AFRL-AFOSR-UK-TR-2020-0042

Optimized tracking of low thrust orbit raising maneuvers

Marco Martorella
CONSORZIO NAZIONALE INTERUNIVERSITARIO PER LE TELECOMUNICAZIONI
VIALE G.P.USBERTI 181/A
PARMA, PARMA, 43124
IT

Final Report

DISTRIBUTION A: Distribution approved for public release.

Air Force Research Laboratory
Air Force Office of Scientific Research
European Office of Aerospace Research and Development
Unit 4515 Box 14, APO AE 09421

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*

2. REPORT TYPE
Final

3. DATES COVERED *(From - To)*
01 Sep 2018 to 31 Aug 2019

4. TITLE AND SUBTITLE
Optimized tracking of low thrust orbit raising maneuvers

5a. CONTRACT NUMBER

5b. GRANT NUMBER
FA9550-18-1-7006

5c. PROGRAM ELEMENT NUMBER
61102F

6. AUTHOR(S)
Marco Martorella

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
CONSORZIO NAZIONALE INTERUNIVERSITARIO PER LE TELECOMUNICAZIONI
VIALE G.P.USBERTI 181/A
PARMA, PARMA, 43124 IT

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
EOARD
Unit 4515
APO AE 09421-4515

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/AFOSR IOE

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
AFRL-AFOSR-UK-TR-2020-0042

12. DISTRIBUTION/AVAILABILITY STATEMENT
A DISTRIBUTION UNLIMITED: PB Public Release

13. SUPPLEMENTARY NOTES

14. ABSTRACT
Orbit determination of Resident Space Objects (RSOs) is a fundamental step to ensure effective Space Situational Awareness (SSA). A particular scenario of interest concerns the transfer of a satellite from a Low Earth Orbit (LEO) to a Geostationary Earth Orbit (GEO). In this project we have developed an Geo Transfer Orbit (GTO) tracker that optimizes the use of sensors on the ground by minimizing their number and therefore the tracking operation cost.

15. SUBJECT TERMS
Astrodynamics, Radar, Satellite Tracking

16. SECURITY CLASSIFICATION OF: a. REPORT	b. ABSTRACT	c. THIS PAGE	17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
Unclassified	Unclassified	Unclassified	SAR		FLAKE, BARRETT
					19b. TELEPHONE NUMBER *(Include area code)* 314-235-6115

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

GRANT NUMBER: FA9550-18-1-7006

TECHNICAL REPORT
(FINAL REPORT)
01 SEP 2018 - 31 AUG 2019

Optimized Tracking of Low Thrust Orbit Raising Maneuvers

Laboratory of Radar and Surveillance Systems

(RaSS) - CNIT, Pisa, Italy
November 2019

Authors: L. Gentile, M. Martorella, W. Faber, W. Zaidi, T. Kelecy

PI: Prof. M. Martorella

Contents

1 Introduction

Orbit determination of Resident Space Objects (RSOs) is a fundamental step to ensure effective Space Situational Awareness (SSA). A particular scenario of interest concerns the transfer of a satellite from a Low Earth Orbit (LEO) to a Geostationary Earth Orbit (GEO). In this scenario, a satellite is not only subject to a number of known physical forces but also to a thrust, which, for cooperative targets, is known a priori with a certain accuracy. In this project we have only considered cooperative GEO Transfer Orbit (GTO) satellites as the time and resources planned for this project would only allow for an initial study to be conducted. Nevertheless, in spite of the limitations inherent in this scenario, important elements have been developed that will provide the basis for building algorithms that would be applicable to non-cooperative GTO satellite tracking. Such scenarios would include the case of tracking a third party/adversary satellite during its GTO. In this project, we have considered satellites equipped with an electric propulsion thruster that maneuvers along a GTO. More specifically, a 100% efficency ion propulsion thruster is considered, so that no propellent and no chemical thruster is needed. In this conditions, the mass decrease due to the consumption of the propellent can be ignored.

Because of the low thrust (on the order of a fraction of Newton), the entire transfer from LEO to GEO usually takes more time with respect to traditional thrusters. Therefore, a close monitoring is needed in order to maintain custody of the object. Several geographically distributed sensors that are capable of detecting and tracking the orbiting object are needed to accomplish this task. The types of sensors that can be used to detect and track RSOs can be diverse and each type may provide measurements that are affected by errors that vary from type to type. For example, radar is typically used where the range is low (less than 1000 km) and Electro-Optical (EO) may be used when the range is large (more than 20,000 km). Moreover, some sensor performance may depend on external conditions, such as weather and lighting conditions (e.g. optical sensors). Therefore, some sensors may be preferred to others as they provide better and more reliable measurements. On the other hand, sensor's time has a cost and also this may vary from type to type and, typically, high-performing sensors have a higher cost. In this project, the complete problem set is considered and modelled. An optimal sensor scheduling is also derived in order to provide the sufficient conditions to periodically track the GTO from the initial to the final phase and at the same time minimise the cost of the entire operation. Results will be shown according to a typical GTO and a discussion will follow in terms of optimal tasking performance vs cost. This technical report concludes with an outlook in terms of research that may be conducted in order to define and develop optimal algorithms for tracking GTOs when thrusting manoeuvres are not known a priori (third party launches).

2 Methods, assumptions and procedures

In this section, we will describe the proposed approach, the physical constraints, the technological assumptions and the simulation procedure, in order to carry out a performance evaluation.

2.1 Proposed method

As previously discussed, this technical report deals with the problem of monitoring an object moving in a GTO from LEO to GEO when equipped with a low-thrust system. Since the complete transfer typically takes several months, a continuous tracking system must be set in place. In an operative scenario, more than one sensor is needed to observe GTO objects. Different types of sensors are considered, such as radar, TT&C and optical systems. Therefore, different types of measurements and different observability conditions for each sensor must be considered.

The processing scheme is depicted in Fig. 1

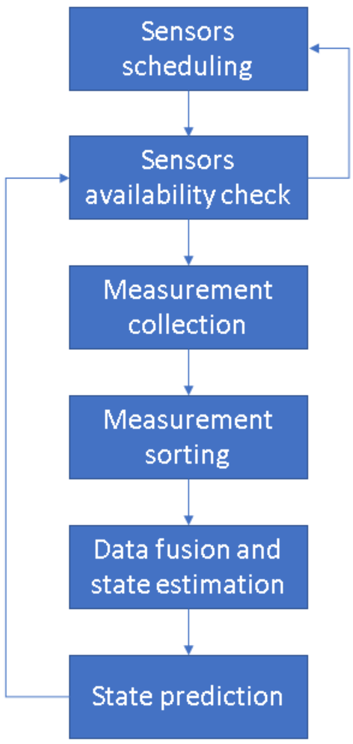

Figure 1: Processing diagram

In the proposed approach, sensor scheduling can be performed before the mission actually starts, taking into account all the constraints for each sensor. This scheduling will not be considered fixed but will change as the availability of each sensor changes and the GTO transitions to GEO.

Once the sensors are assigned, an availability check for each sensor is constantly performed, in order to update the scheduling as the conditions change and reschedule the sensors that will be employed in the future. Each selected sensor measurement is then sent to a central processing unit (Mission Control Unit). The collected measurements are first time-ordered and then a data fusion algorithm is run to estimate the state vector.

