Real-Time Software Receiver for GPS Controlled Reception Pattern Antenna Array Processing

Yu-Hsuan Chen, Jyh-Ching Juang, *National Cheng Kung University, Taiwan*
David S. De Lorenzo, Jiwon Seo, Sherman Lo, Per Enge, *Stanford University, U.S.A.*
Dennis M. Akos, *University of Colorado Boulder, U.S.A.*

BIOGRAPHY

Yu-Hsuan Chen is a Ph. D. candidate in electrical engineering of National Cheng Kung University, Taiwan. He visited GPS Laboratory at Stanford University in 2010. He received his M.S. degree in electrical engineering from National Cheng Kung University, Taiwan in 2002.

David S. De Lorenzo is a research associate at the Stanford University Global Positioning System (GPS) Laboratory. He received his M.S. in Mechanical Engineering from University of California, Davis and his Ph.D. in Aeronautics and Astronautics from Stanford University.

Jyh-Ching Juang received the Ph. D. degree in electrical engineering from University of Southern California, Los Angeles, in 1987. He had been with Lockheed Aeronautical Systems Company before he went back to Taiwan in 1993. Currently, he is a Professor at the Department of Electrical Engineering, National Cheng Kung University, Tainan, Taiwan. His research interests include DSP-based control applications, GNSS navigation design, sensor networks, and advanced signal processing.

Jiwon Seo is a Postdoctoral Scholar at Global Positioning System Laboratory and Space Environment and Satellite Systems Laboratory at Stanford University. He received his B.S. degree in mechanical engineering (division of aerospace engineering) from KAIST (Korea Advanced Institute of Science and Technology) and received M.S. degrees in aeronautics/astronautics and electrical engineering from Stanford and a Ph.D. degree in aeronautics/astronautics from Stanford in 2010. His research interests include ionospheric effects on GPS aviation; alternative positioning, navigation, and timing; and atmospheric remote sensing. Dr. Seo was a recipient of the Samsung Lee Kun Hee Graduate Fellowship for five years.

Sherman Lo is a senior research engineer at the Stanford University Global Positioning System (GPS) Laboratory. He is the Associate Investigator for the Stanford University efforts on the Federal Aviation Administration (FAA) Alternate Position Navigation & Time (APNT) study. .

Per Enge is a Professor of Aeronautics and Astronautics at Stanford University, where he is the Kleiner-Perkins, Mayfield, Sequoia Capital Professor in the School of Engineering. He directs the Stanford University GPS Research Laboratory.

Dennis M. Akos is an Associate Professor with the Aerospace Engineering Science Department at the University of Colorado at Boulder. Dr. Akos completed the Ph.D. degree in Electrical Engineering at Ohio University within the Avionics Engineering Center.

ABSTRACT

This paper demonstrates a real-time software receiver supporting GPS L1 C/A controlled reception pattern antenna (CRPA) processing. The software receiver is implemented on a widely-available recent generation multi-core processor and is capable of processing array signals from either radio frequency (RF) front-end modules or collected datasets. Most importantly, the receiver allocates dedicated satellite-tracking antenna patterns for each baseband receiver channel.

The architecture of such a software receiver needs to be carefully developed so that it can process signals from each individual antenna, calculate the appropriate beam formed composite signals (one for each satellite/receiver channel) and then process those signals. Multitasking and synchronization mechanisms were developed to support the tracking of multiple channels in real time. To achieve real-time capability, parallel operations are necessary to reduce computation complexity. Bit-wise operations are exploited and implemented in the correlator. Additionally, Single Instruction Multiple Data (SIMD) instructions are used to efficiently calculate the covariance matrix for the beam steering algorithm. The architecture supports at least eighty tracking channels (ten channels from each of seven antennas plus ten composite beam formed signals) in real time.

The CRPA software receiver was architected to operate without extensive set up and pre-calibration enhancing its suitability for commercial users. The algorithm design was architected to aid ease of set up. While conventional antenna array system receivers are used with the geometry of antennas and cable lengths known in advance, the algorithm implemented allows for operation without such a priori knowledge.

Two beam steering techniques were tested. First is deterministic beam steering. An adaptive algorithm, Minimum Variance Distortion Response (MVDR) algorithm, was implemented to adaptively maximize signal power. The architecture determines of the carrier phase difference between signals from different antennas for a single satellite in order to build the steering vector. An experiment was conducted to show the enhanced C/No and controlled reception patterns through directing the CRPA toward the direction of satellite of interest. For evaluating interference rejection, a LI GPS simulator is used to build an environment with CDMA and CW interferences. The result shows that the MVDR algorithm has reliable performance than non beam steered and deterministic beam steering under the both type of interferences.

INTRODUCTION

GPS provides 24 hour all-weather position, navigation, and timing (PNT) services worldwide. However, GPS and GNSS signals are relatively weak and thus vulnerable to deliberate and unintentional interference. An electronically-steered antenna array system provides an effective approach to mitigating interference by controlling the reception pattern and steering beams/nulls. As a result, so called controlled reception pattern antenna (CRPA) array have been deployed by organizations such as the US military which seek high levels of anti-jam performance. However, there is a tradeoff in increased cost and computational complexity which to-date has not been acceptable to commercial GNSS users. The research conducted in this paper brings the directive gains and interference rejection benefits of electronically-steered antennas closer to commercial users by implementing CRPA processing on software receivers.

A software receiver was developed for study a civil use of GNSS beam steering. It uses a 7-element antenna array with one RF front-end per antenna to collect the desired signals. The digitalized signals are then transferred to PC by a USB microcontroller board. The signal processing, positioning, and beamforming are accomplished by software [3][4].

