실시간 배경영상과 거리 Ranking을 통한 다개체 추적

서영욱, 차의영
부산대학교 컴퓨터공학과

Multi-Object Tracking using Real-Time Background Image and Ranking Distance Algorithm

Young-Uk Seo, Eui-Young Cha
Dept. of Computer Engineering, Pusan National University

요 약

본 논문은 제한된 영역 안의 다수 물고기를 추적하는 방법을 제안한다. 고정된 카메라로 물고기가 있는 수조의 영상을 얻은 다음 실시간으로 얻는 배경영상을 통해 물고기의 이미지만을 얻는다. 이렇게 얻어진 이미지를 ART2 알고리즘을 통해 clustering을 하고 각각의 물고기라 추정되는 cluster와 이전까지 측정되어진 물고기 좌표와의 거리 계산을 통해 각각의 물고기의 개체 인식을 하게 된다. 본 논문에서는 기존의 물고기 이미지를 얻는 방법을 개선하여 다 개체 추적을 위한 깨끗한 개체 이미지를 얻는 방법과, 각 cluster들과 이전 물고기 위치와의 거리계산을 통한 개체 인식 방법에 대해 초점을 맞추었다.

1. 서론

생물체의 움직임을 관찰하는 일은 생물학적 연구에 있어서 매우 중요한 일이다. 하지만 이러한 작업은 오랜 시간, 지속적으로 이루어져야 하므로 많은 시간과 집중력이 필요한 일이다. 이러한 일을 컴퓨터가 대신함으로서 정확성과 인력 단축의 효과를 볼 수 있다. 이미 이러한 생물체의 추적 시스템을 여러 논문에서 소개된 바 있다. 하지만 이전까지의 논문에서는 단일 개체 추적에 초점을 맞추거나 두 마리의 추적에 관한 짧은 소개에 마치고 있다. 생물학적 연구에 있어서 여러 개체들이 집단으로 서식할 때의 데이터는 단일 개체의 데이터보다 훨씬 많은 정보와 그 개체 집단의 사회성을 볼 수 있는 데 있어 매우 중요하다. 본 논문에서는 두 마리 이상의 다수 개체의 추적과 개체인식을 위한 방법론을 제시한다.

개체 추적을 하기 위해선 먼저 개체가 아닌 영역은 없애고 관심영역인 개체 이미지만 추출하여야 한다. 이전까지의 논문에선 이러한 개체 이미지를 얻는 방법으로 3프레임간의 차영상을 얻는 방법이 주류를 이루었다. 하지만 3프레임간의 차영상을 통한 방법은 3프레임 사이에 개체의 움직임이 적은 경우 개체의 이미지가 깨끗하지 못하거나 형태조차 나타나지 않게 된다. 이런 경우 한 마리의 개체 추적 시에는 그렇게 문제가 되지 않더라도 여러 마리의 개체 추적에는 개체 인식에 문제를 일으키게 된다. 또 다른 방법으론 배경영상을 얻는 방법이 있다. 배경영상을 얻는 방법은 개체가 없는 배경영상을 먼저 이미지로 얻은 다음에 추적을 하거나, 개체가 움직일 때 수동으로 영역을 설정하여 배경영상을 얻는 방법 등이 있다. 이러한 배경 영상으로 추적을 할 경우 얻는 이득은 개체가 움직임을 보이지 않아도 형체를 뽑아 낼 수 있다는 것이다. 하지만 배경영상을 얻기 위해 수동적으로 작업이 필요하고 한번 얻어낸 배경 영상을 이용한 방법은 시간에 따라 변화하는 외부 효과, 다시 말해 조명의 변화, 미세한 배경의 변화 등에 매우 민감한 반응을 보여 장시간의 사용은 어려운 점이 있었다.

본 논문에서 제안하는 방법은 이러한 문제점을 개선하여 배경영상을 실시간으로 얻어 내고 개체의 명암정보와 결합하여 노이즈에도 강하면서 보다 나은 개체 이미지를 얻는 방법을 소개한다. 이렇게 얻어진 이미지를 ART2알고리즘을 통해 clustering하여 각각의 정보를 가지고 물고기의 개체 인식을 하게 한다.

2장에서는 본 논문이 제안하는 깨끗한 개체 이미지

를 얻는 방법을 소개 하며 3장에서는 ART2 알고리즘을 이용한 clustering 방법, 4장에서는 ART2 알고리즘을 통해 구한 cluster들을 이용해 실제 개체와 거리 비교를 통해 개체 인식 방법을, 5장에서는 실험과 결과, 마지막으로 6장에서는 결론과 향후 연구 방향에 대해 언급하며 본 논문을 맺겠다.

