IRSTI 41.03.51 https://doi.org/10.26577/ijmph.2023.v14.i1.02

P. Xu, N.B. Kalieva, Z.B. Rakisheva*

Al-Farabi Kazakh National University, Almaty, Kazakhstan
*e-mail: zaure.ra@gmail.com

(Received 27 March 2023; accepted April 21)

Development of a program for the prediction of placement of spacecraft based on TLE data

Abstract. Accurate prediction of spacecraft placement is crucial for mission completion and scientific research. Based on the archive of the TLE data of the spacecraft, the placements of the Low Earth Orbit spacecraft (Al-Farabi-2), Medium Earth Orbit spacecraft (BEIDOU-3 M20), and High Earth Orbit spacecraft (IPM 2 & BREEZE-M R/B) were predicted in coordinate form using the first set of TLE data and SGP4/SDP4 model in Python. This work also employs the STK software for comparison purposes. The actual placements of these spacecraft are obtained by parsing their TLE data archives, and the errors of the two prediction methods are calculated. The errors are compared to evaluate the similarity and prediction accuracy of the two methods. This comprehensive analysis allows for an assessment of the feasibility of predicting spacecraft placements using a Python program as an alternative to the STK software. The results indicate that the Python-based method can be effectively used for accurate satellite coordinate prediction, offering a more accessible and cost-effective alternative to the STK software.

Key words: placement of spacecraft, placement prediction, TLE data, Celestial Mechanics, Al-Farabi-2 satellite.

Introduction

With the increasing number of spacecraft launches, the widespread use of small satellites worldwide in recent years, and the emergence of space debris and satellite fragments [1], accurately predicting the placement of spacecraft is a crucial research topic. Improving the prediction accuracy of the placement of spacecraft in orbit is of extraordinary significance for the completion of spacecraft missions and scientific research. However, the prediction of the placement of artificial spacecraft requires consideration of various factors such as gravitational resonance caused by atmospheric drag models [2] and orbital decay (especially for LEO artificial satellites), the Earth's oblateness, and the long-term and periodic perturbations of the Sun and Moon's gravities due to the non-ideal two-body motion of the spacecraft.

This work uses the Two Lines Elements (TLE) as the primary reference data for predicting spacecraft placements. TLE is a set of orbital elements based on general perturbation theory that describes spacecraft's placement and motion acceleration and calculates and predicts their placements [3]. Max Lane's mathematical models of spacecraft elements, widely used by the military and NASA, became the standard for the North American Aerospace Defense Command (NORAD), creating the TLE format. TLE consists of two lines of data, including the satellite catalog number, epoch year, satellite epoch, inclination, and so on [4]. The TLE considers various perturbation factors such as Earth's atmospheric drag, oblateness, and sun-moon gravity.

The use of the STK software is a common method to work on spacecraft parameters. For example, Dai Yong [5] investigated the design of a 5G LEO satellite constellation by using STK to determine the satellite's orbital parameters; CP Dang used STK and MATLAB to analyze the relationship between fly-around parameters and the orbital elements [6]. STK (System Tool Kit, formerly known as Satellite Tool Kit) is a commercially available software application that Analytical Graphics, Inc. has developed since 1989 [7]. It incorporates high-precision perturbation models, allowing it to calculate and predict spacecraft positions based on TLE data using the SGP4/SDP4 (Simplified General Perturbation Version 4/Simplified Deep-space Perturbation Version 4) models. As a successful commercial software, STK boasts outstanding 3D graphics rendering capabilities, allowing users to visualize the positions or operational status of satellites, rockets, spacecraft, ground antennas, and even aircraft carriers. Its accurate built-in models and comprehensive data analysis templates allow users to obtain data and charts to analyze the information more intuitively. However, due to STK's high precision calculations and 3D graphics rendering, complex built-in models and sub-programs, and high acquisition and learning costs, STK is only famous among large tech companies, aerospace firms[7], defense industries, and renowned research institutes, and not widely used in public universities, college teachers and students, and amateur astronomers.

A literature review shows that it is possible to use programming languages to implement the corresponding tasks while avoiding the limitations of STK software. For example, Zhao Jian used Visual C++ and OpenGL to visualize satellite trajectories under the Windows system [8]; Xiaoheng Liang designed a cross-platform simulation software for efficient analysis and fast planning of multi-satellite Earth observation missions by C++ [9]. Therefore, this work also implements an alternative to STK software by using a programming language. With the rapid development of programming, increasingly specialized needs are being addressed. In recent years, as the specificities of demands across industries have grown, an increasing number of programming languages have emerged. Compared to the increasingly complex systems of software, programming languages are high-level languages similar to human languages, allowing users to complete specific tasks through designing, writing, running, and debugging programs [10]. Among the numerous programming languages, Python, C, Java, PHP, and others are the most widely used, with convenient system-building capabilities. Python, which supports adding third-party libraries such as SciPy and NumPy, is now widely used [11]. This work will analyze TLE data using Python and predict the placement of spacecraft.

This work will take LEO satellites (Al-Farabi-2, altitude is 577km), MEO satellites (BEIDOU-3 M20 altitude is 21571km), and HEO satellites (IPM 2 & BREEZE-M R/B altitude is 37609km) as examples. First, the TLE data set of satellites from Jan 01, 2023, to Jan 31, 2023, will be obtained. The first TLE data of each satellite will be parsed, and the placement of the satellite will be predicted through Python and STK. Then, all TLE data will be parsed, and the placement information of the satellite in all TLE data will be obtained as a reference for error calculation. Both prediction methods will be evaluated, and the actual significance of the program design method in satellite placement prediction will be obtained compared to the STK software.

Mathematical model

In celestial mechanics, the orbital elements, commonly known as the orbital six elements, are used to describe the position and state of the target celestial body relative to the central celestial body [12, 13]. This theory is widely used in the field of aeronautical and astronautical research. There are:

a – Semi-major axis or specific angular momentum h

e – Eccentricity

i – Inclination

ω – Argument of Perigee

Ω – RAAN, Right ascension of ascending node

M – Mean anomaly or θ – true anomaly (They can be converted through the eccentric anomaly E, where: $\cos\theta = \frac{\cos E - e}{1 - e\cdot\cos E}$, $M = E - e\cdot\sin E$)

These parameters describe the spacecraft's position and motion direction in its orbit, as well as the size, shape, and orientation of the orbit. Based on this theory, the placement and velocity of a satellite can be calculated. The orbital six elements of a satellite can be obtained by parsing TLE data.

