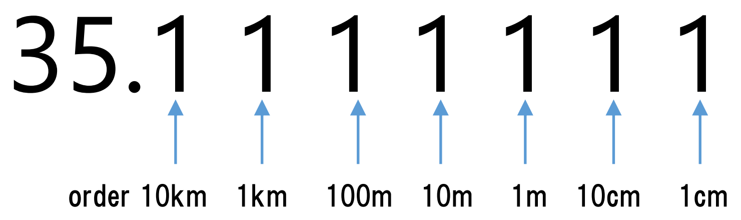
NMEA precision issue on Arduino

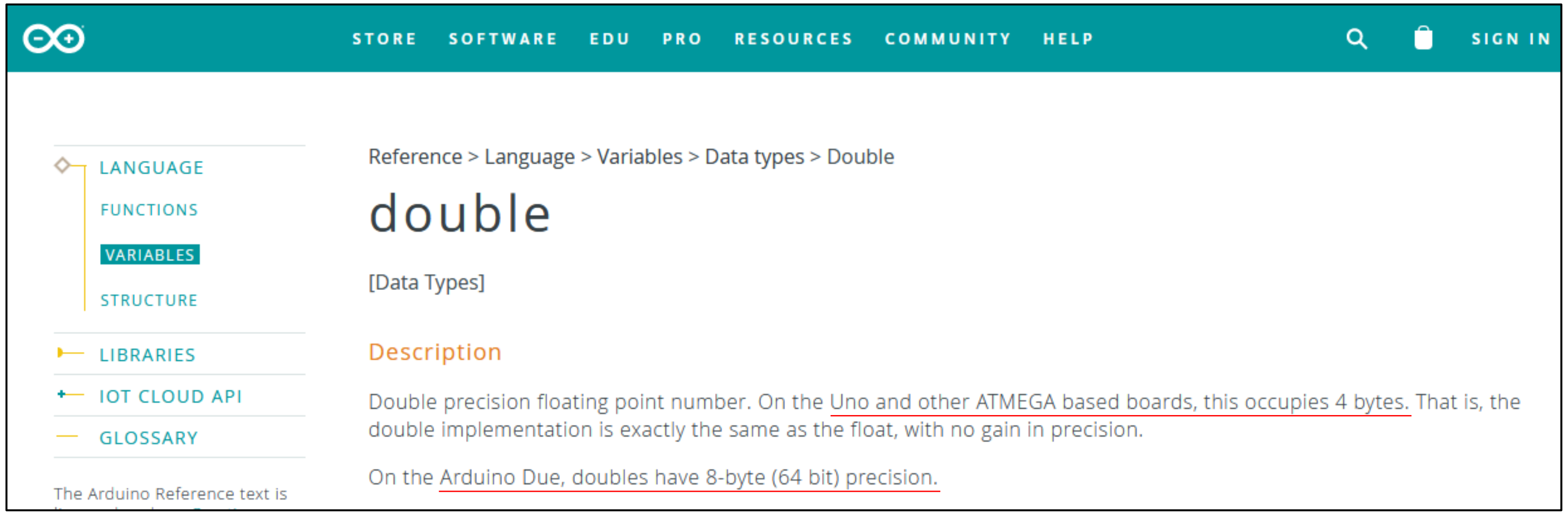
2020/07/29

1. Overview

◆Some Arduino board doesn’t support “double” data type as 8 byte and its precision is same as “float” type (4 byte).

◆4 byte latitude or longitude value don’t have enough precision (especially on RTK, PPP)
For RTK or PPP position use, at least 6 decimal place is necessary.





The Arduino Reference text is

Reference > Language > Variables > Data types > Double

double

[Data Types]

Description

Double precision floating point number. On the Uno and other ATMEGA based boards, this occupies 4 bytes. That is, the double implementation is exactly the same as the float, with no gain in precision.

On the Arduino Due, doubles have 8-byte (64 bit) precision.

1. Overview

◆If we handled NMEA with 4 byte, after 6 digit number are rounded.
The precision of output degree degrade to 10m order!

