

HoloChess

– Hologram과 음성인식을 이용한 체스게임 –

성 명	홍길동		
프로젝트 명	HoloChess(홀로그램과 음성인식을 이용한 체스게임)		
개 발 인 원	5 명	개발기간	2016.9.06 ~ 2016.11.14
개 발 언 어	C#		
개 발 툴	Visual Studio 2015, Unity, Oracle		
사 용 기 술	WPF, OpenCV, Vuforia		

목차

I.	프로젝트 배경	3
i.	프로젝트 주제	3
ii.	프로젝트 개발 목적	3
II.	프로젝트 진행 보고	4
i.	개발도구	4
ii.	프로젝트 목표	5
III.	프로젝트 개발 내용	7
i.	전체 아키텍처	7
ii.	본인 구현부분	9
iii.	알고리즘	12

I. 프로젝트 배경

i. 프로젝트 주제

1. 음성인식과 홀로그램을 통해 소설 '해리포터' 속 마법사 체스 게임을 구현한다.
2. 분야 별 개발 주제
 - A. 음성인식: 음성을 인식하여 체스 말을 제어한다.
 - B. 홀로그램/Unity: 체스 게임의 진행상황을 홀로그램으로 보여준다.
 - C. OpenCV: 얼굴을 인식하여 로그인하고 게임을 시작한다.
 - D. Vuforia: 마커를 인식하여 Mobile App 에서 체스판을 띄운다.
 - E. 서버/데이터베이스: 각 회원의 회원정보와 체스 기보 정보를 저장하고 관리한다.
 - F. AI: 1 인 게임 시 컴퓨터가 상대가 되어 대전한다.
 - G. Android: Android OS 에서 체스 게임을 2D 또는 3D 모델을 통해 관전할 수 있는 Application 을 개발한다.

ii. 프로젝트 개발 목적

1. 목적

: '해리포터'영화에 등장한 마법사 체스를 모방한 게임들이 출시 되었지만 체스의 말을 흉내 낸 정도에 머물렀다. 본 프로젝트에서는 마법사 체스를 3 차원 홀로그램 영상으로 보여주며 음성으로 체스의 말들을 움직이게 함으로 영화와 매우 흡사한 게임을 구현해내며 게임 계의 새로운 장르를 개척하며 대중의 관심을 이끌어 내는데 목적을 둔다.

2. 차별점

: 현재 나와있는 마법사 체스는 기존의 체스와 다를 것 없이 정적인 움직임으로 실제 영화에서 음성을 통해 체스 말을 움직이거나 전투하는 모습을 살리지 못했다. 때문에 홀로그램 화면을 통해 한 나라의 왕이 되어 음성으로 체스 말을 움직이도록 하고 3 차원 입체 영상으로 보여주어 각 체스 말의 동작을 구현하여 최대한 영화와 유사한 게임을 구현한다.

II. 프로젝트 진행 보고

i. 개발도구

1. 기술

#	이름	사용처
1	C#	Unity 를 통한 게임 클라이언트 구현
2	WPF	서버 구축
3	OpenCV	얼굴을 인식하여 로그인 및 게임 시작
4	Vuforia	마커를 인식하여 Mobile App 에 체스판 띄우기
5	Oracle	데이터베이스 개발 및 제어 ① 데이터베이스 테이블 제작 ② DB 에 데이터 입출력

2. 개발 S/W

#	이름	사용처
1	Unity5.4	프로젝트 표준 개발 도구
2	Visual Studio 2010	프로젝트 개발 도구
3	Window10	PC Application 동작 환경
4	Android	Mobile Application 동작 환경
5	Oracle	데이터베이스 생성 및 관리

3. 장비

#	이름	사용처
1	PC	윈도우 PC: 5 대 개인 개발 및 테스트 환경 제공
2	마이크	핀 마이크: 2 대 플레이어의 음성을 입력 받음
3	홀로그램 장치	게임 화면을 홀로그램을 띄움
4	카메라	얼굴을 인식
5	Android Phone	Mobile Application 설치 및 실행

ii. 프로젝트 목표

1. 1 단계: 전체 프로젝트 달성을 위한 기본적인 기능들을 구현한다.

