Proceedings of KFIS Fall Conference, 2003. 5. 3

정수형 퍼지제어기법을 적용한 실시간 고속 퍼지제어 시스템

A Real-time High-speed Fuzzy Control System Using Integer Fuzzy Control Method

손 기성*, 김 종혁*, 성 은무**, 이 상구*
Ki Sung Son, Jong Hyuk Kim, Oun Moo Sung, Sang Gu Lee
한남대학교 컴퓨터공학과*
공주영상정보대학**
Dept. of Computer Engineering, Hannam University*
Kongju Communication Arts College**
E-mail : romeo@para.hannam.ac.kr

요 약

대용량의 퍼지데이터를 처리하기위한 퍼지제어 시스템의 가장 큰 과제는 퍼지추론 및 비퍼지화 단계에서의 수행속도의 개선이다. 본 논문에서는 퍼지제어기의 속도 향상을 위해 [0, 1]사이의 실수값을 갖는 퍼지 소속함수값을 정수형 격자(pixel)에 매핑시켜 정수형 퍼지 소속함수값만을 가지고 퍼지연산을 하는 정수형 퍼지제어기법을 적용한 고속의 정수 연산을 수행하는 퍼지 프로세서와 주변제어 시스템을 FPGA로 설계하여 기존의 퍼지제어 시스템에 비해 매우 빠른 실시간 고속퍼지 제어시스템을 구현한다.

Abstract

In fuzzy control systems having large volumes of fuzzy data, one of the important problems is the improvement of execution speed in the fuzzy inference and defuzzification stages. In this paper, to improve the speedup of fuzzy controllers, we use an integer line mapping algorithm to convert [0, 1] real values in the fuzzy membership functions to integer pixels. Using this, we propose a real-time high-speed fuzzy control system and implement a fast fuzzy processor and control system using FPGAs.

Keywords : Fuzzy Arithmetic, Integer Operation, High-speed Fuzzy Controller

1. 서 론

퍼지 이론은 과거에는 퍼지제어를 중심으로 공학 분야에 국한되어져 왔으나 현재에는 의료진단이나 데이터 마이닝, 전문가시스템 등 새로운 응용분야에서도 퍼지 이론의 효용성을 보이고 있으며 퍼지 이론을 기반으로 설계되는 퍼지 시스템은 인간의 언어적 개념을 정량적 수치로 표시할 수 있다는 장점 때문에 지능.시스템 및 소프트 컴퓨팅 등의 공학적 기술로서 널리 응용되고 있다[1].

퍼지 논리는 기존의 부울 논리를 확장한 것으로 특정 집합 A에 대하여 구성 원소의 소속 정도를 0 또는 1의 이진정보가 아닌 0과 1사이의 실수로 나타낸다. 이러한 퍼지값은 퍼지 추론과 비퍼지화를 거쳐 시스템에서 요구하는 제어값이 된다. 퍼지 추론은 외부에서 입력되는 조건부의 퍼지 정보에 대해 각각의 소속함수를 통해 소속정도를 구하고 이 소속정도들에 퍼지제어 규칙을 적용하여 적합도를 구한다. 개개의 제어규칙에서 얻어진 추론결과들에 대한 비퍼지화를 통해 제어값을 구하게 된다. 입력되는 퍼지 정보의 소속 정도는 정수값이 아닌 0과 1사이의 실수로 표현되기 때문에 이러한 추론 과정을 거치는 동안 퍼지 제어기는 많은 양의 실수연산을 필요로 한다. 소속 함수나 퍼지 제어규칙의 수가 많아질수록

연산량은 더욱 증가하고 그만큼 많은 실수 연산을 하게 된다.

일반적인 산술연산에서 정수와 실수의 연산 속도에는 많은 차이가 있다. CPU 속도에 따라 조금씩 영향을 받지만 정수 연산이 실수 연산에 비해 곱셈의 경우 10배 이상, 나눗셈의 경우 4배정도 빠른 계산 속도를 갖는다. 계산이 복잡해지고 계산량이 많아질수록 계산 속도의 차이는 더욱 커지게 된다.

