논문 2006-43SD-9-7

Foreground 객체 추출을 위한 실시간 SoC 설계

(A Real-time SoC Design of Foreground Object Segmentation)

김 지 수*, 이 태 호*, 이 혁 재**

(Jisu Kim, Tae-Ho Lee, and Hyuk-Jae Lee)

요 약

최근 개발된 영상 압축 표준인 MPEG-4 Part 2는 임의의 영상 객체를 처리할 수 있는 최신의 기능을 포함한다. 이러한 기능을 지원하기 위해서는 효과적인 객체 추출 기술이 요구된다. 본 논문에서는 영상 내에서 실시간으로 객체를 추출해 낼 수 있는 알고리즘을 제안한다. 제안된 알고리즘은 두 단계로 구성된다. 첫 번째 단계는 한 프레임의 영상을 시공간적 watershed transform을 이용하여 여러 영역으로 분할하는 것이고, 두 번째 단계는 분할된 영역 정보를 바탕으로 객체를 추출해내는 것이다. 실시간 처리를 위해서 제안된 알고리즘은 하드웨어와 소프트웨어로 분할하여 구현하고, 계산량이 집중된 연산 부분을 하드웨어 가속기를 사용하여 처리한다. 실험 결과 제안된 시스템은 QCIF 크기의 영상을 초당 15 frame 이상의 속도로 처리하면서도, 정확한 객체 추출 결과를 보였다.

Abstract

Recently developed MPEG-4 Part 2 compression standard provides a novel capability to handle arbitrary video objects. To support this capability, an efficient object segmentation technique is required. This paper proposes a real-time algorithm for foreground object segmentation in video sequences. The proposed algorithm consists of two steps: the first step that segments a video frame into multiple sub-regions using Spatio-Temporal Watershed Transform and the second step in which a foreground object segment is extracted from the sub-regions generated in the first step. For real-time processing, the algorithm is partitioned into hardware and software parts so that computationally expensive parts are off-loaded from a processor and executed by hardware accelerators. Simulation results show that the proposed implementation can handle QCIF-size video at 15 fps and extracts an accurate foreground object.

Keywords: MPEG-4, Hardware/software partitioning, Foreground object segmentation, Watershed transform, Real-time processing

I. 서 론

객체 기반 영상 처리는 실제 생활에 다양하게 사용될 가능성을 가지고 있다. 예를 들어, 뉴스 방송 등에서는 Chroma-key 방법을 사용하여 배경 화면과 인물을 별도로 생성한 후 이를 하나의 화면으로 합성하여 보여주기도 한다. 객체 기반 영상 처리의 실용화를 위하여 최신 비디오 표준인 MPEG-4 part 2[1]에서는 객체 기반 영상 압축 기술을 심도 있게 기술한다. 예를 들어, 영상 내에서 임의의 모양을 갖는 객체는 각각 그 모양과 텍스쳐로 표현하여 처리하도록 한다. 이러한 표준화 작업의 진전에도 불구하고 인간의 도움 없이 실시간으로 객체를 추출하는 기술이 아직까지도 충분히 발전되지 못하였기 때문에, 이러한 최신 영상 압축 기술은 아직 실용화와는 거리가 멀다고 여겨지고 있다. 이러한 한계를 극복하기 위하여 본 논문에서는 객체 추출을 위한 실시간 시스템을 제안하고자 한다.

이제까지 객체 추출을 위해서 많은 알고리즘들이 제안되었다. 그 중에서 공간적, 시간적 정보를 모두 사용하여 원하는 영역을 추출해 내는 방식에 관하여 활발한 연구가 진행되어 왔다[2]-[5]. 공간적 정보를 통해서 추출

* 학생회원, ** 정회원, 서울대학교 전기컴퓨터공학부
(School of Electrical Engineering and Computer Science, Seoul National University)
※ 본 연구보고서는 정보통신부 출연금 등으로 수행한 정보통신연구개발사업의 연구결과입니다.
※ 본 연구보고서의 내용을 발표할 때에는 반드시 정보통신부 정보통신연구개발사업의 연구 결과임을 밝혀야 합니다.
접수일자: 2006년7월1일, 수정완료일: 2006년8월18일

영역의 정확한 경계를 얻어내고, 시간적 정보는 추출하고자 하는 객체의 움직임을 찾아내고 추적한다. 이러한 알고리즘들은 정확도의 측면에서는 우수하지만, 계산의 복잡도가 크기 때문에 실시간의 처리가 어렵다. 또 다른 주요한 접근 방식으로는 배경 등록(background registration) 방식이 있다[6][7]. 이 방식들은 계산량은 적지만, 그 적용 범위가 제한적이다. 왜냐하면 카메라의 움직임이 없어야 하거나, 미리 저장된 배경 정보가 필요 하는 등의 제약 조건이 있기 때문이다.