업무 분야	기능
음성인식	음성을 인식하여 체스 말이 위치한 좌표를 인식하고 이동할 좌표를 인식한다.
Unity	① UI 구성: 로그인 화면과 체스 게임 화면을 구현한다. ② 게임 동작 구현 A. 선택된 체스 말을 표시하도록 한다. B. 각 체스 말의 이동 가능 범위 내에서 움직이게 한다. ③ 홀로그램 A. 체스 게임 화면을 홀로그램 이미지(4 방향 이미지)로 만들고 홀로그램을 띄워본다.
OpenCV	① 얼굴을 인식하여 회원가입과 로그인을 한다

2. 2 단계: 서버 – 클라이언트를 구축하여 얼굴인식 로그인 기능을 추가한다.

업무 분야	기능
1 단계까지 개발된 기능은 기본적으로 포함된다.	
음성인식	2 대의 마이크가 번갈아 가며 음성을 인식하도록 한다.
Unity	① 홀로그램 A. 체스 말의 애니메이션 동작을 실시간으로 홀로그램 영상으로 변환하도록 한다.
홀로그램	홀로그램 장치를 제작한다.
클라이언트	① 회원가입 A. 카메라로 얼굴을 찍고 입력된 회원정보와 함께 서버로 전송한다. ② 로그인 A. 카메라가 얼굴을 인식하면 인식된 얼굴 경로를 서버로 전송한다. B. 로그인이 되면 회원의 정보를 출력한다. C. 두 명 이상 로그인 된 상태일 때 게임을 시작한다.
서버/데이터베이스	① 데이터베이스 A. 회원 테이블을 구성한다.

	<p>B. 체스 테이블을 구성한다.</p> <p>② 서버</p> <p>A. 회원가입: 클라이언트로부터 회원 데이터를 전송받고 데이터베이스에 저장하고 얼굴 이미지에 대한 정보를 저장한다.</p> <p>B. 로그인: 데이터베이스로부터 기존 회원 정보를 불러와 입력된 정보와 얼굴 이미지와 비교하여 로그인한다.</p> <p>C. 게임 승률 저장: 게임 종료 후 클라이언트가 전송한 정보를 이용하여 회원의 레이팅을 계산하고 데이터베이스에 저장한다.</p>
AI	<p>① 가중치 결정: 보호하거나 공격하는 체스 말의 수와 체스 말의 평가치, 기존 체스 전술을 이용하여 각 경우의 수에 대해 가중치(상태 점수)를 결정한다.</p> <p>② 트리 탐색 알고리즘</p> <p>A. 현재 상태에서 한 단계 예측 후 시뮬레이션 한 최종결과에 따라 트리 정보를 업데이트한다.</p> <p>B. 계산된 가중치를 이용하여 가장 최선의 수를 두도록 구현한다.</p>

