平成 26 年度　特別研究報告書

端末間の相対距離と GPS 衛星の誤差傾向について

龍谷大学　理工学部　情報メディア学科

T110487　松原　元希

指導教員　三好　力　教授

概要

近年、様々な機能を搭載したスマートフォンの普及が進んでおり、GPS を使うことで誰もが容易に現在位置を知ることが可能となった。しかし、受信機の時計、衛星の送信情報、電離層やマルチパスなどから引き起される電波伝搬の状態などによって推定位置には誤差が生じる。ここで GPS の誤差要因は、受信機の時計による誤差を除けば、誤差の多くが衛星からの情報や伝搬状況に依存している。すなわち、比較的近距離にある複数の端末では同一衛星からの情報は同一であり、伝搬状況は類似しており、推定位置の誤差は傾向が類似しているのではないかと考えた。本研究では、端末間の相対距離の誤差は単体の推定位置の誤差より小さいのではないのかと考え、Android 端末を用いてプログラミングを行い実際に野外での検証実験を行った。結果としては、傾向は観測できた。

目次

第 1 章　はじめに.....1
第 2 章　既存技術.....3
　2.1 GPS の仕組みについて.....3
　　2.1.1　GPS による位置測定の誤差の要因.....4
　2.2　NMEA 形式のデータについて.....5
　2.3　既存技術における問題点.....8
第 3 章　提案手法.....9
　3.1　提案手法.....9
　3.2　考え方について.....9
第 4 章　実験概要.....11
　4.1　実験目的.....11
　4.2　開発環境.....11
　4.3　実験概要.....11
　4.4　実験結果.....12
　4.5　考察.....14
第 5 章　まとめ.....15
　5.1　まとめ.....15

第１章　はじめに

日本は地震大国であることに加え、津波や火山の噴火など、全国各地自然災害が頻繁に起こっている。1995 年の阪神淡路大震災や 2004 年の新潟中越大地震、そして 2011 年の東日本大震災といった大規模自然災害では多くの犠牲者をだしている。地震が発生し建物の倒壊などが起きた場合、早期に人を発見する必要があり、図 1-1 からも 72 時間以内に発見することができなければ脱水症状や低体温症などが原因で生存率が極端に下がってしまう。

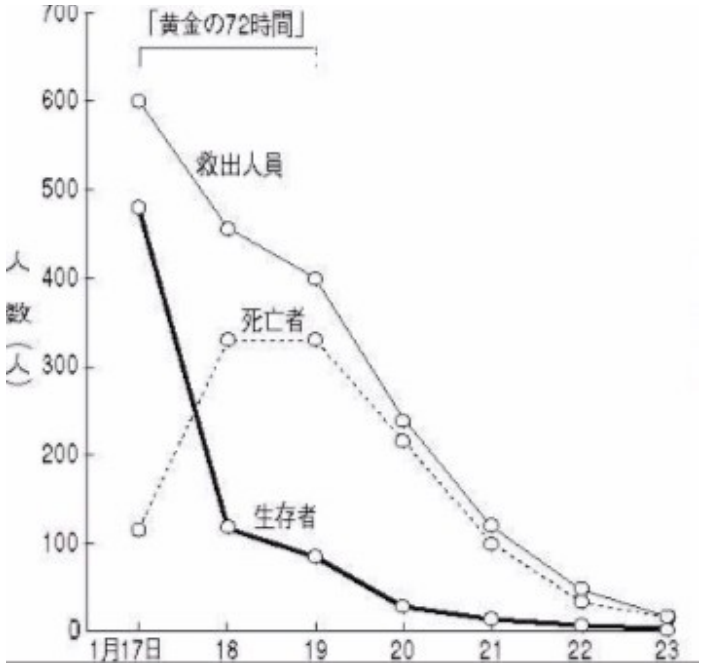

図 1-1　黄金の 72 時間について

近年では図 1-2 からも携帯の保有率はほぼ 100%となり基本的には常に身に着けているという状態である。すなわち、携帯を探すということは人を探すということと同意義であると言える。近年日本では多種多様な機能やアプリケーションによる機能の拡張が可能な、スマートフォンの普及がすすんでいる。その多くの機能の中でも本研究では、スマートフォンから取得可能な GPS 衛星の情報、特に NMEA 形式で取得可能なデータに着目する。