Conventionally, antenna array system receivers perform CRPA with the geometry of the antennas and cable lengths known in advance. In the developed software receiver, the algorithm implemented allows for operation without such a priori knowledge. As the carrier phase difference is related to geometry of antenna as well as line biases of the cables, this can be used as weights for deterministic beamforming or constraints for adaptive beamforming.

The software receiver contains ten channels per antenna for assessing the carrier phase. . So for seven antennas without beamforming we have seventy independent channels. This results in high computational complexity. In order to achieve real-time capability, single instruction multiple data (SIMD) instructions [5] and assembly programming is used to accelerate the operation. Additionally, multi-threading programming is adopted to fully exploit the multi-core resources of the processor.

An experiment was conducted to demonstrate that the carrier to noise ratio (C/No) is enhanced by array processing. With an injected interference, the developed software receiver is also tested with low power CW and CDMA interferences. Comparisons are made between a single antenna, CRPA by deterministic beamforming and MVDR adaptive beamforming.

The paper is organized as follows. First, the algorithms of the CRPA are introduced. Then, the implementation of the CRPA in our developed software receiver is described. The architecture of the software receiver , including hardware and software components, is explained in detail. For verifying the real-time capability, the software analyses, including thread activities and timing, are shown. A calibration procedure by carrier phase precise positioning is implemented for calculating the controlled pattern results. The experiments for enhancing C/No and interference rejection are described and results show the performance of the CRPA software receiver. Finally, some concluding remarks are made.

ALGORITHMS OF THE CONTROLLED RECEPTION PATTERN ANTENNA

The primary goal of a controller reception pattern antenna is to enhance the carrier-to-noise ratio of a selected satellite and reject interference [1]. The basic algorithm is deterministic beamforming which combines the signal of antennas, multiplied by complex weights and then summed over all antennas as shown in equation (1).

$$s(t) = \sum_{i=1}^{M} s_i(t) \cdot w_i = W^T S \tag{1}$$

where $s_i(t)$ is the signal from ith antenna and w_i is the weight associated to the ith antenna. Traditionally, the signal of one of the antennas is set as a reference and its weight is set to 1. The other weights are set as phase shifts relative to the reference antenna represented in equation (2).

$$w_i = \exp\left(-j\left(\Delta\varphi_i^l + \delta L_i\right)\right) \tag{2}$$

where $\Delta\varphi_i^l$ is the phase difference based on antenna geometry and the direction of desired satellite. δL_i is the phase difference resulting from the cable length difference and phase center inaccuracy. $\Delta\varphi_i^l$ can be calculated as

$$\Delta\varphi_i^l = \frac{2\pi\vec{p}_i \cdot \hat{r}^l(\phi,\theta)}{\lambda} \tag{3}$$

where $\vec{p}_i$ is the baseline vector of the ith antenna and $\hat{r}^l(\phi,\theta)$ is the unit vector to satellite l as shown in the figure 1.

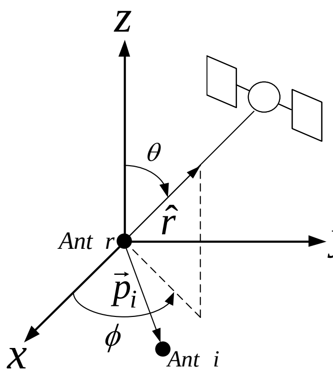


Figure 1. The antenna geometry and direction of satellite showing calculation of $\Delta\varphi_i^l$

Because $\vec{p}_i$ is a known prior and $\hat{r}_l(\phi,\theta)$ can be calculated after positioning, $\Delta\varphi_i^l$ can be obtained. However, $\Delta\gamma_i$ is needed to be re-calibrated whenever any part of antenna array hardware is changed.

Compared to deterministic beamforming, the adaptive scheme uses feedback to find the optimal weights for the beam. The MVDR algorithm [2] uses weights of the deterministic beamforming as the steering vector to constraint the gain of the desired direction to unity while steering the nulls to reject any interference. The null steering is accomplished by calculating the appropriate weights iteratively. The implementable updating equations of MVDR is written as

$$\overline{W_{n+1}} = \gamma\left[I - \mu R\right]W_n + T \tag{4}$$

where R is the covariance matrix and T is the steering vector. Then, in order to keep the magnitude of weight of the reference signal as unity, the weight vector is normalized as shown in equation (5).

$$W_{n+1} = \overline{W_{n+1}} \Big/ \left|\overline{w_{r,n+1}}\right| \tag{5}$$

IMPLEMENTATION OF A CRPA IN A SOFTWARE RECEIVER

For deterministic beamforming and the adaptive MVDR scheme, obtaining the steering vector is the key to implementing a CRPA. However, as mentioned in the previous section, parameters of the steering vector are obtained through calibration. A software receiver has the flexibility of implementing channels to process multi-antenna signals. A set of channels with the same PRN assignment are configured to process the signal of an individual antenna, respectively. The measurement of the phase locked loop is integrated carrier phase (ICP) which integrates carrier phase throughout tracking. The ICP is often used to smooth code pseudorange for improving accuracy of positioning. In our software receiver, ICP differences between different antennas are taken to build the steering vector. Without any calibration prior to execution, the software receiver uses ICP differences to perform CRPA instead of the azimuth/elevation to the satellites and baseline vectors of antennas. Figure 2 shows the block diagram of the implementation of the CRPA in our software receiver.