2.1.1 Sensor scheduling

Several sensor scheduling algorithms can be found in the literature [1], [5], [2]. Most of the proposed algorithm use the Fisher information as a cost function and selects a subset of sensor with the goal of minimising the overall cost. The main problem of such an approach regards the computational complexity. The estimation algorithm must be performed twice: the first time considering a prediction of the measurements for each sensor to evaluate the Fisher information and then a second time, after the scheduling, to actually perform the state vector estimation using the collected measurements. This approach could be useful for short missions, batch algorithms and for a low number of sensors, but in the case of a long mission and for a large number of sensor the algorithm may be computationally expensive.

The proposed approach will consider a cost function defined with a priori information, such as the model and the sensor accuracy, so that the estimation algorithm is performed only during the mission.

The cost function is described by eq. 1

$$C(k) = \frac{HC(k)[(\Delta T_{k-1} + \Delta T_{k+1})\sigma_p + \sigma_m(k)] \cdot SC(k)}{D(k-1,k)} \tag{1}$$

where σ_p is the propagation standard deviation, which accounts for the physical model accuracy; σ_m is the measurement standard deviation, which accounts for the sensor measurement accuracy ΔT_{k-1} is the time between the last observation of the $k-1^{th}$ sensor and the first observation of the k^{th} sensor, ΔT_{k+1} is the time between the last observation of the k^{th} sensor and the first observation of the $k+1^{th}$ sensor; $SC(k)$ is a parameter that takes into account the cost of the sensor usage, as defined in Tab. 1, and $D(k-1,k)$ takes into account the diversity of the measurement between $k-1^{th}$ sensor and k^{th} sensor. Particularly, $D(k-1,k) = 1$ if the type of measurement of the $k-1^{th}$ sensor is the same of k^{th} sensor and $D(k-1,k) = 2$ if the type of measurement of $k-1^{th}$ sensor is not the same of k^{th} sensor; finally, $HC(k)$ is the hard check for the k^{th} sensor, defined as:

$$HC(k) = Optical\ visibility \cdot Sensor\ availability \cdot good\ weater\ conditions \tag{2}$$

where the three elements of the product in eq. 4 are binary flags. It should be noted that all three flags must be set to ”1” in order for the sensor to be used.

Sensor Cost	**Value**
Very low	1/4
Low	1/2
Medium	1
High	2
Very High	4

Table 1: Sensor cost definition

Once defined the single sensor cost, considering $s_i \in S$ as the i^{th} subset of sensors, where S is the set of all sensor partitions, the overall sensor cost can be defined as:

$$C(s_i) = \sum_{k-1,k,k+1 \in s_i} \sum_{t \in T_{obs}} C(k,t) \tag{3}$$

where $C(k,t)$ is the cost of k^{th} sensor evaluated at time t, at which correspond a state vector of the satellite along the trajectory.

The optimal subset of sensor can be evaluated minimizing the overall function as:

$$s_{opt} = \underset{s_i \in S}{arg\ min}\ C(s_i) \tag{4}$$

Once the cost function is evaluated for all the subsets of sensors, in case of change in the availability constraint for one or more sensor a new evaluation of the cost function is not required, but the sub-optimal set of sensor can be selected taking into account the change. For example, if sensor s^* is no more available, the new subset of sensors will be the set with the minimum cost in which sensor s^* is not present.

Let us consider an example in which there are three sensors available: s_1, s_2 and s_3. All the possible configurations of such sensors are listed in Tab. 2.

s1	s2	s3
1	1	1
1	1	0
1	0	1
1	0	0
0	1	1
0	1	0
0	0	1

Table 2: Example: Configurations list

Let us assume that the evaluated costs are listed in Tab. 3. The optimal configuration will be the configuration made by s_1 and s_2.

s1	s2	s3	Cost
1	1	1	3.2726
1	1	0	0.1697
1	0	1	3.2702
1	0	0	2.7279
0	1	1	0.2858
0	1	0	0.3869
0	0	1	2.0866

Table 3: Example: Evaluated costs

Let us consider now the hypothesis in which s_2 is no longer available. Since cost evaluation for all possible configurations has already been performed, it will be sufficient select the configuration with the lowest cost in which s_2 is not considered available. Tab. 4 lists such configurations. In this case the minimum cost configuration consists in s_3 sensor only.

s1	s2	s3	Cost
1	1	1	3.2726
1	1	0	0.1697
1	0	1	3.2702
1	0	0	2.7279
0	1	1	0.2858
0	1	0	0.3869
0	0	1	2.0866

Table 4: Example: Sub-optimal configuration

Once s_2 is again available the optimal configuration will be re-assessed.

Look-up tables such as 3 can be used for any sensor number, as shown in Tab. 5, in order to storage the cost evaluation and allow to adapt dynamically the sensor network in case of changes in sensors state.

s1	s2	s3	...	sn	Cost
1	1	1	...	1	C_1
1	1	0	...	0	C_2
1	0	1	...	1	C_3
...	...	...	...	...	...

Table 5: Configurations look-up table

2.1.2 Data fusion

In order to perform data fusion from such different types of sensors the Unscented Kalman Filter, presented in detail in [4] [3], will be considered.

This type of non-linear filter, represented in Fig. 2, uses the Unscented Transform to estimate the propagated mean value x_{k+1} and the covariance matrix $\mathbf{C}_{k+1}$ of the state vector (usually assumed to be a normal random variable) by using the state vector x_k and Covariance matrix $\mathbf{C}_{k+1}$ previously estimated and measurements y_{k+1}.

