비재귀 Flood-Fill 알고리즘을 이용한 적응적 이미지 Labeling 알고리즘

김 도 현[†] · 강 동 구[†] · 차 의 영[††]

요 약

본 논문에서는 이진화 영상의 물체 분석에서 자주 사용되는 새로운 Labeling 알고리즘을 제안한다. 제안한 Labeling 알고리즘은 기존의 Labeling 과는 달리 복잡한 Equivalent Label Merging/Ordering이 필요하지 않으며 비재귀적인 Flood-filling에 의하여 1 pass에 Labeling이 이루어진다. 또한 Gray-level 이미지에 대해서도 쉽게 확장될 수 있으며, HIPR Image Library를 대상으로 실험한 결과 기존의 방법보다 2배 이상의 빠른 수행 속도를 보였다.

Adaptive Image Labeling Algorithm Using Non-recursive Flood-Fill Algorithm

Do Hyeon Kim[†] · Dong Koo Kang[†] · Eui Young Cha[††]

ABSTRACT

This paper proposes a new adaptive image labeling algorithm for object analysis of the binary images. The proposed labeling algorithm need not merge/order of complex equivalent labels like classical labeling algorithm and the processing is done during only 1 pass. In addition, this algorithm can be extended for gray-level image easily. Experiment result with HIPR image library shows that the proposed algorithm process more than 2 times faster than compared algorithm.

키워드 : 레이블링(Labeling), Flood-Fill Algorithm

1. 서 론

이진화된 영상의 분석에 있어서 Labeling 기법은 요소들에 대한 연결성을 제공함으로써 물체 분석, 잡영 제거 등 여러 가지 방법으로 많이 사용되고 있다. Labeling 기법은 입력 영상의 각각의 pixel에 고유한 숫자 번호(label)를 부여함으로써 주변 pixel에 대한 연결성을 구현하는 기법이다. 이러한 Labeling 기법은 Computer Vision의 초창기에서부터 Rosenfeld, Pfaltz 등에 의해 개발되어 많은 연구 및 개선이 되어왔다[1, 2].

일반적인 Labeling 알고리즘은 두 번의 scan을 통해 입력 영상 이미지에 대해 label된 이미지 정보를 획득하게 된다. 하지만 첫 번째 scan을 통하여 영상의 각 pixel에 대해 임시 Label를 부여하고, label conflict가 일어나는 label에 대하여 equivalence [1]table에 기록한 후, 두 번째 scan에서 이를 처리하여 임시 label을 재조정하는 일반적인 Labeling 기법은 equivalence table의 equivalence class를 효과적으로 처리하고 구현하기가 다소 복잡하고 어려울 뿐만 아니라 처리 시간이 오래 걸리는 단점이 있었다.

이를 개선하기 위한 방법으로 Gonzales와 Woods는 [3]에서 equivalent label을 n×n boolean matrix B(n은 label의 개수)를 이용하여 동치관계로를 이끌어냈고, Haralick과 Shapiro는 equivalence table을 graph structure로 구현하고 각각의 노드에 대한 일반적인 depth-first search로 equivalence label set을 구하는 방법을 사용했다[4]. Stefano 등은 equivalence table을 구성하는 복잡함을 개선하여 1D class array(C)와 frequency array(F)를 두어 equivalent class에 해당하는 두 개의 label을 첫 번째 pass에서 바로 조정하는 방법을 사용하였다[5]. 하지만 [5]에서 제안한 방법은 실험을 통하여 살펴보았을 때 이미지의 크기가 커지고 label의 개수가 많아짐에 따라 class array C를 조정하는 회수가 증가하여 제안하는 알고리즘보다 수행속도가 느려지는 단점이 있었다. 이러한 단점들을 개선하는 새로운 Labeling의 방법으로 Non-recursive Flood-Fill Labeling 기법을 제안한다.