The initial orbital elements $(a_0, e_0, I_0, \Omega_0, \omega_0, M_0)$ and the orbital elements at any time of the spacecraft are given $(a, e, I, \Omega, \omega, M)$.

In ideal conditions, a spacecraft moves in a two-body motion, but the predictions of this ideal state have significant errors due to the perturbation forces in the motion environment. Under the influence of these perturbations, the motion trajectory and velocity of the satellite do not follow the two-body motion model [14]. The eccentricity e, inclination i, right ascension of the ascending node Ω, and orbital period T of the satellite's orbit are constantly changing.

As shown in Figure 1, the perturbation forces experienced by the spacecraft can be represented:

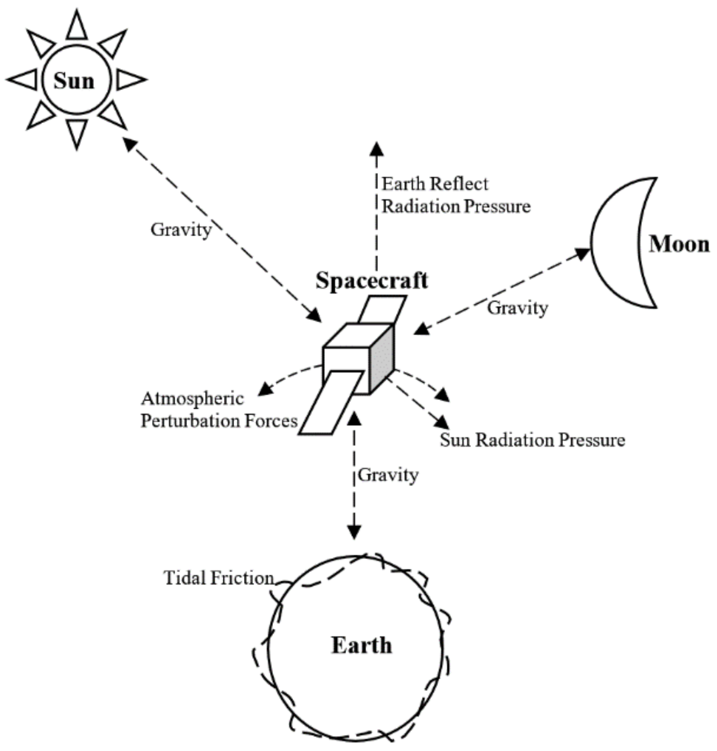


Figure 1 – Perturbation forces act on the spacecraft.

The above perturbation forces need to be substituted into the motion equations. The two-body motion equation can be represented as Equation (1):

$$\ddot{\bar{r}} = -\frac{\mu}{r^3}\bar{r} + \bar{r}_p \qquad (1)$$

$\bar{r}_p$ represents the sum of all the perturbation forces:

$$\bar{r}_p = \ddot{\bar{r}}_{Earth} + \ddot{\bar{r}}_{Sun} + \ddot{\bar{r}}_{Moon} + \ddot{\bar{r}}_{Tidal} + \\ + \ddot{\bar{r}}_{Atmospheric} + \ddot{\bar{r}}_{ERadiation} + \ddot{\bar{r}}_{SRadiation} \quad (2)$$

where,

$\ddot{\bar{r}}_{Earth}$ – Non-spherical and non-homogeneous Earth Perturbation

$\ddot{\bar{r}}_{Sun}$ – Solar Gravitational Perturbation

$\ddot{\bar{r}}_{Moon}$ – Lunar Gravitational Perturbation

$\ddot{\bar{r}}_{Tidal}$ – Tidal Friction

$\ddot{\bar{r}}_{Atmospheric}$ – Atmospheric Perturbation Forces

$\ddot{\bar{r}}_{ERadiation}$ – Earth Reflect Radiation Pressure

$\ddot{\bar{r}}_{SRadiation}$ – Sun Radiation Pressure

The perturbation forces are closely related to factors such as orbital height and orbital period. Earth's non-spherical gravitational force is the largest source of positional offset, and atmospheric drag significantly decreases with increasing orbital height [15]. Solar radiation pressure has a robust periodic nature and is broadly consistent with the orbital period of the spacecraft [16]. Here, the perturbation force due to Earth's non-spherical gravitational force (J_2) is chosen as the basis for mathematical derivation.

Based on period T, spacecraft can be divided into two categories: LEO spacecraft with T < 225 minutes and deep space spacecraft with T > 225 minutes. For the placement prediction of LEO spacecraft, the SGP4 model should be used [17]. The SDP4 model considers both the solar and lunar gravitational influence, so it is more appropriate for deep space spacecraft.

The mathematical derivation of SGP4 is based on the following principles [18]:

1. Kepler's laws of planetary motion, which are the three laws of planetary motion.

2. Atmospheric perturbation (static, non-rotating spherically symmetric atmosphere); fourth-order potential harmonics (determined by the Earth's flat rate uptake J_2, J_3, and J_4); spin-orbit resonances in synchronous and semi-synchronous orbits; solar and lunar gravitational perturbation effects; and

approximate gravitational field models: SGP4 uses a simplified model of the Earth's gravitational field to simplify calculations.

3. Orbit simplification for small bodies: The SGP4 model assumes the orbit of spacecraft motion is stable, which simplifies the model.

The gravitational acceleration of a spacecraft from a non-spherical planet can be given by [19]:

$$\ddot{\bar{r}} = -\frac{\mu}{r^2}\,\bar{u}_r + \bar{r}_p \qquad (3)$$

The first component of the right-hand side is dependent on the spherical planet; during the second term $\bar{r}_p$ means the perturbation acceleration due to the non-spherical planet.

Where perturbation acceleration can be expressed in the following formula:

$$\bar{r}_p = p_r\bar{u}_r + p_\perp\bar{u}_\perp + p_h\bar{h} \qquad (4)$$

And $\bar{u}_r$ is the radial unit vector of the spacecraft, pointing to the radial position.

