안드로이드 기반 로고를 이용한 증강현실 시스템[1)]

임선진*, 정은영*, 정운국*, 정경민*, 문창배*, 김병만*, 이종열**
*금오공과대학교 소프트웨어공학과
**(주)비전앤바이오테크
e-mail:moonyeses@naver.com

Augmented Reality Logo System Based on Android platform

Sun-Jin Lim*, Eun Young Jung*, Un Kuk Jeong*, Kyoung Min Jung*, Chang Bae Moon*, Byeong Man Kim*, Jong-Yeol Yi**
*Dept. of Computer and Software Engineering. Kumoh National Institute of Technology
**Vision&BioTech Corp.

요 약

스마트 폰의 등장과 모바일 인터넷을 제공함에 따라 휴대폰은 음성통신 수단이 아닌 웹을 통하여 서비스를 제공받는 도구 또는 각종 게임 및 응용 어플리케이션을 제공하는 놀이수단으로도 발전하였고, 이로 인하여 사용량도 증가하였다. 사용량의 급증으로 인하여 모바일 광고에 대한 업계의 관심도 증가 하였지만, 한정적인 출력화면에 의하여 제한적일 수밖에 없다. 이를 보완하기 위해, 본 논문에서는 기업의 로고 광고의 효과를 극대화 할 수 있는 안드로이드 기반 로고를 인식하는 증강현실 시스템을 제안 하였고, 이를 구현 하여 실 제폰에 탑재한 후 다양한 성능 분석을 하였다. 실험결과, 그 가능성은 확인하였지만 현 하드웨어 성능으로는 실시간으로 지원하기에는 역부족임을 알 수 있었다.

1. 서론

기존 휴대폰은 주로 음성 통신수단으로 이용되어 왔다. 하지만 스마트 폰의 등장과 모바일 인터넷을 제공함에 따라 휴대폰은 더 이상의 음성통신 수단이 아닌 웹을 통하여 서비스를 제공받는 도구로 발전하였다. 또한 스마트폰을 이용하여 각종 게임 및 응용 어플리케이션을 사용함으로 사용자들의 놀이수단으로도 발전하였다. 이와 같은 사용량의 급증으로 인하여 모바일 광고에 대한 업계의 관심도 증가 하였지만, 한정적인 출력화면에 의하여 제한적일 수밖에 없다.

기업들은 기업의 이미지를 중요시 하고, 이미지 개선을 위하여 기업 로고 디자인에 투자를 하기도 하고, 방송 매체를 통하여 기업의 로고를 광고하기도 한다. 하지만 광고 매체는 시간적인 제약사항으로 한정적인 시간만을 광고할 수 있고, 고비용이 발생 한다. 또한 외국에서 광고를 할 때 후진국의 방송매체를 이용하는 경우 저 비용이 발생하지만, 선진국의 방송매체를 이용하는 경우 고 비용이 발생한다.

일반적으로 기업의 로고를 광고하는 경우 시청자의 지루함을 발생할 수 있고, 지루함으로 오히려 불쾌감을 발생시켜 기업의 이미지가 손상될 수도 있다. 그리고 게임을 통하여 기업의 로고를 광고할 경우 한정적인 화면을 사용하기 때문에 기업의 로고를 사용자가 인지하지 못하고 게임을 진행 할 수 있다. 증강현실을 이용하면 이러한 문제점을 해결할 수 있다. 즉 증강현실에서 사용하는 마커를 로고로 사용하면 사용자는 실질적으로 로고를 접할 수 있기 때문에 기업 로고의 광고 효과 또한 증가하게 될 것이다.

본 논문에서는 기업의 로고를 인식하고, 인식한 로고를 이용하여 증강현실 게임을 제공하여 사용자에게 간접적인 광고효과를 보일 수 있는 안드로이드 기반 로고인식 증강현실 시스템을 제안 한다.

2. 관련연구

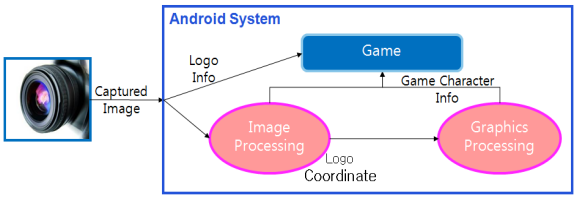
영상 내에서 객체를 검색하기 위한 알고리즘으로 템플릿 매칭[1]이 존재한다. 템플릿 매칭 알고리즘은 템플릿 이미지를 이용하는 방식으로 대상 이미지 상에 가장 유사한 구간을 검색하는 방법이다. 대상 이미지 전체를 탐색하면서 템플릿 이미지와 각각 얼마만큼 유사성이 있는지를 구하는 방법으로 유클리디안(Euclidean)법을 이용한다.

