INSTYTUT ŁĄCZNOŚCI
PAŃSTWOWY INSTYTUT BADAWCZY

Praktyki studenckie 2019 – obliczanie pozycji smartphone'a na podstawie raw measurements systemu Galileo.

Zadania do wykonania:

1. Zapoznanie się z dokumentem European GNSS Agency (GSA) „Using GNSS raw measurements on Android devices".

2. Zapoznanie się z przygotowywaną w Instytucie Łączności aplikacją zapisującą pomiary systemu Galileo wykonywane przez smartphone z systemem Android.

3. Implementacja programu, który na podstawie zebranych pomiarów systemu Galileo oblicza pozycję odbiornika o Preferowany język programowania: Java.

4. Przygotowanie testów jednostkowych wytworzonego oprogramowania.

5. Przygotowanie dokumentacji oprogramowania oraz raportu z prac.

Źródła:

- GSA White Paper „Using GNSS raw measurements on Android devices",
- https://www.gsa.europa.eu/gnss-applications/gnss-raw-measurements,
- Dokumentacja aplikacji zapisującej pomiary systemu Galileo,
- https://developer.android.com/guide/topics/sensors/gnss.

Obliczanie pozycji smartphone’a na podstawie raw measurements systemu Galileo.

RAPORT KOŃCOWY

1. **Treść wykorzystana w celu realizacji projektu:**

https://tiny.pl/tfhkk

https://tiny.pl/tfhk2

https://www.gsa.europa.eu/gnss-applications/gnss-raw-measurements

https://en.wikipedia.org/wiki/RINEX

https://gpsd.gitlab.io/gpsd/NMEA.html#_gll_geographic_position_latitude_longitude

https://www.gsa.europa.eu/system/files/reports/gnss_raw_measurement_web_0.pdf

https://tiny.pl/tfhkz

https://tiny.pl/tfh2q

https://www.ngs.noaa.gov/CORS/RINEX211.txt

https://github.com/google/gps-measurement-tools

http://ejml.org/wiki/index.php?title=Main_Page

2. **Kosmiczny segment systemu Galileo:**

Świadczenie pierwszych usług rozpoczął w grudniu 2016 roku. Jest największym konkurentem dla systemu wyznaczania pozycji GPS. Oferuje usługi związane z określaniem położenia i czasu dla około 400 milionów użytkowników. Składać się będzie z 30 satelitów (27 satelitów operacyjnych i 3 zapasowych), równomiernie rozmieszczonych na 3 orbitach na wysokości około **23 222 km**. Szacuje się, że system satelitarny Galileo osiągnie swą pełną funkcjonalność w 2020 roku.

3. Aplikacje wykorzystane do realizacji projektu:

Aplikacje **GnssLogger**(aplikacja służąca do pobierania Raw GNSS Measurements) oraz **rinex ON** (niezależny od odbiornika format wymiany danych) zostały uruchomione i przetestowane na telefonie SONY XPERIA XZ3. Wynikowe dane uzyskane za pomocą wyżej wymienionych aplikacji zostały przesłane drogą mailowa na komputer oraz zostały podstawione do zmiennych w programie do obliczania pozycji.

4. Działanie programu krok po kroku:

Początkowo należało wyznaczyć wartość pseudoodległości.

Pseudoodległość - odległość odbiornika od satelity, wyliczana na podstawie pomiaru czasu potrzebnego na przebycie przez sygnał nadawany z satelity drogi miedzy nią a odbiornikiem. Pomiar czasu przebiegu sygnału obarczony jest rożnego rodzaju błędami, dlatego wynik pomiaru odległości nazywamy **pseudoodległością**.

