JANT 한국항행학회논문지
Journal of Advanced Navigation Technology

항행 및 항법

J. Adv. Navig. Technol. 19(6): 544-551, Dec. 2015

첨단운전자보조시스템용 이동객체검출을 위한 광학흐름추정기의 설계 및 구현

Design and Implementation of Optical Flow Estimator for Moving Object Detection in Advanced Driver Assistance System

윤 경 한 · 정 용 철 · 조 재 찬 · 정 윤 호*
한국항공대학교 항공전자정보공학부

Kyung-Han Yoon · Yong-Chul Jung · Jae-Chan Cho · Yunho Jung*
School of Electronics and Information Engineering, Korea Aerospace University, Gyeonggi-do 412-791, Korea

[요 약]

본 논문에서는 첨단 운전자 보조 시스템 (ADAS; advanced driver assistance system) 용 이동객체검출 (MOD; moving object detection)을 위한 광학흐름추정기 (OFE; optical flow estimator) 의 하드웨어 구조 설계 결과를 제시하였다. 광학흐름추정 알고리즘은 차량 환경에서 높은 정확도를 나타내는 광역 최적화 (global optimization) 기반 Brox 알고리즘을 적용하였다. Brox 알고리즘의 에너지 범함수 (energy functional)를 최소화 하는 과정에서 생성되는 Euler-Lagrange 방정식을 풀기 위해 하드웨어 구현에 용이한 Cholesky factorization이 적용되었으며, 메모리 접근율 (memory access rate)를 줄이기 위해 시프트 레지스터 뱅크 (shift register bank)를 도입하였다. 하드웨어 구현은 Verilog-HDL을 사용하였으며, FPGA 기반 설계 및 검증이 수행되었다. 제안된 광학흐름추정기는 40.4K개의 logic slice 및 155개의 DSP48s, 11,290 Kbit의 block memory로 구현되었다.

[Abstract]

In this paper, the design and implementation results of the optical flow estimator (OFE) for moving object detection (MOD) in advanced driver assistance system (ADAS). In the proposed design, Brox's algorithm with global optimization is considered, which shows the high performance in the vehicle environment. In addition, Cholesky factorization is applied to solve Euler-Lagrange equation in Brox's algorithm. Also, shift register bank is incorporated to reduce memory access rate. The proposed optical flow estimator was designed with Verilog-HDL, and FPGA board was used for the real-time verification. Implementation results show that the proposed optical flow estimator includes the logic slices of 40.4K, 155 DSP48s, and block memory of 11,290Kbits.

Key word : Advanced driver assistance system, Moving object detection, Optical flow estimation.

http://dx.doi.org/10.12673/jant.2015.19.6.544

Received 3 December 2015; **Revised** 8 December 2015
Accepted (Publication) 12 December 2015 (30 December 2015)

***Corresponding Author; Yunho Jung**

Tel: +82-2-300-0133
E-mail: yjung@kau.ac.kr

Ⅰ. 서 론

최근 자동차업계에서는 기술 발전으로 운전 환경의 돌발 상황에서 운전자의 판단과 조작을 돕는 보조수단에 대한 수요가 늘어남에 따라, 첨단 운전자 보조 시스템 (ADAS; advanced driver assistance system)에 대한 연구 개발이 활발히 진행되고 있다. ADAS 시스템에 활용되는 기술으로는 BSD (blind spot detection), LCA (lane change assistance) 등이 있는데, 해당 기술들을 구현하기 위해서는 영상처리, 레이더 (radar), 라이다 (laser radar) 기법 등이 적용될 수 있다. 이 중 영상처리 기반 기술은 사용할 수 있는 알고리즘이 다양하며, 직관적이고 여타 기술에 비해 저렴하다는 장점이 있어 다양하게 응용되고 있다 [1].

영상처리 기반 ADAS는 보편적으로 이동 객체 검출 (MOD; moving object detection) 기능을 수행하여 운전자를 보조하며, MOD에 활용되는 알고리즘으로는 대표적으로 FD (frame difference), GMM (Gaussiann mixture model) 기반 BS (background subtraction), 광학흐름추정 (OFE; optical flow estimation) 기법이 존재한다 [2], [3]. FD 알고리즘은 입력된 화면과 이전 화면의 차이값으로 이동 객체를 검출하며, 가장 간단한 구현 복잡도를 지니나 검출 성능이 매우 낮은 문제점이 존재한다. GMM 기반 BS 알고리즘은 각 화소의 분포를 다수의 가우시안 혼합 분포로 정의하여 영상이 입력됨에 따라 분포를 갱신함으로써 배경을 추정한다. 이 후, 추정된 배경과 입력된 화면의 차이로 이동 객체를 검출한다. GMM 기반 BS 알고리즘은 고정 카메라 환경에서 우수한 성능을 지원하지만, 차량 환경과 같은 이동 카메라 환경에서는 성능이 크게 저하되는 특성을 지닌다. 이에 반해 광학흐름추정 알고리즘은 각 화소의 움직임을 추적하며 화소 이동의 크기로 이동 객체를 추정하며, 이 특성 때문에 이동 카메라 환경에서 GMM 기반 BS 알고리즘이나 FD 알고리즘에 비해 우수한 검출 성능을 지원한다.

