

SVA2

(SpaceCraft Visualizaion and Analysis tool)
version 2.7

取扱説明書 付録

スクリプト機能説明書

2020年7月6日(ver2.5版)

2020年10月5日(ver2.5.24版)

2022年10月5日(ver2.7.2版)

2023年11月5日(ver2.7.6版)

有限会社スフィアソフト

改訂履歴

版数	発行日	改訂箇所	改訂内容
2.7.6	2023/11/5	2.2	定数に関わる記述を追加した
2.7.6	2023/11/5	2.2	定数に関わる記述を追加した

1. 概要	1
(1) 衛星制御スクリプト.....	1
(2) シナリオスクリプト.....	1
(3) ユーザ変数定義スクリプト.....	1
2. スクリプト仕様	2
2.1 制御文	2
2.2 変数	3
(1) ローカル変数	3
(2) システム変数	4
(3) ユーザ定義変数.....	6
(4) ベクトル／行列変数の四則演算	6
2.3 算術関数.....	7
2.4 状態取得関数.....	9
(1) Print	10
(2) fileOpen.....	10
(3) fprintf	10
(4) PrintTime.....	10
(5) PrintVec.....	10
(6) movAve	11
(7) makeArray	11
(8) store.....	11
(9) getCount.....	11
(10) average	11
(11) fitting.....	12
(12) Clear	12
(13) getGAST	13
(14) getEventTime	13
(15) angle	13
(16) set	13
(17) time.....	14
(18) wait.....	14
(19) getGPT	14
(20) getTimeInterval.....	14
(21) kepler.....	15
(22) loadTable	15
(23) getValues.....	15
(24) countVisibleGnss	16

2.5	衛星制御関数.....	17
	(1) setEulerAngle.....	18
	(2) setAttByQt.....	18
	(3) setAttByTriad.....	18
	(4) setAttByTriad2.....	19
	(5) changeAttByTriad.....	19
	(6) changeAttByTriad2.....	20
	(7) checkMnvDone.....	21
	(8) changeAttByQt.....	22
	(9) rotateParts.....	22
	(10) changeOrbit.....	22
	(11) thrusterBurn.....	23
	(12) setSfKp.....	23
	(13) setSensorAngle.....	23
	(14) makeAttitude.....	23
	(15) setTorq.....	24
	(16) readAttFile.....	24
	(17) setTrackObject.....	24
	(18) getAtt.....	25
	(19) setGimbalAngle.....	25
3.	スクリプト編集機能.....	27
3.1	衛星制御スクリプト.....	27
3.2	シナリオスクリプト.....	29
3.3	ユーザ変数定義スクリプト.....	29
4.	スクリプトのデバッグ.....	30
5.	シミュレーション中のスクリプト実行タイミング.....	31

1. 概要

SVAには、以下のスクリプト制御機能があります。

(1) 衛星制御スクリプト

衛星制御スクリプトは、衛星の ΔV 制御や姿勢制御のロジックを利用者が記述するためのものであり、一般の演算式や関数の他、衛星制御用の各種関数や制御命令などが用意されています。

衛星制御スクリプトは、シミュレーションの各ステップにおいて、軌道位置・速度、姿勢などが更新された後、スクリプトの先頭から最後までが実行され、現在までの状態を基に、次のシミュレーションステップにおける各種制御量を算出することができます。

また、定義済みの衛星制御スクリプトは、衛星制御スクリプトファイルとして保存することができます、他のシミュレーションで流用することができます。

(2) シナリオスクリプト

シミュレーション中の指定時刻におけるアクションを記述することにより、シミュレーション全体の制御を記述するものです。

衛星制御スクリプトでも同様の記述が可能ですが、制御計画が予め決められている場合に、より簡単にシナリオを記述することができます。

また、定義済みのシナリオスクリプトはシミュレーション定義ファイルに組込まれて、保存されます。

(3) ユーザ変数定義スクリプト

ユーザ変数の値を算出するための簡易スクリプトです。

シミュレーション定義ファイルに組込まれて、保存されます。

2. スクリプト仕様

スクリプトの仕様に関する、以下の項目について記述します。

- ・制御文
- ・変数
- ・関数
- ・制御命令
- ・その他

2.1 制御文

スクリプト処理を記述するために使用可能な文を以下に示します。

表 2.1-1 制御文一覧

種別	内容	記述形式/例
init 文	シミュレーション開始後、1 度だけ実行される制御文を定義する。ローカル変数の初期化や姿勢の初期設定に用いる。	init { 代入文; }
if 文	指定した条件式が真の場合と偽の場合の制御文を記述する。 ・ elseif 文は複数定義可能。また省略可能。 ・ else 文は省略可能	if (条件式) { 真の場合の文; } elseif (条件式 2) { 条件式 2 が真の場合の文; } else { 偽の場合の文; }
for 文	繰り返し制御を記述する。	for(i=0;i<100;i=i+1) { 繰り返し実行される文; }
while 文	繰り返し制御を記述する。 繰り返し条件が成立する間、繰り返される。	while(条件式) { 繰り返し実行される文; }
stop 文	シミュレーションを終了する。	
continue 文	繰り返し処理を途中からスキップする。	
break 文	繰り返し処理を中断する。	
代入文	ローカルな変数に演算式の結果を代入する。 文の最後はセミコロン“;”で終了する。	a = b + c*sin(theta);
制御命令 呼出し文	各種制御命令を呼び出す。 文の最後はセミコロン“;”で終了する。	changeAttRate(sc, e1, e2, e3, rate);
演算式	四則演算 (+、-、*、/) と、剰余 (%) を行う。 演算式では各種関数を参照することができる。 演算子の優先度は、一般の処理言語 (C、Fortran) と同様。	
条件式	比較式と論理演算子を組み合わせ、条件式を記述する。 以下の比較演算子と論理演算子が使える。 ・ >、<、<=、>=、== ・ && (AND) ・ (OR)	(a==1 && b==0) (c<=0)
コメント	文字#以降はコメントとなる、。	# コメント 文字列

2.2 定数

スクリプト中に以下の定数を記述することができます。

表 2.2-1 定数一覧

定数名	内容
const.C	光速 (299792.458)[km/s]
const.PI	π (3.14159265358979323846)
const.E	e (2.71828182845905)
const.Mu	地心重力定数 μ (3.986004415e5) [km ³ /s ²]
const.Re	地球赤道半径(6378.1363)[km]
const.Rp	地球極半径 (6356.752314245)[km]
const.DEGRAD	度→Radian 変換係数
const.RADDEG	Radian→度変換係数

※ 網掛け部分は、次バージョンで対応の予定。

2.3 変数

スクリプト内では、変数として、ローカル変数、システム変数、およびユーザ定義変数を使用することができます。

また、単純変数の他、2次元までの配列を扱うことができる。

(1) ローカル変数

以下、ローカル変数の特性を示します。

- ・ ローカル変数は宣言なしに使うことができる。
- ・ 値の設定なしに参照することはできない。
- ・ 変数は2次元までの配列の要素を持つことができ、ベクトルや行列を表現することができる。
- ・ 配列要素を参照することができる。配列要素のインデックスは0始まりとする。

以下、記述例を示す。

```
a = b + c;
v1 = vector(a, b, c);
v2 = vector(0, 0, 1);
v = v1 * v2;
d = v[0] + v[1] + v[2];
mat = zeroMat(3, 3);
mat[2, 2] = 1.0;
```

(2) システム変数

システム変数は、衛星位置や姿勢などの、内部データをスクリプト内で参照するために用意されており、以下のような特性を持っています。

(a) システム変数の参照形式

以下の形式で参照する。

オブジェクト名. データ名. 座標系

(i) オブジェクト名

衛星名、天体名、センサ名、地上局名、および、**Earth** を指定することができる。時刻などの普遍的なデータは、**Earth** を用いて参照する。

また、衛星制御スクリプトは衛星毎に記述するが、対象となる衛星は、「**\$SC**」として参照することができる。(これにより、記述したスクリプトを他の衛星にも流用できるようになる)

