공학석사학위논문

GPS Carrier Phase / INS Integrated Smartphone Pedestrian Dead-Reckoning Using User Context Classifying Deep Learning

사용자 상황인지 딥러닝을 사용한 GPS 반송파 / 관성 센서 결합 스마트폰 보행자 항법

2019 년 2 월

서울대학교 대학원

기계항공공학부

양 서 연

GPS Carrier Phase / INS Integrated Smartphone Pedestrian Dead-Reckoning Using User Context Classifying Deep Learning

사용자 상황인지 딥러닝을 사용한 GPS 반송파 / 관성 센서 결합 스마트폰 보행자 항법

지도교수 여 재 익

이 논문을 공학석사 학위논문으로 제출함

2018 년 12 월

서울대학교 대학원

기계항공공학부

양 서 연

양서연의 공학석사 학위논문을 인준함

2018 년 12 월

위 원 장 _____

부위원장 _____

위 원 _____

Abstract

GPS Carrier Phase / INS Integrated Smartphone Pedestrian Dead-Reckoning Using User Context Classifying Deep Learning

Seo Yeon Stella Yang

School of Mechanical and Aerospace Engineering

The Graduate School

Seoul National University

In this research, the overall construction of the smartphone GPS / INS pedestrian dead reckoning system is detaily described with considering the smartphone sensor measurement properties. Also, the recent android GNSS API which can provide the raw GPS measurement is used. With carrier phase, the cycleslip compensated velocity determination is considered. As a result, the carrier phase /INS integrated pedestrian dead reckoning shows the more precise navigation accuracy than NMEA. Moreover, The deep learning approach is applied in the user context classification to change the parameters in the pedestrian dead reckoning system. The author compares the effect of several transformed inputs for the LSTM model and validate each classification performances.

Main Term: GNSS, Android GNSS API, Carrier phase, Cycleslip, INS, Extended quaternion kalman filter, Pedestrian dead reckoning, GPS / INS integration, Sensor signal deep learning, User context classification, LSTM.

Student number : 2016-20743

Contents

List of Figures

List of Tables

Chapter 1. Introduction

1.1 Motivation and Backgrounds

Smartphone becomes an essential device in this age and many users spend the much time with this machine. On the phone, there are many sensors which can detect the user's environment. The smartphone sensor data provide the knowledge to analyze the user's behaviors. The attitude and position information is one of the most important advice and is related to inertial sensors and GPS. To acquire accurate info, the smartphone sensor properties are needed to analyze deeply.

Because of the fast changing trend about the smartphone, the newest hardware/software upgrade contents must be considered. Google announced the GNSS API in 2016. After that notice, there are many trials to use this raw GNSS measurement in smartphone navigation [1],[2],[3],[4]. Moreover, Xiaomi launchs the dual frequency phone which can receive L1, L5 band signal [5],[6]. The dual frequency GNSS measurement can eliminate the ionic delay error and this can lead to precise navigation result.

To get the position info of the walking user, there are a lot of smartphone-based pedestrian dead-reckoning research is conducted so far [7],[8],[9],[10]. Many algorithms which can improve the performance of PDR is suggested. The zero velocity update and zero angular rate update method are used to enhance the accuracy in Zhang,W.'s work [11]. The map matching methods, wifi and magnetic field fingerprinting, can be integrated with PDR at an indoor environment [12],[13],[14]. About outdoor navigation, the main problem is GPS signal could be deteriorated with the city environment. To overcome the issue, Li-Ta Hsu's work

shows the 3D-GNSS pedestrian smartphone positioning in urban place [15]. Also P. Baranski used the raster map information in the urban environment navigation [16].

Today, there are machine learning and deep learning approachs to detect the user's behavior context with a smartphone inertial sensor data [17],[18],[19],[20]. Also, this user context ML/DL could be implemented in smartphone dead-reckoning. The user walking context classification can change the mode adaptively. With this variable mode, the walking detection, step counting, and stride length can find the more suitable parameters at each mode [21],[22],[23].

With this background, the author developed the smartphone GPS / INS integrated pedestrian dead reckoning system and tried to higher the performance with the raw GPS measurement.

1.2. Research Purpose and Contribution

In this research, the overall construction of the smartphone GPS / INS pedestrian dead reckoning system is detaily described with considering the smartphone's GPS, INS sensor measurement property. For this, the recent android GNSS API which can get the raw GNSS measurement is used and the API related information is given. Moreover, the carrier phase / doppler based velocity estimation in android environment is investigated. As a result, this raw GPS/ INS integration improved the position, velocity navigation accuracy. At last, the deep learning approach and the model comparision with the transformation methods could help the INS sensor data context classification study.

1.3. Contents and Methods of Research

This research analyzes the INS sensor noise and bias model properties with the experimental data from the Galaxy S8 smartphone. After the INS sensor model is determined, the pedestrian dead reckoning system is developed. Also, the smartphone GNSS API measurement is investigated and the carrier phase-based velocity determination is performed. With this information, the overall GPS/ INS integrated pedestrian dead reckoning system is constructed and the performance of the navigation system is analyzed with the contrast to NMEA GPS. At last, the user context classification is considered for the adaptive changing parameters in PDR. With the LSTM model, the 3 different transformation methods, time window cutting, STFT, and CNN encoding is compared about the prediction performance.

Chapter 2. Smartphone GPS / INS measurements analysis

In this chapter, the overall smartphone based GPS / INS pedestrian dead reckoning system and related experiment result will be explained. First, the Android smartphone environment to acquire GPS/INS measurements will be introduced. Secondly, GPS raw measurement properties and position, velocity navigation methods will be discussed. Thirdly, INS measurement properties and the attitude determination Kalman filter will be studied. Fourthly, the pedestrian dead-reckoning algorithms will be followed. Finally, I will represent developed smartphone GPS / INS integrated the pedestrian dead-reckoning system.

2.1 Smartphone GNSS measurements

2.1.1 Android Raw GNSS Measurements API

In 2016, Google announced the Android APIs for retrieving raw GNSS measurements. This APIs can be accessed after Nougat (Android 7.0) version. Also, Google supports the opensource analysis tool which can decode the raw measurement and shows the several navigation-related information.

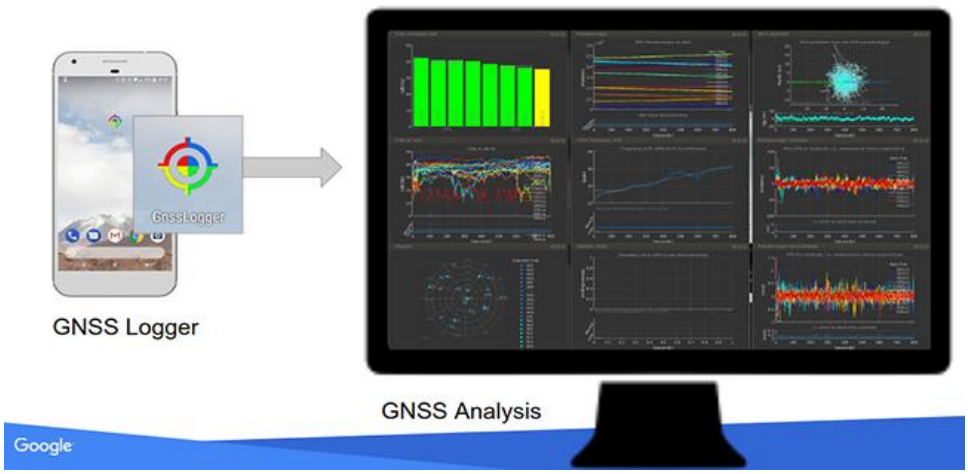


Figure 2-1. Google GNSS Analysis Opensource Tools

The accessibility of raw GNSS measurements depends on smartphone devices. The table of "Android devices that support raw GNSS measurements" from the webpage [25] shows the data dependency of each smartphone devices.
With API, the raw measurement data which can be obtained is the following table.

Table 2-1. List of Raw measurements

List of raw measurement.		
Name	Data type	remark
Raw	String	Header
ElapsedRealtimeMillis	double	
TimeNanos	int64	Nano Second
LeapSecond	int64	
TimeUncertaintyNanos	int64	Nano Second
FullBiasNanos	int64	Nano Second
BiasNanos	double	Nano Second
BiasUncertaintyNanos	double	Nano Second
DriftNanosPerSecond	double	
DriftUncertaintyNanosPerSecond	double	
HardwareClockDiscontinuityCount	double	

Svid	double	
TimeOffsetNanos	double	Nano Second
State	double	
ReceivedSvTimeNanos	int64	Nano Second
ReceivedSvTimeUncertaintyNanos	int64	Nano Second
Cn0DbHz	double	
PseudorangeRateMetersPerSecond	double	
PseudorangeRateUncertaintyMetersPerSecond	double	
AccumulatedDeltaRangeState	double	
AccumulatedDeltaRangeMeters	double	
AccumulatedDeltaRangeUncertaintyMeters	double	
CarrierFrequencyHz	double	
CarrierCycles	int64	
CarrierPhase	double	
CarrierPhaseUncertainty	double	
MultipathIndicator	double	
SnrInDb	double	
ConstellationType	double	Global Systems

In this research, we chose the “Galaxy S8” as a platform and the device has the following properties.

Table 2-2. Supported Raw measurements in Galaxy S8

Model	Ver	AGC	NM	ADR	HW clock	L5 Support	Global systems
Samsung Galaxy S8	7	no	yes	yes	yes	no	GPS
							GLONASS
							GALILEO
							BeiDou
							QZSS

(Ver: Android version, AGC: Automatic Gain Control,
NM: Navigation Message, ADR: Accumulated Delta Range)

2.1.2 Raw GPS Measurements Properties

For the GPS navigation, the raw GPS measurements properties need to be investigated. The measurement equations can be expressed as below.

Table 2-3. GPS measurement components

Pseudorange(Code) :

$$\rho_u^i = d_u^i + B_u - b^i + T_u^i + I_u^i + \partial R_u^i + M_u^i + \varepsilon_{u,\rho}^i \quad (2.1)$$

Doppler(Delta Psuedorange) :

$$f_u^i = d_u^i + B_u - b^i + T_u^i - I_u^i + \partial R_u^i + M_u^i + \varepsilon_{u,f}^i \quad (2.2)$$

Carrier Phase(Accumulated Delta-Range) :

$$\phi_u^i = d_u^i + B_u - b^i + T_u^i - I_u^i + \partial R_u^i + M_u^i + N_u^i \lambda + \varepsilon_{u,\rho}^i \quad (2.3)$$

d_u^i : Distance between i th satellite and user
B_u : Receiver clock error of the user
b^i : i th satellite clock error
T_u^i : Troposphere delay error
I_u^i : Ionosphere delay error
∂R_u^i : Satellite location error
M_u^i : Multipath error
ε_u^i : Receiver noise
λ : Wavelength of carrier phase
N_u^i : Integer ambiguity

The noise level of psuedorange is meter scale, Doppler noise level is sub-meter/sec scale and Carrier phase noise level has mm scale.

2.1.3 Smartphone NMEA Location Provider

Location manager API provides periodic reports on the geographical location of the device. There are 3 type of location providers in Android [26].

Table 2-4. Smartphone Location Providers

Provider	Characteristics	Accuracy
GPS provider (GPS, AGPS)	Name of the GPS location provider. This provider determines location using satellites. Depending on conditions, this provider may take a while to return a location fix.	20ft
	Requires the permission android.permission.ACCESS_FINE_LOCATION.	
Network provider (AGPS, CellID, WiFi MACID)	Name of the network location provider. This provider determines location based on the availability of cell tower and WiFi access points. Results are retrieved by means of a network lookup.	200ft
	Requires either of the permissions android.permission.ACCESS_COARSE_LOCATION or android.permission.ACCESS_FINE_LOCATION.	
Passive Provider (CellID, WiFi MACID)	A special location provider for receiving locations without actually initiating a location fix. This provider can be used to passively receive location updates when other applications or services request them without actually requesting the locations yourself. This provider will return locations generated by other providers.	5300ft / 1mile
	Requires the permission android.permission.ACCESS_FINE_LOCATION,	

	although if the GPS is not enabled this provider might only return coarse fixes.	

GPS provider gives the location which is determined with the triangle signal from remote satellites. Receiver module typically connects to the host system via UART but use another form of Peripheral I/O. To access the position information, the Android system needs to acquire permission. There are 2 types of permission with different location accuracy, “ACCESS_FINE_LOCATION” and “ACCESS_COARSE_LOCATION”.

GPS hardware typically reports location information as ASCII strings in the NMEA standard format. Each line of data is a comma-separated list of data values known as a sentence. While each GPS module may choose to report different portions of the NMEA protocol, most devices send one or more of the following sentences

GPGGA (Fix Information): Includes position fix, altitude, timestamp, and satellite metadata.

GPGLL (Geographic Latitude/Longitude): Includes position fix and timestamp.

GPRMC (Recommended Minimum Navigation): Includes position fix, speed, timestamp, and navigation metadata.

In 2013 Google I/O, the Fuse location provider was introduced. The fused location provider has the battery efficient property and can combine the GPS / WIFI / CELL / INS sensor information. Even more, it can achieve the approximate location at the indoor environment.

In this research, the author uses the fix information(GPGGA) of GPS provider with ACCESS_FINE_LOCATION permission as NMEA location.

2.1.4 Pseudorange Based Position Estimation

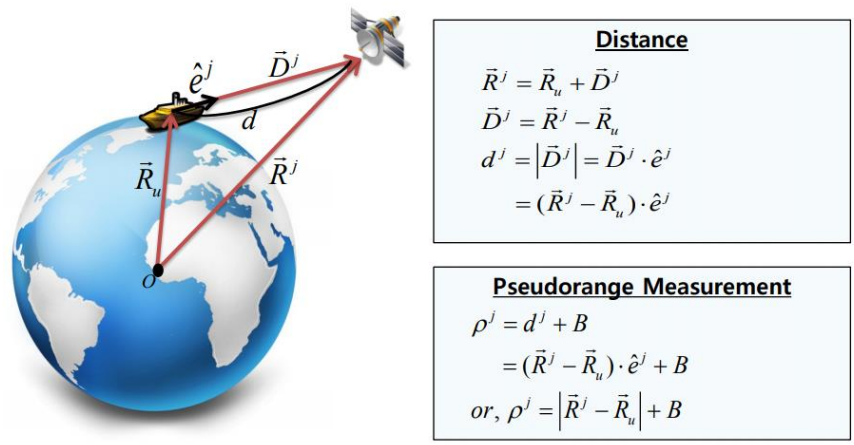


Figure 2-2. Pseudorange Standalone Positioning

About the pseudo-range measurement, the distance between the user and satellite can be expressed with the sight unit vector(user to satellite) and ECEF location vectors. Also with the assumption that the other noises are small (or eliminated), the receiver clock error is the main error of measurement.

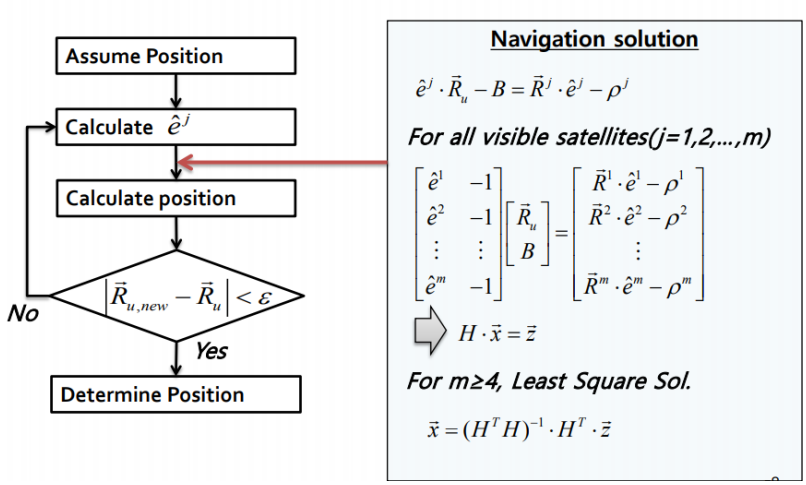


Figure 2-3. Position determination Algorithm

The position determination is performed with the least square method. Also, if the variance of the pseudo-range is considered, the weighted least square solution which is more precise can be used.

$$\vec{x} = (W \cdot H^T)^{-1} W \cdot \vec{z} \quad \text{(W : matrix with 1/(noise sigma))} \qquad (2.4)$$

2.1.5 Position Determination Experiment

To determine the position with the raw pseudorange data, the weighted least square standalone positioning method in the upper section is used. The satellite clock error is eliminated with the ephemerides information and the other error effects are assumed as small.

Using Galaxy S8 android app, the walking data is acquired at open sky playground during about 30 min, 12 times experiment. For the true reference trajectory, the precise Trimble GEO-XR device logs the position at the same time.

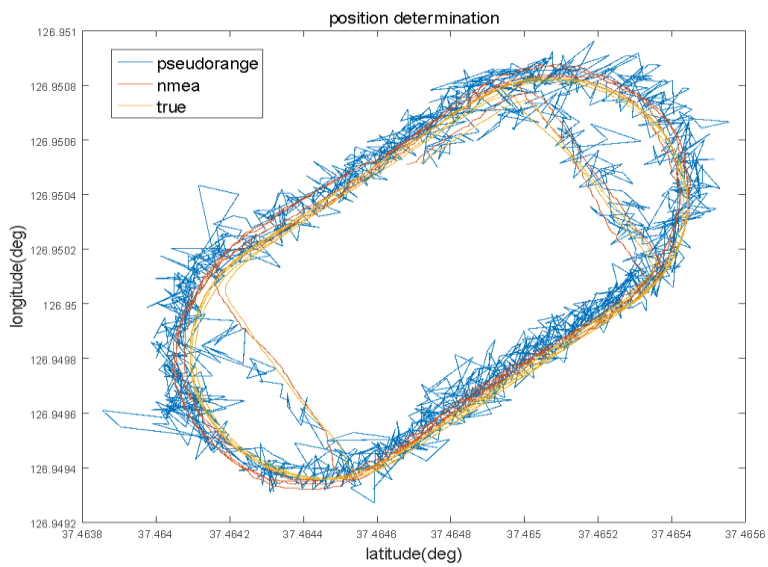


Figure 2-4. Position Determination Experiment

Table 2-5. The mean errors of the position determination

Name	Mean Error (deg)	Mean Error (deg)	Mean Error (m)	Mean Error (m)
NMEA	2.8095e-05	2.7539e-05	2.7805	2.5397
Psuedorange	6.1566e-05	5.5452e-05	9.3837	5.4730

The pseudorange positioning precision was lower than the NMEA API based. The reason would be the troposphere, ionosphere errors remain. The navigated position showed the noisy property than NMEA. But it was a dynamic and real-time situation so that the average position accuracy could be different in a static situation. To find the exact error reason, the processing code inside the NMEA smartphone GPS API is needed to be investigated more.

2.2 Smartphone INS Measurements

2.2.1 Android Sensor Manager API

Smartphone devices have many sensors which can detect the phone user's information. The INS sensor like the accelerometer, gyroscope, magnetometer, and barometer can give the smartphone attitude and position related information.

In the Android system, there is the sensor manager API [27] which can get the smartphone INS sensors data. In this research, the accelerometer, gyroscope, magnetometer sensors are used.

After the 26 level API version (8.0), Android system supports "TYPE_ACCELEROMETER_UNCALIBRATED". The former Android developer could get the calibrated accelerometer only but after ver 8.0 (Oreo), the uncalibrated one can be accessed.

Also,"TYPE_GYROSCOPE_UNCALIBRATED","TYPE_MAGNETIC_FIELD_UNCALIBRATED" is available after API level 18.

The smartphone's sensor axes are depended on hardware orientation. So that the axes could be different by smartphones. In this research, the Galaxy S8 smartphone is used and the sensor axes are the following figure.

