The Journal of The Institute of Internet, Broadcasting and Communication (IIBC)
Vol. 14, No. 5, pp.243-248, Oct. 31, 2014. pISSN 2289-0238, eISSN 2289-0246

http://dx.doi.org/10.7236/JIIBC.2014.14.5.243
JIIBC 2014-5-34

3D센서의 Depth frame 데이터를 이용한 이동물체 감지

Detection of Moving Objects using Depth Frame Data of 3D Sensor

이성호*, 한경호**

Seong-Ho Lee*, Kyong-Ho Han**

요 약 외부 광원여부에 상관없이 3D정보를 수신할 수 있는 Microsoft의 3D 모션 센서인 키넥트의 Depth frame을 사용하여 물체의 움직임 영역을 감지할 수 있는 방법을 연구하였다. 센서로부터 수신되는 Depth 정보 중 주로 물체의 경계면에 존재하는 노이즈를 제거하기 위해 픽셀의 x, y좌표에 대한 블러링 기법과 z좌표에 대한 주파수 필터를 적용하였다. 또한 인접 픽셀들의 변화량에 따른 군집화 필터를 적용함으로서 움직이는 물체 영역을 추출할 수 있었고 필터 설정에 따라 기준이상의 빠른 움직임을 감지할 수 있도록 하여 이동형 로봇에 응용할 수 있도록 하였다. 특히 IR 카메라에 의하여 만들어지는 Depth frame을 이용함으로써 주야간 모두 제약 없이 사용할 수 있다. 또한 직진 방향으로 움직이는 물체에 대해서도 입체적으로 감지할 수 있어 무인 로봇영역에 응용할 수 있다.

Abstract This study presents an investigation into the ways to detect the areas of object movement with Kinect's Depth Frame, which is capable of receiving 3D information regardless of external light sources. Applied to remove noises along the boundaries of objects among the depth information received from sensors were the blurring technique for the x and y coordinates of pixels and the frequency filter for the z coordinate. In addition, a clustering filter was applied according to the changing amounts of adjacent pixels to extract the areas of moving objects. It was also designed to detect fast movements above the standard according to filter settings, being applicable to mobile robots. Detected movements can be applied to security systems when being delivered to distant places via a network and can also be expanded to large-scale data through concerned information

Key Words : 3d camera, 3d sensor, Kinect, Depth frame, Motion detection, Robot

Ⅰ. 서 론

3D카메라가 3차원 정보를 획득하기 위한 방법은 크게 접촉식과 비접촉식으로 나뉘어지며 비접촉식 방식은 다시 TOF (Time of flight) 방식과 스테레오 방식 그리고 구조광 방식(structured light)으로 나뉘어진다.

2010년 12월에 출시된 Microsoft 키넥트 센서는 게임 콘솔인 XBOX 360의 주변기기로 출시되어 사람의 움직임을 감지하여 게임을 컨트롤 하는 UI(User interface)로 사용되고 있다. 구조광 방식을 사용하는 키넥트는 기존의 3D 카메라들에 비해 저렴한 가격으로 출시되어 3D 카메라의 대중화를 기대할 수 있으며 그에 따른 연구가 활발히 진행되고 있다.

특히 2011년 Windows OS용으로 출시된 Kinect for Windows는 Near Mode를 지원하여 센서와 인체간 거리 확보가 안된 일반 컴퓨터 사용 환경에서 사용할 수 있게

*준회원, 단국대학교 전자전기공학부
**정회원 단국대학교 전자전기공학부
접수일자: 2014년 8월 25일, 수정완료: 2014년 9월 25일
게재확정일자 : 2014년 10월 10일

Received: 25 August, 2014 / Revised: 26 September, 2014
Accepted: 10 October, 2014
*Corresponding Author: ianlee@dankook.ac.kr
Dept. of Electronics and Electrical Engineering, Dankook University, Korea

