Real Time Processing을 위한 Image Processing Unit의 설계

*김진욱, 김석태
부경대학교 정보통신공학과

Design of Image Processing Unit for Real Time Processing

*Jin Wook Kim, Seok-Tae Kim
Dept. of Telematics Eng. Pukyong National University

요약

Image Processing은 Image Data가 대량이고 내재된 정보가 병렬로 연관성을 가진다는 측면에서 실시간 처리가 용이하지 않다. 본 연구에서는 High Speed Real Time Image Processing을 위한 IPU(Image Processing Unit)와 이를 구동하기 위한 High Speed Real Time Image Processing Language인 IPASM(Image Processing Assembly)을 제안한다. 우선 IPU의 기본개념을 설명하고 IPU의 구현을 위한 IPLU(Image Processing Logic Unit)를 설계한다. 그 후 Window98 환경에서 구동 가능한 IPASM Interpreter를 실제로 제작하여 IPU의 동작방식을 간접적으로 진단한다.

1. 서론

실시간 비젼시스템은 항공기나 미사일의 실시간 유도시스템, 실시간 표적겨냥시스템, FA(Factory Automation)의 실시간 제어시스템 등과 같이 군사, 산업 전반에 걸쳐 폭 넓게 활용되고 있다. 그러나 화상 데이터 자체가 대량이고 내재된 정보가 병렬로 연관성을 가지고 있어서 실시간 비젼시스템을 구현하는 것은 쉽지않다[1]. 특히 도면화상, 지도화상, 항공화상 등에는 기하학적인 특징을 나타내는 정보 뿐만 아니라 그에 관련된 많은 정보가 내재되어 있어 사용자의 요구에 따라 실시간으로 화상 속의 목적 데이터를 분리, 추출, 인식하여 시스템을 제어하는 것은 더욱 어렵다[2,3].

지금까지의 실시간 화상처리 알고리즘은 크게 두 종류가 있는데 프레임그레버를 이용하는 알고리즘[4-6]과 광전자공학 등 광학적 도구를 이용하는 알고리즘[7-9]이다. 프레임그레버를 이용하는 방법은 CCD카메라에서 화상신호를 받아서 처리하므로 고속 화상처리에 한계가 있으며 기능이 제한적이었다. 또한 광학적 접근법은 장비의 크기가 커지는 단점이 있으며 광학소자 자체의 노이즈로 인하여 도면이나 지도화상 등 정밀한 화상에는 부적합하였다.

본 논문의 알고리즘은 실시간 화상처리 전용 VLSI인 IPU(Image Processing Unit)를 제안하여 NTSC나 PAL신호를 거치지 않고 Image Processing Language IPASM(Image Processing Assembly)으로 CCD소자의 화상정보를 처리한다. 따라서 하드웨어가 간단해지면서 처리속도는 빠른 장점이 있다.

2.에서 IPU를 제안 하고 3.에서 IPASM 인터프리터를 이용하여 IPU의 설계와 동작방식을 진단한다.

2. IPU(Image Processing Unit)

2.1 고속화상처리의 비젼시스템.

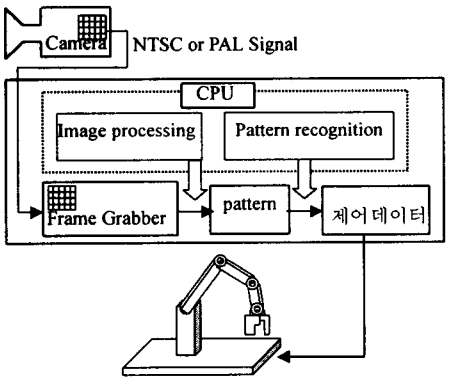

그림 1. 기존의 비젼시스템.

