Parallel Particle Filter Algorithm and Its FPGA Implementation

Xiaofeng Lu, Shuhui Wang, Zhiying Du, Dongbin Pei, Dingyuan Zheng, Tongchun Zuo

School of Communication and Information Engineering Shanghai University, Shanghai Municipality, China
luxiaofeng@shu.edu.cn, wongshgogo@gmail.com, duzhiying@live.com, dongbinpei@126.com

Abstract - Visual target tracking is the key problem in intelligent video processing. Particle Filter is classic and effective in visual target tracking algorithms, but it needs to analyze a large amount of probability statistic, leading to high algorithm complexity and low calculation efficiency. FPGA provides a competitive alternative for hardware acceleration to these applications. In this paper, we modify Particle Filter algorithm and propose a FPGA-based hardware accelerating architecture. Experiments show the embedded architecture is robust and shows a great real-time performance**.**

Index Terms - parallel Particle Filter, FPGA, embedded vision

1. Introduction

Visual target tracking is the key problem in intelligent video processing applications. It is related to the image processing and pattern recognition. In military, precision guided system [1] uses the accurate tracking technique to get target location. In transportation, intelligent transportation system [2] can analyze the traffic flow statistics and anomaly detection by tracking cars. In security, the intelligent monitoring system [3] will raise the alarm if the image captured by camera is anomalous.

Target tracking was proposed firstly by Wax in 1955. CamShift and Particle Filter are two widely used algorithms in many target tracking techniques because of their excellent performance in real time and robustness.

Bradski proposed the CamShift algorithm [4-6] on basis of MeanShift algorithm [7-10]. CamShift algorithm is also called self-adaption MeanShift algorithm.

Particle Filter algorithm [11-12] is based on Bayesian estimation and Monte Carlo method, and it is suitable for the non-linear and non-Gaussian systems presented by any state space model that is approximate to the optimal estimation [13]. It is a great improvement to Kalman Filter algorithm [14-16].

Particle Filter and CamShift are well developed on PC. PC shows powerful computing performance, but it has great disadvantage of large size, high cost and power consumption. Compared to PC systems, embedded systems are less powerful in storage space and computing capability, but they are flexible and less consuming. Considering the cost, embedded system is a better choice for target tracking algorithm implementation.

Lots of researchers use DSP, ARM or FPGA to implement target tracking algorithm. Pan Lian [17] proposed a pedestrian tracking system on ARM. Li Yanbin [18] modified Particle Filter on DSP. Deng Minxi [19] built a real-time target tracking system on DSP. Saeed Ahsan [20] used the parallel structure of FPGA to track the moving objects and proposed the FPGA-based real-time target tracking on a mobile platform.

Compared to PC, ARM and DSP, FPGA has a great performance in parallel structure. If there are 100 particles need to be computed, PC will cost plenty of time due to the serial processing pattern. FPGA processes the 100 particles in parallelism and shorten the computing time.

2. Particle Filter Algorithm

Particle Filter is an optimal algorithm based on Bayesian estimation and Monte Carlo method. It uses sequential process to measure data by recursive fashion, so it is unnecessary to store and process the previous data and saves storage space. The algorithm flow chart is shown in Fig.1.

The steps of the Particle Filter are shown as follows:

Initialization: $t = 0$.

Collect the particle set $\{ x_0^{(i)}, i = 1,2,...,N \}$ from the prior distribution $p(x_0)$.

If $t \geq 1$, $\{x_{t-1}^{(i)}, w_{t-1}^{(i)}\}_{i=1}^{N}$ is the particle set from posteriori distribution at the $t-1$ moment.

Importance sample:

Sample the particles $\tilde{x}_t^{(i)} \sim I(x_t \mid x_{t-1}^{(i)}, y_t)$ from the importance distribution $I(x_t \mid x_{t-1}^{(i)}, y_t)$.

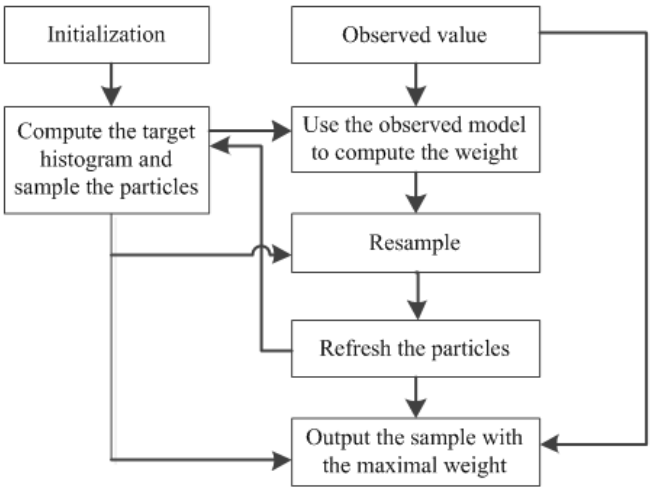

Fig. 1 Flow chart of Particle Filter algorithm.