되었으며 2012년 10월부터 제공된 Microsoft Kinect SDK 1.6을 사용할 경우 IR카메라를 통해 전달되는 영상 정보를 사용할 수 있어 조도가 낮아 RGB카메라의 사용이 불가능한 곳에서도 이미지 촬영이 가능하게 되었다.

a.RGB Mode b.IR Mode

그림 1. Kinect 의 RGB영상과 IR영상
Fig. 1. RGB, IR Data of Kinect Sensor

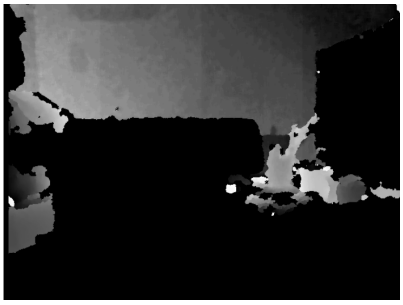


그림 2. Kinect 의 Depth Stream 영상
Fig. 2. Depth Stream picture of Kinect sensor

Microsoft 사의 키넥트 센서는 RGB카메라를 통해 수신되는 Color Stream과 Depth 센서를 통해 수신되는 Depth Stream이 별개의 모듈로 동작하며 RGB카메라를 가려도 Depth Stream 수신에는 문제가 없다. 역시 RGB 영상 역시 IR 송광부나 IR카메라를 가려도 Color Stream을 획득하는 데에는 문제가 없으나 Color Stream이 IR모드로 동작할 때에는 IR송광부와 IR카메라를 통하여 Color Frame과 Depth Frame이 수신되므로 IR송광부나 IR카메라를 가릴 경우 수신되지 않는다.

본 논문에서는 조도 및 외부의 광원에 상관없이 사용할 수 있는 Depth Stream을 이용하여 움직이는 물체를 감지하는 실험을 하였다.

II. 노이즈 제거를 위한 기본필터링

1. 기본필터링 구현 방법

본 연구에서는 Windows7 OS가 설치된 PC에 Kinect for Windows SDK v1.6 를 설치한 후 Kinect로 부터 Depth Stream을 해상도 320 * 240, 속도 30fps로 수신 하였다. 수신되는 Depth Stream에서 원하는 이동 물체를 감지하기 위해 움직이지 않는 영역에서 발생되는 노이즈를 제거한 후 최종으로 물체의 군집 영역 데이터를 획득하였다. 노이즈를 제거하기 위한 첫번째 단계에서는 2D 영상처리에서 사용되는 블러링(Bluering) 기법을 Depth Image의 x, y좌표값에 적용하여 주로 물체의 경계부분에서의 발생하는 불안정된 영역을 감소시켰고 2단계와 3단계에서는 시간의 흐름에 따른 픽셀(pixel)의 변화량을 사용한 필터를 적용하였으며 변화 여부를 결정하는 Threshold값을 변수화하여 피사체와 실험환경에 따라 최적화할 수 있도록 하였다.

2. 시간차 프레임의 픽셀 비교와 실험

수신된 대상 프레임의 모든 픽셀(Pixel)과 대상 프레임 직전에 수신된 프레임의 모든 픽셀을 각각 비교한 후 비교값이 ThresHold를 초과하는 경우에만 움직임이 있다고 판단하였다.

$$Qi(x,y) = | Q(i-1)(x,y) - Qi(x,y) | > ThresHold \quad (x \in X, y \in Y) \tag{1}$$

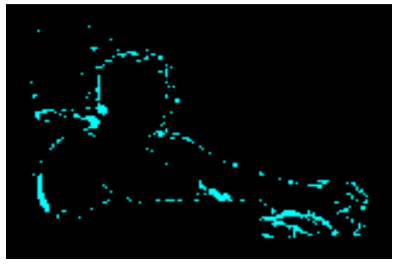


그림 3. 현재프레임과 이전프레임간의 차이영상
Fig. 3. Difference between the before frame and after frame

3. 주파수필터

영상에서 발생하는 노이즈와는 달리 물리적으로 실제 움직이는 물체의 경우 Depth정보가 빠르게 반복적으로 변화될 수 없는 Depth Image특성을 사용하여 시간 주파수 필터링을 적용하였다.

