다중 카메라를 이용한 3차원 개체 추적 시스템

이상걸*, 구경모*, 서영욱*, 차의영*
*부산대학교 컴퓨터공학과
e-mail:jalnaga@nate.com

A 3D Object Tracking System Using a Multi-camera

Sang-Geol Lee*, Kyung-Mo Koo*, Young-Wook Seo*, Eui-Young Cha*
*Dept of Computer Engineering, Pusan National University

요 약

본 시스템은 어항속에 있는 물고기 움직임을 추적하기 위해 두 대의 카메라로부터 동시에 독립된 영상을 획득하고 획득된 영상을 처리하여 좌표를 얻어내고 3차원 좌표로 생성해내는 시스템이다.

제안하는 방법은 크게 두 대의 카메라로부터 동시에 영상을 획득하는 방법과 획득된 영상에 대한 처리 및 물체 위치 검출, 그리고 3차원 좌표 생성으로 구성된다. Frame grabber를 사용하여 두 개의 카메라로부터 동시에 영상을 획득하며, 3개의 연속된 프레임에 대한 차영상과 ART2(Adaptive Resonance Theory)를 이용하여 각각의 영상에서의 물고기 위치를 검출한다. 검출된 각각의 좌표를 병합하여 3차원 좌표를 생성하며, 추적 결과는 OpenGL을 이용하여 3차원으로 재생한다.

1. 서론

우리가 살고 있는 이 지구에는 수많은 종류의 생물체들이 함께 살고 있다. 그 수많은 생물체들은 각기 다른 환경에서 살며, 그들만의 독특한 행동을 지니고 있다. 이런 특정한 생물체들의 행동이 어떤 특정 상황에서 어떻게 나타나는지 관찰, 분석해 봄으로써 여러 정보들을 추출해내는 연구들이 진행되어 왔다[1,2,3].

이런 생물체들의 행동을 분석하기 위해서는 생물체의 움직임을 관찰해야 한다. 움직임 관찰 대상에는 여러 가지가 있을 수 있으나, 본 논문에서는 물속의 물고기를 대상으로 하였다. 이때 물고기가 물속에서 다닌 궤적을 물고기의 움직임으로 관찰해 볼 수 있다. 이 움직임을 사람이 계속해서 관찰하기엔 문제가 많으므로, 움직임을 정보화해야 할 필요성이 있다. 따라서 물고기의 행동을 분석하는데 가장 기본이 되는 움직임을 추적하는 시스템이 필요한 것이다.

물고기 추적 문제에 있어 여러 가지 해결 방법을 생각해 볼 수 있지만 본 논문에서는 컴퓨터와 두 대의 카메라를 이용하는 방법을 제안한다. 기본적으로 카메라 한 대를 이용하여 2차원으로 움직임 궤적을 추적할 수도 있다[4]. 하지만, 물고기의 움직임이 평면상이 아닌 공간상에서 이루어지는 것이므로 보다 실제적이고 정확한 분석을 위해서 본 논문에서는 3차원 좌표 데이터를 생성하는 시스템을 제안한다.

2. 전체 시스템 개요

전체 시스템 개요는 그림 1과 같다.

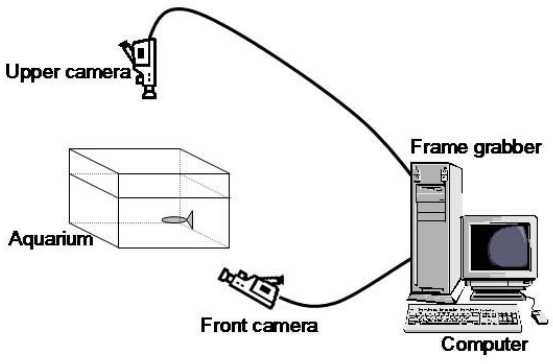

그림 1. 3차원 물고기 추적 시스템의 개요

먼저, frame grabber를 통하여 두 대의 카메라로부터 영상을 획득한다. 획득한 영상은 연속된 3개의

프레임으로 보관한다. 각 카메라별 영상에 대하여 연속된 3개의 영상을 가지고 영상 처리와 ART2를 이용하여 물고기의 위치를 검출하게 된다. 이때 움직임과 잡영에 대한 임계치를 두어 잡영을 처리한다. 이렇게 구해진 각각의 좌표를 병합하여 3차원 좌표로 생성하게 된다. 그림 2는 제안하는 시스템의 구조를 도식화하여 보여준다.

