



Calhoun: The NPS Institutional Archive
DSpace Repository

Theses and Dissertations

1. Thesis and Dissertation Collection, all items

2022-09

**ASSESSMENT OF ELECTRO-OPTICAL IMAGING
TECHNOLOGY FOR UNMANNED AERIAL
SYSTEM NAVIGATION IN A GPS-DENIED ENVIRONMENT**

Chan, Yi Cheng

Monterey, CA; Naval Postgraduate School

<http://hdl.handle.net/10945/71102>

Copyright is reserved by the copyright owner.

Downloaded from NPS Archive: Calhoun



Calhoun is the Naval Postgraduate School's public access digital repository for research materials and institutional publications created by the NPS community. Calhoun is named for Professor of Mathematics Guy K. Calhoun, NPS's first appointed -- and published -- scholarly author.

Dudley Knox Library / Naval Postgraduate School
411 Dyer Road / 1 University Circle
Monterey, California USA 93943

<http://www.nps.edu/library>



**NAVAL
POSTGRADUATE
SCHOOL**

MONTEREY, CALIFORNIA

THESIS

**ASSESSMENT OF ELECTRO-OPTICAL IMAGING
TECHNOLOGY FOR UNMANNED AERIAL SYSTEM
NAVIGATION IN A GPS-DENIED ENVIRONMENT**

by

Yi Cheng Chan

September 2022

Thesis Advisor:
Second Reader:

Oleg A. Yakimenko
Fotis A. Papoulias

Approved for public release. Distribution is unlimited.

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			<i>Form Approved OMB No. 0704-0188</i>	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington, DC 20503.				
1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE September 2022	3. REPORT TYPE AND DATES COVERED Master's thesis	
4. TITLE AND SUBTITLE ASSESSMENT OF ELECTRO-OPTICAL IMAGING TECHNOLOGY FOR UNMANNED AERIAL SYSTEM NAVIGATION IN A GPS-DENIED ENVIRONMENT			5. FUNDING NUMBERS	
6. AUTHOR(S) Yi Cheng Chan				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000			8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A			10. SPONSORING / MONITORING AGENCY REPORT NUMBER	
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.				
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release. Distribution is unlimited.			12b. DISTRIBUTION CODE A	
13. ABSTRACT (maximum 200 words) Navigation systems of unmanned aircraft systems (UAS) are heavily dependent on the availability of Global Positioning Systems (GPS) or other Global Navigation Satellite Systems (GNSS). Although inertial navigation systems (INS) can provide position and velocity of an aircraft based on acceleration measurements, the information degrades over time and reduces the capability of the system. In a GPS-denied environment, a UAS must utilize alternative sensor sources for navigating. This thesis presents preliminary evaluation results on the usage of onboard down-looking electro-optical sensors and image matching techniques to assist in GPS-free navigation of aerial platforms. Following the presentation of the fundamental mathematics behind the proposed concept, the thesis analyzes the key results from three flight campaign experiments that use different sets of sensors to collect data. Each of the flight experiments explores different sensor setups, assesses a variety of image processing methods, looks at different terrain environments, and reveals limitations related to the proposed approach. In addition, an attempt to incorporate navigational aid solutions into a navigation system using a Kalman filter is demonstrated. The thesis concludes with recommendations for future research on developing an integrated navigation system that relies on inertial measurement unit data complemented by the positional fixes from the image-matching technique.				
14. SUBJECT TERMS GPS-denied environment, image matching technique, navigation, unmanned aerial system, image matching navigation algorithm, electro-optical imaging, Global Positioning Systems			15. NUMBER OF PAGES 117	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UU	

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release. Distribution is unlimited.

**ASSESSMENT OF ELECTRO-OPTICAL IMAGING TECHNOLOGY FOR
UNMANNED AERIAL SYSTEM NAVIGATION IN A GPS-DENIED
ENVIRONMENT**

Yi Cheng Chan
Civilian, DSO National Laboratories
AE, Nanyang Technological University, Singapore, 2013

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN SYSTEMS ENGINEERING

from the

**NAVAL POSTGRADUATE SCHOOL
September 2022**

Approved by: Oleg A. Yakimenko
Advisor

Fotis A. Papoulias
Second Reader

Oleg A. Yakimenko
Chair, Department of Systems Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Navigation systems of unmanned aircraft systems (UAS) are heavily dependent on the availability of Global Positioning Systems (GPS) or other Global Navigation Satellite Systems (GNSS). Although inertial navigation systems (INS) can provide position and velocity of an aircraft based on acceleration measurements, the information degrades over time and reduces the capability of the system. In a GPS-denied environment, a UAS must utilize alternative sensor sources for navigating. This thesis presents preliminary evaluation results on the usage of onboard down-looking electro-optical sensors and image matching techniques to assist in GPS-free navigation of aerial platforms. Following the presentation of the fundamental mathematics behind the proposed concept, the thesis analyzes the key results from three flight campaign experiments that use different sets of sensors to collect data. Each of the flight experiments explores different sensor setups, assesses a variety of image processing methods, looks at different terrain environments, and reveals limitations related to the proposed approach. In addition, an attempt to incorporate navigational aid solutions into a navigation system using a Kalman filter is demonstrated. The thesis concludes with recommendations for future research on developing an integrated navigation system that relies on inertial measurement unit data complemented by the positional fixes from the image-matching technique.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

I.	INTRODUCTION.....	1
A.	BACKGROUND	1
B.	MOTIVATION AND PROBLEM DEFINITION	2
C.	THESIS STRUCTURE	6
II.	MATHEMATICAL FOUNDATION OF VISION-BASED ODOMETRY	9
A.	DERIVATION OF CHANGE IN UAS POSITION	9
B.	FEATURE EXTRACTION	11
C.	HOMOGRAPHY AND TRANSFORMATIONS	14
III.	IMAGE PROCESSING	19
A.	FEATURE DETECTION	19
B.	FEATURE MATCHING	24
C.	OUTLIER REJECTION.....	25
D.	TRANSFORMATION DECOMPOSITION.....	26
IV.	EXPERIMENTING WITH QUANTUM TRINITY F90+ MAPPING UAS.....	29
A.	TEST SETUP	29
B.	DATA PROCESSING	31
C.	KEY OBSERVATIONS	35
V.	PROCESSING TASE 200 DATA.....	47
A.	TEST SETUP	47
B.	DATA PROCESSING	49
C.	KEY OBSERVATIONS	52
VI.	DATA FROM SELF-BUILT SENSOR SUITE	57
A.	TEST SETUP	57
B.	DATA PROCESSING	60
C.	KEY OBSERVATIONS.....	63
VII.	INCORPORATING DEVELOPED NAVAID SOLUTION ON UAS.....	73
VIII.	CONCLUSION	79
A.	SUMMARY OF WORK DONE	79

B. RECOMMENDATIONS FOR FUTURE WORK	80
APPENDIX. NAVAID IMPLEMENTATION SCRIPT	83
LIST OF REFERENCES	89
INITIAL DISTRIBUTION LIST	93

LIST OF FIGURES

Figure 1.	GPS trilateration of four satellites. Source: Gilliland (2019).	2
Figure 2.	3D point cloud map generated from stereovision sensors. Source: Perez-Grau et al. (2018).....	4
Figure 3.	Example of image matching. Adapted from Yakimenko and Decker (2017).....	6
Figure 4.	Relationship between UAS state, camera orientation, and target point of interest during transit	9
Figure 5.	Image with corner detector	12
Figure 6.	Image with blob detector	13
Figure 7.	Geometric transformations of checkerboard. Source: MathWorks (n.d.).....	17
Figure 8.	Vision-based NAVAID image processing sequence	19
Figure 9.	Grass field aerial views.....	20
Figure 10.	Visualization of feature points detected using different methods.....	21
Figure 11.	Number of features detected with respect to detection algorithms.....	22
Figure 12.	CPU execution time to extract features with respect to detection algorithms	22
Figure 13.	Average CPU time required for feature detection with respect to detection methods	23
Figure 14.	CPU execution time with respect to number of features detected.....	24
Figure 15.	Two consecutive aerial images of grass field	25
Figure 16.	All feature matchings between two consecutive images from SURF detection method.....	25
Figure 17.	Matched inliers between two consecutive images from SURF detection method.....	26
Figure 18.	Recovered target image.....	27

Figure 19.	a) Trinity F90+ mapping drone with b) Sony RX1R II camera. Source: Quantum Systems (n.d.).	29
Figure 20.	Trinity F90+ flight over waypoint options. Source: Quantum Systems (n.d.).	31
Figure 21.	Trajectory overview of Camp Roberts flight campaign	32
Figure 22.	3D flight profile of Camp Roberts flight campaign	32
Figure 23.	Terrain elevation (gray) and preplanned flight altitude (blue)	33
Figure 24.	Flight telemetry plots of Camp Roberts flight campaign	34
Figure 25.	Sample images from Camp Roberts flight campaign	35
Figure 26.	Number of features detected in Camp Roberts flight imagery using ORB, SURF, and FAST methods	35
Figure 27.	Estimated yaw comparison with reference GPS, for SURF strongest 5,000 points	38
Figure 28.	Estimated yaw comparison with reference GPS, for ORB strongest 8,000 points	38
Figure 29.	Estimated yaw comparison with reference GPS, for FAST strongest 8,000 points	39
Figure 30.	Compiled flight telemetry data with estimated yaw for SURF strongest 5,000 points	40
Figure 31.	Estimated flight trajectory overlaid on flight profile	41
Figure 32.	Example of segmented flight analysis	42
Figure 33.	Segmented flight trajectory estimation using SURF strongest 5,000 points	43
Figure 34.	Improved segmented flight trajectory estimation, SURF strongest 5,000 points	44
Figure 35.	Improved segmented flight trajectory estimation, ORB strongest 8,000 points	45
Figure 36.	Improved segmented flight trajectory estimation, FAST strongest 8,000 points	46
Figure 37.	TASE 200 sensor in SkyIMD enclosure	47

Figure 38.	Overview of flown trajectory with TASE 200 system.....	49
Figure 39.	3D flight profile of flown trajectory with TASE 200 system	49
Figure 40.	Sample TASE 200 images	50
Figure 41.	Number of features detected using FAST, ORB, and SURF methods from TASE 200 imagery files.....	50
Figure 42.	Matched inliers on raw TASE 200 images	51
Figure 43.	Matched inliers on cropped TASE 200 images	52
Figure 44.	Comparison of SURF method trajectory estimations between raw and cropped imagery at 2,000 ft	53
Figure 45.	Trajectory estimations for down leg segment at 2,000 ft.....	54
Figure 46.	Trajectory estimations for up leg segment at 2,000 ft	54
Figure 47.	Trajectory estimations for down leg segment at 4,000 ft.....	55
Figure 48.	Trajectory estimations for up leg segment at 4,000 ft	55
Figure 49.	Trajectory estimations for up leg segment at 6,000 ft	56
Figure 50.	Cessna-172 aircraft	57
Figure 51.	Trajectory overview of Cessna flight campaign	58
Figure 52.	3D flight profile of Cessna flight campaign	58
Figure 53.	Typical UAS flight profile	59
Figure 54.	Mounting setup of acquisition equipment on Cessna wing strut	59
Figure 55.	Flight telemetry plots of Cessna flight campaign	61
Figure 56.	Sample images acquired from GoPro Hero 4	62
Figure 57.	Number of features detected in Cessna flight imagery using ORB, SURF, and FAST methods	62
Figure 58.	Trajectory estimations for straight level flight at 2,000 ft	64
Figure 59.	Trajectory estimations for straight level flight at 3,000 ft	64
Figure 60.	Trajectory estimations for straight level flight at 4,000 ft	65

Figure 61.	Flight telemetry for left-bank maneuver at 2,000 ft.....	66
Figure 62.	Trajectory estimations for left-bank maneuver at 2,000 ft	67
Figure 63.	Flight telemetry for left-bank maneuver at 3,000 ft.....	68
Figure 64.	Trajectory estimations for left-bank maneuver at 3,000 ft	69
Figure 65.	Flight telemetry for left-bank maneuver at 4,000 ft.....	70
Figure 66.	Trajectory estimations for left-bank maneuver at 4,000 ft	71
Figure 67.	Trajectory estimations for right-bank maneuver at 2,000 ft	71
Figure 68.	Trajectory estimations for right-bank maneuver at 3,000 ft	72
Figure 69.	Trajectory estimations for right-bank maneuver at 4,000 ft	72
Figure 70.	Block diagram of INS and GPS integration.....	73
Figure 71.	Block diagram of INS and NAVAID integration	73
Figure 72.	Axes convention definition relative to iPhone device	75
Figure 73.	Trajectory comparison of INS-GPS reference, INS-only, and INS-NAVAID FAST	76
Figure 74.	Trajectory comparison of INS-GPS reference, INS-only, and INS-NAVAID ORB.....	76
Figure 75.	Trajectory comparison of INS-GPS reference, INS-only, and INS-NAVAID SURF	77

LIST OF TABLES

Table 1.	Summary of feature extraction methods	13
Table 2.	Elementary transformations	15
Table 3.	Minimum requirements for geometric transformations	17
Table 4.	Summary of estimated transformation parameters	27
Table 5.	Technical data of Trinity F90+ mapping drone. Adapted from Quantum Systems (n.d.).....	30
Table 6.	Metadata of images captured by Sony RX1R II	30
Table 7.	Sensitivity analysis results for SURF detection method.....	36
Table 8.	Sensitivity analysis results for ORB detection method.....	36
Table 9.	Sensitivity analysis results for FAST detection method	37
Table 10.	Flight segments with corresponding start and end frame index	42
Table 11.	Summary of corrections applied on initializing yaw angles	43
Table 12.	Pertinent specifications of TASE 200 sensor.....	47
Table 13.	TASE 200 meta data format specification	48
Table 14.	Comparison of estimated transformation parameters	52
Table 15.	QStarz BT-Q1000eX sensor specifications. Adapted from QStarz International Co. Ltd. (n.d.)	60

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

BRIEF	Binary Robust Independent Elementary Features
BRISK	Binary Robust Invariant Scalable Keypoints
COTS	Commercial off-the-shelf
CPU	Central processing unit
CV	Computer vision
DEM	Digital elevation map
EO	Electro-optics
eVTOL	Electric vertical take-off and landing
FAST	Features from Accelerated Segment Test
fps	Frame per second
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
Harris	Harris-Stephens corner detector
IMU	Inertial measurement unit
INS	Inertial navigation system
IR	Infrared
LED	Light-emitting diodes
LNA	Low noise amplifier
LoS	Line-of-sight
MEMS	Micro-electromechanical system
MinEigen	Minimum Eigenvalue algorithm
MSAC	M-estimator Sample Consensus
MSER	Maximally Stable External Regions
MSL	Mean sea level
NAVAID	Navigational aid
NED	North-East-Down
NIR	Near infrared
ORB	Oriented FAST and rotated BRIEF
PNT	Positioning, navigation, and timing
RANSAC	Random Sample Consensus

RGB	Red, green, blue
RMS	Root mean square
RVI	Reference view image
SURF	Speeded-Up Robust Features
UAS	Unmanned aircraft systems
UAV	Unmanned aerial vehicle
UHR	Ultra high resolution
VIO	Visual inertial odometry

EXECUTIVE SUMMARY

Navigation systems of unmanned aircraft systems (UAS) are heavily dependent on the availability of Global Positioning Systems (GPS) or other Global Navigation Satellite Systems (GNSS) to enable beyond line-of-sight autonomous operations. With global positioning information provided by GPS signals, human operators of UAS can track, apply flight profile corrections, and steer the UAS to its intended area of operations.

In the event of a GPS-denied scenario, a UAS will have to depend on its inertial navigation system (INS) as the only source of information for navigation (Cole 2017). Although the INS and its inertial measurement unit (IMU) can continue to provide position and velocity of the aircraft based on acceleration measurements, the information degrades over time and reduces the navigation capability of the UAS. This is undesirable for a UAS conducting long-endurance (long-range) missions. While sensors such as barometers, magnetometers, and air data systems can augment GPS in providing UAS position and attitude information, this thesis seeks to study the feasibility of using a vision-based sensor as a navigation aid (NAVAID) solution.

The key aspects of the proposed image-matching technique for manned and unmanned aerial vehicles operating in GPS-denied environments have been evaluated previously by Han (2017), and Yakimenko and Decker (2017, 2019). In these studies, the matching was done between the imagery obtained by the aerial vehicle and previously stored satellite imagery. Nevertheless, the risk of a map database being inaccurate is high, particularly for disaster areas with terrain deformation. In addition, for UASs flying at faster speeds over unknown territory, vision-based mapless navigation methods may appear more attractive. Hence, this thesis aims to study the feasibility of a mapless navigation solution, where pose estimation is computed from imagery acquired from onboard downward-looking electro-optical (EO) sensors.

To address the feasibility of adopting a vision-based NAVAIID for UAS navigation, this thesis aims to fulfill the following objectives:

- Viability of real-time application. Different image processing methods are investigated and recommended based on their computational execution time.
- Imagery resolution and frame rate requirements. A range of EO sensors, ranging from dedicated high-resolution aerial mapping sensors to relatively cheaper commercial off-the-shelf (COTS) camera sensors, are used.
- Comparison against reference GPS data. Pose estimation results from different flight altitudes and flight maneuvers are compared against reference GPS data to understand vision-based NAVAID performance and limitations.
- Implementation of NAVAID pose estimations into UAS navigation.

The position of a feature or any target point of interest in a local tangent coordinate plane was derived from its position in the camera coordinate plane, by transforming it to the body coordinate frame and subsequently to the normal tangent plane. The proposed approach assumed that target points with known geolocations were not available. By processing two consecutive images, the change of position of the target point was determined, which is used in place of GPS signals by the UAS navigation system.

To facilitate real-time implementation on the UAS, feature detection algorithms were selected based on computing processing unit (CPU) time and number of features detected. Acquired images were also preprocessed into grayscale format to leverage on the benefits of faster processing speed and eliminate any sensitivity issues of color images (Ebner 2007). Features from the Accelerated Segment Test (FAST), Oriented FAST and rotated BRIEF (ORB), and Speeded-Up Robust Features (SURF) were identified for subsequent NAVAID feasibility studies due to their low processing time per feature detected, indicating better efficiency and superior performance among detection methods investigated.

With the assumption of low image distortion at high flight altitudes, similarity geometric transformation was adopted for pose estimation. Decomposition of the transformation matrix allowed determination of pixel displacement, rotation, and scale factor of matched feature points between consecutive images. It was shown that low resolution imagery over a feature-poor landscape did not hinder the performance of the proposed solution, as long as there was an adequate number of feature matches across images.

The proposed NAVAID solution was executed on three flight campaigns, on different flying platforms and sensor suites. The preliminary image matching feasibility study was conducted on images acquired from the Trinity F90+ mapping drone, equipped with an ultra-high resolution (UHR) camera payload. Subsequent image matching attempts on imagery from the TASE 200 sensor flight campaign required preprocessing to crop out perspective distortion due to the forward looking EO sensor. The third flight campaign featured a self-built sensor suite, consisting of a high-resolution GoPro Hero 4 camera and an iPhone 7 running the Sensor Log application, mounted on a manned Cessna-172 aircraft. The self-built sensor suite offered more flexibility in both data acquisition rate and quality of imagery collected, compared to the previous two flight campaigns.

The current assumption of using similarity transformation for pose estimation showed promising results for straight level flight profiles. The segmented flight profile analysis approach was able to estimate straight level flight trajectories with good accuracy, even with non-periodic or low fps rate video feed. However, it was noted that similarity transformation may not be sufficiently robust to handle perspective distorted imagery due to a forward-looking sensor or bank maneuvers. Image preprocessing may need to be executed to correct distortions before proceeding with the image matching and pose estimation.

Implementation of the NAVAID solution was successfully demonstrated in MATLAB, using a Kalman filter. INS-NAVAID integrated pose results showed better performance compared to INS-only pose calculations, with lower RMS position errors from the proposed solution. This further illustrates the potential of using a vision-based NAVAID for UAS operating in GPS-denied environments.

References

- Cole, Sally. 2017. "Ensuring navigation in GPS-denied environments." *Military Embedded Systems*. November 20, 2017. <https://militaryembedded.com/comms/satellites/ensuring-navigation-gps-denied-environments>.
- Ebner, Marc. 2007. *Color Constancy*. New York: Wiley.
- Han, Keng Siew Aloysius. 2017. "Test and Evaluation of an Image-Matching Navigation System for a UAS Operating in a GPS-Denied Environment." Master's thesis, Naval Postgraduate School. <https://calhoun.nps.edu/handle/10945/56131>.
- Yakimenko, Oleg, and Ryan Decker. 2019. Image-matching navigation method and apparatus for aerial vehicles. U.S. Patent 10,515,458, filed February 28, 2018, and issued December 24, 2019. <http://hdl.handle.net/10945/64482>.
- . 2017. "On the Development of an Image-Matching Navigation Algorithm for Aerial Vehicles." In *2017 IEEE Aerospace Conference*, 1–9. <https://doi.org/10.1109/AERO.2017.7943742>.

ACKNOWLEDGMENTS

First and foremost, I am extremely grateful to my thesis advisor, Distinguished Professor Oleg Yakimenko, for his invaluable advice, continuous support, and patience during my studies for the master's degree at the Naval Postgraduate School. His exceptional knowledge in both unmanned systems and MATLAB has been pivotal in the completion of this thesis.

Additionally, I would like to express my gratitude to Professor Fotis Papoulias, SE technical writing Senior Lecturer Barbara Berlitz, and Thesis Processor Rebecca Pieken for their coaching and editing of this thesis. Their guidance and assistance have significantly improved the flow of the writing and polished my thesis.

I would also like to thank Chris Bley from Insight Up Solutions/UAV Centre, Santa Cruz, California, for providing the valuable UAS telemetry and imagery data to realize this work.

My decision to embark on my master's degree would not have been possible without the support of DSO National Laboratories, Singapore. I am extremely grateful to my directors and mentors who recommended me for a scholarship to sponsor my studies. I would also like to thank my colleagues for following up with my work.

Finally, I would like to express my gratitude to my family for their encouragement and support, which helped me in the completion of this paper. Special gratitude goes to my supportive wife, Valerie, for her understanding and company as I seek to further my studies.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

Autonomous vehicles have become increasingly popular and are employed in civilian and commercial applications, as well as military operations. They are attractive as a supplement to the human workforce (Liew 2019) and have become effective force multipliers, replacing humans to perform dull, dirty, and dangerous work. Not only are they economical to develop, but they can also greatly increase a user's strategic advantage over near-peer competitors (Department of the Navy 2021).

A. BACKGROUND

Unmanned aerial systems (UAS), a category of autonomous vehicles, are equipped with an inertial navigation system (INS) and Global Positioning System (GPS) or equivalent Global Navigation Satellite System (GNSS) receivers, to enable beyond line-of-sight autonomous operations. Specifically, UASs depend on the GPS to provide global positioning, navigation, and synchronized timing based on data received from multiple GPS satellites in orbit. At any point in time, there needs to be a minimum of four satellites in line-of-sight (LoS), shown in Figure 1, to solve for latitude, longitude, altitude, and time (PNT). With knowledge of the current UAS position information, the human operator can track, apply flight profile corrections, and steer the UAS to its intended area of operations at various flight altitudes.

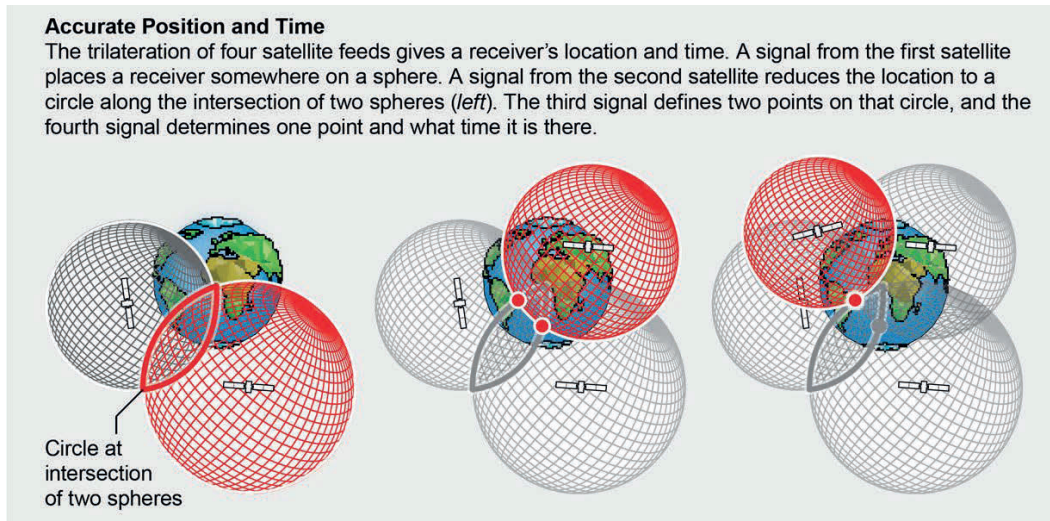


Figure 1. GPS trilateration of four satellites.
 Source: Gilliland (2019).

B. MOTIVATION AND PROBLEM DEFINITION

In the event of a GPS-denied scenario, due to either natural phenomena or human intervention, the UAS will have to depend on its INS as the only source of information for navigation (Cole 2017). Particularly for military operations, the ease of access to GPS spoofing and jamming technologies has made the GPS signal very vulnerable, adversely affecting mission success. Without the GPS, the INS and its inertial measurement unit (IMU) can still provide pose information, but it degrades rapidly due to the accumulation of integration drift over time. This is undesirable especially for long-endurance (long-range) UASs. Periodic corrections should be applied on the INS positioning data to improve its fidelity, using a reference source such as GPS signals. While sensors such as barometers, magnetometers, and air data systems can augment a GPS in providing UAS position and attitude information, this thesis seeks to study the feasibility of using a vision-based sensor as a navigation solution.

Vision-based navigation has been extensively researched in robotics. Desouza and Kak (2002) presented a literature survey on robotic vision-based navigation, while Lu et al. (2018) summarized navigation solutions for unmanned aerial vehicles (UAV) using onboard optical sensors with CV techniques. Autonomous robots have used map building

methods to generate virtual representations of the surrounding environment to execute path planning and obstacle avoidance (Courbon et al. 2010). An internal model of the environment was generated during the first fly past on the intended flight path. As the UAV revisited the same path, due to persistent surveillance requirements, the platform would be able to execute image matching between its current image and its internal database of key images. Localization of the UAV could then be established when high correlation was achieved between matching images. The UAV would steer towards the desired end state image should its current image appear distorted with reference to the database, facilitating autonomous navigation.