객체를 추적하기 위한 알고리즘으로 Color Segment를 이용한 MeanShift 알고리즘이 존재하고, MeanShift 알고리즘은 조명이나 주의 색상에 민감하게 반응하며 추적율이 낮고, MeanShift 알고리즘의 성능을 보완한 알고리즘으로 CAMShift[1, 2] 알고리즘이 존재한다. CAMShift 알고리즘은 검출된 객체의 영역의 Hue값의 분포를 이용하여 변화된 위치를 예측하고 탐지 후 객체의 중심을 찾아 객체를 실시간으로 추적하는 알고리즘이다. CAMShift 알고리즘은 Hue 값을 이용하여 조명의 영향을 줄여주는 방법이다.

1) 본 논문은 2010년도 교육역량강화사업의 KIT 가족연구실 지원 사업으로 연구 되었습니다.

3. 안드로이드 기반 로고인식 증강현실 시스템

본 논문의 시스템 구조도는 (그림 1)과 같이 전체적으로 Android System 기반으로 이미지 처리 모듈, 그래픽 처리 모듈, 게임 모듈로 구성되며, 각 모듈은 스레드를 이용하여 독립적 동작하도록 구축되었다. 최초 카메라에서 영상을 입력받아 이미지 처리모듈과 게임모듈로 영상을 전달한다. 이미지 처리모듈에서는 전달받은 영상을 이용하여 로고 인식 및 추적하고, 게임모듈과 그래픽 처리모듈로 로고 정보를 전달한다. 게임모듈에서는 로고의 기본 정보를 영상처리 모듈에서 입력받아 로고의 기본 정보를 제공하고, 그래픽 처리 모듈에서는 로고의 좌표를 입력받아 게임 캐릭터 생성 및 영상을 합성한다.



(그림 1) 시스템 구조도

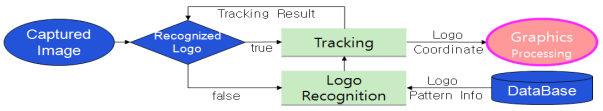
3.1 안드로이드 시스템

본 논문에서 사용한 Application Framework에서는 Activity Manager, View System, Resource Manager 등을 이용하여 액티비티의 생명 주기를 제어하고, 액티비티의 사용자 인터페이스를 구성하고, DB파일과 로고 이미지, 캐릭터등을 제공하였다. 그리고 Open GL/ES, SQLite[3], Webkit[4], Media Framework[4]등의 라이브러리를 사용하였는데 이들은 카메라에서 입력 받은 영상과 생성시킨 캐릭터를 합성, DB를 구축, 웹을 통한 기업의 사이트 제공, 게임의 사운드를 제공하였다. 또한 안드로이드 전용 라이브러리인 Core Libraries와 Dalvik Virtual Machine을 이용하여 하드웨어 사용 및 사용자 인터페이스와 다양한 리소스 관리를 위한 클래스를 제공받는다.

3.2 이미지 처리

카메라 영상을 캡쳐하는 과정은 카메라 장치로부터 입력된 원시 데이터를 캡쳐 스레드를 통하여 이미지 처리모듈에서 사용할 이미지를 생성시키고, 생성한 이미지를 이미지 처리 모듈로 전송 한다.

이미지 처리모듈에서는 영상이 입력되면 (그림 2)과 같이 데이터베이스의 로고 정보를 바탕으로 로고를 인식하고 이를 이용하여 로고를 추적하게 된다. 만약 추적과정 중에 객체 추적에 실패하면 로고 인식과정을 통하여 로고를 다시 찾게 된다. 이미지 처리과정이 완료되면 그래픽 처리를 위하여 로고가 위치한 좌표를 최종적으로 그래픽 처리 과정으로 전달한다.

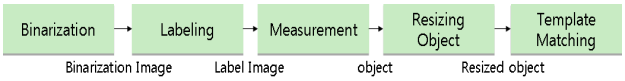


(그림 2) Image Processing 내부 구성

3.2.1 로고 인식

본 논문에서는 로고 인식을 위하여 템플릿매칭을 이용하였다. 하지만 템플릿매칭만 사용하는 경우 템플릿 로고의 사이즈와 대상 로고의 사이즈가 동일하지 않은 경우 로고 인식성능이 크게 저하되기 때문에 로고를 인식하기 위해 (그림 6)과 같은 전처리 과정을 수행하였다. 먼저 이진화를 통하여 이진화된 이미지를 획득한 후 레이블링을 통하여 레이블된 객체를 획득하고, 이를 확대하여 템플릿 이미지와의 유사도를 계산하였다.