본 논문에서 제안하는 알고리즘은 다음과 같은 조건을 만족시키는 것을 목표로 한다. 실시간 수행이 가능할 정도로 계산량이 적으면서, 비교적 정확한 결과를 얻을 수 있고, 실제 응용에 적용되기 위한 제약이 많지 않아야 한다. 이러한 알고리즘의 개발을 위해서 본 논문이 기여하는 부분은 크게 세 가지이다. 첫 번째로, 기존에 존재하는 알고리즘들을 효과적으로 재구성하여 계산량을 감소시킨 알고리즘을 제안한다. 본 논문의 두 번째 기여는 foreground 객체가 화면의 가운데에 위치한다고 가정하고, 이를 바탕으로 효과적인 foreground 객체 추출 알고리즘을 개발하였다. 세 번째 기여한 바는 효과적인 하드웨어/소프트웨어 분할 기법을 사용하여 알고리즘을 구현함으로써 저비용의 SoC 시스템의 구성을 가능하게 한다는 점이다. 이러한 접근 방식을 통하여 실시간 동작이 가능하면서도 동시에, 타 알고리즘과 비교하여 추출 결과의 차이가 크지 않은 결과를 얻을 수 있다.

본 논문의 구성은 다음과 같다. Ⅱ장에서는 전체 알고리즘의 개요에 대해서 서술하고, Ⅲ장과 Ⅳ장에서는 각각 영역 분할 알고리즘과 foreground 영역 추출 알고리즘에 대해서 자세하게 설명한다. Ⅴ장에서는 임베디드 시스템에 적합한 구현 방법에 대해서 다루고, Ⅵ장에서는 시뮬레이션 결과를 분석한다. 마지막으로 Ⅶ장에서는 결론을 제시한다.

Ⅱ. 알고리즘 개요

본 논문에서 제안하는 알고리즘은 크게 두 가지 단계로 구분된다. 첫 번째는 입력으로 들어온 영상 이미지를 분할하는 것이고, 두 번째는 영역 분할된 결과에서 foreground 영역을 추출해내는 것이다.

첫 번째 단계인 영상 분할은 watershed transform[8]을 기반으로 이루어진다. 일반적인 watershed transform은 그 결과가 과분할 된 상태로 나오게 되기 때문에, 이를 해결하기 위하여 적절한 선처리 및 후처리가 필요하다. 또한 기본적인 watershed transform은 공간적 정보만을 이용하여 영역을 분할하기 때문에, 동영상에서의 전후 영상 간의 관계, 즉, 시간적 정보를 이용할 수 없다. 본 논문에서는 단절된 하나의 이미지에서가 아니라 동영상에서의 영상 분할을 목표로 하므로, 영역 분할 시에 시간적 정보를 이용할 수 있도록 하였다. 따라서 전체 알고리즘 상에서 watershed transform은 두 가지 종류의 모드를 적응적으로 사용한다. 첫 번째는 공간적 정보만을 사용한 watershed transform이고, 두 번째는 공간적, 시간적 정보를 모두 사용한 watershed transform이다. 공간적 정보만을 이용하는 방식은 이전 프레임의 정보를 이용할 수 없을 경우에 사용되고, 그렇지 않은 경우에는 공간적, 시간적 정보를 모두 사용한다. 영상 압축 기술에서 흔히 사용하는 용어를 이용하여, 첫 번째 방식을 "I-type" 이라고 하고, 두 번째 방식을 "P-type" 이라고 한다. I-type 과정에서는 참고문헌 [5]에서와 유사한 과정을 사용하고, P-type 과정에서는 참고문헌 [6, 7]에서 사용되는 방법을 응용하여 전체 알고리즘의 계산량을 줄일 수 있도록 한다.

두 번째 단계인 foreground 영역 추출은, 첫 단계의 결과로 분할된 여러 영역 가운데에서 foreground 영역을 찾아내는 것이다. Foreground 영역을 추출하기 위해서, 본 논문에서는 foreground 영역이 영상 화면의 중앙에 위치하고 있다는 공간적 가정을 통해 foreground 영역을 판별해 낸다. 이러한 가정을 이용하여 연산량을 많이 감소시키면서도 foreground 영역을 비교적 정확하게 찾아낼 수 있다.

Ⅲ. 영역 분할

그림 1은 영역 분할 알고리즘의 흐름을 보여준다. 영역 분할 단계는 크게 6가지 과정으로 이루어지는데, 그 과정은 영상 평탄화, gradient 계산, 영상 변화 검출, marker 추출, watershed transform, 영역 병합의 순서로 이루어진다. Marker 추출은 영상 변화 검출 과정에서 판단한 결과에 따라 현재 영상이 I-type인지 P-type인지를 선택하여 수행한다. 영역 분할 알고리즘의 세부적 내용은 아래와 같다.