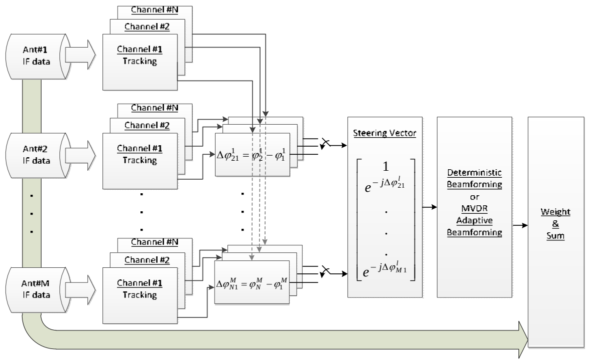

Channel #N
Channel #2
Ant#1 IF data
Channel #1 Tracking
Ant#2 IF data
$\Delta\varphi_{21}^1 = \varphi_2^1 - \varphi_1^1$
Ant#M IF data
$\Delta\varphi_{N1}^M = \varphi_N^M - \varphi_1^M$
Steering Vector
$\begin{bmatrix} 1 \\ e^{-j\Delta\varphi_{21}^l} \\ \vdots \\ e^{-j\Delta\varphi_{M1}^l} \end{bmatrix}$
Deterministic Beamforming or MVDR Adaptive Beamforming
Weight & Sum

Figure 2. Block diagram of the implementation of CRPA in a software receiver

Once all of the channels assigned to a given satellite are tracking, the ICPs are needed to re-initialize according to in-phase and quadrature-phase of individual correlator output. It takes several milliseconds to average the phase. Then, the ICPs are continuously integrated and the phase difference between the reference antenna and others are computed at runtime. Any of the tracked satellite can be selected by the operator and then the ICP difference is utilized to calculate the steering vector. For deterministic beamforming, the weight vector is equal to steering vector. For the MVDR algorithm, it is obtained through an adaptive procedure as expressed in equations (4) and (5) every millisecond. The beam formed signal is calculated by multiplied weights with signals and then sum over all antennas.

ARCHITECTURE OF SOFTWARE RECEIVER

The developed CRPA real-time software receiver runs on a PC platform and uses USB for data input. It currently operates on GPS L1 C/A signal. There are a total of 80 channels of which ten are allocated to process real data for each antenna as well as ten tracking channels to process complex data for the beam formed composite signal. ICP measurements from each channel assigned to track the same satellite are collected to build the steering vector. Only one satellite is selected for beam steering (signal beam). The weight updating rate of MVDR algorithm is 1 KHz. Moreover, positioning is dedicated to a beam formed composite signal. There is a GUI to show ICP differences, weights and C/No of all channels to illustrate the beamforming performance as well as the positioning result.

A. HARDWARE

The 7-element antenna elements are arranged in circle with one wavelength between the each on a circular plane of aluminum. In the RF front-end, the L1 signal is down-convertered to a 4.1304 MHz intermediate frequency and sampled at 16.3676 MHz. The front-end outputs 2-bit real IF data. The clocks of the front-ends must be perfectly synchronized for array signal processing. A function generator set as sinusoidal wave output at 16.3676 MHz. It is divided to two branches by a 1-to-2 splitter. One is used the drive the clock input of USB microcontroller. The other one is further divided by a 1-to-8 splitter to eight branches which are used as reference clock for each front-end. The USB microcontroller serves as a bridge from RF front-end to PC. Its 16-bit parallel digital interface I/O is connected to seven 2-bit RF front-ends outputs. The resulting data rate is 32 MByte/s close to the limit of USB 2.0 data rate of 40 MByte/s. It is necessary to have an efficient strategy to reach such a high data rate. The hardware including antenna array, RF front-end and USB microcontroller board was developed in [6]. Since the developed software receiver implements as many as 80 channels, it required a multi-core processor with multi-thread support. Further, the processor needs to support single instruction multiple data (SIMD) instructions to account for the high computational complexity. The implementation of SIMD instructions for Intel is called as MMX/SSE/SSE2 [7]. The 128-bit register of SSE can be divided to several items in terms of bytes, words, or double words and perform parallel mathematical and comparative operations. The used processor is the quad-core Intel Core i7 which runs eight threads at a time and supports SIMD instructions sets including MMX and SSE (1, 2, 3, 4, 4.1, 4.2). The hardware set-up of the CRPA software receiver is depicted in the figure 3.

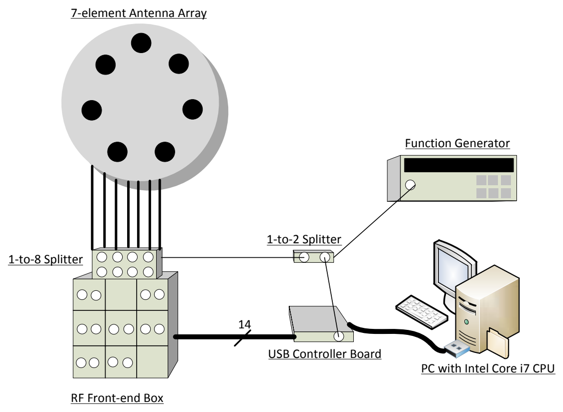


Figure 3. Diagram of hardware set-up of CRPA software receiver

B. SOFTWARE

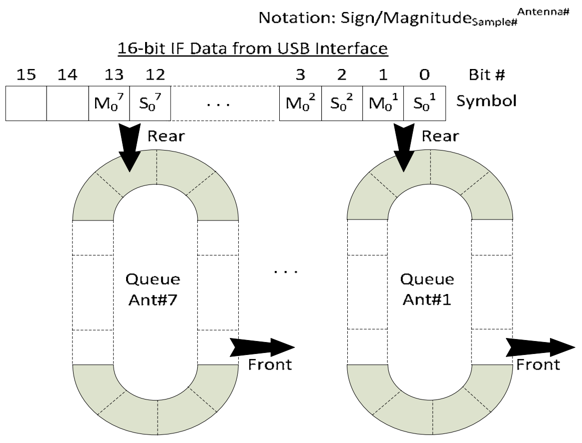
The software is developed with Visual Studio under 32-bit version of Microsoft Windows XP. Most of source code is programmed using C++. The functions with high computational complexity are programmed by inline assembly such as correlation operation and covariance matrix calculation. The components used are listed in the Table 1.