$$C1 = \sum_{I=QueueSize}^{0} | Gi(x,y) - G(i+1)(x,y) | > ThresHold \tag{2}$$

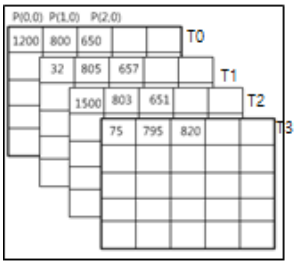


그림 4. Depth Frame의 예
Fig. 4. Examples of the Depth Frame

그림 4 에서 Threshold를 100으로 가정하고 변화카운트가 2개 이하만을 유효하다고 할 때 p(0, 0)은 변화카운트가 3개이므로 노이즈로 판단하며 P(1,0)은 '가.2단계필터' 에서 제거되므로 변화되지 않은 Pixel로 인식된다. P(2,0)은 최근 프레임과 이전 프레임과의 변화가 있으며 변화카운트가 1이므로 실제 유효한 Pixel로 인식되게 된다.

아래의 모든 실험은 30Frame을 기준으로 수행 되었다.

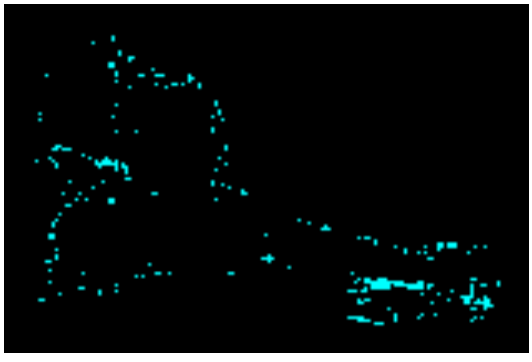


그림 5. 변화 카운트가 25 이하인 픽셀
Fig. 5. Changed count <= 25

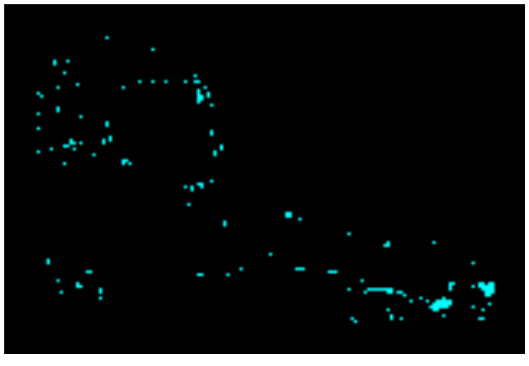


그림 6. 변화 카운트가 20 이하인 픽셀
Fig. 6. Changed count <= 20

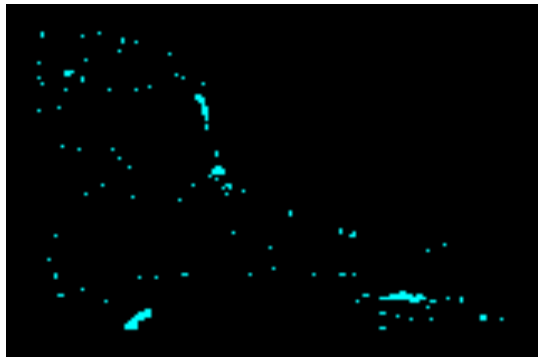


그림 7. 변화 카운트가 15 이하인 픽셀
Fig. 7. Changed count <= 15

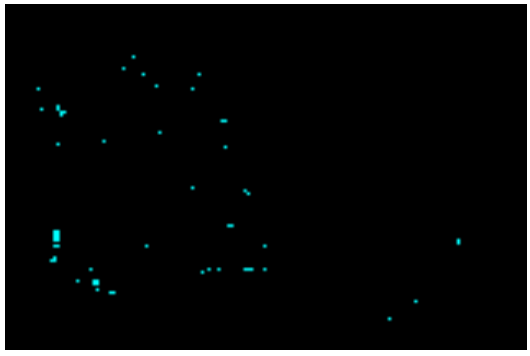


그림 8. 변화 카운트가 10 이하인 픽셀
Fig. 8. Changed count <= 10

위 그림8 에서 보는 것과 같이 주파수 필터를 적용하여 동일한 픽셀에서 지정횟수만큼 변화되는 경우 Noise로 인식하여 제거한 경우 물체와 배경 및 물체와 물체간의 경계에서 발생하는 noise가 감소되는 것을 알 수 있었다.