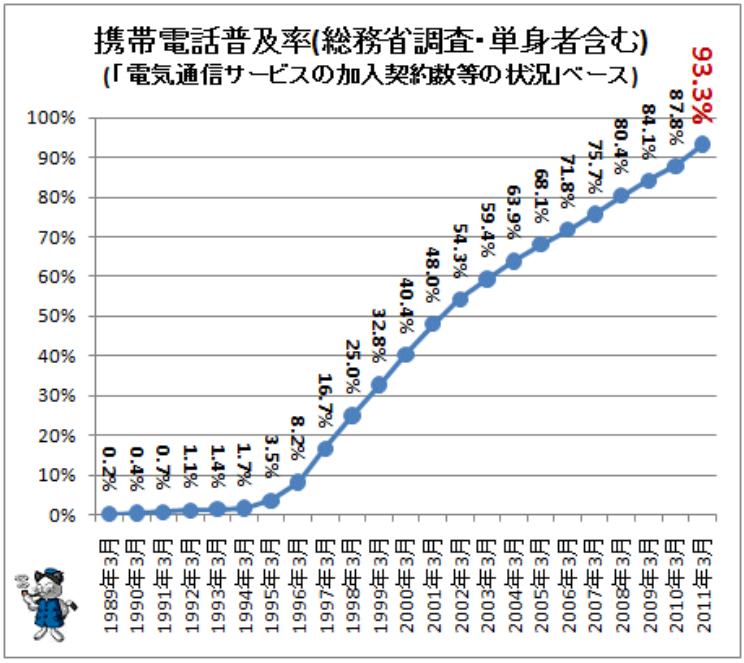

図 1-2　総務省調査の携帯普及率の推移

GPS はナビゲーションシステムなどで主に使用されている。ナビゲーションシステムは車などで使用されていることが多く、目的地を登録すると、正確に目的地まで案内してくれる。しかし、車ではなく、徒歩での移動の際など、自分の本当に正確な位置を測定しようとすると、その現在位置には数 10m 前後の誤差が生じてしまう。その原因としては、GPS は使用する端末や環境、情報取得可能な衛星数などによってその精度に変化が生じるからである。高層ビルなどの多い場所より開けた場所では単純に取得可能な衛星が増えることからも、その精度は大きく向上する。しかし、周囲に電波を遮断する建物が少ない場所で観測を行ったとしても、衛星から送信される電波が空気中などで屈折することなどで起こるマルチパスなどにより、GPS 情報に誤差が生じてしまう。

そこで本研究では探したい目的としている端末の相対位置を正確に探索するために、使用衛星と情報を取得する時間を統一するという手法を提案する。この手法を用いることにより、現在位置の正確な特定はできないが、取得した現在位置と実際の位置との誤差の大きさは、目的の端末でも、探す側の端末でも同様になるのではないかと考えた。これを他の端末から NMEA 形式のデータを取得できるとして、実際に Android 端末のスマートフォン端末を用いて検証するため実験を行い、検討する。

第 2 章　既存技術

2.1 GPS の仕組みについて

GPS とは Global Positioning System の略で、全世界で使用可能な位置測定システムである。GPS は 24 時間使用可能で、誰もが簡単に使用でき、誤差 10m 前後で位置を測定できるシステムのことを示す。その GPS 衛星は高度約 2 万 km のところを 30 個以上飛行おり、その中で最低でも 4 機以上 GPS 衛星が観測可能であれば、それぞれの GPS 衛星から、GPS 衛星の軌道情報、時間のデータを取得できる。以下に単独測位の場合の原理を説明する。

まず、複数の GPS 衛星から電波を発信した時刻などのデータが送信される。そのデータを受信機が受信した時刻と GPS 衛星から送信された時刻の差から算出された、電波が衛星と受信機を移動するのにかかった時間に、光の速さをかけることにより GPS 衛星から受信機間の距離を計測する。算出された時間を t 、光の速さを c とし、距離を D とする。この距離 D は地心直交座標で、電波を発した GPS 衛星の位置を(Xs,Ys,Zs)、受信機の位置を(X,Y,Z)とすれば、

$$D = \sqrt{(Xs - X)^2 + (Ys - Y)^2 + (Zs - Z)^2} = t \times c \qquad (1,1)$$

と表せる。3 つの衛星の情報を取得できれば t,c,Xs,Ys,Zs は取得できることにより、連立方程式を用いて X,Y,Z を算出することで現在位置を特定できる。

しかし、受信機の時計は GPS 衛星に搭載されている時計に比べ、精度が劣る。実際には水晶時計がスマートフォンには搭載されており、約 11.5 日で 1 秒の誤差が生じるが、GPS 衛星に搭載されているルビジウム、セシウム原子時計には約 300 年で 1 秒の誤差が生じる。上記のことから電波を送信する衛星側の時刻には誤差は少ないが、受信器側の現在時刻を求めるために用いる時計に誤差があり、それらの誤差を ε として考慮した式が、次式である。

$$D = \sqrt{\{(Xs - X)^2 + (Ys - Y)^2 + (Zs - Z)^2\}} + \varepsilon \times c \qquad (1.2)$$

求める変数にεが追加し、4つの衛星を使用し、式(1,2)に当てはめることによりX,Y,Z,εを求めることができる。 以上の原理により現在位置の測定を行っている。[1]