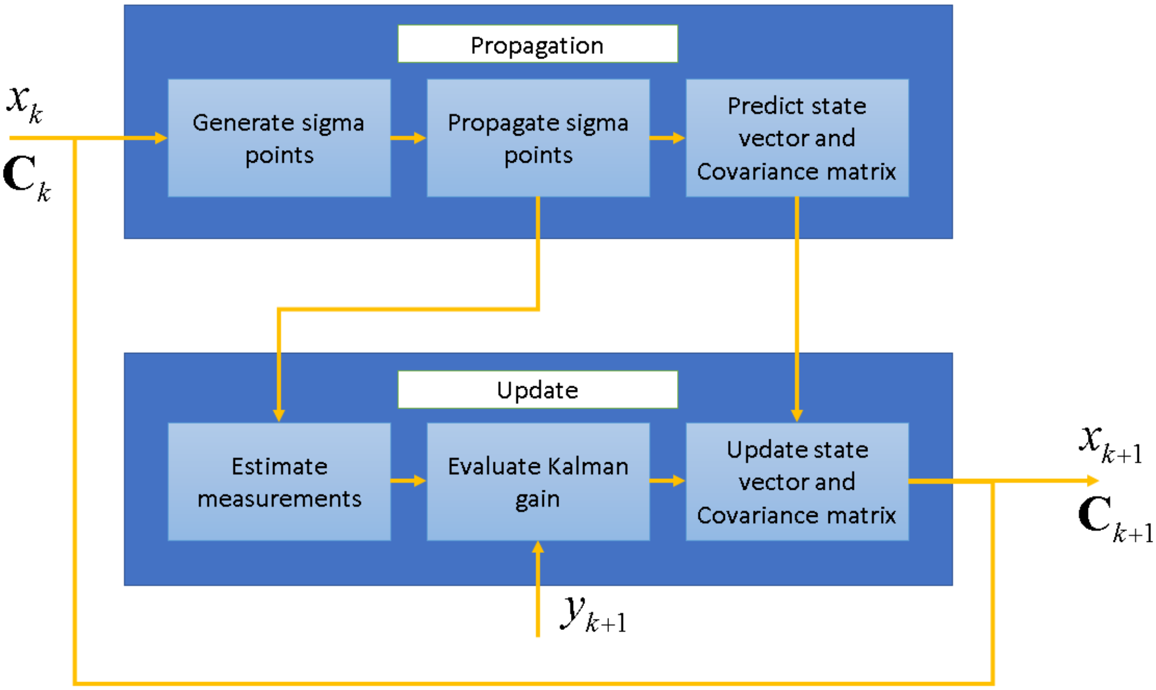

Figure 2: Unscented Kalman Filter processing scheme

Table 6: UKF algorithm summary

Prediction
$\mathbf{S}_{k-1} = Cholesky(\hat{\mathbf{C}}_{k-1})$ $\mathbf{X}_{i,k-1} = \hat{\mathbf{x}}_{k-1} \pm \sqrt{n_x}\mathbf{s}_{i,k-1}$ where $\mathbf{S} = [\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_{n_x}]$ $w = \frac{1}{2n_x}$ $\mathbf{X}_{i,k} \Leftarrow \dot{\mathbf{X}}_i = f(\hat{\mathbf{X}}_{i,k-1}, t)$ $\hat{\mathbf{x}}_k = \sum_{i=1}^{2n_x} w_i \mathbf{X}_{i,k}$ $\mathbf{C}_k = \sum_{i=1}^{2n_x} w_i (\mathbf{X}_{i,k} - \hat{\mathbf{x}}_k)(\mathbf{X}_{i,k} - \hat{\mathbf{x}}_k)^T$
Update
$\mathbf{Y}_i = h(\mathbf{X}_i, t)$ $\hat{\mathbf{y}} = \sum_{i=1}^{2n_x} w_i \mathbf{Y}_i$ $\mathbf{C}_{yy} = \sum_{i=1}^{2n_x} w_i (\mathbf{Y}_i - \hat{\mathbf{y}})(\mathbf{Y}_i - \hat{\mathbf{y}})^T + \mathbf{R}$ $\mathbf{C}_{xy} = \sum_{i=1}^{2n_x} w_i (\mathbf{X}_i - \hat{\mathbf{x}})(\mathbf{Y}_i - \hat{\mathbf{y}})^T$ $\mathbf{K} = \mathbf{C}_{xy}\mathbf{C}_{yy}^{-1}$ $\hat{\mathbf{X}}_{i,k} = \hat{\mathbf{x}} + \mathbf{K}(\mathbf{y} - \hat{\mathbf{y}})$ $\hat{\mathbf{C}}_k = \mathbf{P}_k - \mathbf{K}\mathbf{C}_{yy}\mathbf{K}^T$

From the knowledge of the initial state vector and covariance matrix, the algorithm generate a set of $2n$ sigma points (where n is the number of elements in the state vector to be estimated). These sigma points are then propagated by using the physical model ($x_{k+1} = f(x_k, t_{k+1})$), the propagated sigma points are than used to estimate the mean value and the covariance matrix of the propagated state vector. Propagated sigma points are also used to estimate the measurements, by using a function which relate the state vector to the measurements vector ($y_{k+1} = g(x_{k+1})$) according to the sensor.

Once obtained the measurement vector from the sensors, the difference between this vector and the predicted measurement vector is evaluated to obtain the Kalman gain. Kalman gain is finally used to update the propagated state vector and covariance matrix.

It can be observed that Unscented Kalman Filter processing has the same structure of linear and extended Kalman Filters (LKF and EKF respectively). The main difference with the UKF avoids the linearization process, which is necessary both in LKF and EKF. The error introduced by the linearization process could be negligible in certain application, but in this case also a small error could imply bigger error during time, due to a continuous propagation of the state and consequently of the error. Since the Unscented Kalman Filter doesn't require the linearization of propagation and measurement functions this algorithm minimizes the error propagation and maintains accuracy in the estimation.

UKF steps are described in Tab. 6.

2.2 Simulator

In order to perofm performance evaluation of the proposed algorithm a MATLAB simulator has been developed. The simulator scheme is shown in Fig. 3.

The simulator is composed of three main blocks: Orbit generation, sensors manager and filter processing.

Orbit generator blocks take as input the initial epoch and orbit parameters, the mission plan, which specifies the burns scheduling for the transfer orbit, and the physical model. Once specified, all the parameters in this block generate the trajectory for the mission duration at the specified time step.

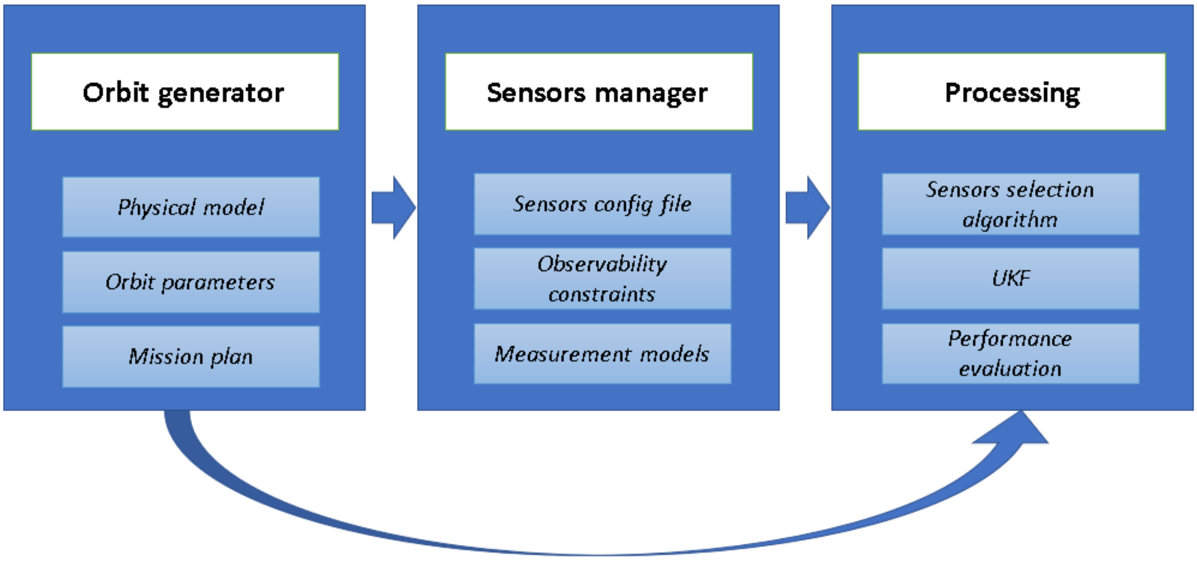

Figure 3: Developed simulator scheme

Once the orbit has been generated over the mission duration, the sensors management block is used to generate the observation for all the sensors contained in a configuration. The *.json file shown in Fig. 4, fig. 5 and Fig. 6 shows an example of the configuration file.