따라서 본 논문에서는 실수 연산으로 인한 퍼지 추론 속도 저하 문제를 해결하기 위해 퍼지 소속함수 그래프를 정수형 격자에 매핑시켜 실수를 각각에 대한 정수로 변환 후 퍼지 추론을 하여 연산 속도를 향상시키는 정수형 퍼지제어 기법을 적용하여 실제 퍼지 제어 시스템의 정수형 퍼지연산을 하기위한 프로세서를 설계하여 시스템의 성능을 평가한다.

2. 정수형 퍼지제어 알고리즘

2.1 정수형 퍼지제어 기법의 필요성

일반적으로 퍼지제어 시스템에서 소속함수의 모양은 삼각형 또는 사다리꼴의 형태를 갖고 있다. 이러한 소속함수의 형태를 퍼지제어 시스템에 저장시킬 때는 소속함수의 많은 실수 데이터를 모두 저장할 수 없기 때문에 삼각형 또는 사다리꼴의 정점만을 저장하여 내부에서 기울기를 계산하여 실수의 적당한 간격으로 소속함수의 값을 실수 연산하여 퍼지제어 시스템에 적용한다. x축은 퍼지변수의 값이고, y축은 [0, 1]의 소속도이다. 이때 x축이 256개 y축이 32개를 갖는 정수형 격자를 생각하면, 퍼지 소속함수를 이러한 정수형 격자에 근사화하여 매핑 시킬 수 있다. 이러한 정수형 격자에서는 기울기를 구하여 각각의 x축에 대응되는 y축의 정수형 격자를 계산할 때 곱셈, 나눗셈이 필요 없이 단순한 정수의 덧셈 연산만으로 고속으로 연산할 수 있으며 비퍼지화 단계에서도 실수연산이 아닌 정수연산으로 종래의 실수형 연산에 비하여 상당히 빠른 연산결과를 얻을 수 있다.

2.2 퍼지제어 알고리즘

정수형 변환 알고리즘을 적용한 퍼지시스템을 표 1에서 알고리즘으로 나타내 보았다. 퍼지 추론단계는 Mamdani의 최소·최대 연산법(Min·Max method)을 이용하여 수행하도록 하고 추론된 결과에 대한 비퍼지화는 무게 중심법(center of area method)을 이용해 해당 결과값을 계산하도록 하였다. 먼저 전건부를 보면 r은 rule의 개수 이며 m은 전건부의 소속함수의 개수이다. C가 소속함수값을 갖는 집합이고 A가 입력변수들의 집합이라 할 때 MAX값을 구해서 d 집합에 넣고 전건부 소속함수로부터 나온 소속정도값을 D 집합이라 한다. 결국 D 집합의 값은 후건부의 입력값이 된다. n값은 후건부 소속함수의 개수이며 P 집합은 후건부의 소속정도값으로 D 집합과 MIN연산을 통하여 얻어진다. 이렇게 얻어진 D 집합은 정수형 변환알고리즘을 거쳐 최적화된 정수형태의 값을 배열형태로 갖게 된다. D 집합은 x축의 개수만큼의 배열 256개로 각각의 값들은 0부터 31사이의 정수값을 갖게 된다. 256개의 배열은 후건부 소속정도값들의 MAX 연산을 거치면서 계속적으로 갱신되며 최종적으로 이 값들은 B집합이 된다. B집합의 256개 배열값은 비퍼지화 과정에서 정수형 연산을 할때 필요하게 된다. 본 연구에서는 FPGA설계의 용이성과 정수형 알고리즘의 최적화를 고려하여 256개의 배열에 후건부 소속정도값을 저장하도록 하였다. 비퍼지화 단계는 B집합을 무게중심법을 이용하여 최종 결과값을 얻을 수 있다.

표 1. 정수형 연산 퍼지제어 알고리즘

```
퍼지추론단계
For       i = 1  to  r
{
  For     j = 1  to  m
  {
          dj = MAX( Cij(x) , Aj(x) )
          Di = MIN( Di , dj )
  }
  For     k = 1  to  n
  {
          Pik(y) = MIN( Di , Sik(y) )
  }
  Inti(y) = IntFunction( Pi )
}

For       i = 1  to  r
{
  For     k = 1  to  n
  {
          Bk(y) = MAX( Bk(y) , Inti(y) )
  }
}

비퍼지화
For k = 1   to  256
  {
          x += ( k · Intk )
          y += Intk
  }
b = x / y
```