Table 1. Components list of software architecture

Component	**Description**
USB driver USB C++ library	Provided by chip manufacturer.
Software correlator	Hand-coded inline assembly using SSE instruction set based on bit-wise parallel algorithm [8].
Tracking Positioning	Use GPL-GPS [9] open source codeand modify interface to software correlator [10].
MVDR adaptive beamforming -Covariance Calculation	Hand-coded inline assembly using SSE instruction set for calculating covariance.
-Weight Update	Hand-coded C++ code for weight update
Weight-and-Sum and quantization of composite signal	Hand-coded inline assembly using SSE instruction set.
System Program	Arrange threads to achieve real-time capability

The IF data transfer uses the C++ library provided by chip manufacturer for communicating with USB driver. The procedure of the IF data transfer is depicted in the figure 4.

The width of data transfer for one sample is 16-bits for the entire IF data stream from all antennas. The data is further separated into a circular queue of individual antennas. The size of the entry of the queue is C/A code period (one msec). New IF data is stored into the rear index of the queue and the processing of the data is started from front index of queue.



Notation: Sign/Magnitude$_{\text{Sample\#}}^{\text{Antenna\#}}$
16-bit IF Data from USB Interface
15 14 13 12 3 2 1 0 Bit #
M_0^7 S_0^7 . . . M_0^2 S_0^2 M_0^1 S_0^1 Symbol
Rear
Rear
Queue Ant#7
Queue Ant#1
Front
Front

Figure 4. Procedure of IF data transfer from the USB interface to the circular queues

To fully exploit the resources of Intel Core i7 CPU, multiple threads are created and arranged in a way such that at most 8 threads need be executed. Figure 5 shows the flowchart of the threads.

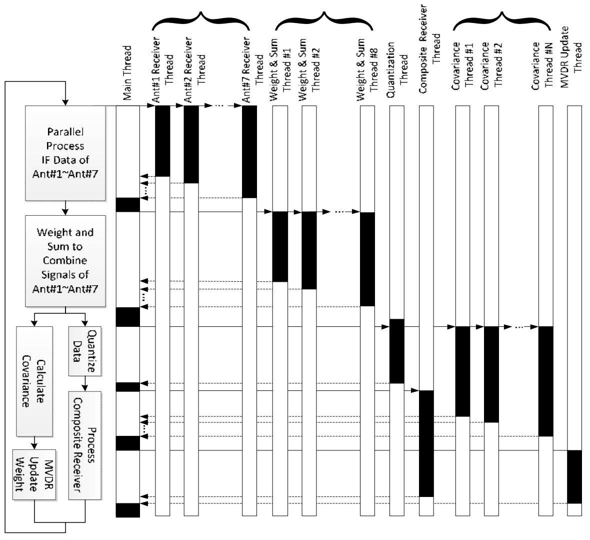
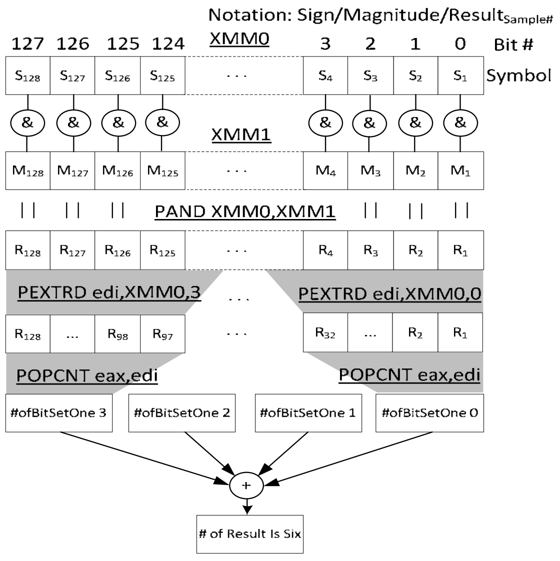


Figure 5. Flowchart of threads every msec

The main thread controls all the working threads. At first, seven receiver threads process the IF data obtained from the individual queue of antennas. Software correlation and acquisition/tracking are performed within each receiver thread. Next, the weight-and-sum operations for beam forming are separated into 8 threads and run simultaneously. The combined data is further quantized to complex 2-bit outputs, and then processed within a composite antenna receiver thread. In the composite receiver thread, not only is software correlation, acquisition and tracking performed, but the position solution is also calculated. In parallel, the covariance matrix is calculated by averaging over one msec within N threads where N depends on the number of elements in the covariance matrix. In addition, the MVDR weight update is performed in another thread. The whole procedure must be finished within one msec to achieve real-time capability.

CODING EXAMPLES TO ENABLE REAL-TIME

The bit-wise parallel algorithm [8] represents an incoming signal as a bit of a variable or register, and then performs a parallel logic operation instead of multiplication. In the software receiver, the algorithm is implemented using the SIMD instructions. 128 samples can be processed at a time using the 128-bit XMM registers. After the bit-wise operation, the results are stored within the bits of the register and an accumulation (counting the number of bit set to one) is required. Conventionally, this operation is performed by addressing a look-up table in the memory. The memory size for 16-bit table is 64KB [8]. However, with a CPU with SSE 4.2 can use POPCNT instruction [11] which can count the number of bits set to one for a 32-bit register in a 32-bit operating system. An example of counting the number of correlation results being six using SSE instructions shows in the figure 6.