$\bar{u}_\perp$ is the transverse spacecraft unit vector perpendicular to the radial vector.

$\bar{h}$ is the normal unit vector of the spacecraft, pointing to the spacecraft's motion direction.

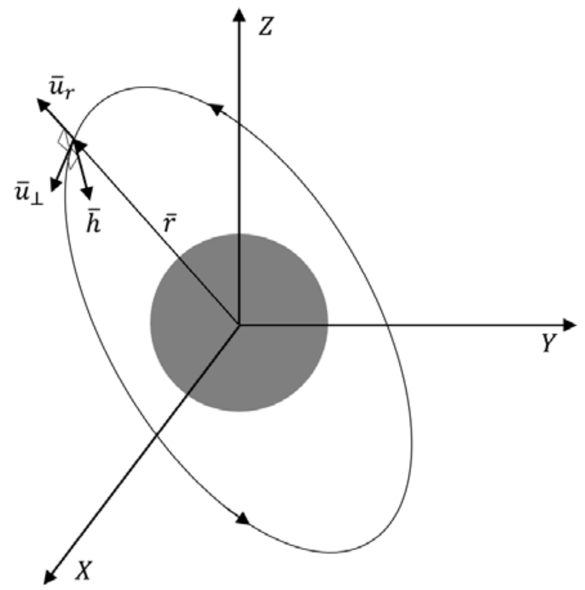


Figure 2 – Unit vectors of a spacecraft

The terms of perturbation are all proportional to J_2 and they are functions of orbital parameters, so the terms can be written as follows:

$$p_r = -\frac{3\mu\cdot J_2\cdot R^2}{2r^4}\left[-3\sin^2 i\sin^2(\omega+\theta)+1\right] \qquad (5)$$

$$p_\perp = -\frac{3\mu\cdot J_2\cdot R^2}{2r^4}\sin^2 i\sin[2\omega+2\theta] \qquad (6)$$

$$p_h = -\frac{3\mu\cdot J_2\cdot R^2}{2r^4}\sin 2i\sin(\omega+\theta) \qquad (7)$$

These relations are derived by Prussing and Conway [14], who also shows how p_r , $p_\perp$ and p_h induce time rates of change in all of the orbital parameters.

The model calculates the position and velocity of a satellite in Earth-centered inertial (ECI) coordinates. You can refer to the following formulas and the reference for the SGP4/SDP4 model [3]:

SGP4 model for near-Earth satellites (period less than 225 minutes):

$$rECI = f(TLE, t, SGP4) \qquad (8)$$

SDP4 model for deep-space satellites (period greater than 225 minutes):

$$rECI = f(TLE, t, SDP4) \qquad (9)$$

where $rECI$ represents the position vector of the satellite in Earth-centered inertial coordinates, TLE is the Two-Line Element data, t is the time of interest, and SGP4/SDP4 are the specific perturbation models used for calculation.

After obtaining the ECI coordinates using the SGP4/SDP4 model, we need to convert them to Earth-centered Earth-fixed (ECEF) coordinates for practical use. This can be done using the following formula [20]:

$$rECEF = Rz(-GMST) * rECI \tag{10}$$

where $rECEF$ is the position vector in ECEF coordinates, Rz is the rotation matrix about the z-axis, and $GMST$ is the Greenwich Mean Sidereal Time.

For easier interpretation and usage, the ECEF coordinates can be converted to latitude, longitude, and altitude (LLA) using the following formulas [20] :

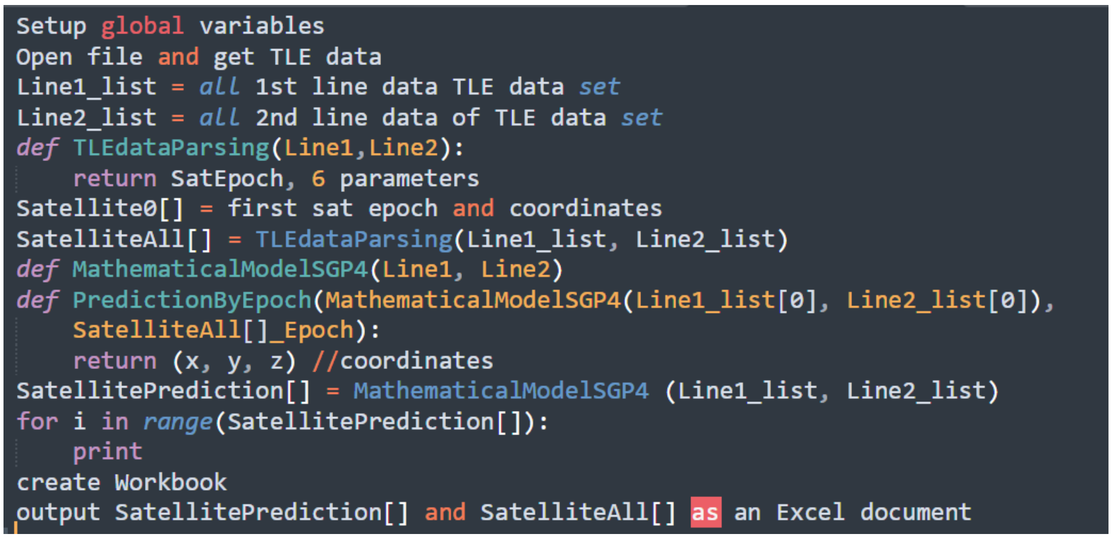
$$\phi = atan2\left(z, \sqrt{x^2 + y^2}\right) \tag{11}$$

$$\lambda = atan2\,(y, x) \tag{12}$$

$$h = \sqrt{(x^2 + y^2 + z^2)} - R_e \tag{13}$$

where ϕ is the latitude, λ is the longitude, h is the altitude, and R_e is the Earth's radius.