2. 개체영상 획득

개체추적을 하기에 먼저 필요한 작업은 이미지에서 무엇이 추적되어 질 대상이고 무엇이 배경영상인지를 구분하는 일이다. 그러므로 깨끗한 개체영상 획득은 개체 추적에 있어 매우 중요한 작업이다. 개체영상 획득 방법은 여러 가지가 있겠는데 본 논문에선 배경영상을 실시간으로 추출하고 개체 특유의 명암 정보를 이용하여 개체 이미지를 추출하는 방법을 제안한다.

배경영상을 실시간으로 추출하는 방법은 물고기가 동적인 생물체라는 것을 이용한다. 어느 시간 간격 동안 어느 임의의 구간에서 볼 수 있는 이미지 정보는 동적인 물체보다는 배경 영상이 훨씬 많다. 이를 이용하여 각 pixel 정보를 누적시킨 다음 평균을 함으로써 동적인 생물체의 pixel 정보는 소멸시키고 배경영상의 pixel 정보만 남게 하는 기법을 이용하여 배경영상을 추출하게 된다. 실제 적용한 식은 다음과 같다.

$$BI_n(x,y) = \begin{cases} \dfrac{CI_n(x,y) + BI_{n-1}(x,y)*(n-1)}{n} & (n < k) \\ \dfrac{CI_n(x,y) + BI_{n-1}(x,y)*(k-1)}{k} & (n \geq k) \end{cases}$$

BI : 배경영상, CI : 현재영상, x,y : pixel 좌표
n : 현재프레임번호
k : CI정보를 보장하기 위한 상수 값

위의 식에서 k의 존재는 현재영상정보의 보장을 위해 설정한 상수이다. n이 커질수록 현재 영상의 정보는 거의 0에 가깝게 되어 배경영상에 더 이상 변화를 줄 수가 없게 된다. 이럴 경우 처음 배경영상을 구할 때부터 개체의 움직임이 없었다면 개체 자체가 배경영상이 된다. 그리고 조명이 변하거나 미세한 배경의 변화가 생기더라도 배경정보를 더 이상 바꿀 수 없어 noise를 남기게 된다. 이것을 방지하기 위해 배경영상에 포함되는 현재영상정보를 줄여나가다가 현재영상정보가 배경영상에 미세한 영향을 주는 선에서 현재영상정보를 줄여주는 값을 고정시키게 해줌으로써 해결하게 하였다. 그림 1은 실제 어항 영상과 실시간으로 얻은 배경영상을 보여주고 있다. 그림을 보면 물고기가 있던 자리엔 물고기의 어렴풋한 형태만 남게 되고, 물고기가 지나왔던 자리에 희미한 선을 남기는 현상을 볼 수 있다.

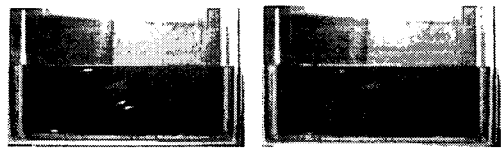

그림 1 (a)실제 어항영상 I (b)실시간으로 얻은 배경영상

이렇게 구해진 배경영상과 현재영상의 차영상을 구하게 되면 물고기의 형태를 뽑아낼 수는 있지만 희미한 선까지 남게 되는 현상이 생긴다. 이러한 선을 없애기 위해 물고기가 배경보다 밝다는 점을 이용하여 물고기라고 보기엔 어두운 pixel 정보를 모두 없애 버린다. 공식은 다음과 같다.

$$RCI_n(x,y) = \begin{cases} CI_n(x,y) & (CI_n(x,y) \geq LC) \\ 0 & (others) \end{cases}$$

RCI : 명암정보를 이용해 배경을 지운 이미지
LC : 물고기 명암정보의 하한값

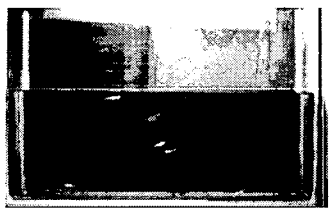

그림 2 명암 정보를 이용해 얻은 이미지

그림 2는 위의 공식으로 구해진 명암 정보를 이용한 이미지이다. 현재영상과 구해진 배경영상과의 차영상을 명암 정보를 이용해 얻은 이미지와 AND 연산을 함으로써 최종적인 물고기 이미지를 구하게 된다.

$$FI = (CI - BI) * RCI$$

FI : 구하고자 하는 물고기 이미지

그림 3 최종적으로 얻어진 물고기 이미지

3. ART2 알고리즘을 이용한 Clustering

실시간 배경영상 획득을 이용해서 물고기만 남아 있는 이미지가 추출되었다면 이제 전체 이미지에서 각각의 물고기들을 이루고 있는 pixel들을 Clustering 하여야 한다. 이를 위해 ART2 알고리즘을 이용하여 각 물고기들을 Clustering하고 중심좌표를 얻게 하였다. ART2 알고리즘은 입력 데이터의 Clustering에 유용한 방법으로 입력 데이터와 cluster 중심과의 차이 값을 이용한 방법이다. 간단히 ART2알고리즘의 내용을 살펴보기로 하자.