2장에서는 일반적인 Labeling 방법 및 Stefano에 의해 개

† 준 회 원 : 부산대학교 대학원 전자계산학과
†† 종신회원 : 부산대학교 전자계산학과 교수
논문접수 : 2001년 10월 11일, 심사완료 : 2002년 5월 23일

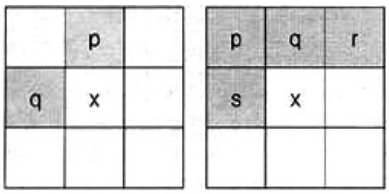
1) 일반적인 Labeling 과정에서 현재 픽셀의 위쪽 픽셀의 label과 왼쪽 픽셀의 label이 서로 일치하지 않는 경우를 말한다.

선된 Labeling의 3가지 방법에 대하여 살펴본다. 3장에서는 제안하는 Non-recursive Flood-fill Labeling을 설명하고, 4장에서는 이것을 이용한 Labeling의 확장성 및 gray-level 이미지에서의 Labeling에 대해 살펴본다. 5장에서는 HIPR Image Library DB의 각 category별 이미지에 대하여 Stefano에 의해 개선된 방법과의 비교 실험 결과를 살펴보고, 6장에서는 향후 연구 방향 및 결론을 맺는다.

2. Connected Components Labeling Algorithm

2.1 Classical Connected Components Labeling Algorithm

일반적인 Labeling 알고리즘은 두 번의 scan을 통해 입력 영상 이미지에 대해 label된 이미지 정보를 획득하게 된다. 먼저 첫 번째 scan을 통하여 이진화된 이미지를 위에서 아래로, 좌에서 우로 스캔하면서 1인 pixel x에 대하여 이미 방문한 몇 개의 주변 pixel(N)를 살펴 영상의 각 pixel에 label을 붙인다. 4-연결성에 의한 Labeling의 경우, 현재 pixel x의 상단 pixel p와 좌측 pixel q를 검사하고($N_4 = \{p, q\}$), 8-연결성에 의한 Labeling의 경우는, 주변의 4개 pixel $N_8 = \{p, q, r, s\}$를 검사하여 수행된다.



N_4 N_8

4-연결성에 의한 Labeling을 살펴보면, (a) p = 1, q = 0이거나 p = 0, q = 1인 경우 1인 pixel의 label을 x에 부여한다. (b) 만약 p = 1, q = 1인 경우 p와 q의 label이 같으면 그 label을 현재 pixel x의 label로 하고, (c) p와 q의 label이 다르면, label conflict가 발생한 경우로 어느 한쪽의 label을 x에 할당하고, p와 q의 label을 equivalence table에 입력한다. 이것은 같은 connected component이지만 scan에 의하여 다른 label로 처리되어 온 경우로 두 label이 동일하게 조정될 수 있도록 equivalence table을 두어 나중에 merge하게 된다. (d) 그 외의 경우 즉, p = 0, q = 0인 경우 새로운 label을 x에 부여한다.

위와 같은 첫 번째 labeling의 과정이 모두 끝나면 입력 영상의 각각의 pixel에 대하여 임시적인 label이 부여된 상태가 되지만 이 label은 label conflict가 발생한 경우 고유한 label이 아니므로 equivalence table의 equivalent label set에서 최소의 label을 찾아 이를 같은 label로 merge함으로써 고유한 equivalence class의 label identifier를 얻게 된다. 이러한 equivalence merging이 끝나면 두 번째 scan을 통하여 다시 이미지 pixel 각각의 임시 label에 대하여 해당하는 equivalence class의 고유한 label identifier를 부여함으로써 labeling을 완료하게 된다. 이와 같은 과정은 <표 1>에 요약되어 있다.

〈표 1〉 Sequential Connected Components Algorithm using 4-connectivity

1. Scan the image left to right, top to bottom
2. If the pixel is 1, then
  (a) If only one of its upper and left neighbors has a label, then copy the label.
  (b) If both have the same label, then copy the label.
  (c) If both have different labels, the copy the upper's label and enter the labels in the equivalence table as equivalent labels.
  (d) Otherwise assign a new label to this pixel and enter this label in the equivalence table.
3. If there are more pixels to consider, then go to step 2.
4. Find the lowest label for each equivalent set in the equivalence table.
5. Scan the picture. Replace each label by the lowest label in its equivalent set.