Based on the above mathematical model, combined with the description given on the website and the GitHub blog [21], the program was designed and written on the Python platform, and this program was named Placement_Prediction.py. The pseudocode of the Placement_Prediction.py program is given in Figure 3.



```
Setup global variables
Open file and get TLE data
Line1_list = all 1st line data TLE data set
Line2_list = all 2nd line data of TLE data set
def TLEdataParsing(Line1,Line2):
    return SatEpoch, 6 parameters
Satellite0[] = first sat epoch and coordinates
SatelliteAll[] = TLEdataParsing(Line1_list, Line2_list)
def MathematicalModelSGP4(Line1, Line2)
def PredictionByEpoch(MathematicalModelSGP4(Line1_list[0], Line2_list[0]),
    SatelliteAll[]_Epoch):
    return (x, y, z) //coordinates
SatellitePrediction[] = MathematicalModelSGP4 (Line1_list, Line2_list)
for i in range(SatellitePrediction[]):
    print
create Workbook
output SatellitePrediction[] and SatelliteAll[] as an Excel document
```

Figure 3 – Pseudocode of Placement_Prediction.py

Experimental process of placement prediction

To evaluate the accuracy of the SGP4/SDP4 model in predicting the positions of satellites with different orbital altitudes, in this work, satellites with eccentricities less than 0.1 ($e < 0.1$) and nearly circular orbits have been selected. Three different types of satellite orbits are considered, based on their altitude: Low Earth Orbit (LEO) with an altitude of less than 2000 km, Medium Earth Orbit (MEO) with an altitude between 2000 km and 35786 km, and High Earth Orbit (HEO) with an altitude higher than 35786 km [22]. The archive of TLE data for these three satellites over a period of one month was obtained from the website [23]. The first set of data was used as the initial data. The obtained TLE was transformed into orbital elements, and the geocentric coordinates of the satellite were calculated through the mathematical programming model, predicting the satellite coordinates for approximately one month. Then, the initial TLE data was directly used as input into the STK software to predict the satellite coordinates for one month. The process is illustrated in Figure 3.

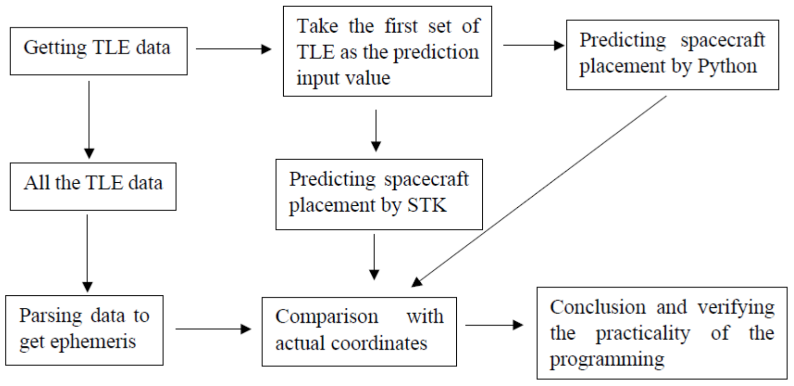

Getting TLE data
Take the first set of TLE as the prediction input value
Predicting spacecraft placement by Python
All the TLE data
Predicting spacecraft placement by STK
Parsing data to get ephemeris
Comparison with actual coordinates
Conclusion and verifying the practicality of the programming

Figure 4 – Experimental flowchart

The errors were calculated by comparing two sets of coordinates (x_A, y_A, z_A) and (x_B, y_B, z_B), which respectively represent the actual and predicted coordinates:

$$\Delta l = \sqrt{(x_A - x_B)^2 + (y_A - y_B)^2 + (z_A - z_B)^2} \quad (14)$$

And the accuracy of the prediction:

$$\%A = 100 - \frac{\Delta l}{\sqrt{x_A^2+y_A^2+z_A^2}} \quad (15)$$

Since Placement_Prediction.py was designed and written for TLE data parsing and spacecraft position prediction, the step size of spacecraft coordinate prediction can be changed arbitrarily according to the requirements, however, Placement_Prediction.py did not have the function of fitting the data, so the actual coordinates of the spacecraft at any given moment could not be inverted from a set of TLE data.

We assumed that the one-day TLE data of Jan 06, 2023, for satellite Al-Farabi-2 was acquired (as shown in Table 1), and there were three sets of TLE data for that day as follows:

Table 1 – TLE data of Jan 06, 2023

1 43805U 18099AZ 23008.17327929 .00005821 00000-0 48179-3 0 9998
2 43805 97.5961 75.5667 0013990 299.0463 60.9370 14.99187897223757
1 43805U 18099AZ 23008.64048900 .00005897 00000-0 48795-3 0 9996
2 43805 97.5962 76.0225 0013852 297.3414 62.6464 14.99194019223823
1 43805U 18099AZ 23008.90746400 .00006030 00000-0 49886-3 0 9997
2 43805 97.5962 76.2829 0013563 295.8858 64.1089 14.99198321223863

After parsing by the Placement_Prediction.py program, the epochs and actual coordinates of the three sets of TLE data were found to be:

Table 2 – Epochs and actual coordinates of TLE data of Jan 06, 2023

Epoch	2023 Jan 06 04:09:31.3307	2023 Jan 06 15:22:18.2496	2023 Jan 06 21:46:44.8896
Coordinates	(1765.899, 6717.656, -3.948)	(1712.559, 6731.741, -3.188)	(1682.110, 6739.717, -2.433)

After obtaining the epoch, the time Jan 06 04:09:31.3307 was provided as input to the Placement_Prediction.py prediction program, which gave the following predicted coordinates for the remaining moments Jan 06 15:22:18.2496 and Jan 06 21:46:44.8896.

