'93 한국자동제어학술회의논문집 (1993.10.20~22)

서보모터의 고정도 속도검출을 위한 M/T 방식의 하드웨어 구현

°채상락*, 박정일*, 이석규**
*영남대학교 공과대학 전자공학과, **영남대학교 공과대학 전기공학과

Hardware Implementation of the M/T Method for High Accuracy Speed Detection of a Servo Motor

°Sangrak Chae*, Jungil Park*, Suckgyu Lee**
*Department of Electronic Engineering, **Department of Electrical Engineering, Yeungnam University

ABSTRACT

In this paper, the new M/T method for motor speed detection is proposed. This method can be able to reduce the dead time compared with it when Ohmae's M/T method is implemented. And the comparison of the dead time length between the Ohmae's method and the proposed method is analyzed quantitatively. Actually we implemented the new proposed M/T method using the hardware and software and verified the effectiveness of the proposed M/T method.

1. 서 론

현재 산업기기에서 가장 많이 이용되고 있는 서보모터의 속도검출기로는 인크리멘탈 엔코더와 같은 디지탈식 검출기가 많이 이용되고 있다. 근래에 와서 서보모터의 저속제어에 관심을 갖게 되었고 엔코더를 이용하여 속도를 검출하는 경우에는 저속을 검출하기 위해서는 여러가지 애로점이 따른다. 그래서 저속을 검출하기 위해서 속도 관측기를 이용하거나[1,5] 저속에서 속도를 검출할 때 Dead Time이 길어지는 문제점 때문에 순시속도 검출에도 관심을 갖게 되었다.[2,3]

일반적으로 엔코더를 사용하여 속도를 검출하는 방법으로는 고속과 저속에서 각각 다른 방법이 사용된다. 고속에서는 속도가 빨라서 단위 시간당 카운트되는 엔코더의 펄스 갯수가 많기 때문에 엔코더의 펄스수를 일정시각마다 헤아려서 속도를 검출하는 M 방식이 이용된다. 또 저속에서는 속도가 느려서 단위시간당 카운트할 수 있는 엔코더의 펄스 갯수가 적기 때문에 검출 분해능이 떨어지게 된다. 그래서 매우 높은 주파수를 이용하여 엔코더의 펄스 간격을 측정하여 속도를 검출하는 T 방식이 이용된다.

1982년 Ohmae등이 이 두가지 방법을 결합하여 고속이나 저속에서 동시에 사용할 수 있는 M/T 방식을 제안하였다.[1] 그러나 이 논문에서 제안한 M/T 방식은 하드웨어로 구현함에 있어서 엔코더의 펄스와 샘플링 시각을 동기화 시켜야 하기 때문에 초저속에서는 샘플링 구간이 가변되는 현상이 발생하게 된다. 또 초저속일수록 속도를 검출해서 제어입력을 인가하는 Dead Time이 길어 지게 된다. 그래서 본 논문에서는 샘플링 시각과 검출구간이 동기화될 필요성이 없으면서 Dead Time을 줄일 수 있는 M/T 방식을 구현하였다. 또한 기존의 M/T 방식과 새로 구현한 M/T방식의 Dead Time을 정량적으로 비교하였으며, 새로 제안한 M/T 방식을 실제 하드웨어로 제작하여 PC와 인터페이스하여 속도검출을 수행하였다. 이 방식은 제어해야 하는 속도의 영역에 무관하게 정확한 속도를 검출할 수 있어서 향후 서보 드리이브의 성능 향상에 도움이 되리라고 본다.

2. 속도 검출

2.1 기존의 속도검출 방식의 원리 [1]

그림1은 Ohmae의 M/T방식의 속도 검출원리를 설명한 그림이다.

검출시간(T_d)과 샘플링시간(T_s)의 시작점과 엔코더 펄스는 동기가 되어 있다. 윗 그림은 엔코더 펄스를 엔코더 펄스의 상승에지에서 시작하여 짧은 펄스폭을 갖도록 파형정형한 그림이고 아랫쪽 그림은 엔코더 펄스폭을 측정하는데 이용되는 주파수가 높은 클럭 파형이다. 그림1에서 측정된 회전속도(rpm)는 다음과 같이 나타낼 수 있다.

$$N_r(rpm) = \frac{60 f_c m_1}{P m_2} \quad (1)$$

여기에서 P는 엔코더의 한바퀴당 발생되는 펄스수이고, f_c는 클럭펄스의 주파수이며, m_1과 m_2는 각각 검출시간(T_d)동안 카운트된 엔코더 펄스수와 클럭 펄스수이다.