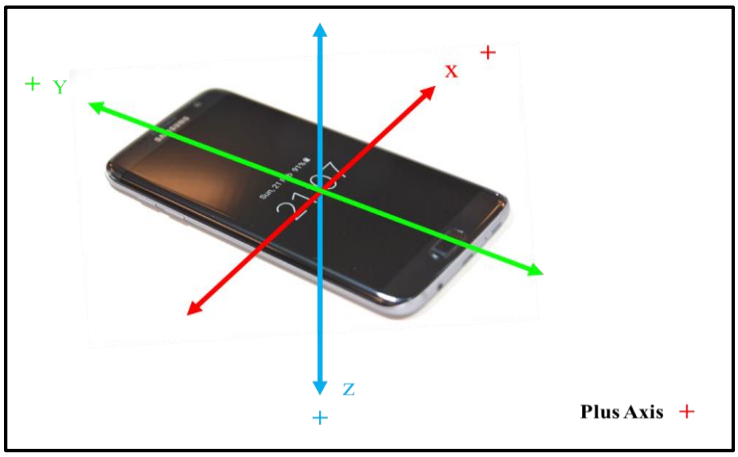


Figure 2-5. Galaxy S8 Smartphone Axis

2.2.2 INS Measurements Properties

For the navigation, the used sensor, accelerometer, gyroscope, and magnetometer, measurement models are needed to be analyzed. These INS sensor measurements have each inherent characters. The following equations illustrate the sensor measurements property [28].

Table 2-6. INS measurement equations

Accelerometer specific force	$\overrightarrow{f_m(k)} = S^f[C_i^b(k)(\vec{g} + \overrightarrow{a_{body}(k)})]^T + \overrightarrow{b_0^f} + \overrightarrow{b_r^f(k)} + \overrightarrow{v_f}$
Gyroscope angular rate	$\overrightarrow{w_m(k)} = S^w(k)\overrightarrow{w(k)} + \overrightarrow{b_0^w} + \overrightarrow{b_r^w(k)} + \overrightarrow{v_w}$
Magnetometer magnetic field	$\overrightarrow{m_m(k)} = S^m C_i^b(k)\vec{h} + \overrightarrow{b_0^w} + \overrightarrow{b_r^w(k)} + \overrightarrow{v_w}$

C_i^b : from inertial to the body frame rotation matrix

$\vec{S}$: scale factor

$\vec{b_0}(k)$: constant bias

$\vec{b_r}(k)$: random bias

$\vec{v} \sim (0, \sigma^2)$: Gaussian noise

With these measurement models, we want to know the user's phone attitude information. The sensor is attached to the smartphone's body so that the equation is expressed as the body frame. Also, there are some parameters, scale factor, constant bias and random bias, which is needed to define in the models. The scale factor is the parameter which scales the measured signal. The constant bias has character changing with each on/off time. The random bias represents the sensor's temporal, time-related quantum reaction. And the sensor noise is assumed as a normal distribution.

2.2.3 Noise level, Constant bias, Scale factor, Calibration

To know the motion related states, the upper sensor property constants need to be calibrated. In the following chapters, we will consider this calibration method in each sensor types [29].

Constant bias, Noise level: The constant bias changes randomly with the smartphone power on/off times but when power is on state, it persists the static characteristic. If the power on mode is maintained and main signal becomes zero, the fixed constant bias and the noise level can be detected. But, making the zero magnetic field is not easy. Also, the magnetometer's measurement has special property of distortion from the iron effect. By this reasons, the magnetometer need a special calibration method, ellipsoid fitting, and this calibration will be analyzed in

the next chapter separately. In this chapter, we will investigate the accelerometer and gyroscope calibration. With the assumption which the random noise is not influential, the mean constant bias, constant bias variance and noise sigma are founded by the following method.

Table 2-7. Noise sigma determination.

Accelerometer noise level	Gyroscope noise level
$f_m = b_c + w$ $Std(f_m) = Std(w) \quad (\because b_c : deterministic)$ $w \sim N(0, \sigma_w^2)$	$\omega_m = b_c + w$ $Std(\omega_m) = Std(w) \quad (\because b_c : deterministic)$ $w \sim N(0, \sigma_w^2)$

Table 2-8. Mean constant bias determination.

Accelerometer constant bias	Gyroscope constant bias
$\boldsymbol{f_m(1) = b_c(1) + w(1)}$ $\vdots$ $\boldsymbol{f_m(n) = b_c(n) + w(n)}$ $\boldsymbol{\sum f_m(i) = nb_c + \sum w(i)}$ $\boldsymbol{\tilde{b} = \frac{\sum f_m(i)}{n} = b_c + \frac{\sum w(i)}{n}}$	$\boldsymbol{\omega_m(1) = b_c(1) + w(1)}$ $\vdots$ $\boldsymbol{\omega_m(n) = b_c(n) + w(n)}$ $\boldsymbol{\sum \omega_m(i) = nb_c + \sum w(i)}$ $\boldsymbol{\tilde{b} = \frac{\sum \omega_m(i)}{n} = b_c + \frac{\sum w(i)}{n}}$

Scale factor :

If the mean constant bias is determined with the upper method, we can eliminate it. After that remove, scale factor can be calcurated.

Table 2-9. Scale factor determination.

Accelerometer Scale factor	Gyroscope Scale factor
- **Constant bias elimination** ➔ $f_m = Sf + w,\ w \sim N(0, \sigma_w^2)$, - $S = const,\quad f = g$ $\boldsymbol{f_m(1) = Sf(1) + w(1)}$ . . $\boldsymbol{f_m(n) = Sf(n) + w(n)}$ $\boldsymbol{\sum f_m(i) = nSf + \sum w(i)}$ $\boldsymbol{\tilde{S} = \frac{\sum f_m(i)}{nf} = S + \frac{\sum w(i)}{nf}}$	- **Constant bias elimination** ➔ $\omega_m = S\omega + w,\ \ w \sim N(0, \sigma_w^2)$, - $S = const$ $\boldsymbol{\omega_m(1) = S\omega(1) + w(1)}$ . . $\boldsymbol{\omega_m(n) = S\omega(n) + w(n)}$ $\boldsymbol{\sum \omega_m(i) = \sum S\omega(i) + \sum w(i)}$ $\boldsymbol{\tilde{S} = \frac{\sum \omega_m(i)}{\sum \omega(i)} = S + \frac{\sum w(i)}{\sum \omega(i)}}$

To calculate the accelerometer scale factor, we need the known acceleration. Using gravity acceleration, this scale factor can be calculated. By tilting the axis to the gravity direction, we can make the gravity acceleration. About gyroscope, the known angular rate is needed. The known angle also could be used instead (integrated angular rate with time). The author chooses the known angular rate as a reference and uses the turntable which can measure the rotation rate with the encoder and speed controller.

2.2.4. Accelerometer, Gyroscope Calibration Experiment

With the background of the above section, the calibration experiment is performed with Galaxy S8 smartphone.

To measure the white noise and constant bias, the 2 poses which make the vertical to gravity(0 acceleration at 2 axes) is used. To know the exact gravity direction, a level meter measures the horizon line. 1 Hour of the experiment is performed.

Figure 2-6. Gravity to Z − axis

Figure 2-7. Gravity to Y − axis

Mean white noise

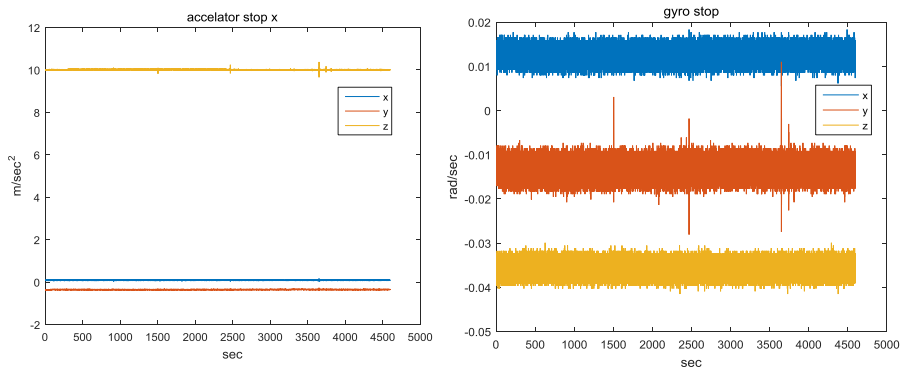


Figure 2-8. White noise experiment 1

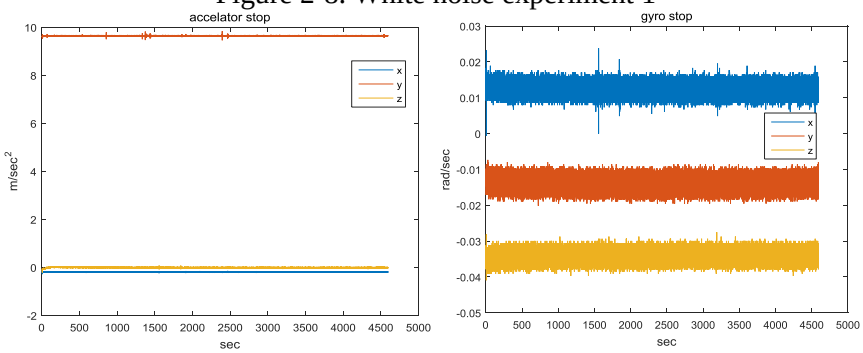


Figure 2-9. White noise experiment 2

Table 2-10. White noise experiment result

	Acc. x	Acc. y	Acc. z	Gyro x	Gyro y	Gyro z
$\sigma\_w\hat{}$	0.0094	0.0094	0.0141	0.0013	0.0015	0.0012
	0.0093	0.0094	0.0127	0.0013	0.0014	0.0012
Average	0.00935	0.0094	0.0134	0.0013	0.00145	0.0012

Constant Bias Experiment

The constant bias is varying with the power on/off times. So the 10 times of the experiment is done each 5 mins. In an accelerometer case, to eliminate the gravity effect, two poses is considered.

Accelerometer constant bias :

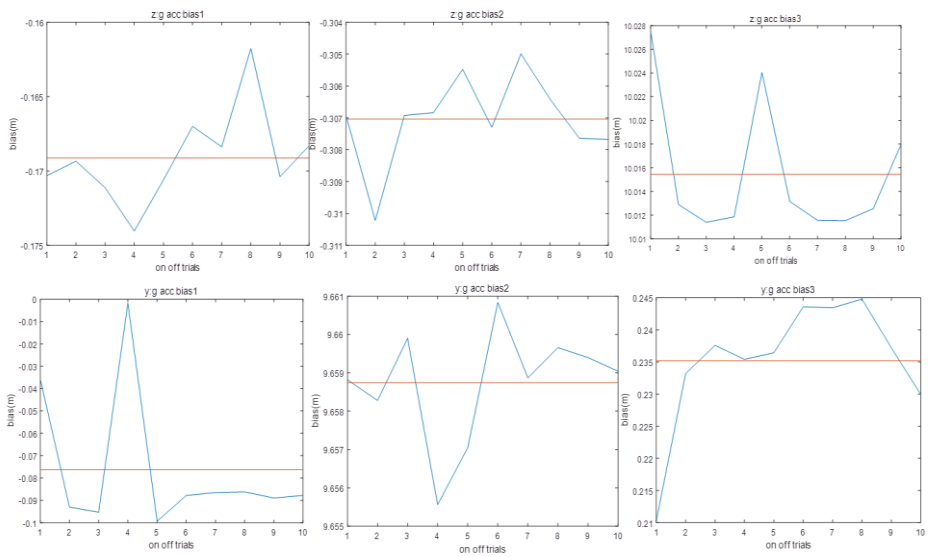


Figure 2-10. Acc. constant bias experiment

Table 2-11. Acc. constant bias experiment result

b_c	Accel. x	Accel. y	Accel. z
Pose 1 experiment Mean	−0.1691	−0.3070	−
Pose 2 experiment Mean	−0.1597	−	0.2352
Overall Mean	−0.1644	−0.3070	0.2352

Gyro Constant bias :

As a reason that gyro bias is not affected by gravity, 1 pose, 2 experiments are done.

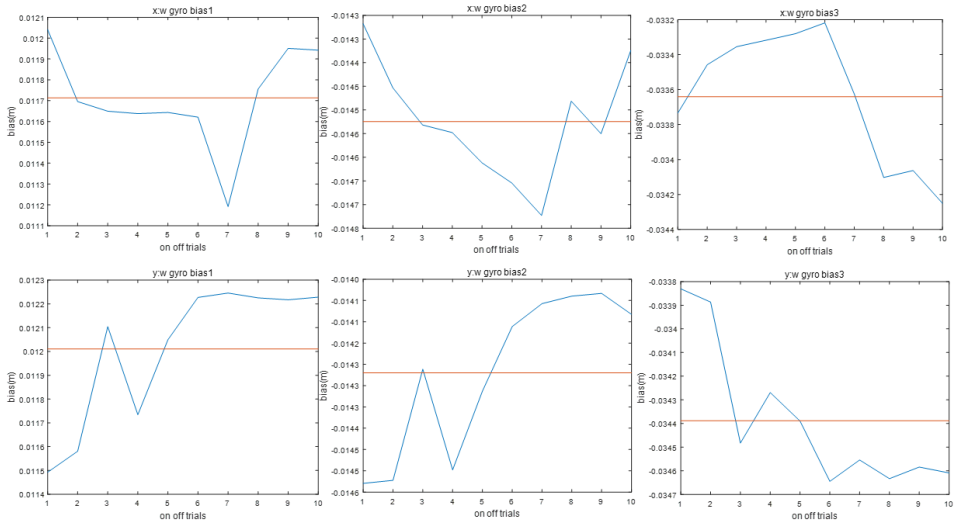


Figure 2-11. Gyro constant bias experiment

Table 2-12. Gyro constant bias experiment result

b_c	Gyro x	Gyro y	Gyro z
1 experiment Mean	0.0117	−0.0145	−0.0336
2 experiment Mean	0.0120	−0.0143	−0.0344
Overall Mean	0.01185	−0.0144	−0.034

Scale Factor experiment

With the upper section background, the scale factor is calculated.

10 time on/off experiments is done.

Accelerometer Scale factor:

The gravity acceleration is used as reference acceleration.

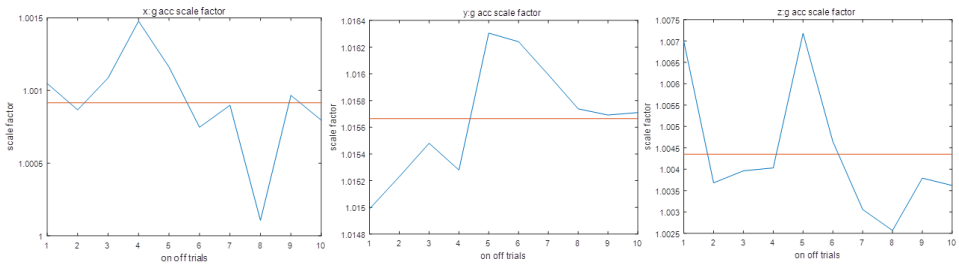


Figure 2-12. Accel. scale factor experiment

Table 2-13. Accel. scale factor experiment result

	Sx	Sy	Sz	Sσx	Sσy	Sσz
Mean Scale factor	1.0009	1.0157	1.00155	3.5386 e−04	4.3278 e−04	0.00155

Gyro Scale factor :

The angular velocity controlled turn table is used to find the gyro scale factor.

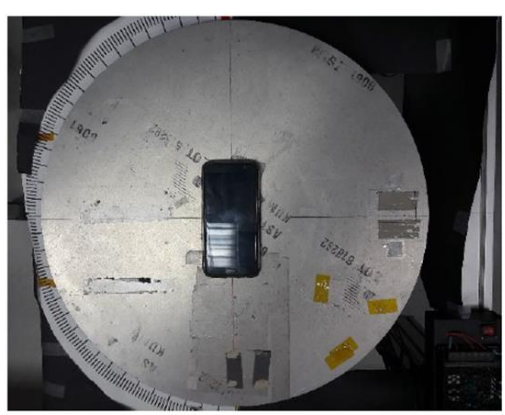


Figure 2-13. Gyro angular velocity controlled turn table

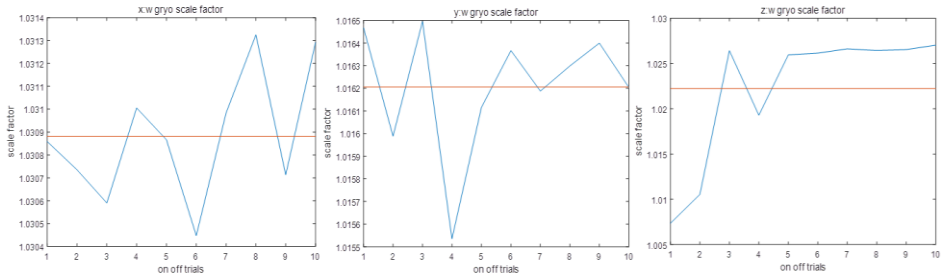


Figure 2-14. Gyro scale factor experiment

Table 2-14. Gyro scale factor experiment result

	Sx	Sy	Sz	Sσx	Sσy	Sσz
Mean Scale Factor	1.0309	1.0162	1.0222	7.9584 e−08	8.1220 e−08	5.4688 e−05

2.2.5 Magnetometer Ellipse Fitting Calibration

Magnetometer detects the earth magnetic field and can find the north pole direction. The local earth magnetic field can be predicted with the models like world magnetic model (WMM) and international geomagnetic reference filed (IGRF). But the magnetometer measurement can be influenced by the user environment.

The distortion is caused by the nearby iron environment called iron effect. The iron effect can divide as 2 types, hard iron effect, soft iron effect. The hard iron effect comes from the constant magnetic field and results the bias shift in measurement. The soft iron effect appears the ellipsoidal distortion which has different scales in the each direction.

To compensate this iron effect, the special calibration is needed. There are many methods to calibrate but the ellipse fitting calibration method is considered in this research.

The ellipse equation can be represented as follow.

$$Q(x,y,z) = ax^2 + by^2 + cz^2 + 2dxy + 2exz + 2fyz + gx + hy + kz + j = 0 \quad (2.5)$$

If the ellipse is not far displaced from the sphere the j term can set as 1.

$$Q(x,y,z) = ax^2 + by^2 + cz^2 + 2dxy + 2exz + 2fyz + gx + hy + kz + 1 = 0 \quad (2.6)$$

At least 9 attitude information is given, the 9 parameters can be calculated with the least square method.

$$u = (M^T M)^{-1} M^T y \quad (2.7)$$

$$u = [a,b,c,d,e,f,g,h,k]^T \quad (2.8)$$

$$M = \begin{bmatrix} x_1^2 & y_1^2 & z_1^2 & 2x_1y_1 & 2x_1z_1 & 2y_1z_1 & x_1 & y_1 & z_1 \\ x_2^2 & y_2^2 & z_2^2 & 2x_2y_2 & 2x_2z_2 & 2y_2z_2 & x_2 & y_2 & z_2 \\ \vdots & & & & & \vdots & & & \\ x_n^2 & y_n^2 & z_n^2 & 2x_ny_n & 2x_nz_n & 2y_nz_n & x_n & y_n & z_n \end{bmatrix} \quad (2.9)$$

$$y = [-1,-1,-1,-1,-1,-1,-1,-1,-1]^T \quad (2.10)$$

After finding the ellipse parameters, the rotation, offset(center to 0,0) calibration, and radius scale adjustment can be performed. Finally, the ellipsoid distortion can compensated for the spheric shape.

Using MATLAB, the ellipsoid calibration algorithm is performed to the magnetometer data.

