ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

A FEEDBACK STAR IDENTIFICATION ALGORITHM VIA REGULARIZED PATTERN RECOGNITION USING A UNIQUE FEATURE EXTRACTION

Ph.D. THESIS

Erdem Onur ÖZYURT

Department of Aeronautics and Astronautics Engineering

Aeronautics and Astronautics Engineering Programme

JULY 2024

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

A FEEDBACK STAR IDENTIFICATION ALGORITHM VIA REGULARIZED PATTERN RECOGNITION USING A UNIQUE FEATURE EXTRACTION

Ph.D. THESIS

Erdem Onur ÖZYURT
(511172118)

Department of Aeronautics and Astronautics Engineering

Aeronautics and Astronautics Engineering Programme

Thesis Advisor: Prof. Dr. Alim Rüstem ASLAN

JULY 2024

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

ÖZGÜN ÖZNİTELİKLER İLE REGÜLARİZASYON VE ÖRÜNTÜ TANIMA TABANLI GERİ BİLDİRİMLİ YILDIZ TANIMA ALGORİTMASI

DOKTORA TEZİ

Erdem Onur ÖZYURT
(511172118)

Uçak ve Uzay Mühendisliği Anabilim Dalı

Uçak ve Uzay Mühendisliği Programı

Tez Danışmanı: Prof. Dr. Alim Rüstem ASLAN

TEMMUZ 2024

Erdem Onur ÖZYURT, a Ph.D. student of ITU Graduate School student ID 511172118 successfully defended the thesis entitled "A FEEDBACK STAR IDENTIFICATION ALGORITHM VIA REGULARIZED PATTERN RECOGNITION USING A UNIQUE FEATURE EXTRACTION", which he/she prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Prof. Dr. Alim Rüstem ASLAN** Istanbul Technical University	.....
Jury Members :	**Prof. Dr. Cengiz HACIZADE** Istanbul Technical University	.....
	Prof. Dr. Faruk BAĞCI Turkish-German University	.....
	Asst. Prof. Dr. Demet ÇİLDEN GÜLER Istanbul Technical University	.....
	Asst. Prof. Dr. Tuncay Yunus ERKEÇ Turkish National Defense University	.....

Date of Submission : **12 June 2024**
Date of Defense : **11 July 2024**

FOREWORD

I would like to thank my thesis advisor Prof. Dr. A. Rüstem ASLAN, who has given me the opportunity to benefit from the equipment in the USTTL (Space Systems Design and Test Laboratory) located in the Department of Aeronautics and Astronautics Engineering at Istanbul Technical University, for leading the way to become an astronautics engineer. I would like to thank the very competent USTTL team and the lead engineer MSc. Boğaç KARABULUT who have been engaged in the project Sharjah-Sat-1, in which the star sensor CubeStar, which is simulated in this study, is used. I would also like to thank Prof. Dr. Cengiz HACIZADE and Prof. Dr. Faruk BAĞCI, for their very valuable suggestions and corrections that pave the way for better ideas and implementations. Finally, I should pay my respects to Prof. Dr. Bilge GÜNSEL from Multimedia Signal Processing and Pattern Recognition Laboratory at Istanbul Technical University, who helped me gain a distinguished vision in signal processing, and to Asst. Prof. Cuma YARIM from the Department of Aeronautics and Astronautics Engineering at Istanbul Technical University, from whom I have learned a lot about the fundamentals of astronautics engineering.

July 2024

Erdem Onur ÖZYURT
(Electronics and Communication Engineer, M. Sc.)

TABLE OF CONTENTS

ABBREVIATIONS

ADCS : Attitude Determination and Control System
AU : Astronomical Unit
CCD : Charge Coupled Device
CME : Camera Motion Estimation
CMOS : Complementary Metal-Oxide-Semiconductor
DS : Database Shrinkage
ESOQ : Estimators of Optimal Quaternion
FOAM : Fast Optimal Attitude Matrix
FoV : Field of View
FN : False Negative
FP : False Positive
FSF : False Star Filtering
GA : Grid Algorithm
GV : Geometric Voting Algorithm
IA : Iteration Algorithm
ICRS : International Celestial Reference System
IG : Improved Grid Algorithm
kNN : k-Nearest Neighbor
K-L : Karhunen-Loève
LASSO : Least Absolute Shrinkage and Selection Operator
LSI : Lost-in-space Star Identification
LPT : Log-Polar Transform Algorithm
LSE : Least Square Error
MG : Match Group Algorithm
MGA : Modified Grid Algorithm
MLESAC : Maximum Likelihood Estimator Sample Consensus
MSE : Mean Squared Error
OSV : Oriented Singular Value Algorithm
PA : Pyramid Algorithm
QUEST : Quaternion Estimator
RANSAC : Random Sample Consensus
RoS : Region of Search
RSI : Recursive Star Identification
SGS : Simplest General Subgraph
SVD : Singular Value Decomposition Algorithm
TN : True Negative
TP : True Positive
UBV : Ultraviolet Blue Visual
VP : One-Dimensional Vector Pattern Algorithm
1NN : One Nearest Neighbor
1SE : One Standard Error

SYMBOLS

a	: Penalty term in $\mathscr{P}_a(\cdot)$
A	: Mean of brightness for objects in $\mathscr{S}$
$\hat{A}$	: Normalized mean of brightness for objects in $\mathscr{S}$
A_{ij}	: Mean of brightness for a pair of objects i and j
a_i	: Brightness of an object i in $\mathscr{S}$
$\mathbf{C}_k$	: Cluster k
c_i	: Centroid of an object i
D	: Number of vectors in disparity list
$\mathscr{D}$	: Disparity list
$\overrightarrow{d}_i$	: Disparity vectors in $\mathscr{D}$
$\overrightarrow{d}_k$	: Candidate core disparity vector in $\mathbf{C}_k$
$\overrightarrow{d}_l$	: Unlabeled disparity vector
d_i	: Euclidean distances yielded by $\mathscr{F}_{\mathrm{Ed}}(\cdot)$
d_{ab}	: Angular distance line between objects a and b
f_i	: i false stars injected into image frame
$\overrightarrow{\tilde{f}}^t_i$	: Estimated false stars at image t
$\overrightarrow{f}$	: Ground truth inertial observation boresight vector
$\overrightarrow{\tilde{f}}$	: Estimation of inertial observation boresight vector
$\mathbf{F}$	: Boresight template matrix
$\mathscr{F}_{\mathrm{d}}(\cdot)$	: Function for generating disparity list
$\mathscr{F}_{\mathrm{D}}(\cdot)$	: Function for setting dictionary setup
$\mathscr{F}_{\mathrm{db}}(\cdot)$	: Function for density-based clustering
$\mathscr{F}_{\mathrm{Ed}}(\cdot)$	: Function for calculation of Euclidean distances
$\mathscr{F}_{\mathrm{H}}(\cdot)$	: Function for estimating affine transformation matrix
$\mathscr{F}_{\mathrm{LSI}}(\cdot)$	: LSI function
$\mathscr{F}_{\mathrm{N}}(\cdot)$	: Function for normalization
$\mathscr{F}_{\mathrm{R}}(\cdot)$	: Function for regularization
$\mathscr{F}_{\mathrm{RoS}}(\cdot)$	: Function for database shrinkage
$\mathscr{F}_{\mathrm{RSI}}(\cdot)$	: RSI function
$\mathscr{F}_{\mathrm{trace}}(\cdot)$	: Function for tracing back $\overrightarrow{\tilde{s}}^t_i$ and $\overrightarrow{\tilde{f}}^t_i$
$\mathscr{F}_{\mathrm{1NN}}(\cdot)$	: Function for implementation of 1NN classifier
$\mathscr{F}_{\psi}(\cdot)$	: Function for extraction of feature vector
$\mathscr{F}_{\theta}(\cdot)$	: Function for extraction of polar angle
$\tilde{\mathbf{H}}$	: Estimated affine transformation matrix
h_{ij}	: Elements of $\tilde{\mathbf{H}}$
$\mathbf{I_o}$	: Observed digital image frame
K	: Factor of τ_{ε_ψ} for invalid estimation
l	: Distance metric
L	: Number of feature vectors in an image frame

m : Number of database vectors in a single row along right ascension
M : Length of regularization weight vector $\overrightarrow{\omega}$
m_{ab} : Slope angle of d_{ab} with respect to image frame
$\min_p$: Minimum number of vectors within ε to form $\mathbf{C}_k$
$\mathbf{M_V}$: Absolute visual magnitude
n : Number of stars in an image frame
$\hat{n}$: Normalized number of stars in an image frame
N : Total number of database feature vectors
$\mathcal{N}(\cdot)$: Normal distribution
N_o : Number of detected objects in an image frame
N_{RoS} : Number of database feature vectors bounded by the RoS
p : Overlapping ratio
p_{min} : Minimum value of overlapping ratio to avoid information loss
$\mathscr{P}_a(\cdot)$: Function for penalty term in Elastic Net
$\tilde{\mathbf{R}}$: Rotation part of $\tilde{\mathbf{H}}$
r : Polar radius of a frame object
$\hat{r}$: Normalized polar radius of a frame object
r_i : Polar radius of an object in an image frame
$\overrightarrow{\tilde{s}}^t_i$: Estimated true stars at image t
$\mathscr{S}$: Set that contains objects detected in a frame
$\mathbf{T}$: Dictionary template matrix
t_j : Dictionary vectors
t_5 : The most significant dictionary vector
$\mathscr{U}(\cdot)$: Uniform distribution
$\tilde{\mathbf{U}}$: Translation part of $\tilde{\mathbf{H}}$
$\mathscr{V}^t$: Set of feature vectors extracted from image frame t
$\overrightarrow{v}^t_k$: Feature vectors in $\mathscr{V}^t$
$\mathbf{V_{mag}}$: Visual magnitude
x : The measure of intrusion
X : Horizontal coordinate component of a frame object
x_i : Horizontal pixel coordinates in an image frame
Y : Vertical coordinate component of a frame object
y_i : Vertical pixel coordinates in an image frame
α : Right ascension in degrees
$\tilde{\alpha}$: Estimation of observation right ascension in degrees
δ : Declination in degrees
$\tilde{\delta}$: Estimation of observation declination in degrees
ε : Radius for neighborhood of $\overrightarrow{d}_k$
ε_{LSE} : The least square error
ε_f : Error for estimated inertial boresight vector
ε_γ : Error for mean of rotation motion
ε_ψ : Error for reconstructed observation feature vector
ε_ρ : Error for mean of translation motion
ε_θ : Error for estimated rotation angle about $\overrightarrow{f}$
η : Threshold for distance metric l
ϕ : FoV angle in one direction in circular database frames
ϕ_x : FoV angle in the direction of right ascension

ϕ_y : FoV angle in the direction of declination
γ : Rotation between observation images
$\tilde{\gamma}$: Estimated rotation between observation images
Γ : One side of the RoS
κ : Brightness threshold
λ : Non-negative regularization parameter
μ_A : Mean of brightness of all objects in $\mathscr{S}$
μ_d : Mean of dislocations of stars transformed by $\tilde{\mathbf{H}}$
μ_j : Noise factor
μ_r : Mean of polar radii of all objects in $\mathscr{S}$
$\overrightarrow{\omega}$: Regularization weight vector
π : Trigonometric parallax in milliarcsec
$\overrightarrow{\psi}$: Observation feature vector
$\hat{\overrightarrow{\psi}}$: Normalized observation feature vector
$\tilde{\overrightarrow{\psi}}$: Reconstruction of observation feature vector
$\overrightarrow{\rho}$: Translation vector
$\tilde{\overrightarrow{\rho}}$: Estimated translation vector
$\rho_{\mu\sigma}$: Correlation coefficient between μ_d and σ_d
σ_A : Brightness standard deviation
$\hat{\sigma}_A$: Normalized brightness standard deviation
σ_d : Standard deviation of dislocations of stars transformed by $\tilde{\mathbf{H}}$
σ_r : Polar radius standard deviation
$\hat{\sigma}_r$: Normalized polar radius standard deviation
σ_ρ : Standard deviation of translation motion error
σ_γ : Standard deviation of rotation motion error
$\overrightarrow{\tau}$: Database feature vector
$\hat{\overrightarrow{\tau}}$: Normalized database feature vector
τ_{ε_ψ} : Error threshold for ε_ψ
θ : Polar angle of a frame object
$\tilde{\theta}$: Estimation of rotation angle about $\tilde{\overrightarrow{f}}$
θ_δ : Ground truth rotation angle about $\overrightarrow{f}$ with respect to $\delta = 0^\circ$
Θ : Rotation template matrix

LIST OF TABLES

LIST OF FIGURES

Page

A FEEDBACK STAR IDENTIFICATION ALGORITHM VIA REGULARIZED PATTERN RECOGNITION USING A UNIQUE FEATURE EXTRACTION

SUMMARY

This thesis presents a star identification algorithm integrated with preprocessing. Star sensors, which are highly reliable for attitude determination use of spacecrafts and satellites, relies on star identification algorithms. The star identification algorithm proposed in this study is capable of functioning either in lost-in-space method or recursive method. Both methods utilize a unique feature extraction scheme. This novel approach of feature extraction method extracts a single vector from each captured image instead of treating each star as a separate object. This cumulative approach aims to save a significant amount of memory space while taking the entire catalog into account for elevated accuracy. A database containing stars from the catalog is constructed using the unconventional features extracted from each corresponding field-of-view. The databases may differ in size and detail dependent on the parameters of overlapping ratio and brightness threshold. These parameters have a significant effect on accuracy and complexity of the method. The method aims to estimate the inertial boresight vector and the rotation angle about it. This is a novel approach that is carried out by matching frames but not matching individual stars, star pairs, star triangles or star polygons. Both star identification methods rely on pattern recognition and regularization successively. First, a 1NN classifier is used to perform a coarse estimation with limited accuracy specified by the characteristics of the database with predetermined parameters. The coarse estimation is exactly the database vector that is most similar to the observation vector. Subsequently, a dictionary is generated using the neighbor database vectors of the most similar database vector. The final estimation is obtained by conducting a regularization method for fine estimation. A solution coefficient vector is yielded through regularization. The estimates of boresight vector and rotation angle are retrieved using the solution coefficient vector. This is the output of the lost-in-space star identification method. Since the lost-in-space algorithm is very sensitive to false stars, an additional false star filtering algorithm is developed. This algorithm is based on density-based clustering. A disparity list is created using two successive image frames. After estimating true stars by implementing density-based clustering on the disparity list, false stars are removed. Using the successive frames containing only estimated true stars, an affine transformation matrix is obtained by a regression analysis procedure. In order to overcome the issues tackling the lost-in-space star method, the recursive star identification method is developed. Apart from the algorithmic structure taken from the lost-in-space method, it possesses an update mechanism that ensures usage a much smaller portion of the database to reduce computational complexity and average run time. Also, the false star filtering avoids sensitivity to false stars. Thus, the integrated algorithm not only increases accuracy but also reduces computational complexity and average run time. The performance of the

proposed algorithms is evaluated in a simulation environment also developed within scope of this study. The simulation environment allows generation of image frames with given sensor parameters and database information. It also allows realization of different noise scenarios including all types of noise. Camera motion can be simulated with random or biased iterations. After conducting a number of experiments to specify the optimal parameters to be used in different steps of the algorithms, a vast number of experiments are carried out to evaluate the algorithm performances. the performance of lost-in-space star identification is statistically evaluated by means of error plots, precision rate curves and identification rate curves. And, complexity analysis is carried out by measuring database size and average run time. It is observed that accuracy increases by using databases with larger overlapping ratio but compromises complexity. The proposed algorithm outperforms state-of-the-art methods in terms of accuracy but lacks behind in terms of database size and average run time. After optimal parameter selection, the performance of false star filtering is evaluated by means of confusion matrix and statistical indicators while the algorithm for camera motion estimation is evaluated through distance error measurement. Both algorithms show superior performance. The recursive star identification algorithm is also subject to experiments for optimal parameter selection. After selecting optimal parameters of error threshold and scan region, it is compared to the lost-in-space star identification method in terms of accuracy and complexity. It is concluded that memory usage and run time is dramatically decreased without compromising accuracy. Besides, the issue of false star sensitivity is resolved. Nevertheless, the algorithm is still sensitive to missing stars. The studies continue to extend the false star filtering algorithm also to detect missing stars. Since the algorithm is based on features obtained using the whole field-of-view, a big object blocking the field-of-view incurs missing stars so that accuracy is significantly deteriorated. Thus, apart from the sun, the moon also impairs the proposed method when comes into view. The proposed method is also being developed to overcome this issue by splitting the field-of-view into smaller portions. Finally, the proposed method has a number of procedures dependent on parameters. Therefore, the whole process is highly dependent on parameters. This allows flexibility if the procedure is thoroughly followed. Many following studies are planned to be carried out to investigate each parameter's effect on reliability and complexity.

ÖZGÜN ÖZNİTELİKLER İLE REGÜLARİZASYON VE ÖRÜNTÜ TANIMA TABANLI GERİ BİLDİRİMLİ YILDIZ TANIMA ALGORİTMASI

ÖZET

Bu çalışmada, ön işleme ile entegre edilmiş bir yıldız tanıma algoritması sunulmaktadır. Uzay araçları ve uyduların yönelim belirleme amaçlı kullanımında oldukça güvenilir olan yıldız izleyicileri; güneş sensörü, ufuk sensörü ve manyetizma sensörü gibi cihazlara göre çok daha yüksek doğruluk sağlar. Bu sebeple yüksek hassasiyet gerektiren uzay görevlerinde kullanılan başlıca yönelim belirleme cihazlarındandır. Yıldız izleyiciler, yıldız tanıma algoritmalarına dayanmaktadır. Bu çalışmada önerilen yıldız tanıma algoritması hem ön bilgisiz yöntemle hem de özyinelemeli yöntemle çalışabilmektedir. Her iki yöntemde de benzersiz bir öznitelik çıkarımı kullanılmaktadır. Öznitelik çıkarımındaki bu yeni yaklaşım, her bir yıldızı ayrı bir nesne olarak ele almak yerine, çekilen her görüntüden tek bir vektör çıkarmaktadır. Öznitelik çıkarım işleminde sensör özellikleri belirleyici olmaktadır. Öznitelik vektörü; her görüntü karesinden elde edilen tekil nesnenin parlaklık, konum ve yıldız sayısı bilgisini içermektedir. Bu kümülatif yaklaşım, yüksek doğruluk sağlamak için tüm kataloğu hesaba katarken önemli miktarda bellek alanından tasarruf etmeyi amaçlamaktadır. Katalogdaki yıldızları içeren bir veritabanı, ilgili her bir görüş alanından çıkarılan geleneksel olmayan öznitelikler kullanılarak oluşturulur. Veritabanları, örtüşme oranı ve parlaklık eşiği parametrelerine bağlı olarak boyut ve ayrıntı bakımından farklılık gösterebilir. Bu parametreler yöntemin doğruluğu ve karmaşıklığı üzerinde önemli bir etkiye sahiptir. Gözlem öznitelik vektörleri elde edilirken de aynı öznitelik çıkarım yöntemi kullanılmaktadır. Yöntem, eylemsiz eksende doğrultu vektörünü ve bu vektör etrafındaki dönüş açısının kestirimini yapmayı amaçlamaktadır. Bu, ayrı ayrı yıldızları, yıldız çiftlerini, yıldız üçgenlerini veya yıldız poligonlarını eşleştirmek yerine görüntü çerçevelerini eşleştirerek gerçekleştirilen yeni bir yaklaşımdır. Her iki yıldız tanıma yönteminde de ardışık olarak örüntü tanıma ve regülarizasyon yöntemleri kullanılmaktadır. İlk olarak, önceden belirlenmiş parametrelerle veritabanının özellikleri tarafından belirlenen sınırlı doğrulukla kaba bir kestirim gerçekleştirmek için bir 1NN sınıflandırıcı kullanılır. Kaba kestirim tam olarak gözlem öznitelik vektörüne en çok benzeyen veritabanı öznitelik vektörüdür. Daha sonra, benzerliği en yüksek veritabanı öznitelik vektörüne komşu olan veritabanı öznitelik vektörleri kullanılarak bir sözlük oluşturulur. Nihai kestirim, ince kestirimden elde edilen sonuçtur. Bu kestirim sonucu bir regülarizasyon yöntemi uygulanarak elde edilir. Regülarizasyon ile bir çözüm katsayısı vektörü elde edilir. Doğrultu vektörü ve dönüş açısı kestirimleri çözüm katsayısı vektörü kullanılarak elde edilir. Bu vektör, ön bilgisiz yıldız tanıma yönteminin sonuç çıktısıdır. Ön bilgisiz yıldız tanıma algoritması yalancı yıldızlara karşı çok hassas olduğundan, ilave bir yalancı yıldız filtreleme algoritması geliştirilmiştir. Bu algoritma bir gözetimsiz öğrenme yöntemi olan yoğunluk

tabanlı kümeleme yöntemine dayanmaktadır. Birbirini izleyen iki görüntü çerçevesi kullanılarak bir fark listesi oluşturulur. Bu fark listesi üzerinde yoğunluk tabanlı kümeleme yöntemi uygulanarak gerçek yıldızların kestirimi yapıldıktan sonra yalancı yıldızlar filtrelenir. Sadece kestirimi yapılan gerçek yıldızları içeren ardışık çerçeveler kullanılarak, bir regresyon analizi prosedürü ile bir ilgin dönüşüm matrisi elde edilir. Ek olarak, ön bilgisiz yıldız tanıma yönteminde karşılaşılan sorunların üstesinden gelmek için özyinelemeli yıldız tanıma yöntemi geliştirilmiştir. Ön bilgisiz yıldız tanıma yönteminden alınan algoritmik yapının yanı sıra, hesaplama karmaşıklığını ve ortalama çalışma süresini azaltmak için veritabanının çok daha küçük bir kısmının kullanılmasını sağlayan bir güncelleme mekanizması özyinelemeli yıldız tanıma algoritmasına entegre edilmiştir. Ayrıca, yalancı yıldız filtrelemesi, yalancı yıldızlara karşı hassasiyetin giderilmesini sağlamıştır. Böylece, tam entegre algoritma sadece doğruluğu artırmakla kalmamış, aynı zamanda hesaplama karmaşıklığını ve ortalama çalışma süresini de azaltmıştır. Önerilen algoritmaların performansı, yine bu çalışma kapsamında geliştirilen bir simülasyon ortamında değerlendirilmiştir. Simülasyon ortamı, verilen sensör parametreleri ve veritabanı bilgileri ile görüntü çerçevelerinin oluşturulmasına izin vermektedir. Ayrıca, tüm gürültü türlerini içeren farklı gürültü senaryolarının gerçekleştirilmesi de bu simülasyon ortamında mümkündür. Kamera hareketi rastgele veya eğilimli iterasyonlarla simüle edilebilmektedir. Algoritmaların farklı adımlarında kullanılacak en uygun parametreleri belirlemek için bir dizi deney gerçekleştirilmiştir. Yıldız tanıma algoritmalarında farklı veritabanları kullanılabilir. Bu veritabanları parlaklık eşik aralığı ve görüş alanı gibi sensör parametrelerine ve algoritmaya özgü bir parametre olan örtüşme oranına bağlı olarak farklı özellikler kazanabilir. Bu parametrelerin farklı değerleri için veritabanlarının kestirim sonucuna etkisi incelenmiştir. Sensör parametrelerinden bağımsız olarak örtüşme oranının oldukça belirleyici olduğu ve bu oran arttıkça doğruluk artış gözlenirken hesap karmaşıklığında da artış gözlemlenmiştir. Bunun yanında regülarizasyon parametreleri de uygun şekilde seçilmiştir. Yalancı yıldız filtreleme algoritmasında uygulanan yoğunluk tabanlı kümeleme yönteminin parametrelerinin seçilmesi için ayrı deney setleri gerçeklenmiş ve filtreleme algoritmasına uygun parametre değerleri dinamik olacak şekilde belirlenmiştir. Özyinelemeli yıldız tanıma algoritmasında kullanılan parametrelerin kestirim sonucuna olan etkileri incelenmiştir. Bu parametrelerden biri olan yeniden oluşturulmuş gözlem öznitelik vektörünün kestirim sonucuna etkisi farklı gürültü koşulları altında gerçekleştirilen simülasyon deneylerinde incelenmiştir. Buna göre bu algoritmada kullanılacak güncelleme mekanizmasında bu hata parametresinin bir eşik değerine göre durumuna göre bir tetikleme modelinin geliştirilmesi uygun görülmüştür. Bunun yanında veritabanının küçültülmesi amacıyla kamera hareket kestiriminden alınan sonuçların mesafe hatalarına göre doğruluk ve tutarlılık ayrıntılı bir şekilde incelenmiştir. Buna göre küçültülmüş veritabanının konumu ve büyüklüğü bu kestirim sonuçlarından faydalanılarak uygun şekilde dinamik olarak belirlenmiştir. En uygun parametreler belirlendikten sonra algoritma performanslarını değerlendirmek için çok sayıda deney gerçekleştirilmiştir. Ön bilgisiz yıldız tanıma performansı, hata grafikleri, hassasiyet oranı eğrileri ve tanıma oranı eğrileri aracılığıyla istatistiksel olarak değerlendirilmiştir. Bu simülasyon deneyleri kestirim sonuçlarının eğilim kazanmasının engellenmesi için veritabanının en fazla bölgesinin kullanılması ilkesine göre gerçekleştirilmiştir. Ayrıca, veritabanı boyutu ve ortalama çalışma süresi ölçülerek karmaşıklık analizi yapılmıştır. Daha büyük örtüşme

oranına sahip veritabanları kullanıldığında doğruluğun arttığı ancak karmaşıklıktan ödün verildiği gözlemlenmiştir. Önerilen ön bilgisiz yıldız tanıma algoritması, doğruluk açısından son teknoloji yöntemlerden daha iyi performans gösterirken, veritabanı boyutu ve ortalama çalışma süresi açısından geride kalmaktadır. Optimum parametre seçiminden sonra, yalancı yıldız filtreleme yönteminin performansı karışıklık matrisi ve istatistiksel göstergeler aracılığıyla değerlendirilirken, kamera hareket kestirimi yönteminin performans değerlendirmesi için mesafe hata ölçümü kullanılmıştır. Her iki algoritma da üstün performans göstermektedir. Özyinelemeli yıldız tanıma algoritmasında hata eşiği ve tarama bölgesi için optimum parametreler seçildikten sonra, doğruluk ve karmaşıklık açısından ön bilgisiz yıldız tanıma yöntemiyle performans ve karmaşıklık bakımından karşılaştırılmıştır. Güncelleme mekanizmasının tetiklenme modeli ayrıntılı bir şekilde incelenmiştir. Deneyler, farklı gürültü senaryoları kullanılarak farklı yıldız tanıma ve veritabanı parametre değerleri için tekrarlanmıştır. Hassasiyet oranı eğrileri ile doğruluk analizi yapılırken karmaşıklık analizi ile kullanılan bellek miktarı ve çalışma süresi hesaplanmıştır. Buna göre doğruluktan ödün vermeden bellek kullanımının ve çalışma süresinin önemli ölçüde azaltıldığı sonucuna varılmıştır. Ayrıca, yalancı yıldız hassasiyeti sorunu da çözülmüştür. Bununla birlikte, algoritma hala eksik yıldızlara karşı hassastır. Yalancı yıldız filtreleme algoritmasının eksik yıldızları da tespit edecek şekilde genişletilmesi için çalışmalar devam etmektedir. Algoritma tüm görüş alanı kullanılarak elde edilen özelliklere dayandığından, görüş alanını engelleyen büyük bir nesne eksik yıldızlara neden olmaktadır ve böylece doğruluk önemli ölçüde bozulmaktadır. Dolayısıyla, güneşin yanı sıra ay da görüş alanına girdiğinde önerilen yöntemi bozmaktadır. Önerilen yöntem, görüş alanını daha küçük parçalara bölerek bu sorunun üstesinden gelebilecek şekilde geliştirilmektedir. Son olarak, önerilen yöntemde parametrelere bağlı bir dizi prosedür bulunmaktadır. Bu nedenle, tüm süreç büyük ölçüde parametrelere bağlıdır. Bu durum, prosedürün yeterince dikkatli bir şekilde takip edilmesi halinde esneklik sağlamaktadır. Her bir parametrenin güvenilirlik ve karmaşıklık üzerindeki etkisini araştırmak için birçok müteakip çalışma yapılması planlanmaktadır.

1. INTRODUCTION

The research problem is stated in detail so that the purpose of this thesis is completely comprehended. An extensive literature review is provided for the reader to conceive the proposed methods and follow the given results within the context of the study.

1.1 Problem Statement

This thesis aims to propose a solution for star identification problem. There are studies that propose solutions based on either pattern-based or isomorphism-based structures. The type of the star identification algorithm has impact on the performance including accuracy and robustness to noise. Considering that the performance of a star identification algorithm is also dependent on noise, the proposed algorithm also offers a false star filtering. The LSI algorithm comes up with a novel feature extraction method by handling the whole image frame as a single object rather than conventionally processing each star objects. Besides, the estimation is based on two stages, including coarse estimation and fine estimation, respectively carried out by the procedures of pattern recognition and regularization. On the other hand, the preprocessing scheme aims at false star filtering since the proposed algorithm is very sensitive to false stars. The outputs of the filtering algorithm are also used to estimate camera motion. The outputs of the camera motion estimation are provided to the star identification algorithm so that the RSI algorithm is achieved by integrating an update mechanism into the LSI algorithm. This allows reducing computational complexity without compromising accuracy boosted through false star filtering.

1.2 Purpose of Thesis

This thesis presents an autonomous and adaptive star identification algorithm. The algorithm operates in two modes including the lost-in-space mode and the recursive mode. It can switch between two modes by means of a feedback mechanism. The

LSI algorithm is based on a cascade structure combining a pattern recognition method and a regression method [1]. The pattern recognition method is used to generate a dictionary that is used by the supervening regression method to make the final attitude estimation. The RSI method is also based on the same scheme as the LSI method except that it exploits an additional FSF method [2] developed within the scope of this thesis. The FSF method, in addition to filtering false stars to increase accuracy, provides the RSI algorithm with affine transformation parameters yielded by the CME method, enabling it to work on a much smaller subset of the whole database.

This thesis aims to ensure attitude determination with higher accuracy compared to state-of-the-art methods in real-time implementation by decreasing computational complexity via improvements in star detection and star identification. The preprocessing phase of the algorithm includes denoising and removal of dead pixels and non-stellar objects after detection of stars in order to import the objects that are most likely to be stars to the star identification process, which increases accuracy and lessens computational burden. The star identification algorithm is designed to employ a pair of methods including dictionary-based star matching and motion estimation, thus expected to maintain higher accuracy with less amount of computation. The dictionary-based star matching method is a lost-in-space application whereas the motion estimation requires a priori information yielded by the former method. An integral simulation medium that provides centroid and brightness of the detected stars is designed. The simulation medium allows star detection and star identification by offering the necessary parameters and tools. The input images are to be simulated by using parameters of a specific CCD image sensor including resolution and FoV, by assuming that the FoV of the sensor is arbitrarily directed to the sky. The simulated input images are to be also subject to noise addition, including position noise, magnitude noise and other types of noise. The developed algorithm is to be compared with the state-of-the-art methods in terms of rate of identification, runtime, size of data storage, rate of false positive, resistance against false identification, resistance against missing identification, resistance against position noise, resistance against magnitude noise and computational complexity. The comparison is based on

the performance measured in terms of accuracy, time, computational complexity, and cost.

1.3 Literature Review

The accuracy level of attitude determination in space missions is of critical importance in maintaining the functions of the spacecraft such as communication and power supply, and often in the success of the mission payload. Accordingly, in recent years, in parallel with the developments in spacecraft technology, the number of researches for high-accuracy attitude determination has increased. Among various attitude determination methods, star sensors stand out with their high accuracy due to developments in electronic and optical technologies. With advances in technology and downsizing of technical staff, low-cost cubesats are increasingly used more widely. Similar trends allow star sensors to become available on cubesats. One of the most important components of star tracking systems is the star identification algorithm, which has a high data processing load.

1.3.1 Star sensor as a spacecraft component

Astronomical attitude determination has helped mankind take "giant leaps" throughout history. Many devices have been invented for such use throughout ages, and their accuracy has been improved constantly. The commencement of the space age in the 1960s made astronautical attitude determination gain even more key importance since attitude determination is an essential necessity for space missions. Scientific payloads requiring high accuracy have triggered research for even more accurate and precise attitude determination capabilities. The devices used for attitude determination in spacecraft or satellites include sun sensors, earth horizon sensor, magnetometer, gyroscope and star sensors. Star sensors, of interest to this thesis, basically aim to make attitude determination by matching some images captured by a CCD to the reference stars obtained from a reference star catalog. Star sensors provide the most accurate attitude determination results at any point independent from any celestial body including earth and sun. Also, their capability of autonomous operation makes them

the most convenient attitude determination sensor for far space missions. However, the significant advantages lead to higher complexity and cost.

In recent years, significant advancements have occurred in star sensor technology, particularly concerning automation and technical capabilities. Progress in key components like microprocessors, optical systems, and algorithms has greatly enhanced the precision of attitude determination. Nonetheless, the demand for attitude accuracy has intensified, driven by requirements for improved communication, high-resolution tracking, and other applications [3]. Various sensors can be used to determine a spacecraft's attitude, with star sensors increasingly favored for their unparalleled accuracy. Sun sensors offer advantages such as light weight, low power consumption, and affordability, but they are inactive without sunlight and provide limited accuracy. Horizon sensors, suitable for near-earth satellites, boast ease of analysis and integration with other sensors but suffer from limited accuracy, low-precision reference pointing, and higher costs. Magnetometers present a low-cost, low-power option for low-altitude Earth orbits, yet they are hampered by limitations in magnetic field modeling, resulting in reduced accuracy and imprecise reference pointing. Gyroscopes excel in providing highly accurate attitude data for short periods and are spatially independent, but they are expensive, necessitate another sensor for initial calibration, and introduce error accumulation over time. Star sensors, though complex and costly, offer superior accuracy and can function independently without requiring additional space or auxiliary sensors [4].

An increase in use of CubeSats for space missions has been increasing the level of space accessibility. A demand of high accuracy settles attitude determination and control systems, which CubeSats rely on for a successful mission, into a vital role. The star sensor is a component of attitude determination for a spacecraft. Its capability of high accuracy makes it a prominent choice in attitude determination as the space missions require higher and higher accuracy in the long run. Attitude determination comprises the steps including star detection and centroiding, star identification and attitude determination. Star identification is the key process within the implementation of attitude determination, which is simply the process of identification of stars within a given frame of observation captured by the star sensor with reference to a star

catalogue, thus transforming an observed body vector into a vector defined in an inertial frame, ready for use by the spacecraft since the Newtonian dynamics are applicable in an inertial frame. In advance of star identification, a chosen method is employed to detect stars and implement centroiding, which is required to satisfy high accuracy as it provides the successive star identification algorithm with information of star body vectors. Accuracy may vary with regards to the properties of the star sensor such as FoV, resolution and saturation sensitivity. ADCS is vital for a spacecraft, which has led to development of many devices and algorithms to obtain this information, namely attitude determination. Besides many devices and methods that help determine attitude, star sensors offer the most accurate estimation of attitude [3] thanks to the sophisticated attitude determination algorithms they rely on, which are, in particular, utilized not only for attitude determination, but also for other spacecraft-related tasks including estimation of angular velocity [5], space surveillance through estimation of orbits of nearby objects [6,7], space navigation in interplanetary trajectories [8] and positioning on an extraterrestrial celestial body [9,10]. The pointing accuracy of some of the state-of-the-art ADCS systems are listed in a study aiming to investigate different CubeSat missions in this regard [11], in which the pointing accuracy is given as 0.4°, 0.008°, 0.2°, 0.002° and 0.013° respectively for the ADCS models TUD, iADCS-100, CubeADCS, XACT and MAI-400. Furthermore, the pointing accuracy is shown to be 0.1° for ARCUS ADC, 1° for iADCS-200 and -400 and CubeADCS 3-Axis Small, Medium and Large, 0.007° for XACT-15, -50 and -100 and Flexcore, 0.5° for CubeADCS 3-Axis Small, Medium and Large with Star Tracker, and 5° for CubeADCS Y-Momentum in the given report [12].

The European Space Agency's published standards offer guidance for both currently employed and in-progress star sensor systems [13]. This comprehensive manual establishes terminology and technical specifications, delineating the performance and specific aspects relevant to star sensor functionality. It serves as a structured and systematic reference for industry practitioners. Contained within this standards manual are performance specifications and evaluation criteria for star sensor components, capabilities, types, reference axis set, environmental factors (e.g., temperature, radiation), lighting conditions (dynamic and diffuse), and standard functional

interfaces. The model used to characterize star sensor capabilities comprises three primary units: an image capture unit, a detection unit, and a data processing unit. The image capture unit gathers photons from celestial objects within the star sensor's FoV, directing them to the detection unit. Here, photons are converted into electrical signals, subsequently transmitted to the data processing unit for attitude estimation [14]. Refer to Figure 1.1 for a visual depiction of these units.

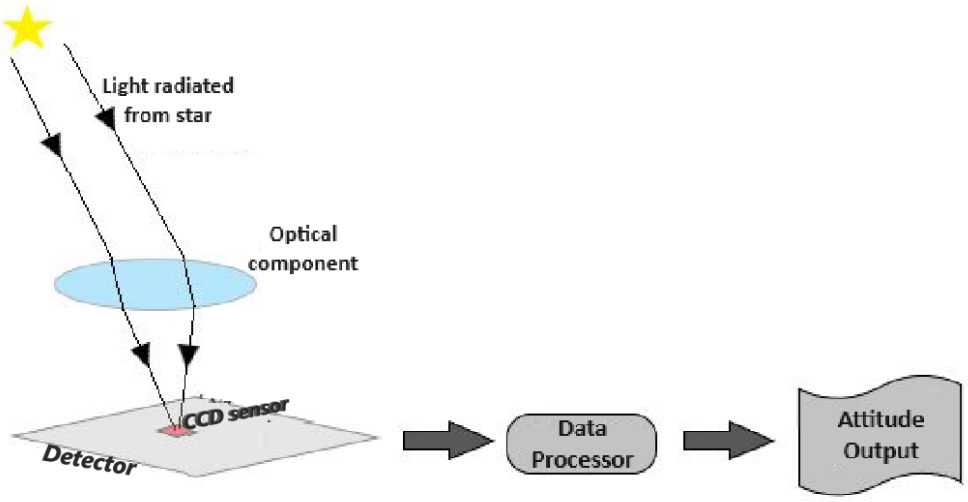

Figure 1.1 : Star sensor model.

As per the standards guidance [13], simulations serve as efficient means to verify performance metrics. Through a series of simulations, results can be statistically analyzed to estimate performance. However, due to the limited nature of simulation sets, the accuracy of performance estimates depends on the number of simulations conducted. Performance conditions for the statistical set should be established considering worst-case scenarios, including plate temperature, radiation flux, and exposure to various light sources such as solar, lunar, terrestrial, and planetary within specified ranges. Maximum angular velocity and acceleration magnitudes should be utilized, with worst-case projections defined accordingly in the direction reference axis set. Simulations should demonstrate the impact of individual star errors on overall accuracy. Based on validation and overall performance criteria, probabilities of correct, incorrect, and invalid attitude determinations should be determined for autonomous attitude determination, accounting for all possible initial attitudes within a defined region of the sky. An autonomous star sensor is expected to maintain operation, even

with reduced performance, under non-nominal conditions, such as when non-stellar objects such as the moon, planets, artificial satellites or cometary dust enter its FoV. In addition, tracking is expected to be maintained in the event of a solar flare. High-energy protons emitted during a solar flare interact with the pixel imaging field. Each proton can affect hundreds of pixels, leaving an electron trail at various signal levels. This effect can be precisely quantified if the direction and energy of the incoming particles and the sensitivity of the pixels are known. The orbital and space radiation model used and the requirements for its use must be provided to determine the level of perturbation. The orbital model is important because the solar flare flux scales with the square of the distance to the sun.

1.3.2 Star identification algorithms

A star sensor processes an observation captured by a camera onboard the spacecraft to match it with a catalog after being converted to an expression of body vector, thereby estimating an inertial vector pointed by the FoV of the camera. This process is called autonomous star identification, which helps determine attitude of a spacecraft. A study [14] splits star identification algorithms into two distinct categories, including LSI algorithms and RSI algorithms. While the latter technique is dependent on a priori information regarding attitude, the former technique requires no such information.

Star identification is, in fact, a subprocess of attitude determination carried out by use of a star sensor, comprising image acquisition and filtering, star identification and attitude estimation and filtering [3]. This scheme is illustrated in Figure 1.2. More clearly, star identification is an intermediary step between both end of the attitude determination, which intakes body vectors and yields inertial vectors. Plus, it also involves its peculiar steps including feature extraction, database search and error checking. The step of feature extraction intakes body vectors of the whole or partial set of stars and derives their selected features by using their specific properties such as brightness and position to output feature vectors. The step of database search intakes features extracted in the previous step to output the possible matches with the catalog in a given probabilistic manner depending on the algorithm. The step of error checking intakes possible matches to output inertial vectors. The complicated

mathematical nature of the star identification process requires more time and energy. Time and energy are of great importance for a space mission besides high accuracy. Thus, it is crucial that star sensors provide attitude determination results within the shortest time interval by consuming the least amount of energy without compromising on accuracy to accomplish space missions which require a huge amount of financial and labor investment. This thesis aims to develop a star identification algorithm that allows the operation to be completed within a shorter time without compromising on accuracy thanks to improvements and innovative approaches. This would allow the ADCS of a spacecraft to operate real-time with high accuracy when equipped with a star sensor using the algorithm developed. The parametric nature of the algorithm, as well, allows the features including resolution and FoV to be expressed in the process to make it customizable to any CCD image sensor, which sustains flexibility and cost-effectiveness.

The limited electrical power sources of a spacecraft, which is the case particularly in small satellites including cubesats that are used for several purposes such as education, Earth remote sensing, science and defense [15]. Besides, the limited amount of time required to make an estimation accurate enough to meet the given criteria and constraints compel researchers to develop star identification algorithms with less computational complexity without compromising accuracy which is vital for telecommunications [16], energy harvesting [17] and payload functions [18,19] in most cases. Considering that small satellites, particularly as they have been evolving into cubesats, are employed in various superior-technology-demanding missions including interplanetary missions [20]–[23], on-Earth-orbit [24]–[26] missions and interstellar missions [27] because of their capability of low-cost production in a short period of time, star sensors are gaining importance to acquire higher accuracy with lower power consumption and less weight.

The sub-processes of the star identification algorithm is illustrated in Figure 1.2. The inputs to the star identification algorithm can include the object vector, accuracy, and brightness of the stars detected in the image. These inputs are generated and transmitted by the star detection and centroiding algorithm, which identifies stars in the image and pinpoints their positions with sub-pixel accuracy. Absolute brightness

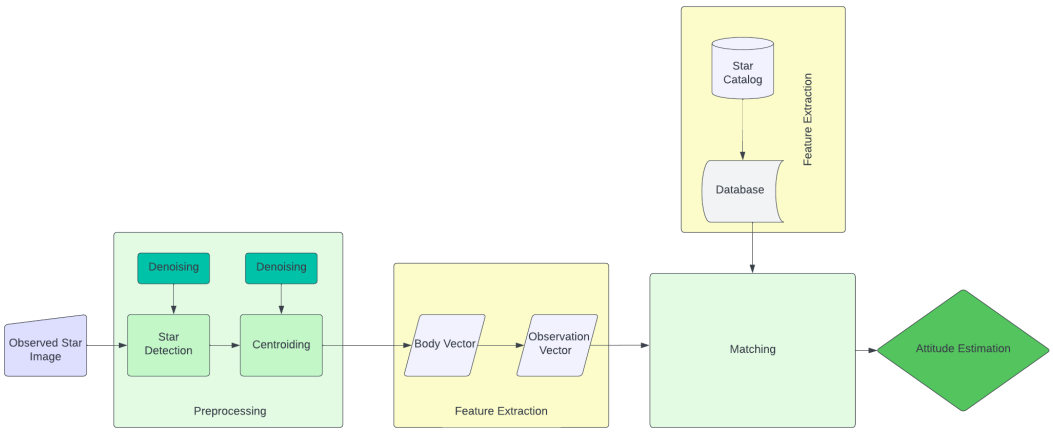

Figure 1.2 : Basic process of star identification.

measurements can be misleading due to noise in the input image, so using the relative brightness of stars in relation to each other yields more accurate results. In the star identification algorithm, position and brightness information are used to extract features. These features are then matched with the star tracker database using the relevant algorithm. The match with the lowest solution error probability is accepted as correct, and the reference inertial vectors of the stars in the image are determined. If no match is found, the identification is considered invalid. A false positive, where an incorrect match is accepted as valid, can severely impair the spacecraft's operational capability. Therefore, it is preferable to declare the identification invalid rather than risk a false positive. Once the star identification process is completed without prior knowledge, the system can transition to recursive identification methods. These methods use the previously obtained attitude estimation for faster processing. An estimator, such as the Davenport q-method [28,29], QUEST [30,31], FOAM [32], ESOQ [33,34], or SVD [35], is then used to determine the rotation angle between the reference inertial vectors from the star identification algorithm and the body vectors [36].

A star catalog is integrated into the star sensor's database prior to its usage onboard the spacecraft's ADCS. It is crucial to select a guide star catalog suitable for the mission. Stars within the catalog are encoded as vectors by parsing the attributes essential for algorithmic use. During the mission, the center of mass of the captured star image, acquired through the optical elements and the CCD detector, is measured

with sub-pixel precision. Subsequent to this, the star identification process, based on matching feature vectors extracted from test images with those in the catalog, can be implemented. This can be achieved through improved triangle algorithms utilizing angular distances, triangle patterns, radial and cyclic star patterns, log-polar transformation, or artificial neural networks [3].

In a study [14], star identification algorithms were categorized into two groups. The first group comprises lost-in-space algorithms that operate without prior knowledge, while the second group consists of recursive algorithms requiring prior information but offering enhanced efficiency. The feature extraction phase of the star identification algorithm is further subdivided into two categories: subgraph isomorphism-based feature extraction and pattern-related feature extraction [37]. In the former, stars are treated as vertices of a subgraph, with angular distances between stars defining edge weights. The stars are identified when the corresponding subgraph is matched from the database. Algorithms employing this approach include the polygon angular matching algorithm [38], the triangle algorithm [39], the color matching algorithm [40], the group matching algorithm [41], and the pyramid algorithm [42]. In the latter category, each star is associated with a neighboring star based on its relative position, and the closest match to the measured pattern is sought in the pattern database. These algorithms encompass the grid algorithm [43], SVD method [44], Log-Polar transformation algorithm [45], hidden Markov model-based algorithm [46], genetic algorithm-based identification algorithm [47], K-L transform algorithm [48], ordered set of points algorithm [49], labeling technique algorithm [50], and star identification algorithms based on Euclidean distance transform, Voronoi mosaics, and k-nearest neighbor classification [51].

A crucial component of the star identification algorithm is its integrated database, which includes star attributes necessary for identification. This database is sourced from established catalogs such as Hipparcos [52] and the Bright Star Catalog [53], containing the positions and brightness of stars across the galaxy. Parameters derived from this data contribute to star identification, emphasizing the importance of establishing a database that allows for rapid and efficient access to these parameters. During database generation, consideration must be given to the characteristics of the

star detection sensor. The sensor's capability to detect stars above a certain brightness threshold dictates that only stars surpassing this threshold should be included in the algorithm, minimizing processing load [37]. The distribution of stars across the sky is non-uniform, varying greatly within different regions of the sensor's FoV. To ensure uninterrupted attitude determination, the database must guarantee the detection of a minimum number of stars in any sky orientation within the sensor's FoV, enabling the algorithm to function effectively. The size of the database is closely linked to the sensor's FoV. While there's no strict requirement for the database size, considerations of scanning methods and star patterns guide its generation to ensure optimal functionality.

The computational complexity of star identification algorithms is mostly determined by the number of stars in the star catalog used, the average number of stars captured in the FoV of the camera and the number of stars in a pattern depending on the type of algorithm. The first star sensor based on a CCD was developed by [54], after which many researchers were involved in studies focused on development and implementation of algorithms that would be able to make real-time star identification. A couple of years later, an algorithm was developed based on patterns derived from angular features of stars in a database selected from a given catalog, which could successfully identify star triplets despite its limitation to perform in real-time due to the need for a priori information of attitude and despite its poor rate of estimation update lagging far behind requirements [55]. In the following years, more researchers were focused on the subject to develop faster algorithms by reducing computational complexity, as based on sorting of sides of triangles formed by combining angular separation of stars [56], area of such triangles and sum of star luminous magnitudes [57] as well as proposal of a permutation matrix addressing the order of star triplets and choice of pattern parameters independent of the selection order of stars [58]. Next, a morphological approach was employed to implement a lost-in-space solution to the problem in real-time, which makes use of angular separation between a pivot star and two closest stars to it and the angle between two lines corresponding to those angular separations and neglects luminous magnitudes, and the computational complexity of which is constrained by both the number of stars in the FoV and the

number of stars in patterns in the feature extraction process with a possibly reduced size of database [39]. Later on, other studies tried to improve this morphological approach. Some of them took into account luminous magnitude of stars by also integrating more complicated patterns of angular separation and dynamic listing of features [59], a sequential filtering algorithm [60], while some focused on database search problem by implementing binary search tree [61], search-less algorithm [62], k-vector search [63], pyramid algorithm [42] and bit-by-bit linear database search [64]. Besides, additional algorithms were developed by means of improvements in the phases of feature extraction and database search including a grid algorithm based on star pattern recognition [43], use of neural with a very fast feature extraction process despite the computational burden due to massive amount of parallel layers networks [65,66]. On the other hand, another star identification approach robust to calibration errors was developed [67], which used the angles between the lines of angular separation within a triangle formed by star triplets instead, in order to eliminate variations due to change in temperature, making it appropriate to use to recalibrate camera. Similarly, another study published an approach robust to errors occurring in CMOS active pixel sensors by using only one of angles within a triangle formed by a star triplet [68]. Furthermore, an RSI method [69] was implemented to reduce process time by means of either k-vector method or star neighborhood approach.

One study [70] introduced an LSI algorithm combining an algorithm using extended images with combined images for feature extraction and group catalog for catalog search by claiming them to be more effective for robustness to various errors, where HIPPARCOS/TYCHO star catalog [71] is used to compare star neighborhood approach [69] and group catalog approach [72]. Another study regarded LSI algorithms to be the most critical component of a star sensor system and surveyed several influential works in the field [73]. The phase of feature extraction was categorized under two definitions including subgraph isomorphism and pattern association [43]. In the case of subgraph isomorphism, an isomorphic subgraph obtained from an observation must be matched with the database within a given tolerance of similarity. This type of feature extraction has been used by the polygon angular matching algorithm [38], the triangle algorithm [39], the color matching

algorithm [40], the group matching algorithm [41] and the pyramid algorithm [42]. The other case involves estimation of the closest match between the pattern obtained from an observation and the catalog based on relative positioning to neighboring stars, which has been used by the grid algorithm [43], the SVD method [44], the log-polar transform algorithm [45], the hidden-Markov model based algorithm [46], the genetic algorithm based identification [74], K-L transformation algorithm [48], the ordered set of points algorithm [49], the labeling technique [50] and the voronoi tessellation and k-nearest classification based algorithm [51].

The SVD algorithm is based on determination of attitude directly without need of additional steps [44]. This algorithm iteratively performs pattern recognition and extracts pattern-related features. The process begins by selecting the brightest star as the reference direction vector, then extracting vectors from four stars within the FoV. These vectors are parsed, and the obtained singular values are compared to those in the database. If a match is not found, a different star is chosen as the reference direction vector. Once a match is successful, the attitude is determined. The method of oriented SVD of triangles [75] is compared to the grid algorithm [43] and the original SVD method [44]. The method of oriented SVD of triangles demonstrated superior performance in handling brightness and position noise, although the original SVD method is noted for its faster processing speed.

The grid algorithm performs feature extraction for pattern recognition [43]. This algorithm has weaknesses in handling brightness and position noise. To address these issues, significant improvements were made in another study [76]. The template matching part of the grid algorithm, which reduced its robustness to brightness and position noise, is removed. Instead, a cost function is introduced that considers the difference between the measured pattern and the database pattern, using the relative brightness values of the stars as weights. This allows the use of grayscale signals instead of binary signals, enhancing robustness. Compared to the original grid algorithm [43], this improved version demonstrated a higher recognition rate in the presence of brightness and position noise and false stars. However, the increased computational complexity due to the higher number of stars has slowed the algorithm's progress. Further optimizations have been made in the improved grid algorithm [77].

Radiometric clusters based on the relative brightness of the stars are generated to reduce the probability of incorrectly selecting the pivot star. Matching classification errors are minimized by retaining only the two highest-scoring matches for each direction and ignoring others. The optimal threshold for rejecting incorrect matches is calculated using Bayesian decision theory. Although the computational complexity increases due to the number of pivot stars, the robustness to position noise and non-catalog stars is significantly enhanced.

In the log-polar transform-based star identification algorithm [45], pattern-based feature extraction is employed. Initially, star patterns are converted from Cartesian coordinates to polar coordinates. The log-polar transformation generates features that are invariant to rotation and scale. This is achieved by translating the star image to position the guide star at the origin. The coordinates produced by the log-polar transformation are stored in a sparse matrix. By projecting the columns of this matrix, a vector is obtained, with a size equal to the number of rows in the matrix. Similar vectors are derived from the stars in the database in the same manner. This allows patterns to be encoded as sequences, and matches can be searched using these sparse vectors. When compared to the grid algorithm [43], the log-polar transform algorithm demonstrated superior performance in handling brightness and position noise. However, the high computational complexity associated with sequence matching reduces the algorithm's speed.

In the adaptive ant colony algorithm [78], in accordance with pattern-based feature extraction, circles are drawn around the star that has brightness close to the average value of the stars in an image. The angular distance between all star pairs within this circle is then calculated. Using these angular distances, the shortest path through all the stars is determined. Similarly, the shortest paths within the circles are identified in the database. The binary scanning method continues until a circle is found that matches the shortest path within the circle in the test image. However, the presence of false stars can severely distort the feature pattern.

In the image-based identification algorithm [79], the sensor image is compared to database images by maximizing an objective cost function. Subsequently, a new image

is created from these coordinates. Similar to the grid algorithm [43], the image is rotated around a pivot star, with the star closest to the pivot star on the horizontal axis. Each star is modeled as a Gaussian probability distribution, with variance depending on system performance and star brightness. The correlation between images is calculated by multiplying the sensor data image with the database images. The correlation value, which indicates similarity, is directly proportional to this product. The cost calculation, which involves cross-correlation between two functions, is made using the Fourier transform. This process is repeated for all pivot stars in the database. Another image-based star identification algorithm [80] uses an image processing technique known as the shortest distance transform. A binary image is converted into a color map where each pixel is colored according to the nearest star. To ensure accurate comparison, the sensor image and database images are of equal size, meaning their FoVs match. Since sensor images may be rotated and translated relative to the database images, similar to the grid algorithm, the sensor image is adjusted to align with the database. In this method, the database image is translated to align the center of mass of a set number of the brightest stars. The brightest stars' centers of mass are then rotated and translated to match those in the database, using the smallest angle of rotation. This is done for all images in the database. During the matching phase with the test image, a vector containing the distances between stars in the image and those in the database is extracted. The decision phase uses the sum of these distances and the number of database stars within two pixels of the image stars to determine similarity. To streamline the comparison, a threshold is set on distance and angle attributes to exclude a significant portion of database images. This algorithm is reported to be highly robust against position noise and false stars.

In the group matching algorithm [41], after selection of a pole star, pairs are formed with neighboring stars. This method is based on subgraph isomorphism for feature extraction, and uses angular distances to find matching sets of star pairs. However, it lacks robustness against false stars and requires high storage capacity. To address these issues, the pole star algorithm [81] is developed, which combines a subgraph isomorphism-based approach in the matching phase with pattern-based features to identify matching candidates. Once a reference star is selected, angular distances of

all stars located between a lower and upper threshold radius are calculated. Rings are obtained by quantizing these distances into a set number of circles. A binary vector, indicating the presence or absence of stars in these rings, is created. The pattern vectors are merged with all selected reference stars to form a database indexed by binary vector positions. Each row in the index corresponds to stars matching the generated pattern, and the count of positively indexed rows for each star is incremented. When this count exceeds a threshold, the star is identified as a candidate. Subsequent iterations generate pairs, triangles, and polygons from the candidate stars using subgraph isomorphism feature extraction until a match is found. This algorithm outperforms Liebe's triangle algorithm [39] and the grid algorithm [43] in terms of robustness to brightness and position noise, though the triangle algorithm is faster. A similar algorithm [64] uses radial and cyclic attributes, expressing radial attribute patterns in vectors and creating a database similar to the pole star algorithm. Instead of subgraph isomorphism, pattern-based feature extraction is used. A bit pattern is generated from cyclic vectors based on angular distances between the pole star and two other stars. Radial-based matching is performed for each star, comparing cyclic patterns to candidate stars. This method, like the original grid algorithm, is sensitive to brightness and position noise when selecting the initial reference star. Another algorithm [82] incorporates a voting mechanism in three steps: single matching, iterative scanning, and verification. During single matching, the angular distance between a pole star and its neighbor is calculated, and star pairs within proximity to this distance are searched in the database. The voting score for both stars in each measured pair increases, and when the count exceeds a threshold, the database star is considered matched. This often results in multiple star matches, which are iteratively reduced by increasing the threshold proportionally. In the validation phase, more than four star matches are required. The multi-poles algorithm [83] uses subgraph isomorphism for feature extraction, matching angular distances between the pole star and its neighbors to database distances using the k-vector technique [63]. The pole star and its neighbors are selected from the most common stars among candidate matches. In the two-step verification, clusters are compared, and distances checked against the database to eliminate false stars. Lastly, the dynamic cyclic pattern matching algorithm [75] enhances robustness to false stars

during pole star selection. Pattern vectors are obtained using discrete center angles and compared with database vectors to produce a similarity score. The false star effect is mitigated through a validation step similar to the multi-poles algorithm.

Deep learning-based algorithms currently face challenges with hardware complexity due to their dense parallel processing requirements, making them less feasible for star identification tasks. However, advancements in processor technology enables the use of deep learning in space applications. RPNet [84] is a representative learning-based star identification network that uses pattern-based features to convert the angular distances between a guide star and its neighboring stars into discrete values, forming the input set. This pattern is generated in a multidimensional space using an encoder-decoder structure. The encoder and decoder are trained with artificial star images. Once the encoder is trained, the classifier is trained using the encoder's outputs. Compared to the grid algorithm [43], RPNet excels in handling false stars and brightness and position noise. It also boasts a fast analytical performance despite a large FoV. However, it does not reduce the storage space required for the database, and the neural network itself has a substantial memory footprint.

