FPGA를 이용한 NCC기반의 실시간 스테레오 매칭 프로세서 구현

김병진*, 배상민**, 고광식***
*경북대학교 IT대학 전자공학부
e-mail : kimbj123@nate.com

FPGA implementation of NCC-based real-time stereo matching processor

Byeong-Jin Kim*, Sang-Min Bae**, Kwang-Sik Koh*
*School of Electronics Engineering, Kyung-pook National University

요 약

스테레오 비전 시스템에서 전통적인 매칭 알고리즘으로 SAD(Sum of Absolute Differences), SSD(Sum of Squared Differences), NCC(Normalized Cross Correlation) 등 다양한 알고리즘이 존재한다. 그러나 하드웨어로 실시간 처리를 위한 시스템을 구현하기 위해서는 리소스가 한정 되어있다는 제약 때문에 많은 연구에서 SAD 혹은 RT(Rank Transform), CT(Census Transform)를 많이 사용하게 된다. FPGA 내부에는 BRAM(Block RAM)과 MAC(multiply-accumulator)인 DSP슬라이스가 이미 존재한다. 본 논문에서는 BRAM과 DSP로직을 활용해서 전통적인 매칭 알고리즘 중에서 연산기 사용이 가장 많은 NCC를 FPGA로 실시간 처리 가능한 하드웨어 구조를 제안한다.

1. 서론

3D 거리 정보 획득은 로보틱스, 오토모티브 같은 분야에서 많이 요구되고 있다. 3D 거리 정보를 획득하는 방법에는 크게 능동적(Active) 방법과 수동적(Passive) 방법으로 나뉠 수 있다.

능동적인 방법으로는 TOF(Time-Of-Flight) 센서를 이용이 있다. 그리고 프로젝터와 카메라를 이용해서 특정패턴(Fringe, Stripe, etc)을 물체에 투영하고 카메라에서 얻어진 패턴을 물체가 있을 때와 없을 때의 패턴을 비교해서 거리정보를 획득하는 방법 등이 있다.

수동적인 방법으로는 가장 많이 사용되는 방법이 두 대의 카메라를 이용한 스테레오 비전 시스템이다. 그 외에도, 카메라의 위치를 이동시켜 두 장의 이미지를 사용해서 물체의 크기 변화를 이용해서 거리 정보를 획득하는 방법, 카메라 렌즈의 초점에 따라 물체의 흐릿한 정보를 사용해서 거리를 획득하는 방법 등이 있다.

또한 수동적인 방법에 능동적 방법을 적용해서 스테레오 카메라에 패턴 또는 질감(Texture)을 투영하여 거리 정보를 획득하는 방법도 있다.

수동적인 방법 중 하나인 스테레오 카메라 시스템에서 거리 정보를 계산하는 것은 다른 3D 센싱 방법들보다 많은 이점이 있다.[1]

첫째로, 수동적(Passive) 센싱 방법이다. 능동적(Active) 방법의 경우에는 신호를 프로젝션을 해야 하거나 높은 전력이 필요하거나, 안전 문제, 환경의 제약이 따른다.

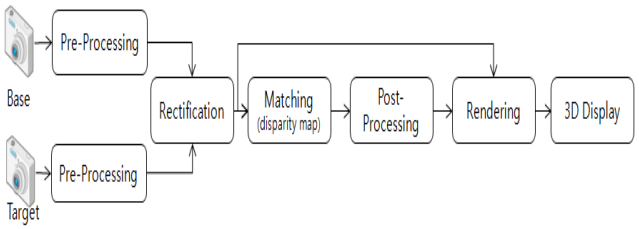
둘째로, 스테레오 센싱은 거리 정보를 정확하게 포함하는 컬러 혹은 흑백 이미지를 제공한다. 이 이미지는 전통적인 2D 방법을 사용하거나 컬러와 거리 이미지 데이터를 결합하는 새로운 방법으로 해석 할 수 있다.

셋째로, 동작범위와 거리 Z 해상도는 유연성을 가진다. 렌즈의 뷰 필드와 이미지 사이즈, 렌즈간의 거리에 따라서 달라지기 때문이다.

