SATELLITE GEODESY

2021

Doğu İlmak

Content

First Question

Definition of the Problem:

Before I go into the solution of the first problem, I would like to specify what the problem is:

The true anomaly (v) value will be generated in cell A2, satellite orbit major half-axis (a) in cell B2, and satellite orbital axiality (e) in cell C2. Calculate the prifocal (P, Q, W) coordinates of the satellite according to these data.

You can see the given values from figure 1.

	A	B	C
1	v°	a (m)	e
2	153.6206897	9131.00835200	7.704
3			
4			
5			
6			
7	öğrenci No -->	140609008	

Fig.1 True anomaly (v◦), satellite orbit major half-axis (a) and satellite orbital axiality (e).

Solution of the problem

Here, Fig.2 is taken as reference for the calculation of P, Q and W coordinates. The equation to find the value of P, Q and W from the figure:

$$r(PQW) = \frac{a(1 - e^2)}{1 + e\cos(v)} \cdot \begin{matrix} \cos(v) \\ \sin(v) \\ 0 \end{matrix}$$

$\frac{a(1-e^2)}{1+e\cos(v)}$ differential equation of orbit; When multiplied by cos (v) and sin (v), the values of P, Q and W are obtained:

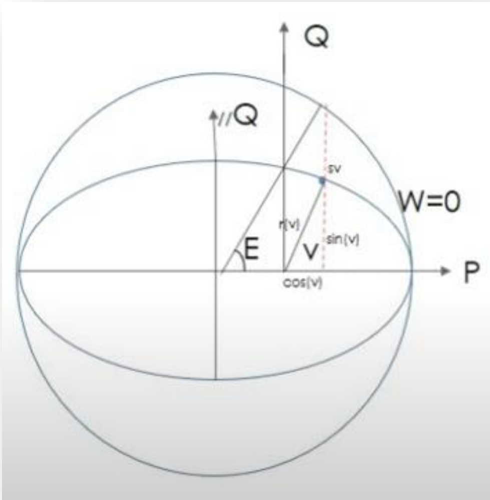

Fig.2 Professional coordinates (P, Q, W), differential equation of orbit (r (v)) and Iteration angle representation.

Satellite's Perifocal Coordinates:

$$P = \frac{91.310083520\text{m} \cdot (1-7.704^2)}{1+e \cdot \cos(153^0.62068966)} \,.\cos(153^0.62068966) = \mathbf{-80.8821594551\ km}$$

$$Q = \frac{91.3100835200\text{m} \cdot (1-7.704^2)}{1+e \cdot \cos(153^0.62068966)} \,.\sin(153^0.62068966) = \mathbf{40.1138553510\ km}$$

W = 0 km

Profikal coordinates have been calculated by taking the values of a value in meters and kilometers as reference. Since the calculations are in 2 dimensional plane, omega value is taken as 0. Calculations are made in units of kilometers.

20	**Meter**	
21	90283.13856554	r(v)
22	P	-80882.1594551001
23	Q	40113.8553510409
24	W	0
25	**Kilometer**	
26	90.28313857	r(v)
27	**P**	-80.8821594551
28	**Q**	40.1138553510
29	**W**	0

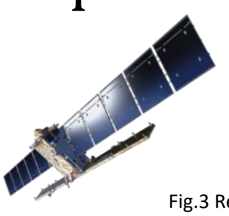

Fig.3 Representation of professional coordinates (P, Q, W) and differential equation of orbit (r (v)) with different units in Microsoft Excel.

For meter values:

r(v)	=(B2*(1-C2*C2))/(1+C2*COS(RADYAN(A2)))
P	=A21*COS(RADYAN(A2))
Q	=A21*SİN(RADYAN(A2))
W	0

For kilometremeter values:

r(v)	=(B2*(1-C2*C2))/(1+C2*COS(RADYAN(A2)))/1000
P	=B22/1000
Q	=B23/1000
W	0

Second question

Definition of the Problem:

Before I go into the solution of the second problem, I would like to specify what the problem is:

The inclination angle (I◦) in cell A2, the recessation angle () in cell B2, and the angle of perige argument () in cell C2 will be produced. From the PQW coordinates of the point obtained from the 1st group, calculate the Earth-centered Intern (ECI) coordinates according to the given.