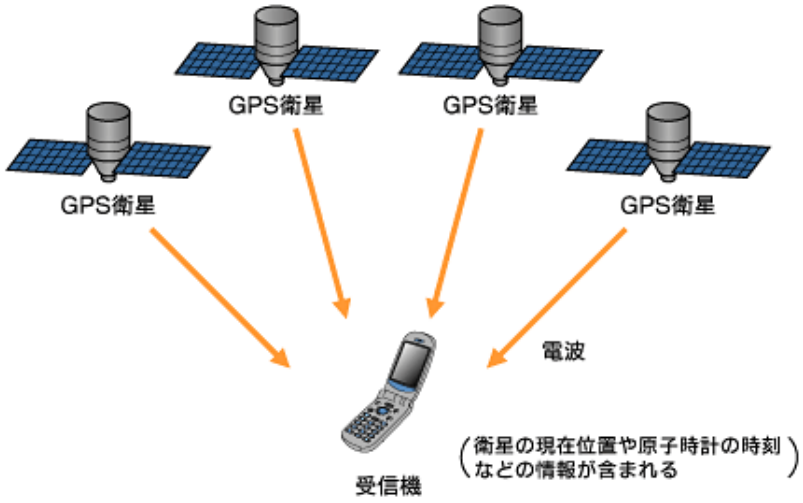

図 2-1　GPS 衛星 4 台による位置探索についての図

2.1.1　GPS による位置測定の誤差の要因

誤差の要因について。誤差の要因は衛星側と受信機側に分けて考えると、そのほとんどが衛星側に依存している誤差である。以下にその要因を示す。

・送られてくる衛星の時刻に生じる nanosec 単位での誤差

・衛星の現在位置情報（軌道情報）による誤差

・利用者側の受信機が時間補正を行う際に生じる現在時刻の誤差

・電離層での伝搬速度の変化によって生じる誤差

・マルチパスによる誤差

2.2 NMEA 形式のデータについて

NMEA フォーマットの情報は、センテンスの集まりである。1つのセンテンスは、「$」で始まり、「(改行(\r\n))」で終わり、センテンスは、「,」で区切られた単語の集まる。それぞれの単語の意味は、データタイプによって異なる。センテンスの最初の単語は、データタイプを表し、センテンスの最後の単語は、「*」以降がチェックサム値を表す。

表 2-1 はデータタイプが GPRMC の場合の単語の説明を示している。表 2-2 はデータタイプが GPGGA の場合の単語の説明を示している。表 2-3 はデータタイプが GPGSA の場合の単語の説明を示している。表 2-4 はデータタイプが GPGSV の場合の単語の説明を示している。表 2-5 はデータタイプが GPVTG の場合の単語の説明を示している。[3]

表 2-1 GPRMC の説明

単語例	説明	意味
85120.307	協定世界時(UTC)での時刻。日本標準時は協定世界時より9時間進んでいる。hhmmss.ss	UTC時刻:08時51分20秒307
A	ステータス。V = 警告、A = 有効	ステータス:有効
3541.1493	緯度。dddmm.mmmm 60分で1度なので、分数を60で割ると度数になる。Googleマップ等で用いられる ddd.dddd度表記は、(度数 + 分数/60)で得ることができる。	緯度:35度41.1493分
N	北緯か南緯か。N = 北緯、South = 南緯	北緯
13945.3994	経度。dddmm.mmmm 60分で1度なので、分数を60で割ると度数になる。Googleマップ等で用いられる ddd.dddd度表記は、(度数 + 分数/60)で得ることができる。	経度;139度45.3994分
E	東経か西経か。E = 東経、West = 西経	東経
0	地表における移動の速度。000.0～999.9[knot]	移動の速度:000.0[knot]
240.3	地表における移動の真方位。000.0～359.9度	移動の真方位:240.3度
181211	協定世界時(UTC)での日付。ddmmyy	UTC日付:2011年12月18日
	磁北と真北の間の角度の差。000.0～359.9度	
	磁北と真北の間の角度の差の方向。E = 東、W = 西	
A	Differential(干渉測位方式), E = Estimated(推定)	モード:自律方式
*6A	チェックサム	チェックサム値:6A

表 2-2　GPGGA の説明

単語例	説明	意味
85120.307	協定世界時(UTC)での時刻。日本標準時は協定世界時より9時間進んでいる。hhmmss.ss	UTC時刻:08時51分20秒307
3541.1493	緯度。dddmm.mmmm 60分で1度なので、分数を60で割ると度数になる。Googleマップ等で用いられる ddd.dddd度表記は、(度数 + 分数/60) で得ることができる。	緯度:35度41.1493分
N	北緯か南緯か。N = 北緯、South = 南緯	北緯
13945.3994	経度。dddmm.mmmm 60分で1度なので、分数を60で割ると度数になる。Googleマップ等で用いられる ddd.dddd度表記は、(度数 + 分数/60) で得ることができる。	経度;139度45.3994分
E	東経か西経か。E = 東経、West = 西経	東経
1	位置特定品質。0 = 位置特定できない、1 = SPS(標準測位サービス)モード、2 = differenctial GPS(干渉測位方式)モード	位置特定品質:SPS(標準測位サービス)モード
8	使用衛星数	使用衛星数:8個
1	水平精度低下率	水平精度低下率:1.0
6.9	アンテナの海抜高さ	アンテナの海抜高さ:6.9[m]
M	[m]	メートル
35.9	ジオイド高さ	ジオイド高さ:35.9[m]
M	[m]	メートル
	DGPSデータの最後の有効なRTCM通信からの時間。空 = DGPS不使用	DGPS不使用
0	差動基準地点ID	差動基準地点ID:0000
*5E	チェックサム	チェックサム値:5E