따라서 본 논문에서는 거리 정보를 획득하는 방법으로 스테레오 시스템을 사용하였으며, 핵심이 되는 정합 알고리즘으로 NCC 알고리즘을 사용하여 FPGA로 실시간 처리가 가능한 구조를 제안하고, 구현해 본다. 2장에서는 정합 알고리즘에 대해 살펴보고, 3장에서 실시간 처리 가능한 하드웨어 구조를 제안하며, 4장에서는 구현에 대한 결과를 나타냄으로써 논문을 결론짓는다.

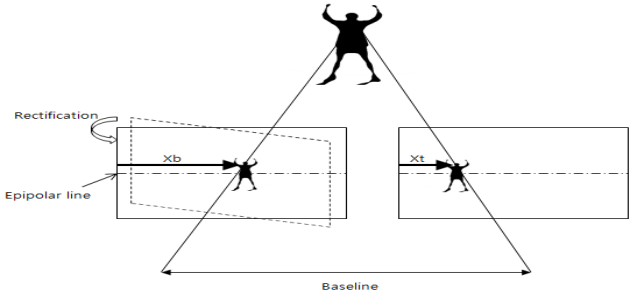
2. 스테레오 비전

스테레오 비전은 일반적인 처리과정이 (그림1)과같이 진행된다. 두 카메라에서 받은 이미지를 전처리 과정으로 노이즈 제거, 혹은 시차(disparity)를 찾기 위해 도움이 되는 필터 등을 처리하고, 다음 단계에서 이미지 교정(rectification and calibration[2])과정을 거치게 된다. 이미지 교정과정을 거치게 되면 두 이미지의 대응점이 같은 라인에 존재하기 때문에, 시차를 계산하기 위한 처리 과정이 효율적으로 된다(그림2참조). 다음 후처리과정을 거치고, 랜더링으로 이미지에 3D 효과를 적용한 후 보여준다.



(그림 1) Stereo vision system flow chart

본 논문에서 핵심사항은 시차맵을 찾는 과정을 하드웨어로 효과적으로 처리하기 위한 방안을 모색한다.



(그림 2) The one-dimensional search space for corresponding pixels

I) 스테레오 정합 방법

스테레오 정합 과정은 이미지 교정과정을 거친 이미지에 대해 에피폴라 라인(epipolar line)에서 대응하는 점(corresponding point)을 찾는 과정을 말한다. 대응점을 찾기 위해 사용되는 전통적인 정합 함수는 차이를 기반으로 하는 SAD, SSD 등이 있고, 상관관계 기반으로 하는 NCC, RT, CT 등이 있다. 차이를 기반으로 하는 함수의 경우에 계산이 간단하나, 노이즈가 많은 환경에서 비교적 좋지 않은 결과를 나타낸다. 따라서 본 논문에서는 상관관계 기반 함수로 NCC를 사용한다. NCC(normalized cross correlation)는 두 이미지에서 유사도(similarity) 혹은 차이도(dissimilarity)를 평가할 때 많이 사용된다.

정합 함수를 기반으로 대응점을 찾는 방법을 크게 전체적인 정합(Global Matching)과 국소적인 정합(Local Matching)두 가지로 나눌 수 있다. 전체적인 방법으로는 DP(Dynamic Programming), BP(Belief Propagation) 등이 있고, 국소적인 방법은 WTA(Winner Takes All)방법이 일반적이다. 국소적인 방법에 비해 전체적인 정합 방법은 성능에서 우수하나, 실시간 처리하기에 계산과정이 복잡하고, 시간복잡도가 크다. 따라서 본 논문에서는 국소적인 방법기반 정합비용집계(matching-cost aggregation)로 SNCC(summed normalized cross-correlation)[3]를 사용한다. 간단한 설명은 2.2에서 소개한다.

II) NCC 기반 정합 알고리즘

국소적인 방법은 윈도우 사이즈와 윈도우 형태, 정합 함수에 따라 정합비용(Matching cost) 계산에 많은 영향을 준다. Middlebury에 평가되어 있는 여러 알고리즘들 중 상위권에 위치하고 있는 알고리즘은 대부분 전체적인 정합 방법을 사용하거나 최적화(optimization) 윈도우를 찾아서 정합하는 방법, 혹은 여러 가지 필터와 함께 반복적인 처리를 통해서 해결하는 방법 등을 사용한다. 이러한 방법은 성능은 우수하지만, 자원이 한정되어 있는 하드웨어로 처리하기 어려울 뿐 아니라, 실시간으로 처리하는데 적당하지 않다. 그 중에서 국소적으로 처리하면서 상당한 성능을 나타내고 있는 알고리즘으로 SNCC 방법이 있다. SNCC방법은 2단계로 나누어서, 1단계로 NCC로 정합비용을 계산하고, 다음 단계에서 2D 윈도우내의 평균을 계산하여 WTA로 최댓값을 찾는 방법이다.