© 2014. The authors - Published by Atlantis Press

At the moment y_t, compute the weight of the particle $\{\tilde{x}_t^{(i)}\}_{i=1}^N$:

$$w_t^{(i)} = w_{t-1}^{(i)} p(y_t \mid \tilde{x}_t^{(i)}) p(\tilde{x}_t^{(i)} \mid \tilde{x}_{t-1}^{(i)}) / I(\tilde{x}_t^{(i)} \mid \tilde{x}_{t-1}^{(i)}, y_t). \quad (1)$$

Normalize the particle weight:

$$\tilde{w}_t^{(i)} = w_t^{(i)} / \sum_{i=1}^{N} w_t^{(j)}. \quad (2)$$

Resample:

$$\hat{N}_{eff} = \sum_{i=1}^{N} (\tilde{w}_t^{(i)})^2. \quad (3)$$

If $\hat{N}_{eff} < N_{th}$, then define the resample particle as

$$\{x_t^{(i)}, 1/N\}_{i=1}^N. \quad (4)$$

If $\hat{N}_{eff} \geq N_{th}$, then define the resample particle as

$$\{x_t^{(i)}, w_t^{(i)}\}_{i=1}^N = \{\tilde{x}_t^{(i)}, \tilde{w}_t^{(i)}\}_{i=1}^N. \quad (5)$$

Output the particle set $\{x_{0:t}^{(i)} : i = 1,2,..., N\}$.

Posteriori probability:

$$p(x_{0:t} \mid y_{1:t}) = \hat{p}(x_{o:t} \mid y_{1:t}) = \frac{1}{N} \sum_{i=1}^{N} \delta_{x_{0:t}^{(i)}}(dx_{0:t}). \quad (6)$$

State approximation: $\hat{x}_t = \sum_{i=1}^{N} \tilde{w}_t^{(i)} x_t^{(i)}$.

According to the law of large numbers, define the expectation of the function $g_t(x_{0:t})$ as

$$E(g_t(x_{0:t})) = \int g_t(x_{0:t}) p(x_{0:t} \mid y_{1:t}) \approx \frac{1}{N} \sum_{i=1}^{N} g_t(x_{0:t}^{(i)}). \quad (7)$$

At the moment of $t-1$, the posteriori probability $p(x_{t-1} \mid y_{1:t-2})$ of the target state variable is represented by approximating particle set $\{\tilde{x}_{t-1}^{(i)} : i = 1,2,..., N\}$.

The weight of the particles is $1/N$. Compute the weight $\tilde{w}_{t-1}^{(i)}$ of the particle $\tilde{x}_{t-1}^{(i)}$. The particles with high accuracy have high weight, while those with big error have low weight. Then, we get the particle set $\{\tilde{x}_{t-1}^{(i)}, \tilde{w}_{t-1}^{(i)}\}_{i=1}^N$. Particles with high weight will derive more particles than those with low weight. The derived particles show uniform distribution as well. Therefore, we get the particle set $\{\tilde{x}_{t-1}^{(i)}, 1/N\}_{i=1}^N$ to predict next particle set $\{\tilde{x}_t^{(i)}, 1/N\}_{i=1}^N$.

3. Hardware Implementation of Particle Filter

The module partition of the Particle Filter hardware architecture is shown in Fig.2. It includes initialization, target selection, color histogram statistics of the target and each particle, particle weight computation, target capture, resampling and RAM. Initialization is to clear RAM. Then, set the size and location of the target. The display module is to output the original image and computing result.

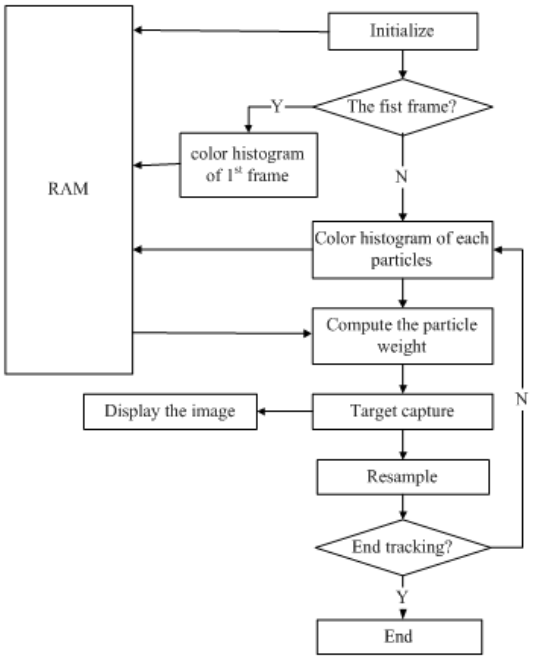

Fig. 2 Module partition of Particle Filter hardware architecture.