Table 3 – Epochs and prediction of coordinates of TLE data of Jan 06, 2023

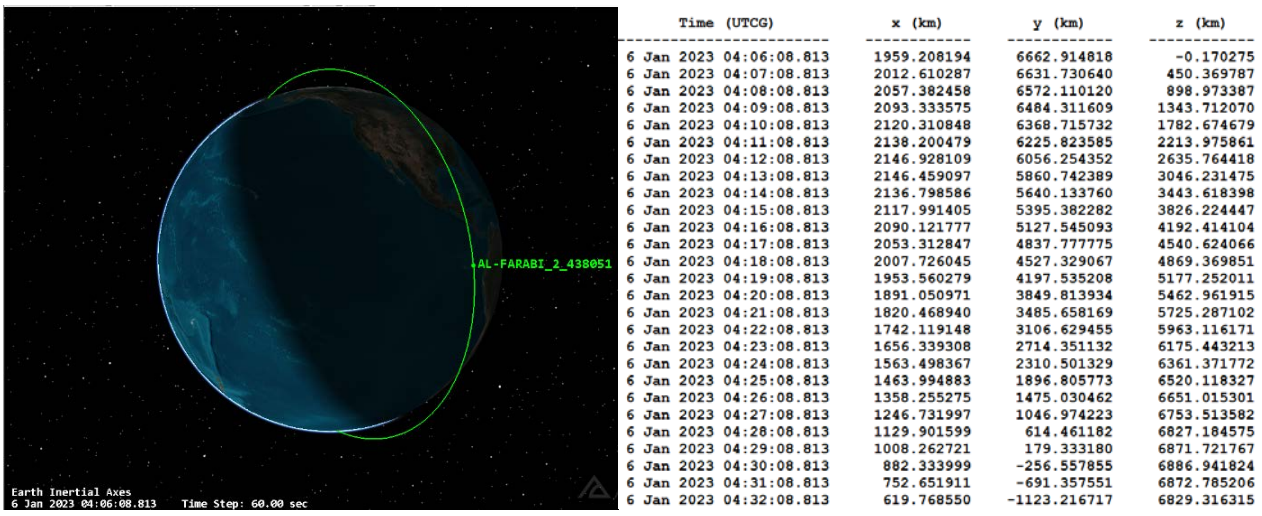
Epoch	2023 Jan 06 04:09:31.3307	2023 Jan 06 15:22:18.2496	2023 Jan 06 21:46:44.8896
Coordinates	(1765.899, 6717.656, -3.948)	(1711.813, 6731.857, -9.574)	(1681.075, 6739.726, -10.825)

Based on the above coordinates data, Placement_Prediction.py could calculate the coordinates errors. This prediction method was efficient with a short calculation time (less than one second), which was very suitable for real-time solving.

As the following figure shows, the software STK could give a 3D model with three sets of TLE data and the trajectory.

As shown above, the STK program could give satellite coordinates for any moment of the day and also predict satellite coordinates based on time intervals. It is not difficult to understand that the built-in algorithm of STK software supports all the functions of the Placement_Prediction.py program, but the aim to complete the prediction of coordinates could be fully realized by the Placement_Prediction.py program.

Taking the satellite Al-Farabi-2 (altitude is 577km) as an example for LEO satellite placement prediction, prediction started on Jan 06 at 04:06:08. The errors of the two prediction methods are shown in Figure 6 and Figure 7.



```
       Time (UTCG)              x (km)         y (km)         z (km)
------------------------    ------------   ------------   ------------
6 Jan 2023 04:06:08.813      1959.208194    6662.914818      -0.170275
6 Jan 2023 04:07:08.813      2012.610287    6631.730640     450.369787
6 Jan 2023 04:08:08.813      2057.382458    6572.110120     898.973387
6 Jan 2023 04:09:08.813      2093.333575    6484.311609    1343.712070
6 Jan 2023 04:10:08.813      2120.310848    6368.715732    1782.674679
6 Jan 2023 04:11:08.813      2138.200479    6225.823585    2213.975861
6 Jan 2023 04:12:08.813      2146.928109    6056.254352    2635.764418
6 Jan 2023 04:13:08.813      2146.459097    5860.742389    3046.231475
6 Jan 2023 04:14:08.813      2136.798586    5640.133760    3443.618398
6 Jan 2023 04:15:08.813      2117.991405    5395.382282    3826.224447
6 Jan 2023 04:16:08.813      2090.121777    5127.545093    4192.414104
6 Jan 2023 04:17:08.813      2053.312847    4837.777775    4540.624066
6 Jan 2023 04:18:08.813      2007.726045    4527.329067    4869.369851
6 Jan 2023 04:19:08.813      1953.560279    4197.535208    5177.252011
6 Jan 2023 04:20:08.813      1891.050971    3849.813934    5462.961915
6 Jan 2023 04:21:08.813      1820.468940    3485.658169    5725.287102
6 Jan 2023 04:22:08.813      1742.119148    3106.629455    5963.116171
6 Jan 2023 04:23:08.813      1656.339308    2714.351132    6175.443213
6 Jan 2023 04:24:08.813      1563.498367    2310.501329    6361.371772
6 Jan 2023 04:25:08.813      1463.994883    1896.805773    6520.118327
6 Jan 2023 04:26:08.813      1358.255275    1475.030462    6651.015301
6 Jan 2023 04:27:08.813      1246.731997    1046.974223    6753.513582
6 Jan 2023 04:28:08.813      1129.901599     614.461182    6827.184575
6 Jan 2023 04:29:08.813      1008.262721     179.333180    6871.721767
6 Jan 2023 04:30:08.813       882.333999    -256.557855    6886.941824
6 Jan 2023 04:31:08.813       752.651911    -691.357551    6872.785206
6 Jan 2023 04:32:08.813       619.768550   -1123.216717    6829.316315
```

Figure 5 – STK program for placement prediction

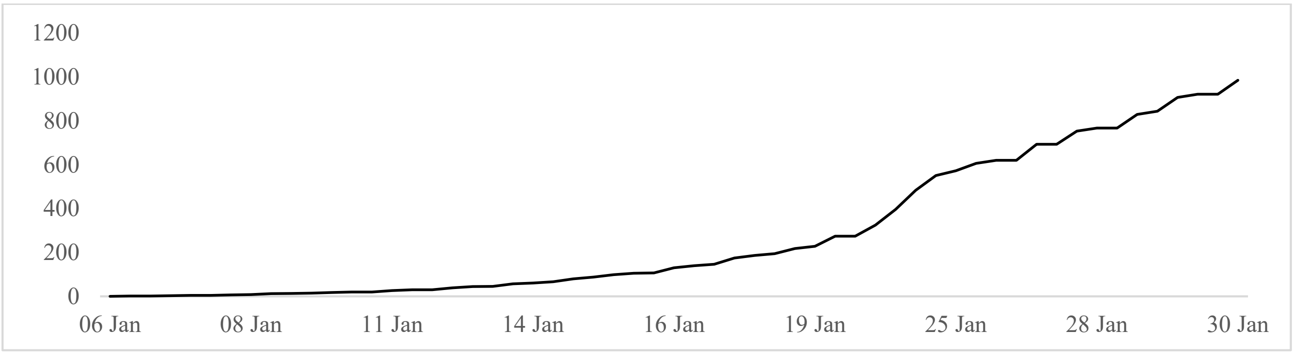


Figure 6 – Errors of LEO Placement in Placement_Prediction.py (unit: km)