Stereo cameras are preferred over monocular cameras for map-based navigation due to improved depth perception, facilitating precise routing and internal model generation. As illustrated in Figure 2, features extracted from images are transformed into the 3D point clouds depicting the exact location of physical objects. Mallet, Lacroix, and Gallo (2000) demonstrated that the 3D point cloud produced from images acquired from stereovision enabled pose estimation with small error magnitudes over several meters of displacement. While it is known that visual odometry suffers from cumulative errors, Perez-Grau et al. (2018) adopted a key-framing approach, where reference key frames containing the UAV's pose information were generated when feature tracking flow exceeded an allowable threshold.

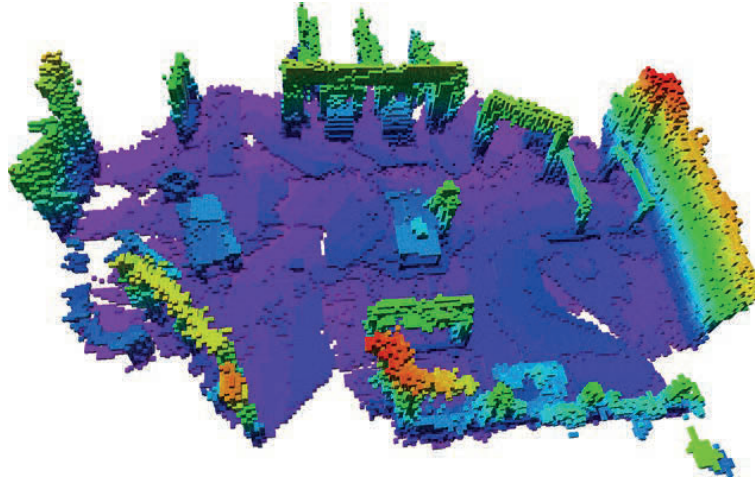


Figure 2. 3D point cloud map generated from stereovision sensors.
Source: Perez-Grau et al. (2018).

Preloaded digital models of the environment are also used in map-based navigation when UASs do not have the opportunity to create their own environment models. Digital models, such as a digital elevation map (DEM) or digital maps, acquired from satellite or other UAS sources are uploaded onto the UAS platform before flight. Image matching will then be performed during flight to determine the correlation between the observed scene and the preloaded database (Zhang, Liu and Wu 2011; Kaniewski and Grzywacz 2017).

Reference beacons, such as an infrared (IR) lamp or a light-emitting diode (LED), implemented at known coordinate positions have also been shown to enable precision navigation. These reference beacons are less sophisticated and have relatively few features as compared to digital models. Gui et al. (2013) utilized IR lamps for precision recovery and landing for UAVs, using optical filters to improve sensitivity of electro-optics (EO) to specific electromagnetic wavelengths. Yakimenko et al. (2002) leveraged on a ship's smokestack as a reference point for shipboard landing and successfully demonstrated real-time flight implementation. In 2005, Valasek et al. proved that LEDs installed on an air refueling drogue basket, used in conjunction with optical sensors, can provide a feasible solution for autonomous air refueling operations.

For UASs flying at faster speeds over unknown territory, vision-based mapless navigation methods may be more attractive. Using computer vision (CV) and image

processing techniques, the mapless navigation system could execute image-matching techniques on imagery acquired by an onboard optical sensor commonly found on UASs conducting aerial surveillance. Instead of conducting image matching with a map model, this method matches detected features across sequential images to determine the corresponding transformations. Since optical sensors are passive devices, they are less susceptible to disruptions, unlike GPS signals, making vision-based UAS navigation a promising area for further development. In fact, some progress in this area has already been demonstrated by Miller et al. (2010), Madison et al. (2007), as well as Sasiadek and Walker (2008). Kong, Egan, and Cornell (2006) successfully demonstrated the concept of using the Canny edge detection algorithm to perform feature matching of consecutive images. The concept was able to track the position of the UAV for a flight time of 20 seconds, within acceptable error tolerances. It was identified that a more advanced feature-matching algorithm be developed to cater for real-time flight applications, which this thesis research aims to achieve. Indeed, many challenges still remain. For example, one of the most recent publications exploring visual inertial odometry (VIO) for an outdoor UAS flight highlighted the challenges of varying lighting conditions and camera pitch orientation on trajectory estimation performance (Bednář et al. 2022).

The key aspects of the proposed image-matching technique for manned and unmanned aerial vehicles operating in GPS-denied environments have been evaluated previously by Han (2017), and Yakimenko and Decker (2017, 2019). In these studies, the matching was done between the imagery obtained by the aerial vehicle and previously stored satellite imagery. As shown in Figure 3, adapted from Yakimenko and Decker (2017), the a) in-flight acquired image was matched with b) the corresponding satellite reference view image (RVI) to generate c) where the flight image was imposed on the RVI. This thesis continues the aforementioned line of efforts by investigating an image-matching technique based on the imagery coming from an onboard EO sensor only.

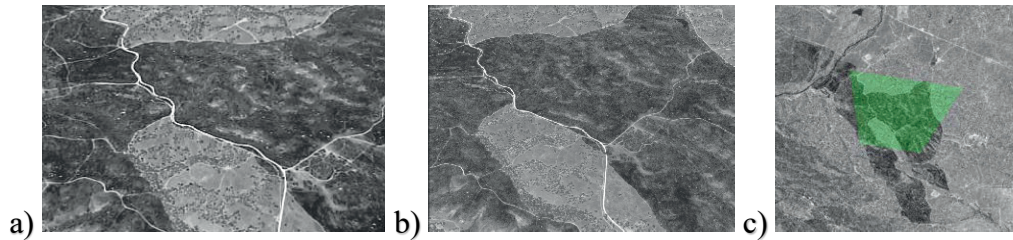


Figure 3. Example of image matching.
Adapted from Yakimenko and Decker (2017).

Specifically, the work presented in this thesis aims to assess the feasibility of using onboard EO sensors as a vision-based navigational aid (NAVAID) when the GPS becomes unavailable during UAS operations. In order to establish the viability of real-time application, different image processing methods are investigated and recommended based on their computational execution time. Imagery resolution requirements are also determined using various EO sensors, ranging from dedicated high-resolution aerial mapping sensors to relatively cheaper commercial off-the-shelf (COTS) camera sensors. Lastly, pose estimation results from different flight altitudes and flight maneuvers are compared against reference GPS data to understand vision-based NAVAID performance and limitations.

C. THESIS STRUCTURE

To address the problem formulated in Part B of this chapter, the subsequent chapters of this thesis are organized as follows:

Chapter II introduces the basics of visual-based odometry. The chapter presents the derivation of change in UAS position, followed by an overview of the feature detection algorithms available and geometric transformations.

Chapter III details the image processing procedure, which includes feature detection, matching, outlier rejection, and eventual geometric transformation decomposition to obtain pose estimation.

The experiment performed using the Quantum Systems Trinity F90+ mapping UAS, featuring an ultra-high resolution (UHR) Sony RX1R II 42.4 MP camera is described

in Chapter IV. While the UAS is a good mapping drone in the industry, it does not necessarily support research work as time synchronized telemetry and imagery data are not available.

In Chapter V a description of the experiment conducted using a manned aircraft equipped with a TASE 200 sensor is described. This sensor does provide telemetry synchronized with the imagery but of a lower resolution quality.

Chapter VI recounts the attempt to prototype a cheap sensor-telemetry suite featuring high-resolution imagery and high-rate telemetry data, built using COTS parts.

Then, the attempts to implement NAVAID estimations into UAS navigation architecture in the MATLAB environment are described in Chapter VII.

Finally, Chapter VIII provides concluding remarks about the work completed in this thesis and offers recommendations for further research.

THIS PAGE INTENTIONALLY LEFT BLANK

II. MATHEMATICAL FOUNDATION OF VISION-BASED ODOMETRY

A. DERIVATION OF CHANGE IN UAS POSITION

The position of a target point of interest on a local tangent coordinate plane $\{n\}$, $P_T^{\{n\}}$, is represented on an image plane shown in Figure 4. This target point position can be derived from its position in the camera coordinate frame $\{c\}$, $p_c^{\{c\}}$, through a series of transformations; firstly by rotating and translating it to the body coordinate frame $\{b\}$ via ${}^{\{b\}}R$ and $p_c^{\{b\}}$, and subsequently rotating and translating it to the normal tangent plane $\{n\}$ via ${}^{\{n\}}R$ and $P_{a/c}^{\{n\}}$

$$P_T^{\{n\}} = P_{a/c}^{\{n\}} + {}^{\{n\}}R(p_c^{\{b\}} + {}^{\{b\}}R p_T^{\{c\}}) \quad (1)$$

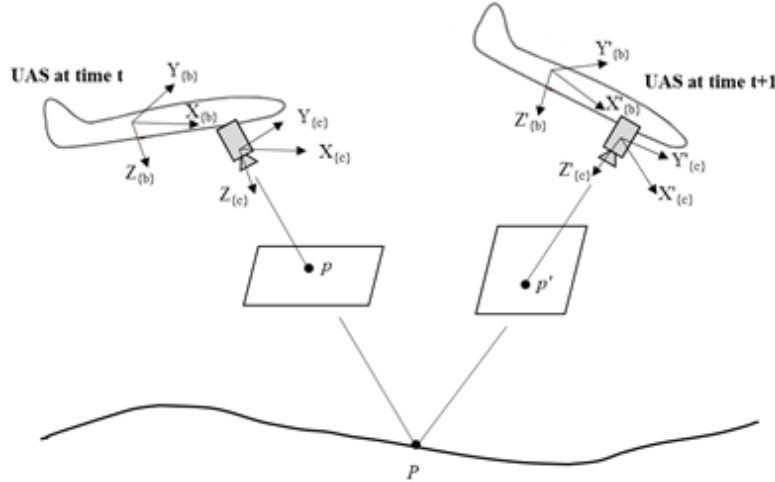


Figure 4. Relationship between UAS state, camera orientation, and target point of interest during transit

By rearranging terms in Equation (1) with the known positions of the target point in $\{n\}$ and $\{c\}$, an unknown UAS position can be expressed as

$$P_{a/c}^{\{n\}} = P_T^{\{n\}} - {}^{\{n\}}R(p_c^{\{b\}} + {}^{\{b\}}R p_T^{\{c\}}) \quad (2)$$

The rotation matrix ${}^{\{n\}}R_{\{b\}}$ can be generated from Euler angles, provided by INS measurements, with the rotation order of “ZYX”; the rotation order is critical as it will affect the result of matrix multiplication (Yakimenko and Slegers 2015, 267).

$${}^{\{n\}}R_{\{b\}} = R_z(\psi)R_y(\theta)R_x(\varphi) \quad (3)$$

$${}^{\{n\}}R_{\{b\}} = \begin{pmatrix} \cos \psi & -\sin \psi & 0 \\ \sin \psi & \cos \psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos \theta & 0 & \sin \theta \\ 0 & 1 & 0 \\ -\sin \theta & 0 & \cos \theta \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos \varphi & -\sin \varphi \\ 0 & \sin \varphi & \cos \varphi \end{pmatrix}$$

$${}^{\{n\}}R_{\{b\}} = \begin{pmatrix} \cos \theta \cos \psi & \sin \varphi \sin \theta \cos \psi - \cos \varphi \sin \psi & \cos \varphi \sin \theta \cos \psi + \sin \varphi \sin \psi \\ \cos \theta \sin \psi & \sin \varphi \sin \theta \sin \psi + \cos \varphi \cos \psi & \cos \varphi \sin \theta \sin \psi - \sin \varphi \cos \psi \\ -\sin \theta & \sin \varphi \cos \theta & \cos \varphi \cos \theta \end{pmatrix}$$

The location of the camera on the UAS, represented by $p_c^{\{b\}}$, is fixed, and its orientation, ${}^{\{b\}}R_{\{c\}}$, can be defined with the gimbal’s known pan and tilt angle values

$${}^{\{b\}}R_{\{c\}} = R_y(Tilt)R_z(Pan) \quad (4)$$

Unlike other map-based navigation research presented earlier in Chapter I, which relies on known reference target points on visual images, the current proposed approach assumes that there is no available target point with a known geolocation; $P_T^{\{n\}}$ is therefore unknown. Fortunately, by post-processing two consecutive images, it can be inferred that any change in view captured in the images will be due to the difference in position and orientation of the camera when the two images were captured. Revisiting Equation (2), after including differences in position and orientation, the following is obtained:

$$P_{a/c}^{\{n\}} + \Delta P_{a/c}^{\{n\}} = P_T^{\{n\}} - ({}^{\{n\}}R_{\{b\}} + \Delta {}^{\{n\}}R_{\{b\}})[p_c^{\{b\}} + ({}^{\{b\}}R_{\{c\}} + \Delta {}^{\{b\}}R_{\{c\}})(P_T^{\{c\}} + \Delta P_T^{\{c\}})] \quad (5)$$

The change in the target point position $\Delta p_T^{\{c\}}$ could have been produced by the change in UAS position $\Delta p_{a/c}^{\{n\}}$ and attitude $\Delta {}^{\{n\}}R_{\{b\}}$, as well as gimbal orientation $\Delta {}^{\{b\}}R_{\{c\}}$. Subtracting Equation (5) from Equation (2) and ignoring any second order terms yields

$$\Delta P_{a/c}^{\{n\}} = -\Delta {}^{\{n\}}R_{\{b\}}(p_c^{\{b\}} + {}^{\{b\}}R_{\{c\}}P_T^{\{c\}}) - {}^{\{n\}}R_{\{b\}}\Delta {}^{\{b\}}R_{\{c\}}P_T^{\{c\}} - {}^{\{n\}}R_{\{b\}}{}^{\{b\}}R_{\{c\}}\Delta P_T^{\{c\}} \quad (6)$$

It is interesting to note that the term $P_T^{\{n\}}$ is no longer present, indicating that there is no requirement to know the geolocation of the target point and that only the change of the target point's position $\Delta P_T^{\{c\}}$ is the unknown term to resolve. Equation (6) can be further rearranged to obtain

$$\Delta P_{a/c}^{\{n\}} = -(P_T^{\{n\}} - P_{a/c}^{\{n\}}) \frac{\Delta_{\{b\}}^{\{n\}} R}{\{n\} R_{\{b\}}} - \{n\} R \Delta_{\{c\}}^{\{b\}} R p_T^{\{c\}} - \{n\} R_{\{c\}}^{\{b\}} R \Delta p_T^{\{c\}} \quad (7)$$

Using the expression for the change of the direction cosine matrix caused by rotation of $\{b\}$ with respect to $\{n\}$ (where p , q , and r are the components of the corresponding angular velocity vector)

$$\Delta_{\{b\}}^{\{n\}} R = \{n\} R [I + S(\omega_{\{b\} wrt \{n\}}^{\{b\}}) \Delta t], S(\omega_{\{b\} wrt \{n\}}^{\{b\}}) = \begin{pmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{pmatrix} \quad (8)$$

Equation (7) can then be rearranged and eventually expressed as

$$\Delta P_{a/c}^{\{n\}} = -(P_T^{\{n\}} - P_{a/c}^{\{n\}}) [I + S(\omega_{\{b\} wrt \{n\}}^{\{b\}})] - \{n\} R \Delta_{\{c\}}^{\{b\}} R p_T^{\{c\}} - \{n\} R_{\{c\}}^{\{b\}} R \Delta p_T^{\{c\}} \quad (9)$$

B. FEATURE EXTRACTION

If the camera is displaced when acquiring two consecutive images, it can be safely assumed that some features are present in both images, and the features can be correlated from one image frame to the other through geometric transformations. In order to utilize Equation (6) to determine the change in UAS position, all corresponding parameters of the UAS and its gimbal need to be known, as well as the detection of displaced target points represented by $\Delta p_T^{\{c\}}$.

Image matching or image registration is fundamentally based on matching features in images, using computer vision algorithms. These algorithms find specific combinations of pixels, such as corners, blobs, and edges within the image, which can also be referred to as local features. The local features differ from their immediate surrounding pixels by texture, color, or intensity and can be distinctively identified. It is important that these

features remain locally invariant, so that feature detection can be executed even when the image is rotated or altered in scale. The identification of a correct match of target points between two images can be made using the Hamming-distanced-based metric to facilitate the similarity matching.

The corner detection algorithm works specifically on the intersection of two edges; when a small survey window is placed at a corner on an image, there will be a great change in intensity when the survey window is displaced in any direction. Figure 5 illustrates the corner detector at the corner in the image. This detection algorithm is applicable for point tracking and image registration with little or no scale change. Examples of corner detection algorithms include the Features from Accelerate Segment Test (FAST) (Rosten and Drummond 2005), Minimum Eigenvalue algorithm (MinEigen) (Shi and Tomasi 1994), and the Harris-Stephen corner detector (Harris) (Harris and Stephens 1988).

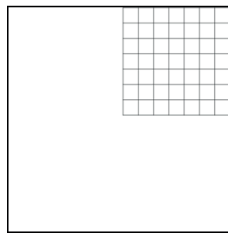


Figure 5. Image with corner detector

Improved corner detectors subsequently created to handle scale and rotation changes provide robustness in the image matching process. Oriented FAST and rotated BRIEF (ORB) (Rublee et al. 2011) is a multiscale corner detector that can handle rotated images, while Binary Robust Invariant Scalable Keypoints (BRISK) (Leutenegger, Chli and Siegwart 2011) is another multiscale corner detector that can also handle scale variations.

Blob detection algorithms are designed to detect regions in the image that are of the same properties when compared to surrounding pixels, illustrated in Figure 6. These region properties include brightness and color. Blob detection algorithms are also multiscale detectors, with the ability to handle changes in scale and rotation of images. Examples of blob detection algorithms include Speeded-Up Robust Features (SURF) (Bay et al. 2008) and

KAZE (Alcantarilla, Bartoli and Davison 2012). MSER (Nistér and Stewénus 2008) is also a blob detection method, but it is more robust to affine transformation than SURF and KAZE.

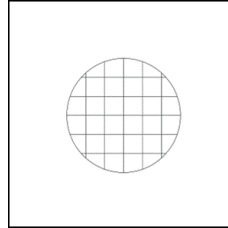


Figure 6. Image with blob detector

Table 1 provides the summary of the different detector algorithms analyzed for feature extraction, emphasizing their invariance to scale and rotation, which is a necessary attribute for the vision-based UAS navigation concept.

Table 1. Summary of feature extraction methods

Detector	Feature Type	Invariance		Typical Use	
		Scale	Rotation	Finding Point Correspondences	Classification
Corner Detector (FAST, MinEig, Harris)	Corner	No	No	Yes	No
ORB	Corner	No	Yes	Yes	No
BRISK	Corner	Yes	Yes	Yes	No
SURF	Blob	Yes	Yes	Yes	Yes
KAZE	Blob	Yes	Yes	Yes	Yes
MSER	Blob (Region with uniform intensity)	Yes	Yes	Yes	No

Good features in images are desired for feature extraction algorithms to work effectively. They shall be distinctive from neighboring pixels, uniquely localizable, and robust to changes in viewing conditions and presence of noise.

After good features are identified across two images, they are matched and labeled as inliers. Elimination of outliers is facilitated by the M-estimator Sample Consensus (MSAC) algorithm. The MSAC algorithm is a variant of the Random Sample Consensus (RANSAC) algorithm but known to be more accurate with the use of a loss function (Choi, Kim, and Yu 2009). Due to the randomized nature of the MSAC algorithm, the exclusion of outliers will differ across multiple runs and is not repeatable. Fortunately, the high number of matched inliers in feature-rich images can mitigate the issue of handling outliers that are not excluded.


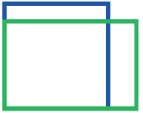

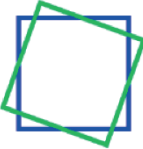
C. HOMOGRAPHY AND TRANSFORMATIONS

With a set of inliers identified across two images, a matching 2D geometric transformation to determine transformation matrix ${}_{\{i1\}}^{\{i2\}}R$ can be executed. Through the geometric transformation, $u-v$ coordinates of the points in the first image $i1$ can be expressed in the coordinate system of the second image $i2$, as follows:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix}_{i2} = {}_{\{i1\}}^{\{i2\}}R \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}_{i1} \quad (10)$$

Table 2 summarizes the elementary geometric transformations, which will be used in combination for subsequent transformation methods.

Table 2. Elementary transformations

Transformation	Geometric Interpretation	Transformation Matrix	Notes
Translation		${}^tR = \begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix}$	t_x specifies the displacement along the u -axis t_y specifies the displacement along the v -axis
Scaling (non-proportional)		${}^sR = \begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{pmatrix}$	s_x specifies the scale factor along the u -axis s_y specifies the scale factor along the v -axis
Shear		${}^{sh}R = \begin{pmatrix} 1 & sh_x & 0 \\ sh_y & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	sh_x specifies the shear factor along the u -axis sh_y specifies the shear factor along the v -axis
Rotation		${}^\theta R = \begin{pmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{pmatrix}$	θ specifies the angle of rotation

A transformation that preserves shapes, such as ratios of length, angles, and ratio of areas and parallel lines, is called a similarity transformation. A similarity transformation includes translation, proportional scaling, rotation, or combinations. The transformation matrix of similarity type is expressed as

$${}_{\{i1\}}^{\{i2\}}R_s = {}^tR {}^sR {}^\theta R = \begin{pmatrix} s \cos \theta & -s \sin \theta & t_x \\ s \sin \theta & s \cos \theta & t_y \\ 0 & 0 & 1 \end{pmatrix} \quad (11)$$

The transformation matrix ${}_{\{i1\}}^{\{i2\}}R$, expressed in four unknown parameters t_x , t_y , s , and θ , can be obtained through optimization. Thereafter, these parameters can be decomposed from the transformation matrix by expressing them as elements of the matrix, generating

$$t_x = r_{31}, t_y = r_{32}, s = \sqrt{r_{11}^2 + r_{21}^2}, \theta = \tan^{-1}(r_{21} / r_{11}) \quad (12)$$

An affine transformation is one that preserves lines and parallelism, which includes non-proportional scaling and shear transformations in addition to those in similarity transformation. The transformation matrix of affine type is expressed as

$${}_{\{i1\}}^{\{i2\}}R_a = {}^tR^sR^{sh}R^\theta R = \begin{pmatrix} s_x (\cos \theta + sh_x \sin \theta) & -s_x (\sin \theta - sh_x \cos \theta) & t_x \\ s_y (\sin \theta + sh_y \cos \theta) & s_y (\cos \theta - sh_y \sin \theta) & t_y \\ 0 & 0 & 1 \end{pmatrix} \quad (13)$$

The affine transformation matrix ${}_{\{i1\}}^{\{i2\}}R$ is expressed in seven unknown parameters t_x , t_y , s_x , s_y , sh_x , sh_y , and θ . While t_x , t_y can be found using Equation (12) that was established previously, the remaining five unknown parameters cannot be defined independently with only four equations. By assuming proportional scaling, where $s_x = s_y = s$, the other parameters can be deduced using

$$s = \frac{r_{11}r_{12} + r_{12}r_{22}}{\sqrt{(r_{11} - r_{22})^2 + (r_{21} + r_{12})^2}}, \theta = 2 \tan^{-1} \left(\frac{\sqrt{(r_{11} - r_{22})^2 + (r_{21} + r_{12})^2} - (r_{21} + r_{12})}{(r_{11} - r_{22})} \right) \quad (14)$$

$$sh_x = \frac{r_{11}^2 - r_{22}r_{11} + r_{12}^2 + r_{21}r_{12}}{r_{11}r_{21} + r_{12}r_{22}}, sh_y = \frac{r_{21}^2 + r_{12}r_{21} + r_{22}^2 - r_{11}r_{22}}{r_{11}r_{21} + r_{12}r_{22}}$$

A projective transformation is one that maps lines to lines but does not necessarily preserve parallelism. An example of a projective transformation is a perspective transformation, where parallel lines intersect toward infinity and circles distort to become ellipses. In a 2D projective transformation, unlike in the affine transformation shown in Equation (15), there are no restrictions on the last row of the transformation matrix. Equation represents how $u-v$ coordinates of the points in the first image $i1$ can be

expressed in the coordinate system of the second image i_2 through a projective transition matrix.

$$\begin{pmatrix} u\lambda^{-1} \\ v\lambda^{-1} \\ \lambda \end{pmatrix}_{i_2} = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & 1 \end{pmatrix} = \begin{pmatrix} u \\ v \\ 1 \end{pmatrix}_{i_1} \quad (15)$$

Figure 7 provides graphical representations of similarity, affine, and projective geometric transformations of a checkerboard.

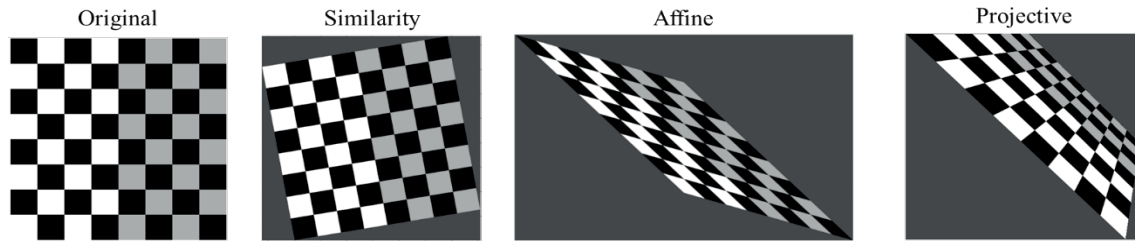


Figure 7. Geometric transformations of checkerboard.
Source: MathWorks (n.d.)

Indisputably, having a greater number of matched pairs of points will increase the accuracy of the estimated transformation matrix. However, depending on the quality of the images acquired, the possibility exists that the detection algorithm returns a small number of matched points. Therefore, it is crucial to be aware of the minimum number of matched pairs of points for each transformation type, which are summarized in Table 3.

Table 3. Minimum requirements for geometric transformations

Transformation Type	Number of Unknown Parameters	Minimum Number of Matched Pairs of Points
Similarity	4	2
Affine	6	3
Projective	8	4

HIS PAGE INTENTIONALLY LEFT BLANK

III. IMAGE PROCESSING

Image processing for a vision-based NAVAID can be broadly categorized into four phases: feature detection, feature matching, outlier rejection, and transformation decomposition. The four phases, illustrated in Figure 8, were executed in the specified order on every pair of consecutive images that are acquired. All NAVAID image processing was executed in MATLAB.