We choose the H-component histogram as the feature model to track the target.

If the input image is the first frame, this system generates the color histogram and prepares for the weight computing.

If the input image is the first frame, this system generates the color histogram and prepares for the weight computing.

We use the H-component as the RAM address to do read-write operation. The read data equals the written data adding one.

Particle Filter computes the posteriori probability distribution of the target state variables by a series of random particles. We use the method shown in Fig.3 to produce random particles (x, y).

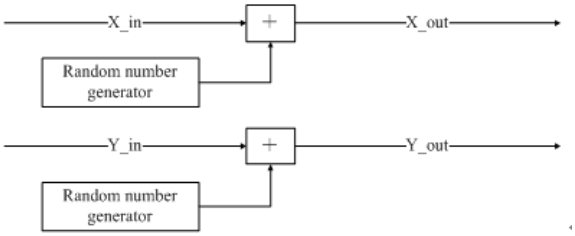

Fig. 3 Particles generator.

Given the coordinate of each particle, we use the method mentioned in the previous section to get the color histogram of each particle.

Computing the weight of the particles is the key to Particle Filter, involving lots of multiplication, division, radication and superposition. We can take advantage of the FPGA parallel structure to compute the weight of each particle in parallelism, but we still need compute enough particles to have a great experimental effect. In this paper, we use 128 particles, so it costs lots of hardware consumption. Considering the slot time between each frame, we use the time multiplexing to compute the weight of the particles, as shown in Fig.4.

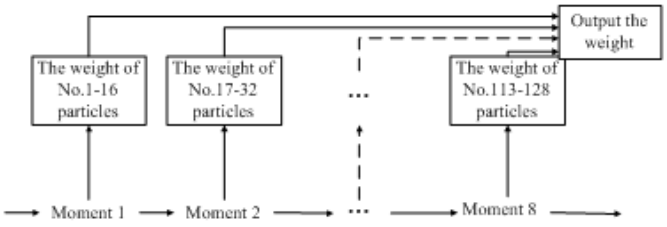

Fig.4 Compute the weight of the particles.

After getting the weight of each particle, we sort them and get the particle with the biggest weight. Choose the particle ordinate as the target ordinate.

Particles are random in the Particle Filter. Particles with small weight contribute little, but they lead to severe degeneracy. Resampling deletes the particles with small weight and replaces them with large weighted particles.

We use the RAM on FPGA to store the color histograms of the target and each particle.

4. Experimental Results

As shown in Fig.5, we use the multimedia circuits design resources on the DE2-115 to build this FPGA-based target tracking system. The camera is to collect the video signal. We implement CamShift algorithm to compare its performance with Particle Filter, and use DE2-115 board to implement them. The display is to output the processed video signal. The left part of the display is experimental results of CamShift algorithm and the right part is Particle Filter algorithm.

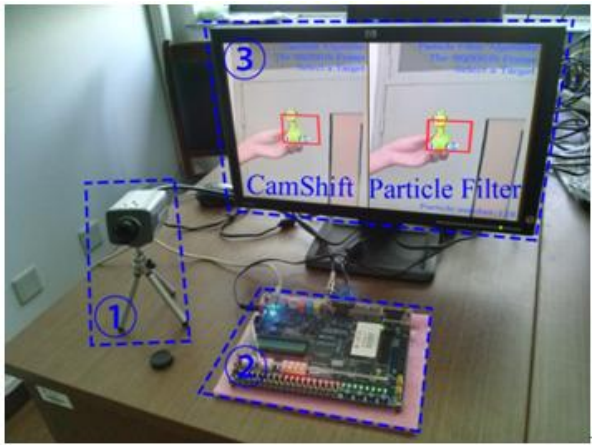

Fig.5 FPGA-based visual target tracking system.

A. *Qualitative Experimental Results*

1) Complex Scenario: In a colorful and complex scenario, the tracking effects of CamShift and Particle Filter algorithms are shown in Fig.6 Both algorithms choose the color as their features, so colorful scenario causes the interference to target tracking. Experiment shows CamShift and Particle Filter are still effective in complex scenario, but the shift extent of CamShift is more serious than Particle Filter.