表 2-3 GPGSA の説明

単語例	説明	意味
A	モード。M = 手動、A = 自動	モード:自動
3	特定タイプ。1 = 存在しない。2 = 2D特定。3 = 3D特定	特定タイプ:3D特定
29,26,05,10,02,27,08,15,,,,,	衛星番号。最大12個列挙。	衛星番号:29,26,05,10,02,27,08,15,,,,,
1.8	位置精度低下率	位置精度低下率:1.8
1	水平精度低下率	水平精度低下率:1.0
1.5	垂直精度低下率	垂直精度低下率:1.5
*3E	チェックサム	チェックサム値:3E

表 2-4　GPGSV の説明

単語例	説明	意味
3	総GSVセンテンス数	総GSVセンテンス数:3個
1	このセンテンスの番号	3個中の1個目のセンテンス
12	ビュー内の総衛星数	ビュー内の総衛星数:12個
26	衛星番号	衛星番号:26
72	衛星仰角。00～90度	衛星仰角:72度
352	衛星方位角。000～359度	衛星方位角:352度
28	C/No（キャリア／ノイズ比）。00～99dB	C/No:28dB
5	衛星番号	衛星番号:05
65	衛星仰角。00～90度	衛星仰角:65度
66	衛星方位角。000～359度	衛星方位角:66度
37	C/No（キャリア／ノイズ比）。00～99dB	C/No:37dB
15	衛星番号	衛星番号:15
50	衛星仰角。00～90度	衛星仰角:50度
268	衛星方位角。000～359度	衛星方位角:268度
35	C/No（キャリア／ノイズ比）。00～99dB	C/No:35dB
27	衛星番号	衛星番号:27
33	衛星仰角。00～90度	衛星仰角:33度
189	衛星方位角。000～359度	衛星方位角:189度
37	C/No（キャリア／ノイズ比）。00～99dB	C/No:37dB
*7F	チェックサム	チェックサム値:7F

表 2-5　GPVTG の説明

単語例	説明	意味
204.3	地表における移動の真方位。000.0～359.9度	移動の真方位:240.3度
T	[True course]	True course
	地表における移動の磁方位。000.0～359.9度	
M	[Magnetic course]	Magnetic course
0	地表における移動の速度。000.0～999.9[knot]	移動の速度:0.0[knot]
N	[knot]	[knot]
0	地表における移動の速度。0000.0～1800.0[km/h]	移動の速度:0.0[km/h]
K	[km/h]	[km/h]
A	モード, N = データなし, A = Autonomous（自律方式）, D = Differential（干渉測位方式）, E = Estimated（推定）	モード:自律方式
*05	チェックサム	チェックサム値:05

2.3 既存技術における問題点

既存技術で最も問題となっていることは位置測定の正確性である。GPS によって生じた数mから数十 m の誤差は数十m以内にいる人探索を目的としている場合、数 m 以内の誤差でなければ探す対象の端末と自身の端末双方の誤差により算出される相対距離が絶対距離に対し大きく異なる恐れがあり、実用性は低いと言える。

第３章　提案手法

3.1　提案手法

タブレット端末の GPS で取得する推測された位置情報は実際の位置とは数 10m 前後の誤差が存在する事が多い。GPS を用いて位置推測を行った場合の、誤差の要因については、その多くは衛星側に依存している。そのことから、位置探索を行う際に「同時刻に同じ衛星から送信される信号」を取得することができれば、その誤差の傾向は一定になると推測した。

よって、複数端末間受信した NMEA データを交換するようなアプリを搭載しておくことにより、その NMEA データを受信した他の端末が、自身が取得する NMEA データの条件、すなわち「位置情報と各センテンスにおける衛星番号と時刻」を条件とし、これが同一であるデータを使用することにより、各端末で推測した絶対位置よりも誤差の少ない相対距離と相対位置を算出するシステムを提案する。

3.2　考え方について

図 3-1 に 1 端末で位置情報を取得した場合、推測される位置と推測される位置の誤差範囲を表す円と実際の位置を示す。GPS を用いて取得する推測された位置情報は図 3-1 のように実際の位置とは数 10m 前後の誤差が存在する。

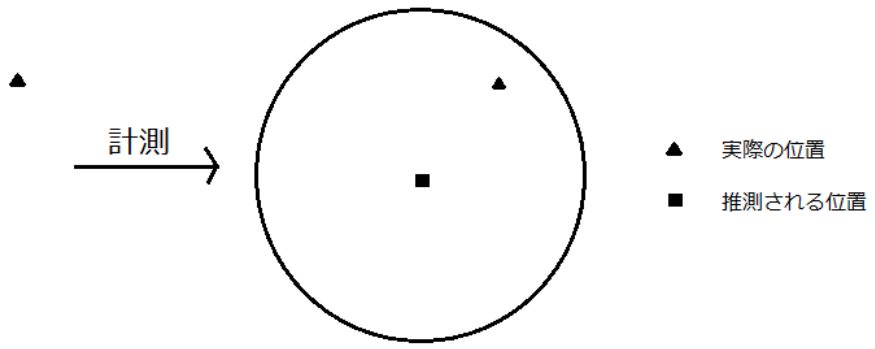

図 3-1　計測された位置情報について

図 3-2 に 2 端末でそれぞれ位置情報を取得した場合、推測される位置と推測される位

置の誤差範囲を表す円と実際の位置をそれぞれ示す。2 端末で別々に位置情報を取得した場合もそれぞれ同様に誤差は存在するが、「推測される位置同士の距離」と「実際の位置同士の距離」には大きな違いが存在することが図 3-2 からも推測される。