1. 영상 평탄화

영상 평탄화란 영상 내에서 급격하게 변화하는 부분을 완만하게 변화하도록 만들어주는 과정이다. 영상 평

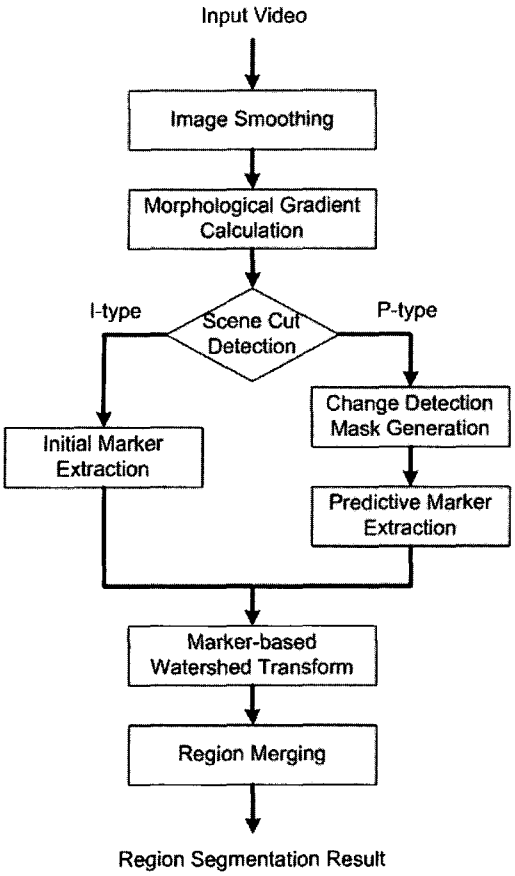


그림 1. 영역 분할 알고리즘의 흐름도
Fig. 1. The flow of the region segmentation algorithm.

탄화를 하는 목적은 noise나 미세한 부분의 정보를 제거하여 영상이 과분할 되는 것을 막기 위함이다. 이를 위해서 morphological open-close 방법[9]이 이용된다.

2. Gradient 계산

Gradient는 영상에서의 edge 정보를 수치로 나타낸 값이라고 볼 수 있다. Frame의 전체 pixel의 gradient값을 구하기 위하여 morphological gradient[10]를 사용한다. $f(x,y)$가 입력 이미지라고 하고, $\oplus$와 $\ominus$가 각각 morphological dilation과 erosion을 나타내며, B는 3×3 structuring element라고 하면, morphological gradient $G(f)$는 다음 식 (1)과 같이 표현된다.

$$G(f) = (f \oplus B) - (f \ominus B) \quad (1)$$

그림 2에서 (a)는 평탄화된 후의 입력 이미지에서 한 행의 pixel값들을 나타낸 것이고, (b)의 그래프는 그 입력 이미지로부터 gradient level을 구한 것이다.

3. 영상 변화 검출

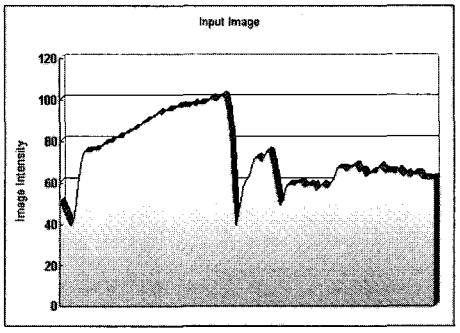
영상 변화 검출 단계에서는 이전 frame과 현재 frame의 차이를 구하여, 한 pixel에서 특정 값 이상의 차이가 존재할 경우 해당 pixel에서 변화가 있었다고 판단하고, 이를 사용하여 변화 검출 마스크 (change detection mask) 를 만든다. 이를 바탕으로 frame 전체에서의 변화 정도를 측정하여 현재 frame이 I-type인지 P-type인지 구분한다. 만약 마스크에 해당하는 픽셀의 개수가 특정 값 이상이면 I-type 과정을 선택하고, 아니면 P-type 과정을 선택한다.

4. Marker 추출

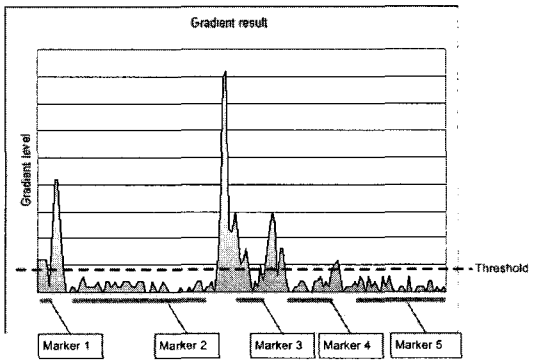
Marker 추출 단계는 I-type인지 P-type인지의 여부에 따라 분기되어 수행된다. 우선 I-type일 경우의 marker 추출 과정은 initial marker 추출이라고 한다. 이 경우 marker는 일정 값 이하의 gradient level을 갖는 연속된 pixel들의 집합을 말한다. 앞의 gradient 계산 과정을 통해 얻은 결과를 threshold값과 비교하여 marker를 구한다. 그림 2 (b)는 initial marker를 추출한 결과가 표시되어 있다. 점선은 threshold값을 나타내고, 그 이하의 값을 가지는 연속된 pixel들의 집합이 각각 marker들로 지정된 것을 확인할 수 있다. 이와 같이 추출된 marker는 watershed transform의 초기 영역으로 사용된다.

P-type일 경우에는 공간적 정보만이 아니라 시간적 정보도 이용하여 영역을 분할한다. 시간적 정보를 이용하는 방식에는 여러 가지가 있다. 예를 들어, motion estimation과 같은 방식을 통해 비교적 정확한 시간적 정보를 얻어낼 수 있는데, 이 경우에는 너무 큰 연산량이 필요하다. 따라서 본 논문에서는 적은 계산량으로 시간적 정보를 이용하는 방법을 사용한다. 이를 위하여 이전 단계에서 구한 변화 검출 마스크를 사용한다.

