Scenario 3 - Low Light - $Z_{T/E} = 1.2\text{m}$

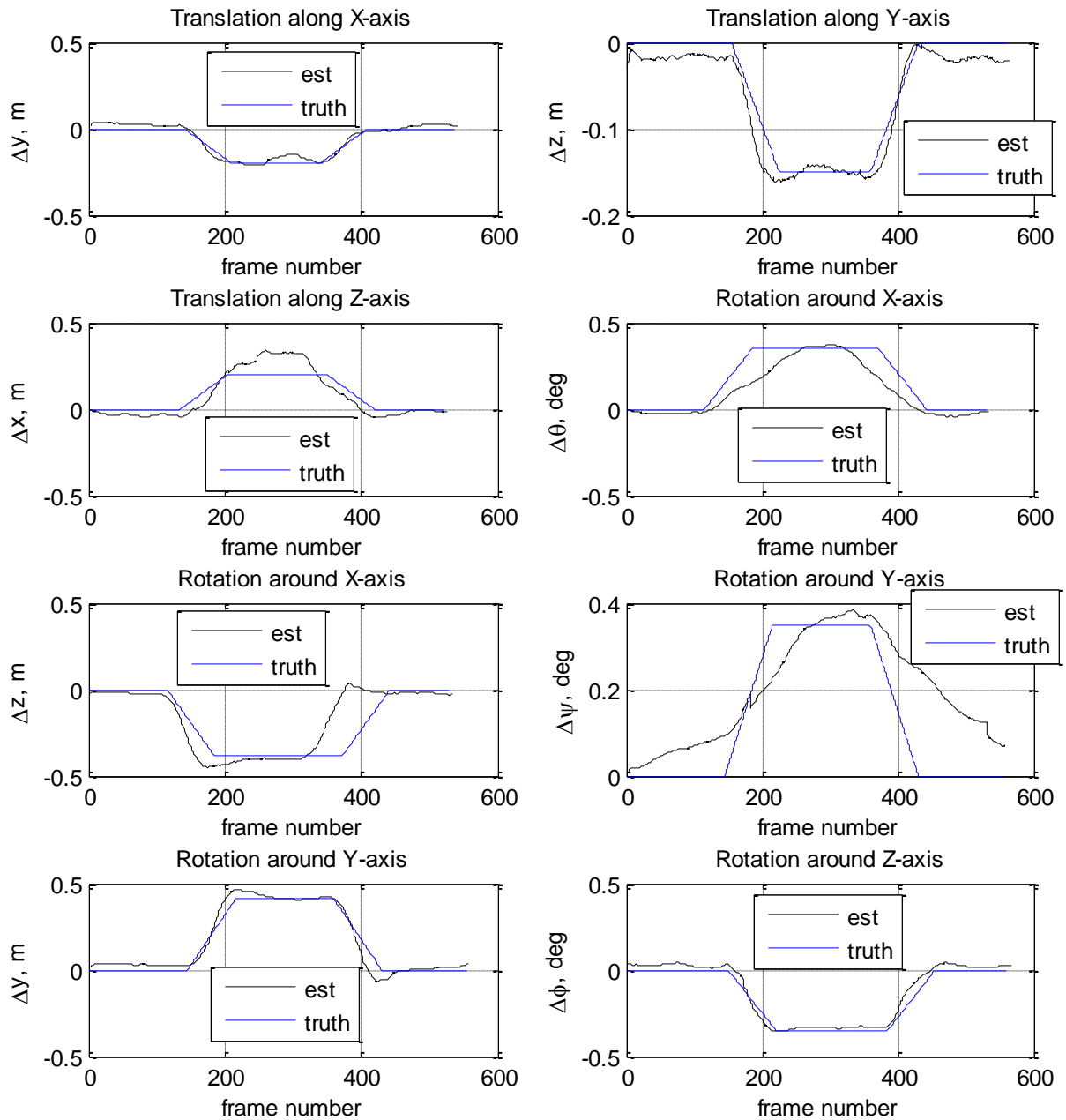


Figure 76. Experimental tracking tests. Scenario 3 – Low light condition.

Table 14. Detected (blue) and reference (red) features at the extreme positions during the rotations of the experimental tracking tests – Full light condition

	Rotations Full Light - Scenario 3 - $Z_{TE} = 1.2m$	
Motion	Camera 1	Camera 2
Home		
RotX		
RotY		
RotZ		

Table 15. Detected (blue) and reference (red) features at the extreme positions during the translation of the experimental tracking tests – Full light condition

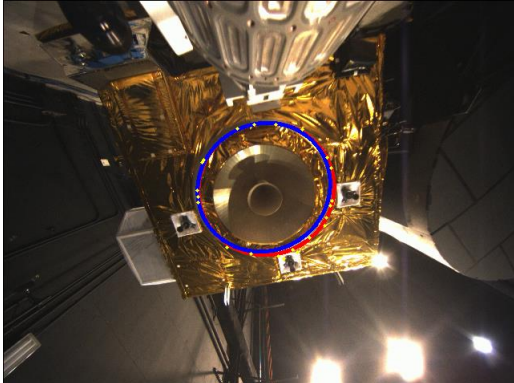
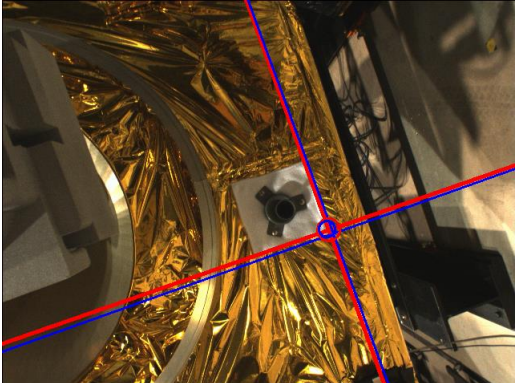
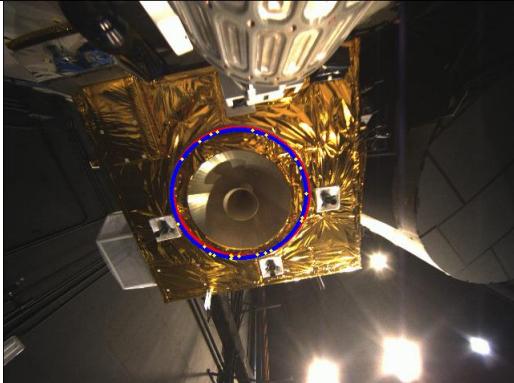
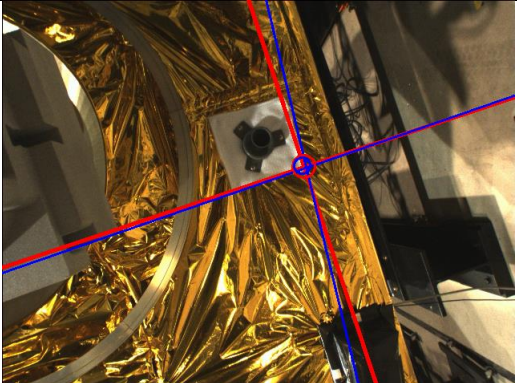
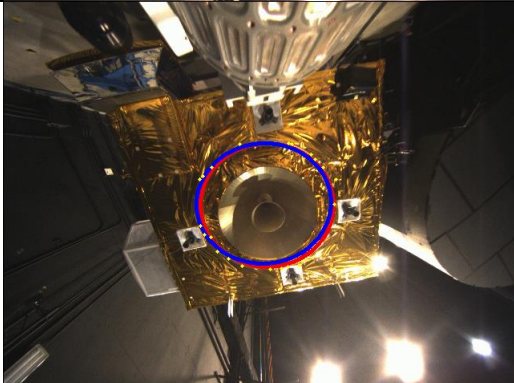
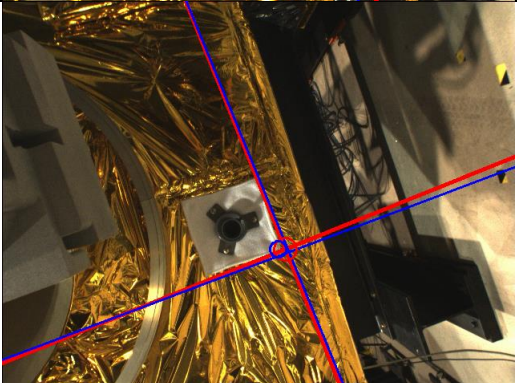
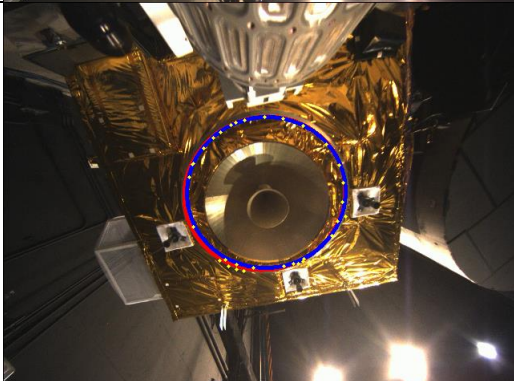
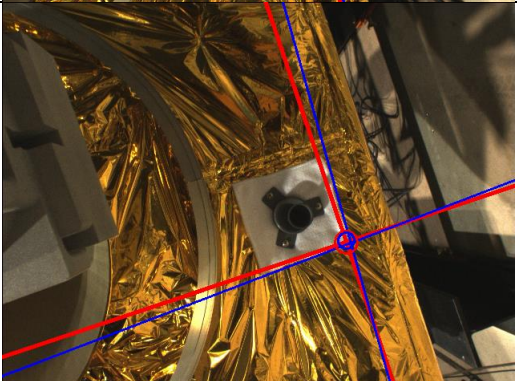
	Translations Full Light - Scenario 3 - $Z_{TE} = 1.2m$	
Motion	Camera 1	Camera 2
Home		
TransX		
TransY		
TransZ		

Table 16. Detected (blue) and reference (red) features at the extreme positions during the rotations of the experimental tracking tests – Low light condition