2.2 SEL(Simple and Efficient Labeling) Algorithm

일반적인 Labeling 알고리즘은 첫 번째 scan에서 기록된 임시 label에 대하여, 동일한 label을 merge하기 위하여 복잡한 equivalence merging processing을 거치게 된다. 이와 같은 복잡한 구조 처리를 간소화하고 실제적인 구현을 쉽게 하기 위해서 Stefano등은 label conflict가 있을 때 바로 equivalent label을 조정하여 불필요한 label 처리를 줄이는 방법을 제안하였다. 여기서는 제안하는 알고리즘과의 비교를 위한 선행 단계로 SEL을 간단하게 살펴보겠다.

SEL에서는 Equivalence Label Merging을 위한 자료구조로 Class Array(C)를 두었다. 이는 각각의 임시 label에 해당하는 실제적인 label identifier를 저장하고 있는 1차원 배열로 C[i]는 label i에 대한 실제적인 label identifier이다. C[i]는 처음에 i값으로 초기화된 후 label conflict가 있을 때 conflict된 label li, lj에 대하여 C[li]와 C[lj]가 다른 경우에 대해서 그때까지의 생성된 label과 비교하여 동일한 label을 아래와 같이 조정된다. [Version 1]

```
for(k = 1 ; k <= nLabelCount ; k++) {
   if ( C[k] == C[lj] ) C[k] = C[li] ;
}
```

위 과정을 통해서 생성된 Class Array(C)을 이용하여 두 번째 scan에서 이미지의 임시 label에 해당하는 값을 Class Array(C)의 index로 참조함으로써 연결된 요소에 대한 unique한 labeling을 가능하게 했다.

[Version 1]에서의 이와 같은 방법은 label merging을 첫 번째 pass에서 바로 조정함으로써 label conflict를 줄였지만, label의 개수가 증가함에 따라 Class Array(C)를 조정하는 overhead가 커지는 문제점이 있다. 이를 개선하여 conflict가 일어난 label이 최근에 생성된 label일 경우 그에 해당하는 1개의 Class Array element만 조정함으로써 Class Array 전체를 탐색하는 불필요한 연산을 줄이는 방법[Ver-

sion 2]과 모든 label에 대한 Frequency Array(F)를 두어 [Version 2]의 방법을 일반화한 방법[Version 3]이 있다.

3. NFL(Non-recursive Flood-fill Labeling) Algorithm

기존의 Labeling 방법은 두 번의 scan을 통하여 label의 값을 조정하는 방법을 사용하였다. 따라서 label에 대한 conflict가 발생하게 되고 이를 위하여 고유한 label identifier를 구하기 위하여 merging 등에 의하여 많은 시간이 소요되었다. 이를 개선하기 위한 다른 방법으로 Flood-fill Algorithm을 이용한 비재귀적인 Labeling방법인 NFL(Non-recursive Flood-fill Labeling)을 제안한다.

Flood-filling은 Graphic Image Editing Tool에서 흔히 사용되는 color filling방법이다. 이는 recursion으로 다음과 같이 쉽게 구현될 수 있다[6].

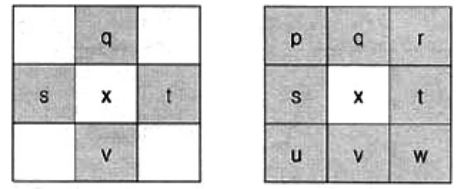
```
// Recursive flood-filling
flood_fill(int x, int y) {
  if( read_pixel(x, y) == TARGET_VALUE ) {
    write_pixel( x, y, FILL_VALUE );
    flood_fill(x-1, y);
    flood_fill(x+1, y);
    flood_fill(x, y-1);
    flood_fill(x, y+1);
  }
}
```

위와 같이 System Stack을 이용한 recursive한 function call은 context switching에 의한 overhead뿐만 아니라 Stack Memory Dump Fail의 가능성이 높다. 특히 Labeling의 경우, component의 최대 pixel 개수만큼의 function call이 일어나므로 위험한 방법이다. 따라서 Non-recursive한 방법이 요구된다. 이를 위하여 NxM 이미지에 대한 Label Stack(LS)을 정의한다. LS는 System Stack을 간단하게 대체하기 위한 자료구조로 각 pixel의 위치에 대한 방문 flag와 리턴위치를 가지는 Matrix이다.

```
class LS
{
    boolean bVisited;
    position pReturnPos;
};
```