이 단계의 자세한 과정이 그림 3에 나와 있다. 그림 3의 (a)는 이전 frame에서의 영역 분할 결과로서, 전체 이미지가 Region 1 및 Region 2로 분할된 예를 보여준다. 그림 3 (b)는 전 단계에서 구한 변화 영역 마스크이다. 현재 frame과 이전 frame을 비교하여 변화가 생긴 pixel들은 검은색으로 표시되었다. 그림 3 (c)에서는 이전 frame에서의 영역 분할 결과에 변화 영역 마스크를 적용 (masking) 시킨 결과이다. 마스크의 검은색 부분은 모두 '0'이 되고, 흰색 부분은 원래 값을 그대로 유지한다. 마스크에 의해 '0'이 된 pixel들은 undefined 영역



(a) 입력 이미지에서 한 행(row)의 pixel 값



(b) (a)의 입력에서의 gradient 계산 결과 및 initial marker 추출 결과

그림 2. Gradient 계산 및 initial marker 추출 결과
Fig. 2. Gradient calculation and initial marker extraction result.

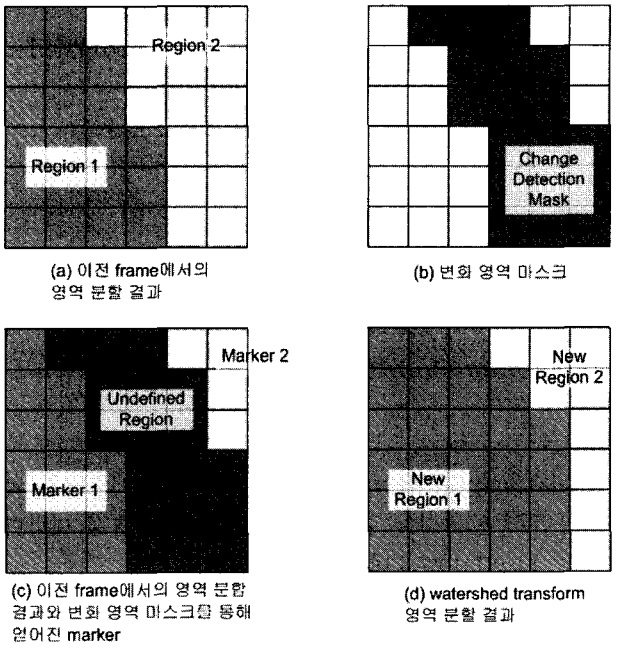
으로 분류된다. 그 이외의 영역들은 이전 frame에서의 영역에 따라 marker로 지정된다. Undefined 영역은 나중의 watershed transform에 의해 어떤 영역이 될지 결정되고, 그 결과가 그림 3 (d)와 같이 될 수 있다.

5. Watershed Transform

앞의 단계에서 구한 marker정보와 pixel의 gradient 값을 통해 watershed transform[11]을 수행한다. 이 방법에 대한 자세한 설명은 참고문헌[11]을 참조한다.

6. Region Merging

Watershed transform 단계를 거치면, 전체 영상은 영역별로 분리된다. 이전 단계에서 과분할을 피하기 위해 여러 절차가 추가되었지만, 그래도 필요 이상으로 분할된 영역들이 존재하게 된다. 따라서 이러한 영역들을 하나로 만들어야 할 필요가 있다. 이를 위해서, 우선적으로 각 영역들 사이의 인접 관계를 파악하여 그래프 형식으로 표현한다. 그 다음에 인접한 두 영역이 병합될 수 있는지 판단하고, 만약 병합되어야 할 경우에는 하나로 합친 후에 인접 영역 그래프를 업데이트한다. 이 때, 두 인접 영역의 병합 여부의 판단은 두 가지 조



(a) 이전 frame에서의 영역 분할 결과
(b) 변화 영역 마스크
(c) 이전 frame에서의 영역 분할 결과와 변화 영역 마스크를 통해 얻어진 marker
(d) watershed transform 영역 분할 결과

그림 3. P-type에서의 marker 추출 과정
Fig. 3. P-type marker extraction.

건으로 이루어지는데, 첫 번째는 영역별 색의 평균값의 비교이고, 두 번째는 두 인접 영역들 사이의 edge에서 weak edge (여기서 weak edge란, 두 인접 영역의 경계 중에서 작은 gradient 값을 가지는 픽셀을 말한다)의 비율이다. 이러한 방식으로 영역 병합을 수행한 후에, 일정 기준 이하의 크기를 가지는 영역을 대상으로 다시 한번 영역 병합을 수행하게 된다. 이 때, 일정 크기 이하의 영역은 인접 영역 중에서 가장 유사한 영역과 병합된다.

위와 같은 과정을 모두 거치고 나면, 영역 분할 단계가 완료된다. 이렇게 생성된 영역 분할 결과는 두 번째 단계인 foreground 영역 추출 단계의 입력으로 사용된다.