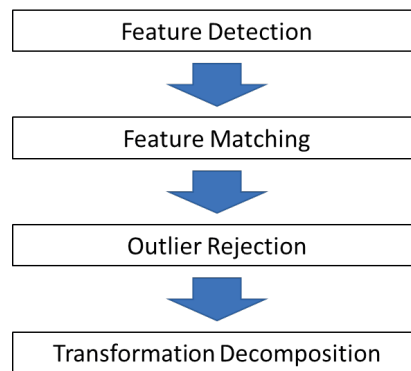


Figure 8. Vision-based NAVAID image processing sequence

A. FEATURE DETECTION

While color images provide a wider range of enhanced features for feature detectors (Van de Weijer, Gevers, and Bagdanov 2006; Burghouts and Geusebroek 2009), they are also known to be sensitive to noise, exposure, and quality of the camera sensor (Ebner 2007). To overcome these issues and take advantage of faster processing speed, grayscale images are preferred and are explored for applicability of the identified feature detection algorithms. Figure 9 shows aerial views of the same grass field in a) EO RGB and b) grayscale formats. Near infrared (NIR) format image was available, illustrated in Figure 9 c), and also included for subsequent performance comparison of detection algorithms.

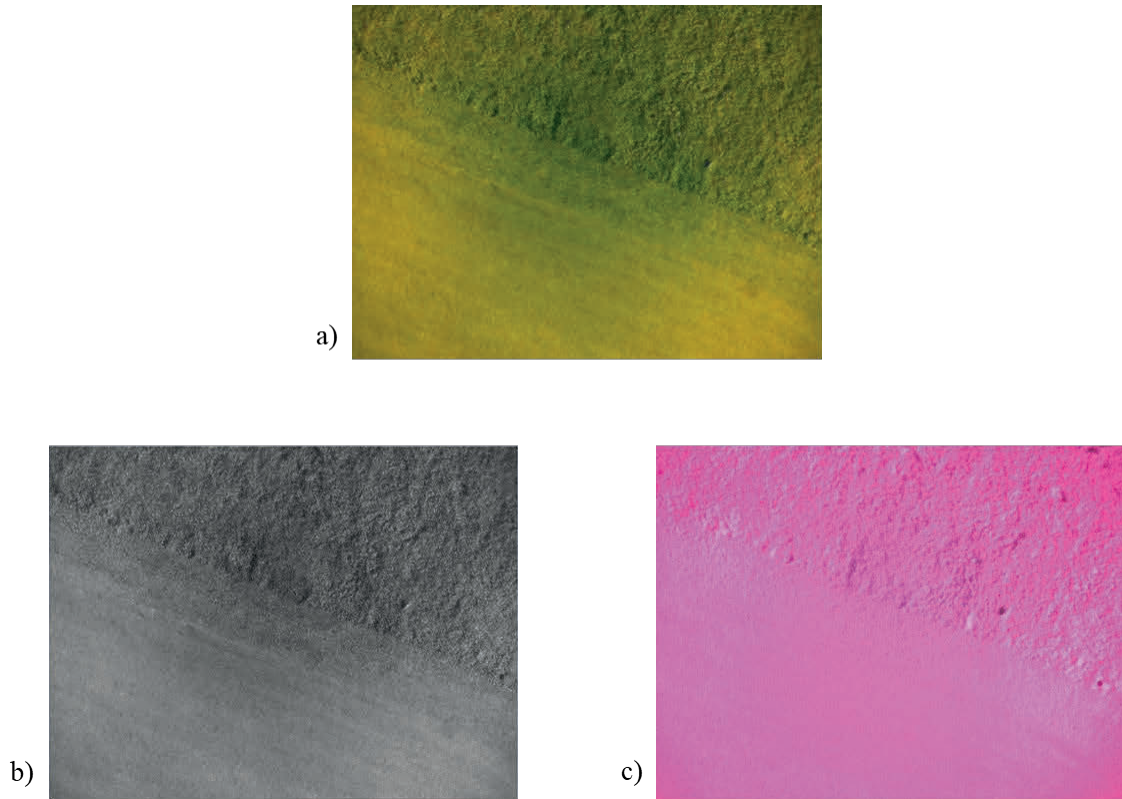


Figure 9. Grass field aerial views

It was observed that even for a featureless landscape, the detection algorithms were proven effective, with a substantial number of features detected, as shown in Figure 10. Analysis of feature detection results reveals that a) the ORB method found 558,273 feature points, consuming just 0.585 s of central processing unit (CPU) time in the MATLAB interpretative environment; b) the BRISK method found 21,289 features in 0.358 s; c) the FAST method found 15,088 features in 0.118 s; d) the Harris method found 25,109 features in 1.77 s; e) the KAZE method found 188,643 features in 21.8 s; f) the MSER method found 3,804 features in 1.85 s; g) the MinEig method found 221,381 features in 21.8 s; and h) the SURF method found 11,706 features in 0.316 s.

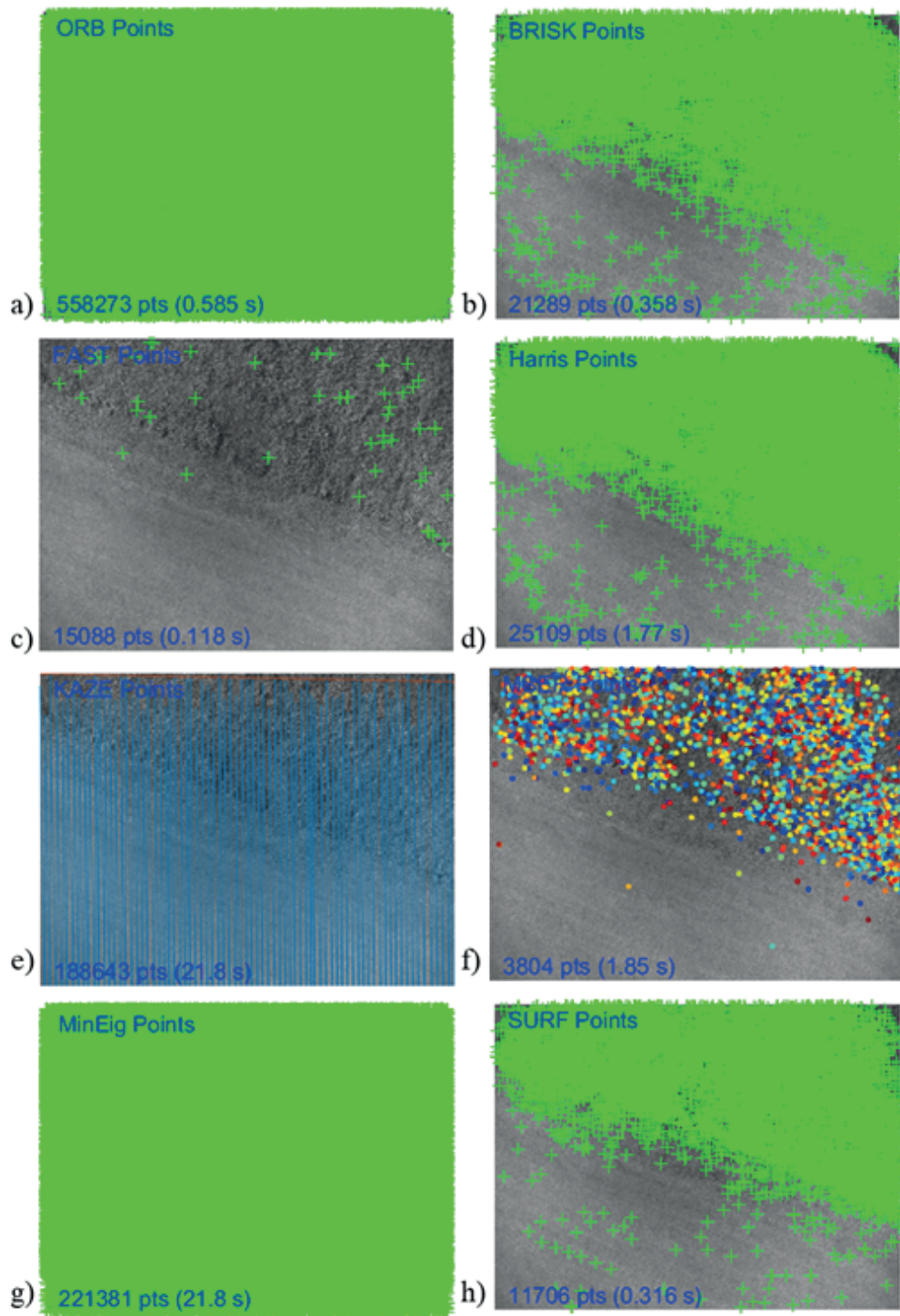


Figure 10. Visualization of feature points detected using different methods

The performance of the various detection algorithms was relatively consistent, regardless of image format, between EO and NIR. As seen in Figure 11, the ORB method was able to detect the greatest number of features among methods invariant to scale and rotation changes. While the Harris, KAZE, and MinEig methods do not pale in comparison in terms of number of features detected, they require more processing time than the ORB method to find features in images, as illustrated in Figure 12. It is also interesting to note that the FAST method outclassed all other methods, having the least processing time required, with the SURF method not far behind.

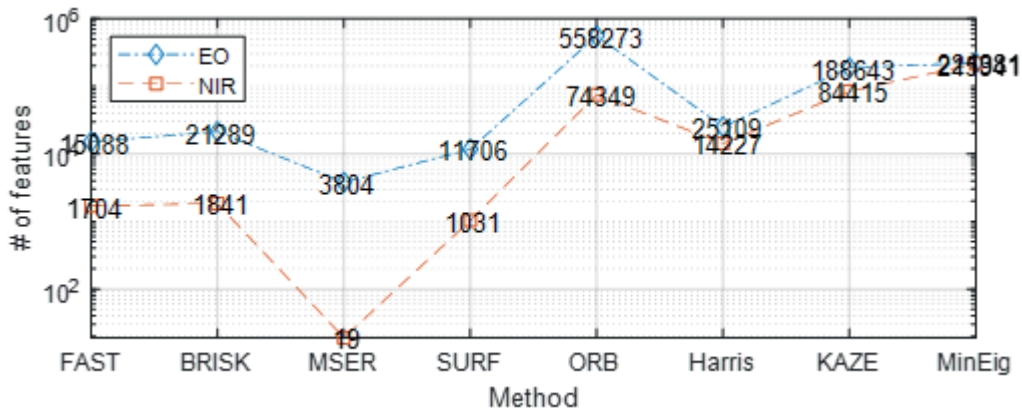


Figure 11. Number of features detected with respect to detection algorithms

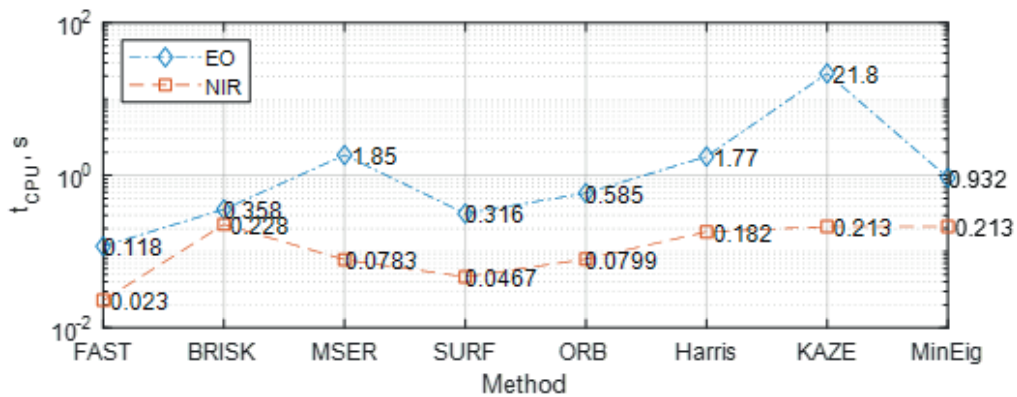


Figure 12. CPU execution time to extract features with respect to detection algorithms

For the vision-based NAVAID to be useful in real-time applications, it is crucial for the processing time to be short. By normalizing the results from Figure 11 and Figure 12, representing them in terms of processing time required per feature detected, as illustrated in Figure 13, the performance of various detection methods can be better appreciated. Having a low processing time per feature detected highlights the feasibility of implementation for real-time flight application. Clearly, the ORB method surpassed the other detection methods. In addition, Figure 14 depicts an alternative interpretation of the performance of detection methods. An efficient detection algorithm can be found at the lower right corner of Figure 14, which translates to having found more features in less time. Considering only results from processing the EO image, it can be observed that both the ORB and the FAST detection methods have superior performance over other algorithms, and so, they are selected for subsequent image processing. The SURF detection method shall also be included in subsequent analysis to explore the feasibility of using a blob feature detection method.

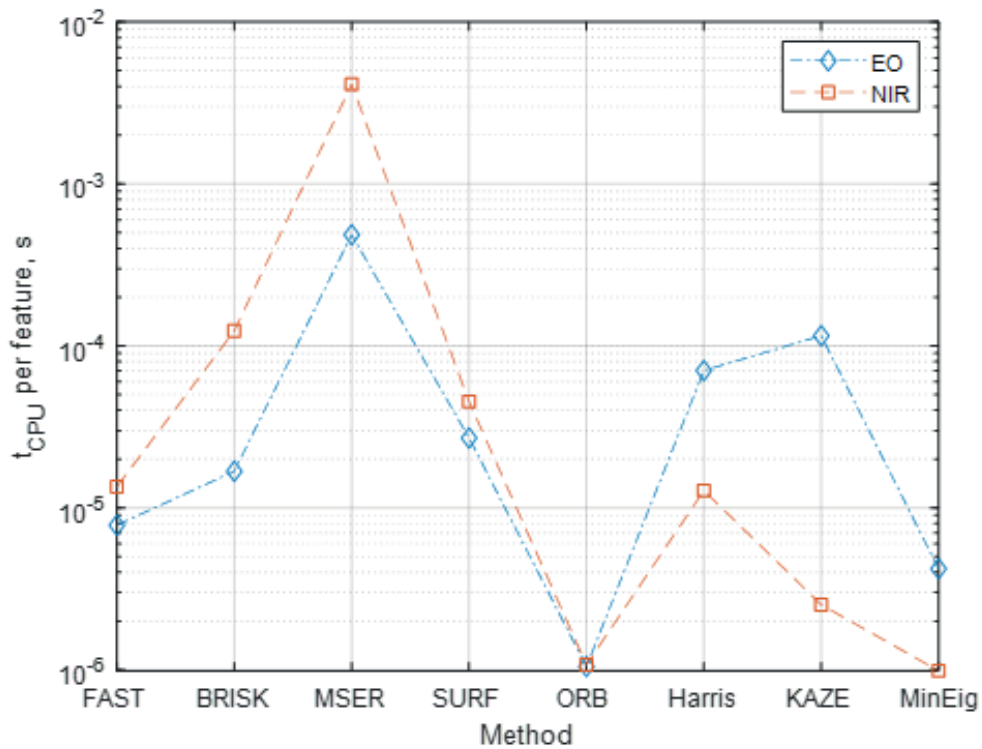


Figure 13. Average CPU time required for feature detection with respect to detection methods

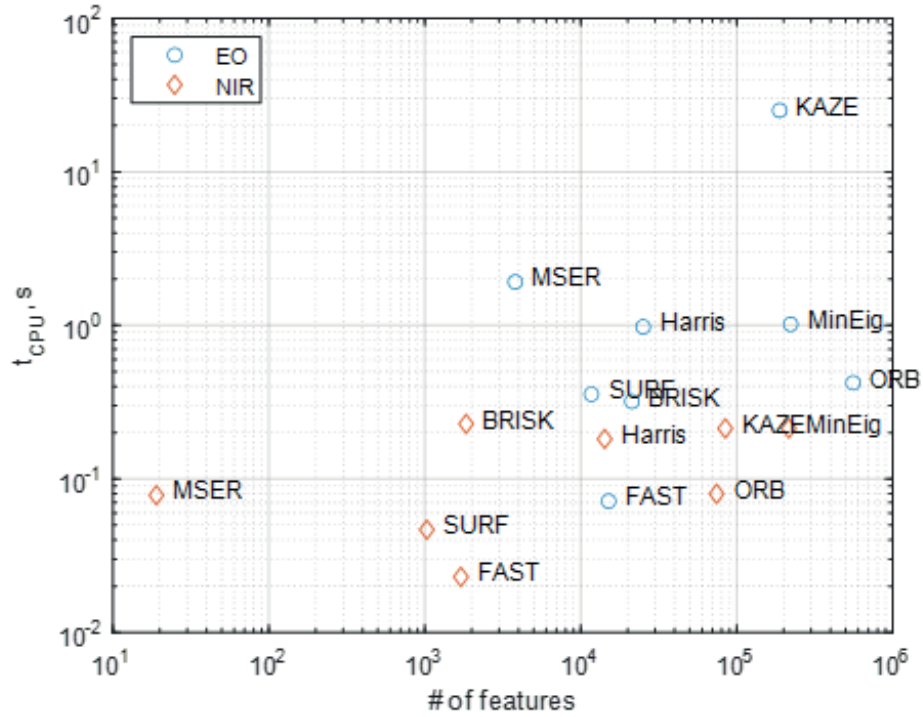
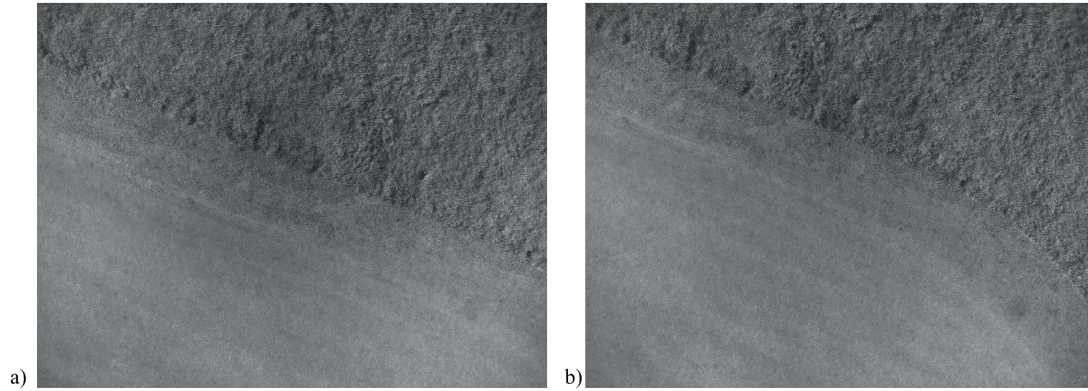


Figure 14. CPU execution time with respect to number of features detected

B. FEATURE MATCHING

Extracted features from two consecutive images will be processed for matching to pair similar features observed in both images. Figure 15 shows a sample of two consecutive aerial images of a grass field that was used to illustrate feature matching. After features were detected on each image, Hamming distance was used to compute the similarity metric and lines were drawn between each pair of matched features, shown in Figure 16. It was observed that the lines consisted of both inliers and outliers matches, which indicated a need to apply MSAC to exclude outliers before subsequent transformation decomposition.



Two consecutive grayscale images captured sequentially: a) Reference image and b) Target image taken some time unit after.

Figure 15. Two consecutive aerial images of grass field

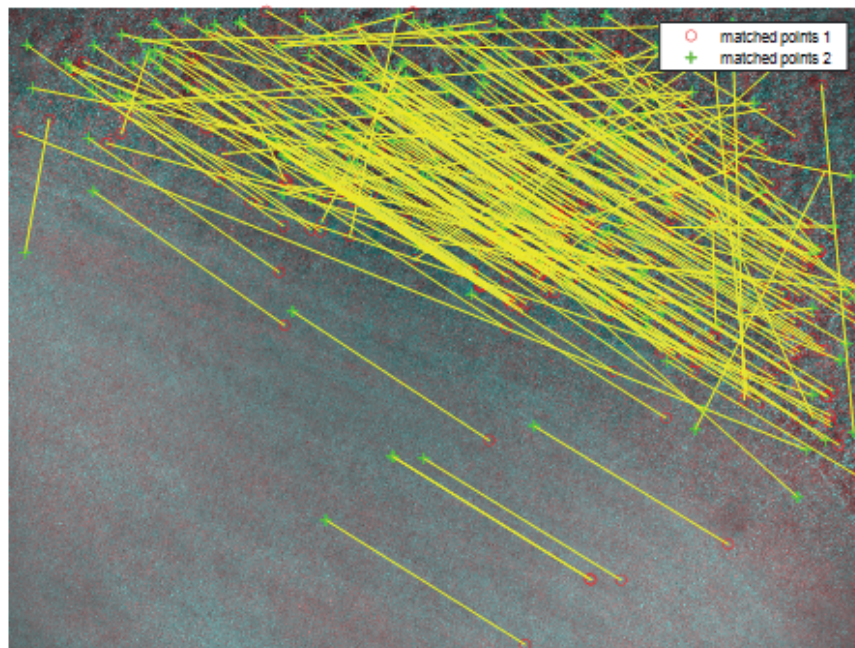


Figure 16. All feature matchings between two consecutive images from SURF detection method

C. OUTLIER REJECTION

Removal of outlier feature matchings was executed using the MSAC algorithm, with the remaining inliers shown in Figure 17. Similar to the RANSAC algorithm, the MSAC algorithm is an iterative method to detect outliers, such that remaining inliers are

coherent to estimate the transformation matrix. Although the percentage of matched features at 2.12% seemed low, it had 248 matched features and fulfilled the minimum requirements for geometric transformation stated in Table 3.

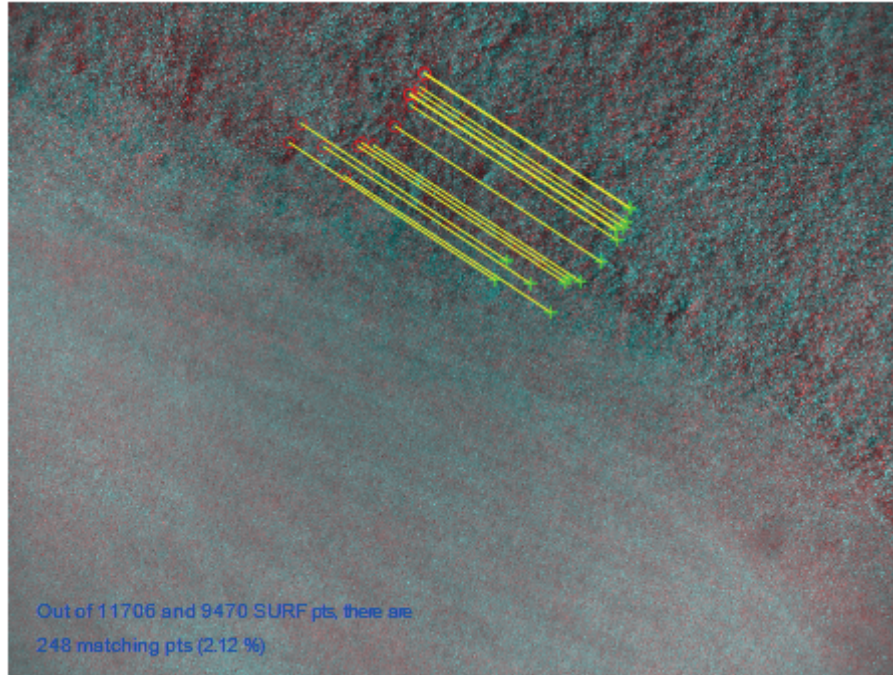


Figure 17. Matched inliers between two consecutive images from SURF detection method

D. TRANSFORMATION DECOMPOSITION

While affine and projective transformations consider non-proportional scaling and shear transformations, similarity transformation was assumed sufficient for aerial imagery captured at high cruise altitude. The geometric transformation matrix ${}_{\{i1\}}^{\{i2\}}R$ was estimated by using the remaining matched inliers in MATLAB. Using similarity transformation matrix decomposition, reflected in Equation (12), translation, scaling, and rotation parameters were determined by mapping to elements of the matrix. Results of the transformation decomposition are summarized in Table 4, with the transformed image illustrated in Figure 18.

Table 4. Summary of estimated transformation parameters

Transformation Parameter	Value
t_x	-896.361 pix
t_y	-647.977 pix
s	1.407
θ	-0.946°

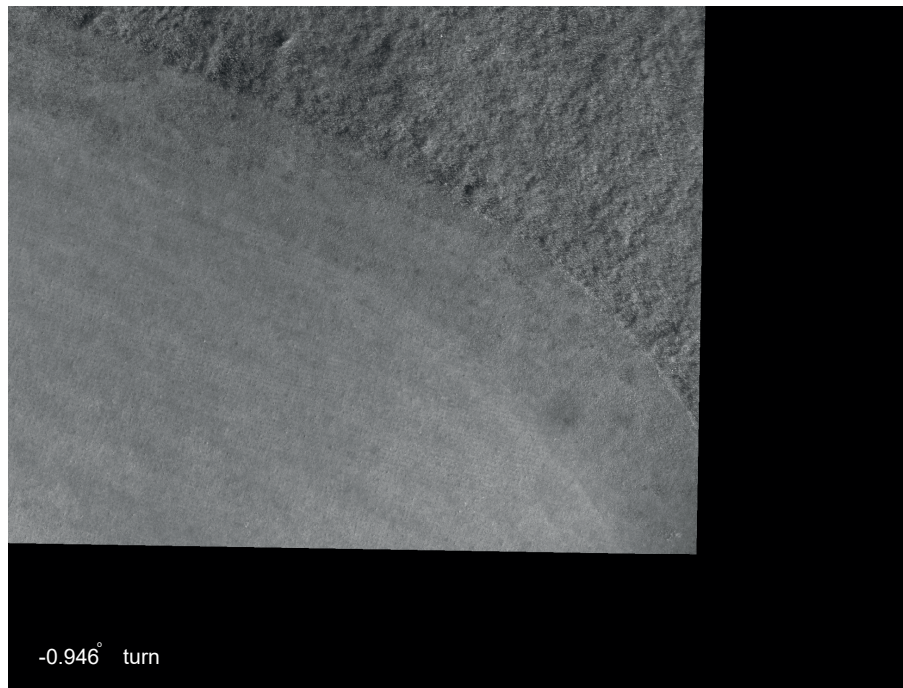


Figure 18. Recovered target image

THIS PAGE INTENTIONALLY LEFT BLANK

IV. EXPERIMENTING WITH QUANTUM TRINITY F90+ MAPPING UAS

This chapter describes the experiments involving the Quantum Systems Trinity F90+ mapping UAS equipped with an UHR EO sensor. While it is ideal to have synchronized time-stamped EO imagery and telemetry data, the following work investigates the feasibility of generating a vision-based NAVAID solution using unsynchronized data collected from other flight missions.