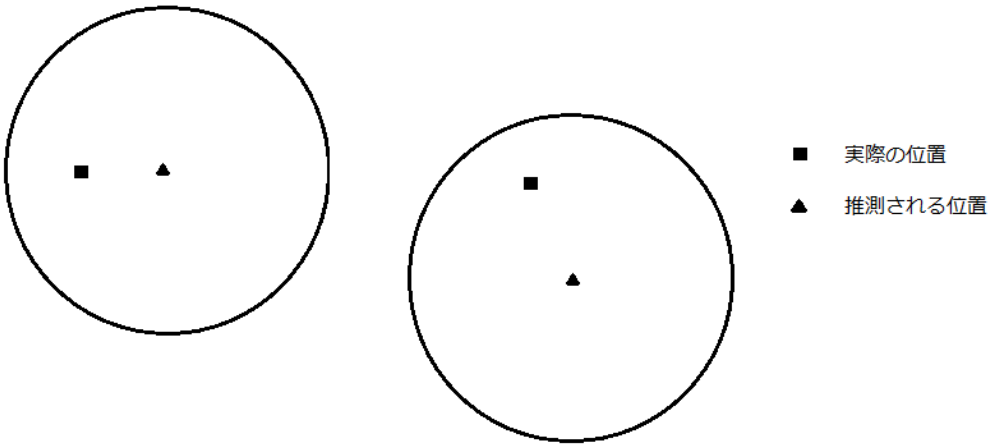

図 3-2　2 端末の位置とその誤差の円について

2.1.1 にも記したように GPS を用いて位置推測を行った場合、誤差の要因についてはその多くは衛星側に依存して生じている。すなわち、位置探索を行う際に「位置情報と各センテンスにおける衛星番号と時刻」を自身の端末と探したい目的としている端末双方において同一にすることにより、両端末で推測した絶対位置よりも誤差の少ない相対距離と相対位置を算出できるのではないかと考えた。図 3-3 に同一条件の場合の例を示す。

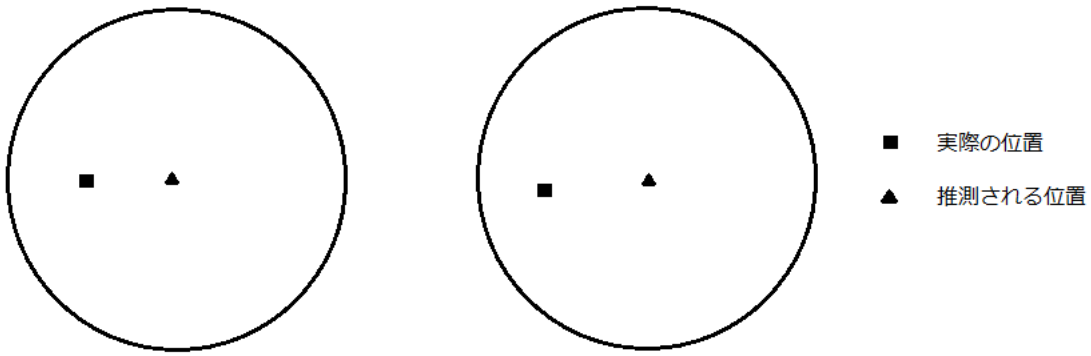

図 3-3　同一条件の場合の図

第 4 章　実験概要

4.1　実験目的

提案システムの実用性を調べるため、提案手法で求めた相対位置が各端末で推測した絶対位置に比べて、どれくらい正確かを調べる事を目的とする。

4.2　開発環境

今回自身の位置情報と使用衛星数、時刻を取得するために Android アプリを自作し使用した。[2]　[4]　[5]　[6]

実際の端末間の距離の測定にはレーザー距離計を用いた。表 4-1 に詳細を示す。

表 4-1　開発環境について

OS	Windows8.0
開発ツール	eclipse
開発言語	Java
タブレットの Android バージョン	3.2
レーザー距離計	GLM150

4.3　実験概要

ランダムに数 10m 離れたほぼ同条件の位置に端末を 1 台ずつ配置した。配置した両端末に位置情報、使用衛星数、使用衛星番号のデータが格納された NMEA 情報を 100 回取得するようにプログラミングを行い、取得された両端末の NMEA 情報の中から両端末の各センテンスにおける衛星番号と時刻が同一であるものを選択し、その位置情報から相対距離の算出を行う。