그림 1의 기존의 비젼시스템 동작방식은 다음과 같다. 우선 카메라의 비디오신호를 프레임그레버가 화상데이터로 만들어 컴퓨터에 입력한다. 컴퓨터는 이를 가공해서 필요한 정보를 추출한 뒤 인식하여 최종적으로 제어데이터를 제어시스템에 전달한다. 현재 FA에서는 이러한 시스템이 주류를 이루고 있으나 이 시스템은 실시간 병렬처리에는 부적합하다. 그 이유는 카메라의 NTSC나 PAL 신호가 직렬 신호이므로 병렬 알고리즘을 직접 적용하는 것이 불가능하다는 점, CPU는 워드단위 데이터의 고속 순차처리를 목적으로 설계되어 화상의 고속처리에서는 병목현상이 있을 수 있다는 점 때문이다. 실시간 화상처리가 가능한 비젼시스템은 CCD소자에서 받아들이는 영상을 NTSC나 PAL

신호가 아닌 행렬 데이터로 IPU에 보냄으로써 CCD 소자의 광학신호를 바로 처리할 수 있다. 또 IPU는 화상 단위로 행렬 병렬처리를 할 수 있으므로 CPU에서 발생하는 병목현상을 없앨 수 있다. 행렬데이터는 인식하기 쉬운 패턴으로 가공 처리한 뒤 패턴인식시스템으로 넘겨준다. 패턴인식시스템에서는 이를 인식하여 최종적으로 제어데이터를 제어시스템에 전달한다. 그림으로 표현하면 그림 2와 같다.

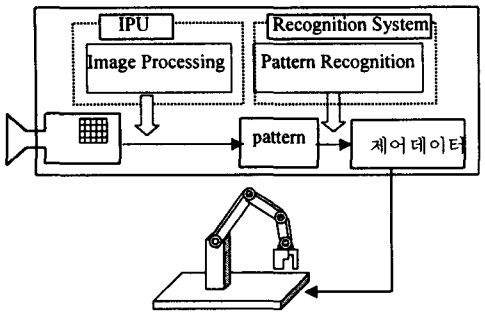

그림 2. 실시간 화상처리가 가능한 비젼시스템.

2.2 IPU의 설계

2.2.1 IPU의 인터페이스

IPU의 인터페이스는 CCD가 외부로부터 받은 광학신호, 클럭셋(버스클럭, 연산클럭, 제어클럭), 명령어를 입력으로 외부 메모리나 시스템에 화상을 출력한다. CCD의 크기를 $2^m \times 2^n$이라고 두고 한 셀의 분해력이 p bit 이면 IPU의 인터페이스는 Pixel Address n bit와 m bit, p bit 데이터 버스가 고려될 수 있으며 이를 그림으로 표현하면 그림 3과 같다.

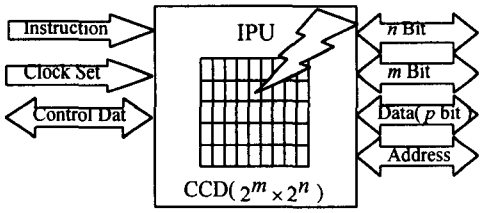

그림 3. IPU의 인터페이스.

2.2.2 IPUL(Image Processing Logic Unit)의 설계

화상처리연산은 Point 연산, Local 연산, Global 연산으로 나뉜다. IPUL은 화상처리 연산 중 Point 연산과 Local 연산을 구현하기 위하여 설계된 병렬 화상처리 전용로직이며 7가지 레지스터 HTR(Histogram Table Register), PAMR(Parallel Arithmetic Matrix Register), OMR(Operand Matrix Register), PABMR(Parallel Arithmetic Binary Matrix Register), BOMR(Binary Operand Matrix Register), TR(Temporal Register), TMR(Temporal Matrix Register)와 유기적으로 동작한다. 이를 그림으로 표현하면 그림 4와 같다.