A. TEST SETUP

The Trinity F90+ mapping drone in Figure 19 is an electric vertical take-off and landing (eVTOL) fixed-wing mapping drone developed by Quantum Systems GmbH, Germany. It was primarily designed for geographical mapping purposes, combining both the eVTOL feature of a classic multicopter and the long range of a fixed-wing UAV. Its robust design also allows multiple payload configurations to be implemented, quickly and interchangeably. For the flight campaign over Camp Roberts, the Sony RX1R II payload configuration was selected, which provided high resolution aerial photographs. A summary of the Trinity F90+ mapping drone technical data and image metadata is presented in Table 5 and Table 6, respectively.



Figure 19. a) Trinity F90+ mapping drone with b) Sony RX1R II camera.
Source: Quantum Systems (n.d.).

Table 5. Technical data of Trinity F90+ mapping drone. Adapted from Quantum Systems (n.d.).

“Max. Take-off Weight”	“5.0 kg (11.0 lb)”
“Max. Flight Time”	“90+ min” “60 min (locked)”
“Max. Range (Area Coverage)”	“100 km = 700 ha” “(62 mi = 1730 ac)”
“Maximum Flight Altitude (MSL)”	“4500 m (14,763.8 ft)”
“Command and Control Range”	“5 – 7.5 km (3.1 – 4.7 mi)”
“Payload (with compartment)”	“max. 700 g (1.54 lbs)” Sony RX1R II 35 mm CMOS sensor with 42.4 megapixels
“Optimal Cruise Speed”	“17 m/s (33 kn)”
“Wind Tolerance (ground)”	“up to 9 m/s (17.5 kn) < 1500 m MSL” “up to 7 m/s / (13,6 kn) 1500 m – 3000 m MSL” “up to 5 m/s / (9.7 kn) > 3000 m MSL”
“Wind Tolerance (cruise)”	“up to 12 m/s (23.3 kn)”
“Battery Weight”	“1.5 kg (3.3 lb)”
“Telemetry Link & RC Transmitter Frequency”	“2.4 GHz”

Table 6. Metadata of images captured by Sony RX1R II

Dimensions (width by height)	7,952 pix by 5,304 pix
Horizontal/Vertical Resolution	350 dpi
F-stop	f/4
Exposure Time	1/2000 s
ISO Speed	ISO-250
Focal Length	35 mm

Flight planning for the Trinity F90+ drone was handled entirely by its proprietary software QBase 3D. Users do not have the option to define a fixed image acquisition rate. Instead, the acquisition rate was governed by the percentage of surveillance overlap setting prescribed in QBASE 3D and the drone’s autopilot would decide the instance to trigger the camera shutter. The image acquisition time interval was also affected by in-flight wind

conditions, which caused variations in flight speeds. There were several options for the drone to fly over waypoints, depicted in green, catered for various aerial mapping requirements, illustrated in Figure 20.

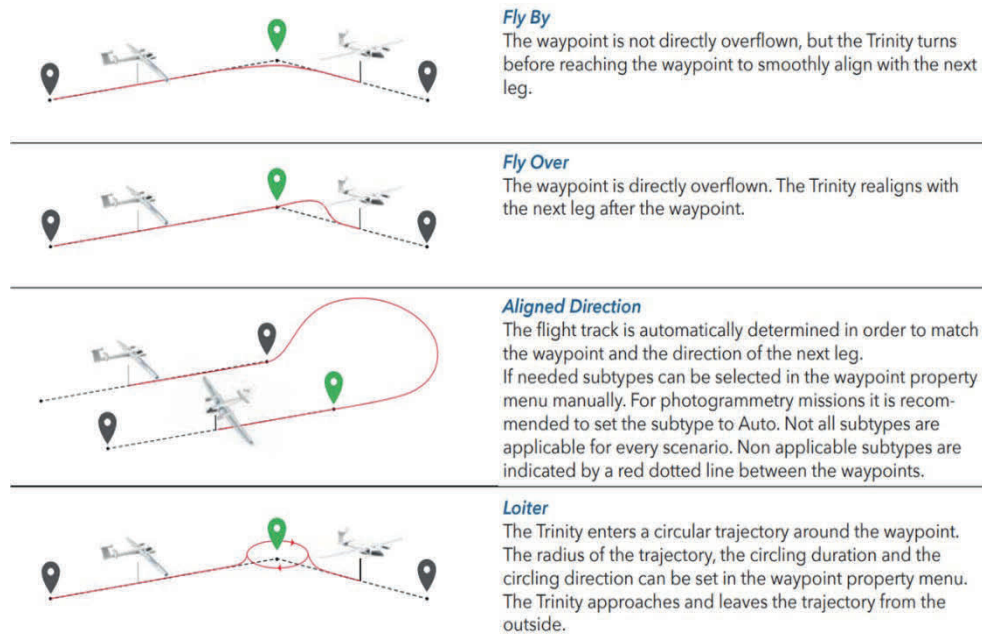


Figure 20. Trinity F90+ flight over waypoint options.
Source: Quantum Systems (n.d.).

B. DATA PROCESSING

For the feasibility study of generating a vision-based NAVAID solution, the Trinity F90+ mapping drone executed a flight over Camp Roberts, San Miguel, California. A completed set of telemetry data with corresponding imagery files was obtained post flight for the development of a vision odometry algorithm.

The flight campaign commenced from Camp Roberts, transiting over mountainous and open grazing land, before returning to base. An overview of the flight trajectory, plotted using post-flight telemetry data in Google Earth, is illustrated in Figure 21. The star icon in Figure 21 depicts the start and end points of the flight campaign. The flight began with a transition to the west, turned north, and followed a clockwise profile before returning to ground. Due to frequent terrain elevation changes, shown in Figure 22 and Figure 23,

the drone had to make frequent altitude alterations, which proved to be a challenge in subsequent trajectory matching. Telemetry data, containing the corresponding GPS coordinates, altitude, and attitude of the UAS when each image was acquired, was stored in *KML* and *CSV* files.



Figure 21. Trajectory overview of Camp Roberts flight campaign

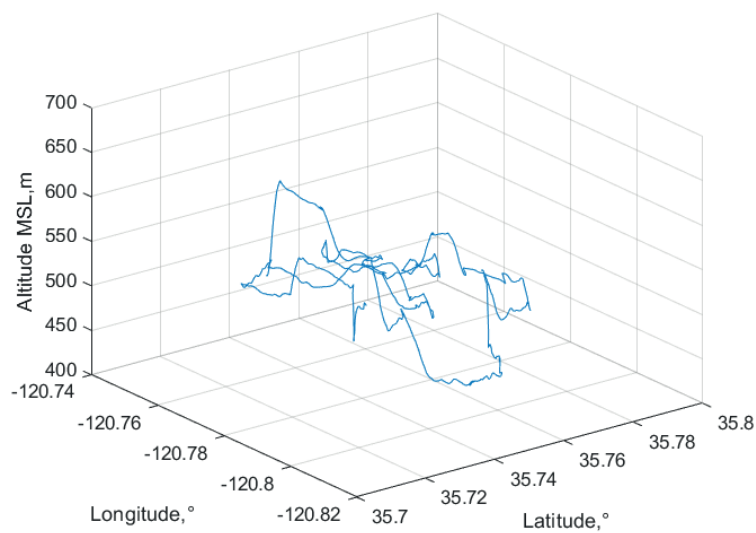


Figure 22. 3D flight profile of Camp Roberts flight campaign

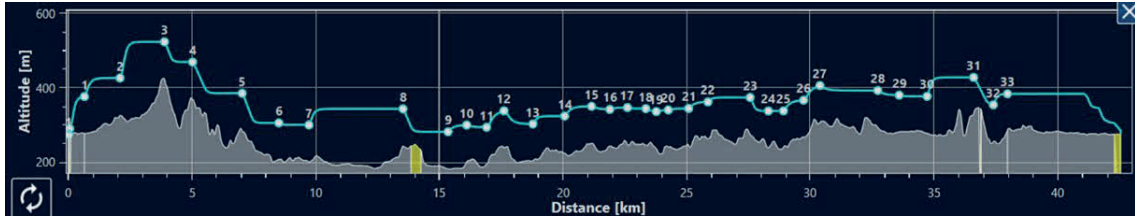


Figure 23. Terrain elevation (gray) and preplanned flight altitude (blue)

Reference flight telemetry data, recorded by Trinity F90+ drone, is plotted in Figure 24. As the drone is a COTS product, it does not support the customization of the recorded flight parameters. Fortunately, it did record crucial parameters such as corresponding GPS coordinates, drone altitude, and attitude (Euler angles) at each trigger instance of the camera payload. With GPS coordinates tagged to every image frame, the distance travelled by the drone between two image frames was computed. By correlating the translation transformation from the transformation matrix to the distance travelled across two consecutive frames, its corresponding displacement was obtained.

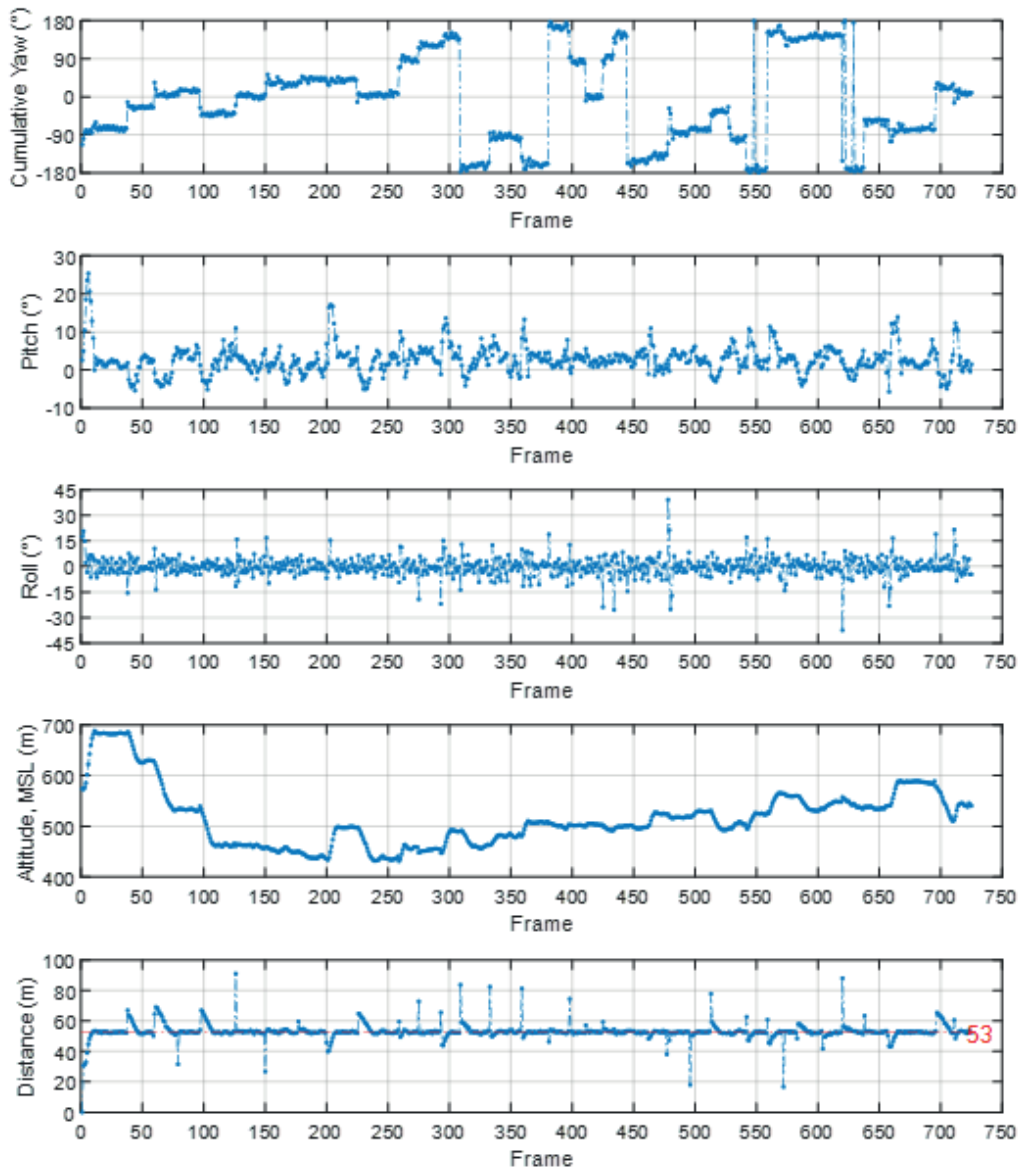


Figure 24. Flight telemetry plots of Camp Roberts flight campaign

For a flight over varying terrains and vegetation, shown in Figure 25, one can expect a large magnitude of features to be extracted by the detection algorithms. The number of features detected by the ORB, SURF, and FAST algorithms in every image are illustrated in Figure 26. It was observed that the minimum average number of features extracted across all three algorithms exceeded 20,000. This was more than sufficient to fulfill the minimum requirements for finding proper geometric transformations. Time

duration required to process all images by the FAST detection method was also noted to be the fastest among the three algorithms investigated. This observation is consistent with the summary of the CPU execution time presented earlier in Figure 12.



Figure 25. Sample images from Camp Roberts flight campaign

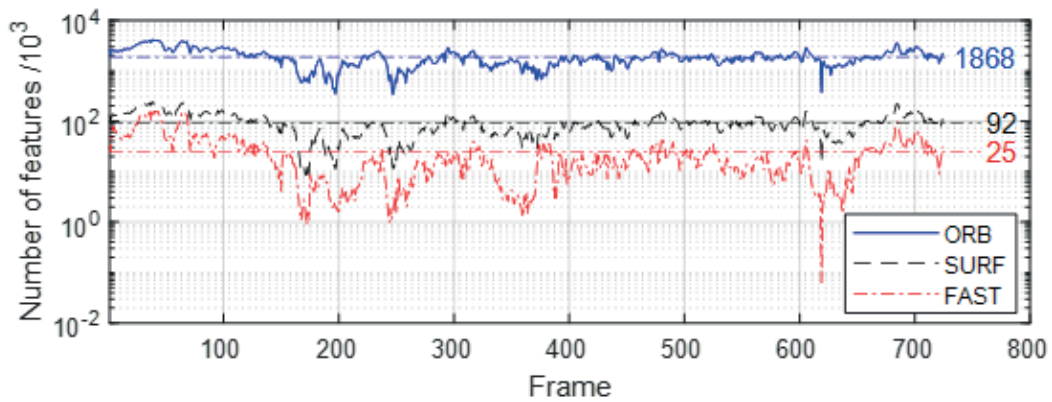


Figure 26. Number of features detected in Camp Roberts flight imagery using ORB, SURF, and FAST methods

C. KEY OBSERVATIONS

For real-time in-flight applications, the number of features that need to be processed ought to be reduced, without affecting the accuracy of the visual odometry. A sensitivity

analysis on the number of features available and their impact on estimation error was conducted. Varying the number of strongest feature points, selected based on their respective detection metrics, was analyzed and their error with reference flight telemetry data computed. Results of the sensitivity analysis are presented in Table 7, Table 8, and Table 9. It was observed that the SURF detection method performs relatively well with 5,000 strongest feature points, providing a fairly accurate estimation in a short computation time. Both ORB and FAST required at least 8,000 strongest features to be available before they were able to facilitate any trajectory estimation.

Table 7. Sensitivity analysis results for SURF detection method

Number of Strongest Feature Points	Average Number of Matched Points	Mean Yaw Error (deg)	Standard Deviation of Mean Yaw Error (deg)	CPU Time (s)
1000	137	7.08	42.99	6811
5000	618	5.80	41.33	6958
8000	947	6.16	44.47	7267
10000	1159	6.09	43.48	7640
15000	1660	5.86	42.29	10602
20000	2132	6.04	43.28	12225

Table 8. Sensitivity analysis results for ORB detection method

Number of Strongest Feature Points	Average Number of Matched Points	Mean Yaw Error (deg)	Standard Deviation of Mean Yaw Error (deg)	CPU Time (s)
1000	N.A.	N.A.	N.A.	N.A.
5000	N.A.	N.A.	N.A.	N.A.
8000	202	6.63	43.65	3070
10000	257	6.59	43.67	3301
15000	399	6.57	43.67	3816
20000	545	6.50	43.62	5049

Table 9. Sensitivity analysis results for FAST detection method

Number of Strongest Feature Points	Average Number of Matched Points	Mean Yaw Error (deg)	Standard Deviation of Mean Yaw Error (deg)	CPU Time (s)
1000	N.A.	N.A.	N.A.	N.A.
5000	N.A.	N.A.	N.A.	N.A.
8000	97	7.22	43.08	1594
10000	115	6.88	42.14	1764
15000	151	6.89	44.73	2394
20000	178	6.40	41.74	3076

Estimated yaw angles extracted from the transformation matrix were assessed and plotted against reference GPS telemetry data to provide visual comparison. Figure 27, Figure 28, and Figure 29, show the estimation results for the SURF, ORB, and FAST detection methods, respectively. It was also observed that although large numbers of features were detected across all image frames, the number of matches across consecutive frames was significantly less. From the SURF detection results in Figure 27, 5,000 points were consistently detected but only an average of 618 matched features were discovered. Similarly for the ORB method an average of 7,956 features were detected but only an average of 202 matched features, shown in Figure 28, while the FAST method, shown in Figure 29, detected an average of 7,035 features but discovered only an average of 97 matched features.

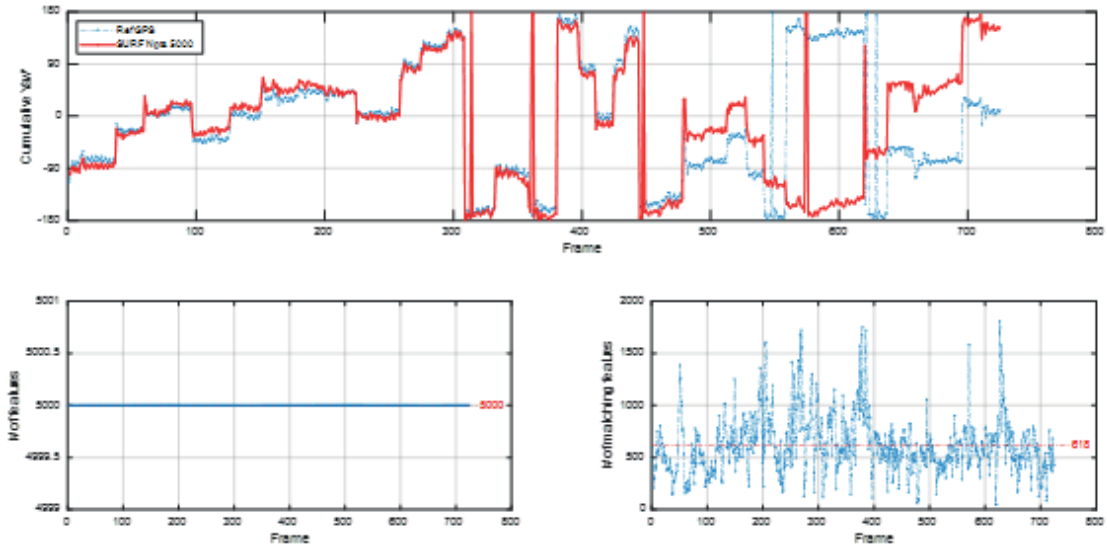


Figure 27. Estimated yaw comparison with reference GPS, for SURF strongest 5,000 points

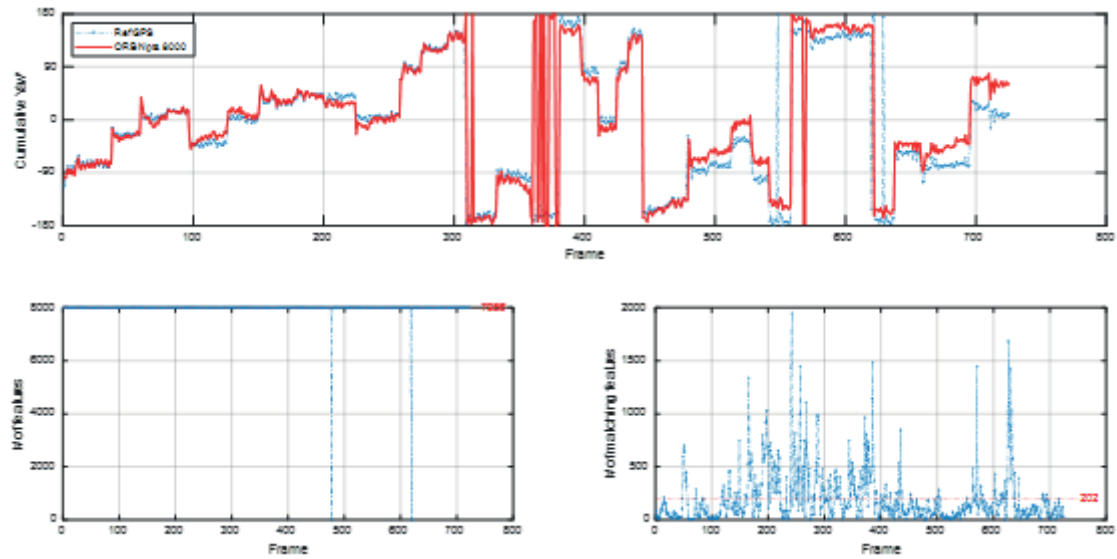


Figure 28. Estimated yaw comparison with reference GPS, for ORB strongest 8,000 points

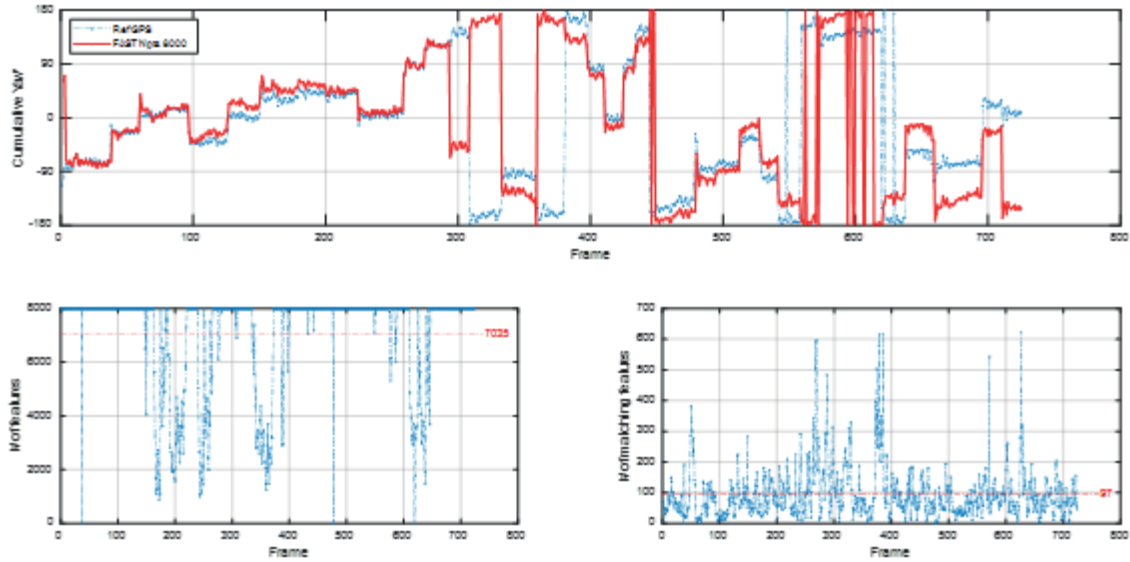


Figure 29. Estimated yaw comparison with reference GPS, for FAST strongest 8,000 points

Deviations were observed in the yaw angle comparison plots across all three detection methods, beginning at about the 300th image frame. Due to cumulative computation of yaw angles, small deviations resulted in huge magnitude differences as estimation errors accumulated with increasing image frames. An assessment of estimated yaw angles for the SURF method, illustrated in Figure 30, suggested that these deviations were likely to be caused by changes in aircraft altitude, roll, and pitch angles.

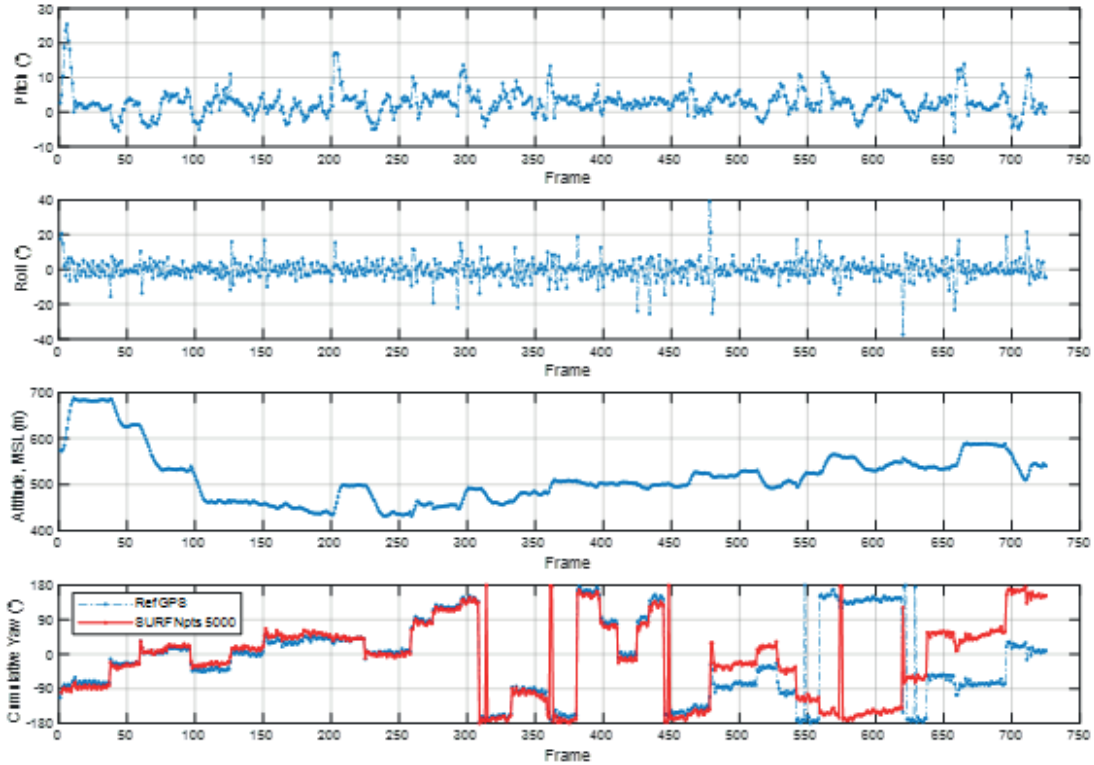


Figure 30. Compiled flight telemetry data with estimated yaw for SURF strongest 5,000 points

Flight over waypoints, illustrated earlier in Figure 20, was also a contributing factor to erroneous yaw estimation due to the large change in yaw heading whenever the drone executed turn maneuvers using fly over or loiter over waypoints. Figure 31 shows the attempt to overlay the flight trajectory estimated from visual odometry, plotted in blue, on the actual flight profile over Camp Roberts. Numbering in the figure indicates instances of camera trigger and their respective numeric labels. Large deviations in displacement were observed, particularly near the halfway mark where there were numerous turn maneuvers over waypoints.

