**매직 큐브**

최규화, 이관구, 유창민

한국산업기술대학교 게임공학부

gusdl576@naver.com, [dufdif@gmail.com](mailto:dufdif@gmail.com), min940913@naver.com

**요약**

컴퓨터 게임에는 여러 장르가 있고, 사람들마다 취향이 다르기 때문에 각자 선호하는 게임의 장르가 있다. 그러나 귀여운 외모의 캐릭터가 나오는 캐쥬얼한 게임은 남녀노소 불문하고 그 수요가 많다. 따라서 우리는 1년동안 졸업연구를 진행하여 큐브모양의 귀여운 캐릭터와 쉬운 조작 감, 다이나믹한 게임플레이로 최대한 많은 연령대가 즐길 수 있는 온라인 대전 슈팅게임을 구현하였다. 우리가 졸업작품을 구현하면서 맞닥뜨렸던 다양한 문제와 해결하기 위해 했던 노력들을 이렇게 기록으로 남겨서 앞으로 프로젝트를 진행할 때 도움이 될 수 있게 하고자 한다.

**1. 서론**

처음 기획할 때부터 어떤 게임을 만들어야 할지 고민했던 우리 팀은 진입장벽이 낮고, 남녀노소 누구나 재미있게 즐길 수 있는 캐쥬얼한 게임을 만들고자 하는 공통의 생각을 가지고 있었다. 특히 이러한 게임에 동화 풍의 모델링과 귀여운 캐릭터가 추가되면, 저 연령층 및 여성들에게도 인기가 많기 때문에 다양한 연령층에 대한 시장성을 확보에 유리하다. 따라서 우리는 이런 점들을 충족시키는 게임을 기획, 구현하였다.

매직 큐브의 플랫폼은 PC이고 장르는 온라인 대전슈팅게임으로 최대 5명의 플레이어가 다양한 스킬들을 사용하여 개인전투를 하여 점수를 획득, 1등이 되는 것을 목표로 하는 게임이다.

이 게임의 특징은 다음과 같다.

1) 최대 5인 플레이가 가능한 온라인 게임이다.

2) 10개의 캐릭터와 6개의 스킬이 있다.

3) 이 게임은 독특한 규칙으로 플레이어에게 재미를 준다. 규칙은 다음과 같다.

가) 이 게임은 3분의 제한 시간이 있다.

나) 제한시간 후 가장 많은 점수를 얻은 플

레이어가 승리한다.

다) 각 플레이어를 처치 시 10점을 얻는다.

라) 2층에는 중립 몬스터가 존재하고, 처치 시 100점을 얻는다.

마) 1등 플레이어는 공격력, 체력 버프를 받고 크기가 커진다.

4) 물리 효과를 구현하여 플레이어와 물리 효과가 적용된 오브젝트 간의 상호작용을 할 수 있다.

결과적으로 이러한 게임 특징들을 이용한 플레이로 우리는 여태까지의 게임처럼 팀 게임 이 아닌 철저한 개인의 전략을 사용하고 점수 시스템을 도입함으로써 플레이어의 컨트롤 뿐만이 아닌 눈치싸움을 유도하는 게임의 의도를 적용하였다.

**2. 개발 내용**

본 게임의 클라이언트는 3차원 그래픽 응용 프로그램을 다양한 Windows 플랫폼(데스크톱, 모바일, Xbox)에서 작성하기 위한 low-level 렌더링 라이브러리인 DirectX 12로 개발하였다. 서버는 C++언어와 호환성이 좋고 Windows 플랫폼에서 제공하는 고성능 및 대용량 서버 구축에 효율적인 Overlapped IO모델인 Boost/asio를 이용하여 제작 및 클라이언트와의 동기화를 진행하였다.

**2.1 클라이언트 개발 내용**

**2.1.1 클래스의 생성 및 정보**

클라이언트에서는 각 객체들을 수월하게 관리할 수 있도록, STL라이브러리의 vector와 list를 적극 활용하였다. 클래스는 크게 FrameWork, mainFrameWork, Scene, Shader, GameObject, MMPE, Light, Player등의 클래스로 이루어져 있다. 각 설명은 다음과 같다.

1) FrameWork : DirectX에서 제공되는 여러 Com객체들을 생성 및 초기화를 하고, 윈도우 생성을 한다.