In this work, a novel dictionary-based star identification method is proposed [1]. The method is based on a star identification algorithm which blends pattern recognition and regularization using an unconventional set of feature vectors. Accordingly, a set of feature vectors is created to construct a database and obtain observation feature vectors. Unconventionally, feature vectors are based on neither graphical nor morphological features, but rather a single object is constructed for each observation frame comprising several features. Thereby, a cumulative approach is developed, by which the computational complexity could be reduced significantly thanks to less amount of feature vectors. A database is constructed using the given process of feature extraction, which is used in a dictionary-based matching method. The database feature vector, which has the highest similarity with the observation feature vector, is detected by means of the 1NN classifier, a simpler version of kNN classifier. The 1NN classifier, which was initially proposed in early studies [85,86], has been used and improved in several star identification algorithms [43,49,51,87]. The similarity is measured using Euclidean distances between the observation feature vector and database feature

vectors. Within the 1NN classifier, the binary search method [88], which has also been used and improved in several star identification algorithms [61,66,75,78], is used to detect database feature vector with the highest similarity corresponding to the smallest norm value of difference with the observation vector. Subsequently, a dictionary is constructed using the most similar database feature vector as well as its neighboring database feature vectors. Next, a process of regression analysis is executed in the regularization process by means of variable selection and regularization, through which a solution vector is constructed by linear combination of the chosen dictionary feature vectors multiplied by a solution coefficient vector, thus fitted to the observation. This process is specifically implemented by means of the LASSO regression [89]. This regression method has been used in a study that make use of a dictionary retrieved by a scheme of hybrid mechanism blending particle filtering and deep learning [90,91] by using another study as a baseline method [92]. Because of the unorthodox structure of the algorithm, the solution vector could yield both the direction of the boresight vector and the rotation about it. The simulated experiments show the effectiveness of the algorithm and its robustness against types of noise regarding position and luminous magnitude, but high sensitivity to missing stars and false stars. The complexity of the proposed algorithm is empirically indicated in terms of database size and average run time for different values of the overlap ratio that is a parameter specifying the structure of database. The identification rates in the presence of noise are compared to some state-of-the-art methods including SGS [93], IG [94], IA [82], PA [42], MG [95], GA [43] and GV [96]. The identification rate data is retrieved from the studies [93,94] for the competing methods. The computational complexity of the proposed algorithm is calculated. Its average run time and database size are compared to the state-of-the-art methods including PA [42], MGA [97], LPT [45], VP [98], GA [43], SVD [44] and OSV [75]. The information regarding database size and average run time is retrieved from the studies [75,98] for the competing methods.

1.3.3 False star filtering

Star identification includes image preprocessing, feature extraction and matching. Of them, image preprocessing involves two steps including noise removal and centroiding.

Mostly, the center of gravity method yields the fastest results for centroiding despite a reduction in accuracy due to sensitiveness to noise [99]. Noise is, mostly, removed by linear filtering, median filtering, morphological filtering or etc [3]. The types of input noise, to which a star sensor is subject, include position noise, missing and false stars as well as brightness noise [73]. Position noise may emerge due to thermal deformation, optical flaws or calibration errors, while magnitude noise is incurred by the sensitivity of the sensor. Interfering stars include false stars and missing stars respectively caused by reflecting objects in the FoV and solar flare and blockages in the FoV and dead pixels. Recent studies analyzed the performance of the proposed star identification methods for their robustness against false star noise [93,100,101], or simulators were developed to make noise injection including false stars [102]. Lastly, a study proposed an FSF algorithm to be used as a preprocessing algorithm for existing star identification algorithms, which utilizes the difference between motion of objects in the FoV by implementing angular distance tracking and star voting on multiple consecutive frames [103].

The methods used by a star sensor should ensure improvements in cost, quality, success and lifetime of a space mission by saving mass and electric power thanks to higher accuracy and less complexity that help decrease storage and memory space. These improvements may be achieved in either star detection and centroiding or star identification. Real implementations inevitably have to compensate the effects of input noise on the body vector [73]. The process of removal of noise out of an image captured by a star sensor is regarded as a type of preprocessing improvement. Furthermore, as to the standards of the European Space Agency, a star sensor is required to resume operating in the presence of non-star objects in the FoV or solar flare which induces high energy protons to interact with the imaging field, which are considered sources of noise. The sources of noise involve false stars due to reflecting objects and solar flare, shifted stars due to thermal deformation or optical flaws and missing stars due to blockage of FoV and dead pixels [84]. These phenomena pose a threat against achievement of high accuracy; thus, they are required to be removed where possible. False stars are stimulated by reflecting objects such as other satellites or dust and solar flare while missing stars due to blockage of the FoV and dead pixels.

False stars can be classified into three types: transient, stationary and drifting false stars. False stars are introduced by radiation impinging particles due to a large number of electrons and protons hitting the imaging sensor particularly in strong radiation zones [102] in case of transient false stars, reflections of sunlight from faraway space objects in case of stationary false stars, and appearance of nearby objects or particles on the image plane as called plume interference [103]. There have been some algorithms to cope with these types of false stars.

Star sensors may use some methods of denoising as a part of preprocessing in advance to star identification, and alternatively, some star identification methods are capable of overcoming noise. The occurrence of noise on star images may decrease accuracy of star identification algorithm and pose a risk for a space mission. Thus, denoising methods can be used to remove the effects of noise. FSF is a process that aims to detect and remove false stars from the captured image. As to the standards of the European Space Agency, a star sensor is required to resume operating in the presence of non-star objects in the FoV or solar flare which induces high energy protons to interact with the imaging field, which are considered sources of noise [13]. Considering the different types of false stars, many algorithms have been developed to filter them. Transient false stars can be distinguished by their characteristic position randomness in the image plane stimulated by strong radiation [104]. There are algorithms that achieve to filter transient stars using such characteristics in two time-sequential frames [105]. Stationary false stars can be successfully filtered by existing algorithms when the number of false stars is much less than true stars [42,76] while other algorithms are more effective when the number of false stars increases further [83,106]. On the other hand, although the methods used to filter transient false stars fail to drifting false stars, there are studies focusing on this issue [103]. Moreover, many star identification algorithms [80,96,107] have reported the algorithm performance in terms of robustness against false stars as well as other types of noise. Lately, a study proposed a star identification method based on spectral graph matching, in which a feature called the neighbor graph is constructed, thus making star identification through similarity of features, also by inserting types of noise including position, missing star and false star [101]. Plus, another study,

asserting that derivation of reproducible results and comparison of algorithms are difficult tasks due to nontrivial structure of star sensors, proposed to use simulated star images by aiming to accelerate the development time through a standard verification methodology, where the simulator is capable of inserting specific noise to create real-world conditions [102]. There is another study focusing on the problem of star identification in cases of generic calibrated and non-calibrated cameras, narrow-FoV calibrated and non-calibrated cameras using invariant theory by developing a compact algorithm to simultaneously calibrate camera and compute attitude [108]. Another recent study proposed a subgraph isomorphism-based star identification algorithm that evaluates validity of different subgraphs to achieve a faster process by choosing the simplest general subgraphs by means of voting, building isomorphic subgraph and verification respectively [109]. An additional recent study also proposed a subgraph isomorphism-based strategy, namely multilayer voting algorithm, including an initial match which adopts a triangle unit through a triangle voting scheme and uses singular values of each triangle unit to search for the match, and a verification process used to eliminate incorrect candidates [93].

1.3.4 Performance evaluation

The varying principles and application environments of star identification algorithms make it challenging to directly measure their performance metrics. This has led to inconsistencies in the literature regarding these metrics, necessitating a structured and complementary approach to compare algorithmic performances [73]. To select the most suitable star identification algorithm for a star tracker, a thorough structural comparison is essential. In one study [14], algorithms were assessed based on their analytical asymptotic performance during the feature extraction step, database search, and the use of independent pattern attributes relative to the number of stars in a pattern. However, this approach does not fully capture the system performance of the star tracker, especially in the presence of random noise such as false stars. Analytical performance alone does not accurately reflect real-world conditions. Different algorithms report their performance using various FoV and noise levels, making comparisons difficult. In a study [43], a structural comparison of algorithms using

two different feature extraction methods is conducted. The algorithms' sensitivity is evaluated in different test environments, and their robustness is examined unless simulations reflect peak performance. This approach makes comparisons more informative and reproducible, as it accounts for differences in noise levels. In simulation-based evaluations, a sensitivity analysis of performance criteria should be conducted by varying the test environment [73]. The theoretical accuracy limits of star trackers are estimated in a study [110], based on Earth's position in the galaxy. These physical limits have gained significance as spacecraft components continue to decrease in mass, power consumption, and cost. The accuracy of a star tracker's sensor is influenced by stellar distribution, sensor dimensions, and exposure time. When the sensor diameter and exposure time are sufficiently large, the optimal orientation estimation is achieved through the covariance matrix of the rotation vector error [111].

Table 1.1 : Characteristics of star identification algorithms.

Algorithm	Feature extraction	Type	Complexity
Singular value decomposition [44]	Pattern	Singular value	$\mathcal{O}(n)$
Pole star [81]	Hybrid	Pole star	$\mathcal{O}(bn)$
Modified grid [76]	Pattern	Grid	$\mathcal{O}(bn)$
Radial and cyclic [64]	Pattern	Pole star	$\mathcal{O}(fn)$
Log-polar transform [45]	Pattern	Log-polar transform	$\mathcal{O}(n)$
Adaptive ant colony [78]	Pattern	Adaptive ant colony	$\mathcal{O}(\log(n))$
Correlation approach [79]	Pattern	Image-based	$\mathcal{O}(n)$
Optimal image matching [80]	Pattern	Image-based	$\mathcal{O}(n)$
Iterative [82]	Subgraph	Pole star	$\mathcal{O}\left(b(\Delta mn)^2\right)$
Optimized grid [77]	Pattern	Grid	$\mathcal{O}(\alpha n)$
Multi-poles [83]	Subgraph	Pole star	$\mathcal{O}(k)$
Oriented singular value [75]	Hybrid	Singular value	$\mathcal{O}(n)$
Representation learning-based [84]	Pattern	Deep learning	$\mathcal{O}(1)$
Radial and dynamic cyclic [112]	Pattern	Pole star	$\mathcal{O}(\log(n))$

A list that compares a number of star identification algorithms in terms of computational complexity is given in Table 1.1. According to the list, n, b, f, Δm, α and k respectively refer to number of database stars, number of stars in a pattern, average number of stars in the FoV, reduced number of star pairs in the database fraction, number of pivot stars and number of candidates. Most algorithms lack an optimal scanning approach to enhance speed performance. However, methods like the adaptive ant colony and oriented SVD transformation have significantly improved performance by reducing database scanning time through binary scanning. The multi-poles algorithm achieved better database scanning performance using the k-vector technique, though it compromised speed by employing multiple iterations to enhance reliability. The deep learning method delivered the highest scanning performance by eliminating the scanning phase, making its complexity independent of the problem size. Nonetheless, due to the neural network's layered structure, which occupies considerable memory space, it always produces a result. Therefore, it is crucial to add appropriate validation layers to prevent false positive matches, which are unacceptable for the star tracker.

Table 1.2 : Scoring criteria for star identification algorithms.

Criterion	Definition
Precision rate [113]	Percentage of instances with estimated error is within the given threshold
Identification rate	Ratio of instances correctly identified with respect to a predefined threshold
Run time	Amount of time required to make estimation including all phases
Storage usage	Storage memory size allocated
Position noise	Robustness to average position noise in the image
Brightness noise	Robustness to average brightness noise in the image
False stars	Robustness to average number of false stars in the image
Missing stars	Robustness to average number of false stars in the image
Complexity	The level of complexity of implementation

The simulation test scoring criteria for the star identification algorithm are detailed in Table 1.2. The simulation data used for star identification is derived from an

existing star catalog. For a specified number of image frames, an attitude is created, and all stars within the FoV within a brightness boundary are identified from the catalog. A test image is then generated from this image frame using a camera and sensor model, with noise added through these models. The input elements for the star identification algorithm are position and brightness. To minimize variations during star detection and centroiding, these input elements should remain consistent each time they are used in the star identification algorithm. Key parameters to consider when generating the test data include the noise levels to be added to the images. These noise levels should be realistically modeled with sufficient tolerance to suit the specific application environment. The test simulation environment should account for factors such as platform type, space environment, camera system FoV size and resolution, the criticality of star tracker measurements, sensor speed and accuracy levels. All selected parameters should be clearly specified, and a sensitivity analysis of the parameters and their interactions should be conducted [73].

1.4 Document Outline

This thesis presents a novel star identification algorithm. The algorithm uses an unconventional feature extraction method. Besides, it is capable of switching between two modes, lost-in-space and recursive, by means of an update mechanism. The update mechanism is driven by feedback from the FSF&CME algorithm, also developed as a preprocessing implementation within the scope of this thesis. In Chapter 2, the LSI method is described. After presentation of contributions and limitations as well as preliminaries and assumptions, the problem is stated. The methodology is explained with necessary definitions and propositions, including feature extraction, database generation, and algorithm description. In Chapter 3, subsequent to problem statement, methodology of FSF&CME algorithm is described, including feature extraction, false star filtering and camera motion estimation. Chapter 4 presents the RSI algorithm based on the LSI algorithm. Its adaptive structure supported by the FSF&CME is described after declaration of contributions and limitations. In Chapter 5, the simulation environment is introduced, including integration of parameters to generate database and observation images with noise injected and translated and rotated in

accordance with convenient probabilistic functions. Chapter 6 provides very detailed experimental analyses for the algorithms. The experiments are based simulation implementations carried out for parameter selection, performance evaluation and complexity analysis for all three algorithms. Conclusions are given in the last Chapter 7 in addition to improvements and recommendations.

2. LOST-IN-SPACE STAR IDENTIFICATION METHOD

The proposed LSI algorithm is based on a regularized pattern recognition method, in which unconventional features are used. Feature vectors are based on neither graphical nor morphological features but rather on a single frame object is constructed for each observation frame comprising five features. Thus, a database with less amount of feature vectors can be generated, where each feature vector contains information corresponding to a single image frame with a given FoV. This leads to a significant reduction in the computational complexity. The final estimation is yielded through sequential implementation of pattern recognition and L_1 regularization. Pattern recognition and L_1 regularization are realized by using the 1NN classifier [85,86] and the LASSO regression [89] respectively. The algorithm sequentially makes two estimations. While the first estimation is the coarse estimation made by the 1NN classifier, the second estimation is the fine estimation ensured by the LASSO regression. The database feature vector with the highest similarity to the observation feature vector is detected by the 1NN classifier. The 1NN classifier has been used and improved in several star identification algorithms [49,51,87]. The similarity is measured using Euclidean distances between the observation feature vector and database feature vectors. The 1NN classifier is implemented using the binary search method [88], which has also been used and improved in several star identification algorithms [75,78]. It is simply used to detect the database feature vector with the highest similarity that corresponds to the smallest distance to the observation vector in the feature space. Subsequently, a dictionary is constructed using the most similar database feature vector and its neighboring database feature vectors. Next, the LASSO regression is implemented as the regularization process for fine estimation. Accordingly, a solution vector is constructed by linear combination of the chosen dictionary feature vectors multiplied by a solution coefficient vector so that an optimized solution that fits the observation is estimated by iteration. The solution vector yields both the camera boresight vector and the rotation about it.

The phases of the algorithm are explicitly shown in Figure 2.1. The body vector of the observation is transformed into the observation feature vector in the phase of feature extraction. However, the algorithm is already supplied with the database which contains the database feature vectors extracted from the star catalog using the same feature extraction procedure. The observation feature vector is processed sequentially through the pattern recognition and regularization stages. Finally, an inertial vector solution is estimated, which contains the camera boresight vector and the rotation angle about it.

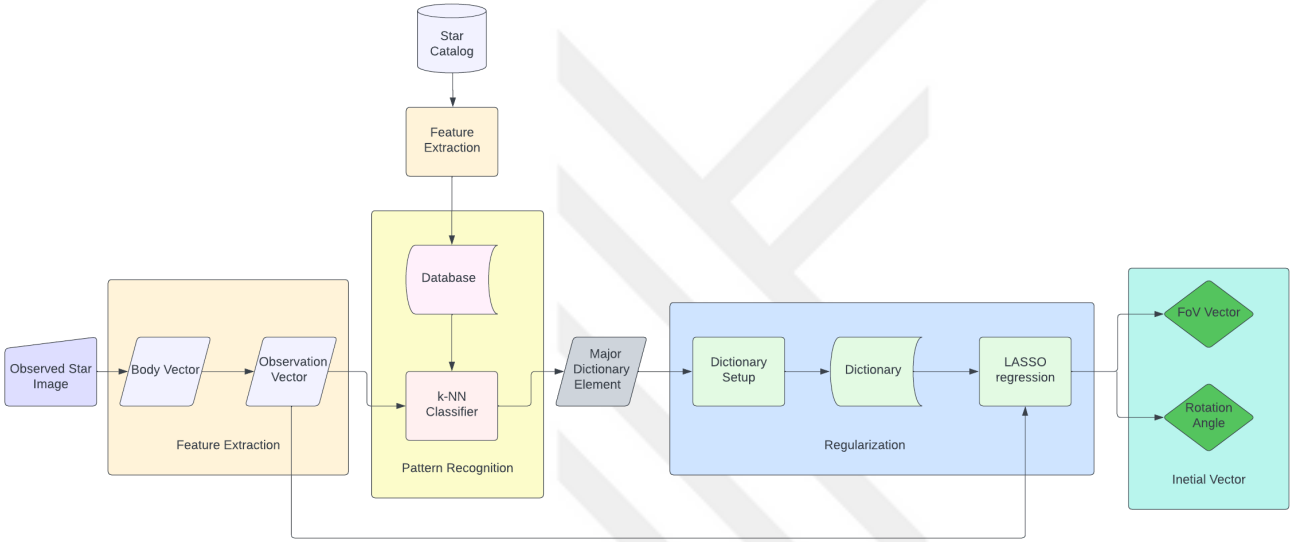

Figure 2.1 : The flowchart of the dictionary-based star matching method.

2.1 Contributions and Limitations

This work presents a novel LSI method so that the issues faced by space missions demanding increasingly higher accuracy with less complexity, in particular for small satellites, are addressed through a diversified approach unlike the previous works based on morphological approach [39,42,59]–[64]. The results promise as high accuracy as the state-of-the-art commercial star trackers while the algorithm maintains acceptable database size and average run time. The first contribution includes maintenance of a small storage unit thanks to a novel approach of feature extraction that ensures a compact database by allocating a unique feature vector for each camera frame, which is called a frame object. Secondly, the computational complexity is reduced by implementing a compact database accompanied by the sequential use of the 1NN classifier via a binary search of Euclidean distances and the LASSO regularization technique. The cascade structure of the algorithm ensures a high accuracy by means of a coarse estimation followed by a fine estimation. Moreover, the algorithm directly yields an estimation of the camera boresight vector and the rotation angle about it, providing a complete spatial attitude information. Lastly, the results produced by the algorithm are independent from rotation of the camera plane about the boresight vector thanks to the feature extraction process bounded within a circular FoV. On the other hand, the proposed LSI method is very sensitive to false stars and missing stars despite very high robustness to position and brightness noise. It requires a larger database allocation in comparison to most of its competitors in return for a higher accuracy.

2.2 Preliminaries and Assumptions

The LSI algorithm relies on pattern recognition and regularization. The methods used for this purpose include 1NN classification and LASSO regression. Before stating the problem, explaining the methodology in detail and providing and interpreting the results, the preliminaries including concepts, variables, functions and constants are introduced so that a neat discourse and a clear understanding are ensured.

The 1NN classifier requires a database of feature vectors. The database is generated using the Hipparcos catalog [71] in this study. This catalog offering high-quality scientific data provides a range of parameters for each star contained. The parameters used in this study include right ascension and declination in degrees with respect to J1991.25 ICRS as well as the visual magnitude and the absolute visual magnitude with respect to the Johnson UBV Photometric System. The absolute visual magnitude is obtained using

$$M_V = V_{mag} + 5\log\pi - 10 \tag{2.1}$$

where V_{mag}, M_V and π represent the visual magnitude, the absolute visual magnitude and the trigonometric parallax respectively. The values of parameters used to extract feature vectors are given in Table 2.1 as provided by the Hipparcos catalog [71]. The absolute visual magnitude is called brightness in this study.

Table 2.1 : Parameters used for feature extraction.

Label	Symbol	Description	Field	Unit
HIPP	-	Identifier	H1	-
V	V_{mag}	Magnitude	H5	mag
RA	α	Right ascension	H8	deg
Dec	δ	Declination	H9	deg
Par	π	Parallax	H11	mas

The database contains feature vectors. Each feature vector is extracted from a single frame extracted from the catalogue. A feature vector corresponds to a frame object. This is a single object obtained from each image frame with specific characteristics of the star sensor, including FoV, resolution and brightness. The frame object contains the spatial gravity center of the stars within the given image frame. Thus, a feature vector consists of five components including the number of stars in the frame, brightness mean, brightness standard deviation, log-polar brightness-weighted positional mean including polar radius and polar angle and polar radius standard deviation. The spatial dimensions are subject to log-polar transformation [45]. The features are derived as follows

$$n \triangleq \mathrm{n}(\mathscr{S}) \tag{2.2}$$

where n is the number of stars in the image frame, and $\mathscr{S}$ is a set that contains the detected objects in the given image frame,

$$A \triangleq \frac{\sum_{i=1}^{n} a_i}{n} \tag{2.3}$$

where A is the brightness mean, and a_i is the brightness of each object i in the set $\mathscr{S}$,

$$r \triangleq \sqrt{X^2 + Y^2} \quad \text{and} \quad \theta \triangleq \arctan\left(Y/X\right) \tag{2.4}$$

where r is the polar radius, θ is the polar angle, as well as X and Y are the frame object's centroid coordinates with respect to the image center, which are obtained by

$$X \triangleq \frac{\sum_{i=1}^{n} a_i \cdot x_i}{n} \quad \text{and} \quad Y \triangleq \frac{\sum_{i=1}^{n} a_i \cdot y_i}{n} \tag{2.5}$$

where x_i and y_i represent the spatial pixel coordinates in the image frame,

$$\sigma_A \triangleq \frac{\sum_{i=1}^{n} \left(a_i - \mu_A\right)^2}{n} \tag{2.6}$$

where σ_A is the brightness standard deviation, and μ_A is the mean of brightness of all objects in the set $\mathscr{S}$, defined such that

$$\mu_A \triangleq \frac{\sum_{i=1}^{n} a_i}{n} \tag{2.7}$$

and finally,

$$\sigma_r \triangleq \frac{\sum_{i=1}^{n} \left(r_i - \mu_r\right)^2}{n} \tag{2.8}$$

where σ_r is the polar radius standard deviation, and μ_r is the mean of polar radii of all objects in the set $\mathscr{S}$, defined such that

$$\mu_r \triangleq \frac{\sum_{i=1}^{n} r_i}{n} \tag{2.9}$$

where r_i is the polar radius of each object in the image frame.

A database feature vector is obtained from the reference catalog in accordance with the characteristics of the sensor used in the simulation. It is defined as

$$\overrightarrow{\tau} \triangleq \begin{bmatrix} n & A & r & \sigma_A & \sigma_r \end{bmatrix}^{\mathsf{T}} \tag{2.10}$$

with the features included. Database feature vectors should contain discriminating features that allows an optimal selection of the most similar database feature vector in

the process of pattern recognition and a successful regularization. However, the big gap between the numeric values of separate features potentially incurs a dominating effect in favor of the features with higher numeric values. Thus, the normalized database feature vectors are introduced so that this dominating effect could be mitigated, which is defined such that

$$\hat{\overrightarrow{\tau}} \triangleq \begin{bmatrix} \hat{n} & \hat{A} & \hat{r} & \hat{\sigma_A} & \hat{\sigma_r} \end{bmatrix}^{\mathsf{T}} \tag{2.11}$$

where the normalized features are calculated with respect to their true minimal and maximal values separately, such that $\hat{n} = \frac{n-\min\{n_i\}}{\max\{n_i\}-\min\{n_i\}}$, $\hat{A} = \frac{A-\min\{A_i\}}{\max\{A_i\}-\min\{A_i\}}$, $\hat{r} = \frac{r-\min\{r_i\}}{\max\{r_i\}-\min\{r_i\}}$, $\hat{\sigma_A} = \frac{\sigma_A-\min\{\sigma_{Ai}\}}{\max\{\sigma_{Ai}\}-\min\{\sigma_{Ai}\}}$ and $\hat{\sigma_r} = \frac{\sigma_r-\min\{\sigma_{ri}\}}{\max\{\sigma_{ri}\}-\min\{\sigma_{ri}\}}$ providing that $i = 1, \cdots, N$, and N is the number of database feature vectors.

On the other hand, an observation feature vector is extracted from each observed image frame. The observation feature vector, denoted by $\overrightarrow{\psi}$, contains the same types of features as a database feature vector. Furthermore, a normalized observation feature vector $\hat{\overrightarrow{\psi}}$ is generated using the minimal and maximal numeric values of the database features. When using the nearest neighbor classifier to detect the most similar normalized database feature vector to the normalized observation feature vector, the measure of distance is calculated by means of the Euclidean distance as follows

$$\| \hat{\overrightarrow{\psi}} - \hat{\overrightarrow{\tau}} \|_2 = \sqrt{\sum_{i=1}^{5} (\hat{\psi}_i - \hat{\tau}_i)^2} \tag{2.12}$$

where L_2 norm is calculated by means of the Euclidean distance between a normalized observation vector $\hat{\overrightarrow{\psi}}$ and a normalized database feature vector $\hat{\overrightarrow{\tau}}$.

The regularization process is implemented by means of the LASSO regression which requires a dictionary so that a better approximation could be achieved for the sake of fine estimation. A dictionary template matrix is a 5×9 matrix containing 9 database feature vectors, which is defined such that

$$\mathbf{T} \triangleq \begin{pmatrix} \overrightarrow{t}_1 & \cdots & \overrightarrow{t}_9 \end{pmatrix} \tag{2.13}$$

where $\overrightarrow{t}_5$ is the most similar database feature vector to the observation feature vector that is detected by the 1NN classifier, and the other vectors $\overrightarrow{t}_j$ are its neighboring vectors. The LASSO regression yields a regularization weight vector $\overrightarrow{\omega}$ that contains 9 elements corresponding to each vector t_j in the dictionary template matrix $\mathbf{T}$.

An estimation of the observation inertial boresight vector $\tilde{\vec{f}}$ is defined as

$$\tilde{\vec{f}} \triangleq \begin{bmatrix} \tilde{\alpha} \\ \tilde{\delta} \end{bmatrix} \tag{2.14}$$

where $\tilde{\alpha}$ and $\tilde{\delta}$ are the estimations of right ascension and declination. The estimated inertial boresight vector is obtained by matrix multiplication of the regularization weight vector $\vec{\omega}$ and the boresight template matrix

$$\tilde{\vec{f}} = \mathbf{F}\ \vec{\omega} \tag{2.15}$$

where $\mathbf{F}$ is the boresight template matrix defined as

$$\mathbf{F} \triangleq \begin{bmatrix} \alpha_1 & \cdots & \alpha_9 \\ \delta_1 & \cdots & \delta_9 \end{bmatrix} \tag{2.16}$$

where α_i and δ_i respectively correspond to the values right ascension and declination of the dictionary feature vectors. In addition, an estimation of rotation angle about the estimated inertial boresight vector can also be obtained through the following matrix multiplication

$$\tilde{\theta} = \Theta\ \vec{\omega} \tag{2.17}$$

where the estimated rotation angle is expressed with respect to the linear curve $\delta = 0^\circ$ in the inertial frame, and the rotation template matrix is defined as

$$\Theta \triangleq \begin{bmatrix} \theta_1 & \cdots & \theta_9 \end{bmatrix} \tag{2.18}$$

where θ_i denotes the rotation angle corresponding to the relevant dictionary feature vector. The reconstruction of the observation feature vector can also be made by the following matrix multiplication

$$\tilde{\vec{\psi}} = \mathbf{T}\ \vec{\omega} \tag{2.19}$$

where $\tilde{\vec{\psi}}$ is the reconstructed observation feature vector.

Three types of error indicators are defined and used. The first one is the error for the estimated observation feature vector ε_ψ defined as the norm of the difference vector between the reconstructed observation feature vector and the observation feature vector as follows

$$\varepsilon_\psi = \| \vec{\psi} - \tilde{\vec{\psi}} \|_2 \tag{2.20}$$

where $\vec{\psi}$ and $\tilde{\vec{\psi}}$ are the observation feature vector and the reconstructed observation feature vector respectively. The second one is the error for the estimated inertial boresight vector ε_f defined as the norm of the difference vector between the estimated inertial boresight vector and the ground truth inertial boresight vector as follows

$$\varepsilon_f = \| \vec{f} - \tilde{\vec{f}} \|_2 \tag{2.21}$$

where $\vec{f}$ and $\tilde{\vec{f}}$ are respectively the ground truth inertial boresight vector and the estimated inertial boresight vector. Lastly, the third error indicator is the error for the estimated rotation angle ε_θ defined as the difference between the estimated rotation angle about $\tilde{\vec{f}}$ and the ground truth rotation angle about $\vec{f}$ as follows

$$\varepsilon_\theta = \| \theta_\delta - \tilde{\theta} \|_2 \tag{2.22}$$

where θ_δ and $\tilde{\theta}$ are respectively the ground truth rotation angle about $\vec{f}$ with respect to $\delta = 0^\circ$ and the estimated rotation angle about $\tilde{\vec{f}}$.

The sensor parameters are required to be introduced to the algorithm so that the corresponding database could be generated in advance to implementation. In this study, the sensor parameters of the CubeStar [114] assembled in the project SharjahSat-1 [115] are simulated. Thus, the database feature vectors $\vec{\tau}_i$ and the observation feature vectors $\vec{\psi}$ are generated in accordance with the values given in Table 2.2. CubeStar is a miniature star tracker specifically intended for, but not limited to low-power, performance-critical CubeSat applications [114]. Note that the brightness limit is arbitrary to better present and illustrate the results of the study in accordance with the parametric characteristics of the algorithm, while the on-board star catalog of CubeStar is actually a reduced list of the Hipparcos catalog with 410 stars brighter than 3.8 V_{mag}.

Table 2.2 : The sensor parameters of the CubeStar simulated.

Parameter	Value
FoV	$42^\circ \times 42^\circ$
Resolution	937×937 pixels

Besides the sensor parameters, two other parameters control the database creation. The overlap ratio and the brightness threshold are highly relevant to the overall performance of the algorithm. The overlap ratio, denoted by p, is a threshold used to determine the

amount of overlap area for the successive catalogue regions used to extract database feature vectors. As the overlap ratio increases, so does the number of database feature vectors. On the other hand, the brightness threshold, denoted by κ, specifies the brightness sensitivity of the sensor in terms of M_V. A smaller brightness threshold causes fewer number of stars to be detected by the simulated sensor so that the database feature vectors are extracted by using fewer number of stars.

2.3 Problem Statement

This section of the study proposes a LSI algorithm based on pattern recognition and regularization. An optimisation method is used to solve the star identification problem. The aim is to determine the criteria that would allow the given observed image of stars to be matched with the catalogue. To this end, a dictionary-based star matching method is developed, where a pattern recognition approach leads the way by using a novel feature extraction accompanied by a 1NN classifier for coarse estimation and a regularization operator by regression by LSE for fine estimation. The algorithm is divided into three sections, including the 1NN classifier by means of the binary search method using the Euclidean distances between the observation vector and each database vector, the dictionary setup and the regularization. The 1NN classifier is used to make a coarse approximation of the solution. It finds the database vector with the highest similarity. The highest similarity database vector is used to set up the dictionary used in the regularization process, which allows fine solution approximation. Finally, a maximum likelihood prediction is obtained from the resulting solution vector.

2.4 Methodology

The algorithm to be employed for the LSI method in this thesis consists of three stages including feature extraction, pattern recognition and regularization. A block diagram of the algorithm is given in Figure 2.1, which reveals the inputs, outputs and processes. The initial raw input is the observed digital image frame while the final output is a pair of inertial vectors representing the boresight vector and the rotation angle about it. The database is generated in advance and embedded into the system. It is generated through the process of feature extraction using the star catalog [71]. Similarly, the observation

feature vector is extracted from the body vectors of the objects in the observed image frame using the same feature extraction process. The 1NN classifier within the pattern recognition block uses the observation feature vector and the database as two inputs while outputting the major dictionary element which is the most similar database feature vector. The LASSO regression scheme within the regularization block utilizes the observation feature vector and the dictionary generated from the database through the dictionary setup process as two inputs while outputting the final inertial vectors.

2.4.1 Features and database

The number of stars in the FoV, brightness mean and brightness standard deviation were used to construct feature vectors in a study [116]. However, this study sets a single frame object for each frame given within a specific FoV and, for each image frame, generates a separate feature vector by implementing a log-polar transformation [45] on the body vector assuming that the center point of the frame is the origin. The feature vector comprises five elements including the number of stars n in the image frame, the brightness mean of the stars A, the polar radius r of the frame object, the brightness standard deviation σ_A and the polar radius standard deviation σ_r of the stars within the set $\mathscr{S}$. Also, an additional feature is extracted from the frame object, the polar angle θ. However, this feature is not included in the definition of feature vector, rather it is used to estimate the rotation of the camera plane about the boresight vector. A set of feature vectors is obtained to construct the database, $\overrightarrow{\tau}_i$, defined in Equation 2.10. The observation feature vector $\overrightarrow{\psi}$ is also defined using the same features. The Hipparcos star catalog [71] is used to generate database feature vectors. The entries to be used for the purpose of simulation and database generation are given in Table 2.1. These entries are bounded by some constraints specified by the sensor parameters including overlapping ratio p and brightness threshold κ.

2.4.1.1 Feature extraction

The process of feature extraction yields a single feature vector for each single image frame. The single feature vector representing a single frame allows less amount of computation so that a higher overlap ratio used when extracting feature vectors may

significantly increase accuracy. The centroid and brightness of the stars detected in an image frame are processed to obtain a feature vector. A single body vector is yielded by calculating positional and brightness mean of the stars in the frame. The description of the features and the process of derivation of a feature vector from an image frame is shown in Figure 2.2. In the right-hand side, the single frame object is shown, which has the features derived using the positions and brightness values of the stars in the image frame on the left-hand side including the number of stars n in Equation 2.2, the brightness mean A in Equation 2.3, the polar radius r and the polar angle θ in Equation 2.4, the brightness standard deviation σ_A in Equation 2.6 and the polar radius standard deviation σ_r in Equation 2.8. The derivation of a circular image which is used to obtain a feature vector through the same procedure is depicted in Figure 2.14.

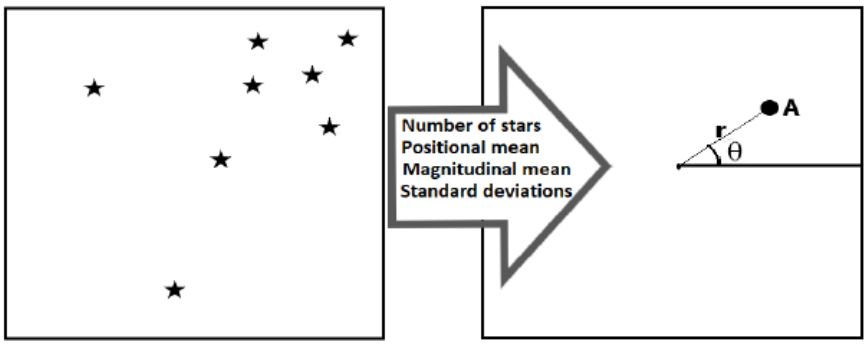

Figure 2.2 : Description of a feature vector and its formation from an image.

The features are examined in detail to check whether they are convenient to include in the feature vector. The values of features reveal a significant variation over the whole range of right ascension and declination. Thus, it is reasonable to select the features that can contribute to derivation of meaningful results in the 1NN method and the regression method thanks to their spatially fluctuating structure. The distribution of the features over the whole database is investigated. The fluctuations of the features n, A, r, θ, σ_A and σ_r are shown over the whole database. The overlap ratio is set to the value $p = 0.9$ while no brightness threshold is set, implying that the database feature vectors are obtained from the database images overlapping by 90% and all stars in the database are involved in the process without a brightness threshold. The fluctuating behavior of the features is illustrated at a glance over the whole database given that $\alpha \in [0^\circ, 360^\circ]$ and $\delta \in [-90^\circ, 90^\circ]$. The sensor parameters are selected from Table 2.2.

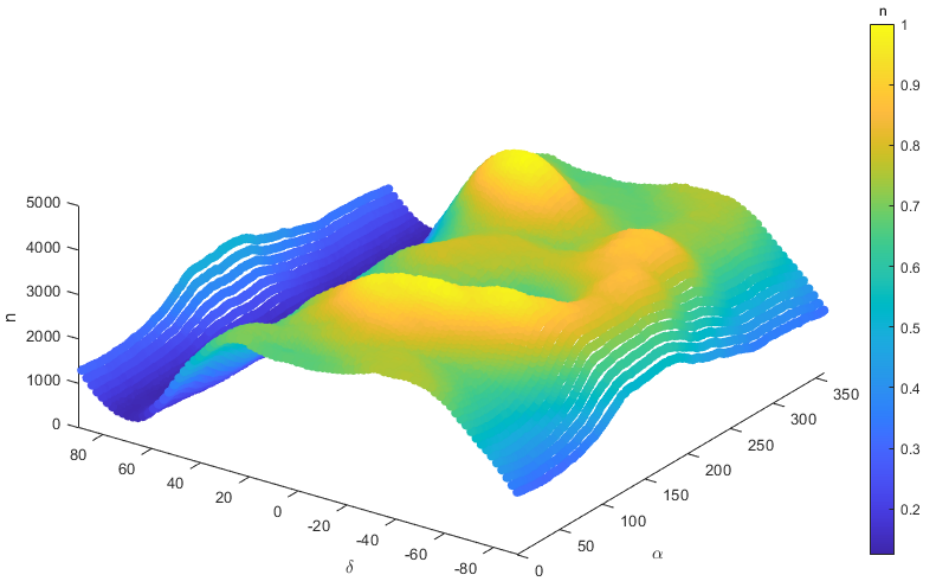

Figure 2.3 : Distribution of the number of stars n over the database.

Figure 2.3 shows the distribution of the values of the feature n over the whole database. It is shown that the number of stars corresponding to each image frame fluctuates within a wide interval ranging from between a few hundreds up to nearly 5000 stars in an image frame. Note that the stellar intensity is the highest in the vicinity of $\delta = 0°$ while it gradually declines and hits lower values in the pole region $\delta = \pm 90°$. The lowest stellar density is given within the interval $\delta = [60°, 80°]$. The stellar density tends to be higher in the southern hemisphere where declination is negative $\delta < 0°$. A general overview of the feature n gives a rough idea that it is a convenient feature to be used as an element of the feature vector because its value neither stays constant nor follows a repeating pattern that would avoid it from having a discriminating characteristic.

According to Figure 2.4 revealing the distribution of the values of the feature A over the whole database, the values of brightness mean varies within the interval $A \in (8,9)$ in terms of V_{mag}. The fluctuating pattern of A is not similar to that of n, which is required to avoid linear dependency of the features. Unlike the feature n, the feature A has distributed extrema over the $\alpha \times \delta$ plane. Moreover, the extrema of each feature have different positions on the plane, which is also required to involve different discriminating patterns of the features in the process.

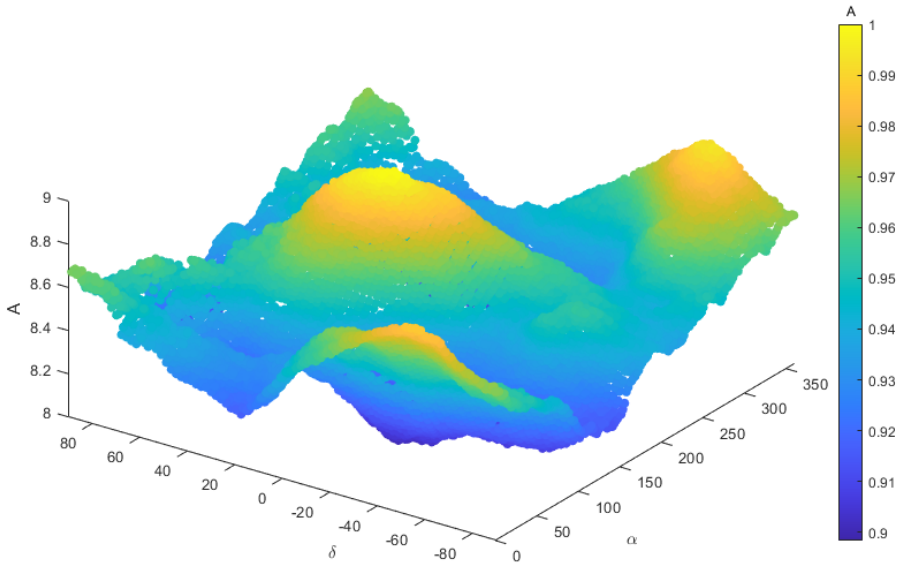

Figure 2.4 : Distribution of the brightness mean A over the database.

Figure 2.5 shows the distribution of the values of the feature r over the whole database. The values of polar radius is given in terms of degrees in the graph. There are two bands of extreme values where the value of r reaches up to 10° for all values of α when the declination is bounded around $\delta \approx 60^\circ$ and $\delta \approx 80^\circ$. Although this pattern is followed by the feature n in the same region, the rate of changes are completely different since the rate of change of r is very large in terms of δ where the local maxima and the local minima of the feature r are gathered within the band $\delta = [60^\circ, 80^\circ]$ where the feature n has only the local minima. On the other hand, the remaining regions of the database have a different pattern compared to the former two features, which fosters the discriminating behavior of the feature vector with the given element. Note that the rate of change of the feature r tends to increase in the pole regions while it is flatter in the equator region because a larger number of stars n in the equator region attracts the frame object towards the center of the image frame. Yet, the fluctuating characteristic of the feature r resumes despite smaller values in the equator region.

Figure 2.6 illustrates the distribution of the values of the feature θ over the whole database. While the values of θ tend to avoid large changes in some regions, they exhibit abrupt and large changes in some other regions. The feature θ is the rotation of the frame object about the center of the image frame with respect to the positive horizontal axis of the image frame. Thus, when the feature r approaches zero as the

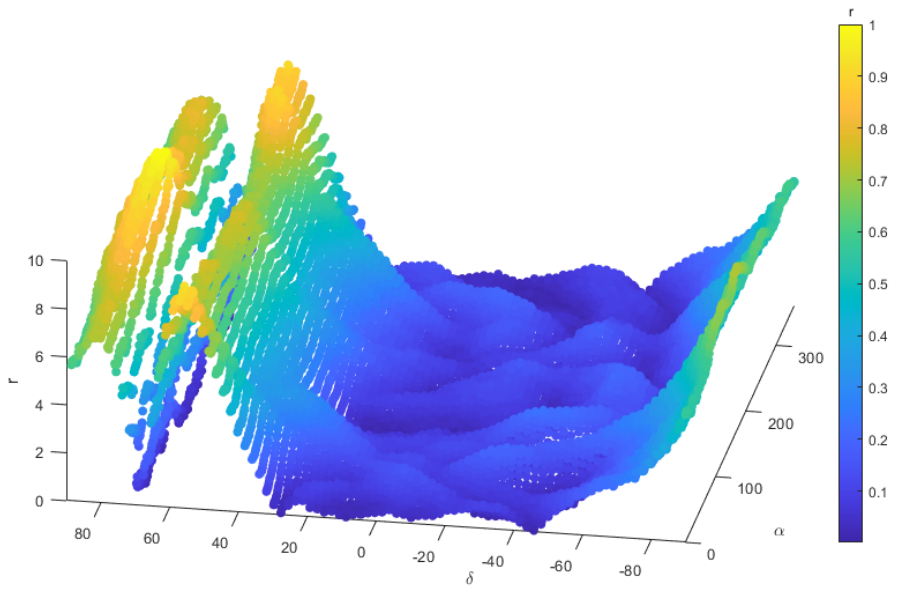

Figure 2.5 : Distribution of the polar radius r over the database.

frame object is attracted towards the center of the image because of the larger values of the feature n, the value of the feature θ may exhibit large changes due to transition of the frame object across the center of the image frame. This irregular pattern of the feature θ may lead to false approximation in the phase of pattern recognition. Therefore, it is excluded from the feature vector. Instead, it is exploited in the phase of the rotation angle of the estimated boresight vector.

According to Figure 2.7 showing the distribution of the values of the feature σ_A over the whole database, the feature σ_A seems to have a fluctuating pattern similar to the feature A at the first glance. However, the locations of the extrema are different from those of the feature A, and the surface gradient of the feature σ_A is more frequently oscillated. These properties not only makes this feature have discriminating characteristic but also prevents it from being linear dependent on the feature A.

Figure 2.8 depicts the distribution of the values of the feature σ_r over the whole database. The feature σ_r also exhibits a pattern similar to the feature r, in particular, within the range of $\delta \in [60^\circ, 80^\circ]$. Except that, it follows a unique pattern in the remaining features while its values are bounded within a larger interval in comparison with the feature σ_r. Thus, the feature σ_r also has a discriminating characteristic without a clearly visible dependency on other features.

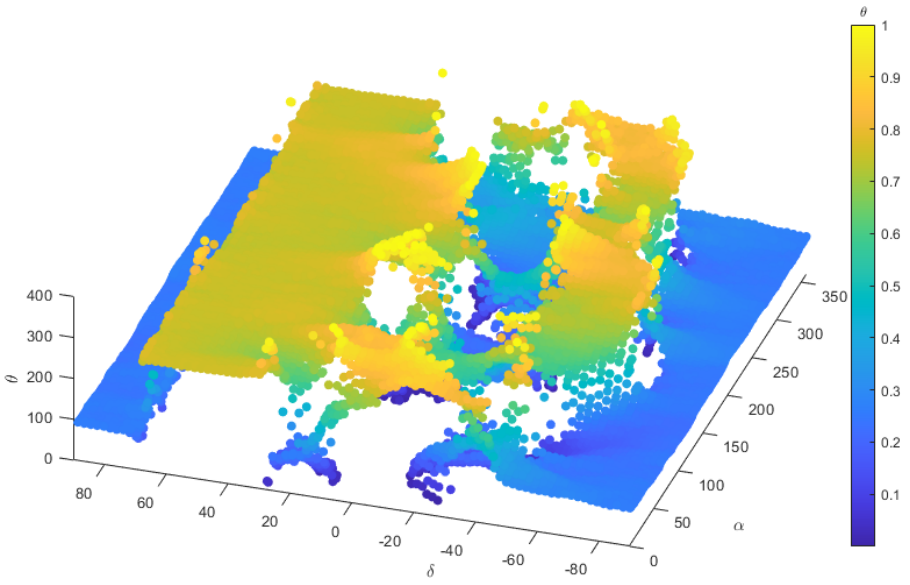

Figure 2.6 : Distribution of the polar angle θ over the database.

A more detailed investigation is performed on the features to reveal and verify that they provide statistically meaningful information to be used in the process of regression with regards to their discriminating characteristics. Accordingly, the effects of the magnitude threshold κ and overlap ratio p on the variation of the feature vectors are examined. Such an investigation uncovers the level of variation on the features, thus, the extent that the vectors assigned to each database image are separated from each other. The variations of the features n, A, r, θ, σ_A and σ_r are shown separately. The consistent results provide meaningful information that will allow interpretation of each parameter for such conditional variations.

In Figure 2.9, the variations of the features are shown along a full zonal cycle of right ascension $\alpha \in [0°, 360°]$ within a declination interval $\delta \in [-6°, 36°]$ by changing the overlap ratio p from 0.6 up to 0.95. No brightness thresholding is applied, that is, all stars in the database are taken into consideration. The first two features n and A reveal a smooth fluctuation along α. The fifth feature σ_A also fluctuates rather smoothly compared to σ_r in accordance with A considering that both are originated from brightness properties. It is a good indication that the features n and A do not stay constant within a limited declination zone where the stellar density is the highest. The features r and θ fluctuates much more frequently compared to other features.

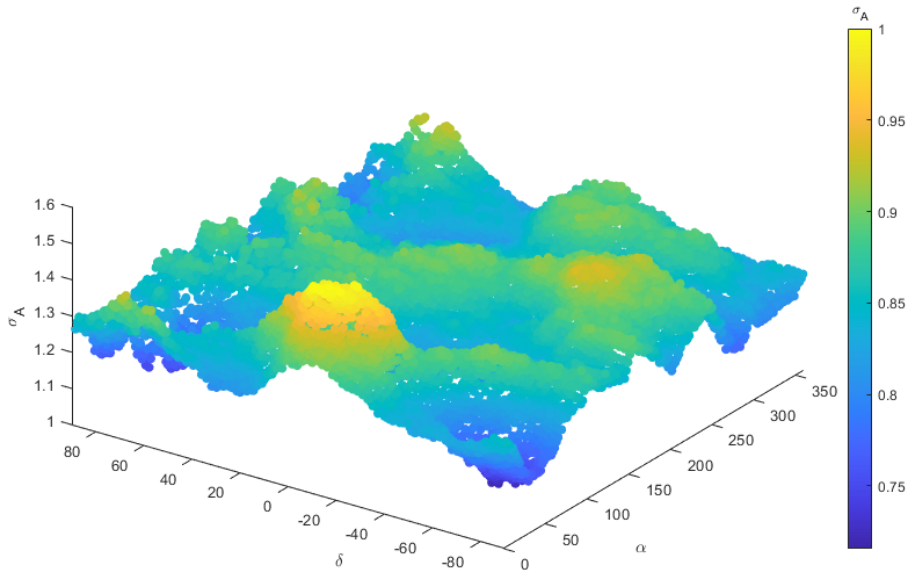

Figure 2.7 : Distribution of the brightness standard deviation σ_A over the database.

Additionally, the effect of the weighting method is apparent, making the fluctuation steeper. When r has high values, the variation of θ is small. However, small values of r lead to dramatic changes in θ. This phenomenon can be manipulated to enable the algorithm to gain robustness since a higher value of r obviously makes θ less significant and vice versa. Besides, the irregular pattern exhibited by the feature θ makes it convenient for use in estimation of the rotation angle about the boresight vector rather than inclusion in the feature vectors.

In Figure 2.10, the variations of the features are shown along a full zonal cycle of declination $\delta \in [-90°, 90°]$ within a right ascension interval $\alpha \in [0°, 42°]$ by changing the overlap ratio p from 0.6 up to 0.95. No brightness thresholding is applied so that all stars in the database are taken into consideration. The relation between n and A is not the same as that in Figure 2.9. On the contrary, they are more likely to be directly proportional in this case. The relation between n and r is clearly visible, where the polar radius decreases as the stellar density increases, which also drives the polar angle θ into a more unstable state. The features σ_A and σ_r reveal unique properties that are capable of contributing to the identification algorithm.

In Figure 2.11, the variations of the features are shown along a full zonal cycle of right ascension $\alpha \in [0°, 360°]$ within an interval of declination $\delta \in [-6°, 36°]$ with

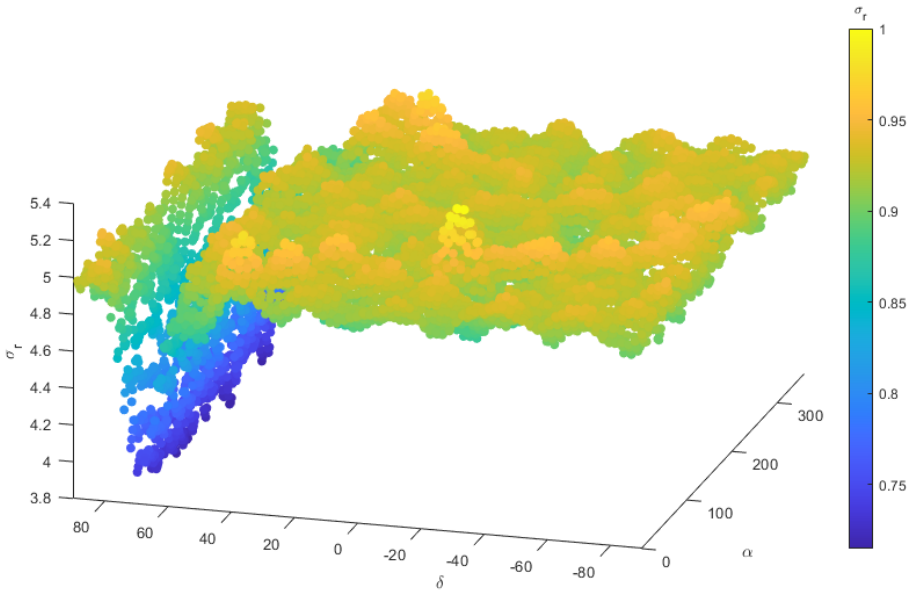

Figure 2.8 : Distribution of the polar radius standard deviation σ_r over the database.

different values of the brightness threshold κ. The value of the overlap ratio is assigned to $p = 0.9$. The effect of varying lower bound brightness threshold κ on the features is revealed. The behaviors of the features are investigated subject to different lower bounds of the brightness threshold including $\kappa \longrightarrow -\infty$ (no threshold), and $\kappa = 1,3,5,7,9$. A change in stellar density n exists for all thresholds. The relation between n and A is likely to be inversely proportional as is the case in Figure 2.9. It is shown that the value of κ has a significant effect on the behavior of r. The relation between r and θ is still true as in Figure 2.9.

In Figure 2.12, the variations of the features are shown along a full zonal cycle of declination $\delta \in [-90°, 90°]$ within right ascension interval $\alpha \in [0°, 42°]$ with different values of the brightness threshold κ. The value of the overlap ratio is assigned to $p = 0.9$. The effect of a lower bound brightness threshold κ on the features is shown. The features are calculated for the same values of κ. The curves of the features appear to resemble those in Figure 2.10 except that the curves tend to deviate more within the same amount of increase in κ. As of $\kappa \leq 7$, the difference between the curves becomes significant in the given scale. For instance, the number of stars n decreases and the brightness mean A increases dramatically in case of $\kappa = 9$. This is very likely because of a steep decline in the number of stars in the database due to a high lower

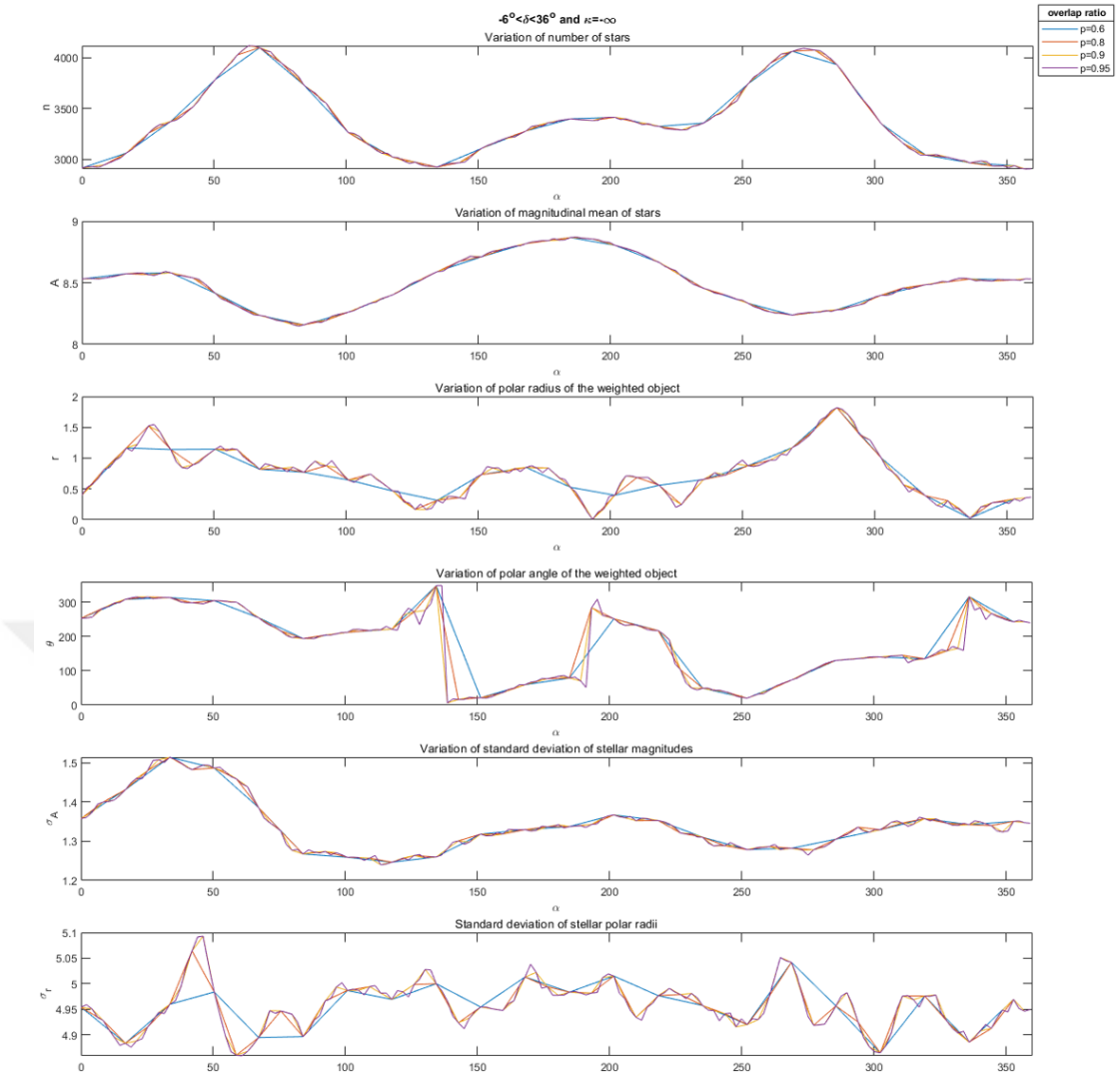

Figure 2.9 : Variation of features along the direction of right ascension when declination is within $[-6°, 36°]$ with varying overlap ratio.

bound threshold κ. Also, higher values of κ cause less amount of variation for n and A but more for r and θ. The effects of the weighting model given in Equation 2.5 is distinguishable in Figure 2.11 and Figure 2.12, as the increasing values of κ do not have a significant effect on the nature of the curves of r, θ and σ_r where the weighting model is implemented. It is certain that the features σ_A and σ_r will have a contributing effect on the success of convergence.

