Original Research Paper

Embedded Architecture for Object Tracking using Kalman Filter

Ahmad Abdul Qadir Al Rababah

Faculty of Computing and Information Technology in Rabigh,
King Abdulaziz University, Rabigh 21911, Kingdom of Saudi Arabia

Abstract: Intelligent video is a new area of research fairly wide allowing to do a study, analysis, or interpretation of digital video such as motion analysis. However, for a video surveillance system, a motion analysis task of digital video includes the detection of moving objects and their tracking. The object detection allows the location of the regions of interest, which represents a change of movement. The purpose of tracking is to maintain the identity of objects detected over time by the estimation or the location of their position in each frame of the sequence. The most popular tracking algorithm is the Kalman filtering. In this study a hardware architecture for moving object tracking using Kalman filter on a FPGA board, is proposed.

Keywords: Moving Object Tracking, Kalman Filter, Hardware Architecture, FPGA

Introduction

Smart video surveillance is a process of automatically identifying in video sequences, objects, behaviors or events preset by a user or learned by the system. It analyzes and transforms data from one (or more) camera into a semantic interpretation directly usable by a human operator. For example, when an abnormality is detected, the system can send an alert to the staff so that it can decide on the appropriate intervention to set up. Generally, video surveillance systems consist of the following steps: Detection of moving objects, tracking and behavioral analysis of such objects (Yilmaz *et al*., 2006).

Motion detection techniques can be considered as a preprocessing to reduce analyzing an amount of information. A moving object is detected if the position changes relative to that of a set of static objects or if it is located in an image acquired at the time t of the sequence at a different position from that occupied in the previous image. The object motion in the video can therefore be detected by comparing the contents of at least two frames from the sequence.

The object tracking is to associate objects detected in the current image with those detected in previous images. This is to maintain object identity and the time evolution of their positions (or other characteristic). It can be considered as a problem of spatial and temporal location of objects in the scene (Junzo *et al*., 2010). It is a classic problem and widely studied in the field of

motion analysis, because of its many applications among which we can note video surveillance, robotics, video compression, the study of the objects deformation. Thus, many tracking algorithms have been proposed in recent years (Blackman, 1986).

The most popular tracking algorithm is the Kalman filter. It is essentially a set of recursive equations used together to help modeling and evaluating precisely the movement of a linear dynamic system. In this study an embedded architecture for moving object tracking using Kalman filter, is proposed.

This paper is organized as follows: After introducing, section 2 presents a description of the Kalman filter. Its hardware implementation is then described in section 3. Section 4, shows the implementation results and some comparison. Finally, concluding remarks are given in section 5.

The Overview of Kalman Filter

Definition

The algorithms to identify the state from measurements are called filters because their action is to restore the system state by eliminating the uncertainties and noises that disturb the information we have on the state. The Kalman filter is based on a diagram of the type prediction-correction. It uses innovative information to correct the prediction of the state taking into account the previous estimate.

The principle of Kalman filtering is to be placed in a probabilistic modeling of uncertainty modeling and noise measurement to calculate the innovation gain. The reconstruction of the state according to measurements is then a problem of Bayesian estimation, i.e., we determine the distribution probability of the posteriori state given the available measures (Breton, 2004).

In the tracking process, the Kalman filter 'looks' at an object when it moves, i.e., it takes the information on the state of the object at that moment. Then, it uses this information to predict where the object is located in the next frame. The position of the object is then corrected considering the prediction and also observation.

The Kalman filter provides an estimate of a process using a feedback system in the form of noisy measurements. In this manner, the Kalman filter equations fall into two types of equations: Prediction equations (extrapolation) and update equations (correction). The extrapolation equations are responsible for propagating the estimated present state and error covariance to obtain the *a priori* estimates of the next step. While the correction equations are responsible for the feedback, i.e., introduce a new measure with the estimated *a priori* to obtain an improved estimated

posteriori. Therefore, the Kalman filter finally resembles a predictor-corrector filter (Fig. 1).

