Maria Mańk 311590

Zadanie 2 – Astronomia geodezyjna

Cele:

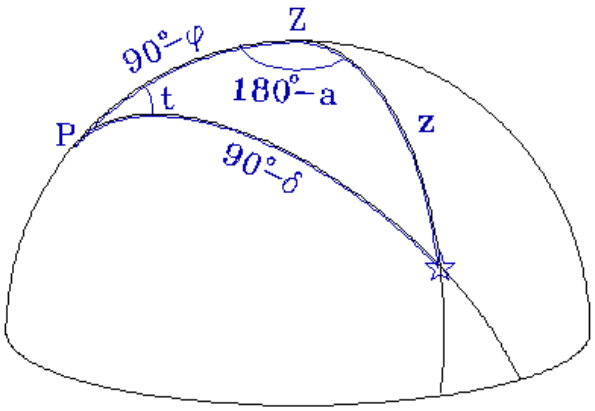
Dla wybranej gwiazdy(o znanej **rektastenzji (α)** i **deklinacji (δ)**) ze swojego znaku zodiaku dla 3 miejsc na ziemi(półkula północna, południowa, okolice równika):

- przeliczyć współrzędne w układzie horyzontalnym (Az, h) dla danego miejsca obserwatora,
- przedstawić współrzędne gwiazdy na wykresie w danym szeregu czasowym(Wykres zależności wysokości od czasu, wykres zależności azymutu od czasu),
- zaprezentować wyniki na sferze w układzie horyzontalnym, obrazując w ten sposób pozorny ruch gwizady.

Teoria:

Pozorny ruch gwiazd – po kilkugodzinnej obserwacji jesteśmy w stanie zaobserwować ruch gwiazd na niebie wokół osi Ziemi. Jednak nie jest to rzeczywisty ruch, a złudzenie wynikające z ruchu obrotowego Ziemi, gdyż to obserwator się porusza i zmienia się jego kąt patrzenia na gwiazdę i odległość od niej, a nie gwiazda porusza się.

Trójkąt paralaktyczny – trójkąt na sferze niebieskiej, którego wierzchołkami są: obserwowany obiekt astronomiczny, zenit i biegun świata. Umożliwia wyprowadzenie wzorów i otrzymanie współrzędnych horyzontalnych.



$$\sin a \cos B = \cos b \sin c - \sin b \cos c \cos A$$
$$\sin b \cos C = \cos c \sin a - \sin c \cos a \cos B$$
$$\sin c \cos A = \cos a \sin b - \sin a \cos b \cos C$$

$$\frac{\sin a}{\sin A} = \frac{\sin b}{\sin B} = \frac{\sin c}{\sin C}$$

Dane:

Gwiazdą którą wybrałam jest Denebola, z gwiazdozbioru lwa. Gwiazda ta ma rektastencję równą 11.81777778 i deklinację 14.57672222 (stopni).

Natomiast miejsca jakie wybrałam, to okolice Paryża i dwa losowo wybrane miejsca zgodne z wytycznymi.

```
fi1= np.deg2rad(45)
lb1 = np.deg2rad(17)
fi2= np.deg2rad(1)
lb2= np.deg2rad(25)
fi3= np.deg2rad(-60)
lb3= np.deg2rad(190)
```

Biblioteki:

Do uzyskania zamierzonego efektu wykorzystałam 4 biblioteki. Numpy do przeliczeń matematycznych. Jdcal do otrzymania daty juliańskiej oraz tkinter i matplotlib do prezentacji wyników za pomocą wykresów.

```
import numpy as np
import jdcal as jc
import tkinter as tk, tkinter.ttk as ttk
import matplotlib.backends.backend_tkagg as TkAgg
from matplotlib.figure import Figure
```

Poszczególnie funkcje znajdujące się w programie:

Funkcja licząca średni czas gwiazdowy Greenwich na podstawie daty juliańskiej

```
def GMST (jd):
    t = (jd - 2451545)/36525
    g = 280.46061837 + 360.98564736629*(jd - 2451545) + 0.000387933*(t**2) - (t**3)/38710000
    g = g%360
    return g
```