2) mainFrameWork : 가장 게임 진행에 핵심이 되는 클래스로, Scene클래스를 생성하고, FrameAdvance함수를 통해서 게임의 진행을 담당한다.

FrameAdvance에서는 먼저 커맨드 알로케이터랑 커멘드 리스트를 초기화 하고, 씨저렉트랑 뷰포트를 생성한다. 전면버퍼를 리소스베리어를 통해 상태를 전이 시키고, 후면버퍼로 바꾼다. 이후 후면버퍼를 초기화하고, 업데이트 함수를 호출 후 렌더함수를 호출해 렌더링을 한다.

3) Scene : 렌더링 되야 하는 게임오브젝트와 광원과 쉐이더 객체들을 생성하고 제거한다. 또한 매 프레임마다 Tick함수를 호출하여, 제거되야 할 게임오브젝트를 제거하고, 모든 게임오브젝트의 Tick함수를 호출한다. 또한 렌더링 함수를 가지고 있어서, 루트시그니처랑 쉐이더객체를 Set하고, UI렌더를 호출한다.

4) Shader : Scene클래스의 하위 멤버객체로 들어가며, 모든 PSO를 생성하고 연결한다. Scene클래스가 가진 게임오브젝트를 공유받아서, 게임종류별로 PSO를 연결해서 렌더링 하는 역할을 맡는다.

5) GameObject : 각 오브젝트를 생성하는 클래스이다. 생성자에서는 오브젝트마다 게임 플레이에 필요한 정보들(체력, 공격력 등)을 초기화하고 텍스쳐를 불러온다. 리지드바디 오브젝트에서는 이뿐만 아니라 물리효과에 필요한 정보들 까지 초기화한다.

**2.1.2 MD5 포멧을 활용한 모델출력 및 애니메이션 재생**

우리는 3dMax에서 제작한 3D모델을 블랜더 프로그램을 이용해서 MD5로 추출한 파일을 게임 내 리소스 데이터로 활용하였다. 생성된 모델을 렌더링하고, 애니메이션을 재생할 수 있도록 간단한 함수들을 추가하고, 클래스 등을 추가하였다. MD5내부에 있는 weight와, UV, 쿼터니언 등을 이용해 정점을 계산하고, 계산된 결과를 Mesh클래스의 Vertex리스트에 저장하였다. 이후 Joint 클래스에 애니메이션 정보를 담아서, hlsl에서 애니메이션을 사용하는 객체라면, 애니메이션 보간을 이용해, 정점들의 위치를 갱신을 하여 자연스러운 애니메이션을 재생하였다.

**2.1.3 물리효과 직접구현**

리지드바디, 질점 같은 물리 효과를 직접 구현하여 게임 내에서 좀더 다이나믹하고, 전략적인 게임플레이를 도모하였다. 물리효과의 세부적인 설명은 다음과 같다.

1) GenetatorGravity : 리지드바디와 질점 객체에 중력을 매 프레임마다 가하도록 인터페이스를 제공한다.