광학흐름추정 알고리즘은 국소 지역의 움직임이 동일하다는 전제 하에 움직임 벡터 (u,v)를 추정하는 국소 최적화 (local optimization) 기반 Lukas-Kanade 알고리즘과 frame 전체에 대한 에너지 범함수 (energy functional)를 정의하고 이 에너지 범함수를 최소화하는 (u,v)를 연산하는 광역 최적화 (global optimization) 기반 Horn-Schunck 알고리즘으로 분류되며, 이 두 기법이 발표된 이후 이를 보완한 다양한 알고리즘이 출현되었다 [4]-[10]. Horn- Schunck 알고리즘은 (u,v) 뿐만이 아닌 픽셀 자체의 밝기 값에도 영향을 받는다는 단점이 존재하여, 이를 보완하기 위해 여러 변형된 알고리즘이 제안되었으며, 이 중 Brox 알고리즘은 Horn-Schunck 알고리즘의 gradient 계수를 화소의 밝기 값에 제한시키지 않는다는 보완점을 제시하여 자동차 환경에서는 여타 알고리즘에 비해 높은 정확도를 보인다 [11]. 본 논문에서는 이러한 Brox 알고리즘의 효율적인 하드웨어 구조 및 FPGA 기반 구현 결과를 제시한다.

본 논문의 구성은 다음과 같다. 2장에서 광역 최적화 기반 광학흐름추정 알고리즘 중 대표적으로 활용되는 Brox 알고리즘을 설명하고, 3장에서는 제안된 하드웨어 구조 설계 결과를 제시한다. 4장에서는 FPGA 기반 구현 및 검증 결과에 대해 다루며, 마지막으로 5장에서는 본 논문의 결론을 맺는다.

Ⅱ. Brox의 광학흐름추정 알고리즘

2-1 기본 전제

각 화소의 명암 값이 이동한 다음 화면에서도 같은 값을 가진다는 전제가 광학흐름추정에서 적용된다. 식 (1)은 해당 개념인 항상성 을 나타낸다.

$$I(x,y,t) = I(x+u, y+v, t+1) \tag{1}$$

식 (1)에서 적용된 전제만으로는 작은 밝기 변화에서 정확한 광학흐름추정이 용이하지 않으므로, 식 (1)에서 편미분을 적용한 수식 또한 이용하게 된다. 식 (2)는 이러한 항상성 변화도에 대한 수식을 보여준다.

$$\nabla I(x,y,t) = \nabla I(x+u, y+v, t+1) \tag{2}$$

여기서, ∇는 편미분 연산자 $(\partial_x, \partial_y)^T$이다.

일반적으로 근접한 화소는 유사한 움직임을 가진다. 벡터의 평탄성을 의미하는 이 개념은 근접한 화소의 움직임이 동일하다는 의미이며, 식 (3)으로 표현된다.

$$|\nabla u|^2 + |\nabla v|^2 = 0 \tag{3}$$

2-2 에너지 범함수

식 (4)는 광역 최적화 기반 광학흐름추정 알고리즘의 기본적인 에너지 범함수를 나타내며, 식 (4)-(6)는 식 (1)-(3)에서 언급된 전제로 표현된, Brox algorithm의 E_{Data} 항과 E_{Smooth} 항을 나타낸다.

$$E(u,v) = E_{Data} + \alpha E_{Smooth} \tag{4}$$

$$E_{Data} = \int_{\Omega} \Psi(|I(\chi+w) - I(\chi)|^2 + \gamma|\nabla I(\chi+w) - \nabla I(\chi)|^2) d\chi \tag{5}$$

$$E_{Smooth} \equiv \int_{\Omega} \Psi(|\nabla u|^2 + |\nabla v|^2) d\chi \tag{6}$$

여기서, $\Psi(s^2) = \sqrt{s^2+\epsilon^2}$ 는 제곱항의 영향을 상쇄하기 위해 도입되었으며, ϵ의 값은 일반적으로 10^{-3}와 같이 작은 값을 사용한다. χ와 ω는 $\chi = (x,y,t)^T$, $\omega = (u,v,1)^T$ 이고, γ와 α는 weight parameter를 각각 나타낸다. E_{Data} 항은 식 (1)의 항상성과

식 (2)의 항상성 변화도의 합으로 구성되며, E_{Smooth} 항은 평탄성 값으로 구성된다. 에너지 범함수 $E(u,v)$가 최소값을 가질 때의 (u,v) 값이 식 (4)의 해가 된다.

2-3 Euler-Lagrange 방정식

식 (4)의 최소값을 만족시키는 벡터 (u,v)를 찾기 위해서 Euler-Lagrange 방정식이 적용된다 [12]. 식 (7)과 (8)은 식 (4)에 Euler-Lagrange 방정식을 적용한 결과를 보여준다.

$$\Psi'(I_z^2+\gamma(I_{xz}^2+I_{yz}^2))(I_xI_z+\gamma(I_{xx}I_{xz}+I_{xy}I_{yz})) - \alpha\cdot div(\Psi'(|\nabla_3 u|^2+|\nabla_3 v|^2)\nabla_3 u)=0 \quad (7)$$

$$\Psi'(I_z^2+\gamma(I_{xz}^2+I_{yz}^2))(I_yI_z+\gamma(I_{yy}I_{yz}+I_{xy}I_{xz})) - \alpha\cdot div(\Psi'(|\nabla_3 u|^2+|\nabla_3 v|^2)\nabla_3 v)=0 \quad (8)$$

여기서 I_x, I_y, I_z, I_{xx}, I_{xy}, I_{yy}, I_{xz}, 그리고 I_{yz}는 다음과 같다.