Equations

The Kalman filter equations provide a smart computational way for estimating the state of a process. It uses two different types of equations in a prediction of the state variable; these being both the extrapolation equations and the update equations.

The extrapolation equations are seen as prediction equations as they are responsible for predicting the current state and covariance matrix, used in time t+1 to predict the previous state. k is representative of the time step, so the prediction equations are basically indicative of k+1.

The update equations are seen as correction equations as they are responsible for correcting the errors that have been made in the extrapolation equations. They are back propagating to get new values for the prior state to improve the prediction for the next state.

In the Kalman filter, the equations describe the state of the successive time system; through these equations we can predict the future state using the state without depending on previous states in the calculations. Figure 2 shows the whole operation of the Kalman filter.
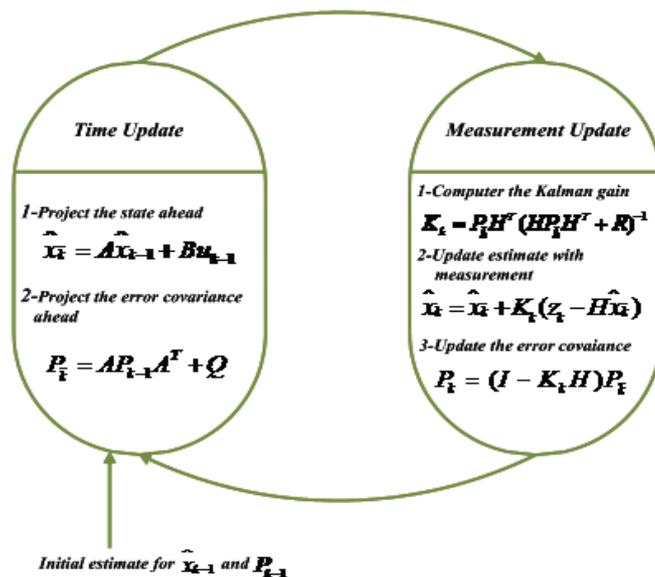

Fig. 1. Kalman filter cycle

Fig. 2. The whole operation of the Kalman filter

Table 1. Definition of variables in the Kalman filter equations

P_k	Posteriorerrorcovariance
$\hat{x}_k$	Posteriorstate estimate
Z_k	State measurement
K	Kalman gain
R	Measurementerror covariance
Q	Random white noise
A	Variable
B	Variable
u_k	Control variable
H	Matrix-valuedfunction

For computing the extrapolation equations, we have to project the state estimation and the state covariance matrix as follows:

$$\hat{x}_{\bar{k}} = A\hat{x}_{k-1} + Bu_{k-1}$$

$$P_{\bar{k}} = AP_{k-1}A^T + Q$$

Where:

$\hat{x}_{\bar{k}}$ and $P_{\bar{k}}$ = Respectively the *a priori*state estimate and estimate error covariance

$\hat{x}_{k-1}$ and P_{k-1} = The *a posteriori* state estimate and estimate error covariance

The definition of remaining additional variables in the Kalman filter equations is presented in Table 1.

When dealing with the measurement update equations, three measurements should be computed:

- The Kalman gain denoted by K_k:

$$K_k = P_{\bar{k}}H^T(HP_{\bar{k}}H^T + R)^{-1}$$

- The posterior state estimate denoted by $\hat{x}_k$:

$$\hat{x}_k = \hat{x}_{\bar{k}} + K_k(z_k - H\hat{x}_{\bar{k}})$$

- The posterior error covariance denoted by P_k:

$$P_k = (I - K_kH)P_{\bar{k}}$$

Hardware Implementation

The Kalman filter algorithm is broken up into its constituent operations. These operations include addition, subtraction, multiplication and reciprocal function. Figure 3 shows the proposed architecture of Kalman filter that consists of 16 mathematical operations including various matrix manipulations, which can be computed in 10 calculations cycle. Each calculation cycle involves at least one constituent operation. For example, in the first calculation cycle, two independent multiplications can be calculated at the same time, since the input of one does not depend on the output of the other.
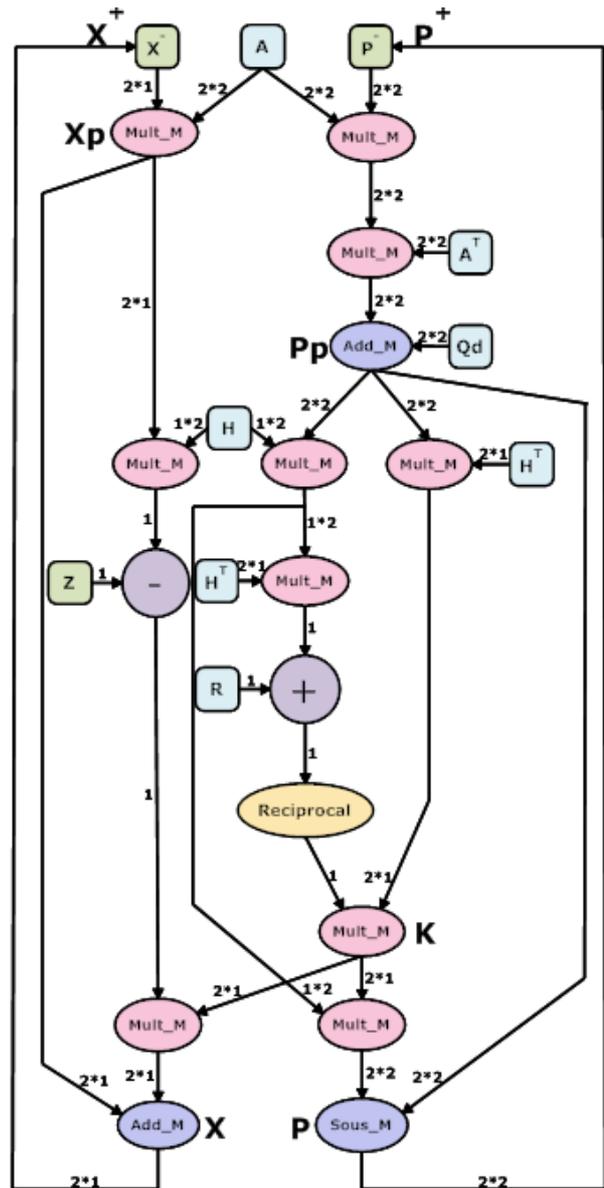

Fig. 3. The proposed architecture of Kalman filter

For embedded applications, real-time constraint is primary considerations. So, speed optimization is most critical for the Kalman filter design proposed in this study.