Ⅳ. Foreground 영역 추출

영역 분할된 이미지에서 foreground 영역을 판별하는 방법은 여러 가지가 있을 수 있으나, 본 논문에서는 foreground가 중앙에 위치하고 있다는 가정으로부터 foreground 영역을 판별해 낸다. 이렇게 하면 타 알고리즘에 비해 적은 연산량으로도 foreground 영역을 추출해 낼 수 있다. 알고리즘의 자세한 순서는 다음과 같다.

1. Foreground Object Mask 정의

Foreground 영역을 추출한다는 것은 전체 영상을

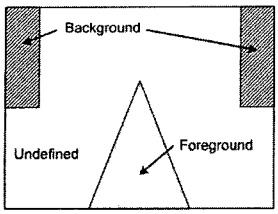


그림 4. Foreground object mask 정의
Fig. 4. Foreground object mask.

foreground 영역과 배경으로 분리한다는 뜻이다. 따라서 foreground 영역이 중앙에 위치한다는 가정을 바탕으로, foreground 영역, 배경 영역, 그리고 미지정 영역의 세 분류를 가지는 mask를 미리 정해둔다. 그림 4는 정의된 mask이다. Foreground 영역과 배경으로 표시된 부분이 있고, 그 이외의 나머지 부분은 미지정 영역이다.

2. 인접 영역 그래프 생성

앞에서 언급한 영역 분할의 결과를 인접 영역 그래프로 표현한다. 각 영역은 그래프의 정점 (vertex) 이 되고, 각 영역들이 서로 인접한 경우에는 해당하는 정점들을 간선 (edge) 으로 연결한다. 이 때 간선의 가중치는 각 영역들 사이의 유사한 정도에 따라 결정된다.

3. 그래프에 mask 정보 적용

생성된 인접 영역 그래프에 미리 지정된 mask를 적용시킨다. 그렇게 되면 각 영역, 즉 그래프의 정점은 foreground, 배경, 그리고 미지정 영역의 세 가지 중에 한 가지 값을 가지게 된다.

4. 미지정 영역의 분류

그래프의 정점 중에서 미지정 영역으로 선택된 정점을 foreground나 배경 둘 중 하나로 선택하는 작업을 한다. 우선 미지정 영역 중에서 foreground이나 배경 영역을 이웃으로 하는 정점을 선택한 후, 그 중에서 정점 사이의 가중치가 가장 작은 것을 선택하여, 이웃하는 정점의 종류에 따라 foreground 또는 배경으로 분류한다. 한 정점이 분류된 후에는 다시 처리 대상이 되는 정점들을 찾고, 미지정 영역으로 선택된 모든 정점이 분류될 때까지 이 과정을 반복한다.

그림 5는 foreground 영역 추출 과정을 보여주고 있다. 그래프에서 정점은 각각 분할된 영역을 나타내고, 영역의 인접 여부에 따라 연결되어 그래프 구조를 가지게 된다. 영역과 영역 사이에는 가중치를 갖는 간선이

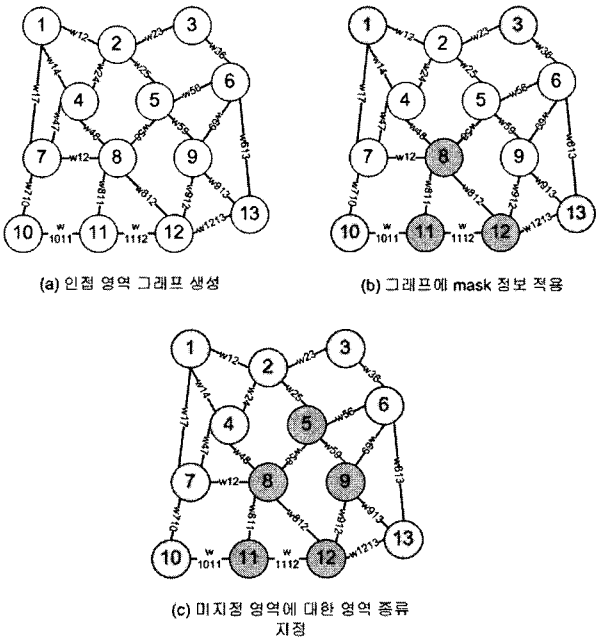


그림 5. 인접 영역 그래프에서 foreground 영역추출 과정
Fig. 5. Foreground object extraction with an adjacency graph.

있어서 이를 바탕으로 미지정 영역의 분류를 수행하게 된다. 그림 5 (a)는 인접 영역 그래프의 예시를 보여준다. 총 13개의 영역이 있다고 가정하고, 인접 영역은 가중치를 갖는 간선에 의해 연결되어 있다. 그림 5 (b)는 미리 정의된 mask 정보를 적용한 결과이다. 영역 1, 3, 10, 13은 배경 영역으로 선택되고, 영역 8, 11, 12는 foreground 영역으로 선택되었다. 그림 5 (b)에서 흰색 정점 2, 4, 5, 6, 7, 9는 미지정 영역으로 분류된 영역이다. 그림 5 (c)는 미지정 영역에 대하여 영역 종류를 지정한 후의 결과를 보여주고 있다. 미지정 영역 중에서 2, 4, 6, 7은 배경 영역으로, 5, 9는 foreground 영역으로 분류된 결과를 보여준다.