例) **\$SC.Px** : 当該スクリプトで制御対象とする衛星の慣性系位置 X 成分。

(ii) データ名

各オブジェクト毎に使用可能なデータ名については、SVA 取扱説明書の「付録1 システム変数一覧」を参照のこと。

(iii) 座標系

座標系は、原点オブジェクト@座標フレーム
フレームとして、以下を指定できる。

座標フレーム	説明	原点オブジェクト
J2K	J2000 赤道座標系	天体
TOD	TOD 座標系	天体
BF	天体の場合は、天体固定座標系。衛星の場合は機体固定系	天体、衛星
EC	J2000 黄道座標系	天体
VF	視野座標系 (視線方向が Z 軸)	センサ
無指定	地上局の場合、ENU となる	地上局

(b) システム変数の特性

以下にシステム変数の特性を示します。

- ・ 値は、全て倍精度実数と扱われる。
- ・ スクリプト内で値を変更することはできない。(式の左辺に用いることはできない)
- ・ 位置や速度などのベクトル変数が用意されており、ベクトル演算に用いることができる。

(c) 参照例

以下に参照例を示す。

SAT1.Px	慣性座標系における、SAT1 衛星の位置(X)
SAT1.Px.Earth@BF	地球固定座標系における SAT1 衛星の位置(X)
SAT1.Px.Tokyo	地上局 Tokyo のローカル座標系における SAT1 衛星の位置(X)
SAT1.Pos	慣性座標系における、SAT1 衛星の位置ベクトル
SAT1.Pos.Earth@BF[2]	地球固定座標系における SAT1 衛星の位置(Z 成分)
\$SC.Vel	制御対象衛星の速度ベクトル (衛星制御スクリプトでのみ、使用可)
Earth.GHA	グリニッジ時角

(3) ユーザ定義変数

ユーザ定義変数は、ユーザ変数定義機能で登録された変数である。

通常は、簡単なスクリプトで定義し、表示等に用いることを目的としているが、衛星制御スクリプトで算出されたデータを、外部出力（データ一覧やプロット表示）を可能とするため、ユーザ定義変数を利用することができる。

つまり、あらかじめ登録したユーザ定義変数に、算出結果を代入することにより、各種画面表示で参照することが可能である。

(a) 参照形式

ユーザ定義変数の参照形式は、以下のようである。

```
a = b + uvar.変数名 ;
uvar.変数名 = 演算式 ;
```

変数名は、「ユーザ変数定義機能」で登録した名称を指定する。

(b) ユーザ定義変数の特性

- ・ 値は、全て倍精度実数として参照する。
- ・ スクリプト内で値を変更することができる。
ただし、ユーザ変数定義において、変数のスクリプトを空欄にすることが必要である。
- ・ ベクトルや配列は使用できない。

(4) ベクトル／行列変数の四則演算

ベクトル／行列変数の四則演算について、2つの項の組合せに対する結果を以下に示す。

演算子	項 1 (a)	項 2 (b)	結果(c)
c = a + b	V	V	V
	V	S	V
	S	V	V
c = a - b	V	V	V
	V	S	V
	S	V	V
c = a * b	V	V	V または S 項 1 の列数と項 2 の行数が等しい場合のみ有効
	V	S	V
	S	V	V
c = a / b	V	V	不可
	V	S	V
	S	V	不可

S : スカラ V : ベクトル／行列

なお、剰余(%)にベクトル変数を適用することはできない。

2.4 算術関数

使用可能な算術関数を以下の表に示します。

表 2.4-1 算術関数一覧

名称	書式	説明
Sin	$x = \sin(\theta)$	角度 $\theta(\text{rad})$ の sine を返す。
Cos	$x = \cos(\theta)$	角度 $\theta(\text{rad})$ の cosine を返す。
Tan	$x = \tan(\theta)$	角度 $\theta(\text{rad})$ の tangent を返す。
Asin	$\theta = \text{asin}(x)$	指定値 x の arc-sine(rad) を返す。
Acos	$\theta = \text{acos}(x)$	指定値 x の arc-cosine(rad) を返す。
Atan	$\theta = \text{atan}(x)$	指定値 x の arc-tangent(rad) を、 $-\pi/2 \sim \pi/2$ で返す。
atan2	$\theta = \text{atan2}(y, x)$	指定値 y/x の arc-tangent(rad) を、 $-\pi \sim \pi$ で返す。
dsin	$x = \text{dsin}(\theta)$	角度 $\theta(\text{deg})$ の sine を返す。
dcos	$x = \text{dcos}(\theta)$	角度 $\theta(\text{deg})$ の cosine を返す。
dtan	$x = \text{dtan}(\theta)$	角度 $\theta(\text{deg})$ の tangent を返す。
dasin	$\theta = \text{dasin}(x)$	指定値 x の arc-sine(deg) を返す。
dacos	$\theta = \text{dacos}(x)$	指定値 x の arc-cosine(deg) を返す。
datan	$\theta = \text{datan}(x)$	指定値 x の arc-tangent(deg) を、 $-90 \sim 90$ で返す。
datan2	$\theta = \text{datan2}(y, x)$	指定値 y/x の arc-tangent(deg) を、 $-180 \sim 180$ で返す。
Log	$y = \log(x)$	x の自然対数を返す。
log10	$y = \log_{10}(x)$	x の常用対数を返す。
Exp	$y = \exp(x)$	e を底とする x のべき乗を返します。
Pow	$y = \text{pow}(x, p)$	x の p 乗を返す。
Abs	$y = \text{abs}(x)$	x の絶対値を返す。
Sqrt	$y = \text{sqrt}(x)$	x の平方根を返す。
Sq	$y = \text{sq}(x)$	x の 2 乗を返す。
Deg	$y = \text{deg}(x)$	x を radian から degree に変換する。
Rad	$y = \text{rad}(x)$	x を degree から radian に変換する。
Rand	$y = \text{rand}(x)$	$0 \sim x$ の範囲の一樣乱数を返す。c は定数。
Randn	$y = \text{randn}(\text{ave}, \text{sigma})$	平均値 ave 、分散 sigma の正規乱数を返す。
Reduce	$y = \text{reduce}(x, \text{min}, \text{max})$	指定値 x を、 $\text{min} \sim \text{max}$ の範囲に正規化する。
min	$v = \text{min}(v1, v2)$	$v1$ と $v2$ の小さい方を返します。
max	$v = \text{max}(v1, v2)$	$v1$ と $v2$ の大きい方を返します。

※ setTrigFuncUnit 関数の設定により、degree/radian を切替えることができる。
setTrigFuncUnit("DEG")を呼び出すことで、以降の三角関数が degree 対応となる。
また、setTrigFuncUnit("RAD")で、radian に戻る。