NCC 함수의 일반적인 식은 다음과 같다.

$$NCC(d) = \frac{\sum_W (B(W) - \overline{B}(W)) \cdot (T(W+d) - \overline{T}(W+d))}{\sqrt{\sum_W (B(W) - \overline{B}(W))^2 \cdot \sum_W (T(W+d) - \overline{T}(W))^2}} \quad (1)$$

$$where,$$
$$B : base\ image$$
$$T : target\ image$$
$$W : windows\ size (M \times N)$$
$$d : disparity$$
$$\overline{X} : average\ of\ X$$

NCC 함수 식(1)에서 윈도우 내의 좌, 우 영상의 화소값의 공분산(covariance)을 분자로 하고, 표준편차를 분모로 가진다. 시차(disparity)는 Base image를 좌, 우 어떤 영상을 기준으로 할 것인지에 따라서 + or - 값을 가진다. d의 값의 범위는 $[d_{\min} \sim d_{\max}]$의 범위를 가지게 된다.

3. 실시간 처리 가능한 NCC 하드웨어 구조

NCC의 일반식을 픽셀 단위로 처리하기에는 많은 시간이 소요 될 뿐 아니라, 실시간 처리에서 너무 많은 연산기가 소요된다. 윈도우 사이즈가 커짐에 따라서 연산기의 개수는 기하급수적으로 늘어난다. 또한 반복적인 계산이 동일하게 이루어지기 때문에 하드웨어로 설계하기에는 적합하지 않다. 따라서 (1)의 수식을 변형한다.

$$NCC(d) = \frac{\sum_W (B(W) \cdot T(W+d)) - MN(\overline{B}(W) \cdot \overline{T}(W+d))}{\sqrt{\sum_W B^2(W) - MN\overline{B}^2(W)} \cdot \sqrt{\sum_W T^2(W+d) - MN\overline{T}^2(W)}} \quad (2)$$

각 매개변수(parameter)간에 의존성을 가진 수식(1)에서와는 달리 식(2)로 변형함으로써, 각 항목의 독립적인 계산이 가능하다.

여기서, 핵심 사항은 $\sum$ 계산과 $B \cdot T$을 어떻게 할 것인가에 대해 초점을 맞춘다. 지금까지 연구된 방법으로 빠른 NCC 계산을 위해서 제안된 방법으로 SUM-TABLE를 이용한 방법이 존재한다.[4]

Sum-Table에 의한 방법은 다음식과 같이 계산 된다.

$$S(x,y) = g(x,y) + S(x-1,y) + S(x,y-1) - S(x-1,y-1) \quad (3)$$

$$with\ S(x,y) = 0\ when\ either\ x,y < 0$$

$$\sum_{i=-\frac{m}{2}}^{\frac{m}{2}} \sum_{j=-\frac{n}{2}}^{\frac{n}{2}} g(x+i,y+i) = S(x+\frac{m}{2}, y+\frac{n}{2}) - S(x-\frac{m}{2}-1, y+\frac{n}{2}) \quad (4)$$
$$- S(x+\frac{m}{2}, y-\frac{n}{2}-1) + S(x-\frac{m}{2}-1, y-\frac{n}{2}-1)$$

$g(x,y)$는 2차원 합, $x = 0,1,2,\cdots,M-1$, $y = 0,1,2,\cdots,N-1$을 나타낸다.

수식(3),(4)의 식으로 하드웨어를 설계할 경우 단점이 저장해야 할 bit 수가 이미지 전체를 총 합한 bit만큼 필요하다. 만약 이미지 사이즈가 커져버린다면, 합에 필요한 데이터 bit는 8bit 이미지 일 경우, $255 \times width \times height$ 만큼 필요하다. 자원이 제한된 FPGA에서 구현하기에는 적당하지 않은 방법이다.

본 논문에서 제안하는 방법은 SUM-Table과 비슷하지만, 데이터 저장을 위해 사용되는 메모리 버퍼의 1-word는 $255 \times M \times N$ (M,N : window size)로 제약된다.