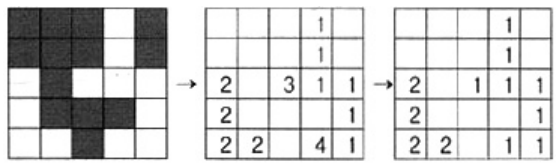
이와 같은 LS N×M개를 이용하여 이미지의 각 pixel을 방문하면서 방문 flag(bVisited)를 setting하고 복귀 위치(pReturnPos)를 저장함으로써 간단하게 System Stack의 기능을 하게 된다.

LS를 사용한 NFL Algorithm은 먼저 첫 번째 scan을 통하여 이진화된 이미지를 위에서 아래로, 좌에서 우로 스캔하면서 방문하지 않은 1인 pixel x에 대하여 Non-recursive Flood-filling을 수행한다.

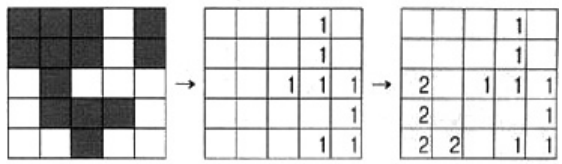


N_4 N_8

Non-recursive Flood-filling에서는 먼저 현재 위치 및 종료 위치를 설정하고 방문 flag을 설정한 다음 현재 위치에 대하여 현재 label 번호를 붙인다. 그 다음 주변 pixel(N : $N_4 = \{q, s, t, v\}$ or $N_8 = \{p, q, \cdots, w\}$) 중에서 방문하지 않은 1인 pixel n을 찾는다. 이때 n이 존재할 경우 pixel n의 복귀 위치를 현재 pixel x의 위치로 설정해둔다. 그 다음 현재 pixel의 위치를 다시 pixel n의 위치로 설정하여 이를 반복함으로써 Flood-filling이 구현된다. 만약 방문해야 할 주변 pixel n이 존재하지 않을 경우, 현재 pixel의 위치를 현재 pixel의 리턴위치로 설정함으로써 이전 위치로의 이동이 이루어지게 된다. 이때 리턴된 pixel의 위치가 종료 위치와 일치하게 되면 현재 label 번호에 대한 더 이상의 Flood-filling은 이루어지지 않고 다음 label에 대한 Flood-fill을 계속하게 된다.



(a) Classical Algorithm



(b) Proposed Algorithm

□ foreground ■ background

(그림 1) Labeling Process

살펴본 바와 같이 NFL은 기존의 알고리즘처럼 unique한 label을 위하여 2 Pass를 거치면서 Equivalent label에 의한 Label Merging, Mapping, Ordering이 필요없으므로 많은 연산량을 줄일 수 있다. 또한 8-연결성으로 확장할 경우에도 간단하게 확장할 수 있는 구조이다.

이와 같은 과정은 <표 2>, <표 3>에 요약되어 있다. 여기서 target_value는 1을 나타내는 값으로 이미지의 전경(foreground)에 해당하는 값이다. 즉, target_value가 설정이 되면 주변 pixel의 값이 target_value와 동일한 경우 연속적으로 label값이 채워져 나가게 된다.

〈표 2〉 NFL (Non-recursive Flood-fill Labeling) Algorithm

1. Set label_number 1
2. Scan the image left to right, top to bottom
3. If the pixel was not visited and its value pixel(x, y) = target_value, then
Execute Non-recursive Flood-filling Algorithm routine
Increment label_number
4. If there are more pixels to consider, the go to step 3.