※ 関数名は、大文字と小文字を区別しない。

表 2.4-2 ベクトル／行列演算関数一覧

名称	書式	説明
vector	v = vector(x1,x2,x3) v = vector(x1,x2,x3,x4) v = vector(x1,x2,x3,x4,x5) v = vector(x1,x2,x3,x4,x5,x6)	成分が、x1～x6 の行ベクトルを返す。
rowVec	v = rowVec(x1,x2,x3) v = rowVec(x1,x2,x3,x4) v = rowVec(x1,x2,x3,x4,x5) v = rowVec(x1,x2,x3,x4,x5,x6)	成分が、x1～x6 の列ベクトルを返す。
dot	x = dot(v1,v2)	ベクトル v1 と v2 の内積を返す。 ベクトルは、列／行に関わらず、要素数が等しければ演算可能。
cross	v = cross(v1,v2)	ベクトル v1 と v2 の外積を、行ベクトルとして返す。 v1、v2 の要素数は 3 に限定される。
norm	x = norm(v)	ベクトル v のノルムを返す。
eyeMat	mat = eyeMat(n, m)	n 行、m 列の単位行列を返す。
zeroMat	mat = zeroMat(n, m)	n 行、m 列の 0 行列を返す。
rotMat	mat = rotMat(code, a, b, c)	回転順序 (123,213 等) と、回転角(a,b,c) [deg]を指定し、回転行列を作成する。
transpose	mat2 = transpose(mat1)	指定行列／ベクトルを転置する。
inverse	mat2 = inverse(mat1)	逆行列を作成する。
solveEq	x = solveEq(A, b)	[A]{x}={b}を解く。
mat2q	q = mat2q(mat)	座標変換行列をクォータニオンに変換する。
q2mat	mat = q2mat(q)	クォータニオンを座標変換行列に変換する。
mulQ	q3 = mulQ(q1, q2)	q3 = q1 * q2 を算出する。(※)
invQ	q = invQ(q)	q の逆クォータニオンを返す。(※)
getEulerAng	getEulerAng(mat, code, a, b, c) getEulerAng(q, code, a, b, c)	指定した行列、またはクォータニオンから、指定回転順序のオイラー角[deg]を求める。 回転順序は、123,213 などのコードで指定する。
unitVec	v1 = unitVec(v2);	ベクトル v2 を単位ベクトル化する。
getRotAxis	v = getRotAxis(q);	クォータニオン q に対する回転軸ベクトルを得る。
getRotAngle	angle = getRotAngle(q);	クォータニオン q に対する回転角[度]を得る。

※ クォータニオンの積と逆クォータニオンの定義

$$\text{積定義: } q = \begin{bmatrix} \mathbf{q} \\ q_4 \end{bmatrix}, q' = \begin{bmatrix} \mathbf{q}' \\ q'_4 \end{bmatrix} \text{ として、 } q \cdot q' = \begin{bmatrix} q'_4 \mathbf{q} + q_4 \mathbf{q}' + \mathbf{q} \times \mathbf{q}' \\ q_4 q'_4 - \mathbf{q} \cdot \mathbf{q}' \end{bmatrix}$$

$$\text{具体的に書き下ろすと、} \quad q \cdot q' = \begin{bmatrix} q'_4 q_1 + q_4 q'_1 + q_2 q'_3 - q_3 q'_2 \\ q'_4 q_2 + q_4 q'_2 + q_3 q'_1 - q_1 q'_3 \\ q'_4 q_3 + q_4 q'_3 + q_1 q'_2 - q_2 q'_1 \\ q_4 q'_4 - q_1 q'_1 - q_2 q'_2 - q_3 q'_3 \end{bmatrix}$$

$$\text{また、逆クォータニオンは、} \quad q^{-1} = \begin{bmatrix} -\mathbf{q} \\ q_4 \end{bmatrix}$$

2.5 状態取得関数

状態取得用関数の一覧を以下に示します。

表 2.5-1 状態取得用関数一覧

No.	関数名	概要
1	print	ログ画面に変数値を表示する。
2	printTime	ログ画面に時刻変数を表示する。
3	printVec	ログ画面にベクトル変数を表示する。
4	movAve	変数の移動平均値を返す。
5	makeArray	n 個の要素を持つ配列を作成する。
6	store	makeArray で作成した配列に変数値を格納する。
7	getCount	配列に格納されているデータ数を返す。
8	average	配列に格納されているデータについて平均値を取得する。
9	fitting	配列に格納されているデータについて n 次関数による最小二乗 fitting を行う。
10	clear	配列に格納されているデータをクリアする。
11	getGAST	指定時刻のグリニッジ時角を取得する。
12	getEventTime	指定衛星について、前回と今回のシミュレーションステップ間で、発生したイベントの時刻を返す。
13	findEventTime	現在時刻を基点に、次または前回の軌道イベント時刻を求める。
14	getCartesian	指定時刻の衛星位置・速度を取得する。
15	getKeplerian	指定時刻の衛星ケプラー要素を取得する。
16	angle	object1 から見た、object2 と object3 のなす角を求める。
17	set	シミュレーション経過時間を返す。
18	time	現在のシミュレーション時刻を返す。
19	wait	指定時刻まで待ち合わせる
20	getGPT	ベクトルと地球楕円体の交点位置を求める。
21	getTimeInterval	シミュレーション時間間隔を取得する。
22	Kepler	ケプラー方程式を解く。f = Kepler (M, e);
23	loadTable	csv 形式のファイルをテーブルとして読み込む
24	getValues	loadTable で読み込んだテーブルを検索し、テーブルの項目値を取得する。
25	countVisibleGnss	姿勢センサ (アンテナ) に対して、可視となる GNSS 衛星数を返す。
26	time2str	時刻変数 (UnixTime 形式) を文字列に変換する。 print 文での出力書式には %s を用いる。

(1) Print

説明	ログ画面に変数値を表示する。	
形式	print(format,x1,x2,...,xN);	
パラメータ	format	フォーマット文字列 C 言語の書式指定子を使用可能。 オブジェクトの NAME 変数については、%s を指定する。
	x1~xN	出力対象変数。0 個以上の任意の数の変数を指定可能。
例	print("a=%g b=%g",a,b);	

(2) fileOpen

説明	出力ファイルをオープンする。	
形式	n = fileOpen(filepath);	
パラメータ	filepath	出力ファイルパス名を文字列で指定する。 n = fileOpen("c:¥temp.txt");
戻り値	ファイル識別番号。fprintf で指定する。	

(3) fprintf

説明	指定したファイルに変数値を出力する。	
形式	fprintf(n,format,x1,x2,...,xN);	
パラメータ	n	fileOpen でオープンされたファイル識別番号。
	format	フォーマット文字列 C 言語の書式指定子を使用可能。 オブジェクトの NAME 変数については、%s を指定する。
	x1~xN	出力対象変数。0 個以上の任意の数の変数を指定可能。
例	fprintf(n, "a=%g b=%g",a,b);	

(4) PrintTime

説明	ログ画面に時刻変数値を表示する。	
形式	printTime(label,t)	
パラメータ	Label	データ値の前に付与するラベル文字列
	t	変数 t を時刻と見做して、 yyyy/mm/dd hh:mm:ss 形式で表示する。
例	printTime("昇交点通過時刻",t);	

(5) PrintVec

説明	ログ画面にベクトル変数（配列）の全要素を表示する。	
形式	printVec(label,vec)	
パラメータ	Label	データ値の前に付与するラベル文字列
	vec	変数 vec のすべての要素を表示する。
例	printVec("位置ベクトル", \$SC.Pos);	

(6) movAve

説明	変数の移動平均値を返す。 n 個分の変数のバッファに、変数値を格納し、最新の n 個のデータによる平均値を算出する。	
形式	y = movAve(x,n)	
パラメータ	x	平均値をとる変数
	n	データ点数
戻り値	移動平均値	

(7) makeArray

説明	n 個の要素を持つ配列を作成する。 (n 個の成分を持つベクトル変数と等価)	
形式	makeArray(array,n)	
パラメータ	array	配列変数
	n	最大要素数
戻り値	なし	

(8) store

説明	makeArray で作成した配列に変数値を格納する。 要素数を超えた場合は、捨てられる。	
形式	store(array,x)	
パラメータ	array	配列変数
	n	最大要素数
戻り値	なし	

(9) getCount

説明	makeArray で作成した配列の格納データ数を返す	
形式	n = getCount(array)	
パラメータ	array	配列変数
戻り値	なし	

(10) average

説明	配列に格納されたデータの平均値を返す	
形式	ave = average(array)	
パラメータ	array	配列変数
戻り値	平均値	

(11) fitting

説明	配列に格納されているデータについて n 次関数による最小二乗 fitting を行う。 ただし、 $n \leq 3$ とする。	
形式	error = fitting(x,y,n,a)	
パラメータ	x	x (配列変数)
	y	y (配列変数)
	n	次数
	a	係数 (配列変数)
戻り値	フィッティング残差	