As regards the hardware design of the Kalman filter, Fixed point was chosen due to the precision it allows for and the speed benefits it affords. If an application can be done in fixed-point arithmetic, it will probably run faster than in any other format because fixed point is the natural language of the processor (Greg and Bishop, 2006). A 32-bit fixed-point binary number represents all numbers. The 32-bit fixed-point representation uses15 bits for the integer portion, 16 bits for the fractional portion and one bit for the sign.
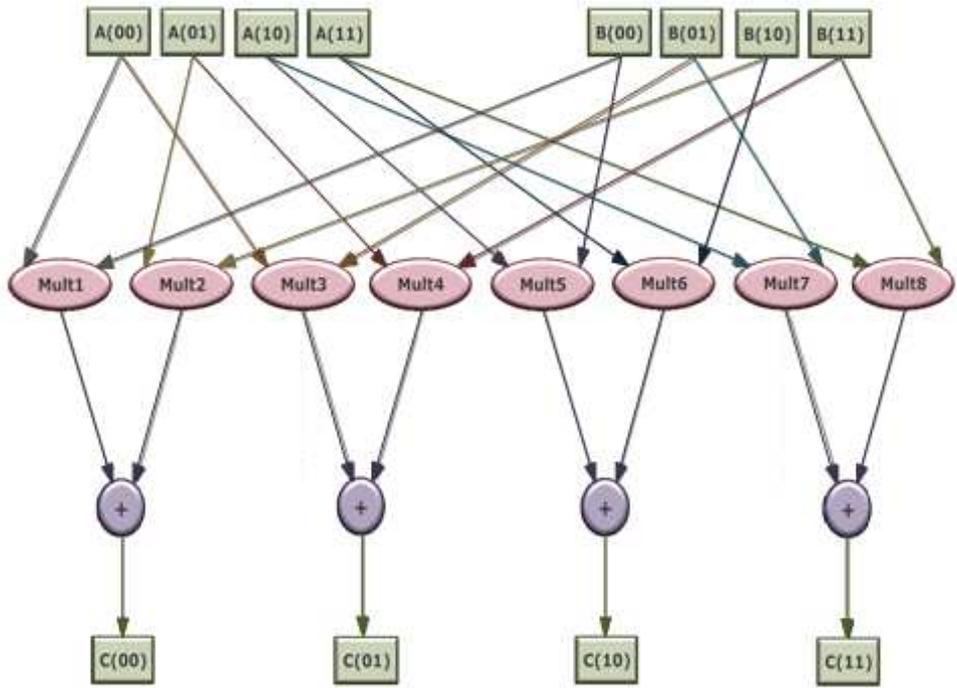
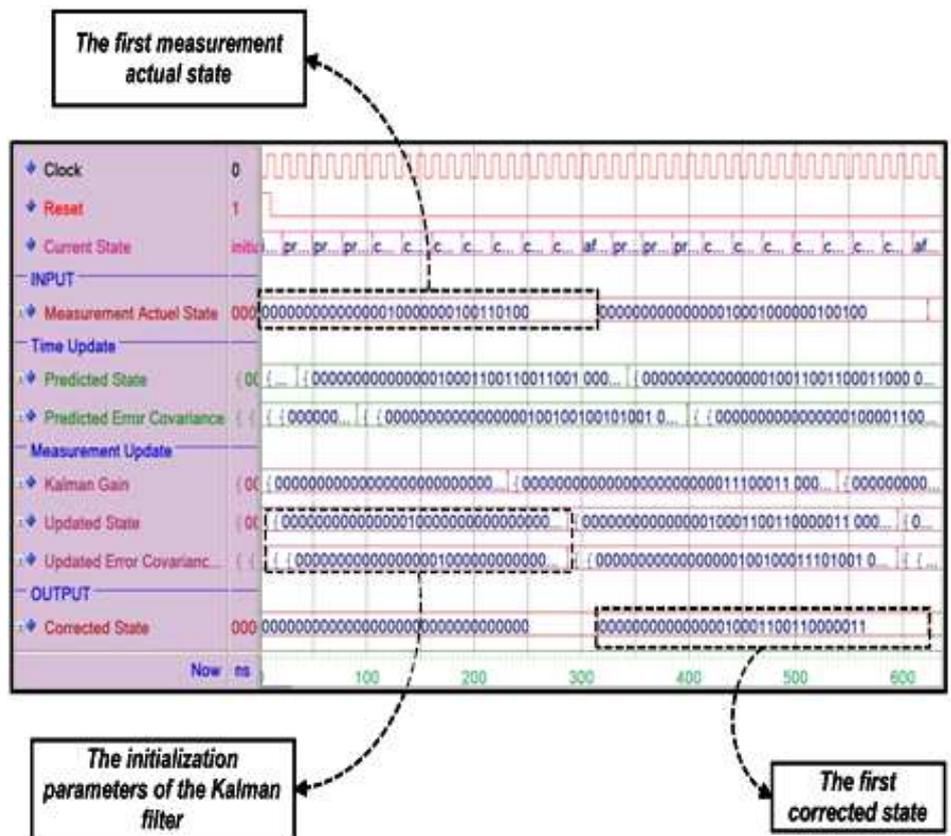Fig. 4. Graphical representation of two 2×2 matrix multiplication