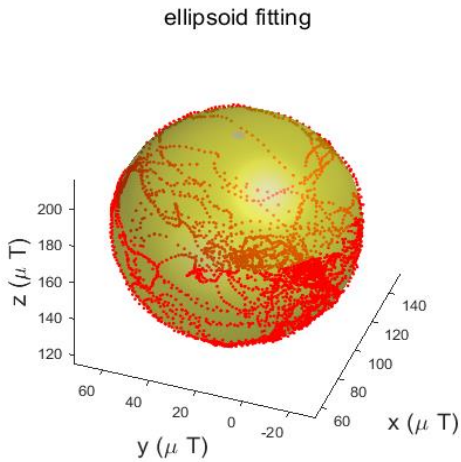


Figure 2-15. Ellipsoid Fitting

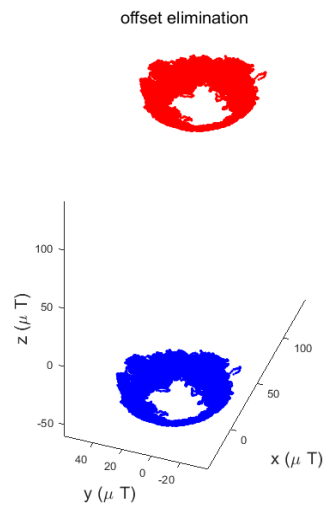


Figure 2-16. Offset Elimination
(Red : offset exist / Blue : offset is eliminated)

2.2.6 Random Bias, Allan Variance Exiperiment

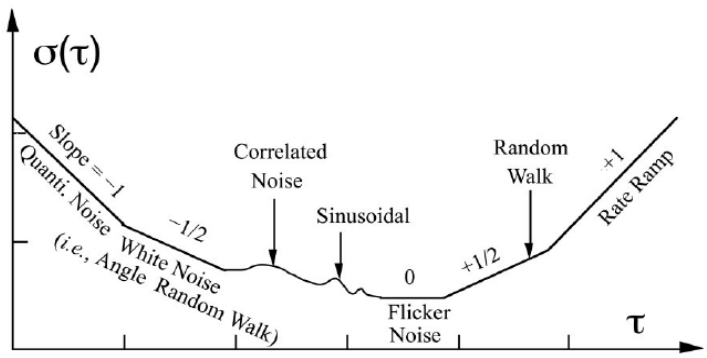


Figure 2-17. Allan variance

The random bias is the error source of the sensor measurement which has statistical changing property with time. To know the sensor's different noise types, Allan variance can be helpful.

$$
\begin{aligned}
&m_{meas}(t) = m_{\text{true}}(t) + b_1(t) + b_0 + w(t) \text{ where} \\
&w(t): \text{ (wideband) noise, } w(t) \sim N\left(0, \sigma_w^2\right) \\
&b_0: \text{ constant bias (null shift)} \\
&b_1(t): \text{ time-varying bias, } \dot{b}_1(t) = -\frac{1}{T_c} b_1(t) + w_{b1}(t), w_{b1} \sim N\left(0, q_c^2\right)
\end{aligned}
\tag{2.11}
$$

<Gauss markov process bias model>

The random bias can be modeled as gauss markov process bias. In this model, the parameter N_w, T_c, q_c can be found by the Allan variance investication.

To analyze the sensor property of Galaxy S8. 11-hour sensor data is logged. The Allan variance analysis result is present.

Gyro :

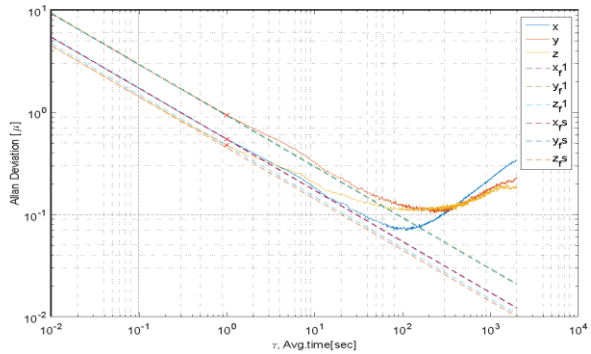


Figure. 2-18 Gyro allan variance - white noise

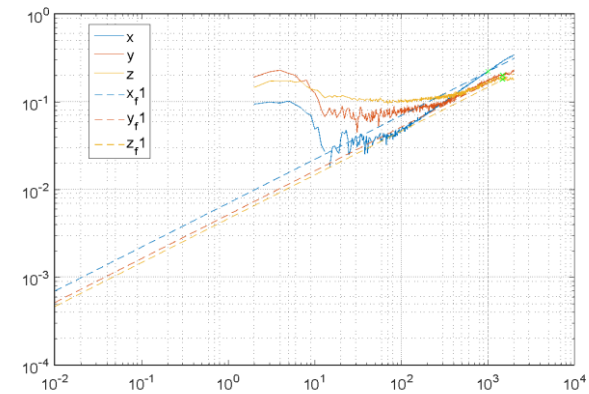


Figure. 2-19 Gyro allan variance - random walk

Table 2-15. Gyro random bias parameters

$\sigma_w(ARW)$ $([°/\sqrt{s}],[°/\sqrt{hr}])$	T_c (s)	$q_c(RRW)$ $([°/\sqrt{s}/s],[°/s^{3/2}])$
0.1242e-03	3.6016e+04	0.1665e-05
0.1412e-03	2.0633e+04	0.1530e-05
0.1260e-03	1.1997e+04	0.1534e-05

Accelerometer :

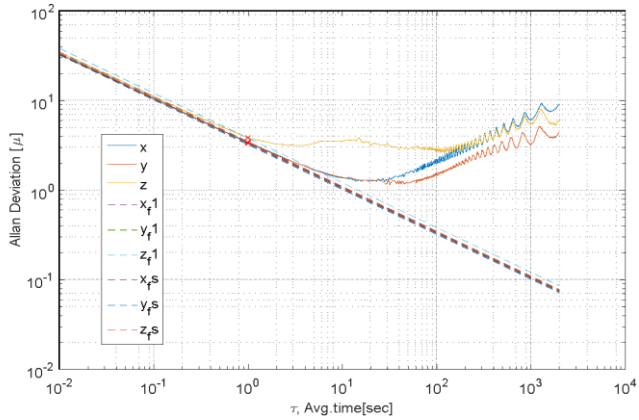


Figure. 2-20 Accelerometer allan variance - white noise

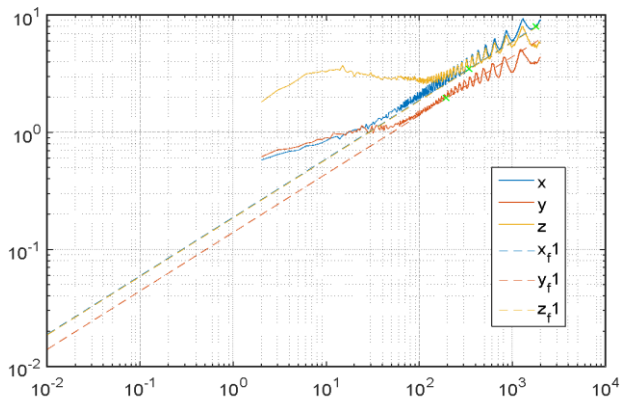


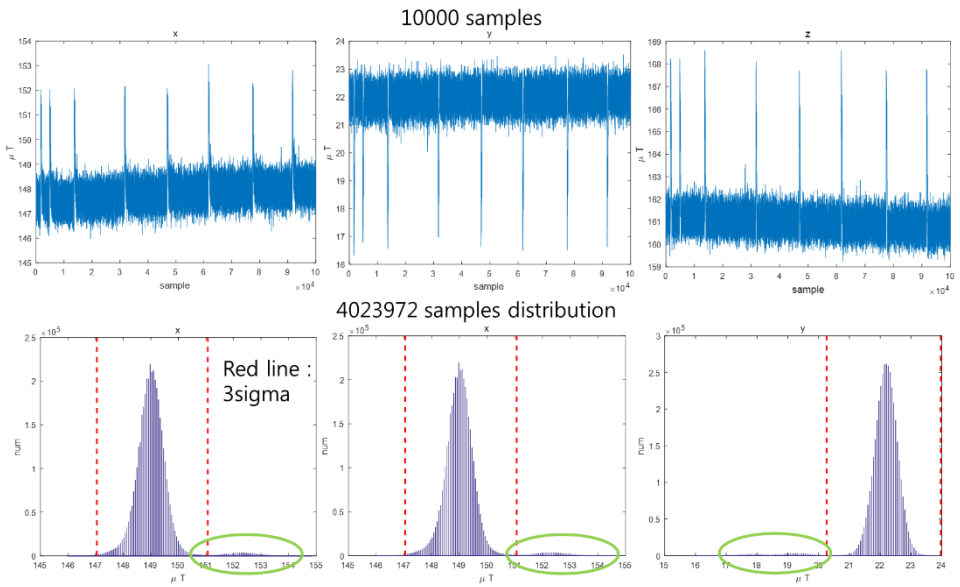
Figure. 2-21 Accelerometer allan variance - random walk

Table 2-16. Accel. random bias parameters

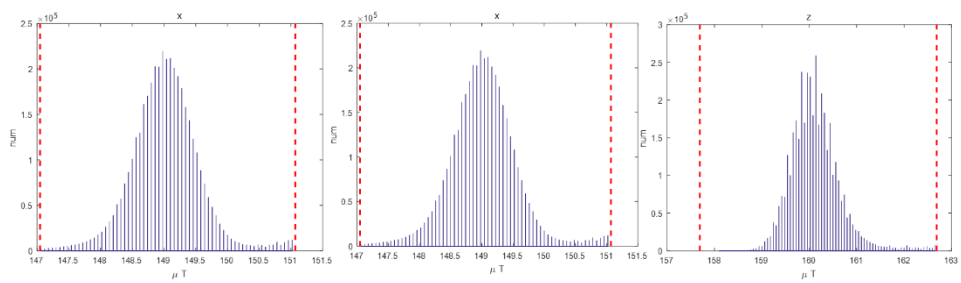
$\sigma_w(ARW)$ $([°/\sqrt{s}],[°/\sqrt{hr}])$	T_c (s)	$q_c(RRW)$ $([°/\sqrt{s}/s],[°/s^{3/2}])$
0.0010	5.7443e+03	0.1320e-03
0.0010	3.9452e+03	0.1305e-03
0.0012	6.3465e+03	0.1715e-03

Magnetometer :

When the Galaxy S8 smartphone raw magnetometer is received, the measurement shows strange ticks. So the noise histogram was not gaussian distribution so the preprocessing process was needed before the allan variance analysis.



Before preprocessing distribution



After preprocessing distribution

Figure. 2-22 Magnetometer Preprocessing

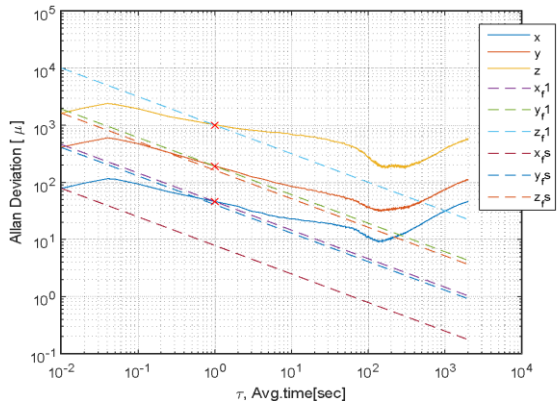


Figure 2-23. Magnetometer Allan variance - white noise

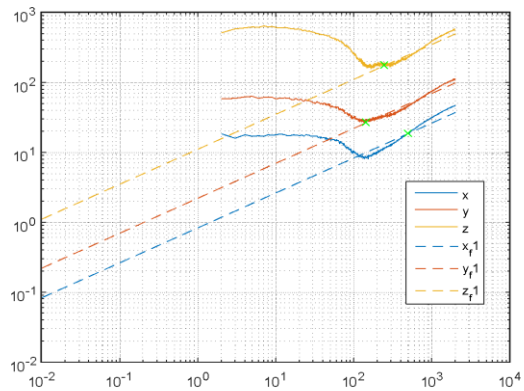


Figure 2-24. Magnetometer Allan variance - random walk

Table 2-17. Magnetometer random bias parameters

$\sigma_w(ARW)$ $([°/\sqrt{s}],[°/\sqrt{hr}])$	T_c (s)	$q_c(RRW)$ $([°/\sqrt{s}/s],[°/s^{3/2}])$
0.0933	1.6202 e+05	0.0022
0.0926	0.2364 e+05	0.0008
0.1132	1.3894 e+05	0.0015

The allan variance analysis shows the white noise and random walk property. With this sensor property, the bias is modeled. But, the magnetometer gauss markov process is not well implemented in kalman filter empirically. So that the random walk bias model is use in the filter.

2.3 Developed Android Smartphone App

The author developed the Android app for logging the INS sensor and GPS data. The Android Studio is used as IDE. The Android App UI is following the figure.

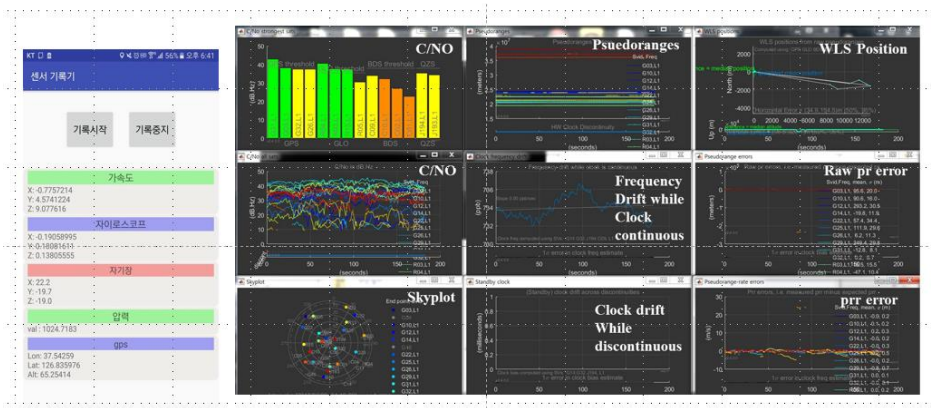


Figure 2-25. Developed Android App UI and Raw GNSS parsing

This App can log the uncalibrated accelerometer, gyroscope, magnetometer and barometer data (which is not used) with 100hz sampling rate. To acquire the constant frequency, the asynchronized task coding is applied. Also the power saved, doze mode are blocked to avoid the logging speed change. Also, it can log the NMEA GPS data and raw GNSS data as 1hz. The raw GNSS code reference the Google GPS logger opensource and is integrated with the developed app. The data is saved as .txt format in the smartphone local built-in memory. Moreover, the additional time-synchronizing preprocessing is needed because the raw GPS data is logged in separated .txt and has the different time formnat(GPS Time). The time synchronization is performed by the matching the smartphone system clock in each time data. The Galaxy S8 smartphone supports the other GNSS system (GLONASS, BEIDO, QZSS) satellites data but the research only dealt the GPS satellites for the navigation.

Chapter 3. Pedestrian Dead Reckoning

3.1 Pedestrian Dead-Reckoning System

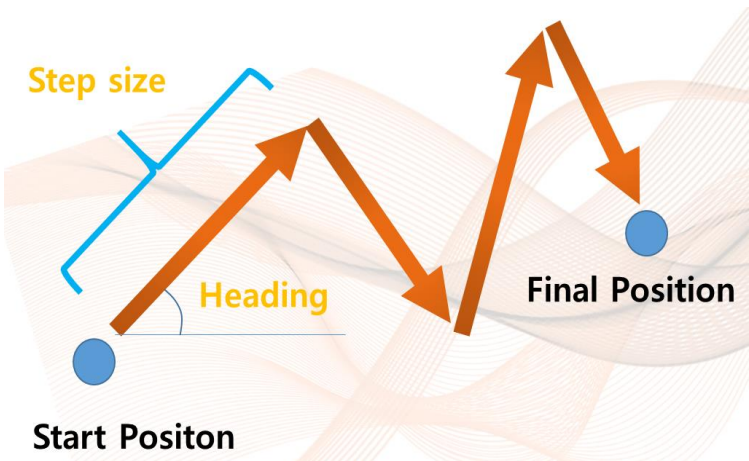


Figure 3-1. Pedestrian dead reckoning concept

The upper section shows the GPS / INS measurement properties. In this section, we will investigate the INS based navigation method, pedestrian dead reckoning. To perform the pedestrian dead reckoning, we need the information, heading, step size, and step length. About heading, the direction can be derived from the Euler yaw angle which can be achieved from the Kalman filtering result. Also, the user's step size and step length can be derived from the several algorithms. But the user can walk or not, the walking state needs to be determined. The step detection needs to conduct only in this context. In this section, the heading estimated quaternion extended Kalman filter will be constructed. Next, the walking detection method will be explained. Moreover, the step counting algorithm and stride length equation will be presented.

3.1.1 Attitude Determination Quaternion Kalman Filter

The attitude representation can have many types of form. The Euler angle is one of the attitude representation methods and intuitively understandable. But the Euler angle has a discontinuity problem and can make the gimbal lock effect. The smartphone is usually used diverse attitudes. For this reasons, the quaternion is chosen as the form which can represent the user motion. The quaternion equation is followed.

$$q = q_1 + q_2 i + q_3 j + q_4 k \qquad (3.1)$$

q_1 : scalar

$$q_2 i + q_3 j + q_4 k \ : \text{vector} \qquad (3.2)$$

The unit quaternion has unit norm size. By this unit quaternion's property, the next equation is given.

$$1 = \sqrt{q_1^2 + q_2^2 + q_3^2 + q_4^2} \qquad (3.3)$$

With this expression, the smartphone attitude extended Kalman filter equation can be derived.