2.3 정수형 변환 알고리즘

퍼지 소속함수의 그래프가 삼각형 또는 사다리꼴 형태를 하는 경우 이 함수는 직선들의 연속임을 알 수 있다. 전건부 소속함수의 그래프는 매우 단순한 형태의 직선들의 연속이라고 할 수 있지만 추론결과에 대한 그래프를 보면 매우 복잡한 형태의 직선들의 연속이 된다. 퍼지 추론시 전건부의 삼각 함수나 사다리꼴 함수 형태의 소속함수에서 임의의 x에 대한 y값을 얻는 것은 그리 많은 양의 계산이 필요하지 않다. 하지만 후건부를 거치면서 소속정도값들을 그래프로 보면 좀더 복잡한 다각형 형태를 띠며 임의의 하나의 좌표값을 구하는 것이 아니라 모든 x축값에 대한 y축값을 연산해야 하므로 직선의 방정식만으로 모든 y축값을 구하게 되면 많은 양의 실수의 곱셈과 나눗셈 연산을 필요로 한다. 본 논문에서 제시하는 정수형 변환 알고리즘은 모든 x축값에 대한

y축값을 구할 때 실수의 곱셈이나 나눗셈이 아닌 정수의 덧셈만을 사용하여 결과값을 얻어내므로 기존의 방법보다 매우 빠르다.

정수형 변환 알고리즘은 실수형의 소속함수를 정수형 격자에 매핑시켜 정수형의 값을 갖도록 한다. 격자좌표에서 직선은 시작점의 좌표와 끝점의 좌표를 잇는 직선 위에 위치한 격자 점들을 이어 표현된다.

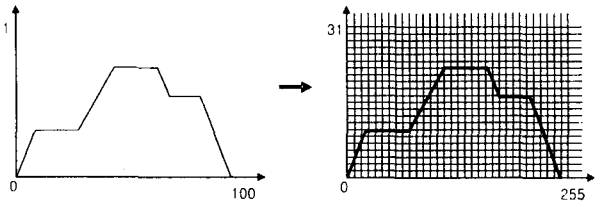

그림 1. 소속함수의 격자 좌표 표현

그림 1은 후건부의 소속정도값들의 MAX값들을 그래프로 나타낸 예이다. 그림을 보면 연속된 직선임을 볼 수 있다. 연속된 직선을 두개의 좌표를 갖는 여러 개의 직선으로 나누고 각각의 직선에 대해 다음의 방법으로 시작점에서 끝점을 잇는 직선의 점들에 대한 격자좌표를 빠르게 얻어낼 수 있다.

일반적으로 직선의 식은 기울기와 y축 접점을 이용하여 $y=mx+c$로 표현된다. 여기서 m은 직선의 기울기이고 c는 직선이 y축과 만나는 점의 y값이다. 시작점을 (x_0, y_0), 끝점을(x_k, y_k)라고 하고 그 사이의 점들을 시작점에서부터 순서대로 (x_1, y_1), (x_2, y_2),....,(x_{k-1}, y_{k-1})이라고 하자. 여기서 x, y의 값은 정수이다. 이 직선의 기울기가 양수이며 1보다 작을 경우에는 x의 값을 x_0에서 1씩 증가하면서 x값에 대응하는 y좌표를 구하고, 만약 1보다 큰 경우에는 y의 값을 y_0에서 1씩 증가하면서 y값에 대응하는 x의 좌표를 계산하는 방식으로 시작과 끝점사이의 좌표들을 계산한다. 기울기가 양수이며 1보다 작을 경우 $x_{k+1}(=x_k+1)$에서의 y좌표 값은 $y=m(x_{k+1})+c$를 이용하여 계산을 하면 y값은 정수가 아닌 실수 값을 가질 수도 있다. 그러나 격자의 좌표는 정수이므로 y의 값에 따라 적당한 정수의 값을 선택하는데 정수 좌표 선택 방법은으로 임의 좌표 x_k에 해당하는 y좌표 즉 y_k값을 이용한다. x가 x_k에서 $x_k+\frac{1}{2}$의 y값이 y_k에 가까운지 y_k+1에 가까운지를 판단하여 만약 y_k쪽에 가깝다면 x_k+1에서의 y좌표는 y_k로 그렇지 않고 y_k+1에 가깝다면 x_k+1에서의 y좌표는 y_k+1을 선택한다[2].