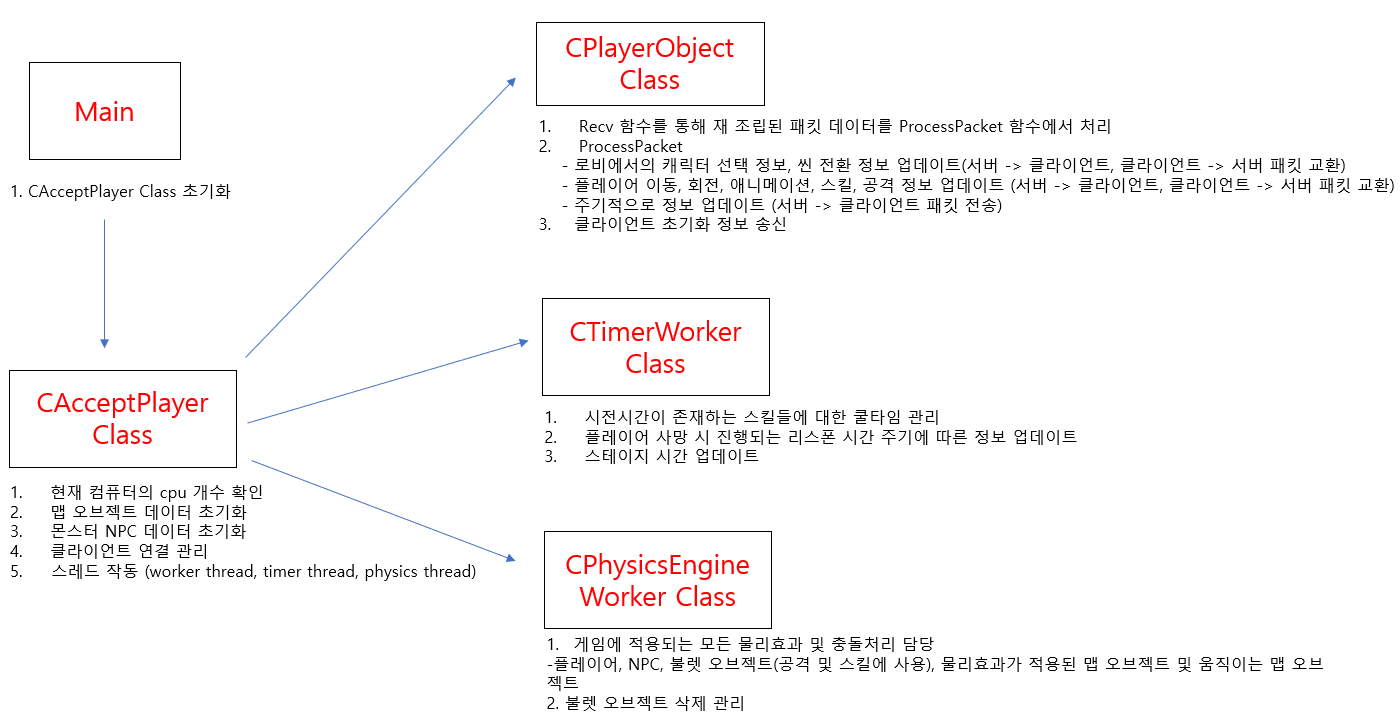
2) RigidBody : 속도/ 가속도/중점/방향/질량/토크/힘 등 리지드바디의 정보들을 담고 있으며, 적분함수를 통해 위치와 방향정보를 갱신한다.

3) PhysicsPoint : 리지드바디와 비슷하지만 방향의 변화를 처리하지 않으므로 각속도 등을 변경하지 않는다. 플레이어 객체나, 탄환, 고정된 객체에 적용된다.

4) AfterGravitySystem : 중력에 대한 후 처리를 한다. 물체가 땅에 파고들거나, 리지드바디 객체가 바닥에 부딪혀서 튕기거나 하는 것을 처리한다.

5) CollisionSystem : 물체끼리의 충돌을 처리한다. 충돌 후 속도와 겹쳤을 때 처리 등이 일어난다.

**2.2 서버 개발 내용**

**2.2.1 클래스 생성 및 정보**

메인 스레드에서 AcceptPlayer 클래스를 호출한다. 이 클래스에서는 맵, 맵 오브젝트, NPC 몬스터 정보 등을 초기화하는 부분을 거친다.

다음에 이 프로그램에서 IOCP의 worker thread 역할을 하는 io\_service::run 객체를 호출하는 부분, 물리엔진 및 충돌 처리가 작동되는 부분, 타이머 관련 작업을 하는 부분이 스레드를 할당 받아 작동된다.

이 때, 클라이언트로부터 접속이 완료되면 Accept 함수 부분의 async\_accept의 handler가 호출되어 이 부분에서 클라이언트 정보를 담은 객체를 생성하고 아이디를 할당하여 서버에서 관리할 수 있도록 한다.

클라이언트와 접속이 완료되었다면 Recv함수를 호출하여 클라이언트로부터 받은 데이터 패킷을 재조립하여 이를 사용하여 ProcessPack

et함수를 통해 패킷 처리를 진행한다.

또한 물리엔진 및 충돌 처리 작업을 담당하는 스레드와 플레이어 리스폰 타임, 스킬 쿨타임, 스테이지타임 업데이트 작업을 담당하는 스레드가 병렬적으로 수행된다.

