http://dx.doi.org/10.3745/KIPSTB.2011.18B.4.181

안드로이드 기반 로고를 이용한 증강현실 시스템

정 은 영[†] · 정 운 국[††] · 임 선 진[†††] · 문 창 배[††††] · 김 병 만[†††††]

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 놀이 수단으로도 발전하였고, 이로 인하여 사용량도 증가하였다. 사용량의 급증으로 인하여 모바일 광고에 대한 업계의 관심도 증가 하였지만, 한정적인 출력 화면에 의하여 광고 효과가 제한적일 수밖에 없다. 이를 보완하기 위해, 본 논문에서는 기업의 로고 광고의 효과를 극대화 할 수 있는 안드로이드 기반 로고 인식 증강 현실 시스템을 제안하였고, 이를 구현하여 실 제 스마트 폰에 탑재한 후 다양한 성능 분석을 하였다. 실험결과, 그 가능성은 확인하였지만 현 하드웨어 성능상 실시간으로 지원하기에는 역부족임을 알 수 있었다.

키워드 : 증강현실, 로고인식, 안드로이드, 영상처리

Augmented Reality Logo System Based on Android platform

Eun Young Jung[†] · Un Kuk Jeong[††] · sun-jin Lim[†††] · Chang Bae Moon[††††] · Byeong Man Kim[†††††]

ABSTRACT

A mobile phone is becoming no longer a voice communication tool due to smartphones and mobile internet. Also, it now becomes a total entertainment device on which we can play game and get services by variety applications through the Web. As smartphones are getting more popular, their usages are also increased, which makes the interest of advertising industry in mobile advertisement increased but it is bound to be limited by the size of the screen. In this paper, we suggest an augmented reality logo system based on Android platform to maximize the effect of logo advertisement. After developing software and mounting it on a real smartphone, its performances are analyzed in various ways. The results show the possibility of its application to real world but it's not enough to provide real time service because of the low performance of hardware.

Keywords : Augmented Reality, Recognition of Logo, Android, Image Processing

1. 서 론

기존 휴대폰은 주로 음성 통신 수단으로 이용되어 왔다. 하지만 스마트 폰의 등장과 모바일 인터넷을 제공함에 따라 휴대폰은 더 이상의 음성 통신 수단이 아닌 웹을 통하여 서비스를 제공받는 도구로 발전하였다. 또한 스마트 폰을 이용하여 각종 게임 및 응용 어플리케이션을 사용함으로 사용자들의 놀이 수단으로도 발전하였다. 이와 같은 사용량의 급증으로 인하여 모바일 광고에 대한 업계의 관심도 증가하였지만, 한정적인 출력 화면에 의하여 그 활용도가 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, 광고 매체를 이용하는 것은 고비용이 발생한다. 또한 외국에서 광고를 할 때 후진국의 방송 매체를 이용하는 경우 저 비용이 발생하지만 선진국의 방송 매체를 이용하는 경우 고 비용이 발생한다.

일반적으로 기업 로고를 광고하는 경우 시청자의 지루함이 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 진행할 수 있다. 증강현실을 이용하면 이러한 문제점을 해결할 수 있다. 즉 증강현실에서 사용하는 마커를 로고로 사용하면 사용자는 실질적으로 로고를 접할 수 있기 때문에 기업

※ 본 연구는 금오공과대학교 학술연구비에 의하여 연구되었음.
† 정 회 원 : 웹케시(주) 사원
†† 정 회 원 : (주)한국무역정보통신 사원
††† 정 회 원 : (주)오픈앤와이즈 사원
†††† 준 회 원 : 금오공과대학교 컴퓨터공학부 소프트웨어공학과 박사과정
††††† 준 회 원 : 국립금오공과대학교 컴퓨터공학부 교수(교신저자)
논문접수 : 2010년 10월 27일
수 정 일 : 1차 2011년 2월 22일, 2차 2011년 3월 17일
심사완료 : 2011년 3월 17일

로고의 광고 효과 또한 증가하게 될 것이다.

대기업의 로고는 해당 국민이면 모르는 사람이 없을 것이다. 하지만 외국인의 경우 해당국 대기업의 로고를 모르는 사람들도 존재할 수 있다. 아무리 기업 제품의 기능 및 서비스가 좋다고 하더라도 해당 제품을 타국 제품이라 인지한다면 국가적 손실이 발생할 것이다. 예를 들어, 삼성이나 엘지에서 제작된 모바일 폰의 경우 중국에 포함된 기업의 제품이라 인지하는 경우 해당 기업은 대한민국의 회사가 아닌 중국의 회사로 인지하여 중국의 제품 경쟁력만 높일 수 있다. 이로 인해 국가적 손실이 발생할 수 있다. 하지만 로고를 증강현실 게임으로 제공하고, 게임 중 사용자의 요청으로 해당 로고가 의미하는 기업 사이트를 제공하면 해당 기업이 어느 국가에 포함된 기업인지 사용자는 알 수 있을 것이다. 또한 자연적으로 기업의 사이트를 통해 광고할 수 있고, 국가적 경쟁력 또한 향상될 것이다.

로고의 광고는 기업의 광고에 있어 중요한 요소라 할 수 있다. 본 논문에서는 기업의 로고를 인식하고, 인식한 로고를 이용하여 증강현실 게임을 제공하여 사용자에게 간접적인 광고 효과를 보일 수 있는 안드로이드 기반 로고 인식 증강현실 시스템을 제안한다. 본 논문의 구성은 다음과 같다. 2장에서는 선행 연구, 3장에서는 본 논문의 핵심인 로고 인식 및 트랙킹을 위해 사용한 영상처리 기술, 증강 현실과 3D 처리를 사용하기위해 적용한 그래픽 기술을 설명하고, 4장에서는 실제 구현 결과를 토대로 데모를 수행한 실험 결과, 마지막으로 5장에서 본 논문을 결론짓는다.

2. 관련 연구

증강현실(Augmented Reality : AR)이란 인간과 컴퓨터 간의 상호 작용(Human Computer Interface : HCI)을 이용하여 인지 능력을 증강하는 기술로써 주로 시각이 사용되고, 시각을 통하여 느낌을 증강시키는 것이다[1, 2]. 즉, 증강현실은 실제 환경에 가상의 환경을 접목한 기술이라 할 수 있을 것이다. 또한 증강현실 기술은 교육, 엔터테인먼트, 광고, 방송, 군사, 산업 등 다양한 분야에서 각광받고 있다[3].