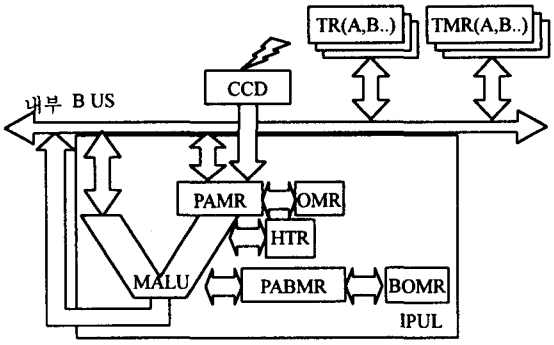

그림 4. IPLU의 내부구조

2.2.3 MALU(Matrix Arithmetic Logic Unit)

MALU는 Point 연산 중 픽셀의 그레이레벨 수치조정, 화상끼리의 집합연산이나 산술연산, 히스토그램, 이치화 등이 지원되도록 설계한다. 빠른 병렬연산을 하려면 픽셀에 ALU가 하나씩 대응되어 연산을 처리하면 되지만 하드웨어비용이 증가하는 문제점이 있다. 그러나 고속연산 클럭을 따로 사용하여 행렬연산을 하면 적은 비용으로 빠른 연산속도를 얻을 수 있다. 이를 그림으로 나타내면 그림 5와 같다.

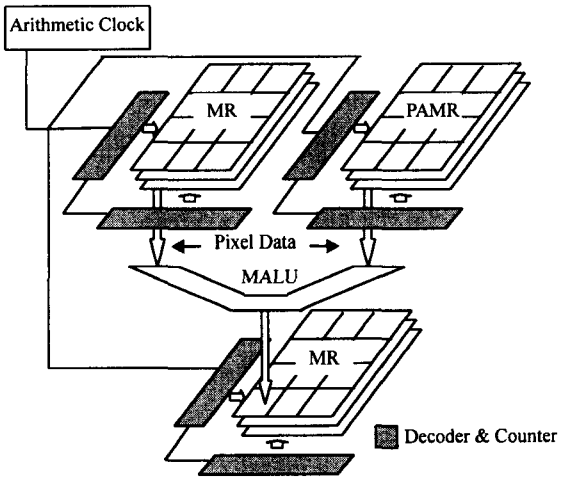

그림 5. MALU의 기본구조.

2.2.4 HTR

HTR은 화상의 히스토그램 수치가 저장되는 레지스터이며 2^p개의 h bit 카운터로 구성된다. h는 화상의 전체 픽셀 수를 MPA라 두었을 때 식(1)과 같이 표현된다. 여기서 MPA은 다시 식(2)

와 같이 표현 되므로 h는 식(3)과 같다. HTR을 기준으로 화상의 이치화를 할 수 있으며 이치화상은 PABMR에 저장된다. HTR의 내부구조는 그림 6에 나타내었다.

$$h \le \log MPN \quad (1)$$

$$MPN = 2^m \times 2^n \quad (2)$$

$$h \le m + n \quad (3)$$

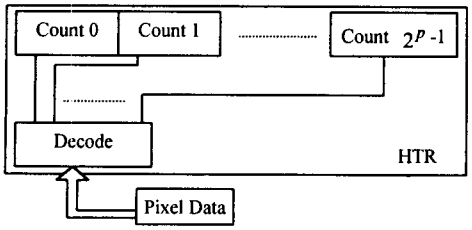

그림 6. HTR의 내부구조

2.2.5 PAMR과 PABMR

PAMR과 PAMBR은 각각 $L_H \times L_v$ 크기의 OMR, BOMR을 연산자로 PALU가 그레이화상과 이진화상의 Local 병렬연산을 PALU가 연산할 수 있도록 설계된다. 즉, $L_H \times L_v$ 크기의 그레이 모폴로지연산자나 그레이 필터로 이진 모폴로지연산, 그레이 모폴로지연산, 그레이 필터링연산, CA(Cellular Automata)연산을 병렬로 처리할 수 있다. 기본적인 내부구조는 그림 7과 같다.