위와 같은 방법으로 정수연산만을 사용하여 모든 점들의 좌표를 구하는 알고리즘을 표 2에서 나타낸다.

표 2. 정수형 변환 알고리즘

```
IntFunction(x1,y1,x2,y2)
{
  x = x1 , y = y1
  dx = x2 - x1 , dy = y2 - y1
  setPixel(x, y)
  p = 2dy - dx
  for (k = 0; k < dx ; k++) {
        x = x + 1
        if (p < 0)
              p = p + 2dy
        else
              { y = y + 1, p = p + 2 (dy - dx ) }
        setPixel(x, y)
  }
}
```

하나의 직선은 두개의 포인터값을 갖으며 두개의 포인터 값 (x_0, y_0), (x_k, y_k)을 알면 x_0에서부터 x_k 까지 사이의 모든 x값에 대응되는 y값을 위의 정수형 변환 알고리즘으로부터 정수형으로 얻을 수 있다.

3. FPGA를 이용한 프로세서 설계

3.1 정수형 변환 알고리즘의 FPGA 설계

전건부에서 소속도를 구할 때에는 직선의 방정식을 사용하도록 설계하였으며 후건부 소속함수로부터 추론 결과를 구하기 위해서는 정수형 변환 알고리즘을 사용하도록 설계하였다. 후건부 소속함수로부터 얻은 추론 결과는 256개의 배열을 갖는 레지스터에 0에서 31사이의 정수값으로 저장되도록 하였다. 레지스터 값은 하나의 제어규칙으로부터 추론 결과를 얻을 때마다 레지스터에 이미 저장되어 있는 값과 MAX연산을 하여 갱신된다. 모든 제어규칙을 지나면 레지스터에는 최종 추론결과가 남게 된다. 표 3은 정수형 변환 알고리즘을 FPGA로 설계하기 위한 VHDL 코드의 일부이다.

표 3. 정수형 변환 프로세서 설계

```
architecture int_conversion of fuzzy is
    signal px,py: integer:=0:

begin
  process (clk,x1,y1,x2,y2)
  variable x,y,dx,dy,p,k : integer:=0;
  begin
    x := x1;
    y := y1;
    dx := x2 - x1;
    dy := y2 - y1;
    p := 2*dx -dy
    if(clk'event and clk = '1') then
       if ( k < dx ) then
             x := x + 1;
             k := k + 1;
           if ( p < 0 ) then
              p := p + 2*dy;
           else
              y := y + 1;
              p := p + 2 * (dy - dx);
           end if;
       end if;
    end if;
    px <= x;
    py <= y;

  end process;
end int_conversion;
```

3.2 비퍼지화 프로세서의 FPGA 설계

표 4는 비퍼지화 단계를 수행하기 위한 정수형 프로세서를 설계한 것으로 VHDL코드의 일부이다. 이 단계에서는 정수형 알고리즘 적용단계에서 추론결과를 저장한 레지스터값을 가져와 무게중심법을 이용한 비퍼지화를 통해 최종 결과값을 얻는데 사용된다.

표 4. 비퍼지화 프로세서 설계

```
architecture defuzzy of fuzzy is
        type reg_int is array (0 to 255) of integer;
    signal b: integer:=0;

begin
        process
        variable x,y : integer:=0;
        begin
                for i in 0 to 255 loop
                        x := x + (i*reg_int(i));
                        y := y + reg_int(i);
                end loop;
                b <= x / y;
        end process;
end defuzzy;
```

4. 실험 결과 및 분석

본 논문에서는 정수형 퍼지제어 시스템의 성능평가를 위해 세개의 입력변수와 하나의 출력변수를 갖는 퍼지제어기를 VHDL을 사용하여 FPGA로 설계하였으며 성능비교를 위해 실수연산만을 사용하는 퍼지제어기를 소프트웨어적인 방법으로 설계하였다. 사례에 대한 초기 설정은 3개의 전건부에 대해 각각 5개의 퍼지항과 125개의 제어 규칙을 갖게 하여 실험해 보았다. 기존의 퍼지제어 시스템의 성능과 본 논문에서 제안한 방법인 정수형 퍼지제어 시스템의 성능을 평가하여 표 5와 같은 결과를 얻었다.