Notation: Sign/Magnitude/Result$_{\text{Sample\#}}$
127 126 125 124 XMM0 3 2 1 0 Bit #
S_{128} S_{127} S_{126} S_{125} . . . S_4 S_3 S_2 S_1 Symbol
&
XMM1
M_{128} M_{127} M_{126} M_{125} . . . M_4 M_3 M_2 M_1
PAND XMM0,XMM1
R_{128} R_{127} R_{126} R_{125} . . . R_4 R_3 R_2 R_1
PEXTRD edi,XMM0,3
PEXTRD edi,XMM0,0
R_{128} … R_{98} R_{97} . . . R_{32} … R_2 R_1
POPCNT eax,edi
POPCNT eax,edi
#ofBitSetOne 3
#ofBitSetOne 2
#ofBitSetOne 1
#ofBitSetOne 0
+
# of Result Is Six

Figure 6. Example of counting the number of correlation results being using SSE instructions

As mentioned in the previous section, the weight-and-sum operation is to combine the IF data by multiplying the complex weights and summing over all antennas. This operation can be performed in parallel using SSE instructions. Figure 7 shows implementation of the weight-and-sum operation using SSE instruction. At first, the real inputs and complex weights are loaded into the XMM registers in terms of bytes. Then, the multiply packed signed integers and store low (PMULLW) instruction is performed to multiply real inputs with complex weights in parallel. Finally, a parallel addition is performed three times to obtain the summations of real and imagery components.

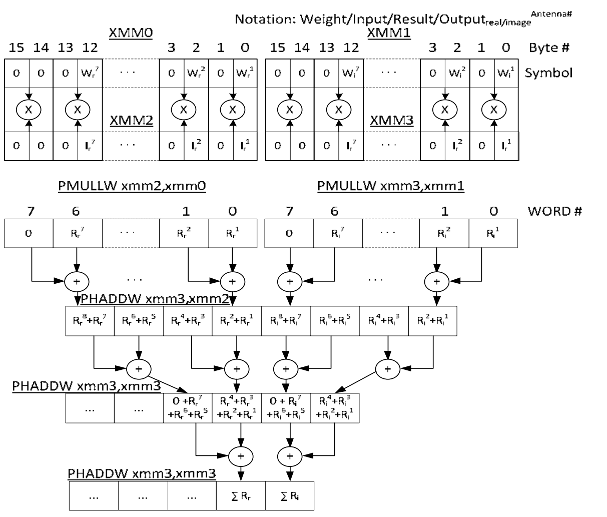


Figure 7. Example of the weight-and-sum operation using SSE instructions

CALIBRATION OF THE ANTENNA ARRAY BY CARRIER PHASE PRECISE POSITIONING

Although the software receiver can act as a CRPA in real-time without any prior knowledge, the calibration of the antenna array in the post-processing mode will aid in examining the CRPA performance of receiver. The gain pattern of the composite antenna is critical to show the performance of CRPA. The gain is calculated using relative positions of antennas and weights as shown in equation (6).

$$GP(\phi,\theta)=\sum_{i=1}^{M} w_i \times \exp\left[j\left(\frac{2\pi \vec{p}_i \cdot \hat{r}_l(\phi,\theta)}{\lambda} + \Delta\gamma_i \right)\right] \qquad (6)$$

where baseline vector $\vec{p}_i$ and cable length differences $\Delta\gamma_i$ are obtained through calibration. Figure 8 shows the used calibration procedure of the antenna array. In the CRPA software receiver, the ICP measurements are collected from the channels of the individual antennas for one minute. The azimuth/elevation of satellites are also obtained using the beam formed composite signal. The single difference ICP between signals of antennas and reference antenna j is represented as [12]:

$$\varphi_{ij}^{k} = \lambda^{-1} r_{ij}^{k} + \delta L_{ij} + N_{ij}^{k} + \varepsilon_{ij}^{k} \qquad (7)$$

where r_{ij} is differential range toward the k satellite between ith and jth antenna, N_{ij}^{k} is the integer associated to φ_{ij}^{k}, ε_{ij}^{k} is the phase error. The double difference ICP between satellites and reference satellite l is represented as:

$$\varphi_{ij}^{kl} = \lambda^{-1} r_{ij}^{kl} + N_{ij}^{kl} + \varepsilon_{ij}^{kl} \qquad (8)$$

The cable length difference term is subtracted in the double difference. Based on the distance of the antenna position close to one wavelength, equation (8) can be written as:

$$\varphi_{ij}^{kl} = \lambda^{-1}\left(-\hat{r}^{k} - \left(-\hat{r}^{l}\right)\right) p_{ij} + N_{ij}^{kl} + \varepsilon_{ij}^{kl} \qquad (9)$$

where $\hat{r}^{k}$ is the unit vector to satellite k, p_{ij} is baseline vector between ith and jth antenna. By combining all the double difference measurements of the pair ijth antennas, the observations equation is represented as:

$$\begin{bmatrix} \varphi_{ij}^{21} \\ \varphi_{ij}^{31} \\ \vdots \\ \varphi_{ij}^{K1} \end{bmatrix} = \lambda^{-1} \begin{bmatrix} -\hat{r}^{2} - \left(-\hat{r}^{1}\right) \\ -\hat{r}^{3} - \left(-\hat{r}^{1}\right) \\ \vdots \\ -\hat{r}^{K} - \left(-\hat{r}^{1}\right) \end{bmatrix} p_{ij} + I_{K-1} \cdot \begin{bmatrix} N_{ij}^{21} \\ N_{ij}^{31} \\ \vdots \\ N_{ij}^{K1} \end{bmatrix} + \begin{bmatrix} \varepsilon_{ij}^{21} \\ \varepsilon_{ij}^{31} \\ \vdots \\ \varepsilon_{ij}^{K1} \end{bmatrix} \qquad (10)$$

$$\Gamma = \lambda^{-1} G p_{ij} + N + \mathrm{E}$$

From the positioning results of composite channels, the azimuth and elevation of satellites are used to manipulate matrix G. Before solving the relative antenna positions, the integer vector N needs to be resolved. The LAMBDA method, specified in reference [13], is used. By substituting the solutions of N and p_{ij} into the equation (7), the cable length differences are calculated after filtering the noise term.