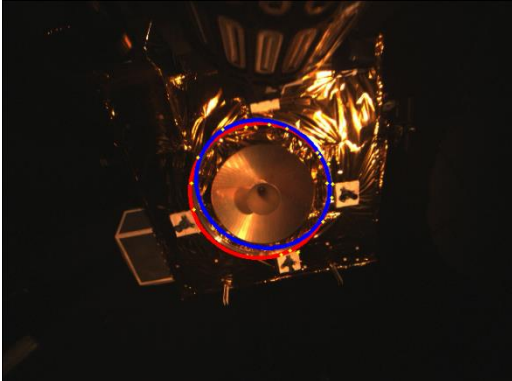
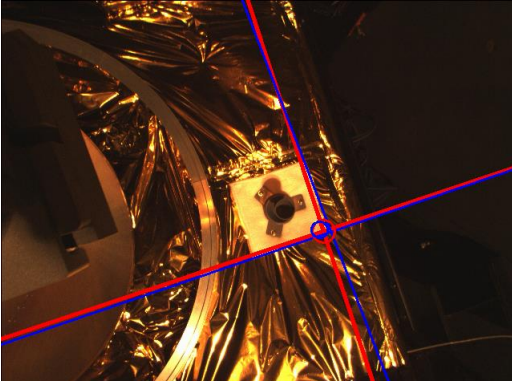
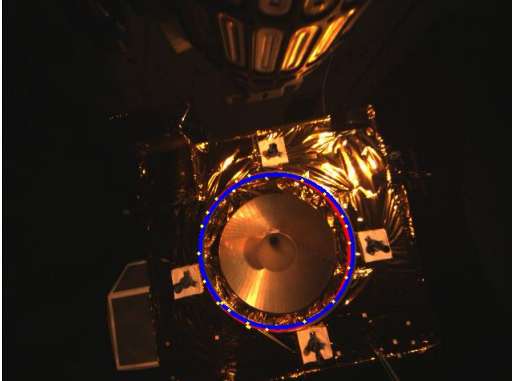
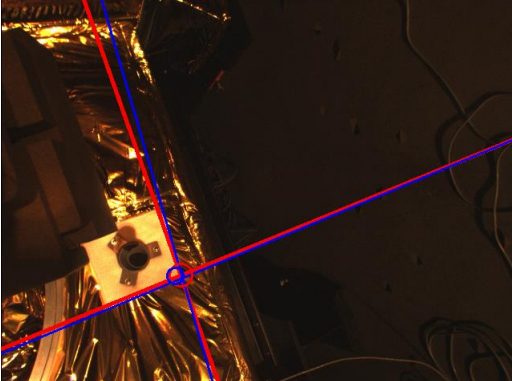
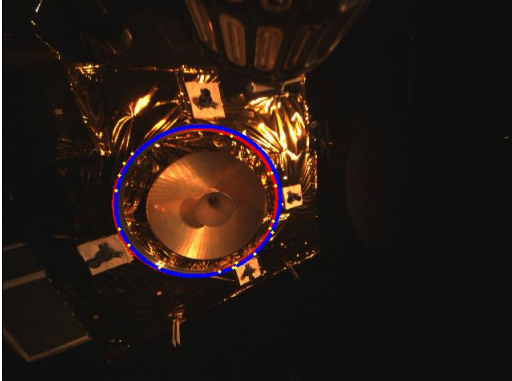
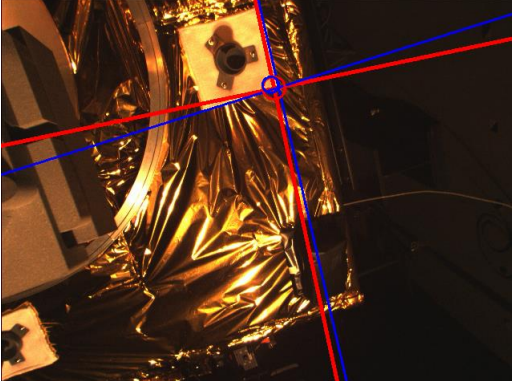
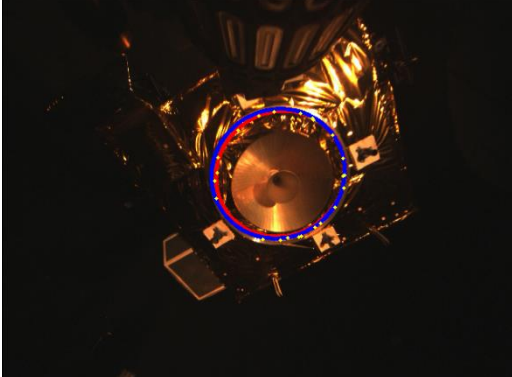
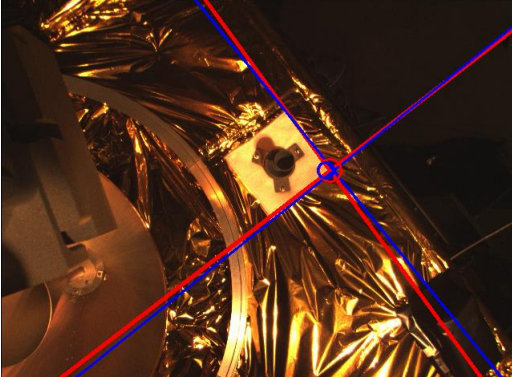
	Rotations Low Light - Scenario 3 - $Z_{TE} = 1.2m$	
Motion	Camera 1	Camera 2
Home		
RotX		
RotY		
RotZ		

Table 17. Detected (blue) and reference (red) features at the extreme positions during the translations of the experimental tracking tests – Low light condition