Ⅲ. 움직인 영역을 획득하기 위한 군집 필터링

기본 필터링을 사용해서 Depth frame에 존재하는 물체와 물체간의 노이즈와 배경과 물체 경계간에 발생하는 noise를 감소하였다. 물체가 움직인다면 주파수 필터에 의해서 물체의 움직인 영역과 그렇지 않은 영역이 구분되어 질것으로 예상 되었으며 실험은 아래의 [그림 9]와 같다.

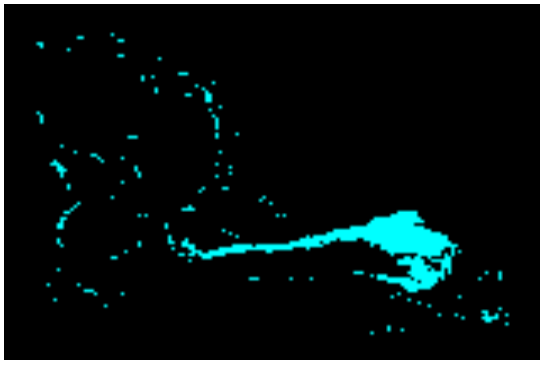


그림 9. 빠른 손 움직임에 대한 영상
Fig. 9. Image for quick hand movements

그림 9 는 손을 빠르게 움직였을 때의 화면으로 실제 움직인 부분의 경우 움직이지 않은 부분에 비하여 픽셀이 군집된 것을 확인할 수 있다. 군집되지 않은 픽셀이나 실제 움직임으로 나타낼 수 없는 작은 단위로 군집된 부분을 제거하는 방식을 사용하여 움직인 부분만을 찾아내었다. 군집여부의 판별은 대상 픽셀을 기준으로 주변의 픽셀이 시간에 따른 변화가 있는지를 판단하여 같은 군집에 포함시킬지를 결정하였다.

비교대상 픽셀의 선정은 대상픽셀을 중심으로 360도를 검사하고자 하는 각도 나눈 후 대상픽셀로부터 지정된 거리(distance)만큼 떨어진 것 까지를 비교대상으로 하였다.

```
public Point GetPositionByAngle(int angle, int distance)
{
    double newRadian = Math.PI * (double)angle / 180.0;

    int xValue = (int)Math.Round(Math.Sin(newRadian) * distance);
    int yValue = (int)Math.Round(Math.Cos(newRadian) * distance) * -1;
    Point rstValue = new Point(xValue, yValue);
    return rstValue;
}
```

```
private void MageAnglePositions(int intervalAngle, int distanceValue, int screenWidth)
{
    int intervalDegree = (360 / intervalAngle);

    anglePos = new int[intervalAngle][];

    for (int i = 0; i < intervalAngle; i++)
    {
        //체크하는 길이만큼 배열을 생성한다.
        anglePos[i] = new int[distanceValue];
        int targetAngle = intervalDegree * i;

        for (int j = 0; j < distanceValue; j++)
        {
            Point pnt = GetPositionByAngle(targetAngle, j + 1);

            int bufferPosition = (int)(pnt.Y * screenWidth + pnt.X);
```

위 소스코드는 처리 속도 향상을 위해 대상 픽셀을 기준으로 군집화 대상을 미리 계산해 놓은 후 실제 연산시 대상 픽셀의 x값과 y값을 기준으로 군집화 대상의 픽셀들과 비교한 것이다.