All features have a contributing effect on the success of convergence both in the regression procedure and in the classification procedure. Five features excluding the polar angle θ are used in generating the database feature vector set $\{\overrightarrow{\tau}_i\}$ and the

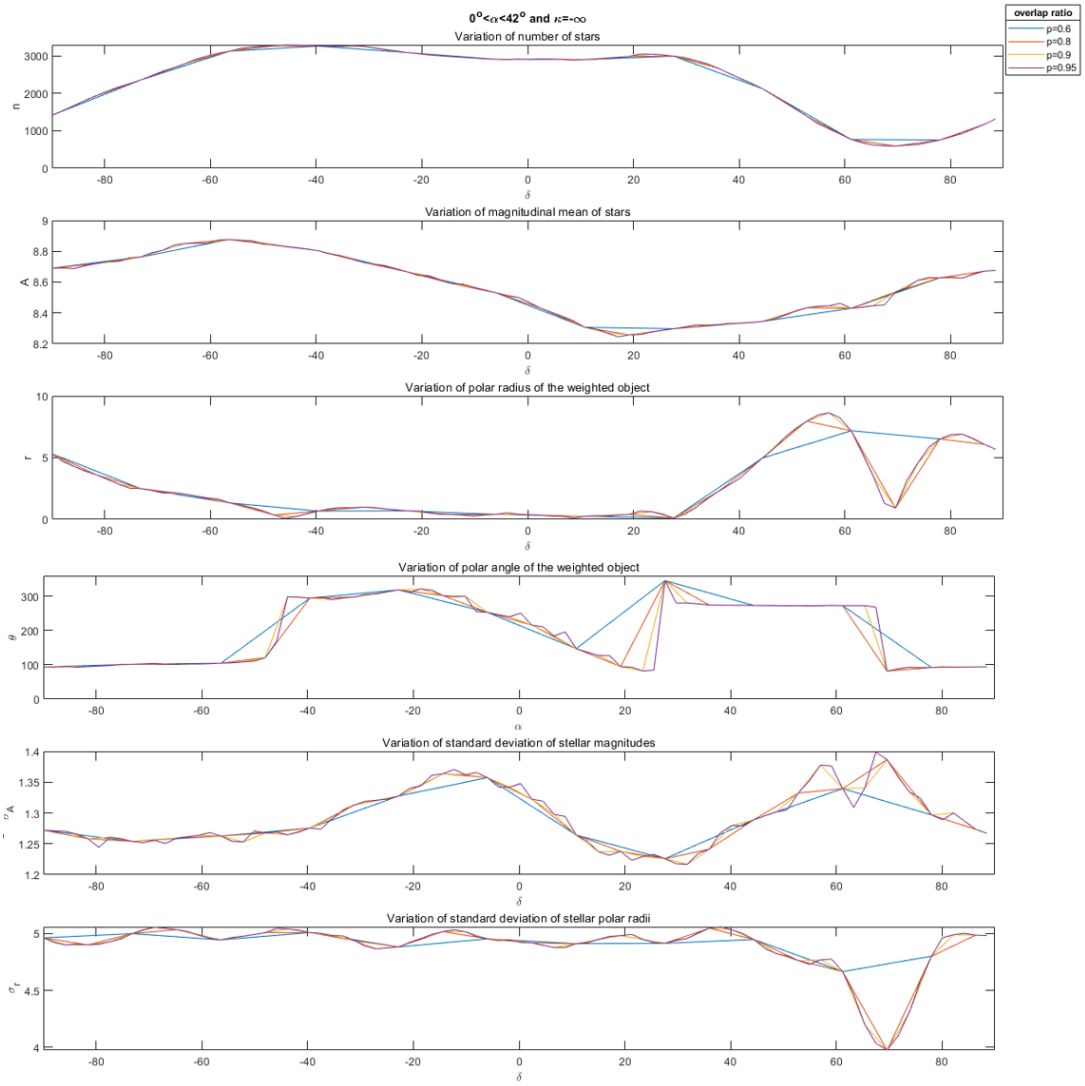

Figure 2.10 : Variation of features along the direction of declination when right ascension is within $[0°, 42°]$ with varying overlap ratio.

observation feature vector $\overrightarrow{\psi}$ while θ is used in the reconstruction phase which aims to make an estimation of the rotation angle of the estimated boresight vector $\overrightarrow{\tilde{f}}$ with respect to the axis of right ascension ($\delta = 0°$). Additionally, the curves of the features tend to have a larger change for variations of the brightness threshold κ than it is for variations of the overlap ratio p. For instance, n decreases and A increases dramatically for larger values of κ. This is very likely because of a steep decline of stellar density in the database due to a high lower bound threshold κ. The effects of the weighting model appear in variations of κ values, as the increasing values of κ do not have a tremendous effect on r, θ and σ_r where the weighting model is implemented.

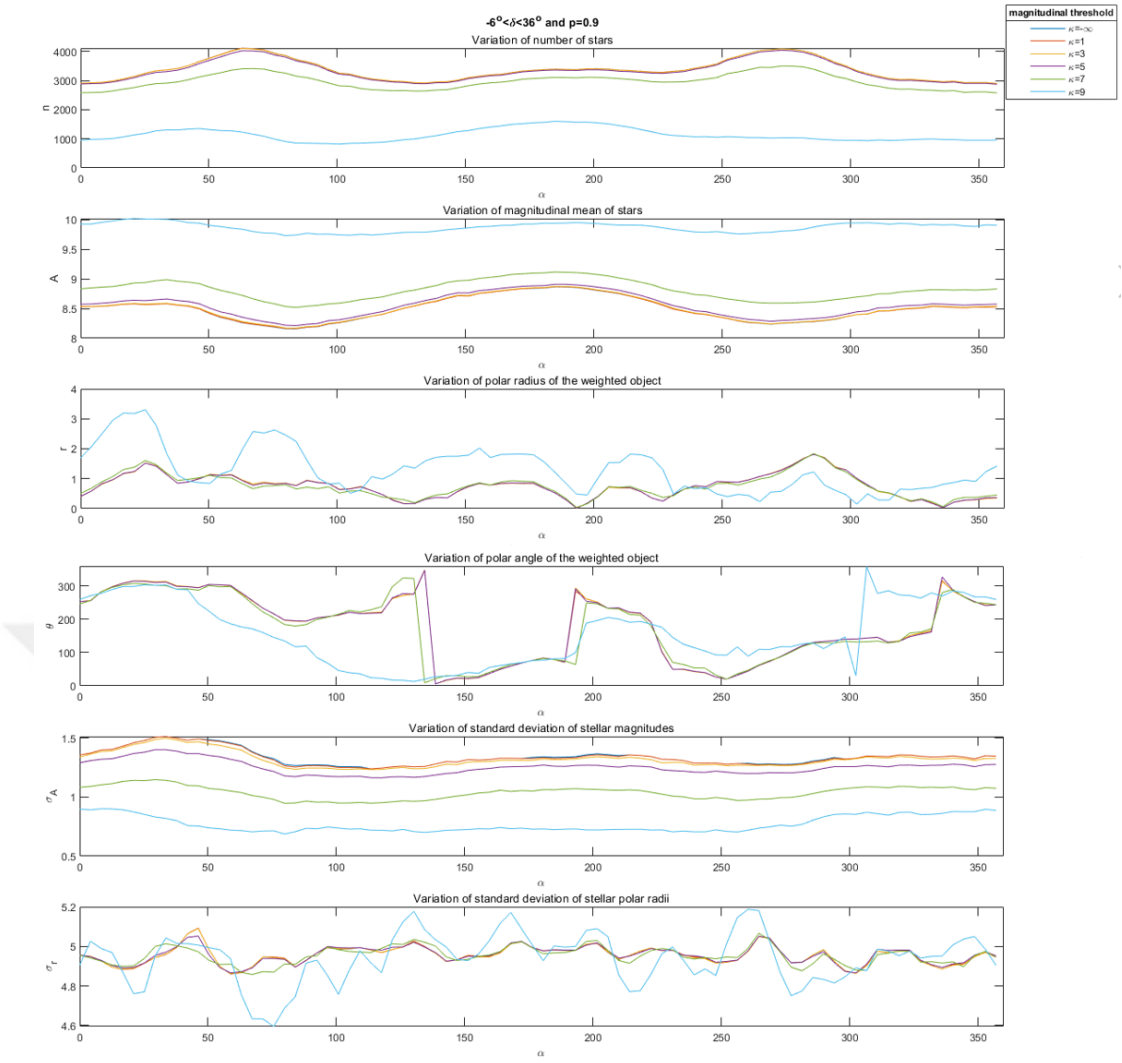

Figure 2.11 : Variation of features along the direction of right ascension when declination is within $[-6°, 36°]$ with varying brightness threshold.

2.4.1.2 Database generation

The capability of representing each single image frame with a single six-element feature vector allows a decrease in the computational complexity so that the amount of overlap may be increased when picking the image frame from the star catalog database. The scheme for derivation of the set of feature vectors from the database is illustrated in Figure 2.13. The gray regions reveal the overlapping regions in the figure where the horizontal and vertical axes represent right ascension and declination respectively. In this scheme, the total number of image frames, that is the total number of database

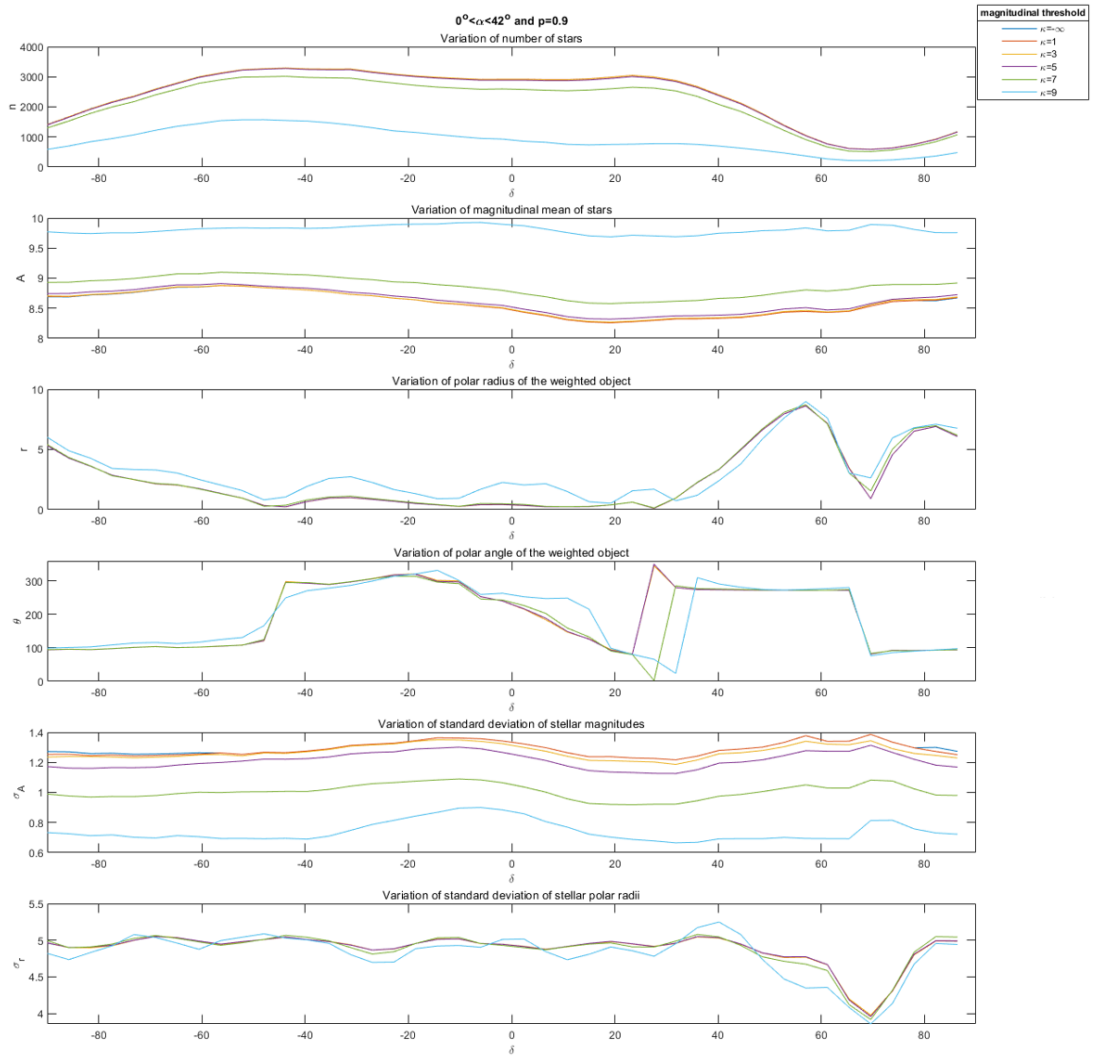

Figure 2.12 : Variation of features along the direction of declination when right ascension is within $[0°, 42°]$ with varying brightness threshold.

feature vectors, is given by

$$N = \frac{2\pi^2}{\phi_x \cdot \phi_y \cdot (1-p)^2} \tag{2.23}$$

where ϕ_x and ϕ_y represent the FoV angles in the directions of right ascension and declination respectively, and p is the overlap ratio. Since a single feature vector is obtained for each image frame, the set of feature vectors consist of N feature vectors. It is apparent that an increase in the FoV values ϕ_x and ϕ_y causes a decline in the number of feature vectors N while an increase in the overlap ratio p leads to an increase.

Considering that a full-FoV feature extraction that is implemented on the whole image frame may impose some difficulties in the star identification algorithm due to

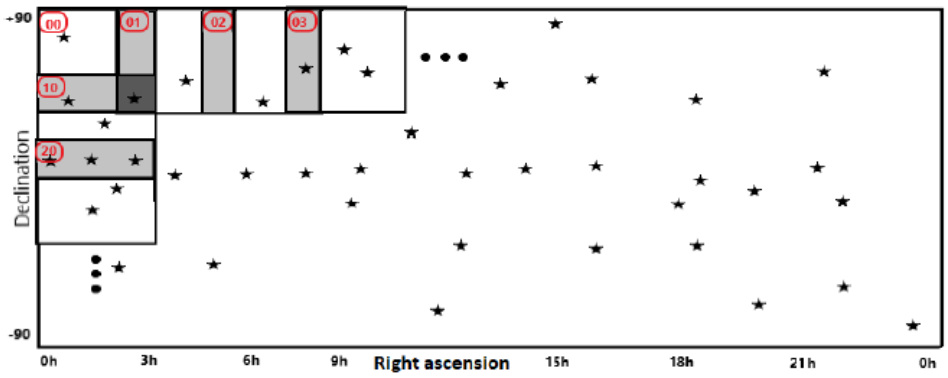

Figure 2.13 : Selection of image frames from the star catalog database to generate feature vectors.

geometrical verification, an alternative feature extraction scheme is proposed, which is called partial-FoV feature extraction that is implemented on a circular image frame. The circular image frame is obtained by rotating the whole rectangular frame about its origin and extracting the intersecting circular frame, which is identical to extraction of the circle with the maximal area that fits into the frame as given in Figure 2.14. In this scheme of feature extraction, the feature vector is obtained by carrying out the same procedure as implemented in the full-FoV feature extraction except that this is implemented on the circular frame. Note that some information is lost as the region outside the circular frame in the whole frame is excluded. This information is gained by maintaining an overlap ratio that will allow covering all stars in the database. Moreover, a partial-FoV feature extraction ensures independence on rotation of the camera plane in the estimation process, which allows an easier implementation of geometric approximation.

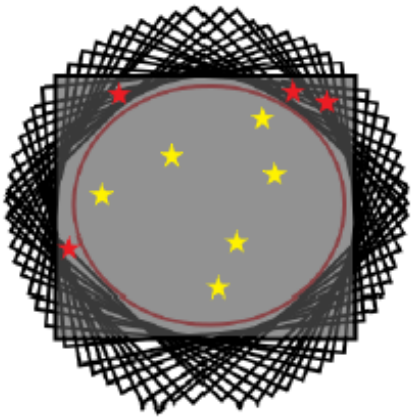

Figure 2.14 : Derivation of a circular image frame from a whole image frame.

The issue of information loss incurring due to non-overlapping regions in case of partial-FoV feature extraction may limit the level of accuracy, and lead to poor estimation. Therefore, an investigation is carried out to address this issue through calculation a value of the overlapping ratio p that would be sufficient to avoid any information loss. The calculation is based on a simple geometric analysis.

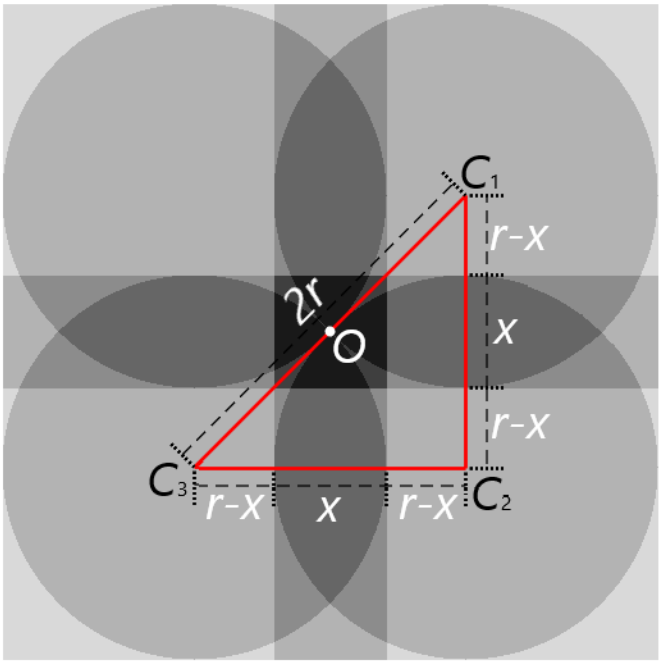

Figure 2.15 : Calculation of the value of the overlapping ratio p sufficient for a database without information loss.

In Figure 2.15, four representative database image frames are given side by side. Each of these image frames corresponds to a separate database feature vector. The surrounding four square image frames are used in case of use of full-FoV feature extraction whereas the inner circles correspond to database feature vectors derived by using partial-FoV feature extraction. Each layer of overlapping is shaded by an additional tone of gray color. For instance, considering the circles, while each circle is shaded by the lightest gray tone in case of no overlapping, a darker tone is used when two circles overlap. Due to the periodic pattern used in selection of image frames, three circles do not coincide to overlap. Rather, four circles may overlap. In this scheme, since it is aimed to calculate the overlapping ratio with the least value to ensure no

information loss, it is assumed that the circular frames aligned diagonally are tangent to each other while the ones aligned side by side overlap with an overlapping ratio p that ensures no information loss. By this means, a star that is excluded from a partial-FoV feature extraction although taking part in the full-FoV image frame is included within at least one of the adjacent circular image frames. The point O is the point where the diagonal circular frames are tangential, given that the points C_1, C_2 and C_3 are the centers of the given circular frames. Since the circles with the center points C_1 and C_3 are tangential to each other, the length of the line segment $\overline{C_1C_3}$ is $2r$ where the radius of a circle is r. The length x represents the measure of intrusion for two circles aligned side by side. Thus, the lengths of the line segments $\overline{C_1C_2}$ and $\overline{C_2C_3}$ are $2r-x$. Given that the angle $\widehat{C_1C_2C_3}$ is a right angle, the Pythagoras theorem may be used to define x in terms of r such that

$$2 \cdot (2r-x)^2 = 4r^2 \tag{2.24}$$

which yields

$$x = \left(2-\sqrt{2}\right) r \tag{2.25}$$

where the measure of intrusion x is expressed in terms of the radius r.

The FoV angles ϕ_x and ϕ_y specify the diameter of a circular image frame, which are equal in accordance with the simulation parameters given Table 2.2 used in this study. Then, assuming that $\phi_x = \phi_y = \phi$, the radius of a circular frame is obtained $r = \dfrac{\phi}{2}$. Given that the overlapping ratio is obtained by

$$p = \frac{x}{\phi} \tag{2.26}$$

where p is defined by the ratio between the measure of intrusion and the FoV angle, a value overlapping ratio is yielded by substituting ϕ in Equation 2.25

$$p_{min} = \frac{\left(2-\sqrt{2}\right)\dfrac{\phi}{2}}{\phi} = 0.2929 \tag{2.27}$$

where p_{min} is the minimum value of p to avoid any information loss when using the partial-FoV feature extraction. This implies that an overlapping ratio larger than 29.29% would be sufficient to avoid information loss. In the simulations carried out in this study, the values of overlapping ratio are far more than this value of p_{min}.

2.4.2 Mathematical model and algorithm description

The dictionary-based star matching method consists of two stages, which include the 1NN classification through binary search by Euclidean distance that allows selection of the database feature vector with the highest similarity to the test feature vector and generation of a linear combination of the selected database feature vector and the neighboring database feature vectors that maximizes the convergence of similarity criterion through the minimization of LSE and L_1 minimization, specifically LASSO regression. The proposed method is categorized as an LSI algorithm that needs no a priori information retrieved from the previous iteration, which also requires a step for setting up a dictionary between two phases mentioned above and also a normalization process in feature extraction because of possible dominance of high-valued features in the process of 1NN classification due to a big difference in the scales of the numeric values of the features.

Algorithm 1 Lost-in-space star identification

1: $\overrightarrow{\psi} \leftarrow \mathscr{F}_{\psi}\left(\mathrm{I_o}\right)$ (▷) Observation feature vector
2: $\theta \leftarrow \mathscr{F}_{\theta}\left(\mathrm{I_o}\right)$ (▷) Observation polar angle
3: $\hat{\overrightarrow{\psi}} \leftarrow \mathscr{F}_{\mathrm{N}}\left(\overrightarrow{\psi}\right)$ (▷) Normalization
4: **for** $i = 1 : N$ **do** (▷) N: Number of database feature vectors
5: $\quad d_i \leftarrow \mathscr{F}_{\mathrm{Ed}}\left(\hat{\overrightarrow{\psi}}, \hat{\overrightarrow{\tau}}_i\right)$ (▷) Euclidean distances
6: **end for**
7: **for** $i = 1 : N$ **do** (▷) N: Number of database feature vectors
8: $\quad \overrightarrow{t}_5 \leftarrow \mathscr{F}_{\mathrm{1NN}}\left(d_i\right)$ (▷) 1NN method
9: **end for**
10: $\mathbf{T} \leftarrow \mathscr{F}_{\mathrm{D}}\left(\overrightarrow{t}_5, \hat{\overrightarrow{\tau}}_i\right)$ (▷) Set dictionary template
11: $\overrightarrow{\omega} \leftarrow \mathscr{F}_{\mathrm{R}}\left(\mathbf{T}, \hat{\overrightarrow{\psi}}\right)$ (▷) Regularization
12: $\tilde{\overrightarrow{\psi}} \leftarrow \mathbf{T}\ \overrightarrow{\omega}$ (▷) Reconstruction
13: $\tilde{\overrightarrow{f}} \leftarrow \mathbf{F}\ \overrightarrow{\omega}$ (▷) FoV estimation
14: $\tilde{\theta} \leftarrow \Theta\ \overrightarrow{\omega}$ (▷) Rotation estimation

The LSI method is illustrated in Algorithm 1, which provides a brief description at a glance. Each line represent a major step of the algorithm. The first and second lines denote extraction of an observation feature vector $\overrightarrow{\psi}$ and a polar angle θ from an observed digital image $\mathrm{I_o}$ using the functions $\mathscr{F}_{\psi}$ and $\mathscr{F}_{\theta}$ used for extracting feature vector and polar angle respectively. The functions $\mathscr{F}_{\psi}$ and $\mathscr{F}_{\theta}$ yield $\overrightarrow{\psi}$ and θ in

accordance with Equation 2.4 and 2.10 as explained in Section 2.4.1 in detail. The third line unveils the normalization function through the function $\mathscr{F}_{\mathrm{N}}$ that inputs $\overrightarrow{\psi}$ and outputs the normalized observation feature vector $\overrightarrow{\hat{\psi}}$ with respect to Equation 2.11. The fifth line within the first *for* loop reveals the calculation of the Euclidean distances between $\overrightarrow{\hat{\psi}}$ and each normalized database feature vector $\overrightarrow{\hat{\tau}}_i$ as preserved in a set containing the elements d_i using the function of Euclidean distance $\mathscr{F}_{\mathrm{Ed}}$ where $i = 1, \cdots, N$ and N is the number of database feature vectors. The function $\mathscr{F}_{\mathrm{Ed}}$ is based on Equation 2.12. The procedure of detection of the database feature vector most similar to $\overrightarrow{\hat{\psi}}$ is shown in the eighth line in the second *for* loop where $\overrightarrow{t}_5$ is the most significant vector in the template detected by the 1NN function $\mathscr{F}_{\mathrm{1NN}}$ that finds the minimal distance from the set $\{d_i\}$ through binary search. The tenth line shows the dictionary setup through the corresponding function $\mathscr{F}_{\mathrm{D}}$ that detects the database feature vectors neighboring $\overrightarrow{t}_5$. The eleventh line reveals the regularization weight vector $\overrightarrow{\omega}$ approximated by the regularization function $\mathscr{F}_{\mathrm{R}}$ that uses the LASSO implementing L_1 regression. The last three lines show the estimations obtained by multiplying the dictionary template $\mathbf{T}$, the FoV template $\mathbf{F}$ and the rotation template Θ by $\overrightarrow{\omega}$ to yield the reconstructed feature vector $\overrightarrow{\tilde{\psi}}$, the estimated boresight vector $\overrightarrow{\tilde{f}}$ and the estimated rotation angle $\overrightarrow{\tilde{\theta}}$.

2.4.2.1 Pattern recognition

The scheme of pattern recognition is based on a 1NN classifier with binary search using euclidean distance. This part of the algorithm implements a search for the feature vector in the database that has the highest similarity with the given test feature vector. This search algorithm is based on the detection of the minimal Euclidean distance. The 1NN classifier is employed to detect the smallest Euclidean distance between the observation vector and database vectors. However, the numeric values of different types of elements are on different scales. For instance, while the number of stars n may vary around several hundreds in numerical value depending on the value of the brightness threshold κ, polar angle varies within $[0^\circ, 360^\circ]$. In order to remove the exaggerated effect of the accompanying features with larger values, a normalization process is applied in advance to the implementation of the classification method.

A set of Euclidean distances are obtained using Equation 2.12 within the function $\mathscr{F}_{\text{Ed}}$ defined in Algorithm 1 as

$$\{d_i\} = \mathscr{F}_{\text{Ed}}\left(\hat{\overrightarrow{\psi}}, \hat{\overrightarrow{\tau}}_i\right) = \|\hat{\overrightarrow{\psi}} - \hat{\overrightarrow{\tau}}_i\|_2 \tag{2.28}$$

where $\|\overrightarrow{v}\|_2 = \sqrt{\langle \overrightarrow{v}, \overrightarrow{v} \rangle}$ denotes the L_2 norm on the vector $\overrightarrow{v}$, $\langle \cdot \rangle$ representing the operation of inner product, and $i = 1, \cdots, N$ with N defined in Equation 2.23. Then, 1NN classifier is, by means of the method of binary search [88], applied on the set $\{d_i\}$ to detect the most similar database feature vector, which is defined as

$$\overrightarrow{t}_5 = \mathscr{F}_{\text{1NN}}(\{d_i\}) = \hat{\overrightarrow{\tau}}_k = \min_{d_i}\left\{\hat{\overrightarrow{\tau}}_i\right\} \tag{2.29}$$

where k is the index of the database vector satisfying the minimal Euclidean distance.

2.4.2.2 Dictionary setup

The process of dictionary setup ends up with a dictionary as shown in Figure 2.16. The left-hand side shows the positions of the vectors in a scaled image with the axes of right ascension and declination, where it is hard to distinguish the elements of the dictionary because the circular frame patches overlap by at least 96%. The right-hand side shows the dictionary and the test vector without scale for illustrative purposes. The red circle is the patch of the observation image, from which the observation feature vector is derived. The blue circle is the image patch which is detected by the 1NN classification. Besides, the black circular image patches are the neighboring elements of the database feature vector with closest Euclidean distance, which is estimated by 1NN classifier.

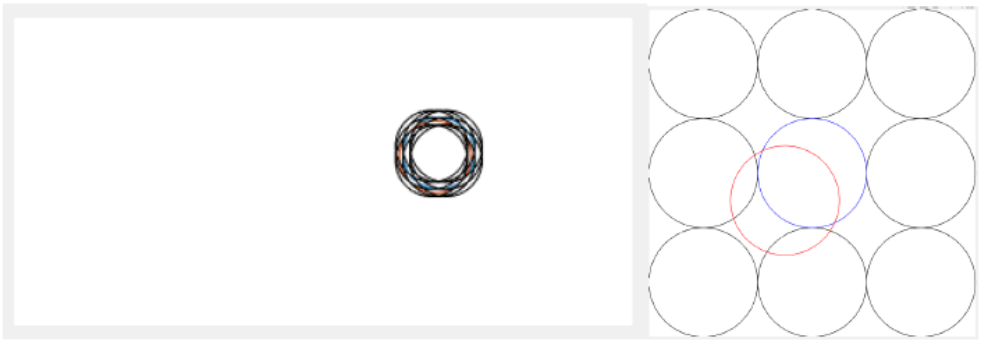

Figure 2.16 : An example of a dictionary with nine elements arranged with respect to the test vector represented by a red circle.

The function $\mathscr{F}_{\mathrm{D}}$ given in Algorithm 1 is used to retrieve the dictionary template matrix $\mathbf{T}$ such that

$$\mathbf{T} = \mathscr{F}_{\mathrm{D}}\left(\overrightarrow{t}_5, \left\{\hat{\overrightarrow{\tau}}_i\right\}\right) = \left(\overrightarrow{t}_1, \overrightarrow{t}_2, \cdots, \overrightarrow{t}_9\right) \tag{2.30}$$

where, providing that $\overrightarrow{t}_5 = \hat{\overrightarrow{\tau}}_k$ with respect to Equation 2.29, $\overrightarrow{t}_1 = \hat{\overrightarrow{\tau}}_{k-m-1}$, $\overrightarrow{t}_2 = \hat{\overrightarrow{\tau}}_{k-m}$ and $\overrightarrow{t}_3 = \hat{\overrightarrow{\tau}}_{k-m+1}$ construct three vectors in the top row of the dictionary template, $\overrightarrow{t}_4 = \hat{\overrightarrow{\tau}}_{k-1}$, $\overrightarrow{t}_5 = \hat{\overrightarrow{\tau}}_k$ and $\overrightarrow{t}_6 = \hat{\overrightarrow{\tau}}_{k+1}$ construct three vectors in the middle row of the dictionary template, and $\overrightarrow{t}_7 = \hat{\overrightarrow{\tau}}_{k+m-1}$, $\overrightarrow{t}_8 = \hat{\overrightarrow{\tau}}_{k+m}$ and $\overrightarrow{t}_9 = \hat{\overrightarrow{\tau}}_{k+m+1}$ construct three vectors in the bottom row of the dictionary template where $m = \dfrac{2\pi}{\phi_x \cdot (1-p)}$ is the number of database feature vectors in a single row with a constant value of right ascension, ϕ_x is the FoV angle in the direction of right ascension, and p is the overlap ratio.

2.4.2.3 Regularization and solution

The process of regularization is primarily based on a minimization of the LSE. The least square error is expressed as

$$\varepsilon_{\mathrm{LSE}} = \min_{\mathbf{T}} \| \overrightarrow{\psi} - \mathbf{T}\ \overrightarrow{\omega} \|_2^2 \tag{2.31}$$

which minimizes the square of the L_2 norm of the difference between the observation feature vector and the approximated vector obtained through linear combination of the feature vectors in the dictionary, where $\overrightarrow{\psi}$ is the observation vector, $\overrightarrow{\omega}$ is the weight vector and $\mathbf{T}$ is the dictionary matrix comprising dictionary feature vectors. $\mathbf{T}$ is designated to comprise the feature vectors obtained by using the closest Euclidean distances and the neighboring feature vectors so that it is expressed as given in Equation 2.13. The least square error minimization is depicted in Figure 2.17.

The LASSO regularization includes the minimization of LSE and L_1 norm. It stands for the least absolute shrinkage and selection operator, which is deduced from Elastic Net [117]. The elastic net solves the following regularization problem

$$\min_{\overrightarrow{\omega}_k} \left(\frac{1}{2M} \| \hat{\overrightarrow{\psi}} - \mathbf{T}\ \overrightarrow{\omega}_k \|_2^2 + \lambda\ \mathscr{P}_a(\overrightarrow{\omega}_k) \right) \tag{2.32}$$

where M is the number of feature vectors (the number of columns) in the dictionary matrix $\mathbf{T}$, $\hat{\overrightarrow{\psi}}$ is the normalized observation feature vector, λ is a non-negative

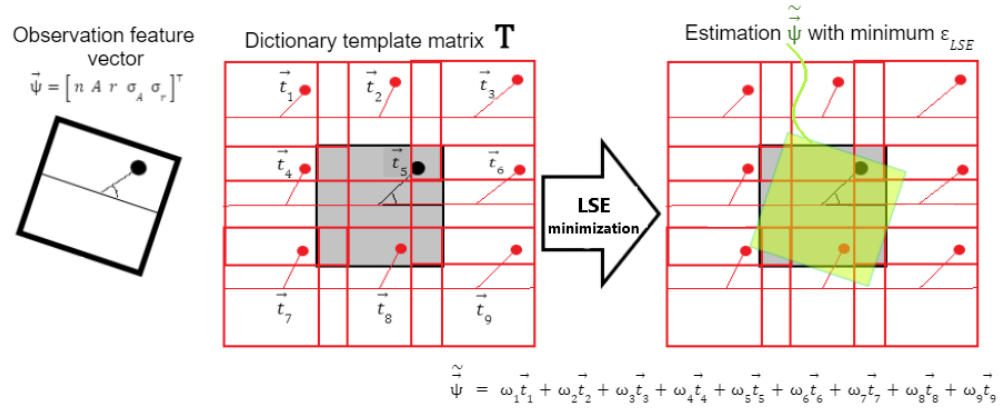

Figure 2.17 : Minimization of the least square error.

regularization parameter, and $\vec{\omega}$ is the regularization weight vector of length M. Furthermore, $\mathscr{P}_a(\vec{\omega})$ is defined as

$$\mathscr{P}_a(\vec{\omega}) = \frac{1-a}{2}\,\|\vec{\omega}\|_2^2 + a\,\|\vec{\omega}\|_1 = \sum_{j=1}^{L}\left(\frac{1-a}{2}\,\omega_j^2 + a\,|\omega_j|\right) \tag{2.33}$$

which enforces the regularization problem into LASSO regression for $a = 1$ by eliminating the part with L_2 norm.

In this study, the regularization problem is enforced into the LASSO regression [89]. Thus, the problem is reduced to the minimization of LSE and L_1 norm by substituting $a = 1$ in Equation 2.32. This study solves the regularization problem by

$$\vec{\omega} = \mathscr{F}_R\left(\mathbf{T}, \hat{\vec{\psi}}\right) = \min_{\vec{\omega}_k}\left(\frac{1}{2M}\sum_{i=1}^{M}\left(\hat{\psi}^i - \mathbf{T}^{i,*}\,\vec{\omega}_k\right)^2 + \lambda\sum_{j=1}^{N}|\omega_k^j|\right) \tag{2.34}$$

where $M = 5$ is the number of features in a feature vector (the number of rows in the dictionary template matrix $\mathbf{T}$), $\hat{\psi}^i$ is the i^{th} component of the normalized observation feature vector, $\mathbf{T}^{i,*}$ is the i^{th} row of the dictionary template matrix, $N = 9$ is the number of feature vectors in the dictionary template matrix $\mathbf{T}$, λ is a non-negative regularization parameter, $\vec{\omega}_k$ is the regularization weight vector of length N, and ω_k^j is the j^{th} component of $\vec{\omega}_k$. Each vector $\vec{\omega}_k$ is obtained by means of the minimization of LSE and L_1 norm where the L_1 norm of $\vec{\omega}_k$ is multiplied by λ and suppressed. In this study, 100 $\vec{\omega}_k$ vectors are obtained ($k = 100$), where λ is incremented by $5 \cdot 10^{-2}$ within the interval $(0,5]$ ($\lambda = 0.05, 0.10, \cdots, 5$). The regularization weight vector $\vec{\omega}_k$ corresponding to the lowest mean squared error (MSE) is considered the solution vector $\vec{\omega}$.

Once the regularization weight vector $\vec{\omega}$ is obtained, the observation vector is reconstructed in the form of

$$\hat{\vec{\psi}} = \mathbf{T}\ \vec{\omega} \tag{2.35}$$

and the estimated boresight vector and the estimated rotation angle are yielded by

$$\hat{\vec{f}} = \mathbf{F}\ \vec{\omega} \tag{2.36}$$

and

$$\hat{\theta} = \Theta\ \vec{\omega} \tag{2.37}$$

where the matrices $\mathbf{T}$, $\mathbf{F}$ and Θ are defined in Equations 2.13, 2.16 and 2.18 respectively.

3. FALSE STAR FILTERING AND CAMERA MOTION ESTIMATION

This chapter focuses on the removal of false stars using a morphological approach. This is followed by the estimation of the camera motion between two time-sequential images. The proposed method is based on a method of unsupervised classification, namely density-based clustering [118]. It uses the isomorphic feature vectors extracted from at least two time-sequential images. The detection of false stars also allows the estimation of translation and rotation motions of the star sensor camera. After centroiding, a set of feature vectors is obtained for each star pair in each frame. A feature vector consists of three elements. These include the mean brightness, the angular separation and the slope of each star pair. A list of disparities is then generated. This list contains the Euclidean distances between each pair of feature vectors, each of which is selected from two temporally consecutive frames. One disparity in the list also contains vectors of three elements. The first two elements in the disparity vector are used to determine the star objects and to label the non-star objects. The third element is also used to retrieve the motion parameters, including translation and rotation. This is achieved by using tolerances corresponding to magnitude and angular separation to account for the presence of noise from displaced stars. A pair is assumed to consist of stellar objects if the pair satisfies criteria enforced by tolerances set using density-based clustering, an unsupervised learning method. The use of an unsupervised learning method ensures a dynamic threshold setting, which allows for a high level of accuracy under varying conditions. The third elements of the disparity vectors corresponding to the star object pairs are checked to see if they are within the given tolerance after the star objects have been detected in the images. An object is labelled as a non-star if it is not involved in any of the pairs meeting the criterion. Finally, an estimate of the rotational camera motion is derived, since it contains information about the change in tilt. The translation motion is also obtained by calculating the difference between the labelled star objects after reversing the rotation motion in the last image frame. The star sensor camera CubeStar [114], which is simulated in the experimental setup of this

study, is used in the SharjahSat-1 [115] project. The effectiveness and accuracy of the method is demonstrated in the empirical results, and the additional CME parameters are used in the development of the RSI method in Chapter 4.

3.1 Problem Statement

The aim of the algorithm is to filter out false stars in a given image frame. An additional algorithm is used to estimate camera motion after estimating true stars to remove false stars. Firstly, the task of FSF is transformed into an unsupervised classification problem, in which the method of density based clustering [118] is used. The problem of optimization is expressed as

$$\mathbf{C}_k\left(\overrightarrow{d}_i\right) = \left\{\overrightarrow{d} \in \mathscr{D} : \|\overrightarrow{d}_i - \overrightarrow{d}_k\|_2 \leq \varepsilon \ , \ \mathrm{n}\{\mathbf{C}_k\} \geq \min_p\right\} \tag{3.1}$$

where C_k is the k^{th} cluster, $\overrightarrow{d}$ is a disparity vector in the disparity list $\mathscr{D}$, $\overrightarrow{d}_k$ is the candidate core vector for the cluster C_k, and $\overrightarrow{d}_i$ is the test vector while ε and $\min_p$ are the parameters of density-based clustering that respectively denote the radius for neighborhood of the vector $\overrightarrow{d}_k$ and the minimum number of vectors in the given neighborhood ε so that C_k is allowed to be called a cluster. Otherwise, the related vectors are considered outliers. Also, note that $\mathrm{n}\{C_k\}$ stands for the number of elements of the cluster C_k, and $\|\cdot\|_2$ for L_2 norm.

The second step is the retrieval of an affine transformation matrix for the CME. This procedure is called MLESAC [119], which uses the same sampling strategy as RANSAC [120] in generating presumptive solutions by randomly selecting a minimal set of observations and evaluating their likelihood until a good solution is found, selecting the solution with the maximum likelihood as opposed to the number of inliers. Within this scope, an affine transformation matrix is retrieved using the relation between the estimated true stars in two subsequent image frames such that

$$\begin{bmatrix} \overrightarrow{c}_i^{t+1} \\ 1 \end{bmatrix} = \tilde{\mathbf{H}} \begin{bmatrix} \overrightarrow{c}_i^{t} \\ 1 \end{bmatrix} \tag{3.2}$$

where $\overrightarrow{c}_i^{t}$ and $\overrightarrow{c}_i^{t+1}$ are the i^{th} centroid vectors of true stars estimated from two time-sequential image frames, and $\tilde{\mathbf{H}} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ 0 & 0 & 1 \end{bmatrix}$ is the estimated affine

transformation matrix, in which $\tilde{\mathbf{R}} = \begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{bmatrix}$ and $\tilde{\mathbf{U}} = \begin{bmatrix} h_{13} \\ h_{23} \end{bmatrix}$ account for rotation and translation respectively. A cost function consisting of a distance metric is used for the estimation process. In addition, the RSI algorithm is provided with the necessary recursive information by the performance metrics, including the mean of the distances between the transformed stars and the true stars and their standard deviation.

3.2 Methodology

There are two estimation phases in the proposed method. The first phase, given by equation 3.1, estimates true stars in an input pair of two temporally sequential images, while filtering out false stars by marking them as outliers. The second phase, given by equations 3.2 and 3.11, uses the estimated true stars to obtain an affine transform matrix to estimate camera motion. The FSF method is based on a subgraph isomorphism approach that is applied to two time-sequential images. This approach requires two sets of feature vectors. These are obtained separately from two time-sequential images. Using the two sets of feature vectors, a list of disparity vectors $\mathscr{D}$ is derived. These are obtained by computing a norm of difference between each pair of feature vectors in two separate sets. Disparity vectors satisfying the given criteria are labelled as star objects, true stars, while those failing the criteria are labelled as non-star objects, false stars. The CME requires the object centroid vectors obtained from two time-sequential image frames $\overrightarrow{c}_i^t$ and $\overrightarrow{c}_i^{t+1}$ to yield the estimated slope angle difference $\Delta\tilde{m}_{ij}$.

Algorithm 2 False star filtering and camera motion estimation

1: $\mathscr{V}^t \leftarrow \mathscr{F}_v\left(\{\overrightarrow{c}_i^t\}, \{A_i^t\}\right)$ (▷) Feature vector set
2: $\mathscr{D} \leftarrow \mathscr{F}_d\left(\mathscr{V}^t, \mathscr{V}^{t+1}\right)$ (▷) Disparity list
3: $C \leftarrow \mathscr{F}_{\text{db}}\left(\mathscr{D}\right)$ (▷) Density-based clustering
4: $\left\{ \overrightarrow{\tilde{s}}_i^{t+1}, \overrightarrow{\tilde{f}}_i^{t+1} \right\} \leftarrow \mathscr{F}_{\text{trace}}\left(C, \mathscr{D}\right)$ (▷) Estimating false stars
5: $\tilde{\mathbf{H}} \leftarrow \mathscr{F}_H\left(\overrightarrow{s}_i^t, \overrightarrow{\tilde{s}}_i^{t+1}\right)$ (▷) Estimating affine transformation matrix

Algorithm 2 shows the whole process. The first line shows the function $\mathscr{F}_v$ which is used to generate the feature vector set $\mathscr{V}^t$ corresponding to the image frame t from the centroid and brightness information of the objects. The second line defines the function $\mathscr{F}_d$, which returns the disparity list $\mathscr{D}$ from the feature vector sets corresponding to two

time-sequential images. The third line reveals the function $\mathscr{F}_{\text{db}}$ which is used to obtain the main cluster **C** from the disparities list. The fourth line defines the function $\mathscr{F}_{\text{trace}}$. This function estimates the false stars $\overrightarrow{\tilde{f}}_i^{t+1}$ and the true stars $\overrightarrow{\tilde{s}}_i^{t+1}$ in the last frame using the cluster **C** and the disparity list $\mathscr{D}$. The fifth line shows the function $\mathscr{F}_H$, which returns the affine transformation matrix $\tilde{\mathbf{H}}$. This is obtained by matching the true stars in the temporal-sequential frames.

3.2.1 Feature extraction and disparity list

A set of feature vectors is extracted for each temporal-sequential image before implementing subgraph isomorphism. As shown in 3.1, the feature vectors contain the slope of the line segments connecting each pair of stars of given centroid and brightness separately for some temporal-sequential test images.

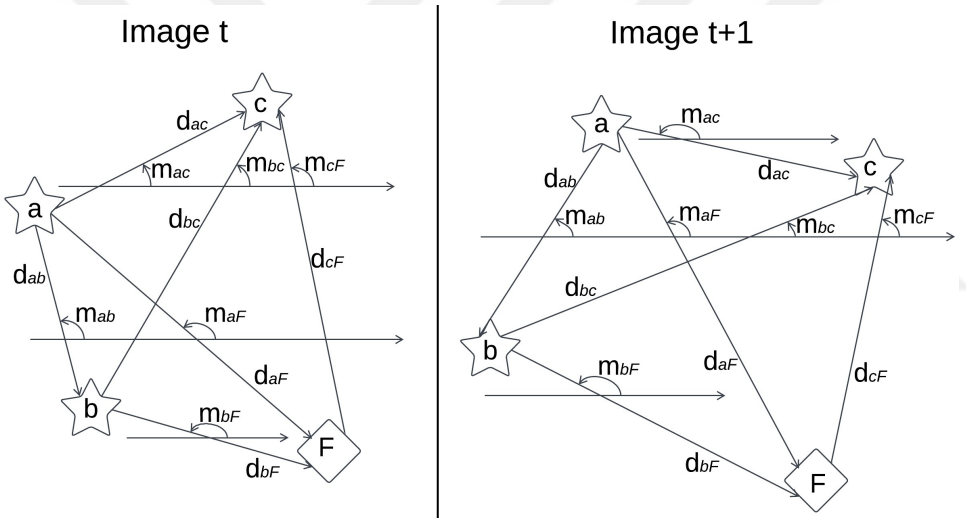

Figure 3.1 : Pairs of stars in two temporal-sequential test images.

Note that the objects labeled as a, b and c are star objects, whereas the object labeled as F is assumed to be a non-star object in 3.1. The lengths d_{ab}, d_{ac}, d_{bc}, d_{aF}, d_{bF} and d_{cF} represent the angular separations between the corresponding objects, and m_{ab}, m_{ac}, m_{bc}, m_{aF}, m_{bF} and m_{cF} denote the slope of the corresponding line segments with respect to a line assumed to be horizontally aligned with respect to the image frame plane. Thus, a feature vector $\overrightarrow{v}_k$ is defined such that

$$\overrightarrow{v}_k \triangleq \begin{bmatrix} A_{ij} \\ d_{ij} \\ m_{ij} \end{bmatrix} \tag{3.3}$$

where A_{ij}, d_{ij} and m_{ij} respectively denote the average brightness of the object pair, angular separation of the object pair and slope angle of the line connecting the pair. The feature A_{ij} is defined as

$$A_{ij} \triangleq \frac{A_i + A_j}{2} \tag{3.4}$$

where A_i and A_j are the brightness values of the objects i and j. The feature d_{ij} is defined as

$$d_{ij} \triangleq \| \overrightarrow{c}_i - \overrightarrow{c}_j \|_2 \tag{3.5}$$

where $\overrightarrow{c_i}$ and $\overrightarrow{c_j}$ are the centroids of the objects i and j, and $\| \cdot \|_2$ is the L_2 norm, and the feature m_{ij} is defined as

$$m_{ij} \triangleq \arctan \overrightarrow{c_i c_j} \tag{3.6}$$

where $\overrightarrow{c_i c_j}$ is the slope of the angle of the angular distance d_{ij}. In each image frame with N_o detected objects, the number of pairs of stars is computed by

$$L = \binom{N_o}{2} = \frac{N_o}{(N_o - 2) \cdot 2!} \tag{3.7}$$

which yields L, the number of feature vectors for each image frame. An image frame can now be represented by a set of feature vectors defined as

$$\mathscr{V}^t \triangleq \left\{ \overrightarrow{v}_i^t \right\} \tag{3.8}$$

which contains every feature vector extracted from the image frame at time t.

The disparity list $\mathscr{D}$ contains a set of disparity vectors $\overrightarrow{d}_k$ that can be generated using two sets of feature vectors, $\mathscr{V}^t$ and $\mathscr{V}^{t+1}$, obtained separately from two time-sequential image frames. For all possible combinations of the pairs, the differentiation operation is performed in pairs. Thus, a disparity vector is obtained by

$$\overrightarrow{d}_k \triangleq \overrightarrow{v}_i^t - \overrightarrow{v}_j^{t+1} = \begin{bmatrix} \Delta A_{ij} \\ \Delta d_{ij} \\ \Delta m_{ij} \end{bmatrix} \tag{3.9}$$

where ΔA_{ij}, Δd_{ij} and Δm_{ij} respectively correspond to difference of magnitudes, difference of angular separations and rotation. The number of vectors in the disparity list $\mathscr{D}$ is given by

$$D = L_1 \cdot L_2 \tag{3.10}$$

where L_1 and L_2 are the number of feature vectors in the sequential image frames.

3.2.2 Detection of false stars

The density clustering algorithm [118] is implemented on the disparity list $\mathscr{D}$ in such a way that the disparity vectors corresponding to the true star pairs will be grouped in the same cluster, i.e. the cluster of disparity vectors corresponding to the true stars. The rest of the disparity vectors are dropped from the cluster. The algorithm first selects an unlabelled disparity vector $\overrightarrow{d}_l$ from the disparity list $\mathscr{D}$ as the current vector. This is used to initialise the cluster C. Then the vectors within the ε neighbourhood of $\overrightarrow{d}_l$ are detected. These vectors form a set of neighbours. If the number of vectors in the set of neighbours is less than $\min_p$, $\overrightarrow{d}_l$ is called an outlier. Otherwise, $\overrightarrow{d}_l$ is said to be a core vector which belongs to the cluster C. This operation is repeated for each of the neighbour vectors until no new neighbour vector can be found that belongs to the cluster C. In the case that a vector is labelled as an outlier, a new vector is selected as the current vector to continue the procedure until all the vectors in $\mathscr{D}$ have been iterated. Since the labels of each object detected in the image frame are preserved as pairs in the disparity vectors, the true stars are easily extracted from the cluster C.

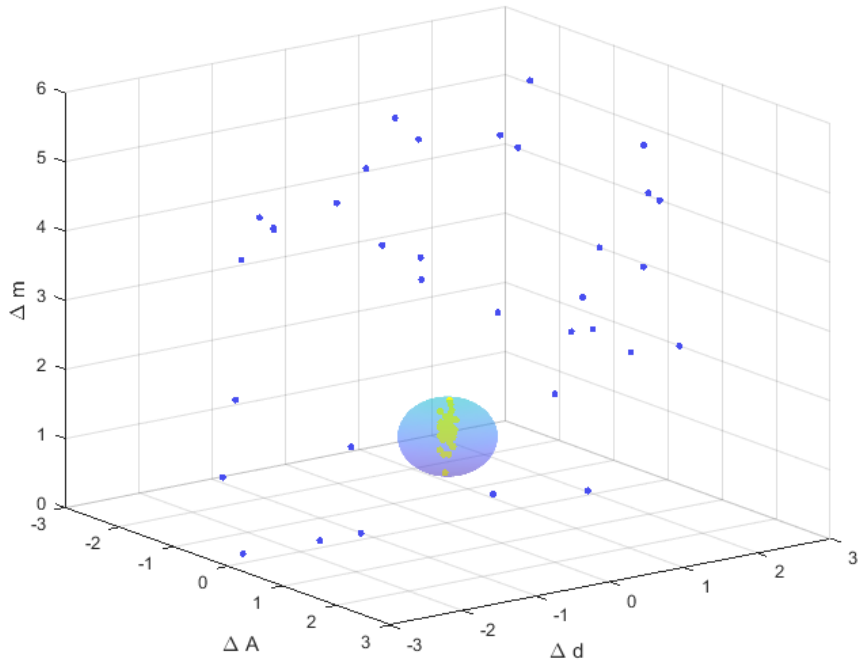

Figure 3.2 : The cluster of disparity vectors corresponding to true stars, which is generated by the density-based clustering algorithm.

Figure 3.2 shows an example of a cluster containing a set of disparity vectors within the disparity list $\mathscr{D}$. Each point in the figure represents a disparity vector $\overrightarrow{d}_i$. This vector corresponds to all combinations of star pairs in the time-sequential frames. The space of the plot spans three dimensions including δA, δd and δm. The yellow points bounded by the blue sphere of ε correspond to the true star pairs. Other points outside the sphere are labelled as outliers. Thus, the objects corresponding to the cluster $\mathbf{C}$ obtained according to Equation 3.1 are considered true stars, while others are outliers or false stars.

3.2.3 Camera motion estimation

The CME algorithm is implemented using the true stars obtained by the FSF algorithm. An affine transformation matrix transforms the centroids of the true stars in frame t into the centroids of the true stars in frame $t+1$. The equation 3.2 is used to estimate the affine transformation matrix. The accuracy of the estimate is determined by the proximity of the transformed stars to the true stars. The translation and rotation information is included in the estimated affine transformation matrix. In accordance with the MLESAC procedure, the process of estimation of $\tilde{\mathbf{H}}$ is subject to a cost function defining a distance metric

$$l = \sum_{i}^{N} \min \| \tilde{\overrightarrow{c}}_i^{t+1} - \overrightarrow{c}_i^{t+1} \|_2 \text{ subject to } \| \tilde{\overrightarrow{c}}_i^{t+1} - \overrightarrow{c}_i^{t+1} \|_2 \leq \eta \tag{3.11}$$

where $\tilde{\overrightarrow{c}}_i^{t+1}$ is the projection of $\overrightarrow{c}_i^{t}$ based on $\tilde{\mathbf{H}}$, and η is a threshold in accordance with $\begin{bmatrix} \tilde{\overrightarrow{c}}_i^{t+1} \\ 1 \end{bmatrix} = \tilde{\mathbf{H}} \begin{bmatrix} \overrightarrow{c}_i^{t} \\ 1 \end{bmatrix}$. Once the estimated affine transformation matrix $\tilde{\mathbf{H}}$ is obtained, the rotation angle can be retrieved in addition to $\tilde{\mathbf{R}}$ and $\tilde{\mathbf{U}}$.

$$\tilde{\mathbf{R}} \triangleq \begin{bmatrix} \cos\tilde{\gamma} & -\sin\tilde{\gamma} \\ \sin\tilde{\gamma} & \cos\tilde{\gamma} \end{bmatrix} \tag{3.12}$$

Using the definition of rotation given in Equation 3.12, the estimated rotation angle $\tilde{\gamma}$ can be obtained.

4. RECURSIVE STAR IDENTIFICATION METHOD

This chapter presents the RSI algorithm. The algorithm is based on the LSI algorithm, previously proposed in the thesis. It differs by an update mechanism integrated by means of the motion estimation outputs retrieved from the preprocessing phase which implements the CME algorithm.

4.1 Contributions and Limitations

The RSI algorithm is based on the dictionary-based matching in the same manner as the LSI algorithm except that it is supported by an update mechanism. The update mechanism allows the RSI algorithm to reduce computational complexity and run time in comparison with the LSI algorithm without compromising accuracy. Thus, thanks to the integration of the FSF algorithm, it achieves much higher accuracy than the LSI algorithm, especially in the presence of false stars in the scene. Furthermore, using the motion parameters yielded by the CME algorithm, it shrinks the database so that the computational complexity is reduced significantly, which also helps run time of the coarse estimation be decreased to insignificant levels compared to the run time of fine estimation.

4.2 Methodology

The RSI algorithm works in collaboration with the algorithms proposed in the previous chapters of this thesis. The algorithm uses the same feature extraction method as the LSI algorithm. It is also based on the successive implementation of coarse estimation and fine estimation. While the 1NN classifier is used in the coarse estimation, the LASSO regression accounts for fine estimation. It, in addition to the baseline algorithm, benefits from the FSF algorithm and the CME algorithm to further increase accuracy and significantly lower run time. Since the LSI algorithm is highly sensitive to false stars, the FSF algorithm is developed to overcome this

issue. Also, the CME algorithm implemented subsequent to the FSF algorithm in the preprocessing phase provides the RSI algorithm with motion parameters. Hence, the RSI algorithm is dependent on an update mechanism through a threshold. The reconstructed observation error ε_{ψ} is tracked, and the update mechanism is activated in case it exceeds a predetermined error threshold $\tau_{\varepsilon_{\psi}}$.

Algorithm 3 All algorithms integrated

for $t = 0 : T$ **do** $(\triangleright)\ T \to \infty$
  if $t = 0$ **then** $(\triangleright)$ Capture the first image
    $\left[\overrightarrow{\psi}^{0}\right] \leftarrow \mathscr{F}_{\psi}\left(\mathbf{I}^{0}\right)$ $(\triangleright)$ Get observation feature vector
    $\left[\overrightarrow{\tilde{f}}^{0}, \overrightarrow{\tilde{\theta}}^{0}, \varepsilon_{\psi}^{0}\right] \leftarrow \mathscr{F}_{\mathrm{LSI}}\left(\overrightarrow{\psi}^{0}, \left\{\overrightarrow{\tau}_{i}\right\}\right)$ $(\triangleright)$ First estimation w/o FSF&CME
  else if $t \neq 0$ **then** $(\triangleright)$ Subsequent images captured
    $\left[\tilde{\mathbf{I}}^{t}\right] \leftarrow \mathscr{F}_{\mathrm{FSF}}\left(\mathbf{I}^{t}\right)$ $(\triangleright)$ False star filtering
    $\left[\tilde{\mathbf{H}}^{t}\right] \leftarrow \mathscr{F}_{\mathrm{CME}}\left(\tilde{\mathbf{I}}^{t}, \mathbf{I}^{t-1}\right)$ $(\triangleright)$ Motion estimation
    $\left[\left\{\overrightarrow{\tilde{\tau}}_{j}\right\}\right] \leftarrow \mathscr{F}_{\mathrm{RoS}}\left(\tilde{\mathbf{H}}^{t}, \left\{\overrightarrow{\tau}_{i}\right\}\right)$ $(\triangleright)$ Database shrinkage
    $\left[\overrightarrow{\psi}^{t}\right] \leftarrow \mathscr{F}_{\psi}\left(\tilde{\mathbf{I}}^{t}\right)$ $(\triangleright)$ Get observation feature vector
    $\left[\overrightarrow{\tilde{f}}^{t}, \overrightarrow{\tilde{\theta}}^{t}, \varepsilon_{\psi}^{t}\right] \leftarrow \mathscr{F}_{\mathrm{RSI}}\left(\overrightarrow{\psi}^{t}, \left\{\overrightarrow{\tilde{\tau}}^{t}_{j}\right\}\right)$ $(\triangleright)$ RSI estimation
    if $\varepsilon_{\psi}^{t} > \tau_{\varepsilon_{\psi}}$ **then** $(\triangleright)$ If error threshold exceeded
      $\left[\overrightarrow{\tilde{f}}^{t}, \overrightarrow{\tilde{\theta}}^{t}, \varepsilon_{\psi}^{t}\right] \leftarrow \mathscr{F}_{\mathrm{LSI}}\left(\overrightarrow{\psi}^{t}, \left\{\overrightarrow{\tau}_{i}\right\}\right)$ $(\triangleright)$ Switch to LSI estimation
    end if
  end if
  if $\varepsilon_{\psi}^{t} > K\ \tau_{\varepsilon_{\psi}}$ **then** $(\triangleright)$ In case of very large error
    Invalid estimation
  end if
end for

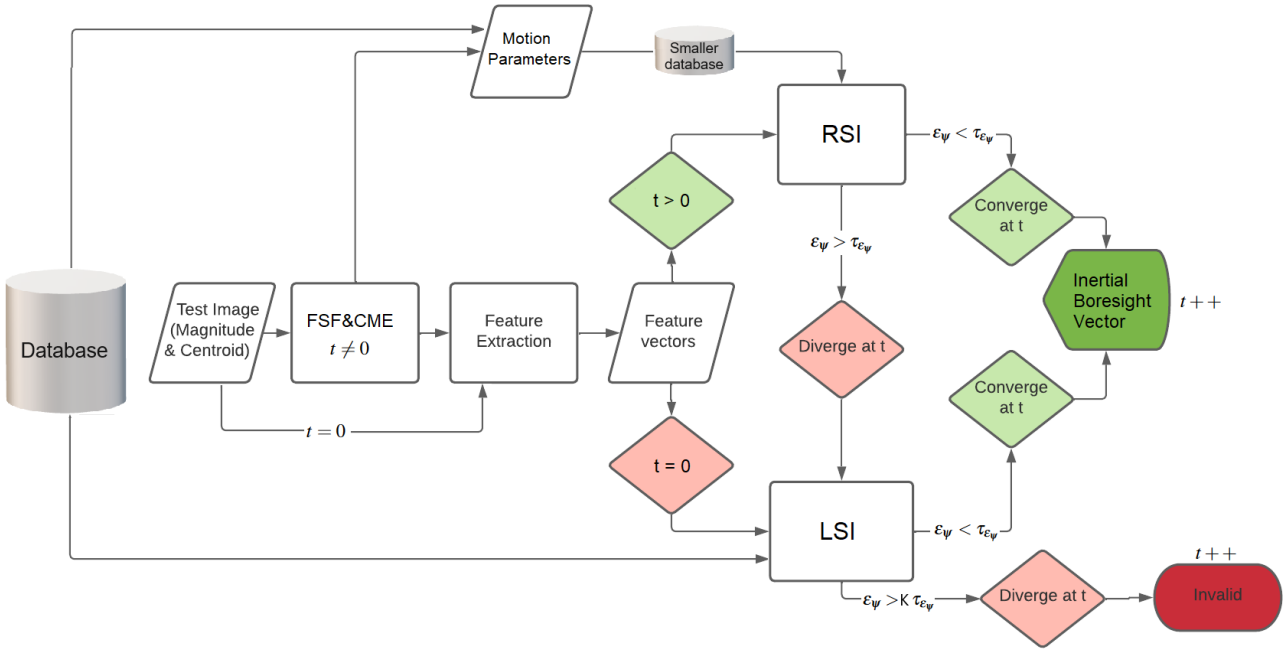

Figure 4.1 : The flowchart of the RSI algorithm.

