2012 32nd International Conference on Distributed Computing Systems Workshops

Performance Comparison of Positioning Algorithms for Complex GPS Systems

Wei Li[1] Zimu Yuan[1] Biao Chen[2] Wei Zhao[2]

[1] *Institute of Computing Technology, CAS*
[2] *University of Macau*
liwei@ict.ac.cn, yuanzimu@ict.ac.cn, bchen@umac.mo, weizhao@umac.mo

Abstract—**In this paper, we first present a classification for existing GPS positioning algorithms based on their features of mathematical operations in the calculating process and the accuracy of the solutions solved. Then we systematically evaluate and compare the performance of GPS algorithms for practical environments. Performance metrics include normalized execution time and the absolute error. We examine the performance of traditional Newton-Raphson algorithm as well as those newly proposed ones. We find that while the traditional Newton-Raphson algorithm does deliver a reasonable performance, others may over-perform it by a good margin.**

I. INTRODUCTION

Global Positioning System (GPS) is now widely utilized for positioning and navigational purposes. Highly accurate and super efficient GPS algorithms are critical to providing all-weather and all-time positioning services. The basic idea of GPS positioning techniques is the so-called *Trilateration Model*, which translates the positioning problem into a multivariate quadratic system [27].

Though Trilateration appears to be a simple mathematical problem, obtaining an accurate location is not a straightforward task, especially considering the existence of errors in the system. Basically, there are two main factors affecting the accuracy of positioning: *distance measurements* and *satellite geometry* [30]. Distance measurements will contain errors when satellites' signals transmit in the atmosphere. This brings errors into the quadratic system to be solved. On the other hand, the satellite geometry will magnify the above errors when solving that system [30]. In fact, large measurement errors and a bad geometry will bring significant errors into calculated locations. In order to obtain stable and accurate solutions, GPS algorithms need to carefully consider these factors.

Various algorithms have been proposed to solve the GPS problem. However, most of these algorithms focus on cases when there are no measurement errors. The effectiveness and efficiency of these algorithms have not been investigated systematically when measurement errors and redundant measurement are considered. For many researchers, an obvious question is, while theoretically these algorithms can find same or similar solutions, why is the Newton-Raphson algorithm is adopted as a de facto standard approach for GPS positioning in practice only?

In this paper, we intend to address this question. We first classify GPS systems into two kinds. The first kind is *simple systems*, which make simplicity assumptions and do not take into account measurement errors. A study of simple systems helps to reveal the mathematical nature of different algorithms. The second kind is *complex systems*, which are designed to deal with practical issues such as measurement errors and redundant measurements.

We start our evaluation by classifying the GPS algorithms based on their mathematical operations in the solution process. Based on the identified simple systems, a classification of GPS algorithms is presented. Basically, GPS algorithms can be classified as *direct* or *iterative* ones, depending on whether the solution is obtained directly or iteratively. Furthermore, GPS algorithms can be divided as exact and approximate ones, depending on whether an approximation is made in the solution process.

Representative algorithms from these classes are selected for performance analysis and comparison for complex systems that involve measurement errors and redundant measurements. We evaluate these algorithms in terms of performance metrics including *normalized execution time* and *the absolute error*. By comparing execution times, we quantitatively demonstrate the tradeoff between accuracy and execution time. The performance observations have not been previously reported and can be used as a guidance for further improvements on and selection of GPS algorithms.

The paper is organized as follows: Section 2 discusses the related work; Section 3 presents the model of trilateration and introduces a classification of GPS algorithms; In Section 4, we discuss strategies for complex systems and compare their performance. Section 5 concludes the paper with a discussion on future work.

II. PREVIOUS WORK

Various algorithms have been proposed based on a different understanding of Trilateration. Most of this work focuses on how to deal with a system of equations with or without measurement errors.

A comprehensive analysis of GPS algorithms has not been reported. No systematic classification of GPS algorithms has been proposed before. The work in [13] summaries previous direct algorithms such as [2] [3] [4] [6] [7]. However, it only covers the cases in which there are no measurement errors. Though algorithms supporting redundant measurements are mentioned, systematic analysis and comparison were not presented. Some literatures have presented the comparison on accuracy and execution time for certain algorithms [16] [19] [23] [29]. However, these papers compare individual algorithms but not different classes of algorithms.