Funkcja zwracająca kąt godzinny (w radianach) na podstawie daty, długości geodezyjnej i rektastensji.

```
def kh (y, m, d, h, lb, alfa): #dobrze
    jd = sum(jc.gcal2jd(y, m, d))
    gm = GMST(jd)
    UT1 = h * 1.002737909350795
    S = UT1*15 + lb + gm #czas gwiazdowy (stopnie)
    t = S - alfa*15 # kat godzinny (stopnie)
    t = np.deg2rad(t) #kat godzinny radiany
    return t
```

Dwie funkcje wynikające z trójkąta paralaktycznego.

Funkcja licząca Azymut gwiazdy w radianach na podstawie fi, deklinacji i kąta godzinnego (t) (czyli współrzędnych równikowych godzinnych i fi). Uwzględniająca ćwiartkę w jakiej znajduje się dany kąt.

```
def A(fi, dek, t): #w poprawnych jednostkach już
    g = -np.cos(dek)*np.sin(t)
    d = np.cos(fi)*np.sin(dek)-np.sin(fi)*np.cos(dek)*np.cos(t)
    a = np.arctan(g/d)
    ad = np.rad2deg(a)
    if g>0:
        if d<0:
            ad = 180 + ad
        if d> 0 and ad<360:
            return a
    elif g<0:
        if d>0:
            ad= 360 + ad
        elif d<0:
            ad+=180
    if ad > 360:
        ad-=360
    elif ad < 0:
        ad+=360
    a = np.deg2rad(ad)
    return a #też w radianach
```

Funkcja licząca odległość zenitalną.

```
def z(fi, dek, t):
    res = np.arccos(np.sin(fi)*np.sin(dek)-np.cos(fi)*np.cos(dek)*np.cos(t))
    return res
```

Funkcja przeliczająca współrzędne biegunowe na kartezjańskie, umożliwiające prezentacje wyników w przestrzeni 3d.

```
def toxyz (r, z, A):
    x = r*np.sin(z)*np.cos(A)
    y = r*np.sin(z)*np.sin(A)
    z = r*np.cos(z)
    return x, y, z
```

Funkcja zwracająca tablice godzin, azymutów, odległości zenitalnych i współrzędnych xyz na sferze dla danych fi i lambda. Jest i zatem funkcja wywołująca wcześniej napisane funkcje dla danej lokalizacji.

```
def licz(fi, lb):
    x1 = []
    y1= []
    z1 = []
    od1 = []
    A1= []
    h = []
    for i in range(24):
        t = kh(2022, 8, 13, i, lb, a)
        d = z(fi, dek, t)
        Az = A(fi, dek, t)
        h.append(i)
        resl1 = toxyz(1, d, Az )
        od1.append(d)
        A1.append(np.rad2deg(Az))
        x1.append(resl1[0])
        y1.append(resl1[1])
        z1.append(resl1[2])
    return h, A1, od1, x1, y1, z1
```