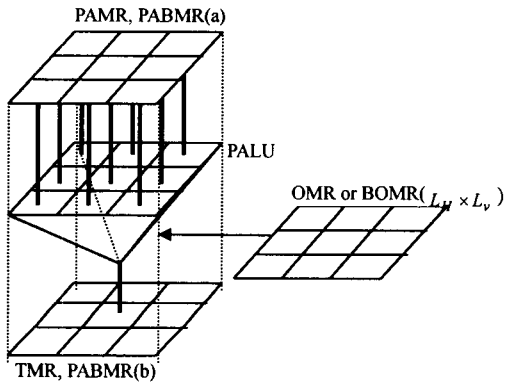

그림 7. PAMR과 PABMR의 기본적인 내부구조

2.2.6 TR

IPLU의 연산결과가 저장되는 레지스터는 MTR(A,B,c,…), TR(A,B,c,…)두 가지 종류가 있다. MTR은 화상처리연산의 결과가 일시 저장되는 행렬 레지스터이고 TR은 히스토그램과 같이 수치연산의 결과가 일시 저장되는 레지스터이다. TR은 히스토그램 값이 저장 될 수도 있어야 하므로 h bit보다 크게 구성되어야 한다. TR은 하드웨어의 용량에 따라 TRA, TRB, TRC… 등으로 구성될 수 있다.

2.2.7 TMR

TMR은 IPLU의 연산결과 중 화상데이터가 저장되는 행렬레지스터이며 하드웨어의 용량에 따라 TMRA, TMRB, TMRC.. 등으로 구성될 수 있다.

3. IPASM(Image Processing Assembly)

IPU에 제어명령을 내리기 위해서는 이진 명령어 집합이 필요하다. 이 명령어 집합을 체계화 한 것이 Image Processing Language인 IPASM이다.

IPASM은 기본적으로 두개의 오퍼랜드를 사용한다. 이는 MALU가 2개의 행렬레지스터에 연결되는데 기인한다. 또 병렬연산은 화상이 PAMR이나 PABMR에 로드하여야 된다. 이는 다른 행렬레지스터에서는 병렬연산 기능이 지원되지 않기 때문이다. 또 CCD의 화상은 PAMR로 로드되기 때문에 Load AMX, CCD 는 잘못된 코드이다. 이러한 일련의 규칙들을 점검하여 오류를 지적해주고 기계어로 만들어주는 컴파일러 소프트웨어를 Image Processing Assembler라 정의한다.

일 예로 CCD카메라를 통하여 화상을 읽어 히스토그램을 구하고 이치화 하는 알고리즘을 IPASM으로 표현하면 다음과 같다.

```
Load PAM, CCD
Get HT
GetThreashold ( AX )
Binarization AX
```

여기서 'Get HT'는 히스토그램을 구해서 HT에 저장하라는 명령어이고 'GetThreashold (AX)'는 Threshold 값을 구하여 AX에 저장하는 함수이며 'Binarization AX'는 PAMR의 화상을 AX를 기준으로 이진화 하라는 의미이다.

IPASM은 기존의 수식표현보다 화상처리 알고리즘을 더욱 명확히 표현할 수 있으며 이를 함수화 하여 라이브러리로 만들어 두면 화상처리 알고리즘에 관한 노하우가 데이터베이스로 저장될 수 있다.

3.1 IPASMI(IPASM Interpreter)의 제안

IPASMI는 Window98 환경에서 IPASM의 소스코드와 VIPU(Virtual IPU)를 시뮬레이션할 수 있는 소프트웨어이다.

VIPU는 IPASMI 내부에 C++ 라이브러리로 연결되어 있어서 IPASMI가 호출하면 동작하게 구성한다. 이를 그림으로 표현하면 그림 8과 같다.

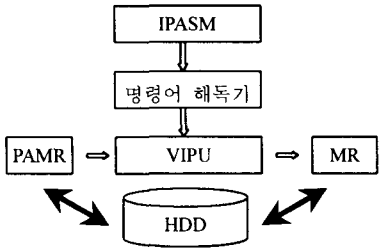

그림 8. IPASMI 의 동작방식.

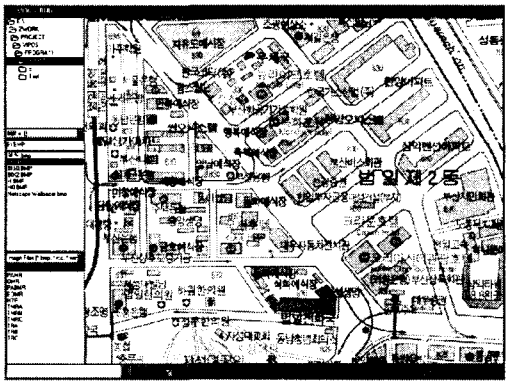

그림 9. IPASMI 의 실행화면

IPASMI 는 PC 에서 실시간 화상처리시스템을 개발하기 이전에 알고리즘을 시뮬레이션 함으로써 제품개발 시 발생할 수 있는 오류를 최소로 줄일 수 있다.