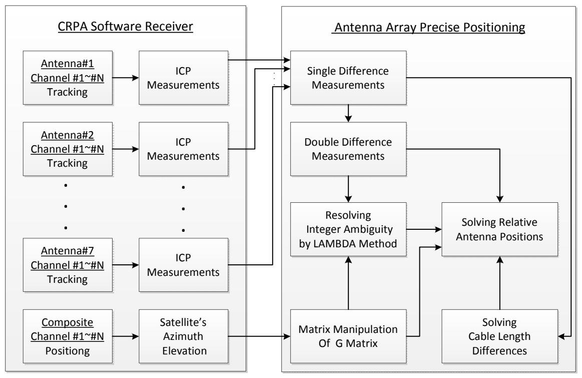

CRPA Software Receiver
Antenna Array Precise Positioning
Antenna#1 Channel #1~#N Tracking
ICP Measurements
Antenna#2 Channel #1~#N Tracking
ICP Measurements
Antenna#7 Channel #1~#N Tracking
ICP Measurements
Composite Channel #1~#N Positioning
Satellite's Azimuth Elevation
Single Difference Measurements
Double Difference Measurements
Resolving Integer Ambiguity by LAMBDA Method
Solving Relative Antenna Positions
Matrix Manipulation Of G Matrix
Solving Cable Length Differences

Figure 8. Block diagram of the calibration procedure of antenna array

ANALYSIS OF THE THREAD ACTIVITIES AND TIMING PERFORMANCE

In order to analyze the activities of the threads of the CRPA real-time software receiver, the Intel Thread Profiler collector program is utilized. This is a thread analyzer application to understand the threading patterns in multi-threaded software. This collector program is executed with our software receiver and outputs a timeline diagram containing execution flow of all the threads as shown in the figure 9. The resulting execution flow exactly follows the designed flow shown in figure 5.

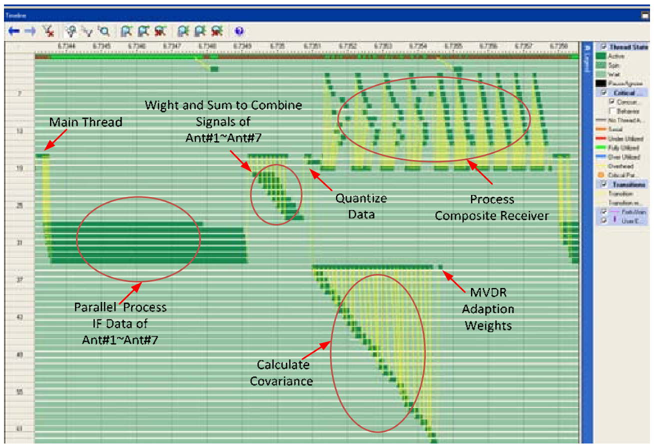

Main Thread
Wight and Sum to Combine Signals of Ant#1~Ant#7
Quantize Data
Process Composite Receiver
Parallel Process IF Data of Ant#1~Ant#7
Calculate Covariance
MVDR Adaption Weights

Figure 9. Execution flow of the threads of the CRPA real-time software receiver collected from the Intel Thread Profiler collector program

The execution time of the threads is measured by counting the clock cycles of CPU. The threads needed to execute in one msec are divided into three parts and execution times is measured 1000 trials. The results are represented as a box plot in the figure 10. The mean execution times of each part are listed in the table 2. The total mean execution time is less than one msec and shows that the CRPA software receiver can achieve real-time capability.

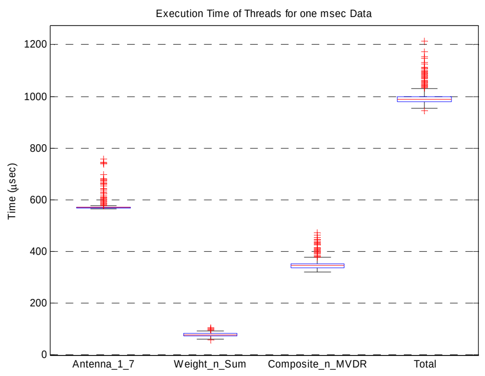


Figure 10. Box plot of the execution time illustrating the real time capability of the CRPA

Table 2. Execution time of threads

Thread Name	Mean Execution Time (μsec)
Process IF Data of Antenna #1 ~ #7	572.93
Weight and Sum	75.23
Composite Receiver and MVDR weight update	345.81
Total	993.98

EXPERIMENT FOR ENHANCING THE C/NO

An experiment is conducted in open field (low multipath environment) to examine the performance of CRPA. Figure 11 shows the hardware set-up of the software receiver.

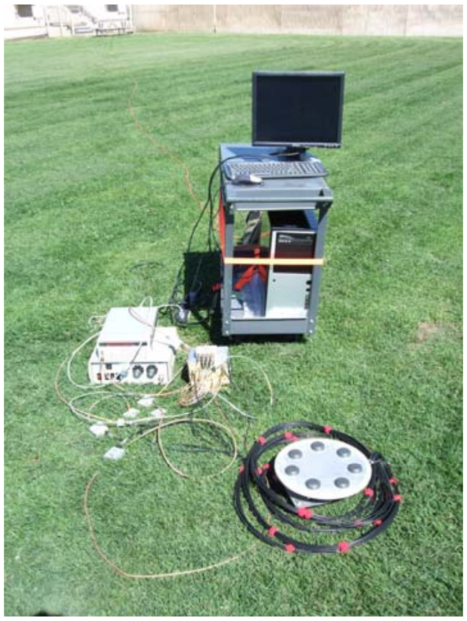


Figure 11. Hardware set-up of the software receiver

To determine the beam formed antenna gain pattern for each satellite in view, the software receiver runs in post-processing mode. In each run, the receiver performs CRPA by MVDR adaptive beamforming toward one of satellites. Figure 12 shows the filtered C/No of all satellites in view. The CRPA begins to perform from 30th second. There is more than 6dB gain through the CRPA for all satellites tested.