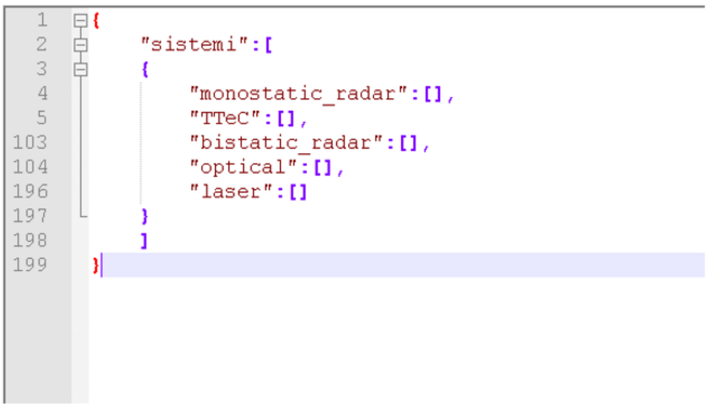

Figure 4: Sensors config file structure

The file content specifies the sensor type (TT&C or optical), its geodetic position, its weekly availability, the accuracy, the elevation constraint and the cost, as defined in Tab. 1. In this block, for each time step, the availability of the sensor and the Line of Sight between the sensor and the satellite is verified. For optical sensors, the lighting conditions are verified,

```
    {
        "monostatic_radar":[],
        "TTeC":[
            {
                "TTeC_id":"TTeC001",
                "Name":"Altair_Alcor_TTeC",
                "Available":"True",
                "Latitude":8.726659,
                "Longitude":167.735033,
                "Altitude":3134,
                "Min_Azimut":-180,
                "Max_Azimut":180,
                "Min_Elevation":25,
                "Max_Elevation":85,
                "Range_resolution":30,
                "Range_rate_resolution":1,
                "Cost":1,
                "Availability":[1, 1, 1, 1, 1, 1, 1]
            },
            {
                "TTeC_id":"TTeC002",
                "Name":"AN_FPS_85",
                "Available":"True",
                "Latitude":30.572399,
                "Longitude":-86.214952,
                "Altitude":3134,
                "Min_Azimut":-180,
                "Max_Azimut":180,
                "Min_Elevation":25,
                "Max_Elevation":85,
                "Range_resolution":30,
                "Range_rate_resolution":1,
                "Cost":1,
                "Availability":[1, 1, 1, 1, 1, 1, 1]
            },
```

Figure 5: TT&C sensors config example

```
        "bistatic_radar":[],
        "optical":[
            {
                "Optical_id":"opt001",
                "Name":"Altair_Alcor_Opt",
                "Available":"True",
                "Latitude":8.726659,
                "Longitude":167.735033,
                "Altitude":15,
                "Min_Azimut":-180,
                "Max_Azimut":180,
                "Min_Elevation":25,
                "Max_Elevation":85,
                "ScanTime":1e-1,
                "FOV_x":0.65783,
                "FOV_y":0.44,
                "Resolution":0.00026667,
                "Cost":1,
                "Availability":[1, 1, 1, 1, 1, 1, 1]
            },
            {
                "Optical_id":"opt002",
                "Name":"Diego_Garcia",
                "Available":"True",
                "Latitude":-7.295956,
                "Longitude":72.388964,
                "Altitude":15,
                "Min_Azimut":-180,
                "Max_Azimut":180,
                "Min_Elevation":25,
                "Max_Elevation":85,
                "ScanTime":1e-1,
                "FOV_x":0.65783,
                "FOV_y":0.44,
                "Resolution":0.00026667,
                "Cost":0.5,
                "Availability":[1, 1, 1, 1, 1, 1, 1]
```

Figure 6: Optical sensors config example

such that during day time the sensor is considered unavailable. For each sensor and for each set of measurements a random error is then added according to the specified sensor measurement accuracy.

Finally, the processing block designate the selected sensors optimizing the cost function defined by eq. 1, collects the measurements obtained from each selected sensor and sort them. Sorted measurements are then used by the UKF to estimate the state vector of the satellite. Once completed the state vector estimation, performance evaluation is performed comparing the mean position covariance between the estimated state using only the selected sensors and the covariance obtained using all the sensors without the optimization.

2.3 Scenario

2.3.1 Satellite and generated orbit

The proposed scenario will consider a cubic satellite with an area to mass ratio of $\frac{A}{m} = 0.294\frac{m^2}{kg}$ equipped with four BHT8000 Ion thruster, with a total thrust of 499 mN. The mission starts on 24th July 2019 at 10:16:28 UTC and ends on 21st November 2019 at 10:16:28 UTC, for a complessive duration of 120 days. The initial and final orbit parameters are listed in Tab. 7.

Epoch UTC	SMMA [km]	ECC	INC [deg]	RAAN [deg]	AOP [deg]	TA [deg]
24-Jul-2019 10:16:28	24371.1685	0.7301	7	0	0	0
21-Nov-2019 10:16:28	43670	0.001	0.9264	349.957	9.15	357.43

Table 7: Initial and final orbit parameters

The first step of the mission modifies the inclination of the orbit. When the satellite approaches the apogee ($\pm 20 deg$) a burn occurs in the direction normal to the orbital plane. Once the inclination reaches the final value the orbit eccentricity is modified in order to transform the orbit from elliptical to quasi-circular; this is carried out with burns in a tangential direction. Finally, the semimajor axis is raised to obtain a GEO orbit. The generated orbit is shown in Fig. 7 and 8, in which each axis is scaled with respect to the Earth radius.

The orbit is obtained by using a Runge-Kutta propagator, with a step of 60 seconds and a

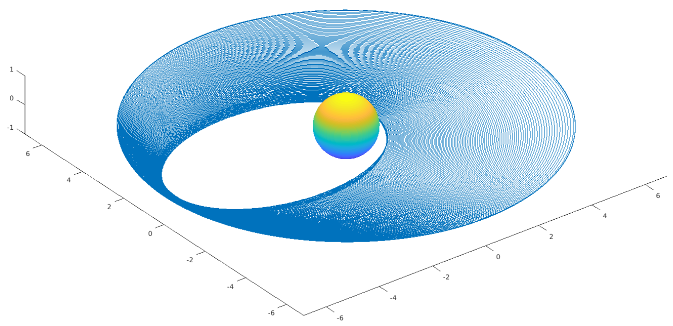

Figure 7: Generated orbit

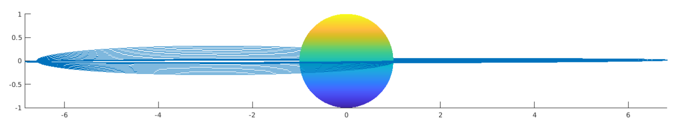

Figure 8: Generated orbit, side view

relative and absolute tolerance of 10^{-9}. The physical model includes the gravitational model (EGM2008 degree 20), the atmospheric drag model, the gravitational effect of sun and moon and the Solar Radiation Pressure, with an overall accuracy of $10^{-9} m/s^2/s$, which implies an accumulated error of 3.7325 m/day.

2.3.2 Sensor network and availability constraints

To validate the proposed method two scenario are being considered. The proposed sensor network for each scenario are listed in Tab. 8 and Tab. 9, in which an availability for each day of the week is specified and the cost refers to Tab. 1.

