

로봇 팔의 시간지연제어기의 가속도 평가를 위한 Kalman 필터의 FPGA 임베디드 설계

Embedded Kalman Filter Design Using FPGA for Estimating Acceleration of a Time-Delayed Controller for a Robot Arm

전 효 원, 정 슬*

(Hyo-won Jeon and Seul Jung)

Abstract : In this paper, an embedded Kalman filter for a time-delayed controller is designed on an FPGA to estimate accelerations of the robot arm. When the time-delayed controller is used as a controller, the inertia estimation along with accelerations is needed to form the control law. Although the time-delayed controller is known to be robust to cancel out uncertainties in the nonlinear systems, performances are very much dependent upon estimating the acceleration term $\ddot{q}(t-\lambda)$ along with inertia estimation $\hat{D}(t-\lambda)$. Estimating accelerations using the finite difference method is quite simple, but the accuracy of estimation is poor specially when the robot moves slowly. To estimate accelerations more accurately, various filters such as the least square fit filter and the Kalman filter are introduced and implemented on an FPGA chip. Experimental studies of following the desired trajectory are conducted to show the performance of the controller. Performances of different filters are investigated experimentally and compared.

Keywords : FPGA, Kalman filters, robot arms, time-delayed controller

I. 서론

PID 제어기는 현재까지 사용되는 제어기 중 가장 실용적이며 많이 사용되는 제어기이다. 선형시스템 제어에 있어서 매우 만족스런 결과를 얻을 수 있기 때문에 아직까지도 대부분의 산업공정에서 사용되고 있다. 하지만 기술이 발달되고 시스템이 점점 복잡해짐에 따라 시스템의 비선형 특성을 고려해야 하는 문제점들이 야기되고 있다. 이는 PID 제어기로 해결할 수 없는 매우 어려운 문제이다. 따라서, 비선형 문제를 해결하기 위한 제어기들이 많이 연구되었는데, 그 중 하나가 시간지연제어기다[1-4].

시간지연제어기는 매우 간단한 구조의 강진한 제어기로 알려져 있다. 시스템을 나타내는 모델링이 정확하지 않을 경우에 이전 샘플 값을 사용하여 불확실한 요소를 없애주는 방식이다. 또한 실제 하드웨어 구현이 간편하고 실용적인 제어기이다. 이전 연구에서는 시간지연제어기를 구현하여 로봇 팔의 제어에 사용하였다[5].

선행 연구로 일반적으로 많이 사용하는 고전제어기인 PID 제어기와 비선형 PID 제어기를 FPGA로 구현하였다[6-8]. 또한 신경회로망 제어기를 PID 제어기와 함께 DSP로 구현하여 실험하였다[9].

FPGA는 다른 제어 시스템보다 최적화시킨 제어기를 설계할 수 있으며, 하나의 chip으로 모든 제어를 가능하게 하는 장점이 있다. 또한 다른 MCU에 비해 입출력 포트가 많아 그 사용에 제한이 없어 많은 모터를 동시에 제어할 수 있다는 것이 가장 큰 장점이다. 하지만 디지털 연산으로 인한 소수점

연산에는 어려움이 있고 또한 chip마다의 용량적 차이로 인해 원하는 연산을 위한 용량을 고려해야 하는 것이 단점이다.

본 논문에서는 시간지연제어기를 system-on-chip기반으로 구현하기 위해 FPGA에 제어기를 설계했을 경우에 나타나는 문제점들을 살펴보고자 한다. PID 제어기는 시스템의 모델을 알 수 없을 때 임의적인 P, I, D의 이득값으로 그 값을 추론하지만 시간지연제어기는 빠른 sampling time을 이용하여 현재 모델에서 이전 샘플의 동적 모델을 뺄으로써 모델링하지 못했던 불확실성 요소를 제거한다. 따라서, PD 제어기를 기준으로 로봇의 관성 모델링 값인 \hat{D} 을 실험을 통해 추론하여 설정하면 원하는 제어를 할 수 있게 된다. 하지만, 시간지연제어기가 잘 작동하기 위해서는 관성값과 더불어 정확한 가속도 정보가 필요하다.

가장 간단한 가속도 평가 방법은 두 번의 미분을 통해서 구할 수 있는 finite difference 방법이다. 이 방법은 미분 시에 잡음이 많이 들어가게 되어 정확한 가속도의 평가가 어려운 것이 사실이다. 또한 움직임이 느린 경우에는 속도를 평가하는 것이 불가능하다[9].

그러므로, 본 논문에서는 시간지연제어기에서 필요한 가속도의 값을 보다 정확하게 구하기 위해 다양한 필터를 사용하였다. Least square fit 필터와 Kalman 필터를 FPGA에 구현하여 실험하였다. 먼저 정수 기반으로 필터를 설계하고, 그 다음에 고정 소수점 기반으로 필터를 설계하여 비교해 보았다.

휴머노이드 로봇 팔에 부착된 일반적인 DC 모터의 위치제어를 제어하기 위해 시간지연제어기를 임베디드하여 성능을 실험하였다. 정확한 가속도값을 추측하기 위해 다양한 필터를 구현하였으며, 각 필터에 대한 제어기의 성능을 실험적으로 비교해 보았다.