Table 3-1. Sensor Model

Sensor Model	
Accel	$\overrightarrow{f_k} = {}^{a}S * [C_n^b(q_k)(\overrightarrow{q_k})(-\vec{g} + \overrightarrow{a_{bodyk}})] + {}^{a}\overrightarrow{b_{0k}} + {}^{a}\overrightarrow{b_{1k}} + {}^{a}\overrightarrow{v_k}$
Gyro	$\overrightarrow{w_k} = {}^{g}S * [\overrightarrow{w_{k0}})] + {}^{g}\overrightarrow{b_{0k}} + {}^{g}\overrightarrow{b_{1k}} + {}^{g}\overrightarrow{v_k}$
Mag	$\overrightarrow{m_k} = {}^{m}S * [C_n^b(q_k)(\overrightarrow{q_k})(\vec{h})] + {}^{m}\overrightarrow{b_{0k}} + {}^{m}\overrightarrow{b_{1k}} + {}^{m}\overrightarrow{v_k}$

S: scale factor , $C_n^b(q_k)$: dcm,
$\vec{g}$: gravity, $\overrightarrow{h}$: geomagnetic vector, $\overrightarrow{a_{bodyk}}$: body acceleration ,
$\overrightarrow{w_k}$: angular velocity, $\overrightarrow{b_{0k}}$: constant bias,
$\overrightarrow{b_{1k}}$: random bias , $\overrightarrow{v_k}$: measurement noise

Table 3-2. Bias Model

Bias Model	
Accel	$^a\overrightarrow{b_{1k}} = \left(1 - \frac{1}{\tau_a}\right)^a \overrightarrow{b_{1k-1}} + ^{ab}\overrightarrow{w_k}$
Gyro	$^g\overrightarrow{b_{1k}} = \left(1 - \frac{1}{\tau_a}\right)^g \overrightarrow{b_{1k-1}} + ^{gb}\overrightarrow{w_k}$
Mag	$^m\overrightarrow{b_{1k}} = \ ^m\overrightarrow{b_{1k-1}} + ^{mb}\overrightarrow{w_k}$

Table 3-3. Dynamic Model

Dynamic Model
$\overrightarrow{q_{k+1}} = \exp(\frac{1}{2}W_K)\overrightarrow{q_{k+1}} - \frac{T_S}{2}\xi_K(\ \ ^g\overrightarrow{b_{0k}} + ^g\overrightarrow{b_{1k}}) + ^q\overrightarrow{w_k}$
$^q\overrightarrow{w_k} = -\frac{T_S}{2}\xi_K + ^g\overrightarrow{v_k}$
$W_k = \begin{bmatrix} \overrightarrow{w_k}\times & \overrightarrow{w_k} \\ -\overrightarrow{w_k} & 0 \end{bmatrix}, \qquad \xi_k = \begin{bmatrix} [e\times] + q_4 I_3 \\ -e^T \end{bmatrix}$
$C_n^b = \begin{bmatrix} q_1^{\ 2} - q_2^{\ 2} - q_3^{\ 2} + q_4^{\ 2} & 2(q_1q_2 + q_3q_4) & 2(q_1q_3 - q_2q_4) \\ 2(q_1q_2 - q_3q_4) & -q_1^{\ 2} + q_2^{\ 2} - q_3^{\ 2} + q_4^{\ 2} & 2(q_2q_3 + q_1q_4) \\ 2(q_1q_3 + q_2q_4) & 2(q_2q_3 - q_1q_4) & -q_1^{\ 2} - q_2^{\ 2} + q_3^{\ 2} + q_4^{\ 2} \end{bmatrix}$

Table 3-4. Process Model

Process Model
$\overrightarrow{x_{k+1}} = F\overrightarrow{x_k} + G\overrightarrow{w_k}$
$\begin{bmatrix} \overrightarrow{q_{k+1}} \\ {}^{a}\overrightarrow{b_{1k+1}} \\ {}^{g}\overrightarrow{b_{1k+1}} \\ {}^{m}\overrightarrow{b_{1k+1}} \end{bmatrix} = \begin{bmatrix} \exp(\frac{1}{2}T_s) & 0 & -\frac{T_s}{2}\xi_k & 0 \\ 0 & \left(1-\frac{1}{\tau_a}\right) & 0 & 0 \\ 0 & 0 & \left(1-\frac{1}{\tau_b}\right) & 0 \\ & & & 1 \end{bmatrix} \begin{bmatrix} \overrightarrow{q_k} \\ {}^{a}\overrightarrow{b_{1k}} \\ {}^{g}\overrightarrow{b_{1k}} \\ {}^{m}\overrightarrow{b_{1k}} \end{bmatrix} + \begin{bmatrix} I & 0 & 0 & 0 \\ 0 & I & 0 & 0 \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & I \end{bmatrix} \begin{bmatrix} {}^{q}\overrightarrow{w_k} \\ {}^{a}\overrightarrow{w_k} \\ {}^{b}\overrightarrow{w_k} \\ {}^{m}\overrightarrow{w_k} \end{bmatrix}$

Table 3-5. Measurement Model

Measurement Model
$\overrightarrow{z_{k+1}} = h(\overrightarrow{x_k}) + \overrightarrow{v_k}$
$\begin{bmatrix} \overrightarrow{f_{k+1}} \\ \overrightarrow{m_{k+1}} \end{bmatrix} = \begin{bmatrix} H_1 & I & 0 & 0 \\ H_2 & 0 & 0 & \mathrm{I} \end{bmatrix} \begin{bmatrix} \delta\overrightarrow{q_k} \\ \delta^{a}\overrightarrow{b_{1k}} \\ \delta^{g}\overrightarrow{b_{1k}} \\ \delta^{m}\overrightarrow{b_{1k}} \end{bmatrix} + \begin{bmatrix} {}^{a}\overrightarrow{v_k} \\ {}^{m}\overrightarrow{v_k} \end{bmatrix}$
$H_1 = 2g* \begin{bmatrix} q_3 & -q_4 & q_1 & -q_2 \\ q_4 & q_3 & q_2 & q_1 \\ -q_1 & -q_2 & q_3 & q_4 \end{bmatrix}$ $H_2 = 2* \begin{bmatrix} Aq_1 + Bq_3 & -Aq_2 - Bq_4 & -Aq_3 + Bq_1 & Aq_4 - Bq_2 \\ Aq_2 + Bq_4 & Aq_1 + Bq_3 & -Aq_4 + Bq_2 & -Aq_3 + Bq_1 \\ Aq_3 - Bq_1 & Aq_4 - Bq_2 & Aq_1 + Bq_3 & Aq_2 + Bq_4 \end{bmatrix}$ $A = h\cos\varphi, B = h\sin\varphi$

With this model, the smartphone attitude is represented as the quaternion form. Because quaternion is not intuitively known. The extended Kalman filtering quaternion is transformed to Euler angle.

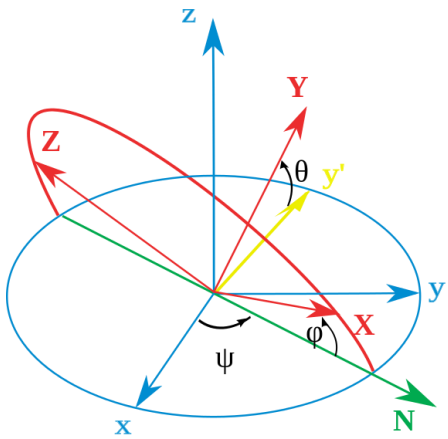


Figure 3-2. Euler Angle

(ϕ : roll, θ : pitch, ψ :yaw)

For dead reckoning, the heading information corresponds to euler yaw angle. But the exact walking heading could be different from the smartphone's attitude heading. There are many algorithms which can detect the main walking direction. For example, there are PCA and vertical component usage. But this research is conducted with the environment which the user walking direction is horizontal to smartphone's heading. So we assume the heading is equal to the user's main walking heading.

3.1.2 Attitude Determination Simulation , Experiment

Quaternion Extended Kalman Filter Simulation

The author validates the extended Kalman filter model's performance with simulation. Given the experimental noise and bias information, the sensor output is generated. Using this measurement, the Kalman filter shows the state estimation ability.

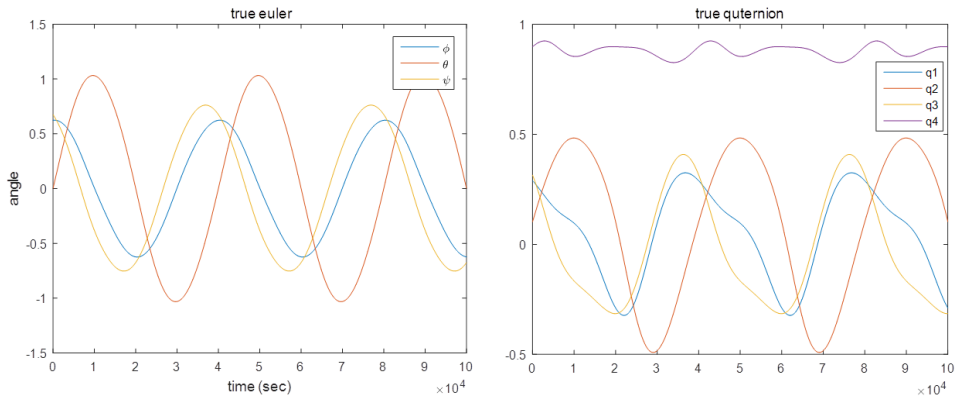


Figure 3-3. True Euler, quaternion

The true Euler is generated with the random signal (Sine waveform) and the quaternion is derived from the transform of Euler angle.

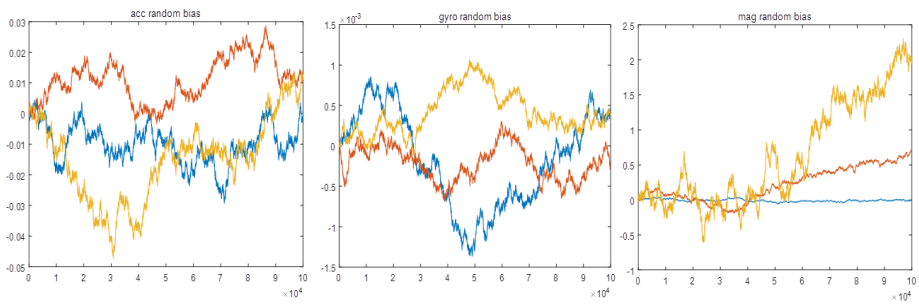


Figure 3-4. Random bias modeling
(Acc, Gyro, Mag)

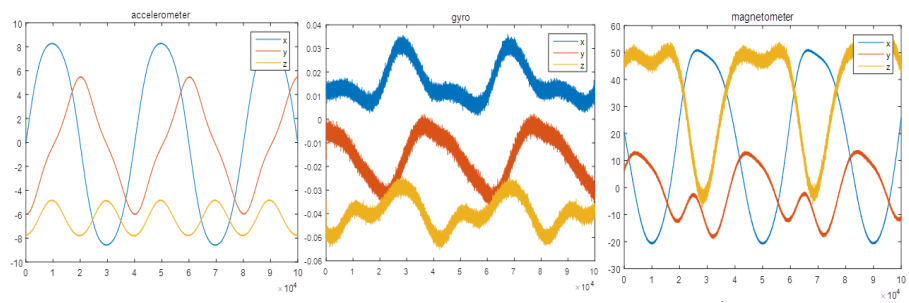


Figure 3-5. Final + constant bias + random bias + white noise
(Acc, Gyro, Mag)

The Sensor data is generated with the sensor model. With this simulated signals, the extended quaternion Kalman filter calculate the states(quaternion and bias). The result is given.

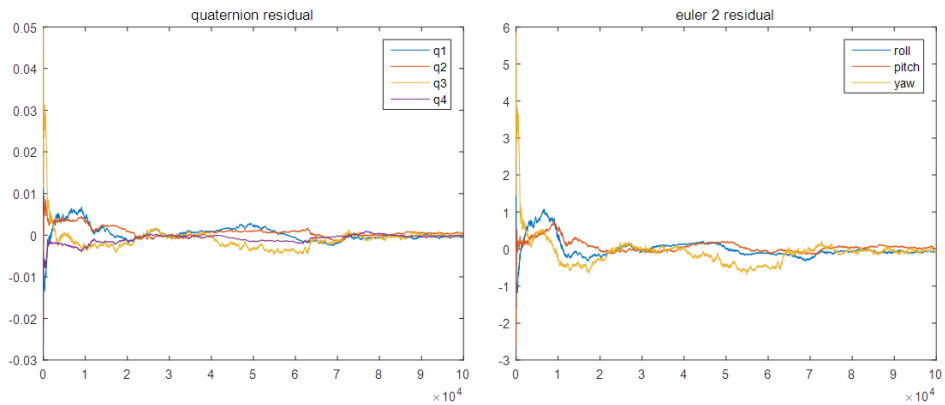


Figure 3-6. The error residual of the quaternion and Euler angle

Table 3-6. Converge error residual in simulation

Quaternion	q0	q1	q2	q3
RMS residual	0.0013	0.0010	0.0017	0.0009
Euler	Roll (deg)	Pitch (deg)	Yaw (deg)	
RMS residual	0.1669	0.1132	0.1804	

Quaternion Extended Kalman Filter Experiment

To tune the filter parameter for real data, the experiment is done. The true attitude is measured compared to the other INS sensor which gives the Euler angle information. The pixhawk 2.4.8 board inertial sensor out is used as reference attitude.

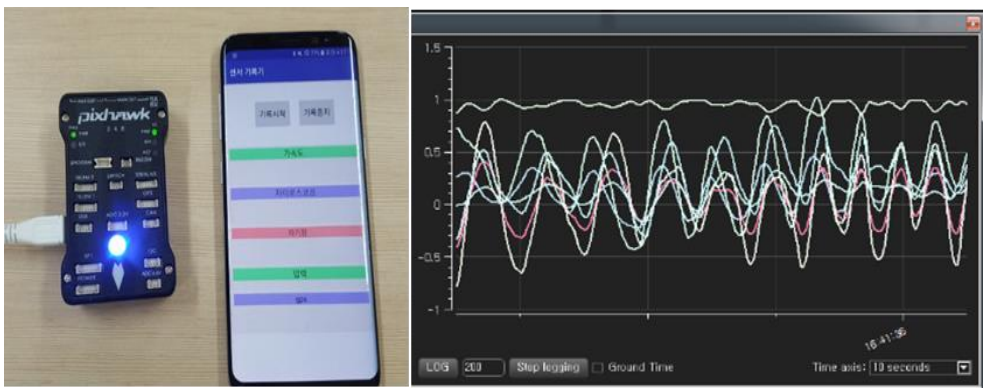
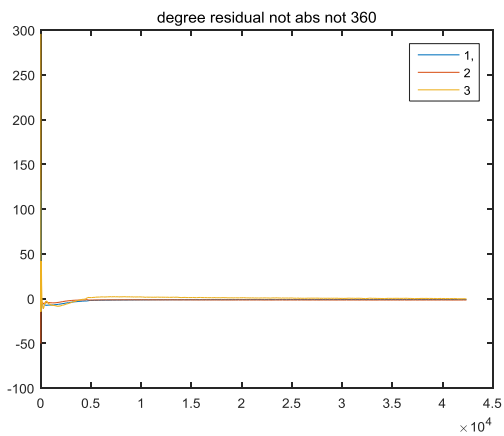


Figure 3-7. Pixhawk 2.4.8 board quaternion reference output



(x axis: samples(0.01s), y axis: deg)

Figure 3-8. Euler residual to reference sensor measurement

Table 3-7. Convergence error residual in experiment

Roll RMS residual	-1.7172 deg
Pitch RMS residual	-1.7529 deg
Yaw RMS residual	1.9553 deg

3.1.3 Walking Detection

For pedestrian dead reckoning, the remained work is finding the step information. The step needed to detect only on walking phase. To detect walking, the windowing standard deviation threshold method is used. the procedure of the algorithm is writing beneath.

Table 3-8. Walking detection

1. Calculate acceleration magnitude M = sqrt(ax^2+ay^2+az^2).
2. Moving average to M.
3. Low pass filter : cutoff frequency 3.5Hz.
4. With the not overlapped window, standard deviation is calculated in each windows.
5. Setting a threshold which can distinguish the walking phase.
6. Detect the time epoch which is over the threshold and has minimum duration.
   (the walking time is defined when std value is over the threshold(+) to below the threshold(-))

The window std walking detection is performed to the open data which is used in “Walk Detection and Step Counting on Unconstrained Smartphones - Agata Brajdic, Robert Harle”. After the validation to the open data set. the Galaxy S8 phone experiment is performed (Sec 3.1.5). The open dataset analysis result is given.

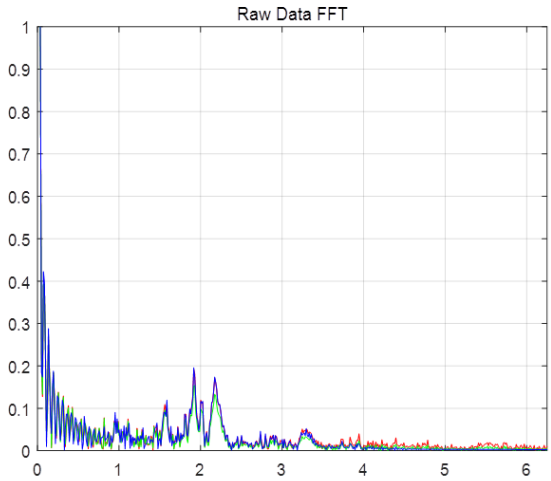
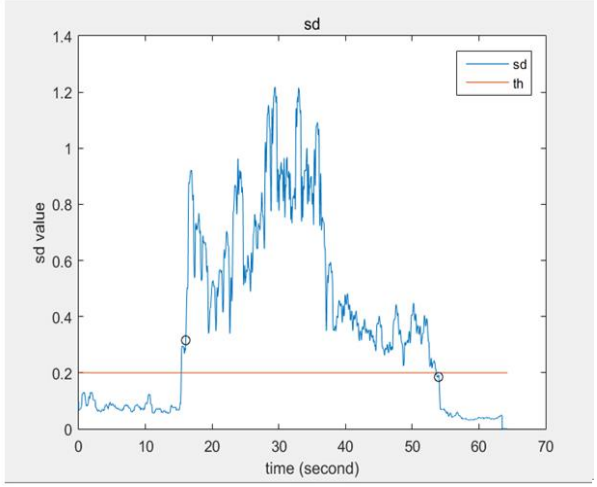
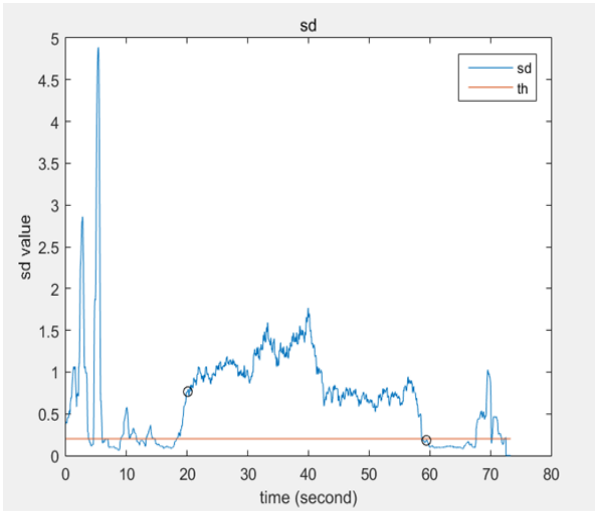


Figure 3-9. Low pass filtering (x : frequency, y:FFT accM)

(black circle : true walking start and end time)





(Minimum duration rule help to ignore the not walking signal)

Figure 3-10. Walking detection with windowed std threshold method

Performance

There are 5 users and 5 datasets are used in each test set.

Table 3-9. Walking detection performance for the open dataset

	Testset 1	Testset 2	Testset 3	Overall Mean
mean wd error	0.75 s	0.77 s	1.61 s	0.78 s

(Used std threshold: 0.2)

3.1.4 Step Counting, Stride Length

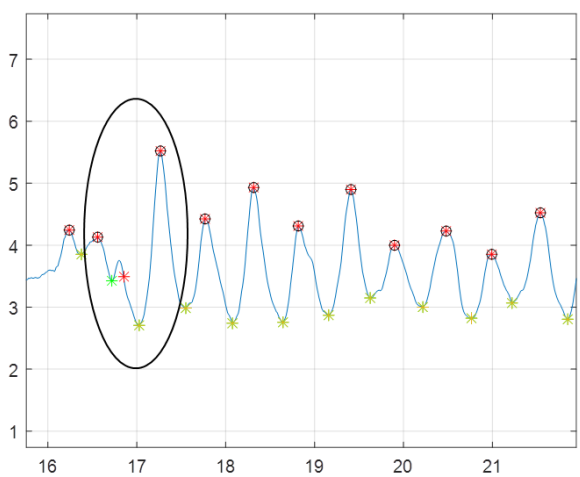
Step counting and stride length algorithm based on the peak, valley detection. With the peak and valley magnitudes, the step information can be inferred. The algorithm is given in following table

Table 3-10. Peak /Valley Detection Algorithm

1. Calculate acceleration magnitude M = sqrt(ax^2+ay^2+az^2)
2. Moving Average to M
3. The step counting algorithm is applied in only walking detected phase.
4. With comparison between previous and current M value, save the maximum and minimum value continuously.
   The maximum/minimum mode is alternatively changed because the step signal has peak and valley pair. the last max/ min value is saved at each mode.

   maximum mode : if max<x1 , save x1, continue
   if x1<max, not save, mode change as min
   minimum mode: if min> x2, save x2, continue
   if x2<min, not save, mode change as max

5. condition1: The peak must be higher than mean M value
6. condition2: With walking frequency considered, the peak and valley time interval is longer than the minimum time threshold
   peak time: t1 , valley time : t2 , (t1-t2) > threshold



7. condition3: 1 valley must appear between peaks at least. If there are many valleys between peaks, the lower is chosen. Also, if there is two peak which has not one valley between them, the higher is chosen.