V. Implementation

본 논문의 알고리즘은 실시간 임베디드 시스템에 적합하도록 연산량을 최대한 적게 하는 것이 필요하다. 이를 위해서 영상을 down-sampling하여, 그 결과를 입력으로 하여 영상 분할 및 foreground영역 추출을 수행함으로써 연산량을 줄이도록 한다. 수평과 수직 각 방향으로 2 대 1의 비율로 down-sampling하여, 전체적으로 처리해야 하는 데이터의 크기를 4분의 1로 줄임으로써 계산량을 감소시킨다.

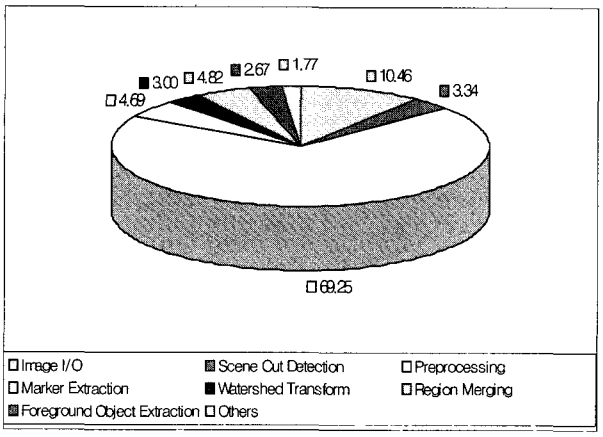


그림 6. 제안된 알고리즘의 프로파일링 결과
Fig. 6. Profiling result of the proposed algorithm.

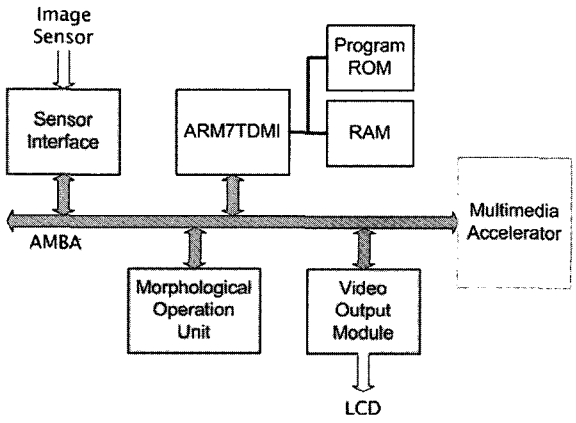


그림 7. 제안된 object segmentation 시스템을 포함하는 SoC의 구조도
Fig. 7. Block diagram of the SoC including the proposed object segmentation system.

계산 속도를 늘이기 위한 또 다른 방법으로써 알고리즘 중에서 연산량이 많은 특정 부분을 하드웨어로 설계한다. 이를 위해서 본 논문의 알고리즘을 순수하게 소프트웨어로 수행한 결과를 프로파일링 하였다. 그 결과를 그림 6이 보여준다. 그림 6에서 'Preprocessing'으로 표시된 부분이 영상 평탄화와 gradient 계산 부분을 나타내는데, 이 부분이 전체 계산량의 약 70%를 차지한다. 이 부분은 morphological 연산들이 대부분을 차지하는데, 이 연산들은 병렬 처리 및 파이프라인 처리를 효과적으로 할 수 있으므로, 통합하여 하나의 하드웨어 블록으로 만들기에 적당하다. 실험 결과, 이 부분을 하드웨어로 설계하여 해당 단계의 수행 시간을 소프트웨어 대비 약 10%로 줄일 수 있다.

본 논문에서 제안된 시스템은 Magnachip 0.18um 공정을 사용한 칩 (ViDan3500) 으로 제작될 예정이다. 제안된 시스템을 포함하는 칩의 블록 다이어그램은 그림 8과 같다. 이 칩은 제안된 영역 추출 알고리즘 뿐만 아니라 MPEG-4, JPEG, H.264, 및 각종 오디오 코덱을 포함한 다양한 멀티미디어 기능이 구현되었다. 그림 7은 제안된 영역 추출 알고리즘을 구현하는 부분을 주로 보여주고 있으며, 멀티미디어 기능을 처리하는 부분은 모두 통합하여 하나의 블록인 Multimedia Accelerator로 나타내었다.

제안된 알고리즘의 소프트웨어는 ARM7TDMI 프로세서에서 구동하고, Morphological Operation Unit이 영상 평탄화와 gradient 계산을 위한 morphological 연산을 수행한다. Image Sensor Interface와 Video Output Interface는 영상 데이터의 입출력을 담당한다. Morphological Operation Unit의 게이트 수는 16,612 게이트이며, 프로그램 크기는 18.9Kbytes, 그리고 필요한 메모리 크기는 100Kbyte 이내이다.