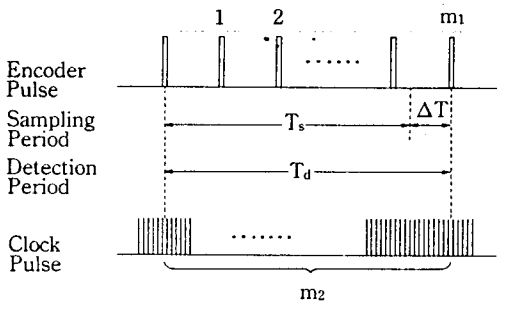

Fig. 1. Speed detection principle of the Ohmae's M/T method.

그림2는 기존의 Ohmae가 M/T방식을 구현한 하드웨어 블록도이다. 처음에 마이크로프로세서가 'Start Flip- Flop'에 검출시작 신호를 보낸다. 그러면 엔코더의 펄스에 동기를 맞추어서 샘플링 T_s 타이머가 스타트된다. 이때 높은 클럭 펄스를 헤아리는 T 카운터와 엔코더 펄스를 헤아리는 M 카운터가 동작을 시작한다. 이 후, T_s 타이머의 규정된 시간 이후에 처음으로 엔코더의 펄스가 발생하는 시점에서 'Stop Flip-Flop'이 동작하여 마이크로프로세서에 인터럽트를 인가함과 동시에 T 카운터와 M 카운터를 정지시킨다. 이 다음 마이크로프로세서가 T 카운터와 M 카운터를 읽어서 속도를 계산한 후 다음 시퀀스를 반복한다. 이렇게 속도를 검출하는 과정이 그림3의 순서도이다.

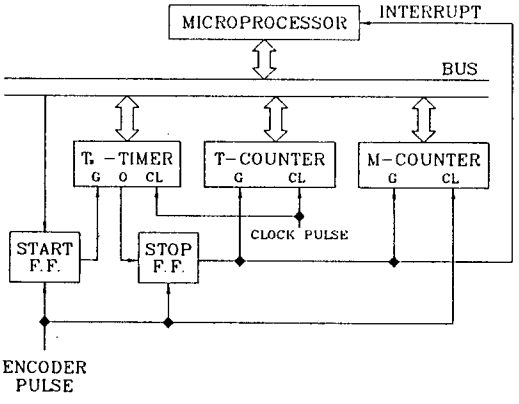

Fig. 2. Speed measuring circuit of the Ohmae's M/T method.

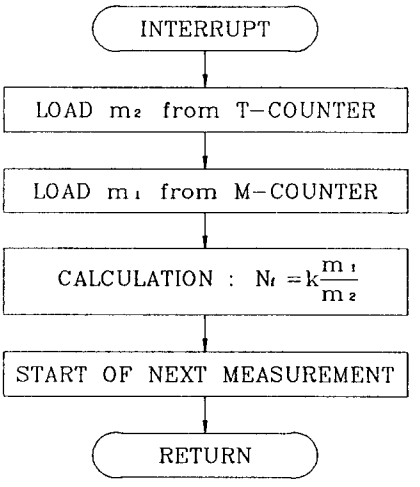

Fig. 3. Speed detection flowchart of the Ohmae's M/T method.

2.2 제안 속도 검출 방법 원리

이 절에서는 같은 精度를 가지면서 Dead Time을 줄일 수 있는 M/T 방식을 설계하기로 한다.

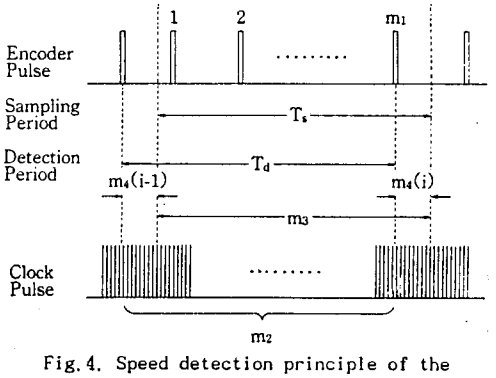

Fig. 4. Speed detection principle of the proposed M/T method.