With this algorithm, the performance is analyzed with the same dataset.

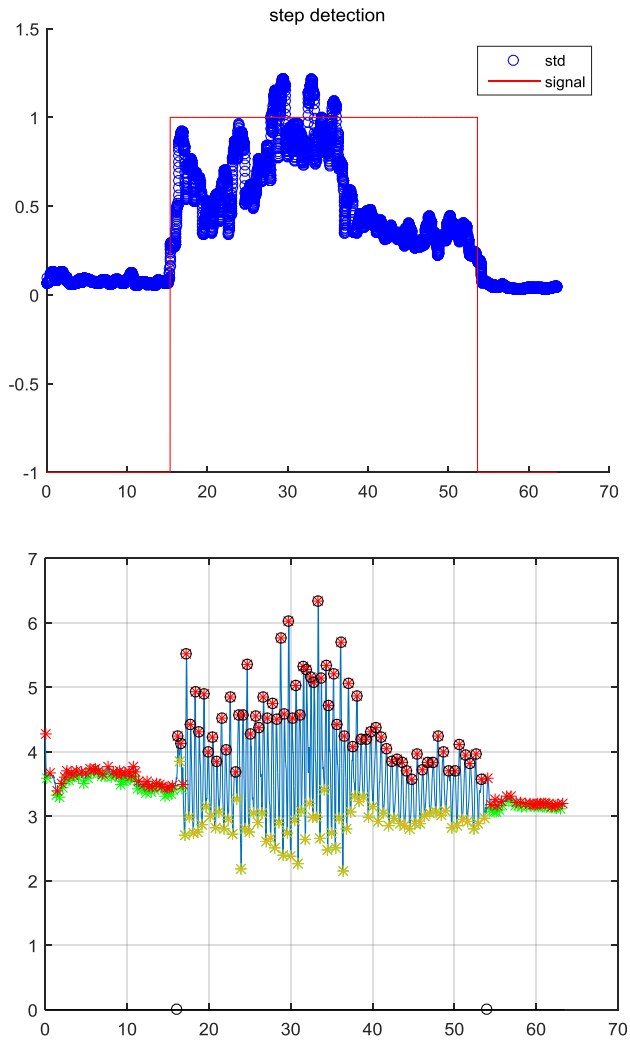


Figure 3-11. Peak/ Valley detection method

Table 3-11. Peak / Valley detection performance for the open dataset

	With 21 walking dataset (21 users)
Step error mean	5.1 step

Stride Length is related with the peak and valley acceleration magnitude. In the "Smartphone-based Pedestrian Dead Reckoning as an Indoor Positioning System - Azkario Rizky Pratama" research [8], the author said the stride can be derived from the following equation.

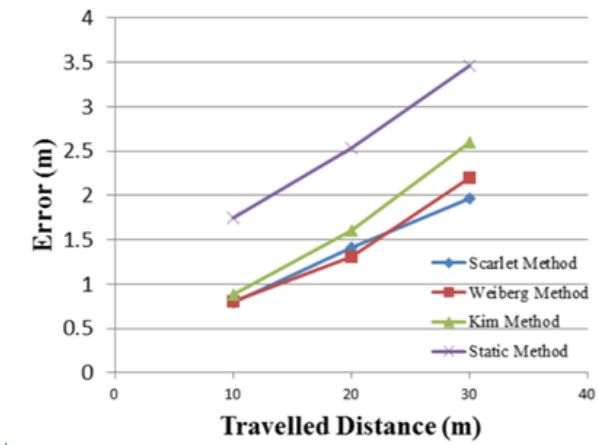


Figure 3-12. Step length method comparison

Table 3-12. Step length algorithms.

1. Static	$step\_size = height \cdot k$
2. Weinberg	$step\_size = k \cdot \sqrt[4]{a_{\max} - a_{\min}}$
3. Scarlet	$step\_size = k \cdot \frac{\frac{\sum_{k=1}^{N} \lvert a_k \rvert}{N} - a_{\min}}{a_{\max} - a_{\min}}$

4. Kims	$step\_size = k \cdot \sqrt[3]{\frac{\sum_{k=1}^{N} \lvert a_k \rvert}{N}}$
5. Linear	$L_S = \alpha_{S_i,C_j} \cdot fre + \beta_{S_i,C_j} \cdot var + \gamma_{S_i,C_j}$,

The Weiberg method and Scarlet method shows a good result in this paper. With this background, our research used Weiberg method to determine the stride length.

$$Stepsize = K * \sqrt{peak - valley} \qquad (3.4)$$

(The constant K is the tuned value and this is experimentally chosen.)

3.1.5 Pedestrian Dead Reckoning Experiment

To choose the fine threshold, the pedestrian dead reckoning experiment is conducted and the following figure gives the experiment environment.

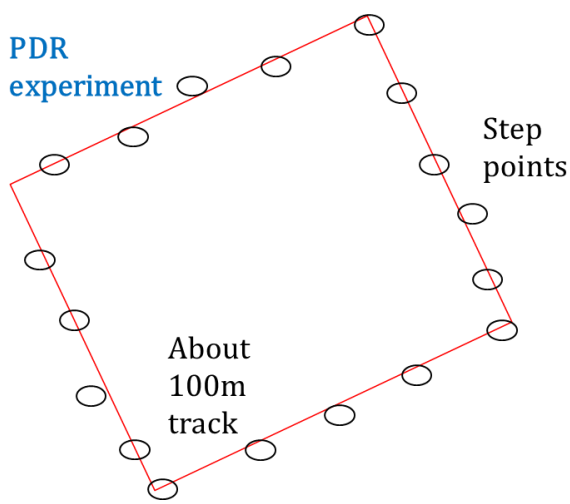
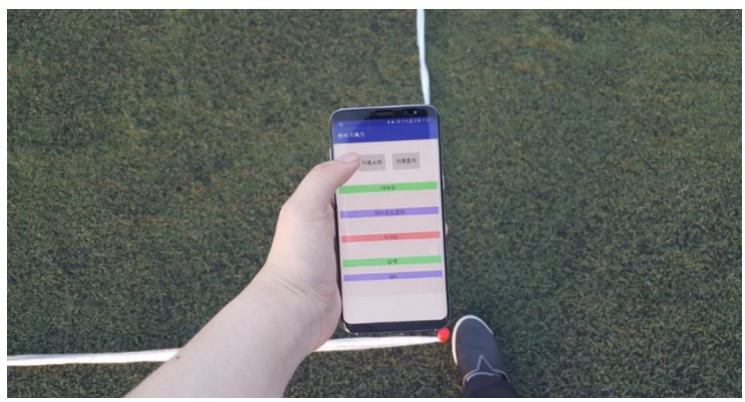


Figure 3-13. Pedestrian dead reckoning fitting experiment

Table 3-13. PDR experiment settings.

	Length (cm)	Heading (deg)	Step number
Track line 1	951	325	17
Track line 2	956	415	18
Track line 3	943	145	17
Track line 4	943	235	17
Times	Each 3 turns, 5 times experiment is done		
Place	Seoul National University President Grace		



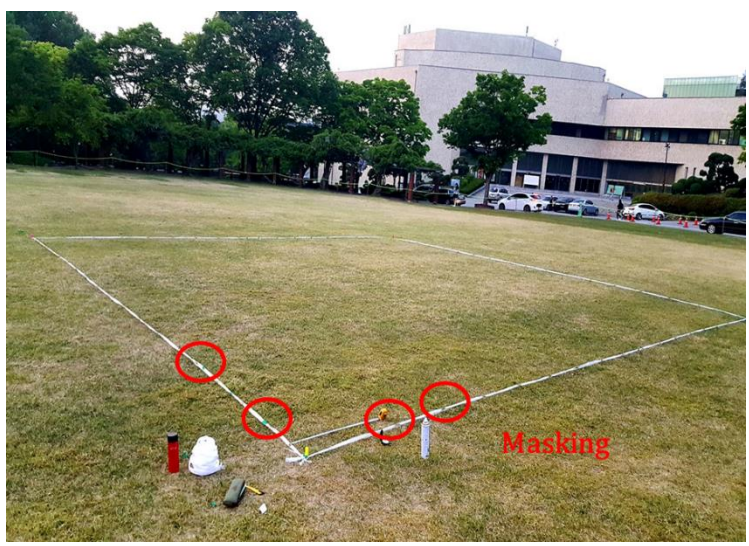


Figure 3-14. PDR fitting experiment

Table 3-14. The error of step counting.

WD threshold	0.2		
Error type	inbound	outbound	rms
Test1	-6.1908	6.3454	8.865106
Test2	-0.6657	8.7298	8.755145
Test3	-0.4706	9.8074	9.818684
Test4	-0.4325	10.3372	10.34624
Test5	-0.4112	10.7994	10.80723
Mean	-1.63416	9.20384	9.718481
WD threshold	0.3		
Error type	inbound	outbound	rms
Test1	-13.6318	3.9617	14.19581
Test2	-1.7537	5.6785	5.943133
Test3	-1.0302	6.5199	6.600788
Test4	-0.9459	6.8652	6.930058
Test5	-0.8351	7.208	7.256215
Mean	-3.63934	6.04666	8.185201
WD threshold	0.5		
Error type	inbound	outbound	rms
Test1	-26.3452	1.9159	26.41477
Test2	-8.5462	3.1794	9.118449
Test3	-5.4796	3.8276	6.684051
Test4	-4.8706	4.1268	6.383825
Test5	-4.3186	4.4669	6.213172
Mean	-9.91204	3.50332	10.96285

(inbound : step error within wd phase, outbound : step error outside wd phase)

Upper table shows that the std threshold 0.3 is the best result. (8step/207step = 3% error)

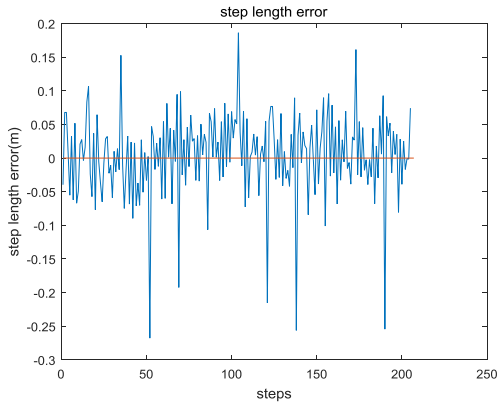


Figure 3-15. Step length error

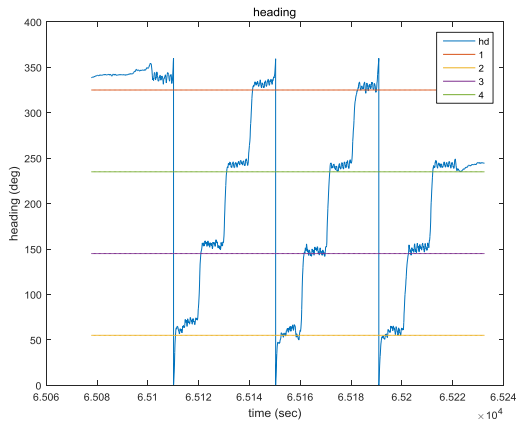


Figure 3-16. heading

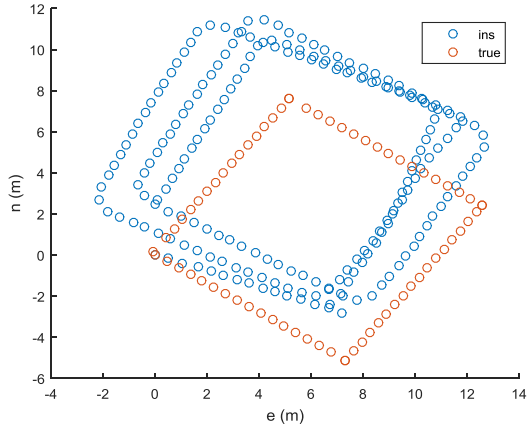


Figure 3-17. pedestrian dead reckoning trajectory. (not use the saved gain)

To tune the suitable stride length parameter, the comparison between real step size and detected step size is conducted. The result was about 0.05 mean error of stride length has the K : 0.6417 value. Also, the heading shows a bias about 10 deg at start time and the bias becomes smaller as the kalman filter converge. Considering this effect, the converging kalman gain is saved and used as initial state. This helped the heading bias converge fast.

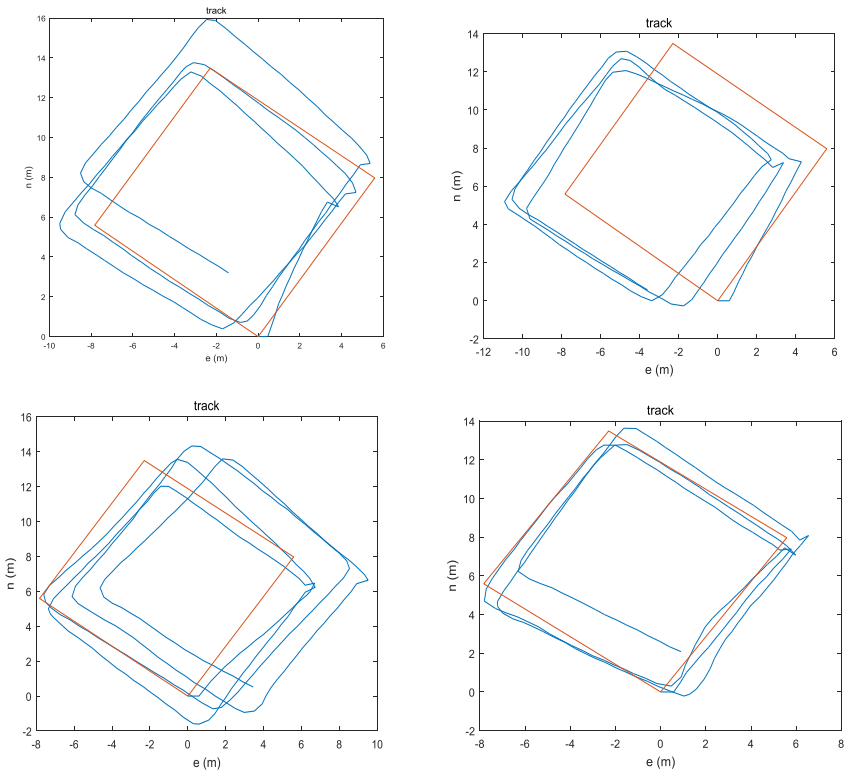


Figure 3-18. pedestrian dead reckoning trajectories. (use the saved gain)

Chapter 4. Carrier phase / INS integrated Pedestrian Dead Reckoning

In this chapter, the carrier phase measurement is investigated and the android enviroment cycle slip compensation will be suggested. To choose an appropriate threshold in the slip detection, the simulation is performed with considering miss detection and false alarm scheme. After cycle slip elimination, the carrier phase based velocity determination method will be introduced. Finally, the gps/ ins integrated pedestrian dead reckoning is conducted and the performance will be analized.

4.1 Carrier phase Cycleslip Compensation & Velocity Determination

4.1.1 Carrier phase Cycleslip Compensation

When the GPS receiver lock losses some cycles of the carrier and relocked, the Integer Ambiguity component $N_u^i\lambda$ of carrier phase measurement is jumped. This is called as 'Cycle slip' effect. This discontinuity becomes the main error in the carrier-based navigation. The cycle slip comes from 3 causes. when the sight between user and satellites is blocked some object like a tree, tunnel, and loof etc, the cycle slip error could appear. Also, low SNR can loose the cycle. The receiver's intenal problem also contributed to the cycle slip effect.

To eliminate this error, the relation between Doppler and Carrier phase needs to be explained.

$$\phi_u^i = d_u^i + B_u - b^i + T_u^i - I_u^i + \partial R_u^i + M_u^i + N_u^i\lambda + \varepsilon_{u,\phi}^i \text{ (m)} \quad (4.1)$$

$$f_u^i = \dot{d}_u^i + \dot{B}_u - \dot{b}^i + \dot{T}_u^i - \dot{I}_u^i + \partial\dot{R}_u^i + \dot{M}_u^i + \varepsilon_{u,f}^i \text{ (m/s)} \quad (4.2)$$

Except for the Ambiguity term, the other components are almost same. But the dimension (meter, meter/sec) and the noise levels are different. So that the time differenced carrier phase has a similar value with Doppler when the cycle slip does not occur.

$$f_u^i \approx \delta\phi_u^i \quad (\text{if } N_u^i\lambda = 0) \tag{4.3}$$

For this reason, the cycle slip can be detected with below equation.

$$\begin{aligned} \delta\phi_u^i &= (\phi_u^i(t+1) - \phi_u^i(t)) / \Delta t \approx f_u^i + N_u^i\lambda \\ \delta\phi_u^i - f_u^i &\approx N_u^i\lambda \end{aligned} \tag{4.4}$$

With the TDCP(Time Differenced Carrier Phase) minus Doppler parameter($\delta\phi_u^i - f_u^i$), the cycle slip can be detected with the setting threshold.

4.1.2 Android Environment Cycle slip Detection

The GNSS measurement API supports some constant which can give the information of the receiver like cycle slip, multipath and lock/sync state. The following table shows the carrier phase cycle slip related constant.

Table 4-1. Accumulated Delta Range(carrier) related constant in API.

Constants	
int	ADR_STATE_CYCLE_SLIP
	The state of the getAccumulatedDeltaRangeMeters() has a cycle slip detected.
int	ADR_STATE_HALF_CYCLE_REPORTED

	Reports whether the flag ADR_STATE_HALF_CYCLE_RESOLVED has been reported by the GNSS hardware.
int	ADR_STATE_HALF_CYCLE_RESOLVED
	Reports whether the value getAccumulatedDeltaRangeMeters() has resolved the half cycle ambiguity.
int	ADR_STATE_RESET
	The state of the getAccumulatedDeltaRangeMeters() has detected a reset.
int	ADR_STATE_UNKNOWN
	The state of the value getAccumulatedDeltaRangeMeters() is invalid or unknown.
int	ADR_STATE_VALID
	The state of the getAccumulatedDeltaRangeMeters() is valid.

The ADR_STATE_CYCLE_SLIP parameter is investigated.

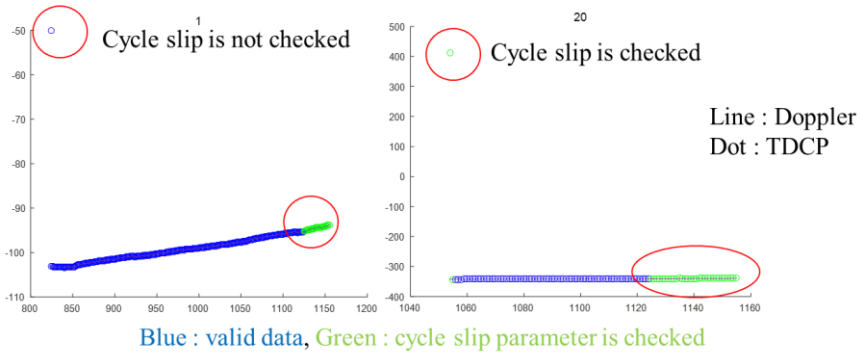


Figure 4-1. ADR_STATE_CYCLE_SLIP parameter in GNSS API

The result shows the cycle slip parameter is turned on either cycle slip or not. For this reason, the ADR_STATE_CYCLE_SLIP parameter is not sufficient for detecting cycle slip. The author found this parameter is related with the accumulated deltarange uncertainty meters. When the signal quality(SNR) becomes lower then the uncertainty sigma value goes up.