(12) Clear

説明	配列に格納されているデータを消去する。	
形式	clear(array)	
パラメータ	array	配列変数
戻り値	なし	

(13) getGAST

説明	指定時刻のグリニッジ時角を取得する。	
形式	gast = getGAST(t);	
パラメータ	t	時刻
戻り値	グリニッジ時刻(deg) 0~360。	
備考	現在時刻のグリニッジ時角は、システム変数を参照することができる。	
処理概要		

(14) getEventTime

説明	指定衛星について、前回と今回のシミュレーションステップ間で、指定した軌道イベントの発生有無を判定し、発生した場合にその時刻を返す。 なお、特別摂動法でなければ、findEventTime が使える。	
形式	t = getEventTime(sc, evType);	
パラメータ	sc	衛星オブジェクト名
	eveType	イベント種別を表す、以下の文字列 “PASS_AN”： 昇交点通過イベント “PASS_DN”： 降交点通過イベント
戻り値	イベント時刻 (1970/1/1 00:00UTC からの通算秒)	
処理概要	イベント発生時刻は、前回と今回の緯度引数と時刻から線形補間で求める。	

(15) angle

説明	object1 から見た、object2 と object3 のなす角を求める。	
形式	ang = angle(object1,object2,object3);	
パラメータ	object1	object1 の名称
	object2	object2 の名称
	object3	object3 の名称
戻り値	角度[deg]	

(16) set

説明	シミュレーション経過時間を返す。	
形式	t = set()	
パラメータ	なし	
戻り値	シミュレーション開始時からの経過時間[s]	

(17) time

説明	現在のシミュレーション時刻を返す。 または、指定された日付文字列(UTC)に対応する時刻を返す。	
形式	t = time() t = time(時刻文字列);	
パラメータ	時刻文字列	“YYYY/MM/DD hh:mm:ss.ttt”形式で、時刻を指定する。 (ttt は秒の小数部で、精度は最大 6 桁まで有効)
戻り値	シミュレーション時刻 (1970/1/1 00:00UTC からの経過秒)	

(18) wait

説明	指定時刻まで待ち合わせる。	
形式	wait(時刻文字列)	
パラメータ	時刻文字列	“YYYY/MM/DD hh:mm:ss”形式で、時刻を指定する。
戻り値	なし。	
備考	本関数は、シナリオスクリプトでのみ使用可能である。	

(19) getGPT

説明	指定ベクトルと地球楕円体の交点位置を求める。	
形式	getGPT(pos, dir, lat, lon)	
パラメータ	pos	ベクトルの始点位置 (ベクトル変数)
	dir	方向ベクトル (ベクトル変数)
	lat	指定ベクトルと地球楕円体の交点緯度(rad)
	lon	指定ベクトルと地球楕円体の交点経度(rad)
戻り値	0 : 正常、-1 : 交点無し	

(20) getTimeInterval

説明	シミュレーション時間間隔を取得する。	
形式	intervalSec = getTimeInterval();	
パラメータ	なし	
戻り値	シミュレーション時間間隔(秒)	

(21) kepler

説明	ケプラー方程式を解き、真近点離角を求める	
形式	ta = kepler(ma,e)	
パラメータ	ma	平均近点離角[rad]
	e	離心率
戻り値	真近点離角[rad]	

(22) loadTable

説明	CSV 形式のファイルをテーブルとして読み込む CSV 形式のファイルの内容は、数値または時刻形式とする。 時刻形式は先頭の列にのみ記述可能である。 また、#文字を 1 カラム目に置くことでコメント行とみなされる。	
形式	nLine = loadTable(ファイル名、keyType、table)	
パラメータ	ファイル名	ファイルパス名または、ファイル名をダブルクォーテーションで括弧付した文字列で指定する。 当該ファイルをスクリプトファイルと同じフォルダに置いた場合に限り、ファイル名だけの指定が可能である。
	keyType	以下の定数で検索する際のキー種別を指定する。 0 : 数値 1 : 時刻(YYYY/MM/DD hh:mm:ss[.ssss]) ¹ 2 : 時刻(hh:mm:ss[.ssss]) ※ 時刻データは、内部形式(数値)に変換される。 3 : index (getValues ではレコードインデックスを指定する)
	table	読込先のテーブル名
戻り値	読み込んだライン数 (-1 のときは異常)	

(23) getValues

説明	loadTable で読み込んだテーブルから、指定キーのレコードを検索し、レコード内の項目を取得する。	
形式	status = getValues(table, key, type, v1, v2, ..., vN)	
パラメータ	table	テーブル変数名
	key	キー値を指定する。
	type	0 : key が完全一致するレコードを取得 1 : key が $v[i-1] \leq key < v[i]$ のとき、i-1 番目のレコードを取得 2 : key が $v[i-1] \leq key < v[i]$ のとき、 i-1 番目と i 番目のレコード中の各項目について線形補間より、key に対応する値を取得する。
	v1 ... vN	テーブルの 2 列目以降の値が、v1..vN の並び順に格納される。
戻り値	合致したレコードインデックス -1=合致レコードなし	

¹ [] 内の記述は省略可能であることを表す。

(24) countVisibleGnss

説明	衛星センサ（アンテナ）または地上アンテナから、可視となる GNSS（GPS 衛星または QZS）衛星数を求める。 可視となる条件は、指定した衛星センサ（アンテナ）の FOV 内に見えることである。 また、SV ヘルスが BAD の GNSS 衛星は除く。	
形式	count = countVisibleGnss(sensorObject, type)	
パラメータ	sensorObject	衛星センサ（アンテナ名）または、地上局アンテナを指定する。
	type	カウントする GNSS 種別を指定する。 0 : GPS 衛星 1 : GPS 衛星 + QZS
戻り値	可視 GNSS 衛星数	

2.6 衛星制御関数

以下では、衛星の軌道、姿勢、パドルなどの制御を行うための関数について示します。

表 2.5-1 衛星制御関数一覧

No.	名称	機能
1	setEulerAngle	基準座標系に対するオイラー角により、衛星姿勢を設定する。
2	setAttByQt	クォータニオンにより、衛星姿勢を設定する。
3	setAttByTriad	衛星姿勢を、Triad 法により設定する。 衛星機体の基準方向ベクトルは、±X/Y/Z 軸方向のいずれかを指定する。
4	setAttByTriad2	衛星姿勢を、Triad 法により設定する。 衛星機体の基準方向ベクトルを任意ベクトルで指定する。
5	setObjectAtt	機体座標系に対するセンサーの回転角を設定する。
6	changeAttByTriad	衛星姿勢を、Triad で指定された姿勢に回転する。 角加速度と最大角速度、およびマヌーバ角から作成した角速度プロファイルに従って、姿勢回転を模擬する。
7	changeAttByTriad2	同上。Triad の設定方法が異なる。
8	checkMnvDone	ChangeAtt*関数による制御が完了したか否かを返す。
9	changeAttByQt	衛星姿勢を、Quaternion 指定された姿勢に回転する。 角加速度と最大角速度、およびマヌーバ角から作成した角速度プロファイルに従って、姿勢回転を模擬する。
10	rotateParts	形状モデルのパーツに対して、回転角を設定する。
11	changeOrbit	ΔV を与えて、瞬時に軌道要素(速度成分)を変更する。
12	thrusterBurn	ΔV と噴射時間を与えて軌道要素変更を行う。
13	setSfKp	大気密度モデルパラメータとして、ソーラーフラックスと地磁気指標値を設定する。
14	setSensorAngle	センサーのジンバル角を設定する。
15	makeAttitude	Triad 法により、姿勢パラメータ(クォータニオン)を求める。
16	setTorq	衛星にトルクを与える。 ダイナミクスモードが ON の場合のみ有効。
17	setTrackObject	衛星アンテナ(センサ)の追尾オブジェクトを設定し、駆動を開始する。 また、追尾オブジェクトをクリアし、駆動を停止する。
18	readAttFile	姿勢プロファイルから姿勢データを読み込む
19	getAtt	姿勢プロファイルから読み込んだ姿勢データを時刻指定で取得する。
20	setGimbalAngle	センサ/アンテナのジンバル角を駆動レートとともに設定する。
21	makeMnvPlanIP	面内軌道制御計画を作成する。
22	makeMnvPlanOP	面外軌道制御計画を作成する。