$$j^* = \min\|X - C_j\|$$

if $\|X - C_{j^*}\| < \rho$ (Vigilance Test)

$$\begin{cases} true: & C_{j^*}^{new} = \dfrac{X + C_{j^*}^{old}\|cluster_{j^*}^{old}\|}{\|cluster_{j^*}^{old}\| + 1} \\ false: & C_k \end{cases}$$

X : Input data C_j : Center of class j

ρ : cluster radius C_{j^*} : New Center of class j

C_k : new cluster

한 이미지에 대해 ART2 Clustering이 끝나고 나면 k 개의 cluster들이 생기게 되며 각각의 중심 좌표와 그 cluster를 이루고 있는 좌표의 개수 정보를 가지게 된다.

4. 개체 인식을 위한 Distance Ranking Algorithm

실시간으로 우리가 원하는 개체라고 생각되어 지는 cluster들을 추출할 수 있다면, 각 cluster들이 어떠한 개체를 나타내는지를 추정함으로써 다 개체 추적이 끝나게 된다. 여기엔 여러 가지 방법이 쓰일 수 있겠지만 실시간 처리를 고려하여 기본적인 방법인 거리 값 비교로 각 개체를 구분하도록 하였다. 거리 값을 이용하는 개체 인식 방법은 이전 cluster와 현재 cluster의 좌표 정보가 있을 때 가장 가까운 것끼리 짝을 맞춰주는 방법이다. 여기서 다 개체 추적 시 하나 더 고려해야 할 것은 무조건 거리가 가장 짧다고 해서 짝을 맞춰주게 되면 문제가 생기게 된다.

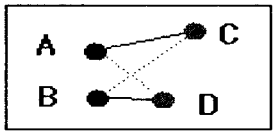

그림 4 Cluster 간 Matching

그림 4를 보면 A와 B cluster가 이전 프레임에서 확인되었고, 현재 C와 D cluster가 있다고 가정하자. 이 때 A와 B 입장에선 D와 가장 짧은 거리를 가지게 된다. 만약 시스템에서 A부터 C, D 와의 거리 값을 계산하여 처리하면 A-D, B-C 로 짝을 이루게 될 것이다. 하지만 D 입장에선 A가 아닌 B와 가장 가까운 거리를 가지고 있고, C 입장에선 B가 아닌 A와 가장 가까운 거리를 가지고 있다. 올바른 계산이 되려면 A-C, B-D처럼 묶여져야 할 것이다. 이러한 문제점을 해결하기 위해 시스템이 먼저 처리하는 cluster가 없도록 이전 cluster들과 현재 cluster들과의 거리 값을 모두 계산하여 2차원 Table에 저장한 다음 가장 짧은 짝을 찾아 cluster를 matching시켰다.

다음과 같이 실제 개체라 생각되는 좌표와 현재 프레임에서 얻은 cluster간의 모든 거리를 2차원 Table에 기록한다.

$$Table = \begin{matrix} d_{RO_1-C_1} & \cdots & d_{RO_1-C_j} & \cdots & d_{RO_1-C_M} \\ \vdots & & \vdots & & \vdots \\ d_{RO_i-C_1} & \cdots & d_{RO_i-C_j} & \cdots & d_{RO_i-C_M} \\ \vdots & & \vdots & & \vdots \\ d_{RO_N-C_1} & \cdots & d_{RO_N-C_j} & \cdots & d_{R_N-C_M} \end{matrix}$$

RO : 실제 개체라 생각하는 좌표

C : 현재 프레임에서 얻어낸 cluster

N : RO 의 개수

M : C 의 개수

d : RO 와 C와의 거리

이 상태에서 각 row내에서 정렬을 시킨다. 예로,

```
        -->
3 2 5       2 3 5
4 1 7       1 4 7
4 4 2       2 4 4
```

(거리 값 sorting)

```
2 1 3
2 1 3
3 1 2
```

(cluster index)

와 같이 바꾸어 준다. 이 상태가 되면 각 RO에서 가장 가까운 cluster부터 순서대로 나열해 놓은 결과가 된다. 이 상태에서 첫 번째 column부터 마지막 column까지 옮기면서 가장 거리가 짧은 짝끼리 맞춰주게 된다. 위의 예를 다시 보면, 각 column의 거리 순위는 다음과 같이 된다.