제안된 방법은 다음의 수식으로 정리 된다.

$$P(n) = g(n) + P(n-1) \quad \begin{cases} g(n) : data \\ P(n) = 0, when\ n < 0 \end{cases}$$

(5)

$$A(n) = P(n) - P(n-M) \quad (6)$$

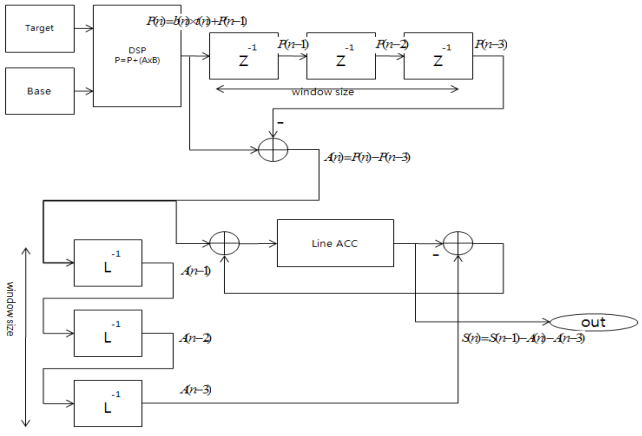
$$S(n) = S(n-1) + A(n) - A(n-N) \quad (7)$$

$A(n)$: 1D 윈도우 합

$S(n)$: 2D 윈도우 합

식(6)과 (7)의 식으로 구성을 함으로써, 메모리의 데이터 사이즈는 윈도우 내의 총합을 한 비트수로 Sum-Table를 사용하는 방법보다 현저하게 줄어드는 것을 알 수 있다.

식(6),(7)을 이용해서 FPGA에 구현하면 (그림3)와 같은 구조로 구현 할 수 있다.



(그림 3) $\sum_{W} B(W) \cdot T(W)$, window size : 3x3

DSP 로직의 경우 곱셈(multiplier)과 덧셈(add/sub)을 한꺼번에 처리가 가능하고, 사용한 FPGA 내부의 DSP 슬라이스에 누산기(Accumulator)가 포함되어 있기 때문에, 수식(5)에서 $g(n) = B \cdot T$ 로 계산하게 되면 P(n)은 DSP 로직 하나로 구현이 가능하다. FPGA내부에 이미 구현되어있는 슬라이스를 사용함으로써, 자원의 효율성을 높인다.

<표 1 > compare proposed method and other method

Window size : MxN (W : image width H : image height)		The sum-table method		The traditional method		The proposed method	
		Add/ Sub	Mul	Add/ Sub	Mul	Add/ Sub	Mul (mac)
Calculation of NCC	$\sum\sum B$	6	0	MxN-1	0	4	0
	$\sum\sum B^2$	6	1	MxN-1	MxN	3	1
	$\sum\sum B \cdot T$	6	1	MxN-1	MxN	3	1
Total		18	2	3xMxN-3	2xMxN	10	2
1 word of memory (data : 8bit)		255xHxW		255		255xMxN	

4. 시스템 구현 및 성능 분석

I) 시스템 구현

제안한 실시간 스테레오 비전 시스템은 Xilinx에서 제공하는 System Generator을 사용하여 설계되었고, Virtex-4 XC4VSX35-10 에 구현되었다. 합성툴은 ISE 13.2버전으로 내부에 있는 Xilinx synthesis tool을 사용하였다. 제안된 시스템의 최대 동작 주파수는 80 Mhz이고 구현(implementation) 후 리소스 사용량은 <표 2>에 나타내었다.

구현한 시차(disparity)범위는 [0 ~ 15]이고, 테스트는 Middlebury[5]에서 제공하는 이미지를 다른 보드에서 실시간으로 출력하고, 처리 후 결과를 입력받아 PC에서 처리한 데이터와 비교함으로써, 구현 검증을 하였다. 그 결과 PC에서 처리한 영상과, FPGA에서 처리한 영상의 결과는 동일한 결과를 얻을 수 있었다. 테스트에 사용된 이미지는 middlebury에서 제공하는 이미지 중에서 동일한 시차맵 범위를 가지는 (그림4)의 첫 번째 이미지 Tusukuba 이미지를 사용하여 검증하였다.