The whole algorithm is described in Algorithm 3. The first estimation is performed in accordance with the LSI algorithm since there is no a priori information. As from the acquisition of the second image frame at time $t = 1$, the proposed preprocessing is employed. First, the false stars are filtered out using the FSF algorithm $\mathscr{F}_{\text{FSF}}$. It is followed by the CME algorithm $\mathscr{F}_{\text{FSF}}$, which yields the motion parameters. Next, the function $\mathscr{F}_{\text{RoS}}$ utilizes the estimated affine transformation $\tilde{\mathbf{H}}^t$ to shrink the database. The input of $\mathscr{F}_{\text{RoS}}$ is all of the feature vectors in the database $\overrightarrow{\tau_i}$ while the output is the shrunk database $\overrightarrow{\tilde{\tau}}^t_j$ where $j \ll i$. After obtaining the observation feature vector $\overrightarrow{\psi}^t$, the RSI algorithm is implemented using the function $\mathscr{F}_{\text{RSI}}$. In addition to the estimates $\overrightarrow{\tilde{f}}^t$ and $\overrightarrow{\tilde{\theta}}^t$, the error for the reconstructed feature vector ε^t_ψ is given by the RSI algorithm. After retrieval of the estimations, it is checked whether ε^t_ψ exceeds the predetermined error threshold τ_{ε_ψ}. In case of a larger error, the estimaitons and the error is updated by the LSI algorithm $\mathscr{F}_{\text{LSI}}$. Finally, it is checked whether the error is very large, larger than $K \cdot \tau_{\varepsilon_\psi}$. If it is so, the estimation at t is considered a valid estimation. The whole algorithm is depicted in Figure 4.1 for a better visualization.

4.2.1 Update mechanism

The update mechanism is based on two steps. The first step includes the shrinkage of the database, and the second step conditionally enforces the use of the whole database instead of the shrunk one. The update mechanism decides that the estimation of the RSI algorithm is not safe enough in case the condition $\varepsilon_\psi < \tau_{\varepsilon_\psi}$ is not met. Thus, the estimation is recurred with the whole database in accordance with the LSI algorithm. A similar update mechanism based on thresholding by an intrinsic error measurement has been developed in an object detection algorithm [90]. Database shrinkage is performed by the function $\mathscr{F}_{\text{RoS}}$, which is derived by Equations 3.2 and 3.11 described in Section 3.2.3. And, the reliability of the estimation is checked by the following conditional test

$$\left\{\overrightarrow{\tau}^t\right\} = \begin{cases} \left\{\overrightarrow{\tau}^t_j\right\} & , \varepsilon^t_\psi < \tau_{\varepsilon_\psi} \\ \left\{\overrightarrow{\tau}_i\right\} & , \varepsilon^t_\psi \geq \tau_{\varepsilon_\psi} \end{cases} \tag{4.1}$$

where the database used for the final estimation is denoted by $\left\{\overrightarrow{\tau}^t\right\}$. $\left\{\overrightarrow{\tau}_i\right\}$ represents the whole database, which is shrunk to $\left\{\overrightarrow{\tau}^t_j\right\}$ to be used by the RSI algorithm upon status of the given condition.

4.2.2 Determination of scan region

The scan region is a smaller square patch of the whole database. The corresponding database feature vectors to be used by the RSI algorithm are picked from the scan region. The center of the scan region is determined by the database vector with the corresponding degrees in the catalog, pointed by the estimated affine transformation matrix $\tilde{\mathbf{H}}^t$. The reliability of $\tilde{\mathbf{H}}^t$ is measured by distance errors of the CME algorithm under different noise scenarios. The values of mean and standard deviation are also computed for the distance errors. The consistency is also ensured by investigating the correlation coefficient and p-value between mean values and standard deviation values. The correlation coefficient of two random variables is a measure of their linear dependence [121,122], given by

$$\rho\left(\mu_d, \sigma_d\right) = \frac{1}{n-1} \sum_{i=1}^{n} \left(\frac{\mu_d^i - \mu_{\mu_d}}{\sigma_{\mu_d}} \right) \left(\frac{\sigma_d^i - \mu_{\sigma_d}}{\sigma_{\sigma_d}} \right) \tag{4.2}$$

where μ_d and σ_d respectively denote mean and standard deviation of distance errors. The empirical values given in Table 6.5 imply a strong correlation between μ_d and σ_d and a low probability of observing null hypothesis, a claim of observing no correlation between the random variables. Thus, since μ_d and σ_d are correlated, the size of the RoS square can be determined consistently in addition to determination of the center. One side of the RoS square is determined as per Equation 6.5 after the behavior of the RoS is investigated empirically. The details are provided in Section 6.3.

5. SIMULATION ENVIRONMENT

A simulation environment is designed and developed using MATLAB so that the experiments could be carried out. The simulation tool allows creation of star objects with the given values of centroid and brightness values in the given coordinates. It also enables injection of noise including position noise, brightness noise, false stars and missing stars. In addition to test images, the images assumed to be used in the database are also created using the simulation environment. Therefore, the simulation tool is capable of adjusting the parameters regarding the sensor such as FoV and resolution as well as the parameters required to add noise into the images. The noise parameters are selected using the probabilistic distribution functions such as normal distribution and uniform distribution.

5.1 Database and observation

A number of experiments are carried out with database feature vectors extracted from the catalog [71] using a series of overlap ratios $p = 0.96$, $p = 0.97$, $p = 0.98$, $p = 0.99$, $p = 0.991$, $p = 0.992$, $p = 0.993$, $p = 0.994$ and $p = 0.995$ with a magnitude threshold $\kappa = 11.5$. A bunch of test vectors are extracted from the catalog bounded in the region $\alpha \in [95.0^\circ, 137.0^\circ]$ and $\delta \in [-85.0^\circ + k \cdot 10^\circ, -43.0^\circ + k \cdot 10^\circ]$ for $k = 0, \cdots, 24$. 25 test feature vectors are extracted for the experiment. Each observation feature vector is normalized to eliminate domination of large-value elements in the phase of binary search by Euclidean distance. After normalization, the database vector closest to the observation vector is detected through the 1NN classification. Next, the dictionary is created by merging the neighbor vectors into the dictionary. And then, the LASSO regularization is performed to estimate the regularization coefficients.

An example is shown hereinafter. A database of feature vectors is generated with an overlap value $p = 0.96$ and a magnitude threshold $\kappa = 11.5$, which implies that the sky patches from which the feature vectors are obtained overlap by 96% and it is assumed

that the stars with a visual magnitude value greater than 11.5 M_V are detected by the star sensor in accordance with the capability of the sensor with a FoV $42^\circ \times 42^\circ$. The observation vector is assumed to be captured by the sensor covering the region ranging within right ascension $\alpha \in [95.0^\circ, 137.0^\circ]$ and declination $\delta \in [-85.0^\circ, -43.0^\circ]$. The observation feature vector corresponding to the observation image is $\overrightarrow{\psi}$, which is normalized to get $\hat{\overrightarrow{\psi}}$ as given below in Equation 5.1.

$$\overrightarrow{\psi} = \begin{pmatrix} n \\ A \\ r \\ \sigma_A \\ \sigma_r \end{pmatrix} = \begin{pmatrix} 26 \\ 11.96 \\ 2.95 \\ 0.37 \\ 5.35 \end{pmatrix} \xrightarrow{normalized} \hat{\overrightarrow{\psi}} = \begin{pmatrix} \hat{n} \\ \hat{A} \\ \hat{r} \\ \hat{\sigma_A} \\ \hat{\sigma_r} \end{pmatrix} = \begin{pmatrix} 26.60 \\ 43.59 \\ 20.03 \\ 47.75 \\ 64.41 \end{pmatrix} \tag{5.1}$$

Note that the polar angle feature θ is not normalized and separated from the feature vector to be used in the approximation of the estimated rotation angle $\tilde{\theta}$.

After normalizing both the observation feature vector and the database feature vectors, the 1NN classification is performed and the classifier detects that the 735^{th} database feature vector $\overrightarrow{\tau}_{735} = \begin{pmatrix} 26 & 11.96 & 3.54 & 0.37 & 5.39 \end{pmatrix}^T$ has the highest similarity with the observation feature vector $\overrightarrow{\psi}$. The neighboring database feature vectors are picked into the dictionary template. Thus, the dictionary contains 509^{th}, 510^{th}, 511^{th} database feature vectors at the top row, 734^{th}, 735^{th}, 736^{th} database feature vectors in the middle row and 959^{th}, 960^{th}, 961^{th} database feature vectors at the bottom row. The dictionary template is shown in Figure 5.1. Thus, the vectors in the dictionary template matrix $\mathbf{T}$ are assigned to be $\overrightarrow{t}_1 = \hat{\overrightarrow{\tau}}_{509}$, $\overrightarrow{t}_2 = \hat{\overrightarrow{\tau}}_{510}$, $\overrightarrow{t}_3 = \hat{\overrightarrow{\tau}}_{511}$, $\overrightarrow{t}_4 = \hat{\overrightarrow{\tau}}_{734}$, $\overrightarrow{t}_5 = \hat{\overrightarrow{\tau}}_{735}$, $\overrightarrow{t}_6 = \hat{\overrightarrow{\tau}}_{736}$, $\overrightarrow{t}_7 = \hat{\overrightarrow{\tau}}_{959}$, $\overrightarrow{t}_8 = \hat{\overrightarrow{\tau}}_{960}$ and $\overrightarrow{t}_9 = \hat{\overrightarrow{\tau}}_{961}$.

Figure 5.1 shows that the database feature vector closest to the observation feature vector is extracted from the region of sky patch bounded by right ascension $\alpha \in [94.4^\circ, 136.4^\circ]$ and $\delta \in [-85.2^\circ, -43.2^\circ]$ by means of 1NN classifier. Also, the angular values of right ascension and declination are given in yellow and the corresponding feature vectors are given in green in the figure. Thus, the dictionary template $\mathbf{T}$ is obtained where the columns are represented by the database feature vectors corresponding to the sky patch image in the order of Figure 5.1 such

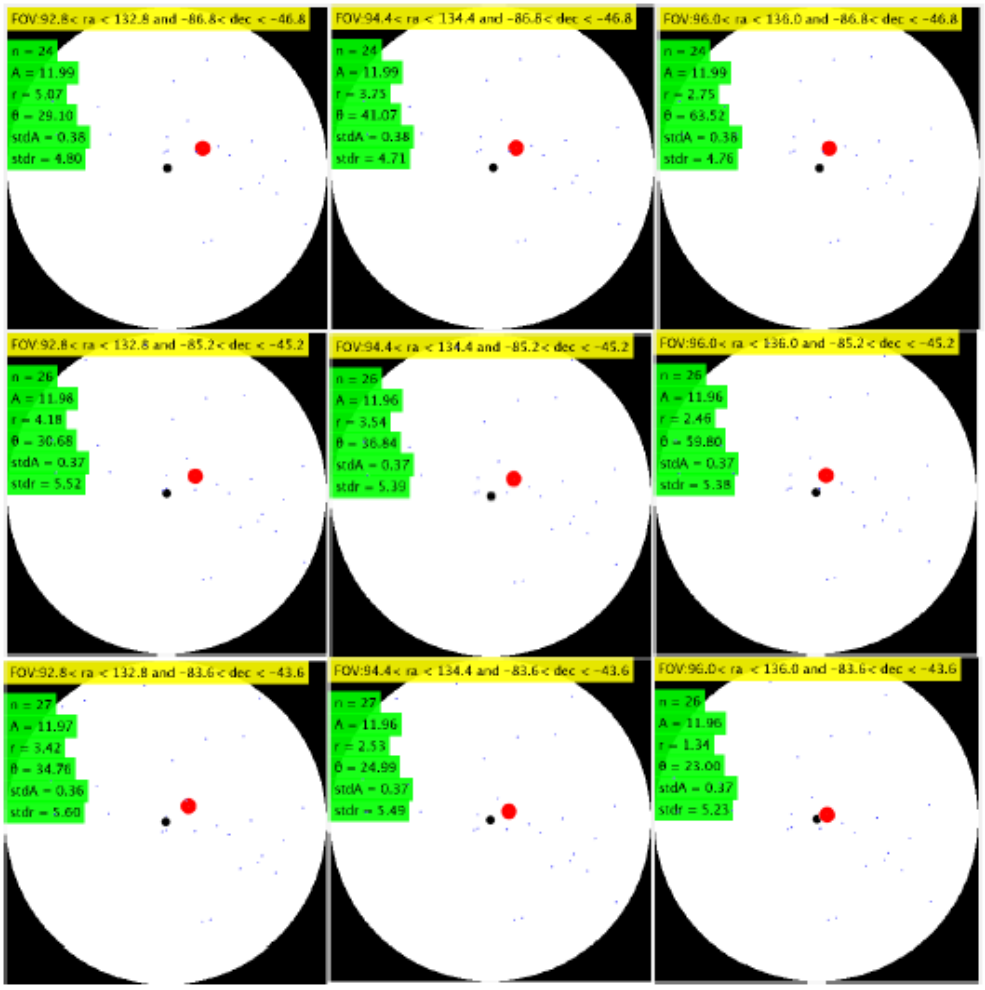

Figure 5.1 : An example of dictionary template.

that $\overrightarrow{\tau}_{509} = \begin{pmatrix} 24 \\ 11.99 \\ 5.07 \\ 0.38 \\ 4.80 \end{pmatrix}$, $\overrightarrow{\tau}_{510} = \begin{pmatrix} 24 \\ 11.99 \\ 3.75 \\ 0.38 \\ 4.71 \end{pmatrix}$, $\overrightarrow{\tau}_{511} = \begin{pmatrix} 24 \\ 11.99 \\ 2.75 \\ 0.38 \\ 4.76 \end{pmatrix}$, $\overrightarrow{\tau}_{734} = \begin{pmatrix} 26 \\ 11.98 \\ 4.18 \\ 0.37 \\ 5.52 \end{pmatrix}$, $\overrightarrow{\tau}_{735} = \begin{pmatrix} 26 \\ 11.96 \\ 3.54 \\ 0.37 \\ 5.39 \end{pmatrix}$, $\overrightarrow{\tau}_{736} = \begin{pmatrix} 26 \\ 11.96 \\ 2.46 \\ 0.37 \\ 5.38 \end{pmatrix}$, $\overrightarrow{\tau}_{959} = \begin{pmatrix} 27 \\ 11.97 \\ 3.42 \\ 0.36 \\ 5.60 \end{pmatrix}$, $\overrightarrow{\tau}_{960} = \begin{pmatrix} 27 \\ 11.96 \\ 2.53 \\ 0.37 \\ 5.49 \end{pmatrix}$ and $\overrightarrow{\tau}_{961} = \begin{pmatrix} 26 \\ 11.96 \\ 1.34 \\ 0.37 \\ 5.23 \end{pmatrix}$. Moreover, the rotation template matrix Θ is obtained with the corresponding polar angles such that $\theta_{509} = 29.10°$, $\theta_{510} = 41.07°$, $\theta_{511} = 63.52°$,

$\theta_{734} = 30.68^\circ$, $\theta_{735} = 36.84^\circ$, $\theta_{736} = 59.80^\circ$, $\theta_{959} = 34.76^\circ$, $\theta_{960} = 24.99^\circ$ and $\theta_{961} = 23.00^\circ$.

After performing the LASSO regularization, two types of regularization coefficient vector are computed. While $\vec{\omega}_{MSE}$ is the weight vector corresponding to the λ value with the minimum MSE, $\vec{\omega}_{1SE}$ is the weight vector corresponding to the λ value such that the MSE is within 1SE of the minimum MSE, where λ is defined in Equation 2.34. They are obtained such that $\vec{\omega}_{MSE} = \begin{pmatrix} 0 \\ 0 \\ 0.04 \\ 0 \\ 0.45 \\ 0.26 \\ 0 \\ 0.26 \\ 0 \end{pmatrix}$ and $\vec{\omega}_{1SE} = \begin{pmatrix} 0 \\ 0 \\ 0.03 \\ 0 \\ 0.44 \\ 0.26 \\ 0 \\ 0.27 \\ 0 \end{pmatrix}$ is yielded in this example. Next, using the regularization weight vector $\vec{\omega}_{MSE}$ or $\vec{\omega}_{1SE}$, the estimated feature vector can be reconstructed from the dictionary template matrix $\mathbf{T}$. The estimated feature vectors $\tilde{\vec{\psi}}_{MSE}$ and $\tilde{\vec{\psi}}_{1SE}$ obtained in this example such that $\tilde{\vec{\psi}}_{MSE} = \mathbf{T} \times \vec{\omega}_{MSE} = \begin{pmatrix} 24.71 \\ 11.94 \\ 2.80 \\ 38.37 \\ 0.35 \\ 5.08 \end{pmatrix}$ and $\tilde{\vec{\psi}}_{1SE} = \mathbf{T} \times \vec{\omega}_{1SE} = \begin{pmatrix} 24.14 \\ 11.94 \\ 2.72 \\ 37.06 \\ 0.34 \\ 4.95 \end{pmatrix}$ where $\vec{\omega}$ has the elements corresponding to the feature vectors in the dictionary template with the given locations from top left to bottom right in the order given in Figure 5.1. The norm of the difference between the observation feature vector $\vec{\psi} = \begin{pmatrix} 26 \\ 11.96 \\ 2.95 \\ 40.72 \\ 0.37 \\ 5.35 \end{pmatrix}$ and the estimated feature vector yields both types of error such that $\varepsilon_{\psi_{MSE}} = \|\vec{\psi} - \tilde{\vec{\psi}}_{MSE}\| = 2.69$ and $\varepsilon_{\psi_{1SE}} = \|\vec{\psi} - \tilde{\vec{\psi}}_{1SE}\| = 4.13$. Finally, a new template is generated using the FoV values of the dictionary template feature vectors where each column corresponds to the relevant feature vector with the corresponding values of right ascension α and declination δ such that $\mathbf{F} = \begin{pmatrix} \alpha_1 & \alpha_2 & \alpha_3 & \alpha_4 & \alpha_5 & \alpha_6 & \alpha_7 & \alpha_8 & \alpha_9 \\ \delta_1 & \delta_2 & \delta_3 & \delta_4 & \delta_5 & \delta_6 & \delta_7 & \delta_8 & \delta_9 \end{pmatrix}$. Thus, the estimated boresight vector $\tilde{\vec{f}} = \begin{pmatrix} \tilde{\alpha} \\ \tilde{\delta} \end{pmatrix}$ can be obtained by a cross product such that

$\tilde{\vec{f}}_{MSE} = \mathbf{F} \times \vec{\omega}_{MSE} = \begin{pmatrix} 94.88 \\ -84.85 \end{pmatrix}$ and $\tilde{\vec{f}}_{1SE} = \mathbf{F} \times \vec{\omega}_{1SE} = \begin{pmatrix} 94.86 \\ -84.81 \end{pmatrix}$. Note that the bottom left point of the FoV is taken as a reference in calculation of the values. Lastly, the error of estimated boresight vector is calculated through the derivation of norm of difference between the estimated boresight vector $\tilde{\vec{f}}$ and the observed boresight vector $\vec{f}$ such that $\varepsilon_{f_{MSE}} = \| \vec{f} - \tilde{\vec{f}}_{MSE} \| = 0.1983^\circ$ and $\varepsilon_{f_{1SE}} = \| \vec{f} - \tilde{\vec{f}}_{1SE} \| = 0.2347^\circ$. The algorithm finally yields the estimated angular values for the observed star image. The accuracy of the estimation can be measured with either $\varepsilon_{f_{MSE}}$ or $\varepsilon_{f_{1SE}}$ on the type of error used in the estimation.

5.2 Noise injection

A star sensor is exposed to several types of noise in the course of image acquisition and detection of body vectors. The factors leading to noise include any neighboring object, satellite or dust particles, activation of some pixels saturated due to radiation in the space environment and dislocation of star images due to thermal deformation or optical flaws, as well as dead pixels or blockage of FoV. Major types of noise encountered are exemplified in Figure 5.2. The types of input noise, to which a star sensor is subject, include positional noise, missing and false stars as well as magnitude noise [73]. Thermal deformation, optical flaws or calibration errors may stimulate positional noise, and the sensitivity of a sensor accounts for magnitude noise. Furthermore, false stars and missing stars are respectively triggered by reflecting objects in the FoV and solar flare and blockages in the FoV and dead pixels [84].

Figure 5.2 : Types of noise encountered in star sensors.

The simulated images are generated using the same sensor model [114] given in Table 2.2. While a Gaussian noise model is used for both positional and magnitude noise, false stars and missing stars are selected within a simulated observation image

arbitrarily using a uniform probabilistic distribution. A pixel in a given simulated image covers a region of FoV such that

$$\text{FoV}_p = \frac{42^\circ}{937} \times \frac{42^\circ}{937} \approx 0.0448^\circ/pixel \times 0.0448^\circ/pixel \tag{5.2}$$

since the sensor has a FoV $42^\circ \times 42^\circ$ and a resolution 937×937 pixels according to Table 2.2.

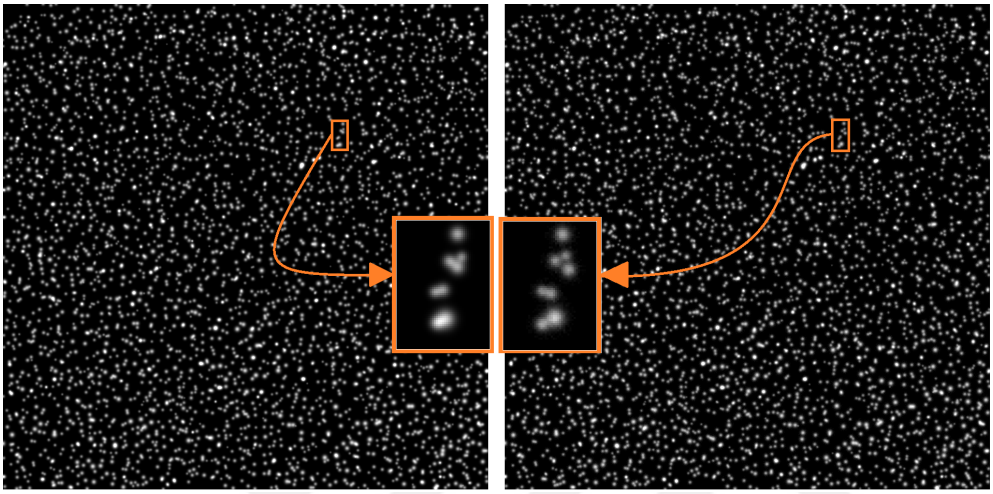

Figure 5.3 : A pair of star images obtained from the Hipparcos database without noise at the left-hand side and with noise added at the right-hand side.

Figure 5.3 shows an image of stars at the left-hand side without noise and an image of stars with noise added at the right-hand side to exemplify the procedure of noise injection. The images are simulated with a lower bound brightness threshold $\kappa = 3.8$, a FoV $42^\circ \times 42^\circ$ and a resolution 937×937 pixels. The image is located within an interval of right ascension $\alpha \in [0^\circ, 42^\circ]$ and declination $\delta \in [0^\circ, 42^\circ]$, where two types of noise are injected: positional noise and magnitude noise. In order to observe the effects of noise addition, a smaller patch is extracted from the image and zoomed in. The patch has a FoV $2^\circ \times 2.7^\circ$ and a resolution 45×62 pixels, which is located within $\alpha \in [28.1^\circ, 30.1^\circ]$ and $\delta \in [29.2^\circ, 31.9^\circ]$ within the whole image. The stars are created by generating a 2D Gaussian function, center of which is located on the position of the star given in the catalog, that is, the mean and the variance of the function are the exact location and the magnitude of the star respectively. The right-hand side image is created by adding positional noise and magnitude noise to each star object separately. The addition of positional noise is carried out by imposing a shift effect

on the centroid of a star through an individual Gaussian function along horizontal and vertical axes. In this image sample, the amount of positional noise is determined by a normal distribution $\mathcal{N}\left(\mu_p, {\sigma_p}^2\right)$ where $\mu_p = 0$ and $\sigma_p = 1$ in terms of pixels. The amount of brightness noise is also determined by a normal distribution $\mathcal{N}\left(\mu_m, {\sigma_m}^2\right)$ where $\mu_m = 0$ and $\sigma_m = 5$ in terms of pixel intensity. The effect of both types of noise is clearly revealed in the enlarged patch, where the stars' positions are shifted and the pure Gaussian shape of the stars in the left-hand side are degraded in the image at the right-hand side. On the other hand, the simulation also allows injection of false stars into an image and removal of true stars from an image. A false star is injected into an image using two uniform distributions, one to determine the position within a given FoV where $\mathcal{U}\left(\alpha_0, \alpha_1\right)$ and $\mathcal{U}\left(\delta_0, \delta_1\right)$ are respectively two uniform distributions that specify the boundaries of right ascension and declination in the given FoV, and the other to determine brightness within a given boundary specified by the characteristic of the sensor where $\mathcal{U}\left(m_L, m_H\right)$ is the uniform distribution specifying the boundaries of brightness. A true star is removed from an image using a uniform distribution as well, where the index of the star to be removed is selected arbitrarily from the set of stars in the image by means of $\mathcal{U}\left(1, n\right)$ where n is the number of true stars in the given FoV.

5.3 Translation and rotation

An image frame is translated and rotated to obtain time-sequential images such that a motion effect of the camera is sustained by inserting the corresponding stars into the image with the selected noise conditions. After the initial image is simulated at time t_0 within a selected region in the given FoV, the subsequent image is simulated by translating the image at $t-1$ by $\begin{bmatrix} x_t \\ y_t \end{bmatrix}$ drawn from a uniform distribution $\mathcal{U}\left(-\overrightarrow{\rho}, \overrightarrow{\rho}\right)$ and rotating it by θ_t drawn from a uniform distribution $\mathcal{U}\left(-\theta, \theta\right)$.

Figure 5.4 shows two sequential images. The left-hand side image is assumed to be captured at time t, which corresponds to the region bounded by $\alpha \in (0^\circ, 42^\circ)$ and $\delta \in (0^\circ, 42^\circ)$ with stars brighter than -12 $\mathrm{M_V}$ and a false star labeled number 3. The right-hand side image at time $t+1$ is simulated by translating the image at time t by

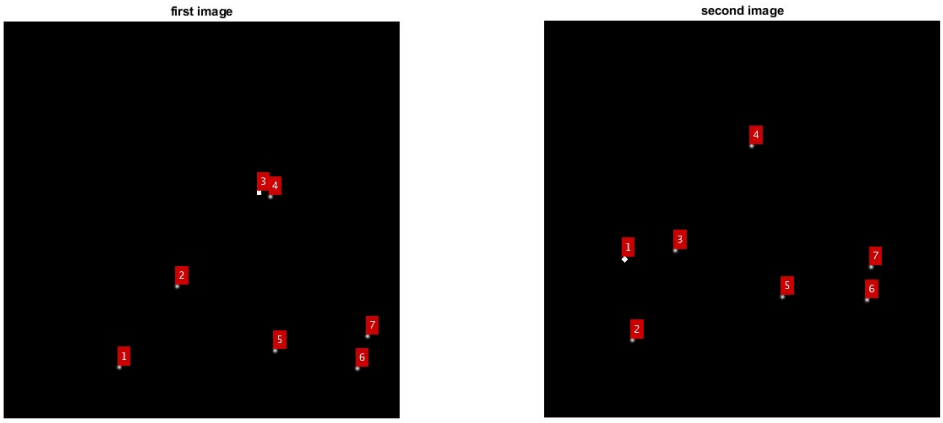

Figure 5.4 : Temporal-sequential rotated and translated star images with a false star injected at time t on the left-hand side and at time $t+1$ on the right-hand side.

-5° in the direction of declination and rotating it by 10° counterclockwise. The image at time $t+1$ has also another arbitrarily-positioned false star labeled number 1.

6. EXPERIMENTAL RESULTS AND DISCUSSIONS

This chapter presents the results of the experiments carried out in the simulation environment. The experiments are divided into three sections including the results provided for the LSI algorithm, the FSF&CME algorithm and the RSI algorithm. The LSI algorithm is the baseline method while the RSI algorithm is an enhanced version of the LSI algorithm improved by the results yielded from the FSF&CME algorithm. Since all algorithms depend on specific parameters, the concepts of parameter selection are provided with the corresponding case studies for all three algorithms. Parameter selection is made to choose the optimal parameters that maintain the most convenient use of the methods used in the algorithms for the given application. The performance evaluations are carried out using statistical indicators that complies with the architecture of each algorithm. The performance of the LSI algorithm and RSI algorithm is evaluated by error plots, precision rate and identification rate. On the other hand, the performance of the FSF algorithm is tracked by the confusion matrix and the derived statistical indicators while the spatial deviation of the transformed stars is used to evaluate the performance of the CME algorithm. The experiments are carried out using the simulation environment developed using MATLAB for the study. The simulation environment allows noise injection including position noise, brightness noise, missing stars and false stars.

6.1 Tests on Lost-in-Space Star Identification Algorithm

The simulation includes modelling the CubeStar sensor [114] which provides a FoV $42^\circ \times 42^\circ$ and a resolution 937×937 in pixels. The feature vectors used in the experiments include five features as given in Equation 2.11. The partial-FoV feature vectors are extracted from circular image frames instead of conventional rectangular ones. The feature vectors are extracted from the corresponding circular patch images obtained from the star catalog database corresponding to the related coordinates. The

dictionary-based star matching method involves the 1NN classification, the dictionary setup and the regularization process respectively. The 1NN classification is realized by selecting the most significant feature vector in the dictionary through binary search of the smallest Euclidean distance, followed by the dictionary setup through selection of the neighboring vectors in the database. Subsequently, the regularization process is carried out, in which the LSE minimization is implemented in order to realize a regression by minimizing L_1 norm, which is called LASSO regularization.

6.1.1 Parameter selection and case study

A number of experiments are carried out to evaluate the performance of the algorithm by simulating the sensor [114]. The performance is evaluated on 10 distinct databases, all of which are created using the parameters lower-bound brightness threshold $\kappa = 11.5$. However, the databases differ in the application of overlap ratio p, since a different overlap ratio is applied on each database such that $p_1 = 0.96$, $p_2 = 0.97$, $p_3 = 0.98$, $p_4 = 0.99$, $p_5 = 0.991$, $p_6 = 0.992$, $p_7 = 0.993$, $p_8 = 0.994$ and $p_9 = 0.995$ and $p_10 = 0.996$ assuming that the indices correspond to the number of the database. Thereby, the number of database feature vectors are calculated using Equation 2.23 by substituting $\phi_x = \phi_y = 42^\circ$, and p_i. They are obtained such that $N_1 = 22959$, $N_2 = 40816$, $N_3 = 91836$, $N_4 = 367346$, $N_5 = 453514$, $N_6 = 573979$, $N_7 = 749687$, $N_8 = 1020408$, $N_9 = 1469388$ and $N_{10} = 2295918$. Note that the number of database vectors, despite increasing accuracy of estimation, leads to higher computational complexity in the binary search used when performing the 1NN method. A bulk of observation vectors are extracted using arbitrary FoV values, and the proposed method is evaluated for all observation vectors using the databases created, that is, 1701 observation vectors are extracted and the experiments are carried out. Some rotation of camera plane with respect to the inertial frame is also added to the observation vectors. The observation images are extracted from the catalog by means of the simulator tools, each of which covers a FoV with right ascension $\alpha \in [2^\circ + k \cdot 5^\circ, 44^\circ + k \cdot 5^\circ]$ and $\delta \in [-88^\circ + m \cdot 5^\circ, -46^\circ + m \cdot 5^\circ]$ where $k = 0, \cdots, 59$ and $m = 0, \cdots, 29$. Also, each set of 1701 observation images are rotated by $\theta_\delta = 80^\circ + t \cdot 80^\circ$ where $t = 0, \cdots, 4$ with respect to the curve $\delta = 0$ in the inertial frame so that it is distinguishable that the

estimation results are independent from the rotation of the camera plane. The results are given in terms of both MSE-type error and 1SE-type error for each database. Also, the curves of precision rates are given for each database. It is shown that the estimation is independent of the rotation of the camera plane. Precision rate curves are constructed in reference to precision plot defined in a benchmark used as an evaluation metric for object tracking [123]. The star sensor is simulated using the characteristics given in Table 2.2, and the Hipparcos catalog is used with the entries given in Table 2.1. The results are given and discussed in this chapter, including both MSE- and 1SE-type error for both the estimated boresight vector and the estimated rotation angle as well as precision rates yielded using all 10 databases.

6.1.2 Statistical performance evaluation

The performance evaluation results are given in terms of error plots and precision rates. The error plots are given for both MSE- and 1SE-type error by implementing the method on the sets of observation images rotated with the given angles using all 10 databases. Therefore, 100 sets of experiments are carried out, all of which involves 1701 trials for each observation vectors extracted from the set of observation images. Also, noise injection is performed to evaluate performance of the algorithm when positional noise, brightness noise, false stars and/or missing stars are present in the observations.

6.1.2.1 Error plots

The error plots contain color-scaled illustration of MSE- and 1SE-type errors obtained for the observation images with the given FoV. The errors $\varepsilon_{f_{MSE}}$ are given for the estimated boresight vectors. The horizontal and vertical axes of the plots represent right ascension and declination respectively. The color scales range between minimum accuracy and maximum accuracy where maximum accuracy is 0° with green color index and minimum accuracy is indexed in red color. Each error type is depicted in two different scales for better illustrative purposes, where the same experiment is given in the scales for an accuracy range pointing to an error interval between $[0^\circ, 0.1^\circ]$ and $[0^\circ, 1^\circ]$. Although the observation images overlap when mapped onto the inertial plane,

a separate representative $42^\circ \times 42^\circ$ FoV square is allocated for each observation image in the illustration filled with the color implying the level of accuracy.

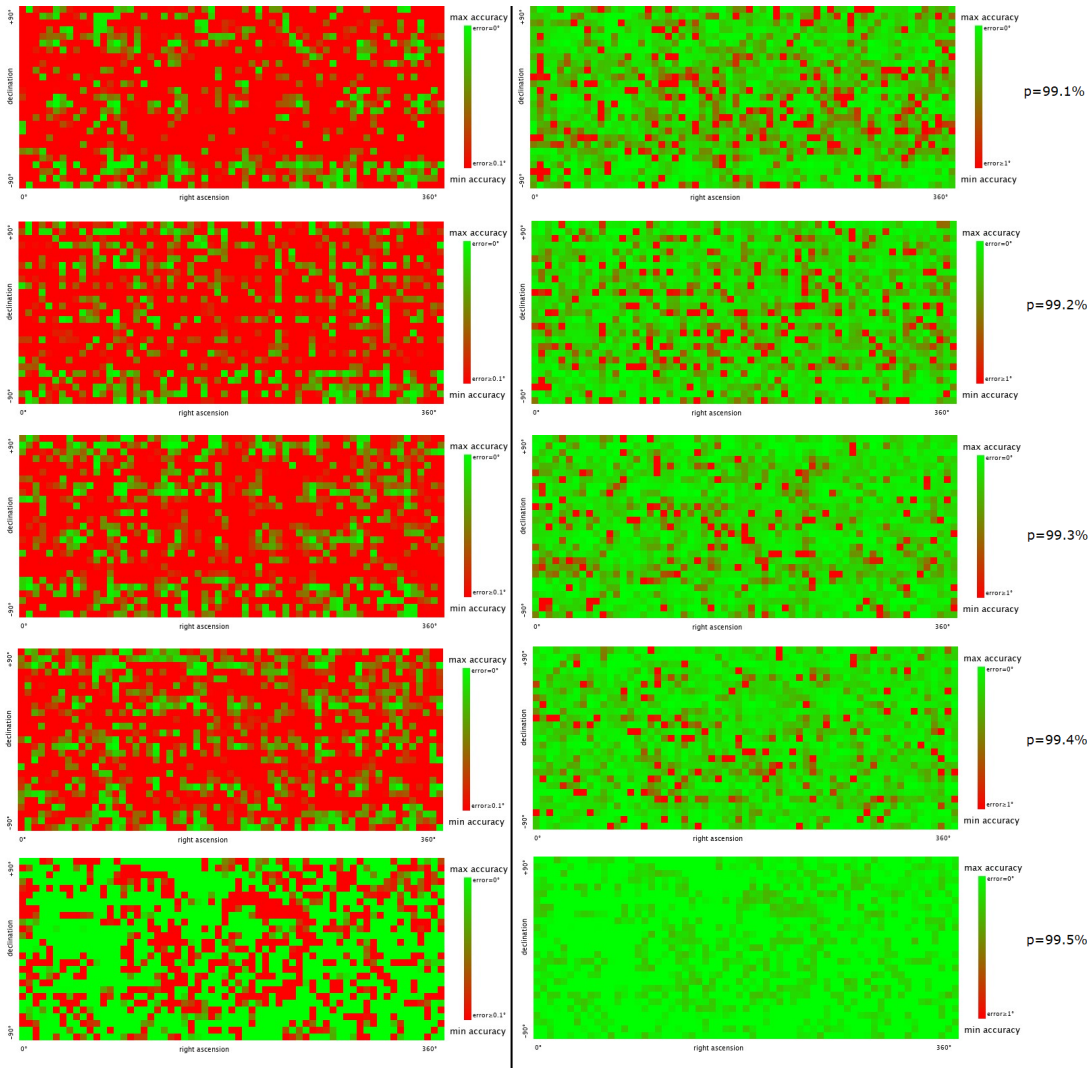

Figure 6.1 : Distribution of MSE-type errors over the database with the given overlap ratios.

Figure 6.1 shows the MSE-type error $\varepsilon_{f_{MSE}}$ yielded by the boresight vector $\widetilde{\overrightarrow{f}}_{MSE}$ estimated using the MSE-LASSO regression for the set of observation vectors scattered over the inertial frame plane, where the horizontal and vertical axes represent right ascension and declination respectively. Each row corresponds to a different database used in the experiment with the given overlap ratio varying between $p = 99.1\%$ and $p = 99.5\%$. Although there are 10 sets of experiments carried out, only 5 of them are depicted, and the results of the rest are provided in the precision rate curves. Each row in the figure represents a database that is used in the implementation of the method with

the values of overlap ratio p given in the right-hand side of the row. The overlap ratios applied when generating the related databases are $p = 99.1\%$, $p = 99.2\%$, $p = 99.3\%$, $p = 99.4\%$ and $p = 99.5\%$ respectively from the top to the bottom. On the other hand, each column represents a different scale of accuracy for each database. Note that the plots on the column on the left-hand side depict accuracy pointing to a MSE-type error within $0^\circ < \varepsilon_{f_{MSE}} < 0.1^\circ$, while the ones in the right-hand side refer to a range of accuracy $0^\circ < \varepsilon_{f_{MSE}} < 1^\circ$. The error rates equal to or above the maximal end of the scale ($\varepsilon_{f_{MSE}} \geq 0.1^\circ$ or $\varepsilon_{f_{MSE}} \geq 1^\circ$) are depicted in full-red color. The values of errors remaining between the selected range are only color-scaled, while the values exceeding the range limit are considered a saturation case. The scheme of color scaling is expressed as

$$\text{error color} = \begin{cases} r \cdot \dfrac{\varepsilon_f}{\varepsilon_1} + g \cdot \left(1 - \dfrac{\varepsilon_f}{\varepsilon_1}\right), & \text{if } \varepsilon_0 \leq \varepsilon_f < \varepsilon_1 \\ r, & \text{if } \varepsilon_f \geq \varepsilon_1 \end{cases} \tag{6.1}$$

where error color is the color assigned to the corresponding error for the estimated boresight vector $\tilde{\vec{f}}$, r and g are the values for color channels of red and green, plus ε_0 and ε_1 are the minimal and maximal error limits. Using Equation 6.1, the estimated boresight vectors with less error are highlighted with green color, while the ones with higher error rates are highlighted with red color, gradually in combination.

The accuracy increases as the overlap ratio p increases both on the left column and on the right column in Figure 6.1. It is apparent that there is no single irrelevant estimation when the database with $p = 99.5\%$ is used in the proposed method since none of the errors for the estimated boresight vectors exceed an error rate of 1° ($\varepsilon_{fMSE} \leq 1^\circ$ for all $\tilde{\vec{f}}_{MSE}$ with $p = 99.5\%$). Also, a significant part of the estimated boresight vectors are accumulated below an error rate of 0.1° for $p = 99.5\%$.

Figure 6.2 shows the 1SE-type error $\varepsilon_{f_{1SE}}$ yielded by the boresight vector $\tilde{\vec{f}}_{1SE}$ estimated using the 1SE-LASSO regression for the set of observation vectors scattered over the inertial frame plane. The plots are also built using the scheme given in Equation 6.1. Although the results are very similar, the differences are visible in the color scale if observed thoroughly.

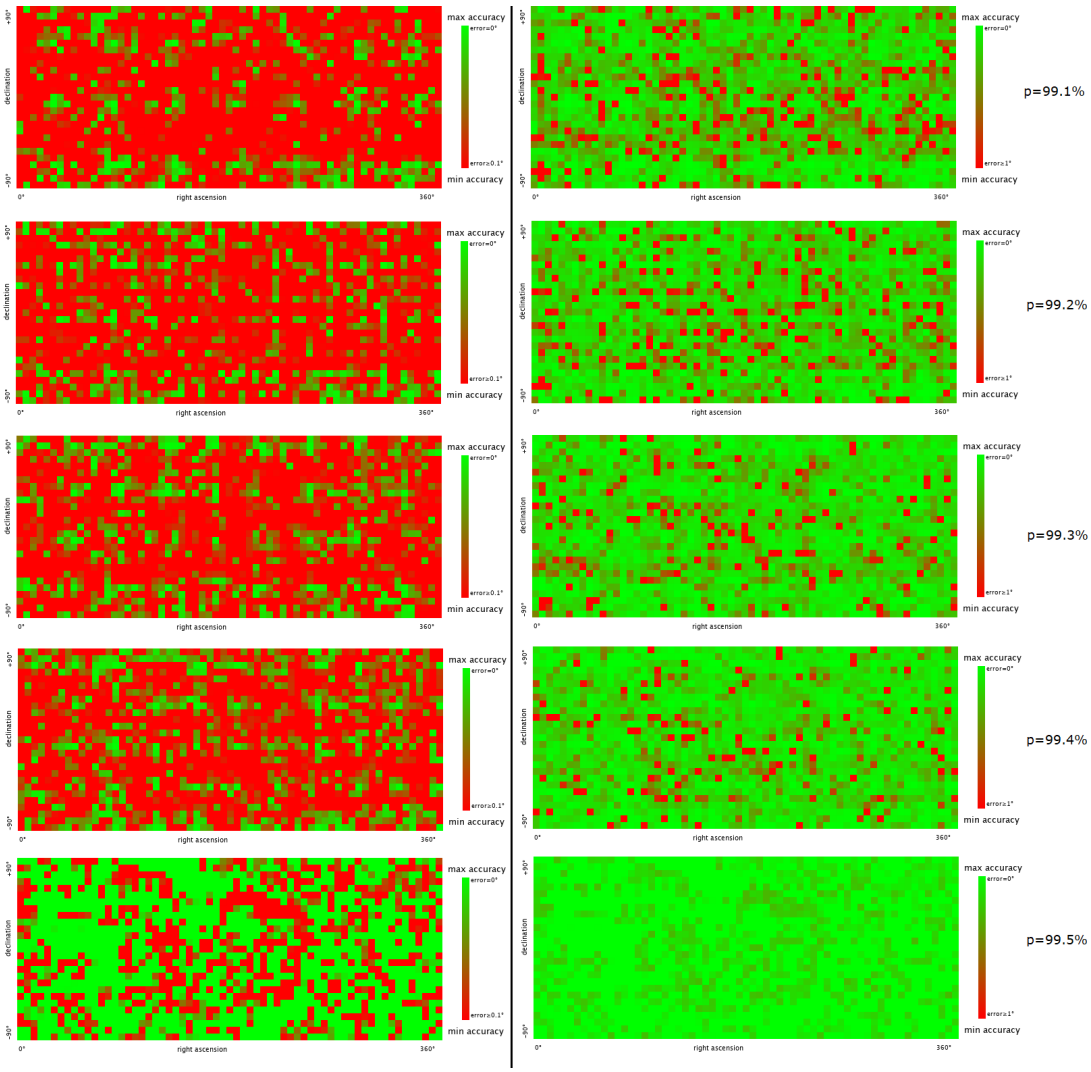

Figure 6.2 : Distribution of 1SE-type errors over the database with the given overlap ratios.

Next, it is shown that the estimation is independent from the rotation of the camera plane. Figure 6.3 depicts the error plot for a set of experiments carried out using the same observation images, but this time rotated with respect to the inertial frame. Note that the accuracy ranges within $0^\circ < \varepsilon_{f_{MSE}} \leq 1^\circ$, and the overlap ratio is selected to be $p = 98\%$. The observation images with same boresight vectors orthogonal to the camera plane as in the previous experiments are extracted, but additional observation images are involved in these experiments. These additional observation images include the ones rotated about an angle $\theta_\delta = 0^\circ + t \cdot 80^\circ$ with respect to the inertial frame where $t = 0, \cdots, 4$. Thus, five sets of observation images are obtained including the ones that are not rotated and the ones rotated about 80°, 160°, 240° and 320° respectively. Also,

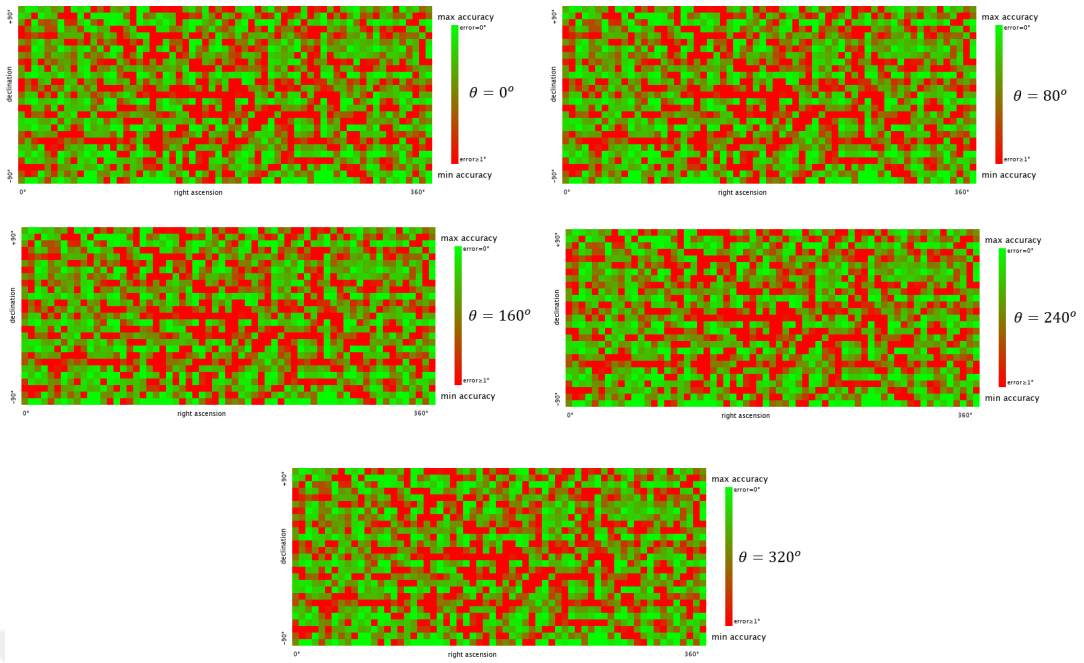

Figure 6.3 : MSE-type error for the estimated boresight vectors using the database with $p = 98\%$, which implies that the estimation is independent from the rotation of the camera plane.

note that the MSE-type error is used in these experiments. The fact that the error plots are exactly identical for different rotation angles implies that the estimation is unrelated to the rotation of the camera plane.

6.1.2.2 Precision rate

The precision rate curves graph the error of the estimated boresight vector ε_f versus precision rate. The curve contains information of 1701 estimations $\overrightarrow{\tilde{f}}$ yielded from the corresponding observation vectors $\overrightarrow{\psi}$ extracted from the observation images. The horizontal axis represents a threshold for the error rate, and the vertical axis corresponds to the percentage of estimations with error rates below the threshold. Thereby, the curves illustrate the percentage of estimations satisfying an error rate falling below the value appointed in the horizontal axis.

The precision rate curves are given for both MSE- and 1SE-type error in Figure 6.4 yielded by the method for the estimated boresight vectors through experiments executed on 1701 observation images using the databases generated by the overlap ratios ranging between $p = 96\%$ and $p = 99.6\%$. The percentages of error rates are

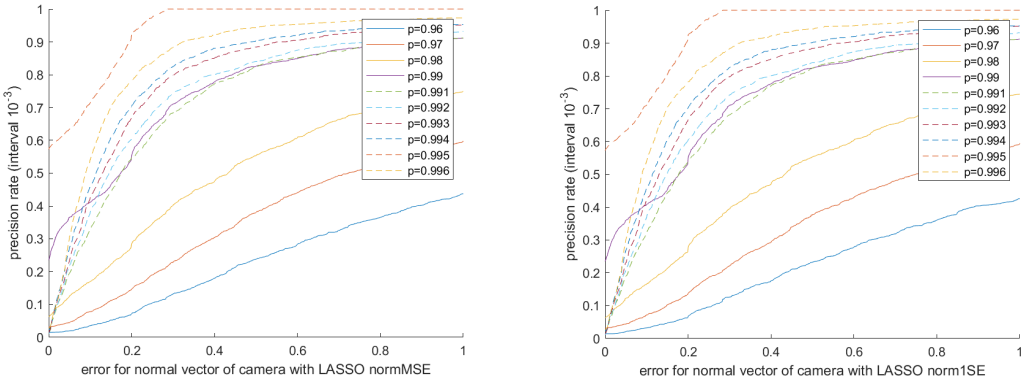

Figure 6.4 : Precision rate of MSE- and 1SE-type error for the estimated boresight vectors.

incrementally calculated for 10^{-3} degrees. First of all, both types of errors reveal a very similar curve as expected. Moreover, it is apparently shown that an increase in the overlap ratio incurs a certain increase in accuracy. The best performance is yielded by the database with an overlap ratio $p = 99.5\%$, in which the accuracy reaches above 90% in case of a threshold value 0.2° and even close to 100% in case of a threshold value 0.3°. This implies that all 1701 estimated FoV vectors have an error rate less than 0.3% in case the database with an overlap ratio $p = 99.5\%$ is used.

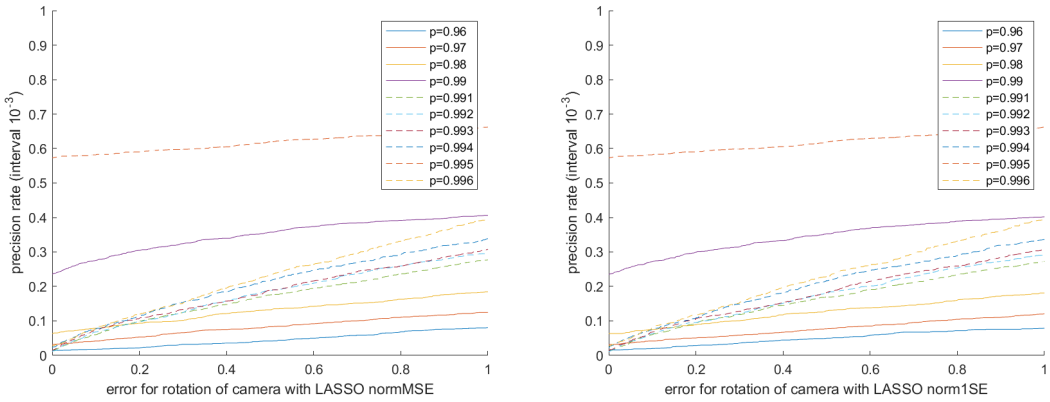

Figure 6.5 : Precision rate of MSE- and 1SE-type error for the estimated rotation angle.

Next, the precision rate curves of the estimated rotation angles are given for both MSE- and 1SE-type error in Figure 6.5 as yielded in the same series of experiments. As is the case in the precision rate curves given in Figure 6.4, the overlap ratios range between $p = 96\%$ and $p = 99.6\%$ and the percentages of error rates are calculated in the increments of 10^{-3} degrees. Similarly, in line with an increasing threshold value,

the accuracy is increasing much slower in comparison with the previous precision rate curves belonging to the estimated boresight vectors. Moreover, the precision rate hardly climbs over 60% accuracy in case of an error threshold 0.2° and approaches to 70% accuracy when the error threshold is close to 1° for the curve of the database with $p = 99.5\%$, which yields the best results.

Furthermore, the results are provided for noise-injected images. For this purpose, the same set of 1701 experimental images are used. Four types of noise are injected, including positional noise, magnitude noise, false stars and missing stars. The performance evaluation is revealed using the database with overlap $p = 99.5\%$ where the best performance is achieved in comparison to the reference precision rate curve achieved using test images without noise. Also, note that the precision rate curves are constructed for the MSE-type error.

Figure 6.6 depicts the performance of the boresight vector estimations in terms of precision rate using the observations injected with noise including positional noise, magnitude noise and both types of noises. The implementation of positional noise injection is made by use of six different values of mean and variance in the normal distribution $\mathcal{N}\left(\mu_p, {\sigma_p}^2\right)$ given in Section 5.2, representing the amount of deviation of the centroid of the noise-added star from the centroid of the true star in terms of pixels in the direction of right ascension and declination with a given mean μ_p and standard deviation σ_p. The first three experiments in two plots at the top row of the figure show results for $\mu_p = 0$ and $\sigma_p = 1$, $\sigma_p = 2$ and $\sigma_p = 3$, whereas the latter three experiments differ in the values of μ_p since μ_p is also drawn from a normal distribution such that $\mu_p \approx \mathcal{N}(0,1)$, noting that it is denoted as mu in the legend. Apparently, the results imply that the algorithm is very robust against positional noise since the curves rapidly converge to the curve of precision rate without noise. For a better understanding, precision rate curves are shown for a maximum error 0.05°, in which it is shown that the curves rapidly converge despite a dramatic decline in success for an accuracy $\varepsilon_{f_{MSE}} < 0.01^\circ$.

The implementation of magnitude noise injection is also made by use of six different values of mean and variance in the normal distribution $\mathcal{N}\left(\mu_m, {\sigma_m}^2\right)$ given in Section

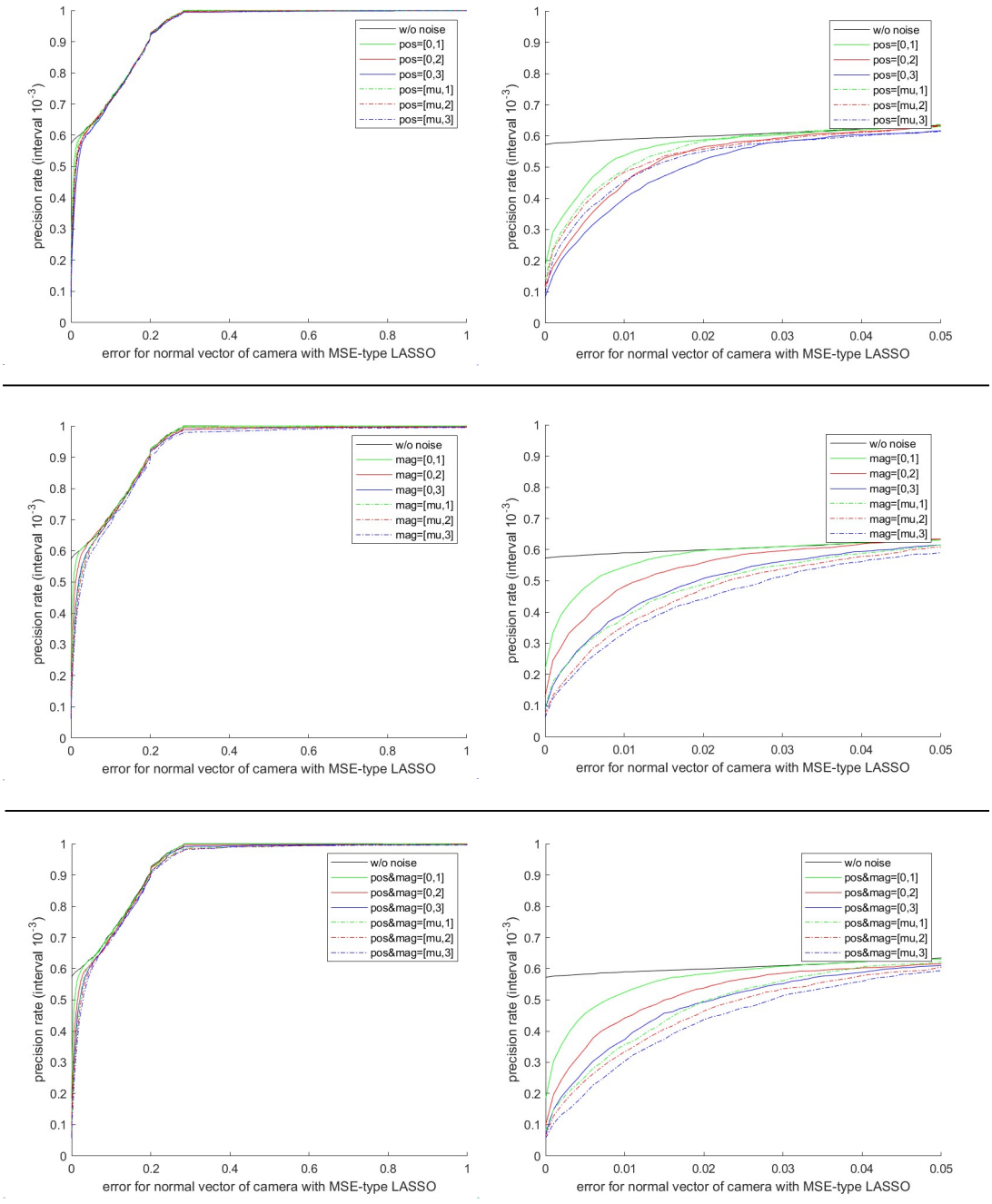

Figure 6.6 : The curves of precision rate of MSE-type error for the estimated boresight vectors ($\varepsilon_{f_{MSE}}$) in the presence of positional noise (top), magnitude noise (middle) and both (bottom).