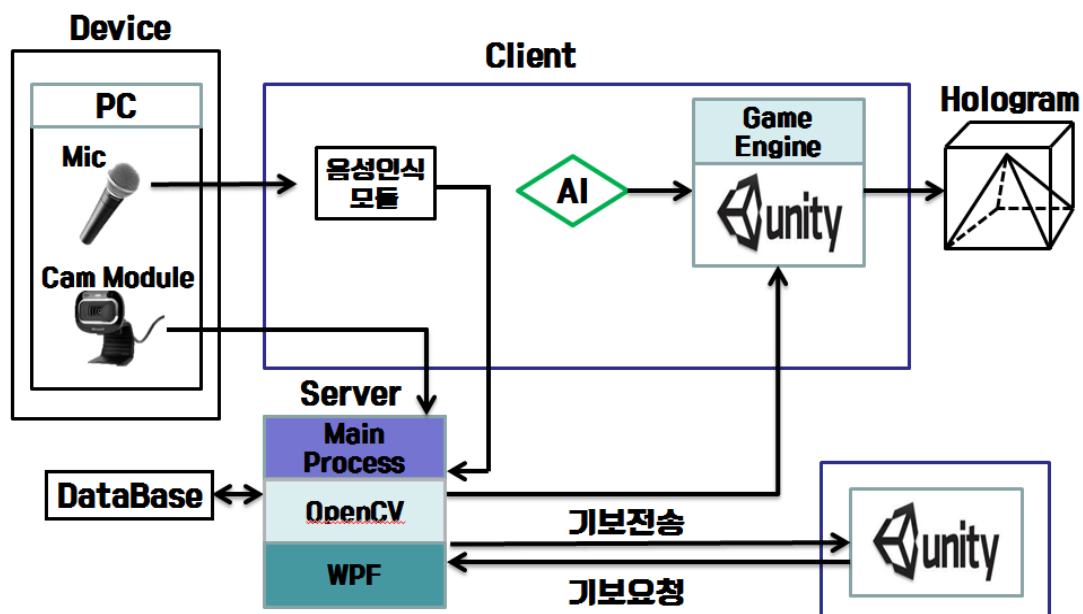
3. 3 단계:

업무 분야	기능
2 단계까지 개발된 기능은 기본적으로 포함된다.	
OpenCV	<p>① 얼굴을 haarcascades 를 이용하여 검출을 한다..</p> <p>② 정확도 높이기</p> <p>A. 사진의 숫자가 많으면 많을수록 인식률이 높아지고 적을수록 인식 확률이 많이 떨어진다.</p> <p>B. 임계값 조절로 최적의 값을 조절하기.</p> <p>③ 다른 opencv 활용방안으로 연구하고 개발하기(손 인식을 통한 오프닝 영상 제작)</p>
Android	<p>① UI 구현</p> <p>A. 로그인 화면과 마커를 이용해 체스를 관전할 수 있는</p>

	<p>AR 화면, 2D 로 체스를 관전할 수 있는 화면을 구현한다.</p> <p>B. 서버로부터 기보 정보를 받아 진행 중이거나 종료된 게임을 관전할 수 있도록 한다.</p>
--	---

III. 프로젝트 개발 내용

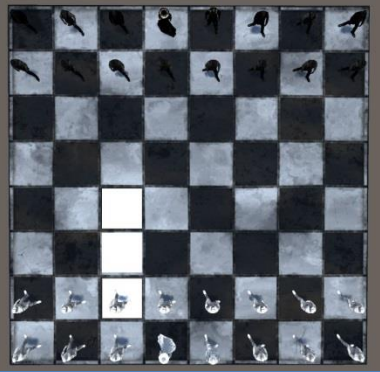
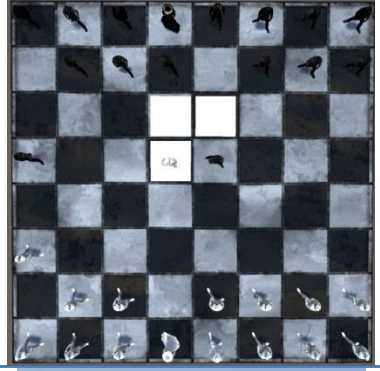


i. 전체 아키텍처



	파일 명칭	기능
1	Client.exe	<p>Cam module control</p> <ul style="list-style-type: none"> 캠 모듈을 제어하여 이미지 촬영 <p>OpenCV</p> <ul style="list-style-type: none"> 캡처한 얼굴 이미지를 서버로 전송 <p>VoiceRecognition module control</p> <ul style="list-style-type: none"> 음성을 입력 받아 문자열로 치환하여 전송 인식된 문자열을 통해 해당 위치의 체스말 이동. <p>Game Engine</p> <ul style="list-style-type: none"> 인식된 명령과 이미지를 바탕으로 게임을 진행 게임 화면을 2D UI 와 3D 게임화면을 출력 <p>AI</p> <ul style="list-style-type: none"> 1 인 플레이 시 컴퓨터와 대전할 알고리즘 설정