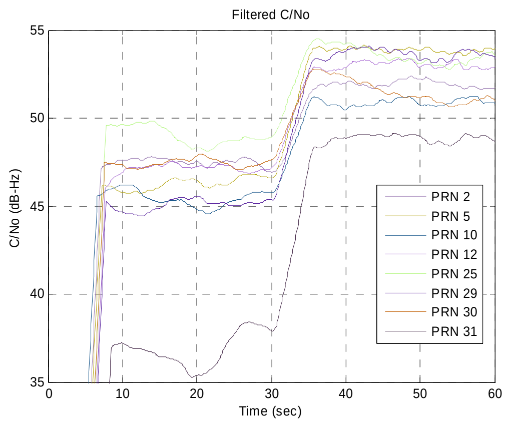


Figure 12. The filtered C/No of the software receiver with and w/o CRPA beam steering

The ICP measurements and azimuth/elevation of satellites are recorded for 60 seconds. Then, the calibration of antenna array using carrier phase precise positioning is performed and the result is shown in the figure 13. It is close to the physical antenna array geometry. Moreover, with relative antenna positions and weights from CRPA, the gain patterns of composite antenna after CRPA toward one satellite are calculated by equation (6). Figure 14 shows the resulting gain patterns toward the selected satellite. The corresponding sky plot is shown in the center of the figure. . There is a high gain in the direction of satellite which is the basis as to why the C/No is enhanced by the CRPA processing.

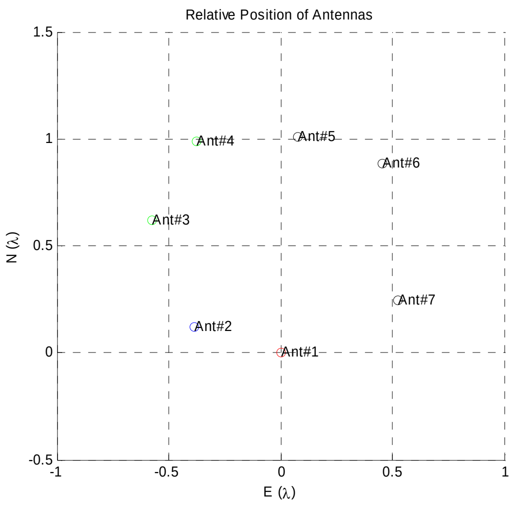


Figure 13. Relative position of the antennas from the carrier phase precise positioning

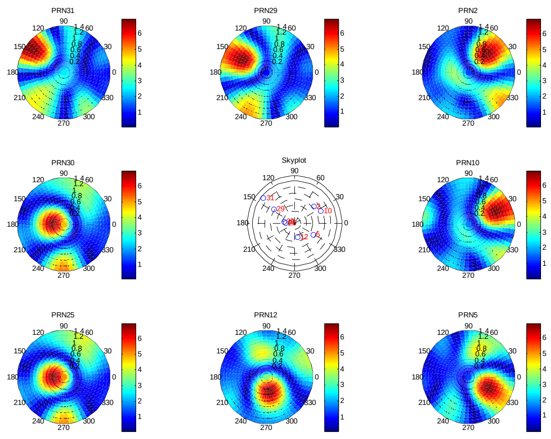


Figure 14. The gain patterns of the composite antennas by the CRPA toward the specified satellite in a sky plot format

EXPERIMENT FOR INTERFERENCE REJECTION

In order to examine the interference rejection performance of the CRPA software receiver, a single channel GPS simulator provides injected, via splitters, interference which is combines with received signal from four elements of the antenna array. The hardware set-up is depicted in figure 15. The GPS simulator generates two types of interference. One is CDMA interference obtained by setting a PRN number which is currently unallocated. The other is CW interference by turning off C/A code spreading. Figure 16 shows the power spectral density of IF signal for three cases: w/o interference, with CDMA interference, and with CW interference. The spectrum of the signal with CDMA interference has a higher power lobe within 2 MHz bandwidth than no interference case. The spectrum of signal with CW interference has a peak in the center frequency of IF.

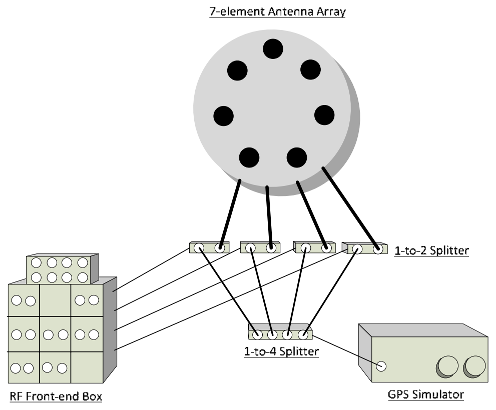


Figure 15. Diagram of the hardware set-up of the CRPA software receiver with the interference using GPS simulator