5.2, representing the amount of deviation of the pixel intensity of the noise-added star from the pixel intensity of the true star with a given mean μ_m and standard deviation σ_m. The configuration of the experiments in the middle row of the figure are similar to the configuration in the experiments of positional noise, and μ_m is similarly drawn from a normal distribution such that $\mu_m \approx \mathcal{N}(0,1)$ in the latter three experiments. The results show a slight slower convergence than the performance in the presence of positional noise, but still prove a very high robustness against magnitude noise.

Two figures at the bottom row depict the results of precision rate where both positional noise and magnitude noise are applied in the same order as the former cases. Although the effect of both noises deteriorates the success slightly, the algorithm proves to be very robust against positional and magnitude noise since a rapid convergence to the reference curve without noise is satisfied.

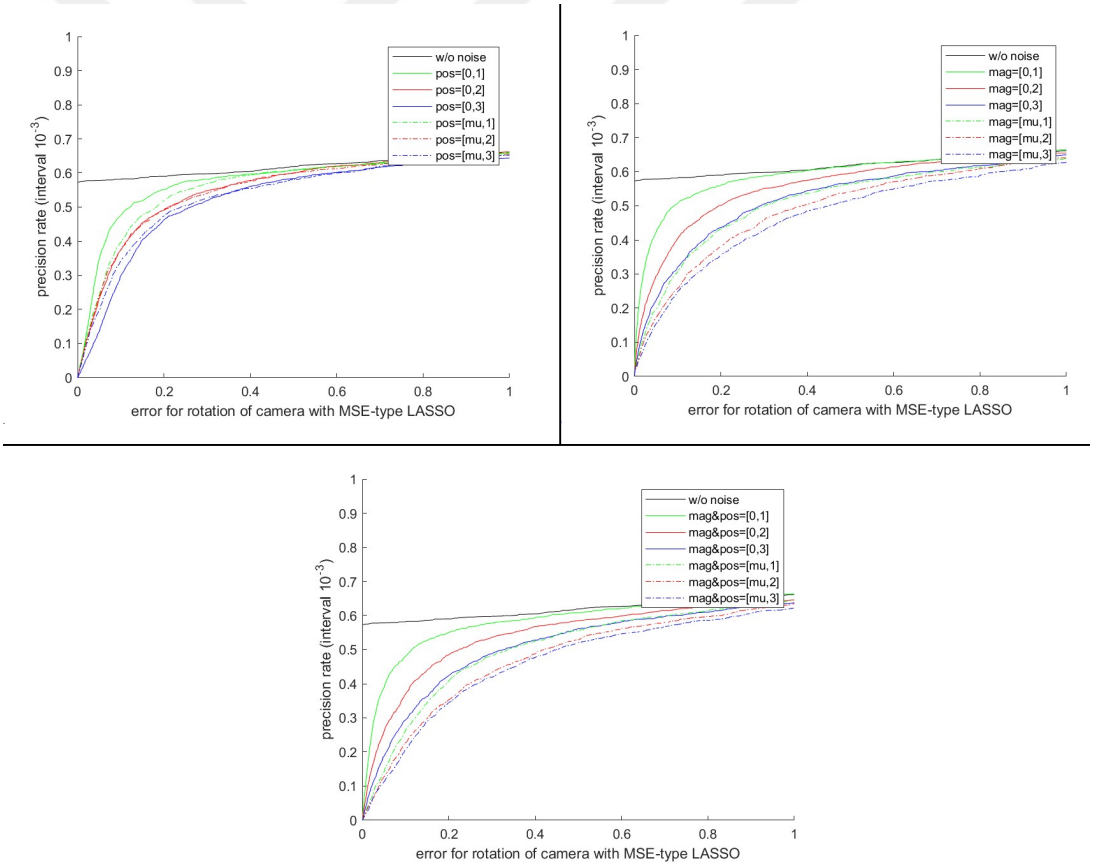

Figure 6.7 : The curves of precision rate of MSE-type error for the estimated camera rotation ($\varepsilon_{\theta_{MSE}}$) in the presence of positional noise (top left), magnitude noise (top right) and both (bottom).

Precision rate curves are also constructed for MSE-type error for the estimated camera rotation angle $\varepsilon_{\theta_{MSE}}$ in Figure 6.7. The same sets of observation feature vectors as the set in Figure 6.6 are used, in which positional and magnitude noise are injected. Despite an inferior success compared to the curves constructed for $\varepsilon_{f_{MSE}}$, a high robustness against positional and magnitude noise is also proven according to the curves.

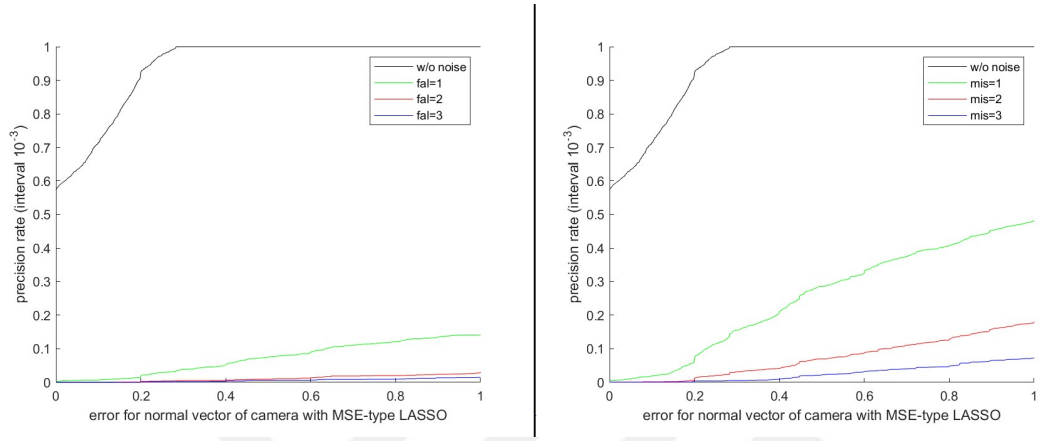

Figure 6.8 : The curves of precision rate of MSE-type error for the estimated boresight vectors ($\varepsilon_{f_{MSE}}$) in the presence of false stars (left) and missing stars (right).

Lastly, the precision rate curves are shown in the presence of false stars and missing stars in Figure 6.8. False stars are injected into the observations with arbitrary position and magnitude in accordance with a uniform distribution defined in Section 5.2. A false star's position is specified by means of two uniform distributions $\mathscr{U}(\alpha_0, \alpha_1)$ and $\mathscr{U}(\delta_0, \delta_1)$ on condition that the FoV of the observation lies within $\alpha \in [\alpha_0, \alpha_1]$ in the direction of right ascension and $\delta \in [\delta_0, \delta_1]$ in the direction of declination, and its magnitude is specified by means of another uniform distribution $\mathscr{U}(m_L, m_H)$ where m_L and m_H are determined by the limits of the sensor, which is $m_L = 11.5$ and $m_H = 15$ in this case. Moreover, to examine the effect of missing stars, the true stars to be removed from the observation are selected in accordance with a uniform distribution $\mathscr{U}(1, n)$ where each true star is indexed with a number between 1 and n, and n is the number of stars in the observation. Accordingly, the left-hand side plot in the figure shows the precision rate curves in the presence of one, two and three false stars in comparison with the reference curve without noise with a maximum error of 1°. It is clear that the algorithm is highly sensitive to false stars since the success is dramatically worsened.

It could not even reach to 20% for an error of 1° in the presence of one false star, and it declines in an accelerating manner for additional false stars. However, it should be noted that the effect of false stars is highly correlated to the number of stars in the FoV of the observation. Thus, a sensor capable with a wider magnitude interval could be more robust to false stars. The effect of missing stars is also investigated in the right-hand side plot in the figure where one, two and three true stars are removed from the observations respectively. The algorithm is proven to be more robust to missing stars in comparison with false stars, yet still very sensitive since the precision rate could not reach to 50% for an error of 1° in case of one missing star. And, the success is reduced similarly with additional missing stars.

6.1.2.3 Identification rate

Because of the fact that the proposed method offers a novel approach which yields a solution vector for the whole FoV rather than matching stars, it is difficult to make a direct comparison with conventional methods. Thus, centroiding error is considered to be an indicator to make a comparison with the proposed method. For example, performance of a method is evaluated with misalignment errors (star tracker's alignment errors) of 3, 30 and 100 arcmins, corresponding to 0.05° (6.75 pixels), 0.5° (67.5 pixels) and 1.67° (> 100 pixels) in the study [70] that proposes an LSI system for small satellites. Another study claims a centroiding accuracy of about 0.1° corresponding to 2.76 pixels [124], which is proposed for agile satellites where fine attitude determination for rapid maneuverability. In a similar way, one study defines buffer radius that determines the limit of accuracy which is selected as 0.3° corresponding to 19.2 pixels [75]. Considering the given values of centroiding errors, the values of success rates that correspond to 16.98 arcmins ($0.283^\circ = 6.33$ pixels) in the precision rate curves for the errors of the estimated boresight vectors ε_f available in Section 6.1.2.2 are assumed appropriate for a comparison, since this value is better than and close to the values given for small satellites in terms of both pixels and degrees and still allows comparison with agile satellites.

In this regard, the curves of identification rate versus noise are given so as to evaluate the performance of the proposed method with the selected parameters comparatively.

The results include the curves of the proposed method and other methods including SGS, IG, IA, PA, MG, GA and GV. SGS, IA, MG and GV were implemented in MATLAB operated on Windows 10 with 2.3 GHz dual-core CPU, which used the SAO catalog in the simulation to produce 10^4 simulated image orientations [93]. On the other hand, IG, PA and MG were implemented in MATLAB operated on Windows with Core i3 2.5 GHz CPU, which used J2000 star catalog in the simulation to produce 10^4 simulated image orientations [94]. The proposed method was implemented in MATLAB in Windows 10 Pro operating system with Core i5 2.70 GHz CPU. In the proposed method, the Hipparcos catalog is used to produce 1701 simulated images.

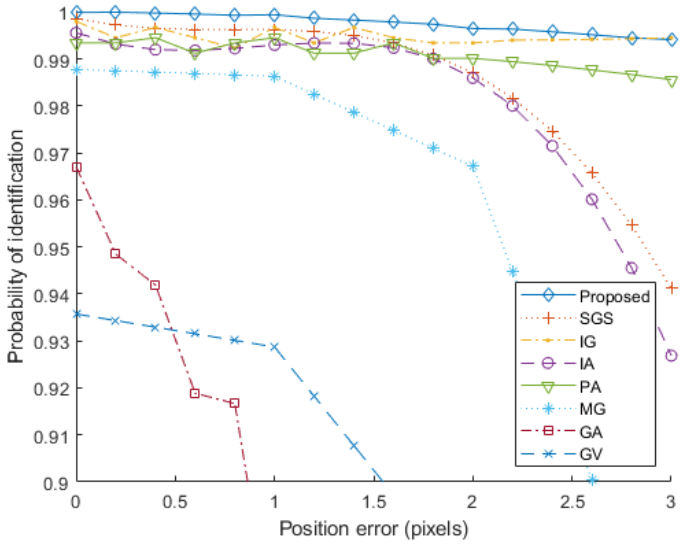

Figure 6.9 : Identification rate of the proposed method implemented with MSE-type error and $p = 99.5\%$ database in comparison with other methods in the presence of position noise.

Figure 6.9 shows the identification rate curves of the algorithms versus positional noise with the given standard deviation values in terms of pixels. The proposed algorithms achieve 99.99% identification rate with no noise injected, while SGS, IG, IA and PA achieve 99.86%, 99.80%, 99.56% and 99.34% respectively. As the standard deviation of positional noise increases, the identification rate of the proposed algorithm significantly outperform others, while the performance of SGS and IA decreases dramatically. Finally, the proposed method achieves 99.41% by outperforming IG and PA achieving 99.40% and 98.56% respectively for a standard deviation 3 pixels for positional noise.

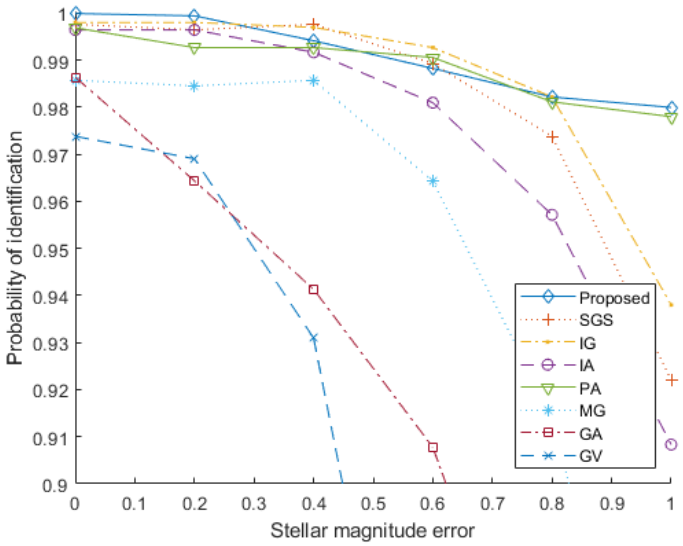

Figure 6.10 : Identification rate of the proposed method implemented with MSE-type error and $p = 99.5\%$ database in comparison with other methods in the presence of magnitude noise.

Figure 6.10 shows identification rate curves versus magnitude noise with the given standard deviation values in terms of stellar magnitude. The proposed method outperforms others in case of no noise with close rivals including SGS, IG, IA and PA. As the standard deviation of magnitude noise increases, the performance of SGS, IG and IA decreases dramatically, although the proposed method with an identification rate 97.99% in case of standard deviation 3 M_V outperforms PA achieving 97.80%. Figures 6.9 and 6.10 reveal that the proposed method is rather robust to positional noise and magnitude noise in comparison with the other methods. However, the curves of identification rate versus false stars tell a different story in Figure 6.11, since the identification rate of the proposed method is disrupted with introduction of false stars. This phenomenon shows that the proposed algorithm is extremely sensitive to false stars.

In addition, it is shown that the proposed algorithm with an overlap ratio $p = 99.5\%$ ensures an accuracy of 0.3° with a probability of 100% in case of no noise and with probabilities of 99.94%, 99.59% and 99.18% in case of both positional noise and magnitude noise injection with standard deviations $\sigma = 1$, $\sigma = 2$ and $\sigma = 3$ respectively for positional noise in terms of pixels and magnitude noise in terms of pixel intensity. The proposed algorithm also achieves the accuracy rates 0.2°,

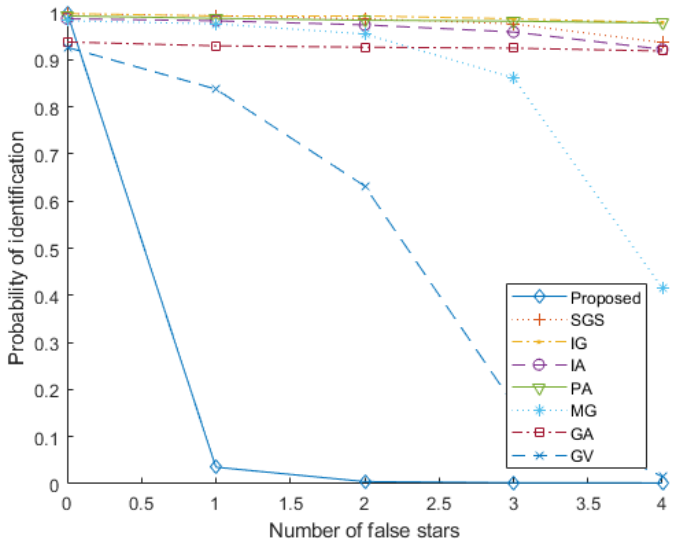

Figure 6.11 : Identification rate of the proposed method implemented with MSE-type error and $p = 99.5\%$ database in comparison with other methods in the presence of false stars.

0.1°, 0.013°, 0.008° and 0.002° given in the studies [11,12] respectively with the probabilities 92.24%, 71.43%, 59.20%, 58.67% and 57.79% without noise injection, while achieving with probabilities 91.89%, 91.59% and 90.65% with both types of noise injected with $\sigma = 1$, and 71.19%, 70.66% and 70.08% with $\sigma = 2$, and 35.57%, 24.40% and 18.99% with $\sigma = 3$.

6.1.2.4 Complexity analysis

The complexity of an algorithm is as significant as its accuracy since it is an indicator for real time implementation. The proposed algorithm is evaluated in terms of database size and average run time to measure its feasibility in real time. Time complexity of the proposed algorithm involves time required for feature extraction, 1NN implementation, dictionary setup, LASSO regression and calculation of estimated vectors. Noting that time complexity of feature extraction is $\mathcal{O}(f)$ where f is the number of stars in the FoV dependent on the magnitude threshold κ, that of 1NN implementation is $\mathcal{O}(N)$ where N is the number of database feature vectors dependent on the overlap ratio p, that of dictionary setup is $\mathcal{O}(1)$, that of LASSO regression is $\mathcal{O}\left(K^3 + K^2 n\right)$ where K and n are the number of variables in the feature vector and the number of database feature vectors in the dictionary [125] and that of calculation of estimated vectors is

$\mathcal{O}(1)$, the amount of time except 1NN implementation and LASSO regression can be neglected. Therefore, there is a tradeoff between this two steps in the algorithm. As the overlap ratio p increases, N also increases, leading to an increase in time complexity of the implementation of the 1NN classifier besides the database size, but also a decrease in time complexity of LASSO regression because of faster convergence to solution thanks to higher similarity between the most significant database feature vector $\overrightarrow{t}_5$ in the dictionary and the observation feature vector $\overrightarrow{\psi}$.

Table 6.1 : Database size and average run time of the proposed algorithm with different values of overlap ratio p.

Overlap ratio	Database size (kB)	Average run time (ms)					
		Feature extract	1NN	Dictionary setup	LASSO	Estimation	Total
$p = 96\%$	99	0.071	0.491	0.021	13.072	0.030	13.613
$p = 97\%$	176	0.071	0.974	0.020	12.535	0.029	13.559
$p = 98\%$	392	0.071	1.914	0.027	12.411	0.034	14.386
$p = 99\%$	1575	0.071	8.899	0.025	10.246	0.030	19.200
$p = 99.1\%$	1945	0.071	11.441	0.016	9.333	0.027	20.816
$p = 99.2\%$	2464	0.071	15.042	0.022	9.317	0.028	24.409
$p = 99.3\%$	3290	0.071	20.138	0.022	8.876	0.028	29.064
$p = 99.4\%$	4380	0.071	26.957	0.022	8.053	0.027	35.059
$p = 99.5\%$	6300	0.071	38.577	0.022	7.480	0.029	46.108
$p = 99.75\%$	25225	0.071	287.686	0.022	5.371	0.029	293.179

Table 6.1 shows the required database size in kB and the average run time in ms as measured in the experiments. Since the feature vectors are embedded into database after being normalized, a precision of 8 bits is sufficient for each element in the database. It is apparent that the value of overlap ratio p has a significant effect on both the database size and the average run time because it determines the number of database feature vectors N. An increase in p leads to an increase in database size and a decrease in average run time. Note that average run time required for the steps including feature extraction, dictionary setup and estimation tends to stay constant and negligible in contrast to 1NN implementation and LASSO regression in accordance with the theoretical expressions. In addition, an increase in p constantly leads to an increase in average run time of 1NN implementation but a gradual decrease in that of LASSO regression. Total average run time tends to increase in parallel to an increasing value of p.

Table 6.2 : Database size and average run time of the proposed algorithm in comparison with the other algorithms.

Algorithm	Database size (kB)	Average run time (ms)
PA	130.57	27.5
MGA	7380	394.6
LPT	665.89	73.8
VP	280.72	7.8
GA	331	185.6
SVD	1483	13.4
OSV	145	35.2
Proposed LSI algorithm	6300	46.1

Table 6.2 shows the database size and average run time of the proposed algorithm implemented with $p = 99.5\%$ yielding the best results and other algorithms including PA, MGA, LPT, VP, GA, SVD and OSV. The proposed method achieved 6.3 MB in database size and 46.1 ms in average run time, which are, thanks to the outperforming results in identification rate, enough to claim that the method is able to compete with the given methods despite failing to exhibit the best results in terms of database size and average run time. The proposed algorithm performs similarly to PA and OSV, and outperforms MGA, LPT and GA, although it lags behind VP and SVD in terms of average run time. On the other hand, its 6300 kB database size is larger than its competitors in general as it outperforms only MGA with a database size of 7380 kB.

The cause for its comparatively larger database size is the high value of overlap ratio $p = 99.5\%$ used in the generation of the database. Note that the database only seizes 99 kB in case of an overlap ratio $p = 96\%$. The cost of high database size is compensated with much higher identification rate under positional noise and magnitude noise or in case of no noise injection.

6.1.3 Summary

An LSI method is proposed in this chapter. The method runs into a cascade structure of 1NN classifier, dictionary setup and a regularization operator and estimator sequentially. The 1NN classifier uses Euclidean distances between the observation feature vector and the database feature vectors as inputs and outputs the most similar feature vector in the dictionary. The dictionary setup is carried out through selection of the closest neighbor feature vectors of the most similar feature vector from the database. The regularization operator is based on L_1 regression that is called the LASSO regression, in which the dictionary is used to estimate a regularization weight vector $\overrightarrow{\omega}$. It is used to reconstruct an estimated boresight vector and an estimated rotation angle. Throughout the process, a novel set of features is used. Feature extraction allows acquisition of a feature vector for each image frame with a given FoV, which helps the information of all stars in the given FoV be accumulated in a single feature vector. A feature vector of the given FoV comprises the number of stars n, the mean of magnitude of stars A, the polar radius and angle of the magnitude-weighted centroid of the stars r and θ and the standard deviation of magnitude and polar radii of each star σ_A and σ_r. Five features are used in the matching process while the remaining one θ is used in the estimation step to predict the planar rotation angle of the camera. Thus, instead of creating a separate feature vector for each star, a single feature vector saves run time and memory size without compromising performance. The features are examined in detail to show their contribution to the solution when empirically exposed to the effect of the different values of parameters, overlap ratio p and magnitude threshold κ, that determine the key characteristics of the database. The identification algorithm employs the 1NN classifier and the regularization operator. The Hipparcos catalog and the CubeStar star sensor are used to generate database and observation

images. The developed simulator allows generation of both database feature vectors $\vec{\tau}$ and observation feature vectors $\vec{\psi}$ as well as noise injection in addition to the execution of the algorithm. Noise injection involves injection of positional noise, magnitude noise, false stars and missing stars into an observation image. A case study is performed, in which 1701 observation images are used to output the estimations. The results are illustrated in the error plots and precision rate curves. The precision rate curves ensures to decide that the database generated with $p = 99.5\%$ yields the best results. This database is used to compare the proposed algorithm with some state-of-the-art algorithms in terms of identification rate, database size and average run time.

This study claims that the proposed algorithm make contributions including high accuracy, independence of the estimation from the rotation of the camera plane about the boresight vector, a new approach of estimation with two solutions including the estimated boresight vector $\tilde{\vec{f}}$ and the estimated rotation angle $\tilde{\theta}$, and reduction in computational complexity and shrinkage in database size. The levels of pointing accuracy offered by the state-of-the-art spacecraft ADCS units are achieved with the probabilities given in Section 6.1.2.3. Higher accuracy is also achieved with positional noise and magnitude noise as shown in Figures 6.9 and 6.10, while the performance is deteriorated dramatically in case of injection of false stars and missing stars as shown in Figures 6.8 and 6.11. The accuracy with false and missing stars can be increased by using a wider range of magnitude threshold κ to increase the number of stars in the FoV. In addition, a contributing study can be used to add a preprocessing step that aims to remove false stars [2]. Independence of the algorithm from the rotation of camera plane is shown in Figure 6.3, which avoids additional burden in the regularization step. A pair of estimated solutions including $\tilde{\vec{f}}$ and $\tilde{\theta}$ ensures a complete spatial attitude information. Despite a very high accuracy in the estimation of $\tilde{\vec{f}}$ in Figures 6.9 and 6.10, the accuracy of the estimated rotation angle $\tilde{\theta}$ remains low as shown in Figure 6.7. New research can be made to search for better results by carrying out more experiments by focusing on the parameters of the LASSO regression, or the estimation of $\tilde{\vec{f}}$ can be executed in two directions instead of estimating $\tilde{\theta}$, which requires an unwanted additional camera. It is shown that a higher accuracy costs a

larger database size and a longer average run time in Table 6.1. Nevertheless, this cost does not prevent the proposed algorithm from managing to compete with some of the state-of-the-art methods as shown in Table 6.2, considering that its very high accuracy 99.99% without noise and its continual higher accuracy with injected positional noise and magnitude noise. To overcome the decline of accuracy in the presence of false stars, a new FSF method is developed using unsupervised learning presented in Section 3. Moreover, additional benefits harvested from this algorithm pave the way to develop an RSI algorithm presented in Section 4.

6.2 Tests on False Star Filtering and Camera Motion Estimation

This section presents the experiments and their results carried out to evaluate the performance of the FSF&CME algorithm. Firstly, since the FSF algorithm is based on an unsupervised learning method, preliminary tests are performed to detect the optimal parameters required for the method. After choosing the optimal values for the parameters ε and $\min_p$, the performance of the algorithm is statistically exhibited for both FSF and CME in a case study where noise is also present. Next, a wide range of series of experiments are carried out under different conditions to evaluate the performance. A confusion matrix is used for demonstration, and the performance indicators are derived from the confusion matrix.

6.2.1 Parameter selection and case study

The parameters ε and $\min_p$ are required to be determined appropriately so that a single cluster that contains the disparity vectors corresponding to the feature vectors of true star pairs can be generated. A series of simulations are carried out to determine the parameters that would yield the best results in accordance with the algorithm.

For selection of clustering parameters, three groups of experiments are carried out as shown in Figure 6.12. The first group involves 10 sets of 1000 estimations to examine the effect of the clustering parameter ε on the performance by implementing the algorithm on the image frames with position noise drawn from a normal distribution $\mathscr{N}(0^\circ, 0.1^\circ)$ but neither false stars nor brightness noise when the other parameter $\min_p = 3$ is chosen for all cases. The left-hand side plot shows the average number of

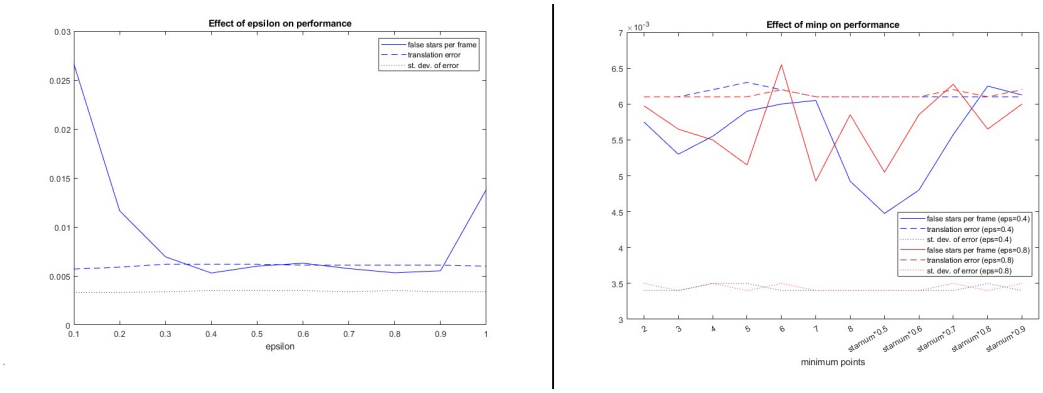

Figure 6.12 : Evaluation of performance for selection of the optimal clustering parameters.

false stars per image frame despite absence of false stars, mean of translation error ε_ρ and the corresponding standard deviation σ_ρ. While changing ε causes no significant difference in ε_ρ and σ_ρ, the average number of stars that is expected to be close to zero has smaller values when $\varepsilon = 0.4$ and $\varepsilon = 0.8$. Next, the second group of experiments is carried out to find the optimal value of $\min_p$ for $\varepsilon = 0.4$ revealed by the blue curve in the right-hand side plot, and the third group for $\varepsilon = 0.8$ by the red curve in the right-hand side plot. It is shown that the highest accuracy is given in the case with $\varepsilon = 0.4$ and $\min_p = 0.5 \cdot \mathrm{n}\{\overrightarrow{c}_i\}$ where $\mathrm{n}\{\overrightarrow{c}_i\}$ represents the number of detected object centroids (star candidates) in the given image frame. Thus, the parameters can be dynamically updated for the images with different number of stars. The extensive package of experiments are carried out to evaluate performance by using the values of $\varepsilon = 0.4$ and $\min_p = 0.5 \cdot \mathrm{n}\{\overrightarrow{c}_i\}$ for the clustering parameters, which implies that for each candidate core point selected in the disparity list $\mathscr{D}$ would constitute a cluster with the points in the list $\mathscr{D}$ satisfying that they are in $\varepsilon = 0.4$ neighborhood and on condition that the total number of points is at least $\min_p = 0.5 \cdot \mathrm{n}\{\overrightarrow{c}_i\}$, half the number of detected object centroids in the image frame. After selection of the optimal clustering parameters, 36 sets of 1000 experiments are carried out using the same initial conditions and the same probabilistic distributions for the parameters of translation and rotation. The experiment sets include variations of 1001 subsequent image frames with different number of false stars f_i $(i = 0, 1, \cdots, 5)$ and different values of noise factor μ_j $(j = 0, 1, \cdots, 5)$. Noise factor μ_j corresponds to the number of pixels in case of position noise and number of bit values in case of brightness noise.

The algorithm is implemented on two sequential images given in Figure 5.4. The true stars are successfully matched. The object labeled number 3 in the image t and the object labeled number 1 in the image $t+1$ are not matched, thus called false stars. Moreover, the estimations of rotation $\tilde{\gamma}$ and translation $\tilde{\overrightarrow{\rho}}$ components of camera motion are obtained. In this example, the motion estimates are calculated by backtracking each star object. Thus, they fall into some intervals such that $\tilde{\gamma} \in (9.75^\circ, 10.58^\circ)$ and $\tilde{\overrightarrow{\rho}} \in \left(\begin{bmatrix} -0.04^\circ \\ -3.99^\circ \end{bmatrix}, \begin{bmatrix} 0.22^\circ \\ -3.54^\circ \end{bmatrix} \right)$ while the true values are $\gamma = -10^\circ$ and $\overrightarrow{\rho} = \begin{bmatrix} 0^\circ \\ -5^\circ \end{bmatrix}$. Using the centroids of 7 objects, a separate set of 21 feature vectors are generated for both images, which are $\mathscr{V}^t = \{\overrightarrow{v}_i^t\}$ and $\mathscr{V}^{t+1} = \{\overrightarrow{v}_i^{t+1}\}$. Using two sets of feature vectors, the disparity list $\mathscr{D}$ is generated, which contains 441 disparity vectors $\overrightarrow{d}$. The disparity list is illustrated in Figure 3.2. Then, the true stars are matched by applying the density-based clustering on the disparity list. The affine transformation matrix containing motion parameters is subsequently obtained using the RANSAC algorithm.

Table 6.3 : Results of false star detection for the case study.

f_i	Type of noise	Accuracy	Precision	Recall
0	None	1	1	1
0	Brightness	1	1	1
0	Position	1	1	1
0	Both	1	1	1
1	None	1	1	1
1	Brightness	98.75%	98.59%	1
1	Position	1	1	1
1	Both	1	1	1
2	None	1	1	1
2	Brightness	98.89%	98.59%	1
2	Position	1	1	1
2	Both	98.89%	98.59%	1

In addition, another case study is simulated, in which 10 matching results are obtained from 11 sequential images. After the initial image frame is selected arbitrarily, the following images are simulated using the distributions $\mathscr{U}\left(\begin{bmatrix} -25 \\ -25 \end{bmatrix}, \begin{bmatrix} 25 \\ 25 \end{bmatrix} \right)$ for translation and $\mathscr{U}(-2.5, 2.5)$ for rotation. 12 sets of experiments are carried out, which includes images injected with brightness noise and/or position noise accompanied by no false stars, 1 false star and 2 false stars, which is stated by f_i.

Brightness noise and position noise are added using the normal distributions $\mathcal{N}(0,1)$ in terms of pixel intensity for brightness and $\mathcal{N}(0,1)$ in terms of pixels for position. Note that there are 7 true stars in all image frames. In Table 6.3, the results of 12 experiments are given in terms of accuracy, precision and recall. The algorithm manages to successfully detect 7 true stars in all experiments. Three exceptional incidents occur when one false star is mistakenly classified as true stars in one experiment with 1 false star and brightness noise as well as in two other experiments with 2 false stars and brightness noise and both types of noise. While accuracy shows overall success of the model, precision indicates success when predicting the true stars. On the other hand, recall shows whether the model can find all true stars. According to the table, the algorithm perfectly detects all true stars without incorrectly labeling them as false stars in case of no false stars. The impurities in the cases of 1 false star and 2 false stars arise due to three exceptional occurrences where false stars are incorrectly classified as true stars.

Table 6.4 : Estimation errors for the case study.

f_i	Type of noise	ε_γ	σ_γ	ε_ρ	σ_ρ
0	None	$4.6 \cdot 10^{-14}$	$1.4 \cdot 10^{-14}$	$6.8 \cdot 10^{-15}$	$3.8 \cdot 10^{-15}$
0	Brightness	$1.1 \cdot 10^{-2}$	$5.6 \cdot 10^{-2}$	$7.5 \cdot 10^{-3}$	$7.6 \cdot 10^{-3}$
0	Position	$6.6 \cdot 10^{-1}$	$1.8 \cdot 10^{-1}$	$2.9 \cdot 10^{-2}$	$1.6 \cdot 10^{-2}$
0	Both	$7.2 \cdot 10^{-1}$	$1.8 \cdot 10^{-1}$	$3.1 \cdot 10^{-2}$	$1.9 \cdot 10^{-2}$
1	None	$7.4 \cdot 10^{-1}$	$1.4 \cdot 10^{-14}$	$6.8 \cdot 10^{-15}$	$3.8 \cdot 10^{-15}$
1	Brightness	$7.1 \cdot 10^{-1}$	$2.1 \cdot 10^{-2}$	$3.1 \cdot 10^{-3}$	$2.4 \cdot 10^{-3}$
1	Position	$6.8 \cdot 10^{-1}$	$1.9 \cdot 10^{-1}$	$3.1 \cdot 10^{-2}$	$1.7 \cdot 10^{-2}$
1	Both	$6.8 \cdot 10^{-1}$	$1.8 \cdot 10^{-1}$	$3.3 \cdot 10^{-2}$	$1.6 \cdot 10^{-2}$
2	None	$7.4 \cdot 10^{-1}$	$1.4 \cdot 10^{-14}$	$6.8 \cdot 10^{-15}$	$3.8 \cdot 10^{-15}$
2	Brightness	$7.5 \cdot 10^{-1}$	$2.3 \cdot 10^{-2}$	$3.2 \cdot 10^{-3}$	$2.5 \cdot 10^{-3}$
2	Position	$6.7 \cdot 10^{-1}$	$1.4 \cdot 10^{-1}$	$2.7 \cdot 10^{-2}$	$1.7 \cdot 10^{-2}$
2	Both	$6.7 \cdot 10^{-1}$	$1.7 \cdot 10^{-1}$	$2.9 \cdot 10^{-2}$	$1.7 \cdot 10^{-2}$

The estimation errors and their standard deviations are also obtained. Table 6.4 shows the estimation errors, where ε_γ and σ_γ represent the error of the rotation component and the corresponding standard deviation in terms of degrees while ε_ρ and σ_ρ represent the same for the translation component. It is shown that both the estimate errors and the corresponding standard deviations increase as higher amount of noise is added. On the other hand, injection of false stars into the scene does not significantly affect

success, implying that the algorithm is robust to false stars. However, more extensive and varied experiments are carried out to make a more detailed statistical performance analysis.

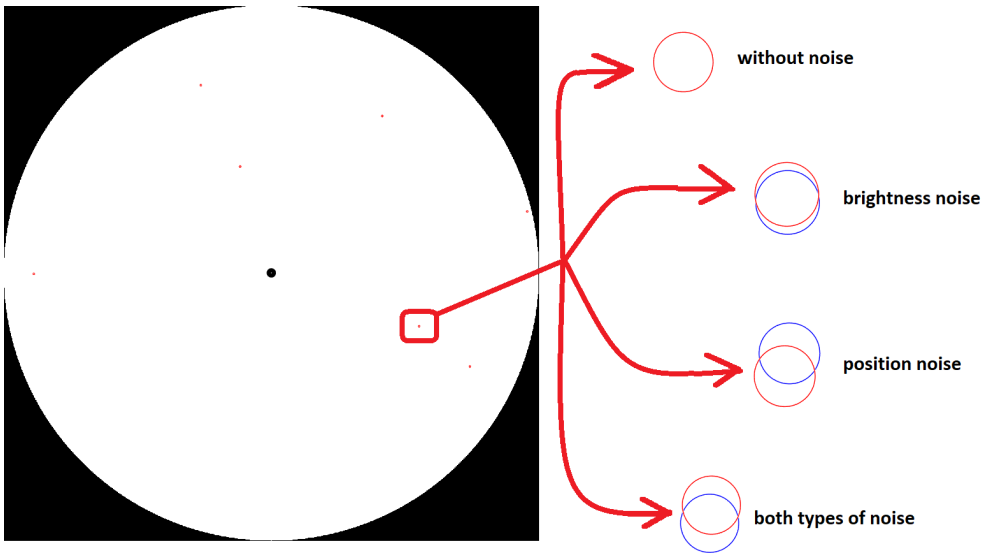

Figure 6.13 : True stars and reconstructed stars.

An example image shows a comparison of true stars and reconstructed stars in Figure 6.13. An image with no false stars is selected to indicate the effect of noise addition more clearly. The reconstructed stars are obtained by applying affine transformation matrix in the subsequent image. Note that this image is a single example taken from the case study. The estimated affine transformation matrix is $\tilde{\mathbf{H}} = \begin{bmatrix} 0.9996 & -0.0290 & 3.552710^{-15} \\ 0.0290 & 0.9996 & 2.131610^{-14} \\ 0 & 0 & 1 \end{bmatrix}$ which preserves the estimated rotation matrix $\tilde{\mathbf{R}}$ and the estimated translation matrix $\tilde{\mathbf{U}}$. A single star is magnified to show the effect of noise on the performance where blue and red colors represent true and reconstructed stars respectively. While the error is significant in case of no noise where the reconstructed star exactly fits the true star, the size of error becomes significant in the presence of noise.

6.2.2 Statistical performance analysis

A number of experiments are carried out to analyze the performance of the algorithm in detail. The density-based clustering parameters ε and $\min_p$ are selected as obtained from the case study in Figure 6.12. Using these optimal clustering parameters,

the extensive package of experiments are implemented to evaluate performance of the algorithm in terms of statistical measures including accuracy, precision, recall and F1-score. Note that the initial image frame corresponds to right ascension $\alpha \in [87.41^\circ, 129.41^\circ]$ and declination $\delta \in [-8.86^\circ, 33.14^\circ]$, and the subsequent image frames are shifted by $\mathscr{U}\left(\begin{bmatrix}-25^\circ\\-25^\circ\end{bmatrix}, \begin{bmatrix}25^\circ\\25^\circ\end{bmatrix}\right)$ and rotated by $\mathscr{U}(-2.5^\circ, 2.5^\circ)$ in all experiments.

6.2.2.1 Confusion matrix and indicators

The statistical measures are derived from the confusion matrix containing TP, FP, FN and TN. TP and FP imply that the estimator decides a true star when the object is a true star or a false star respectively. FN and TN imply that the estimator decides a false star when the object is a true star or a false star respectively. Thus, TP and TN are correct decisions while FP and FN are incorrect decisions. Accuracy is the most used measure despite not being the most appropriate measure when target classes are unbalanced. Precision shows the relevancy of the selected data items, that is, it indicates the ratio of true positives out of positive estimations. Recall reveals the amount of relevancy in selections by proportioning correctly estimated true stars to all true stars in the scene. F1-score, on the other hand, is the harmonic mean of precision and recall [126]. They are formulated as follows.

$$\text{accuracy} = \frac{\text{TP}+\text{TN}}{\text{TP}+\text{TN}+\text{FP}+\text{FN}} \tag{6.2a}$$

$$\text{precision} = \frac{\text{TP}}{\text{TP}+\text{FP}} \tag{6.2b}$$

$$\text{recall} = \frac{\text{TP}}{\text{TP}+\text{FN}} \tag{6.2c}$$

$$\text{F1-score} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{6.2d}$$

The confusion matrices of the experiments are provided for different scenarios implemented. They are given in Appendix A. The tables cover all combinations of experiments including all numbers of false stars varying between $f_i = 0$ and $f_i = 5$ and all types of noise factors varying between $\mu_j = 0$ and $\mu_j = 5$.

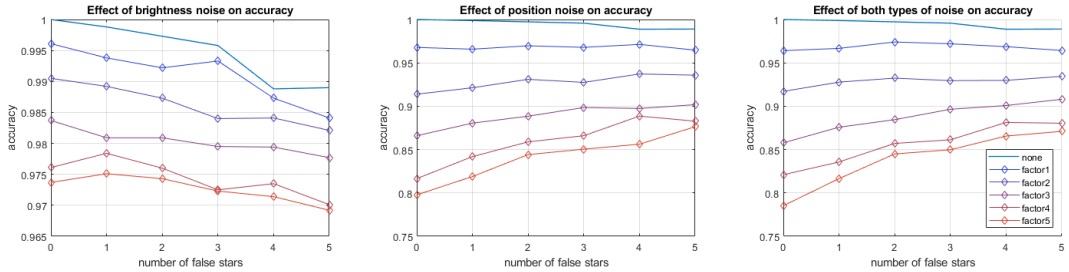

Figure 6.14 : Accuracy of the FSF algorithm in the presence of noise.

Figure 6.14 shows the achieved accuracy (see Equation 6.2a) of the algorithm for different number of false stars in the scene in three cases including brightness noise, position noise and both. It is shown that the accuracy is 100% in case of no false stars and no noise, that is, all true stars are classified correctly without mistakenly labelling any true star as a false star in all 1000 trials. Accuracy is 100% in case of no false stars and no noise. The top left plot of the figure shows accuracy when only brightness noise is injected to the frames. Accuracy, in this case, declines as the noise and the number of false stars increase. It drops down to 96.92% in the worst case when the number of false stars and the noise factor both take the peak value of 5. In case of position noise addition illustrated in the top right plot of the figure, accuracy also decreases as the number of false stars and the noise factor increase. However, as more false stars are introduced into the scene, the accuracy tends to increase as the noise factor increases, which implies that the robustness of the method against false stars is, in contrast to the robustness behavior when brightness noise is present, boosted in the presence of intenser position noise. For instance, when the noise factor is 5, the accuracy increases from 79.78% with no false stars injected up to 87.64% with 5 false stars injected. On the contrary, it slightly decreases in case the noise factor is 1. The bottom plot exhibits accuracy when both types of noise are combined, in which the effect of position noise dominates the same of brightness noise. Both types of noise combined reduces accuracy down to 78.53%. Note that an increase in noise factor stimulates higher accuracy as the number of false stars increases.

The precision values are revealed in Figure 6.15. Precision, given in Equation 6.2b, helps visualize the reliability of the model. The method proves to be very robust in filtering false stars against both types of noise. Although the precision performance is

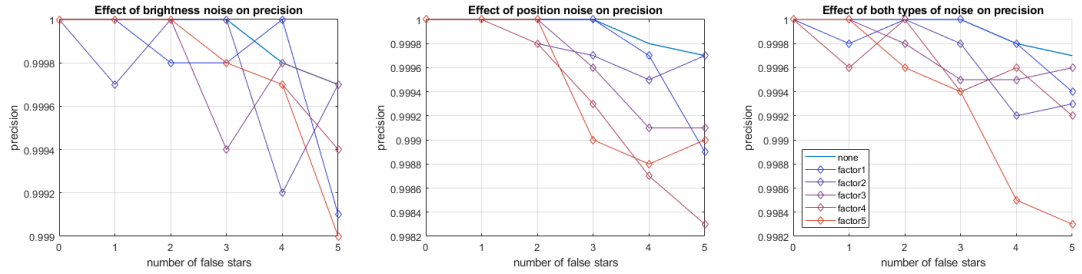

Figure 6.15 : Precision of the FSF algorithm in the presence of noise.

slightly degraded with the introduction of additional false stars and with an increased noise factor, it does not fall under 99.83% in the worst case when 5 false stars are introduced and both types of noise are injected with a noise factor 5. This implies that the algorithm is very consistent when estimating true stars.

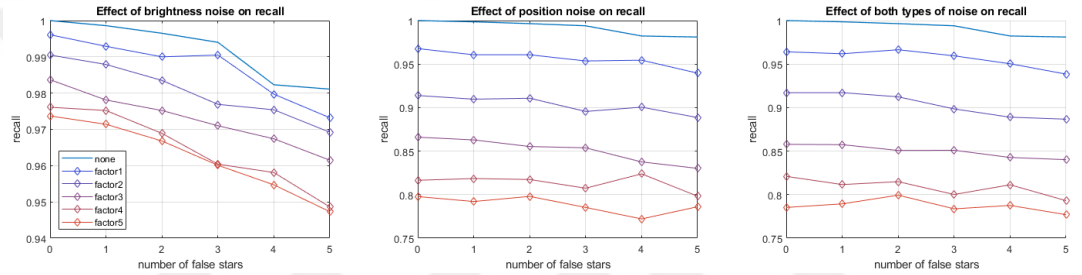

Figure 6.16 : Recall of the FSF algorithm in the presence of noise.

Figure 6.16 shows the recall values, which is formulated in Equation 6.2c. Recall is a measure that can be used as an indicator in the cases when the cost of prediction of false negative. In the top left plot of the figure, it is shown that the method is robust against brightness noise in terms of recall despite a descent with an increase in the number of false stars and the value of noise factor. However, the recall values decrease much more in case of the presence of position noise when the value of recall drops down to 77.21% in the worst case as seen in right top plot of the figure. Surprisingly, the combination of both types of noise causes a very slight increase in the recall values rather than a decrease in comparison with the cases of only position noise as seen in the bottom plot of the figure. Therefore, the algorithm is sensitive to position noise when examined with respect to recall despite higher values of precision. The tradeoff between precision and recall is slightly in favor of precision. Particularly in the presence of position noise, the number of false negatives is larger than the number of false positives. The number of false stars that the algorithm fails to detect are larger than the number of true stars

labelled as false stars. Considering that a correct detection of true stars is important in terms of success of a star identification algorithm, a higher precision is preferable. On the other hand, a smaller value of recall leads to a poorer filtering of false stars, thus incurring an increase in computational complexity and decrease in success of a star identification algorithm. Nevertheless, the proposed false filtering algorithm achieves a performance for recall higher than 95% in all cases in the presence of brightness noise and higher than 85% in most cases in the presence of position noise.

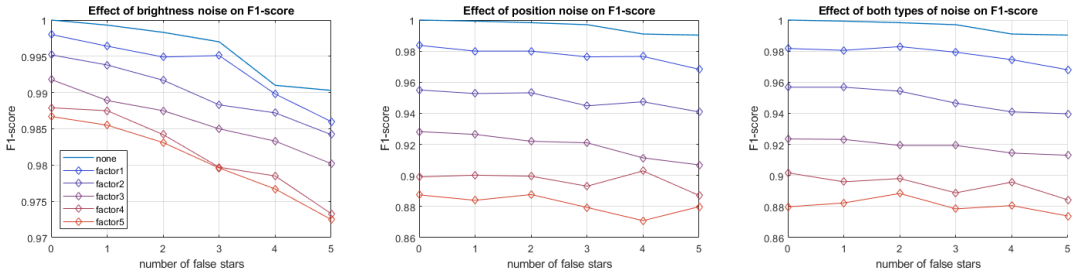

Figure 6.17 : F1-score of the FSF algorithm in the presence of noise.

F1-score is a measure that indicates harmonic mean of precision and recall as seen in Equation 6.2d. Harmonic mean avoids neglecting outliers since outliers are taken into account for a better interpretation. Figure 6.17 shows the values of F1-score for the proposed algorithm. Similar to the curves of precision and recall, the values of F1-score also decrease as both the number of false stars and the level of noise factor increase. While the values of precision and recall fall down to 99.90% and 94.73% in the worst case when $f_i = 5$ and $\mu_i = 5$, F1-score descends to 97.25% in the presence of brightness noise. On the other hand, in the presence of position noise, the former two measures drop down to 99.83% and 77.21% respectively, while the latter falls down to 87.09%.

6.2.2.2 Morphological error measurement

The CME performance is evaluated by means of measurements of morphological features. The morphological features are obtained from the estimated affine transformation matrix $\tilde{\mathbf{H}}$ applied on the coordinates of the stars. The stars proved not to be false stars in the former test image frame are transformed using $\tilde{\mathbf{H}}$ in accordance with Equation 3.2.

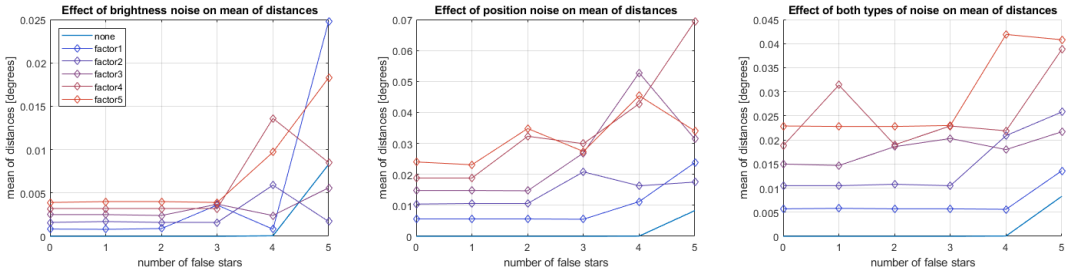

Figure 6.18 : Error of CME is given by the mean of the distances between the centroids of the true stars at t and the centroids of the stars at $t+1$ transformed using the affine transformation matrix in the presence of noise.

Figure 6.18 shows the mean of distances between the stars transformed using $\tilde{\mathbf{H}}$ and the true stars for each pair of images of all the trials in the experiments. The mean values is a good indicator for the accuracy of CME. The means of distances indicate the amount of error in terms of degrees given by

$$\mu_d = \frac{\sum_{i=1}^{n} \| \tilde{\vec{c}}_i^{t+1} - \vec{c}_i^{t+1} \|}{n} \tag{6.3}$$

where $\tilde{\vec{c}}_i^{t+1}$ is derived using Equation 3.2. In the absence of noise, the error is very low in the orders of 10^{-15}, but climbs up to 0.0083° in the presence of 5 false stars. The error tends to increase when the noise factor and the number of false stars increase, and reach to maximum value at 0.0419° when the noise factor is $\mu_j = 5$ and there are 4 false stars in the scene.

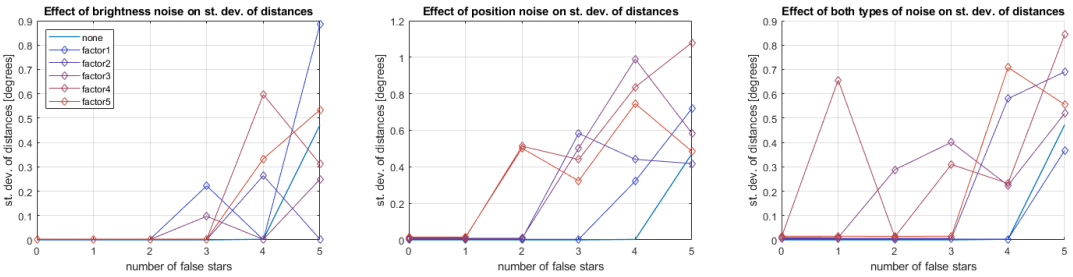

Figure 6.19 : Standard deviation of error of CME is obtained from the distances between the centroids of the true stars at t and the centroids of the stars at $t+1$ transformed using the affine transformation matrix in the presence of noise.

Another morphological feature used to evaluate the performance of CME is standard deviation of distances σ_d. It is calculated such that

$$\sigma_d = \frac{\sum_{i=1}^{n} \left(\| \tilde{\vec{c}}_i^{t+1} - \vec{c}_i^{t+1} \| - \mu_d \right)^2}{n} \tag{6.4}$$

which is based on the distances between the transformed stars and the true stars. The values of standard deviation of distances σ_d are given in Figure 6.19. These demonstrations are significant because they can give feedback to the LSI algorithm to shrink the database used in the phase of 1NN classification so that run time can be reduced while increasing accuracy further. It is promising to see that the values increase as the values of distances that represent an indicator of error increase. Therefore, a larger region can be chosen from the database when implementing the 1NN classifier so as not to miss the most significant dictionary vector.

6.2.3 Summary

This section analyzes the empirical results of the FSF&CME algorithm. First, a case study is investigated, which has been performed to select the parameters required for the algorithm optimally. The density-based clustering method used in the FSF algorithm is dependent on the parameters ε and $\min_p$, which represent radius of neighborhood and minimum number of vectors within this neighborhood. After performing experiments under different noise scenarios, it is decided that the optimal values are $\varepsilon = 0.4$ and $\min_p = 0.5 \cdot \mathrm{n}\{\overrightarrow{c}_i\}$ to implement the FSF algorithm. Next, the FSF algorithm is implemented under various noise scenarios. The performance is investigated using the corresponding confusion matrices and its relevant statistical indicators. This is followed by implementation of the CME algorithm using the experimental setup of the FSF algorithm. The performance of the CME algorithm is evaluated using morphological error measurements, namely distance errors. The FSF algorithm achieves very high accuracy and precision under various noise scenarios. It performs still effective but less than accuracy and precision in terms of recall and F1-score. This implies that the FSF algorithm is robust to false alarms. It is more likely to falsely label true stars as false stars, rather than falsely labelling false stars as true stars. The distance error is obtained by calculating spatial distances between star objects in the successive frames after transforming one of them using the estimated transformation matrix $\tilde{\mathbf{H}}$. The means and standard deviations are calculated for all

trials in all experiments. Both statistical indicators reveal superior performance, which implies that the CME algorithm is not only reliable but also consistent.

6.3 Tests on Recursive Star Identification Algorithm

This section presents the experiments and their results carried out to evaluate the performance of the RSI algorithm. The RSI algorithm, in fact, relies on the same fundamental methodological structure as the LSI algorithm proposed in this thesis except that the RSI method retrieves a priori information of motion estimated by the FSF algorithm also proposed in this thesis. The CME results are used to determine a RoS to speed up the process of 1NN classification in the LSI method, followed by an update mechanism that detects a limit deviation from the required accuracy in the process of the RSI algorithm. Firstly, the performance measurements derived from the former experiments are used to choose the optimal parameters for both the update mechanism and the scheme of RoS selection. Then, the performance of the RSI algorithm is evaluated in terms of MSE-type error in comparison with the LSI method. Additionally, the curves of precision rate and identification as well as average run time are provided, exhibiting the improvements in comparison with the LSI method.

6.3.1 Parameter selection and case study

The set of experiments carried out to evaluate the performance of the proposed LSI algorithm are used to watch the patterns of errors and detect a meaningful correlation that leads the algorithm to deviate from the required amount of accuracy.

Figure 6.20 shows the patterns of the error for the estimated inertial boresight vector ε_f and the error for the reconstructed observation vector ε_ψ, respectively defined in Equations 2.21 and 2.20. The series of experiments used for the performance evaluation of the LSI algorithm are used to extract the given curves. The instances between 560th and 600th trials are used for demonstration. Note that the red line and the dashed blue line respectively represent the curves of ε_f and ε_ψ. The noise factors are $f_i = 0$ and $\mu = 0$ in this scheme, which implies that neither noise nor false stars are injected in these trials. The vertical axis is represented in terms of $(1 + \log \varepsilon)$ for a better illustration of a relationship between the variables. The direct proportionality

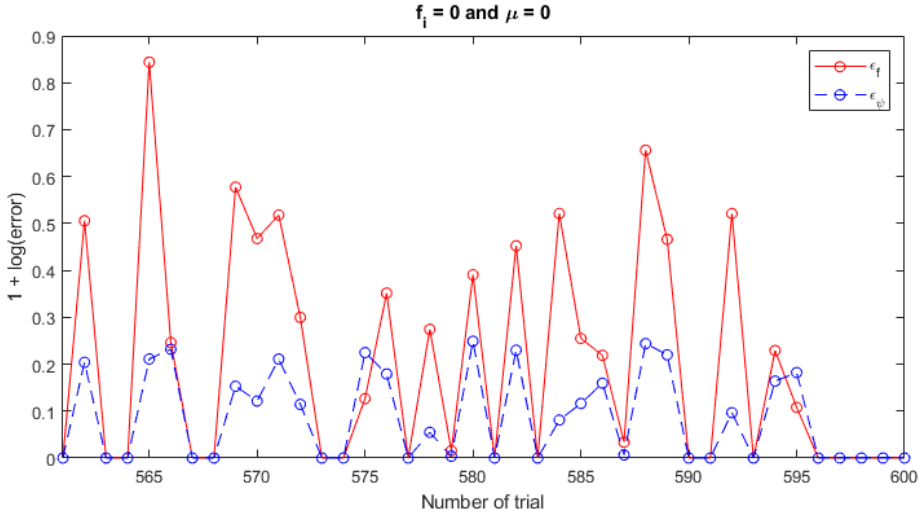

Figure 6.20 : The patterns of ε_ψ and ε_f derived from the test trials 560-600 using the LSI algorithm for the case without noise.

is clearly visible in the figure. A deviation from the required accuracy as the value of ε_f increases is, at all instances, accompanied by an increase in the value of ε_ψ. This correlation can be used to trigger an alarm signal for a deviation of the estimated inertial boresight vector $\vec{\tilde{f}}$ from the required accuracy. In this case, the error for the reconstructed observation vector ε_ψ serves as an alarm indicator.