2	Server.exe	<p>Server</p> <ul style="list-style-type: none"> - 로그인, 로그아웃, 회원가입, 이미지 수신, 기보 정보 송수신 및 저장, 결과 정보 저장 <p>OpenCV</p> <ul style="list-style-type: none"> - 클라이언트에게서 받은 이미지를 회원가입 시 저장한 이미지와 비교하여 통해 데이터베이스에 있는 회원정보를 확인 - 확인된 정보를 클라이언트로 전송 <p>WPF</p> <ul style="list-style-type: none"> - WPF 를 통해 서버를 On/Off 하는 기능과 패킷을 직관적으로 확인 가능하게 하고 DB 정보 확인 및 수정, 삭제가 가능하도록 구현
3	Data.db	<p>데이터베이스</p> <ul style="list-style-type: none"> - 회원 데이터베이스를 구성 - 회원 이미지 라벨, 아이디, 패스워드, 회원 이름, 전적, 승률, 접속정보 정보를 관리 - 체스 기보 번호, 흑,백 아이디, 결과, 체스 기보, 체스게임상태 정보를 관리.
4	Android	<p>Android</p> <ul style="list-style-type: none"> - 현재 게임 중이거나 이미 종료된 게임을 선택하여 다시 볼 수 있음 - 2.5 초에 한번씩 서버로 기보를 요청 - 서버로부터 기보를 전송 받아 버튼 클릭을 통해 다시 볼 수 있도록 함

ii. 본인 구현부분

1. 게임로직	
<ul style="list-style-type: none"> > 선택된 체스 말의 이동 가능 경로 표시 > 체크, 체크메이트, 스테일메이트 상황 확인 > 체크 상태일 때 체스 말의 이동 제한 > 특수상황(프로모션, 앙파상, 캐슬링) 구현 	
	<p>게임화면</p> <p>선택한 말의 이동 가능 경로 표시</p>
	<p>앙파상</p> <p>상대 폰이 2 칸 이동한 경우 이동 경로를 공격할 수 있는 룰</p>
	<p>프로모션</p> <p>폰이 상대 진영의 끝까지 가면 퀸으로 승격하는 룰</p>
	<p>캐슬링</p> <p>킹과 룯이 한번도 움직이지 않았을 때 킹과 룯의 위치를 바꾸는 룰</p>



2. AI

- > 현재 체스 판의 상태를 Root node 한 Tree 구현
- > MiniMax Algorithm 과 Alpha-Beta Pruning 이용하여 현 상태에서 최적의 수를 두도록 구현
- > 특수상황(프로모션, 앙파상, 캐슬링) 구현

알고리즘 파트에서 설명




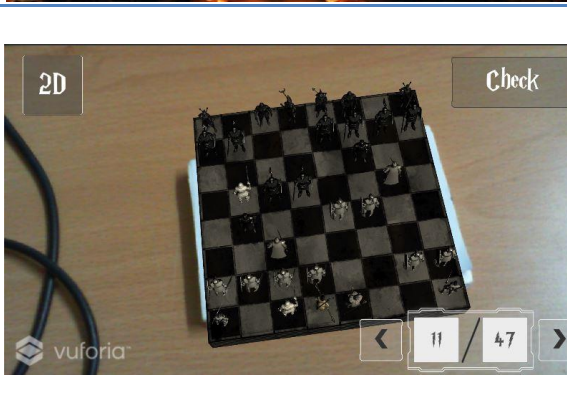

3. Mobile App

- > 로그인 기능 구현
- > MyInformation, Watch a Game, Replay the Game 버튼 구현
- > MyInformation 버튼 클릭 시 로그인한 회원 정보 출력
- > Watch a Game 버튼 클릭 시 다른 회원이 진행 중이거나 종료한 게임 목록 출력
- > Replay the Game 버튼 클릭 시 로그인한 회원이 이전에 한 게임 목록 출력
- > 출력된 게임 목록 중 하나 클릭 시 AR 을 이용하여 마커 인식 시 체스 판과 체스 말을 띄우도록 구현
- > 2D 버튼 구현하여 클릭 시 게임 화면을 2D 게임화면으로 볼 수 있도록 구현
- > 2.5 초에 한번씩 서버로부터 기보를 받아와 실시간 갱신하도록 구현