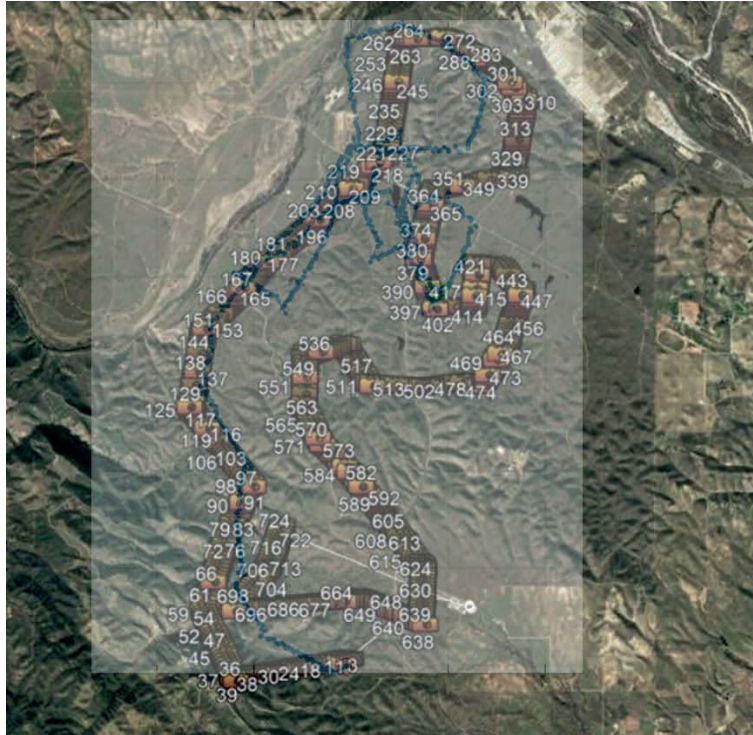


Figure 31. Estimated flight trajectory overlaid on flight profile

In order to overcome the challenge of accumulating estimation errors, a segmented computation approach was adopted. It was hypothesized that by treating flight segments in between flight over waypoints as straight and level flight with relatively small change in yaw heading, estimation errors would be reduced. For instance, Figure 32 shows a flight section where the segmented flight analysis approach was implemented. Image frames 19 to 37 before the turning waypoint would be considered as one segment, whereas image frame 38 onwards after the turn would be considered as another flight segment. Table 10 shows the flight segments with their corresponding start and end image frame index. Every start index frame then serves as the initialization point for each flight segment, providing reference telemetry data for computation.

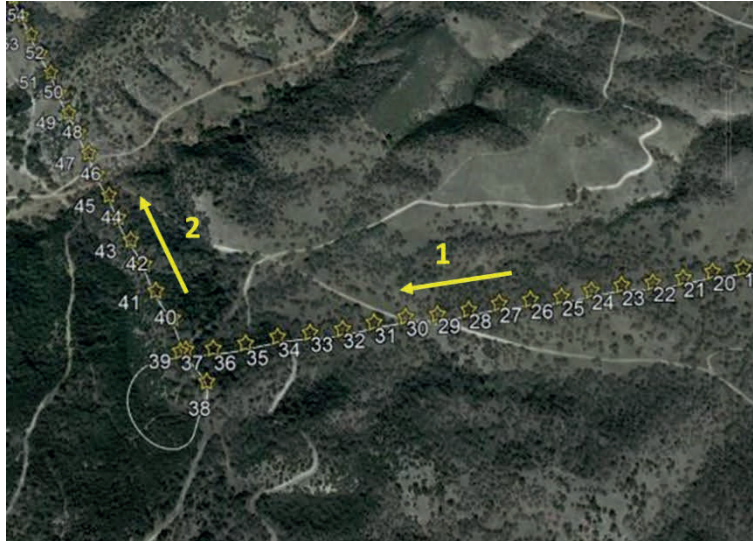


Figure 32. Example of segmented flight analysis

Table 10. Flight segments with corresponding start and end frame index

S/N	Start Index	End Index	S/N	Start Index	End Index
1	1	37	14	411	424
2	38	96	15	425	433
3	97	125	16	434	444
4	126	224	17	445	461
5	225	258	18	462	512
6	259	274	19	513	527
7	275	292	20	528	541
8	293	308	21	542	558
9	309	332	22	559	619
10	333	358	23	620	637
11	359	380	24	638	695
12	381	397	25	696	710
13	398	410	26	711	725

It was observed that trajectory estimations deviate from reference flight telemetry data, even after executing segmented flight analysis. Flight segments with large deviations were highlighted and labeled in Figure 33, indicating the need to further minimize estimation errors. An optimization approach to minimize distance RMS error for each flight segment was explored to improve the estimation results. Corrections to initializing yaw angles at each start index were introduced, with the goal to minimize overall distance RMS

error. Table 11 presents the summary of angle corrections applied to initializing yaw angles required to minimize the distance RMS error. It was observed that correction angles of various magnitudes were required for all the flight segments. This could be due to the latency and unsynchronized properties between the recorded telemetry and acquired imagery files. In some flight segments, corrective magnitudes above 100 degrees were seen. Nonetheless, estimated flight trajectory showed encouraging results. The final improved segmented flight trajectory estimation, illustrated in Figure 34, had most flight segments tracing over the actual flight trajectory from the reference GPS source.

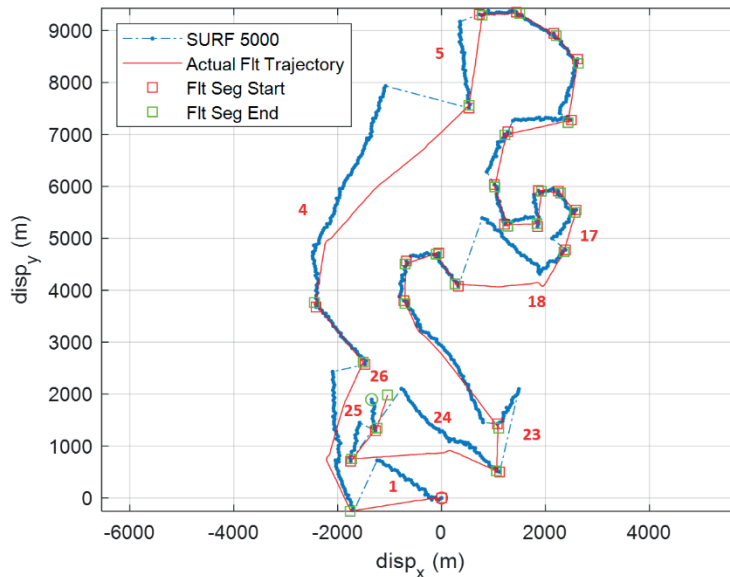


Figure 33. Segmented flight trajectory estimation using SURF strongest 5,000 points

Table 11. Summary of corrections applied on initializing yaw angles

	SURF	ORB	FAST
S/N	ΔYaw (deg)		
1	-35.3	-27.7	-175.7
2	6.1	4.9	4.5
3	-0.8	-3.0	-6.2
4	16.6	20.6	17.0
5	14.5	8.9	11.1
6	3.7	8.0	10.4

7	-2.6	-5.6	-3.4
8	-2.2	0.4	-168.7
9	-8.5	-12.0	-11.0
10	-14.0	-12.7	-16.4
11	-15.1	-15.8	-14.2
12	-2.5	-5.6	-3.0
13	16.5	16.3	12.9
14	11.3	10.3	11.6
15	6.2	1.2	5.2
16	-8.1	-7.7	-9.3
17	-21.4	-20.0	-21.1
18	-30.8	28.8	58.0
19	1.3	-1.0	2.6
20	-10.5	-7.2	-10.7
21	-8.2	-7.3	-7.3
22	0.4	7.8	3.7
23	146.8	-43.1	-43.6
24	-26.2	-24.7	-35
25	25.3	25.8	18.8
26	25.8	22.1	27.3

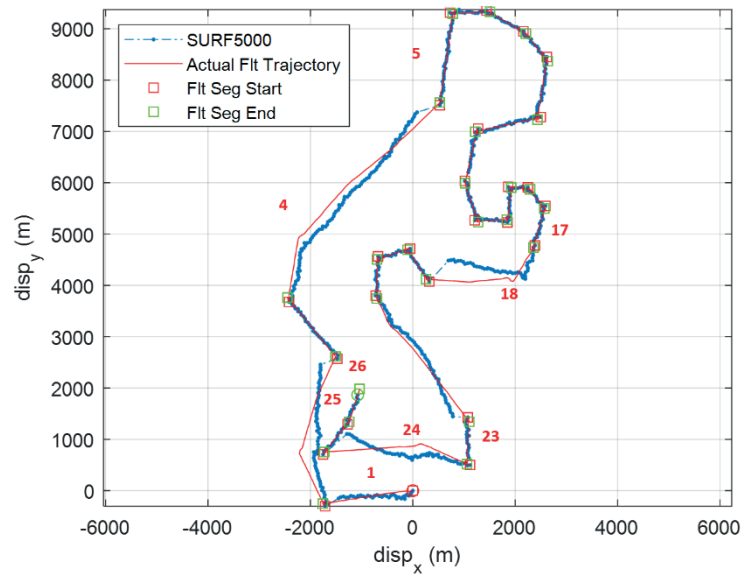


Figure 34. Improved segmented flight trajectory estimation, SURF strongest 5,000 points

Improved flight trajectory estimations for the ORB and FAST detection methods are illustrated in Figure 35 and Figure 36, respectively. While estimation results from all three detection methods were largely similar, inferior estimates were observed from the ORB and FAST methods in some flight segments, in particular flight segment 18.

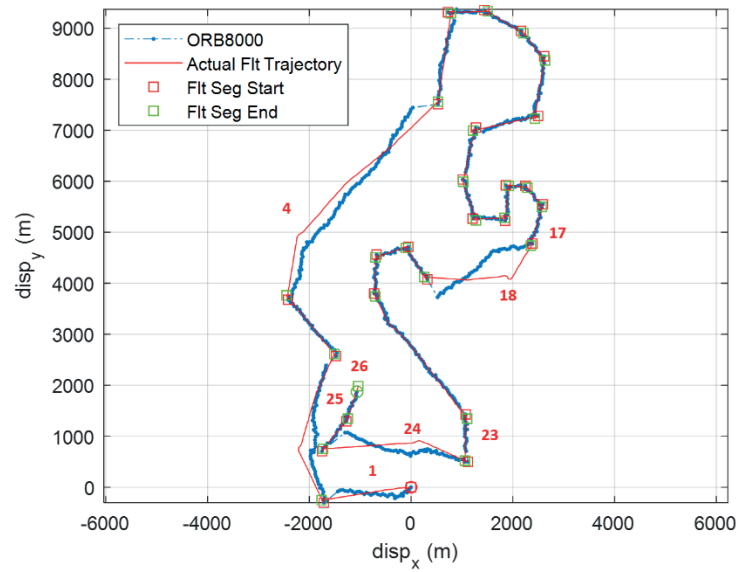


Figure 35. Improved segmented flight trajectory estimation, ORB strongest 8,000 points

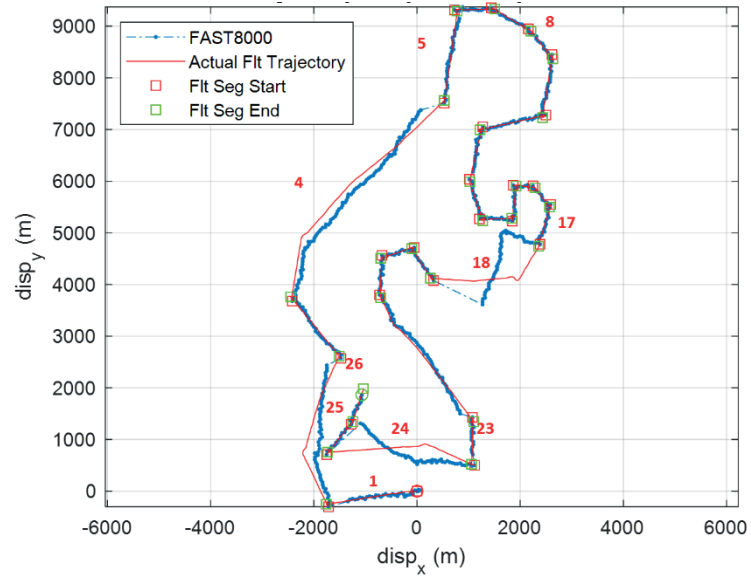


Figure 36. Improved segmented flight trajectory estimation, FAST strongest 8,000 points

V. PROCESSING TASE 200 DATA

The TASE 200 EO system was developed and utilized on a variety of UASs, featuring both daylight and infrared camera sensors. The TASE 200 comes with an integrated GPS/INS function that allows time-synchronized capturing and recording ground truth information during flight.

A. TEST SETUP

The TASE 200 sensor used to collect flight telemetry and imagery data has the specifications presented in Table 12. Figure 37 shows the enclosure developed by SkyIMD to enable the usage of this sensor on a manned aircraft, with the GPS receiver visible on top of the enclosure.

Table 12. Pertinent specifications of TASE 200 sensor

Image Dimension	640 pix by 480 pix
Horizontal Field-of-view	Variable, from 10.5° to 25.26°
Ground Sampling Distance at 137 m	71 cm/pix
Recording Rate	30 Hz
Storage Media	Laptop



Figure 37. TASE 200 sensor in SkyIMD enclosure

The TASE 200 system outputs data using a proprietary format in a series of files. Each file contains about 30 frames, each preceded by the corresponding telemetry data.

The super header for each file determines the size and format of the telemetry as well as the size of the images. The variability of the size of each file is dictated by the real-time operations requirement. Table 13 shows specific parameters captured and embedded into the meta-data file for each image frame.

Table 13. TASE 200 meta data format specification

#	Parameter	Byte(s)	#	Parameter	Byte(s)
1	GPS Day	41	25	Mount Roll	260-263
2	GPS Hour	42	26	Mount Pitch	264-267
3	GPS Minute	43	27	Mount Yaw	268-271
4	GPS Second	44-47	28	VN	76-79
5	Second since reset	136-139	29	VE	80-83
6	Second since midnight	12-15	30	VD	84-87
7	Gimbal Lat	56-63	31	Heading	316-319
8	Gimbal Lon	65-71	32	HFOV	168-171
9	Gimbal Alt	72-75	33	VFOV	172-175
10	Gimbal Pan	24-271	34	HFOVmax	176-179
11	Gimbal Tilt	28-31	35	HFOVmin	180-183
12	Gimbal Roll	32-35	36	Zoom	186-187
13	Image Lat	192-199	37	HFOVmaxC2	212-215
14	Image Lon	200-207	38	HFOVminC2	216-219
15	Image Alt	208-211	39	Transx	
16	Axis Pan Rate	140-143	40	Transy	
17	Axis Tilt Rate	144-147	41	GPS Satellites	48-49
18	Axis Roll Rate	148-151	42	GPS Status	50-51
19	Mount Pan Rate	152-155	43	GPS PDOP	52-55
20	Mount Tilt Rate	156-159	44	Magx	310-311
21	Mount Roll Rate	160-163	45	Magy	312-313
22	Roll	88-91	46	Magz	314-315
23	Pitch	92-95	47	Focus	256-257
24	Yaw	96-99			

Facilitating proof of concept of image-matching techniques in a previous research, the TASE 200 was flown west of King City, California, in the standard holding patterns at altitudes 2,000 ft, 4,000 ft, and 6,000 ft MSL, shown in both Figure 38 and Figure 39.

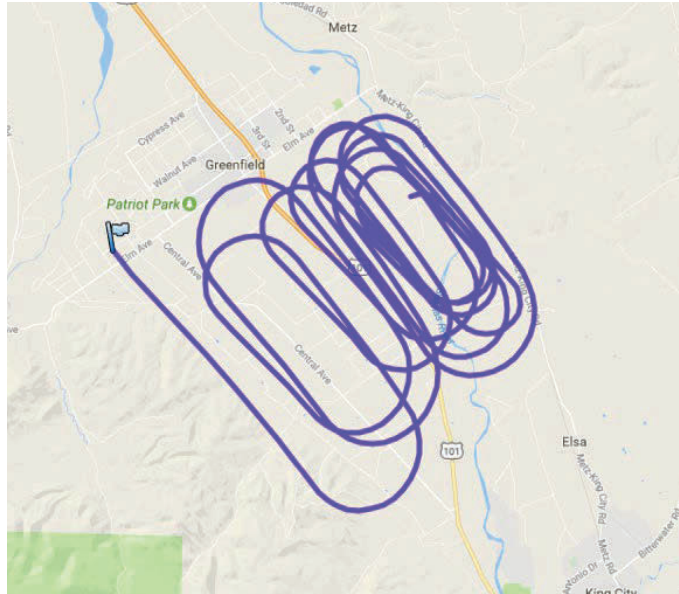


Figure 38. Overview of flown trajectory with TASE 200 system

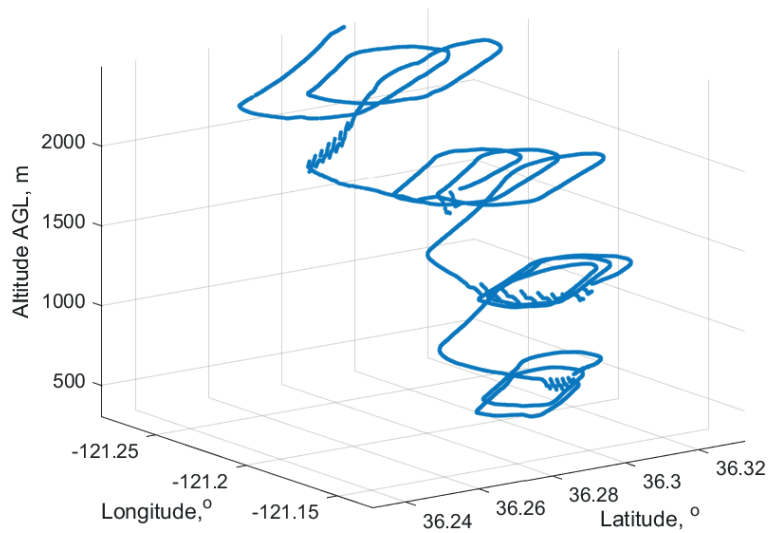


Figure 39. 3D flight profile of flown trajectory with TASE 200 system

B. DATA PROCESSING

The 30 Hz (30 fps) video recorded by the TASE 200 sensor was down sampled to 0.5 Hz to explore the practicality of using less frequently updated images for image matching. Assuming there are sufficient feature matches across down sampled image frames, processing time can be saved from skipped images, facilitating real-time

implementation. Figure 40 illustrates sample TASE 200 imagery output, in grayscale format. Similar to the images returned from the flight over Camp Roberts, agricultural fields were observed in all of the images. The number of features detected from a straight level flight segment at 2,000 ft, shown in Figure 41, was still significant to produce the required geometric transformations.

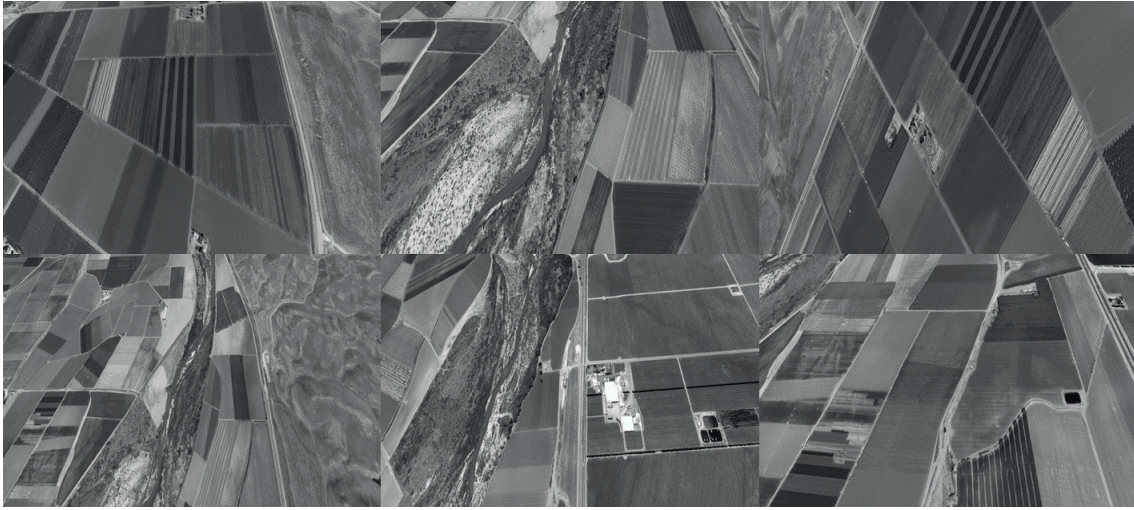


Figure 40. Sample TASE 200 images

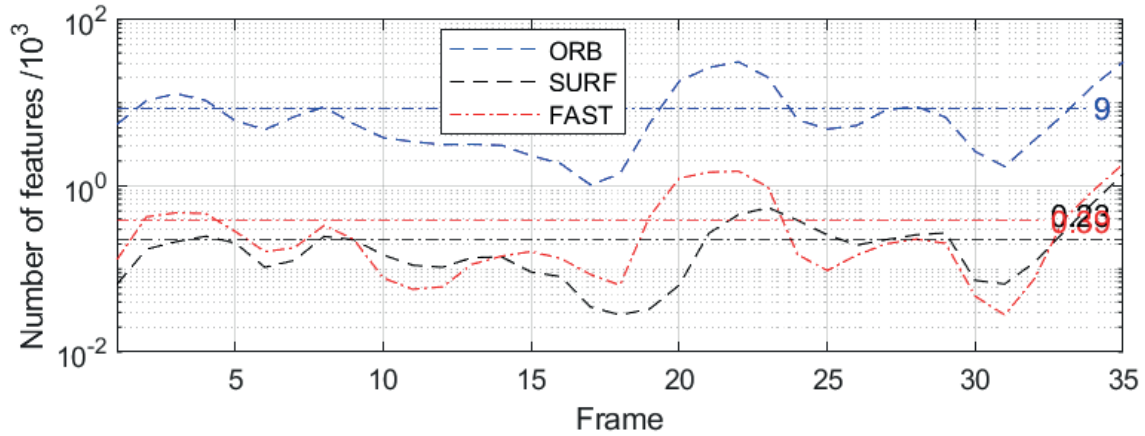


Figure 41. Number of features detected using FAST, ORB, and SURF methods from TASE 200 imagery files

Categorization of straight level flight segments into groups of largely similar altitude were executed to assess the performance of the NAVAID at different altitudes.

Due to the camera mean pitch angle of 45° , acquired images were seen to undergo shearing, likely due to projective transformation. This phenomenon greatly affected the transformation results, which were based on similarity transformations. In order to overcome this, images were cropped at the sides, leaving the center sections which were less distorted. Figure 42 illustrates the matched inliers between two consecutive images acquired at 2,000 ft, with clear indication of shearing as the trace lines converged toward the top of the image frame. Figure 43 shows the trace lines preserving parallelism after the image was cropped at the sides. Table 14 provides a summary of the estimated results of transformation parameters, with the most significant difference observed in transformation matrices t_x and t_y . Results presented in the following Part C are processed using cropped images.

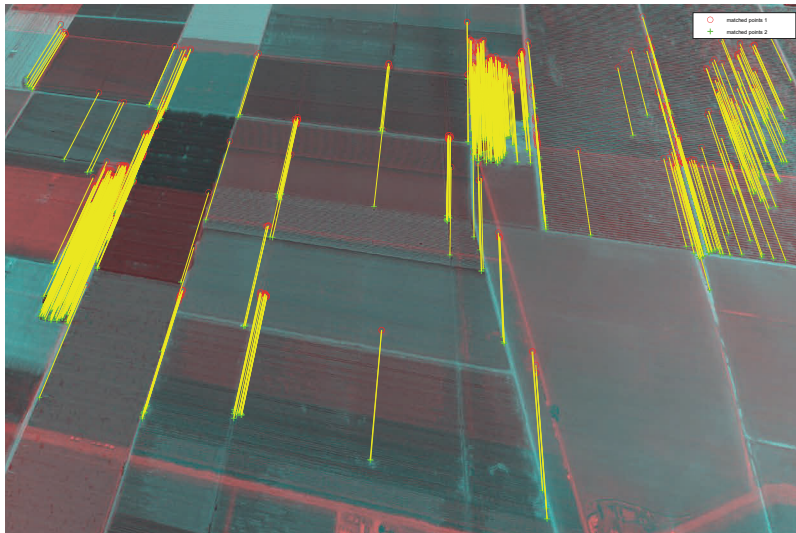


Figure 42. Matched inliers on raw TASE 200 images

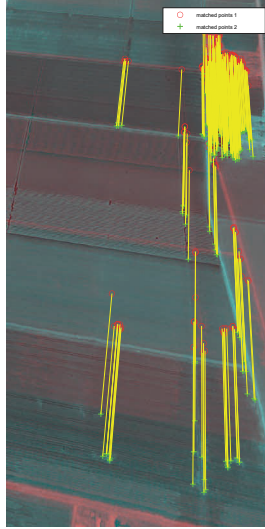


Figure 43. Matched inliers on cropped TASE 200 images

Table 14. Comparison of estimated transformation parameters

Transformation Parameter	Before Crop	After Crop
t_x	-140.566pix	-86.273 pix
t_y	178.558 pix	124.559 pix
s	1.1139	1.174
θ	0.9969°	-1.694°

C. KEY OBSERVATIONS

Similar to the analysis approach for the Camp Roberts flight campaign presented in Chapter III, the three feature detection methods were utilized to analyze the TASE 200 imagery. Yaw angle refinements were not introduced since the flight telemetry and imagery were time-synchronized. Cropped images were used to overcome the shear distortion effects on the images while using the similarity geometric transformation approach. Figure 44 illustrates the estimated trajectory of using raw and cropped images acquired at 2,000 ft, with the trajectory from cropped images matching closer to actual flight trajectory from the reference GPS.