ARToolKit[4]은 데스크톱에서 마커 기반의 증강현실을 실현 가능하게 도와주는 오픈 소스 라이브러리이고, 뉴질랜드의 HITLab에서 만들어졌으며 현재 가장 널리 사용되고 있다. 또한 NYARToolKit이라는 JAVA와 C# 라이브러리의 참고 모델이 되기도 하였다. ARToolKit은 현재 SGI IRIX, PC Linux Mac OS X, PC Windows 등에서 동작이 가능하도록 제작되어있다. 또한 ARToolKit은 검정색 테두리의 사각형인 Marker를 찾는 방법으로 Find Furthest Point라는 방법을 사용하며, 가상의 이미지가 사용자의 시각에서 실제 세상의 객체와 정확하게 맞아 떨어지도록 실시간으로 정밀하게 계산되도록 한다. 다시 말해 실제 카메라의 위치와 마커의 위치 관계를 계산해야한다.

[5]에서는 2개의 모바일 폰을 이용하여 사용자에게 증강현실 서비스를 제공하는 방법을 제안하였다. 제공하는 서비스로는 2인용 증강현실 테니스게임이다. 이때 사용한 통신 매체는 블루투스이고, 서버 클라이언트 방식으로 한 대의 모바일 폰은 서버로 작동하고 다른 한 대의 모바일 폰은 클라이언트 방식이다. [5]에서는 ARToolKit을 사용하여 카메라 입력 영상에서 마커를 인식하고, OpenGL ES를 사용하여 테니스 라켓과 볼을 렌더링 하였다.

[6]에서는 증강현실기반 3D 제공 기술을 본체를 통하여 제공하는 방법을 탈피하여, 모바일 폰에서도 3D 조작 기술을 제공하는 방법을 제안하였다. [6]에서 사용한 방법은 ARToolKit Library를 사용하여 카메라 입력 영상에서 마커를 인식하고, OpenGL ES를 사용하여 3D 객체를 렌더링 한다. 이 상태에서 keypad 입력장치를 통해 특정 객체에 이동, 회전과 같은 조작을 가하면 해당 객체가 화면에 렌더링 되도록 하였다.

[7]에서는 바퀴벌레 혐오증을 치료하기 위해 모바일 플랫폼에서 증강현실을 이용한 기능성 게임을 제안하였다. 게임은 사용자가 바퀴벌레 퍼즐의 한 조각을 받으면 그 퍼즐을 완성함으로써 게임을 성공적으로 끝낼 수 있다.

[5, 6, 7]에서는 모두 마커를 이용하여 증강현실 게임을 진행하는 방식이기 때문에 로고의 광고적 효과가 없지만 본 논문에서는 기존 논문에서 사용한 마커를 이용하는 방법이 아닌 기업의 로고를 이용하여 증강현실 게임을 제공하기 때문에 기업 로고에 대한 광고적 효과가 발생할 수 있다.

모바일 폰에서 물체를 인식한다고 하면 보통 정형화되어 있는 그림과 물체 등을 인식하고 추적하게 된다. 하지만 [8]에서는 이를 보안하여 펜으로 직접 그린 그림을 인식 및 추적을 가능하게 하였다. [8]에서는 외곽선 피쳐를 사용하여 보다 유연한 인식과 추적을 가능케 하는 알고리즘도 제안하였다. 하지만 본 논문에서는 펜으로 직접 그린 그림을 인식하여 추적하는 방법이 아닌 로고를 인식하여 추적하는 방법을 제공함으로 광고적 효과를 볼 수 있다.

영상 내에서 객체를 검색하기 위한 알고리즘으로 템플릿 매칭[9]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.

영상에서 객체를 검색하는 다른 알고리즘은 SURF[9, 14]가 존재한다. SURF 알고리즘은 흑백 영상으로부터 대응점 정합을 이용하여 스케일, 회전 변환에 불변하는 특징 점을 찾는 방법으로 정합 속도가 빠르고, 좋은 정합 성능을 보여 널리 사용되고 있다. 하지만 흑백 영상에서 지역 공간 정보를 사용하기 때문에 유사한 패턴이 존재하는 영상에서 대응점의 정합 성능이 떨어지게 된다.

SURF 알고리즘의 정합 성능을 개선한 알고리즘으로 확장된 SURF[15] 알고리즘이 존재하고, 확장된 SURF 알고리즘은 빠른 속도를 가지며 컬러 공간의 불변 특성과 광역 특징을 활용한 것이다. 이 알고리즘에 사용되는 컬러 공간은

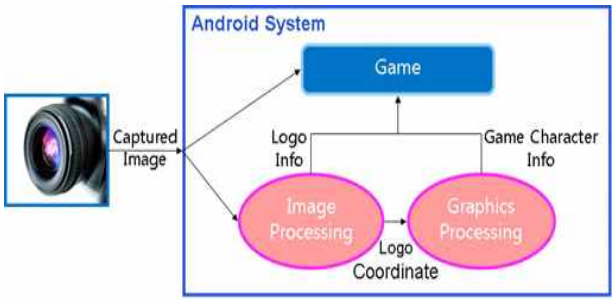
조명의 세기 뿐 아니라 강한 빛의 영향에 강인한 컬러 공간을 이용한다. 그리고 유사한 지역적 특징을 갖는 특징점을 구분하기 위해 지역적 특징뿐 아니라 광역적 범위에서 얻은 곡률값을 이용한다. 즉, 컬러 정보와 광역 정보를 이용한 확장 SURF 알고리즘은 기존의 SURF 알고리즘과 비교하여 보다 강인하게 대응점을 찾는다. 하지만 SURF 알고리즘은 특징 점의 위치가 유사한 로고가 발생하는 경우 이를 잘 못 인식할 수도 있다. 그리하여 본 논문에서는 인식의 정확도를 높이기 위해 템플릿 매칭을 사용하였다.

객체를 추적하기 위한 알고리즘으로 Color Segment를 이용한 MeanShift 알고리즘이 존재하고, MeanShift 알고리즘은 조명이나 주위 색상에 민감하게 반응하며 추적율이 낮고, MeanShift 알고리즘의 성능을 보완한 알고리즘으로 CAMShift[9, 10] 알고리즘이 존재한다. CAMShift 알고리즘은 검출된 객체 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.

Particle Filter[16]는 Bayes 추정에 바탕을 둔 고차원의 비선형, 비 가우스형의 상태 공간에 적응 가능한 시계열 필터이다. Bayes 추정은 시각 t에서 상태 벡터를, 시각 t-1에서 예측한 사전 확률과 시각 t에서 관측 결과에 의한 사후 확률에서 계산한 확률 밀도 분포에 의해 추정하는 방식으로 객체를 추적하는 필터이다.