Our work reported in this paper is the first on classification of GPS algorithms and comparison of their performance in terms of accuracy and execution time measurements, especially in real environments that involves measurement errors and redundant measuring.

III. Models Approaches for Simple Systems

A. *The Model and Simple Systems*

The challenge of a GPS system is to determine the position of a device (usually called a *receiver*) by measuring the distance between the receiver and a set of *m* satellites based on the trilateration method [22]. Let the location of satellite be (x_i, y_i, z_i), and the receiver's location be (x^e, y^e, z^e). The center of the earth is denoted as $(0,0,0)$, the origin point of the coordinate system. Note that the clock at the receiver may not be precisely synchronized with those in satellites. Then, a GPS system typically takes the following steps to determine (x^e, y^e, z^e): first, the receiver receives a signal from satellite S_i and uses it to compute the measured distance, $\hat{\rho}_i$, between itself and S_i; then, the following equations can be established:

$$\sqrt{(x_i - x^e)^2 + (y_i - y^e)^2 + (z_i - z^e)^2} = \hat{\rho}_i - \varepsilon^R \tag{3-1}$$

where i = 1, 2, …, m and ε^R is the distance error brought by imprecise clock at the receivers. The receiver's location (x^e, y^e, z^e) is determined by solving the system in (3-1).

In reality, many additional errors have to be considered in addition to imprecise clock at the receiver. To better appreciate the GPS algorithmic ideas, we will start by considering solution approaches that only deal with the clock error. We refer to these kinds of systems as *simple systems*. We will then discuss complex systems in Section 5.

B. *Approaches for Simple Systems*

We now consider the approaches that can be used to solve the system in (3-1). Note that (3-1) has four unknowns, namely (x^e, y^e, z^e) and ε^R and, hence, four equations are sufficient to solve (3-1). Thus, in this section, we will assume that four satellites are available, i.e., *m* = 4, unless otherwise specified.

Based on utilized solution strategies, GPS algorithms can be partitioned into two classes: *iterative* and *direct algorithms*. The main idea of an iterative algorithm is as follows: starting from an initial guess, the algorithm attempts to solve a simple system by finding successive approximations to the solution. On the contrary, direct algorithms attempt to solve the problem with a number of transformations. Each transformation converts the system of equations into another form (equivalent or approximated). For both classes of algorithms, sometimes it's necessary to make approximations. Therefore, the solution obtained will be approximate; otherwise, the solution will be exact. We call them *approximate* or *exact*, respectively. Therefore, we present a classification in Table 3.1.

Table 3.1 A Classification of GPS Algorithms

	Exact	Approximate
Iterative	[13][25][33], [21][31], [35]	N/A[1]
Direct	[3] [6] [32], [5], [7], [18], [20], [23], [24], [26], [29]	[14][16][9], [28]

1) *Exact Iterative Algorithms*

This class of algorithms will iteratively calculate until the solution is sufficiently close to the true position. A typical algorithm is *Newton-Raphson* (or NR in short) algorithm [21] [31]. We use this algorithm to illustrate the basic ideas behind these algorithms. For (3-1), let

$$r_i = \sqrt{(x_i - x^e)^2 + (y_i - y^e)^2 + (z_i - z^e)^2} + \varepsilon^R \tag{3-2}$$

and

$$f_i(\vec{X}) = r_i - \hat{\rho}_i = 0 \tag{3-3}$$

where i = 1, 2, 3 and 4. $\vec{X}$ is vector $(x^e \quad y^e \quad z^e \quad \varepsilon^R)^T$ which represents the solutions to be calculated.