スマートフォンの Android 端末で取得した NMEA 形式の情報についてを表 4-2 に記載する。

表 4-2　NMEA 形式のデータ

$PGLOR	FIX	1.0*3E																	
$GPRMC	53626	A	3457.7936	N	13556.3489	E	0		130115			A*76							
$GPGGA	53626	3457.7936	N	13556.3489	E	1	5	0.5	173.8	M	34.3	M		*65					
$PGLOR	STA	53627	0.005	0	-2469	0	5	1	PWR	D*2F									
$PGLOR	SAT	15	38	1F	24	44	1F	13	36	1F	21	37	1F	5	48	1F	28	32	3*63
$PGLOR	SIO	TxERR	0	RxERR	0	TxCNT	200	RxCNT	1428	DTMS	1000	DTIN	1	TOUT	75*45				
$GPGSV	3	1	12	15	66	342	40	24	53	204	45	13	41	47	38	21	39	286	39*7A
$GPGSV	3	2	12	5	32	119	48	18	25	310	12	28	12	53	32	42	49	172	*74
$GPGSV	3	3	12	50	48	164		26	41	48		40	6	259		12	0	170	*74
$GPGSA	A	3	5	13	15	21	24								1.9	1.3	4*3B		

表 4-2 について、今回実験に使用したデータは位置情報と現在時刻の取得のために GPRMC を、現在位置測定に使用した衛星の取得のために GPGSV を使用した。表 4-2 の順でこれを 1 セットとし、一秒ごとに出力するように設定を行った。GPRMC で取得した緯度経度を 60 進数から 10 進数に変換を行い、2 端末のユークリッド距離を算出し、相対距離とした。

評価については 2 点間の距離の誤差について、相対距離と絶対距離の差を誤差とし、その「両端末の各センテンスにおける衛星番号と時刻が同一」である場合と「全データ」の場合で、その誤差について平均と分散を算出し、比較を行った。

4.4　実験結果

2 端末を 30m 離したランダムな位置に端末を配置し、同一箇所で 100 回位置情報を取得することを 20 回行った。表 4-3 は条件を満たしている場合と全データの場合について、2 端末間の距離にどれだけ誤差があるかを平均と分散を算出し、比較を行った結果である。

表 4-3　2 端末間の距離誤差の平均

	平均(m)	分散(m)
条件一致	10.467	5.343
全データ	11.656	6.720

表 4-4 は、条件一致に比べ全データについては誤差がどれだけ軽減されているのかを算出した表である。

表 4-4　誤差の軽減率について

	平均(%)	分散(%)
誤差の軽減率	10.199	20.487

表 4-3、表 4-4 から 2 端末間の相対距離の平均と分散は軽減されていることがわかった。

今回タブレット端末で位置情報を取得した際に、出力された位置情報はそれ以前に計測された位置情報に依存して変動していることが多かった。このことから使用衛星が連続して一様である場合、どれだけ誤差が軽減されているかを求めるため、両端末の各センテンスにおける衛星番号と時刻が一致し、なおかつその条件でデータが連続する場合、その連続したデータの「最初と最後の相対距離と絶対距離の差」を平均して算出した。結果として 0.392m 減っていることが確認された。表 4-5 に計算に用いたデータの一例を示す。

表 4-5　連続データの相対距離と絶対距離の差について

	時間	相対距離と絶対距離の差
条件不一致	34802	16.3983205513
条件一致	34803	16.2632264275
条件一致	34804	16.1479613135
条件一致	34805	16.0674992627
条件一致	34806	16.0487920518
条件一致	34807	16.0352315944
条件一致	34808	16.0260560376
条件一致	34809	15.9875782702
条件一致	34810	15.9262052257
条件一致	34811	15.9148039079
条件不一致	34812	15.8937587833

4.5 考察

以上の実験結果から考察する。表 4-4 より誤差が平均では約 10％軽減され、分散では約 20％軽減されているという結果となった。誤差の軽減率が大きくならなかったことの要因としては受信機のキャリアノイズ比が、受信機において同一衛星からデータを取得していた場合でも数値に違いがある場合があったということと、出力された位置情報はそれ以前に計測された位置情報に依存して変動していることが多かったことから、位置情報の大きな修正が行われなかったことが軽減率の低下を引き起こしたと推測される。しかし、両端末の各センテンスにおける衛星番号と時刻が一致し、なおかつその条件でデータが連続する場合、その連続したデータの「最初と最後の相対距離と絶対距離の差」が約 0.4m 改善されたことからも使用衛星が一様であれば誤差は軽減された。

第５章　まとめ

5.1　まとめ

本論文では、2 端末の各センテンスにおける衛星番号と時刻が同一である場合、端末間の相対位置が各端末で推測した絶対位置に比べて、どれくらい正確になるのかということを検証するため、スマートフォンの Android 端末を用いてプログラミングを行い、実際に野外で検証実験を行った。

実験結果より、誤差は軽減され目的の端末をより正確に探索できるようになった。よって両端末の各センテンスにおける衛星番号と時刻が一致する場合、誤差は軽減するということが確認された。

