FPGA ACCELERATION MODULE DESIGN FOR SYSTEMATIC RESAMPLING IN PARTICLE FILTERS

Xinran Zhang, Hongfei Lian, Qiuyu Liu, Hongqi Fan

College of Electronic Science and Technology, National University of Defense Technology, Changsha 410073, China

Abstract: The resampling algorithm addresses the degeneracy problem in particle filters, but its high computational load limits real-time applications. This paper proposes an FPGA-accelerated implementation of the systematic resampling algorithm, significantly improving the processing speed of particle filters.

Keywords: systematic resampling algorithm, FPGA, particle filter

As a powerful nonlinear state estimation method, particle filtering demonstrates unique advantages in target tracking, autonomous driving, sound source localization, and other fields[1-3]. However, the high computational complexity of its resampling process severely restricts the real-time performance of particle filters, limiting their application in embedded real-time scenarios such as intelligent unmanned aerial vehicles and guidance radars[4]. To address this issue, this paper proposes an FPGA acceleration scheme based on systematic resampling. By parallelizing computations, the scheme improves computational efficiency while ensuring algorithmic accuracy, significantly enhancing the processing speed of particle filters to meet real-time requirements. This advancement expands the application of particle filters in edge devices with stringent real-time demands, such as intelligent unmanned aerial vehicles and guidance radars.

The implementation of the resampling algorithm on FPGA is divided into two main stages. First, the code is developed according to the algorithmic principles and validated for correctness using MATLAB. Subsequently, the algorithm is ported into the Vitis HLS environment, where test benches are developed for simulation. This chapter focuses on the simulation of the resampling algorithm within Vitis HLS and provides a comparative performance analysis with MATLAB simulations.

Taking 100 particles as an example, first input the particle weights and random numbers. After processing in Vitis HLS, a new set of particle weights is generated. Then, in MATLAB, plot both the original particle weights before resampling and the new weights produced by the simulation.

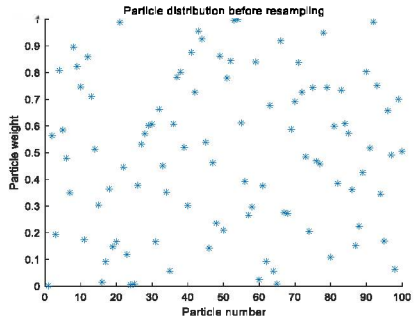

Fig.1. Particle distribution before resampling

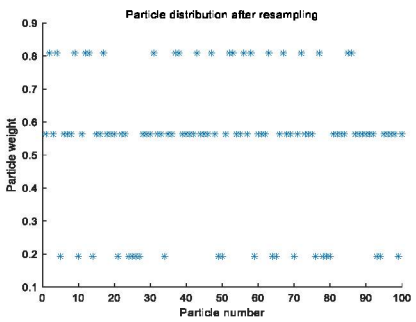

Fig.2. Particle distribution after resampling

By comparing Figure 1 and Figure 2, it can be observed that after resampling, particles with larger weights are duplicated multiple times, while those with smaller weights are replicated less frequently, t hereby verifying the correctness of the algorithm's functionality.

The timing diagram generated by the simulation is shown in figure 3:

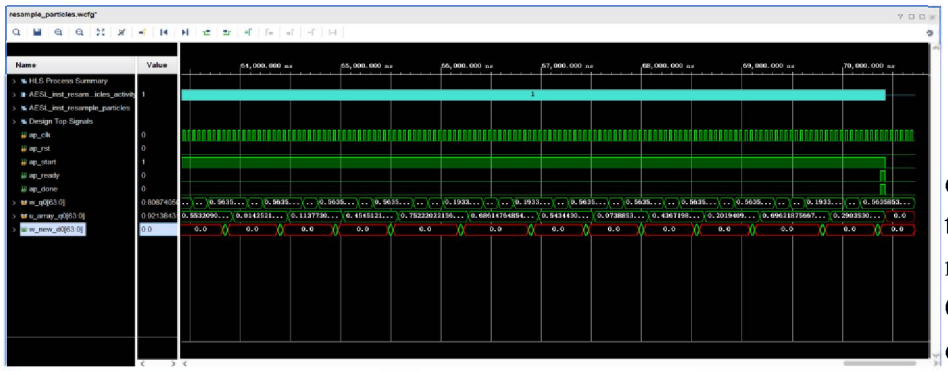

Fig.3. Timing diagram for the resampling algorithm

Figure 3 demonstrates that the simulation results match the C simulation outputs, confirming the functional correctness of the hardware implementation. The input weights and random numbers are properly sampled at clock rising edges, while resampling results are generated within the expected cycle count. Critical control signals (e.g., start, done) remain stable at clock edges with no metastability risks, verifying the successful FPGA implementation of the resampling algorithm.