대부분 기업의 로고는 단색을 많이 사용하기 때문에 본 논문에서는 Hue값을 이용하여 이진화 하였고, 로고는 단순한 모형을 유지 하지만 서로 다른 사이즈를 사용하기 때문에 레이블된 객체의 가로:세로 비율과 데이터베이스에 저장되어 있는 로고들의 가로:세로 비와 비교하여 1차적으로 템플릿 매칭을 위한 대상 이미지 즉, 로고 이미지를 획득한다. 레이블링으로 받아온 객체를 데이터베이스에서 받아온 이미지와 동일한 사이즈로 resizing 후 2차적으로 템플릿 매칭의 유클리디안(Euclidean)법을 사용하여 객체를 판별하였다.



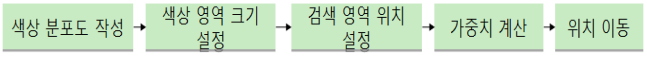
(그림 3) 로고인식의 전처리 과정 및 Template Matching

템플릿 매칭을 이용하여 획득한 예는 (그림 4)와 같다. (그림 4a)는 전체 영상, (그림 4b)는 템플릿 이미지, (그림 4c)는 최종적으로 획득한 결과 영상의 예를 보여준다. 본 논문 실험에서는 로고에 모자이크 처리를 하지 않은 상태에서 실험을 하였고, 논문 기술시 수작업으로 로고에 모자이크 처리를 하였다.

(a) 전체 영상 (b) 템플릿 이미지 (c)결과영상
(그림 4) 템플릿매칭의 예

3.2.2 로고 추적

템플릿 매칭으로 획득한 영상을 추적하기 위한 알고리즘으로 본 논문에서는 CAMShift 알고리즘을 사용하였고, CAMShift 알고리즘은 검출된 객체의 영역의 Hue값의 분포를 이용하여 변화된 위치를 예측하고 탐지 후 객체의 중심을 찾아 객체를 실시간으로 추적하는 알고리즘으로 (그림 5)와 같다.



(그림 5) Camshift 순서도

$$M_{00}=\sum_x\sum_y I(x,y) \quad (1) \qquad M_{01}=\sum_x\sum_y xI(x,y) \quad (2)$$

$$M_{10}=\sum_x\sum_y yI(x,y) \quad (3) \qquad (x_c,y_c)=\left(\frac{M_{10}}{M_{00}},\frac{M_{01}}{M_{00}}\right) \quad (4)$$

$$M_{20}=\sum_x\sum_y x^2I(x,y) \quad (5) \qquad M_{02}=\sum_x\sum_y y^2I(x,y) \quad (6)$$

$$M_{11}=\sum_x\sum_y xyI(x,y) \quad (7)$$

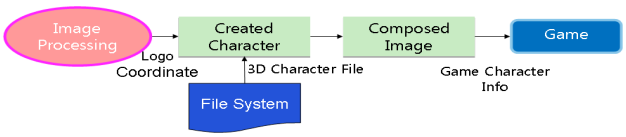
$$a=\frac{M_{20}}{M_{00}}-x_c^2,\ b=2\left(\frac{M_{11}}{M_{00}}-x_cy_c\right),\ c=\frac{M_{02}}{M_{00}}-y_c^2 \quad (8)$$

$$l=\frac{\sqrt{(a+c)+\sqrt{b^2+(a-c)^2}}}{2},\ W=\frac{\sqrt{(a+c)-\sqrt{b^2+(a-c)^2}}}{2} \quad (9)$$

CAMShift 알고리즘은 우선 (그림 5)와 같은 처리 과정을 거치고, 크게 두 가지로 나누어 볼 수 있는데 n-1 시간의 프레임 처리와 n 시간에서의 프레임 처리로 나누어 볼 수 있다. n-1 시간에서는 식 (1)부터 (3)까지의 처리가 이루어지고, n 시간에서의 처리과정은 식 (5)부터 식 (9)까지 처리가 된다. 식 (1)에서는 객체가 포함된 윈도우의 moment를 계산하고, 식 (2), (3)에서는 x축과 y축에 대한 무게 중심을 구하기 위한 각 moment를 구한 후 식 (4)을 이용하여 무게중심을 계산한다. 이때 n-1 시간의 무게중심과 n 시간의 무게 중심의 차이가 발생하는 경우, 임계점보다 크다면 n-1 시간에서의 무게 중심과 n 시간의 중심 사이의 거리가 기본 임계점보다 작아질 때까지 검색 영역의 무게 중심을 n-1 시간의 객체 위치에서 n 시간의 객체 위치로 재조정한다. n시간에서 객체의 사이즈를 획득하기 위해 식 (5), (6), (7), (8), (9)를 이용하는데, 식 (8)을 이용하여 객체의 높이와 폭을 측정한 후 객체의 높이와 폭을 식 (9)를 통하여 획득한다[1, 2, 5].