$$I_x=\partial_x I(\chi+w) \quad (9)$$
$$I_y=\partial_y I(\chi+w) \quad (10)$$
$$I_z=I(\chi+w)-I(\chi) \quad (11)$$
$$I_{xx}=\partial_{xx} I(\chi+w) \quad (12)$$
$$I_{xy}=\partial_{xy} I(\chi+w) \quad (13)$$
$$I_{yy}=\partial_{yy} I(\chi+w) \quad (14)$$
$$I_{xz}=\partial_x I(\chi+w)-\partial_x I(\chi) \quad (15)$$
$$I_{yz}=\partial_y I(\chi+w)-\partial_y I(\chi) \quad (16)$$

2-4 비선형성 소거

식 (7)과 (8)에서 나타난 Euler-Lagrange 방정식은 (u,v)에 대해 높은 비선형성을 지닌다. 이를 선형화하기 위해 다음의 방법을 적용한다. $w^k=(u^k,v^k,1)^T$, $k=0,1,2$이며 $w^0=(0,0,1)^T$라 한다면 식 (17)과 식 (18)으로 전개된다.

$$\Psi'((I_z^{k+1})^2+\gamma((I_{xz}^{k+1})^2+(I_{yz}^{k+1})^2)) \cdot(I_x^k I_z^{k+1}+\gamma(I_{xx}^k I_{xz}^{k+1}+I_{xy}^k I_{yz}^{k+1})) - \alpha\cdot div(\Psi'|\nabla u^{k+1}|^2+|\nabla v^{k+1}|^2)\nabla u^{k+1})=0 \quad (17)$$

$$\Psi'((I_z^{k+1})^2+\gamma((I_{xz}^{k+1})^2+(I_{yz}^{k+1})^2)) \cdot(I_y^k I_z^{k+1}+\gamma(I_{yy}^k I_{yz}^{k+1}+I_{xy}^k I_{xz}^{k+1})) - \alpha\cdot div(\Psi'|\nabla u^{k+1}|^2+|\nabla v^{k+1}|^2)\nabla v^{k+1})=0 \quad (18)$$

여기서 $(u^k,v^k)=(u^k+du^k,v^k+dv^k)$이며, 미지수 벡터 (du^k,dv^k)를 구하고, 식 (17)과 식 (18)에서 남아있는 비선형성을 소거하기 위해 미분 계수에 Taylor 확장을 수행한다.

$$I_z^{k+1}\approx I_z^k+I_x^k du^k+I_y^k+dv^k \quad (19)$$
$$I_{xz}^{k+1}\approx I_{xz}^k+I_{xx}^k du^k+I_{xy}^k dv^k \quad (20)$$
$$I_{yz}^{k+1}\approx I_{yz}^k+I_{xy}^k du^k+I_{yy}^k dv^k \quad (21)$$

미분 계수를 Taylor 확장으로 전개한 식 (19), (20), (21)과 $du^{k,0}=0$, $dv^{k,0}=0$를 식 (17)과 식 (18)에 대입하면 식 (22)와 식 (23)을 얻는다.

$$\begin{aligned}(\Psi')_{Data}^{k,l}\cdot(&I_x^k(I_z^k+I_x^k du^{k,l+1}+I_y^k dv^{k,l+1})\\&+\gamma(I_{xx}^k(I_{xz}^k+I_{xx}^k du^{k,l+1}+I_{xy}^k dv^{k,l+1})\\&+I_{xy}^k(I_{yz}^k+I_{xy}^k du^{k,l+1}+I_{yy}^k dv^{k,l+1})))\\&-\alpha\cdot div((\Psi')_{Smooth}^{k,l}\nabla(u^k+du^{k,l+1}))=0\end{aligned} \quad (22)$$

$$\begin{aligned}(\Psi')_{Data}^{k,l}\cdot(&I_y^k(I_z^k+I_x^k du^{k,l+1}+I_y^k dv^{k,l+1})\\&+\gamma(I_{yy}^k(I_{yz}^k+I_{xy}^k du^{k,l+1}+I_{yy}^k dv^{k,l+1})\\&+I_{xy}^k(I_{xz}^k+I_{xx}^k du^{k,l+1}+I_{xy}^k dv^{k,l+1})))\\&-\alpha\cdot div((\Psi')_{Smooth}^{k,l}\nabla(u^k+dv^{k,l+1}))=0\end{aligned} \quad (23)$$

여기서 $(\Psi')_{Data}^{k,l}$, $(\Psi')_{Smooth}^{k,l}$는 식 (24)와 식 (25)와 같이 전개된다.

$$\begin{aligned}(\Psi')_{Data}^{k,l}=\Psi'((&I_z^k+I_x^k du^{k,l}+I_y^k dv^{k,l})^2\\&+\gamma((I_{xz}^k+I_{xx}^k du^{k,l}+I_{xy}^k dv^{k,l})^2\\&+(I_{yz}^k+I_{xy}^k du^{k,l}+I_{yy}^k dv^{k,l})^2))\end{aligned} \quad (24)$$

$$(\Psi')_{Smooth}^{k,l}=\Psi'(|\nabla_3(u^k+du^{k,l})|^2+|\nabla_3(v^k+dv^{k,l})|^2) \quad (25)$$

식 (22)와 식 (23)을 행렬 형태로 변환하면 식 (26)과 같이 변환된다.