The resource utilization report for the resampling implementation on FPGA is presented in Table I:

TABLE Ⅰ. The result of C synthesis

Indicators	Target	Estimated	Uncertainty	Latency(cycles)	Latency(ns)
Result	50.00ns	33.917ns	13.50ns	406	2.03E4
Indicators	Interval	BRAM	DSP	FF	LUT
Result	406	0	0	217	111

According to Table I, the clock cycle is 50.00 ns. Subtracting the uncertainty of 13.50 ns yields an actual cycle of 33.917 ns. The total cycle time for completing the resampling algorithm is 105 clock cycles (2.03E4 ns), while the MATLAB implementation requires 2.81E4 ns. This demonstrates that the algorithm meets the low-latency requirements of the resampling algorithm while occupying minimal DSP resources. Therefore, the algorithm can achieve high throughput in particle filters when processing large-scale inputs.

When the number of particles is 100, the algorithm's speedup ratio $\frac{t_{Matlab}}{t_{FPGA}}$=1.4. As the particle count NN varies, the speedup ratio also changes, as shown in Figure 4.

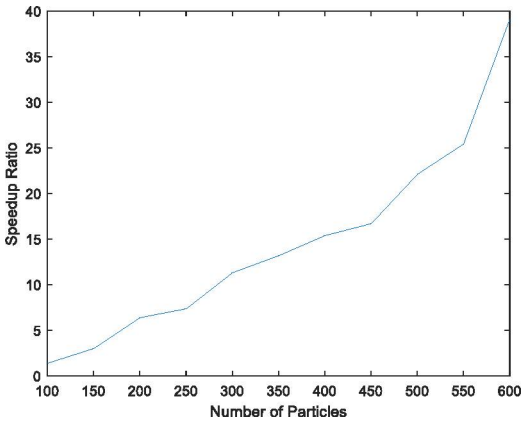

Fig.4. Speedup Ratio versus Number of Particles Diagram

As shown in the figure, the speedup ratio increases progressively with the number of particles, demonstrating more pronounced acceleration effects at higher particle counts. While the weight distribution also influences the relationship between speedup ratio and particle number, this aspect is not explored in the present study.

To address the challenges of high computational complexity and insufficient real-time performance in resampling algorithms, this paper proposes an FPGA-based hardware acceleration scheme utilizing systematic resampling. By implementing a parallel computing architecture, the solution significantly enhances processing efficiency while maintaining algorithmic accuracy, thereby dramatically improving the particle filter's execution speed. Experimental results demonstrate that the proposed approach achieves high energy efficiency and superior real-time performance, effectively enabling the deployment of particle filters in latency-critical edge computing applications such as intelligent UAVs and guidance radar systems.

Reference

[1]Wang Ning, Duan Rui, Zhou Xiaoyi. A Box Particle Filtering Approach for Target Tracking Under Measurement Uncertainty[J]. Journal of Electronics & Information Technology, 2024, 46(9): 3654-3661.

[2]Gao Yan, Fu Chunyun, Yang Zhong, Yang Guanlong. Vehicle Speed Estimation Based on Improved Particle Filter Algorithm[J]. Journal of Chongqing University, 2024, 47(3): 44-52.

[3]Liu Wangsheng, Pan Haipeng, Wang Minghuan. Composite Model Particle Filter for Indoor Sound Source Localization Using Multi-Feature Fusion [J]. Acta Armamentarii, 2024, 45(3): 975-985.

[4]A. KRISHNA, A. VAN SCHAIK, C. S. THAKUR. FPGA Implementation of Particle Filters for Robotic Source Localization[J]. IEEE Access, 2021, 9: 98185–98203

[5]Su Zhibao, Lu Jilian. Research Methods for Formation Control of Multiple Mobile Robots [J]. Robot, 2003, 25(1): 88-91.