**2.2.2 서버 구조 및 특징**

C++과 호환성이 좋고 Windows에서 제공하는 IO 중 고성능 및 대용량 서버를 구축하는데 사용되는 Overlapped IO를 지원하는 Boost asio를 이용하여 MMOG 서버를 구축하였다. 대부분의 게임에서 채택하고 있는 서버-클라이언트 모델이며 async IO를 이용하여 IO작업을 하는 동시에 다른 작업 또한 할 수 있도록 병렬성을 증대 시켰으며 main 스레드를 제외한 8개의 스레드를 사용하여 6개의 스레드는 패킷 송수신 및 패킷 처리 작업을 하는데 사용하고 1개의 스레드를 사용하여 물리엔진 및 충돌 처리 작업에 사용하고 나머지 1개의 스레드는 시간의 변화에 따라 업데이트되어야 하는 작업 처리에 사용하였다. 게임 상에서 일어나는 모든 충돌 처리 및 물리엔진을 서버에서 구동 시키는 방식을 채택하여 클라이언트에 성능 상의 부담을 덜 준다.

**2.2.3 서버의 효율을 높이기 위한 작업**

서버의 효율 및 대용량 서버를 구축하기 위해 main 스레드를 제외한 총 8개의 스레드를 이용하여 멀티 스레드 기반의 서버를 구축하였다. 패킷 송수신에 있어 과부화를 줄이기 위해 패킷 구조체를 이루는 데이터들의 타입을 char, short 형으로 구성하여 패킷 사이즈를 압축했고 주기적으로 업데이트 되어야하는 캐릭터 상태, 위치, 공격 스킬 정보, 변화되는 맵 오브젝트 데이터 등을 통합하여 패킷을 송수신함으로써 패킷 송수신 빈도수를 최대한 줄이는 방향으로 구현하였다. STL에서 지원하는 컨테이너들 중 상대적으로 가장 탐색, 삽입, 삭제 성능이 좋은 비 정렬 연관컨테이너인 unordered\_

map 을 사용하여 수가 많은 맵 오브젝트들을 관리하여 서버의 효율을 높였다.

**2.2.4 개발 중 발생한 멀티 스레드 프로그래밍의 문제점과 해결법**

공격 스킬 중 해머불렛(이하 불렛)이라는 스킬이 있는데 이 스킬을 사용했을 때 상대방 플레이어와 충돌하여 사라져야 될 불렛이 사라지지 않고 여전히 남아있는 버그가 발생했었다. 이 스킬의 전반적인 관리는 서버에서 이루어진다. 다시 말해, 불렛 생성 시 서버에서 id가 부여되며 해당 id를 이용하여 어떤 불렛과 상대방이 맞았는지 유무를 판단한다. 이 스킬은 특성상 상대 플레이어와 맞으면 그 주변에 또 다른 불렛이 생성된다. 불렛이 처음 생성될 때에는 클라이언트로부터 해당 스킬 키를 눌렀다는 정보를 서버의 패킷 송수신처리를 담당하는 worker\_thread에서 id를 부여하여 생성하고 상대방이 맞았을 때 생성될 때는 물리엔진 및 충돌처리를 담당하는 physics\_thread에서 id가 부여되어 생성된다. 불렛에 대한 id를 증가시키는 부분이 서로 다른 스레드에서 중첩적으로 발생할 가능성이 있어 위와 같은 현상이 발생되는 것이라 생각하고 불렛 id가 중첩되어 출력되는 지 확인해보았다. 그 결과 id가 중첩되는 현상이 발생하였고 이 부분이 스레드 간 동기화가 제대로 이뤄지지 않아 발생하는 문제라 생각했고 mutex lock을 사용하여 스레드 간 공유자원에 대한 동기화를 진행하였다. 그 결과 해당 버그를 해결할 수 있었다.