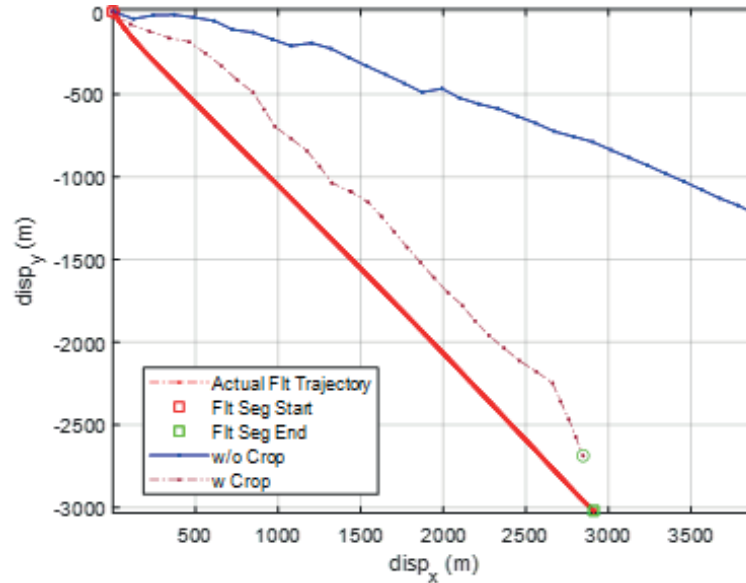


Figure 44. Comparison of SURF method trajectory estimations between raw and cropped imagery at 2,000 ft

At flight altitude of 2,000 ft, estimation performance of both the FAST and SURF methods was relatively similar for the down leg flight segment, shown in Figure 45. For the up leg flight segment, the SURF method was observed to trace over the actual flight trajectory for some waypoints, seen in Figure 46. Estimated positions of flight segment end points were observed to be less than 400 m from the true end point for most detection methods.

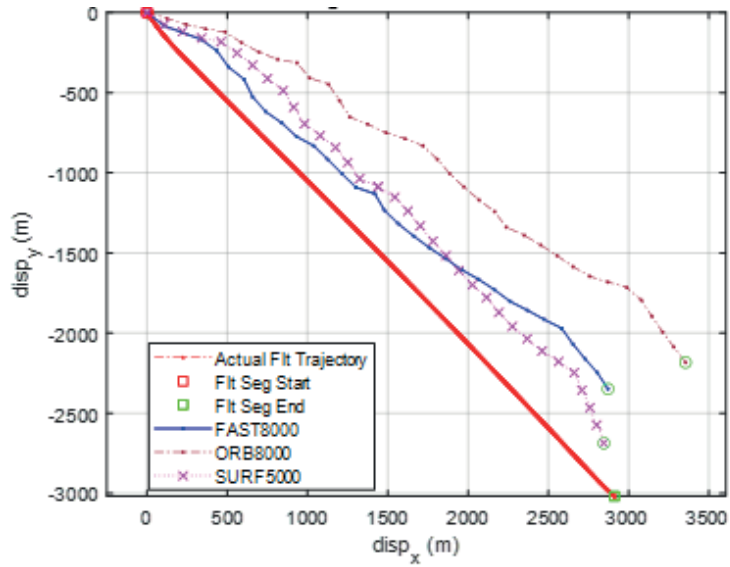


Figure 45. Trajectory estimations for down leg segment at 2,000 ft

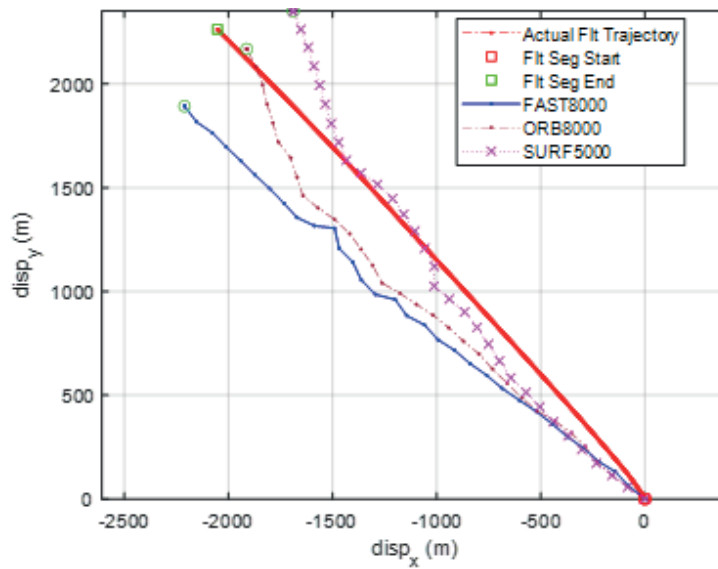


Figure 46. Trajectory estimations for up leg segment at 2,000 ft

For both the down and up leg flight segments at 4,000 ft, the SURF method had superior performance over the other two detection methods, tracing close to the actual flight trajectory. The respective trajectory estimations are illustrated in Figure 47 and

Figure 48. Estimated positions of flight segment end points were observed to be more than 400 m from the true end point for all detection methods.

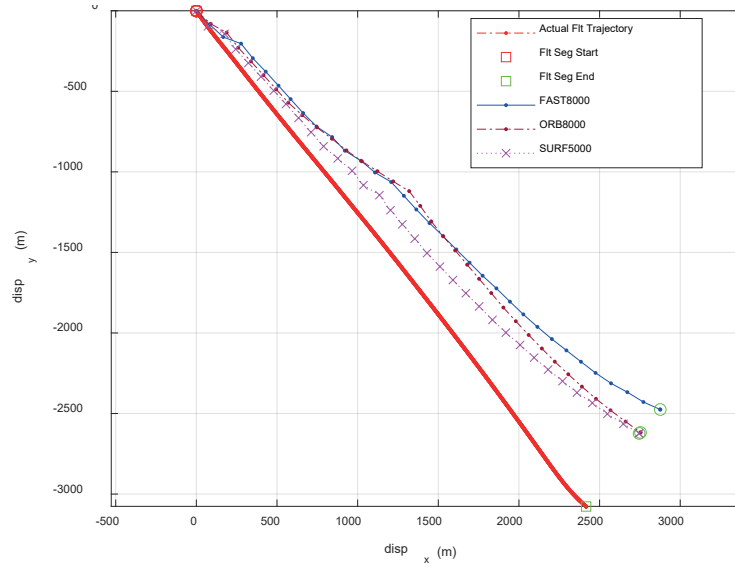


Figure 47. Trajectory estimations for down leg segment at 4,000 ft

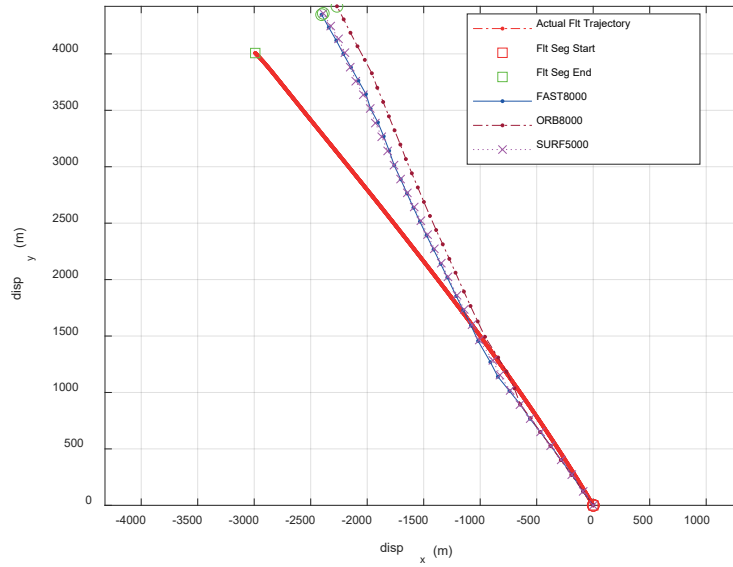


Figure 48. Trajectory estimations for up leg segment at 4,000 ft

At the highest flight altitude of 6,000 ft, all three detection methods seemed to underperform, showing large deviations from the actual flight trajectory. Estimated positions of flight segment end points were observed to be more than 700 m from the true end point for all detection methods. Trajectory estimations for the up leg flight segment at 6,000 ft are illustrated in Figure 49.

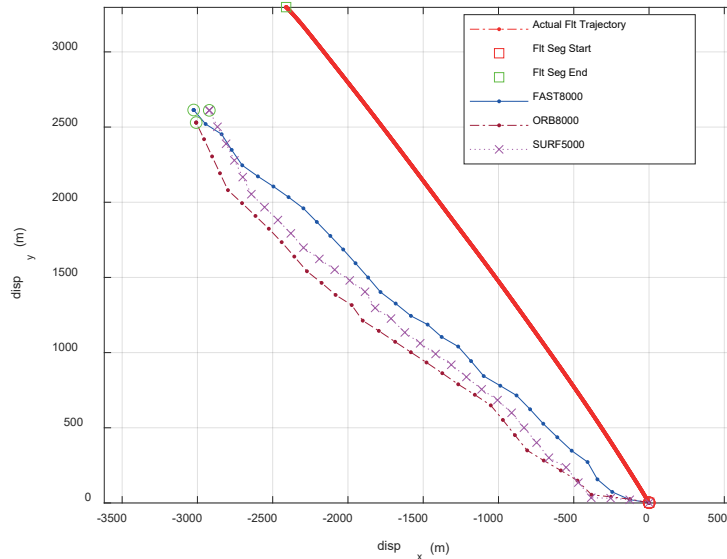


Figure 49. Trajectory estimations for up leg segment at 6,000 ft

From the results, it was observed that estimation performance deteriorates with increasing altitude. This could be due to poor image resolution at higher altitudes, while representing higher distance per pixel. Inaccuracies in estimated pixel displacement from the geometric transformation of a sheared image also contributed to the large deviations in trajectory matching. While the SURF method compared better with actual flight trajectory, it should be noted that its required computation time was the highest among the three detection methods.

VI. DATA FROM SELF-BUILT SENSOR SUITE

The third and final set of flight telemetry and imagery data was collected from a self-built sensor suite. The self-built sensor suite was made up of COTS sensors, featuring a high-resolution GoPro Hero 4 camera and a standard iPhone 7 running a telemetry recording application. The sensor suite was mounted on a manned Cessna-172 aircraft, which offered flexibility and manual control over synchronization of data acquisition.

A. TEST SETUP

The manned aircraft Cessna-172, shown in Figure 50, was equipped with a GoPro and an iPhone to provide an alternate source of data set to further establish the applicability of the NAVAID. An overview of the Cessna aircraft flight trajectory, plotted using post-flight telemetry data, is illustrated in Figure 51 and Figure 52.



Figure 50. Cessna-172 aircraft

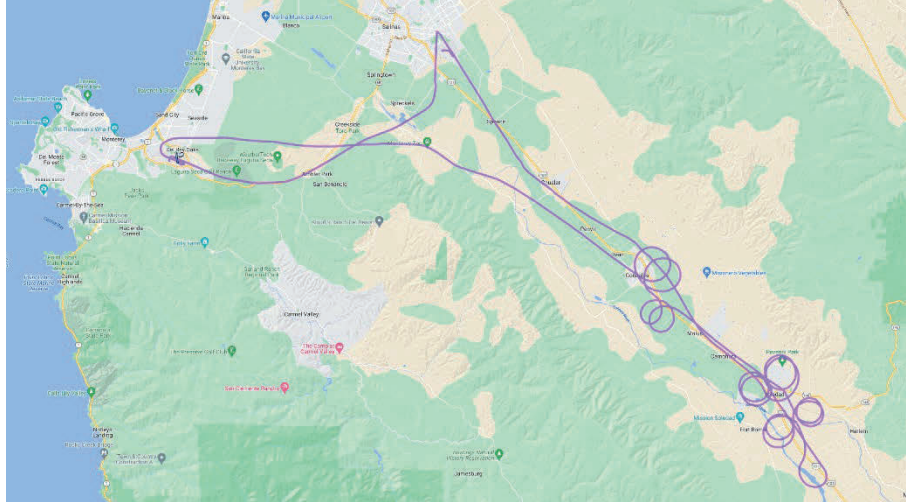


Figure 51. Trajectory overview of Cessna flight campaign

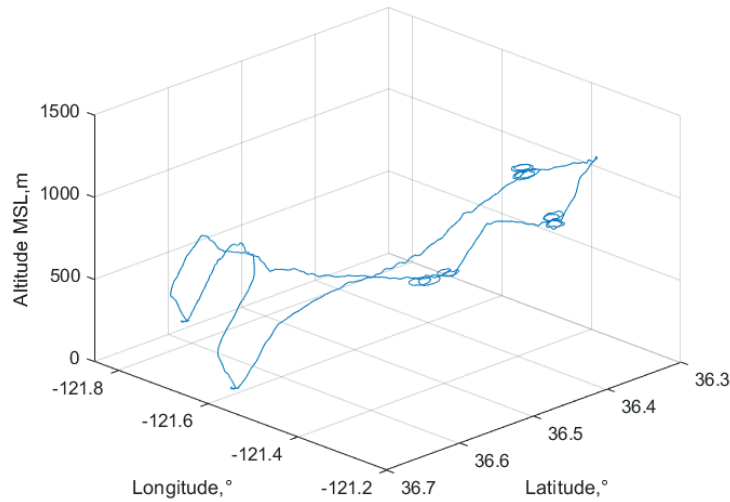


Figure 52. 3D flight profile of Cessna flight campaign

Like the Trinity F90+ drone flight, the Cessna flight aims to acquire in-flight aerial videos with corresponding flight telemetry recording. A typical flight can be categorized into five segments, illustrated in Figure 53. Considering potential GPS-denial scenarios that a UAS will encounter, straight level (cruise) and loitering flight segments were the focus for assessing performance of visual odometry.

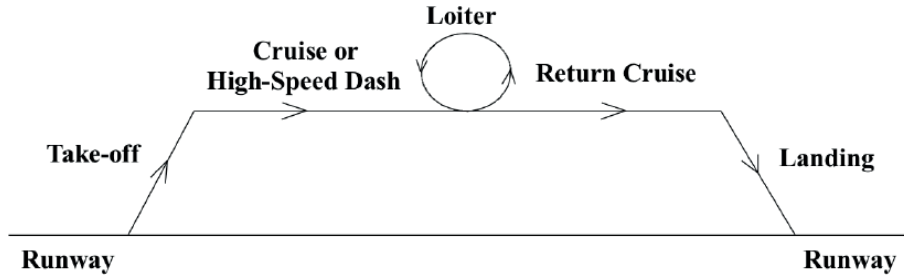


Figure 53. Typical UAS flight profile

The GoPro Hero 4 camera, collecting $1,920 \text{ pix} \times 1,080 \text{ pix}$ images at 30 Hz rate, were the sole source of aerial imagery. An iPhone 7, equipped with micro-electromechanical system (MEMS) accelerometers, barometer, and GPS receivers, as well as installed with the SensorLog application, were utilized to record in-flight telemetry data. The telemetry data can be recorded at a rate of up to 100 Hz and saved as a *CSV* file. Both the GoPro and the iPhone were installed on the Cessna wing strut, using 3D printed adapters fitted on the Airfilm Camera Systems' wing strut mount. The mounting setup is illustrated in Figure 54.



Figure 54. Mounting setup of acquisition equipment on Cessna wing strut

Along with the setup shown in Figure 54, a secondary telemetry recorder QStarz BT-Q1000eX GPS data logger with a 10 Hz log rate was also used to provide redundancy. The list of recorded parameters included GPS coordinates, speed, and distance. This data logger features a battery life of 42 hours and can log up to 400,000 data points. Other specifications of the data logger are presented in Table 15.

Table 15. QStarz BT-Q1000eX sensor specifications. Adapted from QStarz International Co. Ltd. (n.d.)

Device size, mm	“72.2(L) × 46.5(W) × 20(H)”
GPS Chip	MTK II GPS Module
Frequency, MHz	L1, 1575.42
Antenna	“Built-in patch antenna with low noise amplifier (LNA)”
Acquisition rate (cold start/hot start), s	33/1
Accuracy (CEP), m	< 3
Memory size, Mb	8
Interface	Bluetooth

B. DATA PROCESSING

Flight telemetry data recorded by the iPhone SensorLog application was post-processed using MATLAB and plotted in Figure 55. Sample images acquired by the GoPro are illustrated in Figure 56. In contrast to the data from the flight campaign with the Trinity F90+ drone and its proprietary QBase 3D software, the telemetry data in this campaign had to be manually synchronized to the imagery captured by the GoPro.

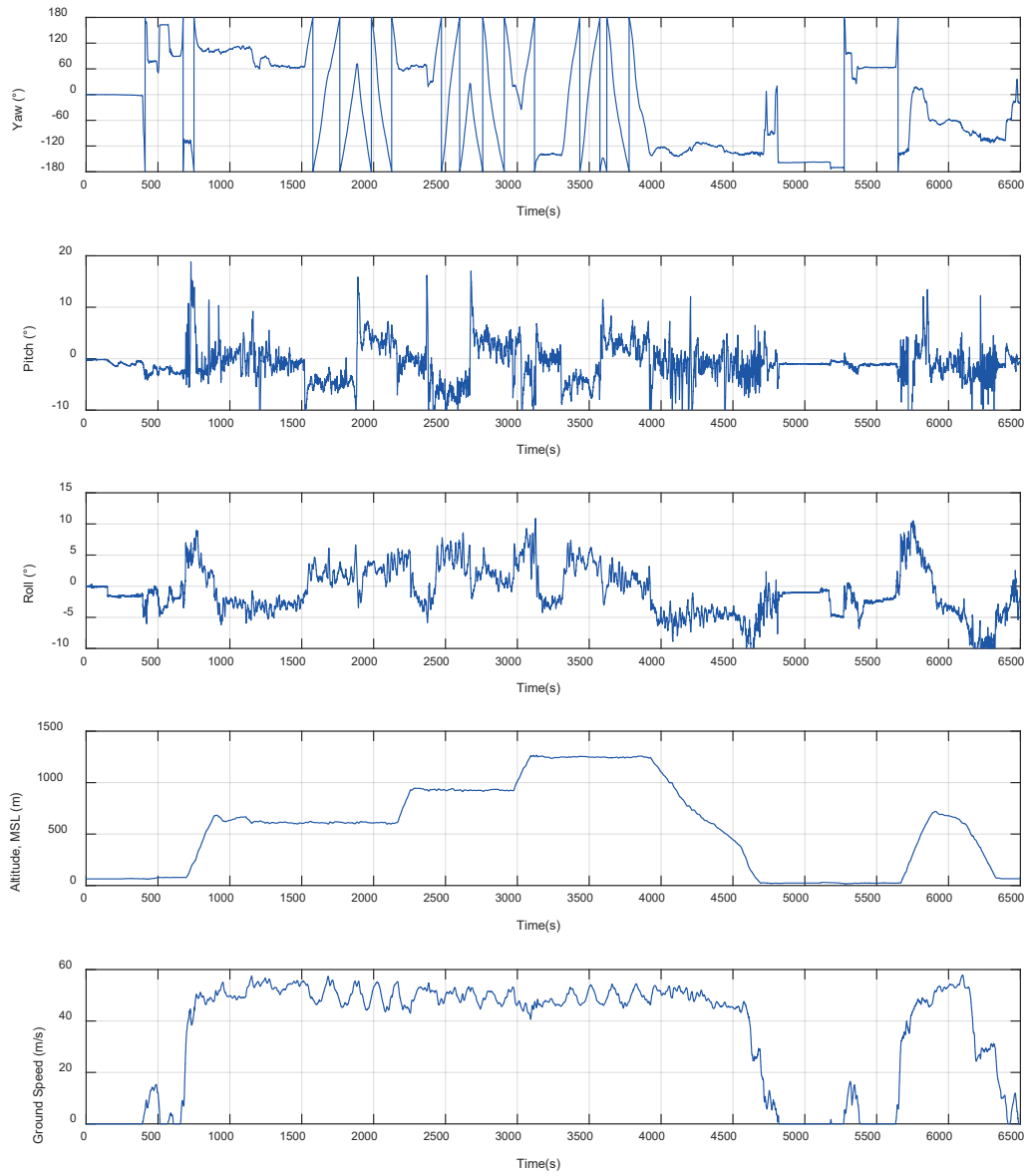


Figure 55. Flight telemetry plots of Cessna flight campaign



Figure 56. Sample images acquired from GoPro Hero 4

Similar to the Trinity F90+ flight presented earlier, the minimum average numbers of features extracted across all three algorithms, shown in Figure 57, were more than sufficient to fulfill the minimum requirements for proper geometric transformations.

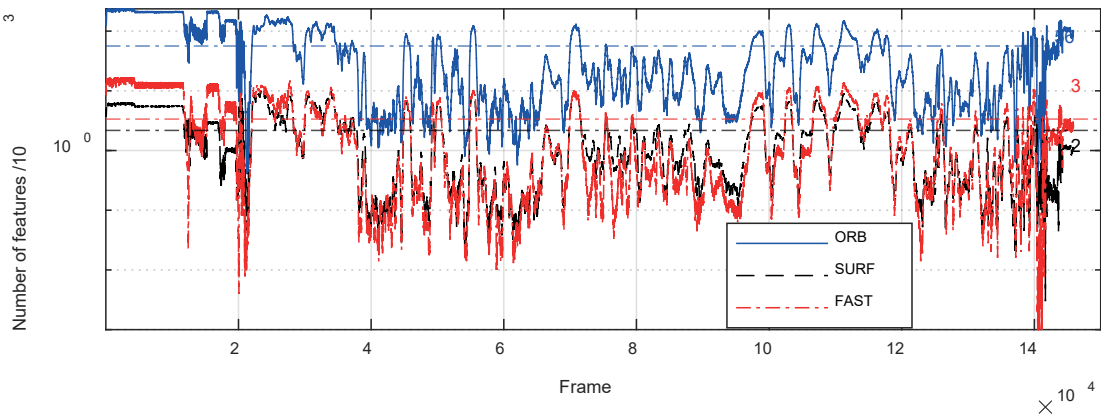


Figure 57. Number of features detected in Cessna flight imagery using ORB, SURF, and FAST methods

The recorded imagery video was also analyzed at 1 Hz, instead of the recorded 30 Hz, to overcome imagery distortions due to engine- and aerodynamics-induced vibrations. At the original rate of 30 Hz, the imagery distortions were found to corrupt pose estimations. Although a significant number of features were detected, presented in Figure 57, pixel movements were not significant. Moreover, they were inaccurate due to the

rolling shutter effect. Nevertheless, enough features were still detected despite multiple image frames being skipped, even when using the 1 fps video.

C. KEY OBSERVATIONS

With lessons learnt from Trinity F90+ and TASE 200 imagery analysis, the segmented approach was adopted for the Cessna flight imagery analysis. Flight segments were categorized into straight level, left-bank, and right-bank maneuvers at altitudes of 2,000 ft, 3,000 ft and 4,000 ft MSL. The categorization facilitated the ease of estimation comparison and assessment across various flight altitudes for the same maneuver.

Flight trajectory matchings for straight level flight legs were relatively good across all altitudes. With appropriate initialization of time-synchronized telemetry, image matching enabled executing pose estimation with a good level of accuracy. Figure 58, Figure 59, and Figure 60 show the trajectory estimations and matching with actual flight trajectories for 2,000 ft, 3,000 ft and 4,000 ft, respectively. It was noted that as flight altitude increases, trajectory matchings deteriorate, with large deviations observed amid the 4,000 ft straight level flight segment. Similar to pose estimations of TASE 200 imagery at high altitudes, poor accuracy in estimated pixel displacement may have caused the increasing deviations at higher altitudes. The impact of poor estimation results between a single pair of images could be seen in the FAST detection method shown in Figure 49. Due to its cumulative properties, poor estimation accumulates and its performance declines over time.