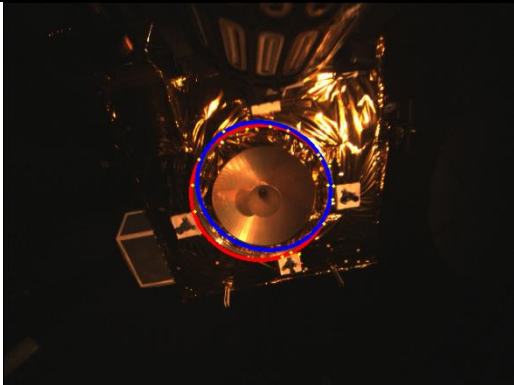
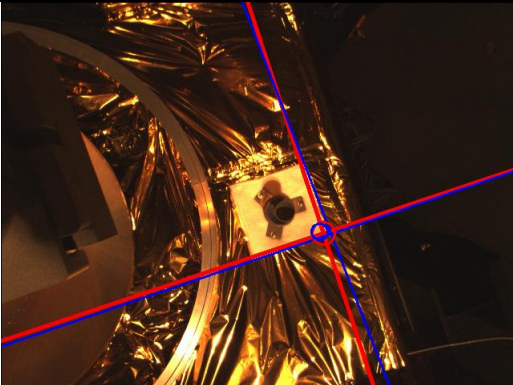
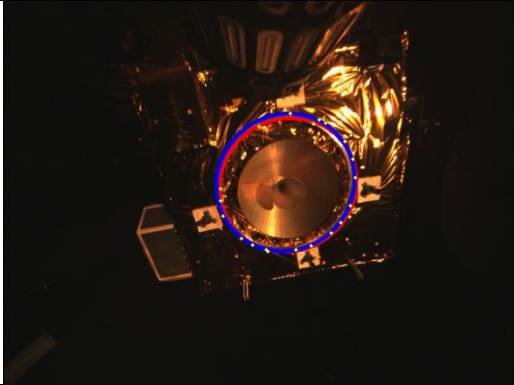
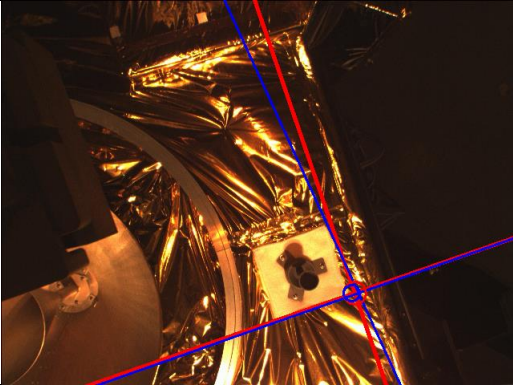
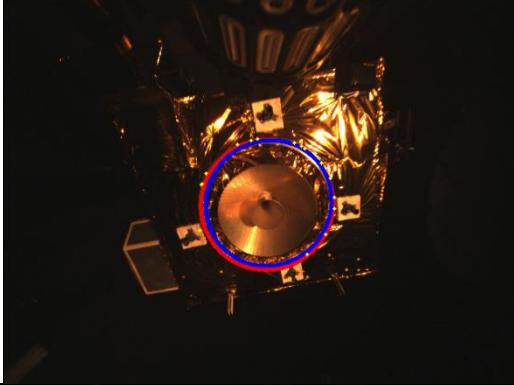
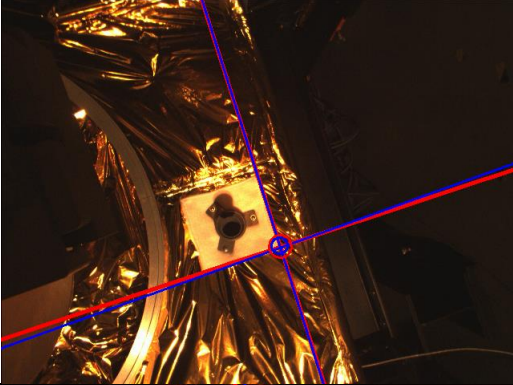
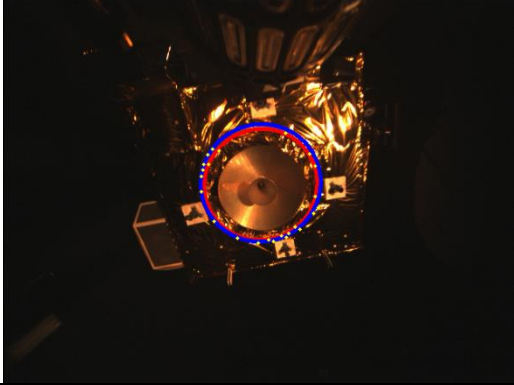
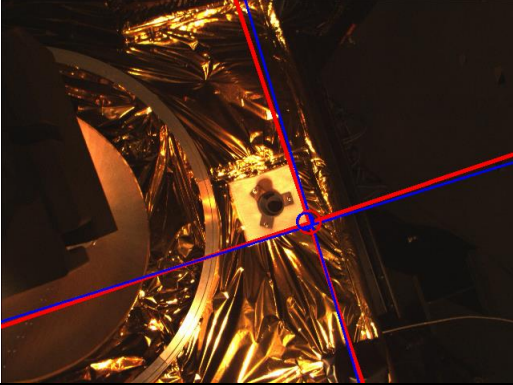
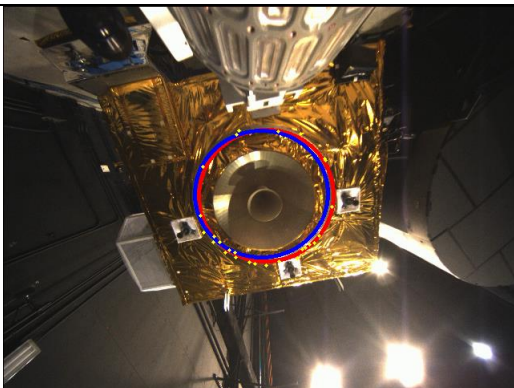
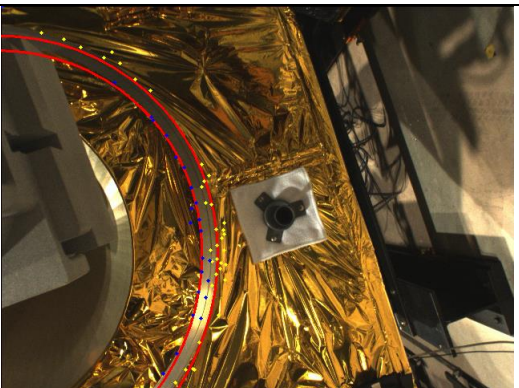
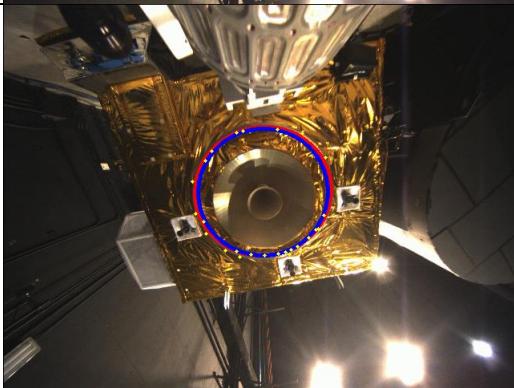
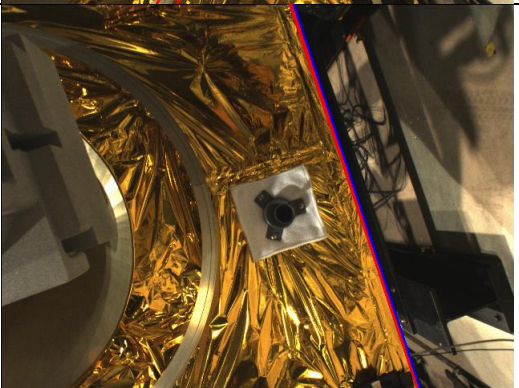
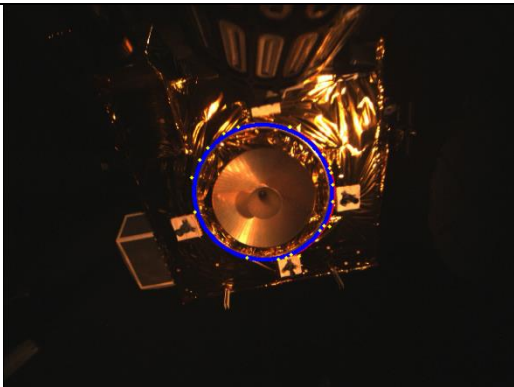
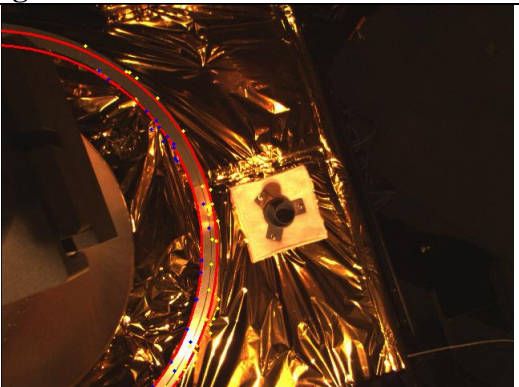
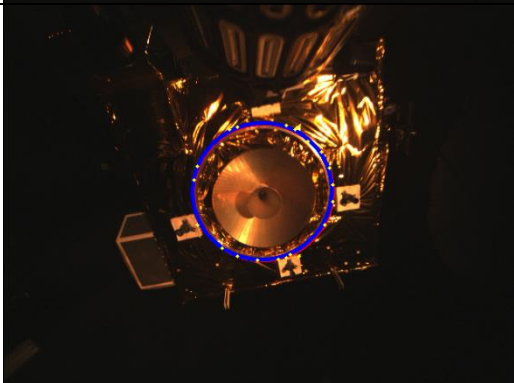
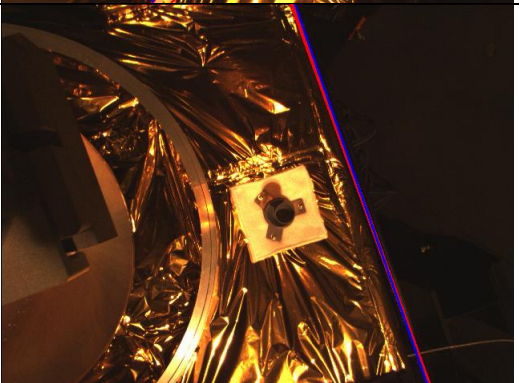
	Translations Low Light - Scenario 3 - $Z_{TE} = 1.2m$	
Motion	Camera 1	Camera 2
Home		
TransX		
TransY		
TransZ		

Table 18. Detected (blue) and reference (red) features at the home positions during the motions of the experimental tracking tests –Scenarios 1 and 2 – Full and low light conditions

		Home Position-Full Light - $Z_{TE} = 1.2m$	
		Camera 1	Camera 2
Scenario1			
Scenario2			
		Home Position-Low Light - $Z_{TE} = 1.2m$	
Scenario1			
Scenario2			

12.2.4 Approach Tests

Figure 77 shows the actual and estimated values of the z-component of the distance of the target w.r.t end-effector during the approach, for the three scenarios. Screenshots of the camera views at the initial and final positions of the approaches are presented in Table 19 through Table 22 for the three scenarios and the two light conditions.

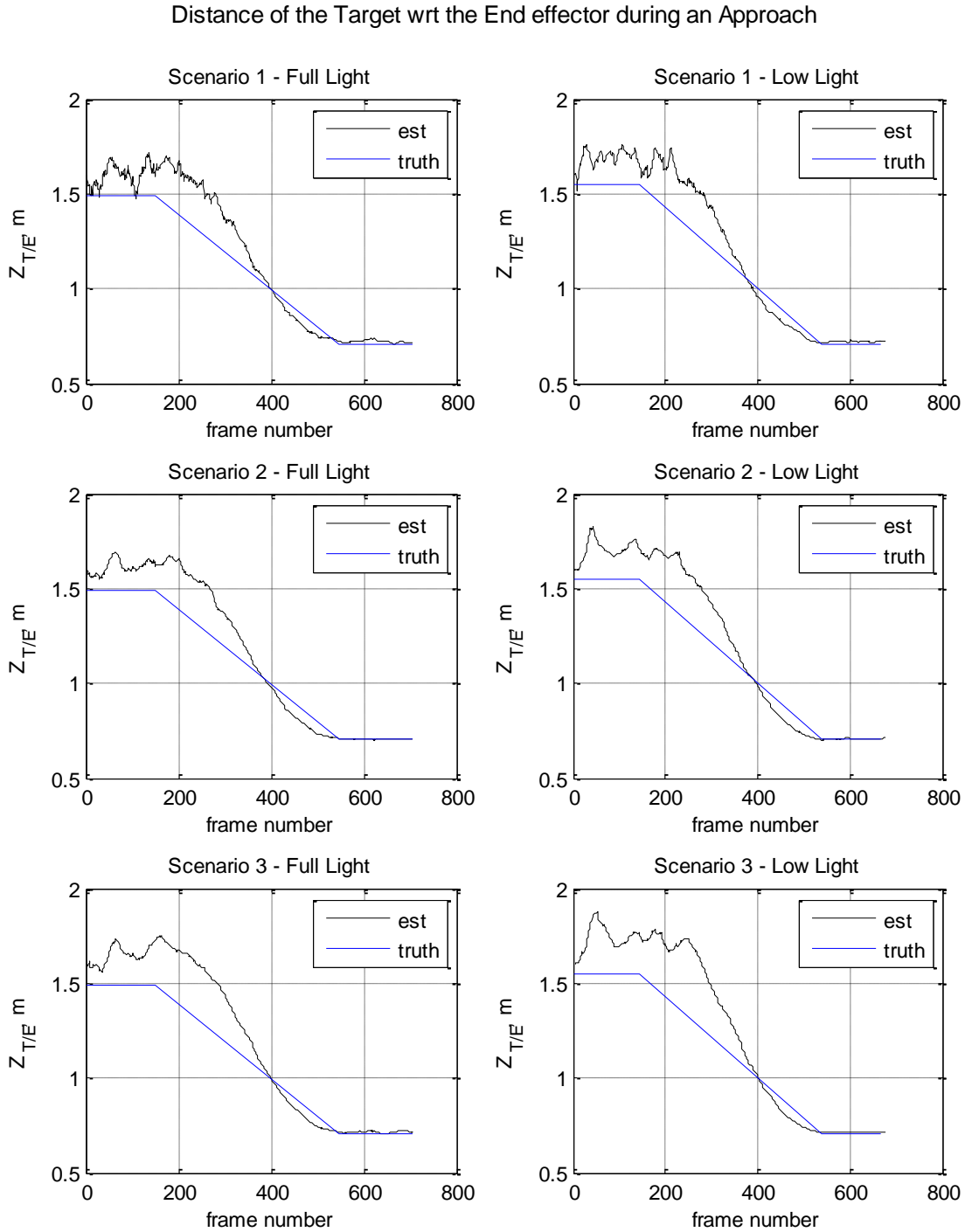


Figure 77. Variation of the z-component of the position of the target w.r.t the end-effector during the approach of the experimental tests

Table 19. Detected (blue) and reference (red) features at the initial positions during the approach of the experimental tests – Full light condition