VI. Evaluation

1. Simulation results

실험을 위하여 QCIF 크기, 15fps의 Akiyo 영상을 사용하였다. 그림 8은 I-type일 경우에 중간 과정 및 최종

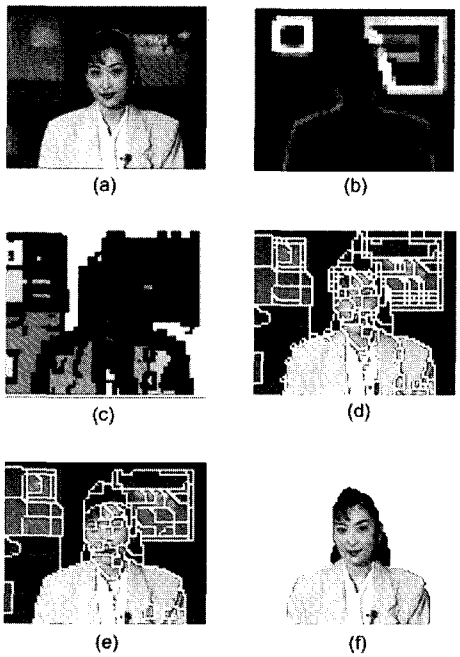


그림 8. I-type에서 foreground 추출. (a) 입력 이미지, (b) gradient 계산 결과, (c) initial marker, (d) watershed transform 결과, (e) region merging 결과, (f) 최종 foreground 영역 추출 결과
Fig. 8. I-type foreground extraction result. (a) Input image, (b) gradient result, (c) initial marker, (d) watershed transform, (e) region merging, (f) final foreground extraction result.

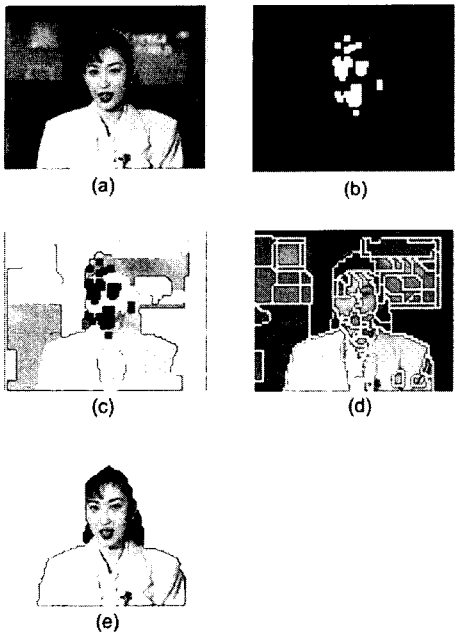


그림 9. P-type에서 foreground 추출. (a) 입력 이미지, (b) 변화 검출 마스크, (c) marker 추출, (d) 영역 분할 결과, (e) foreground 영역 추출 결과

Fig. 9. P-type foreground extraction result. (a) Input image, (b) change detection mask, (c) predictive marker, (d) region segmentation result, (e) final foreground extraction result.

foreground 영역 추출 결과이다. 그림 8 (a)는 입력 이미지이고, (b)는 gradient 계산 결과, (c)는 initial marker 추출 결과이다. 그림 8 (d)는 watershed transform을 통해 영역 분할된 결과이고, (e)는 region merging 단계를 수행한 후의 최종 영역 분할 결과이다. 영역 분할 결과를 이용해 foreground 영역 추출을 수행한 결과는 그림 8 (f)와 같다.

현재 frame이 P-type일 경우의 중간 과정 및 결과를 그림 9에 나타내었다. 입력 영상 이미지인 그림 9 (a)는 실험 영상의 15번째 frame이다. 그림 9 (b)는 변화 검출 마스크를 구한 결과이고, (c)는 이를 통해 추출한 marker이다. 그림 9 (d)는 최종적으로 영역 분할된 결과이고, (e)는 이를 바탕으로 추출한 foreground 영역의 결과이다.

본 논문에서 제안한 알고리즘은 카메라의 움직임이 있는 상황에서도 만족할 만한 결과를 보여준다. 이 부분을 실험하기 위하여, 추가적으로 두 가지의 다른 특징을 가지는 영상을 사용하였다. 한 영상은 카메라의 움직임이 없는 영상 (Claire) 이고, 다른 한 영상은 카메라의 움직임이 비교적 큰 영상 (Foreman) 이다. 이에 대한 실험 결과는 그림 10에서 확인할 수 있는데, 카메라의 움직임이 존재하는 영상의 경우에도 움직임이 없을 때와 마찬가지로 만족할 만한 영역 추출의 결과를 얻을 수 있다.

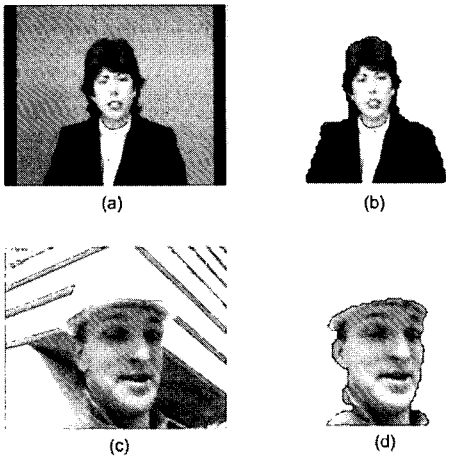


그림 10. 카메라의 움직임이 영역 추출에 미치는 영향. (a) Claire 영상 (고정된 카메라), (b) (a)에서 추출 결과, (c) Foreman 영상 (움직임이 있는 카메라), (d) (c)에서 추출 결과

Fig. 10. Insignificant impact of camera motion on extraction results. (a) Claire image (fixed camera), (b) foreground object extracted from (a), (c) Foreman image (moving camera), (d) foreground object extracted from (c).