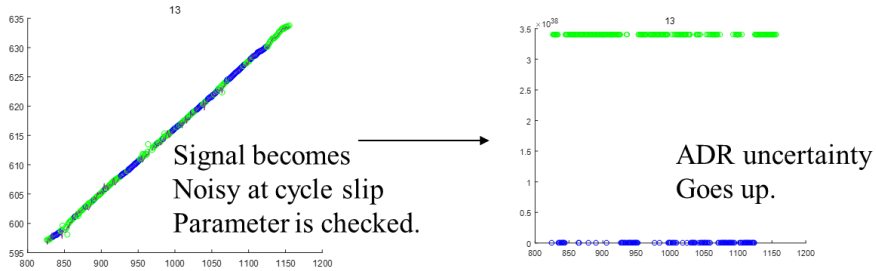


Figure 4-2. ADR uncertainty meters relation with cycle slip parameter.

From this analysis, we can understand the cycle slip parameter represents the low SNR situation. the high carrier uncertainty region has high probability of the cycle slip appearance but the cycle slip could be occurs or not.

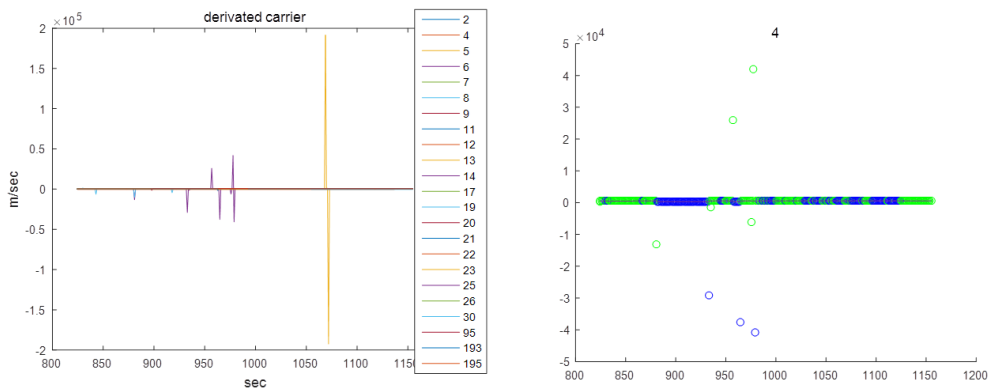


Figure 4-3. (TDCP – doppler) vs cycle slip parameter.

By this reason the TDCP – doppler is used as the parameter to catch the cycle slip. The time derivated carrier phase has the following relation.

Table 4-2. TDCP – doppler parameter equation.

$$\phi_u^i = d_u^i + B_u - b^i + T_u^i - I_u^i + \partial R_u^i + M_u^i + N_u^i \lambda + \varepsilon_{u,\phi}^i \text{, (m)}$$
$$\varepsilon_{u,\phi}^i \sim N(0, \sigma_\phi^2)$$
$$f_u^i = \dot{d}_u^i + \dot{B}_u - \dot{b}^i + \dot{T}_u^i - \dot{I}_u^i + \partial \dot{R}_u^i + \dot{M}_u^i + \varepsilon_{u,f}^i \text{ , (m/sec)}$$
$$\varepsilon_{u,f}^i \sim N(0, \sigma_f^2)$$

$$\delta\phi_u^i = (\phi_u^i(t+1) - \phi_u^i(t)) / \Delta t$$
$$\delta\phi_u^i = \dot{d}_u^i + \dot{B}_u - \dot{b}^i + \dot{T}_u^i - \dot{I}_u^i + \partial \dot{R}_u^i + \dot{M}_u^i + \delta N_u^i \lambda + \delta\varepsilon_{u,\phi}^i \text{ (m/sec)}$$
$$\delta\varepsilon_{u,\phi}^i \sim N(0, \sigma_{\delta\phi}^2)$$

$$\delta\phi_u^i - f_u^i \approx \delta N_u^i \lambda + \delta\varepsilon_{u,\phi}^i - \varepsilon_{u,f}^i$$
$$\varepsilon_{\delta\phi f}^i = \delta\varepsilon_{u,\phi}^i - \varepsilon_{u,f}^i \sim \mathrm{N}(0, {\sigma_{\delta\phi f}}^2) \quad \delta \text{ : Time difference operator in 1 epoch}$$

With this background, the time differenced carrier minus doppler parameter can detected the cycle slips. The parameter's sigma could be estimated with the TDCP and doppler sigma value.

$$\text{real : } \sigma_{\delta\Phi} = 0.1694 \left(\frac{m}{sec}\right) \qquad (4.5)$$

$$\text{real : } \sigma_f = 0.2047 \left(\frac{m}{sec}\right) \qquad (4.6)$$

$$\sigma_{\delta\phi f} = \sqrt{{\sigma_{\delta\phi}}^2 + {\sigma_f}^2} \qquad (4.7)$$

$$\text{real : } \sigma_{\delta\Phi f} = 0.2658 \left(\frac{m}{sec}\right) \qquad (4.8)$$

With this result, the author used the TDCP-doppler parameter sigma value as 0.2658.

4.1.3 False Alarm & Miss Detection Analysis

To set the appropriate cycle slip threshold the simulation is performed. The false alarm and miss detection scheme is applied.

The false alarm means the cycle slip is detected in the wrong epoch. On the other hand, miss detection is the threshold don't catch the cycle slip. The FA(False alarm) and MD(Miss detection) have contradicting property. So that the user need to design the probability which is considered the priority between them.

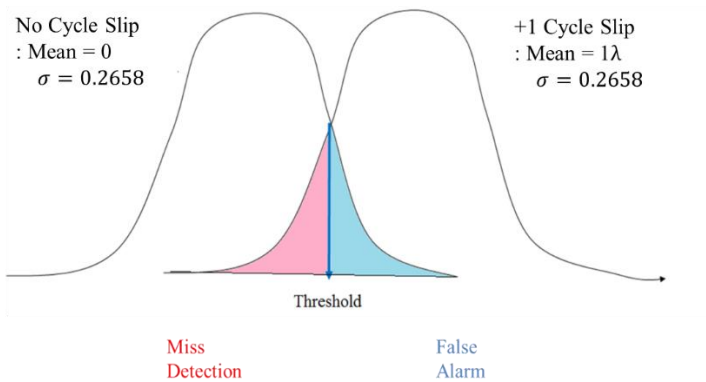


Figure 4-4. False alarm, Miss detection

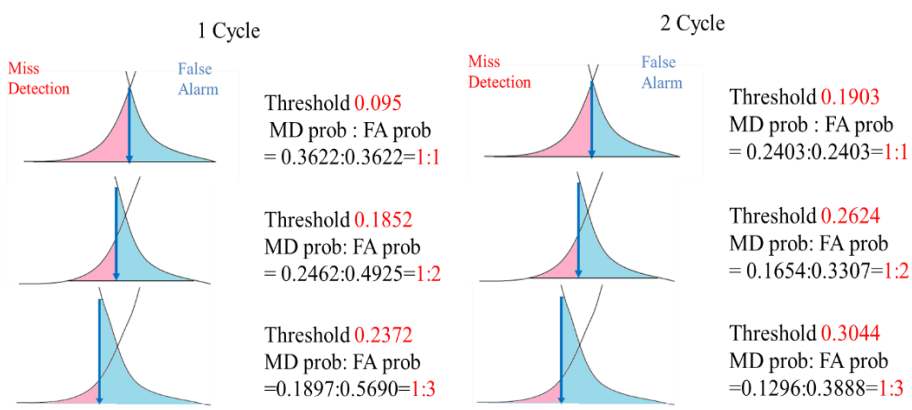


Figure 4-5. The probability for each thresholds

In this research, the each FA / MD is samely important. Also, the upper 1 cycle detection threshold 0.095 (0.3622 prob MD/FA) is used.

4.1.4 Doppler, Carrier Based Velocity Estimation

With the raw measurement Doppler and Carrier, the more precise velocity can be estimated than pseudo-range based [30].

The Doppler shift effect is occurred from the relative motion between user and satellite. This effect can be described as following equation.

$$D_i = f_{Ri} - f_{Ti} = -\frac{f_{Ti}}{c}\left[(v_i - v_u) * \frac{r_i - r_u}{\|r_i - r_u\|} \right] \quad (4.9)$$

(*(dot product), f_{Ti} , f_{Ri} are respectively frequency of the transmitted and received signal.)

$$a_i = (a_x, a_y, a_z) = \frac{r_i - r_u}{\|r_i - r_u\|} \quad (4.10)$$

: user-satellite line of sight vector

If satellite clock error is eliminated with the ephemeris information, the remained main error is receiver clock error. If the receiver clock drift rate relative to satellite clock t_u considered, the drift rate in receiver frequency can be expressed.

$$f_{Ri} = f_i(1 + \dot{t}_u) \quad (4.11)$$

Scaling the Doppler shift with the carrier wavelength, the Doppler frequency(Pseudorange rate) equation is given.

$$f_u^i = (v_i - v_u) * a_i + c\dot{\delta}t_u + \varepsilon_{u,f}^i \quad (4.12)$$

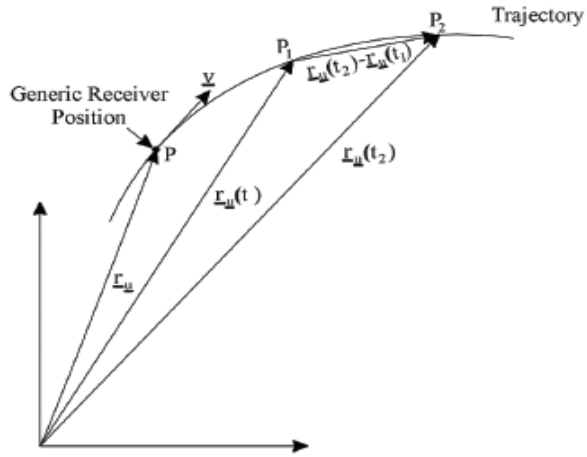


Figure 4-6. Receiver Motion

The satellites velocities can be calculated from the ephemerides. After the removement, the under equation can be derived.

$$f_u^i - v_i * a_i = -v_u * a_i + c\dot{\delta}t_u + \varepsilon_{u,f}^i = -v_u * a_i + B_u + \varepsilon_{u,f}^i \tag{4.13}$$

The right hand v_u , B_u terms can be solved with the least square method.

$$\begin{bmatrix} f_u^i - v_i * a_i \\ f_u^j - v_j * a_j \\ \vdots \\ f_u^k - v_k * a_l \end{bmatrix} = \begin{bmatrix} a_{ix} & a_{iy} & a_{iz} & 1 \\ a_{jx} & a_{jy} & a_{jz} & 1 \\ \vdots & \vdots & \vdots & \vdots \\ a_{zx} & a_{zy} & a_{zz} & 1 \end{bmatrix} \begin{bmatrix} v_{ux} \\ v_{uy} \\ v_{uz} \\ B_u \end{bmatrix} \tag{4.14}$$

$$\overrightarrow{Hx} = \overrightarrow{z}$$
$$\overrightarrow{x} = (H^T H)^{-1} H^T \overrightarrow{z} \tag{4.15}$$

With the upper section condition, $f_u^i \simeq \delta\phi_u^i$ (cycle slip is eliminated), the same method can be applied to the time differenced carrier phase. The carrier phase has the small noise property compare to Doppler. So that the result velocity is expected to have more accurate result.

4.1.5 Cycle slip Compensation & Velocity Determination Experiment

To determine the velocity, the Doppler and carrier phase measurement can be used. But the carrier phase is needed to eliminate the cycle slip. With the choosing threshold in upper section, the cycle slip compensation is performed.

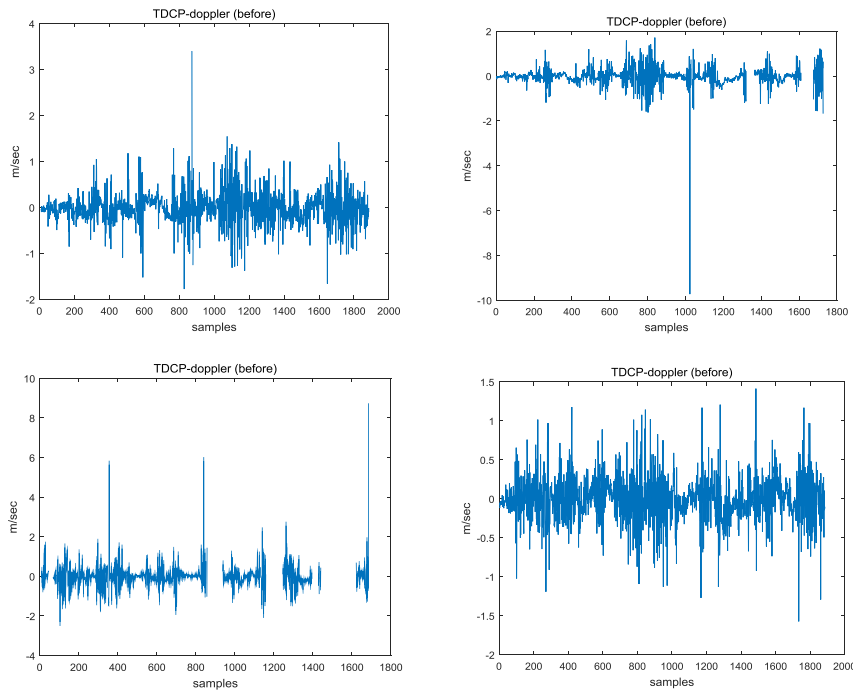


Figure 4-7. Cycle slip effect in carrier phase

The upper threshold parameter(TDCP-doppler) is removed the integer multiple of the carrier phase wavelength ($N_u^i \lambda$).

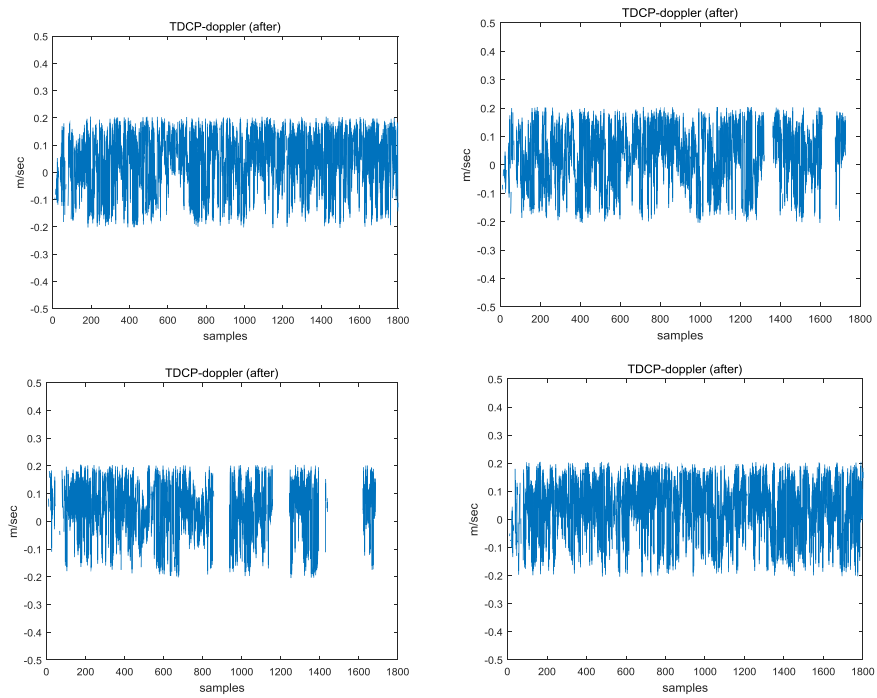


Figure 4-8. Cycle slip compensation

Overall parameter becomes with in 1 lamda.This effect can be checked in the velocity determination result also.

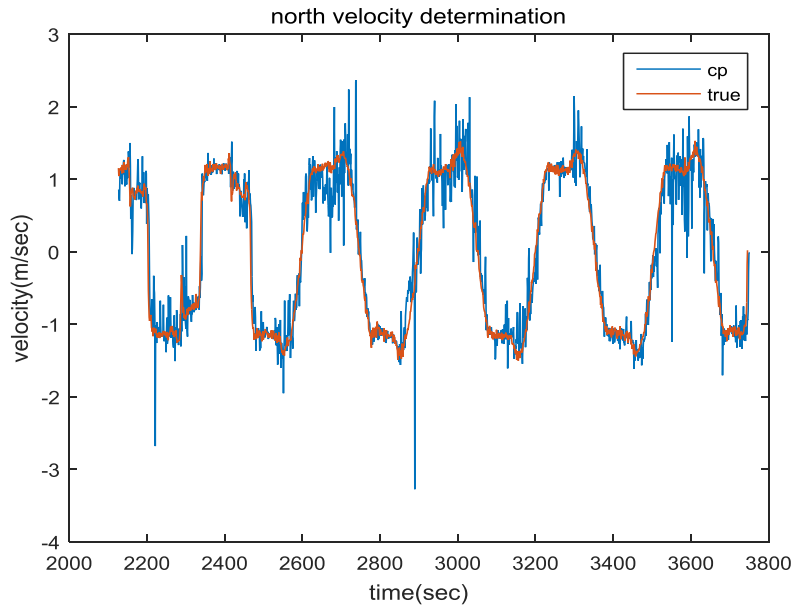


Figure 4-9. Velocity before the cycle slip compensation (North)

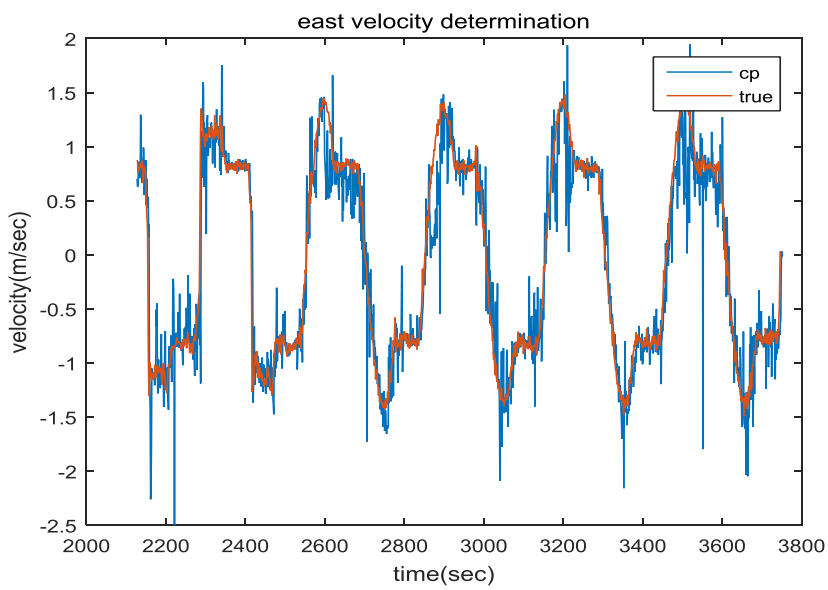


Figure 4-10. Velocity before the cycle slip compensation (East)

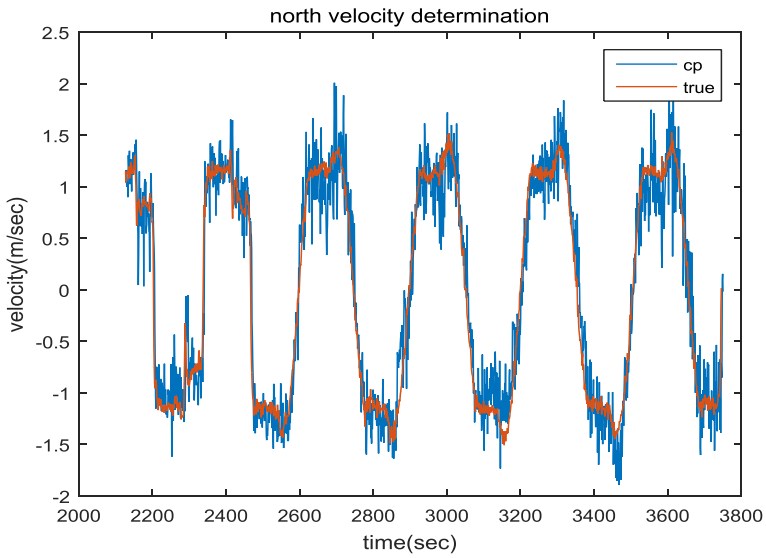


Figure 4-11. Velocity after the cycle slip compensation (North)

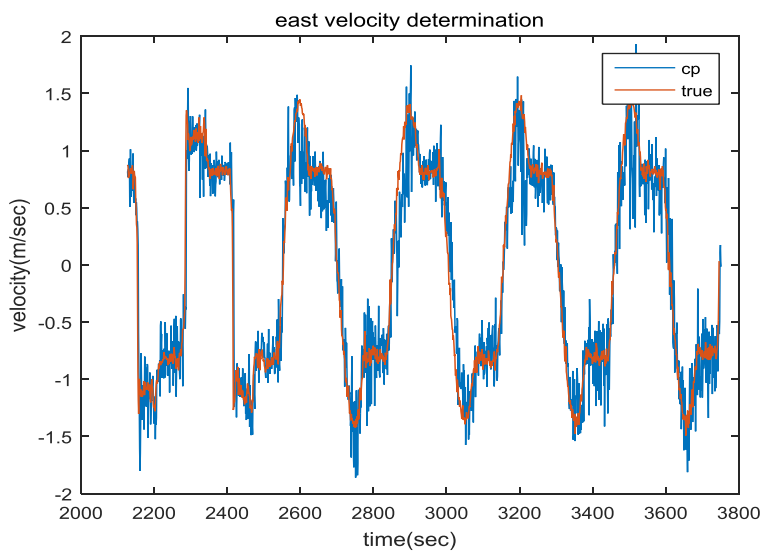


Figure 4-12. Velocity after the cycle slip compensation (East)

The result shows the cycle slip compensation contributed to the velocity error. Finally, the velocity determination accuracy is analyzed. The doppler based velocity also calcurated for the comparison.