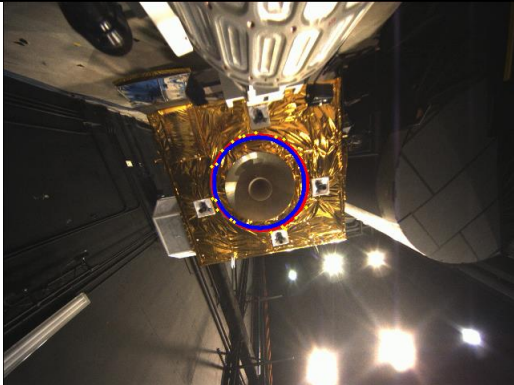
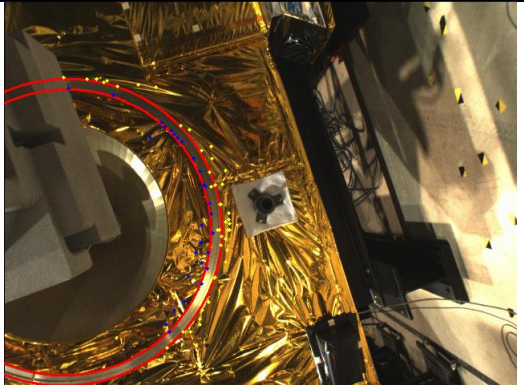
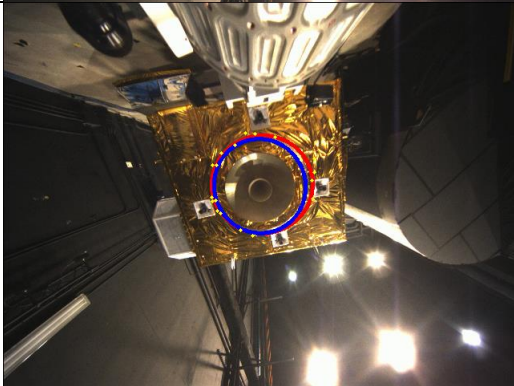
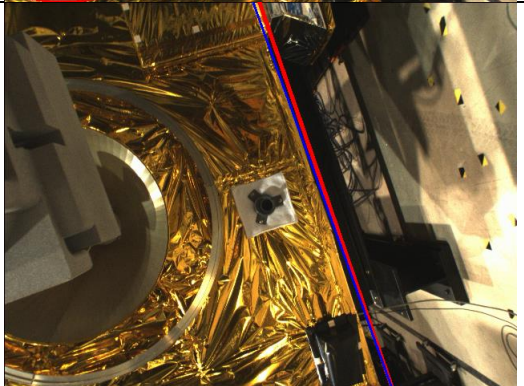
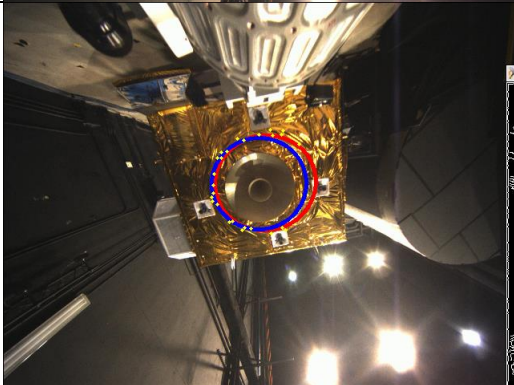
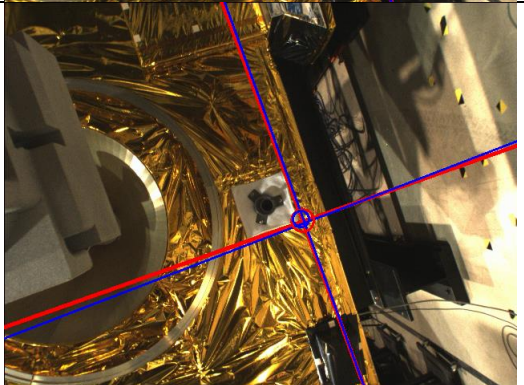
Approach - Initial Position - Full Light - $Z_{TE} = 1.49m$		
	Camera 1	Camera 2
Scenario1		
Scenario2		
Scenario3		

Table 20. Detected (blue) and reference (red) features at the initial positions during the approach of the experimental tests – Low light condition

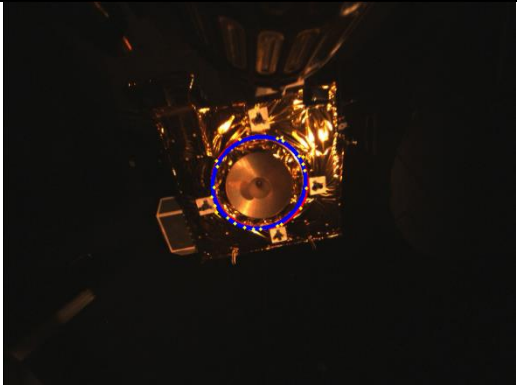
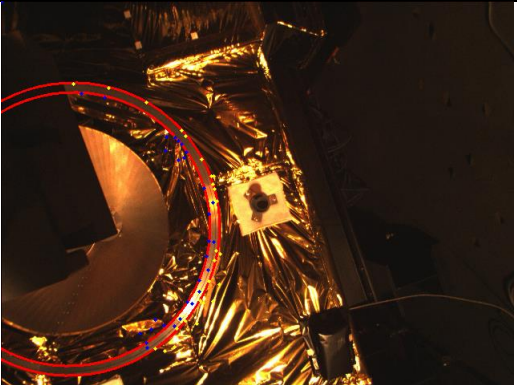
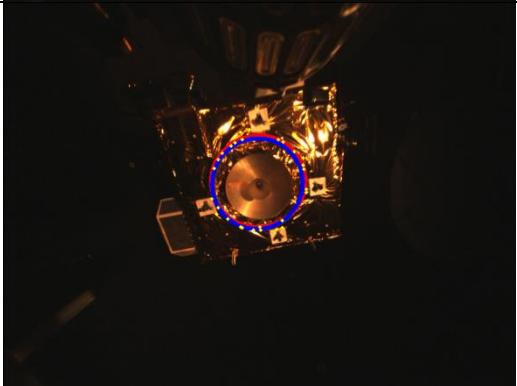
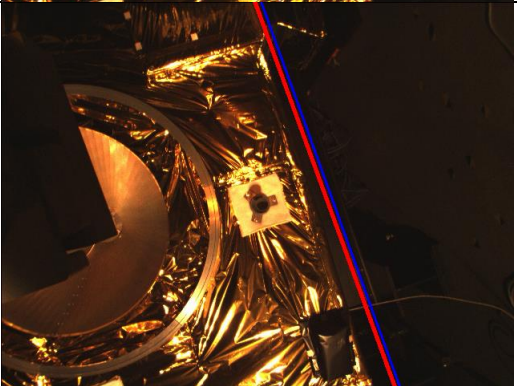
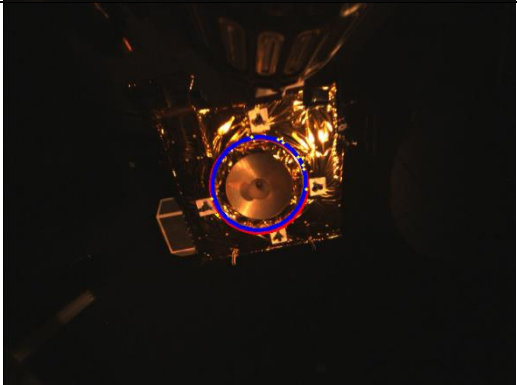
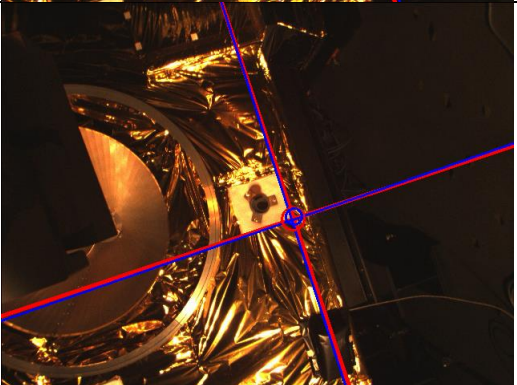
Approach - Initial Position - Low Light - $Z_{TE} = 1.49m$		
	Camera 1	Camera 2
Scenario1		
Scenario2		
Scenario3		

Table 21. Detected (blue) and reference (red) features at the final positions during the approach of the experimental tests – Full light condition