Let $\vec{X}_j$ be the *j*th row of $\vec{X}$. As the algorithm is iterative, we let $\vec{X}^k$ be the solution after *k*th iteration and $\vec{X}_j^{\,k}$ be the *j*th row of $\vec{X}^k$, where *k* = 0, 1, 2, …, and *j* = 1, 2, 3 and 4. Comparing (3-1), (3-2), and (3-3), we see that (3-3) is equivalent to (3-1). Using the Taylor series [21], we can obtain an approximate system for (3-3):

$$f_i(\vec{X}^{k+1}) \approx f_i(\vec{X}^k) + \sum_{j=1}^{4} \frac{\partial f_i}{\partial \vec{X}_j}(\vec{X}^k)(\vec{X}^k - \vec{X}^{k+1}) \tag{3-4}$$

where $\sum_{j=1}^{4} \frac{\partial f_i}{\partial \vec{X}_j}(\vec{X}^k)$ is the partial derivative on $\vec{X}_j$ at the solution $\vec{X}^k$. Since

$$f_i(\vec{X}^{k+1}) = 0 \tag{3-5}$$

we have

$$f_i(\vec{X}^k) + \sum_{j=1}^{4} \frac{\partial f_i}{\partial \vec{X}_j}(\vec{X}^k)(\vec{X}^k - \vec{X}^{k+1}) \approx 0 \tag{3-6}$$

Knowing the value of $\vec{X}^k$, by (3-6), we can obtain a linear system with unknown $\vec{X}^{k+1}$. That is, all other parameters are known in (3-6) except $\vec{X}^{k+1}$. Because there are four equations, we can then derive a unique solution for $\vec{X}^{k+1}$. That is, we have

$$\vec{X}^{k+1} = \vec{X}^k + \frac{f_i(\vec{X}^k)}{\sum_{j=1}^{4} \frac{\partial f_i}{\partial \vec{X}_j}(\vec{X}^k)} \tag{3-7}$$

[1] By definition, this class of algorithms will make certain approximations in the iterative process. To the best of our knowledge, no work of this kind has been reported.

Following the fix point theory [17], we know that the iterative process in (3-7) will converge at the exact solution of (3-3). Other algorithms such as [13] [25] [33] and [35] employ strategies similar to the NR algorithm with certain variations.

2) Exact Direct Algorithms

This class of algorithms uses a number of transformations to convert the original system into an equivalent but calculate-able form. The solutions are, hence, usually in a closed-form. Here, we introduce the main idea behind these algorithms.

Algorithms described [3] [6] [32], [18], [24] and [26] transform the non-linear system in (3-1) to an equivalent polynomial equation with only one unknown. For example, in [3] [6] [32], so-called *Groebner Basis* techniques are used to achieve this purpose. First, (3-1) is converted into (3-8):

$$(x_i - x^e)^2 + (y_i - y^e)^2 + (z_i - z^e)^2 = (\hat{\rho}_i - \varepsilon^R)^2 \qquad (3\text{-}8)$$

Subtracting the first equation from the rest, we will have a linear system and then get a quadratic equation in terms of ε^R:

$$a_2(\varepsilon^R)^2 + a_1\varepsilon^R + a_0 = 0 \qquad (3\text{-}9)$$

where a_0, a_1 and a_2 are known constants. (3-9) may result in two possible solutions for ε^R. Substituting each of these solutions into the expressions of x^e, y^e and z^e, two solutions can be derived. Choosing a solution that meets practical constraints, we can say that (3-1) is solved.

Other algorithms are similar. The difference is that they use different techniques to derive expressions for x^e, y^e and z^e in terms of ε^R. Then, the solution is derived via a polynomial equation similar to (3-9).

3) Approximate Direct Algorithms

This class of algorithms also utilizes a number of transformations. However, some transformations may be approximations. Hence, the solution calculated by this class of algorithms will be an approximation of the true position. Approximations can be either *numerical* or *physical*.

Bancroft algorithm [9] is one that uses numerical approximations. The algorithm first transforms the system in (3-1) into (3-8) and then transform (3-8) into a following approximate quadric equation with one unknown:

$$\langle B\tau, B\tau\rangle\Lambda^2 + 2(\langle B\tau, B\tau\rangle - 1)\Lambda + \langle B\alpha, B\alpha\rangle = 0 \qquad (3\text{-}10)$$

where Λ is unknown; B, τ and α are known. For a detailed discussion on the rationale of the transformation, an interested reader is referred to [9]. Therefore, we obtain a quadratic equation in terms of Λ. Taking the approach similar to the one discussed in (3-9), we can solve (3-10). However, this algorithm uses the location of earth center as the approximate position of the receiver when constructing parameter B [9]. Thus, we eventually obtain an approximate solution of (3-1).