Obliczona została wartość t_{RXGNSS} na podstawie wartości z dokumentu dostarczonego za pomocą GnssLogger(tj. *TimeNanos*, *TimeOffsetNanos, FullBiasNanos, BiasNanos*). Wartość tą wyznacza metoda `trxGNSS`**:**

```
@Override
public double trxGNSS(double TimeNanos, double TimeOffsetNanos, double FullBiasNanos, double BiasNanos) {
    double TrxGNSS = TimeNanos + TimeOffsetNanos - (FullBiasNanos + BiasNanos);
    System.out.println("TrxGNSS: " + TrxGNSS);
    return TrxGNSS;
}
```

Następnie należało przypisać wartość *ReceivedSvTimeNanos* do zmiennej t_{Tx}. W następnym kroku należało obliczyć pomiary czasu t_{RX} z TOW decoded status lub E1C 2nd code status, w zależności od czasu trwania t_{Tx}. Jeżeli czas trwania t_{Tx} jest większy niż 100milisec to obliczamy wartość zmiennej t_{RX} z metody `trxTowDecoded1` lub `trxTowDecoded2` (dwie różne metody wyznaczania):

```
//Trx measurment time for Galileo with TOW decoded counter, first method
@Override
public double trxTowDecoded1(double TrxGNSS, double NumberNanoSecondsPerWeek) {
    double trx = TrxGNSS % NumberNanoSecondsPerWeek;
    System.out.println("trx TOW1: " + trx);
    return trx;
}
```

```java
//Trx measurment time for Galileo with TOW decoded counter, second method
@Override
public double trxTowDecoded2(double TrxGNSS, double weekNumberNanos) {
    double trx = TrxGNSS - weekNumberNanos;
    System.out.println("trx TOW2: " + trx);
    return trx;
}
```

Gdzie weekNumberNanos zostało wyznaczone z metody:

```java
//couting the number of nanoseconds that have occured from the beginning of GPS time to the current WN
@Override
public double weekNumberNanos(double FullBiasNanos, double NumberNanoSecondsPerWeek) {
    double Nanos = Math.floor((-FullBiasNanos) / NumberNanoSecondsPerWeek) * NumberNanoSecondsPerWeek;
    System.out.println("weekNumberNanos: " + Nanos);
    return Nanos;
}
```

Dla E1C 2nd code status:

```java
//Trx measurment time for Galileo with E1C2nd decoded counter
@Override
public double trxE1C2nd(double TrxGNSS, double NumberNanoSeconds100Mili) {
    double trx = TrxGNSS % NumberNanoSeconds100Mili;
    System.out.println("trx E1C2nd: " + trx);
    return trx;
}
```

Mając podane wyżej wartości, można obliczyć pseudoodległość ze wzoru:

$$\rho = \frac{(t_{Rx} - t_{Tx})}{1E9} * c. [s].$$

W kodzie, metoda odpowiedzialna za wyznaczenie pseudorange to `pseudorange_counter`:

```
//couting the pseudorange in meters using TOW or E1C2nd Trx, if Ttx is bigger than 100mili, than TOW Trx is used
@Override
public double pseudorange_counter(double tRX1, double tRX2, double tTX, double NumberNanoSeconds100Mili, double c) {
    System.out.println("TOW if Ttx is bigger than 100milisec");

    double p;
    if (tTX > NumberNanoSeconds100Mili)
        p = ((tRX1 - tTX) / 1000000000) * c;
    else
        p = ((tRX2 - tTX) / 1000000000) * c;

    System.out.println("pseudorange: " + p);
    return p;
}
```

Korzystając z opracowania https://tiny.pl/tfhkz stworzono oraz wykonano operacje na macierzach. Na początku, korzystając z pliku RINEX odczytano informacje o przybliżonych współrzędnych punktu, oraz stworzono macierz na ich podstawie.

```
//approx position of the receiver from RINEX file
double X = 4000000;
double Y = 1000000;
double Z = 5000000;

//creating the approx position of the receiver matrix
double approxpositionofthereceiverMatrix[][] = {{X}, {Y}, {Z}, {0}};
matrix.printapproxpositionofthereceiverMatrix(approxpositionofthereceiverMatrix);
```

W następnym kroku stworzono macierz zawierającą różnice pomiędzy pseudoodległoscią odczytaną z pliku RINEX a obliczoną:

$$\begin{bmatrix} R^1 - \rho_0^1 \\ \vdots \\ R^n - \rho_0^n \end{bmatrix}$$

```
//creating the pseudorange matrix
double pseudorangematrix[][] = {{satelital.RinexPseudorange - ps1st},
                                {satelita2.RinexPseudorange - ps2st},
                                {satelita3.RinexPseudorange - ps3st},
                                {satelita4.RinexPseudorange - ps4st}};
matrix.printPseudorangeMatrix(pseudorangematrix);
```