(1) setEulerAngle

説明	基準座標系に対するオイラー角により、衛星姿勢を設定する。	
形式	setEulerAngle(scName,rotOrder,ang1,ang2,ang3)	
パラメータ	scName	衛星名
	rotOrder	回転順序コード(123,213,etc..)
	ang1-ang3	回転角[deg]

(2) setAttByQt

説明	クォータニオンにより、衛星姿勢を設定する。	
形式	setAttByQt(scName,q1,q2,q3,q4) setAttByQt(scName,q);	
パラメータ	scName	衛星名
	q1-q4	姿勢クォータニオン
	q	姿勢クォータニオン配列

(3) setAttByTriad

説明	衛星姿勢を、Triad 法により設定する。	
形式 1	setAttByTriad(scName,axis1,object1,axis2,object2)	
形式 2	setAttByTriad(scName,axis1,vec1,axis2,vec2)	
パラメータ	scName	衛星名
	axis1	指向方向の機体軸番号(±1 ~ ±3)
	object1	指向天体または、\$VEL(速度方向)、\$OBN(軌道面垂直方向)。
	vec1	指向方向のベクトル。(単位ベクトルでなくても可)
	axis2	第2ベクトル方向の機体軸番号(±1 ~ ±3)
	object2	第2ベクトル方向の天体または、\$VEL(速度方向)、\$OBN(軌道面垂直方向)。
vec2	第2ベクトル方向のベクトル(単位ベクトルでなくても可)	
備考	例) setAttByTriad(SC1, 3, Earth, 2, \$VEL);	

(4) setAttByTriad2

説明	衛星姿勢を、Triad 法により設定する。	
形式 1	setAttByTriad2(scName,axis1,object1,axis2,object2)	
形式 2	setAttByTriad2(scName,axis1,vec1,axis2,vec2)	
パラメータ	scName	衛星名
	axis1	指向方向とする衛星機体方向ベクトル (単位ベクトル)
	vec1	指向方向のベクトル。(単位ベクトルでなくても可)
	axis2	第2ベクトル方向とする衛星機体上の方向ベクトル(単位ベクトル)
	vec2	第2ベクトル方向のベクトル (単位ベクトルでなくても可)

(5) changeAttByTriad

説明	衛星姿勢を、Triad 法で指定された姿勢に回転する。 角加速度と最大角速度、およびマヌーバ角から作成した角速度プロファイルに従って、姿勢回転を模擬する。	
形式	changeAttByTriad(scName,axisNo1,object1,axisNo2,object2,acc,wMax) changeAttByTriad(scName,axisNo1,vec1,axisNo2, vec2, acc,wMax)	
パラメータ	scName	衛星名
	axisNo1	指向方向の機体軸番号(±1 ~ ±3)
	object1	指向天体または、\$VEL (速度方向)、\$OBN(軌道面垂直方向)。
	axisNo2	第2ベクトル方向の機体軸番号(±1 ~ ±3)
	object2	第2ベクトル方向の天体または、\$VEL (速度方向)、\$OBN(軌道面垂直方向)。
	vec1	指向方向のベクトル。(単位ベクトルでなくても可)
	vec2	第2ベクトル方向のベクトル (単位ベクトルでなくても可)
	acc	角加速度[deg/s ²]
wMax	最大角速度[deg/s]	
戻り値	姿勢変更所要時間[sec]	
備考	第1ベクトル方向と第2ベクトル方向として、object を指定した場合は、姿勢変更の間に衛星位置・速度や指向天体位置が変化することを考慮する。	

(6) changeAttByTriad2

説明	衛星姿勢を、Triad法で指定された姿勢に回転する。 角加速度と最大角速度、およびマヌーバ角から作成した角速度プロファイルに従って、姿勢回転を模擬する。	
形式	changeAttByTriad2(scName,axis1,object1,axis2,object2,acc,wMax) changeAttByTriad2(scName,axis1,vec1,axis2,vec2,acc,wMax)	
パラメータ	scName	衛星名
	axis1	指向方向とする衛星機体方向ベクトル(単位ベクトル)
	object1	指向天体または、\$VEL(速度方向)、\$OBN(軌道面垂直方向)。
	axis2	第2ベクトル方向とする衛星機体上の方向ベクトル(単位ベクトル)
	object2	第2ベクトル方向の天体または、\$VEL(速度方向)、\$OBN(軌道面垂直方向)。
	vec1	指向方向のベクトル。(単位ベクトルでなくても可)
	vec2	第2ベクトル方向のベクトル(単位ベクトルでなくても可)
	acc	角加速度[deg/s ²]
	wMax	最大角速度[deg/s]
戻り値	姿勢変更所要時間[s]	
備考	第1ベクトル方向と第2ベクトル方向として、objectを指定した場合は、姿勢変更の間に衛星位置・速度や指向天体位置が変化することを考慮する。	

※補足事項

(a) 入力条件

q_0	現在姿勢(本関数呼び出し時の姿勢)
q_g	目標姿勢(Triadで設定)
a	角加速度[deg/s ²]
ω_{max}	最大角速度[deg/s]

(b) 処理内容

姿勢を $q_0 \rightarrow q_g$ に変更するためのクォータニオンを、 dq としたとき、

$$dq = q_0^{-1} \cdot q_g \text{ と表せる。}$$

dq から、回転軸ベクトルとマヌーバ回転角 θ を求め、 a と ω_{max} から角速度プロファイル(図 2.6-1)を作成し、これを用いて、マヌーバ中の任意時刻(t_x)における回転角

θ_x を算出し、姿勢を決定する。

また、角速度プロファイルの作成時に、Triadの指向方向が天体や衛星速度ベクトル方向の場合は、マヌーバ中の衛星軌道位置の変化を考慮する。

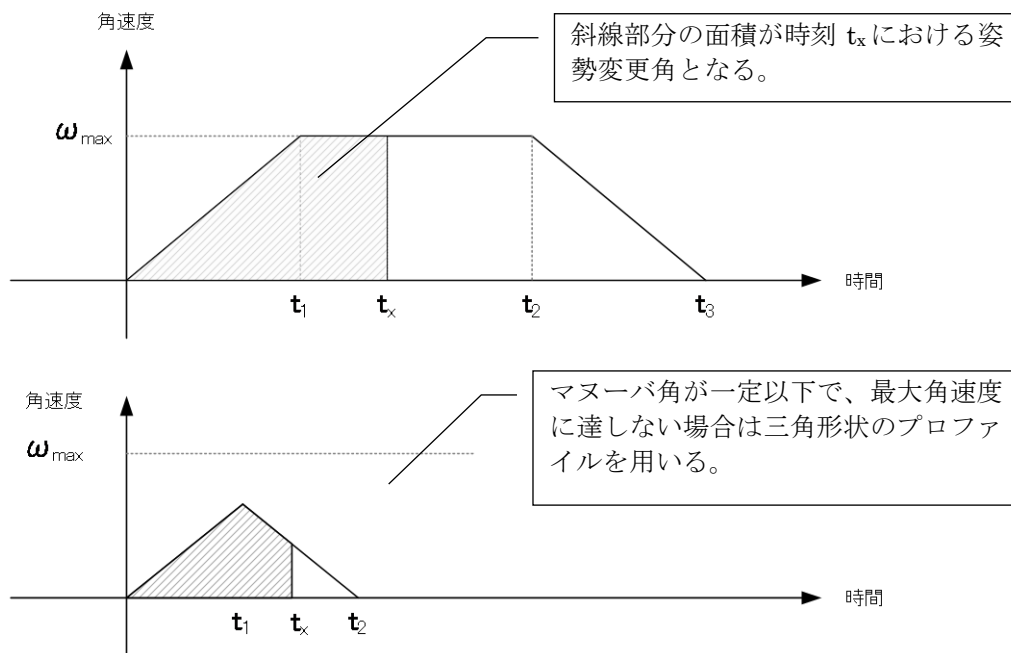


図 2.6-1 角速度プロファイル

(7) checkMnvDone

説明	changeAttByTriad/changeAttByTriad2 で開始した姿勢マヌーバが完了したか否かをチェックする。
形式	checkMnvDone(scName);
パラメータ	scName 衛星名
戻り値	0 = マヌーバ中 1 = マヌーバ完了 changeAttByTriad/changeAttByTriad2 を呼出していない場合も 0 を返す。
備考	本関数は、衛星制御スクリプトとしてのみ使用可能である。 (シナリオスクリプトとしては使用不可)