3. 안드로이드 기반 로고인식 증강현실 시스템

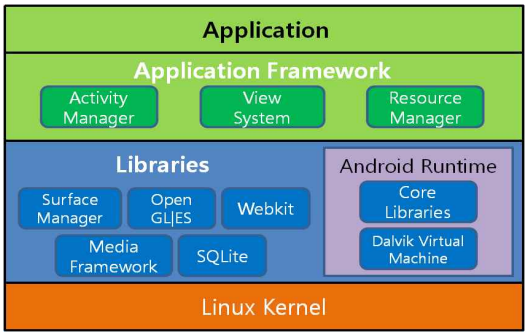
본 논문의 시스템 구조도는 (그림 1)과 같이 전체적으로 Android System 기반으로 이미지처리 모듈, 그래픽처리 모듈, 게임 모듈로 구성되며, 각 모듈은 스레드를 이용하여 독립적 동작하도록 구축되었다. 최초 카메라에서 영상을 입력받아 이미지처리 모듈과 게임 모듈로 영상을 전달한다. 이미지처리 모듈에서는 전달받은 영상을 이용하여 로고를 인식 및 추적하고, 게임모듈과 그래픽처리 모듈로 로고 정보를 전달한다. 게임모듈에서는 로고의 기본 정보를 영상처리 모듈에서 입력받아 로고의 기본 정보를 제공하고, 그래픽처리 모듈에서는 로고의 좌표를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



(그림 1) 시스템 구조도

3.1 안드로이드 시스템

본 논문에서 사용한 Application Framework와 Library는 (그림 2)와 같다. 이를 통해 하드웨어 사용 및 사용자 인터페이스와 다양한 리소스 관리를 위한 클래스를 제공받고, 각 모듈별 기능은 다음과 같다.



(그림 2) 본 논문에서 사용한 Android Architecture

필드 명	자료 형	조건	Not Null	설명
ID	varchar	Primary Key	O	자동배정
Name	varchar		O	회사이름
Location	varchar			회사위치
Representative	varchar			대표자
EstablishYear	varchar			yyyy-mm-dd
WebSite	varchar			
Field	varchar			전문분야
ImgFileName	varchar		O	이미지파일
AssociationWord1	varchar			연관 단어
AssociationWord2	varchar			연관 단어
AssociationWord3	varchar			연관 단어
AssociationWord4	varchar			연관 단어
AssociationWord5	varchar			연관 단어
AssociationWord6	varchar			연관 단어
AssociationWord7	varchar			연관 단어

(그림 3) 기업 정보 Table 설계

• Application Framework
  - Activity Manager와 View System을 이용하여 액티비티의 생명 주기를 제어하고, 액티비티의 사용자 인터페이스를 구성하는데 사용된다. 이것을 이용하여 본 시스템에서는 사용자 인터페이스를 제공하였다.
  - Resource Manager를 이용하여 외부에 위치한 파일을 불러오기 위해 사용하였다. 즉 DB파일과 로고 이미지, 캐릭터등을 제공받기 위해 사용하였다.
• Library
  - Surface Manager는 Open GL/ES을 이용하여 카메라에서 입력 받은 영상과 생성시킨 캐릭터를 화면에 그리기 위한목적으로 사용하였다.
  - SQLite[11]를 이용하여 (그림 3)에서 보는 것과 같은 DB를 구축하였고, 이 데이터를 본 시스템에서 제공받기 위해 사용하였다.
  - Webkit[12]는 기업정보를 웹 브라우저를 통해 보여주기 위해 사용하였다.

- Media Framework[12]는 게임에서 재생되는 Sound를 제공하기 위해 사용하였다.

• Android Runtime

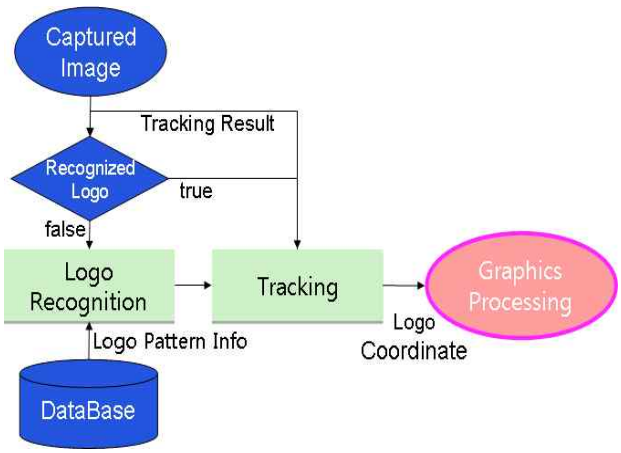
- Core Library와 Dalvik Virtual Machine은 Android System에 기본적으로 사용하는 구성 요소로 안드로이드 전용 라이브러리를 제공하고, Target Machine에서 구동을 위한 최적화된 레지스터 기반의 가상머신의 역할을 수행한다.

3.2 이미지 처리

카메라 영상을 캡쳐하는 과정은 (그림 4)와 같다. 카메라 장치로부터 입력된 원시 데이터를 캡쳐 스레드를 통하여 이미지처리 모듈에서 사용할 이미지를 생성시키고, 생성한 이미지를 이미지처리 모듈로 전송 한다.

(그림 4) Camera Capture 구조

이미지처리 모듈에서는 영상이 입력되면 (그림 5)와 같이 데이터베이스의 로고 정보를 바탕으로 로고를 인식하고 이를 이용하여 로고를 추적하게 된다. 만약 추적 과정 중에 객체 추적에 실패하면 로고 인식 과정을 통하여 로고를 다시 찾게 된다. 이미지처리 과정이 완료되면 그래픽처리를 위하여 로고가 위치한 좌표를 최종적으로 그래픽처리 과정으로 전달한다.



(그림 5) Image Processing 내부 구성

3.2.1 로고 인식

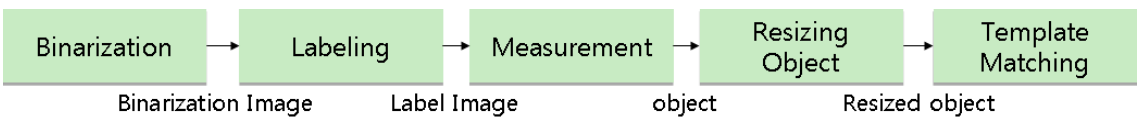
본 논문에서는 로고 인식을 위하여 템플릿매칭을 이용하였다. 하지만 템플릿매칭만 사용하는 경우 템플릿 로고의 사이즈와 대상 로고의 사이즈가 동일하지 않은 경우 로고 인식 성능이 크게 저하되기 때문에 로고를 인식하기 위해 (그림 6)과 같은 전처리 과정을 수행하였다. 먼저 이진화를 통하여 이진화된 이미지를 획득한 후 레이블링을 통하여 레이블된 객체를 획득하고, 이를 확대하여 템플릿이미지와의 유사도를 계산하였다.