	A	B	C
1	i	Ω	ω
2	58.32222222	5.45	268.9310345
3			
4			
5			
6			
7	öğrenci No -->	140609008	

Fig.4 Microsoft Excel representation of the calculated inclination angle (i◦), rectalization angle (Ω◦) and perimeter argument (ω◦) angle.

To find the rotations around all axes, counterclockwise rotation matrices are applied around the x, y and z axes, respectively.

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}$$

$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$

$$R_z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Fig.5 Rotation matrices.

Solution of the problem

Calculation of the matrix values specified in Figure 5 in Microsoft Excel:

$$r_{ECI} = R_3(-\Omega)R_1(-i)R_3(-\omega)r_{PQW}$$

For the calculation of rECI coordinates, rotation matrices in figure 5 will be used. The representation of rotation matrices and transformation matrices in Microsoft Excel is shown on the 6th and 7th figures.

Fig.6 Representation of the results of the transformation matrix on Excel.

	A	B	C	D	E
1	i	Ω	ω		
2	58.32222222	5.45	268.9310345		
3					
4					
5					
6					
7	öğrenci No -->	140609008			
11					
12					
13					
14			1	0	0
15	R1(-i)=	-58.32222222	0	0.525141624	-0.85101485
16			0	0.85101485	0.525141624
17			0.995479461	-0.094977069	0
18	R3(-Ω)=	-5.45	0.094977069	0.995479461	0
19			0	0	1
20			-0.018655886	0.999825964	0
21	R3(-ω)=	-268.9310345	-0.999825964	-0.018655886	0
22			0	0	1

Fig.7 Transformation and

R1 (-I) xR3 (-ω) matrix multiplication

Via Microsoft Excel

representation.

27		-0.018655886	0.999825964	0
28	R1(-i)xR3(-ω)=	-0.52505023	-0.009796982	-0.85101485
29		-0.850866743	-0.015876436	0.525141624
30				
31	**Transformation Matrix**			
32		0.031296181	0.9962367	0.080826896
33	R3(-Ω)*R1(-i)xR3(-ω)	-0.524448601	0.085207845	-0.847167804
34		-0.850866743	-0.015876436	0.525141624

Microsoft Excel Function:

R1(-i)xR3(-ω)=	**=DÇARP(C14:E16,C20:E22)**
R3(-Ω)*R1(-i)xR3(-ω)	**=DÇARP(C17:E19,B27:D29)**

rECI Coordinates:

36		37.43159216		**P=**	-80.8821594551
37	**rECI=**	45.83655056		**Q=**	40.1138553510
38		68.18307452		**W=**	0.0000000000

Microsoft Excel Function:

rECI=	**X**	**=DÇARP(B32:D34,E36:E38)**
	Y	
	Z	

	A	B	C	D	E
1	i	Ω	ω		
2	58.32222222	5.45	268.9310345		
3					
4					
5					
6					
7	öğrenci No -->	140609008			
11					
12					
13					
14			1	0	0
15	**R1(-i)=**	-58.32222222	0	0.525141624	-0.85101485
16			0	0.85101485	0.525141624
17			0.995479461	-0.09497707	0
18	**R3(-Ω)=**	-5.45	0.094977069	0.995479461	0
19			0	0	1
20			-0.01865589	0.999825964	0
21	**R3(-ω)=**	-268.9310345	-0.99982596	-0.01865589	0
22			0	0	1
23					
24					
25					
26					
27		-0.018655886	0.999825964	0	
28	**R1(-i)xR3(-ω)=**	-0.52505023	-0.00979698	-0.85101485	
29		-0.850866743	-0.01587644	0.525141624	
30					
31	**Transformation Matrix**				
32		0.031296181	0.9962367	0.080826896	
33	**R3(-Ω)*R1(-i)xR3(-ω)**	-0.524448601	0.085207845	-0.8471678	
34		-0.850866743	-0.01587644	0.525141624	
35					
36		37.43159216		P=	-80.8821594551
37	rECI=	45.83655056		Q=	40.1138553510
38		68.18307452		W=	0.0000000000