$$\begin{aligned}&(\Psi')_{Data}^{k,l}\cdot\begin{pmatrix}I_x^kI_x^k+\gamma(I_{xx}^kI_{xx}^k+I_{xy}^kI_{xy}^k) & I_x^kI_y^k+\gamma(I_{xx}^kI_{xy}^k+I_{xy}^kI_{yy}^k)\\ I_y^kI_x^k+\gamma(I_{yy}^kI_{yx}^k+I_{xy}^kI_{xx}^k) & I_y^kI_y^k+\gamma(I_{yy}^kI_{yy}^k+I_{xy}^kI_{xy}^k)\end{pmatrix}\cdot\begin{pmatrix}du^{k,l+1}\\dv^{k,l+1}\end{pmatrix}\\&=-(\Psi')_{Data}^{k}\cdot\begin{pmatrix}I_x^kI_z^k+\gamma(I_{xx}^kI_{xz}^k+I_{xy}^kI_{yz}^k)\\I_y^kI_z^k+\gamma(I_{yy}^kI_{yz}^k+I_{xy}^kI_{xz}^k)\end{pmatrix}\\&\quad+\alpha\cdot\begin{pmatrix}div((\Psi')_{Smooth}^{k}\nabla_3(u^k+du^{k,l+1}))\\div((\Psi')_{Smooth}^{k}\nabla_3(u^k+du^{k,l+1}))\end{pmatrix}\end{aligned} \quad (26)$$

식 (26)을 한 프레임 내 전체 픽셀에 대해 행렬 형태로 변환하여 연산을 진행한다 [13]. 해당 변환을 통하여 전체 화소수가 wh일 때, $2wh\times 2wh$의 크기를 가지는 행렬을 **A**, du, dv로 구성된 벡터를 **x**, 그리고 식 (26)의 오른쪽 항으로 표기된 벡터를 **b**라 한다면 $\mathbf{Ax}=\mathbf{b}$ 형태의 행렬식이 생성된다.

이와 같이 생성된 행렬식의 해를 구하기 위한 방법으로는 GS

(Gauss-Seidel) 기법, SOR (successive over relaxation) 기법, LR (local relaxation) 기법 등이 존재하나, 행렬 **A**는 Hermitian positive-definite 조건을 만족하기 때문에, 연산 처리량이 일정하여 실시간 하드웨어 설계 및 구현이 용이한 Cholesky factorization를 이용하여 해를 찾는다 [14]. 식 (27)과 식 (28)은 Cholesky factorization의 연산 과정을 나타낸다.

$$\mathbf{Ax} = \mathbf{LL^*x} = \mathbf{Lz} = \mathbf{b} \tag{27}$$

$$\mathbf{z} = \mathbf{L^*x} \tag{28}$$

2-5 Pyramid algorithm

Brox의 광학흐름추정 알고리즘은 한 프레임에 한 화소 이상의 움직임 벡터 (u,v)를 추적하는 과정에서 에너지 범함수가 국소 최저치에 빠질 수 있기 때문에 pyramid 알고리즘을 적용한다. 기존의 이미지를 $1/2^n$ 크기로 축소하여 (u,v)를 추적한 후, 추정한 (u,v)로 $1/2^{n-1}$ 크기로 축소한 이미지를 뒤틀어 이미지의 차이를 줄인다. 이를 반복하며 추정한 (u,v)를 누적하여 전체 화소의 움직임 (u,v)를 추정한다.

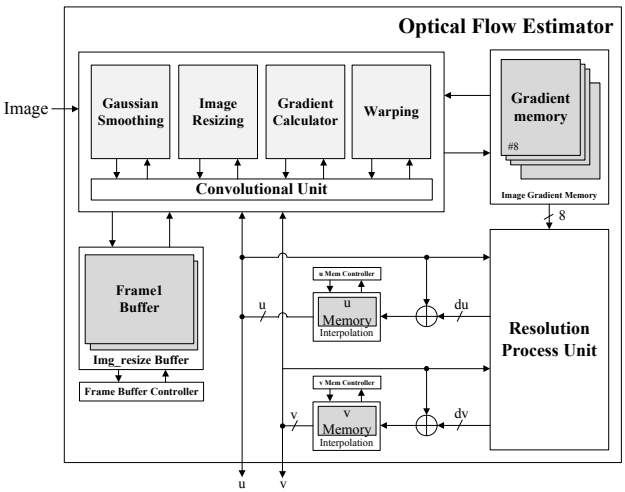

그림 1. 제안된 광학흐름추정기의 구조도
Fig. 1. Hardware structure of the proposed optical flow estimator.

Ⅲ. 하드웨어 구조 설계

3-1 Top block 구조

그림 1과 같이 설계된 광학흐름추정기는 가우시안 이미지 크기 변환 (Gaussian rescaling), 변화도 연산 (gradient calculation) 및 왜곡 (warping)을 수행하는 CU (convolutional unit)와 연산결과를 이용하여 선형 분해 (linear factorization)를 수행하는 RPU (resolution process unit)로 구성된다. RPU에서 Cholesky factorization을 통하여 Euler-Lagrange equation이 적용된 에너지 범함수의 비선형성을 제거하고 벡터 값 (u,v)를 산출한다. 그림 2는 제안된 광학흐름추정기의 타이밍도를 보여준다. 광학흐름추정기의 연산은 입력된 이미지의 가우시안 평활화 (Gaussian smoothing), 이미지 크기 변환 (image resizing) 및 변화도 연산을 진행한 후 RPU에서의 연산을 수행한다. RPU의 1st pyramid와 2nd pyramid 내 inner iteration은 각각 242.88K cycles, 1.67M cycles이 소요되며, RPU의 연산결과로 얻은 du, dv를 통해 왜곡을 수행하며 왜곡된 이미지를 사용하여 다음 pyramid stage에서의 연산을 수행한다. 하나의 프레임에 대한 연산 과정에서 약 7.8M cycles이 소요되므로 250 MHz 환경에서 30 fps (frames per second)의 실시간 처리가 가능하다.