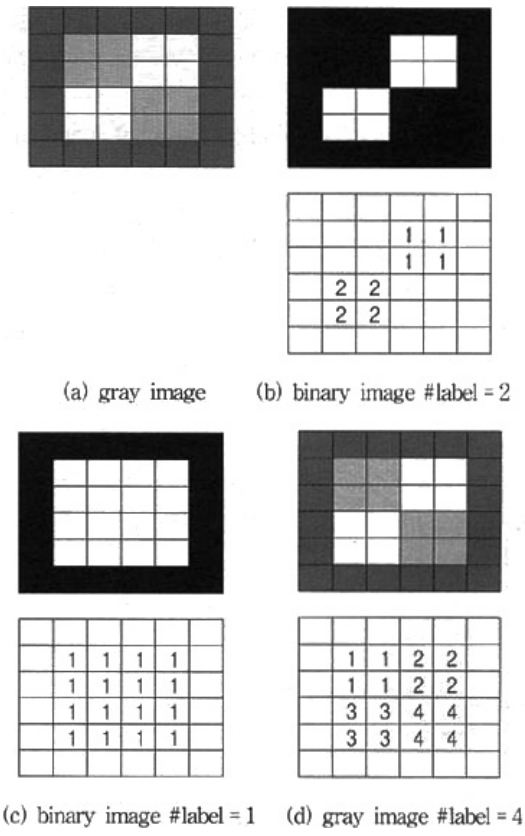
〈표 3〉 Non-recursive Flood-filling Algorithm

1. Set target_value as pixel (x, y) value
2. Set current_position as pixel (x, y) position
3. Set end_position as pixel (x, y) position
4. Set current_position as Visited
5. Set current_position's pixel value to label_number
6. If one of current_position's 4/8 connected position was not visited and its value pixel (x, y) = target_value, then
(a) set return_position of correspond neighbor pixel as current_position
(b) set current_position as correspond neighbor pixel's position
else
reset current_position as current pixel's return_position
7. If current_pixel equals to end_pixel then exit
8. Go to step 4.

4. Adaptive NFL(Non-recursive Flood-fill Labeling) for Extended Approach

일반적으로 Labeling은 이진 영상 처리의 중요한 도구로서 입력 이미지의 각 pixel에 대하여 0 또는 1의 이진값을 요구하고 있다. 따라서 Labeling을 위해서는 이미지의 이진화가 선행되어야 한다.

이미지의 이진화는 임계값(Threshold)을 기준으로 이미지의 각 pixel값을 0 또는 1로 변환하는 방법으로 이런 임계값을 구하기 위한 방법으로 고정 임계치 적용방법, 반복 임계치(Iterative Threshold) 적용방법, 모드법(Mode), P-Tile법, 블록이진화, 적응적 이진화(Adaptive Threshold) 등이 있다. 하지만 이와 같이 이진화를 수행할 경우 영역에 대한 정보가 무시되어 버리기 때문에 이 이진 영상에 대한 Labeling은 의미를 상실하게 되는 치명적인 약점이 있다. 예를 들어, (그림 2)의 gray image (a)에 대하여 기존의 알고리즘은 이진화의 결과에 따라 2가지 경우의 binary image (b), (c)로 Labeling됨으로써 label의 개수가 달라지며 이는 이진화에 상당히 의존적이다. 하지만 <표 4>, <표 5>와과 같이 픽셀값에 적응적으로 수정하면 (d)와 같은 label로 분석이 가능하게 된다. 여기서 T는 주변 pixel와의 merge를 결정하는 임계값이다. 즉, 이미지를 이진화할 경우 손실되는 영역에 대하여 gray-level 이미지 그대로 Labeling함으로써 적응적으로 labeling 분석이 가능하다는 장점이 있다.