그림4를 살펴보면 엔코더 펄스와 샘플링 시점을 동기시키지 않고 그 사이의 폭을 측정함으로써 Ohmae의 방법과 같은 精度로 속도를 검출할 수 있다. 그림4에서 측정된 회전속도(rpm)도 (1)식과 같이 쓸 수 있다. 단, 여기에서 m_2 는 아래의 식과 같이 계산된다.

$$m_2 = m_3 - m_4(i) + m_4(i-1) \quad (2)$$

제안한 속도검출회로를 구현하기 위하여 그림5와 같은 하드웨어를 구성할 수 있다. 카운터1은 높은 입력 주파수를 분주하여 원하는 일정시각 마다 마이크로 프로세서에 인트럽트를 걸어 샘플링 시점을 발생시키기 위한 카운터

이다. 카운터2는 그림3에서의 $m_4(i)$와 $m_4(i-1)$의 폭을 측정하기 위한 것이다. 엔코더 신호와 인터럽트 신호의 상승에지에서 짧은 기간의 펄스를 발생시켜 이 신호로 카운터2의 값을 래치에 래치 시킴과 동시에 클리어 시킨다. 이렇게 카운트된 카운터 값을 읽어 들이기 위해서 CPU는 인터럽트가 걸리면 첫 순간에 래치의 값을 읽어들이면 샘플링 이전의 엔코더 펄스와 샘플링 펄스 사이의 폭이 측정되게 된다. 카운터3는 샘플링 구간 동안의 엔코더 펄스수를 카운트하기 위한 m_1 카운터이다.

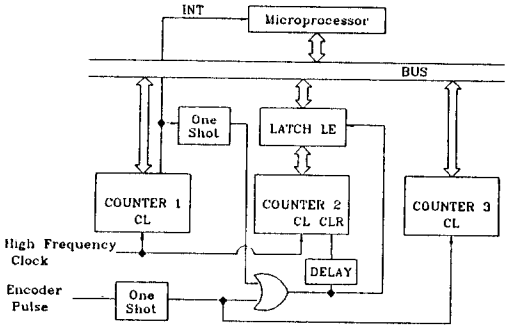

Fig. 5. Proposed speed measuring circuit.

그림6은 측정한 값을 읽어서 속도를 계산하는 과정을 순서도로 나타낸 것이다. 이 순서도에서 j는 그림8에서 볼 수 있는 바와 같이 모터의 속도가 초저속이어서 한 샘플링 구간내에 엔코더 펄스가 발생하지 않는 경우를 고려하기 위하여 도입된 것이다. 원리는 엔코더 펄스가 발생할 때 까지 샘플링 구간이 몇구간 인가를 세어서 속도를 판단하는 것이다. 이것은 결국 T 방식과 동가적으로 된다.

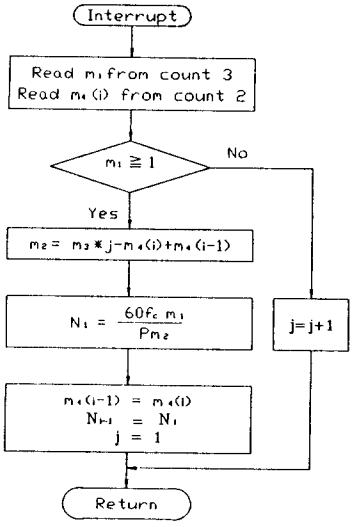

Fig. 6. Speed measurement flowchart of the proposed M/T method.

3. 기존 방식과 제안 속도검출방식의 비교

그림7과 그림8에서 Ohmae가 구현한 기존의 방법과 본 논문에서 제안한 방식을 비교하였다. 그림7의 경우는 저속이며 그림8의 경우는 하나의 샘플링 구간내에 엔코더 펄스가 발생하지 않는 초저속의 경우를 나타낸 것이다. 그림7을 살펴보면 본 논문에서 제안한 방식이 짧은 시간내에 속도를 검출해서 샘플링 구간마다 새로운 제어입력을 인가할 수가 있다. 이에 비해서 아래의 그림은 검출구간이 길고 매번 마다 하나의 엔코더 펄스폭은 측정할 수가 없도록 되어 있으며 외부 변화에 대해서 빠르게 제어입력을 인가할 수 없는 구조로 되어 있다. 즉, 속도를 검출해서 이에 대응한 제어입력을 인가하는 구간이 길어서 Dead Time이 길어지는 효과로 나타난다. 특히 그림8과 같은 초저속에서는 이러한 현상이 더 심해져서 Dead Time이 더 길어져서 속응성 있게 제어를 수행할 수 없게 된다. 표1은 이 Dead Time의 길이를 속도에 따라서 정량적으로 비교한 것이다.