Original Lat=35.1111111,Lon=35.1111111

$GNGGA,084420,3506.666668,N,13906.666669,E,1,12,0.78,3.0,M,0.0,M,,*55

```
Serial.print("NMEA_LAT=");
Serial.println(atof(buf));
```

ASCII -> value

NMEA_LAT=3506.67, NMEA_LON=13906.7

```
pos_llh[0] = NmeallhConv(atof(buf));
```

```
float NmeallhConv(float nmeallh) {//Not used
  float deg = (int)nmeallh / 100;
  float deg2 = nmeallh - deg * 100;
  deg = deg + deg2 / 60;
  return deg;
}
```

NMEA format -> degree

Lat=35.11110687[deg], Lon=139.11109924[deg]

nonsense value

2. Solve

◆We handled degree value as 32bit integer value.

```
pos_llh[0] = (int32_t)NmeallhConv2(buf, 2); //integer high precision
```

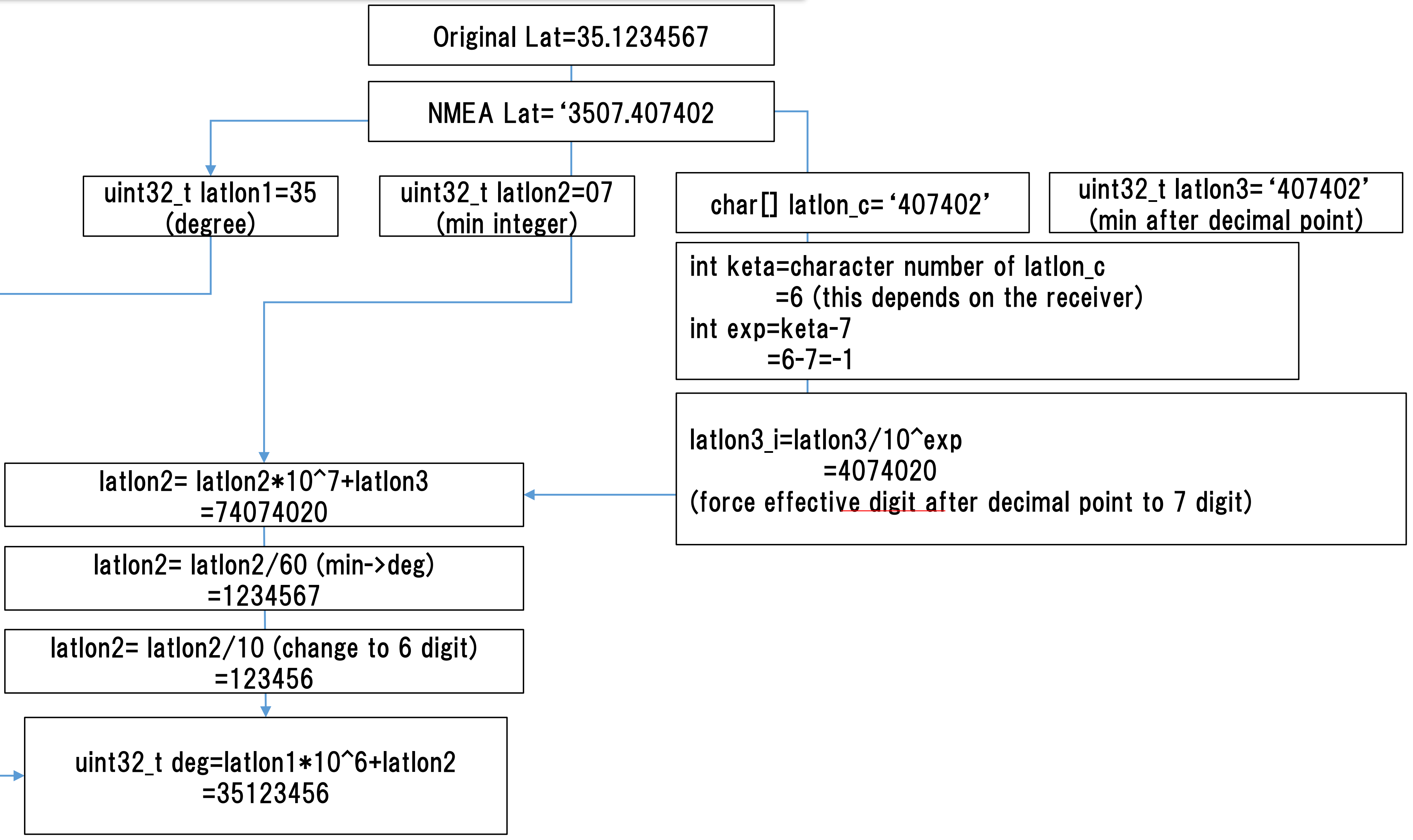
```
int32_t NmeallhConv2(char msg[]) {//NMEA->deg (int32 dd.dddddd)
  int cnt = 0;
  char latlon_c[12];
  uint32_t latlon0=0;
  uint32_t latlon1=0;//deg
  uint32_t latlon2=0;//min before decimal point
  uint32_t latlon3=0;//min after decimal point
  int32_t deg = 0;
  for (int i = 0; i < strlen(msg); i++) {
    if (msg[i] == '.') {
      latlon0=(uint32_t)(atof(latlon_c));
      latlon1=latlon0/100;//deg
      latlon2=latlon0%100;//min
      cnt=0;
      latlon_c[0]='¥0';
      continue;
    }
    latlon_c[cnt]=msg[i];
    cnt++;
    latlon_c[cnt]='¥0';
  }
  latlon3=(uint32_t)(atof(latlon_c));

  int keta = strlen(latlon_c);//after decimal point
  for (int i = 0; i < keta- 7; i++) { //7 digit
    latlon3= latlon3 / 10;
  }
  for(int i=0;i>keta- 7;i--){
    latlon3 = latlon3*10;
  }

  latlon2=latlon2*10000000+latlon3;
  latlon2= latlon2/ 60 / 10; //min->deg 6 digit
  Serial.print(latlon1);Serial.print(".");Serial.println(latlon2);

  deg = latlon1 * 1000000 + latlon2;
  return deg;
}
```