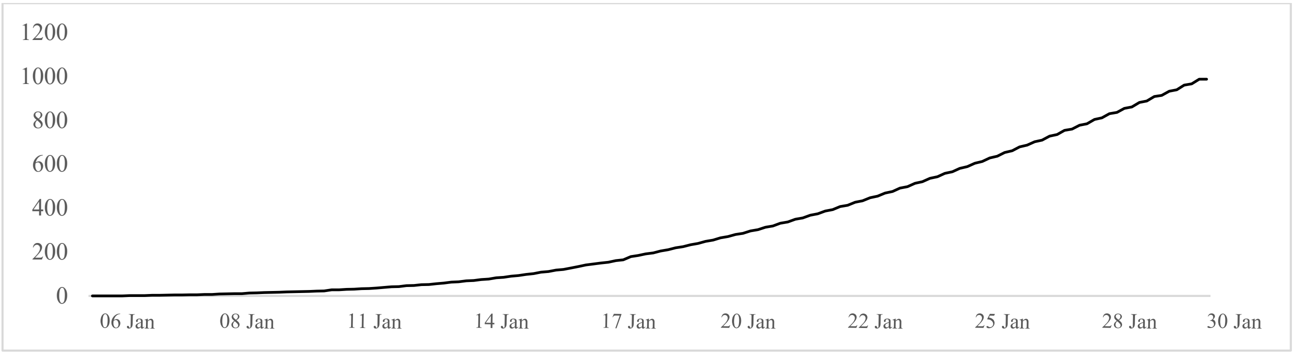


Figure 7 – Errors of LEO Placement in STK (unit: km)

According to the Excel table from the output to obtain the above graph in the overall trend, the predicted results for satellite Al-Farabi-2, Placement_Prediction.py, and STK software gave the same upward trend. At the prediction end time of Jan 30, 2023, the predicted spatial point errors are approximated at 1000km. The prediction results of Placement_Prediction.py show a sudden upward trend from the 21st to the 27th; the excessive prediction errors bigger than 200km make such a sudden upward trend of no research value.

From the data, it appears that Placement_Prediction.py's predictions started to have an error bigger than 10km after Jan 08, while the STK predictions did the same. Therefore, Placement_Prediction.py has the same reference value as STK for LEO predictions over a three-day period, and Placement_Prediction.py can be used as an alternative program to STK.

For comparison in accuracy, the prediction error by Placement_Prediction.py's program achieved 3.09 km on Jan 07 at 10:32:17, and the accuracy was 99.99955% according to Equation (9); the error of STK prediction was 3.167 km, and the accuracy was 99.99954%.

Taking the satellite BEIDOU-3 M20 (altitude is 21571km) as an example for MEO satellite placement prediction, prediction from Jan 01 at 10:48:16, the errors of the two prediction methods are shown in Figure 8 and Figure 9.

In the prediction of MEO satellites from Jan 01 at 10:48:16, Placement_Prediction.py did not have as much data density as STK, so its prediction results did not fluctuate like a wavy line. However, the maximum prediction error of Placement_Prediction.py for 30 days did not exceed 6 km, which is less than half of the maximum value of STK, so the prediction of Placement_Prediction.py is practically useful. In particular, its prediction errors did not exceed 1km in 3 days, which is more accurate than STK's prediction.

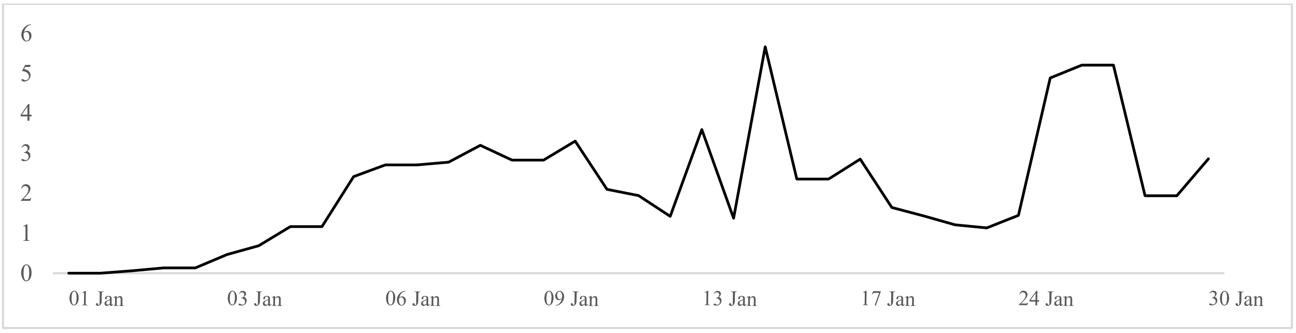


Figure 8 – Errors of MEO Placement in Placement_Prediction.py (unit: km)

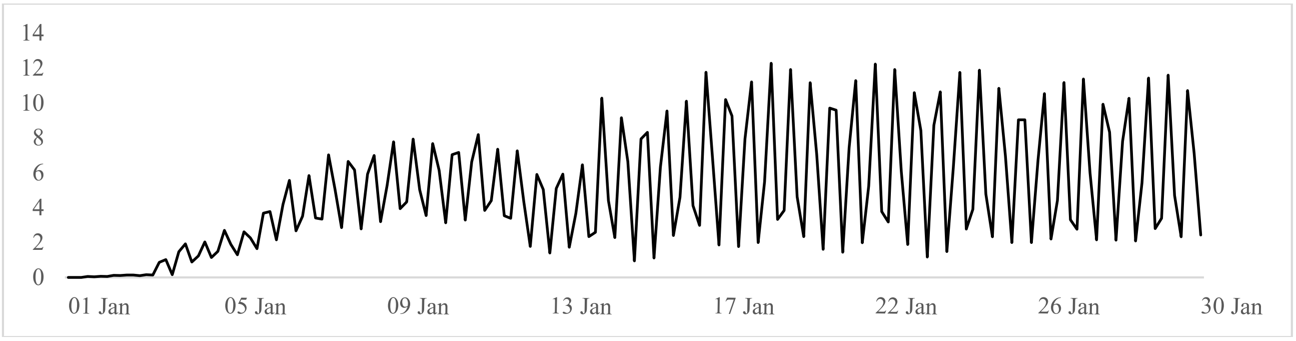


Figure 9 – Errors of MEO Placement in STK (unit: km)

For comparison in accuracy, the prediction error by Placement_Prediction.py's program reached 0.133 km on Jan 02 at 11:51:42, which was calculated according to Equation (9) to obtain an accuracy of 100%; the error of STK prediction was 0.124 km with an accuracy of 100%.