Additional Calculations:

Calculation of rECI Coordinates from P, Q and W Values with Python 3.7 on PyCharm 2020.3 IDE

Fig.8 Representation of Python code that calculates rECI coordinates from perifocal coordinates via Notepad++.

```python
import numpy as np
import math as m


def r_eci(P, Q, W, i, omega, w):

    PQW = np.array([[P],
                    [Q],
                    [W]])

    R1i = np.array([[1, 0, 0],
                   [0, m.cos(m.radians(-i)), m.sin(m.radians(-i))],
                   [0, -m.sin(m.radians(-i)), m.cos(m.radians(-i))]])

    R3omega = np.array([[m.cos(m.radians(-omega)), m.sin(m.radians(-omega)), 0],
                   [-m.sin(m.radians(-omega)), m.cos(m.radians(-omega)), 0],
                   [0, 0, 1]])

    R3w = np.array([[m.cos(m.radians(-w)), m.sin(m.radians(-w)), 0],
                   [-m.sin(m.radians(-w)), m.cos(m.radians(-w)), 0],
                   [0, 0, 1]])

    R1ixR3w = np.matmul(R1i, R3w)
    R3wR1R3 = np.matmul(R3omega, R1ixR3w)
    ECI = np.matmul(R3wR1R3, PQW)

    print("R1(-i):\n", R1i)
    print("R3(-\u03A9):\n", R3omega)
    print("R3(-w):\n", R3w)
    print("R1(-i)xR3(-w):\n", R1ixR3w)
    print("R3(-\u03A9)xR1(-i)xR3(-w):\n", R3wR1R3)
    print("rECI:\n", ECI)


r_eci(-80.87852775, 40.11205411, 0, 58.322222222, 5.45, 268.9310345)
```

Numpy and Math libraries are added for matrix calculations.

The function named r_eci (): is created.

Perifocal coordinates are printed on the PQW array as a 3x1 matrix.

The inclination angle, recessation angle, and perige argument are applied to the rotation matrices R1 (-i), R3 (-Ω°) and R3 (-ω°), respectively.

The R1 (-i) and R3 (-ω°) matrices are multiplied and assigned to the value of R1ixR3w.

Matrix multiplication is made between R3 (-Ω°) and R1ixR3w values. The result matrix is assigned to variable R3wR1R3.

The variable R3wR1R3 is multiplied by the matrix containing the perifocal coordinates requested from us. As a result, the rECI coordinates are obtained.

P, Q, W, i, Ω° and ω° values are written into the function.

All obtained values are printed on the terminal.

Output:

```
R1(-i):
 [[ 1.          0.          0.        ]
 [ 0.          0.52514162 -0.85101485]
 [ 0.          0.85101485  0.52514162]]
R3(-Ω):
 [[ 0.99547946 -0.09497707  0.        ]
 [ 0.09497707  0.99547946  0.        ]
 [ 0.          0.          1.        ]]
R3(-w):
 [[-0.01865589  0.99982596  0.        ]
 [-0.99982596 -0.01865589  0.        ]
 [ 0.          0.          1.        ]]
R1(-i)xR3(-w):
 [[-0.01865589  0.99982596  0.        ]
 [-0.52505023 -0.00979698 -0.85101485]
 [-0.85086674 -0.01587644  0.52514162]]
R3(-Ω)xR1(-i)xR3(-w):
 [[ 0.03129618  0.9962367   0.0808269 ]
 [-0.5244486   0.08520785 -0.8471678 ]
 [-0.85086674 -0.01587644  0.52514162]]
rECI:
 [[37.43159213]
 [45.83655056]
 [68.18307453]]

Process finished with exit code 0
```

You can see the view of the obtained calculations on the terminal on the 9th figure. Calculation has been completed successfully.

Fig.9 Representation of Python code in the PyCharm 2020.3 IDE terminal.