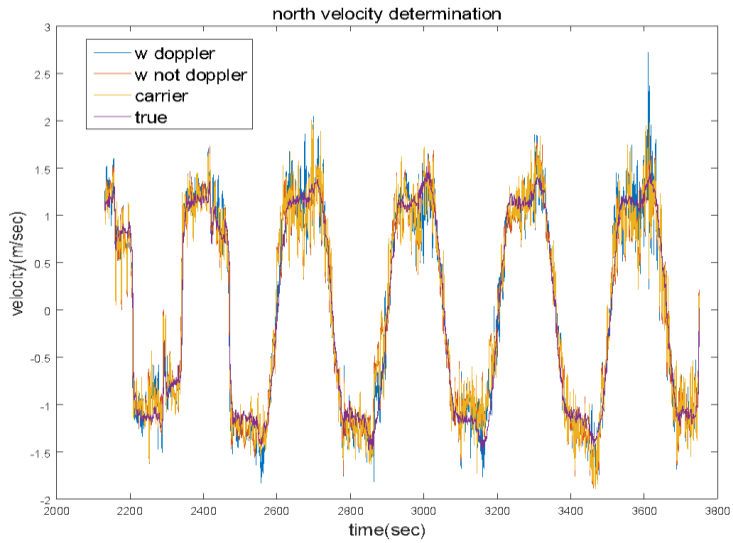


Figure 4-13. Velocity Determination Experiment (North)

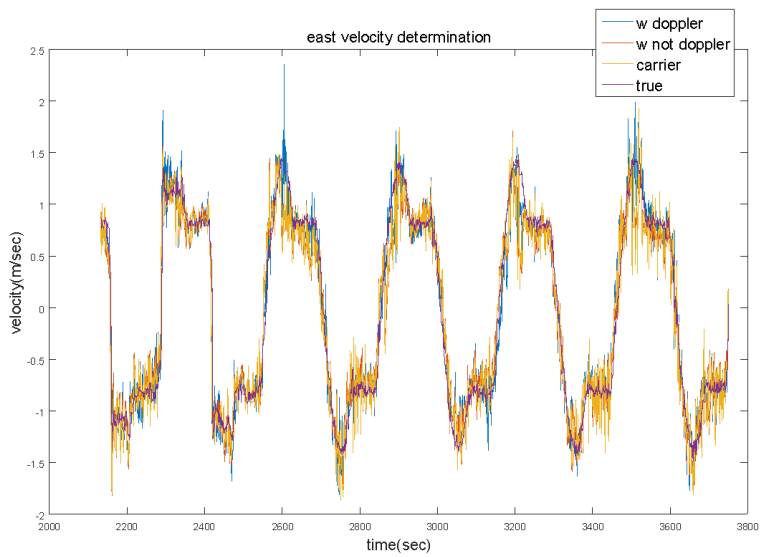


Figure 4-14. Velocity Determination Experiment (East).

Table 4-3. The mean velocity errors of the velocity determination

Name	North Vel. Mean Error (m/sec)	East Vel. Mean Error (m/sec)	Total Mean Error (m/sec)
WLS doppler	0.1950	0.1692	0.1821
Not WLS doppler	0.2048	0.1906	0.1977
carrier	0.1878	0.1674	0.17766
NMEA	0.1816	0.2427	0.21215

The raw measurement supports the doppler and carrier signal uncertainty. But when we use TDCP, the sigma value is not known (it could be approximated but the result was not good). For this reason, the weighted least square could apply to doppler only. The result shows the carrier phase velocity navigation is most accurate

4.2 Raw GPS / INS Integrated Pedestrian Dead Reckoning

4.2.1 GPS / INS Integration

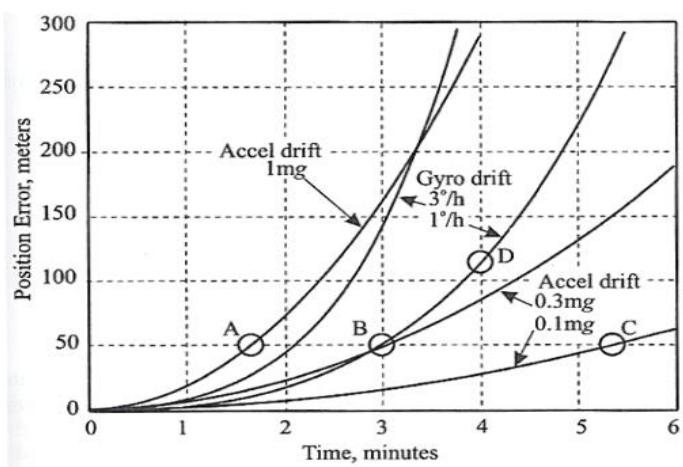


Figure 4-15. INS error property

The INS error is not bounded because the error is accumulated. This bias error is modeled in the extended Kalman filter but it is hard to make the exact model. About the stride detection case, the acceleration magnitude M is not kalman filtered. So that the accumulated drift error can deteriorate the step spot detection. To avoid this INS divergence the GPS/INS integration is used. The GPS gives the absolute position compared to the relative position from the INS. The integration can fix the error.

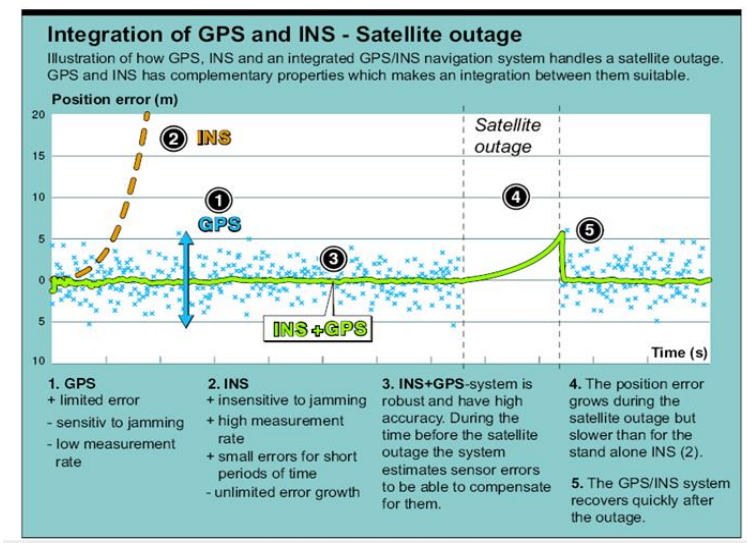


Figure 4-16. GPS / INS Integration

But when GPS goes out, the INS error starts to diverge again. The GPS / INS integration is a well-known method in the navigation field. In this research, the carrier phase based velocity and gps position are used as a measurement to improve the performance.

4.2.2 Position Determination Extended Kalman Filter

To integrate the INS based pedestrian dead reckoning position and the GPS navigated position, the extended Kalman filter is used again.
The filter equation is like below.

Table 4-4. Position extended kalman filter equation

Process Model $\phi_k = \phi_{k-1} + \frac{V_n}{R}\Delta t$ $\lambda_k = \lambda_{k-1} + \frac{V_n}{R\cos\phi}\Delta t$ $V_{Nk} = \frac{(S_{k-1} + \Delta S_{k-1})}{\Delta t}\cos(\psi_{k-1} + \Delta\psi_{k-1}) + w_{k-1}^{V_N}$ $V_{Ek} = \frac{(S_{k-1} + \Delta S_{k-1})}{\Delta t}\sin(\psi_{k-1} + \Delta\psi_{k-1}) + w_{k-1}^{V_E}$ $\Delta\psi_k = \Delta\psi_{k-1} + b_\psi \Delta t + w_{k-1}^{\Delta\psi}$ $\Delta S_k = \Delta S_{k-1} + b_s \Delta t + w_{k-1}^{\Delta S}$
Measurement Model $\phi_{mk} = \phi_k + \nu_{\phi k}$ $\lambda_{mk} = \lambda_k + \nu_{\lambda k}$ $v_{mNk} = v_{Nk} + \nu_{v_N k}$ $v_{mEk} = v_{Ek} + \nu_{v_E k}$ $\psi_{mk} = \psi_k + \Delta\psi_k + \nu_{\psi k}$

The GPS and INS sampling rate is different so that the interpolation is conducted to synchronized the measurement update. The figure shows the interpolation, time update, and measurement update.

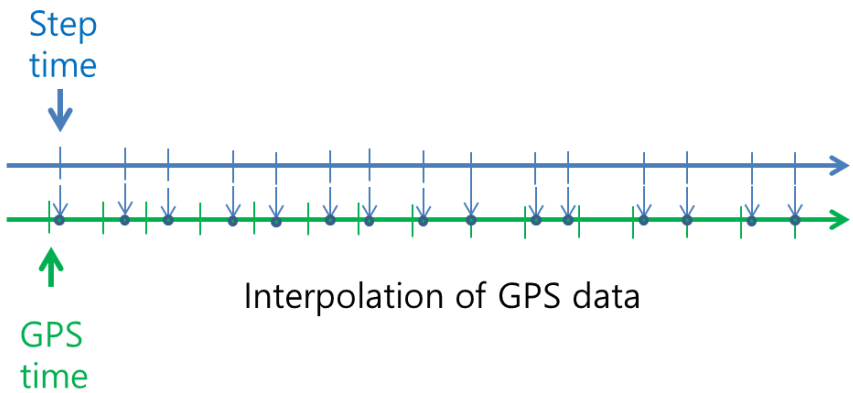


Figure 4-17. GPS data interpolation to step time

4.2.3 Raw GPS / INS Integrated Pedestrian Dead Reckoning Experiment

With the designed position filter, the author calculates the position, velocity, heading error and stride error as states. To judge the performance, the accurate Trimble GEO-XR data is used as a true trajectory. (The experimental data is same with 4.1.5 but the INS data is integrated.

Figure 4-18. The experiment in Seoul National University Open sky Playground

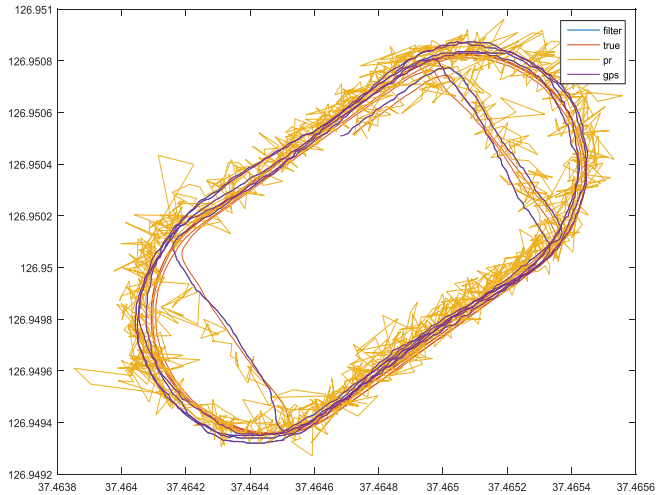


Figure 4-19. GPS / INS integrated pedestrian dead reckoning experiment

Performance

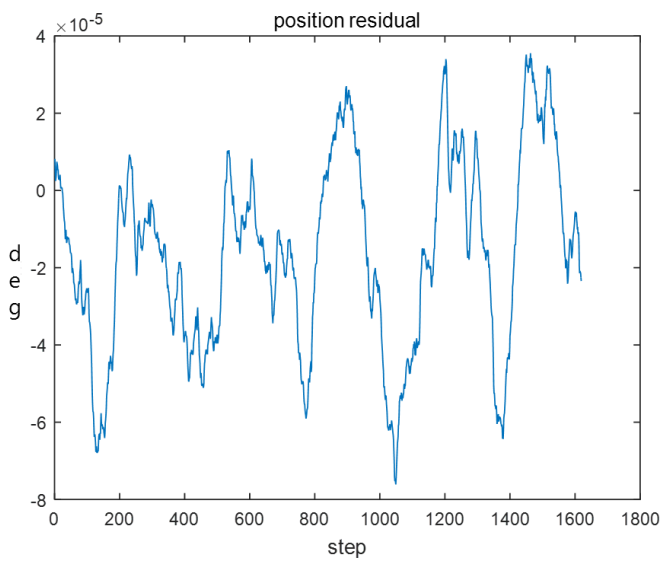


Figure 4-20. Position residual

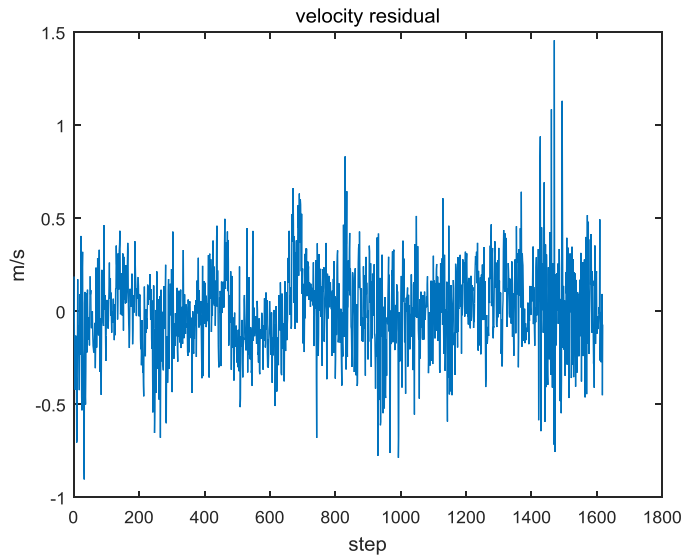


Figure 4-21. Velocity residual

Table 4-5. GPS carrier / INS integrated pedestrian dead reckoning performance.

	Position residual N (deg)	Position residual E (deg)	Velocity residual N (m/sec)	Velocity residual E (m/sec)
NMEA	2.5042e-05	2.6971e-05	0.16	0.21
Mean	2.60065e-05		0.18555	
Filter	2.4049e-05	2.627e-05	0.18	0.14
Mean	2.51595e-05		0.15845	

The carrier based velocity and INS pedestrian dead reckoning information improved the position and velocity accuracy than only NMEA.

Chapter 5. User Context Classification Deep learning for Adaptive PDR

5.1 Smartphone Location / Walking Context Classification

The user walking context can affect the INS pedestrian navigation. For example, the texting context can produce the shaking of the phone. If the choosen threshold is lower, the step can be detected by this action. Also, the walking signal can be different by the phone's location and the hand moving. If the user walks swinging the hand, the signal will have a different period. Moreover, the smartphone location can contribute to changing of the signal for example, it is placed inside the backpack and pocket.

For these reasons, the determination of the walking context is very important for the pedestrian dead reckoning and the changeable threshold, parameters, and algorithms is needed to apply.

A deep learning approach could be used to classify user context. About the time series model, the performance is influenced by the input transformation(or encoding). The author study about the comparsion of several different transformations effect for a single deep learning model and same data set.

5.1.1. Smartphone Location / Walking Context Dataset

The various users and walking mode data set are hard to make so that the author used the open data set which is introduced from "Walking Detection Step Counting An Unconstrained Smartphone" experiment [22]. That paper only uses that data for checking the walking detection and step counting performance but the mode context classification is considered in this research.

There was 7 class in the data. And the dataset was liked below.

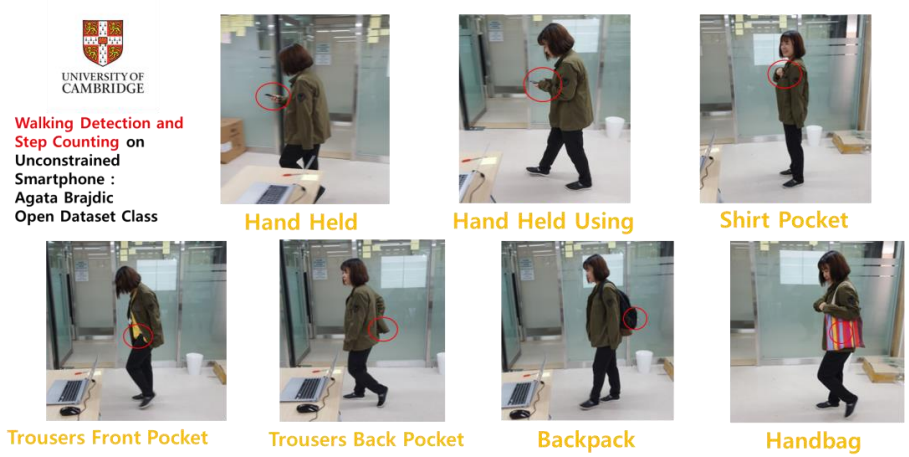


Figure 5-1. Walking context and phone location classes

Table 5-1. Used dataset information

class	Handheld	Handheld using	Shirt Pocket	Trousers front pocket	Trousers back pocket	Backpack ,Handbag	
original class	1	2	3	4	5	6	7
dataset number	27	27	24	25	21	21	6
User information	27 participants were asked to walk a route at three different walk paces: starting with normal, followed by fast, and ending with slow.						