Taking the satellite IPM 2 & BREEZE-M R/B (altitude is 37609km) as an example for HEO satellite placement prediction, from Jan 01 at 03:34:00, the errors of the two prediction methods are shown in Figure 10 and Figure 11.

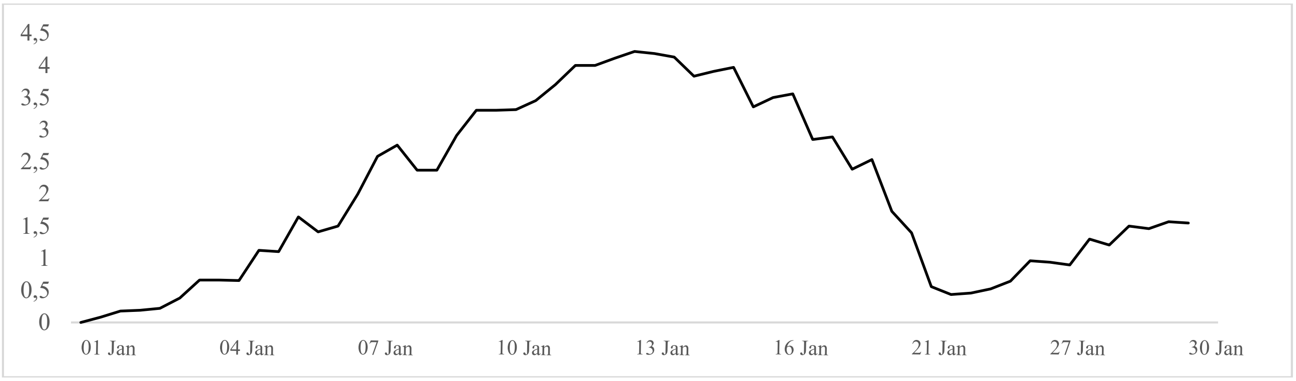


Figure 10 – Errors of HEO Placement in Placement_Prediction.py (unit: km)

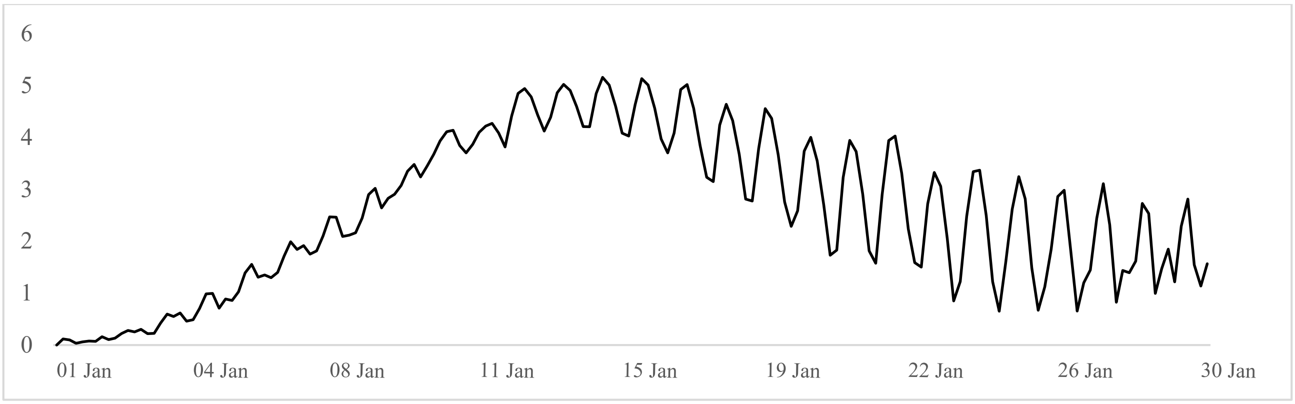


Figure 11 – Errors of HEO Placement in STK (unit: km)

For the prediction of the HEO, it is evident that the errors obtained by the two prediction methods have the same trend. As in the case of the MEO prediction, the errors given by the Placement_Prediction.py program were also smaller than the prediction errors of the STK software. Within 24 hours, the accuracy of both prediction methods reached 100%. The prediction errors were less than 1km in 3 days and even less than 100m in 24h by Placement_Prediction.py, which is highly informative.

Analysis

The overall trend of prediction errors given by Placement_Prediction.py and STK software are very similar in predicting the positions of the three orbiting satellites, with Placement_Prediction.py being significantly smaller and more accurate in predicting the extreme values of errors. The results of both prediction methods are more accurate over a three-day period, especially within 24 hours; the accuracy of the prediction is not less than 99.99%. The errors of both prediction methods are high because satellites in LEO are affected by more regenerative forces. Placement_Prediction.py shows higher superiority in predicting satellite positions in medium and high Earth orbits, with the whole program running in less than 1 second. At the same time, the STK software spends much time in modeling processing.

Conclusion

The errors in position prediction are mainly composed of two parts: the model errors of the prediction algorithm and the errors in the value of orbital parameters caused by the measurement errors, that is, the errors in the obtained TLE data itself [24]. The model error can be analyzed theoretically, and more accurate physical and mechanical models are needed to solve the solution. With the current theoretical and mathematical models, the method used in this work is highly accurate. As a well-developed commercial software, the STK software has a comprehensive algorithm, and the predicted results can be effectively tested for compliance. The final actual accuracy is the result of combining various errors, and the primary purpose of this work is to evaluate the final forecast accuracy.