Sensor	**Latitude**	**Longitude**	**Type**	**Availability MTWTFSS**	**Cost**
Altair/Alcor	8.726659	167.735033	TT&C/Opt	XXXXXXX	1
AN/FPS-85	30.572399	-86.214952	TT&C	XXXXXXX	1
Ascension	-7.973012	-14.400839	TT&C	–X-X–	1
Cobra Dane	52.737295	174.091402	TT&C	-X—X-	1
Diego Garcia	-7.295956	72.388964	Opt	XXXXXXX	1/2
Globus II	70.3671	31.1271	TT&C	XXXXXXX	2
HAYSTACK	42.6233	-71.488198	TT&C	X–X–X	1/2
Maui	20.7083	-156.2571	Opt	-X–X-X	1/2
MOSS (Moron Air Base)	37.182364	-5.604194	Opt	XXXXXXX	1
Socorro	33.817202	-106.659938	Opt	XXXXXXX	1

Table 8: Sensor network and parameters - Scenario 1

Sensor	**Latitude**	**Longitude**	**Type**	**Availability MTWTFSS**	**Cost**
Altair/Alcor	8.726659	167.735033	TT&C/Opt	X–X–X	1
AN/FPS-85	30.572399	-86.214952	TT&C	XXXXXXX	1
Ascension	-7.973012	-14.400839	TT&C	–X-X–	1
Cobra Dane	52.737295	174.091402	TT&C	XXXXXXX	1
Diego Garcia	-7.295956	72.388964	Opt	XXXXXXX	1/2
Globus II	70.3671	31.1271	TT&C	XXXXXXX	2
HAYSTACK	42.6233	-71.488198	TT&C	X–X–X	1/2
Maui	20.7083	-156.2571	Opt	-X–X-X	1/2
MOSS (Moron Air Base)	37.182364	-5.604194	Opt	XXXXXXX	1
Socorro	33.817202	-106.659938	Opt	XXXXXXX	1

Table 9: Sensor network and parameters - Scenario 2

Each TT&C sensor will provide range, range-rate measurements and each optical sensor will provide right ascension and declination measurements. Concerning range and range-rate measurements, all TT&C sensors provide a resolution of 30 m and 1 m/s respectively, concerning right ascension and declination each optical sensor will provide $5 \cdot 10^{-6}$ rad resolution. According to this resolution values, an additive error is applied to the ideal measurements.

Additive error is modeled as a zero mean gaussian random variable having variance expressed by eq. 5, where Δ_{res} is the sensor's resolution.

$$\sigma_{meas} = \frac{\Delta_{res}^2}{12} \quad (5)$$

Optical sensor availability will also depends on the sun position, which makes them available only during night time.

Both TT&C and optical sensor availability is also subject to the satellite elevation relative to the sensor local horizon. Particularly, the satellite is considered in Line of Sight (and the sensor is available) if the elevation is greater than 20 degree. Measurement are acquired during all the LOS period of the satellite. According to this setup, Fig. 9 and Fig. 10 shows the Hard Check availability (yellow means that the considered sensor is available) for all the mission duration for Scenario 1 and Scenario 2 respectively.

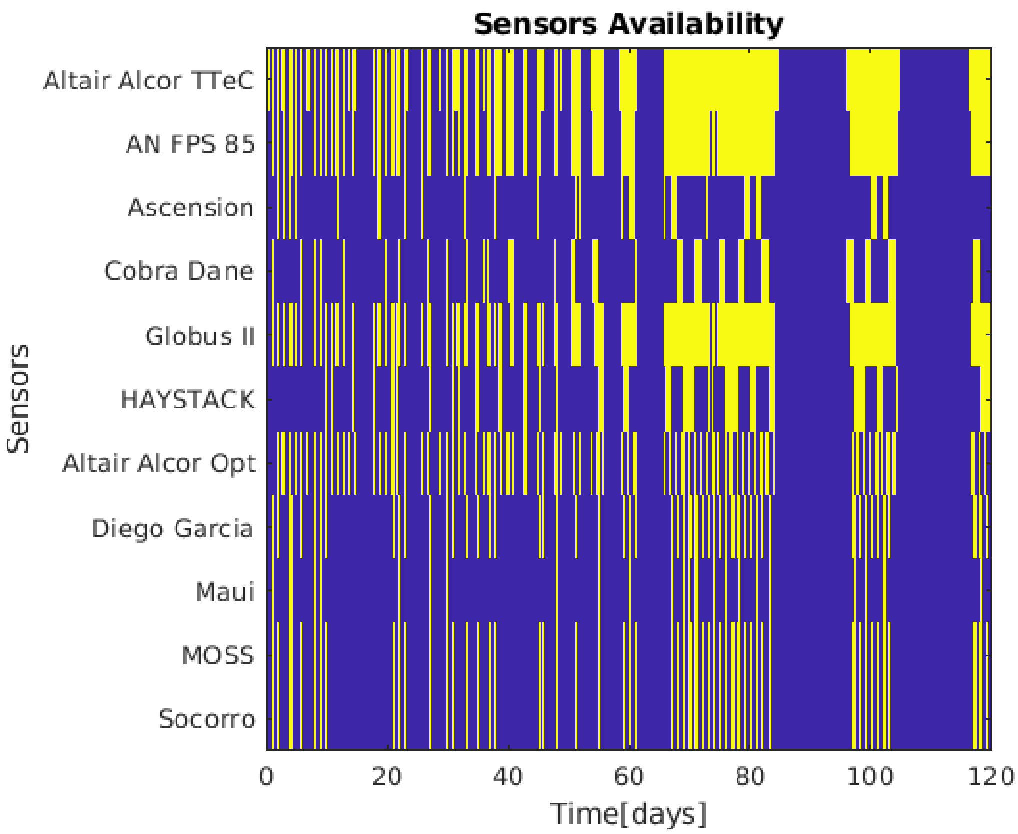

Figure 9: Sensors Availability - Scenario 1

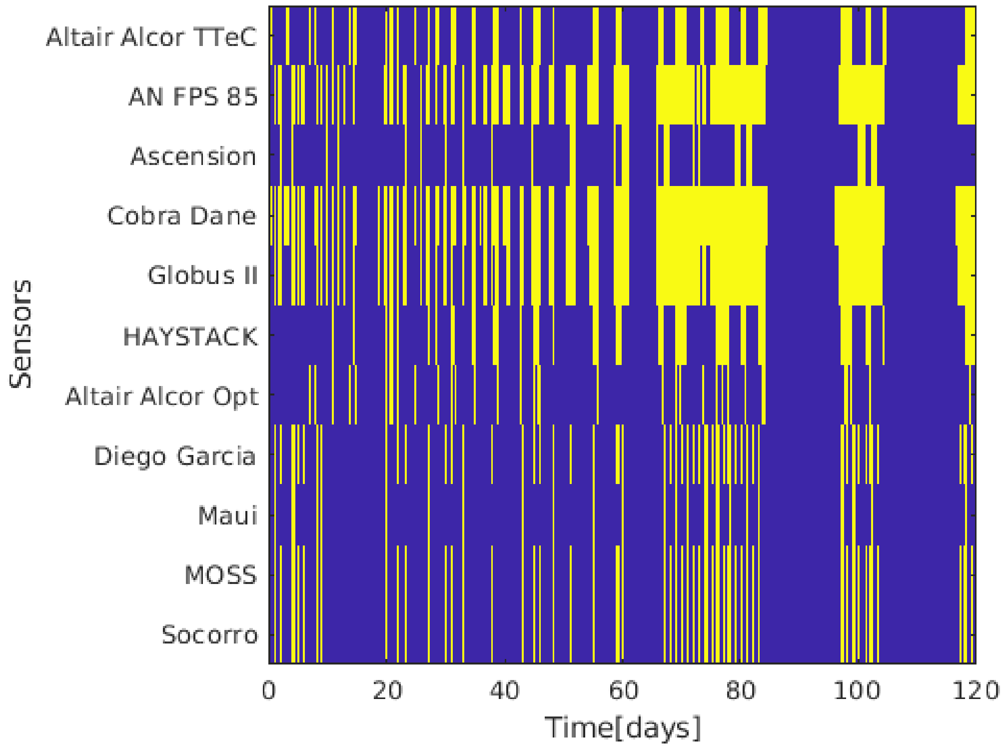

Figure 10: Sensors Availability - Scenario 2

3 Results and Discussions

According to the scenario and the methods previously defined, simulations are carried out in order to perform a performance analisys to validate the proposed approach.