대부분 기업의 로고는 단색을 많이 사용하기 때문에 본 논문에서는 Hue값을 이용하여 이진화 하였고, 로고는 단순한 모형을 유지하지만 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. 이때 로고가 회전되어 있는 경우 템플릿 매칭으로 로고 판별이 불가능하기 때문에 객체의 좌측과 중앙의 좌표를 획득하여 회전율을 계산 후 객체를 회전 변환 시켜주었다. 이렇게 받아온 객체를 데이터베이스에서 받아온 이미지와 동일한 사이즈로 resizing 후 2차적으로 템플릿 매칭의 유클리디안(Euclidean)법을 사용하여 객체를 판별하였다.

템플릿매칭 알고리즘은 템플릿 이미지를 이용하여 이미지 전체에서 가장 유사한 구간을 검색하는 방법으로 대상 이미지 전체를 탐색하면서 템플릿 이미지와 각각 얼마만큼 유사성이 있는지를 구하는 방법으로 유클리디안(Euclidean)법을 이용한다. 하지만 본 논문에서는 이미지 전체를 사용하지 않고, 레이블된 객체를 포함하는 사각형 영역의 이미지(템플릿 이미지 사이즈와 동일한 사이즈)를 사용하였다. 템플릿 매칭에서의 유클리디안 거리는 식 (1)을 이용하여 구한다. 식(1)에서 $T(x', y')$는 템플릿 이미지를 뜻하고, $I(x, y)$는 이미지 전체를 뜻한다. $R(x, y)$는 좌표 (x, y)에서의 매칭 정도를 의미하며, 매칭 수치가 임계치보다 큰 경우 해당 위치의 좌표 (x, y)에 로고가 존재하는 것으로 판별한다[9].

$$R(x,y) = \frac{\sum_{x',y'} [T(x',y') - I(x+x', y+y')]^2}{\sqrt{\sum_{x',y'} [T(x',y')^2 \times \sum_{x',y'} I(x+x', y+y')^2]}} \quad (1)$$

템플릿 매칭을 이용하여 획득한 예는 (그림 7)과 같다. (그림 7a)는 전체 영상, (그림 7b)는 템플릿 영상, (그림 7c)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.



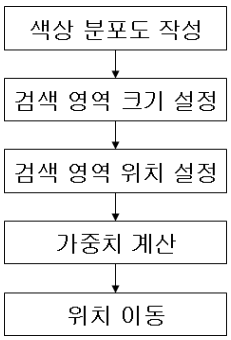
(그림 6) 로고 인식의 전처리 과정 및 Template Matching

(a) 전체 영상 (b) 템플릿 영상 (c)결과영상

(그림 7) 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 8)과 같다.



(그림 8) CAMShift 순서도

$$M_{00} = \sum_x \sum_y I(x,y) \tag{2}$$

$$M_{01} = \sum_x \sum_y xI(x,y) \tag{3}$$

$$M_{10} = \sum_x \sum_y yI(x,y) \tag{4}$$

$$(x_c, y_c) = \left(\frac{M_{10}}{M_{00}}, \frac{M_{01}}{M_{00}} \right) \tag{5}$$

$$M_{20} = \sum_x \sum_y x^2 I(x,y) \tag{6}$$

$$M_{02} = \sum_x \sum_y y^2 I(x,y) \tag{7}$$

$$M_{11} = \sum_x \sum_y xyI(x,y) \tag{8}$$

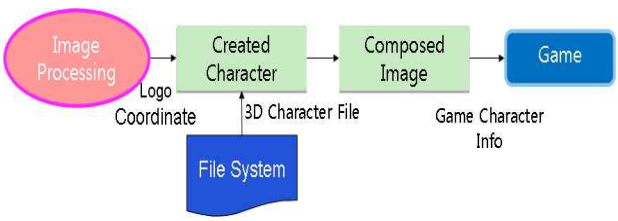
$$a = \frac{M_{20}}{M_{00}} - x_c^2, b = 2\left(\frac{M_{11}}{M_{00}} - x_c y_c \right), c = \frac{M_{02}}{M_{00}} - y_c^2 \tag{9}$$

$$l = \frac{\sqrt{(a+c) + \sqrt{b^2 + (a-c)^2}}}{2}, W = \frac{\sqrt{(a+c) - \sqrt{b^2 + (a-c)^2}}}{2} \tag{10}$$

CAMShift 알고리즘은 우선 (그림 8)과 같은 처리과정을 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 (2)부터 (4)까지의 처리가 이루어지고, n 시간에서의 처리과정은 식 (6)부터 식 (10)까지 처리가 된다. 식 (2)에서는 객체가 포함된 윈도우의 moment를 계산하고, 식 (3), (4)에서는 x축과 y축에 대한 무게 중심을 구하기 위한 각 moment를 구한 후 식 (5)을 이용하여 무게 중심을 계산한다. 이때 n-1 시간의 무게 중심과 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 (6), (7), (8), (9), (10)를 이용하는데, 식 (9)을 이용하여 객체의 높이와 폭을 측정한 후 객체의 높이와 폭을 식 (10)를 통하여 획득한다 [9, 10, 13].

3.3 그래픽 처리

그래픽처리 모듈은 (그림 9)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 시켜주었고, 이미지처리에서 획득한 좌표를 이용하여 투명 뷰에 캐릭터를 합성한다. 합성한 뷰를 게임 모듈로 전송하여 사용자에게 게임을 제공한다.



(그림 9) Graphics Processing

캐릭터 생성 시 사용한 MD2 파일 정보는 (그림 10a)와 같다. 먼저 벡터로 구성된 정점 3개를 (그림 10b)와 같이 화면에 표시 후 (그림 10c)와 같이 3개의 정점들을 서로 연결시킨다. 이때 3개의 정점을 연결시켜 (그림 10d)와 같이 다각형 면을 구성한다. (그림 10b)~(그림 10d)의 과정을 반복 실행하여 (그림 11a)와 같이 메쉬를 구성시킨다. 다각형 면을 구성시킨 후 다각형 면에 스킨(texture)을 입힌다(그림 11b). 이때 스킨(texture)은 파일 시스템에서 받아온다. 또한 캐릭터가 움직일 때는 정점들의 위치가 변하는데 변한 Vertex들의 정보를 추출하고, 추출한 정보를 이용하여 다각형 면을 재정의한다. 재정의된 다각형 면의 표면에 선형 보간법을 이용하여 스킨을 재적용한다(그림 11c).