Because the 7 class(dataset was small, two class(6,7) merged as 1 class. The training and test dataset divided 6:4

5.1.2. Deep Learning Models

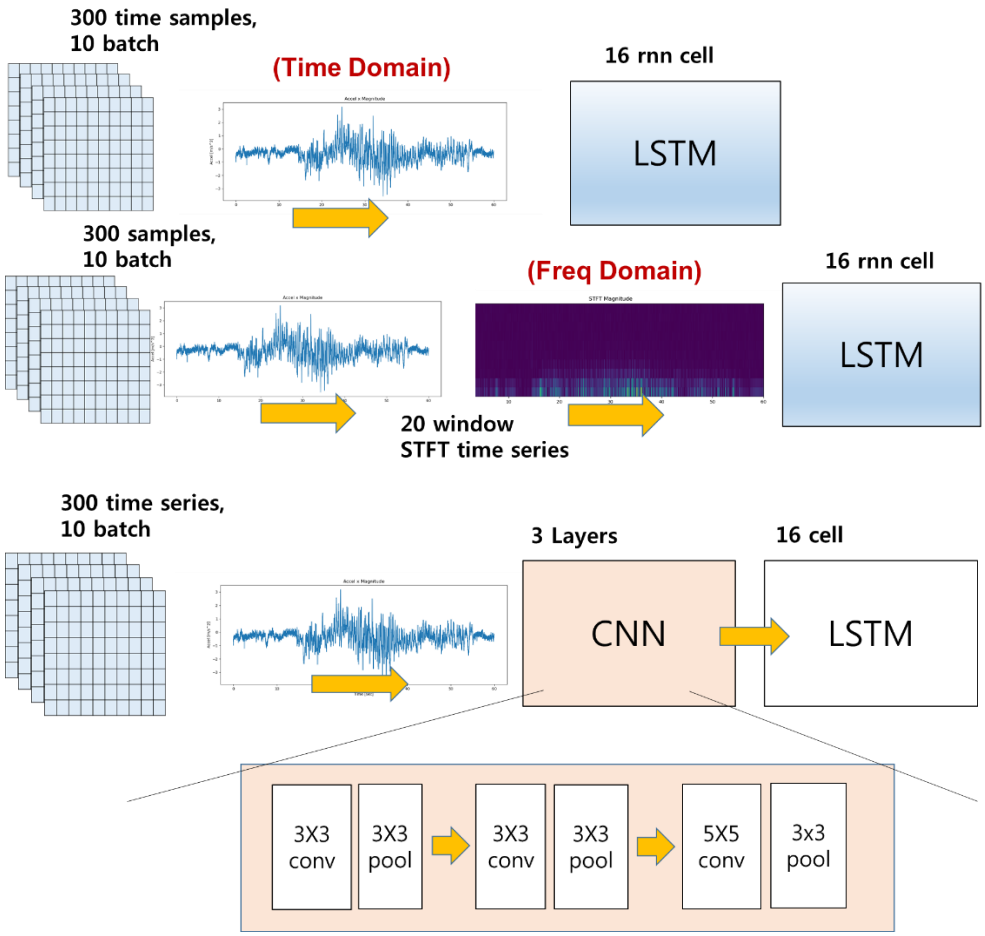


Figure 5-2. Model diagrams

Table 5-2. Model information

Num	1	2	3
Model	Window Cutting Batch + LSTM	CNN + LSTM	STFT + LSTM
Input	300 samples	300 samples,	300 samples,

	10 batch	10 batch	10 batch
Transform		conv2d (3x3) pooling (3x3) conv2d (3x3) pooling (3x3) conv2d (5x5) pooling (3x3)	20 windows STFT
	16 cell 1 layer LSTM	16 cell 1 layer LSTM	16 cell 1 layer LSTM
Loss, Regularizer, Optimizer	Loss: soft max cross entropy Regularizer : L2 regularizer Optimizer : Adams optimizer		

5.1.3. Comparison of Input Transformations

With these models, the author trained the training data. The results is given.

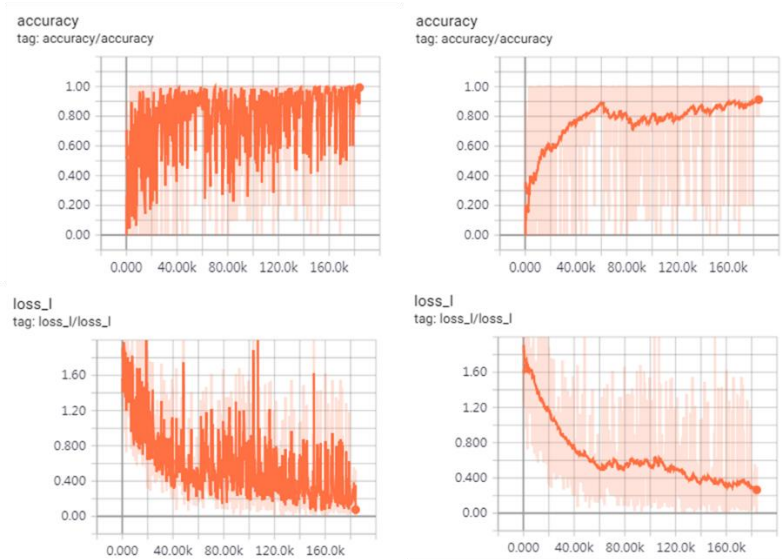


Figure 5-3. Model 1. Train Result

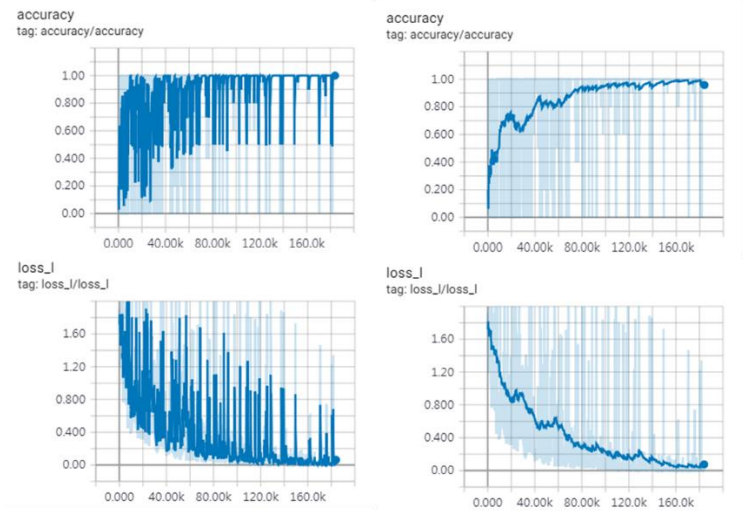


Figure 5-4. Model 2. Train Result

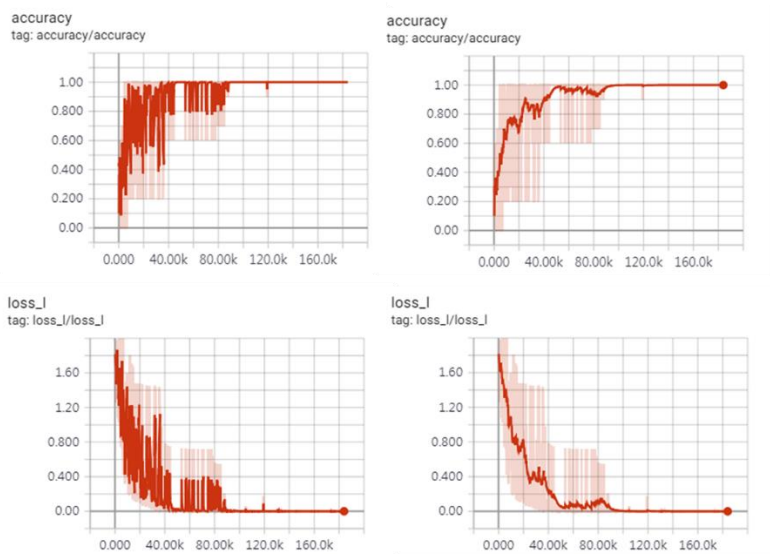


Figure 5-5. Model3. Train Result

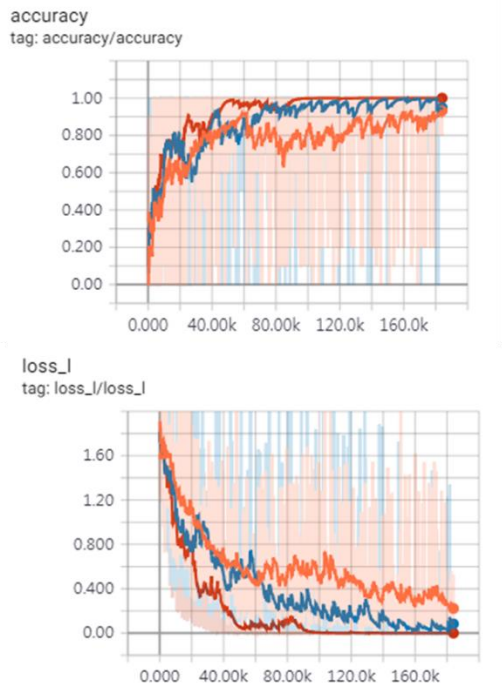


Figure 5-6. Training Result

Table 5-3. Training Result

	Model 1	Model 2	Model 3
Loss	0.2	0.1	0.001
Accuracy (1=100%)	0.85	0.95	1

Because of the small datasetset, the loss is pertubated. But the trend shows the classifying model can be trained. If there is more data, the smoother graph can be achieved.

The Model 3 shows the best training performance. The STFT transform had most fast training speed and converges 1 at 100000 iterations. The CNN encoding gave 0.95 accuracy and just time cutting input results in 0.85 accuracy. It shows the frequency domain analysis is more suitable than time domain in the sensor signal pattern learning. This could be an overfitting model to the training set but we just compare the training performance of each model in this research.

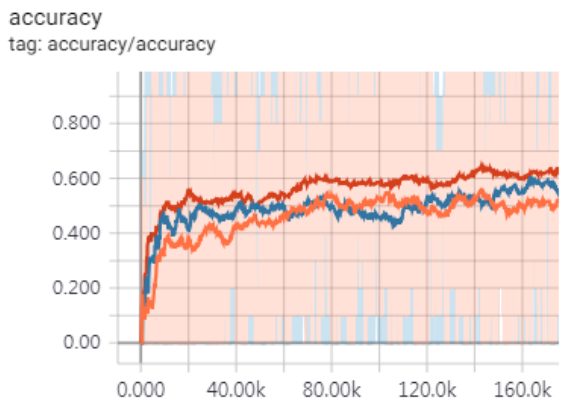


Figure 5-7. Test Result

Table 5-4. Test Result

	Model 1	Model 2	Model 3
Accuracy (1=100%)	0.5	0.6	0.65

The test set performance result shows the same rank (STFT > CNN > Time cutting). The STFT transform input format shows the best performance. But the model's accuracy is constrained because of overfitting. If we find a more suitable model structure, we can get the high accuracy. With this comparison of several models structure performance in the PDR user context classification, we can find the different input feature can influence the result accuracy. This research is performed with Cambridge open data set for comparing the performance of input feature difference. To apply this model for Galaxy S8, we need to get a training data which is acquired from S8. With transfer learning, we could get the proper model for S8 and finally the adaptive PDR parameter control for the different user context could be performed.

Chapter 6. Conclusion & Future work

In this research, the GPS / INS smartphone based pedestrian dead-reckoning was described in detail. The GPS raw carrier phase measurement was used to velocity determination. By performing the cycle slip compensation, the author got the more accurate velocity. Also, GPS/ INS integrated pedestrian navigation showed the improved performance in the position, velocity determination than NMEA.

Moreover, the deep learning scheme was considered in a user walking context classification. The models showed the different performance by transformation method. The STFT gave the best performance and the CNN encoded time-cutting showed the more suitable performance than just time-cutting.

For future research, the INS pedestrian dead-reckoning accuracy needed to be improved for a better result. Moreover, the amount and diversity of dataset will be helpful for the enhancement of the deep learning model accuracy.

Bibliography

[1] 김준성 et al. "안드로이드 GPS 원시데이터의 의사거리를 이용한 측위 정확도 비교". 한국항행학회논문지, vol. 21, pp.514-519, 2017.

Joonseong Gim et al. "Comparison of Positioning Accuracy Using the Pseudorange from Android GPS Raw Measurements". The Journal of Korea Navigation Institute, vol. 21, pp.514-519, 2017.

[2] E. Realini, S. Caldera, L. Pertusini, and D. Sampietro, "Precise GNSS Positioning Using Smart Devices," *Sensors*, vol. 17, no. 10, p. 2434, Oct. 2017.

[3] T. E. Humphreys, M. Murrian, F. van Diggelen, S. Podshivalov and K. M. Pesyna, "On the feasibility of cm-accurate positioning via a smartphone's antenna and GNSS chip," *2016 IEEE/ION Position, Location and Navigation Symposium (PLANS)*, Savannah, GA, 2016, pp. 232-242. doi: 10.1109/PLANS.2016.7479707

[4] G. Lachapelle, P. Gratton, J. Horrelt, E. Lemieux, and A. Broumandan, "Evaluation of a Low Cost Hand Held Unit with GNSS Raw Data Capability and Comparison with an Android Smartphone," *Sensors*, vol. 18, no. 12, p. 4185, Nov. 2018.

[5] R. Warnant, L. Van De Vyvere, and Q. Warnant, "Positioning with Single and Dual Frequency Smartphones Running Android 7 or Later," 2018.

[6] R. TRIGGS, "What is the Xiaomi Mi 8's dual frequency GPS? ," Android Authority, June, 26, 2018. [Online]. Available: https://www.androidauthority.com/dual-frequency-gps-878169/

[7] P. Goyal, V. J. Ribeiro, H. Saran and A. Kumar, "Strap-down Pedestrian Dead-Reckoning system," *2011 International Conference on*

Indoor Positioning and Indoor Navigation, Guimaraes, 2011, pp. 1-7. doi: 10.1109/IPIN.2011.6071935

[8] A. R. Pratama, Widyawan and R. Hidayat, "Smartphone-based Pedestrian Dead Reckoning as an indoor positioning system," *2012 International Conference on System Engineering and Technology (ICSET)*, Bandung, 2012, pp. 1-6.doi: 10.1109/ICSEngT.2012.6339316

[9] Y. Jin, Hong-Song Toh, W. Soh and Wai-Choong Wong, "A robust dead-reckoning pedestrian tracking system with low cost sensors," *2011 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, Seattle, WA, 2011, pp. 222-230. doi: 10.1109/PERCOM.2011.5767590

[10] 김현욱, "개인항법시스템의 방향각 추정을 위한 자세 추정필터,"서울대학교대학원, 서울, 국내석사학위논문, 2009.

[11] W. Zhang, X. Li, D. Wei, X. Ji and H. Yuan, "A foot-mounted PDR system based on IMU/EKF+HMM+ZUPT+ZARU+HDR+compass algorithm," *2017 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, Sapporo, 2017, pp. 1-5. doi:10.1109/IPIN.2017.8115916

[12] Y. Shu, C. Bo, G. Shen, C. Zhao, L. Li and F. Zhao, "Magicol: Indoor Localization Using Pervasive Magnetic Field and Opportunistic WiFi Sensing," in *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 7, pp. 1443-1457, July 2015. doi:10.1109/JSAC.2015.2430274

[13] V. Radu and M. K. Marina, "HiMLoc: Indoor smartphone localization via activity aware Pedestrian Dead Reckoning with selective crowdsourced WiFi fingerprinting," *International Conference on Indoor Positioning and Indoor Navigation*, Montbeliard-Belfort, 2013, pp. 1-10.

doi: 10.1109/IPIN.2013.6817916
[14] R. Ban, K. Kaji, K. Hiroi and N. Kawaguchi, "Indoor positioning method integrating pedestrian Dead Reckoning with magnetic field and WiFi fingerprints," *2015 Eighth International Conference on Mobile Computing and Ubiquitous Networking (ICMU)*, Hakodate, 2015, pp. 167-172. doi: 10.1109/ICMU.2015.7061061
[15] L. Hsu, Y. Gu, Y. Huang and S. Kamijo, "Urban Pedestrian Navigation Using Smartphone-Based Dead Reckoning and 3-D Map-Aided GNSS," in *IEEE Sensors Journal*, vol. 16, no. 5, pp. 1281-1293, March1, 2016. doi: 10.1109/JSEN.2015.2496621
[16] P. Baranski, M. Polanczyk, and P. Strumillo, "Fusion of Data from Inertial Sensors, Raster Maps and GPS for Estimation of Pedestrian Geographic Location in Urban Terrain," (in English), vol. 18, no. 1, p. 145, 2011.
[17] T. Zebin, P. J. Scully and K. B. Ozanyan, "Human activity recognition with inertial sensors using a deep learning approach," *2016 IEEE SENSORS*, Orlando, FL, 2016, pp. 1-3. doi: 10.1109/ICSENS.2016.7808590
[18] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine," in *International workshop on ambient assisted living*, 2012, pp. 216-223: Springer.
[19] C. A. Ronao and S.-B. J. E. S. w. A. Cho, "Human activity recognition with smartphone sensors using deep learning neural networks," vol. 59, pp. 235-244, 2016.
[20] S. Kaghyan, H. J. I. J. o. I. M. Sarukhanyan, and I. I. S. S. Analysis , Bulgaria, "Activity recognition using K-nearest neighbor algorithm on

smartphone with Tri-axial accelerometer," vol. 1, pp. 146-156, 2012.

[21] B. Shin *et al.*, "Motion Recognition-Based 3D Pedestrian Navigation System Using Smartphone," in *IEEE Sensors Journal*, vol. 16, no. 18, pp. 6977-6989, Sept.15, 2016. doi: 10.1109/JSEN.2016.2585655

[22] A. Brajdic and R. Harle, "Walk detection and step counting on unconstrained smartphones," presented at the Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing, Zurich, Switzerland, 2013.

[23] M. Edel and E. Köppe, "An advanced method for pedestrian dead reckoning using BLSTM-RNNs," *2015 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, Banff, AB, 2015, pp. 1-6. doi: 10.1109/IPIN.2015.7346954

[24] J. Kang, J. Lee, and D.-S. Eom, "Smartphone-Based Traveled Distance Estimation Using Individual Walking Patterns for Indoor Localization," vol. 18, no. 9, p. 3149, 2018.

[25] Android Developer, Raw GNSS Measurements. (2019) [Online]. Available: https://developer.android.com/guide/topics/sensors/gnss

[26] N. Idris, " Android Location Providers (gps, network, passive)," *developerlife.com*, Oct. 20, 2010. [Online]. Available: https://developerlife.com/2010/10/20/gps/.

[27] Android Developer, SensorManger. (2019) [Online]. Available: https://developer.android.com/reference/android/hardware/SensorManager

[28] A. M. Sabatini, "Quaternion-based extended Kalman filter for determining orientation by inertial and magnetic sensing," in *IEEE Transactions on Biomedical Engineering*, vol. 53, no. 7, pp. 1346-1356, July 2006. doi: 10.1109/TBME.2006.875664

[29]김문기, "실시간 Cycle Slip 검출과 보상을 통한 강건한 의사위성/초저가 IMU 결합 실내 항법 시스템," 서울대학교 대학원, 서울, 국내석사학위논문, 2017.
[30] M. J. I. G. PETOVELLO, "How does a GNSS receiver estimate velocity?," pp. 38-41, 2015.

초 록

본 논문에서는 스마트폰 Galaxy S8 환경에서 GPS / INS 결합 보행자 항법을 수행하였으며, 스마트폰 센서의 특성을 자세히 분석하였다. 이에 최근 공개된 Android GNSS API 를 사용하여 GPS 원시데이터를 항법에 이용하면서, Cycle slip 을 보정한 Carrier phase 를 이용한 속도 결정법을 사용하였다. 이로 인해 기존의 NMEA GPS 를 사용한 방식의 스마트폰 보행자 항법보다 정밀한 위치, 속도 항법이 가능하였고, 성능을 향상 시켰다. 또한 사용자 상황 분석이 가능한 분류 딥러닝 기법을 사용하여 각 보행 상황에 따른 분류감지가 가능하였음 보였으며, LSTM 의 입력부분을 변화한 몇가지 딥러닝 모델의 성능을 비교하였다. 이를 통해서 사용자의 보행 상황에 따른 적응적 보행자 항법 파라메터 결정이 가능함의 가능성을 보였다.

주요어 : GNSS, 안드로이드 GNSS API, 반송파, 사이클 슬립, 관성센서, 확장 쿼터니안 칼만필터, 보행자 항법, GPS / INS 결합항법, 센서 데이터 딥러닝, 사용자 상황인지, LSTM.

학번 : 2016-20743