(그림 2) Gray Image Labeling

〈표 4〉 Adaptive NFL Algorithm

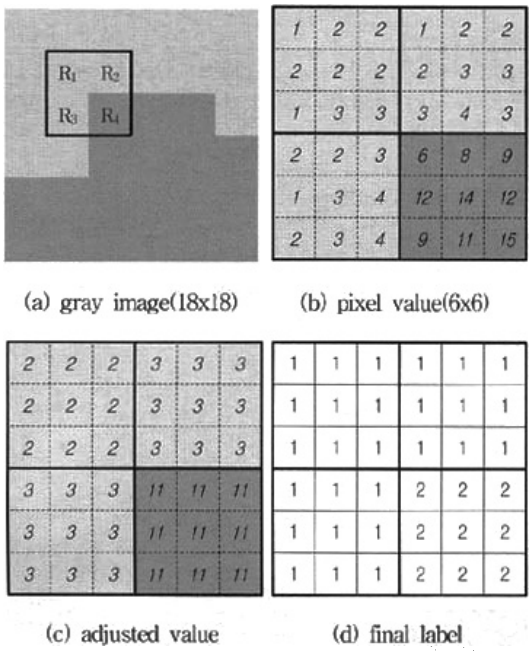
1. Set label_number 1
2. Scan the image left to right, top to bottom
3. If one of current_position's 4/8 connected position was not visited
and value $|\,pixel(x,y) - target\_value\,| < T$, *then*
Execute Adaptive Non-recursive Flood-filling Algorithm routine Increment label_number
4. If there are more pixels to consider, the go to step 3.

〈표 5〉 Adaptive Non-recursive Flood-filling Algorithm

1. Set target_value as pixel(x, y) value
2. Set current_position as pixel(x, y) position
3. Set end_position as pixel(x, y) position
4. Set current_position as Visited
5. Set current_position's pixel value to label_number
6. If one of current_position's 4/8 connected position was not visited and $|\,pixel(x,y) - target\_value\,| < T$, *then*
(a) set return_position of correspond neighbor pixel as current_position
(b) set current_position as correspond neighbor pixel's position
else
reset current_position as current pixel's return_position
7. If current_pixel equals to end_pixel then exit
8. Go to step 4.

이와 같이 gray level 이미지에 대하여 NFL을 이용할 경

우에는 잡영 등 세부적인 고주파수 성분에 의하여 많은 영역으로 나누어질 수 있으므로 이미지를 median filter, mean filter, lowpass filter 또는 n×n block 단위의 평균값으로 픽셀값을 조정하여 이미지에 대한 고주파수 성분을 제거하면 효율적인 gray-level 이미지에서의 영역 labeling이 가능하게 된다.



(그림 3) Enhancement of Gray Image Labeling

(그림 3)에서는 이러한 Gray-level 이미지의 Labeling 과정을 볼 수 있다. gray-level 이미지(a)의 부분 영역 R($R = R_1 \cup R_2 \cup R_3 \cup R_4$)의 pixel값이 (b)와 같을 때 이를 3x3 pixel 블록으로 평균하여 조정된 값은 (c)와 같게 된다. 여기서 3x3 영역 R_1, R_2, R_3, R_4의 각각의 pixel $r_i \{r_i \in R_i\}$에 대하여 임계값 $T (1 \le T \le 6)$로 수정된 NFL을 수행하게 되면, r_1, r_2, r_3의 인접한 pixel에 대한 절대값(absolute value)은 $|r_1 - r_2| < T, |r_1 - r_3| < T$ 이므로 merge되고, r_4에 대해서는 $|r_2 - r_4| > T, |r_3 - r_4| > T$ 이므로 (d)와 같은 최종적인 gray-level 이미지의 Labeling이 이루어진다.