Fig. 5. The simulation results of the proposed architecture

Table 2. FPGA resource occupation

	XC5VLX200T		
	Used	Available	(%)
Number of slice registers	855	207360	0
Number of fully used LUT-FF pairs	746	207360	0
Number of slice FF	475	1126	42
Maximum frequency	115.617 MHz		

Table 3. The comparison table

	Virtex-4 XC4VFX140		Virtex-4 XC4VSX35	
	Our design	Design [9]	Our design	Design [10]
Maximum. Frequency	88.866MHz	41.6 MHz	80.727MHz	40 MHz

Efficiently multiplying matrices is of paramount importance for any Kalman Filter implementation. The vast majority of mathematical operations are performed on matrix. Graphical representation of two 2×2 matrix multiplicationis presented in Fig. 4.

It is necessary to calculate the reciprocal as part the calculation of K that is the Kalman gain or blending factor (Liu *et al*., 2007). A problem arises when performing division in binary. Without properly shifting the decimal place to the left, the answer will not be correct. To achieve the correct answer the dividend must be left-shifted by the number of bits that one wishes to have to the right of the radix point.

Implementation Results and Evaluation

Figure 5 presents the simulation of the proposed architecture of Kalman filter, obtained by Modelsim, where we presented the input, the output and key parameters of the system. The parallelization is the method used for creatinga hardware architecture that meets this requirement. The more parallel calculations that can be done the fewer clock cycles over all that would be needed to compute one iteration of the Kalman filter.

The implementation of the proposed architecture has been done on a Xilinx Virtex 5 XC5VLX200TFPGA device. it is necessary to represent both the integer and fractional portion of a number. Floating point provides a large dynamic range but with relatively poor precision Table 2 shows the requested hardware resources and maximum operation frequency shows in Table 3.

Conclusion

In this study, we proposed a parallel architecture for object tracking using Kalman filter, which allows having optimizations at speed. The implementation of the proposed design has been done on the Virtex 5 XC5VLX200T FPGA device. When comparing the implementation results of our proposed design for Kalman filter with the design proposed on the Xilinx

Virtex-4 XC4VFX140 FPGA device and the design proposed in (Hancey, 2008) on Xilinx Virtex-4 XC4VSX35 FPGA device (see Table), we found that our design is two times faster than other implementations. These optimizations are achieved by focusing on the parallelization and trying to facilitate mathematical operations of this filter.

Ethics

This paper is original and contains unpublished material. The corresponding author is a single author and he has read and approved the content and no ethical issues involved.

References

Blackman, S., 1986. Multiple-Target Tracking with Radar Applications. 1st Edn., Artech House, Dedham, ISBN-10: 0890061793, pp: 449.

Breton, M., 2004. Suivi temporel d'objets en 2D et 3D: Revue de littérature sur le filtre de Kalman et autres méthodes par propagation de densité de probabilité conditionnelle.

Greg, W. and G. Bishop, 2006. An introduction to the Kalman filter. University of North Carolina at Chapel Hil.

Hancey, B., 2008. Characterization and implementation of a real-world target tracking algorithm on field programmable gate arrays with Kalman filter test case. MSc Thesis, Department of the Air Force Air University Air Force Institute of Technology.

Junzo, W., M. Zalili, J. Lakhmi and F. John, 2010. Human Tracking: A State-Of-Art Survey. Knowledge-Based and Intelligent Information and Engineering Systems, Setchi, R., I. Jordanov, R.J. Howlett and L.C. Jain (Eds.), Springer, ISBN-10: 978-3-642-15389-1, pp: 454-463.

Liu, Y. C.S. Bouganis and P.Y.K. Cheung, 2007. Efficient mapping of a Kalman filter into an FPGA using Taylor expansion. proceedings of the International Conference on Field Programmable Logic and Applications, Aug. 27-29, IEEE Xplore Press, Amsterdam, pp: 345-350. DOI: 10.1109/FPL.2007.4380670

Yilmaz, A., O. Javed and M. Shah, 2006. Object tracking: A survey. ACM Comput. Surv., 38: 45-45. DOI: 10.1145/1177352.1177355