出力された位置情報はそれ以前に計測された位置情報に依存して変動していることから、位置測定を行っている各センテンスごとの位置情報を用いて検証した場合は、誤差を小さく計測できる可能性はあると推測される。同様に、マルチパスなどの要素をある程度限定するために、受信する衛星数を減らすなど、受信環境を変えるなど、より正確な位置情報を求める手法の検討が今後の課題である。

謝辞

本論文作成に当たり、日々ご指導いただいた三好力教授には大変深く感謝いたします。また、研究を通じて活発な議論にお付きいただいた三好研究室の皆様や、学友の皆様には深く感謝します。

参考文献

[1] 溝口 早苗, トランジスタ技術編集部 GPS のしくみと応用技術 2009 年
CQ 出版社

[2] Android Developers
http://developer.android.com/intl/ja/reference/android/location/LocationManager.html

[3] GPS の NMEA フォーマット
http://www.hiramine.com/physicalcomputing/general/gps_nmeaformat.html

[4] 中西 葵,中村 祐之,高橋良司 Android SDK 逆引きハンドブック 2012 年
C&R 研究所

[5] 吉井 博史 Android SDK C&R 研究所

[6] Ed Burnette 訳 長尾 高弘 初めての Android O'REILLY

付録

```
package com.example.nmea_mark1;

import java.io.BufferedWriter;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStreamWriter;
import android.app.Activity;
import android.location.GpsStatus.NmeaListener;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.location.LocationProvider;
import android.os.Bundle;
import android.os.Environment;
import android.widget.TextView;
import android.widget.Toast;

public class main extends Activity implements LocationListener, NmeaListener {
	int filecount = 0;
	private LocationManager mLocationManager;
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        mLocationManager = (LocationManager) getSystemService(LOCATION_SERVICE);
        mLocationManager.addNmeaListener(this);
		// requestLocationUpdates と連動して動きます。
		LocationProvider p = mLocationManager.getProvider("gps");
		mLocationManager.requestLocationUpdates(p.getName(), 0, 0, this);
    }
	@Override
	public void onNmeaReceived(long timestamp, String nmea) {
		// TODO 自動生成されたメソッド・スタブ
		String[] data = nmea.split(",");
		// GPGSV,GPGSA,GPRMC,GPVTG,GPGGA
		if (data[0].equals("$GPGGA")) {
		TextView tv = (TextView)findViewById(R.id.nmea_gpgga);
		tv.setText("現在地：" + nmea.trim());
		if(filecount < 500){
			//file 保存先
			String filePath= Environment.getExternalStorageDirectory() + "/sample1122_1.txt";
			// openFileOutput の宣言
			FileOutputStream fos = null;
			//String に戻すと時刻の出力がなぜか可能
			String need_data = "time:" + data[1] + ",ido:" + data[3] + data[2] + ",keido:" + data[5] +data[4];
			if(filecount == 100)Toast.makeText(this, "20%", Toast.LENGTH_LONG).show();
			if(filecount == 250)Toast.makeText(this, "50%", Toast.LENGTH_LONG).show();
			if(filecount == 499)Toast.makeText(this, "完了", Toast.LENGTH_LONG).show();
			try {
				fos = new FileOutputStream(filePath, true);
				BufferedWriter buf = new BufferedWriter(new OutputStreamWriter(fos));
				buf.write(need_data);
				buf.newLine();
				buf.close();
			} catch (FileNotFoundException e1) {
				e1.printStackTrace();
			} catch (IOException e) {
				e.printStackTrace();
			}
			FileOutputStream fas = null;
			FileOutputStream fus = null;
			filePath= Environment.getExternalStorageDirectory() + "/sample1122_2.txt";
			String filePath2= Environment.getExternalStorageDirectory() + "/sample1122_3.txt";
			try {
				fas = new FileOutputStream(filePath, true);
```

```
                    BufferedWriter buf = new BufferedWriter(new OutputStreamWriter(fas));
                    buf.write(data[1]);
                    buf.close();
                    fus = new FileOutputStream(filePath2, true);
                    BufferedWriter baf = new BufferedWriter(new OutputStreamWriter(fus));
                    baf.write(data[1]);
                    baf.close();
            } catch (FileNotFoundException e1) {
                    e1.printStackTrace();
            } catch (IOException e) {
                    e.printStackTrace();
            }
                    filecount++;
}
} else if (data[0].equals("$GPGSV")) {
            TextView tv001 = (TextView)findViewById(R.id.gpgsv001);
            TextView tv002 = (TextView)findViewById(R.id.gpgsv002);
            TextView tv003 = (TextView)findViewById(R.id.gpgsv003);
            TextView tv004 = (TextView)findViewById(R.id.gpgsv004);
            TextView tv005 = (TextView)findViewById(R.id.gpgsv005);
            //TextView tv006 = (TextView)findViewById(R.id.gpgsv006);
            //TextView tv007 = (TextView)findViewById(R.id.gpgsv007);
            //length
            int Num = Integer.valueOf(data[2]);
            int senum =Integer.valueOf(data[3]);
            String needgsvdata = "";
            int secount = 0;
            if (Num == 1){
                    tv001.setText(nmea.trim());
                    if(senum >= 4){
                            needgsvdata = "[" + data[3] + "]" + " , " + data[4] + " , " + data[8] + " , " +
                                            data[12] +  " , " +data[16];
                            secount++;
                    }
            }else if (Num == 2){
                    tv002.setText(nmea.trim());
                    if(senum >= 8){
                            needgsvdata = "[" + data[3] + "]" + " , " + data[4] + " , " + data[8] + " , " +
                                            data[12] +  " , " +data[16];
                            secount++;
                    }
            }else if (Num == 3){
                    tv003.setText(nmea.trim());
                    if(senum >= 12){
                            needgsvdata = "[" + data[3] + "]" + " , " + data[4] + " , " + data[8] + " , " +
                                            data[12] +  " , " +data[16];
                            secount++;
                    }
            }else if (Num == 4){
                    tv004.setText(nmea.trim());
                    if(senum >= 16)needgsvdata = "[" + data[3] + "]" + " , " + data[4] + " , " + data[8] + " , " +