본 논문에서 사용한 선형 보간법은 식 (13)를 이용하였고, 식 (13)의 x_i는 처음 키 프레임의 Vertex 좌표 위치, x_f는

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

(그림 10) 3D 캐릭터 생성 ((a) MD2파일 구조정보, (b)정점, (c)edge, (d)다각형 면)

다음 키 프레임의 Vertex 좌표 위치, x_c는 보간된 키 프레임의 Vertex 좌표 위치를 의미한다. interpolatePercentage는 키 프레임과 키 프레임 사이의 보간 비율을 뜻하고, 1에 가까울수록 다음 키 프레임과 가까운 위치에 보간된다. 이때 키 프레임은 모델의 애니메이션 전체 중 특정한 시간 간격마다의 프레임으로 특정한 순간에서 모델의 개별적인 위치와 자세에 대한 Vertex정보를 담고 있다.

$$x_c = x_i + interplatePercentage \times (x_f - x_i) \quad (13)$$

(a) Mesh Character (b) Character 생성 (c) 객체 움직임 적용

(그림 11) Character 생성 및 움직임 처리의 예

캐릭터와 이미지의 합성은 먼저 투명화 영역을 설정하여 SurfaceView 캐릭터를 생성한다(그림 12a). 투명화 영역은 (그림 12a)의 핑크색 영역이다. 그런 다음 (그림 12b)와 같이 카메라로부터 받아온 영상과 SurfaceView를 합성시켜 (그림 12c)와 같은 영상을 획득한다.

(a) Created Character (b) Capture Image (c) 적용 결과

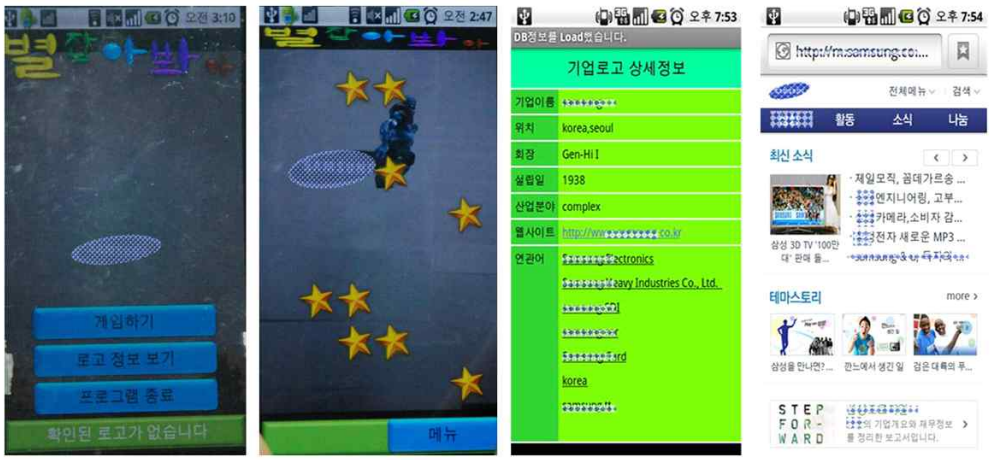
(그림 12) 그래픽 적용의 예

3.4 Augmented Reality Game

본 논문에서 구성시킨 게임 화면은 (그림 13)과 같다. (그림 13a)는 게임의 Main UI, (그림 13b)는 게임 실행 화면, (그림 13c)는 로고가 알려주는 해당 기업정보, (그림 13d)는 (그림 13c)에서 선택한 링크의 사이트이다. SQLite[11]를 이용하여 데이터베이스를 구축하였고, Webkit[12]를 이용하여 해당 사이트의 정보를 제공할 수 있도록 하였다. 메뉴 화면에서 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 제한 시간 안에 캐릭터가 별을 모두 제거하면 종료 된다.

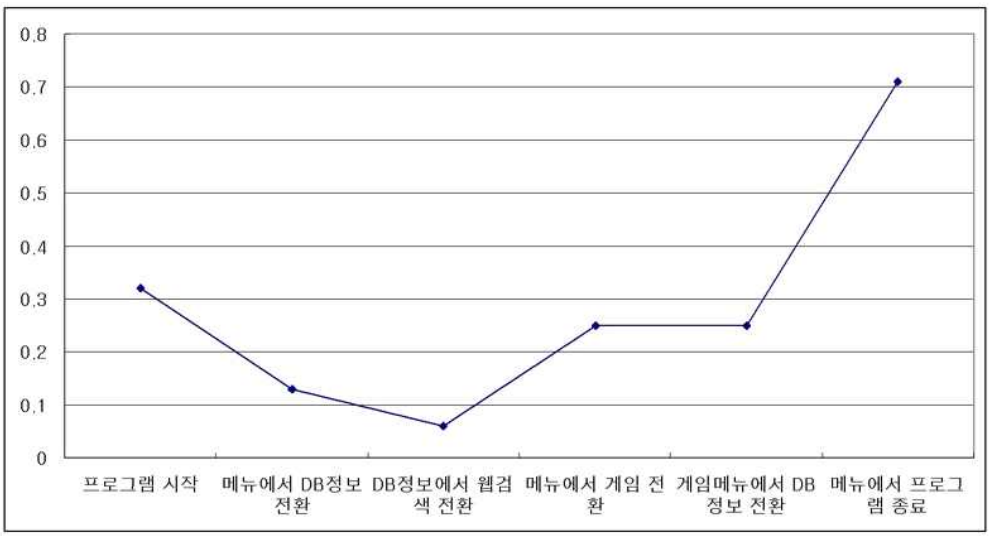
4. 실험 및 성능 평가

본 논문의 개발 환경은 Android OS 버전 2.1을 사용하여 개발하였고, 최종 데모는 Samsung Galaxy A(CPU 처리속



(a) Main UI (b) 게임 실행 화면 (c) 기업 로고 정보 (d) 기업 웹 사이트

(그림 13) 게임 실행화면 및 로고정보 제공화면



(그림 14) 각 단계별 평균 처리시간

도 : 700MHz)에서 테스트하였다. 게임 화면 및 이벤트를 처리하기 위해 Eclipse GALILEO와, SDK API 7을 사용하였고, 영상처리 목적으로 OpenCV를 사용하였다. 그리고 그래픽 합성을 목적으로 OpenGL을 사용하였으며 반응 속도에 대한 실험은 스마트 폰을 이용하였다. 본 논문에서는 세 가지 실험을 실시하였는데 첫 번째 실험은 안드로이드 반응 속도에 대한 실험이고, 두 번째 실험은 영상처리에 관련된 실험, 마지막 실험은 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 320×508 사이즈이다.

4.1 안드로이드 반응 속도

안드로이드의 반응 속도를 체크하기위해 각 구간 프로그램 시작 시간, 메뉴에서 DB 정보 전환 시간, DB 정보에서 웹 검색 전환 시간, 메뉴에서 게임 전환시간, 게임 메뉴에서 DB 정보 전환 시간 그리고 프로그램 종료 시간 등을 기록하였다. 평균 전환 시간은 (그림 14)와 같다. (그림 14)의 y 축은 시간을 의미하며 단위는 초이다.