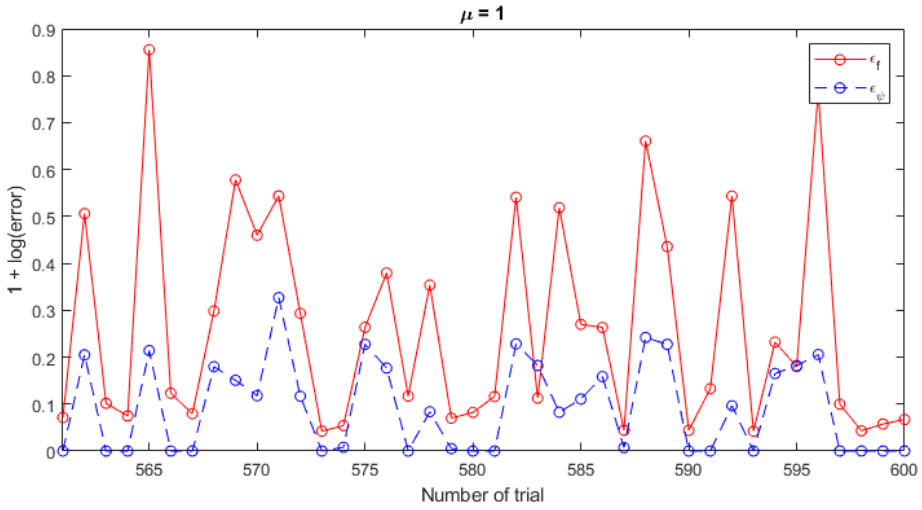

Figure 6.21 : The patterns of ε_ψ and ε_f derived from the test trials 560-600 using the LSI algorithm for the case with a noise factor $\mu_j = 1$.

The same illustration is given in Figure 6.21 for the experiments processed by the LSI algorithm implemented on the simulated test images with position and brightness noise injected. In this case, the noise factor is given as $\mu_j = 1$. Despite slight

differences in comparison with the experiments without noise, the pattern exhibits a similar correlation between the variables.

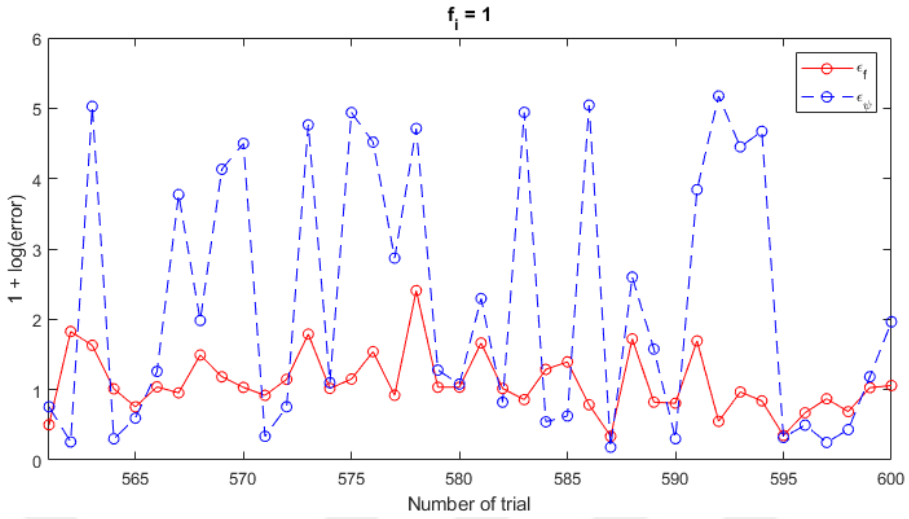

Figure 6.22 : The patterns of ε_ψ and ε_f derived from the test trials 560-600 using the LSI algorithm for the case with 1 false star $f_i = 1$.

Figure 6.22 shows the patterns for the experimental case with 1 false star injected in the test images. The accuracy of the estimations made by the LSI algorithm deteriorates in most cases. Thus, the LSI algorithm is said to be very sensitive to false stars. A pattern of correlation between the variables similar to the case with $\mu_j = 1$, despite a wider scope of alteration in the patterns with respect to the case without noise $\mu_j = 0$, is also promising to allow a detection of estimations with poor accuracy in case of false stars. Therefore, the error ε_ψ can be used as an alarm indicator for a poor accuracy.

Figure 6.23 shows the patterns for the experimental case with 1 missing star injected in the test images. The accuracy is also poor in these estimations, implying that the LSI algorithm is also very sensitive to missing stars. A correlated pattern is also visible in this case. The patterns in the cases with false stars and missing stars are not distinguished as it is in the cases with position and brightness noise injected. However, this is insignificant because of very poor accuracy in these cases. Nevertheless, the incidence of a positive correlation is promising. In all cases, a directly proportional correlation between the error variables allows the error ε_ψ to be used as an alarm indicator for poor accuracy.

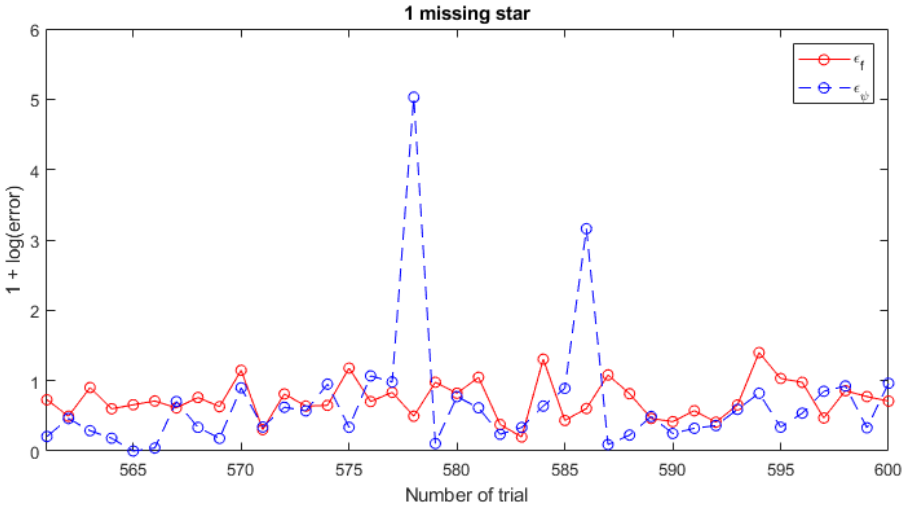

Figure 6.23 : The patterns of ε_ψ and ε_f derived from the test trials 560-600 using the LSI algorithm for the case with 1 missing star.

The same procedure is repeated to investigate the presence of a correlation between the error for the estimated rotation angle ε_θ and the error for the reconstructed observation vector ε_ψ. For this purpose, the patterns of the errors are obtained from the experiments carried out to evaluate the performance of the LSI algorithm. Similar to the patterns of the error for the estimated inertial boresight vector ε_f, a strong correlation between the variables is clearly apparent for the results derived without noise in Figure 6.24. In case of the presence of position and brightness noise in Figure 6.25, there is also a high correlation despite an increase in the values of errors. Figure 6.26 shows the cases of false stars injected to the test images where the patterns of correlation are also similar to the pattern derived for ε_f. The error curves are mostly directly correlated despite some dissimilarities due to high values of error. Lastly, the pattern of the error pair is investigated for missing stars in Figure 6.27. The correlation is again similar to the former pattern with missing stars. There is a correlation between the variables despite partly irrelevancies due to high values of error for this case. Overall, the patterns exhibit a similar scheme of correlation between the errors ε_ψ and ε_θ except that ε_θ has much higher values than ε_f in line with the results derived from the LSI algorithm.

The correlations between the errors are investigated for all types of noise configurations which have been implemented for the performance evaluation of the LSI algorithm. The patterns can be seen in the curves for the correlation between ε_ψ and ε_f for

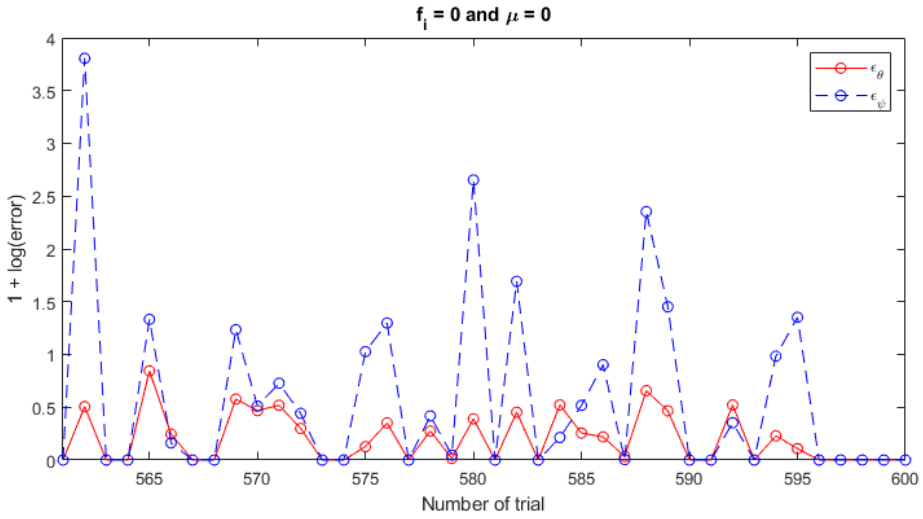

Figure 6.24 : The patterns of ε_ψ and ε_θ derived from the test trials 560-600 using the LSI algorithm for the case without noise.

brightness and position noise factors $\mu_j = 2$ and $\mu_j = 3$ in Figures B.1 and B.2, for the number of false stars $f_i = 2$ and $f_i = 3$ in Figures B.3 and B.4, and for 2 and 3 missing stars in Figures B.5 and B.6. All patterns of errors exhibit a correlation between the variables despite increasing disturbances due to larger amount of noise injected. Therefore, it is reasonable to employ ε_ψ as an alarm indicator for an estimation with a potential of poor accuracy.

Another procedure of analysis is carried out to determine a threshold value for ε_ψ to activate the update mechanism when the threshold is exceeded. The values of ε_ψ are investigated to determine the error threshold τ_{ε_ψ}. For this purpose, the values of the error for the estimated inertial boresight vector ε_f and and the error for the estimated rotation angle ε_θ are examined with respect to varying threshold values forced on the error for the reconstructed observation error ε_ψ.

Figure 6.28 illustrates the curves obtained to investigate the effect of varying values of ε_ψ on ε_f in the presence of brightness and position noise. The horizontal axis represent the ε_ψ thresholds while the vertical axis represents the characteristics of ε_f. In the left-hand side and right-hand side of the figure, the curves of the mean of ε_f and the standard deviation of ε_f are provided for the corresponding values of ε_ψ respectively. For the values of ε_ψ with increments of 0.01, a threshold is determined. An array containing each value of ε_f corresponding to the ε_ψ values below the determined

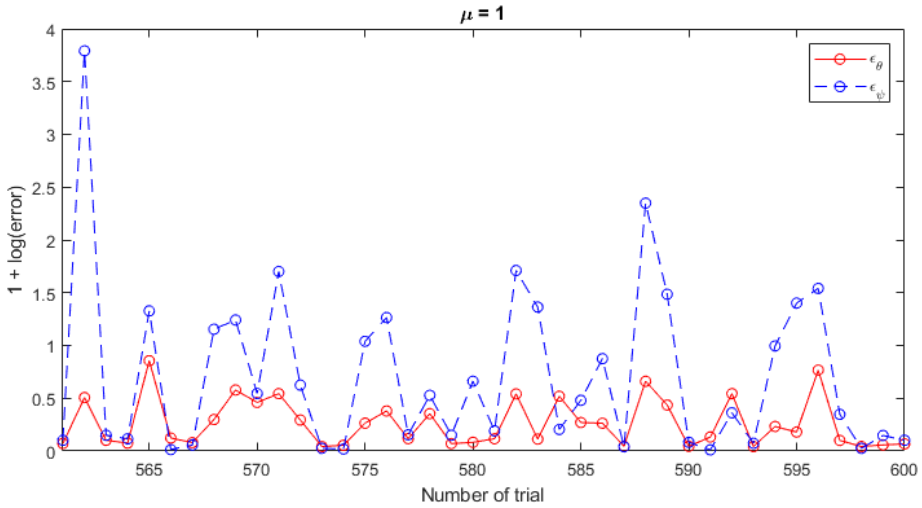

Figure 6.25 : The patterns of ε_ψ and ε_θ derived from the test trials 560-600 using the LSI algorithm for the case with a noise factor $\mu_j = 1$.

threshold is created. Thus, a separate array is generated for each ε_ψ threshold. The figures are created using the characteristics of mean and standard deviation of the corresponding arrays of ε_f lying below the corresponding ε_ψ threshold. According to Figure 6.28, the mean of ε_f steadily increases up to approximately 0.1 for a corresponding threshold value of 1 in case of no noise while the standard deviation is very low, implying stability. As the noise factor increases, both mean and standard deviation increase gradually. For larger values of noise factor, the stability is lost. There are boundary points where the values are subject to bounces for both mean and standard deviation. The bounces in the values of mean point to large decreases in the accuracy within a tiny increment of threshold while that of standard deviation to a instability in the accuracy of estimations. While a value just below a threshold value of 0.5 is critical in case of the noise factor $\mu_j = 3$, the critical value is just above 0.5 for the noise factor $\mu_j = 2$.

Figures 6.29 and 6.30 show the effect of varying values of ε_ψ on ε_f in the presence of false stars and missing stars respectively. First of all, since the values of ε_f are very high in these cases due to poor estimation performance of the LSI algorithm, the behavior of the case without false stars is not distinguishable. However, this graph has already been discussed in Figure 6.28 where the case without noise is given. Except that the cases with 2 and 3 false and missing stars can be claimed to have a

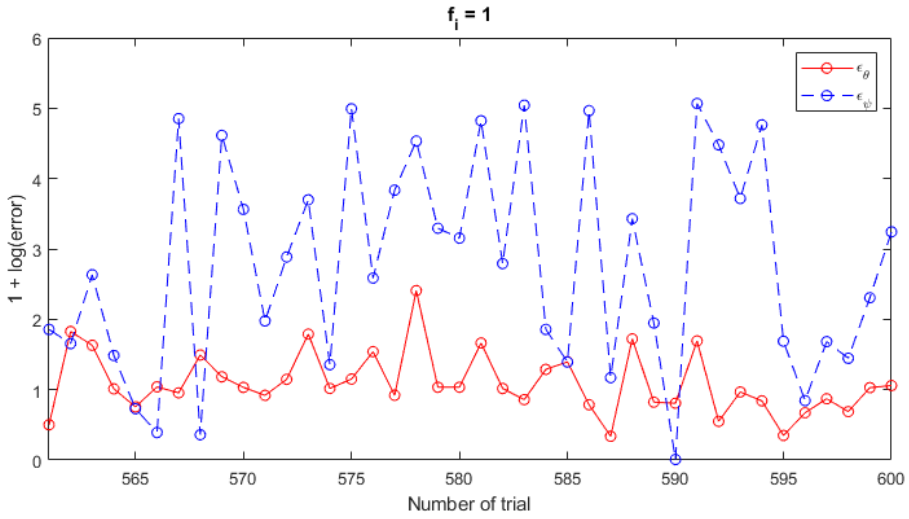

Figure 6.26 : The patterns of ε_ψ and ε_θ derived from the test trials 560-600 using the LSI algorithm for the case with 1 false star $f_i = 1$.

less meaningful patterns despite still an increasing pattern, the curves referring to 1 false star and 1 missing star exhibit an increasing gradient for both mean and standard deviation. Unlike the patterns obtained in Figure 6.28, the increases in gradient are triggered at different points. While the mean values are subject to an initial bounce at around a corresponding threshold value of 0.5, the standard deviation values are triggered to bounce at about 0.25. This means that the accuracy stability of the estimations begin to deviate earlier than a high decrease in estimation accuracy.

The error threshold is also investigated for its effects on the error for the estimated rotation angle ε_θ. Similarly, the curves are extracted for the cases with the presence of position and brightness noise, false stars and missing stars. The results are given in Appendix C. The patterns are very similar to the ones carried out for ε_f except that the values of ε_θ are higher due to poorer accuracy provided by the LSI algorithm for the estimation of rotation angle in comparison with the estimation of the boresight vector. Figure C.1 shows the effect of ε_ψ on ε_θ. The curves of mean and standard deviation of ε_θ are revealed in the presence of position and brightness noise. Similar to Figure 6.28, both of the curves have similar gradients in addition to higher fluctuating curves in case of larger noise factor. According to Figures C.2 and C.3 where the value of errors are much higher than the case with brightness and position noise, the oscillations of the curves are very similar to the ones in Figures 6.29 and 6.30 except that the values

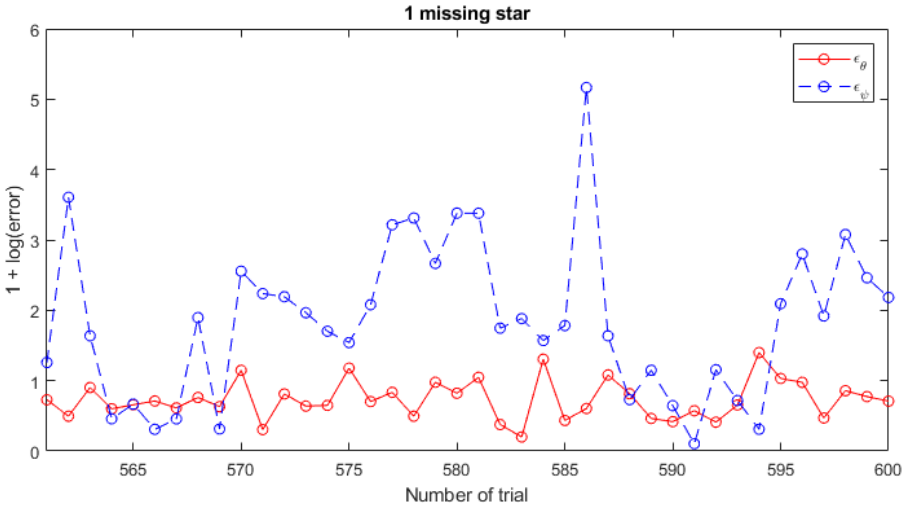

Figure 6.27 : The patterns of ε_ψ and ε_θ derived from the test trials 560-600 using the LSI algorithm for the case with 1 missing star.

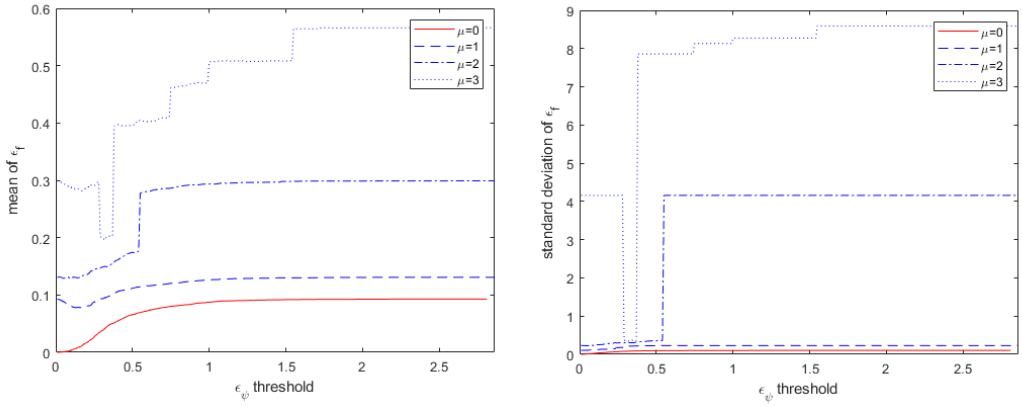

Figure 6.28 : The patterns of ε_f derived from the test results provided by the LSI algorithm with respect to ε_ψ thresholds in the presence of brightness and position noise.

mean and standard deviation are triggered to bounce at the threshold value of 1 and 0.5 respectively. Therefore, it is reasonable to carry out two sets of experiments for the implementation of the RSI algorithm. In the experiments, the error threshold value is assigned to the value of $\tau_{\varepsilon_\psi} = 0.5$ and $\tau_{\varepsilon_\psi} = 1$ to trigger the update mechanism which activates the LSI algorithm to avoid false estimations due to an irrelevant RoS.

The alarm indicator is the key element to implement the feedback scheme. Besides the alarm indicator stimulating the update mechanism when ε_ψ exceeds τ_{ε_ψ}, an indicator is required to implement the initialization procedure after a convergence by means of the update mechanism. This indicator, called DS indicator, is specified by the

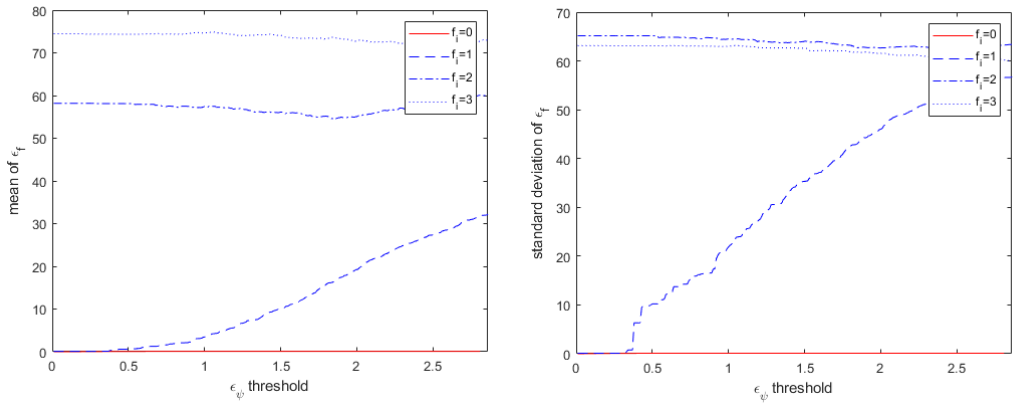

Figure 6.29 : The patterns of ε_f derived from the test results provided by the LSI algorithm with respect to ε_ψ thresholds in the presence of false stars.

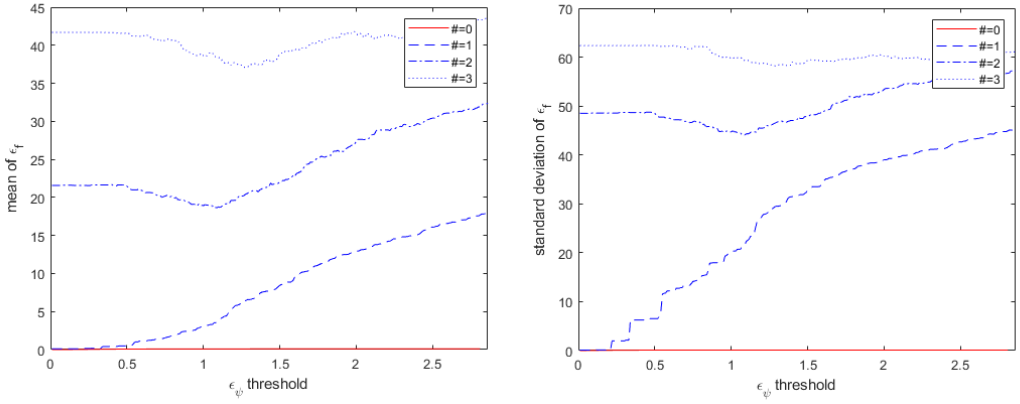

Figure 6.30 : The patterns of ε_f derived from the test results provided by the LSI algorithm with respect to ε_ψ thresholds in the presence of missing stars.

statistical features obtained from the CME algorithm. The experiments carried out to implement the FSF algorithm are used to perform an analysis of correlation between these statistical features. The statistical features shown in Figures 6.18 and 6.19 include the mean of the distances between the true stars and the stars transformed using the estimated affine transformation matrix and their standard deviations. These statistical features are obtained for each set of experiment carried out under different noise conditions.

Figure 6.31 shows the mean of distances between the true stars and the stars transformed using the estimated transformation matrix and their standard deviations without noise for the first 50 trials in the experiment carried out to implement the FSF algorithm. The mean and standard deviation of the distances are calculated using all

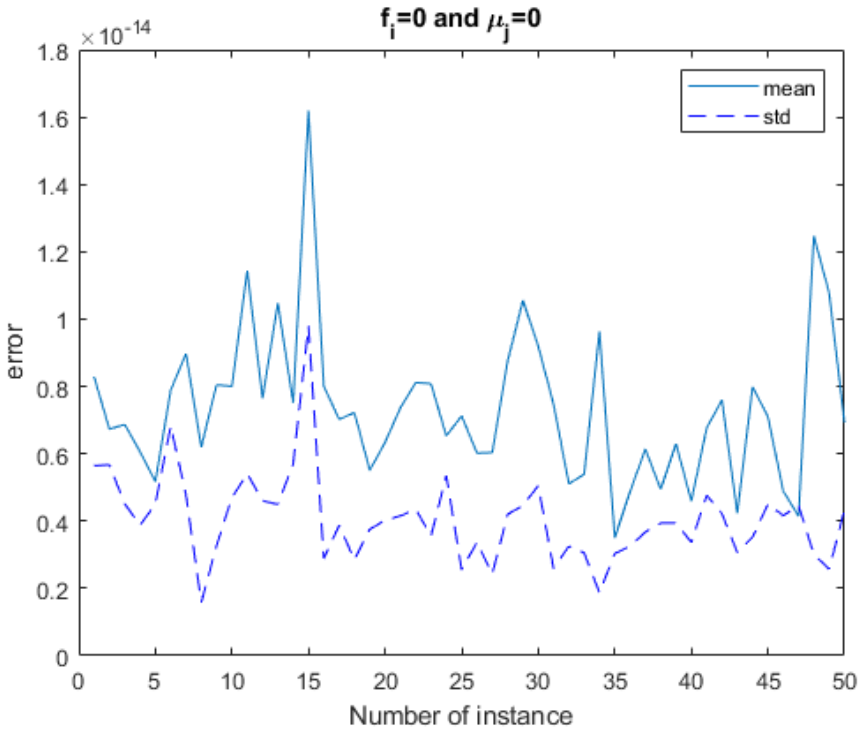

Figure 6.31 : The mean of distances between the true stars and the transformed stars and their standard deviations for the first 50 experiment trials without noise.

stars in each simulated image. The experiment includes 1000 images, thus, 999 trials have been implemented for each sequential pair of images. For a clear demonstration of the relationship between the variables, the first 50 trials are revealed. An apparent correlation between the variables is visible in the graph although they violate the direct proportional relation in a few instances. This phenomenon allows exploitation of these two features to be used in the determination of the RoS in the upcoming implementation of star identification.

The same illustration is made for the same type of experiments under the effect of noise with a noise factor $\mu_j = 1$ and 1 false star as seen in Figure 6.32. There is also a correlation between the variables except that the values of error are larger than the case without noise. Similarly, there are a few trials where the correlation is not applicable. The graphs of the mean and standard deviation curves of the variables are provided to observe the presence of correlation for the cases with increasing noise effects in Figure D.1 for $\mu_j = 2$ and $f_i = 2$, Figure D.2 for $\mu_j = 3$ and $f_i = 3$, Figure D.3 for $\mu_j = 4$ and $f_i = 4$ and Figure D.4 for $\mu_j = 5$ and $f_i = 5$. A similar pattern is valid for all cases but with increasing sizes of error.

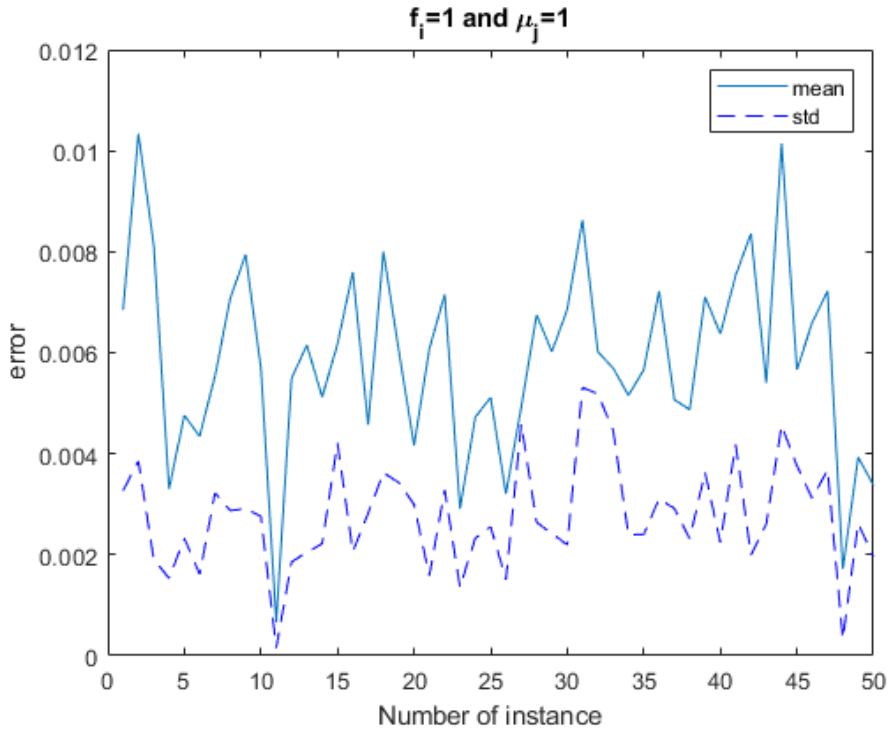

Figure 6.32 : The mean of distances between the true stars and the transformed stars and their standard deviations for the first 50 experiment trials with a noise factor $\mu_j = 1$ in the presence of 1 false star.

The validity of correlation is measured by calculating the correlation coefficient matrix of these two statistical features. The correlation coefficient as well as p-value of two vectors are used to verify the presence of correlation. For this purpose, the statistics of correlation coefficient and p-value are calculated for each case of the mean and standard deviation of distances as given in the figures. The statistics are given in Table 6.5. The p-value statistics is zero at all cases, implying that the null hypothesis is, at each attempt, rejected. The null hypothesis stands for randomness of the variables, or lack of correlation. Thus, this helps claim the presence of correlation between the variables. The scale of correlation is determined by the value of correlation coefficient $\rho_{\mu\sigma}$. As the number of false stars increases together with the noise factor, the correlation coefficient gets close to 1. Therefore, the statistical features are directly proportional in case of larger effect of noise. The estimation of affine transformation matrix yields very successful results in terms of μ_d despite imperfections. Thus, the DS indicator should be dynamically adaptive due to uncertainties in the estimation of motion parameters. Since the statistical features are not totally directly proportional,

Table 6.5 : Correlation coefficient and p-value for μ_d and σ_d.

Noise	Correlation coefficient	p-value
$f_i = 0$ & $\mu_j = 0$	$\rho_{\mu\sigma} = 0.427$	$1.55 \cdot 10^{-45}$
$f_i = 1$ & $\mu_j = 1$	$\rho_{\mu\sigma} = 0.680$	$1.55 \cdot 10^{-136}$
$f_i = 2$ & $\mu_j = 2$	$\rho_{\mu\sigma} = 0.635$	$7.93 \cdot 10^{-114}$
$f_i = 3$ & $\mu_j = 3$	$\rho_{\mu\sigma} = 0.999$	0
$f_i = 4$ & $\mu_j = 4$	$\rho_{\mu\sigma} = 0.997$	0
$f_i = 5$ & $\mu_j = 5$	$\rho_{\mu\sigma} = 0.901$	0

the adaptive property should also reflect the trade-off between the statistical features μ_d and σ_d. Considering that the errors μ_d and σ_d are given in terms of degrees, the RoS is bounded by a region of square with one side of

$$\Gamma_{\text{RoS}} = \Gamma_{\min} + 110^{-\log_{10}(\mu_d)/2} \cdot 110^{-\log_{10}(\sigma_d)/2} \cdot \sigma_d \cdot \mu_d \tag{6.5}$$

in degrees provided that the center of the square is the estimated subsequent candidate boresight vector transformed by the affine transformation matrix $\tilde{\mathbf{H}}$. By this way, the RoS is enlarged for larger values of μ_d and σ_d while the minimal value of Γ_{RoS} is 2 degrees as μ_d approaches zero. As σ_d approaches zero, the RoS is determined only by μ_d. Note that the RSI switches to the LSI algorithm to use the whole database in the case of $\Gamma_{\text{RoS}} \geq 90^\circ$ to guarantee better accuracy against this large error margin.

6.3.2 Statistical performance evaluation

To evaluate the performance of the RSI algorithm, 16 sets of experiments are set up. For each set of experiment, 1001 test images are generated. The sets of experiments are organized with the noise parameters given in Table 6.6.

Table 6.6 : The sets of experiments to evaluate the performance of the RSI algorithm.

f_i \ μ_j	0	1	2	3
0	Set 00	Set 01	Set 02	Set 03
1	Set 10	Set 11	Set 12	Set 13
2	Set 20	Set 21	Set 22	Set 23
3	Set 30	Set 31	Set 32	Set 33

The RSI algorithm, in addition to the FSF algorithm, benefits from the CME algorithm. The outputs of the CME algorithm enables to shrink the RoS and determine the location

of the RoS respectively by means of the parameter Γ_{RoS} and the estimated affine transformation matrix $\tilde{\mathbf{H}}$. Also, an update mechanism is developed using ε_ψ based on the predetermined threshold τ_{ε_ψ}.

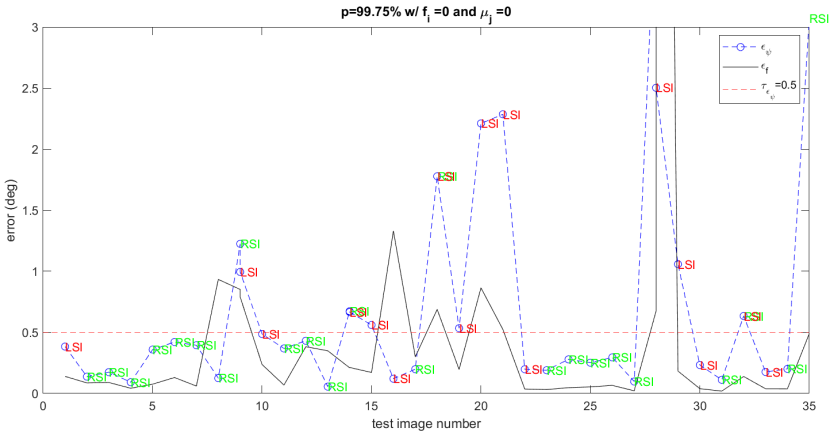

Figure 6.33 : A sequence from the experiment carried out with the database with $p = 99.75\%$ without noise injection to observe the stimulation of the update mechanism in the RSI algorithm with $\tau_{\varepsilon_\psi} = 0.5$.

Figure 6.33 illustrates the patterns of the error for the reconstructed observation vector ε_ψ and the error for the estimated boresight vector ε_f for $\tau_{\varepsilon_\psi} = 0.5$. Once the value of ε_ψ exceeds the threshold τ_{ε_ψ}, the update mechanism is triggered and the LSI algorithm is activated to use the whole database instead of the RSI algorithm that searches a shrunk database. The positive correlation between ε_ψ and ε_f is apparent in most cases. After the LSI method is used in the first image due to lack of feedback information from the MCE algorithm, the RSI algorithm is employed for the next 7 trials because of the condition $\varepsilon_\psi < \tau_{\varepsilon_\psi}$. In the 9th trial, the update mechanism is triggered to activate the LSI algorithm since the threshold τ_{ε_ψ} is exceeded by ε_ψ. Note that the estimation yielding a smaller value for the error ε_ψ is accepted, and the run time is calculated by summing both run time. In this case, since the value of ε_ψ is not reduced below τ_{ε_ψ}, the LSI method is resumed until the condition is satisfied to activate the RSI method. A contrary example is also given. After the update mechanism is triggered in the 18th trial, the condition of $\varepsilon_\psi < \tau_{\varepsilon_\psi}$ is not satisfied for the following 4 trials until when the condition satisfied in the 22nd trial. Thus, the RSI algorithm is used in the 23th trial.

Figure 6.34 shows the patterns of ε_ψ and ε_f for $\tau_{\varepsilon_\psi} = 1$. The values of ε_ψ and ε_f are exactly the same as the values in Figure 6.33. The only difference is that the update

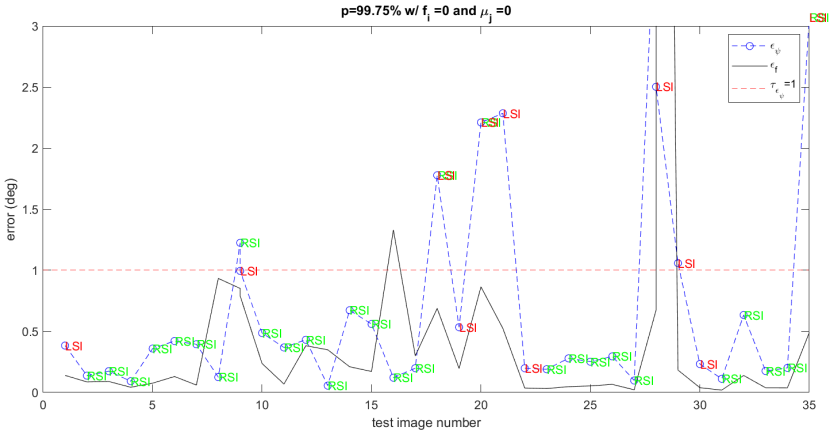

Figure 6.34 : A sequence from the experiment carried out with the database with $p = 99.75\%$ without noise injection to observe the stimulation of the update mechanism in the RSI algorithm with $\tau_{\varepsilon_\psi} = 1$.

mechanism is triggered less often since the value of the predetermined threshold τ_{ε_ψ} is larger.

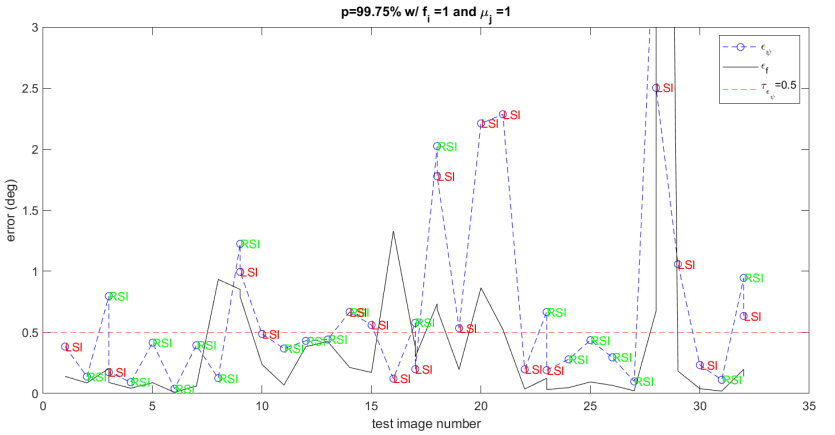

Figure 6.35 : A sequence from the experiment carried out with the database with $p = 99.75\%$ with noise injection $\mu_j = 1$ and $f_i = 1$ to observe the stimulation of the update mechanism in the RSI algorithm with $\tau_{\varepsilon_\psi} = 0.5$.

The patterns of of ε_ψ and ε_f for are also given respectively for $\tau_{\varepsilon_\psi} = 0.5$ and $\tau_{\varepsilon_\psi} = 1$ in Figures 6.35 and 6.36. In this case, the test images are injected with noise $\mu_j = 1$ and $f_i = 1$, implying that the position and brightness noise are added with a noise factor along with one false star. The illustrated curves are not exactly the same when noise added. The stimulation of update mechanism activates the LSI method more often when $\tau_{\varepsilon_\psi} = 0.5$ which reduces ε_ψ in the trials 3, 17, 23 and 32, which also leads to a decrease in ε_f, implying higher accuracy. However, the 15th trial shows

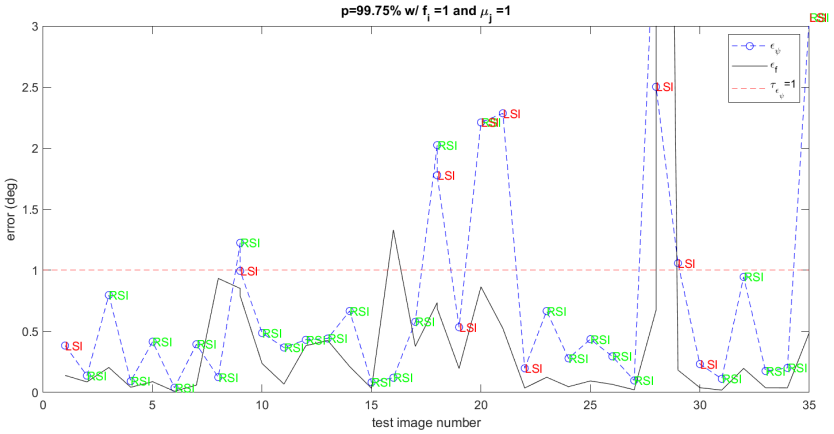

Figure 6.36 : A sequence from the experiment carried out with the database with $p = 99.75\%$ with noise injection $\mu_j = 1$ and $f_i = 1$ to observe the stimulation of the update mechanism in the RSI algorithm with $\tau_{\varepsilon_\psi} = 1$.

a different scenario where the activation of the LSI algorithm lowers the accuracy. This phenomenon reveals that there may be cases when the accuracy is higher with 1NN convergence with a smaller portion of the database. It seems that a smaller value of τ_{ε_ψ} will tend to increase accuracy while leading to larger run time. Moreover, the patterns are also given for the cases of noise injection with the parameters $\mu_j = 2,3$ and $f_i = 2,3$ in Figures E.1, E.2, E.3 and E.4, in which similar patterns can be observed. The detailed performance analyses are given in the next sections.

6.3.2.1 Precision rate

For the sets of given experiments, the curves of precision rate are given. Firstly, the performance of the proposed method is investigated for a combination of different database sizes and threshold values under varying noise conditions. Secondly, additional precision rate curves are obtained to evaluate the effect of noise. In particular, an improvement in performance is investigated in the presence of false stars.

Figures 6.37 and 6.38 show the precision rate curves obtained from the results of the proposed algorithms with different parameters in each case. The results for 6 different scenarios are revealed. While Figure 6.37 shows the results of the set of the experiments without noise injection, Figure 6.38 depicts the precision rate curves for the set of experiments, in which the test images are exposed to one false star and position and brightness noise with a noise factor $\mu_j = 1$. The precision rate curves

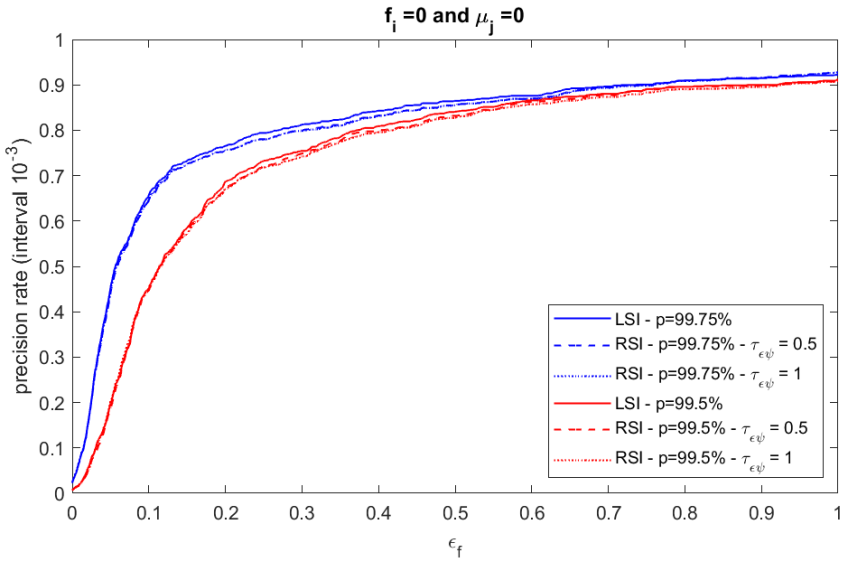

Figure 6.37 : The precision rate curves for $\mu_j = 0$ and $f_i = 0$.

show the percentage of errors upper-bounded by the value of error ε_f projected on the horizontal axis. In addition to the precision rate curves in Figure 6.4, the results for the database generated with an overlap ratio of $p = 99.75\%$ are also taken into account. A larger overlap ratio leads to an increase in accuracy for the RSI algorithm as is the case in the LSI algorithm. The RSI algorithm achieves as high accuracy as the LSI algorithm in the absence of noise as seen in Figure 6.37. Moreover, an increase in the value of threshold τ_{ε_ψ} does not incur a change in accuracy. On the other hand, the accuracy of the RSI algorithm is slightly reduced in comparison with the LSI algorithm with the introduction of noise into the experiment in Figure 6.38. Besides, an increase in the value of τ_{ε_ψ} incurs a slight reduction of accuracy in the presence of noise. Nevertheless, the RSI algorithm, backed up by FSF&CME algorithm, is quite robust to false stars in comparison to the high sensitivity of the LSI algorithm without the FSF algorithm. The precision rate curves are also given for the case of $\mu_j = 2$ and $f_i = 2$ in Figure F.1 and for the case of $\mu_j = 3$ and $f_i = 3$ in Figure F.2. These curves similarly exhibit a high robustness to noise including false stars, as well as slight changes of accuracy between the methods when used with different parameters except that a database with a larger overlap ratio yields better accuracy. Therefore, there is a trade-off between accuracy and runtime since the highest accuracy is yielded by the LSI algorithm supported by with FSF algorithm while the RSI algorithm achieves much shorter run time owing to use of smaller RoS determined by Γ_{RoS}. Note that the

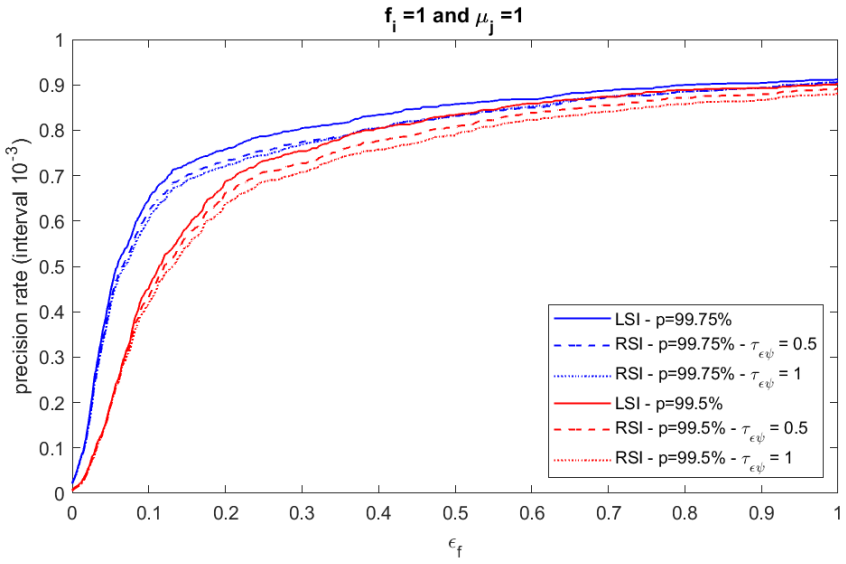

Figure 6.38 : The precision rate curves for $\mu_j = 1$ and $f_i = 1$.

decrease in accuracy is negligible in comparison to the huge reduction in run time. The analysis of run time is given in the next section.

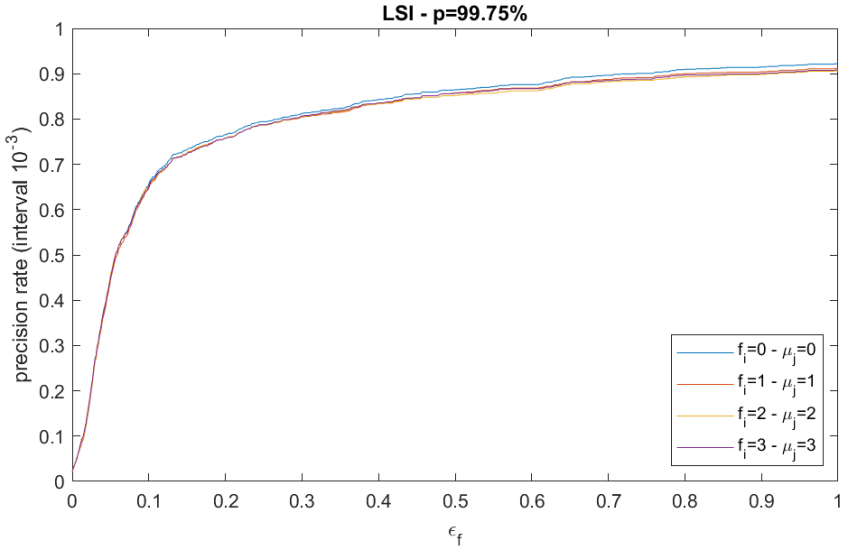

Figure 6.39 : The precision rate curves of the FSF-aided LSI algorithm implemented using a database with $p = 99.75\%$.

Next, the precision rate curves are provided to allow evaluation of the proposed methods with respect to different scenarios of noise injection. Figure 6.39 shows the precision rate curves for the implementation of the FSF-aided LSI algorithm. The LSI algorithm is implemented using a database generated with an overlap ratio of $p = 99.55\%$. Four different curves are given for the implementations of sets of test

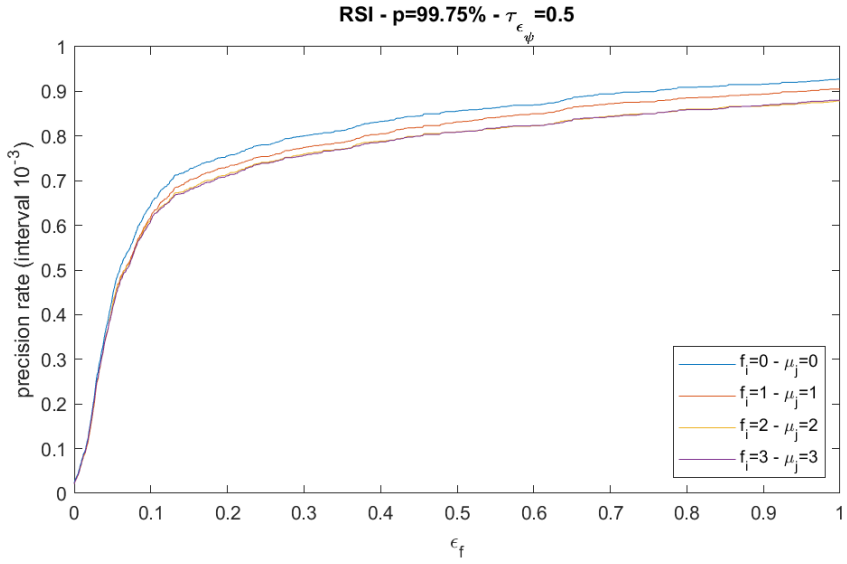

Figure 6.40 : The precision rate curves of the FSF&CME-aided RSI algorithm implemented using a database with $p = 99.75\%$ and a threshold $\tau_{\varepsilon_\psi} = 0.5$.

experiments simulated without noise ($f_i = 0$ and $\mu_j = 0$) and with noise ($f_i = 1,2,3$ and $\mu_j = 1,2,3$). According to the illustration, the LSI algorithm proved to be very robust to noise including false stars after removal of false stars by means of FSF algorithm. The same scenario for the experiments is also implemented using the FSF&CME-aided RSI algorithm. The algorithm uses the same database as the former experiment. Two different scenarios are simulated for that case, in which the threshold triggering the update mechanism is altered. While Figure 6.40 shows the precision rate curves for the threshold assigned to $\tau_{\varepsilon_\psi} = 0.5$, Figure 6.41 depicts the ones for $\tau_{\varepsilon_\psi} = 1$. Despite a slight decrease in terms of accuracy in addition to a larger deviation for larger amount of noise, the reduction in accuracy is very small in comparison to accuracy collapse in the presence of false stars without the FSF algorithm. Since a significant change in accuracy is not the case for $\tau_{\varepsilon_\psi} = 1$, it can be used to decrease run time much further. Additional precision rate curves are also obtained for the experiments carried out with a database generated with an overlap ratio $p = 99.5\%$. Figures F.3, F.4 and F.5 show the precision rate curves for $p = 99.5\%$ under the same noise conditions as it is for $p = 99.75\%$. The pattern are the same as $p = 99.75\%$ except that the accuracy is raised up to 3% in some cases, which is significant considering the importance of high accuracy in ADCS.

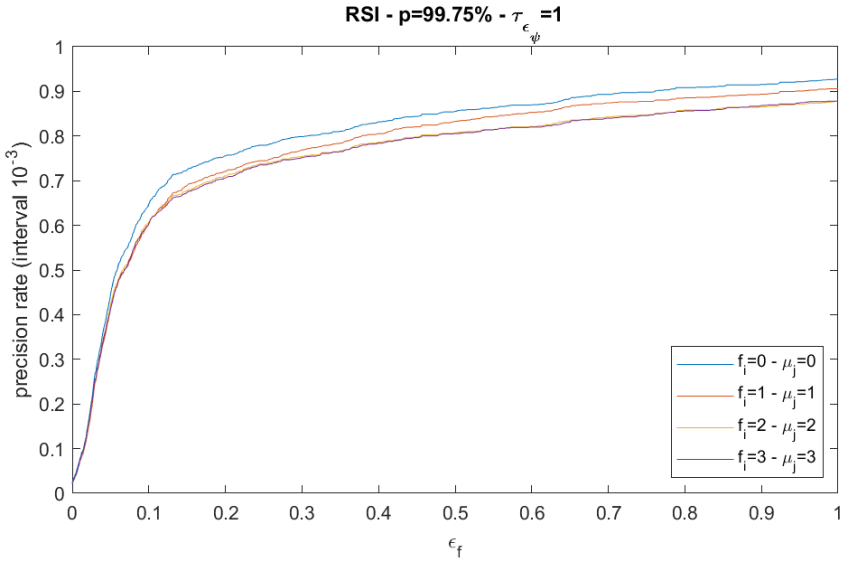

Figure 6.41 : The precision rate curves of the FSF&CME-aided RSI algorithm implemented using a database with $p = 99.75\%$ and a threshold $\tau_{\varepsilon_\psi} = 1$.

The performance indicators are given in terms of the error for the estimated boresight vector ε_f and the error for the estimated rotation angle ε_θ in addition to precision rate curves. The statistical features of the error indicators are given, including mean and standard deviation, in Table H.1 for the experiments carried out using the database generated with $p = 99.5\%$, and in Table H.2 for those with $p = 99.75\%$. The statistical results are provided for the sets of experiments carried out using observations simulated with noise conditions $\mu_j = 0,1,2,3$ and $f_i = 0,1,2,3$ respectively. The performances of the LSI and RSI algorithms are compared. The FSF algorithm is used in all cases to increase accuracy in the presence of false stars. Also, the error threshold for the reconstructed observation vector τ_{ε_ψ} is altered to observe its effect on the performance. In fact, the precision rate curves provide a more meaningful indication of the performance since the mean of error values may demonstrate redundantly large values due to outliers with very large values. Nevertheless, it serves as an indicator to show the effects of the parameters on the accuracy alterations. In the case of $p = 99.5\%$ in Table H.1, the algorithms proved to be very robust to noise, including false stars, in all cases whether τ_{ε_ψ} is used or not since the mean of the error ε_f is always close to 0.18°. In addition, the estimations are very consistent despite increase in noise since the standard deviation of the error ε_f is also close to 0.18° without significant changes. On the other hand, the accuracy of the estimated rotation angle ε_θ is still high compared

to ε_f due to larger values of mean of ε_θ. Its consistency is also higher although slight amount of variation with increase in noise. The introduction of the threshold τ_{ε_ψ} into the system, which converts the LSI algorithm into the RSI algorithm, does not have an adverse impact on the performance. In the worst case where $\mu_j = 3$ and $f_i = 3$, both the mean and the standard deviation of the error ε_f only increase by 0.01° while those of ε_θ by 0.02° in comparison to the LSI algorithm. Also, a larger value of τ_{ε_ψ} has no negative impact on the accuracy although it decreases computational complexity to a large extent. In the case of $p = 99.75\%$, the same pattern is also observed. In addition to that, the values of the error ε_f is reduced by about 0.05° in all cases while a decrease up to 0.15° is observed for the values of the error ε_θ. The standard deviation values are observed to remain nearly constant, which implies that the algorithms function consistently.

6.3.2.2 Complexity analysis

A number of complexity analyses are conducted in terms of run time. Also, the database portion used in the implementation of the FSF&MCE-aided RSI algorithm is determined. Time complexity of the proposed RSI algorithm, just like the LSI algorithm, is determined by feature extraction, 1NN implementation, dictionary setup, LASSO regression and derivation of estimated vectors. The amount of complexity is exactly the same for feature extraction, dictionary setup, LASSO regression and estimation, which are respectively given by $\mathcal{O}(f)$, $\mathcal{O}(1)$, $\mathcal{O}\left(K^3 + K^2 n\right)$ and $\mathcal{O}(1)$ where f is the number of stars in the FoV, K is the number of variables in the feature vector, and n is the number of database feature vectors in the dictionary. On the other hand, the complexity of 1NN implementation is shrunk to $\mathcal{O}(N_{\text{RoS}})$ where N_{RoS} is the number of database feature vectors bounded by the square with one side Γ_{RoS}. Hence, the number of database feature vectors used in the implementation of 1NN classification is given by

$$N_{\text{RoS}} = \frac{\Gamma_{\text{RoS}}^2}{\phi_x \cdot \phi_y \cdot (1-p)^2} \tag{6.6}$$

where Γ_{RoS} is one side of RoS in addition to other notations given in Equation 2.23.

In comparison to the LSI algorithm using the whole database in search of the most significant database feature vector $\overrightarrow{t_5}$, the region of search is significantly contracted

by means of the parameters yielded from the CME algorithm. Therefore, the tradeoff between the 1NN classification and the LASSO regression in terms of run time can be manipulated since the 1NN run time can be reduced to a large extent. Thus, a larger database can be used to increase accuracy further without compromising average run time. The effects of this phenomenon are investigated by plotting run time patterns obtained from the related experiments.

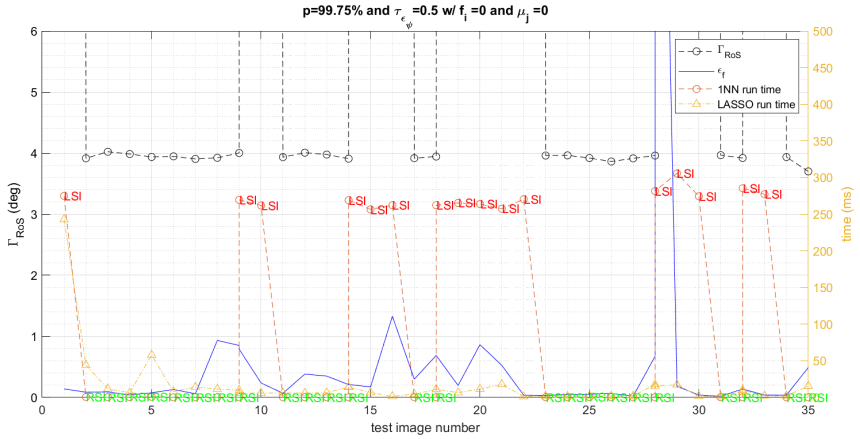

Figure 6.42 : A sequence from the experiment carried out with $p = 99.75\%$ without noise injection to observe the effects of the update mechanism on the run time and memory usage in the RSI algorithm with $\tau_{\varepsilon_\psi} = 0.5$.

