

Politechnika Warszawska  
Wydział Geodezji i Kartografii  
**Przedmiot: Informatyka Geodezyjna**

# PROJEKT 1

## **Program transformacji współrzędnych**

Oświadczam, że niniejsza praca stanowiąca podstawę do uznania osiągnięcia efektów uczenia się z przedmiotu Geodezja Wyższa została wykonana przeze mnie samodzielnie.

Adam Łostowski

Wojciech Litwicki

**Numer indeksu:** 325784 / 325783

Grupa: 3A - poniedziałek - 16.15  
Semestr: 4  
Rok akademicki: 2023/24

Data oddania ćwiczenia: 13 maja 2024  
Zajęcia: Informatyka Geodezyjna  
Prowadzący: mgr inż. Andrzej Szeszko

## Spis treści

<b>1</b>	<b>Cel ćwiczenia</b>	<b>3</b>
<b>2</b>	<b>Wykorzystane narzędzia i programy</b>	<b>3</b>
<b>3</b>	<b>Przebieg ćwiczenia</b>	<b>4</b>
<b>4</b>	<b>Podsumowanie</b>	<b>6</b>
4.1	Rezultat . . . . .	6
4.2	Nabyte umiejętności . . . . .	6
4.3	Zauważone błędy . . . . .	7
4.4	Bibliografia . . . . .	8

## 1 Cel ćwiczenia

Celem ćwiczenia było utworzenie programu/skryptu uruchamianego z pozycji wiersza poleceń komputera osobistego umożliwiającego transformację współrzędnych z jednego układu do drugiego. Transformacja powinna zostać przeprowadzona na pliku tekstowym, a rezultatem powinien być inny plik tekstowy posiadający współrzędne punktów z pliku pierwotnego w innym, podanym przez użytkownika programu, układzie.

```
Współrzędne geocentryczny ECEF stacji pmanentnej GNSS
Obserwatorium Astronomiczno-Geodezyjne w Józefosławiu
X[m]      Y[m]      Z[m]
#-----
3782685.000,1084445.000,5002930.000
3664940.510,1409153.580,5009571.167
3664940.520,1409153.570,5009571.167
3664940.530,1409153.560,5009571.168
3664940.520,1409153.590,5009571.170
3664940.514,1409153.584,5009571.166
3664940.525,1409153.575,5009571.166
3664940.533,1409153.564,5009571.169
3664940.515,1409153.590,5009571.170
3664940.514,1409153.584,5009571.169
3664940.515,1409153.595,5009571.169
3664940.513,1409153.584,5009571.171
```

Rysunek 1: Przykładowy plik wejściowy dla transformacji xyz2blh

```
Podane współrzędne kartezjańskie XYZ zamienione na
geocentryczne phi, lambda, height.
phi[deg] lam[deg] h[m]
#-----
51.999969,15.996886,163.397
52.097272,21.031533,141.400
52.097272,21.031533,141.403
52.097272,21.031533,141.408
52.097272,21.031533,141.410
52.097272,21.031533,141.402
52.097272,21.031533,141.407
52.097272,21.031533,141.411
52.097272,21.031533,141.407
52.097272,21.031533,141.405
52.097272,21.031533,141.408
52.097272,21.031533,141.406
```

Rysunek 2: Przykładowy plik wyniku dla transformacji xyz2blh

## 2 Wykorzystane narzędzia i programy

Do napisania skryptu, który następnie będzie uruchamiany z użyciem wiersza poleceń użyliśmy darmowego środowiska interpretacji języka Python dystrybuowanego przez Anaconda. Do wstępnego wykrycia źródeł pojawiających się błędów wspomagaliśmy się systemem sztucznej inteligencji od OpenAI "ChatGPT". Użycie tego internetowego narzędzia pomagało w zlokalizowaniu i wyeliminowaniu błędów jakie pojawiały się podczas tworzenia skryptu. Dokumentację wyników ćwiczenia utworzyliśmy w oprogramowaniu LaTeX, wykorzystując zintegrowane środowisko TeXstudio.