[그림 10]은 군집필터의 예로써 A부터 I까지 표시된 부분은 변화량이 기준값 이상으로 판단된 픽셀들이며 그 외 픽셀들은 변화량이 기준 이하로 판단되었다고 가정한 경우이다.

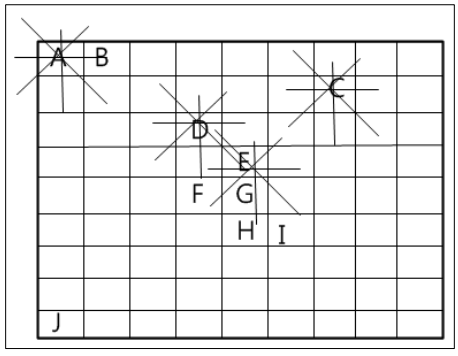


그림 10. 군집필터의 예
Fig. 10. Examples of Cluster filter

8개의 방향에 기준거리(distance)를 1, 인접픽셀을 2이상으로 한 조건에서 군집 검사를 할 때 A=1, D=1 E=2, F=2, G=3의 인접 픽셀을 갖는다. 빠른 속도를 위해서 픽셀 하나의 검사가 완료된 후 군집조건에 해당되지 않는 경우 군집군에서 탈락시켰으며 그로 인해 E픽셀의 경우 인접 픽셀 검사 시 탈락된 D로 인하여 2를 갖게 된다.

$$C = \sum_{n=1}^{Angle} \sum_{d=1}^{Dist} S(\sin(360/angles * n) * d, \cos(360/angles * n) * d * -1) \quad (3)$$

비교대상 픽셀의 선정은 대상픽셀을 중심으로 360도를 검사하고자 하는 각도 나눈 후 대상픽셀로부터 지정된 거리(distance)만큼 떨어진 것 까지를 비교대상으로 하였다.

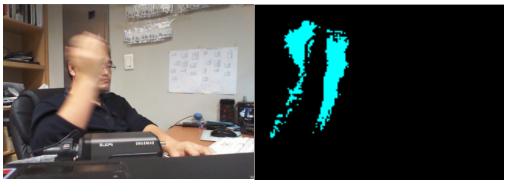
Ⅳ. 실험 및 결과

실험은 Microsft Kinect 센서에서 30Frame에서 Depth frame을 수신받은 후 본 논문에서 제시한 기본 필터링과 군집필터를 사용하여 움직인 영역을 감지하였다.

주파수 필터 사용시 30 Frame에서 초당 10 픽셀을 기

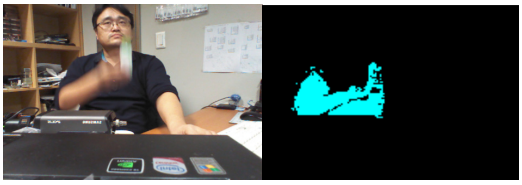
준으로 하여 유효 픽셀을 구분하였다. 또한 군집 필터는 본 논문에서 제시한 방법을 사용 하였으며 대상 픽셀을 기준으로 8개의 방향과 5개의 인접 픽셀 여부를 근거로 필터링 하였다.

아래는 본 논문에서 제시하는 Depth frame에서 주파수 필터와 군집필터를 사용하여 실제 움직인 부분만을 화면에 표시한 실험 결과이다.