2) Complex Target: As shown in Fig.7, we hold two toys in hand as the target and compare the performances of both algorithms. From the 123rd, 272nd and 340th frames, CamShift loses the target, while Particle Filter can track the target accurately. When we track the target with one color, CamShift shows a great performance. If the target includes complex color information, the tracking effect is poor. However, Particle Filter still shows a strong performance with a complex target.

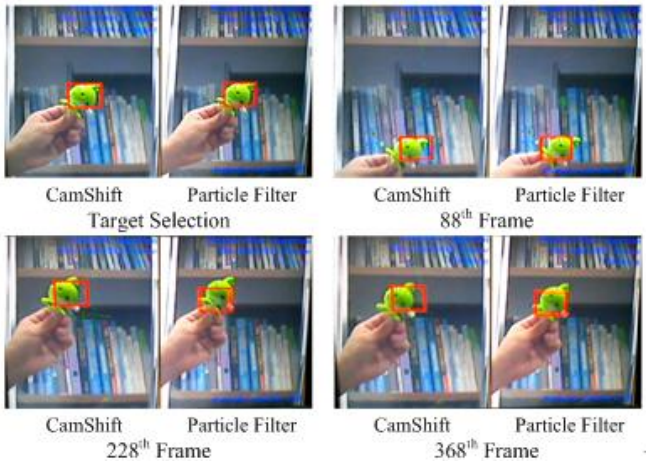

Fig.6 Target tracking effect in a complex scenario.

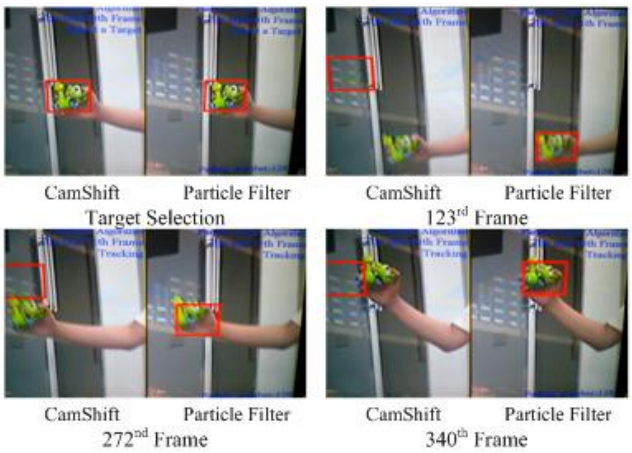

Fig.7 Target tracking effect with a complex target.

3) Occlusion: In practical application, if a car is waiting during the red light and a man is crossing the road, the man will occlude the car. Therefore, we need the algorithm to be

anti-occluded. As shown in Fig.8, from 296th frame, CamShift loses the target, while Particle Filter still has a great performance.

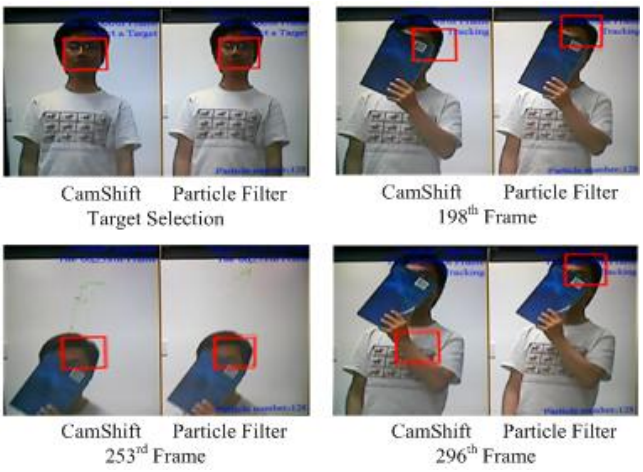

Fig.8 Target tracking effect with an occlusion.

B. Quantitative Experimental Result

Table I shows the operation time to compute one frame with different particle numbers. It costs long time if there are too many particles to compute on PC, while FPGA costs the same time with different particle numbers. With the increasing of the particle numbers, PC will compute them one by one. However, with the help of FPGA parallel structure, Particle Filter can compute all particles in parallelism. Experiment shows compared to PC systems, the operation time has been largely accelerated by the hardware architecture. When we have a large number of particles to be computed, this advantage will be obvious.

5. Conclusion

In this paper, we propose a FPGA-based hardware architecture of Particle Filter. Experiments show Particle Filter is anti-jamming, on condition of complex scenario, complex target and occlusion. Compared to PC, FPGA parallelism largely accelerates the operation time. Experiments show the proposed hardware accelerating architecture has a great performance.

