International Conference on Computational Science and Information Management (ICoCSIM)
Vol. 1 (2012) 223-226

Local Position Estimation Using an Artificial Neural Network Based Model with a Hardware Implementation

Syahrulanuar Ngah, Rohani Abu Bakar and Abdullah Embong

Universiti Malaysia Pahang, Lebuhraya Tun Razak, 26300, Kuantan Pahang, Malaysia

Abstract

Efficient implementation of the activation function is an important part in the hardware design of artificial neural network. Sigmoid function is one of the most widely used activation function. In this paper, an efficient architecture for digital hardware implementation of sigmoid function is presented. The proposed method used second order nonlinear function (SONF) as a foundation and further improves the result by using 320 bits of read only memory (ROM) for storing a differential lookup table (differential LUT). The method proves to be more effective considering the smallest deviation of sigmoid function achieved in comparison to conventional LUT and SONF. Employing this method for hardware-based ANN in the indoor positioning system have shown that, ANN can detect the target position almost as accurate as software implementation with a speed 13 times faster. Thus the proposed idea is suitable to be implemented in a hardware-based ANN for various real-time applications.

Keywords: local positioning, artificial neural network, sigmoid function, second order nonlinear function and differential lookup table;

1. Introduction

The local positioning system gets more and more interest in recent years. Many applications will get benefits from the local positioning system to provide useful services, such as museum tour-guide, hospital health-care and location-based handoff. The position estimation relies on the received physical information, such as time of arrival (TOA), time difference of arrival (TDOA) and received signal strength indicator (RSSI), from several nearby radio access points [1-6]. The positioning algorithm has been employed for converting the received physical information to the targetposition.

This paper employs an artificial neural network (ANN) based model forthe local positioning system. Implementation of ANN falls into two categories: software implementation and hardware implementation [7]. Software implementation offers flexibility for emulating wide range of neural networks model while the hardware is vital in taking advantages of inherent parallelism [8]. The objective of ourresearch is to implement the ANN based model in the hardware for estimating the target positions with minimum error and reducing the computation time as well.

To increase the speed of the local positioning, the ANNbasedmodel has to be installed on a particular hardware. Typically, there are two rationales why ANN needs to be implemented in the hardware. First, although the performance of conventional processor is continually improved, the fastest sequential processoris still unable to provide a real-time response for the ANN witha large number of neurons. Second, most implementation has been done in the form of simulations running on a personal computer (PC) or workstation owing to the lack of portable software for running the ANN [9-10]. Alternatively, realizing the ANN on an FPGA is a smart solution to provide more portable platform for numerous real-world applications.

2. Hardware Implementation of ANN

ANN has been mostly implemented in the software. The benefit is that the designer does not need to know how the neural network elements work. The parallel structure of the ANN makes it potentially fast for hardware implementation. However, to implement the ANN on a digital hardware, special attention must be given to an activation function particularly for the nonlinear activation function. High accuracy positioning required high precision in the output of activation function. The error exists at the output of activation function were accumulated and amplified by the ANN. The final results of the ANN will have a very large error.

One of the most frequently used activation function in back-propagation neuralnetworks applications is the sigmoid function. It can be described by the following mathematical equation:

$$f(x) = \frac{1}{1+e^{-x}} \tag{1}$$

Nevertheless, this function is not suitable for direct hardware implementation. A straightforward implementation of the sigmoid function is unfeasible, since both division and exponentiation are very demanding operations. They require a lot of area

resources and long clock cycle [11]. An approximation is the practical solution. Agreat deal of work has been done on the hardware implementation of the sigmoid function.

Several methods have been reported for the implementation of the sigmoid function in hardware such as look-up table (LUT), coordinate rotation digital computer (CORDIC) function and second order nonlinear function (SONF) [12-13].

2.1 Look-up Table (LUT)

The simplest implementation of sigmoid function is the use of a LUT. Many implementations used a LUT for the approximation. However the amount of hardware memory required for implementing the LUT can be quite large especially if one requires a reasonable approximation. To create a LUT, a limited operating range should be determined. The sigmoid function has a nearly odd property.

$$f(-x) = 1 - f(x) \tag{2}$$

The size of the LUT then can be decreased to half of the desired range. The sigmoid function is nearly 1 for x > 8 and 0 for x < -8. The non-saturation range is between -8 and 8. Therefore the LUT only need to operate between 0 and 8 [12]. This requires 3 integer bits and 8 fractional bits as address bits. The resulting table has 11 address bits selecting the 8 bits fractional portion, using 2 K × 8 bits = 16 Kbits of memories. The deviation between the LUT and sigmoid function is ranging from -0.005 to 0.005.