3-2 Convolution Unit 구조

그림 3은 설계된 CU의 구조도를 보여준다. 가우시안 평활화, 변화도 연산은 convolution 연산을 수행하며, 각각의 연산에 사용되는 filter는 좌우 대칭 형태이기 때문에 곱셈 연산 이전에 덧셈 및 뺄셈 연산을 수행하여 곱셈기의 수를 줄인다. 이미지 크기

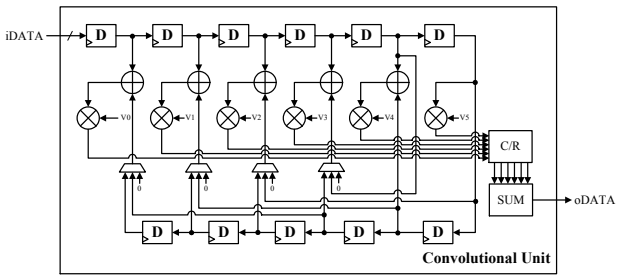

그림 3. CU의 하드웨어 구조도
Fig. 3. Hardware structure of CU.

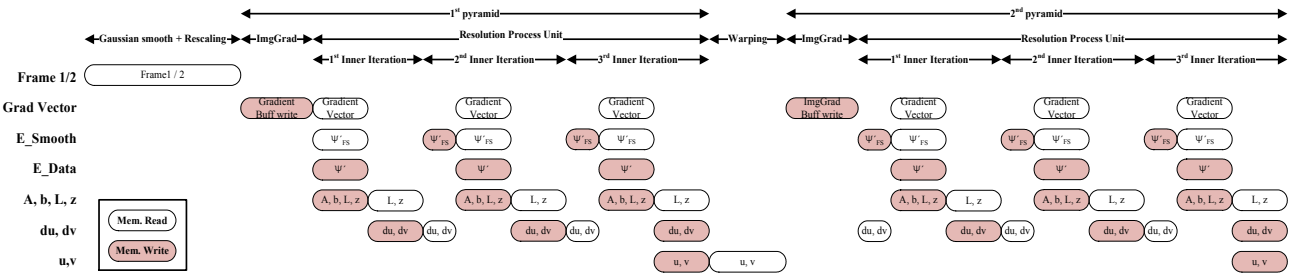

그림 2. 제안된 광학흐름추정기의 타이밍도
Fig. 2. Timing diagram of the optical flow estimator.

변환, 왜곡 연산은 일반적인 보간법을 사용하여 값을 연산한다. 이 연산들은 pipelining이 불가능하므로 CU를 공유하여 연산을 수행한다. 공유 구조를 취함으로서 곱셈기를 8개, 덧셈기를 13개 줄이게 된다.

3-3 Resolution process unit 구조

그림 4와 같이 설계된 RPU는 광학흐름추정기의 에너지 범함수를 최소화하는 (u,v)의 해를 구하는 연산을 수행한다. E_Data calculator와 E_Smooth calculator는 각각 에너지 범함수의 E_{Data} 및 E_{Smooth} 항을 연산하며 A/b calculator는 각 연산 결과를 사용하여 Euler-Lagrange equation의 해를 구하기 위한 행렬 형태로 값을 산출한다. L-matrix calculator 및 backward substituter는 A/b calculator에서 생성한 행렬 연산의 해를 구하는 연산을 수행한다.

3-4 E_Data calculator 구조

RPU의 하위블록인 E_Data calculator에서는 식 (14)로 표현된 $(\Psi')^{k,l}_{Data}$을 구현한다. 그림 5는 E_Data calculator의 구조도를 보여주며, 식 (24)에서 나타나는 $(I_1+I_2du+I_3dv)^2$ 형태의 유사한 식이 반복되기 때문에 공유 구조를 취하여 6개의 곱셈기와 6개의 덧셈기를 줄인다.

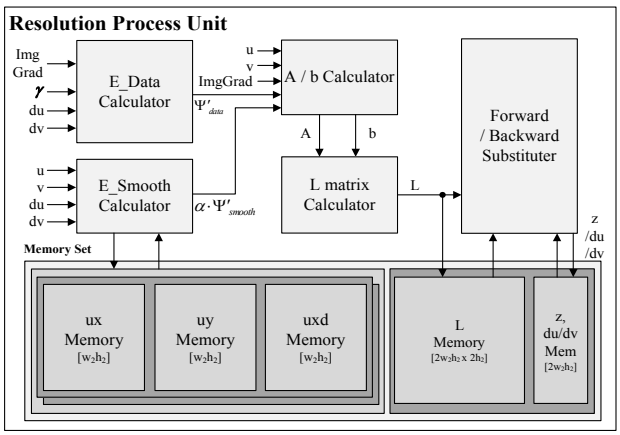

그림 4. RPU의 하드웨어 구조도
Fig. 4. Hardware structure of RPU.

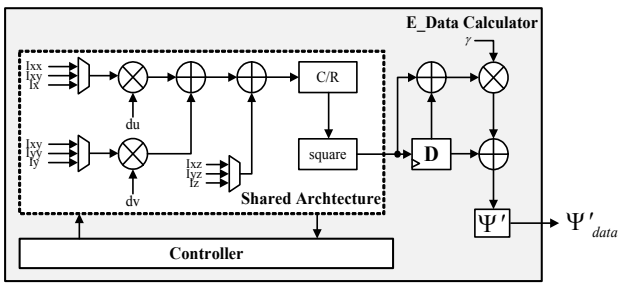

그림 5. E_Data Calculator의 구조도
Fig. 5. Hardware structure of E_Data calculator.