The optimal sensor set, according to the proposed algorithm, previously described, are Altair/Alcor, AN/FPS-85 and Diego Garcia for Scenario 1 and Cobra Dane, AN/FPS-85 and Diego Garcia for Scenario 2, which represents, in both scenarios, the 30% of the sensors set. The overall observation time of this sensor subset is shown in Fig. 11 and 12.

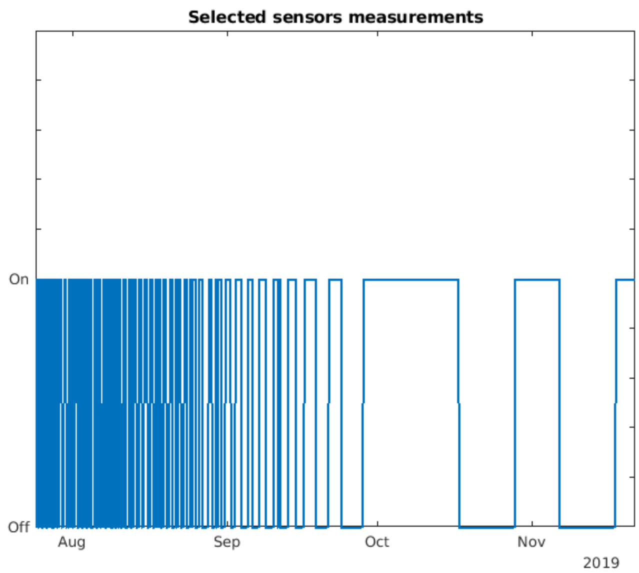

Figure 11: Selected sensors measurements timing - Scenario 1

It can be observed that in both scenarios the measurement time is almost the same except for a time shift.

The algorithm was run for a duration of two days for each scenario, which is an acceptable time for a pre-mission analisys and can be reduced by using faster machines. This operation doesn't need to be run again, because all the subset cost evaluations could be stored and used as a lookup table in case of availability changes.

Once the selected sensor subset is obtained, the UKF is performed using simulated measurements from each sensor, estimating position, velocity and acceleration of the satellite. It must

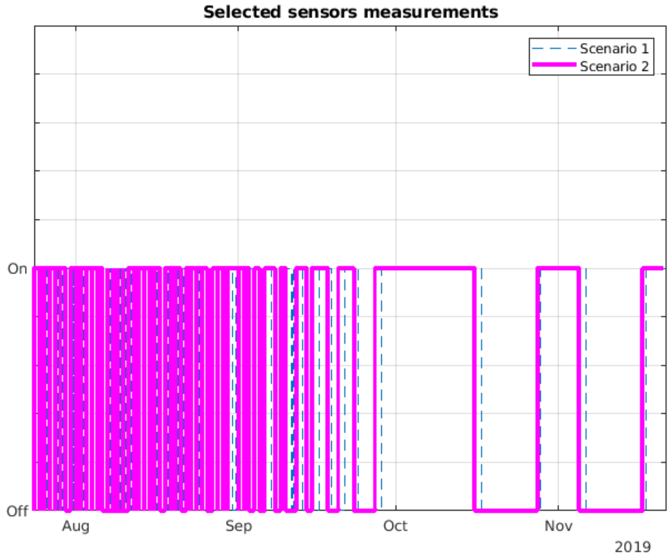

Figure 12: Selected sensors measurements timing - Scenario 2

be observed that in case of cooperative satellite, as in this case, the acceleration and burn time is apriori known by mission control with a certain accuracy (in this case the considered acceleration measurement accuracy is the 10%, 10 mN). Fig. 13 and fig. 14 show the 3σ value for the estimated position considering all the sensors of the network and the selected only sensors in Scenario 1. Fig. 15 and Fig. 16 show the 3σ value for both Scenario 1 and Scenario 2.

It can be observed that in steady state processing, the estimation accuracy is the same as in the case in which all the sensors are used for both scenarios. The difference between the use of a subset of sensors influences the time of the algorithm to reach the steady state condition. Observing Fig. 16, this scenario reaches the steady state condition after a couple minutes later than Scenario 1.

It can be also observed that only the 30% of the available sensors are used in both scenarios which could represent a significant cost reduction. Furthermore, it should be noted that the sensor Globus II, which has the highest cost, has not been selected, highlight the advantage of using such optimization algorithm.

A further analysis has been carried out considering a case in which only one sensor is used to perform orbit determination. The Altair/Alcor sensor has been considered in this case.