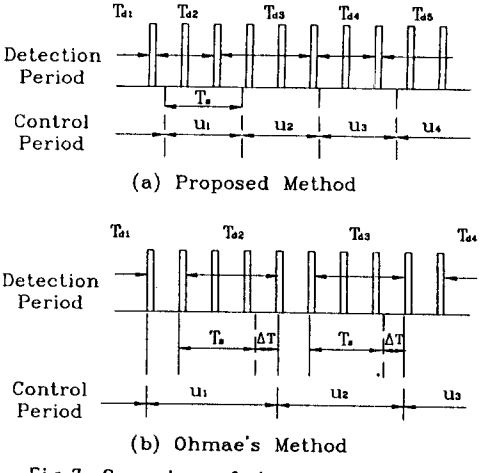

Fig. 7. Comparison of the timing sequences at low speed.

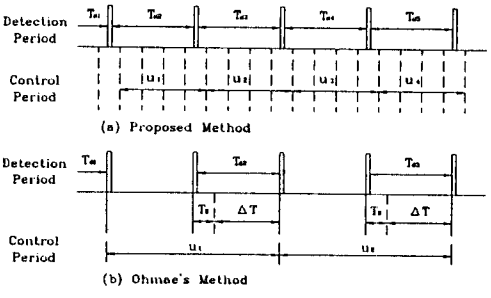

Fig. 8. Comparison of the timing sequences at very low speed.

Table1. Comparison of the dead time

	Low Speed	Very Low Speed
Proposed	T_s+1T_e $(T_s(1+1/m_1))$	T_s+1T_e
Ohmae's Method	$T_s+\Delta T+1T_e$ $(T_s(1+2/m_1))$	$2T_e$

* A parenthesized part is the dead time at worst case.
* $1T_e$: Encoder 1 pulse period
* $2T_e$: Encoder 2 pulse period

그림9는 표1에서 최악의 경우의 Dead Time을 그림으로 나타낸 것이다. 그림에서 볼 수 있듯이 샘플링 구간내의 엔코더 펄스수가 적은 초저속일 수록 본 논문에서 제안한 방식의 Dead Time이 짧아지는 효과가 크게 나타남을 알 수 있다.

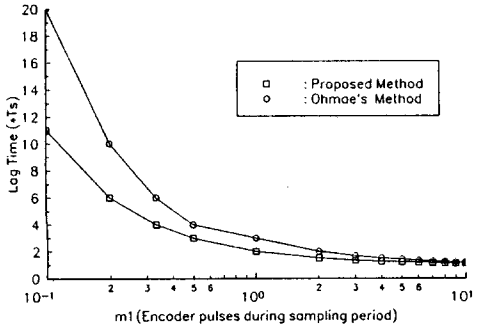

Fig. 9. Comparison graph of the dead time at worst case.

4. 실험 및 고찰

실험을 하기 위한 하드웨어는 그림5의 블록도대로 TTL 소자와 LSI소자(8253, 8255)를 이용하여 구성하였으며, IBM-PC/AT의 slot에 장착할 수 있도록 카운터들과 래치, 8255등을 PC에서 이용하지않고 있는 I/O 번지로 할당하였다. 카운터1과 카운터3은 8253을 이용하였으며 카운터2는 TTL을 사용하여 16비트의 동기카운터로 구성하였다.

속도를 읽어들이는 모든 알고리즘은 C 언어를 사용하여 작성하였으며 compile한 수행파일을 수행시켜 속도를 검출하였다. 속도를 검출하는 샘플링 시간은 10 msec로 하였는데 이것은 카운터1(8253)로 들어오는 PC의 1 MHz의 클럭 주파수를 분주하여 10 msec마다 인터럽트를 걸어 샘플링을 하였다. 엔코더의 펄스간격을 측정하기 위한 카운터2로 들어가는 높은 주파수도 1 MHz의 PC 클럭을 이용하였다. 모터는 ELECTRO-CRAFT 회사의 DC 서보 모터를 이용하였는데, 엔코더의 1회전당 펄스수가 800개 이다.