3-5 E_Smooth calculator 구조

그림 6는 E_Smooth calculator의 구조도를 보여준다. 식 (25)의 $(\Psi')^{k,l}_{Smooth}$ 값을 나타내기 위해 $u+du$, $v+dv$의 편미분을 x, y 축으로 진행한다. (u,v)를 각각 x, y축으로 편미분한 값 u_x, u_y, v_x, v_x의 메모리는 각각의 값에 Ψ' 연산이 적용된 $u_{x,pd}$, $u_{y,pd}$, $v_{x,pd}$, $v_{y,pd}$ 값을 다시 저장하여 메모리 요구량을 576 kbit에서 388 kbit로 줄인다.

3-6 A/b calculator 구조

그림 7은 A/b calculator의 구조도를 제시한다. **A** 행렬의 값을 연산하는 **A** part에서는 식 (26)에서 제시된 **A** 행렬의 $I^k_xI^k_x+\gamma(I^k_{xx}I^k_{xx}+I^k_{xy}I^k_{xy})$를 나타내는 $a_{1,1}$, $I^k_yI^k_y+\gamma(I^k_{yy}I^k_{yy}+I^k_{xy}I^k_{xy})$를 나타내는 $a_{2,2}$가 유사한 형태이기 때문에 공유 구조를 취한다. 또한 $a_{1,2}$, $a_{2,1}$ 값인 $I^k_yI^k_x+\gamma(I^k_{yy}I^k_{yx}+I^k_{xy}I^k_{xx})$, $I^k_xI^k_y+\gamma(I^k_{xx}I^k_{xy}+I^k_{xy}I^k_{yy})$는 실제로 동일한 값이기 때문에 하나의 연산기만을 사용한다. 마찬가지로 **b** 벡터의 값을 구성하는 b_1, b_2의 형태가 유사하므로 공유 구조를 가진다.

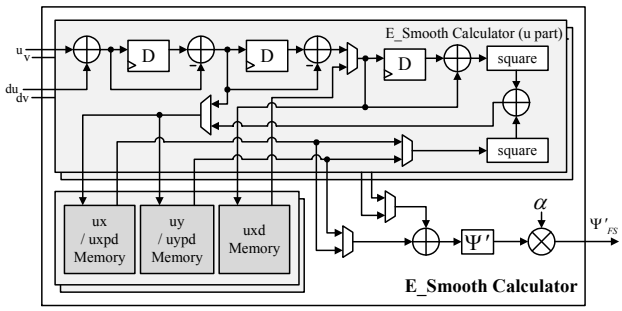

그림 6. E_Smooth Calculator의 구조도
Fig. 6. Hardware structure of E_Smooth calculator.

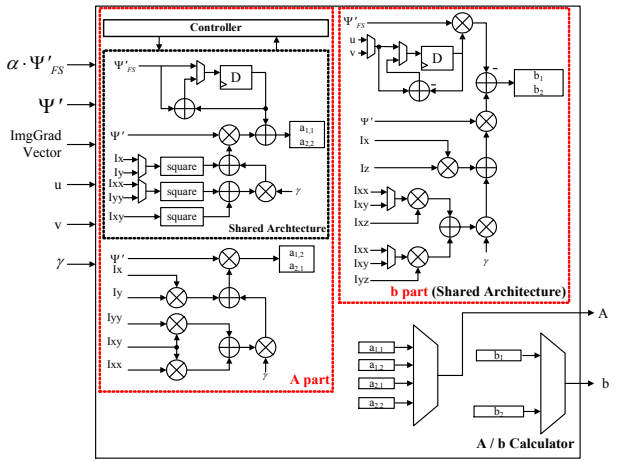

그림 7. A/b Calculator의 구조도
Fig. 7. Hardware structure of A/b calculator.

3-7 L-matrix Calculator

L-matrix calculator에서는 A/b calculator에서 생성한 행렬 및 벡터의 값을 사용하여 (u, v)를 구하는 연산을 수행한다. 이 과정에서 역행렬 연산은 A 행렬이 Hermitian positive-definite 조건을 만족하므로 Cholesky factorization을 사용한다. 이미지의 화소 수가 N_f라고 할 경우 A 행렬의 크기가 $2N_f \times 2N_f$이기 때문에 메모리 접근 회수가 197.74M cycles로 과도하게 증가하며, 이를 줄이기 위해 시프트 레지스터 뱅크(shift register bank)를 도입한다. 그림 8은 시프트 레지스터 뱅크를 적용한 L-Matrix Calculator의 구조도를 보여준다. 레지스터 뱅크는 각 register 값이 전달되는 시프트 레지스터를 나타낸다. 또한 forward substitution은 Cholesky factorization의 **A=LL*** 연산은 **Lz=b** 연산과 동일한 순서의 값을 요구하기 때문에 동시에 진행한다. 생성된 **L**값은 **L** memory에, **z**값은 z memory에 저장하게 된다. Backward substitution은 시프트 레지스터 뱅크에 포함된 곱셈기를 공유하여 사용하게 된다. 시프트 레지스터 뱅크를 도입한 후 전체 메모리 접근 회수는 4.45M cycles으로 도입 이전에 비해 97.75% 감소한다.