2 4 8
1 5 9
3 6 7

1순위부터 마지막 순위까지의 실제 개체와 cluster를 짝을 맞춰주고 이미 짝이 생긴 cluster는 건너뛰는 방법을 사용한다. 예에서 1순위인 Table(2,1) RO-C 짝은 cluster 2와 짝을 이루게 된다. 2순위인 Table(1,1) 역시 cluster 2와 가장 가깝지만 cluster2는 먼저 짝을 이룬 cluster이기 때문에 다음 column에서 짝을 찾아야 한다. 이런 식으로 전체 Table에서 짝을 맞춰주고 난 후 RO와 cluster의 개수가 맞지 않는 경우 남는 cluster가 있을 경우 새로운 개체 RO_{N+1}를 만들어 주며 N은 하나 증가시켜주고 반대로 남는 RO가 있을 경우 개체가 정지해서 cluster가 생기지 않았다고 보고 이전 좌표를 그대로 가지게 된다.

5. 실험 및 결과

실험은 Pentium4 2.0GHz 시스템에서 흑백 CCD 카메라를 이용해 초당 4프레임의 320 * 240 pixel 영상을 입력받아 실험하였다. 물고기는 송사리를 이용하였고 개체 수는 2 마리에서 9 마리까지 다양하게 실험하였고, 밝은 조명과 어두운 조명을 구분하여 실험하였다. 실험 결과 개체가 포개어 지는 경우만 빼고는 개체 인식과 추적을 잘 하는 것을 보여주며 개체가 정지할 경우 기존 방식인 3프레임 차영상은 그 즉시 cluster를 나타내지 못하지만 독특한 배경추출로 인해 잔상 현상이 일어나 개체 정지 후에도 약 3~4초간 개체 위치를 파악할 수 있는 장점이 있었다. 마지막으로 어두운 조명보단 밝은 조명에서 더욱 좋은 결과가 나왔다. 이는 밝은 조명에서 물고기와 배경의 차이가 두드러지기 때문이다. 문제점으론 개체가 포개어져 하나의 cluster를 이룰 경우 하나의 개체만 추적되어 지고 다른 하나는 그 자리에 정지해 있는 것으로 판단하는 문제점이 있었다.

그림 5는 송사리 7마리를 실험에 사용한 모습을 나타내고 **그림 6**은 개체라 생각되어 지는 좌표를 각기 다른 색의 점으로 찍은 모습을 나타내고 있다.

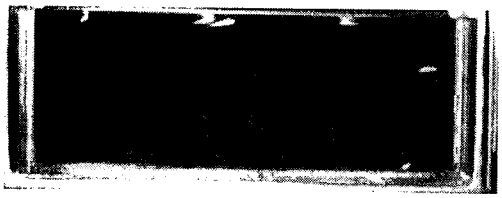

그림 5 실제 어항 영상II

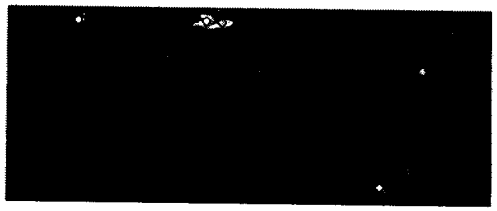

그림 6 다 개체 추적 영상 결과

6. 결론 및 향후 연구과제

본 논문은 기존의 실시간 물고기 추적에서 쓰인 방법에서 탈피하여 독특한 배경영상을 추출하여 각 개체의 이미지를 개선하였고, Distance Ranking Algorithm을 통해 각 개체 인식이 가능하도록 하여 다수의 물고기 추적이 가능하도록 하였다.

향후 개선방향은 개체 겹침으로 인한 문제의 해결이 필요하고, 더욱 정교한 개체 인식을 위하여 거리 값뿐만 아니라 방향성, 개체 크기 등을 이용할 수 있는 방향으로 연구가 진행되어 져야 할 것이다.

[참고문헌]

[1] 김홍수, 차의영, 전태수,"배경 갱신과 반복 Clustering을 이용한 물고기 추적",한국 정보 처리 학회, 제 6권 1호, pp.1375-1378, 1999

[2] 강명호, 오그니언 토파로프, 차의영,"자기 조직화 신경망을 이용한 다중 표적 추적에 관한 연구", 한국정보과학회논문지 제 24권 2호 pp.525-528

[3] *Oron,E.*"Motion estimation and image difference for multi-object tracking"Aerospace Conference, 1999. Proceedings. 1999 IEEE , Volume: 4, 1999 pp.401-409

[4] Hwann-Tzong Chen,Horng-Horng Lin,Tyng-Luh Liu,"Multi-Object tracking using dynamical graph matching"Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on , Volume: 2, 2001 pp. II-210-II-217 vol.2

[5] Laurene Fausett,"Fundamentlas of Neural Networks"Prentice Hall, 1994 pp.246-282