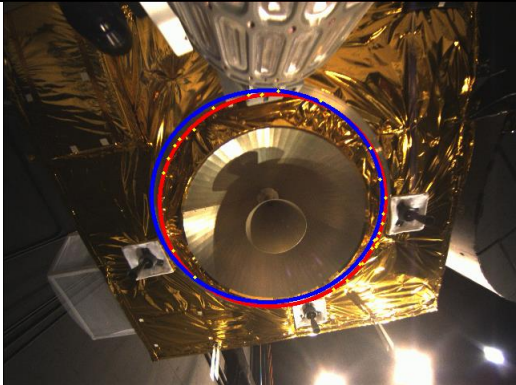
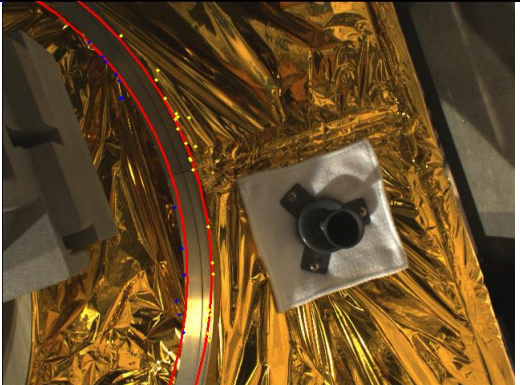
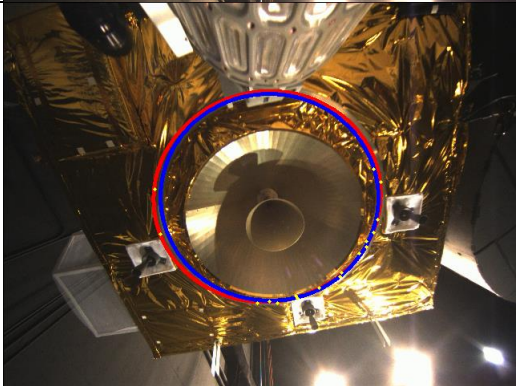
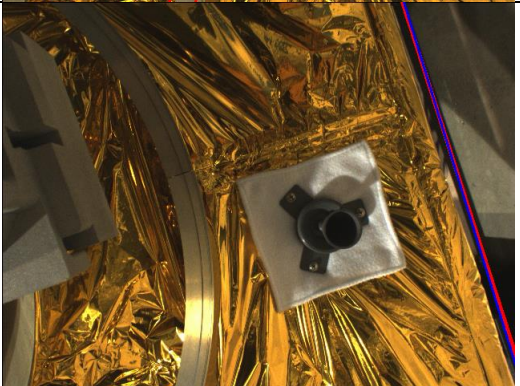
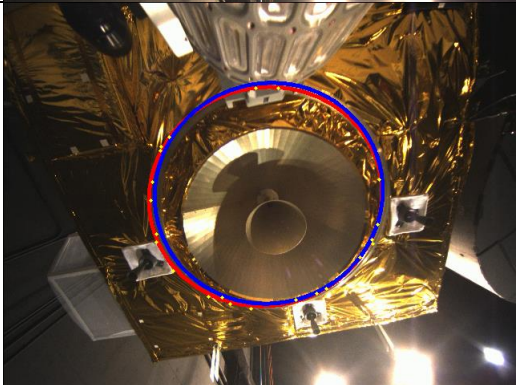
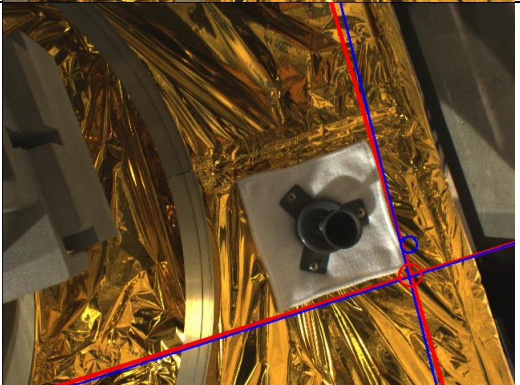
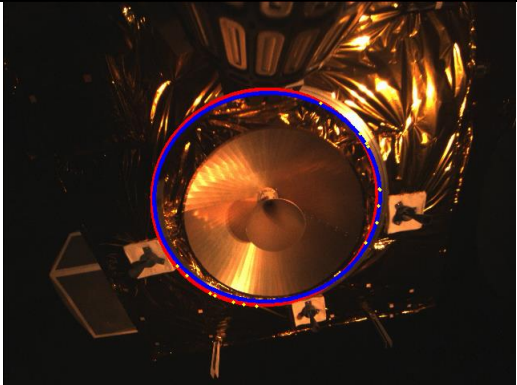
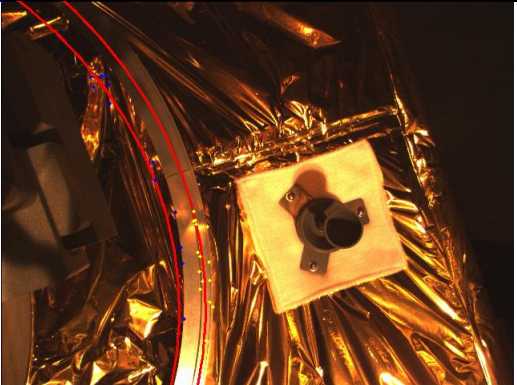
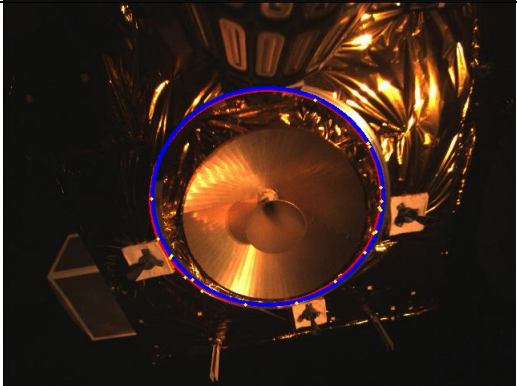
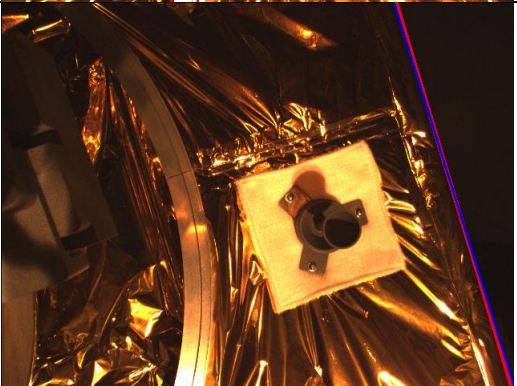
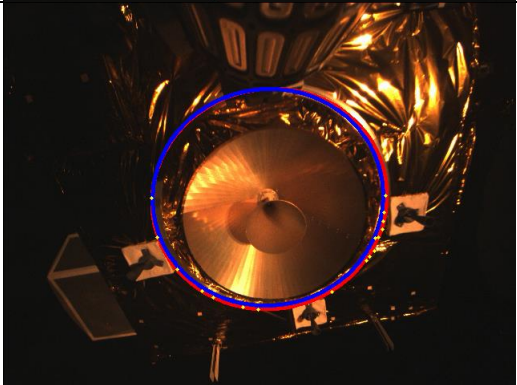
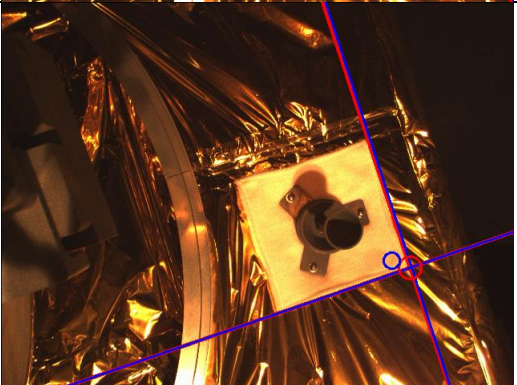
Approach - Final Position - Full Light - $Z_{TE} = 0.7m$		
	Camera 1	Camera 2
Scenario1		
Scenario2		
Scenario3		

Table 22. Detected (blue) and reference (red) features at the final positions during the approach of the experimental tests – Low light condition

Approach - Final Position - Low Light - $Z_{TE} = 0.7m$		
	Camera 1	Camera 2
Scenario1		
Scenario2		
Scenario3		

12.2.5 Processing Time

Table 23 shows the average processing time for each one of the main subsystems of the pose estimation system, when using the implementation presented in Figure 48. The average was taken during an approach (700 measurements approx.) at full light condition and using Scenario 3.

Table 23. Average Processing Time of the Main Subsystems

System	Processing time	% of total time*
Extraction and Detection Camera-1	0.0871 sec	60.8
Extraction and Detection Camera-2	0.0479 sec	33.4
Pose Solver	0.0540 sec	37.7
Circle Pose Calculator	0.0004 sec	0.3
Extended Kalman Filter	0.0018 sec	1.3
Total time	0.1433 sec*	

* Extraction and Detection for Camera-1 and for Camera-2 run in parallel

13 DISCUSSION

This section presents the analysis of the results presented in the previous chapter. The following sections show the discussion of each test result in the same order as they were presented.

13.1 Virtual Tests

13.1.1 Effects of Camera Parameters in the Pose Estimation Error

As expected, the errors in the intrinsic parameters increase the pose estimation error, and larger errors in the parameters imply larger pose error. The extrinsic parameters have a stronger effect on the pose error. The error in the camera-1 parameters has a stronger effect than the ones of camera-2. This is expected since the ellipse in the image of Camera-1 is used by the pose estimation and the circle pose calculator to find pose, in other words, the ellipse provides more information to find the pose.

The image distortion also has a larger effect on the camera-1, since the ellipse points are in the outer part of the image, where the radial distortion is larger.

Since the camera parameters are obtained experimentally in a real system and due to the non-linearities present in a machine vision system (such as lens distortion), the measured parameters are subjected to measurement errors. Due to this, it was important to show that the pose estimation system can still work adequately with small errors in these parameters.

13.1.2 Effects of Relative Pose in the Pose Estimation Error

It can be seen from these tests that the pose estimation error varies slightly (less than 5mm and 1deg) with the range of relative positions used. These small changes are expected due to the nonlinear effects in the system, especially the discretization of the shape of the features by the image pixels and the dependence of the estimated pose on the shape of the features. Although these tests were performed using images without noise or distortions, they show that the mathematical model of the system is robust to these changes.

13.1.3 Dynamic 6DOF Tests

These tests show that the system is capable of tracking the target adequately for each one of the maneuvers performed, the detected features always match the features of interest. Some differences can be seen between the estimated and the actual pose. Since these tests were performed using the CAD model and the relative distance was 1.2m, the system is not able to distinguish between the inner and outer edge of the interface ring, which generates pose errors (see Figure 78).

It can be seen that there are some small occlusions of the target features by the end effector and the heat shield of the apogee thruster, but they do not have any considerably influence in the detection of the features.

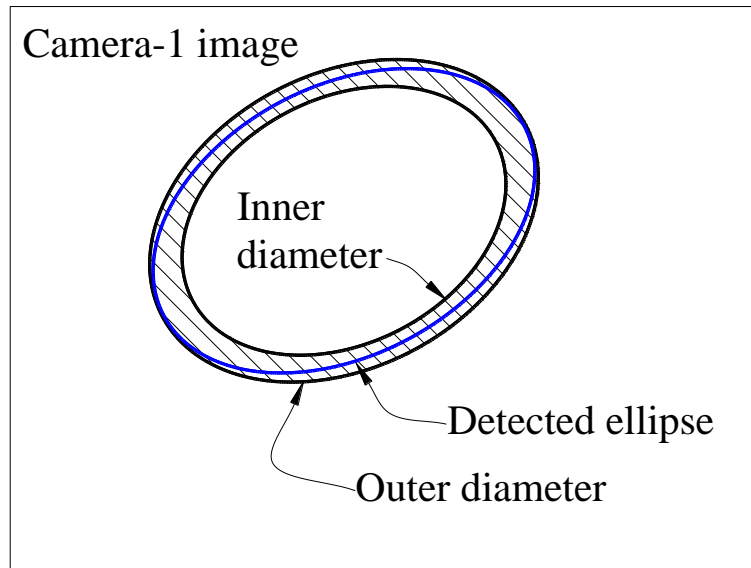


Figure 78. Ring inner and outer diameter miss-detection

13.1.4 Approach Tests

The pose error changes during an approach, being larger when the system is further away from the target, this can be attributed to the inability of the system to identify the inner and outer diameter of the interface ring and the image noise, since the test were performed using the CAD model.