                                    data[12] +  " , " +data[16];
            }else if (Num == 5){
                    tv005.setText(nmea.trim());
            }
            if(filecount < 500){
                    //file 保存先
                    String filePath= Environment.getExternalStorageDirectory() + "/sample1122_2.txt";
                    // openFileOutput の宣言
                    FileOutputStream fos = null;
                    //String に戻すと時刻の出力がなぜか可能
                    if(filecount == 100)Toast.makeText(this, "20%", Toast.LENGTH_LONG).show();
            if(filecount == 250)Toast.makeText(this, "50%", Toast.LENGTH_LONG).show();
                    if(filecount == 499)Toast.makeText(this, "完了", Toast.LENGTH_LONG).show();
                    try {
                            fos = new FileOutputStream(filePath, true);
                            BufferedWriter buf = new BufferedWriter(new OutputStreamWriter(fos));
                            buf.write(needgsvdata);
                            buf.newLine();
                            buf.close();
```

```
                                        } catch (FileNotFoundException e1) {
                                                e1.printStackTrace();
                                        } catch (IOException e) {
                                                e.printStackTrace();
                                        }

                                        String filePath2= Environment.getExternalStorageDirectory() + "/sample1122_3.txt";
                                        // openFileOutput の宣言
                                        FileOutputStream fus = null;
                                        //String に戻すと時刻の出力がなぜか可能
                                        try {
                                                fus = new FileOutputStream(filePath2, true);
                                                BufferedWriter buf = new BufferedWriter(new OutputStreamWriter(fus));
                                                buf.write(nmea);
                                                buf.newLine();
                                                buf.close();
                                        } catch (FileNotFoundException e1) {
                                                e1.printStackTrace();
                                        } catch (IOException e) {
                                                e.printStackTrace();
                                        }
                                        filecount++;

                                }
                        }
                }
                @Override
    protected void onResume() {
        if (mLocationManager != null) {
            mLocationManager.requestLocationUpdates(
                 LocationManager.GPS_PROVIDER,
//               LocationManager.NETWORK_PROVIDER,
                 0,
                 0,
                 this);
        }

        super.onResume();
    }

    @Override
    protected void onPause() {
        if (mLocationManager != null) {
            mLocationManager.removeUpdates(this);
        }

        super.onPause();
    }
                @Override
                public void onLocationChanged(Location location) {
                        // TODO 自動生成されたメソッド・スタブ
                }

                @Override
                public void onProviderDisabled(String arg0) {
                        // TODO 自動生成されたメソッド・スタブ

                }

                @Override
                public void onProviderEnabled(String arg0) {
                        // TODO 自動生成されたメソッド・スタブ

                }

                @Override
```

```
	public void onStatusChanged(String provider, int status, Bundle extras) {
		// TODO 自動生成されたメソッド・スタブ
	}
	protected void onDestroy() {
		super.onDestroy();
		mLocationManager.removeUpdates(this);
		mLocationManager.removeNmeaListener(this);
	}

}
```

main_activity.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.example.nmea_mark1.MainActivity" >

    <TextView
        android:id="@+id/nmea_gpgga"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true"
        android:layout_marginLeft="22dp"
        android:layout_marginTop="22dp"
        android:text="TextView" />

    <TextView
        android:id="@+id/gpgsv002"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/gpgsv001"
        android:text="TextView" />

    <TextView
        android:id="@+id/gpgsv001"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/nmea_gpgga"
        android:layout_marginTop="29dp"
        android:text="TextView" />

    <TextView
        android:id="@+id/gpgsv003"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignRight="@+id/gpgsv002"
        android:layout_below="@+id/gpgsv002"
        android:text="TextView" />

    <TextView
        android:id="@+id/gpgsv004"
```

```
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/gpgsv003"
        android:layout_below="@+id/gpgsv003"
        android:text="TextView" />

    <TextView
        android:id="@+id/gpgsv005"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/gpgsv004"
        android:layout_below="@+id/gpgsv004"
        android:text="TextView" />

</RelativeLayout>
```

Manifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.nmea_mark1"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="21" />
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="main"
            android:label="@string/app_name">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```