(8) changeAttByQt

説明	衛星姿勢を、指定されたクォータニオンに回転する。 角加速度と最大角速度、およびマヌーバ角から作成した角速度プロファイルに従って、姿勢回転を模擬する。	
形式	changeAttByQt(scName, q1, q2, q3, q4, acc, wMax)	
パラメータ	scName	衛星名
	q1	クォータニオン(q1)
	q2	クォータニオン(q2)
	q3	クォータニオン(q3)
	q4	クォータニオン(q4)
	acc	角加速度[deg/s ²]
	wMax	最大角速度[deg/s]
戻り値	姿勢変更所要時間[sec]	
備考	図 2.6-2 の角速度プロファイルに従い、制御を模擬する。	

(9) rotateParts

説明	形状モデルのパーツに対して、回転角を設定する。	
形式	rotateParts (,partsName, rotOrder ,ang1,ang2,ang3)	
パラメータ	partsName	形状モデルのパーツ名
	rotOrder	回転順序コード(123,213,etc..)
	ang1-ang3	回転角[deg]
備考	本関数は、衛星制御スクリプトとしてのみ使用可能である。 (シナリオスクリプトとしては使用不可)	

(10) changeOrbit

説明	軌道要素を瞬時に変更する。	
形式	changeOrbit(scName,dVx,dVy,dVz, coordType)	
パラメータ	scName	衛星名
	dVx,dVy,dVz	慣性系における ΔV 成分[km/s]
	coordType	0 : RTN 座標系、1 : 慣性座標系、 2 : 軌道座標系(LVLH)
備考	軌道生成種別として「SGP4」以外を指定した場合に有効。 RTN : Radial Tangential Nomal 動径方向を X、 速度方向を T、 軌道面方向を N	

(11) thrusterBurn

説明	スラスタによる軌道制御 (ΔV) を行う。	
形式	thrusterBurn(sc,dVx,dVy,dVz,T,coordType)	
パラメータ	scName	衛星名
	dVx,dVy,dVz	慣性系における ΔV 成分[km/s]
	T	増速時間[sec]
	coordType	0 : RTN 座標系、1 : 慣性座標系、 2 : 軌道座標系(LVLH) 3 : 機体座標系
備考	軌道生成種別として特別摂動法を指定した場合に有効。	

(12) setSfKp

説明	大気密度モデルパラメータとして、ソーラーフラックスと地磁気指標値を設定する。	
形式	setSfKp(scName, F107, Kp)	
パラメータ	scName	衛星名
	F107	ソーラーフラックス(f10.7 値)
	Kp	Kp 指標値
備考	大気密度モデルとして、Jacchia-Roberts または MSISE-00 を選択している場合のみ、本設定は有効である。	

(13) setSensorAngle

説明	搭載センサ／アンテナのジンバル角を設定する。	
形式	setSensorAngle(sensorName, angle1, angle2)	
パラメータ	sensorName	センサ／アンテナ名称を指定する・
	angle1	AZ/EL 方式 : AZ 角 X/Y 方式 : X 角 [deg]
	angle2	AZ/EL 方式 : EL 角 X/Y 方式 : Y 角 [deg]
備考	例) setSensorAngle(XANT, az, el);	

(14) makeAttitude

説明	Triad 法により決定された姿勢パラメータを求める。	
形式	q = makeAttitude (scName,axis1,vec1,axis2,vec2)	
パラメータ	scName	衛星名
	axis1	指向方向とする衛星機体方向ベクトル (単位ベクトル)
	vec1	指向方向のベクトル。(単位ベクトルでなくても可)
	axis2	第2ベクトル方向とする衛星機体上の方向ベクトル(単位ベクトル)
	vec2	第2ベクトル方向のベクトル (単位ベクトルでなくても可)
戻り値	姿勢パラメータ (クォータニオン)	
備考		

(15) setTorq

説明	衛星にトルクを与える。	
形式	setTorq(scName,torq)	
パラメータ	scName	衛星名
	torq	トルクベクトル。
備考	姿勢ダイナミクスモードが ON の場合のみ有効。	

(16) readAttFile

説明	姿勢データの時系列ファイルを読み込み、姿勢プロファイルとして保持する。	
形式	profile = readAttFile(filePath)	
パラメータ	filePath	姿勢データ時系列ファイルのパス名
戻り値	姿勢プロファイルデータの識別番号	
備考	filePath に相対パス名("./temp.txt"など) を指定できますが、その際、カレントフォルダは、スクリプトファイルが格納されたフォルダとなります。 ただし、スクリプトファイルが保存されていない場合は、ユーザデスクトップフォルダとなります。	

(17) setTrackObject

説明	衛星アンテナ（センサ）の追尾オブジェクトを設定し、駆動を開始する。 また、追尾オブジェクトをクリアし、駆動を停止する。	
形式	setTrackObject(ant, trackObject, rate);	
パラメータ	ant	衛星アンテナまたはセンサ名称
	trackObject	追尾するオブジェクト名。（局名など） NULL を指定すると、追尾を解除する。
	rate	関数実行後、目的のオブジェクト(地上局など)を指向するために、APM 等の駆動装置を動かす最大速度(deg/sec)を指定する。 例えば AZ/EL アンテナで、関数実行時と指向達成時の AZ/EL 角の大きい方の角度差が 60 度であるようなとき、10 度/sec を指定すると、丁度 6 秒後に指向が達成されるよう、制御される。 また、ゼロを指定すると瞬時に目的のオブジェクトを指向する。
戻り値	なし	

(18) getAtt

説明	姿勢プロファイルデータから、指定時刻の姿勢パラメータを取得する。	
形式	status = getAtt(profile,t,q);	
パラメータ	profile	姿勢プロファイル
	t	時刻。time()関数で取得することができる。
	q	クォータニオン。 事前に、q=vector(0,0,0,0)などで作成しておくこと。
戻り値	姿勢取得結果。 0=指定時刻のデータを取得できた。 -1=指定時刻のデータは存在しない。	
備考	取得した姿勢は、setAttByQt等で、衛星姿勢に反映することができる。	

(19) setGimbalAngle

説明	センサ／アンテナのジンバル角を設定する。	
形式	setGimbalAngle(sensor、angle1、angle2、rate)	
パラメータ	sensor	アンテナ／センサオブジェクト名
	angle1	Azimuth角またはX角[deg]
	angle2	Elebation各またはY角[deg]
	rate	ジンバル角変更レート[deg/s]
備考		