### 3 Przebieg ćwiczenia

Pracę nad programem rozpoczęliśmy od wglądu do przykładowego programu otrzymanego od mgr. inż. Andrzeja Szeszko zawierającego utworzoną klasę o nazwie "Transformacje", która posiadała już zaimplementowaną transformację współrzędnych kartezjańskich do współrzędnych geocentrycznych, oraz utworzenie obiektu o nazwie "geo", który posiadał cechy klasy Transformacje.

Późniejszy kod został utworzony w oparciu o rozwijanie tego skryptu. Pracę własną zaczęliśmy od zaimplementowania do klasy Transformacje pozostałych transformacji współrzędnych podanych w instrukcji do ćwiczenia. Pierwszą zaimplementowaną transformacją był algorytm odwrotny do algorytmu Hirvonena, czyli transformacja ze współrzędnych geocentrycznych do kartezjańskich. Nasz kod do tej transformacji bazował na algorytmie napisanym do jednego z ćwiczeń z semestru trzeciego na zajęciach Geodezji Wyższej. Funkcja wymagała tylko lekkich poprawek w postaci pobierania wartości długości półosi oraz kwadratu mimośrodów z klasy, a nie z podanej przez użytkownika wartości. Była to jedyna transformacja, która za pierwszym razem zwracała plik wyjściowy za pierwszym razem bezbłędnie. W pierwszej wersji po wywołaniu funkcji z pozycji wiersza poleceń program przyjmował jedynie jedną nazwę pliku wejściowego i nie przyjmował żadnej z flag poza nazwą transformacji.

Kolejną dodaną transformacją była transformacja ze współrzędnych kartezjańskich do współrzędnych topocentrycznych. Była to transformacja która sprawiła największe trudności do zaimplementowania. Kod do tej transformacji na początku pozyskaliśmy również z ćwiczenia z Geodezji Wyższej z semestru trzeciego. Pierwotnie zdecydowaliśmy się na rozwiązanie, aby plik tekstowy przyjmował wartości współrzędnych odbiornika różne dla każdego współrzędnego nadajnika. Jeden wiersz w pliku tekstowym miał wtedy sześć kolumn. Wymagałoby to od użytkownika podania w pliku tekstowym dla każdego punktu innych współrzędnych, jednak gdy rozważamy sytuację, że mamy jeden odbiornik i wiele nadajników, rozwiązanie to byłoby bardzo wymagające dla użytkownika we wpisaniu tej samej wartości wiele razy. Ze względu na to, zdecydowaliśmy się na zmianę kodu oraz szablonu na którym się wzorowaliśmy. Wzorzec kodu wzięliśmy z ćwiczeń przedmiotu Geodezji Satelitarnej z semestru czwartego. Druga, zarazem finalna, wersja tej części skryptu pozyskuje z pliku tekstowego jedynie wartości współrzędnych nadajników, a użytkownik przy inicjalizacji programu w wierszu poleceń oczekuje podania przez użytkownika wartości współrzędnych odbiornika. Naszym zdaniem, to rozwiązanie jest bardziej intuicyjne i prostsze do zrozumienia przez użytkownika a zarazem odnosi się do zastosowania tej transformacji, dla którego w większości się ją wykorzystuje.

Zaimplementowanie do klasy transformacji na współrzędne w odwzorowaniu PL2000 oraz PL1992 nie sprawiło większych problemów. Na początku zwracanie błędne wartości zostały poprawione poprzez zaimplementowanie do kodu liniiki zamieniającej wartości pozyskiwane z pliku które są w stopniach na wartości w radianach.



```
deltalb = 1 - radians(lb0)
```

Rysunek 3: Naprawienie błędu poprzez dodanie funkcji transformującej wartość kątową na radiany.