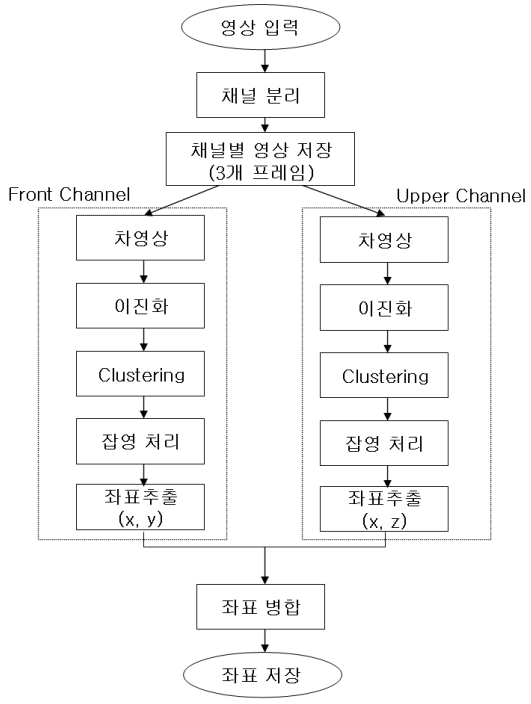

그림 2. 추적 시스템의 구조

3. 영상 획득 및 저장

본 논문에서 제안하는 방법으로 3차원 물고기 추적 시스템을 구현하기 위해서 두 대의 카메라를 이용한다. 이때 두 대의 카메라로부터 영상 획득시 두 영상이 동일한 시점에 동기화 되어 이루어져야만 한다. 따라서 본 시스템에서는 Matrox사의 Meteor-II/Multi-Channel frame grabber를 사용하여 신호를 동기화 하였다. 이 frame grabber는 R, G, B 세 개의 channel을 지원하여 컬러 영상을 각 채널별로 분리해서 받을 수 있는 특징이 있다. 이 특징을 이용하여 R, G Channel에 각각 흑백 CCD 카메라를 연결하고 두 대의 카메라를 Gen-Lock 기능을 이용하여 동기화시켜 동시에 영상을 획득할 수 있도록 하였다.

Meteor-II/Multi-Channel frame grabber를 제어하는데는 MIL(Matrox Image Library)을 이용하였으며, MIL에서 제공되어지는 함수를 이용하여 frame grabber와 카메라를 제어하여 640×480 크기의 영상을 두 대의 카메라로부터 획득하였다.

Frame grabber를 통하여 영상 입력을 받으면 R, G, B 세 개의 채널이 합쳐진 형태로 획득된다. 이 영상을 다시 channel별로 분리하여 R, G channel만 뽑아서 메모리에 저장을 한다. 여기서 R channel은 front channel, G channel은 upper channel로 정의한다.

이렇게 구해진 front, upper channel의 영상으로 channel 별로 3개씩의 frame을 만든다. previous1, previous2, current로 구성되는 3개의 frame에 저장하게 된다. 최초로 영상 입력을 받는 경우 previous1에 저장하고 연속된 다음 영상을 받아 previous2에 저장한다. 계속해서 다음 영상을 받아 current에 저장한다. 이 후 반복해서 영상을 받을 때는 시스템의 성능을 높이기 위해 3번을 모두 다시 받아 저장하는 것이 아니라, 가장 오래된 frame을 삭제하고 뒤의 두 frame과 새로 획득한 영상으로 3개의 frame을 구성한다. 즉 previous2를 previous1에 저장하고 current를 previous2에 저장하고 영상 입력을 받아 current에 저장한다.

4. 물고기 위치 검출

4.1 물고기만 남은 영상 획득

이제 3개의 frame을 가지고 물고기의 위치를 추출할 수 있도록 영상 처리를 수행한다. 먼저 previous1(p1)과 previous2(p2)의 차영상을 구하고 previous2와 current(c)의 차영상을 구한다.

$$DP_{p1,p2}(x, y) = |p1(x, y) - p2(x, y)| \quad (1)$$

$$DP_{p2,c}(x, y) = |p2(x, y) - c(x, y)| \quad (2)$$

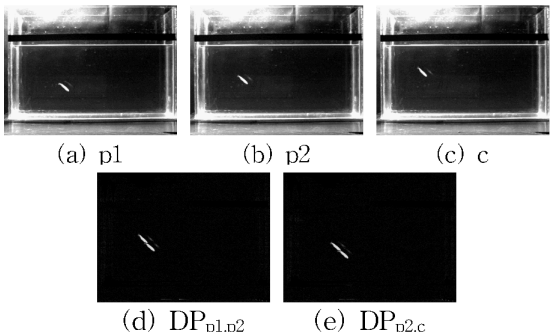

그림 3. 차영상 처리 과정

이렇게 구해진 두 개의 차영상을 각각 이진화 처리를 한다. 이때 이진화 임계치를 두어 임계치를 기준으로 0(검은색) 또는 255(흰색)으로 이진화 한다. 그리고 구해진 두 개의 이진화 영상을 교집합하면 움직임이 있는 부분의 영상만 남게 된다. 움직임이 있는 부분은 흰색(255) pixel이고 그렇지 않은 부분