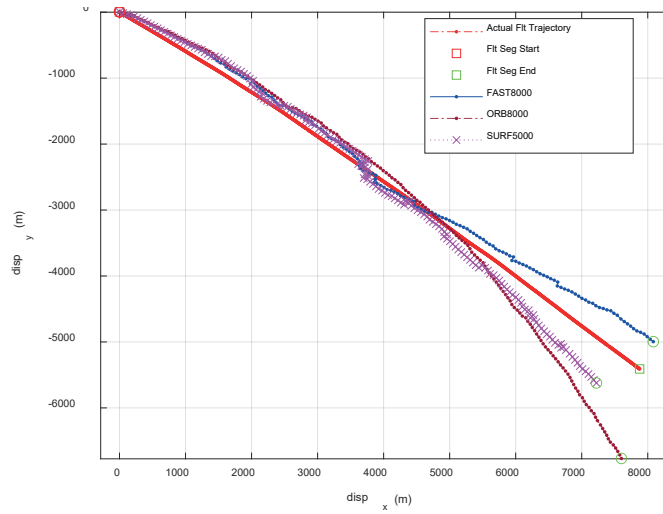


Figure 58. Trajectory estimations for straight level flight at 2,000 ft

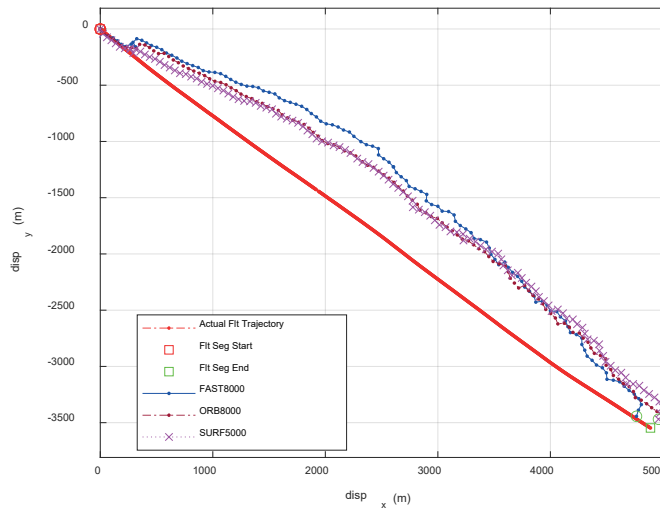


Figure 59. Trajectory estimations for straight level flight at 3,000 ft

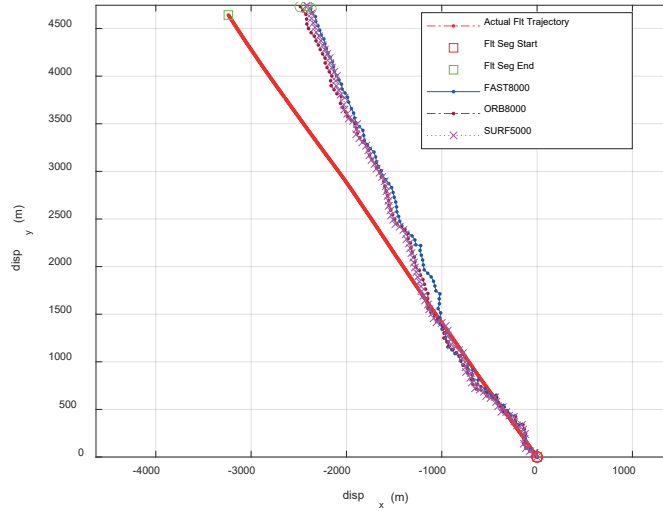


Figure 60. Trajectory estimations for straight level flight at 4,000 ft

On the other hand, flight trajectory matching results were not satisfactory for the bank maneuvers. Although the pilot tried to maintain the pitch and roll angles constant during the bank turns, images perceived by the EO sensor have sheared, meaning that the similarity geometric transformation no longer applied. In addition, all bank maneuvers consisted of two full turns, but estimation results revealed less than one turn in some cases. The preliminary results of the left-bank maneuvers at 2,000 ft, 3,000 ft, and 4,000 ft are illustrated in Figure 62, Figure 64, and Figure 66, respectively, with corresponding flight telemetry plots in Figure 61, Figure 63, and Figure 65.

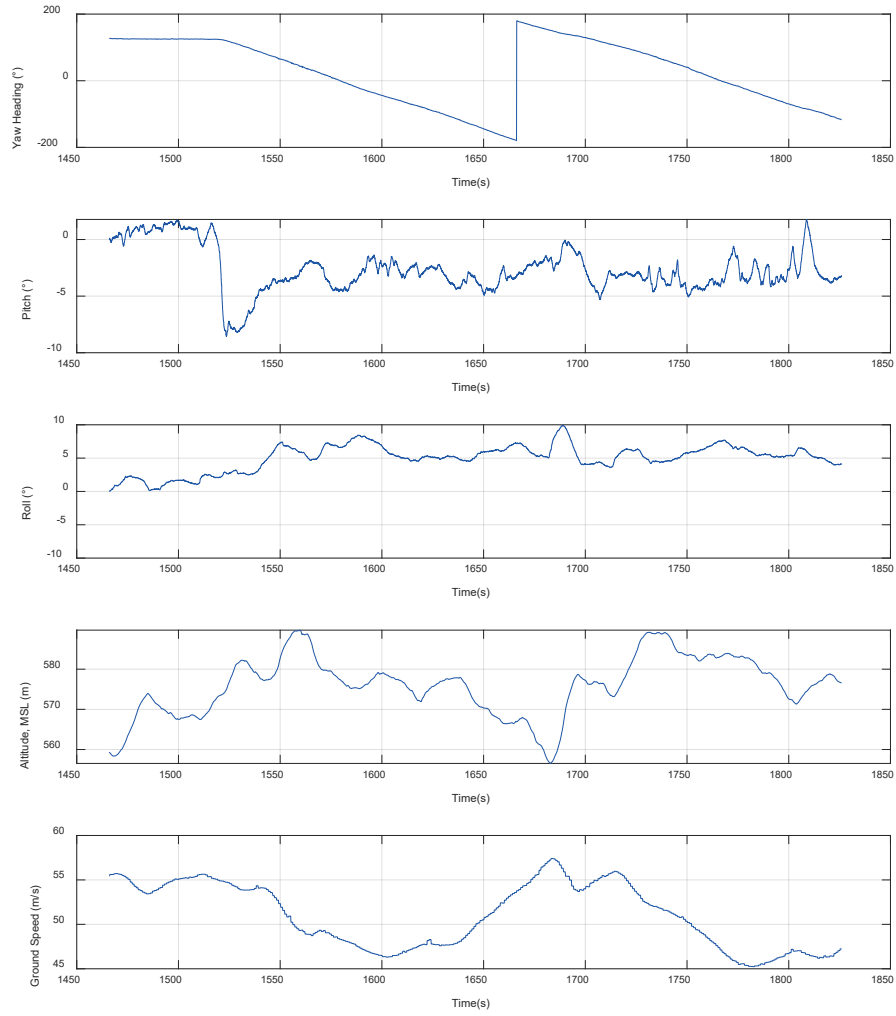


Figure 61. Flight telemetry for left-bank maneuver at 2,000 ft

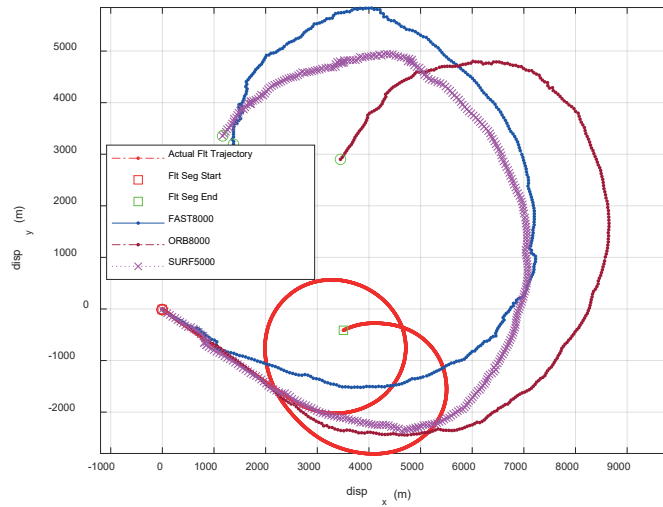


Figure 62. Trajectory estimations for left-bank maneuver at 2,000 ft

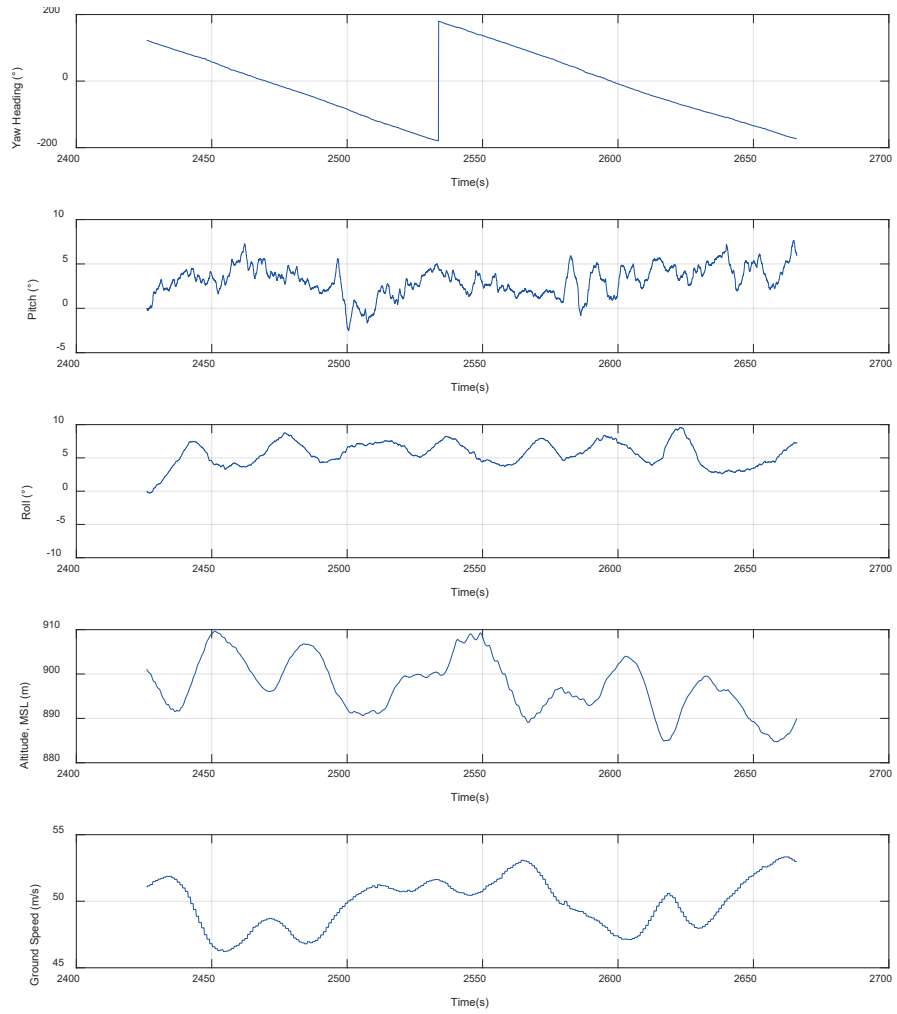


Figure 63. Flight telemetry for left-bank maneuver at 3,000 ft

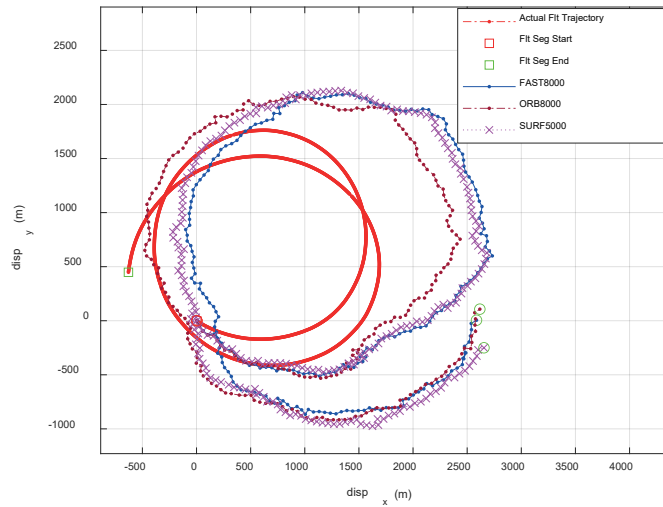


Figure 64. Trajectory estimations for left-bank maneuver at 3,000 ft

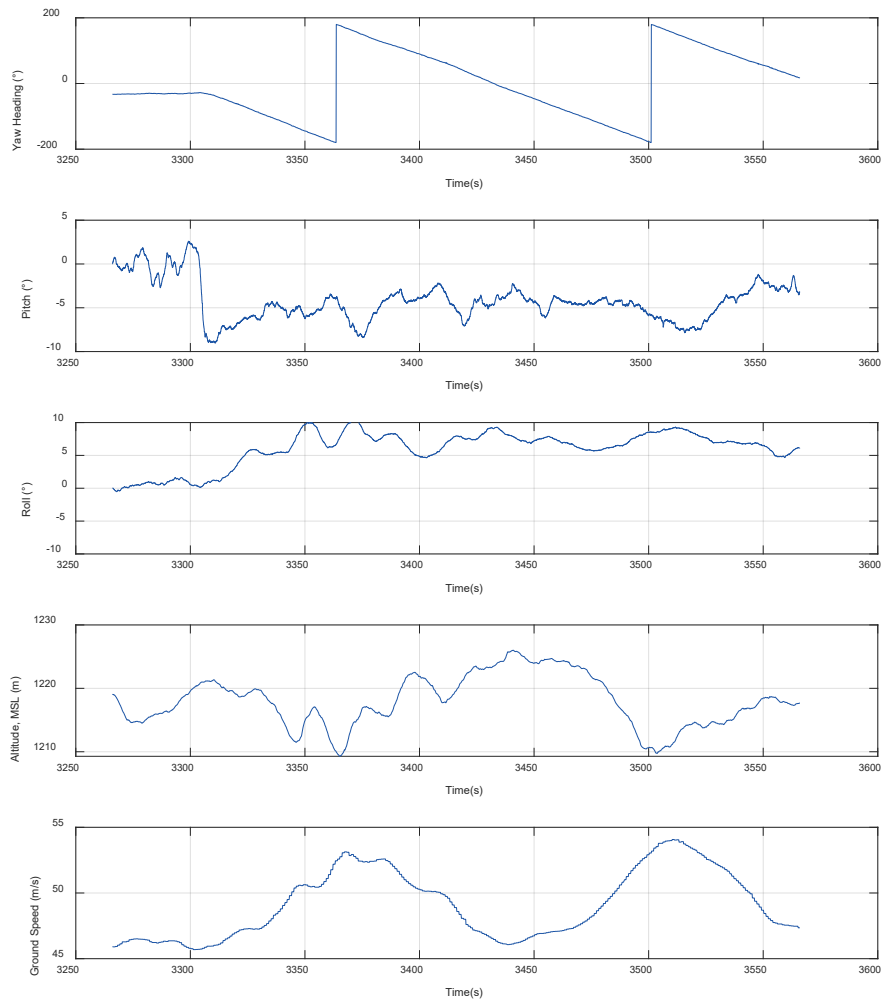


Figure 65. Flight telemetry for left-bank maneuver at 4,000 ft

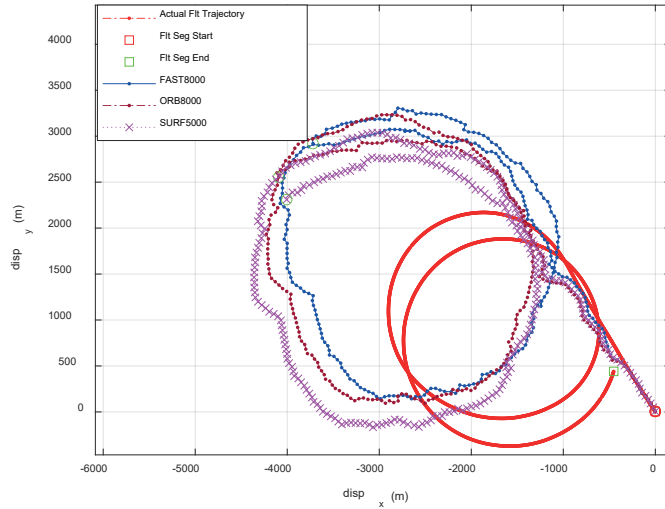


Figure 66. Trajectory estimations for left-bank maneuver at 4,000 ft

Similarly, for the right-bank maneuver, visual-based odometry results showed poor accuracy and were inadequate for the navigation system. The preliminary results of the right-bank maneuvers at 2,000 ft, 3,000 ft, and 4,000 ft are illustrated in Figure 67, Figure 68, and Figure 69, respectively.

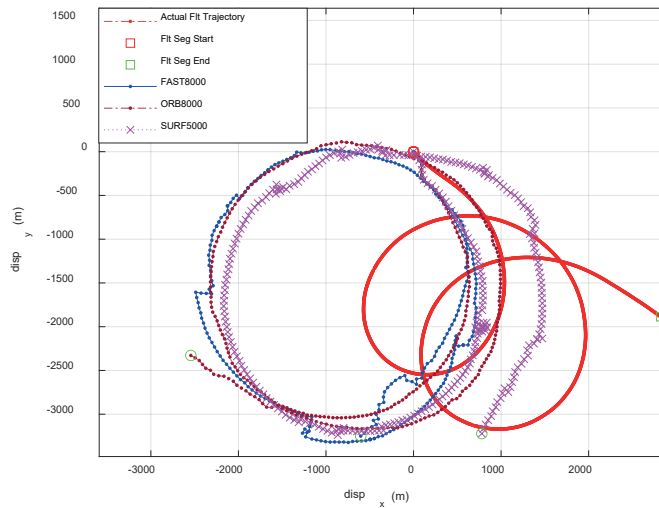


Figure 67. Trajectory estimations for right-bank maneuver at 2,000 ft

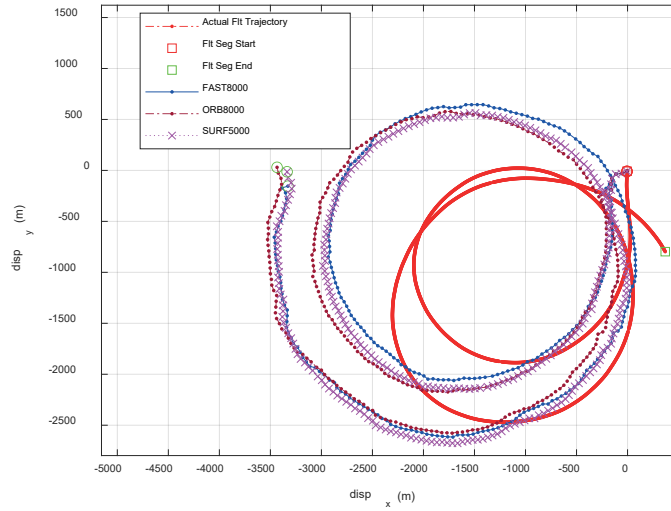


Figure 68. Trajectory estimations for right-bank maneuver at 3,000 ft

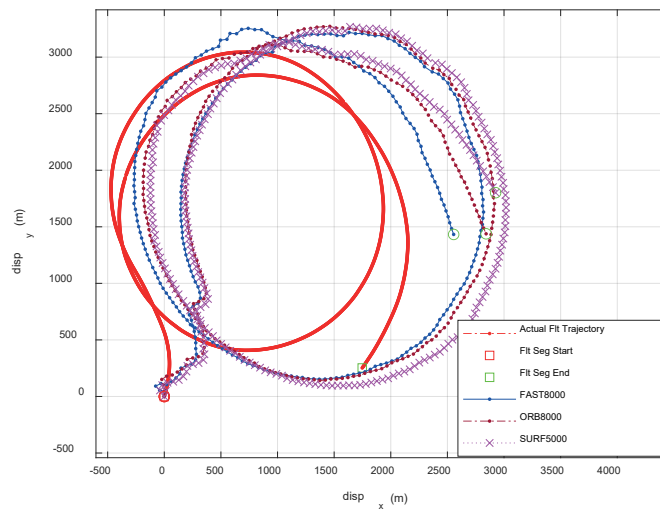


Figure 69. Trajectory estimations for right-bank maneuver at 4,000 ft

VII. INCORPORATING DEVELOPED NAVAID SOLUTION ON UAS

The navigation system typically found on a UAS consists of an IMU and a GPS or equivalent GNSS receiver. A simplified block diagram of such a navigation system is shown in Figure 70, where INS and GPS position and velocity differences are processed by the Kalman filter. The function of the Kalman filter is to estimate the INS errors with reference to GPS signals, which will be used for INS correction.

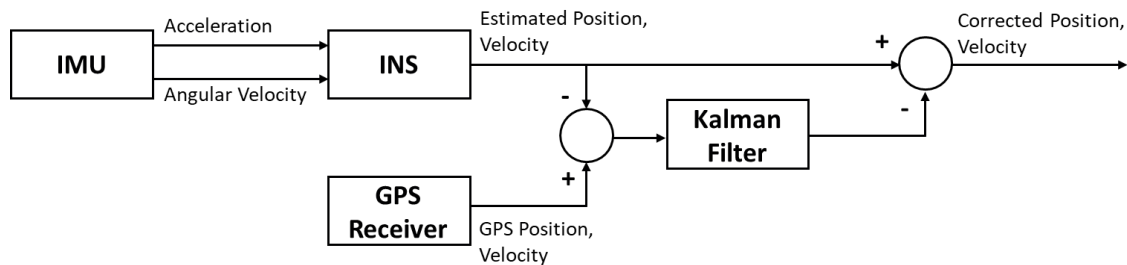


Figure 70. Block diagram of INS and GPS integration

In the event that GPS signals are unavailable, the NAVAID solution augments the navigation system, illustrated in Figure 71. With the proof of feasibility presented in earlier chapter, the NAVAID solution provides position estimates and ensures that the UAS can continue and successfully complete its mission.

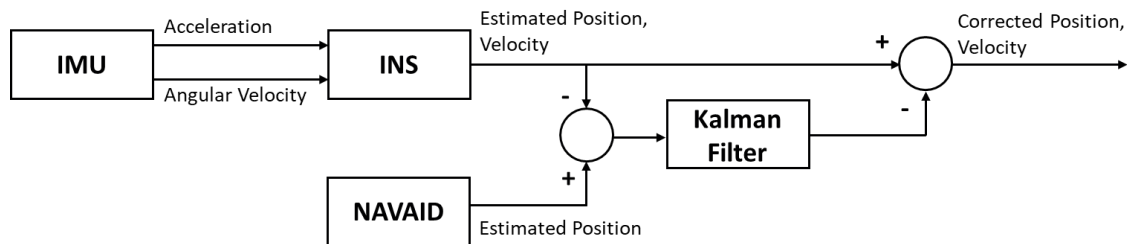


Figure 71. Block diagram of INS and NAVAID integration

The `insfilterErrorState` object from MATLAB (MathWorks n.d.) was utilized to demonstrate the integration of the NAVAID solution to obtain corrected UAS position. This object facilitates the fusion of measurements from both IMU and GPS, estimating UAS pose in the North-East-Down (NED) reference frame. The filter employs a 17-element state vector to track the quaternion orientation, velocity, and position of the UAS, while using an error-state Kalman filter to execute estimation of system state quantities. Either the GPS or the NAVAID position values will be incorporated into the Kalman filter to provide periodic INS error corrections and generated corrected pose results. The NAVAID implementation script is documented in the Appendix.

$$x = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \\ position_N \\ position_E \\ position_D \\ v_N \\ v_E \\ v_D \\ gyrobias_x \\ gyrobias_y \\ gyrobias_z \\ accelbias_x \\ accelbias_y \\ accelbias_z \\ scaleFactor \end{bmatrix} \quad (16)$$

Straight level flight telemetry acquired at 2,000 ft from the Cessna self-built sensor suite was used for to demonstrate the integration of the NAVAID solution into the navigation system. Measurements from accelerometers and gyroscope, embedded in the iPhone device, will be used as the IMU source for the UAS state estimation, with their axes convention defined in Figure 72.

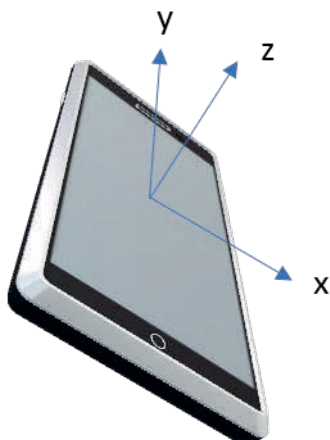


Figure 72. Axes convention definition relative to iPhone device

Comparison of trajectory results revealed that NAVAID solution performance was superior to that of INS-only pose estimation. Position estimation results from the INS-NAVAID using the FAST method performed the best against INS-only, shown in Figure 73. The RMS error between INS-GPS (true reference) and INS-NAVAID was [285.57 m (X, East), 117.88 m (Y, North)], while the RMS error between INS-GPS and INS-only was [610.95, 180.48] m. The SURF method performance was slightly inferior, as illustrated in Figure 75, with an RMS error of [96.91, 357.39] m. The ORB method had an RMS error of [478.46, 208.30] m, shown in Figure 74. In summary, the demonstration of incorporating the NAVAID solution, in place of GPS signals during GPS-denied situations, into a UAS navigation system was promising and provided better pose estimation than the INS-only option.