로그인 화면

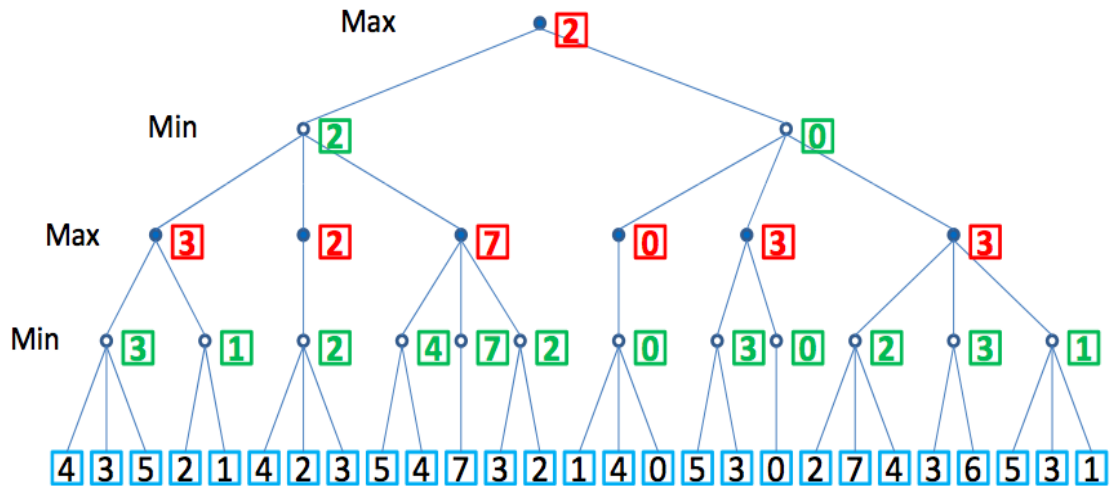
회원 가입 시 입력한 아이디와 비밀번호를 입력하여 로그인을 하도록 한다.

	<h3>My Information</h3> <p>My Information 버튼을 클릭했을 때 오른쪽 화면에 로그인한 회원의 이름과 전적, 승률을 띄우도록 한다.</p>
	<h3>Watch a Game</h3> <p>Watch a Game 버튼을 클릭했을 때 다른 회원이 진행 중이거나 종료한 게임의 목록을 오른쪽에 띄워준다. 목록 맨 하단의 More 버튼을 클릭할 때마다 목록을 10 개씩 추가로 띄워준다.</p>
	<h3>Replay the Game</h3> <p>Replay the Game 버튼을 클릭했을 때 로그인한 회원이 이전에 한 게임 목록을 띄워준다.</p>
	<h3>AR 화면</h3> <p>마커 인식 시 체스 판과 체스 말을 마커 위에 띄워준다. 오른쪽 하단의 좌/우 버튼을 이용해 현 상태의 이전 수와 다음수로 전개할 수 있다. 2D 버튼을 클릭하여 2D 화면으로 이동할 수 있다.</p>
	<h3>2D 화면</h3> <p>게임 전개를 2D 로 볼 수 있다. 동작은 AR 화면과 동일하다. 3D 버튼을 클릭했을 때 다시 AR 화면으로 넘어갈 수 있다.</p>