An algorithm in [28] took a physical approximation approach. It first obtains a linear system in terms of x^e, y^e and z^e as shown in (3-8). It then uses a clock-bias prediction method [14] [28] to obtain an approximate value of ε^R. Denoting $\hat{\rho}_i - \varepsilon^R$ as $\tilde{\rho}_i$, we get a following system:

$$\sqrt{(x_i - x^e)^2 + (y_i - y^e)^2 + (z_i - z^e)^2} = \tilde{\rho}_i \qquad (3\text{-}11)$$

In (3-11), three unknowns are left and ordinary linear algebraic techniques can be used to solve it. As this algorithm uses physical approximation to obtain a solution, we call it *Physically Approximate Direct* algorithm (PAD for short).

IV. Complex Systems and Their Performance

A. Approaches of Solving Complex Systems

In Section 3, we have discussed the different methods taken to solve simple systems. However, due to the limitation of measurement techniques, measurement errors are inevitable in the real world. In the literature, the basic idea to deal with this kind of error has been to treat satellite dependent errors as errors contained in known parameter $\bar{\rho}_i$ [22]. Mathematically, we need to solve

$$\sqrt{(x_i - x^e)^2 + (y_i - y^e)^2 + (z_i - z^e)^2} = \bar{\rho}_i - \varepsilon^R \qquad (4\text{-}1)$$

where $i = 1,\ 2,\ \ldots,\ m$ while considering that $\bar{\rho}_i$ may contains an error.

Although the systems in (4-1) take into account satellite dependent errors and the receiver dependent error, (4-1) is nevertheless similar with (3-1) except that $\bar{\rho}_i$ contains errors. Consequently, a solution obtained from (4-1) will contain an error. As we will see, this error may, unfortunately, be substantial. The key is how to eliminate or reduce the error in the final solution due the error in $\bar{\rho}_i$.

A general strategy to eliminate or reduce the error in the final solution is to take advantage of the (possibility of) hardware redundancy. In other words, we may like to use more than four satellites ($m > 4$), if they are available, to calculate a position. Note that this immediately causes the system in (4-1) to become *over-determined.* A common approach here is to use the *Least Square Method* (LSM) [22] to solve the over-determined system with the hope that the error in the solution can be reduced or eliminated in the process.

Considering the above issues such as errors contained in measurements and utilization of redundant satellites' signals, we call such practical configurations of GPS as *complex systems*. Later we will discuss how to apply different types of GPS algorithms to complex systems.

B. Application of Least Square Methods

It seems that the only thing we need to do is to superimpose LSM to the algorithms discussed in Section 3 for complex systems. Unfortunately, this is not a trivial task. Next, we discuss the feasibility and methodology of applying LSM to these algorithms.

1) Applying LSM to Exact Iterative Algorithms

Exact Iterative Algorithms, as discussed in Section 3.1.1, can utilize least square method to find an optimal solution. As we known, for the NR algorithm, it uses the linear system in (3-7) to iteratively find a solution. In fact, (3-7) can be written as a matrix form:

$$A_k \vec{X}^{k+1} = \vec{D} \tag{4-2}$$

From [21], we know the errors in $\vec{D}$ have a Gaussian distribution with zero mean. Therefore, using *Ordinary Least Squares* [22] to (4-2), we can obtain

$$\vec{X}^{k+1} = \left(A_k{}^T A_k\right)^{-1} A_k{}^T \vec{D} \tag{4-3}$$

2) Applying LSM to Exact Iterative Algorithms

From Section 3.1.2, we know that this approach is to find an equivalent quadratic univariate equation for (3-1). When there are more than four satellites available, this approach cannot utilize least squares techniques to find an optimal estimation. [13] provides an alternative approach (*Groebner Basis* method) to reduce the system scale (i.e., the number of equations) in order to make computable. Unfortunately, as pointed out by [12], such an approach results remarkable large time complexity, making such an approach infeasible for practical uses when we deal with the systems that have satellite dependent errors. Thus, we will not evaluate this type of algorithms for complex systems.