2. 수행시간

본 논문에서 구현한 알고리즘의 수행시간 측정 결과는 표 1과 같다. 그림 7의 하드웨어를 RTL로 구성한 후 동작 주파수를 100MHz로 정하고 시뮬레이션하여 필요한 시간을 측정하였다. 표 1에서와 같은 수행 결과를 바탕으로, 본 논문에서 제안한 알고리즘이 평균적으로 15fps 이상의 속도로 수행될 수 있다는 것을 확인할 수 있다.

Ⅶ. 결 론

본 논문에서는 임베디드 시스템에 적합한 foreground 영역 추출 알고리즘을 제안하고, 이를 구현하였다. 제안된 알고리즘은 비교적 적은 연산량으로도 영상에서 영역을 분할하여 foreground 영역을 추출할 수 있게 해준다. ARM7TDMI를 주 프로세서로 하는 시스템에 구현한 결과 QCIF 크기 영상을 초당 15 frame 처리할 수 있었다.

본 논문에서 제안한 알고리즘으로 영역 추출을 실시

표 1. 수행 시간
Table 1. Processing time.

frame #	frame 구분	수행 시간(ms)
1	I	88.390
2	P	60.239
3	P	58.010
4	P	58.059
5	P	58.901
6	P	58.778

간으로 수행하게 되면 이를 바탕으로 객체 기반 영상 압축 기법 또한 실시간으로 구현하는 것이 가능해진다. 따라서, MPEG-4의 객체 기반 영상 압축 및 처리의 실용화 가능성을 부활시키는 계기가 될 것으로 예상된다.

참 고 문 헌

[1] ISO/IEC 14496-2, Coding of audio-visual objects Part 2: Visual, 2001.

[2] Y. Tsai, C. Lai, Y. Hung, and Z.Shih, "A Bayesian approach to video object segmentation via merging 3-D watershed volumes," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, pp. 175-180, Jan. 2005.

[3] S. Lee, C. Ouyang, and S.Du, "A neuro-fuzzy approach for segmentation of human objects in image sequences," *IEEE Trans. Systems, Man and Cybernetics*, Part B, vol. 33, pp. 420-437, June 2003.

[4] H. Xu, Younis, A.A Younis, and M.R. Kabuka, "Automatic moving object extraction for content-based applications," *IEEE Trans. Circuits Syst. Video Technol.*, vol.14, pp. 796-812, June 2004.

[5] D. Wang, "Unsupervised video segmentation based on watersheds and temporal tracking," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, pp. 539-546, Sept. 1998.

[6] E. Hayman and J. Eklundh, "Statistical background subtraction for a mobile observer," *in Proc. IEEE Int. Conf. Computer Vision*, Oct. 2003, pp. 67-74.

[7] S. Chien, Y. Huang, B. Hsieh, S. Ma, and L. Chen, "Fast video segmentationalgorithm with shadow cancellation, global motion compensation, and adaptive threshold techniques," *IEEE Trans. Multimedia*, vol. 6, pp. 732-748, Oct. 2004.

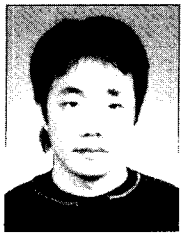
[8] L. Vincent and P. Soille, "Watersheds in digital spaces: An efficient algorithm based on immersion simulations," *IEEE Trans. Pattern Anal. Machine Intell.*, Vol. 13, No. 6, pp. 583-598, June 1991.

[9] L. Vincent and P. Soille, "Morphological grayscale reconstruction in image analysis: Applications and efficient algorithms," *IEEE Trans. Image Processing*, Vol. 2, No. 2, pp. 176-201, Apr. 1993.

[10] P. Salembier and M. Pardas, "Hierarchical morphological segmentation for image sequence coding," *IEEE Trans. Image Processing*, Vol. 3, No. 5, Sept. 1994.

[11] F. Meyer, "Color image segmentation," *in Proc. Int. Conf. on Image Processing and its Applications*, pp. 303-306, 1992.

저 자 소 개



김 지 수(학생회원)

2005년 서울대학교 전기공학부 학사 졸업.

2006년 현재 서울대학교 전기컴퓨터공학부 석사 과정.

<주관심분야 : 컴퓨터 구조, SoC 설계, 멀티미디어 신호처리>

이 혁 재(정회원)

1987년 서울대학교 전자공학과 학사 졸업.

1989년 서울대학교 전자공학과 석사 졸업.

1996년 Purdue대학교 전기컴퓨터공학과 박사 졸업.

2006년 현재 서울대학교 전기컴퓨터공학부 부교수

<주관심분야 : 컴퓨터 구조, 멀티미디어 SoC 설계>

이 태 호(학생회원)

2001년 한양대학교 전자컴퓨터공학과 학사 졸업.

2003년 한양대학교 전자컴퓨터공학과 석사 졸업.

2006년 현재 서울대학교 전기컴퓨터공학부 박사 과정.

<주관심분야 : 컴퓨터 구조, SoC 설계, 멀티미디어 신호처리>