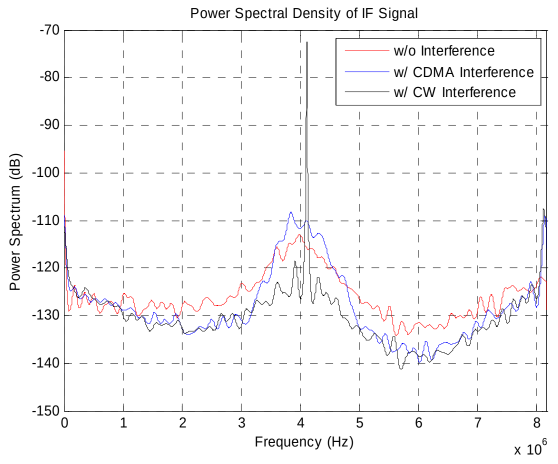


Figure 16. Power spectral density of IF signal with and without interference

The CRPA software receiver processes these signals in the post-processing mode and only combines the four signals antennas which are subject to the interference. Figure 17 shows filtered C/No of the PRN2 in the presence of CDMA and CW interferences. In both cases, the interference starts 30 seconds into the run. Without CRPA, the software receiver will lose lock on the PRN 2 signal when interference is present. The software receiver performs implements the CRPA at 10th second and contributes over 5 dB gain on C/No. When interference is not present, the performance of MVDR is close to deterministic beamforming. However, after interference is present, MVDR has gain about 0.5 dB higher than deterministic beamforming. It should be noted that these are really simple tests of null steering and meant to verify that the algorithms performing correctly.

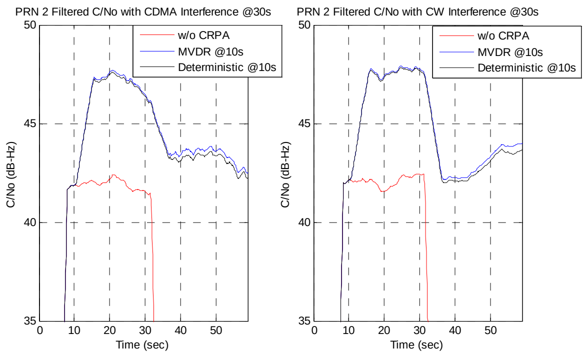


Figure 17. Filtered C/No of the software receiver performing CRPA by MVDR and deterministic beamforming in the cases with CDMA and CW interference

CONCLUSIONS

We implemented a real-time CRPA software receiver for GPS L1 C/A in a PC with a modern processor to demonstrate the feasibility of CRPA technology for civil applications. The developed algorithms implement a complete CRPA without any a prior. The result of experiments shows its performance of enhancing C/No as well as interference rejection. The software receiver can be extended to a GPS L5 by replacing RF hardware components with minimal changes to the software architecture. The Federal Aviation Administration Alternate Position Navigation and Time study is interested in the use of the CRPA with the L5 signal for robust time [14].

Other future work includes leveraging a GPU, which is well known for its parallel structure, to implement all-in-view beamforming implementation and process higher resolution data. The current 2 bit resolution does not provide significant interference rejection as high power interference can saturate the analog to digital converter. Hence processing higher resolution data is needed for robust interference rejection. We are also interested in reducing the number of channel and implementing CRPA software receiver in a single-core processor for mobile device are planned as future activities. This may be useful for allowing CRPA technology to filter into lower cost civil applications.

ACKNOWLEDGMENTS

The authors gratefully acknowledge the invaluable hardware provision of Staffan Backén for antenna array. The authors also acknowledge the electronic assistance of Doug Archdeacon for setting up PC and RF devices. This work has been supported by Federal Aviation Administration and National Science Council, Taiwan under contract NSC98-2917-I-006-106.

REFERENCES

[1] D.S. De Lorenzo, “Navigation Accuracy and Interference Rejection for GPS Adaptive Antenna Arrays,” PhD thesis, Stanford University, 2007

[2] S.P. Applebaum, “Adaptive Arrays,” IEEE Transactions on Antennas and Propagation, vol. 24, no. 5, pp. 585-598, 1976

[3] K. Borre, D.M. Akos, N. Bertelsen, P. Rinder, and S.H. Jensen, A Software-defined GPS and Galileo Receiver: A Single-Frequency Approach, Birkhäuser Boston, 2007.

[4] D.M. Akos, “A Software Radio Approach to Global

Navigation Satellite System Receiver Design," PhD thesis, Ohio University, 1997

[5] G.W. Heckler and J.L. Garrison, "SIMD correlator library for GNSS software receivers," GPS Solutions Volume 10, Number 4, pp. 269-276, 2006

[6] S. Backén, D.M. Akos and M.L. Nordenvaad, "Post-processing dynamic GNSS antenna array calibration and deterministic beamforming," Proceedings of ION GNSS 2008, pp. 1311-1319, 2008

[7] Intel Corporation, "Basic Architecture," Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 1, 2010

[8] B.M. Ledvina, A.P. Cerruti, M.L. Psiaki, S.P. Powell, P.M. Kintner, "A 12-Channel Real-Time GPS L1 Software Receiver," Proceedings of ION NTM 2003, pp. 679-688, 2003

[9] A. Greenberg and T. Ebinuma, "Open Source Software for Commercial Off-the-Shelf GPS Receivers," Proceedings of ION NTM 2005, pp. 2820-2829, 2005

[10] J-C Juang and Y-H Chen, "Accounting for data intermittency in a software GNSS receiver," IEEE Transactions on Consumer Electronics, Volume 55, Issue 2, pp. 327-333, 2009

[11] Intel Corporation, "System Programming Guide, Part 2," Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 3B, 2010

[12] P. Misra and P. Enge, Global Positioning System: Signals, Measurement, and Performance, 2nd Edition, Ganga-Jamuna Press, Lincoln, MA. , 2006

[13] P. de Jonge and C. Tiberius, "The LAMBDA method for integer ambiguity estimation: implementation aspects," Publications of the Delft Geodetic Computing Centre, 1996

[14] D.S. De Lorenzo, S.C. Lo, J. Seo, Y-H Chen and P. Enge "The WAAS/L5 signal for robust time transfer," Proceedings of ION GNSS 2010, 2010