II. 전체 시스템

1. 시간지연제어

* 책임저자(Corresponding Author)

논문접수 : 2008. 5. 29., 채택확정 : 2008. 10. 6.

전효원, 정슬 : 충남대학교 BK21 메카트로닉스 그룹

(atlantis44@naver.com/jungs@cnu.ac.kr)

※ 본 논문은 2006년도 충남대학교 학술연구비지원과 2006년도 학술진흥재단 지역대학우수과학자지원(D00166)에 의해 수행되었으며, 지원에 감사를 드립니다.

일반적인 로봇의 동역학 식은 다음과 같다.

$$D(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) = \tau \quad (1)$$

여기서, q 는 $n \times 1$ 벡터의 조인트, τ 는 $n \times 1$ 벡터의 토크, $D(q)$ 는 $n \times n$ 관성 행렬, $C(q, \dot{q})\dot{q}$ 은 $n \times 1$ 벡터의 Coriolis와 원심력, $G(q)$ 는 $n \times 1$ 벡터의 중력을 나타낸다. 제어 법칙은 다음과 같다.

$$\tau = \hat{D}u + \hat{C} + \hat{G} \quad (2)$$

식 (2)에서 사용하는 제어 입력 u 는 식 (3)과 같이 PD 형태의 제어를 사용하였다.

$$u \cong \ddot{q} = \ddot{q}_d + k_D \dot{e} + k_P e \quad (3)$$

위의 수식으로 식 (4)와 같은 최종적인 시간지연제어를 정리할 수 있다.

$$\tau = \hat{D}(t)u(t) + \tau(t-\lambda) - \hat{D}(t-\lambda)\ddot{q}(t-\lambda) \quad (4)$$

식 (4)에서 λ 는 매우 빠른 sampling time을 의미한다. 본 system에서의 sampling 주파수는 1024Hz이므로 시간지연제어를 사용하기에 충분히 빠른 sampling time을 얻을 수 있다.

식 (3)과 (4)를 합하면 다음과 같다.

$$\tau = \hat{D}(\ddot{q}_d + k_D \dot{e} + k_P e) + \tau(t-\lambda) - \hat{D}\ddot{q}(t-\lambda) \quad (5)$$

여기서 시간지연제어를 구현하기 위해서는 관성행렬 $\hat{D}(t-\lambda)$ 와 가속도 $\ddot{q}(t-\lambda)$ 를 정확하게 구해야 한다. 따라서, 본 논문에서는 관성행렬 $\hat{D}(t-\lambda)$ 를 실험적으로 구하고 가속도 $\ddot{q}(t-\lambda)$ 를 다양한 필터를 사용해서 구하고자 한다.

2. Least square fit filter

식 (5)의 가속도 $\ddot{q}(t-\lambda)$ 는 실제로 센서로부터 얻어지는 값이 아니므로 두 번의 미분을 통하여 간단하게 구할 수 있다. 하지만 인접한 값의 차이를 계산하기 때문에 잡음이 심하다는 단점이 있다. 먼저, 디지털 시스템에서 기본적으로 사용하는 1차 필터인 finite difference 보다 성능이 좋은 least square Fit 3차 필터를 설계하였다[9]. 식 (6)은 속도, 식 (7)은 가속도를 나타낸다.

$$w_k = \frac{11\theta_k - 18\theta_{k-1} + 3\theta_{k-2} - 2\theta_{k-3}}{6T_s} \quad (6)$$

$$\phi_k = \frac{2\theta_k - 5\theta_{k-1} + 4\theta_{k-2} - \theta_{k-3}}{2T_s^2} \quad (7)$$

그러나, 위의 필터도 샘플링 시간 T_s 로 나누어지기 때문에 불안정한 요소를 가진다. 만약 샘플링 시간이 매우 작아지면 상대적으로 분자에 큰 수가 곱해지기 때문에 사용에 제약이 있다.

3. 칼만 필터

칼만 필터는 상태 방정식에서 관측된 데이터와 현재의 상태값을 가지고 다음 상태를 최적으로 추측하는 필터이다. 센서의 측정잡음이나 시스템의 process noise 또한 고려하여 설계하므로 정확도가 매우 뛰어나다. 하지만, 현실적으로 잡음을 정확하게 모델링하기 어려우므로 단점이 될 수도 있다.

본 논문에서는 이러한 잡음을 직접 계산할 수가 없기 때문

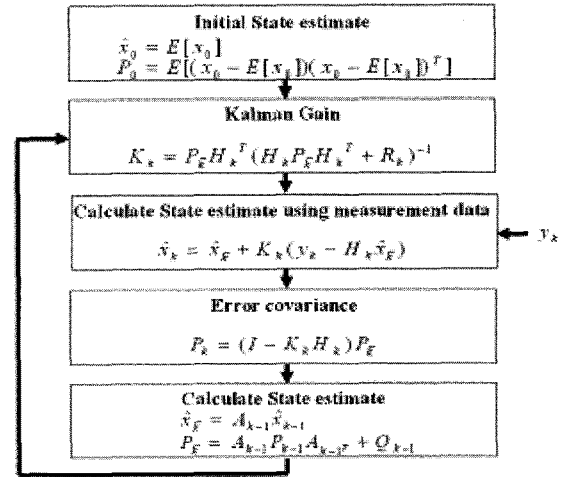


그림 1. 칼만 필터의 흐름도.
Fig. 1. Kalman filter flowchart.