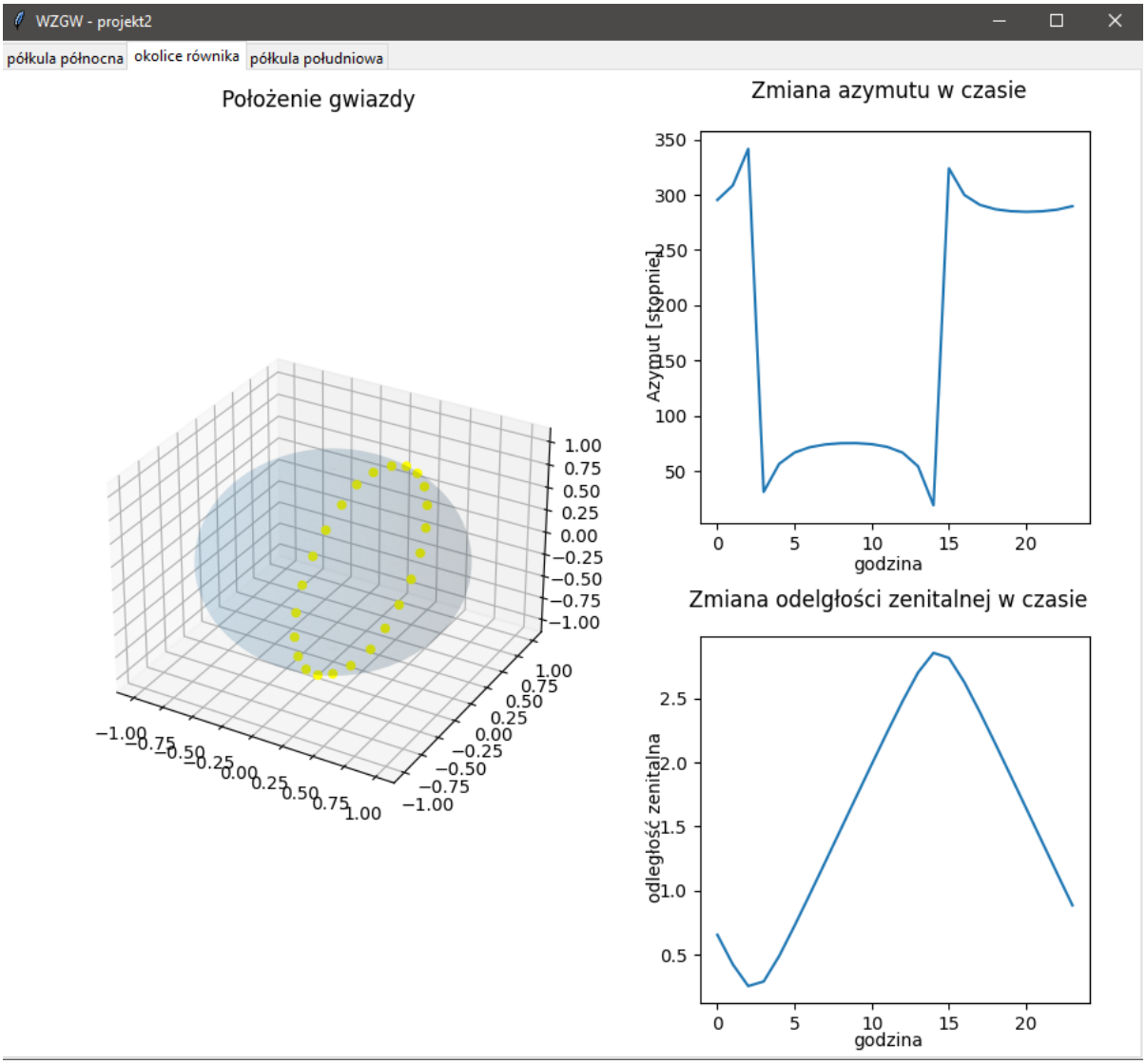
Ostatnia funkcja i jej wywołanie odpowiadają za ostateczną prezentację danych dla poszczególnych miejsc na ziemi – poprzez wywołanie funkcji licz dla tych lokalizacji. Następnie wyniki są wprowadzone do funkcji figury, odpowiedzialnych za ostateczną prezentację danych w okienku GUI.