The error when using the uncalibrated camera is larger (as expected) that when using the calibrated camera, and the error is amplified the further away the end effector is from the target.

For each scenario, the translation error is of the same order of magnitude and has the same trend w.r.t to the relative distance.

When comparing the system using the CAD model and the wireframe model it can be seen that when using the wireframe model the translation error is smaller and it is almost constant during the approach. This is due to the larger noise of the CAD model.

For scenario 3 the larger component of the translation is the y-component. The noisier component is the z-component which is related to the depth, and it is directly related to the size of the detected ellipse, which changes considerably due to the noise of the CAD model.

When using the base of the heat shield a less noisy and more accurate estimated pose is obtained. This is due to a more stable size of the detected ellipse. As mentioned before the inner and outer diameter of the

interface ring can be confused and the CAD model has a textured component around the ring to simulate the MLI which generates noise that affects the ellipse detector. When using the wireframe model-1 (the interface ring is represented as a single circle) a more accurate estimation is obtained than when using all the models, and the error tends to be constant during the approach.

13.2 Experimental Testing

13.2.1 Tests for Static Pose Error

The pose estimation error changes with the relative distance, the light condition and the scenario. As a general trend it can be seen that the pose error, specially the translation error, is smaller when the system is closer to the target. As in the case of the virtual simulation, this can be attributed to the discretization of the shape of the features by the image pixels (less pixels are used to represent a feature when the distance is larger) which means that the detected features can be different from the features in the image corresponding to the actual target, the inability to distinguish between the inner and outer diameter of the ring, and the amplification of the camera calibration errors, as was presented with virtual simulations and the uncalibrated camera.

In almost all the cases the low light condition have a smaller rotation error and smaller standard deviation. When using the low light conditions it was observed that the MLI reflections were reduced and less features were being extracted. The standard deviation of the error are in the majority of the cases small when compared to the mean value of the error.

It is difficult to observe general trends since each scenario is equivalent to a different type of sensor in the EKF, i.e. the noise covariance are different, the amount of information that it provide is different (number of degrees of freedom). The EKF was also using a different noise matrix for each case. Nevertheless it can be said that the system is robust to the variations (target proximity to the target, light conditions and scenarios) since it stable and tracks the features correctly. Each scenario has its own advantages in terms of accuracy and robustness to initial condition, tracking, etc., this will be discussed in Section 13.3.2.

It is important to present the error in the target radial, angular and axial component in order to have a better idea of the magnitude of the error, especially when the interface ring is used as the feature to grasp. It can be seen that the radial error is between 20 and 30mm when the system is at 0.7 from the target. If this were the grasping position, the capture tool have to be able to handle a deviation of that magnitude, in addition to the axial error which is between 10 and 30mm.

13.2.2 Effect of Initial Conditions

When initializing the system with a wrong initial condition, the ellipse corresponding to the interface ring was detected immediately, and it took around 50 iterations (or image frames) for the system to reach the correct state where all the detected features were matching correctly their target counterpart. Scenario-2 was used for these tests since its larger features (ring and panel border) were more robust to these variations. When using Scenario-3, although the ring was being detected correctly, the pad borders and corner were being confused with false features in the MLI. When using Scenario-1 the ring was being detected correctly in camera-1, but the system was sometimes using the wrong normal and/or confusing the ring on camera-2 with MLI noise.

13.2.3 Dynamic 6DOF Tests

During these tests the system was always tracking correctly the corresponding features for each case, and it never lost track, in some cases with small occlusions by the heat shield and the apogee thruster. The system returned correctly to the initial condition after the corresponding maneuver was performed. There were some delays in the tracking, especially when performing rotations, they can be attributed to the system incapability to distinguish between the inner and the outer diameter of the ring (Figure 78). It can be observed that when using Scenario-1 the system was unable to track rotations around the target x-axis when performing rotation around the Z-axis of the end effector.

When performing rotations around the end effector the linear velocity of the target w.r.t to the end effector is not constant, which does not satisfy the constant velocity model of the EKF, nevertheless the system was able to track the target correctly.

13.2.4 Approach Tests

The system performed well during the approach test. The pose error was larger when the system was far away and it was decreasing as it was getting closer to the target, as was also observed in the virtual tests. The noise of the estimated pose was also considerably reduced.

13.2.5 Processing Time

The largest processing time was the corresponding to the extraction and detection system for camera-1, which has the ellipse extraction process. This process runs in parallel with the camera-2 systems and it takes about 60% of the total time. The pose solver takes around 38% of the total processing time.

13.3 General Discussion

13.3.1 Effects of the light conditions

The system performance was different for each light condition. With full light there were more extracted features, and the correct feature was extracted almost all the time, but more false features generated by the MLI were also extracted. This can make the system detected a wrong feature in the case of abrupt change in the position. With low light there was less noise but the feature extraction of the correct feature was more intermittent. In this condition abrupt changes in the position are more tolerable since there are less MLI artifact that can confuse the system. In either case the system performed well and was always detecting the right features.

The videos during the testing were taken with the same camera conditions (iris and focus) although it is a common practice to have machine vision systems with automatic systems that adjust the camera conditions according to the system requirements and operation conditions.

13.3.2 Effects of the scenarios

The system performed differently in each scenario. When using scenario-3 a smaller translation error was obtained, since the features provide more information and were more defined than the one of scenario 2. These features were not rigid, since they are flexible fabrics (see Figure 79), and then their dimension can vary or they are difficult to measure, in addition they are not straight lines as was assumed. Scenario 2 is more robust to errors in the initial conditions since its feature is less affected by the MLI noise (the border used in from MLI). In scenario 3 the pad was surrounded by MLI, then in a case of abrupt motion the system can track a MLI artifacts. The main advantage of Scenario-1 is that it requires less information from the target, the main problem is that in the case of abrupt motions or wrong initial condition it can use the wrong normal. Another disadvantage is that only 5DOF pose is obtained since rotation around the target axial axis are not observable.

In general, using more features has more advantages in terms of robustness and accuracy. Note that only the interface was used for camera-1, but any additional feature could have been used. Different features can also be used according to the relative distance between the objects, since it may occur that some features are only observed from certain distances.

Ellipses provide more information than lines, and lines provide more information than points. It is also easier to track an ellipse than a line, and it is easier to track a line than a point. Especially in this application where the reflections created in the MLI can be incorrectly taken as detected features. Consequently it is recommended to use ellipses and lines in these cases.

The EKF noise covariance matrix was different for scenario 3. The noise covariance matrix has the form,

$$R = \text{diag}(w_n r_1 \quad w_n r_2 \quad w_n r_3 \quad w_c r_4 \quad w_c r_5 \quad w_c r_6 \quad w_\omega r_7 \quad w_\omega r_8 \quad w_\omega r_9 \quad w_v r_{10} \dots \quad w_v r_{11} \quad w_v r_{12} \quad w_q r_{13} \quad w_q r_{14} \quad w_q r_{15} \quad w_q r_{16} \quad w_p r_{17} \quad w_p r_{18} \quad w_p r_{19}) \quad (133)$$

where $r_i, i = 1..15$ are value that are common for all the scenarios, w_n is the weight given to the noise associated to the normal vector, similarly w_c is weight associated to the circle center, w_ω is the weight associated to angular velocity, w_v is the weight associated to the linear velocity, w_q is the weight associated to quaternion given by the pose solver, w_p is the weight associated to target center given by the pose solver. The values of the weight for scenario 2 and 3 are equal each other but different than the values used for scenario 1. The values for the weights and the r_i values were obtained experimentally during the EKF calibration. The calibration process was time consuming due to the fact that there are bias errors in the measurements and if their noise weights are not large enough the EKF will not be able to find a stable solution and will start to oscillate between possible states. In the other hand if the weights are too large the EKF will tend to ignore that particular measurement and the response to changes in that measurement will be slow.



Figure 79. Target features. Pad borders and MLI borders

13.3.3 Virtual images vs. Experimental images

The difference between virtual images and real images are the noise level, the distortion and the errors in the camera parameter (calibration errors). The noise is related to reflection and textures in the components of the target. The distortion and errors in the parameters generates bias error in the estimated pose, meanwhile the noise can result in loss of track or tracking wrong features, especially when combined with calibration

errors. The use of the blue channel when using the real images reduced the MLI noise and provided a cleaner edge map which made possible the detection of the correct features.

Summarizing the main effects that explain the bias error in the pose estimation are:

- Discretization of the feature shapes due to the image pixels
- Inability to distinguish between the inner and outer diameter of the interface ring
- Image distortion
- Calibration errors in the cameras (intrinsic and extrinsic parameters)
- Image noise (MLI reflections)
- Occlusions (heat shield, end effector)
- Features in the target are not rigid (except interface ring)

14 CONCLUSIONS

- A machine vision based pose estimation system was designed, implemented and tested under several conditions at close proximities from the target. The total translation error of the system is between 25 and 50mm and the total rotation error is between 2 and 3deg when the target is at 0.7m from the target.
- The system was able to track and to approach a full scale realistic mockup satellite with an interface ring separation system.
- The system is robust to variations in the type of features to track (scenarios), variations in the light conditions, and variations in the relative position. Robustness is defined in the sense that it keeps track of the target features along the range of parameters used and providing a stable pose estimation.
- The system was also robust to variations in the initial conditions when using Scenario 2. When using other scenarios the system may track wrong features.
- The system is robust to small occlusions, such as the ones generated by the end effector and the heat shield.
- The system was robust to errors in the camera calibration (extrinsic and intrinsic)