에 실험을 통해 알맞은 수치를 정하여 사용하였다. 현재의 값으로 다음 상태의 속도를 구하기 위한 것이므로 상태방정식은 다음과 같다.

$$\begin{aligned} x_{k+1} &= \begin{bmatrix} 1 & T_s \\ 0 & 1 \end{bmatrix} x_k \\ y_k &= [1 \ 0] x_k \end{aligned} \quad (8)$$

구해진 상태방정식을 이용하여 그림 1의 순서대로 칼만 필터를 설계한다.

본 논문에서는 필요한 데이터가 속도와 가속도 값이기 때문에 칼만 필터를 두 번 거쳐 나온 값을 가속도 값으로 정하여 사용하였다.

4. FIR filter

일반적인 system에서 미분 값은 급격한 값의 변화에 의해 잡음이 매우 심하다. 본 논문에서는 이러한 잡음을 제거하기 위해 가장 기본적인 FIR low pass filter를 사용하였다. FIR filter는 IIR filter보다 사용이 간편하고 안정하다는 장점이 있지만 단순한 식으로 인해 그 성능이 떨어진다는 단점을 가지고 있다. 본 논문에서는 5차 filter를 사용하여 잡음을 제거하였다.

필터의 설계 시 통과주파수는 $w_c = 31.4 \text{ rad/s}$ 로 하였다. 따라서 다음 식과 같은 5차 필터를 설계할 수 있다.

$$\begin{aligned} y[n] &= 0.0284x[n] - 0.1427x[n-1] \\ &\quad + 0.3289x[n-2] + 0.3289x[n-3] \\ &\quad - 0.1427x[n-4] + 0.0284x[n-5] \end{aligned} \quad (9)$$

III. FPGA 설계 구현

1. 전체구조

그림 2는 FPGA에서 구현한 시스템 전체의 블록 다이어그램을 나타낸다. FPGA 칩에는 제어기 및 필터, 카운터, PWM 발생기, 그리고 직렬통신과 같은 보조적인 모듈들이 설계된다.

시간지연제어기에서는 모델링의 관성행렬 값인 \hat{D} 에 의해 성능이 좌우된다.

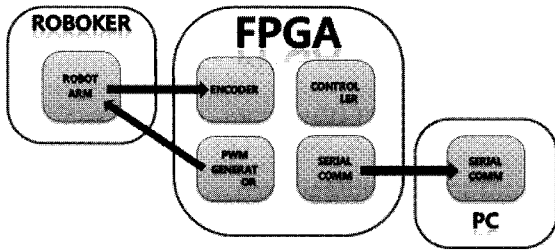


그림 2. 시간지연제어기의 시스템 구조.
Fig. 2 Overall system of time-delayed control.

2. 시간지연제어기 설계

그림 3의 시간지연제어 회로는 (5)의 제어 알고리즘을 설계한 부분이다.

각 부분을 정리하면 다음과 같다.

A: $u = \ddot{q}_d + k_d \dot{e} + k_p e$

B: $\dot{e}, \ddot{q}(t-\lambda)$, 를 구하는 필터

C: $\hat{D}u - \hat{D}\ddot{q}(t-\lambda)$

D: $\tau = \hat{D}u - \hat{D}\ddot{q}(t-\lambda) + \tau(t-\lambda)$

Desired angle과 실제 encoder에서 읽어오는 실제 angle에 의해 오차가 발생하게 되면 내부의 PD 제어기에 의해 오차를 줄여나간다. 하지만 정확한 system의 모델링을 할 수 없으므로 \hat{D} 을 임의의 수로 두고 이전 동적 모델로 실제 모델값을 제거함으로써 원하는 목표치에 도달할 수 있도록 제어한다. 오차는 13bit 이지만 필터의 사용으로 인해 최대 37bit 까지 그 용량이 확대 된다.

3. LSF 필터 설계

그림 4는 가속도를 추정해주는 LSF 3차 필터이다. 가속도

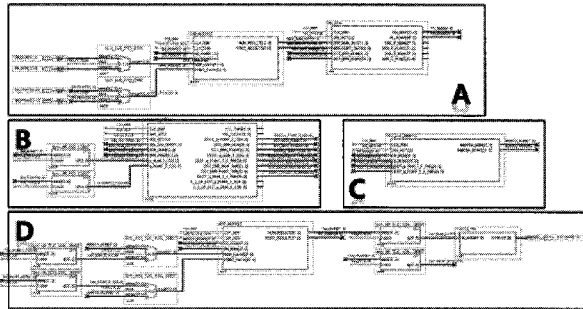


그림 3. 시간지연제어기 중 제어연산 부분.
Fig. 3. Design of a control part of time-delayed controller.