Figure 6.42 shows the patterns of the 1NN run time, LASSO run time as well as the RoS side to observe their relation, taking changes in the error for the estimated boresight vector ε_f into account. The database with $p = 99.75\%$ is used in these experiments, and the threshold $\tau_{\varepsilon_\psi} = 0.5$ is chosen without injecting noise in the simulation. The left-hand side and right-hand side axes are given in terms of degrees and ms respectively. Thus, while the curves of ε_f and Γ_{RoS} correspond to the left axis, the run time curves correspond to the right axis. Once the update mechanism is triggered, the LSI algorithm is activated, which leads to divergence of Γ_{RoS}. The size of the error ε_f is reduced with the introduction of the LSI as soon as the condition $\varepsilon_\psi > \tau_{\varepsilon_\psi}$ is met, which avoids the risk of a complete divergence of the RSI algorithm. While the 1NN run time exceeds 250 ms when the LSI algorithm is active, it is reduced to insignificant levels when the RSI algorithm is active. It is also observed that the activation of RSI algorithm causes a decrease in the LASSO run time. Therefore, the

average run time will be reduced much further while increasing accuracy significantly by use of a larger database.

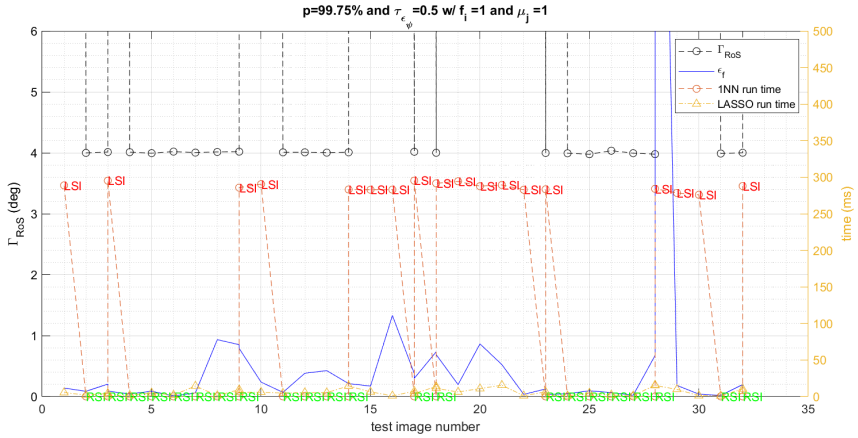

Figure 6.43 : A sequence from the experiment carried out with $p = 99.75\%$ with noise injection ($\mu_j = 1$ and $f_i = 1$) to observe the effects of the update mechanism on the run time and memory usage in the RSI algorithm with $\tau_{\varepsilon_\psi} = 0.5$.

A similar pattern is also observable for the case with noise injection where a noise factor ($\mu_j = 1$) and one false star ($f_i = 1$) are present except a slight increase in the values of Γ_{RoS}, ε_f and 1NN run time. The cases for increasing noise conditions along with $\tau_{\varepsilon_\psi} = 1$ are also given in Figures G.1, G.2, G.3, G.4, G.5 and G.6. The similar patterns revealed, insignificant run time for the RSI algorithm and nearly constant course of Γ_{RoS} when it is not divergent leverage the propositions of consistency in high accuracy and less complexity without compromising in case of more intense noise conditions.

Table 6.7 shows the percentage of the instances when the update mechanism is triggered, and when the condition $\varepsilon_\psi > \tau_{\varepsilon_\psi}$ is satisfied. The given condition is valid for the cases when the error for the reconstructed observation vector exceeds the threshold. The condition cannot be applied on the LSI algorithm since it does not have an update mechanism. The percentage of triggers is different than the percentage of condition satisfaction because the update mechanism can continue to feed the algorithm as long as the condition is satisfied in the subsequent instances. Firstly, an increase in the overlap ratio p has an inverse effect on the activation of the update mechanism, that is, the condition is satisfied less as p increases since the error ε_ψ is smaller. Secondly, an increase in noise also increases the number of instances when the update mechanism is

Table 6.7 : Update mechanism behavior of the algorithm for experiments with varying parameters and noise conditions.

Method	Simulation				Update mechanism	
	p	μ_j	f_i	τ_{ε_ψ}	Trigger	$\varepsilon_\psi > \tau_{\varepsilon_\psi}$
FSF-aided LSI	99.5%	0	0	-	-	-
FSF&CME-aided RSI	99.5%	0	0	0.5	15.1%	35.1%
FSF&CME-aided RSI	99.5%	0	0	1	9.1%	34.8%
FSF-aided LSI	99.5%	1	1	-	-	-
FSF&CME-aided RSI	99.5%	1	1	0.5	19.9%	39.9%
FSF&CME-aided RSI	99.5%	1	1	1	13.9%	40.0%
FSF-aided LSI	99.5%	2	2	-	-	-
FSF&CME-aided RSI	99.5%	2	2	0.5	21.8%	45.1%
FSF&CME-aided RSI	99.5%	2	2	1	16.6%	46.2%
FSF-aided LSI	99.5%	3	3	-	-	-
FSF&CME-aided RSI	99.5%	3	3	0.5	21.9%	45.7%
FSF&CME-aided RSI	99.5%	3	3	1	14.5%	44.9%
FSF-aided LSI	99.75%	0	0	-	-	-
FSF&CME-aided RSI	99.75%	0	0	0.5	12.1%	25.4%
FSF&CME-aided RSI	99.75%	0	0	1	9.0%	24.1%
FSF-aided LSI	99.75%	1	1	-	-	-
FSF&CME-aided RSI	99.75%	1	1	0.5	16.6%	30.0%
FSF&CME-aided RSI	99.75%	1	1	1	13.2%	29.5%
FSF-aided LSI	99.75%	2	2	-	-	-
FSF&CME-aided RSI	99.75%	2	2	0.5	16.8%	33.5%
FSF&CME-aided RSI	99.75%	2	2	1	14.6%	32.9%
FSF-aided LSI	99.75%	3	3	-	-	-
FSF&CME-aided RSI	99.75%	3	3	0.5	16.9%	34.1%
FSF&CME-aided RSI	99.75%	3	3	1	14.5%	34.2%

activated due to higher error values due to higher amount of noise. Lastly, an increase in τ_{ε_ψ} significantly decreases the percentage of triggers while the occurrence of the satisfaction of the condition does not change, which implies that the algorithm tends to resume in the LSI mode for longer time for $\tau_{\varepsilon_\psi} = 1$ than it is for $\tau_{\varepsilon_\psi} = 0.5$ after activation of the update mechanism. The effect of this phenomenon can be discussed more in detail through an investigation of run time statistics. The average run time error for the database generated with $p = 99.75\%$ is nearly 300 ms according to Table 6.1, which makes this database not likely to be used in real-time applications. Whether the use of update mechanism in the RSI algorithm could allow databases larger than

$p = 99.5\%$ to be used in real-time applications is investigated through statistics of database size and average run time.

Table 6.8 : Mean and standard deviation of the RoS and database size used by the algorithm for experiments with varying parameters and noise conditions.

Method	Simulation				RoS in degrees		
	p	μ_j	f_i	τ_{ε_ψ}	μ	σ	kB
FSF-aided LSI	99.5%	0	0	-	-	-	6300
FSF&CME-aided RSI	99.5%	0	0	0.5	3.96	0.05	0.15
FSF&CME-aided RSI	99.5%	0	0	1	3.96	0.05	0.15
FSF-aided LSI	99.5%	1	1	-	-	-	6300
FSF&CME-aided RSI	99.5%	1	1	0.5	4.04	0.02	0.15
FSF&CME-aided RSI	99.5%	1	1	1	4.04	0.02	0.15
FSF-aided LSI	99.5%	2	2	-	-	-	6300
FSF&CME-aided RSI	99.5%	2	2	0.5	4.46	0.02	0.20
FSF&CME-aided RSI	99.5%	2	2	1	4.46	0.02	0.20
FSF-aided LSI	99.5%	3	3	-	-	-	6300
FSF&CME-aided RSI	99.5%	3	3	0.5	4.95	0.02	0.24
FSF&CME-aided RSI	99.5%	3	3	1	4.95	0.02	0.24
FSF-aided LSI	99.75%	0	0	-	-	-	25200
FSF&CME-aided RSI	99.75%	0	0	0.5	3.96	0.05	0.62
FSF&CME-aided RSI	99.75%	0	0	1	3.96	0.05	0.62
FSF-aided LSI	99.75%	1	1	-	-	-	25200
FSF&CME-aided RSI	99.75%	1	1	0.5	4.04	0.02	0.63
FSF&CME-aided RSI	99.75%	1	1	1	4.04	0.02	0.63
FSF-aided LSI	99.75%	2	2	-	-	-	25200
FSF&CME-aided RSI	99.75%	2	2	0.5	4.46	0.02	0.78
FSF&CME-aided RSI	99.75%	2	2	1	4.46	0.02	0.78
FSF-aided LSI	99.75%	3	3	-	-	-	25200
FSF&CME-aided RSI	99.75%	3	3	0.5	4.95	0.02	0.97
FSF&CME-aided RSI	99.75%	3	3	1	4.95	0.02	0.97

Table 6.8 the statistics of the RoS, including mean and standard deviation, obtained by the experiments carried out using different algorithm parameters including p, μ_j, f_i and τ_{ε_ψ}. The statistics is not available for the LSI algorithm because it does not employ the RoS in the 1NN scheme. The RoS is independent from the overlap ratio p while the RoS tends to enlarge as the amount noise injected increases from approximately 4° up to 5°. The standard deviation tends to have very small values, which reveals that the size of the RoS is bounded within a predetermined range by means of dynamical determination of Γ_{RoS} with respect to Equation 6.5. According to the mean of RoS

values in degrees given, the database size used by the RSI algorithm is retrieved according to Equation 6.6. The database size used by the RSI algorithm is negligible in comparison to the LSI algorithm. However, note that the whole database is used upon activation of the update mechanism when the RSI model is switched to the LSI mode.

Table 6.9 : Run time of the algorithm for experiments with varying parameters and noise conditions.

Method	Simulation				Run time (ms)			
	p	μ_j	f_i	τ_{ε_ψ}	1NN		LASSO	
					μ	σ	μ	σ
LSI	99.5%	0	0	-	71.5	2.5	6.6	5.1
RSI	99.5%	0	0	0.5	31.5	35.4	7.6	5.7
RSI	99.5%	0	0	1	15.7	30.0	8.1	6.6
LSI	99.5%	1	1	-	71.7	3.4	6.9	5.3
RSI	99.5%	1	1	0.5	37.0	35.5	8.10	5.8
RSI	99.5%	1	1	1	21.1	32.3	8.4	6.0
LSI	99.5%	2	2	-	71.8	2.7	7.4	5.9
RSI	99.5%	2	2	0.5	41.6	35.2	8.3	6.3
RSI	99.5%	2	2	1	27.2	35.8	9.2	6.9
LSI	99.5%	3	3	-	71.6	2.2	7.4	5.7
RSI	99.5%	3	3	0.5	47.7	40.2	10.1	8.4
RSI	99.5%	3	3	1	25.1	34	9.2	7.4
LSI	99.75%	0	0	-	277.3	2.5	6.6	5.1
RSI	99.75%	0	0	0.5	94.0	135.4	5.3	8.6
RSI	99.75%	0	0	1	60.8	118.8	4.9	4.6
LSI	99.75%	1	1	-	278.7	3.4	6.9	5.3
RSI	99.75%	1	1	0.5	114.8	139.8	5.3	5.0
RSI	99.75%	1	1	1	79.8	127.8	5.4	5.1
LSI	99.75%	2	2	-	281.6	2.7	7.4	5.9
RSI	99.75%	2	2	0.5	154	174.2	7.5	7.6
RSI	99.75%	2	2	1	96.5	135.4	6.0	6.1
LSI	99.75%	3	3	-	289.7	2.2	7.4	5.7
RSI	99.75%	3	3	0.5	129.9	143.9	6.0	5.7
RSI	99.75%	3	3	1	98.9	136.1	6.4	6.7

Table 6.7 shows the run time statistics in terms of ms, obtained in the experiments carried out using different algorithm parameters including p, μ_j, f_i and τ_{ε_ψ}. The introduction of the RSI mode lowers the mean of the run time significantly in comparison with the LSI mode. Despite a slight increase about 2 or 3 ms in the LASSO step, the gain achieved in the 1NN step is very high. Thus, the database

generated with $p = 99.75\%$ can be implemented in real-time since the average run time is lowered down to about 100 ms in the worst case of noise with $\tau_{\varepsilon_\psi} = 1$. The standard deviation peaks when the RSI mode is active because the whole database is used in the 1NN step when the update mechanism activated so that run time takes very long during those instances of estimation. Considering that the accuracy is not affected except that it is now robust to false stars in addition to brightness and position noise, the gain achieved in terms of complexity to reduce database size and run time is a very significant improvement.

6.3.3 Summary

In this section, the RSI algorithm is presented. This algorithm is based on regularized pattern recognition using the same feature extraction as the LSI algorithm. The 1NN classifier and LASSO regression are used as it is used in the LSI algorithm. It differs by an update mechanism integrated into the algorithm by a threshold condition. In addition to the classification and regression parameters, it is also dependent on the update parameters. In order to specify those parameters, a number of experiments are carried out. First, the relationship between the error for the estimated boresight vector ε_f and the error for the reconstructed observation vector ε_ψ is investigated for different noise conditions. A close positive correlation is detected. Thus, the update mechanism is constructed on a threshold that will be applied on ε_ψ. A similar correlation is also detected between ε_θ and ε_ψ, which promotes the proposition. Next, the location and size of the shrunk patch of database is decided by means of the outputs of the CME algorithm. The distance error mean μ_d and standard deviation σ_d are investigated thoroughly. Their very low values and high correlation mean high accuracy and consistency. Hence, these statistical parameters are used to point to the center and one side of the database patch. One side of the patch Γ_{RoS} is determined based on μ_d and σ_d. Statistical performance evaluation is made in terms of precision rate after tracking the update mechanism separately. The activation of update mechanism is investigated for different values of the error threshold τ_{ε_ψ}. After verification of its proper functioning, the performance is compared with the LSI algorithm with different database overlapping ratios under different noise conditions. The update-integrated

algorithm proves to perform very close to the LSI algorithm. Moreover, the algorithm gains robustness to false stars owing to the FSF algorithm. Then, the performance is evaluated in terms of computational complexity and average run time. Memory usage of database and average run time is reduced dramatically. In particular, run time consumption of the 1NN classifier is reduced to insignificant levels when compared to the LSI algorithm because of database shrinkage. Hence, the proposed RSI algorithm proves as high accuracy as the LSI algorithm under different noise scenarios including false stars. In addition, it achieves far less computational complexity and average run time than the LSI algorithm.

7. CONCLUSIONS

7.1 Research Summary and Conclusions

A star identification algorithm integrated with preprocessing techniques is presented in this thesis. Star sensors, which are essential for reliable attitude determination in spacecraft and satellites, depend on these algorithms. The proposed star identification algorithm can be operated either by using the lost-in-space method or by using the recursive method. Both methods use a unique feature extraction scheme, which unconventionally extracts a single vector from each captured image, rather than treating each star as a separate object. This cumulative approach aims to save significant storage space while improving accuracy by processing the entire star catalogue. Using these unconventional features, a database of stars from the catalog is created, with the size and detail of the database varying based on the overlap ratio and brightness threshold parameters. These parameters significantly affect the accuracy and complexity of the method. The algorithm estimates the inertial boresight vector and the rotation angle about it by matching frames rather than individual stars or star patterns. The star identification methods are based on a sequence of pattern recognition and regularization. First, a 1NN classifier provides a coarse estimate by identifying the database vector most similar to the observation vector. A dictionary of neighboring database vectors is then created, and the final estimate is refined by regularization to produce a solution coefficient vector. This vector is used to estimate the boresight vector and rotation angle, which is the output of the lost-in-space star identification. Due to the sensitivity of the lost-in-space algorithm to false stars, an additional filtering algorithm based on density-based clustering is developed. This algorithm removes false stars by analyzing a disparity list generated from successive frames and retaining only the estimated true stars. Using these true stars, an affine transformation matrix is obtained through regression analysis. To address the challenges of the lost-in-space method, a recursive star identification method is developed. This method

incorporates an update mechanism that uses a smaller portion of the database, reducing computational complexity and run time, while filtering out false stars. Thus, the integrated algorithm improves accuracy while reducing computational complexity and run time. The proposed algorithms will be evaluated in a simulation environment that can generate image frames with specified sensor parameters and database information, and simulate different noise scenarios and camera movements. Numerous experiments are performed to determine the optimal parameters for the algorithms and to evaluate their performance. The lost-in-space star identification method is evaluated using error plots, precision rate curves and identification rate curves, while complexity is analyzed by measuring database size and run time. Larger overlap ratios improve accuracy but increase complexity. The proposed algorithm outperforms state-of-the-art methods in accuracy, but falls behind in database size and run time. Optimal parameter selection also involves evaluating the false star filtering algorithm using confusion matrices and statistical indicators, and the camera motion estimation algorithm using distance error measurements. Both algorithms show excellent performance. The recursive star identification algorithm is tested for optimal parameter selection, and compared to the lost-in-space method. It significantly reduces memory usage and run time without compromising accuracy.

7.2 Improvements and Recommendations

The sensitivity to false stars is eliminated, although the algorithm remains sensitive to missing stars. The false star filtering algorithm will be further developed to detect missing stars. As the algorithm relies on features from the entire field-of-view, large objects blocking the view can lead to missing stars, reducing accuracy. This problem is exacerbated by the presence of the Sun or Moon, but efforts are underway to mitigate this by dividing the field-of-view into smaller sections. Finally, the proposed method involves numerous parameter-dependent procedures, making the whole process highly parameter sensitive. This dependency allows flexibility if the procedures are followed

carefully. Future studies are planned to investigate the effect of each parameter on reliability and complexity.

REFERENCES

[1] **Ozyurt, E.O. and Aslan, A.R.** (2024). A lost-in-space star identification algorithm based on regularized pattern recognition, *Acta Astronautica*, *219*, 149–163.

[2] **Ozyurt, E.O. and Aslan, A.R.** (2022). A morphological approach of false star removal and camera motion estimation for star sensors, *11th Nano-Satellite Symposium*, pp.1–5.

[3] **Zhang, G.** (2017). *Star Identification*, Springer Berlin, Heidelberg.

[4] **Toloei, A.**, **Zahednamazi, M.**, **Ghasemi, R. and Mohammadi, F.** (2020). A comparative analysis of star identification algorithms, *Astrophysics and Space Science*, *365*(4), 63.

[5] **Katake, A.**, **Ochoa, J.**, **Zbranek, J.**, **Day, B.**, **Bruccoleri, C. and Goodsell, D.** (2008). Development and Testing of the StarCam SG100: A Stellar Gyroscope, *AIAA Guidance, Navigation and Control Conference and Exhibit*, pp.1–12.

[6] **Ettouati, I.**, **Mortari, D. and Pollock, T.C.** (2006). Space Surveillance using Star Trackers. Part I: Simulations, *2006 AAS Space Flight Mechanics Meeting Conference*, Tampa, FL, pp.2073–2088.

[7] **Abdelkhalik, O.**, **Mortari, D. and Junkins, J.L.** (2006). Space Surveillance with Star Trackers. Part II: Orbit Estimation, *2006 AAS Space Flight Mechanics Meeting Conference*, Tampa, FL, pp.2089–2108.

[8] **Mortari, D.** (2006). Planet and Time Estimation Using Star Trackers, *2006 AAS Space Flight Mechanics Meeting Conference*, Tampa, FL, pp.1853–1866.

[9] **Parish, J.J.**, **Parish, A.S.**, **Swanzy, M.**, **Woodbury, D.**, **Mortari, D. and Junkins, J.L.** (2008). Stellar Positioning System Part I: Applying Ancient Theory to a Modern World, *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, pp.1–15.

[10] **Woodbury, D.**, **Parish, J.J.**, **Parish, A.S.**, **Swanzy, M.**, **Denton, R.**, **Mortari, D. and Junkins, J.L.** (2008). Stellar Positioning System Part II: Overcoming Error During Implementation, *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, pp.1–17.

[11] **Guo, J. and Han, C.** (2016). Where is the limit: The analysis of cubesat ADCS performance, *Small Satellites, System & Services Symposium (4S)*, pp.1–15.

[12] **Yost, B. and Weston, S.** (2021). Small Spacecraft Technology State of the Art Report, **Technical Report**, Small Spacecraft Systems Virtual Institute.

[13] **Requirements, E. and Division, S.**, (2019). Space engineering: Stars sensors terminology and performance specification, European Cooperation for Space Standardization, rev. 2.

[14] **Spratling, B.B. and Mortari, D.** (2009). A Survey on Star Identification Algorithms, *Algorithms*, *2*(1), 93–107.

[15] **Villela, T.**, **Costa, C.A.**, **Brandão, A.M.**, **Bueno, F.T. and Leonardi, R.** (2019). Towards the thousandth cubesat: a statistical overview, *International Journal of Aerospace Engineering*, *2019*, 1–13.

[16] **Aslan, A.R.**, **Yağcı, H.**, **Umit, M.**, **Sofyalı, A.**, **Bas, M.**, **Uludag, M.**, **Ozen, O.**, **Aksulu, M.**, **Yakut, E.**, **Oran, C.**, **Suer, M.**, **Akyol, I.**, **Ecevit, A.**, **Ersöz, M.**, **Öz, I.**, **Gülgönül, S.**, **Dinç, B. and Dengiz, T.** (2013). Development of a LEO communication CubeSat, *2013 6th International Conference on Recent Advances in Space Technologies (RAST)*, pp.637–641.

[17] **Leverone, F.**, **Pini, M.**, **Cervone, A. and Gill, E.** (2020). Solar energy harvesting on-board small satellites, *Renewable Energy*, *159*, 954–972.

[18] **Oztekin, O.**, **Alkaabi, T.**, **Faroukh, Y.**, **Al-Sabt, I.**, **BinAshour, M.**, **Alansaari, M.**, **Alketbi, F.**, **Alhammadi, A.**, **Karabulut, B.**, **Çatal, E.**, **Fernini, I.**, **Aslan, A.R. and Al-Naimiy, H.M.K.** (2022). Development of a novel optical camera subsystem onboard Sharjah-Sat-1 CubeSat, ***R. Navarro and R. Geyl****, editors, Advances in Optical and Mechanical Technologies for Telescopes and Instrumentation V*, volume12188, International Society for Optics and Photonics, SPIE, p.121882M.

[19] **Al., E.K.E.** (2023). The Improved X-ray Detector (iXRD) on Sharjah-Sat-1, design principles, tests and ground calibration, *Experimental Astronomy*, *56*, 99–116.

[20] **Klesh, A.T.**, **Clement, B.G.**, **Colley, C.**, **Essmiller, J.C.**, **Forgette, D.**, **Krajewski, J.**, **Marinan, A.D. and Martin-Mur, T.J.** (2018). MarCO: Early Operations of the First CubeSats to Mars, *32nd Annual AIAA/USU Conference on Small Satellites*, pp.1–6.

[21] **Staehle, R.**, **Blaney, D.**, **Hemmati, H.**, **Jones, D.**, **Klesh, A.**, **Liewer, P.**, **Lazio, J.**, **Lo, M.**, **Mouroulis, P.**, **Murphy, N.**, **Pingree, P.**, **Wilson, T.**, **Puig-suari, P.**, **Williams, A.**, **Svitek, T.**, **Betts, B. and Friedman, L.** (2012). Interplanetary CubeSat Architecture and Missions, *AIAA SPACE Conference and Exposition 2012*, p.5218.

[22] **Walker, R.**, **Binns, D.**, **Bramanti, C.**, **Casasco, M.**, **Concari, P.**, **Izzo, D.**, **Feili, D.**, **Fernandez, P.**, **Fernandez, J.G.**, **Hager, P.**, **Koschny, D.**, **Pesquita, V.**, **Wallace, N.**, **Carnelli, I.**, **Khan, M.**, **Scoubeau, M. and Taubert, D.** (2018). Deep-space CubeSats: thinking inside the box, *Astronomy & Geophysics*, *59*(5), 5.24–5.30.

[23] **Benedetti, G.**, **Bloise, N.**, **Boi, D.**, **Caruso, F.**, **Civita, A.**, **Corpino, S.**, **Garofalo, E.**, **Governale, G.**, **Mascolo, L.**, **Mazzella, G.**, **Quarata, M.**, **Riccobono, D.**, **Sacchiero, G.**, **Teodonio, D. and Vernicari, P.M.** (2019). Interplanetary CubeSats for asteroid exploration: Mission analysis and design, *Acta Astronautica*, *154*, 238–255.

[24] **Curzi, G.**, **Modenini, D. and Tortora, P.** (2020). Large Constellations of Small Satellites: A Survey of Near Future Challenges and Missions, *Aerospace*, *7*(9).

[25] **Farrag, A.**, **Othman, S.**, **Mahmoud, T. and ELRaffiei, A.Y.** (2021). Satellite swarm survey and new conceptual design for Earth observation applications, *The Egyptian Journal of Remote Sensing and Space Science*, *24*(1), 47–54.

[26] **Zhang, J.**, **Cai, Y.**, **Xue, C.**, **Xue, Z. and Cai, H.** (2022). LEO Mega Constellations: Review of Development, Impact, Surveillance, and Governance, *Space: Science and Technology*, *2022*.

[27] **Fil, P.**, **Sengupta, D.**, **Riesco, I.**, **Tortosa, B.S.**, **Krawczyk, B.**, **Ribeiro, G.B.**, **Kwiatkowski, K.**, **Boguslavskiy, M.**, **Liu, C. and Sung, J.** (2023). Mission Concept and Development of the First Interstellar CubeSat Powered by Solar Sailing Technology, *Journal of the British Interplanetary Society*, *76*, 78–86.

[28] **Markley, F.L. and Lerner, G.M.** (1978). Three-axis attitude determination methods., ***J.R. Wertz**, editor, Astrophysics and Space Science Library*, volume 73 of *Astrophysics and Space Science Library*, pp.410–435.

[29] **Shuster, M.D.** (1993). A Survey of Attitude Representations, *The Journal of the Astronautical Sciences*, *41*(4).

[30] **Shuster, M.** (1978). Approximate algorithms for fast optimal attitude computation, *Guidance and Control Conference*, pp.88–95.

[31] **Shuster, M.D. and Oh, S.D.** (1981). Three-axis attitude determination from vector observations, *Journal of Guidance and Control*, *4*(1), 70–77.

[32] **Markley, L.** (1993). Attitude Determination from Vector Observations: A Fast Optimal Matrix Algorithm, *Journal of the Astronautical Sciences*, *41*(2), 261–280.

[33] **Mortari, D.** (1997). ESOQ: A Closed-Form Solution to the Wahba Problem, *Journal of the Astronautical Sciences*, *45*(2), 195–204.

[34] **Mortari, D.** (2000). Second Estimator of the Optimal Quaternion, *Journal of Guidance, Control, and Dynamics*, *23*(5), 885–888.

[35] **Markley, L.** (1988). Attitude Determination Using Vector Observations and the Singular Value Decomposition, *Journal of the Astronautical Sciences*, *36*(3), 245–258.

[36] **Markley, F.L. and Mortari, D.** (2000). Quaternion Attitude Estimation Using Vector Observations, *The Journal of the Astronautical Sciences*, *48*.

[37] **Padgett, C.**, **Kreutz-Delgado, K. and Udomkesmalee, S.** (1997). Evaluation of Star Identification Techniques, *Journal of Guidance Control and Dynamics - J GUID CONTROL DYNAM*, *20*, 259–267.

[38] **Gottlieb, D.** (1978). Mathematical Models of Attitude Hardware, *Spacecraft Attitude Determination and Control; Springer: Leeds, UK*, 256–266.

[39] **Liebe, C.** (1993). Pattern recognition of star constellations for spacecraft applications, *IEEE Aerospace and Electronic Systems Magazine*, *8*(1), 31–39.

[40] **Enright, J. and McVittie, G.R.** (2013). Color star tracking II: matching, *Optical Engineering*, *52*(1), 014406.

[41] **Kosik, J.C.** (1991). Star pattern identification aboard an inertially stabilized aircraft, *Journal of Guidance, Control, and Dynamics*, *14*(2), 230–235.

[42] **Mortari, D.**, **Samaan, M.A.**, **Bruccoleri, C. and Junkins, J.L.** (2004). The Pyramid Star Identification Technique, *NAVIGATION*, *51*(3), 171–183.

[43] **Padgett, C. and Kreutz-Delgado, K.** (1997). A grid algorithm for autonomous star identification, *IEEE Transactions on Aerospace and Electronic Systems*, *33*(1), 202–213.

[44] **Juang, J.N.**, **Kim, H. and Junkins, J.** (2004). An Efficient and Robust Singular Value Method for Star Pattern Recognition and Attitude Determination, *The Journal of the Astronautical Sciences*, *52*, 211–220.

[45] **Wei, X.**, **Zhang, G. and Jiang, J.** (2009). Star Identification Algorithm Based on Log-Polar Transform, *Journal of Aerospace Computing, Information, and Communication*, *6*(8), 483–490.

[46] **Sun, L.**, **Jiang, J.**, **Zhang, G. and Wei, X.** (2016). A Discrete HMM-Based Feature Sequence Model Approach for Star Identification, *IEEE Sensors Journal*, *16*(4), 931–940.

[47] **L., P.**, **B., W. and M., S.** (2003). Star pattern recognition for attitude determination using genetic algorithms, *Algorithms*, *76*, 15–31.

[48] **Zhao, Y.**, **Wei, X.**, **Li, J. and Wang, G.** (2016). Star Identification Algorithm Based on K–L Transformation and Star Walk Formation, *IEEE Sensors Journal*, *16*(13), 5202–5210.

[49] **Zhu, H.**, **Liang, B. and Zhang, T.** (2018). A robust and fast star identification algorithm based on an ordered set of points pattern, *Acta Astronautica*, *148*, 327–336.

[50] **Kim, S. and Cho, M.** (2019). New star identification algorithm using labelling technique, *Acta Astronautica*, *162*, 367–372.

[51] **Roshanian, J.**, **Yazdani, S. and Ebrahimi, M.** (2016). Star identification based on euclidean distance transform, voronoi tessellation, and k-nearest neighbor classification, *IEEE Transactions on Aerospace and Electronic Systems*, *52*(6), 2940–2949.

[52] **Perryman, M.**, **Lindegren, L.**, **Kovalevsky, J.**, **Høg, E.**, **Bastian, U.**, **Bernacca, P.**, **Donati, F.**, **Grenon, M.**, **Van Der Marel, H. and More Authors** (1997). The Hipparcos Catalogue, *Astronomy and Astrophysics*, *323*(1), L49–L52.

[53] **Hoffleit, D.** (1964). *Catalogue of Bright Stars*, Yale Univ. Observatory.

[54] **Salomon, P.M. and Goss, W.C.** (1976). A microprocessor-controlled CCD star tracker, *14th Aerospace Sciences Meeting*, pp.1–6.

[55] **Strikwerda, T.E. and Junkins, J.L.** (1981). Star pattern recognition and spacecraft attitude determination, *Star pattern recognition and spacecraft attitude determination Corps of Engineers*, p.10105.

[56] **Groth, E.J.** (1986). A pattern-matching algorithm for two-dimensional coordinate lists, *Astronomical Journal*, *91*, 1244–1248.

[57] **Sasaki, T. and Kosaka, M.** (1986). A star identification method for satellite attitude determination using star sensors, *15th International Symposium on Space Technology and Science*, volume 2, pp.1125–1130.

[58] **Anderson, D.S.** (1991). *Autonomous star sensing and pattern recognition for spacecraft attitude determination*, Texas A & M University, USA, uMI Order No. GAX91-33905.

[59] **Baldini, D.**, **Barni, M.**, **Foggi, A.**, **Benelli, G. and Mecocci, A.** (1993). A new star-constellation matching algorithm for satellite attitude determination, *ESA Journal*, *17*(2), 185–198.

[60] **Ketchum, E.A. and Tolson, R.H.** (1995). Onboard star identification without a priori attitude information, *Journal of Guidance, Control, and Dynamics*, *18*(2), 242–246.

[61] **Quine, B. and Durrant-Whyte, H.** (1996). A fast autonomous star-acquisition algorithm for spacecraft, *Control Engineering Practice*, *4*(12), 1735–1740.

[62] **Mortari, D.** (1997). Search-less algorithm for star pattern recognition, *The Journal of the Astronautical Sciences*, *45*, 179–194.

[63] **Mortari, D. and Neta, B.** (2000). k-Vector Range Searching Technique, *10th Annual AIAA/AAS Space Flight Mechanics Meeting*, *105*, 449–463.

[64] **Zhang, G.**, **Wei, X. and Jiang, J.** (2008). Full-sky autonomous star identification based on radial and cyclic features of star pattern, *Image and Vision Computing*, *26*(7), 891–897.

[65] **Alvelda, P. and San Martin, A.** (1988). Neural Network Star Pattern Recognition for Spacecraft Attitude Determination and Control, ***D. Touretzky****, editor, Advances in Neural Information Processing Systems*, volume 1, Morgan-Kaufmann, p.314–322.

[66] **Hong, J. and Dickerson, J.A.** (2000). Neural-Network-Based Autonomous Star Identification Algorithm, *Journal of Guidance, Control, and Dynamics*, *23*(4), 728–735.

[67] **Samaan, M.A.**, **Mortari, D. and L., J.J.** (2003). Non-Dimensional Star Identification for Uncalibrated Star Cameras, *Space Flight Mechanics Meeting*, *Paper AAS 03-131*, 913.

[68] **Lamy Au Rousseau, G.**, **Bostel, J. and Mazari, B.** (2005). Star recognition algorithm for APS star tracker: oriented triangles, *IEEE Aerospace and Electronic Systems Magazine*, *20*(2), 27–31.

[69] **Samaan, M.**, **Mortari, D. and Junkins, J.** (2005). Recursive mode star identification algorithms, *IEEE Transactions on Aerospace and Electronic Systems*, *41*(4), 1246–1254.

[70] **Ho, K.** (2012). A survey of algorithms for star identification with low-cost star trackers, *Acta Astronautica*, *73*, 156–163.

[71] **ESA** (1997). *The Hipparcos and Tycho Catalogues*, ESA SP-1200.

[72] **Ho, K. and Nakasuka, S.** (2010). New Star Identification Algorithm for Multiple Star Trackers with Unequal Qualities, *IFAC Proceedings Volumes*, *43*(15), 477–482, 18th IFAC Symposium on Automatic Control in Aerospace.

[73] **Rijlaarsdam, D.**, **Yous, H.**, **Byrne, J.**, **Oddenino, D.**, **Furano, G. and Moloney, D.** (2020). A Survey of Lost-in-Space Star Identification Algorithms Since 2009, *Sensors*, *20*(9).

[74] **Paladugu, L.**, **Seisie-Amoasi, E.**, **Williams, B.G. and Schoen, M.P.** (2006). Intelligent Star Pattern Recognition for Attitude Determination: the "Lost in Space" Problem, *Journal of Aerospace Computing, Information, and Communication*, *3*(11), 538–549.

[75] **Wei, X.**, **Wen, D.**, **Song, Z. and Xi, J.** (2019). Star Identification Algorithm Based on Oriented Singular Value Feature and Reliability Evaluation Method, *Transactions of the Japan Society for Aeronautical and Space Sciences*, *62*, 265–274.

[76] **Na, M.**, **Zheng, D. and Jia, P.** (2009). Modified Grid Algorithm for Noisy All-Sky Autonomous Star Identification, *IEEE Transactions on Aerospace and Electronic Systems*, *45*(2), 516–522.

[77] **Aghaei, M. and Moghaddam, H.A.** (2016). Grid star identification improvement using optimization approaches, *IEEE Transactions on Aerospace and Electronic Systems*, *52*(5), 2080–2090.

[78] **Quan, W. and Fang, J.** (2010). A Star Recognition Method Based on the Adaptive Ant Colony Algorithm for Star Sensors, *Sensors (Basel, Switzerland)*, *10*, 1955–66.

[79] **Yoon, H.**, **Lim, Y. and Bang, H.** (2011). New Star-Pattern Identification Using a Correlation Approach for Spacecraft Attitude Determination, *Journal of Spacecraft and Rockets*, *48*(1), 182–186.

[80] **Delabie, T.**, **Schutter, J.D. and Vandenbussche, B.** (2013). Highly Efficient Attitude-Estimation Algorithm for Star Trackers Using Optimal Image Matching, *Journal of Guidance, Control, and Dynamics*, *36*(6), 1672–1680.

[81] **Silani, E. and Lovera, M.** (2006). Star identification algorithms: novel approach & comparison study, *IEEE Transactions on Aerospace and Electronic Systems*, *42*(4), 1275–1288.

[82] **Li, J.**, **Wei, X. and Zhang, G.** (2015). Iterative algorithm for autonomous star identification, *IEEE Transactions on Aerospace and Electronic Systems*, *51*(1), 536–547.

[83] **Schiattarella, V.**, **Spiller, D. and Curti, F.** (2017). A novel star identification technique robust to high presence of false objects: The Multi-Poles Algorithm, *Advances in Space Research*, *59*(8), 2133–2147.

[84] **Xu, L.**, **Jiang, J. and Liu, L.** (2019). RPNet: A Representation Learning-Based Star Identification Algorithm, *IEEE Access*, *7*, 92193–92202.

[85] **Cover, T.** (1968). Estimation by the nearest neighbor rule, *IEEE Transactions on Information Theory*, *14*(1), 50–55.

[86] **Hart, P.** (1968). The condensed nearest neighbor rule (corresp.), *IEEE transactions on information theory*, *14*(3), 515–516.

[87] **Yoon, Y.** (2013). Autonomous star identification using pattern code, *IEEE Transactions on Aerospace and Electronic Systems*, *49*(3), 2065–2072.

[88] **Williams, L.F.** (1976). A modification to the half-interval search (binary search) method, *Proceedings of the 14th Annual Southeast Regional Conference*, ACM-SE 14, Association for Computing Machinery, New York, NY, USA, p.95–101.

[89] **Tibshirani, R.** (1996). Regression Shrinkage and Selection via the Lasso, *Journal of the Royal Statistical Society (Series B)*, *58*, 267–288.

[90] **Ozyurt, E.O. and Gunsel, B.** (2018). Wami Object Tracking Using L1 Tracker Integrated with a Deep Detector, *2018 25th IEEE International Conference on Image Processing (ICIP)*, pp.2690–2694.

[91] **Gurkan, F. and Gunsel, B.** (2021). Integration of regularized l1 tracking and instance segmentation for video object tracking, *Neurocomputing*, *423*, 284–300.

[92] **Mei, X. and Ling, H.** (2009). Robust visual tracking using l1 minimization, *2009 IEEE 12th International Conference on Computer Vision*, pp.1436–1443.

[93] **Liu, M.**, **Wei, X.**, **Wen, D. and Wang, H.** (2021). Star Identification Based on Multilayer Voting Algorithm for Star Sensors, *Sensors*, *21*(9).

[94] **Li, J.**, **Wei, X.**, **Wang, G. and Zhou, S.** (2020). Improved Grid Algorithm Based on Star Pair Pattern and Two-dimensional Angular Distances for Full-Sky Star Identification, *IEEE Access*, *8*, 1010–1020.

[95] **van Bezooijen, R.W.H.** (1996). Autonomous star trackers for geostationary satellites, ***E.R. Washwell****, editor, GOES-8 and Beyond*, volume2812, International Society for Optics and Photonics, SPIE, pp.847 – 858.

[96] **Kolomenkin, M.**, **Pollak, S.**, **Shimshoni, I. and Lindenbaum, M.** (2008). Geometric voting algorithm for star trackers, *IEEE Transactions on Aerospace and Electronic Systems*, *44*(2), 441–456.

[97] **Lee, H.**, **Oh, C.S. and Bang, H.** (2003). Modified grid algorithm for star pattern identification by using star trackers, *International Conference on Recent Advances in Space Technologies, 2003. RAST '03. Proceedings of*, pp.385–391.

[98] **Luo, L.**, **Xu, L. and Zhang, H.** (2015). An Autonomous Star Identification Algorithm Based on One-Dimensional Vector Pattern for Star Sensors, *Sensors*, *15*(7), 16412–16429.

[99] **Rao, S.**, **Sathyanarayanan, S.**, **Tejaswini, C.M. and Rohit, B.** (2021). Star Sensor Algorithm Using the Overlapping Grid Method for Attitude Determination First, ***L. Garg, H. Sharma, S.B. Goyal and A. Singh****, editors, Proceedings of International Conference on Innovations in Information and Communication Technologies*, Springer Singapore, Singapore, pp.111–121.

[100] **Rijlaarsdam, D.**, **Yous, H.**, **Byrne, J.**, **Oddenino, D.**, **Furano, G. and Moloney, D.** (2020). Efficient Star Identification Using a Neural Network, *Sensors*, *20*(13).

[101] **You, Z.**, **Li, J.**, **Zhang, H.**, **Yang, B. and Le, X.** (2022). An accurate star identification approach based on spectral graph matching for attitude measurement of spacecraft, *Complex & Intelligent Systems*, *8*, 1639–1652.

[102] **Schulz, V.H.**, **Marcelino, G.M.**, **Seman, L.O.**, **Barros, J.S.**, **Kim, S.**, **Cho, M.**, **Villarrubia, G.**, **Leithardt, V.R.Q. and Bezerra, E.A.** (2021). Universal Verification Platform and Star Simulator for Fast Star Tracker Design, *Sensors (Basel, Switzerland)*, *21*.

[103] **Wang, G.**, **Lv, W.**, **Li, J. and Wei, X.** (2019). False Star Filtering for Star Sensor Based on Angular Distance Tracking, *IEEE Access*, *7*, 62401–62411.

[104] **Berrighi, G. and Procopio, D.** (2003). Star Sensors and Harsh Environment, *Proceedings International ESA Conference Spacecraft Guidance*, Navigation and Control Systems, pp.19–25.

[105] **Lauer, M.**, **Jauregui, L. and Kielbassa, S.** (2007). Operational Experience with Autonomous Star Trackers on ESA Interplanetary Spacecraft, *Operational experience with autonomous star trackers on ESA interplanetary spacecraft*, pp.1–6.

[106] **Wang, G.**, **Li, J. and Wei, X.** (2018). Star Identification Based on Hash Map, *IEEE Sensors Journal*, *18*(4), 1591–1599.

[107] **Xie, J.**, **Tang, X.**, **Jiang, W. and Fu, X.** (2012). An Autonomous Star Identification Algorithm Based on the Directed Circularity Pattern, *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *XXXIX-B1*, 333–338.

[108] **Christian, J.A. and Crassidis, J.L.** (2021). Star Identification and Attitude Determination With Projective Cameras, *IEEE Access*, *9*, 25768–25794.

[109] **Liu, H.**, **Wei, X.**, **Li, J. and Wang, G.** (2021). A star identification algorithm based on simplest general subgraph, *Acta Astronautica*, *183*, 11–22.

[110] **Fialho, M.A.A. and Mortari, D.** (2019). Theoretical Limits of Star Sensor Accuracy, *Sensors*, *19*(24).

[111] **Markley, F.L. and L., C.J.** (2014). *Fundamentals of spacecraft attitude determination and control*, Springer.

[112] **Wei, X.**, **Wen, D.**, **Song, Z.**, **Xi, J.**, **Zhang, W.**, **Liu, G. and Li, Z.** (2019). A star identification algorithm based on radial and dynamic cyclic features of star pattern, *Advances in Space Research*, *63*(7), 2245–2259.

[113] **Babenko, B.**, **Yang, M.H. and Belongie, S.** (2011). Robust Object Tracking with Online Multiple Instance Learning, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *33*(8), 1619–1632.

[114] **CubeSpace**, (2020). CubeStar Interface Control Document, CubeSpace, ver. 1.3.

[115] **Aslan, A.R.**, **AlNaimiy, H.M.**, **Fernini, I.**, **Manousakis, A.**, **Shaikh, M.**, **Madara, S.R.**, **Al-Kaabi, T.**, **Karabulut, B.**, **Oztekin, O.**, **Turkoglu, S. and Kalemci, E.** (2019). Space Technology Capacity Building in Support of SDG 2030 Through CubeSat SharjahSat-l, *2019 9th International Conference on Recent Advances in Space Technologies (RAST)*, pp.955–958.

[116] **Udomkesmalee, S.**, **Alexander, J.W. and Tolivar, A.F.** (1994). Stochastic star identification, *Journal of Guidance, Control, and Dynamics*, *17*(6), 1283–1286.

[117] **Zou, H. and Hastie, T.** (2005). Regularization and variable selection via the Elastic Net, *Journal of the Royal Statistical Society, Series B*, *67*, 301–320.

[118] **Ester, M.**, **Kriegel, H.P.**, **Sander, J. and Xu, X.** (1996). A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise, *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, AAAI Press, p.226–231.

[119] **Torr, P. and Zisserman, A.** (2000). MLESAC: A New Robust Estimator with Application to Estimating Image Geometry, *Computer Vision and Image Understanding*, *78*(1), 138–156.

[120] **Fischler, M.A. and Bolles, R.C.**, (1987). Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography, **M.A. Fischler and O. Firschein**, editors, Readings in Computer Vision, Morgan Kaufmann, San Francisco (CA), pp.726–740.

[121] **Fisher, R.A.**, (1992). Statistical Methods for Research Workers, **S. Kotz and N.L. Johnson**, editors, Breakthroughs in Statistics: Methodology and Distribution, Springer New York, New York, pp.66–70.

[122] **Kendall, M.G.** (1979). *The Advanced Theory of Statistics*, Macmillan, 4th ed edition.

[123] **Wu, Y.**, **Lim, J. and Yang, M.H.** (2015). Object Tracking Benchmark, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *37*(9), 1834–1848.

[124] **Kim, K. and Bang, H.** (2020). Algorithm with Patterned Singular Value Approach for Highly Reliable Autonomous Star Identification, *Sensors*, *20*(2).

[125] **Efron, B.**, **Hastie, T.**, **Johnstone, I. and Tibshirani, R.** (2004). Least angle regression, *The Annals of Statistics*, *32*(2), 407 – 499.

[126] **Christen, P.**, **Hand, D.J. and Kirielle, N.** (2023). A Review of the F-Measure: Its History, Properties, Criticism, and Alternatives, *ACM Computing Surveys*, *56*(3).

APPENDICES

APPENDIX A : Confusion Matrices in False Star Filtering
APPENDIX B : Error Patterns in Star Identification
APPENDIX C : Investigation of Error Threshold
APPENDIX D : Statistical Features of Distance Error
APPENDIX E : Patterns of Errors with Different Thresholds
APPENDIX F : Precision Rate Curves
APPENDIX G : Run Time Patterns
APPENDIX H : Error Tables

APPENDIX A : Confusion Matrices in False Star Filtering

Table A.1 : Confusion matrix for $f_i = 0$ with only brightness noise.

			Ground truth	
			True	False
$\mu_j = 0$	Estimation	Positive	6617	0
		Negative	0	0
$\mu_j = 1$			True	False
	Estimation	Positive	6591	0
		Negative	26	0
$\mu_j = 2$			True	False
	Estimation	Positive	6555	0
		Negative	63	0
$\mu_j = 3$			True	False
	Estimation	Positive	6511	0
		Negative	108	0
$\mu_j = 4$			True	False
	Estimation	Positive	6460	0
		Negative	158	0
$\mu_j = 5$			True	False
	Estimation	Positive	6446	0
		Negative	174	0

Table A.2 : Confusion matrix for $f_i = 0$ with only position noise.

			Ground truth	
			True	False
$\mu_j = 0$	Estimation	Positive	6617	0
		Negative	0	0
$\mu_j = 1$			True	False
	Estimation	Positive	6409	0
		Negative	213	0
$\mu_j = 2$			True	False
	Estimation	Positive	6052	0
		Negative	570	0
$\mu_j = 3$			True	False
	Estimation	Positive	5738	0
		Negative	888	0
$\mu_j = 4$			True	False
	Estimation	Positive	5410	0
		Negative	1216	0
$\mu_j = 5$			True	False
	Estimation	Positive	5291	0
		Negative	1341	0

Table A.3 : Confusion matrix for $f_i = 0$ with both brightness and position noise.

			Ground truth	
			True	False
$\mu_j = 0$	Estimation	Positive	6617	0
		Negative	0	0
$\mu_j = 1$			True	False
	Estimation	Positive	6383	0
		Negative	238	0
$\mu_j = 2$			True	False
	Estimation	Positive	6073	0
		Negative	549	0
$\mu_j = 3$			True	False
	Estimation	Positive	5684	0
		Negative	942	0
$\mu_j = 4$			True	False
	Estimation	Positive	5438	0
		Negative	1186	0
$\mu_j = 5$			True	False
	Estimation	Positive	5199	0
		Negative	1421	0

Table A.4 : Confusion matrix for $f_i = 1$ with only brightness noise.

			Ground truth	
			True	False
$\mu_j = 0$	Estimation	Positive	6608	0
		Negative	9	976
$\mu_j = 1$			True	False
	Estimation	Positive	6570	0
		Negative	40	977
$\mu_j = 2$			True	False
	Estimation	Positive	6538	2
		Negative	80	969
$\mu_j = 3$			True	False
	Estimation	Positive	6473	0
		Negative	145	964
$\mu_j = 4$			True	False
	Estimation	Positive	6454	0
		Negative	164	987
$\mu_j = 5$			True	False
	Estimation	Positive	6430	0
		Negative	189	985

Table A.5 : Confusion matrix for $f_i = 1$ with only position noise.

			Ground truth	
			True	False
$\mu_j = 0$	Estimation	Positive	6608	0
		Negative	9	976
$\mu_j = 1$			True	False
	Estimation	Positive	6360	0
		Negative	260	979
$\mu_j = 2$			True	False
	Estimation	Positive	6025	0
		Negative	598	977
$\mu_j = 3$			True	False
	Estimation	Positive	5719	0
		Negative	909	978
$\mu_j = 4$			True	False
	Estimation	Positive	5421	0
		Negative	1202	979
$\mu_j = 5$			True	False
	Estimation	Positive	5245	0
		Negative	1377	981

Table A.6 : Confusion matrix for $f_i = 1$ with both brightness and position noise.

			Ground truth	
			True	False
$\mu_j = 0$	Estimation	Positive	6608	0
		Negative	9	976
$\mu_j = 1$			True	False
	Estimation	Positive	6368	1
		Negative	252	974
$\mu_j = 2$			True	False
	Estimation	Positive	6072	0
		Negative	548	972
$\mu_j = 3$			True	False
	Estimation	Positive	5680	0
		Negative	945	982
$\mu_j = 4$			True	False
	Estimation	Positive	5381	2
		Negative	1249	981
$\mu_j = 5$			True	False
	Estimation	Positive	5228	0
		Negative	1395	979

Table A.7 : Confusion matrix for $f_i = 2$ with only brightness noise.

			Ground truth	
			True	False
$\mu_j = 0$	Estimation	Positive	6594	0
		Negative	23	1958
$\mu_j = 1$			True	False
	Estimation	Positive	6552	1
		Negative	66	1947
$\mu_j = 2$			True	False
	Estimation	Positive	6509	0
		Negative	109	1955
$\mu_j = 3$			True	False
	Estimation	Positive	6453	0
		Negative	164	1957
$\mu_j = 4$			True	False
	Estimation	Positive	6413	0
		Negative	206	1953
$\mu_j = 5$			True	False
	Estimation	Positive	6400	0
		Negative	220	1950

Table A.8 : Confusion matrix for $f_i = 2$ with only position noise.

			Ground truth	
			True	False
$\mu_j = 0$	Estimation	Positive	6594	0
		Negative	23	1958
$\mu_j = 1$			True	False
	Estimation	Positive	6350	0
		Negative	260	1958
$\mu_j = 2$			True	False
	Estimation	Positive	6031	1
		Negative	591	1961
$\mu_j = 3$			True	False
	Estimation	Positive	5663	0
		Negative	958	1969
$\mu_j = 4$			True	False
	Estimation	Positive	5408	1
		Negative	1208	1951
$\mu_j = 5$			True	False
	Estimation	Positive	5283	0
		Negative	1337	1962

Table A.9 : Confusion matrix for $f_i = 2$ with both brightness and position noise.

			Ground truth	
			True	False
$\mu_j = 0$	Estimation	Positive	6594	0
		Negative	23	1958
$\mu_j = 1$			True	False
	Estimation	Positive	6395	0
		Negative	222	1959
$\mu_j = 2$			True	False
	Estimation	Positive	6042	0
		Negative	580	1957
$\mu_j = 3$			True	False
	Estimation	Positive	5638	1
		Negative	989	1946
$\mu_j = 4$			True	False
	Estimation	Positive	5401	0
		Negative	1227	1956
$\mu_j = 5$			True	False
	Estimation	Positive	5297	2
		Negative	1329	1961

Table A.10 : Confusion matrix for $f_i = 3$ with only brightness noise.

			Ground truth	
			True	False
$\mu_j = 0$	Estimation	Positive	6577	0
		Negative	40	2925
$\mu_j = 1$			True	False
	Estimation	Positive	6555	1
		Negative	63	2939
$\mu_j = 2$			True	False
	Estimation	Positive	6464	0
		Negative	153	2949
$\mu_j = 3$			True	False
	Estimation	Positive	6427	4
		Negative	192	2928
$\mu_j = 4$			True	False
	Estimation	Positive	6356	1
		Negative	262	2949
$\mu_j = 5$			True	False
	Estimation	Positive	6356	1
		Negative	264	2929

Table A.11 : Confusion matrix for $f_i = 3$ with only position noise.

			Ground truth	
			True	False
$\mu_j = 0$	Estimation	Positive	6577	0
		Negative	40	2925
$\mu_j = 1$			True	False
	Estimation	Positive	6312	0
		Negative	307	2934
$\mu_j = 2$			True	False
	Estimation	Positive	5930	2
		Negative	691	2935
$\mu_j = 3$			True	False
	Estimation	Positive	5651	2
		Negative	968	2935
$\mu_j = 4$			True	False
	Estimation	Positive	5352	4
		Negative	1277	2930
$\mu_j = 5$			True	False
	Estimation	Positive	5201	5
		Negative	1422	2918

Table A.12 : Confusion matrix for $f_i = 3$ with both brightness and position noise.

			Ground truth	
			True	False
$\mu_j = 0$	Estimation	Positive	6577	0
		Negative	40	2925
$\mu_j = 1$			True	False
	Estimation	Positive	6354	0
		Negative	268	2934
$\mu_j = 2$			True	False
	Estimation	Positive	5950	1
		Negative	672	2929
$\mu_j = 3$			True	False
	Estimation	Positive	5638	3
		Negative	987	2936
$\mu_j = 4$			True	False
	Estimation	Positive	5302	3
		Negative	1324	2935
$\mu_j = 5$			True	False
	Estimation	Positive	5193	3
		Negative	1433	2929

Table A.13 : Confusion matrix for $f_i = 4$ with only brightness noise.

			Ground truth	
			True	False
$\mu_j = 0$	Estimation	Positive	6500	1
		Negative	117	3906
$\mu_j = 1$			True	False
	Estimation	Positive	6483	0
		Negative	134	3918
$\mu_j = 2$			True	False
	Estimation	Positive	6455	5
		Negative	163	3918
$\mu_j = 3$			True	False
	Estimation	Positive	6402	1
		Negative	216	3909
$\mu_j = 4$			True	False
	Estimation	Positive	6340	2
		Negative	277	3925
$\mu_j = 5$			True	False
	Estimation	Positive	6319	2
		Negative	300	3934

Table A.14 : Confusion matrix for $f_i = 4$ with only position noise.

			Ground truth	
			True	False
$\mu_j = 0$	Estimation	Positive	6500	1
		Negative	117	3906
$\mu_j = 1$			True	False
	Estimation	Positive	6317	2
		Negative	301	3924
$\mu_j = 2$			True	False
	Estimation	Positive	5966	3
		Negative	659	3919
$\mu_j = 3$			True	False
	Estimation	Positive	5550	5
		Negative	1076	3904
$\mu_j = 4$			True	False
	Estimation	Positive	5454	7
		Negative	1166	3902
$\mu_j = 5$			True	False
	Estimation	Positive	5112	6
		Negative	1509	3911

Table A.15 : Confusion matrix for $f_i = 4$ with both brightness and position noise.

			Ground truth	
			True	False
$\mu_j = 0$	Estimation	Positive	6500	1
		Negative	117	3906
$\mu_j = 1$			True	False
	Estimation	Positive	6291	1
		Negative	328	3907
$\mu_j = 2$			True	False
	Estimation	Positive	5889	5
		Negative	735	3916
$\mu_j = 3$			True	False
	Estimation	Positive	5584	3
		Negative	1042	3904
$\mu_j = 4$			True	False
	Estimation	Positive	5375	2
		Negative	1250	3924
$\mu_j = 5$			True	False
	Estimation	Positive	5222	8
		Negative	1408	3896

Table A.16 : Confusion matrix for $f_i = 5$ with only brightness noise.

			Ground truth	
			True	False
$\mu_j = 0$	Estimation	Positive	6492	2
		Negative	125	4900
$\mu_j = 1$			True	False
	Estimation	Positive	6439	6
		Negative	177	4890
$\mu_j = 2$			True	False
	Estimation	Positive	6413	2
		Negative	204	4897
$\mu_j = 3$			True	False
	Estimation	Positive	6362	2
		Negative	255	4905
$\mu_j = 4$			True	False
	Estimation	Positive	6277	4
		Negative	340	4903
$\mu_j = 5$			True	False
	Estimation	Positive	6271	6
		Negative	349	4889

Table A.17 : Confusion matrix for $f_i = 5$ with only position noise.

			Ground truth	
			True	False
$\mu_j = 0$	Estimation	Positive	6492	2
		Negative	125	4900
$\mu_j = 1$			True	False
	Estimation	Positive	6220	7
		Negative	400	4889
$\mu_j = 2$			True	False
	Estimation	Positive	5887	2
		Negative	738	4905
$\mu_j = 3$			True	False
	Estimation	Positive	5495	5
		Negative	1124	4875
$\mu_j = 4$			True	False
	Estimation	Positive	5283	9
		Negative	1336	4845
$\mu_j = 5$			True	False
	Estimation	Positive	5208	5
		Negative	1417	4879

Table A.18 : Confusion matrix for $f_i = 5$ with both brightness and position noise.