3-8 Forward / backward substituter

Forward / backward substituter는 L-matrix calculator에서 연산된 **L**을 사용하여 **z**값을 구하는 forward substitution, 그리고 연산된 **z**값과 **L***값으로 **x**값을 구하는 backward substitution 연산을 수행한다.

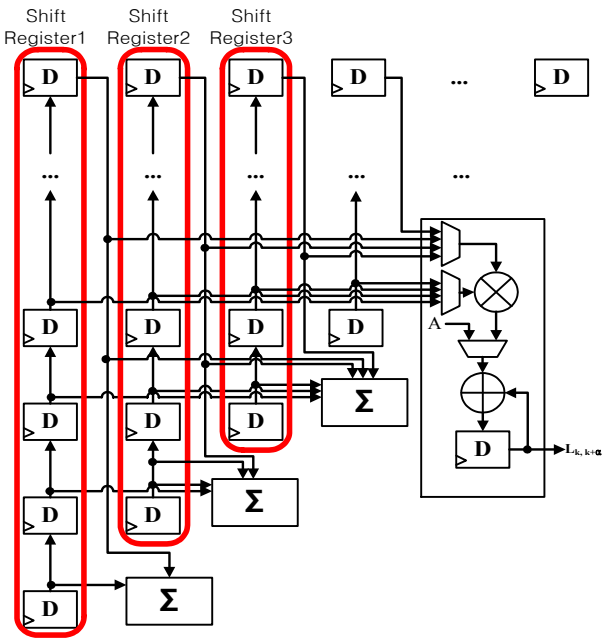

그림 8. 제안된 시프트 레지스터 뱅크가 적용된 L-Matrix calculator의 구조도
Fig. 8. Hardware structure of L-Matrix calculator with shift register bank.

표 1. 제안된 이동 객체 검출기의 복잡도
Table 1. Complexity of the proposed optical flow estimator.

Block		FPGA logic Slices	DSP48s	Block RAM
C.U		112	6	-
R.P.U	E_Data Calc.	462	17	-
	E_Smooth Calc.	384	15	482*K*bit
	A/b Calc.	173	16	-
	L-matrix Calc.	38.9*K*	101	-
	R.P.U Memory	-	-	10,656*K*bit
Frame2 Buffer		-	-	128*K*bit
ImgGrad Mem		-	-	1024*K*bit
Total		40.4*K*	155	11,290*K*bit

그림 9. 제안된 광학흐름추정기의 이동객체검출 성공 예시
Fig. 9. Success examples for the moving object detection by the proposed optical flow estimator.

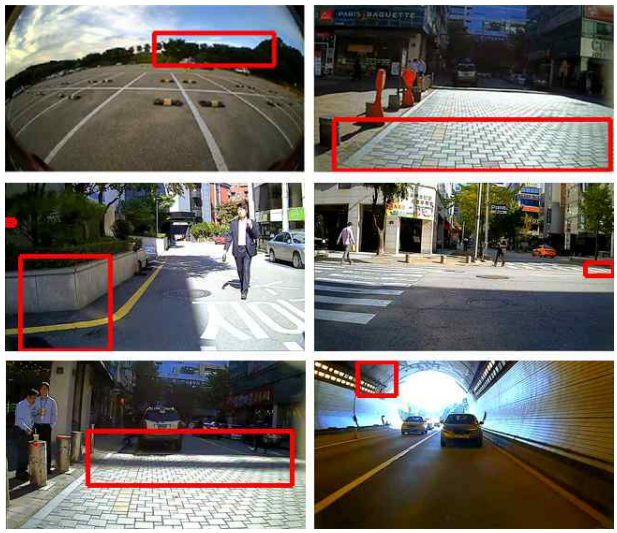

그림 10. 제안된 광학흐름추정기의 이동객체검출 실패 예시
Fig. 10. Failure examples for the moving object detection by the proposed optical flow estimator.

Ⅳ. FPGA 기반 구현 및 검증

4-1 복잡도 분석

제안된 이동 객체 검출기는 Verilog-HDL을 이용하여 설계 후, Xilinx 기반 Virtex7 FPGA device에서 구현 및 검증하였다. 구현 결과 표 1과 같이, 제안된 하드웨어는 총 40.4K개의 logic slices와 총 155개의 DSP48s, 총 11,290 Kbit의 block memory로 구현 가능함을 확인하였다.

4-2 차량 환경에 대한 성능 분석 결과

그림 9와 10은 차량 환경에 대한 광학흐름추정기의 성능을 나타낸다. 화면 내에서 이동 객체를 검출한 경우를 성공으로 판정하였을 경우, 취득한 영상 기준으로 약 95% (13,727 프레임 중에서 13,021 프레임 검출)의 검출율을 보인다. 카메라의 움직임이 작은 환경에서는 그림 9와 같이 이동 객체 검출율이 높게 나타나는 반면, 그림 10과 같이 카메라의 움직임이 큰 환경에서는 배경 전체가 움직이면서 이동 벡터가 크게 나타나 오검출 확률이 증가한다. 이와 같이 카메라의 움직임이 큰 상황에서는 객체 추적을 통해 성능 개선이 가능할 것으로 기대되며, 추후 관련된 연구가 필요할 것으로 판단된다.