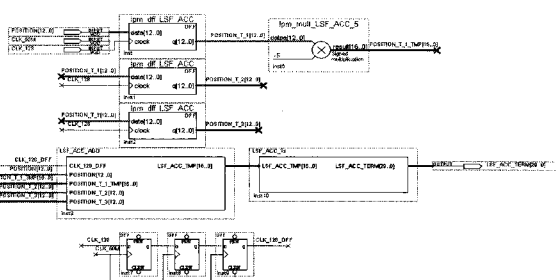


그림 4. LSF 필터설계.
Fig. 4. LSF filter design.

를 구하기 위해 입력되는 위치 값을 1샘플씩 지연시켜 사용한다. 이 값들에 각각 계수를 곱하여 마지막에 제어주기로 나누어준다. 제어주기가 매우 작은 경우에 상대적으로 큰 값이 분자에 곱해지므로 가속도 값에 의해 제어가 불안정해지기도 한다. 본 논문에서는 FIR 필터를 설계하여 LSF 필터와 함께 사용하였다.

사용하는 장치의 용량을 줄이기 위하여 곱하기, 나누기 연산보다는 우측, 좌측 shift를 많이 사용하였다.

4. 정수기반의 칼만 필터 설계

그림 5의 모듈들은 칼만 필터를 FPGA로 구현한 것이다. 사용 용량을 최적화시키기 위해 불필요한 계산은 제거하고 가속도를 추정하는데 필요한 연산만으로 아래와 같은 식을 구현하였다.

A: $\hat{x}_k(1,1) = \hat{x}_{k-1}(1,1) + Ts \cdot \hat{x}_{k-1}(2,1)$ (10)

$\hat{x}_k(2,1) = \hat{x}_{k-1}(2,1)$ (11)

B: $P_k(1,1) = P_{k-1}(1,1) + Ts \cdot \{P_{k-1}(1,2) + P_{k-1}(2,1)\} + Ts^2 P_{k-1}(2,2)$ (12)

$P_k(1,2) = P_{k-1}(1,2) + Ts \cdot P_{k-1}(2,2)$ (13)

$P_k(2,1) = P_{k-1}(2,1) + Ts \cdot P_{k-1}(2,2)$ (14)

$P_k(2,2) = P_{k-1}(2,2) + Q \times 2^{14}$ (15)

C: $Den = P_k(1,1) + R \times 2^{14}$ (16)

$\hat{x}_k(1,1) = \hat{x}_{k-1}(1,1) + \frac{(y_k - \hat{x}_k(1,1))}{Den} P_k(1,1)$ (17)

$\hat{x}_k(2,1) = \hat{x}_{k-1}(2,1) + \frac{(y_k - \hat{x}_k(1,1))}{Den} P_k(2,1)$ (18)

D: $P_k(1,1) = P_k(1,1) - \frac{P_k(1,1)^2}{Den}$ (19)

$P_k(1,2) = P_k(1,2) - \frac{P_k(1,1) \cdot P_k(1,2)}{Den}$ (20)

$P_k(2,1) = P_k(2,1) - \frac{P_k(1,1) \cdot P_k(2,1)}{Den}$ (21)

$P_k(2,2) = P_k(2,2) - \frac{P_k(1,2) \cdot P_k(2,1)}{Den}$ (22)

그림 5의 A block에서는 (10), (11)의 수식이 계산되는 부분으로 이전 추정값인 \hat{x}_{k-1} 를 이용하여 상태 \hat{x}_k 를 추정하는 역할을 수행한다. B block에서는 (12)-(15)의 연산을 하는 것

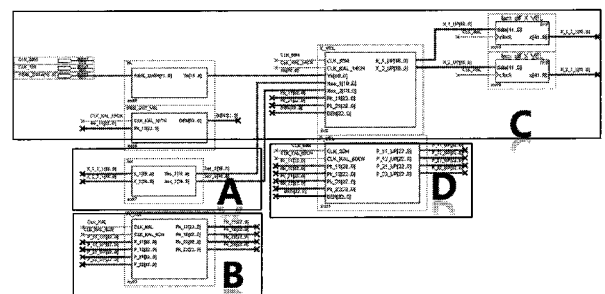


그림 5. 정수기반의 칼만 필터.
Fig. 5. Integer based design of Kalman filter.

로 이전 error covariance를 가지고 P_k 를 계산한다. C block에서는 실제로 측정된 위치 정보를 이용하여 상태를 추정하는 연산을 하는 모듈로 (16)-(18)의 수식이 적용되는 부분이다. $K_k = P_k H_k^T (H_k P_k H_k^T + R_k)^{-1}$ 의 칼만 이득값이 함께 연산된다. y_k 는 실제 data이고 앞에서 구한 \hat{x}_k 와의 오차를 0으로 만들기 위한 연산이 수행된다. (19)-(22)의 수식은 D block에서 연산되는 것으로 상태 추정 값과 error covariance를 업데이트 하는 역할을 수행한다. 오차가 0으로 수렴할 때까지 위의 수식에 의한 연산은 반복된다.