반응 속도를 알아보기 위한 실험 방법은 같은 동작을 50번 반복하여 실험을 하였다. 안드로이드 반응 속도는 시작과 종료 시에 대체적으로 반응속도가 오래 걸리며 이는 Camera 등의 하드웨어 자원 사용 및 해제와 관련된 것이 원인으로 추정된다. 또한 게임 화면 전환에서도 반응 시간이 늦어지는 증상이 있는데 이는 게임 실행 시 영상처리 및 그래픽처리 시간이 발생하는 것이 원인이라 할 수 있다.

4.2 영상처리 알고리즘의 성능 평가

실시간 추적 알고리즘을 선택하기 위해 CAMShift 알고

〈표 1〉 필터의 색상 추적 분석

실험 NO	실험 환경	CAMShift			Particle (분산 25, 파티클수 2000)		
		색상 추적	총 프레임	추적 (%)	색상 추적	총 프레임	추적 (%)
1	한 개의 객체가 좌우로 직선 운동	353	620	56	341	620	55
2	두 개의 객체가 교차되는 경우	856	856	100	856	856	100
3	객체가 유사 객체 인근을 통과하는 경우	794	794	100	794	794	100
4	두 개의 움직이는 객체가 교차하는 경우	706	706	100	632	706	89
5	두 개의 객체가 충돌하는 경우	241	241	100	241	241	100
6	한 개의 객체가 장애물 통과시	353	544	64	320	544	58
7	두 개의 객체가 장애물 통과시	180	647	27	386	647	59
계		3,483	4,408	78	3,570	4,408	80

리즘과, Particle Filter의 성능을 분석을 하였다. 첫 번째 실험은 추적 정확도를 실험하였고, 두 번째 실험은 각 알고리즘의 처리 시간을 비교하였다. 단, 본 실험은 안드로이드 기반이 아닌 윈도우 기반에서 실험을 진행한 것으로 안드로이드 상에서와 다소 차이가 발생할 수도 있지만, 각 알고리즘 간의 성능을 동일한 환경에서 실험하였기 때문에 본 논문에서는 정확도와 처리 속도를 고려하여 추적 알고리즘을 선택하여 안드로이드 기반 로고 인식 시스템에 탑재 하였다.

추적의 정확도를 확인하기 위해 본 논문에서는 <표 1>과 같은 7가지 환경조건을 설정하여 실험하였고, 두 알고리즘 모두 색상 조건에 의해서만 추적 가능하도록 설정하였다. 또한 Particle Filter와 CAMShift 알고리즘은 OpenCV에서 기본적으로 제공하는 알고리즘을 사용하였다.

Particle Filter와 CAMShift 알고리즘의 성능은 <표 1>과 같다. 두 개의 청색 객체가 교차 되는 경우보다 한 개의 청색 객체가 좌우 직성운동을 하는 경우가 낮은 추적률을 보이는데 이는 좌우 직선 운동 시 객체가 화면을 이탈하는 하는 경우가 발생하였기 때문이다. Particle Filter와 CAMShift 알고리즘의 추적률을 종합적으로 분석하면 Particle Filter가 CAMShift보다 우수한 성능을 보이는 것으로 보이지만 안드로이드 기반에 탑재할 알고리즘인 점을 감안하면 처리속도 또한 중요한 요인이다.

두 개의 추적 알고리즘의 처리 시간을 비교 하였을 때 2,000개의 Particle을 이용하여 처리시간을 측정한 결과 평균

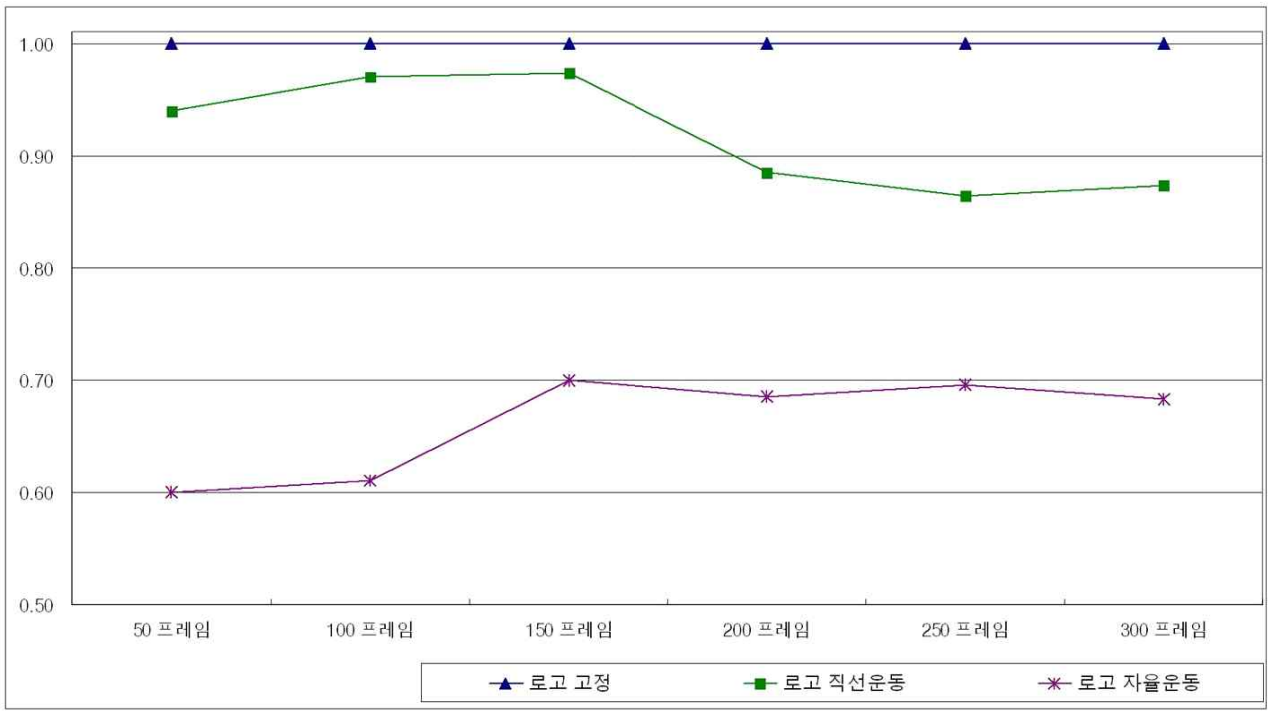
〈표 2〉 필터의 처리 시간 비교 (단위 : ms)