그림10과 그림11은 Ohmae의 M/T 방식을 실제 하드웨어로 구현하는 것은 번거로와서, 샘플링 주기동안 엔코더 펄스 만을 카운트하는 M 방식과 본 논문에서 제안한 M/T 방식을 이용하여 각각 고속과 저속에서 속도를 검출한 결과이다. 예상한대로 M 방식에서는 엔코더의 펄스가 800개, 샘플링 구간이 10 msec인 경우는 분해능이 7.5 rpm이므로 속도를 검출하는데는 오차가 큼을 알 수 있다. 그러나 M/T방식은 고속이나 저속에 관계없이 아주 정확한 속도를 검출하는 것을 알 수 있다. 그림12는 한 샘플링 구간동안에 엔코더 펄스가 카운트 되지 않는 초저속의 경우에 속도를 검출한 결과이다. 분해능이 7.5 rpm 이지만 본 논문에서 제안한 그림6의 프로그램을 이용하면 7.5rpm이하의 속도도 검출할 수 있음을 알 수 있다. 이것은 엔코더 펄스가 발생할 때까지 샘플링 구간을 연장시키는 효과 때문이다. 여기의 모든 그림은 속도검출 값을 실시간으로 PC의 RAM에 저장한 다음 플로팅한 것이다.

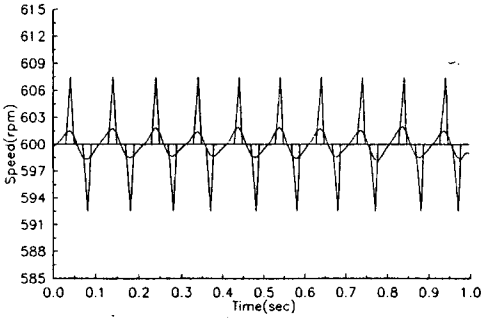

Fig. 10. Speed detection using the M method and the proposed M/T method at high speed.

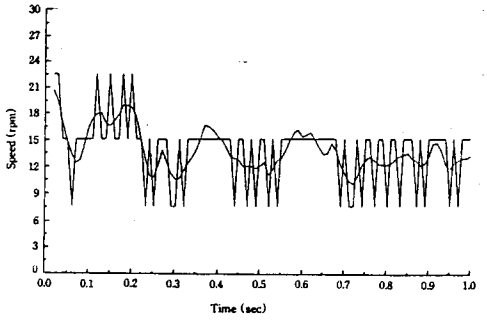

Fig. 11. Speed detection using the M method and the proposed M/T method at low speed.

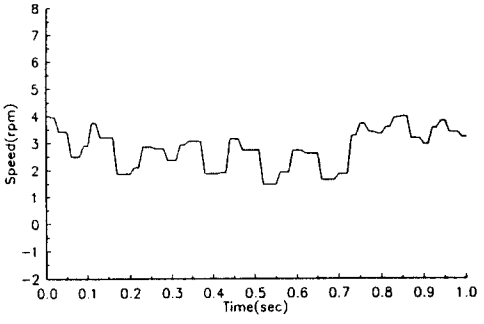

Fig.12. Speed detection using the proposed M/T method at very low speed.

5. 결 론

본 논문에서는 엔코더를 이용하여 모터의 속도를 검출함에 있어서 기존 Ohmae의 M/T 방식보다 Dead Time을 줄일 수 있는 M/T 방식을 제시하였으며, 제시한 방식과 기존의 M/T 방식의 Dead Time의 길이를 정량적으로 비교하였다. 본 논문에서 제시한 방식이 기존의 방식에 비해서 Dead Time이 훨씬 짧음을 보였다.

또한 실제로 제안한 M/T 방식의 속도 검출 하드웨어를 제작하여 PC의 Slot에 장착한 다음 고속과 저속, 초저속에서 속도검출을 수행하여 잘 동작함을 보였다.

본 논문에서 제시한 하드웨어는 간단하여 마이크로 프로세서 레벨에서 구현 가능하기 때문에 향후 서보제어계의 성능향상에 도움이 되리라고 본다.

참 고 문 헌

[1] T.Ohmae et al.,"A Microprocessor-Controlled High Accuracy Wide-Range Speed Regulator for Motor Drives," *IEEE Trans. Ind. Elec.* vol.IE-29, no.3, pp.207-211, Aug.1982.

[2] K. Saito et al., "A Microprocessor-Controlled Speed Regulator with Instantaneous Speed Estimation for Motor Drives," *IEEE Trans. Ind. Elec.*, vol.IE-35, no.1, pp.95-99, 1988.

[3] K. Fujita et al.,"Instantaneous Speed Detection with Parameter Identification for a Servo Systems," *IEEE Trans. Ind. Appl.*, vol.IA-28, no.4, pp.864-872, July/August 1992.

[4] 土手康彦, *ロバスト 高速 サーボ 制御技術*, トリケップス 出版部,PP.251-266, 1991.

[5] 渡邊,"速度推定 オブザーバを 用いた ディジタル サーボ," 電氣學會 論文誌 D, vol.107, no.12, pp. 1468-1474, 1987.