그림 9 는 IPASMI 의 실행 화면이다. 좌측의 상단은 하드디스크의 파일을 볼 수 있도록 GUI 로 구성하였다. 또 좌측하단은 VIPU 내부의 레지스터를 볼 수 있도록 구성되었으며 IPASM 을 편집하여 실행하면 실행과정을 지켜볼 수 있다. 또 이미 만들어진 IPASM 을 디버그 하여서 알고리즘의 오류를 찾을 수도 있다.

4. 결론

실시간 화상처리는 FA 와 군사 등으로 폭 넓게 사용되고 있다. 본 논문에서는, 실시간으로 화상을 고속 처리하기 위해서는 화상처리 전용 유닛이 필요하다는 점에 착안하여 효과적인 시스템을 제안하였다. 먼저 실시간 화상처리 전용 Unit 인 IPU 와 내부의 로직을 설계 하였고 이를 구동하기 위한 IPASM 을 사용하여 효율적인 화상처리 알고리즘을 표현하는 방법을 보았으며 Window98 환경에서 구동하는 IPASM 인터프리터를 제작하여 IPU 의 동작방식을 간접적으로 살펴보았다.

본 논문의 IPU는 그레이화상을 처리할 수 있도록 설계되었으나 같은 개념으로 컬러화상도 처리할 수 있는 전용프로세서도 구성할 수 있다. 또한 고기능의 인식유닛을 탑재해 외부와의 인터페이스를 할 수 있도록 설계할 수 있다.

앞으로는 IPU의 내용들을 IPASMI로 충분히 시뮬레이션 해본 후 실제 IPU를 구현하였을 때의 문제점을 파악하여야 할 것이다.

참고문헌

[1] J.Daugman, "Neural Image Processing Strategies Applied in Real-Time Pattern Recognition", Real-Time Imaging. V.3 N.3, 1077-2014, 1997.1

[2] 김진욱, 김석태, "16 방향모폴로지를 이용한 도면의 선형성분의 추출", 한국통신학회논문지 제 23 권 제 3 호, pp.591-602, 1998

[3] 김진욱, 김석태, "컬러 모폴로지를 이용한 지도의 범례정보 추출", 한국통신학회 하계종합학술발표논문집, Vol.16.No.1,pp.379-382,1997

[4] Gilbert and Yang, "A Real-Time Face Recognition System Using Custom VLSI Hardware", IEEE Workshop on Computer Architectures for Machine Perception, December 1993

[5] Melton, Roy W. Huang, Tsai Chi Alford, Cecil O. Becker, Latika, "VLSI system implementation for real-time object detection", IEEE International Symposium on Circuits and Systems Vol 4, 1996. 12

[6] Zhang YM, Kovacevic R, "Real-Time Sensing of Sag Geometry During GTA Welding", Journal of Manufacturing Science & Engineering, Transactions of the ASME ,V.119 N.2 , 1997. 1

[7] Huang GL, Jin GF, Wu MX, Yan YB, "Developed, binary, image processing in a dual-channel, optical, real-time morphological processor", Applied Optics, V.36 N.23 , 1997. 10

[8] Mario Kreibl, Heiko Schwarzeer, Stephan Teiwes, "Optical processor for real-time detection of defects in textile webs", Optical Pattern Recognition VIII , 1997. 4

[9] N. McArdle, T. Komuro, M. Naruse, H. Yamamoto, H. Sakaida, and M. Ishikawa, "A Smart-Pixel Free-Space Interconnected Parallel Processing System", in Smart Pixels Technical Digest, p59, IEEE/LEOS Summer Topical Meeting, Keystone, Colorado, August 5-9 (1996).