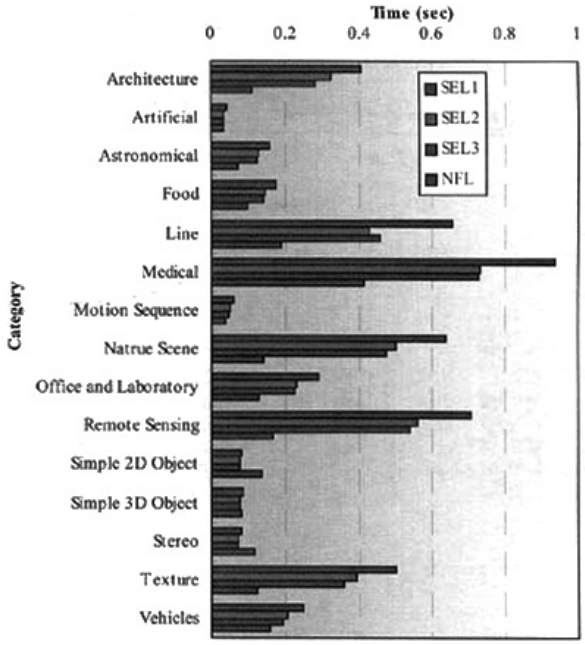
5. Experiments

본 논문에서 제안한 NFL 알고리즘 및 SEL 알고리즘은 Visual C++6.0 MFC를 사용하여 Pentium III 시스템에서 구현하였으며, 비교 알고리즘(SEL)의 세 가지 방법(SEL1, SEL2, SEL3)과의 성능 비교를 위하여 [5]에서 사용된 것과 같은 HIPR Image Library DB (http : //www.dai.ed.ac.uk/HIPR2/)의 15개 Category별로 분류된 245개 이미지에 대하여 비교 실험하였다. 여기서 HIPR Image의 이진화 방법으로는 반복 적용에 의한 임계값을 찾아 이미지를 이진화하는 반복 임계치 적용 방법(Iterative Threshold)을 사용하였다.

1	Architecture (17)	9	Office and Laboratory (12)
2	Artificial (19)	10	Remote Sensing (15)
3	Astronomical (27)	11	Simple 2D Object (13)
4	Food (13)	12	Simple 3D Object (17)
5	Line (20)	13	Stereo (9)
6	Medical (29)	14	Texture (10)
7	Motion Sequence (15)	15	Vehicles (10)
8	Natrue Scene (19)		Total 245 images

〈표 6〉 Category of Test Image

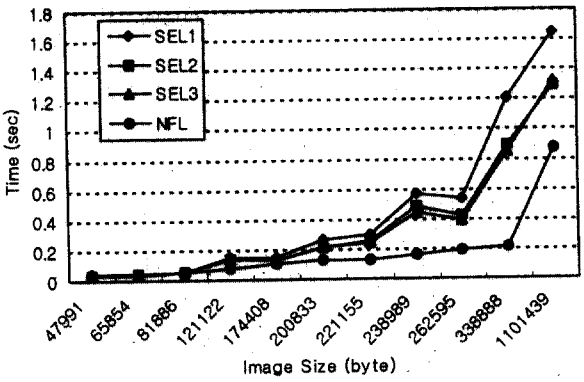
(그림 4)에서는 15개의 Category별 평균 Labeling 시간을 나타내고 있다. 'Architecture', 'Line', 'Medical', 'Nature Scene', 'Remote Sensing', 'Texture'등의 Category에서는 SEL의 방법보다 2배 이상의 빠른 수행 속도를 보이고 있다.



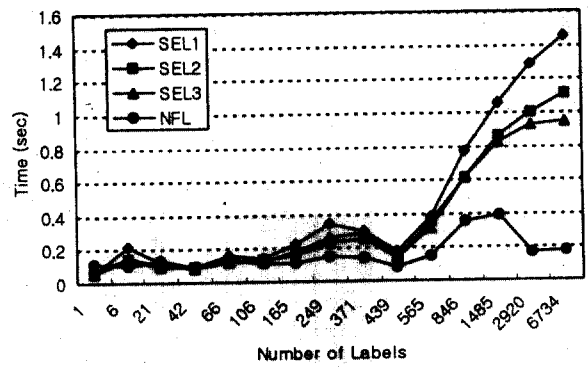
(그림 4) Comparision by Category

(그림 5)와 (그림 6)에서는 이미지의 크기 및 전체 label의 개수에 따른 수행 시간을 비교하고 있다. (그림 5)에서 보는 바와 같이 이미지가 작을 때는 거의 비슷한 성능을 나타내지만 이미지의 크기가 커짐에 따라 최대 5배까지의 속도 차이를 보이고 있다. 마찬가지로 (그림 6)에도 Label 개수가 작을 때는 어느 정도 비슷한 성능을 보이지만 Label의 크기가 커짐에 따라 제안하는 알고리즘의 수행 속도가 상대적으로 매우 빠름을 알 수 있다.