3) Applying LSM to Approximate Direct Algorithms

As we know, algorithm [9] and [28] can provide approximate solutions for (3-1) in a closed-form. Bancroft algorithm can use *Ordinary Least Squares* to obtain an optimal estimation of the true position. In fact, for the system in (3-10), *B* has a following relation with a known parameter *A* (its definition can be found in [9]):

$$AB = I \tag{4-4}$$

where *I* is the identity matrix. From [9], we know that *A* is a $m \times 4$ matrix and it is assumed as a constant matrix without errors. Then, *B* can be calculated by (4-5):

$$B = \left(A^T A\right)^{-1} A^T I \tag{4-5}$$

Similarly, PAD algorithm can also utilize LSM to find an optimal estimation. Next, we will explain how PAD algorithm works. Recalling the system in (3-11) and we write (3-11) as a matrix form in (4-6):

$$A\vec{X} = \vec{D} \tag{4-6}$$

From [28], in (4-6), we know that the errors in $\vec{D}$ are zero-mean, equal-variance and correlated. Then using *Generalized Least Squares* [22], we will have

$$\vec{X} = \left(A^T M^{-1} A\right)^{-1} A^T M^{-1} \vec{D} \tag{4-7}$$

where *M* is the covariance matrix of the errors in $\vec{D}$.

By the above methods, both algorithms will obtain an optimal estimation in the sense of least squares. However, this optimal estimation is meaningful to the approximate solution but not the true position. That is, the final solution is still an approximation of the true position.

C. Performance Evaluation and Comparison

As discussed in Section 4.2, only algorithms in Section 4.2.1 and 4.2.3 are applicable for complex systems. Therefore, in this subsection, we will choose *NR*, *Bancroft* and *PAD* algorithms for comparisons.

1) Performance Metrics

In order to analyze and compare the *accuracy* of GPS algorithms, we would like to introduce metric *Absolute Error (AE)*, which is defined as follows:

$$AE = \sqrt{\left(x^e - x^r\right)^2 + \left(y^e - y^r\right)^2 + \left(z^e - z^r\right)^2} \tag{4-8}$$

where $\left(x^r, y^r, z^r\right)$ is the exact location of the receiver, and $\left(x^e, y^e, z^e\right)$ is the calculated solution by an GPS algorithm. The *Absolute Error* indicates the distance between the exact location and the location derived by the given GPS algorithm. Obviously, the smaller of the AE, the better a GPS algorithm is.

We would like to measure the *efficiency* of an algorithm by its execution time. Let the time for executing algorithm O be τ_O where O may be NR, Bancroft, or PAD algorithm, respectively. We define *Normalized Execution Time (NET)* for algorithm O as follows:

$$NET = \frac{\tau_O}{\tau_{NR}} \times 100\% \tag{4-9}$$

That is, NET measures the execution time of an algorithm in terms of the time taken for executing NR algorithm. We do so as the NR algorithm has been literally a benchmark in studies of GPS algorithms.

2) The Experimental System

We established the following experimental system and collected performance data. First, we collect the real observation data from CORS by randomly selecting a land observation station [15]. The site ID is *SRZN* and the coordinates of this site is (3623420.032,-5214015.434, 602359.096) in meters. The collecting date is 2009/08/12. 24-hour observation data are collected. In each second, all available satellites' coordinates and measured distances are collected as one data item. Each item contains data from 8 to 12 satellites. By this means, real satellite geometry can be obtained. The computer system has an AMD Triple-Core Processor with 3GHz and 1.5MB of Cache, 2GB Physical Memory, and 500GB Disk. Matlab software is used to run all algorithms.

Because the true location of the observation station is known, we can remove all errors from the measured distances. Then we manually add ε_i^s (it satisfies Gaussian distribution and zero mean [20]) and ε^R (In order to guarantee that clock bias prediction is acceptably accurate, it is set as 1 meter. In fact, the accuracy of clock bias prediction in PAD algorithm is less than 1 meter) on the true distances. Such a setting can be seen as an ideal condition for all algorithms. Based on this configuration, performance data in terms of the averages of *Absolute Error* and *Normalized Execution Time* are obtained.