Lat=35.12345678
Lon=139.12345678
↓
int32_t lat=35123456
int32_t lon=139123456

(int32_t type range=±2147483647)

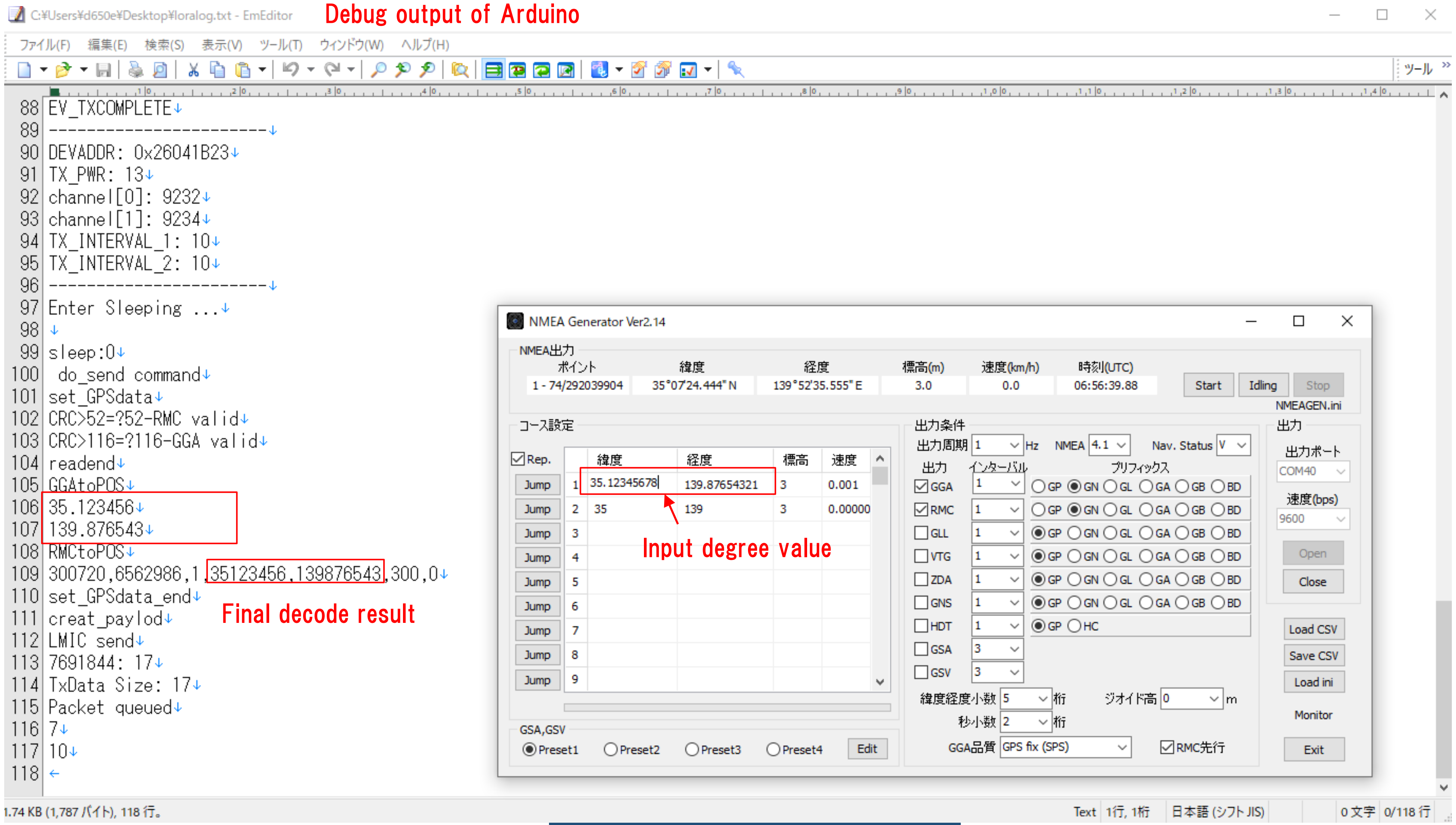
2. Solve

◆Algorithm



```
Original Lat=35.1234567
NMEA Lat= ‘3507.407402

uint32_t latlon1=35
(degree)

uint32_t latlon2=07
(min integer)

char[] latlon_c= ‘407402’

uint32_t latlon3= ‘407402’
(min after decimal point)

int keta=character number of latlon_c
         =6 (this depends on the receiver)
int exp=keta-7
        =6-7=-1

latlon3_i=latlon3/10^exp
          =4074020
(force effective digit after decimal point to 7 digit)

latlon2= latlon2*10^7+latlon3
=74074020

latlon2= latlon2/60 (min->deg)
=1234567

latlon2= latlon2/10 (change to 6 digit)
=123456

uint32_t deg=latlon1*10^6+latlon2
=35123456
```

3. Test


C:¥Users¥d650e¥Desktop¥loralog.txt - EmEditor
Debug output of Arduino
ファイル(F) 編集(E) 検索(S) 表示(V) ツール(T) ウィンドウ(W) ヘルプ(H)
ツール
Input degree value
Final decode result

```
EV_TXCOMPLETE↓
------------------------↓
DEVADDR: 0x26041B23↓
TX_PWR: 13↓
channel[0]: 9232↓
channel[1]: 9234↓
TX_INTERVAL_1: 10↓