15 REFERENCES

- [1] Goddard Space Flight Center, "On-Orbit Satellite Servicing Study Project Report," National Aeronautics and Space Administration, 2010.
- [2] N. Inaba and M. Oda, "Autonomous satellite capture by a space robot: world first on-orbit experiment on a Japanese robot satellite ETS-VII," in *Proceedings of the 2000 IEEE International Conference on Robotics & Automation*, San Francisco, CA, 2000.
- [3] N. Inaba, T. Nishimaki, M. Asano and M. Oda, "Rescuing a Stranded Satellite in Space - Experimental Study of Satellite Captures Using a Space Manipulator," in *Proceedings of the 2003 IEEE/RSJ Intl. Conference on Intelligent Robots and Systems*, Las Vegas, Nevada, 2003.
- [4] A. Ogilvie, J. Allport, M. Hannah and J. Lymer, "Autonomous Satellite Servicing Using the Orbital Express Demonstration Manipulator System," in *i-SAIRAS: International Symposium on Artificial Intelligence, Robotics and Automation in Space*, Hollywood, 2008.
- [5] F. Sellmaier, T. Boge, J. Spurmann, S. Gully, T. Rupp and F. Huber, "On-Orbit Servicing Missions: Challenges and Solutions for Spacecraft Operations," in *SpaceOps 2010 Conference*, Huntsville, Alabama, USA, 2010.
- [6] G. Rouleau, I. Rekleitis, R. L'Archeveque, E. Martin, K. Parsa and E. Dupuis, "Autonomous Capture of a Tumbling Satellite," *Journal of Field Robotics*, vol. 24, no. 4, p. 275–296, 2007.
- [7] A. Petit, E. Marchand and K. Kanani, "Vision-based Space Autonomous Rendezvous : A Case Study," in *International Conference on Intelligent Robots and Systems*, 2011.
- [8] X. D. B. L. and Y. T. , "Pose Determination of Large Non-Cooperative Satellite in Close Range Using Coordinated Cameras," in *International Conference on Mechatronics and Automation*, 2009.
- [9] A. Cropp, P. Palmer, P. McLauchlan and C. Underwood, "Estimating pose of known target satellite," *IEEE Electronics Letters* , vol. 36, no. 15, pp. 1331 - 1332 , 2000.
- [10] B. Wilcox, T. Kam, L. Todd, S. Hayati and B. Bruce, "Autonomous Sensor-Based Dual-Arm Satellite Grappling," in *NASA Conference on Space Telerobotics*, Pasadena, CA, 1989.
- [11] K. Yoshida and H. Nakanishi, "Impedance Matching in Capturing a Satellite by a Space Robot," in *Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Las Vegas, Nevada, 2003.
- [12] G. Hirzinger, B. Brunner, R. Lampariello, K. Landzettel, J. Schott and B.-M. Steinmetz, "Advances in Orbital Robotics," in *Proceedings of the 2000 IEEE international Conference on Robotics & Automation*, San Francisco, CA, 2000.
- [13] N. Inaba, M. Asano and M. Oda, "Monitoring Techniques in a Layered Control Architecture for Reliable Robotic Satellite Capture," in *Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, Japan, 2004.
- [14] F. Aghili and K. Parsa, "An Adaptive Vision System for Guidance of a Robotic Manipulator to Capture a Tumbling Satellite with Unknown Dynamics," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nice, France, 2008.
- [15] F. Terui, H. Kamimura and S. Nishida, "Motion Estimation to a Failed Satellite on Orbit using Stereo Vision and 3D Model Matching," in *9th International Conference on Control, Automation, Robotics and Vision, 2006. ICARCV '06.*, Singapore, 2006.
- [16] G. Z. H. L. J. W. and Z. J. , "Vision-Based System for Satellite On-Orbit Self-Servicing," in *International Conference on Advanced Intelligent Mechatronics*, 2008.
- [17] G.-I. SHAN, B. JI and Y.-f. ZHOU, "A Review of 3D Pose Estimation from a Monocular Image Sequence," in *2nd International Congress on Image and Signal Processing*, 2009.

- [18] Y. Nomura, D. Zhang, Y. Sakaida and S. Fujii, "3-D Object Pose Estimation by Shading and Edge Data Fusion," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, San Francisco, CA, 1996.
- [19] S. Jayawardena, M. Hutter and N. Brewer, "A Novel Illumination-Invariant Loss for Monocular 3D Pose Estimation," in *International Conference on Digital Image Computing: Techniques and Applications*, Noosa, QLD, 2011.
- [20] T. Drummond and R. Cipolla, "Real-Time Visual Tracking of Complex Structures," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 932-946, 2002.
- [21] P. Wunsch, S. Winkler and G. Hirzinger, "Real-Time Pose Estimation of 3-D Objects from Camera Images Using Neural Networks," in *IEEE International Conference on Robotics and Automation*, Albuquerque, NM, 1997.
- [22] C.-H. C. and K. Baker, "Pose Estimation For Servicing of Orbital Replacement Units in a Cluttered Environment," in *IEEE International Conference on Robotics and Automation*, 2004.
- [23] S. Hinterstoisser, S. Benhimane and N. Navab, "N3M: Natural 3D Markers for Real-Time Object Detection and Pose Estimation," in *IEEE 11th International Conference on Computer Vision*, Rio de Janeiro, 2007.
- [24] S. Hati and S. Sengupta, "A GA-Based Integrated Approach to Model-Assisted Matching and Pose Estimation for Automated Visual Inspection Applications," in *Congress on Evolutionary Computation*, 2004.
- [25] C.-H. C. "Target Acquisition using Natural Feature Image Recognition," in *IEEE Aerospace conference*, 2009.
- [26] F. Ababsa and M. Mallem, "Robust Camera Pose Estimation Combining 2D/3D Points and Lines Tracking," in *IEEE International Symposium on Industrial Electronics*, Cambridge, 2008.
- [27] A. Wong, L. Rong and X. Liang, "Robotic Vision: 3D Object Recognition and Pose Determination," in *International Conference on Intelligent Robots and Systems, 1998. Proceedings*, Victoria, BC, 1998.
- [28] A. Comport, E. Marchand, M. Pressigout and F. Chaumette, "Real-Time Markerless Tracking for Augmented Reality: The Virtual Visual Servoing Framework," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 4, pp. 615 - 628, 2006.
- [29] M. Goli, F. Janabi-Sharifi and G. Khosroshahi, "Adaptive Particle Filter Based Pose Estimation Using a Monocular Camera Model," in *International Symposium on Optomechatronic Technologies*, Toronto, ON, 2010.
- [30] L. Xiang, P. Guoxiang, W. Bing and Z. Linfeng, "A Dynamic Features Selection based Algorithm for 3D Objects Motion Estimation," in *Third International Symposium on Intelligent Information Technology Application*, Nanchang, 2009.
- [31] W. Wilson, C. Williams Hulls and G. Bell, "Relative End-Effector Control Using Cartesian osition Based Visual Servoing," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 5, pp. 684 - 696 , 1996.
- [32] A. Shademan and F. Janabi-Sharifi, "Sensitivity Analysis of EKF and Iterated EKF Pose Estimation for Position-Based Visual Servoing," in *Proceedings of 2005 IEEE Conference on Control Applications*, Toronto, Ont., 2005.
- [33] M. Mammarella, G. Campa, M. Napolitano, M. Fravolini, Y. Gu and M. Perhinschi, "Machine Vision/GPS Integration Using EKF for the UAV Aerial Refueling Problem," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 38, no. 6, pp. 791 - 801, 2008.
- [34] F. Janabi-Sharifi and M. Marey, "A Kalman-Filter-Based Method for Pose Estimation in Visual Servoing," *IEEE Transactions on Robotics*, vol. 26, no. 5, pp. 939 - 947, 2010.
- [35] H. Taghirad, S. Atashzar and M. Shahbazi, "Robust solution to three-dimensional pose estimation using composite extended Kalman observer and Kalman filter," *IET Computer Vision*, vol. 6, no. 2, pp. 140 - 152, 2012.
- [36] M. Fravolini, M. Mammarella, G. Campa, M. Napolitano and M. Perhinschi, "Machine Vision Algorithms for Autonomous Aerial Refueling for UAVs using the USAF Refueling Boom Method," in *Innovations in Defence Support Systems – 1*, Springer, 2010.
- [37] R. Haralick, "Determining camera parameters from the perspective projection of a rectangle," *Pattern Recognition*, vol. 22, no. 3, pp. 225-230, 1989.

- [38] R. Haralick, H. Joo, C. Lee, X. Zhuang, V. Vaidya and M. Kim, "Pose Estimation from Corresponding Point Data," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 19, no. 6, pp. 1426 - 1446 , 1989.
- [39] J. McInroy, "Nearly Analytical Pose Estimation," in *IEEE International Conference on Robotics and Automation*, Roma, 2007.
- [40] M. Abidi and T. Chandra, "A New Efficient and Direct Solution for Pose Estimation Using Quadrangular Targets: Algorithm and Evaluation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 5, pp. 534 - 538 , 1995.
- [41] P. Rosin, "Robust Pose Estimation," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 29, no. 2, pp. 297 - 303 , 1999.
- [42] M. Rodrigues and L. Yonghuai, "Distance constraint based iterative structure and pose estimation from a single image," in *International Conference on Image Processing*, Vancouver, BC, 2000.
- [43] A. Ansar and K. Daniilidis, "Linear Pose Estimation from Points or Lines," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 5, pp. 578 - 589, 2003.
- [44] F. Hou and F. Zhu, "A Filter Method for Pose Estimation of Maneuvering Target," in *International Conference on Intelligent Robots and Systems*, 2004.
- [45] Z. Z. J. Y. and D. Z. , "Pose Estimation and Structure Recovery from Point Pairs," in *IEEE International Conference on Robotics and Automation*, 2005.
- [46] Y. K. Y. K. H. W. and M. Chang, "Pose Estimation for Augmented Reality Applications Using Genetic Algorithm," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 35, no. 6, pp. 1295 - 1301, 2005.
- [47] Z. Z. D. Z. and J. Z. , "An Improved Pose Estimation Algorithm For Real-time Vision Applications," in *International Conference on Communications, Circuits and Systems*, Guilin, 2006.
- [48] L. Lucchese, "Closed-form pose estimation from metric rectification of coplanar points," *IEE Proceedings - Vision, Image and Signal Processing*, vol. 153, no. 3, pp. 364 - 378 , 2006.
- [49] O. Tahri, C. Leroux and J. Alexandre, "Pose estimation from less than six non coplanar points," in *IEEE International Conference on Robotics and Automation*, 2006.
- [50] Y. H. F. Z. J. O. Q. W. J. Z. and S. F. , "Robust Analysis of P3P Pose Estimation," in *IEEE International Conference on Robotics and Biomimetics*, 2007.
- [51] Y. Y. Q. C. C. L. and Z. Z. , "Pose Estimation based on Four Coplanar Point Correspondences," in *Sixth International Conference on Fuzzy Systems and Knowledge Discovery*, 2009.
- [52] W. M. S. S. and Y. L. , "Linear Relaxation for Global Pose Estimation," in *8th World Congress on Intelligent Control and Automation*, 2010.
- [53] H. Z. "Three-Step Approach to Camera Pose Estimation," in *Second IITA International Conference on Geoscience and Remote Sensing*, 2010.
- [54] C.-M. C. H.-W. C. T.-Y. L. S.-H. L. and Y.-H. T. , "Robust 3D Object Pose Estimation From A Single 2D Image," in *IEEE Visual Communications and Image Processing*, 2011.
- [55] F. S. J. Z. and Y. L. , "A multi-camera system for precise pose estimation in industrial applications," in *IEEE International Conference on Automation and Logistics*, 2009.
- [56] J. Diaz and M. Abderrahim, "Modified SoftPOSIT algorithm for 3D visual tracking," in *IEEE International Symposium on Intelligent Signal Processing*, 2007.
- [57] D. W. H. R. Y. W. and C. W. , "Feature-tracking Based Pose Estimation for Autonomous," in *3rd International Symposium on Systems and Control in Aeronautics and Astronautics*, 2010.
- [58] Y. L. and Y. W. , "Evaluating 3D-2D Correspondences for Accurate Camera Pose Estimation from A Single Image," in *IEEE International Conference on Systems, Man and Cybernetics*, 2003.
- [59] R. Subbarao, Y. Gene and P. Meer, "Nonlinear Mean Shift for Robust Pose Estimation," in *IEEE Workshop on Applications of Computer Vision*, 2007.