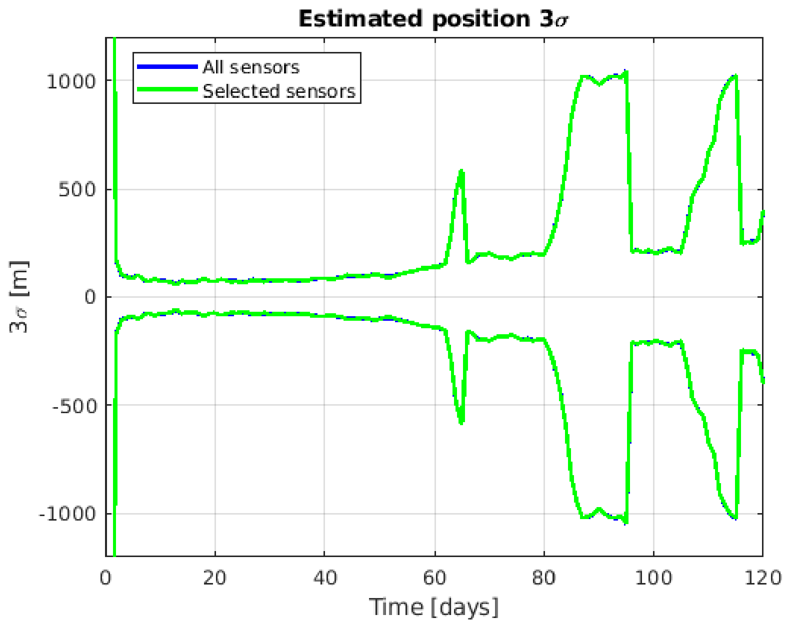

Figure 13: Estimated position 3σ - Scenario 1

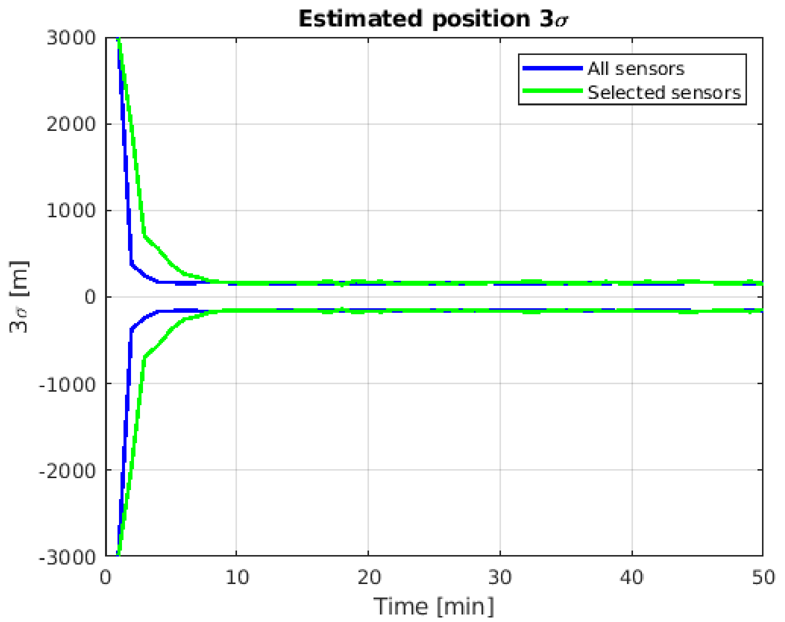

Figure 14: Estimated position 3σ detail - Scenario 1

Fig. 17 shows the 3σ value for all considered scenarios and for the case in which only Altair/Alcor sensor is used. It can be observed that at steady state the performances are almost the same, but to reach this condition it takes a week of measurements, compared to only a few

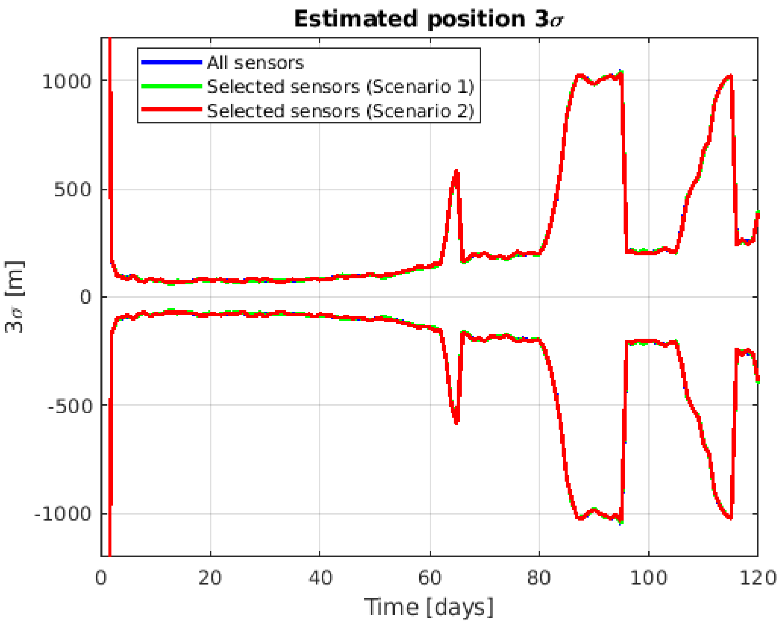

Figure 15: Estimated position 3σ - Scenario 2

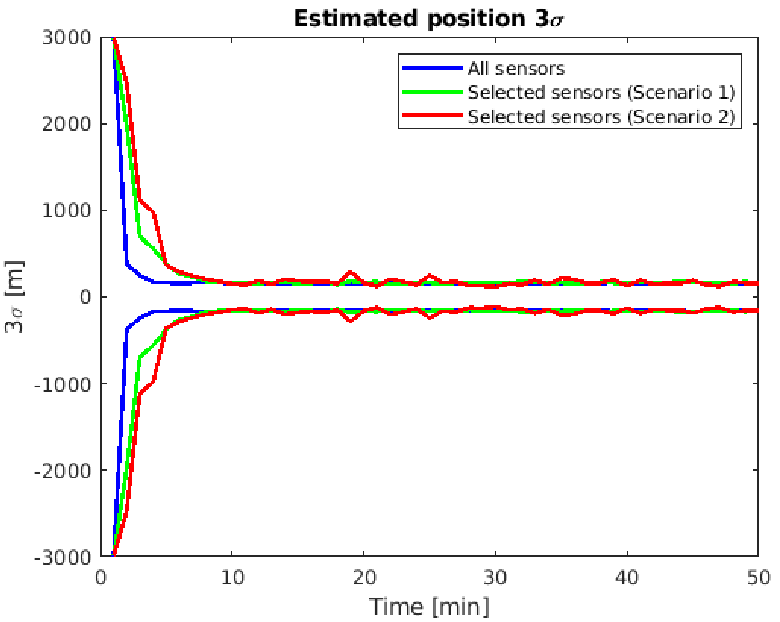

Figure 16: Estimated position 3σ detail - Scenario 2

minutes for the sensor combinations used in scenarios 1 and 2.

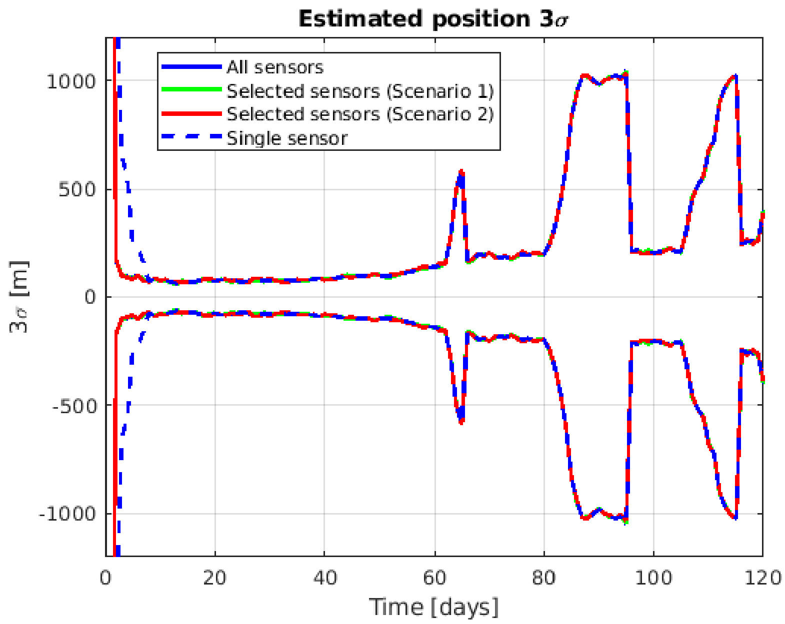

Figure 17: Estimated position 3σ - Single sensor comparison

4 Conclusions

In this technical report a new approach for selecting sensors tasking a GTO Low Thrust satellite has been analyzed. This method performs a selection of an optimal sensor subset from the network by minimizing a newly introduced cost function that considers a priori information such as the model accuracy, the sensor accuracy, sensor diversity and sensor cost. This method has been validated by performing simulations in a cooperative satellite scenario with known acceleration vector and availability of TT&C sensors.

Results have shown that in the case of a cooperating target, a small number of sensors is still sufficient to maintain a good level of performances, compared to using all the available sensors in the network. The only difference is the reduced convergence time, i.e. the time needed for the algorithm to reach a steady state condition, as shown in Fig. 14.