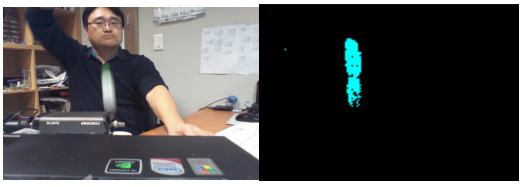
(a) color frame (b) detected area

그림 11. 손을 움직였을때의 결과
Fig. 11. Result of the hand moved



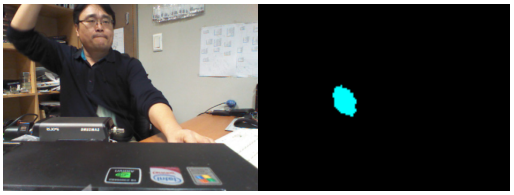
(a) color frame (b) detected area

그림 12. 음료수 병을 위로 던질때의 결과
Fig. 12. Result of the Throw up a bottle



(a) color frame (b) detected area

그림 13. 음료수병 낙하
Fig. 13. Result for the bottle falling



(상) color frame (하) detected area

그림 14. 지폐 낙하
Fig. 14. Result for paper fall

V. 결 론

Kinect의 Depth 이미지를 이용하여 물체간의 경계와 움직임의 판단을 쉽게할 수 있었지만 물체간의 경계등에서 지속적으로 발생하는 z좌표의 변화는 움직이는 부분만을 감지하기 어렵게 하였다. 그로인해 본 논문에서는 블러링화, 주파수필터, 클러스터링 필터를 적용하여 움직이지 않는 부분에서 발생하는 노이즈를 제거하였다. 노이즈가 제거된 영상은 실제 움직이는 부분만을 감지할 수 있었으며 비교적 작은 물체의 감지도 가능하였다. 주로 키넥트를 응용하는 경우 Microsoft Kinect SDK에서 기본 제공되는 Skeleton Stream Data를 이용하여 손쉽게 인체부위의 움직임을 사용할 수 있다. 하지만 감지 대상 물체가 인체가 아니거나 키넥트에 인체의 상반신 이상이 노출되지 못하는 경우 Skeleton Stream이 인식되지 않아 움직이는 영역을 직접 추출해야 되는 경우가 있으며 움직이는 로봇 등에서 배경에 비해 빠르게 움직이는 영역을 감지할 필요가 있을 때 위의 방법들이 적용될 수 있다.

References

[1] Lim, Jung-Geun and Han, Kyongho, "Remote image control by hard motion detection" Journal of IKEEE, Vol 16, No.4, 369~374, December 2012

[2] H. Jeong, J.J. You,K.H. Jang,H.S. Kim,H.W. Jang,J.Y. Park,J.H. Lee, S.W. Nam, "A Technical Trend of Stereoscopic Content Production and User Interaction" Electronics and Telecommunications Trends Vol 27, No 3, 2012

[3] Kwanghyun Ro and Seokkee Lee, "A Research on Context-aware Digital Signage using a Kinect" The Journal of IIBC. Vol. 14, No. 1, pp.265-273, Feb. 28, 2014

[4] Chang-Ho Han and Choon-Suk Oh, "Implementation of a 3D Recognition applying Depth map and HMM" The Journal of IIBC. Vol. 12, No. 2, pp119-126, April 2012

[5] http://en.wikipedia.org/wiki/Time-of-flight_camera

[6] http://en.wikipedia.org/wiki/Stereoscopic_Depth_Rendition

저자 소개

이 성 호(준회원)

- 2012년 : 단국대학교 정보통신(석사)
- 2012년 ~ 현재 : 단국대학교 전자전기 공학부 박사과정

<주관심분야 : Micro Processor Application, security about network and software, Network communication, Mass flow meter>

한 경 호(정회원)

- 1984년 : 서울대학교 대학원 전자공학과 석사
- 1992년 : Texas A&M University, Electrical Engineering Ph.D
- 1992년 ~ 1993년 : 한국전자통신연구원 이동통신연구단 선임연구원
- 1989년 ~ 1992년 : Unix & System Admin, Texas A&M Univ
- 1985년 ~ 1987년 : 한국통신품질보증단전임연구원
- 1984년 ~ 1985년 : 삼성전자HP 연구원
- 2007년 ~ 2008년 : 단국대학교 산업기술연구소장
- 2008년 ~ 2010년 : 단국대학교 정보통신연구원장
- 2014년 ~ 현재 : 단국대학교 전자전기 공학부 교수

<주관심분야 : Micro processor Application, ITS, F/A System, Network communication.>