Po tym, jak w klasie Transformacje znajdowały się już wszystkie potrzebne nam do ćwiczenia

funkcje transformacji, przeszliśmy do części która będzie reagowała na to, co poda użytkownik przy uruchomieniu programu. na początku jedynym obsługiwanym modelem elipsoidy był model WGS84, w późniejszym czasie dodaliśmy funkcję warunkową "if", przy pomocy której program zaczął reagować na podawane przez użytkownika flagi, co umożliwiło wybór modelu elipsoidy. Na tym etapie dodany został model elipsoidy Krasowskiego do klasy Transformacje. Jako że źródła podane na zajęciach nie zawierały informacji o mniejszej półosi tej elipsoidy, najpierw próbowaliśmy wyliczyć ową wartość przy użyciu wzoru 1, co jednak przez dłuższy czas dawało nam błędny wynik. Ze względu na to zdecydowaliśmy się na pozyskanie danych ze strony Szczecińskiego uniwersytetu.

$$e^2 = \frac{a^2 - b^2}{a^2} \quad (1)$$

Zdecydowaliśmy się na dodanie jedynie modelu tej elipsoidy kompatybilnej z transformacją do odwzorowania PL2000.

Na kolejnym etapie zmieniliśmy argument, jaki funkcja "open" pobierała jako plik wejściowy ze współrzędnymi. Od tego momentu argumentem stała się nazwa podana przez użytkownika w postaci flagi, a nie ogólnie narzucona w kodzie.

```
# XYZ TO BLH
coords_plh = []
with open('wsp_int_XYZ.txt') as f:
    lines = f.readlines()
    lines = lines[header_lines:]
```

Rysunek 4: Poprzednia wersja przyjmowania pliku wejściowego.

```
# XYZ TO BLH
coords_plh = []
with open(inp_file_path) as f:
    lines = f.readlines()
    lines = lines[header_lines:]
```

Rysunek 5: Aktualna wersja przyjmowania pliku wejściowego.

Po sprawdzeniu, że na tym etapie funkcja poprawnie zwraca pliki tekstowe z rezultatem naszej transformacji, dodaliśmy warunki "raise" dla wyboru modelu elipsoidy oraz transformacji, które przy wprowadzeniu błędnych flag naprowadzają użytkownika na to, jakich wartości użyć, aby funkcja działała poprawnie. Dodaliśmy również klauzulę, która przy poprawnym wykonaniu całego skryptu informuje o utworzeniu pliku z rezultatem oraz nazwę takowego pliku, alby ułatwić użytkownikowi jego zlokalizowanie.

Jeżeli chodzi o pakiety, jakie musi posiadać użytkownik, zdecydowaliśmy się używać podstawowej biblioteki pythona "math", która sprawdzała się przy transformacjach wszystkich oprócz transformacji ze wsp kartezjańskich na topograficzne. Nie jesteśmy w stanie stwierdzić, czemu współrzędne wychodziły błędne wykorzystując tylko bibliotekę math. Brak innego pomysłu zmusił nas na zaimplementowanie biblioteki numpy. Wymaga to od użytkownika zainstalowania tej biblioteki na komputerze osobistym, aby program działał poprawnie, jednak nie znaleźliśmy innego, lepiej działającego rozwiązania. Choć jedyne wykorzystanie tej biblioteki to wykorzystanie funkcji array, to działa ona idealnie dla naszej transformacji do współrzędnych topograficznych.

```
R = array([[-sin(lam), -sin(phi)*cos(lam), cos(phi)*cos(lam)],
           [cos(lam), -sin(phi)*sin(lam), cos(phi)*sin(lam)],
           [0, cos(phi), sin(phi)]])
```

Rysunek 6: Użycie funkcji array z biblioteki numpy wewnątrz skryptu.