위의 (19), (20)의 수식에서 사용된 Q와 R값은 분산값으로 이 수치는 추정값에 큰 영향을 미친다. 실제적으로 두 분산값은 측정 잡음과 계산 잡음에 의해 계산되는 값이지만 잡음을 측정할 수 없으므로 실험을 통해 임의의 분산값을 찾아 이를 적용하여 사용하였다.

디지털 연산기반의 FPGA에서는 소수점의 사용이 어렵기 때문에 정밀도를 높이기 위해 칼만 필터의 연산 과정에 2^{14} 을 곱해주었다. 실제적으로 사용하는 값은 다시 2^{14} 이 나눠어진 값이므로 불필요한 영향을 끼치지 않는다. 2^{14} 을 곱하는 수로 설정한 이유는 14bit shift를 이용했기 때문이다. 수식 (18)의 $\hat{x}_k(2,1)$ 의 값이 칼만 필터의 출력 값인 속도 값이다. 이 값을 다시 모듈의 y_k 값으로 사용하여 칼만 필터를 총 두 번 거치면 가속도 값을 구할 수 있다.

5. 고정소수점기반의 칼만 필터

III 장 4절에서는 정수기반의 칼만 필터를 설계하였다. 하

지만 연산과정에서 소수점 이하의 수들은 모두 버림을 하기 때문에 정확한 계산이 불가능하여 정확하지 않은 제어 결과가 나온다. 따라서 이번 절에서는 고정 소수점 기반의 칼만 필터를 설계하였다. 그림 6의 A, B, C, D는 III 장 4절의 수식과 동일한 연산을 한다. 고정소수점 기반의 제어기에서는 기본적으로 정수부와 소수점 이하 셋째 자리까지의 소수점부로 나누어 연산한다. 따라서 칼만 필터로 미분하는 데이터도 총 4개의 데이터이다. 4개의 칼만 필터를 병렬로 계산할 경우 과도한 용량을 차지하기 때문에 하나의 필터로 4개의 데이터를 계산하는 시리얼 연산 방식으로 결과값을 구하였다. 제어주기는 128Hz이고, 수식의 계산 주기는 768Hz, 따라서 시리얼 연산 주기는 3072Hz이다.

정수부와 소수점부가 각각 칼만 필터를 통해 미분되고 나면 이를 하나의 수로 만들어주기 위해 소수점부를 1000으로 나누고 몫을 정수와 더해 하나의 수로 만드는 과정이 필요하다. 연산과정에서 소수점 이하의 값을 모두 가지기 때문에 정밀한 제어가 가능하다. 잡음이 작기 때문에 FIR 필터 또한 사용하지 않는다.

6. FIR low pass filter

미분을 통해 estimation을 한 값은 잡음이 매우 많다. 따라서 이 잡음을 제거하기 위해 FIR low pass 필터를 사용하였다.

FIR filter를 설계한 수식인 (13)의 식에 256을 곱하여 정수로 만든 후 결과값을 다시 256으로 나누는 방식을 이용하여 필터를 구현하였다. 여기에서 곱하는 수로 256을 선택한 이유는 위에서도 언급했듯이 곱셈과 나눗셈으로 인한 용량 차지를 줄이기 위해 shift를 이용했기 때문이다.