- [60] W. Xuedong, J. Xinhua, Z. Rongjin and H. Jiashan, "An Application of Unscented Kalman Filter for Pose and Notion Estimation Based on Monocular Vision," in *IEEE International Symposium on Industrial Electronics*, Montreal, Que., 2006.
- [61] S. Aravindan and P. Kaleeswaran, "Pose Estimation of a Low Altitude Aerial Vehicle using Quaternion Theory and Kalman Filter," in *Recent Advances in Space Technology Services and Climate Change*, Chennai, 2010.
- [62] X. Zhang, K. Wang, Z. Zhang and Q. Yu, "A New Line-Based Orthogonal Iteration Pose Estimation Algorithm," in *International Conference on Information Engineering and Computer Science*, Wuhan, 2009.
- [63] L. Qin and F. Zhu, "The Judgment Method for the Unique Solution of Real-time Pose Estimation from Particular Line Correspondences," in *IEEE International Conference on Mechatronics and Automation*, Luoyang, Henan, 2006.
- [64] L. Qin, Y. Hu, Y. Wei, Y. Zhou and H. Wang, "A New Closed-Form Method for Pose Estimation from Three Z-like Lines," in *7th World Congress on Intelligent Control and Automation*, Chongqing, 2008.
- [65] L. Qin, Y. Hu, Y. Wei, Y. Zhou and H. Wang, "Novel Parallelogram Based Pose Estimation Method," in *4th International Conference on Wireless Communications, Networking and Mobile Computing*, Dalian, 2008.
- [66] X. Wu, D. Pi and X. Jiang, "Particle Filter Based Pose and Motion Estimation with Non-Gaussian Noise," in *IEEE International Conference on Industrial Technology*, Mumbai, 2006.
- [67] C. Madsen, "Viewpoint Variation in the Noise Sensitivity of Pose Estimation," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, San Francisco, CA, 1996.
- [68] M. Hutter and N. Brewer, "Matching 2-D ellipses to 3-D circles with application to vehicle pose identification," in *24th International Conference Image and Vision Computing New Zealand*, Wellington, 2009.
- [69] Y. C. Shiu and S. Ahmad, "3D Location of Circular and Spherical Features by Monocular Model-Based Vision," in *IEEE International Conference on Systems, Man and Cybernetics.*, Cambridge, MA, 1989.
- [70] R. Safaee-Rad, I. Tchoukanov, K. Smith and B. Benhabib, "Three-Dimensional Location Estimation of Circular Features for Machine Vision," *IEEE Transactions on Robotics and Automation*, vol. 8, no. 5, pp. 624-640, 1992.
- [71] D. He and B. Benhabib, "Solving the Orientation-Duality Problem for a Circular Feature in Motion," *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 28, no. 4, pp. 506-515, 1998.
- [72] C. Zen and J.-B. Huang, "A Vision-Based Method for the Circle Pose Determination With a Direct Geometric Interpretation," *IEEE Transactions on Robotics and Automation*, vol. 15, no. 6, pp. 1135-1140, 1999.
- [73] Y. Zheng, W. Ma and L. Yuncai, "Another Way of Looking at Monocular Circle Pose Estimation," in *15th IEEE International Conference on Image Processing*, San Diego, CA, 2008.
- [74] F. Ababsa and M. Mallem, "Robust Circular Fiducials Tracking And Camera Pose Estimation Using Particle Filtering," in *IEEE International Conference on Systems, Man and Cybernetics*, Montreal, Que., 2007.
- [75] R. Hanek, N. Navab and M. Appel, "Yet another Method for Pose Estimation: A Probabilistic Approach using Points, Lines, and Cylinders," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Fort Collins, CO, 1999.
- [76] C. Xu, B. Kuipers and A. Murarka, "3D Pose Estimation for Planes," in *IEEE 12th International Conference on Computer Vision Workshops*, Kyoto, 2009.
- [77] S. Ramalingam, S. Bouaziz and P. Sturm, "Pose Estimation using Both Points and Lines for Geo-Localization," in *IEEE International Conference on Robotics and Automation*, Shanghai, 2011.
- [78] A. Gupta, J. Little and R. Woodham, "Using Line and Ellipse Features for Rectification of Broadcast Hockey Video," in *Canadian Conference on Computer and Robot Vision*, St. Johns, NL, 2011.
- [79] Y. R. and H. W. , "Pose and position estimations of regular shapes in monocular images," in *International Conference on Machine Learning and Cybernetics*, 2009.
- [80] J. Shi and C. Tomasi, "Good Features to Track," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Seattle, WA, 1994.

- [81] L. Libuda, I. Grothues and K.-F. Kraiss, "Ellipse Detection in Digital Image Data Using Geometric Features," in *Advances in Computer Graphics and Computer Vision*, Springer, 2007.
- [82] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis," *Communications of the ACM*, vol. 24, no. 6, pp. 381 - 395, 1981.
- [83] F. Viksten, R. Soderberg, K. Nordberg and C. Perwass, "Increasing Pose Estimation Performance using Multi-cue Integration," in *IEEE International Conference on Robotics and Automation*, Orlando, Florida, 2006.
- [84] Y. Shiu and S. Ahmad, "Calibration of wrist-mounted robotic sensors by solving homogeneous transform equations of the form $AX=XB$," *IEEE Transactions on Robotics and Automation*, vol. 5, no. 1, pp. 16-29, 1989.
- [85] X. Du, B. Liang, W. Xu, X. Wang and J. Yu, "Pose Measurement of a GEO Satellite Based on Natural Features," in *International Conference on Virtual Reality and Visualization*, Qinhuangdao, Hebei, China, 2012.
- [86] J. M. Galante, J. V. Eepoel, M. Strube, N. Gill, M. Gonzalez, A. Hyslop and B. Patrick, "Pose Measurement Performance of the Argon Relative Navigation Sensor Suite in Simulated Flight Conditions," in *AIAA Guidance, Navigation, and Control Conference (2012)*, Minneapolis, 2012.
- [87] A. F. Velasquez, J. Luckett, M. R. Napolitano, G. Marani, T. Evans and M. L. Fravolini, "Experimental Evaluation of a Machine Vision Based Pose Estimation System for Autonomous Capture of Satellites with Interface Rings," Unpublished.
- [88] A. F. Velasquez, G. Marani, T. Evans, R. M. Napolitano, A. J. Christian and G. Doretto, "Virtual Simulator for Testing a Vision Based Pose Estimation System for Autonomous Capture of Satellites with Interface Rings," in *21st Mediterranean Conference on Control and Automation*, Crete, Greece, 2013.
- [89] D. Simon, *Optimal State Estimation. Kalman, H_∞ , and Nonlinear Approaches*, Wiley, 2006.
- [90] "Boeing," [Online]. Available: http://www.boeing.com/bds/phantom_works/orbital/oe_images-gallery03.html. [Accessed 2011].
- [91] T. Miyabe, A. Konno and M. Uchiyama, "Automated Object Capturing with a Two-Arm Flexible Manipulator," in *Proceedings of the 2003 IEEE International Conference on Robotics & Automation*, Taipei, Taiwan, 2003.
- [92] A. Heaton, R. Howard and R. Pinson, "Orbital Express AVGS Validation and Calibration for Automated Rendezvous," in *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, Honolulu, Hawaii, 2008.
- [93] Boeing, [Online]. Available: http://www.boeing.com/bds/phantom_works/orbital/oe_images-gallery02.html.
- [94] R. Howard, A. Heaton, R. Pinson and C. Carrington, "Orbital Express Advanced Video Guidance Sensor," in *2008 IEEE Aerospace Conference*, Big Sky, MT, 2008.
- [95] S. Ruel, C. English, M. Anctil and P. Church, "3DLASSO: Real-Time Pose Estimation from 3D Data for Autonomous Satellite Servicing," in *The 8th International Symposium on Artificial Intelligence, Robotics and Automation in Space - iSAIRAS*, Munich, Germany.
- [96] A. Allen, N. Mak and C. Langley, "Development of a Scaled Ground Testbed for Lidar-based Pose Estimation," in *IEEE IROS 2005 Workshop on Robot Vision for Space Applications*, 2005.
- [97] R. N. Jazar, *Advanced Dynamics. Rigid Body, Multibody, and Aerospace Applications*, Wiley, 2011.
- [98] S. Christiansen and T. Nilson, "Docking System Mechanism Utilized on Orbital Express Program," in *39th Aerospace Mechanisms Symposium*, NASA Marshall Space Flight Center, 2008.
- [99] G. Campa, M. Napolitano and L. F. Mario, "Simulation Environment for Machine Vision Based Aerial Refueling for UAVs," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 45, no. 1, 2009.