Union find를 이용한 faster watershed algorithm

Jaemin Lee

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING, SEOUL NATIONAL UNIVERSITY

Abstract

Watershed algorithm은 Computer vision 분야에서 image segmentation을 수행하는 한 방법이다. 기존의 Rainfall과 Flooding 방식의 Watershed algorithm의 경우 우선순위 큐를 이용한다. 그러나, 우선순위 큐를 이용하는 방식이 time-consuming 하기 때문에 본 연구에서는 union-find 기반의 새로운 방식의 Watershed algorithm을 제시한다. 같은 크기의 데이터에서 Union-find의 시간 복잡도는 O(N)으로 우선순위 큐의 O(Nlog(N)) 보다 효율적이다. 본 연구에서는 Union-find 방식의 알고리즘이 더 빠르고 이미지의 형태에 따른 편차 없이 작동함을 입증한다. 향후 연구에서 더 큰 이미지에서의 비교를 진행하고 over-segmentation을 해결할 것이다.

Introduction

Image Segmentation은 digital image를 다수의 픽셀 집합으로 나누는 것으로 Computer vision과 Image processing 분야에서 핵심적인 기술이다. Image Segmentation의 목적은 image를 가공하여 이용하고 분석하기 쉬운 형태로 변환하는 것이다. [1]

Watershed algorithm은 grayscale image에 적용하여 이미지를 지질학적 지도와 같이 밝기를 고도로 하여 그 경계를 나타내는 기술이다.[2] 기존의 방식의 경우 우선순위 큐를 이용한다. 기존 방식의 첫 단계에서는 local minima 또는 maxima를 찾아 우선순위 큐에 저장하게 된다. 이러한 접근은 인접 픽셀들을 조사하여 하나의 중앙 픽셀에 대한 local extremum여부만을 저장함으로 정보의 효율성이 떨어질 수 있다. 또한 이후에 전체 픽셀을 queueing하는 과정에서 O(Nlog(N))의 시간 복잡도가 요구된다. 또한 이미지에 포함된 local extrema 의 수에 따라 수행 시간의 차이가 발생한다.

새로운 방식의 경우 첫 단계에서 local extremum 의 여부가 아닌 window extremum 을 저장하고 이를 바탕으로 union-find하여 정보를 모두 활용하여 O(N)의 시간 복잡도로 같은 결과물을 산출하며 local extrema의 수에 영향 받지 않는 균일한 수행 시간을 가진다.

Methods

본 연구에서는 기존 방식과 새로운 방식을 C++ 언어로 구현하여 비교한다. 256x256 이미지들의 Segmentation을 수행시키며 식별 편의상 각 algorithm의 수행 완료 후 결과를 기존 이미지를 가로세로 4배 확대한 이미지 위에 경계를 표시함으로써 나타낸다.

METHOD1: IMMERSION BASED WATERSHED ALGORITHM.

// Step 1: Initialization
1. Create an empty queue Q
2. Create a label matrix M with the same size as the input image
3. Initialize a variable "label" to 0

// Step 2: Identify regional minima
4. For each pixel P in the input image:
- Find the neighbors of P (8-connectivity)
- Check if P is a regional minimum
- If P is a regional minimum:
- Assign a unique label to P in the label matrix M
- Add P to the queue Q

// Step 3: Flooding process
5. While Q is not empty:
- Dequeue a pixel P from Q
- For each neighbor N of P:
- If N is unlabeled in the label matrix M:
- Label N with the same label as P in the label matrix M
- Add N to the queue Q

// Step 4: Post-processing
6. For each pixel P in the input image:
- If P is unlabeled in the label matrix M:
- Assign a new label to P in the label matrix M

// Step 5: Output
7. Return the label matrix M

METHOD2: UNION-FIND BASED WATERSHED ALGORITHM.

// Step 1: Initialization
1. Create a label matrix M with the same size as the input image
2. Initialize a variable "label" to itself.

// Step 2: Identify regional minima
4. For each pixel P in the input image:
- Find the neighbors of P (8-connectivity)
- Check which pixel is the window minimum
- If P is not a regional minimum:
- Assign the window minimum pixel as the parent of P

// Step 3: Merging process
5. For each pixel P find and merge to the root (greatest parent) without sequence

// Step 4: Output
6. Return the label matrix M

Results

```
Method1(Priority_Queue): 0.876초
#of segment: 5193
Method2(Union-Find): 0.183초
#of segment: 5193
```

```
Method1(Priority_Queue): 1.152초
#of segment: 9500
Method2(Union-Find): 0.196초
#of segment: 9500
```

```
Method1(Priority_Queue): 1.224초
#of segment: 12229
Method2(Union-Find): 0.176초
#of segment: 12229
```