프레임	CamShift	particle			
		2,000	1,000	500	100
1	2.388	7.840	6.277	2.090	0.447
2	1.791	10.405	5.206	2.164	0.416
3	1.701	9.991	4.363	2.187	0.394
4	1.689	11.391	4.442	2.079	0.491
5	1.690	12.292	4.748	2.212	0.439
6	1.721	8.952	4.520	1.974	0.413
~					
849	1.803	12.292	4.203	2.087	0.187
850	1.925	10.747	6.930	1.892	0.187
851	1.894	14.101	4.612	1.874	0.188
852	1.717	11.074	5.761	1.853	0.194
853	1.767	9.162	3.998	1.718	0.193
854	1.692	10.218	5.666	2.737	0.188
855	1.722	11.940	4.809	1.676	0.205
평균	1.915	11.399	5.098	2.176	0.297

11.39ms가 소요된 반면 CamShift를 이용하여 처리 시간을 비교한 결과 평균 1.91ms의 처리속도가 측정되었다. 안드로이드 기반으로 가정하여 비교한다면 처리속도도 중요한 요인이기 때문에 본 논문에서는 <표 2>에서 보는 것과 같이 Particle의 개수를 조절하여 실험한 결과 약 500개의 Particle과 100개의 Particle 사이의 수를 사용하였을 때 CamShift 알고리즘과 동일한 속도가 발생하는 것을 알 수

〈표 3〉 Partilce 수에 따른 추적률 비교

실험 NO	수	100		500		1,000		2,000	
프레임 수		프레임	추적률	프레임	추적률	프레임	추적률	프레임	추적률
5	성공	53	21.9	148	61.16	242	100	242	100
242	실패	189	78.1	94	38.84	0	0	0	0
6	성공	95	17.46	153	28.13	292	53.68	320	58.82
544	실패	449	82.54	391	71.88	252	46.32	224	41.18
계		19.68		44.64		76.84		79.41	



(그림 15) 안드로이드 기반 추적률 및 인식율

있다. 그리하여 본 논문에서는 <표 3>과 같이 Particle의 수를 조절하며 추적율을 계산하였고, 이때 사용한 실험 방법은 <표 1>의 5번 실험과 6번 실험을 이용하였다.

Particle Filter의 Particle 수를 조절하여 추적률을 계산한 결과, 100개의 Particle을 사용하였을 때 추적률은 19.68 %이고, 500개의 Particle을 사용하였을 때 44.64 %의 추적률을 보였다. 동일한 속도로 추적률을 비교하면 Particle Filter를 이용하여 객체를 추적하는 것 보다 CamShift를 이용하는 것이 좋다는 결론이 도출된다. 즉 본 논문에서는 실험을 통하여 CamShift 알고리즘을 이용하여 객체를 추적하는 것이 Particle Filter를 이용하여 추적하는 것 보다 좋은 성능을 보이기에 CamShift 알고리즘을 이용하여 객체를 추적 하였다.

4.3 안드로이드 기반 영상처리 성능평가

본 논문의 실험에서는 4.2절에서 선택한 추적 알고리즘을 사용하였고, 이미지처리 모듈에서 처리하는 이미지의 수를 50 프레임 단위로 증가시키면서 인식율을 측정한 결과 (그림 15)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 자율 운동이 발생하였을 때의 인식율을 계산하였다.

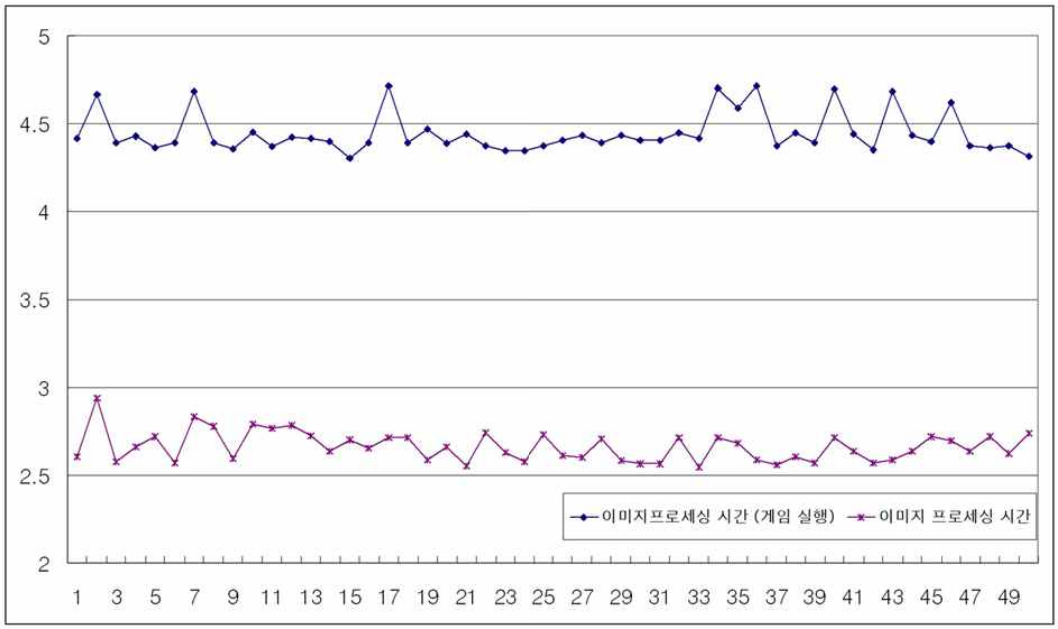
실험 결과, 로고의 움직임을 발생시키지 않았을 때 모든 프레임에서 로고를 인식함으로 100% 인식율을 보였고, 로고의 직선 운동 시 92 %의 인식률을 보였다. 또한 로고의 자율 운동 시 66 %의 인식율을 보였다. 로고의 자율 운동이 발생하였을 때 로고의 직선 운동보다 낮은 인식율을 보이는데 이는 이미지처리 속도가 낮아서 발생하는 원인이라 할 수 있을 것이다. 즉, 한 프레임을 처리하는 시간동안 객체의 이동 변화가 큰 경우 로고의 움직임에 대한 변화를 반영하지 못하는 문제가 발생하기 때문에 로고 인식을 실패한다. 만약 안드로이드 폰의 성능이 좋아지면 인식율 또한 증가할 것이다.

4.4 안드로이드 기반 영상 및 그래픽 처리속도 평가

안드로이드 기반 영상처리 속도는 두 가지를 측정하였다. 첫 번째는 게임 실행 시 영상 처리 속도와 게임 비실행 시 영상처리 속도를 측정하였고, 두 번째는 응용 프로그램 전체 처리 속도를 측정하였다.