은 검은색(0) pixel이다. 다음 식에 의해 구해진 두 개의 차영상을 각각 이진화한다.

$$F_T[i,j] = \begin{cases} 0 & \text{if } F[i,j] \le \theta \\ 255 & otherwise \end{cases} \quad (3)$$

(θ : 이진화 임계치)

그리고 이진화 된 영상을 각각 B1, B2이라고 하면 최종 영상 F는 다음과 같다.

$$F = B1 \bigcap B2 \quad (4)$$

(a) B1($DP_{b1,b2}$) (b) B2($DP_{b2,c}$)

(c) 최종 영상 F

그림 4. 이진화 처리 과정

4.2 ART2를 이용한 cluster의 생성

차영상 기법을 통해서 얻은 움직임이 있는 부분만 남은 영상에 대해서 이제는 잡영을 제거하고 물고기 부분을 찾아서 물고기의 중심점을 얻는 것이 필요하다. 이를 위해 ART2 알고리즘을 이용한 clustering 기법을 사용하였다.

앞의 과정을 통해서 움직임이 있는 부분은 영상에서 pixel이 흰색(255), 아닌 부분은 검은색(0)으로 처리되어 있는 상태다. 입력 데이터는 흰색 pixel들이 되며 각 흰색 pixel들의 거리를 가지고 ART2 알고리즘을 사용하여 cluster를 생성한다. 이렇게 생성된 cluster는 각각 중심점과 member수를 가지게 된다.

ART2 알고리즘을 사용한 cluster 생성은 다음의 4단계로 이루어진다.

step1. 새로운 입력 데이터(흰색 pixel)가 주어지면, 최소 거리의 cluster를 승자로 선택

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

(X : Input data, W_j : Weight of class j)

step2. Vigilance test를 수행

$$\| X - W_{j^*} \| < \rho$$

(ρ : cluster radius)

step3. Vigilance test를 실패하면 다음과 같은 weight값을 가지는 cluster를 생성한다.

$$W_k$$

(k : new cluster)

step4. Vigilance test를 통과하면 승자 cluster에 입력데이터를 포함시키고, 다음 식에 의해 weight값을 수정한다.

$$W_{j^*}^{new} = \frac{X + W_{j^*}^{old} \| cluster_{j^*}^{old} \|}{\| cluster_{j^*}^{old} \| + 1}$$

($\| cluster_j \|$: number of cluster$_j$ member)

이때 cluster radius ρ를 잘 지정해 주어야 한다. ρ값이 너무 크면 모든 pixel들이 하나의 cluster에 다 포함될 수도 있다. 이렇게 되면 물고기로 판정된 cluster의 중심점이 잡영쪽으로 영향을 받아 정확한 좌표 획득이 어렵게 된다. 반대로 ρ값이 너무 작으면 모든 pixel들이 각각의 cluster로 생성될 수 있다. 이렇게 되면 어떤 cluster를 물고기로 판정해야할지 모호해진다. 따라서 물고기 크기를 고려한 ρ값을 지정해야하며 본 시스템에서는 실험적으로 구한 값을 사용하였다.

5. 좌표 선정 및 저장

Clustering까지 마치게 되면 어떤 cluster를 물고기로 볼 것인지 판정하고 좌표를 추출하면 된다. 먼저 물고기 판정은 물고기의 크기가 잡영들 보다는 크기 때문에 생성된 cluster 중 member수가 가장 큰 cluster를 물고기라고 판정한다. 그리고 잡영 임계치를 두어 member수가 잡영 임계치 보다 작은 경우 잡영으로 간주하고 제외시킨다.

그런데 이렇게만 처리하여 물고기를 판정하게 되면 물고기가 움직이지 않고 잡영만 발생하였을 경우 잡영을 물고기로 간주하게 되는 경우가 발생한다. 차영상 기법을 이용하여 움직이는 부분을 찾아내기 때문에 물고기의 움직임이 없을 경우 흰색 pixel들이 나타나지 않게 된다. 이를 처리하기 위해 움직임 임계치를 두어 t-1에 추출된 좌표와 거리를 계산하여 이 임계치를 벗어나면 잡영으로 간주하고 t-1의 좌표를 유지시킨다.

이렇게 잡영에 대한 처리까지 완료하여 새로운 cluster가 물고기로 판정되면 그 cluster의 중심점(weight값)을 물고기의 좌표로 지정하게 된다.

앞의 과정을 통해서 좌표 추출이 이루어지면 그 좌표들을 파일에 저장하여 데이터를 수집하면 된다. Front와 Upper channel 각각 위의 과정을 일정시간 간격으로 수행하여 좌표를 추출해 낸다. 그러면

front channel에서는 x, y 좌표가 추출되고, upper channel에서는 x, z 좌표가 추출된다. 두 개의 좌표 중 x 좌표가 중복되므로 본 시스템에서는 upper channel에서는 z 좌표만 사용하여 병합함으로써 x, y, z 좌표가 생성되고 파일에 시간순서에 따라 기록하게 된다.