3) Performance of Accuracy

Figure 4.1 shows the change of the accuracy when the satellite dependent error ε_i^s changes. We use seven satellites and let ε_i^s change from 0 to 20 meters. We can see that when ε_i^s changes, the accuracy changes in all algorithms. The smaller ε_i^s is, the better the accuracies are. Note that, when ε_i^s is set to 0, accuracies of the NR and the PAD algorithms are close to zero. However, Absolute Error of Bancroft algorithm remains to be more than 30 meters. From Fig. 4.1, we find that the accuracy of Bancroft algorithm is the worst. the NR algorithm has the best accuracy. The accuracy of PAD algorithm is close to that of NR algorithm. Such a result can be explained as follows:

1) From Section 3, we know that the NR algorithm can iteratively find an exact solution for simple systems and an optimal estimation for complex systems.
2) The PAD algorithm can use LSM for complex systems. However, this algorithm uses an approximation to ε^R, which will bring extra errors to the calculated solution. Therefore, the accuracy of the PAD algorithm is close to but does not exactly match that of the NR algorithm. Bancroft algorithm can also use LSM for complex systems and will bring approximate solutions to the true position. Therefore, this algorithm can not achieve the similar accuracy of that of the NR algorithm either.

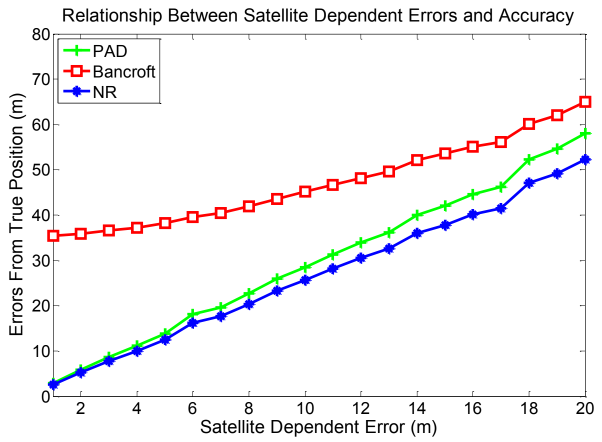

Figure 4.1 Change of the accuracy

Note that in later comparison of execution time of different algorithms, we can see that tradeoff to obtain better accuracies is spending more execution time.

4) Performance of Execution Time

In this subsection, we will evaluate and compare the execution time of three algorithms. Fig. 4.2 gives the results of comparisons. Here we use the same configuration in Section 4.3.3. From Fig. 4.2, we see that the NR algorithm takes longest time to find a solution. Bancroft algorithm is better and the PAD algorithm is the best. Such differences can be explained as follows:

1) The NR algorithm uses an iterative process to find a solution. Typically, four or five iterations are needed to converge. Though each iteration involves solving a linear system, it spends obviously much more time than direct algorithms.
2) Bancroft algorithm uses a direct approach to find a solution. As discussed in Section 3, this algorithm involves finding the root for a univariate quadric equation. In addition, in the real program of Bancroft algorithm, two rounds of executions are needed. That is, the program will use the solution obtained by the first execution for refining. The second execution will find a better solution and no more executions are needed for further refining. If only one execution is performed, its execution time will be less than that of the PAD algorithm. However, this will result in a remarkable loss of accuracy. PAD algorithm also calculates in a direct manner. Just one linear system needs to be solved. This algorithm involves performing only several matrix inversions and multiplies. Consequently it takes the least time to find a solution.

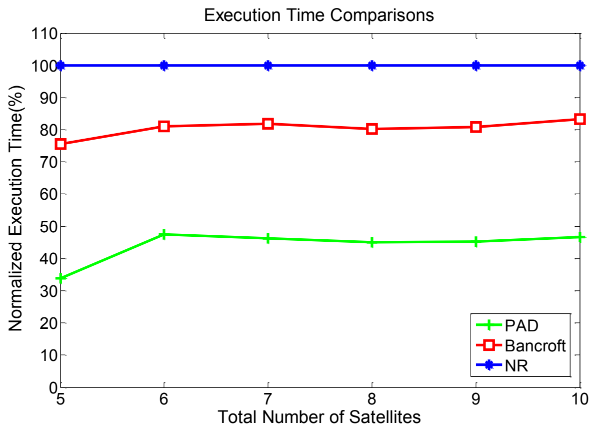

Figure 4.2 Normalized Execution Time

Summaries. Based on the comparisons on accuracies and execution times, we can identify the strengths and weakness for these algorithms:

1) The NR algorithm, which is a representative algorithm of Exact Iterative algorithms, has the best accuracy and uses much more execution time. This algorithm is applicable to applications requiring high accuracies without critical requirements on execution times.
2) For two representative algorithms of Approximate Direct algorithms, PAD algorithm has the feasible accuracy of positioning and least execution time and it fits for applications requiring high accuracies and real-time response; Bancroft algorithm has a limitation on the accuracy of positioning and takes more execution time than that of the PAD algorithm. When clock bias

prediction is not available for PAD algorithm, this algorithm can be an alternative solution.

V. Conclusions and Future Work

In this paper, we analyzed and evaluated GPS algorithms. By a classification of these algorithms, we revealed the mathematical nature of the solution strategies. We systemically evaluated and compared representative algorithms in terms of their performance of accuracy and execution time. Through our observations, justification for the Newton-Raphson algorithm as a de facto standard for GPS positioning can be established. In terms of both execution time and accuracy measurements, this algorithm does delivery reasonably good performance. However, our study also shows the potential of other (newly proposed) algorithms. For example, the PAD algorithm over-perform the NR algorithm in terms of execution time while has similar performance on accuracy. Several observations made in this study can be utilized in practice for selecting and improving the GPS algorithms for different applications situations.

Our work is preliminary and several extensions are possible. For example, we found that no comprehensive studies had been made for exact direct class of algorithms. These algorithms may have the potential for high accuracy and low execution time.

Acknowledgment

This work is supported in part by the National Basic Research (973) Program of China (Grant No.2011CB302502, 2011CB302803), the major national science and technology special projects(Grant No. 2010ZX03004-003-03), and Co-building Program of Beijing Municipal Education Commission.

References

[1] J. S. Abel and J. W. Chaffee, “Existence and uniqueness of GPS solutions,” IEEE Trans. on Aerospace & Electronic Systems, 27(6), 1991, pp. 952-956.

[2] J. L. Awange, Groebner bases, multipolynomial resultants and the Gauss–Jacobi combinatorial algorithms-adjustment of nonlinear GPS/LPS observations, Dissertation, TR 2002(1), Department of Geodesy and Geo-Informatics, University of Stuttgart, 2002.

[3] J. L. Awange and E. W. Grafarend, “Algebraic solution of GPS pseudo-ranging equations,” Journal of GPS Solutions, 4, 2002.

[4] J. L. Awange and E. W. Grafarend, “Explicit solution of the overdetermined three-dimensional resection problem,” Journal of Geodesy, 76, 2003, pp. 605-616.

[5] J. L. Awange and E. W. Grafarend, “Groebner basis solution of the three-dimensional resection problem (P4P),” Journal of Geodesy, 77, 2003, pp. 327-337.

[6] J. L. Awange and E. W. Grafarend, Solving algebraic computational problems in geodesy and geoinformatics, Springer, Berlin, 2005.

[7] J. L. Awange, E. W. Grafarend, Y. Fukuda and S. Takemoto, “Direct polynomial approach to nonlinear distance (ranging) problems,” Earth Planets Space, 55(5), 2003, pp. 231-241.

[8] J. L. Awange, E. W. Grafarend, B. Pal´ancz and P. Zaletnyik, Algebraic Geodesy and Geoinformatics (2nd Edition), Springer, 2010.

[9] S. Bancroft, “An algebraic solution of the GPS equations,” IEEE Trans. on Aerospace & Electronic Systems, 21(7), 1986, pp. 54-59.

[10] S. Bednarz, Adaptive Modeling of Receiver Clock for Integrity Monitoring During Precision Approaches, Master’s Thesis, Dept. of Aeronautics and Astronautics, MIT, 2004.

[11] I. Biton, M. Koifman and I. Y. Bar-Itzhack, “Improved Direct Solution of the Global Positioning System Equation,” Journal of Guidance, Control, and Dynamics, 1998, pp. 1313 - 1320.

[12] B. Buchberger, “Gr¨obner bases: A short introduction for systems theorists,” Proceedings of EUROCAST 2001, LNCS 2178, 2001.