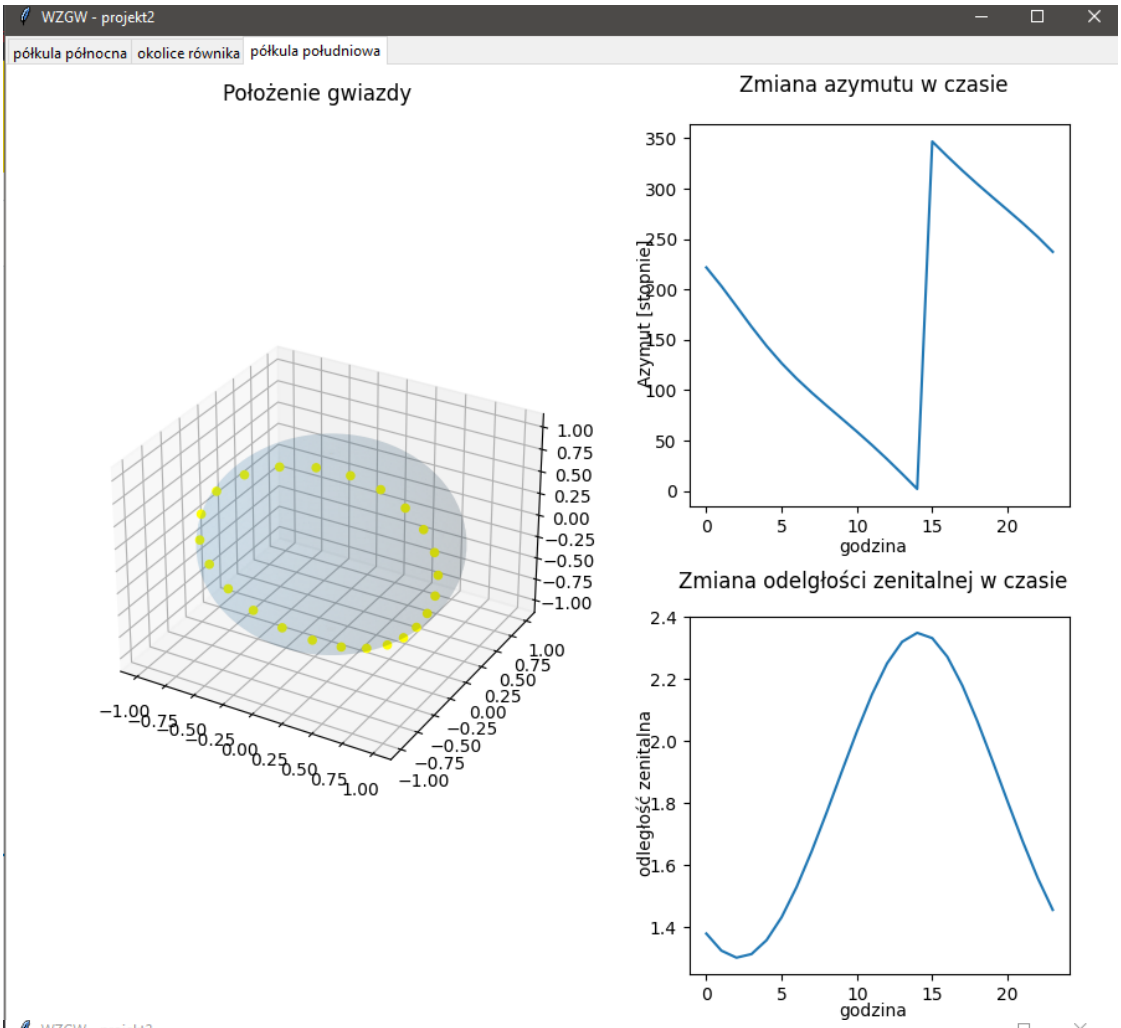
(nie opisuję gdyż ma to małe znaczenie w rozwiązaniu zadania)