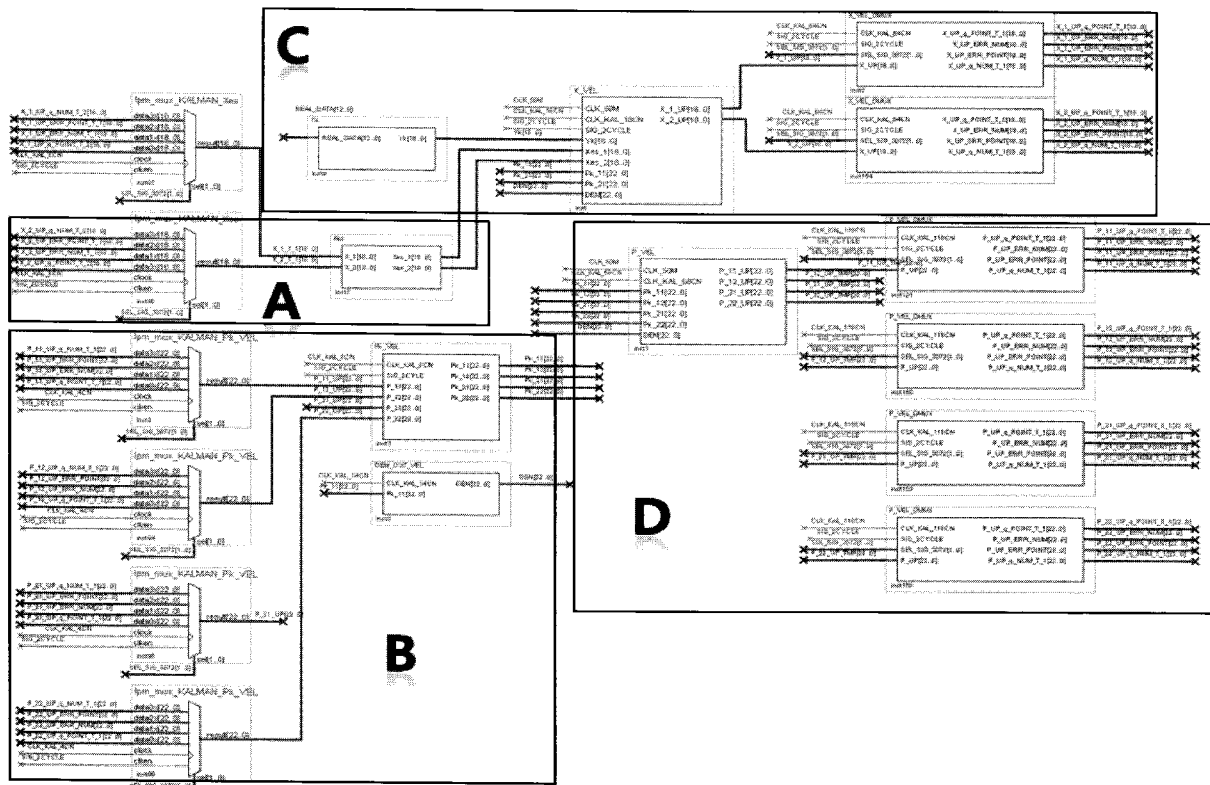


그림 6. 고정소수점 기반의 칼만 필터.
Fig. 6. Kalman filter based on fixed point format.

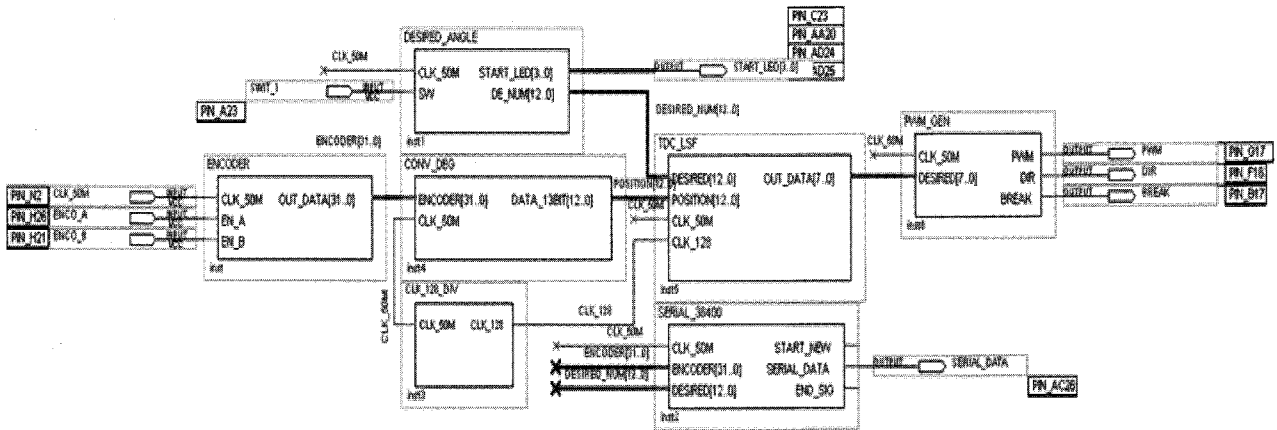


그림 7. 시간지연제어기 전체 모습.
Fig. 7. Overall design of a time delayed controller.

$$y[n] = 7x[n] - 37x[n-1] + 84x[n-2] + 84x[n-3] - 37x[n-4] + 7x[n-5] \quad (23)$$

정수가 된 (23)식을 이용하여 그림 8의 FIR 필터를 완성하였다. 정수기반의 시간지연제어기에서 위의 필터를 사용하지 않으면 불안정한 잡음때문에 제어가 되지 않음을 실험을 통해 확인하였다. 하지만 소수점 기반의 제어기에서는 잡음값이 현저하게 줄기 때문에 FIR 필터를 사용할 필요가 없다. 그림 7은 FPGA로 구현한 제어기의 전체 모습이다.

IV. 실험결과

그림 9는 실험에 사용한 휴머노이드 로봇, ROBOKER이다. 왼쪽 팔 세 번째 관절을 실험 대상으로 삼았다. 시간지연제

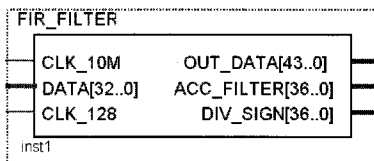


그림 8. FIR 저주파 필터.
Fig. 8. FIR low pass filter module.

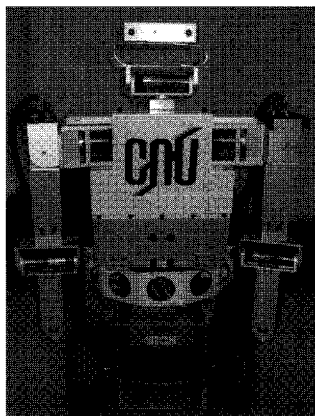


그림 9. 로보커.
Fig. 9. ROBOKER.

어에 사용한 기본적인 P, D 이득값은 각각 15, 1로 주었고 관성 모델 평가치는 $\hat{D} = \frac{1}{128} = 0.0078125$ 로 실험을 통해 구하였다.