			Ground truth	
			True	False
$\mu_j = 0$	Estimation	Positive	6492	2
		Negative	125	4900
$\mu_j = 1$			True	False
	Estimation	Positive	6212	4
		Negative	407	4901
$\mu_j = 2$			True	False
	Estimation	Positive	5871	4
		Negative	750	4889
$\mu_j = 3$			True	False
	Estimation	Positive	5561	2
		Negative	1058	4903
$\mu_j = 4$			True	False
	Estimation	Positive	5251	4
		Negative	1370	4868
$\mu_j = 5$			True	False
	Estimation	Positive	5147	9
		Negative	1477	4886

APPENDIX B : Error Patterns in Star Identification

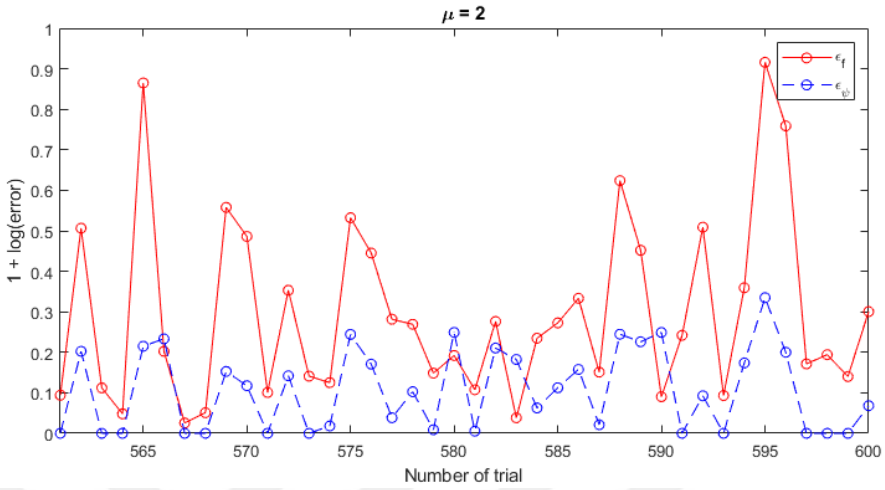

Figure B.1 : The patterns of ε_ψ and ε_f derived from the test trials 560-600 using the LSI algorithm for the case with a noise factor $\mu_j = 2$.

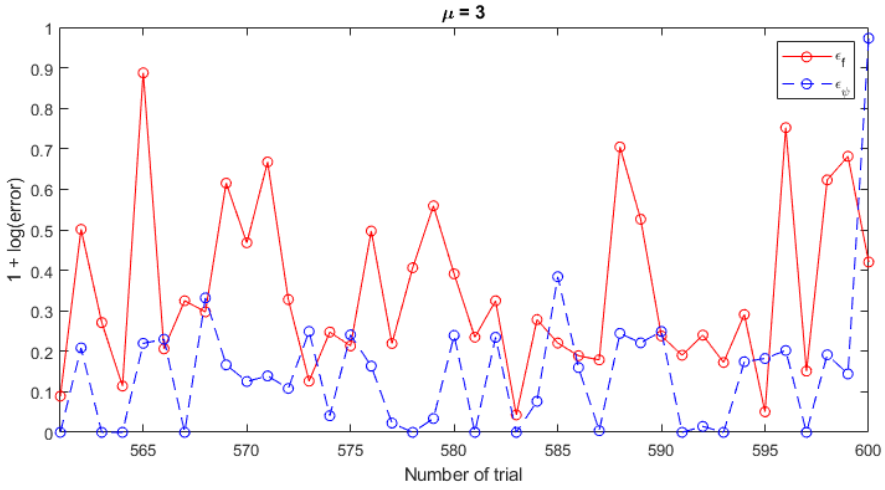

Figure B.2 : The patterns of ε_ψ and ε_f derived from the test trials 560-600 using the LSI algorithm for the case with a noise factor $\mu_j = 3$.

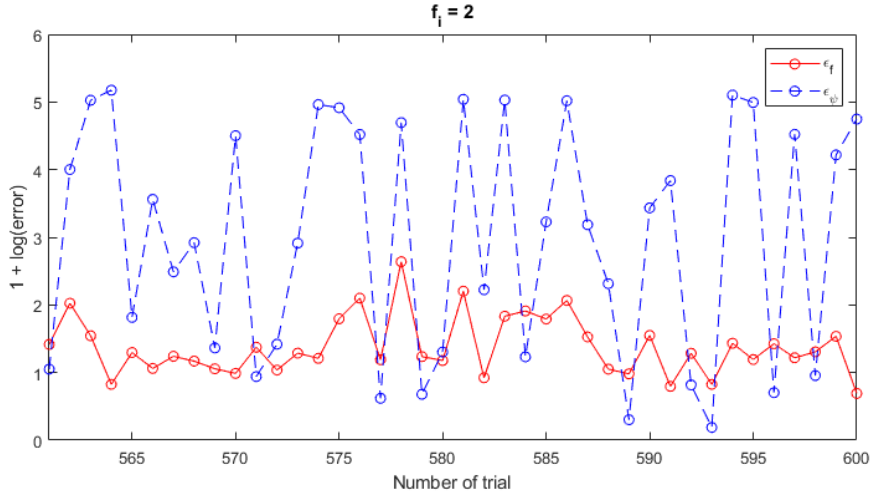

Figure B.3 : The patterns of ε_ψ and ε_f derived from the test trials 560-600 using the LSI algorithm for the case with 2 false stars $f_i = 2$.

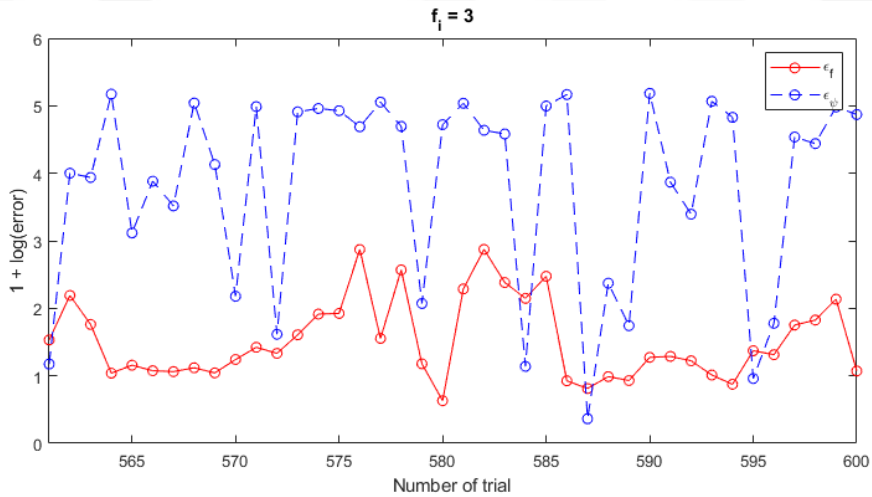

Figure B.4 : The patterns of ε_ψ and ε_f derived from the test trials 560-600 using the LSI algorithm for the case with 3 false stars $f_i = 3$.

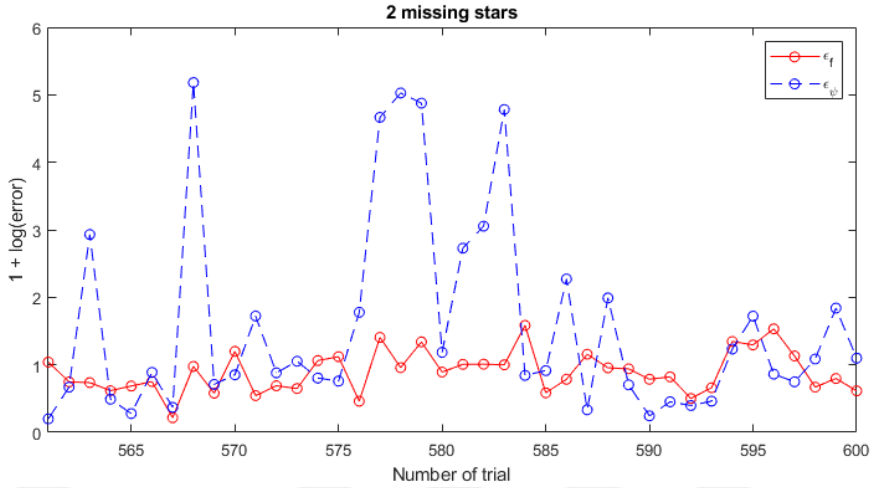

Figure B.5 : The patterns of ε_ψ and ε_f derived from the test trials 560-600 using the LSI algorithm for the case with 2 missing stars.

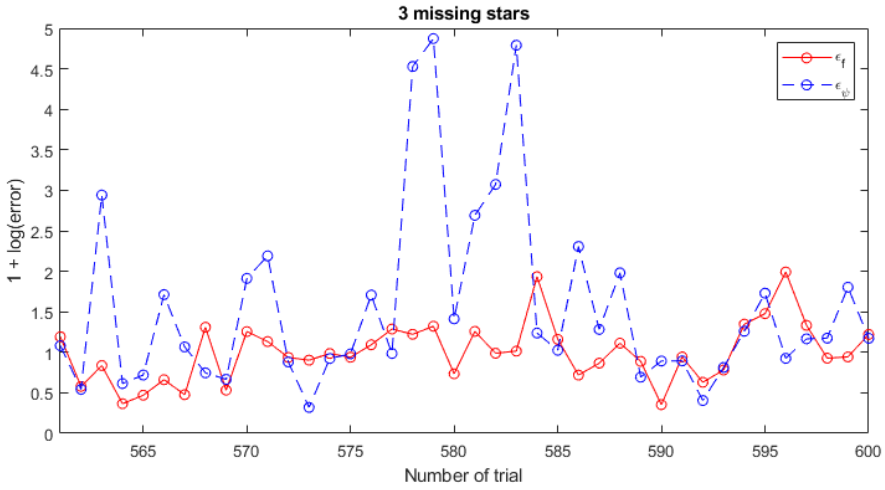

Figure B.6 : The patterns of ε_ψ and ε_f derived from the test trials 560-600 using the LSI algorithm for the case with 3 missing stars.

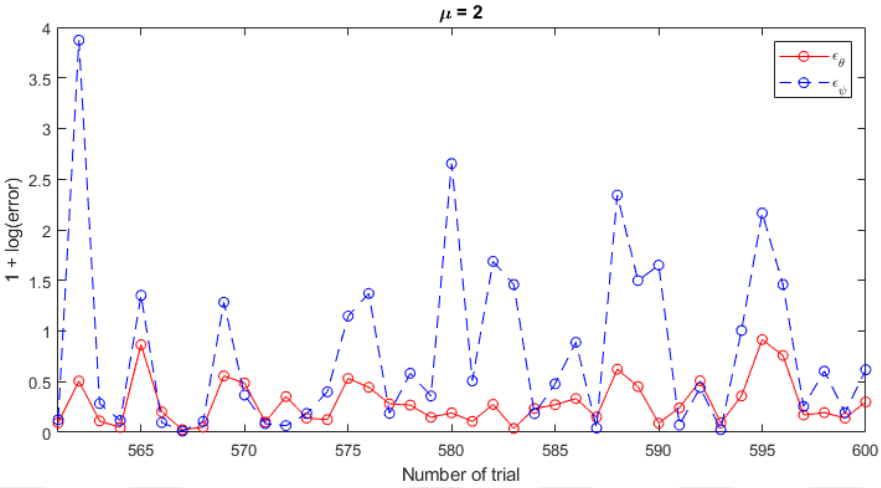

Figure B.7 : The patterns of ε_ψ and ε_θ derived from the test trials 560-600 using the LSI algorithm for the case with a noise factor $\mu_j = 2$.

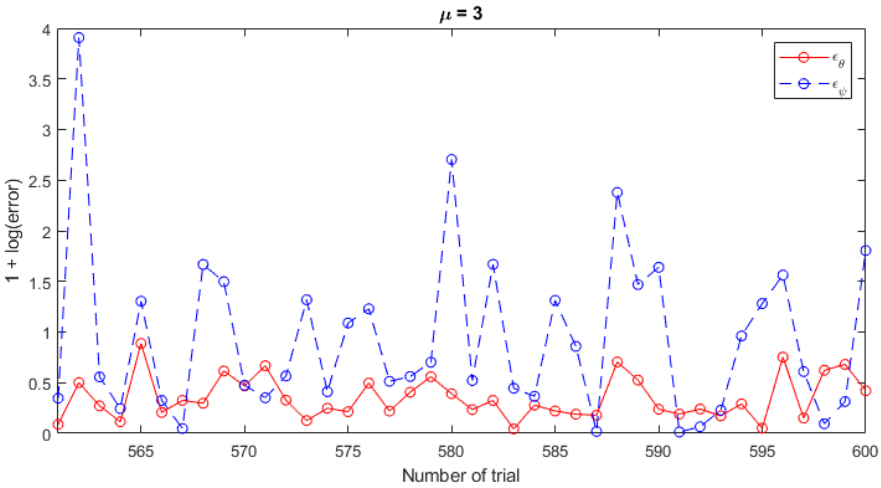

Figure B.8 : The patterns of ε_ψ and ε_θ derived from the test trials 560-600 using the LSI algorithm for the case with a noise factor $\mu_j = 3$.

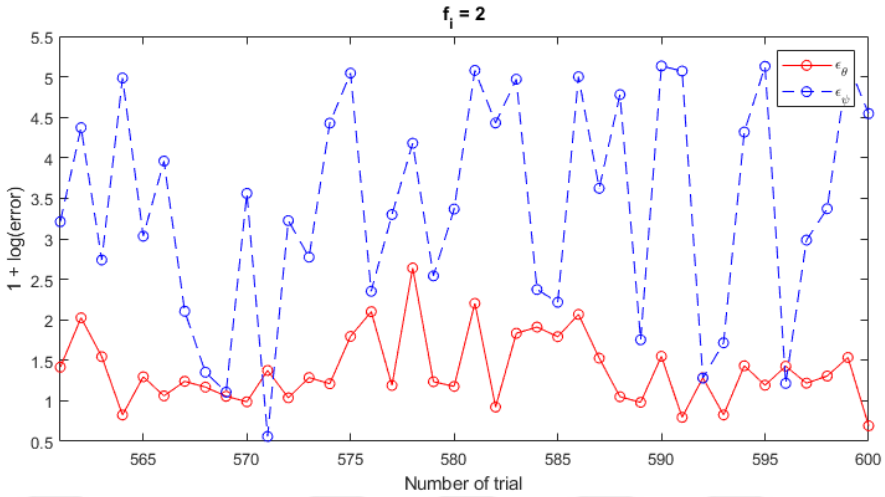

Figure B.9 : The patterns of ε_ψ and ε_θ derived from the test trials 560-600 using the LSI algorithm for the case with 2 false stars $f_i = 2$.

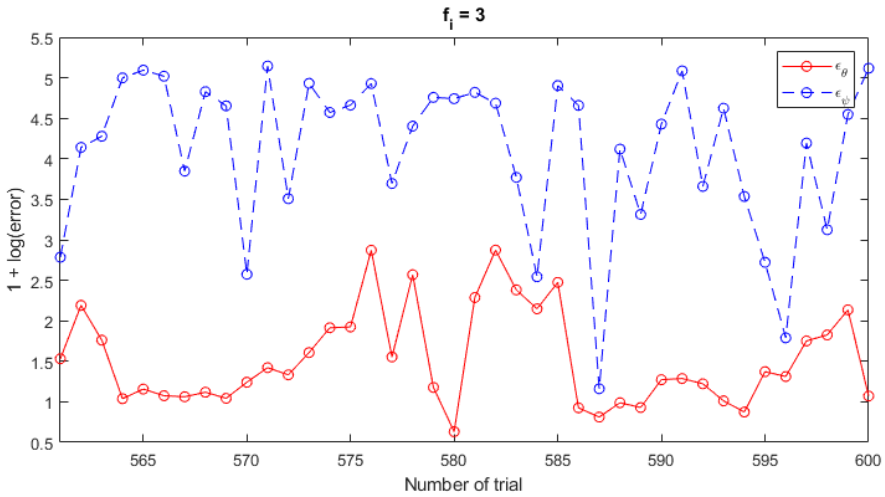

Figure B.10 : The patterns of ε_ψ and ε_θ derived from the test trials 560-600 using the LSI algorithm for the case with 3 false stars $f_i = 3$.

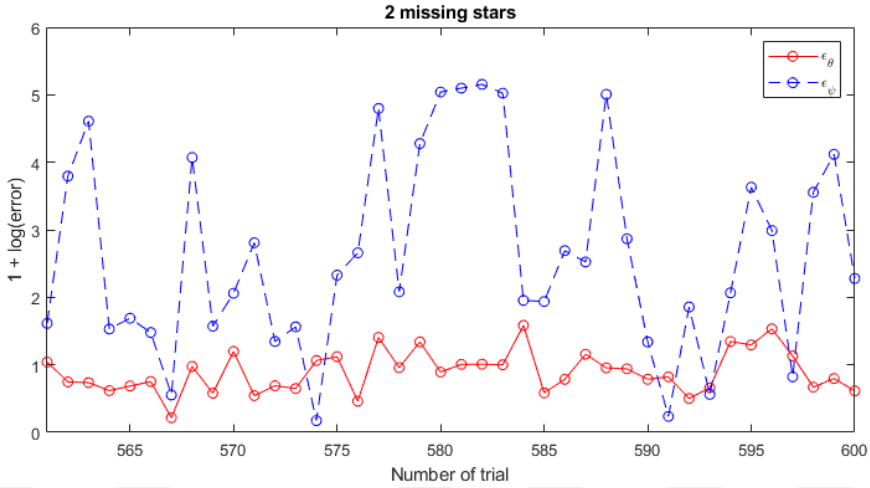

Figure B.11 : The patterns of ε_ψ and ε_θ derived from the test trials 560-600 using the LSI algorithm for the case with 2 missing stars.

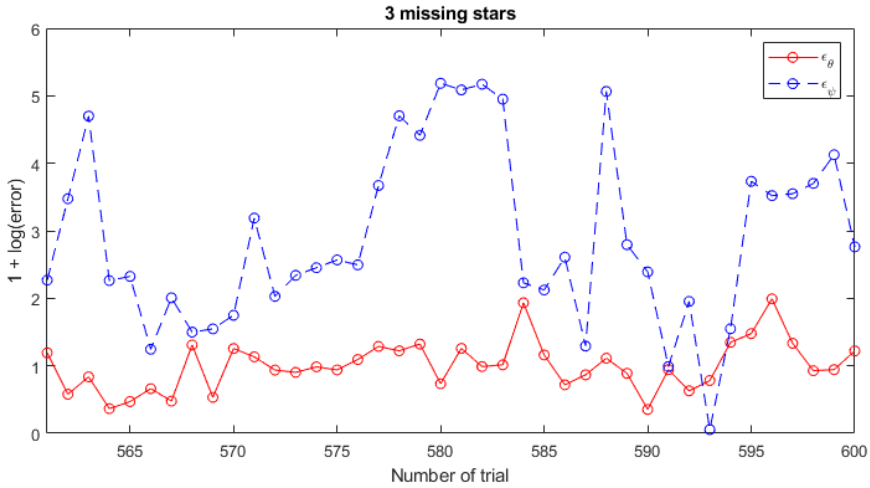

Figure B.12 : The patterns of ε_ψ and ε_θ derived from the test trials 560-600 using the LSI algorithm for the case with 3 missing stars.

APPENDIX C : Investigation of Error Threshold

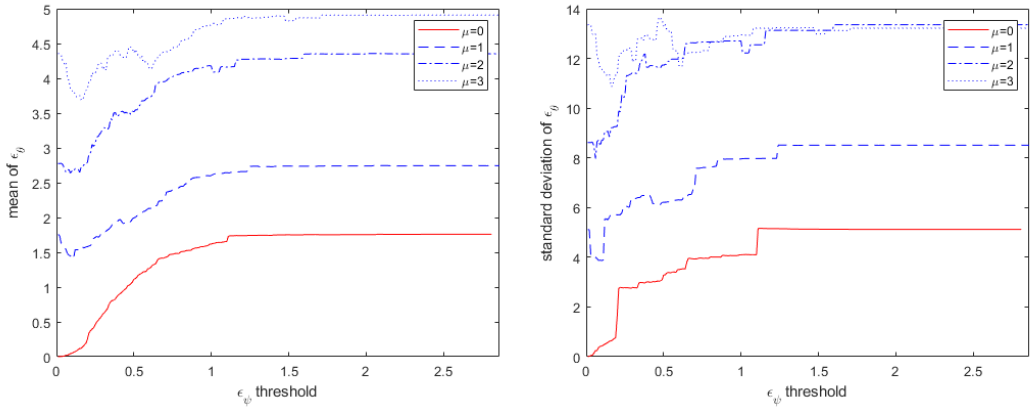

Figure C.1 : The patterns of ε_θ derived from the test results provided by the LSI algorithm with respect to ε_ψ thresholds in the presence of brightness and position noise.

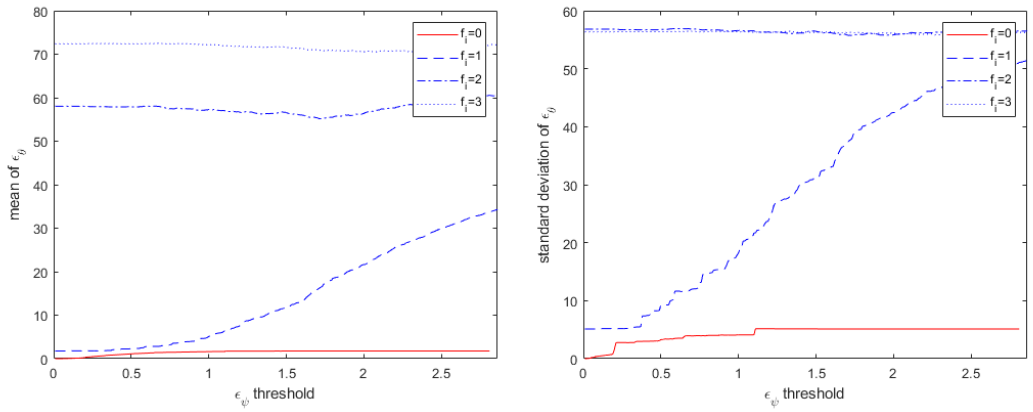

Figure C.2 : The patterns of ε_θ derived from the test results provided by the LSI algorithm with respect to ε_ψ thresholds in the presence of false stars.

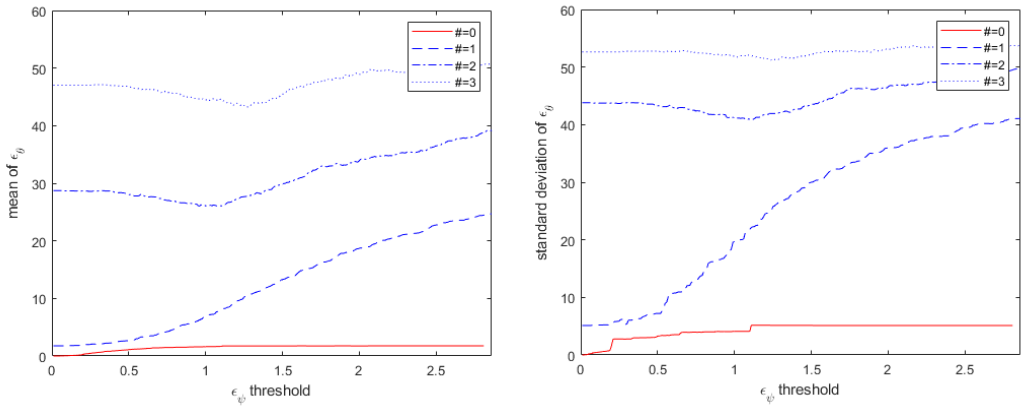

Figure C.3 : The patterns of ε_θ derived from the test results provided by the LSI algorithm with respect to ε_ψ thresholds in the presence of missing stars.

APPENDIX D : Statistical Features of Distance Error

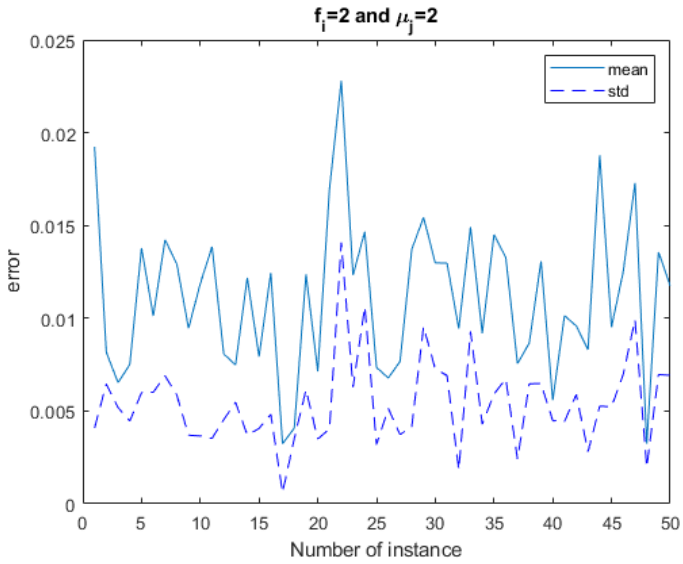

Figure D.1 : The mean of distances between the true stars and the transformed stars and their standard deviations with a noise factor $\mu_j = 2$ in the presence of 2 false stars.

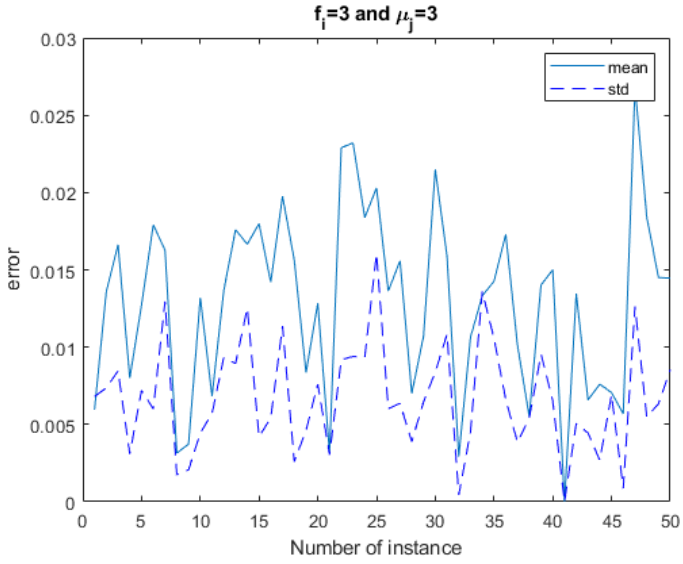

Figure D.2 : The mean of distances between the true stars and the transformed stars and their standard deviations with a noise factor $\mu_j = 3$ in the presence of 3 false stars.

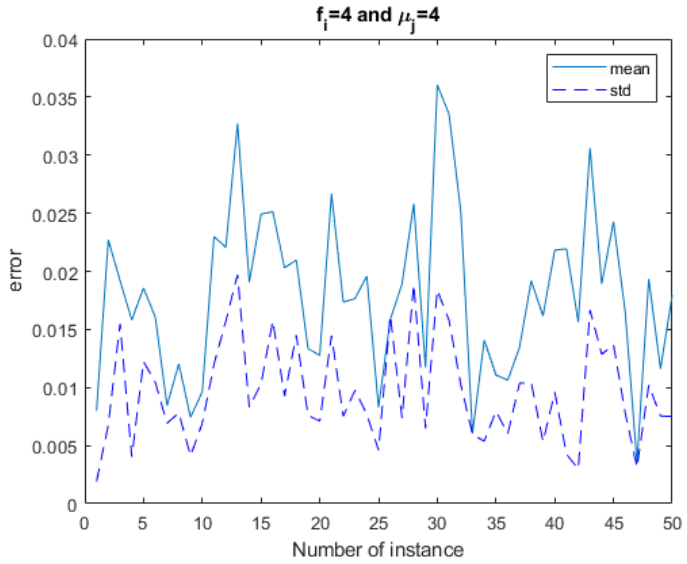

Figure D.3 : The mean of distances between the true stars and the transformed stars and their standard deviations with a noise factor $\mu_j = 4$ in the presence of 4 false stars.

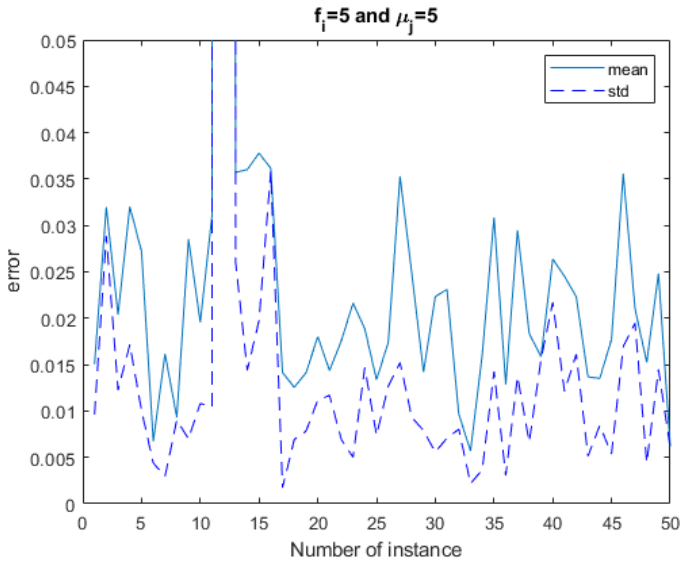

Figure D.4 : The mean of distances between the true stars and the transformed stars and their standard deviations with a noise factor $\mu_j = 5$ in the presence of 5 false stars.

APPENDIX E : Patterns of Errors with Different Thresholds

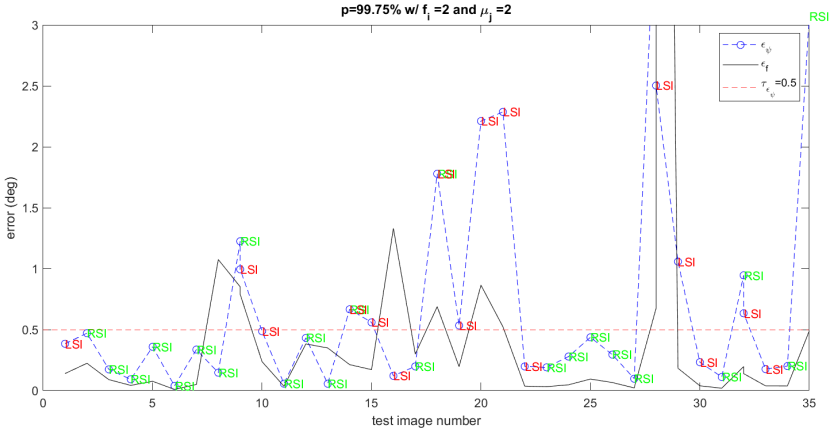

Figure E.1 : A sequence from the experiment carried out with the database with $p = 99.75\%$ with noise injection $\mu_j = 2$ and $f_i = 2$ to observe the stimulation of the update mechanism in the RSI algorithm with $\tau_{\varepsilon_\psi} = 0.5$.

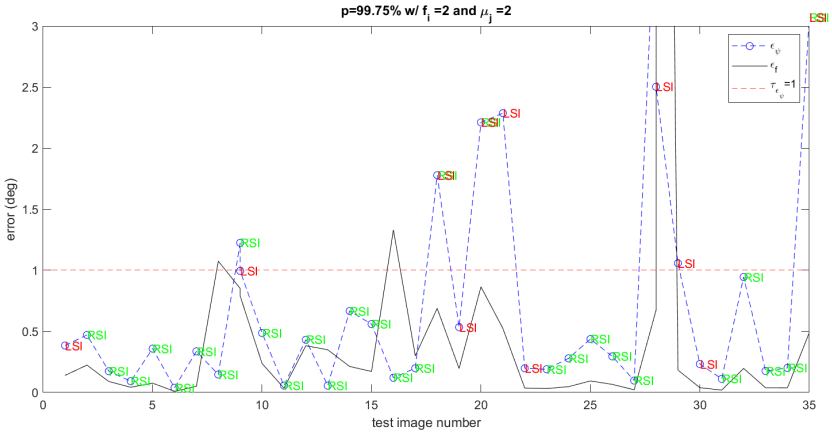

Figure E.2 : A sequence from the experiment carried out with the database with $p = 99.75\%$ with noise injection $\mu_j = 2$ and $f_i = 2$ to observe the stimulation of the update mechanism in the RSI algorithm with $\tau_{\varepsilon_\psi} = 1$.

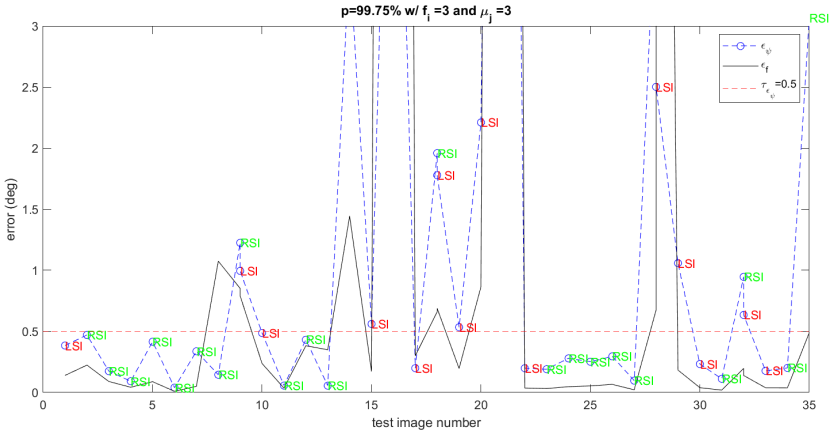

Figure E.3 : A sequence from the experiment carried out with the database with $p = 99.75\%$ with noise injection $\mu_j = 3$ and $f_i = 3$ to observe the stimulation of the update mechanism in the RSI algorithm with $\tau_{\varepsilon_\psi} = 0.5$.

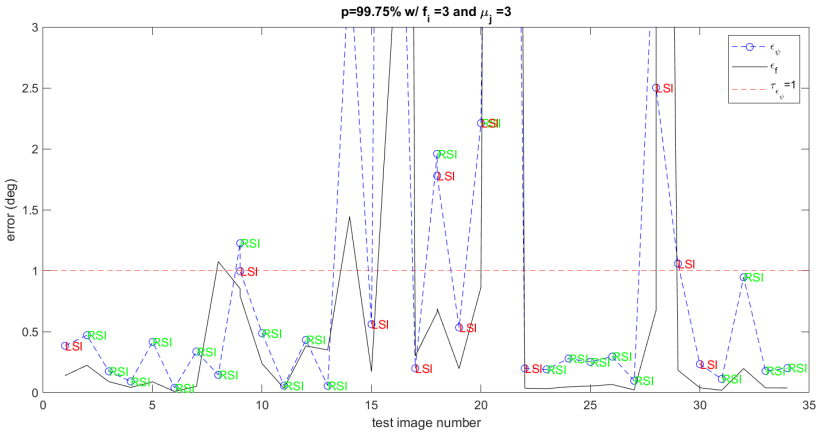

Figure E.4 : A sequence from the experiment carried out with the database with $p = 99.75\%$ with noise injection $\mu_j = 3$ and $f_i = 3$ to observe the stimulation of the update mechanism in the RSI algorithm with $\tau_{\varepsilon_\psi} = 1$.

APPENDIX F : Precision Rate Curves

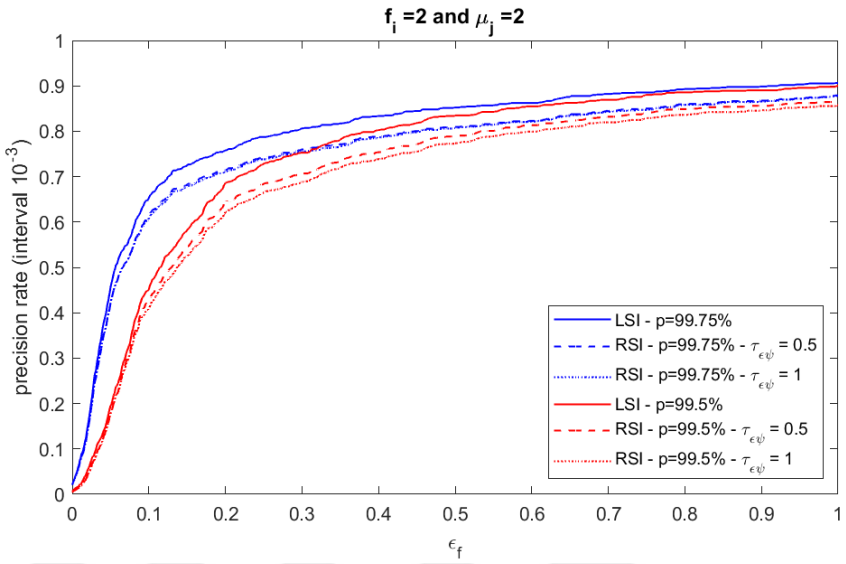

Figure F.1 : Precision rate curves for $\mu_j = 2$ and $f_i = 2$.

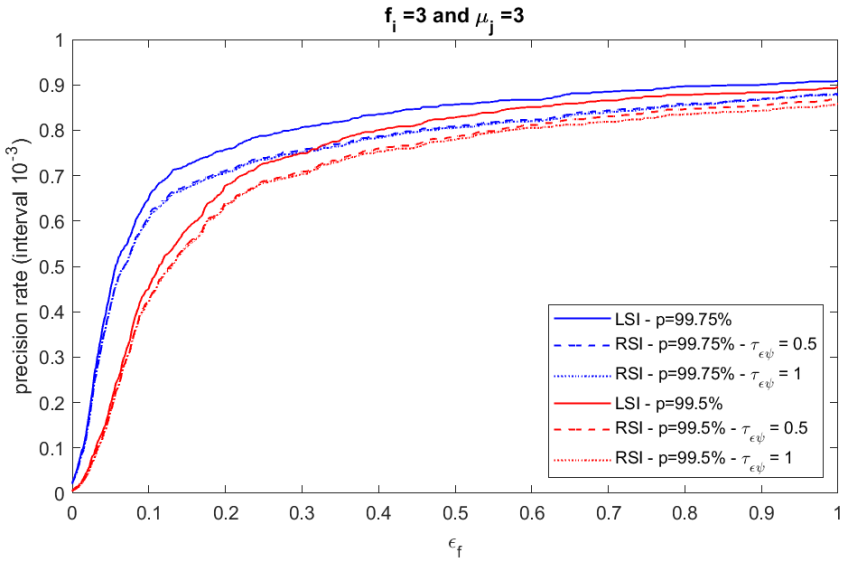

Figure F.2 : Precision rate curves for $\mu_j = 3$ and $f_i = 3$.

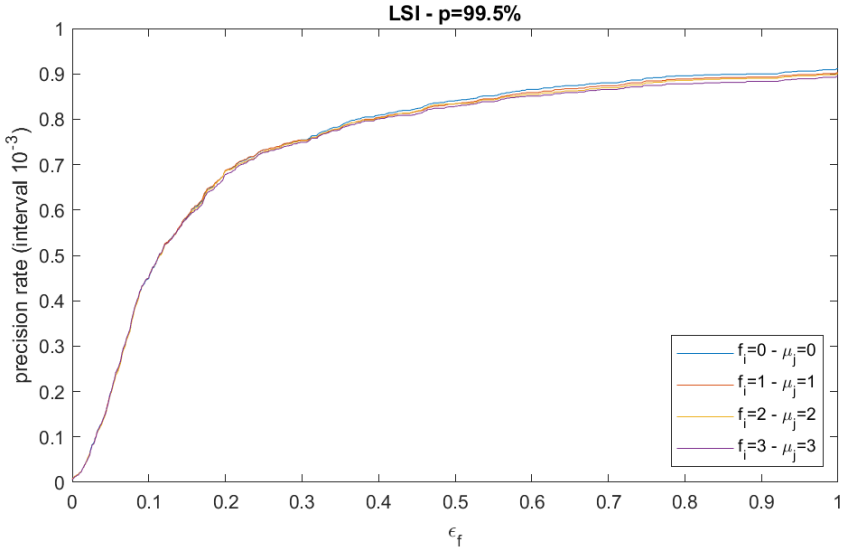

Figure F.3 : The precision rate curves of the FSF-aided LSI algorithm implemented using a database with $p = 99.5\%$.

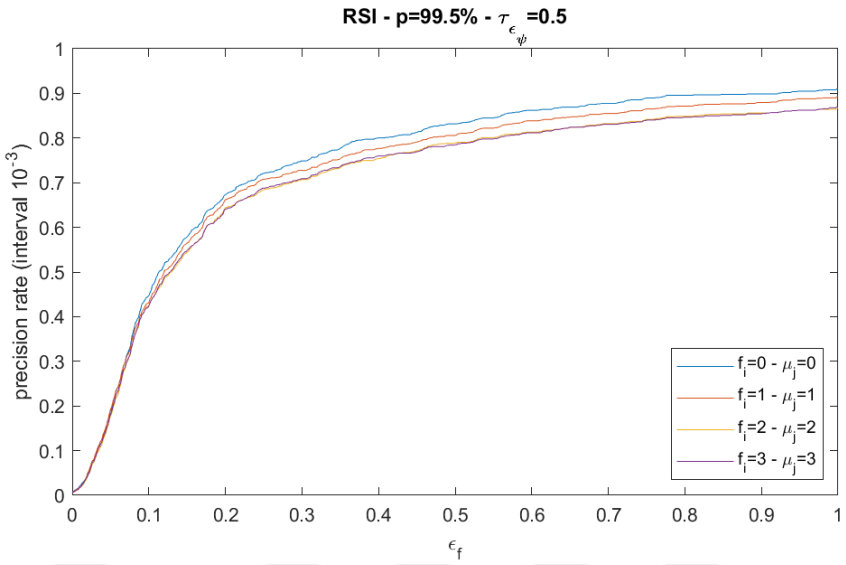

Figure F.4 : The precision rate curves of the FSF&CME-aided RSI algorithm implemented using a database with $p = 99.5\%$ and a threshold $\tau_{\varepsilon_\psi} = 0.5$.

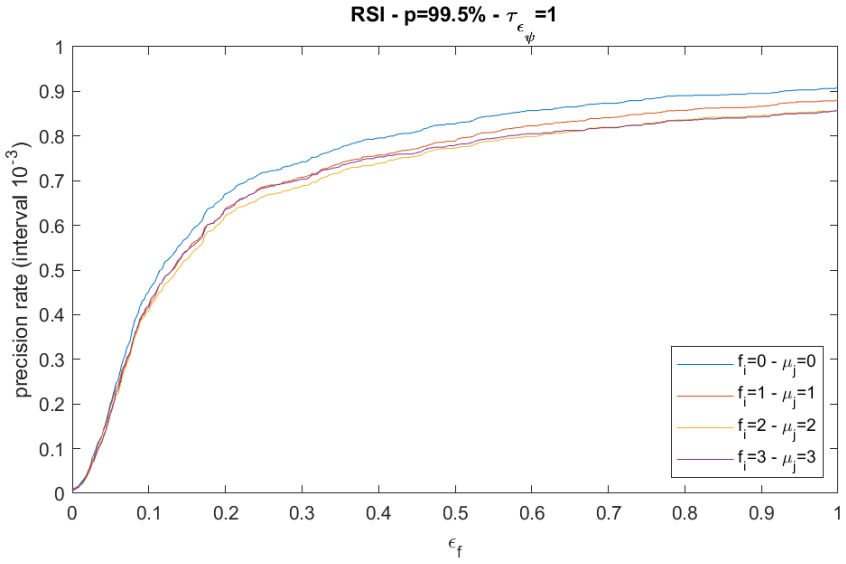

Figure F.5 : The precision rate curves of the FSF&CME-aided RSI algorithm implemented using a database with $p = 99.5\%$ and a threshold $\tau_{\varepsilon_\psi} = 1$.

APPENDIX G : Run Time Patterns

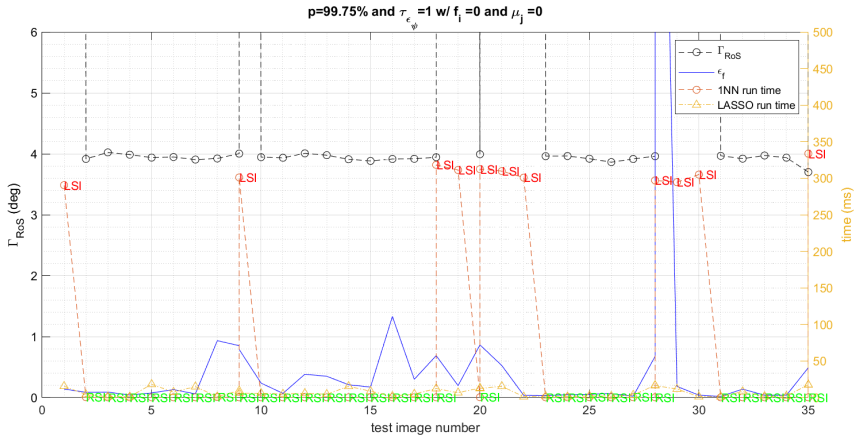

Figure G.1 : A sequence from the experiment carried out with $p = 99.75\%$ without noise injection to observe the effects of the update mechanism on the run time and memory usage in the RSI algorithm with $\tau_{\varepsilon_\psi} = 1$.

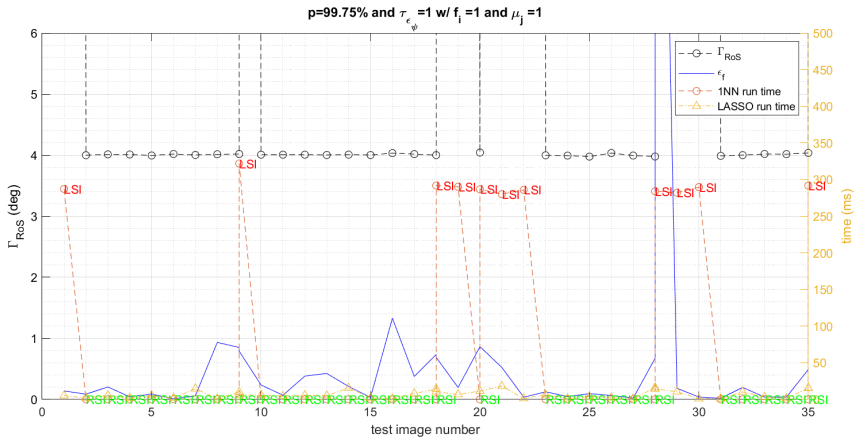

Figure G.2 : A sequence from the experiment carried out with $p = 99.75\%$ with noise injection ($\mu_j = 1$ and $f_i = 1$) to observe the effects of the update mechanism on the run time and memory usage in the RSI algorithm with $\tau_{\varepsilon_\psi} = 1$.

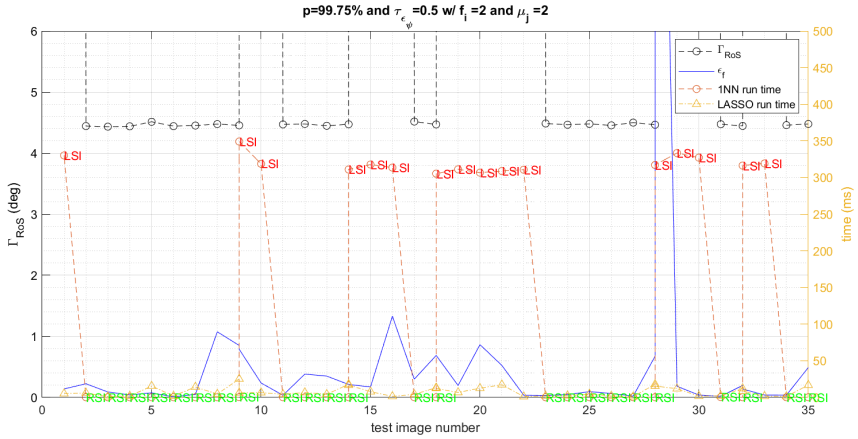

Figure G.3 : A sequence from the experiment carried out with $p = 99.75\%$ with noise injection ($\mu_j = 2$ and $f_i = 2$) to observe the effects of the update mechanism on the run time and memory usage in the RSI algorithm with $\tau_{\varepsilon_\psi} = 0.5$.

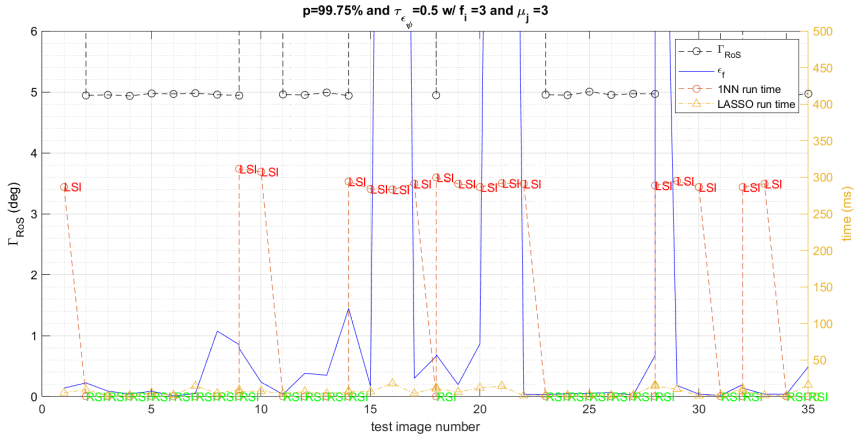

Figure G.4 : A sequence from the experiment carried out with $p = 99.75\%$ with noise injection ($\mu_j = 3$ and $f_i = 3$) to observe the effects of the update mechanism on the run time and memory usage in the RSI algorithm with $\tau_{\varepsilon_\psi} = 0.5$.

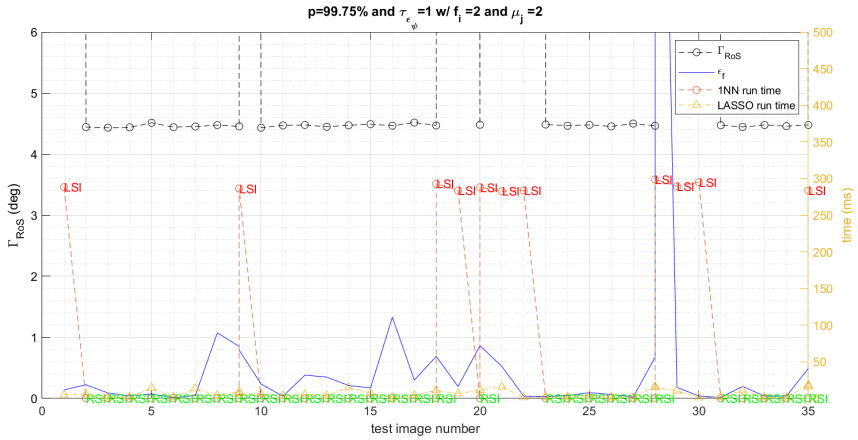

Figure G.5 : A sequence from the experiment carried out with $p = 99.75\%$ with noise injection ($\mu_j = 2$ and $f_i = 2$) to observe the effects of the update mechanism on the run time and memory usage in the RSI algorithm with $\tau_{\varepsilon_\psi} = 1$.

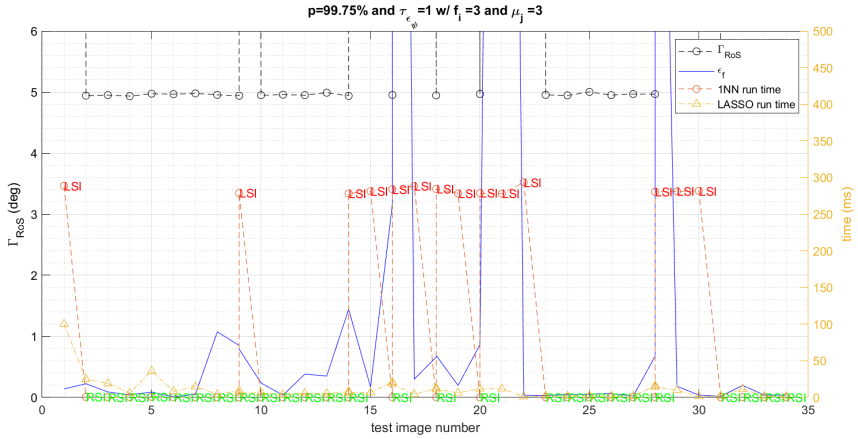

Figure G.6 : A sequence from the experiment carried out with $p = 99.75\%$ with noise injection ($\mu_j = 3$ and $f_i = 3$) to observe the effects of the update mechanism on the run time and memory usage in the RSI algorithm with $\tau_{\varepsilon_\psi} = 1$.

APPENDIX H : Error Tables

Table H.1 : Error means and error standard deviations yielded by the algorithm for experiments with varying parameters and noise conditions using the database with $p = 99.5\%$.

Method	Simulation			Error in degrees					
	μ_j	f_i	τ_{ε_ψ}	ε_f		ε_θ		ε_ψ	
				μ	σ	μ	σ	μ	σ
FSF-aided LSI	0	0	-	0.18	0.18	2.98	2.63	0.49	0.54
FSF&CME-aided RSI	0	0	0.5	0.18	0.18	2.98	2.64	0.48	0.52
FSF&CME-aided RSI	0	0	1	0.18	0.19	3.00	2.65	0.49	0.53
FSF-aided LSI	1	1	-	0.18	0.18	3.00	2.65	0.48	0.54
FSF&CME-aided RSI	1	1	0.5	0.18	0.19	3.01	2.67	0.49	0.54
FSF&CME-aided RSI	1	1	1	0.19	0.19	3.04	2.70	0.50	0.54
FSF-aided LSI	2	2	-	0.18	0.18	3.03	2.73	0.50	0.56
FSF&CME-aided RSI	2	2	0.5	0.18	0.19	3.05	2.76	0.51	0.57
FSF&CME-aided RSI	2	2	1	0.19	0.19	3.08	2.78	0.52	0.58
FSF-aided LSI	3	3	-	0.18	0.18	3.02	2.64	0.49	0.54
FSF&CME-aided RSI	3	3	0.5	0.18	0.19	3.01	2.63	0.50	0.55
FSF&CME-aided RSI	3	3	1	0.18	0.18	2.99	2.63	0.49	0.54

Table H.2 : Error means and error standard deviations yielded by the algorithm for experiments with varying parameters and noise conditions using the database with $p = 99.75\%$.

Method	Simulation			Error in degrees					
	μ_j	f_i	τ_{ε_ψ}	ε_f		ε_θ		ε_ψ	
				μ	σ	μ	σ	μ	σ
FSF-aided LSI	0	0	-	0.13	0.18	2.84	2.63	0.49	0.54
FSF&CME-aided RSI	0	0	0.5	0.14	0.18	2.86	2.62	0.37	0.53
FSF&CME-aided RSI	0	0	1	0.13	0.18	2.85	2.57	0.36	0.53
FSF-aided LSI	1	1	-	0.13	0.18	2.85	2.65	0.48	0.54
FSF&CME-aided RSI	1	1	0.5	0.14	0.19	2.88	2.57	0.38	0.56
FSF&CME-aided RSI	1	1	1	0.14	0.18	2.90	2.55	0.38	0.55
FSF-aided LSI	2	2	-	0.13	0.18	2.88	2.73	0.50	0.56
FSF&CME-aided RSI	2	2	0.5	0.13	0.18	2.91	2.68	0.37	0.54
FSF&CME-aided RSI	2	2	1	0.13	0.18	2.90	2.61	0.37	0.54
FSF-aided LSI	3	3	-	0.13	0.13	2.90	2.64	0.49	0.54
FSF&CME-aided RSI	3	3	0.5	0.13	0.18	2.92	2.65	0.36	0.52
FSF&CME-aided RSI	3	3	1	0.13	0.18	2.93	2.54	0.36	0.52

CURRICULUM VITAE

Name Surname: Erdem Onur ÖZYURT

EDUCATION:

- **B.Sc.:** 2013, Istanbul Technical University, Faculty of Electrical and Electronics Engineering, Department of Electronics and Communication Engineering
- **M.Sc.:** 2018, Istanbul Technical University, Graduate School of Science and Technology, Division of Electronics and Communication Engineering, Telecommunication Engineering Graduate Program

PROFESSIONAL EXPERIENCE AND REWARDS:

- 2014 Assistant Specialist at Perform Asset Management, Inc.
- 2015 Assistant Specialist at Turkish Atomic Energy Authority
- 2015 Engineer at Gitek Engineering and R&D, Ltd. Company
- 2016-2024 Research Assistant at Turkish-German University
- 2024 Engineer at İTÜNOVA Technology, Inc.

PUBLICATIONS, PRESENTATIONS AND PATENTS ON THE THESIS:

- **Ozyurt E. O.**, Aslan R. (2022). A morphological approach of false star removal and camera motion estimation for star sensors. *11th Nano-Satellite Symposium*, October 17-19, 2022, İstanbul, Turkiye. (Presentation Instance)
- **Ozyurt E. O.**, Aslan, R. (2024). A lost-in-space star identification algorithm based on regularized pattern recognition, *Acta Astronautica,* 219, 149-163. (Article Instance)

OTHER PUBLICATIONS, PRESENTATIONS AND PATENTS:

- **Ozyurt E. O.**, Gunsel B. (2018). Wami Object Tracking Using L1 Tracker Integrated with a Deep Detector. *25th IEEE International Conference on Image Processing (ICIP)*, October 7-10, 2018, Athens, Greece, 2690-2694. (Presentation Instance)

- **Ozyurt E. O.**, Kaya A. C., İpekoğlu M. (2020). Prediction of Mechanical Properties of Sintered 316L Open Cell Foam Struts by Image Processing. *9th International Scientific Research Congress*, December 12-13, 2020, Ankara, Turkiye, 117-118. (Presentation Instance)

- Kaya A. C., İpekoğlu M., **Ozyurt E. O.** (2021). Measurement of Area Moment of Inertia from Micrographs by Image Processing. *Global Conference on Engineering Research*, May 2-5, 2020, Bandırma, Turkiye, 537-547. (Presentation Instance)

- Erdoğan A. B., **Ozyurt E. O.**, İpekoğlu M., Gökçen M. G. (2023). Investigation of the Relationship Between Cutting Parameters and Chip Morphology: an Artificial Neural Network Approach. *6th INTERNATIONAL PALANDOKEN SCIENTIFIC STUDIES CONGRESS*, June 24-25, 2023, Palandöken, Turkiye, 229-237. (Presentation Instance)

- **Ozyurt E. O.** (2023). Material Classification by Investigation of 3D Morphological Chip Attributes using Neural Networks. *Leverage of IT in Engineering and Science*, October 30-31, 2023, İstanbul, Turkiye, 26-27. (Presentation Instance)

- **Ozyurt E. O.** (2023). False Star Filtering and Camera Motion Estimation using an Unsupervised Learning Method with Morphological Attributes. *Leverage of IT in Engineering and Science*, October 30-31, 2023, İstanbul, Turkiye, 24-25. (Presentation Instance)

- **Ozyurt E. O.**, Duman A. C. (2023). Adaptive Heat Gain Estimation for Window Filming with Building Models and Estimation of Reduction in Carbon Emission for Turkish Cities. *Leverage of IT in Engineering and Science*, October 30-31, 2023, İstanbul, Turkiye, 22-23. (Presentation Instance)