Acknowledgment

This research was supported in part by Education and Teaching Reform Project of Shanghai University and Innovation Project of Shanghai (CXSJ12-047,CXGJ12-013,CXSJ12-037).

TABLE I Operation Time of Particle Filter Algorithm on PC and FPGA

Particles numbers	PC operation time(s)	FPGA operation time(s)
8	0.0191	0.0167
16	0.0342	0.0167
32	0.0493	0.0167
64	0.1029	0.0167
128	0.1920	0.0167

References

[1] X. Yao, et al, "Precise guidance techniques," *J. Laser and Infrared,* vol. 36, no. 5, pp. 338-340, 2006.

[2] Betke M., et al, "Real-time vision system for automatic traffic monitoring," *J. Image and Vision Computing,* vol. 18, no. 10, pp. 781-794, 2000.

[3] Mohamed F. A., Rama C., and Zheng Q. "Integrated motion detection and tracking for visual surveillance," in Proceeding of the Fourth IEEE International Conference on Computer Vision System. New York: IEEE, 2006.

[4] Bradski, G.R., "Real time face and object tracking as a component of a perceptual user interface," in Applications of Computer Vision, pp. 214-219, 1998.

[5] Collins R T, and Y. Liu, "On-Line selection of discriminative tracking features," *J. IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 27, no. 10, pp. 1631-1643, 2005.

[6] Allen J. G., Richard Y. D., and Jin J. S., "Object tracking using CamShift algorithm and multiple quantized feature spaces," in Pan-Sydney Area Workshop on Visual Information Processing, Australian: Sydney, pp. 1-5, 2003.

[7] Zhou S. K., Chellappa R., and Moghaddam B., "Visual tracking and recognition using appearance-adaptive models in particle filters," *J. IEEE Transactions on Image Processing,* vol. 13, no. 11, pp. 1491-1506, 2004.

[8] Fukunage K., and L.D.Hostetler, "The estimation of the gradient of a density function with application in pattern recognition," *J. Information Theory,* vol. 21, no. 1, pp. 32-40, 1975.

[9] Comaniciu D., Ramesh V., and Meer P., "Real-Time Tracking of Non-Rigid Objects using Mean Shift," *J. Computer Vision and Pattern Recognition,* vol.2, pp. 142-149, 2000.

[10] Yizong Cheng, "Meanshift, mode seeking, and clustering," *J. Pattern Analysis and Machine Intelligence,* vol. 17, no. 8, pp. 790-799, 1995.

[11] Arulampalam M. S., Maskell S., and Gordon N., "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *J. IEEE Transactions on Signal Processing,* vol. 50, no. 2, pp. 174-188, 2002.

[12] Doucet A., Gordon N. J., and Krishnamurthy V., "Particle filters for state estimation of jump Markov linear systems" *J. IEEE Transactions on Signal Processing,* vol. 49, no. 3, pp. 613-624, 2001.

[13] D. Crisan and A.Doucet, "A Survey of convergence results on particle Filter methods for practitioners," *J. IEEE Trans, Speech and Audio Proc.,* vol. 10, no. 3, pp. 173-185, 2002.

[14] R.E.Kalman, "A New Approach to Linear Filter and Prediction Problems," *J. Transactions of the ASME--Journal of Basic Engineering,* vol. 82, no. D, pp. 35-45, 1960.

[15] S. Julier, J. K., Uhlmann, H. F., and Durrant-Whyte, "A New Method for the Nonlinear Transformation of Means and Covariances in Filters and Estimators" *J. IEEE Transactions on Automatic Control,* vol. 45, no. 3, pp. 477-482, 2000.

[16] R. P. Wishner, J. A., Tabaczynski, and M. Athans, "A Comparison of Three Non- linear Filters," J. Automatica, vol. 5, no. 5, pp. 487-496, 1969.

[17] L. Pan, and X. Liu, "The study of target tracking based on ARM embedded platform," *J. Journal of Computers,* vol. 7, no. 8, pp. 2015-2023, 2012.

[18] Y. Li, Z. Cao, and C. Liu, "Particle filter algorithm for target tracking based on DSP" *J. Journal of Optoelectronics Laser,* vol. 20, no. 6, pp. 771-774, 2009.

[19] M. Deng, Q. Guan, and S.Xu, "Intelligent video target tracking system based on DSP," in International Conference on Computational Problem-Solving, 2010.

[20] Saeed, Ahsan, Amin, Adeel, Saleem, and Shehzad. "FPGA based real-time target tracking on a mobile platform," in International Conference on Computational Intelligence and Communication Networks, 2010.