본 논문에서 사용한 제어기 FPGA의 주 클럭은 50MHz이므로 이것을 main frequency로 사용하였고 제어주기는 128Hz로 분주하여 사용하였다. 또한 모터의 사양에 의해 PWM의 주기는 100kHz로 생성하였다.

소수점기반과 정수기반 제어기의 목표 데이터는 각각 (24), (25)이다.

$$y = -0.336x^2 + 6710.886x \quad (24)$$

$$y = -x^2 + 16000x \quad (25)$$

(24)은 이 수치들은 실제 값이 아닌 각각 25비트와 20비트로 좌측 쉬프트를 시킨 값으로 (24)은 라디안 단위기반, (25)는 각도단위기반이다. 따라서 결과 값의 범위는 각각 $[\frac{\pi}{3}, 1060]$ 이다. 그림 10은 제어기의 목표 경로 그래프이다.

칼만 필터를 사용함에 있어 그 성능에 중요한 영향을 주는 요소가 식 (15), (16)에서 사용한 Q와 R값인 분산이다. 이 값

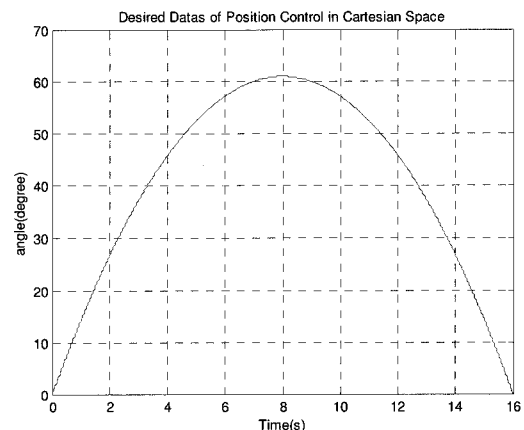


그림 10. 시간지연제어기의 목표값.
Fig. 10. Desired trajectory.

을 계산적으로 알 수 없으므로 실험을 통해 구하였다. 속도의 경우에는 $Q = \text{diag}(1,1)$, $R = \text{diag}(0.0001,0.0001)$ 그리고 가속도의 경우에는 $Q = \text{diag}(0.005,0.005)$, $R = \text{diag}(0.001,0.001)$ 의 값을 사용하였다.

그림 11과 그림 12의 상위 그래프는 점선이 desired data이고 실선이 실제 position data이다.

그림 11에 LSF 필터를 사용할 경우의 추종 결과가 나타나 있다. 추종 오차는 약 3도 정도이다. 그림 12에서 칼만 필터의 경우에도 비슷한 경로 추종 성능을 볼 수 있다. 디지털 연산 기반에서 가속도를 추정하는 필터를 설계하였기 때문에 연산 시 소수점 이하의 자리 수는 모두 버림을 한다. 따라서 정밀한 수치의 계산이 불가능하여 최대 오차가 ± 4 도 정도이다. 따라서 정밀한 제어를 위해서는 소수점 기반의 제어기 설계가 필요하다.

그림 13은 고정소수점 기반의 제어기를 사용한 결과이다. 고정소수점 기반이므로 각도 수치로 실수인 라디안값을 사용할 수 있다. 따라서 연산 시의 오차도 줄고 정밀한 제어를 가능하게 한다. 오차 범위는 ± 0.02 라디안으로 1도 미만이다.

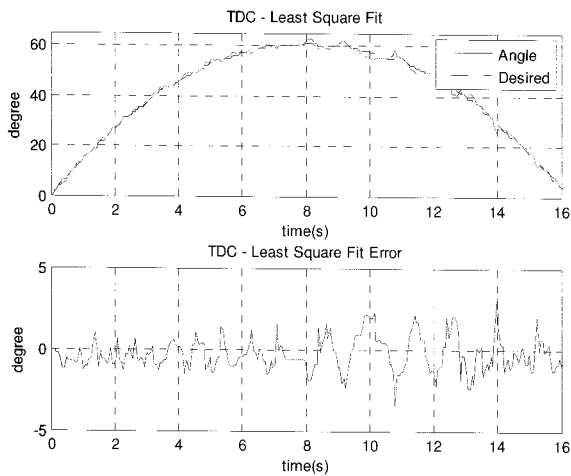


그림 11. LSF 필터를 사용한 제어기의 결과.
Fig. 11. Tracking result using LSF filter.

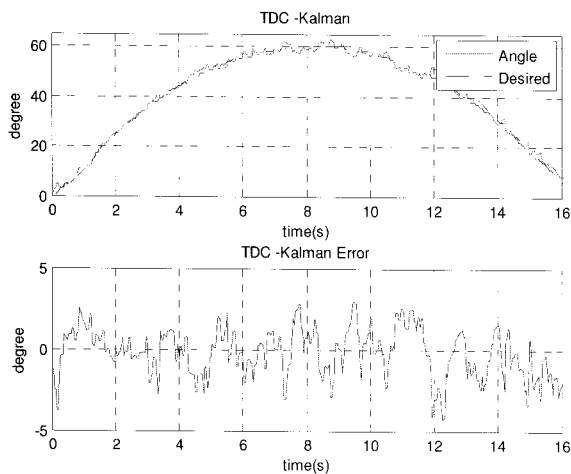


그림 12. 칼만 필터를 사용한 제어기의 결과.
Fig. 12. Tracking result using Kalman filter based on integer format.