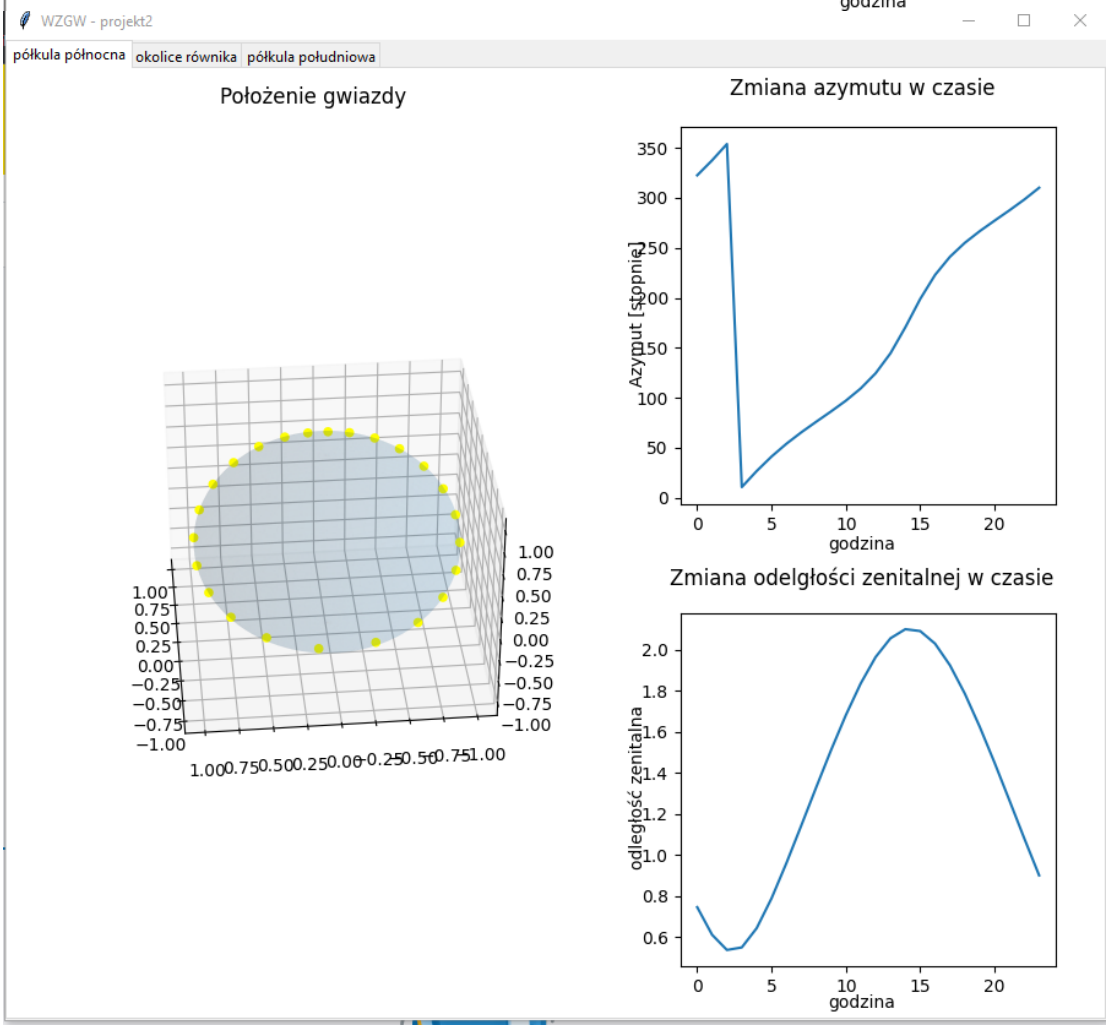
```
def glowna():
    a = licz(fi1, lb1)
    a1= licz(fi2, lb2)
    a2 = licz(fi3, lb3)
    figury(a[0], a[1], a[2], tab1, a[3], a[4], a[5])
    figury(a1[0], a1[1], a1[2], tab2, a1[3], a1[4], a1[5])
    figury(a2[0], a2[1], a2[2], tab3, a2[3], a2[4], a2[5])

glowna()
```

Prezentacja wyników dla poszczególnych półkul w aplikacji okienkowej:




WZGW - projekt2
półkula północna
okolice równika
półkula południowa
Położenie gwiazdy
Zmiana azymutu w czasie
Azymut [stopnie]
godzina
Zmiana odelgłości zenitalnej w czasie
odległość zenitalna
godzina


WZGW - projekt2
półkula północna
okolice równika
półkula południowa
Położenie gwiazdy
Zmiana azymutu w czasie
Azymut [stopnie]
godzina
Zmiana odelgłości zenitalnej w czasie
odległość zenitalna
godzina

Wnioski:

Ze zrealizowanego ćwiczenia wynika że:

- pozorny ruch gwiazdy po sferze zależy od miejsca z jakiego obserwujemy obiekt,
- gwiazdy poruszają się wokół bieguna niebieskiego
- trasa gwiazdy w okolicach równika jest niemal pionowa względem horyzontu