(20) makeMnvPlanIP

説明	面内軌道制御計画を作成する。	
形式	makeMnvPlanIP(sat,baseSat,chgDe,chgDa,dEx,dEy,dA,t1,dv1,t2,dv2);	
パラメータ (入力)	sat	制御対象の衛星名
	baseSat	基準軌道を飛行する衛星名 (基準衛星)
	chgDe	離心率ベクトルの制御有無 (0:なし、1:あり)
	chgDa	軌道長半径の制御有無 (dA の内容) (0:なし、1:Drift Rate 指定、2:軌道の変化量指定)
	dEx	相対離心率ベクトル(X 成分)に対する目標値[km] (相対離心率ベクトルに基準衛星の軌道長半径を乗じて距離で表した値) $dEx = a_0 \cdot \{e \cdot \cos(\omega) - e_0 \cdot \cos(\omega_0)\}$
	dEy	相対離心率ベクトル(Y 成分)に対する目標値[km] $dEy = a_0 \cdot \{e \cdot \sin(\omega) - e_0 \cdot \sin(\omega_0)\}$
	dA	chgDa=1 のとき、基準衛星に対するドリフトレート [m/s] chgDa=2 のとき、相対軌道長半径の目標値 [m]
パラメータ (出力)	t1	1 回目の ΔV 時刻 (UnixTime 表現による絶対時刻)
	dv1	1 回目の ΔV 量 (RTN 座標系成分) [km/s]
	t2	2 回目の ΔV 時刻 (UnixTime 表現による絶対時刻)
	dv2	2 回目の ΔV 量 (RTN 座標系成分) [km/s]
備考	baseSat に制御目標とする理想軌道(大気抵抗や太陽・月引力を受けない軌道)の衛星を指定する。	

(21) makeMnvPlanOP

説明	面外軌道制御計画を作成する。	
形式	makeMnvPlanOP(sat,baseSat,dix,diy,t,dv);	
パラメータ (入力)	sat	制御対象の衛星名
	baseSat	基準軌道を飛行する衛星名
	dIx	相対軌道傾斜角ベクトル X 成分に対する目標値[km] (相対軌道傾斜角ベクトルに基準衛星の軌道長半径を乗じて距離で表した値) $dIx = a_0 \cdot \{(\Omega - \Omega_0) \cdot \sin(i_0)\}$
	dIy	相対軌道傾斜角ベクトル Y 成分に対する目標値[km] $dIy = a_0 \cdot (\Omega - \Omega_0)$
パラメータ (出力)	t	1 回目の ΔV 時刻 (UnixTime 表現による絶対時刻)
	dv	1 回目の ΔV 量 (RTN 座標系成分) [km/s]
備考	baseSat に制御目標とする理想軌道(大気抵抗や太陽・月引力を受けない軌道)の衛星を指定する。	

3. スクリプト編集機能

3.1 衛星制御スクリプト

衛星制御スクリプト編集は以下の手順により行います。

- (1) オブジェクトツリービューから対象衛星の「スクリプト」をクリックし、「スクリプト制御」を選択します。

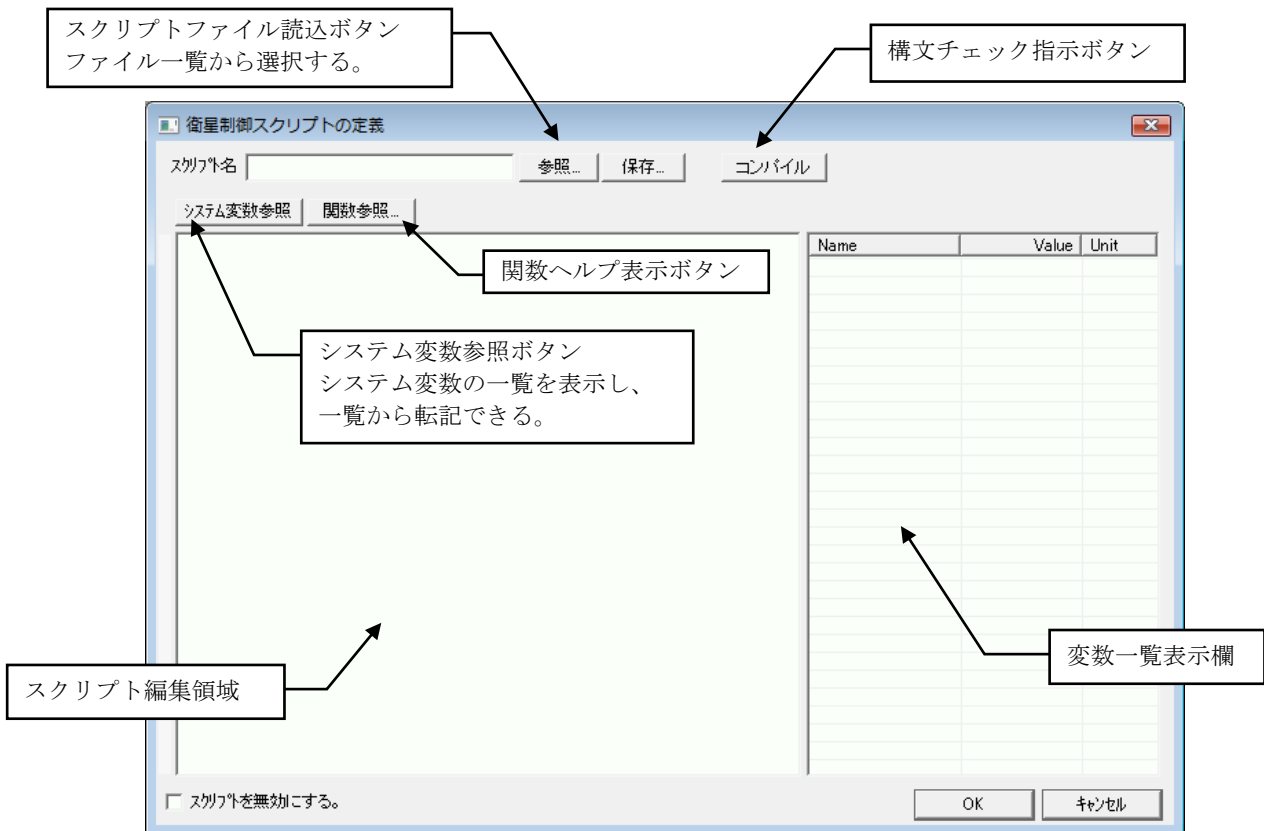
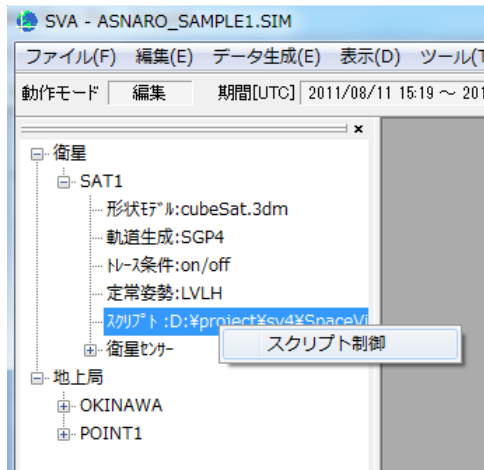


図 3.1-1 編集初期画面

(3) システム変数の参照

衛星制御スクリプト定義画面で、「システム変数参照」ボタンをクリックすると、以下のシステム変数参照画面が表示されます。



オブジェクト、データ項目、座標系の順で選択し、「OK」ボタンをクリックすることで、衛星制御スクリプト定義画面編集領域のカーソル位置に、システム変数が転記されます。