2.2 Cordic function

A CORDIC function is the second alternative to calculate the sigmoid function. A fundamental problem with CORDIC is that it is inherently a sequential approach. This means, the CORDIC function gains accuracy at the cost of latency, where latency is defined as the number of clock cycles required from the start of the calculation until the resulting data is ready. Each additional stage in the CORDIC calculation simultaneously increases the accuracy and the time required to complete the calculation by one or more clock cycles [12]. The 11-stage CORDIC function has a latency of 50 clock cycles. The deviation resulted from the CORDIC function is ranging from -0.005 to 0.005.

2.3 Second Order Nonlinear Function (SONF)

A simple SONF has been reported to approximate the sigmoid function, which is given as follows [13]:

$$\mathrm{f(x)} = \begin{cases} 1 - \frac{1}{2}(1 - \frac{1}{4}\,|\,\mathrm{x}\,|)^2, 0 < \mathrm{x} \le 4 \\ \frac{1}{2}(1 - \frac{1}{4}\,|\,\mathrm{x}\,|)^2, -4 < \mathrm{x} \le 0 \\ 1, 4 < \mathrm{x} \\ 0, \mathrm{x} \le -4 \end{cases}$$

Same as CORDIC function, there are no hardware memory cost since this nonlinear function can be directly implemented using digital techniques. Its operation speed is higher than that in the CORDIC function. Yet this method has a limited accuracy. Figure 1 shows the output produced by SONF and sigmoid function. The deviation produced by the SONF method is ranges from -0.025 to 0.025.

3. Proposed Method

This subsection proposed a new sigmoid function method for the hardware implementations. Figure 2 illustrates the concept of the proposed method. To achieve higher accuracy, two-step approach is implemented. First, SONF is running to calculate initial sigmoid function output. As explained before, the output deviation produced by the SONF is ranging from -0.025 to 0.025. The deviation values then are stored in a LUT. This LUT is called differential LUT. Since the maximum value stored in the differential LUT is 0.025, therefore only 5 bits fractional portion is needed. The input range of the differential LUT is from 0 and 6. Then it is divided into 64 segments, for indexing the value between 0 and 0.025. As a result, the differential LUT provides 5 bits of output data. 48 segments require 3 integer bits and 3 fractional bits to be used as the address bits. The resulting differential

LUT uses 6 address bits to select the 5 bit fractional portion, using 64 × 5 bits = 320 bits of memory.Next, the value in this differential LUT are added to/deducted from the output of the SONF. If the input values for SONF are smaller than zero, then the value that is stored in differential LUT is deducted from the output value of the SONF. Otherwise the output value of the SONF is added to the value stored in the differential LUT. By using this novel approaches,(a combination of SONF and differential LUT) thedeviation achieved by the SONF is reduced to between -0.0022 and 0.0022.

4. Performance Evaluation

Computing speed is the main objective for implementing ANN into hardware. Fast computing times is required to calculate a forward pass through the ANN. Since the complexity of Eq. (1) is already simplified and replaced by the proposed method, it can be implemented directly on the FPGA. Figure 3 shows the proposed model for implementing the ANN into hardware. A total of 50 clock cycles which encloses a frequency of 1 MHz is required for the calculation of forwardpass through the ANN. This allows the ANN tobe evaluated 20,000 times per second. The software-based ANN running on a 2.4 GHzIntel i5processor is capable of performing forward calculation 1500 times per second. It has been confirmed that the computation speed of the hardware-based ANNis 13 times faster than that of the software-based ANN. In-terrn of accuracy, Table 1 shows four examplesof ANN positioning using