표 5. 실험 조건 및 결과

NO	실수 연산		제안된 방법		오차
	결과값	연산속도(㎲)	결과값	연산속도(㎲)	
1	12.13	1963.43	12	88.14	0.13
2	33.19	1974.66	33	84.62	0.19
3	62.09	1967.93	62	84.58	0.09
4	68.09	1964.47	68	84.57	0.09
5	77.17	1966.78	77	84.47	0.17
			⋮		
48	52.08	1963.67	52	84.52	0.08
49	78.15	1965.57	78	84.41	0.15
50	80.06	1967.40	80	84.38	0.06
평균		1967.42		84.78	**0.112

실수연산에 의한 결과값과 제안된 방법에 의한 결과값을 살펴보면 계산된 값이 약간 다른 것을 볼 수 있다. 이것은 실수 값을 정수형 격자에 매핑 시킬 때 생긴 반올림 또는 버림으로 인한 값의 손실에 따른 오차이다. 하지만 이런 오차는 정수격자의 크기를 늘림으로써 오차정도를 줄일 수 있다. 또한 소속 함수를 적절히 조절함으로써 이런 오차로 인한 문제는 해결할 수 있다.

본 실험을 통해 연산 속도를 비교해 보면 실수연산을 사용하는 기존의 퍼지제어 시스템에 비해 FPGA로 설계된 정수형 퍼지제어 시스템이 대략 27.7배 정도 빠르다.

실험 결과에서 알 수 있듯이 제안된 고속 퍼지제어 시스템의 특징은 퍼지 추론 및 비퍼지화 과정에서의 실수 연산이 아닌 정수연산과 이를 적용한 정수형 퍼지프로세서를 설계하여 적용함으로써 기존의 퍼지제어기에 비해 빠른 연산속도를 보인다는 것을 볼 수 있다.

5. 결 론

대용량 퍼지 데이터를 처리하기 위해 설계된 퍼지제어기는 매우 빠른 연산속도를 필요로 한다. 기존의 퍼지 제어기는 실수 연산을 통해 퍼지 추론을 함으로써 추론 결과를 얻기까지 많은 시간을 실수연산에 허비하는 문제를 가지고 있다.

본 논문에서는 지금 까지 실수 연산만을 사용하던 퍼지 제어기에 대해 실수 값을 그에 대응하는 정수값으로 변환하는 방법을 적용하여 정수 연산만으로 퍼지 추론을 할 수 있는 고속 퍼지제어기를 구현하였다.

위의 실험 결과에서 알 수 있듯이 기존의 실수 연산에 의한 퍼지 추론 방법은 결과를 얻기까지 많은 시간이 필요 한데 반해 제안된 방법을 적용한 퍼지 추론은 보다 빠른 시간에 결론을 얻을 수 있다.

본 논문에서 제안된 방법을 복잡한 퍼지 계산이나, 빠른 추론을 요하는 퍼지 제어기에 적용하면 많은 성능 향상 효과를 얻을 수 있을 것이다. 또한, 인공위성으로부터 수집된 원격탐사화상과 같이 대용량의 퍼지데이터에 대한 실시간 고속 처리를 요구하는 위성 영상의 패턴 인식에 응용할 수 있다. 향후 연구로서 본 실험에서는 입력값을 PC의 PCI버스를 통하여 퍼지제어 시스템과 송수신하면서 생기는 지연시간과 프로그램에서 생기는 지연시간을 없애기 위해 독립적인 퍼지제어 시스템의 FPGA 설계가 필요하며 정수 변환시 생기는 오차를 최소화할 수 있는 연구가 필요하다.

참고문헌

[1] J. Yen and R. Langari, *Fuzzy Logic : Intelligence*, Prentice Hall, 1999.

[2] F. S. Hill, *Computer Graphics, 2nd ed*, Prentice Hall, 2001.

[3] E. Cox, *Fuzzy System Handbook*, AP Professional, 1994.

[4] J. Yen and L. Wang, "Simplifying fuzzy rule-based models using orthogonal transformation method," *IEEE Trans. System, an, and Cybernetics-Part B*, vol. 29, no. 1, 1999.