Stworzono również macierz A, z **losowymi** danymi:

```
//creating the A matrix of the increments
double AMatrix[][] = {{1, 2, 3, 4}, {1, 5, 3, 1}, {1, 2, 9, 1}, {5, 2, 3, 1}};
matrix.printAMatrix(AMatrix);
```

W następnym kroku należy wykonać operacje na macierzach, tak jak pokazano poniżej:

$$\hat{dx} = \left[A^T A\right]^{-1} A^T L$$

Zaimplementowano różne metody do obliczania transpozycji, mnożenia macierzy oraz wykorzytsano bibliotekę ejml, dzięki której możliwe było w łatwy sposób wyznaczyć macierz odwrotną.

Metoda odpowiedzialna za wyznaczenie transpozycji:

```
//matrix transponse counter
@Override
public double[][] transponse(double transponse[][], double AMatrix[][]) {
    System.out.println("Matrix A after transponse");
    for (int i = 0; i < 4; i++) {
        for (int j = 0; j < 4; j++) {
            transponse[i][j] = AMatrix[j][i];
        }
    }
    for (int i = 0; i < 4; i++) {
        for (int j = 0; j < 4; j++) {
            System.out.print(transponse[i][j] + " ");
        }
        System.out.println();
    }
    return transponse;
}
```

Metoda odpowiedzialna za wyznaczenie wyniku mnożenia dwóch macierzy:

```
//matrix multiplication counter
@Override
public double[][] matrixMultiplication(double tranponted[][], double AMatrix[][]) {

    double[][] Multiplicated = new double[4][4];
    for (int i = 0; i < 4; i++) {
        for (int j = 0; j < 4; j++) {
            Multiplicated[i][j] = 0;
            for (int k = 0; k < 4; k++) {
                Multiplicated[i][j] += tranponted[i][k] * AMatrix[k][j];
            }
        }
    }
    return Multiplicated;
}
```

Użycia biblioteki macierzy ejml:

```
//Matrix inversion using library ejml
System.out.println("Couting inverted matrix: ");
SimpleMatrix i = new SimpleMatrix(AMultipilaction);
SimpleMatrix inverted = i.invert();
System.out.println(inverted);

//multiplication of the inverted and transponted A matrix
SimpleMatrix cMatrix =  inverted.mult(transpontedejml);

//multiplication of the upper and pseudorange matrix
cMatrix = cMatrix.mult(pseudorangesolutionM);
```

Ostatecznie, pozycja odbiornika jest wyznaczana poprzez sumę macierzy współrzędnych przybliżonych oraz obliczonej wcześniej macierzy dx przyrostów:

$$\begin{bmatrix} \hat{X} \\ \hat{Y} \\ \hat{Z} \end{bmatrix} = \begin{bmatrix} X^0 \\ Y^0 \\ Z^0 \end{bmatrix} + \begin{bmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \end{bmatrix}$$

Do wyliczenia tej sumy użyto biblioteki macierzy ejml:

```
//addition of the approximate position and upper counted matrix
SimpleMatrix dMatrix = approxPosition.plus(cMatrix);
System.out.println(dMatrix);
```

5. WNIOSKI

Algorytm obliczający pseudorange działa poprawnie, jednakowoż jedynie dla dwóch satelit pseudorange wyszedł mniej więcej dobrze (ok. 23000 km), dla dwóch pozostałych wyniki nie są poprawne, gdyż rząd wielkości w danych z pliku **GnssLoger** nie jest poprawny (38954376811111ns a 200972367072045ns). Wyznaczanie pozycji odbiornika było prowadzone wzorując się głównie na pliku https://tiny.pl/tfhkz. Ze względu na brak danych do macierzy dx przyrostów (macierz A), niemożliwe było określenie poprawności zaimplementowanego algorytmu. Jednakowoż dołożono wszelich starań, aby operacje na macierzach wykonywały się poprawnie. Największe problemy i najwięcej czasu zostało poświęcone na wyszukaniu najodpowiedniejszych dokumentów opisujących działanie systemu Galileo oraz metody wyznaczenia pozycji położenia odbiornika.