Figure 1: The duration and result number of segments after segmenting three 256x256 gray images. (From the top; Lena, Caman, NWU)

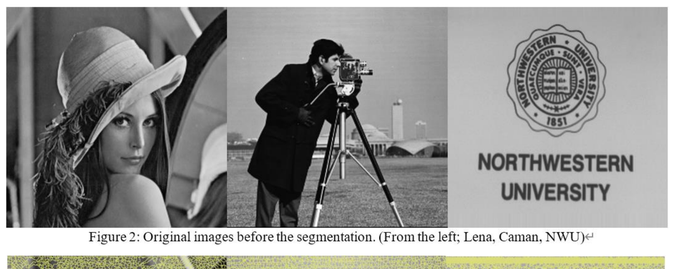

Figure 2: Original images before the segmentation. (From the left; Lena, Caman, NWU)

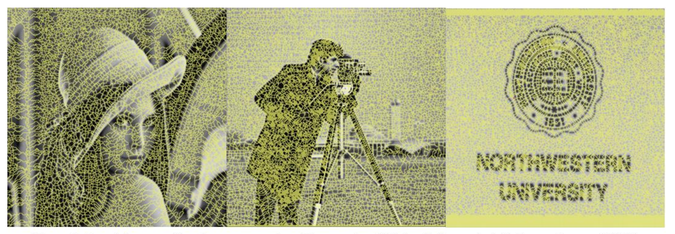

Figure 3: Result images after the segmentation. (Conventional Method) (From the left; Lena, Caman, NWU)

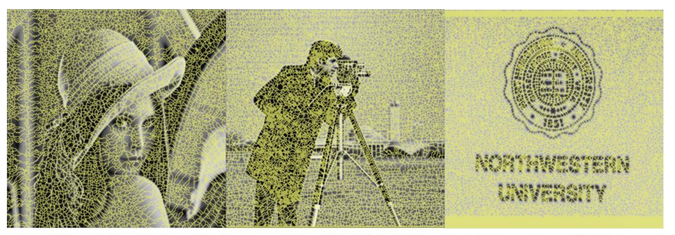

Figure 4: Result images after the segmentation. (New Method) (From the left; Lena, Caman, NWU)

두 방식의 경우 완전히 동일한 Segmentation 결과물을 산출했다. 256x256 size의 이미지 기준 새로운 방식이 5배 이상의 속도 향상을 이루었다. 이미지에 따른 수행시간 편차 또한 기존 방식에 비해 매우 개선되었다.

Discussion

알고리즘의 시간 복잡도 비교에서 예상할 수 있듯이 union-find 기반의 알고리즘이 기존 방식에 비해 큰 성능 향상을 이루었다. 또한 union-find 기반 알고리즘이 이미지에 따라 큰 편차를 보인 기존 방식과 비교하여 일정한 수행시간을 보여주었다. 이미지에 따른 비의존성은 머신러닝의 pre-processing과 같은 병렬 작업을 수행할 때 유의미한 성능 향상을 보일 것이다.

Conclusion

본 연구는 새로운 방식의 Watershed algorithm이 기존 방식에 비해 일반적으로 뛰어난 성능을 보이며 병렬 작업에서 특히 더 강한 성능을 보여줄 수 있음을 증명했다.

추후 연구에서 더 큰 size의 이미지에 대한 실험이 요구된다. 또한 over-segmentation에 대응할 수 있는 추가 작업의 구현이 필요할 것이다.

Reference

[1] L. G. Shapiro and G. C. Stockman, Computer vision. Pearson, 2001.

[2] R. Romero-Zaliz and J. Reinoso-Gordo, "An updated review on watershed algorithms," Soft computing for sustainability science, pp. 235-258, 2018

[3] B. Beucher and C. Lantuejour, "Use if watersheds in contour detection," in The international workshop on image processing: real time and motion detection/estimation, Rennes, France, 1979.

[4] J. Cousty, G. Bertrand, L. Najman, and M. Couprie, "Watersheds, minimum spanning forests, and the drop of water principle," 2007.

[5] B. A. Galler and M. J. Fisher, "An improved equivalence algorithm," Communications of the ACM, vol. 7, no. 5, pp. 301-303, 1964.

[6] J. E. Hopcroft and J. D. Ullman, "Set merging algorithms," SIAM Journal on Computing, vol. 2, no. 4, pp. 294-303, 1973.

[7] R. E. Tarjan and J. Van Leeuwen, "Worst-case analysis of set union algorithms," Journal of the ACM (JACM), vol. 31, no. 2, pp. 245-281, 1984.

[8] M. Fredman and M. Saks, "The cell probe complexity of dynamic data structures," in Proceedings of the twenty-first annual ACM symposium on Theory of computing, 1989, pp. 345-354