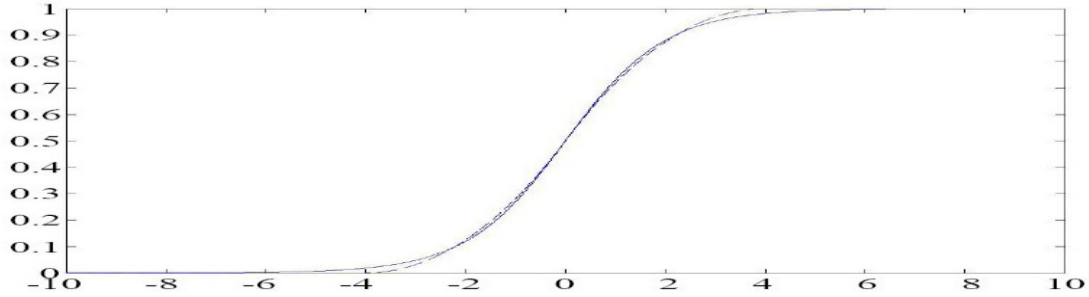

Fig.1 Output of SONF and sigmoid function

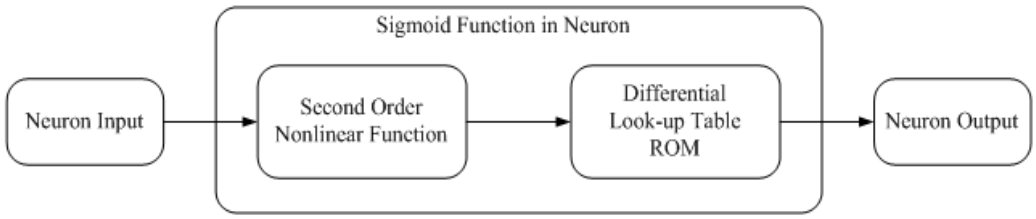

Fig.2 Proposed hardware implementation of sigmoid function

Table 1 Example of ANN output comparison between the result from Matlab and QuartusII

Input four distances (m)				True target position (m)		Target position from Matlab (m)		Target positionfrom QuartusII (m)	
d1	d2	d3	d4	X	Y	x	y	x	y
7.87	3.67	10.46	7.80	7.42	2.62	7.41	2.63	**7.42**	**2.68**
9.59	8.31	6.68	4.66	6.14	7.36	6.14	7.36	**6.15**	**7.32**
6.72	3.65	10.88	9.30	6.59	1.33	6.58	1.33	**6.60**	**1.38**
8.98	1.86	12.30	8.59	8.86	1.47	8.88	1.45	**8.89**	**1.49**

the hardware and software approaches, respectively. Compared with the true target position, the maximum error produced by the hardware implementation is 0.05m. This value is slightly higher than the error produced by the software implementations.

5. Conclusion

An efficient architecture for digital hardware implementation of sigmoid function was proposed. The proposed method employed a combination of second order nonlinear function (SONF) and 320 bits in size of the differential lookup table (differential LUT). The proposed method produced the output of sigmoid function with the deviation ranging from -0.0022 to 0.0022 in comparison to the actual sigmoid function. The achieved deviation is two times better than the conventional lookup table that used 16 Kb of ROM and 10 times than SONF. Further investigation was done by applying the proposed method in hardware-based ANN for the indoor positioning system. The simulation was shown that it can detect the target almost as accurate

as software implementation. The clear advantage is the speed of hardware-based ANN. It is 13 times faster and can be implemented in several of real time applications with higher accuracy.

References

1. 1M. Hata and T. Nagatsu: Mobile location using signals strength measurements in a cellular system, IEEE Trans. on Vehicular Technology, pp. 221-225, 1991.
2. W. C. Jakes: Microwave mobile communications, IEEE Press, 1994.
3. G. Turin, W. Jewell and T. Johnston: Simulation of urban vehicle-monitoring systems, in IEEE Trans. on Vehicular Technology, Vol. VT-21, pp. 9-16, 1972.
4. C. H. Knapp and G. C. Carter: The generalized correlation method of estimation of time delay, IEEE Trans. on Acoustics, Speech and Signal Processing, Vol. 24, No. 4, pp. 320-327, 1976.
5. Y. Gu, A. Lo and I. Niemegeers: A survey of indoor positioning systems for wireless personal networks, IEEE Communication Surveys & Tutorial, Vol. 11, No 1, 2009.
6. H. Liu, H. Darabi, P. Banerjee and J. Liu: Survey of wireless indoor positioning techniques and systems, IEEE Trans. On System, Man and Cybernatic – Part C, Vol. 37, No. 6, pp. 1067-1080, 2007.
7. S. Sahin, Y. Becerikli and S. Yazici: Neural network implementation in hardware using FPGAs, ICONIP, Part III, LNCS 4234, pp 1105-1112, 2006.
8. U. Rucket, A. Funke and C. Pintaske:Accelerator board for neural associative memories, Neurocomputing, Vol. 5, No. 1, pp 39-49, 1993.
9. Y. Liao: Neural networks in hardware: a survey, ECS 250A (Computer Architecture) Project, 2001.
10. I. Aybay,S. Cetinkaya and U. Halici: Classification of neural network hardware, Neural Network World IDG Co. Vol. 6, No. 1, pp. 11-29, 1996.
11. M. T. Tommiska: Efficient digital implementation of the sigmoid function for reprogrammable logic, IEE Proc., Computer and Digital Techniques, Vol. 150, No. 6, pp. 403-411, 2003.
12. R. W. Duren, R. J. Marks, P. D. Reynolds and M. L. Trumbo: Real-time neural network inversion on the SRC-6e reconfigurable computer, IEEE Trans. Neural Networks, Vol. 18, No. 3, pp. 889-910, 2007.
13. M. Zhang, S. Vassiliadis and J. G. D. Frias: Sigmoid generators for neural computing using piecewise approximations,IEEE Trans. Computing., Vol. 45, No. 9, pp. 1045-1049, 1996.