3.4 그래픽 처리

그래픽 처리모듈은 (그림 6)에서 보는 것과 같이 캐릭터를 생성하는 단계와 생성한 캐릭터를 합성하는 단계로 구성된다. 캐릭터 생성 시 파일 시스템에서 캐릭터의 정보를 가진 3D 캐릭터 파일을 획득하여 캐릭터를 형상화 시켜주고, 이미지 처리에서 획득한 좌표를 이용하여 투명 뷰에 캐릭터를 합성한다. 합성한 뷰를 게임모듈로 전송하여 사용자에게 게임을 제공한다.



(그림 6) Graphics Processing

캐릭터 생성 시 벡터로 구성된 정점 3개를 화면에 표시 후 3개의 정점들을 서로 연결시키고, 3개의 정점을 연결시켜 (그림 7a)와 같이 다각형 면을 구성시킨 후 다각형 면에 스킨(texture)을 입힌다(그림 7b). 이때 스킨(texture)은 파일시스템에서 받아온다. 또한 캐릭터가 움직일 때는 정점들의 위치가 변하는데 변한 vertex들의 정보를 추출하고, 추출한 정보를 이용하여 다각형 면을 재정의 한다. 재정의 된 다각형 면의 표면에 선형 보간법을 이용하여 스킨을 재 적용한다(그림 7c).

(a) (b) (c)

(그림 7) Character 생성 및 움직임 처리의 예((a) Mesh Character, (b) Character 생성, (c) 객체 움직임 적용)

캐릭터와 이미지의 합성은 먼저 투명화 영역을 설정하여 SurfaceView 캐릭터를 생성한다(그림 8a). 투명화 영역은 (그림 8a)의 핑크색 영역이다. (그림 8a)와 같이 투명화 영역 설정 후 (그림 8b)와 같이 카메라로 부터 받아온 획득한 영상과 SurfaceView를 합성시켜 (그림 8c)와 같은 영상을 획득한다.

(a) (b) (c)

(그림 8) 그래픽 적용의 예 ((a) Created Character, (b) Capture Image, (c) 적용 결과)

3.5 Augmented Reality Game

본 논문에서 구성시킨 게임화면은 (그림 9)과 같다. (그림 9 a)는 게임의 Main UI, (그림 9 b)는 게임실행화면, (그림 9 c)는 로고가 알려주는 해당 기업정보, (그림 9 d)는 (그림 9 c)에서 선택한 링크의 사이트이다. SQLite[3]를 이용하여 DataBase를 구축하였고, Webkit[4]를 이용하여 해당 사이트의 정보를 제공할 수 있도록 하였다. 메뉴화면에서 입력받은 영상의 로고를 판별하고, 특정 로고가 있으면 로고에 해당하는 기업의 이름을 하단에 표시한다. 그 후 DB에 저장된 로고 정보를 볼 수도 있고 해당 정보를 클릭하여 웹 검색도 가능하다. 게임하기로 넘어가면 화면에 별이 나타나고 로고위에 나타난 3D 캐릭터를 움직여 별과 겹칠 경우 별을 지울 수 있다. 게임은 제한시간 안에 캐릭터가 별을 모두 제거하면 종료 된다.

(a) (b) (c) (d)

(그림 9) 게임 실행화면 및 로고정보 제공화면((a) MainUI, (b) 게임실행화면, (c) 기업로고정보, (d) 기업정보)

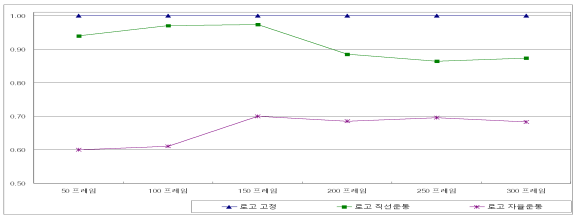
4. 실험 및 성능 평가

본 논문의 개발환경은 Android OS 버전 2.1을 사용하여 개발하였고, 최종 데모는 Samsung Galaxy A(CPU 처리속도 : 700MHz)에서 테스트하였다. 본 논문에서는 세 가지 실험을 실시하였는데 첫 번째 실험은 안드로이드 반응속도에 대한 실험이고, 두 번째 실험은 영상처리에 관련된 실험, 마지막 실험은 영상 및 그래픽처리 속도에 대한 실험을 실시하였다. 또한 본 논문에서 사용한 그래픽 정보는 Vertices 358, Skins 13, TexCoords 469, Triangles 670, Frames 20개로 구성하였고, 카메라에서 받아온 영상 사이즈는 320×508사이즈이다.