Calculation of rECI Coordinates on PyCharm 2020.3 IDE with Python 3.7 from Inclination Angle, Recthesing Angle and Perige Argument

Fig.10 Representation of Python code using Notepad++ to calculate rECI coordinates based on inclination angle, recessation angle and perige argument.

```python
import numpy as np
import math as m


def r_eci(e, a, v, i, omega, w):

    r_v = a*(1 - e**2) / (1 + e * m.cos(m.radians(v)))

    P = r_v * m.cos(m.radians(v))
    Q = r_v * m.sin(m.radians(v))
    W = 0

    PQW = np.array([[P],
                    [Q],
                    [W]])

    R1i = np.array([[1, 0, 0],
                   [0, m.cos(m.radians(-i)), m.sin(m.radians(-i))],
                   [0, -m.sin(m.radians(-i)), m.cos(m.radians(-i))]])

    R3omega = np.array([[m.cos(m.radians(-omega)), m.sin(m.radians(-omega)), 0],
                   [-m.sin(m.radians(-omega)), m.cos(m.radians(-omega)), 0],
                   [0, 0, 1]])

    R3w = np.array([[m.cos(m.radians(-w)), m.sin(m.radians(-w)), 0],
                   [-m.sin(m.radians(-w)), m.cos(m.radians(-w)), 0],
                   [0, 0, 1]])

    R1ixR3w = np.matmul(R1i, R3w)
    R3wR1R3 = np.matmul(R3omega, R1ixR3w)
    ECI = np.matmul(R3wR1R3, PQW)

    print("R1(-i):\n", R1i)
    print("R3(-\u03A9):\n", R3omega)
    print("R3(-w):\n", R3w)
    print("R1(-i)xR3(-w):\n", R1ixR3w)
    print("R3(-\u03A9)xR1(-i)xR3(-w):\n", R3wR1R3)
    print("rECI:\n", ECI)

```

Numpy and Math libraries are added for matrix calculations.

The function named r_eci (): is created.

The differential equation (r_v) formula of the orbit is created in order to calculate the P and Q values. With this equation, the values of P and Q are obtained by multiplying by the cosine and sine of the angle v sırasıyla, respectively.

The calculated perifocal coordinates are printed in the PQW array as a 3x1 matrix.

The inclination angle, recessation angle, and perige argument are applied to the rotation matrices R1 (-i), R3 (-Ω◦) and R3 (-ω◦), respectively.

The R1 (-i) and R3 (-ω◦) matrices are multiplied and assigned to the variable R1ixR3w.

Matrix multiplication is made between R3 (-Ω◦) and R1ixR3w values. The result matrix is assigned to variable R3wR1R3.

The value R3wR1R3 is multiplied by the matrix containing the perifocal coordinates requested from us. As a result, the rECI coordinates are obtained.

The values e, a, v◦, i, Ω◦ and ω◦ are written into the function r_eci.

All obtained values are printed on the terminal.

Output:

```
R1(-i):
 [[ 1.          0.          0.        ]
 [ 0.          0.52514162 -0.85101485]
 [ 0.          0.85101485  0.52514162]]
R3(-Ω):
 [[ 0.99547946 -0.09497707  0.        ]
 [ 0.09497707  0.99547946  0.        ]
 [ 0.          0.          1.        ]]
R3(-w):
 [[-0.01865589  0.99982596  0.        ]
 [-0.99982596 -0.01865589  0.        ]
 [ 0.          0.          1.        ]]
R1(-i)xR3(-w):
 [[-0.01865589  0.99982596  0.        ]
 [-0.52505023 -0.00979698 -0.85101485]
 [-0.85086674 -0.01587644  0.52514162]]
R3(-Ω)xR1(-i)xR3(-w):
 [[ 0.03129618  0.9962367   0.0808269 ]
 [-0.5244486   0.08520785 -0.8471678 ]
 [-0.85086674 -0.01587644  0.52514162]]
rECI:
 [[37.43159213]
 [45.83655056]
 [68.18307453]]

Process finished with exit code 0
```


You can see the view of the obtained calculations on the terminal on the 11th figure. Calculation has been completed successfully.

Fig.11 Representation of Python code in the PyCharm 2020.3 IDE terminal.