6. 실험 및 결과

본 논문에서 제안된 방법으로 3차원 물고기 추적 시스템을 Pentium4 2.0GHz, 512MB RAM의 PC 환경에서 Visual C++ .NET으로 구현하고 실험하였다. 실험에 사용된 물고기는 송사리이며 250ms 간격으로 추적을 하였다.

그림 5. 추적 과정 화면

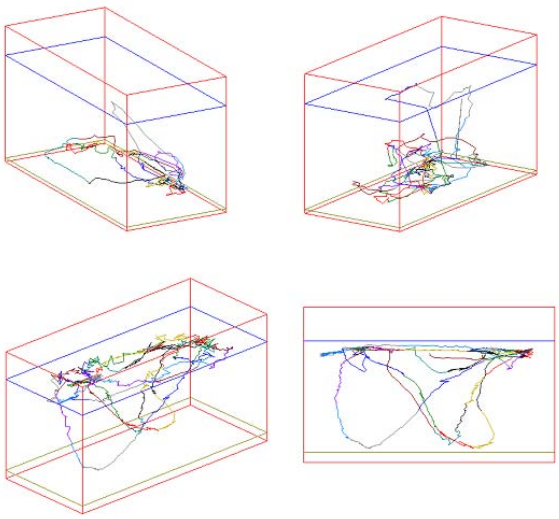

그림 6. 추적 결과 재생

추적 결과 재생은 OpenGL을 이용하여 3차원으로 재생 가능토록 하였으며 그림 6에서 보듯이 다양한 시점으로 볼 수 있다.

7. 결론 및 향후 과제

본 논문은 생물체들의 행동을 분석하고 유용한 정보를 얻어내려는 것에 입각하여 컴퓨터를 이용하여 물고기의 움직임을 3차원으로 추적하여 데이터를 수집하는 시스템을 제시하였다. 단순히 한대의 카메라를 이용하여 2차원 평면상에서 움직임을 추적한 것 보단 더 유용한 정보를 추출하는데 이용될 수 있으리라 본다.

차영상 기법을 이용하여 물체를 추적하는 방법외에 배경 추출을 통한 물체 추적 방법도 있다[4]. 본 시스템 처음 제작시에 배경 추출 기법을 사용하였으나, upper channel의 경우 수면을 촬영하게 되어 물의 반사에 민감하여 잡영 처리가 매우 힘든 문제가 있었으며, 장시간 추적시 물의 증발로 인한 수면 감소, 빛의 세기 변화 등으로 배경 갱신이 수반되어야 했다. 그러나 배경 갱신을 하게 되면 추적 도중 데이터가 연속되지 못하고 잃게 되어 끊기는 문제점이 발생하게 된다. 이러한 이유로 3차원 추적에서는 차영상 기법이 훨씬 효율적이다고 판단하게 되었다.

본 시스템에는 향후 해결해야할 두 가지 문제점이 있다. 첫째, 데이터의 정확도를 높이기 위해서 원근감에 대한 보정이 필요하다. 실험 결과 영상을 보면 알 수 있듯이 카메라로 촬영된 영상이 원근감을 가진 영상이나 이것을 평면으로 간주하고 좌표를 처리함으로써 오차가 존재하게 된다. 이것을 보정해야 하는 문제가 있다. 둘째, 추적된 데이터만으로는 특별한 가치가 없다. 따라서 추적 정보를 유용하게 이용하기 위한 추가의 시스템이 필요하다. 예를 들어, 서론에서 언급한 수질 오염을 검출하는 시스템을 구성하려고 한다면 추적된 결과를 바탕으로 특정 행동 패턴을 분석하여 패턴을 인식하도록 하는 시스템이 필요할 것이다.

참고문헌

[1] Alt, W., Hoffman, G., "Biological Motion," Lecture notes in Biomathematics. Springer-Verlag, Berlin, 1989.

[2] Tourtellot, M.K., Collins, R.D., Bell, W.J., "The Problem of movelength and turn definition in analysis of orientation data," Journal of Theoretical Biology 150, pp.287-297, 1991.

[3] 김철기, 김광백, 차의영, "다층 퍼셉트론을 이용한 유해물질 유입에 따른 송사리의 행동 반응 분석 및 인식," 멀티미디어학회 논문지 제6권 제6호, pp.1062-1070, 2003.

[4] 김홍수, 차의영, 전태수, "배경 갱신과 반복 Clustering을 이용한 물고기 추적," 한국정보처리학회 춘계학술발표논문집, pp.1375-1378. 1999.