[13] J. Chaffee and J. Abel, “On the Exact Solutions of Pseudo-range Equations,” IEEE Trans. on Aerospace and Electronic Systems, 30(4), 1994, pp. 1021-1030.

[14] E. Choi and D.A. Cicci, “Analysis of GPS Static Positioning Problems,” Applied Mathematics and Computation, 140, 2003, pp. 37-51.

[15] CORS, Continuously Operating Reference Stations, National Geodetic Survey, http://www.ngs.noaa.gov/CORS/cors-data.html.

[16] D. J. Dailey and B.M. Bell, “A method for GPS positioning,” IEEE Trans. on Aerospace & Electronic Systems, 32(3), 1996, pp. 1148-1154.

[17] J. Dugundji and A. Granas, Fixed Point Theory, Polish Scientic Publishers, Warsaw, 1982.

[18] E. W. Grafarend and J. Shan, “Closed-form solution of the nonlinear pseudo-ranging equations (GPS),” Artificial Satellites, Planetary Geodesy No. 28 (Special Issue on the 30th Anniversary of the Department of Planetary Geodesy, Warsaw), 31(3), 1996, pp. 133-147.

[19] E. W. Grafarend and J. Shan, “GPS solutions: closed forms, critical and special configurations of P4P”, GPS Solutions, 5(3), 2002, pp. 29-41.

[20] M. S. Grewal, L. R. Weill and A. P. Andrews, Global Positioning Systems, Inertial Navigation, and Integration, John Wiley & Sons, 2001.

[21] E. D. Kaplan (ed.), Understanding GPS: principles and applications, Norwood, Artech House, 1996.

[22] T. Kariya and H. Kurata, Generalized Least Squares, Chichester, UK: John Wiley & Sons, Ltd., 2004.

[23] A. Kleusberg, “Analytical GPS navigation solution,” Geodesy—the challenge of the 3rd millennium, Springer, Berlin, 2003, pp. 93-96.

[24] L. O. Krause, “A Direct Solution to GPS Type Navigation Equations,” IEEE Trans. on Aerospace and Electronic Systems, 23(2), 1987, pp. 223-232.

[25] I. S. Kotsireas, “Homotopies and polynomial systems solving I: Basic Principles,” ACM SIGSAM Bulletin, 35(1), 2001, pp. 19-32.

[26] J. Hoshen, “The GPS Equations and the Problem of Apollonius,” IEEE Trans. on Aerospace and Electronic Systems, 32(3), 1996, pp. 1114-1124.

[27] R. P. Langley, “The Mathematics of GPS,” GPS World, 1991, pp. 44-50.

[28] W. Li, L. Deng, S. Yang, Z. Xu and W. Zhao, Design and Analysis of a New GPS Algorithm, The 30th International Conference on Distributed Computing Systems, 2010, pp. 40-51.

[29] J. B. Lundberg, “Alternative algorithms for the GPS static positioning solution,” Applied Mathematics and Computation, 119(1), 2001, pp. 21-34.

[30] P. Misra and P. Enge, Global Positioning System: Signals, Measurements, and Performance, Ganga-Jamuna Press, Lincoln, Mass., 2001.

[31] P. S. Noe, K. A. Myers and T. K. Wu, “A navigation algorithm for low cost GPS receiver,” Journal of the Institute of Navigation, 25 (2), 1978, pp. 258-264.

[32] B. Paláncz, P. Zaletnyik and J. L. Awange and E. W. Grafarend, “Dixon resultants solution of systems of geodetic polynomial equations,” J. Geod., 82, 2008, pp. 505-511.

[33] B. Paláncz, P. Zaletnyik and L. Kovács, “Homotopy Solution of GPS - N Point Navigation Problem,” 9th International Symposium of Hungarian Researchers on Computational Intelligence and Informatics, 2008, pp. 311-320.

[34] B. Parkinson (ed.) and J. Spilker (ed.), Global positioning system: theory and applications Volume I, Cambridge, Charles Stark Draper Laboratory, Inc., 1996.

[35] M. A. Sturza, “GPS navigation using three satellites and a precise clock,” Navigation: Journal of the Institute of Navigation, 30(2), 1983, pp. 144-156.