첫 번째 측정에서 획득한 결과는 (그림 16a)와 같다. 게임 실행 시 영상처리의 평균 속도는 4.45초, 게임 비실행 시 영상처리의 평균 속도는 2.66초로 측정되었다. 전체 처리 속도를 측정한 결과는 (그림 16b)와 같고, 그래픽 처리를 포함한 전체 평균 처리 속도는 7.23초로, 그래픽 처리를 포함하지 않은 전체 평균 처리 속도는 3.67초로 측정되었다.

알고리즘의 처리 속도가 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 처리 속도는 더 빨라질 것이고, (그림 15)의 로고 자율 운동에서 발생하는 인식율 또한 개선될 수 있다.



(a) 이미지 처리 시간 비교

(b) 전체 처리 시간 비교

(그림 16) 영상 한 프레임 처리 시간 비교

5. 결 론

본 논문에서는 안드로이드 폰 기반 로고 인식 증강현실 게임을 제안하였고, 어플리케이션 개발에 사용된 기술과 알고리즘들을 소개하였다. 로고를 인식하기 위해 템플릿 매칭 알고리즘을 사용하였고, 인식한 로고를 추적하기 위해 CAMShift 알고리즘을 사용하였다. 또한, CAMShift 알고리즘과 Particle Filter의 성능 비교 실험을 하였고, 실험 결과 안드로이드 기반에서는 처리 속도 문제로 인하여 Particle Filter보다는 CAMShift알고리즘이 더 적합한 것을 알 수 있었다.

처리 속도가 현저하게 낮은데 이는 저 사양인 700Mhz의

안드로이드 폰에서 실시간 영상처리 알고리즘을 탑재하였을 때 발생한 결과이다. 만약 안드로이드 폰의 성능이 더 좋아진다면 실시간 영상처리 속도는 자연적으로 개선될 것이다. 또한 본 논문에서 사용한 영상의 사이즈는 320×508을 사용하였기 때문에 영상의 사이즈를 적절히 조절하거나 그래픽을 3차원 그래픽이 아닌 2차원 그래픽으로 제공한다면 더 좋은 성능도 기대할 수 있을 것이다. 또한 로고의 자율 운동 시 로고 인식율이 낮은 것을 확인하였고, 이는 영상처리 속도에서 발생하는 원인이라 할 수 있을 것이고, 안드로이드 폰의 성능이 개선된다면 인식율 또한 개선될 것이다.

참 고 문 헌

[1] R.T.Azuma, “A survey of augmented realty", In Presence: Teleoperators and Virtual Environments 6, pp.355-385, 1997.

[2] P. Milgram and H. Takemura, “Augmented Reality: A Class of Displays on the Reality-Virtuality Continuum." Proc. of SPIE Telemanipulator and Telepresence Technologies, Vol. 2351, pp.282-292, 1994.

[3] Sang-Goog Lee, “Recent Advances in Augmented Realty”, SAIT(Samsung Advanced Institude of Technology) Technical Report, pp.72-76, Feb., 2005.

[4] Kato, H., Billinghurst, M., Marker Tracking and HMD Calibration for a video-based Augmented Reality Conferencing System, In Proceedings of the 2nd International Workshop on Augmented Reality (IWAR 99), pp.85-94, 1999, USA.

[5] A. Henrysson, M. Billinghurst, and M. Ollila, "Face to Face Collaborative AR on Mobile Phones", IEEE and ACM International Symposium on Augmented Reality 2005, pp.80-89, 2005.

[6] A. Henrysson, M. Billinghurst and M. Ollila, "Virtual Object Manipulation using a Mobile Phone", In the 15th International Conference on Artificial Reality and Telexistence (ICAT 2005), pp.164-171, Christchurch, New Zealand, Dec., 2005.

[7] C. Botella, J. Breton-Lopez, S. Quero, R.M. Banos, A. Garcia-Palacios, I. Zaragoza, M. Alcaniz, “Treating cockroach phobia using a serious game on a mobile phone and augmented reality exposure: A single case study” Original Research Article, Computers in Human Behavior, Vol.27, pp.217-227, Available online 25 August, 2010.

[8] Hagbi N, Bergi O, El-Sana J, Billinghurst M, "Shape Recognition and Pose Estimation for Mobile Augmented Reality", Mixed and Augmented Reality, pp.65-71 (7 pages), ISMAR 2009.

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

[10] 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.

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

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

[13] 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.

[14] H. Bay, T. Tuytelaars, and L. V. Gool, "Surf: Speeded up robust features", European Conference on Computer Vision, Vol.3951, pp.404-417, 2006.

[15] 윤현섭, 한영준, 한헌수, "컬러 불변 특징과 광역 특징을 갖는 확장 SURF(Speeded Up Robust Features) 알고리즘", 대한전자공학회, 제 46권, 제 6호, pp.58~67, 2009.

[16] MICHAEL ISARD AND ANDREW BLAKE, "CONDENSATION—Conditional Density Propagation for Visual Tracking", Received July 16, 1996; Accepted March 3, 1997.

정 은 영

e-mail : lovecat6725@naver.com

2011년 금오공과대학교 컴퓨터공학부 소프트웨어공학과(학사)

2011년~현 재 웹케시(주) 사원

관심분야 : 인공지능, 모바일

정 운 국

e-mail : jukyellow@ktnet.com

2011년 금오공과대학교 컴퓨터공학부 소프트웨어공학과(학사)

2011년~현 재 (주)한국무역정보통신 사원

관심분야 : 임베디드시스템, 웹서비스

임 선 진

e-mail : lovecat6725@naver.com

2011년 금오공과대학교 컴퓨터공학부 소프트웨어공학과(학사)

2011년~현 재 (주)오픈앤와이즈 사원

관심분야 : 모바일, 데이터베이스

문 창 배

e-mail : moonyeses@naver.com
2007년 금오공과대학교 컴퓨터공학부 소프트웨어공학과(학사)
2010년 금오공과대학교 컴퓨터공학부 소프트웨어공학과(석사)
2010년~현 재 금오공과대학교 컴퓨터공학부 소프트웨어공학과 박사과정
관심분야 : 패턴인식, 영상처리, 인공지능, 신호처리, 감성공학

김 병 만

e-mail : bmkim@kumoh.ac.kr
1987년 서울대학교 컴퓨터공학과(학사)
1989년 한국과학기술원 전산학과(석사)
1992년 한국과학기술원 전산학과(박사)
1992년~현 재 국립금오공과대학교 컴퓨터공학부 교수
1998년~1999년 미국 UC, Irvine 대학 방문교수
2005년~2006년 미국 콜로라도 주립대학 방문교수
관심분야 : 인공지능, 정보검색, 정보보안