Ⅴ. 결 론

본 논문에서는 광학흐름추정 알고리즘 기반의 이동 객체 검출기의 하드웨어 구조와 구현 결과를 제시하였다. 설계된 이동 객체 검출기는 FPGA 기반 실시간 검증되었으며, SXGA급 해상도의 30 fps 영상에 대해 250 MHz의 동작 주파수로 실시간 처리가 가능하다. 카메라의 움직임이 큰 상황에서는 객체 추적에 대한 과정이 존재하지 않기에 성능이 저하되는 현상을 보였으며. 이를 보완하기 위한 추가적인 알고리즘 연구를 추후 진행할 예정이다.

감사의 글

본 논문은 산업통상자원부 및 한국산업기술평가관리원의 산업원천기술개발사업 (10052009)의 일환으로 수행되었음.

참고 문헌

[1] D. Cheda, D. Ponsa, “Camera egomotion estimation in the ADAS context,” in *IEEE Conference on Intelligent Transportation System,* Funchal: Portugal, pp.1415-1420, Sep. 2010.

[2] Z. Chaohui, D. Xiaohui, X. shuoyu, S. Zheng, “An improved moving object detection algorithm based on frame difference and edge detection,” in *Fourth International Conference on Image and Graphics*, Sichuan: China, pp. 519-523, Aug. 2007.

[3] C. Stauffer, W. E. L Grimson, “Adaptive background mixture models for real-time tracking,” in *IEEE Computer Society Conference on CVPR,* Fort Collins: CO, Vol. 2, pp. 246-252, Jun. 1999.

[4] B. Lucas, T. Kanade, “An iterative image restoration technique with an application in stereo vision,” in *Proceedings of International Joint Conference on Artificial Intelligence*, Vancouver: Canada, Vol. 81, pp. 674-679, Aug. 1981.

[5] B. Horn, B. Schunk, “Determining optical flow,” in *Technical Symposium East of International Society for Optics and Photonics*, Washington: D.C., pp. 185-203, Nov. 1981.

[6] C. Schnorr, “Segmentation of visual motion by minimizing convex non-quadratic functionals,” in *Pattern Recognition, Conference A: Computer Vision and Image Processing, Proceedings of the 12th IAPR International Conference,* Jerusalem: Israel, Vol. 1, pp. 661-663, Oct. 1994.

[7] R. Deriche, P. Kornprobst, G. Aubert, “Optical-flow estimation while preserving its discontinuities: A variational approach,” in *Second Asian Conference on Computer Vision*, Singapore: Singapore, pp. 69-80, Dec. 1995.

[8] L. A. Leon, J. E. Monreal, M. Lefebure, J. S. Perez, “A PDE model for computing the optical flow,” *Proceedings of Congreso de Ecuaciones Diferenciales y Aplicaciones XVI*, Gran Canaria: Spain, pp. 1349-1356, Sep. 1999.

[9] H. Oh, H. Lee, and J. H. Baek, “Moving Object Tracking in UAV Video using Moving Estimation,” *Journal of Advanced Navigation Technology*, Vol. 10, No. 4, pp. 400-405, Dec. 2006.

[10] D. Kim, S. Rho, and E. Hwang, “Real-time Multi-Objects Recognition and Tracking Scheme,” *Journal of Advanced Navigation Technology*, Vol. 16, No. 2, pp. 386-398, Apr. 2012.

[11] T. Brox, Andres Bruhn, Nils Papenberg, Joachim Wickert, “High accuracy optical flow estimation based on a theory for warping,” in *8th European Conference on Computer Vision*, Prague: Czech Republic, pp. 25-36, 2004.

[12] O. P. Agrawal, “Formulation of Euler–Lagrange equations for fractional variational problems,” *Journal of Mathematical Analysis and Applications*, Vol. 272, pp. 368-379, Aug. 2002.

[13] R. J. LeVeque, Finite difference methods for ordinary and

partial differential equations, *Society for Industrial and Applied Mathematics*, pp.35-46, 2007.

[14] J. Kim, C. Park, and Y. Lee, “A new nonlinear method for optical flow estimation,” *The Journal of Korean Institute of Communications and Information Sciences*, Vol. 26, No. 4, pp. 463-470, Apr. 2001.

윤 경 한 (Kyung-Han Yoon)
2014년 2월 : 한국항공대학교 전자 및 항공전자공학과 (공학사)
2014년 3월~현재 : 한국항공대학교 항공전자정보공학부 석사과정
※관심분야 : 영상처리 시스템, VLSI 신호처리, 영상처리 SoC

정 용 철 (Yong-Chul Jung)
2015년 8월 : 한국항공대학교 전자 및 항공전자공학과 (공학사)
2015년 9월~현재 : 한국항공대학교 항공전자정보공학부석사과정
※관심분야 : 영상처리 시스템, VLSI 신호처리, 영상처리 SoC

조 재 찬 (Jae-Chan Cho)
2015년 8월 : 한국항공대학교 전자 및 항공전자공학과 (공학사)
2015년 9월~현재 : 한국항공대학교 항공전자정보공학부 석사과정
※관심분야 : 영상처리 시스템, VLSI 신호처리, 영상처리 SoC

정 윤 호 (Yunho Jung)
1998년 2월 : 연세대학교 전자공학과
2000년 2월 : 연세대학교 전기전자공학과 (공학석사)
2005년 2월 : 연세대학교 전기전자공학과 (공학박사)
2005년 ~ 2007년 : 삼성전자 책임연구원
2007년 ~ 2008년 : 연세대학교 연구교수
2008년 ~ 현재 : 한국항공대학교 부교수
※관심분야 : 무선 통신 시스템, 항공통신 시스템, 영상처리 시스템, 모뎀 SoC, 영상처리 SoC