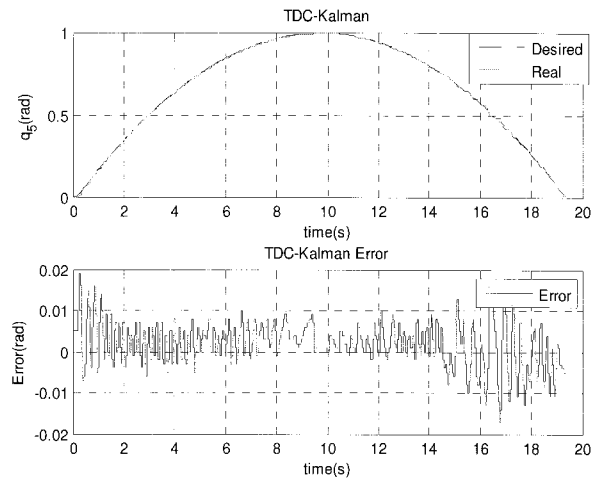


그림 13. 고정소수점 기반의 칼만 필터를 사용한 제어기의 결과.
Fig. 13. Tracking result using Kalman filter based on fixed point format.

V. 결론

본 논문에서는 FPGA를 사용하여 시간지연제어기를 설계하고 로봇 팔을 제어하였다. 시간지연제어 방식에서 필요한 미분값 $\ddot{q}(t-\lambda)$ 의 측정 문제를 해결하기 위해 LSF 필터와 칼만 필터를 이용하여 추정하였다. 디지털연산기반의 제어기로서 정밀한 제어가 불가능하여 계산상의 오차가 많이 발생하는 것을 해결하기 위해 고정소수점 기반의 제어기를 설계하였다. 소수점 이하 셋째 자리 수까지 사용하여 연산 시 오차를 줄였다. 정밀한 제어가 가능하다는 것을 실험을 통해 확인할 수 있다.

참고문헌

- [1] T. C. Hsia, L.S. Gao, "Robot Manipulator Control Using Decentralized Linear Time-Invariant Time-Delayed Joint Controllers," *IEEE conf. on Robotics and Automation*, pp. 2070-2075, 1990.
- [2] K. Yousef-Toumi and O. Ito, "A time delay controller for systems with unknown dynamics," *Journal of Dynamic Systems, Measurement, and Control*," vol. 112, pp. 133-142, 1990.
- [3] K. Yousef-Toumi, "Stability of Uncertain Linear System With Time Delay," *Trans. of ASME*, vol. 113, pp. 558-567, Dec. 1991.
- [4] K. Yousef-Toumi, "Analysis of Linear Time Invariant System With Time Delay," *Trans. of ASME*, vol. 114, pp. 544-555, Dec. 1992.
- [5] S. Jung, T. C. Hsia, and R. G. Bonitz, "Force Tracking Impedance control for robot manipulators with an unknown environment: Theory, Simulation, and Experiment," *Journal of Robotics Research*, vol. 29, no. 9, pp. 765-774, 2001.
- [6] 김성수, 정술, "FPGA를 이용한 범용 모션 컨트롤러 개발," 제어·자동화·시스템공학 논문지, 제10권 제1호, pp. 73-80, 2004.
- [7] 김성수, 정술, "DSP와 FPGA를 이용한 지능 제어기의 하드웨어 구현," 제어·자동화·시스템공학 논문지, 제10권 제10호, pp. 922-929, 2004.

- [8] 김정섭, 전효원, 정슬, “ROBOKER팔의 제어를 위한 FPGA기반 비선형 제어기의 임베디드 하드웨어 구현,” 제어 · 자동화 · 시스템공학 논문지, 제13권 제12호, pp. 1153-1159, 2007.



전 효 원

1984년 1월 8일생. 2006년 충남대학교 메카트로닉스공학과 졸업. 2008년 충남대학교 메카트로닉스공학과 석사졸업. 현재 LIG 넥스원연구소 근무. 관심분야는 임베디드 제어 시스템 설계, 신호 처리 및 필터설계, 인간과 로봇의 상호

작용을 위한 협제어 응용공학.

- [9] 조현택, 정슬, “속도 추정 기반의 2 자유도 도립진자의 안정화를 위한 입력 보상 방식의 분산신경망 제어기에 관한 실험적 연구,” 제어 · 자동화 · 시스템공학 논문지, 제10권 제4호, pp. 342-350, 2004.



정 슬

1964년 9월 11일생. 1988년 미국 웨인주립대 전기 및 컴퓨터 공학과 졸업. 1991년 미국 캘리포니아대 데이비스 전기 및 컴퓨터 공학과 석사. 동 대학 박사. 1997년~현재 충남대학교 메카트로닉스공학과 교수. 관심분야는 지능 제어

및 지능 로봇 시스템, 디지털 필터 시스템, 임베디드 제어기 설계, 로봇과 인간의 상호작용.