Do sprawdzenia poprawności otrzymanych wyników wykorzystaliśmy dane oraz rezultaty z ćwiczeń z poprzednich semestrów, które zostały sprawdzone i zatwierdzone przez prowadzących. Dokładność naszych wyników przedstawiona została do milimetra dla wartości w metrach oraz sekundy kątowej dla wartości kątowych. Z taką dokładnością nasze wyniki były zgodne z tymi wzorcowymi.

Porównanie wyników z naszego programu oraz z wynikami z semestru trzeciego na tych samych danych	
Współrzędne kartezjańskie X: 3782685.0, Y: 1084445.0, Z: 5002930.0	
Wynik xyz2blh z naszego programu	Wynik xyz2blh z ćwiczenia z sem 3
51°59'59"	51°59'59"
15°59'48"	15°59'48"
163.397 m	163.397 m

Jak zaprezentowaliśmy w tabeli powyżej, obydwa zestawy wartości są takie same zachowując oczekiwaną dokładność.

Przebieg naszej pracy i kolejne poprawione wersje kodu oraz instrukcji przesyłaliśmy do zdalnego repozytorium GitHub wraz z komentarzem do wprowadzonych zmian. Zapewniło nam to możliwość pracy z dala od siebie, o różnych porach i czytelność dokonanych postępów. Dodatkowo, w wypadku wprowadzenia niewłaściwej zmiany byliśmy w stanie w łatwy sposób przywrócić zmieniony plik.

## 4 Podsumowanie

### 4.1 Rezultat

[https://github.com/SergeantMcWolfie/projekt\\_git\\_1.git](https://github.com/SergeantMcWolfie/projekt_git_1.git)

### 4.2 Nabyte umiejętności

- współpraca w wieloosobowym zespole z wykorzystaniem systemu kontroli wersji git
- pisanie kodu obiektowego w Pythonie
- implementowanie algorytmów pochodzących ze źródeł zewnętrznych (tj. takich, których nie wymyśliliśmy sami)
- tworzenie dokumentów w latex
- współpraca w wieloosobowym zespole z wykorzystaniem systemu kontroli wersji git
- tworzenie narzędzi w interfejsie tekstowym (cli) potrafiących przyjmować argumenty przy wywołaniu
- pisanie użytecznej dokumentacji

### 4.3 Zauważone błędy

Przy transformacji z układu kartezjańskiego do odwzorowania PL1992 wartości mogą wychodzić błędne ze względu na to, że dla różnych odwzorowań wykorzystywane są różne wartości elipsoidy Krasowskiego, użyta przez nas elipsoida jest kompatybilna z odwzorowaniem PL2000. Różnice nie są znaczące względem wartości prawdziwych, jednak nie zachowują dokładności milimetrowej, oczekiwanej przez nas w rozwiązaniu.

## 4.4 Bibliografia

Parametry elipsoid:

[http://uriasz.am.szczecin.pl/naw\\_bezp/elipsoida.html](http://uriasz.am.szczecin.pl/naw_bezp/elipsoida.html)

Pomoc w znalezieniu błędów skryptu:

<https://chat.openai.com>

Praca w zdalnym repozytorium:

Plik z komendami do git otrzymany na zajęciach

Transformacja współrzędnych do układu topocentrycznego:

<https://notatek.pl/>

transformacja-wspolrzednych-geocentrycznych-odbiornika-do-wspolrzednych-topocentrycznych

Internetowy edytor równan w LaTeX:

<https://latex.codecogs.com/eqneditor/editor.php?lang=pl-pl>

Materiał źródłowy do utworzenia skryptów transformacji blh2xyz, bl2two, bl2nine:

Skrypty do ćwiczeń z przedmiotu Geodezja Wyższa autorstwa dr inż. Dominika

Próchniewicza, sem. akademicki 2020/21

Materiał źródłowy do utworzenia skryptów transformacji xyz2neu:

Skrypt do ćwiczeń z przedmiotu Geodezja Satelitarna autorstwa dr hab inż. Ryszarda

Szpunara