TX_INTERVAL_2: 10↓
------------------------↓
Enter Sleeping ...↓
↓
sleep:0↓
 do_send command↓
set_GPSdata↓
CRC>52=?52-RMC valid↓
CRC>116=?116-GGA valid↓
readend↓
GGAtoPOS↓
35.123456↓
139.876543↓
RMCtoPOS↓
300720,6562986,1,35123456,139876543,300,0↓
set_GPSdata_end↓
creat_paylod↓
LMIC send↓
7691844: 17↓
TxData Size: 17↓
Packet queued↓
7↓
10↓
←
```

1.74 KB (1,787 バイト), 118 行。 Text 1行, 1桁 日本語 (シフト JIS) 0 文字 0/118 行

NMEA Generator Ver2.14

NMEA出力

ポイント	緯度	経度	標高(m)	速度(km/h)	時刻(UTC)
1 - 74/292039904	35°07'24.444" N	139°52'35.555" E	3.0	0.0	06:56:39.88

Start Idling Stop NMEAGEN.ini

コース設定 [✓ Rep.]

		緯度	経度	標高	速度
Jump	1	35.12345678	139.87654321	3	0.001
Jump	2	35	139	3	0.00000
Jump	3				
Jump	4				
Jump	5				
Jump	6				
Jump	7				
Jump	8				
Jump	9				

出力条件: 出力周期 1 Hz; NMEA 4.1; Nav. Status V

出力	インターバル	プリフィックス
[✓ GGA]	1	GP [✓ GN] GL GA GB BD
[✓ RMC]	1	GP [✓ GN] GL GA GB BD
GLL	1	[✓ GP] GN GL GA GB BD
VTG	1	[✓ GP] GN GL GA GB BD
ZDA	1	[✓ GP] GN GL GA GB BD
GNS	1	[✓ GP] GN GL GA GB BD
HDT	1	[✓ GP] HC
GSA	3	
GSV	3	

緯度経度小数 5 桁; ジオイド高 0 m; 秒小数 2 桁; GGA品質 GPS fix (SPS); [✓ RMC先行]