3.2 シナリオスクリプト

省略

3.3 ユーザ変数定義スクリプト

省略

4. スクリプトのデバッグ

スクリプトのデバッグは、スクリプト定義画面の変数一覧と、スクリプト中に記述した `print` 文の出力を確認することによって行います。

`print` 文の出力は、以下のようなログ画面に出力されます。

また、ログ画面は、メニューバーから「ウインドウ」→「メッセージウインドウ」をセンタすることにより表示されます。

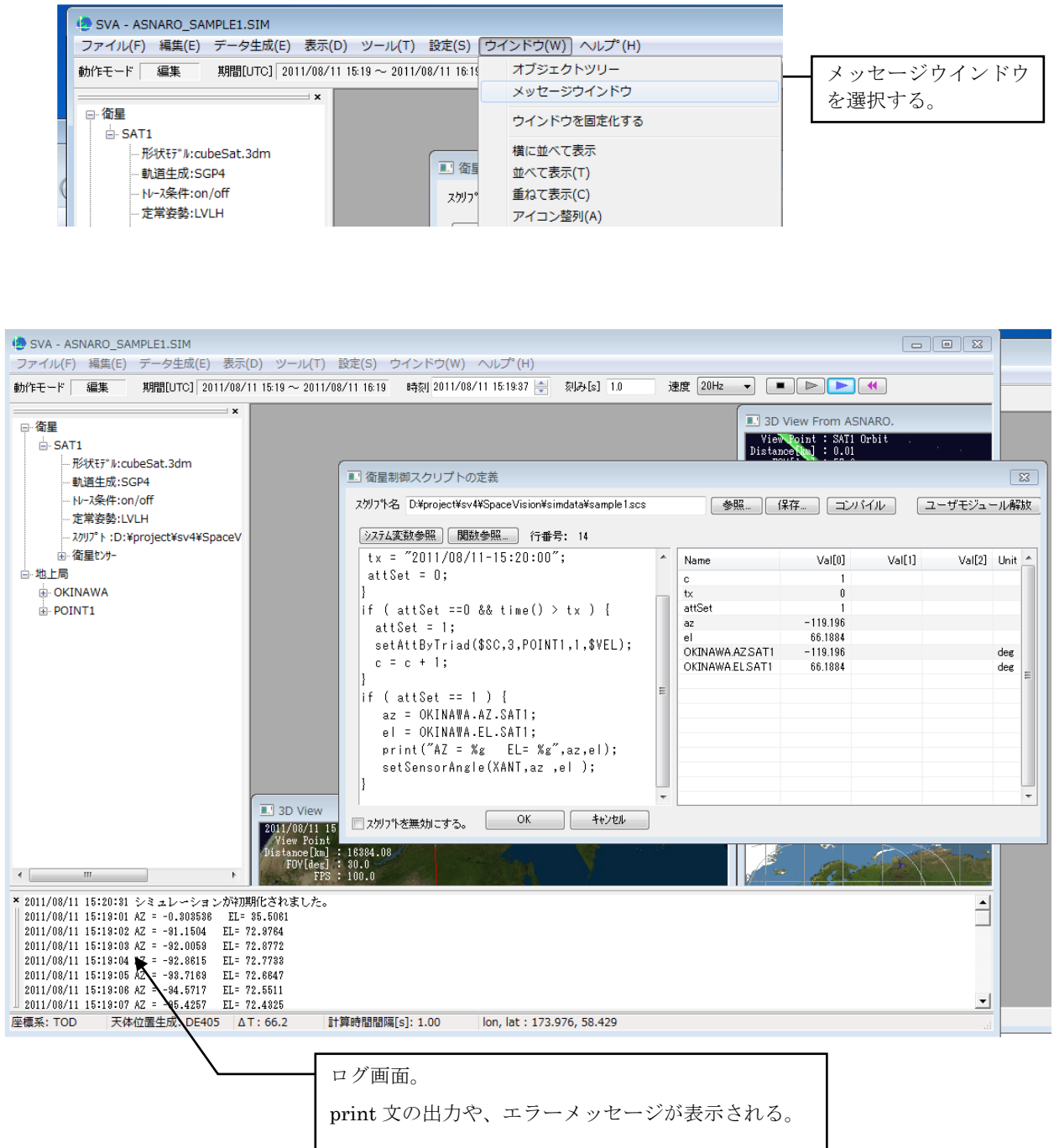


図 3.3-1 スクリプトのデバッグイメージ

5. シミュレーション中のスクリプト実行タイミング

SVA におけるシミュレーションの実行と、スクリプト実行のタイミングについて、以下の処理フローに示します。

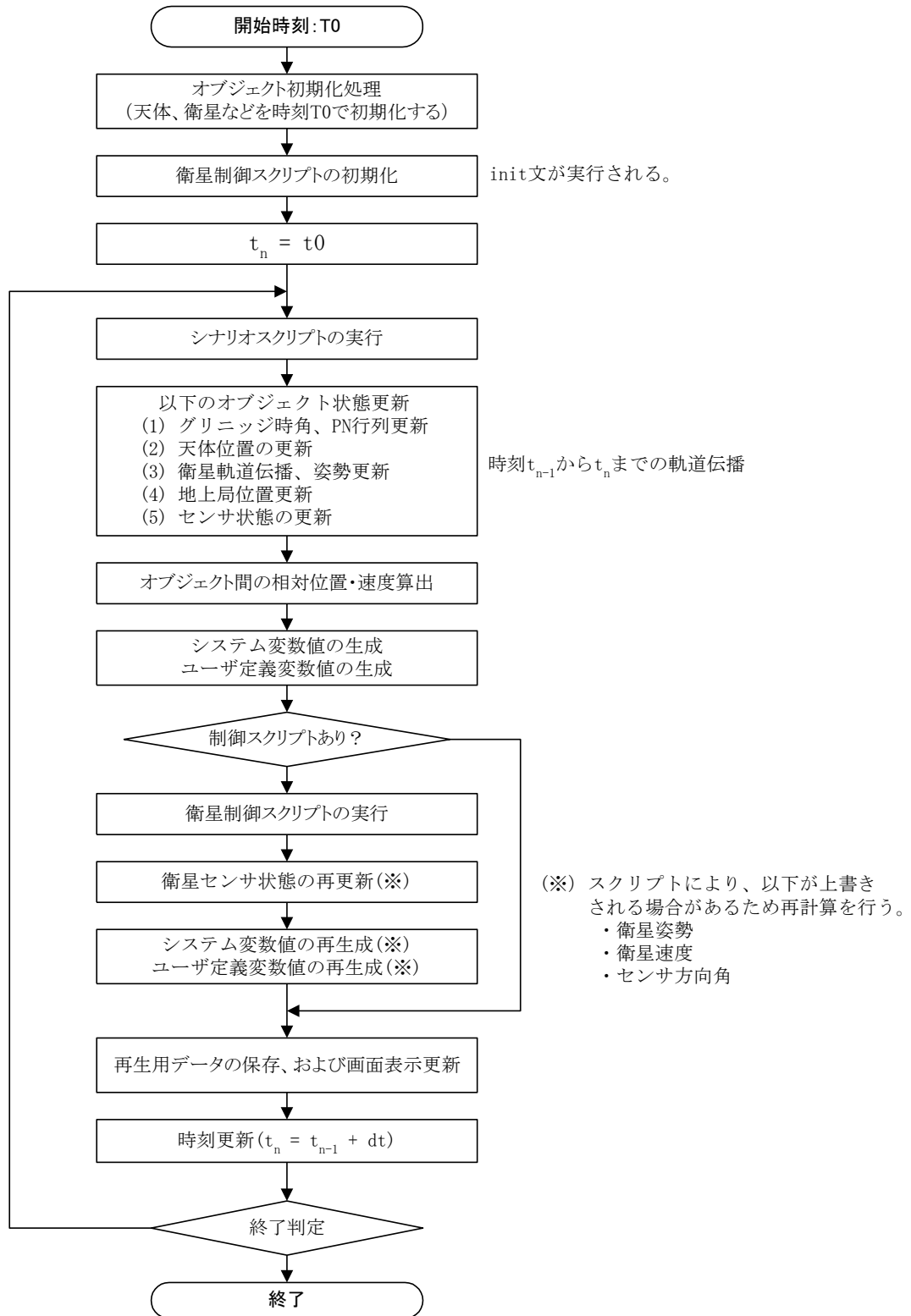


図 3.3-1 シミュレーションフロー