<표 2 > Amount of used total resource
(virtex-4 xc4vsx35-10)

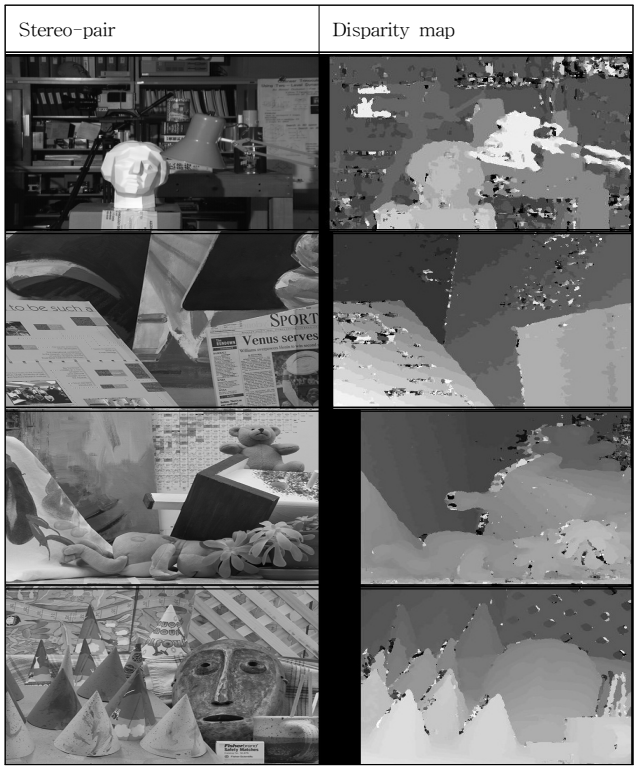
Logic Utilization	Used	Available	Utilization
Number of Slice Flip Flops	26,901	30,720	87%
Number of 4 input LUTs	10,857	30,720	35%
Number of occupied Slices	14,188	15,360	92%
Total Number of 4 input LUTs	10,949	30,720	35%
Number of FIFO16/RAMB16s	168	192	87%
Number of DSP48s	124	192	64%

II) 성능 분석 및 결론

스테레오 비전 처리를 FPGA에 구현한 논문들은 많이 존재한다. SAD 기반, Census 변환, 위상기반 방법, 등이 있고, 명암도 기반으로만 하는 것이 아니라 컬러 기반으로 처리를 하는 경우도 있다. 하지만 이 경우에도 많은 경우

에 NCC를 사용하는 경우는 찾아보기 힘들다. 따라서 우리는 본 논문에서 NCC를 기반으로 실시간으로 처리 가능한 스테레오 비전처리를 FPGA에 효율적인 방법으로 구현 하였다.

Sum-Table을 사용할 경우 기존의 방법으로는 구현하기 힘든 실시간 처리가 가능하나, 연산기의 사용량은 감소하는 대신 메모리와 레지스터의 사용량이 크게 증가하는 단점이 있었다. 제안한 구조를 사용할 경우 메모리 사용량을 크게 감소 줄일 수 있다. 본 논문에서 구현한 내용을 토대로 예를 들면, VGA(640x480)이미지의 8bit 데이터일 때, 윈도우 사이즈는 3x3을 사용했을 경우, Sum-Table은 27bit, 제안한 방법은 12bit의 메모리 1-word 사이즈가 필요하다. 따라서 44%의 bit 사이즈 감소 효과를 볼 수 있다. 또한 연산기의 사용량도 Sum-Table에 비해 적게 사용한다.<표1 참조>



(그림 4) Evaluation result of stereo matching algorithm using stereo-pair in middlebury

5. Acknowledgement

본 연구는 경북대학교 IT융복합글로벌인재양성센터의 지원을 받아 로봇을 원격제어하기 위한 고해상도 영상처리 모듈 개발 과제로 수행되었음.(No. 1345-0966-50)

참고문헌

[1] John Iselin Woodfill, Gaile Gordon, Ron Buck, “Tyzx DeepSea High Speed Stereo Vision System”, 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops

[2] G. Bradski and A. Kaehler, “Learning OpenCV: Computer Vision with the OpenCV library.” O′Reilly Media, Inc., 2008.

[3] Nils Einecke, Julian Eggert, “A Two-Stage Correlation Method for Stereoscopic Depth Estimation”, 2010 Digital Image Computing, pp.227-234

[4] D.M. Tsai and C.T. Lin “Fast normalized cross correlation for defect detection”

[5] Middlebury stereo vision page [Online]. Available: http://vision.middlebury.edu/stereo/