Although this may be expected, the algorithm that has been developed establishes the basis for more complex scenarios to be considered such as that of a non-cooperative satellite. This is recommended as a direct follow-on activity which would also incorporate the estimation of the low thrust acceleration into the UKF.

5 Appendix

In this section the MATLAB main file and the cost function minimization code are included:

```
close all;
clear all;
clc;

load_libraries;
load('Orbit.mat');

sensors = jsondecode(fileread('sensors.json'));

tt = floor(days(Orbit.time - Orbit.time(1)));
dd = mod(tt, 7)+1;

time_on_tmp = [];

%% Hard check evaluation
II = 1;

for kk = 1:length(sensors.TTeC)
    time_on = zeros(size(tt));
    Index_Aval = find(sensors.TTeC(kk).Availability == 1);
    for jj = 1:length(Index_Aval)
        time_on(find(dd== Index_Aval(jj)) ) = 1;
    end
```

```
    time_on_tmp = [time_on_tmp; time_on*II];
    II = II+1;
    sensors.TTeC(kk).time_on = time_on;
end

for kk = 1:length(sensors.optical)
    time_on = zeros(size(tt));
    Index_Aval = find(sensors.optical(kk).Availability == 1);
    for jj = 1:length(Index_Aval)
        time_on(find(dd== Index_Aval(jj)) ) = 1;
    end

    time_on_tmp = [time_on_tmp; time_on*II];
    II = II+1;
    sensors.optical(kk).time_on = time_on;
end



for kk = 1:length(Orbit.time)
    for rr = 1:length(sensors.optical)
        [Az, El] = SolarAzEl(Orbit.time(kk), ...
            sensors.optical(rr).Latitude ,...
            sensors.optical(rr).Longitude, ...
            sensors.optical(rr).Altitude);
        if El > -15
            sensors.optical(rr).time_on(kk) = 0;
```

```
        end
    end
end




for kk = 1:length(Orbit.time)
    [p, v] = eci2ecef(Orbit.State_ECI(kk, 1:3).', ...
                    datevec(Orbit.time(kk)),...
                    EOP, Orbit.State_ECI(kk, 4:6).', Parametri);
    for rr = 1:length(sensors.TTeC)
        sensors.TTeC(rr).osservato(:, kk) = ...
            TTeC_MeasurementFunction(p,...
            lla2ecef([sensors.TTeC(rr).Latitude, ...
            sensors.TTeC(rr).Longitude,...
            sensors.TTeC(rr).Altitude]), datevec(Orbit.time(kk)), ...
            Parametri);
    end

    for rr = 1:length(sensors.optical)
        sensors.optical(rr).osservato(:, kk) = ...
            AngularMeasurementFunction(p,...
            lla2ecef([sensors.optical(rr).Latitude,...
            sensors.optical(rr).Longitude, ...
            sensors.optical(rr).Altitude]),...
            datevec(Orbit.time(kk)), Parametri);
    end
```

```
end


for rr = 1:length(sensors.TTeC)
    sensors.TTeC(rr).obs_time = ...
    double(sensors.TTeC(rr).osservato(3, :)>20 & ...
    sensors.TTeC(rr).time_on);
end


for rr = 1:length(sensors.optical)
    sensors.optical(rr).obs_time = ...
    double(sensors.optical(rr).osservato(3, :)>20 & ...
    sensors.optical(rr).time_on);
end


Sorted_measurements = sorting_Measurements(sensors);
Selected_measurements = ...
        CostFunctionOptimization(Sorted_measurements, ...
       length(sensors.optical) + length(sensors.TTeC), 1e-12);


Index = find(Orbit.time == Selected_measurements(1).Time);
t_prev = (Orbit.time(Index, :));
X_prev = Orbit.State_ECI(Index, :);


for kk = 2:length(Selected_measurements)
      t_next = datevec(Selected_measurements(kk).Time);
      y_next = Selected_measurements(kk).dato;
```

```
        if(strcmp(Selected_measurements(kk).Type, 'Opt'))
            measurementFunction = 'OpticalMeasurementFunction';
        else
            measurementFunction = 'TTeCMeasurementFunction';
        end
        sensore_lla = Selected_measurements(kk).Position_lla;
        Res = Selected_measurements(kk).Resolution;
        N = ((Res/sqrt(12))*eye(2)).^2;
        [X_next, P_next] = UKF_step(X_prev, P_prev, ...
        t_prev, t_next, y_next, Parametri, sensore_lla, N, Q, ...
        RSO, Sistema, measurementFunction);
        X_prev = X_next;
        P_prev = P_next;
        t_prev = t_next;

        X_stim(kk).Dato = X_next;
        X_stim(kk).Cov = P_next;
end




function [Selected_measurements, MeanCost] = ...
    CostFunctionOptimization(Sorted_measurements, N_sensors, sigma_p)
        Selected_measurements = [];
        Scenarios = [];
        for kk = 0:(2^length(sensors))-1
```

```
        Scenarios = [Scenarios; (dec2bin(kk, 10))];
    end



CostFunctionMatrix = zeros(length(Scenarios), ...
    length(Sorted_measurements));
for kk = 2:length(Sorted_measurements)-1
    for jj = 1:length(Scenarios)
        for rr = 1:N_sensors
            Avail = ...
            str2num(Scenarios(Sorted_measurements(kk).SensorNumber)
            Sorted_measurements(kk).HardCheck = ...
            Sorted_measurements(kk).time_on * );

                if Sorted_measurements(kk-1).Type == ...
                Sorted_measurements(kk).Type
                    D = 1;
                else
                    D = 2;
                end
                CostFunctionMatrix = ...
                Sorted_measurements(kk).HardCheck * ...
                    Sorted_measurements(kk+1) - ...
                    Sorted_measurements(kk-1) * sigma_p * ...
                    Sorted_measurements(kk).sigma_sensor * ...
                    Sorted_measurements(kk).Cost   /D;
            end
```

```
            end
        end

        MeanCost = mean(CostFunctionMatrix, 2);
        Selected_Scenario = Scenarios(find(MeanCost == min(MeanCost)));

        for kk = 2:length(Sorted_measurements)-1
            for jj = 1:length(Selected_Scenario)
                if Sorted_measurements(kk).SensorNumber == jj
                    Selected_measurements = ...
                    [Selected_measurements Sorted_measurements(kk)];
                end
            end
        end
end
```

References

[1] S. Chakravorty D. A. V. Raihan, W. R. Faber and I. I. Hussein. Parallel markov chain monte carlo for sensor scheduling. *Proceedings of the AAS/AIAA Astrodynamics Specialist Conference*, 2018.

[2] R. Linares and R. Furfaro. Dynamic sensor tasking for space situational awareness via reinforcement learning. *Advamced Maui Optical and Space Surveillance Technologies Conference*, 2016.

[3] Simo Srkk. On unscented kalman filtering for state estimation of continuous-time nonlinear systems. 2007.

[4] E. A. Wan and R. Van Der Merwe. The unscented kalman filter for nonlinear estimation. In *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No.00EX373)*, pages 153–158, 2000.

[5] R. S. Erwin Z. Sunberg, S. Chakravorty. Information space receding horizon control for multisensor tasking problems. *IEEE Transactions On Cybernetics*, 46(6):1325–1336, 2016.