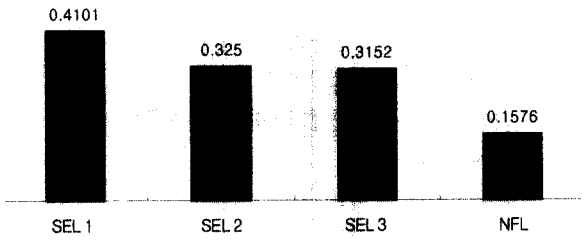
전체적으로 245개의 HIPR Image에 대한 평균 수행 시간은 (그림 7)에서 보는 바와 같이 NFL(0.1576) > SEL3(0.3152) > SEL2(0.325) > SEL1(0.4101)로써 2배 이상의 빠른 수행 속도를 보였다.



(그림 5) Comparision by Image Size



(그림 6) Comparision by Number of Labels



(그림 7) Comparision of Average Labeling Time

6. Conclusion and Future Work

본 논문에서는 기존의 Sequential Connected Component Labeling Algorithm과는 다른 접근방법으로 Non-recursive Flood-fill Labeling Algorithm을 제안하였다. 기존의 Labeling Algorithm의 구현을 위하여 필요했던 복잡한 Equivalent Label Processing의 과정을 필요로 하지 않으며, Flood-fill Algorithm과 간단한 LS(Label Stack)를 이용하여 쉽게 Labeling을 구현할 수 있으며 4-연결성에 의한 Labeling 뿐만 아니라 8-연결성에 의한 방법도 쉽게 처리할 수 있다. 처리 속도면에서도 HIPR 이미지로 테스트해본 결과 2배 이상의 빠른 수행 속도를 나타내었다. 또한 확장된 NFL 방법으로는 Labeling의 전처리 단계인 이진화 과정에서 발생할 수 있는 물체 정보 유실을 방지하기 위해서 gray-level로 확장될 수 있어 적응성있게 사용할 수 있음을 알 수 있다. 향후 연구 방향으로는 불필요하게 사용될 수 있는 LS(Label Stack)에 대한 메모리 최적화에 관련된 연구가 필요하며 확장된 NFL을 사용하여 gray-level 이미지에 대한 효과적인 영역분할(Region Segmentation)에 관련된 연구를 추진할 계획이다.

참 고 문 헌

[1] A. Rosenfeld, J. Pfaltz, Sequential Operations in Digital Picture Processing. Journal of the ACM, 13(4) : pp.471-494, October, 1966.
[2] A. Rosenfeld, A. C. Kak, Digital Picture Processing, Academic Press, Vol.2, pp.241-242, 1982.
[3] R. Gonzales, R. Woods, Digital Image Processing, Addison-Wesley, pp.42-45, 1992.
[4] R. Haralick, L. Shapiro, Computer and Robot Vision, Academic Press, Vol.2, pp.241-242, 1982.
[5] Luigi Di Stefano, Andrea Bulgarelli. A simple and efficient connected components labeling algorithm. Image Analysis and Processing, 1999.
[6] Edward Angel. Interactive Computer Graphics. 291, Addsion Wesley Longman, 1997.
[7] Ramesh Jain,et. Machine Vision, MacGraw-Hill, pp.44-47, 1995.

김 도 현
e-mail : unlimmit@hanmail.net
2001년 부산대학교 전자계산학과 졸업(학사)
2001년~현재 부산대학교 대학원 전자계산학과 석사
관심분야 : 패턴인식, 영상처리, 컴퓨터비젼, 신경회로망

강 동 구
e-mail : dkkang1@harmony.cs.pusan.ac.kr
2001년 부산대학교 전자계산학과 졸업(학사)
2001년~현재 부산대학교 대학원 전자계산학과 석사
관심분야 : 문자인식, 영상처리, 컴퓨터비젼, 자동화시스템

차 의 영
e-mail : eycha@hyowon.cc.pusan.ac.kr
1979년 경북대학교 전자공학과 졸업(학사)
1982년 서울대학교 대학원 전자계산학과 (이학석사)
1998년 서울대학교 대학원 컴퓨터공학과 (공학박사)
1981년~1985년 한국전자기술연구소 연구원
1995년~1996년 University of London 방문교수
1985년~현재 부산대학교 전자계산학과 교수
관심분야 : 컴퓨터비젼, 신경회로망 이론, 웨이블릿, 워터마킹