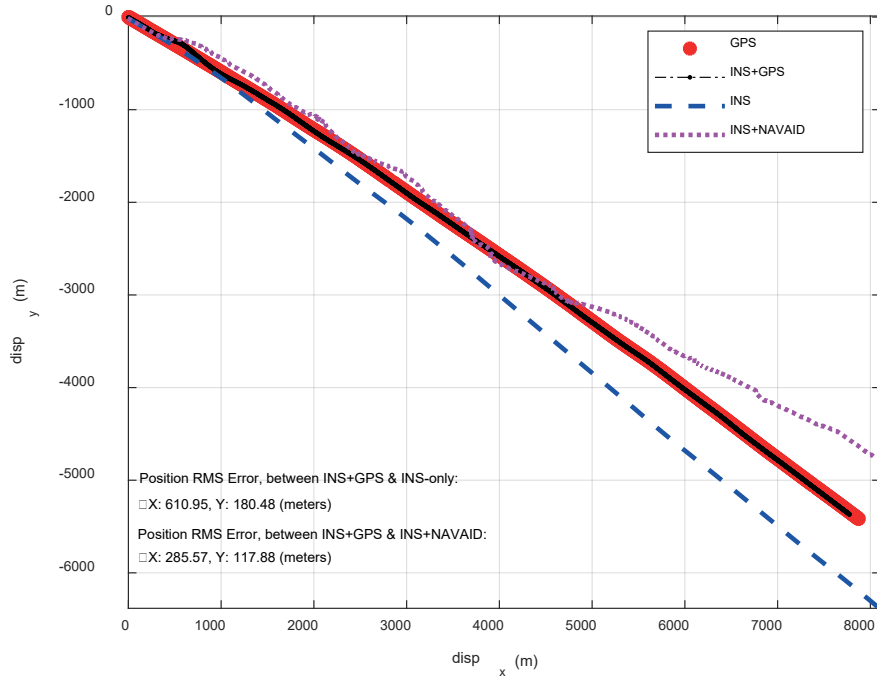


Figure 73. Trajectory comparison of INS-GPS reference, INS-only, and INS-NAVAID FAST

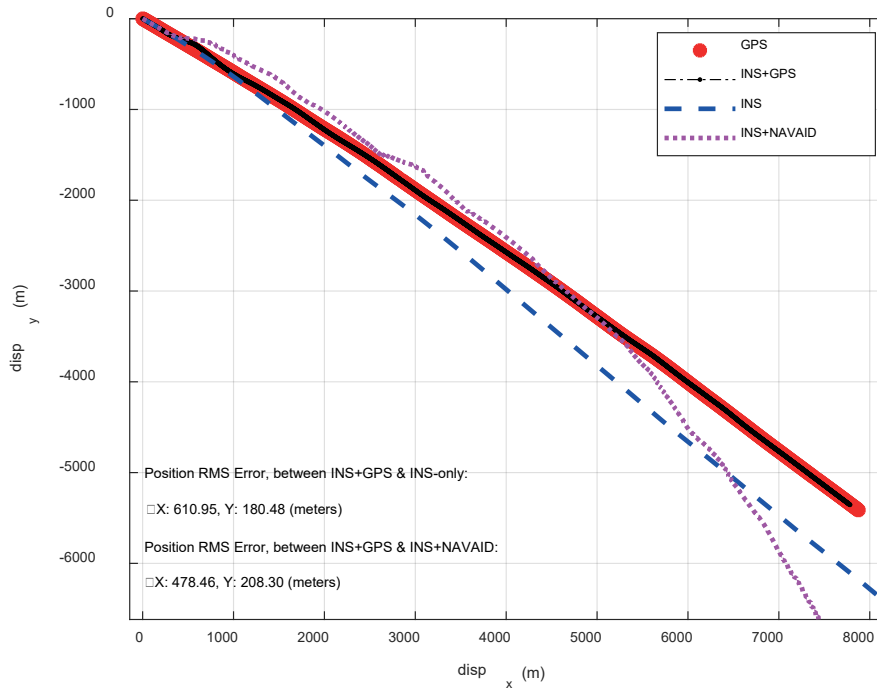


Figure 74. Trajectory comparison of INS-GPS reference, INS-only, and INS-NAVAID ORB

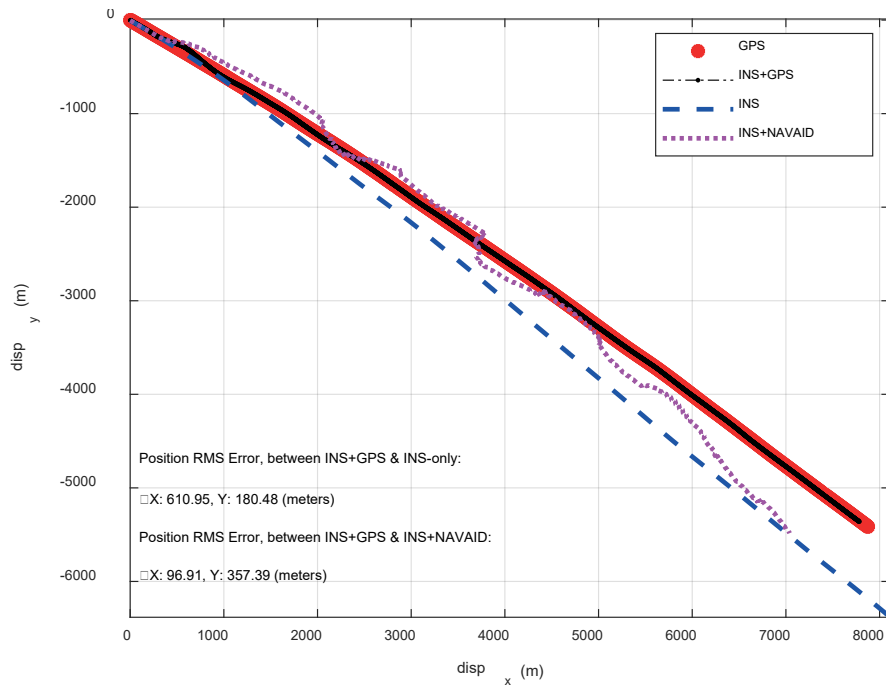


Figure 75. Trajectory comparison of INS-GPS reference, INS-only, and INS-NAVAID SURF

THIS PAGE INTENTIONALLY LEFT BLANK

VIII. CONCLUSION

A. SUMMARY OF WORK DONE

This thesis has demonstrated the feasibility of using on-board electro-optical sensors to provide a vision-based NAVAID solution for a UAS in a GPS-denied environment. The proposed method utilized feature detection algorithms and image matching techniques to deduce geometric transformations across consecutive images and generate pose estimates. A successful attempt incorporating the NAVAID solution, in place of GPS signals, into the navigation system demonstrated the potential for implementing image-matching navigation onto UASs.

The FAST, ORB, and SURF detection methods were selected over other detection algorithms for their short computation time per feature detected. This parameter is crucial for UAS implementation as frequent updates on UAS state are required for the flight computer and mission planner. The number of features detected by the three selected methods were deemed sufficient even when flying over a feature-poor landscape. Low resolution imagery did not hinder the performance of the proposed solution, provided there was an adequate number of feature matches across consecutive image frames.

Furthermore, the current assumption of using similarity transformation for pose estimation showed promising results for straight level flight profiles. The segmented flight profile analysis approach was successful in estimating straight level flight trajectories with good accuracy, as observed in results from all three flight campaigns. Additional image preprocessing was required for the TASE 200 sensor flight campaign, to crop out perspective distortion caused by the forward-looking EO sensor. This indicates that similarity transformation may not be sufficiently robust to handle uncorrected imagery.

In addition, image matching was effective in handling non-periodic imagery acquired from Trinity F90+ and low fps rate video feed from the TASE 200 sensor. This provides a basis for implementing vision-based navigation at a reduced sampling rate to

decrease computation load. Although a 30 fps video was acquired from the self-built sensor suite, it was down sampled to 1 fps to overcome the issue of insufficient and incorrect pixel movement due to rolling shutter effect. Nevertheless, the 1 Hz update rate of the NAVAID solution is adequate for flight and similar to the GPS update rate.

On the other hand, the preliminary results for pose estimation on bank maneuvers performed poorly when compared to reference flight telemetry. This was probably due to large changes in aircraft altitude and attitude distorting image perception. These changes will have to be considered for image preprocessing before proceeding with the analysis of the image-matching navigation.

INS-NAVAID integrated pose results were shown to be better performing than INS-only pose calculations. The proposed INS-NAVAID architecture was executed in MATLAB environment, using a Kalman filter, to provide periodic INS corrections for more accurate pose estimates. As anticipated, the RMS position errors from the INS-NAVAID solution were fewer than for INS-only.

In conclusion, this thesis demonstrated the feasibility of using on-board electro-optical sensors to enhance pose estimation accuracy during GPS-denied situations due to unavailability, degradation, or spoofing. Ongoing improvements to mobile computing also reduce the processing requirements and execution duration for UAS implementation. Hence, UASs will be more resilient as they become less reliant on GPS signals, increasing their mission success rates.

B. RECOMMENDATIONS FOR FUTURE WORK

Results from the analysis conducted in this study highlighted the limitations of using similarity geometric transformation to handle perspective distorted imagery. In consideration of typical UAS missions, which often involve persistent surveillance over the intended area of operations, changes to EO gimbal orientation and bank maneuvers cannot be avoided. With this in mind, future research areas can involve:

- Pose estimation using projective geometric transformations to account for image distortion due to flight maneuvers and changing gimbal orientation.
- Exploration and development of new feature detection algorithms which may offer more feature detections in a shorter computing time.
- Flight demonstrations of real-time NAVAID implementation on a research-oriented UAS platform.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX. NAVAID IMPLEMENTATION SCRIPT

Script for implementing the NAVAID using the Kalman Filter

```
close all, clear all, clc

Load CF090622_Data.mat
SN_num = 1; % Refer to FltSeg.txt for SN label & description

data_list = {'_Data_SURF5000_dn30'}; % To change name
% _ORB8000_dn30.mat
% _FAST8000_dn30.mat
% _SURF5000_dn30.mat

eval(sprintf('load SN%d%s.mat',SN_num,data_list{1}))

stime = SN_Seg(find(SN_Seg(:,1)==SN_num),2); % start time in sec
etime = SN_Seg(find(SN_Seg(:,1)==SN_num),3); % end time in sec

% Pose Estimation in Flight Segments
[val1,ind1] = min(abs(Ref2(:,1)-stime)); %Ref2 - Ext iPhone
[val2,ind2] = min(abs(Ref2(:,1)-etime));
raw_Ref_data = Ref2(ind1:ind2,:);

origin = Ref_data(1,2:4);
[xEast,yNorth] = latlon2local(Ref_data(:,2),Ref_data(:,3),0,origin);
```

Convert XY Distance into GPS Position. Replace null entries with last known value.

```
if find(compiled(:,3)==0)
    rep_data = compiled(find(compiled(:,3)==0,1)-1,:);
    compiled(find(compiled(:,3)==0),:) = repmat(rep_data,size(find(compiled(:,3)==0)));
end

theta = compiled(:,3)/180*pi();
tx = compiled(:,4); ty = compiled(:,5);
dist_ppix = compiled(:,7);

temp_yaw = [0; theta/pi()*180];
temp_tx = [0; tx]; temp_ty = [0; ty];
dist_ppix = [0; dist_ppix];

temp_yaw(1) = temp_yaw(1) + Ref_data(1,6);
temp_tx(1) = temp_tx(1);
temp_ty(1) = temp_ty(1);
cTheta = wrapTo180(cumsum(temp_yaw));
temp_xt = temp_tx.*dist_ppix.*cosd(-cTheta)-temp_ty.*dist_ppix.*sind(-cTheta);
temp_yt = temp_tx.*dist_ppix.*sind(-cTheta)+temp_ty.*dist_ppix.*cosd(-cTheta);
```



```

if ~(size(Ref_data(1:10:end,1),1)==size(temp_Xt,1))
    temp_Xt = interp1(1:size(temp_Xt,1),temp_Xt,1:size(Ref_data(1:10:end,1),1))';
    temp_Yt = interp1(1:size(temp_Yt,1),temp_Yt,1:size(Ref_data(1:10:end,1),1))';
end

Xt = cumsum(temp_Xt); %xEast
Yt = cumsum(temp_Yt); %yNorth

new_T_Alt = [Ref_data(1:10:end,[1,4])];

new_del_T_Alt = new_T_Alt - new_T_Alt(1,:);

[lat,lon,alt] = local2latlon(Xt,Yt,new_del_T_Alt(:,2),origin);

navaid_data = [lat lon new_T_Alt(:,2)]; % Use original altitude data, to replace GPS
new_navaid_data = interp1(new_T_Alt(:,1),navaid_data,raw_Ref_data(:,1));

```

Initialization of logged data

```

eul = raw_Ref_data(1,[13:15])-Ref2(1,[13:15]); %yaw pitch roll
trueOrient = eul2quat(eul); % in quaternion, compact form
truePos = [0 0 new_T_Alt(1,2)]; %XYZ, starting position as (0,0)
initialStateCovariance = (1e-09)*eye(16);
accelData = [raw_Ref_data(:,8) raw_Ref_data(:,9) (raw_Ref_data(:,7))]*-9.81; %AccX, AccY, AccZ
gyroData = raw_Ref_data(:,[11 12 10]); %Rot_X, Rot_Y, Rot_Z
gpsLLA = Ref_data(:,2:4); % Lat Lon Alt
imuFs = 30; % IMU acquisition rate
gpsFs = 10; % GPS acquisition rate
navaidFs = 1; % NAVAID update rate
Rpos = eye(3);

% Create INS filter to fuse IMU and GPS data using an error-state Kalman filter.
initialState =
[trueOrient,truePos,raw_Ref_data(1,5)*cos(deg2rad(Ref_data(1,6))),raw_Ref_data(1,5)*sin(deg2rad(Ref_data(1,6))),0,zeros(1,6),1].';
filt_INS_GPS = insfilterErrorState;
filt_INS_GPS.IMUSampleRate = imuFs;
filt_INS_GPS.ReferenceLocation = origin;
filt_INS_GPS.State = initialState;
filt_INS_GPS.StateCovariance = initialStateCovariance;

% Preallocate variables for position and orientation.
% Allocate a variable for indexing into the GPS data.
numIMUSamples = size(accelData,1); % accx, accy, accz
estOrient = ones(numIMUSamples,1,'quaternion');
estPos = zeros(numIMUSamples,3);
estOrient_INS_GPS = estOrient; estPos_INS_GPS = estPos;
estOrient_INS = estOrient; estPos_INS = estPos;
estOrient_INS_NAVAID = estOrient; estPos_INS_NAVAID = estPos;
gpsIdx = 1; navaidIdx = 1;

```

Fuse accelerometer, gyroscope, and GPS data. The outer loop predicts the filter forward at the fastest sample rate (the IMU sample rate).

```
for idx = 1:numIMUSamples

    % Use predict to estimate the filter state based on the accelData and
    % gyroData arrays.
    predict(filt_INS_GPS,accelData(idx,:),gyroData(idx,:));

    % GPS data is collected at a lower sample rate than IMU data.
    % Fuse GPS data at the lower rate.
    if mod(idx, imuFs / gpsFs) == 0
        % Correct the filter states based on the GPS data.
        fusegps(filt_INS_GPS,gpsLLA(gpsIdx,:),Rpos);
        gpsIdx = gpsIdx + 1;
    end

    % Log the current pose estimate
    [estPos_INS_GPS(idx,:), estOrient_INS_GPS(idx,:)] = pose(filt_INS_GPS);
end
```

Fuse only INS data, no GPS

```
filt_INS = insfilterErrorState;
filt_INS.IMUSampleRate = imuFs;
filt_INS.ReferenceLocation = origin;
filt_INS.State = initialState;
filt_INS.StateCovariance = initialStateCovariance;

for idx = 1:numIMUSamples

    % Use predict to estimate the filter state based on the accelData and
    % gyroData arrays.
    predict(filt_INS,accelData(idx,:),gyroData(idx,:));

    % Log the current pose estimate
    [estPos_INS(idx,:), estOrient_INS(idx,:)] = pose(filt_INS);
end
```

Replace GPS with NAVAID, fuse accelerometer, gyroscope, and NAVAID data

```
filt_INS_NAVAID = insfilterErrorState;
filt_INS_NAVAID.IMUSampleRate = imuFs;
filt_INS_NAVAID.ReferenceLocation = origin;
filt_INS_NAVAID.State = initialState;
filt_INS_NAVAID.StateCovariance = initialStateCovariance;

for idx = 1:numIMUSamples
```

```

% Use predict to estimate the filter state based on the accelData and
% gyroData arrays.
predict(filt_INS_NAVID, accelData(idx,:), gyroData(idx,:));

% GPS data is collected at a lower sample rate than IMU data.
% Fuse GPS data at the lower rate.
if mod(idx, imuFs / navaidFs) == 0
    % Correct the filter states based on the GPS data.
    fusegps(filt_INS_NAVID, navaid_data(navaidIdx,:), Rpos);
    navaidIdx = navaidIdx + 1;
end

% Log the current pose estimate
[estPos_INS_NAVID(idx,:), estOrient_INS_NAVID(idx,:)] = pose(filt_INS_NAVID);
end

```

Calculate the RMS errors between the known true position and the output from the error-state filter

```

pErr1 = estPos_INS_GPS(:, [1 2]) - estPos_INS(:, [1 2]); % Error between INS+GPS & INS
pRMS1 = sqrt(mean(pErr1.^2));
fprintf('Position RMS Error, between INS+GPS & INS-only:\n');
pRMS1str = sprintf('\tX: %.2f, Y: %.2f (meters)\n\n', pRMS1(1), pRMS1(2));
fprintf(pRMS1str)

pErr2 = estPos_INS_GPS(:, [1 2]) - estPos_INS_NAVID(:, [1 2]); % Error between INS+GPS &
INS
pRMS2 = sqrt(mean(pErr2.^2));
fprintf('Position RMS Error, between INS+GPS & INS+NAVAID:\n');
pRMS2str = sprintf('\tX: %.2f, Y: %.2f (meters)\n\n', pRMS2(1), pRMS2(2));
fprintf(pRMS2str)

% Visualize the true position and the estimated position
figure
plot(xEast, yNorth, 'r*', 'Linewidth', 2) % GPS only
hold on; grid on; axis equal;
plot(estPos_INS_GPS(:, 2), estPos_INS_GPS(:, 1), 'k.-.') %,'Linewidth', 2
plot(estPos_INS(:, 2), estPos_INS(:, 1), 'b--', 'Linewidth', 2); %,'Linewidth', 2
plot(estPos_INS_NAVID(:, 2), estPos_INS_NAVID(:, 1), 'm:', 'Linewidth', 2); %,'Linewidth', 2

xlabel('disp_x (m)'); ylabel('disp_y (m)');
legend('GPS', 'INS+GPS', 'INS', 'INS+NAVAID', 'Location', 'best') %

str = {
    'Position RMS Error, between INS+GPS & INS-only:'
    pRMS1str
    'Position RMS Error, between INS+GPS & INS+NAVAID:'
    pRMS2str
};

```

```
gtext(str) % Manual insert text in figure
```

Published with MATLAB® R2022a

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

- Alcantarilla, Pablo Fernández, Adrien Bartoli, and Andrew J. Davison. 2012. “KAZE Features.” In *Computer Vision – ECCV 2012*, 214–27. https://doi.org/10.1007/978-3-642-33783-3_16.
- Bay, Herbert, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. 2008. “Speeded-Up Robust Features (SURF).” *Computer Vision and Image Understanding* 110 (3) (June): 346–59. <https://doi.org/10.1016/j.cviu.2007.09.014>.
- Bednář, Jan, Matěj Petrlík, Kelen Cristiane Teixeira Vivaldini, Martin Saska. 2022. “Deployment of reliable visual inertial odometry approaches for unmanned aerial vehicles in real-world environment.” Paper presented at the 2022 International Conference on Unmanned Aircraft Systems, Dubrovnik, Croatia.
- Burghouts, Gertjan J., and Jan-Mark Geusebroek. 2009. “Performance Evaluation of Local Colour Invariants.” *Computer Vision and Image Understanding* 113 (1) (January): 48–62. <https://doi.org/10.1016/j.cviu.2008.07.003>.
- Choi, Sunglok, Taemin Kim, and Wonpil Yu. 2009. “Performance Evaluation of RANSAC Family.” In *British Machine Vision Conference (BMVC)*, 1–12. <http://dx.doi.org/10.5244/C.23.81>.
- Cole, Sally. 2017. “Ensuring navigation in GPS-denied environments.” Military Embedded Systems. November 20, 2017. <https://militaryembedded.com/comms/satellites/ensuring-navigation-gps-denied-environments>.
- Courbon, Jonathan, Youcef, Mezouar, Nicolas Guénard, and Philippe Martinet. 2010. “Vision-Based Navigation of Unmanned Aerial Vehicles.” *Control Engineering Practice* 18 (7) (June): 789–99. <https://doi.org/10.1016/j.conengprac.2010.03.004>.
- Department of the Navy. 2021. *Department of Navy Unmanned Campaign Framework*. Report No. AD1125317. Washington, DC: Department of the Navy. <https://apps.dtic.mil/sti/citations/AD1125317>.
- Desouza, Guilherme N., and Avinash C. Kak. 2002. “Vision for Mobile Robot Navigation: A Survey.” *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24 (2) (February): 237–67. <https://doi.org/10.1109/34.982903>.
- Ebner, Marc. 2007. *Color Constancy*. New York: Wiley.
- Gilliland, Ben. 2019. “GPS Is Easy to Hack, and the U.S. Has No Backup.” *Scientific American*, December 1, 2019. <https://www.scientificamerican.com/article/gps-is-easy-to-hack-and-the-u-s-has-no-backup/>.

- Gui, Yang, Pengyu Guo, Hongliang Zhang, Zhihui Lei, Xiang Zhou, Jing Du, and Qifeng Yu. 2013. "Airborne Vision-Based Navigation Method for UAV Accuracy Landing Using Infrared Lamps." *Journal of Intelligent & Robotic Systems* 72 (2) (March): 197–218. <https://doi.org/10.1007/s10846-013-9819-5>.
- Han, Keng Siew Aloysius. 2017. "Test and Evaluation of an Image-Matching Navigation System for a UAS Operating in a GPS-Denied Environment." Master's thesis, Naval Postgraduate School. <https://calhoun.nps.edu/handle/10945/56131>.
- Harris, Chris, and Mike Stephens. 1988. "A Combined Corner and Edge Detector." In *4th Alvey Vision Conference*, 147–51. <http://dx.doi.org/10.5244/C.2.23>.
- Kaniewski, Piotr, and Wojciech Grzywacz. 2017. "Visual-Based Navigation System for Unmanned Aerial Vehicles." In *2017 IEEE Signal Processing Symposium (SPSymposium)*, 1–6. <https://doi.org/10.1109/SPS.2017.8053686>.
- Kong, W. Y., Gregory K. Egan, and Terry Cornall. 2006. "Feature Based Navigation for UAVs." In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 3539–43. <https://doi.org/10.1109/IROS.2006.281640>.
- Leutenegger, Stefan, Margarita Chli, and Roland Y. Siegwart. 2011. "BRISK: Binary Robust Invariant Scalable Keypoints." In *2011 International Conference on Computer Vision*, 2548–55. <https://doi.org/10.1109/ICCV.2011.6126542>.
- Liew, Maegan. 2019. "Doing More with Less: The Singaporean Armed Forces Prepare for Manpower Shortages and Evolving Regional Challenges." *ASEAN Today*, November 5, 2019. <https://www.aseantoday.com/2019/11/doing-more-with-less-the-singaporean-armed-forces-prepare-for-manpower-shortages-and-evolving-regional-challenges/>.
- Lu, Yuncheng, Zhucun Xue, Gui-Song Xia, and Liangpei Zhang. 2018. "A Survey on Vision-Based UAV Navigation." *Geo-Spatial Information Science* 21 (1) (January): 21–32. <https://doi.org/10.1080/10095020.2017.1420509>.
- Madison, R., G. Andrews, P. DeBitetto, S. Rasmussen, and M. Bottkol. 2007. "Vision-Aided Navigation for Small UAVs in GPS-Challenged Environments." In *AIAA Infotech@Aerospace 2007 Conference and Exhibit*. <https://doi.org/10.2514/6.2007-2986>.
- Mallet, Anthony, Simon Lacroix, and Laurent Gallo. 2000. "Position Estimation in Outdoor Environments Using Pixel Tracking and Stereovision." In *2000 IEEE ICRA Conference*, (4) (April): 3519–3524. <https://doi.org/10.1109/ROBOT.2000.845279>.
- MathWorks. n.d. "Create a Gallery of Transformed Images." Accessed April 2, 2022. <https://www.mathworks.com/help/images/creating-a-gallery-of-transformed-images.html>.

- MathWorks. n.d. “insfilterErrorState” Accessed July 10, 2022. <https://www.mathworks.com/help/fusion/ref/insfiltererrorstate.html#d123e61769>.
- Miller, Mikel M., Andrey Soloviev, Maarten Uijt de Haag, Michael Veth, John Raquet, Timothy J. Klausutis, and Jimmy E. Touma. 2010. “Navigation in GPS Denied Environments: Feature-Aided Inertial Systems.” NATO Report RTO-EN-SET-116.
- Nistér, David, and Henrik Stewénus. 2008. “Linear Time Maximally Stable Extremal Regions.” In *Computer Vision – ECCV 2008*, 183–96. https://doi.org/10.1007/978-3-540-88688-4_14.
- Perez-Grau, Francisco J., Ricardo Ragel, Fernando Caballero, Antidio Viguria, and Anibal Ollero. 2018. “An Architecture for Robust UAV Navigation in GPS-denied Areas.” *Journal of Field Robotics* 35 (1) (January): 121–45. <https://doi.org/10.1002/rob.21757>.
- QStarz International Co. Ltd. n.d. “GPS eXtreme Recorder Specifications.” Accessed July 8, 2022. <http://www.qstarz.com/Products/GPS%20Products/BT-Q1000EXQR-S.htm>.
- Quantum Systems. n.d. “Trinity F90Plus Mapping Drone.” Accessed May 30, 2022. <https://www.quantum-systems.com/project/trinityf90plus-mapping-drone/>.
- Rosten, Edward, and Tom Drummond. 2005. “Fusing Points and Lines for High Performance Tracking.” In *Tenth IEEE International Conference on Computer Vision (ICCV)*, 1508–1515. <https://doi.org/10.1109/ICCV.2005.104>.
- Rublee, Ethan, Vincent Rabaud, Jurt Konolige, and Gary Bradski. 2011. “ORB: An Efficient Alternative to SIFT or SURF.” In *2011 International Conference on Computer Vision*, 2564–71. <https://doi.org/10.1109/ICCV.2011.6126544>.
- Sasiadek, Jurek, and Mark Walker. 2008. “Vision-Based UAV Navigation.” In *AIAA Guidance, Navigation and Control Conference and Exhibit*. <https://doi.org/10.2514/6.2008-6667>.
- Shi, Jianbo, and Carlo Tomasi. 1994. “Good Features to Track.” In *1994 Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 593–600. <https://doi.org/10.1109/CVPR.1994.323794>.
- Valasek, John, Kiran Gunnam, Jennifer Kimmet, John L. Junkins, Declan Hughes, and Monish D. Tandale. 2005. “Vision-Based Sensor and Navigation System for Autonomous Air Refueling.” *Journal of Guidance, Control, and Dynamics* 28 (5) (September): 979–89. <https://doi.org/10.2514/1.11934>.
- Weijer, Joost van de, Theo Gevers, and Andrew D. Bagdanov. 2006. “Boosting Color Saliency in Image Feature Detection.” *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28 (1) (January): 150–56. <https://doi.org/10.1109/TPAMI.2006.3>.

- Yakimenko, Oleg, Isaac Kaminer, W. Jerry Lentz, and P. A. Ghyzel. 2002. "Unmanned Aircraft Navigation for Shipboard Landing Using Infrared Vision." *IEEE Transactions on Aerospace and Electronic Systems* 38 (4) (October): 1181–200. <https://doi.org/10.1109/TAES.2002.1145742>.
- Yakimenko, Oleg, and Ryan Decker. 2019. Image-matching navigation method and apparatus for aerial vehicles. U.S. Patent 10,515,458, filed February 28, 2018, and issued December 24, 2019. <http://hdl.handle.net/10945/64482>.
- . 2017. "On the Development of an Image-Matching Navigation Algorithm for Aerial Vehicles." In *2017 IEEE Aerospace Conference*, 1–9. <https://doi.org/10.1109/AERO.2017.7943742>.
- Yakimenko, Oleg, and Nathan Slegers. 2015. "Equations of Motion." In *Precision Aerial Delivery Systems: Modeling, Dynamics, and Control*, edited by Oleg Yakimenko, 263–352. Reston, VA: American Institute of Aeronautics and Astronautics.
- Zhang, Jun, Weisong Liu, and Yirong Wu. 2011. "Novel Technique for Vision-Based UAV Navigation." *IEEE Transactions on Aerospace and Electronic Systems* 47 (4) (October): 2731–41. <https://doi.org/10.1109/TAES.2011.6034661>.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
Ft. Belvoir, Virginia
2. Dudley Knox Library
Naval Postgraduate School
Monterey, California