iii. 알고리즘

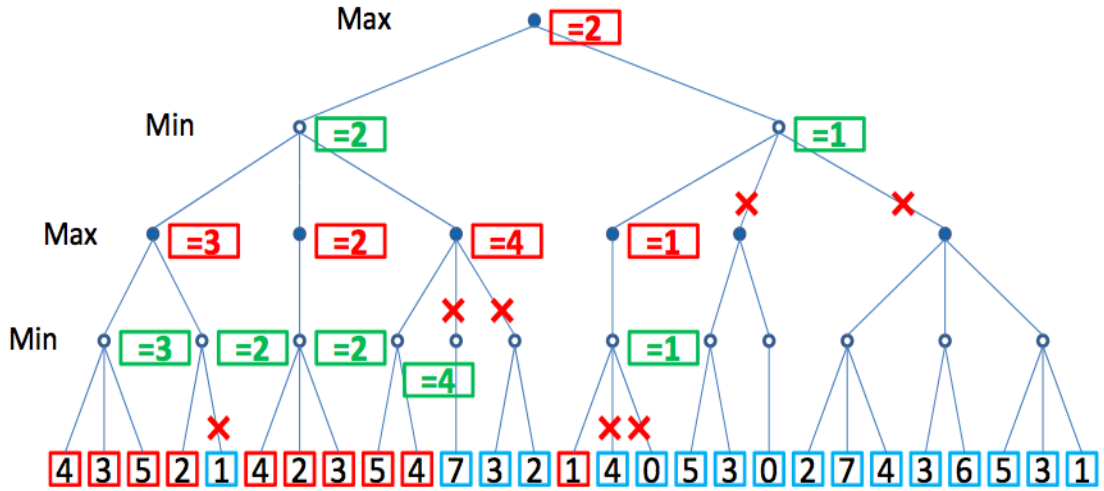
➤ AI

◆ Min-Max Algorithm



- 오목, 바둑, 체스 등 2 인용 제로섬 게임의 AI 개발 시 주로 사용하는 알고리즘이다.
- 흑과 백이 번갈아 두며 흑색이 AI 라고 할 때, Root Node 에는 현재 체스 판의 상태가 들어가고 그 자식 노드에는 흑의 이동 가능한 모든 경우의 수를 추가한다. 그리고 흑의 자식 노드에는 백의 이동 가능한 모든 경우의 수를 추가한다.
- 트리의 끝에 위치하는 터미널 노드가 가지는 값은 클수록 흑에 유리하고 백에 불리하도록 계산한다.
- 흑과 백 모두 자신에게 가장 유리한 수를 고를 것이므로 백의 차례일 때는 가장 작은 수를, 흑의 차례일 때는 가장 큰 수를 가지는 노드를 최선의 수로 고른다.
- 위의 이미지로 보면 Min 이라고 써있는 계층이 백의 차례 Max 라고 써있는 계층이 흑의 차례이다.

◆ Alpha-Beta Pruning



- Min-Max Algorithm 의 속도 향상을 위한 알고리즘이다.
- 모든 자식 노드를 방문해 값을 비교하게 되면 속도가 떨어지므로 전혀 가능성이 없는 노드에는 방문하지 않도록 해 속도를 향상시키는 알고리즘이다.
- Alpha = $-\infty$, Beta = ∞ 로 두고 시작한다.
- Min level 일 때 그 자식 노드들의 값과 Alpha 값을 비교하여 더 큰 값이 Beta 값이 되도록 한다.
- Min Node 가 가질 수 있는 값은 Beta 보다 작거나 같은 값이다.
- Max level 일 때 그 자식 노드들의 값과 Beta 값을 비교하여 더 작은 값이 Alpha 값이 되도록 한다.
- Max Node 가 가질 수 있는 값은 Alpha 보다 크거나 같은 값이다.
- Alpha 값과 Beta 값을 비교하였을 때 Alpha 값이 Beta 값보다 크거나 같다면 그 사이에 값이 존재할 수 없으므로 더 이상의 탐색을 하지 않는다.
 - ✓ Alpha = 3, Beta = 2 일 때, 3 보다 크면서 2 보다 작은 값은 있을 수 없으므로 더 이상의 탐색은 하지 않는다.