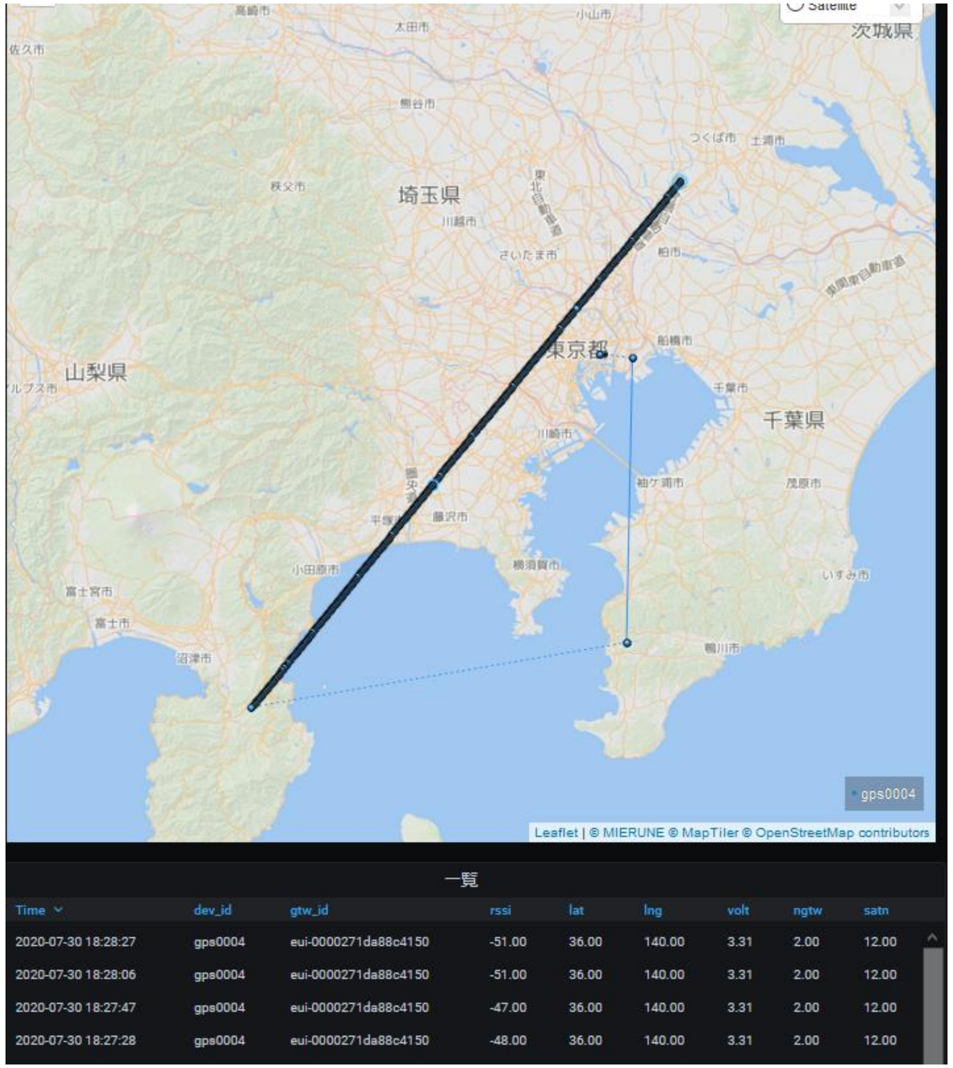
GSA,GSV: [✓ Preset1] Preset2 Preset3 Preset4 Edit

出力: 出力ポート COM40; 速度(bps) 9600; Open Close; Load CSV; Save CSV; Load ini; Monitor; Exit

3. Test

◆NMEA simulation test for bug check

From (35, 139) to (36,140)



一覧

Time	dev_id	gtw_id	rssi	lat	lng	volt	ngtw	satn
2020-07-30 18:28:27	gps0004	eui-0000271da88c4150	-51.00	36.00	140.00	3.31	2.00	12.00
2020-07-30 18:28:06	gps0004	eui-0000271da88c4150	-51.00	36.00	140.00	3.31	2.00	12.00
2020-07-30 18:27:47	gps0004	eui-0000271da88c4150	-47.00	36.00	140.00	3.31	2.00	12.00
2020-07-30 18:27:28	gps0004	eui-0000271da88c4150	-48.00	36.00	140.00	3.31	2.00	12.00