After inputting the initial TLE data in the program Placement_Prediction.py, the obtained coordinate prediction results have low errors compared with those obtained in STK using the model SGP4 for LEO spacecraft and the model SDP4 for deep-space spacecraft, especially when the spacecraft does not make any orbital maneuvers, and the prediction results of both achieve nearly the same results. The prediction results were compared with the actual data, and their prediction errors were satisfactory, especially the realistic prediction results of Placement_Prediction.py, whose prediction errors were less than one hundred meters in 24h. Moreover, in practical applications, TLE data can be refreshed regularly [25], so it is essential to predict with high accuracy within 24 hours.

STK software is quite powerful but requires a specific computer configuration to guarantee computing capabilities. Otherwise, it will affect the computation or even flash and crash. Programming, in contrast, allows us to do as we need and does not have high requirements for computer hardware. Therefore, using programming to achieve accurate satellite coordinates prediction through TLE data can be widely used as an alternative to the STK software.

Acknowledgment

This research was funded by grant number AP09260469 Development of a control system able to keep the required configuration of spacecraft formation for the performed mission purposes, taking into account the uncertainties caused by external disturbances.

References

1. Carrico, Timothy, John Carrico, Lisa Policastri, and Mike Loucks. "Investigating Orbital Debris Events Using Numerical Methods with Full Force Model Orbit Propagation." *Adv. Astronaut. Sci* 130, no. PART 1 (2008): 407-26.
2. Sampaio, JC, E Wnuk, R Vilhena de Moraes, and SS Fernandes. "Resonant Orbital Dynamics in Leo Region: Space Debris in Focus." *Mathematical Problems in Engineering* 2014 (2014).
3. Vallado, David, Paul Crawford, Ricahrd Hujsak, and TS Kelso. "Revisiting Spacetrack Report# 3." Paper presented at the AIAA/AAS Astrodynamics Specialist Conference and Exhibit, 2006.
4. "Celestrak." https://celestrak.org/.
5. Yong, Dai, Jiang Song, Wang Dayang, Bai Yang, Xu Huichen, and Zhao Jincheng. "A Stk-Based Constellation Architecture Implementation for 5g Low-Orbit Satellites." Paper presented at the 2022 IEEE 4th International Conference on Power, Intelligent Computing and Systems (ICPICS), 2022.
6. Dang, Chang Ping, Yuan Wen Cai, Tian Le, Yi Bin Zhao, and Chao Jun Xin. "Based on Stk and Matlab Satellite Formation's Configuration Design and Simulation." Paper presented at the Advanced Materials Research, 2013.
7. "Ansys Stk Software for Digital Mission Engineering and Systems Analysis." https://www.ansys.com/products/missions/ansys-stk.
8. Jian, Zhao, and Zhao Yan. "The Design and Implementation of Constellation Simulation and Evaluation Software." Paper presented at the Sixth International Symposium on Instrumentation and Control Technology: Signal Analysis, Measurement Theory, Photo-Electronic Technology, and Artificial Intelligence, 2006.
9. Liang, Xiaoheng, Yang Gao, and Shenggang Liu. "A C++ Based Modular Simulation Software of Satellites Earth Observation Mission Planning." Paper presented at the MATEC Web of Conferences, 2020.
10. Yifan, Zhang. "Exploring the Development and Application of Computer Programming Languages." *Wireless Internet Technology* 17, no. 24 (2021): 112-13.
11. Virtanen, Pauli, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski*, et al.* "Scipy 1.0: Fundamental Algorithms for Scientific Computing in Python." *Nature methods* 17, no. 3 (2020): 261-72.
12. Duboshin, G. N. *Nebesnaya Mekhanika: Metody Teorii Dvizheniya Iskusstvennykh Nebesnykh Tel: Uchebnoye Posobiye.* Moscow: Nauka, 1983.
13. Duboshin, G. N. . *Nebesnaya Mekhanika. Osnovnyye Zadachi I Metody.* 2 ed. Moscow: Nauka, 1968.
14. Prussing, John E, and Bruce A Conway. *Orbital Mechanics.* Oxford University Press, USA, 1993.
15. Alipbayev, K, Z Rakisheva, A Sukhenko, and D Ahmedov. "An Analysis of the Perturbing Factors in the Spacecraft Motion Simulation Model." (2010).
16. Zhou, Ying Fu. "A Study for Orbit Representation and Simplified Orbit Determination Methods." Queensland University of Technology, 2003.
17. Yifan, Liu. "Research on Low Earth Orbit Spacecraft Orbit Prediction Strategy Based on Sgp4 Model." Harbin Institute of Technology, 2009.
18. Miura, Nicholas Z. "Comparison and Design of Simplified General Perturbation Models (Sgp4) and Code for Nasa Johnson Space Center, Orbital Debris Program Office." (2009).
19. Curtis, Howard. *Orbital Mechanics for Engineering Students.* Butterworth-Heinemann, 2013.
20. Montenbruck, O., and E. Gill. *Satellite Orbits: Models, Methods, and Applications.* Springer Berlin Heidelberg, 2000. https://books.google.kz/books?id=hABRnDlDkyQC.
21. Rhodes, Brandon, "A Python Implementation of the Sgp4 Model with Automatic Downloading of Tle Elements from Norad Database," https://github.com/brandon-rhodes/python-sgp4.
22. Global Change Master Directory, National Aeronautics and Space Administration. *Ancillary Description Writer's Guide.* 2008.
23. "Space Track." https://www.space-track.org/.
24. Picone, JM, JT Emmert, and JL Lean. "Thermospheric Densities Derived from Spacecraft Orbits: Accurate Processing of Two‐Line Element Sets." *Journal of Geophysical Research: Space Physics* 110, no. A3 (2005).
25. Mukundan, Arvind, and Hsiang-Chen Wang. "Simplified Approach to Detect Satellite Maneuvers Using Tle Data and Simplified Perturbation Model Utilizing Orbital Element Variation." *Applied Sciences* 11, no. 21 (2021): 10181. https://www.mdpi.com/2076-3417/11/21/10181.