4.1 영상처리 성능평가

본 논문에서는 추적 알고리즘의 성능을 평가하기위해 50프레임 단위로 증가시키면서 인식율을 측정한 결과 (그림 10)와 같다. 인식율을 계산하기 위한 첫 번째 실험은 로고의 움직임이 발생하지 않았을 때의 인식율을 계산하였고, 두 번째 실험은 로고가 직선 운동할 때의 인식율을 계산하였으며, 마지막 실험은 로고의 무작위 자율운동이 발생하였을 때의 인식율을 계산하였다.

실험 결과, 로고의 움직임을 발생시키지 않았을 때 모든 프레임에서 로고를 인식함으로 100% 인식율을 보였고, 로고의 직선운동 시 92 %의 인식률을 보였다. 또한 로고의 자율운동 시 66 %의 인식율을 보였다.



(그림 10) 안드로이드 기반 추적률 및 인식율

4.2 영상 및 그래픽 처리속도 평가

안드로이드 기반에서 영상 처리 속도는 두 가지로 측정하였다. 첫 번째는 게임 실행 시 영상처리 속도와 게임 비실행 시 영상처리를 측정하였고, 두 번째는 응용프로그램 전체 처리 속도를 측정하였다.

첫 번째 측정에서 획득한 결과는 게임 실행 시 영상처리 평균처리 속도는 4.45초, 게임 비실행 시 형상처리 평균 속도는 2.66초로 측정되었다. 두 번째 측정한 결과는 그래픽처리를 포함한 전체 평균 처리속도는 7.23초로, 그래픽처리를 비 포함한 전체 평균 처리속도는 3.67초로 측정되었다.

알고리즘의 처리속도가 오래 걸리는 단점이 있지만 본 논문에서 사용한 영상의 사이즈는 320×508을 사용하였고, 그래픽의 경우 3차원 그래픽을 사용하였기 때문이라 할 수 있다. 또한 700MHz CPU를 사용하였기 때문에 만약 하드웨어 성능이 개선된다면 본 논문에서 제시한 알고리즘의 처리속도는 더 빨라질 것이다.

5. 결론

본 논문에서는 안드로이드 폰 기반 로고인식 증강현실 게임을 제안하였고, 어플리케이션 개발에 사용된 기술과 알고리즘들을 소개하였다. 로고를 인식하기 위해 템플릿 매칭 알고리즘을 사용하였고, 인식한 로고를 추적하기 위해 CAMShift 알고리즘을 사용하였다. 본 논문에서 제시한 시스템의 성능을 평가한 결과 영상처리속도가 현저하게 낮은데 이는 저 사양인 700Mhz의 안드로이드 폰에서 실시간 영상처리 알고리즘을 탑재 하였을 때 발생한 결과이다. 만약 안드로이드 폰의 성능이 더 좋아 진다면 실시간 영상처리 속도는 자연적으로 개선될 것이다. 또한 본 논문에서 사용한 이미지 사이즈는 320×508을 사용하였기 때문에 영상의 사이즈를 적절히 조절하거나 그래픽을 3차원 그래픽이 아닌 2차원 그래픽을 제공한다면 더 좋은 성능도 기대할 수 있을 것이다.

참고문헌

[1] Intel Inc., “Open Source Computer Vision Library,”http://developer.intel.com, 1999-2001

[2] Join G. Allen, Richard Y.D.Xu, Jesse S. Jin, "Object Tracking Using CamShift Algorithm and Multiple Quantized Feature Spaces", Proceedings of the Pan-Sydney area and workshop on visual information processing, pp.3-7(4pages), 2004

[3] Reto Meier, "Professional Andorid Application Development", 제이펍출판사, pp.201-210, 2009

[4] Mark Murphy, "알짜만 골라 배우는 안드로이드 프로그래밍", 에이콘출판사, pp.243-261, 2009

[5] Xiao Gang, Chen Yong, Chen Jiu-jun, Gao fei, "Automatic Camshift tracking algorithm based on fuzzy inference background difference combining with twice searching", E-Health Networking, Digital Ecosystems and Technologies(EDT), 2010 International conference, pp.1-4(4pages), 2010