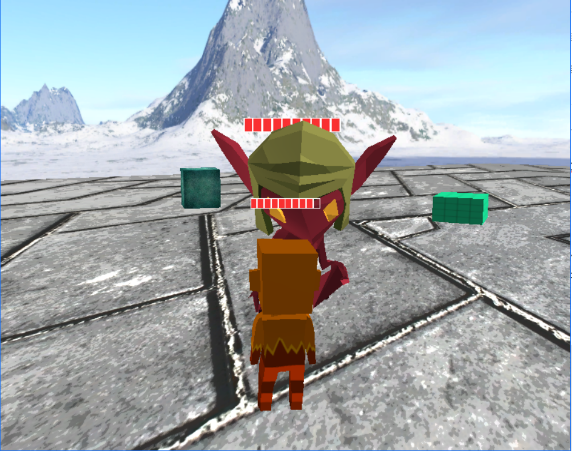
**3. 결론**

**3.1 게임 스크린샷**

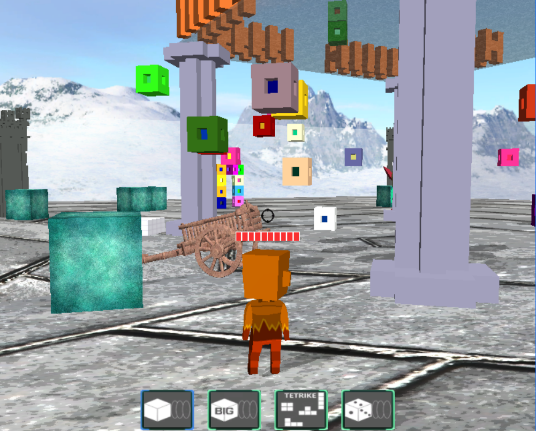
  
[사진1] 맵, 오브젝트 및 캐릭터의 모습



[사진2] 파티클 효과 및 물리 효과



[사진3] 중립 보스 몬스터



[사진4] 스킬UI및 오브젝트



[사진5] 조명 및 셰이더 적용

3.2 장단점

매직큐브의 장점은 첫번째로 서론에서 언급한 것처럼 네모나고 귀여운 캐릭터와 쉬운 조작감, 동화 풍의 그래픽이다. 따라서 넓은 연령층에서 게임을 즐길 수 있다. 또한 팀전이 아닌 개인전에 점수 시스템을 도입해서 1등 플레이어에게 버프를 부여하고 1등 플레이어를 처치 시에 많은 점수를 받을 수 있기 때문에 좀더 다이나믹한 플레이가 가능하다. 마지막으로 물리 효과를 구현하여 리지드바디 오브젝트와 플레이어의 상호작용을 이용해 다양한 플레이가 가능하다.

단점으로는 먼저 단조로운 모델과 UI가 있다. 우리 팀에 그래픽 담당이 없었기 때문에 모델은 인터넷에서 로우 폴리곤으로 이루어진 무료 모델을 사용하였다. 또한 UI도 포토샵 프로그램을 사용해서 직접 만들었다. 플레이어의 관점에서 게임을 할 때 먼저 눈에 보이는 것은 그래픽이기 때문에 우리의 게임은 단조로워 보일 수 있다. 두번째 단점은 쉐이더 부분이다. 특히 그림자 부분에서 셀프 섀도우가 적용이 되지 않고 있기 때문에 그림자가 부자연스럽게 보인다.

**4. 부록**

**4.1 게임 실행 환경**

OS : Windows7이상

CPU : Intel core i5-5200U이상

GPU : D3D12지원 GPU

RAM : 4GB이상

**4.2 게임 설치**

1) MagicCube 폴더를 클릭해서 연다.

2) MagicCube 폴더 안에 MaigcCubeServer 폴더를 클릭하여 연 후 Server\_BoostAsio.ex

e 실행파일을 실행시킨다.

3) MagicCube 폴더 안에 MagicCubeClient 폴더를 클릭하여 연 후 Init Direct3D.exe 실행파일을 클릭하여 실행시킨다.

4) 이하 4.3항목의 게임 진행을 읽어보고 게임을 플레이하면 된다.

**4.3 게임 진행**

게임을 시작하면 MagicCube 타이틀이 보여지며 자동으로 캐릭터 선택 창으로 넘어간다. 캐릭터 선택 창에서 플레이어는 고유한 4가지 스킬을 가진 총 10가지의 캐릭터 중 하나를 키보드 숫자 키 0 ~ 9을 눌러서 선택할 수 있다. 자신과 함께 플레이할 다른 플레이어들이 모두 게임시작버튼인 SPACE키를 누르면 메인 스테이지로 씬이 전환된다. 메인 스테이지는 최대 5인이 플레이 할 수 있도록 오각형모양으로 구현 되어있으며 플레이어들이 생성되는 1층과 NPC 몬스터인 임프가 존재하는 2층으로 구성되어 있다. 스테이지 플레이 타임인 2분 30초 동안 플레이어는 4가지 스킬을 적절히 사용하여 상대방의 체력을 0으로 만들어 죽이거나 2층에 존재하는 NPC 몬스터를 처치하여 점수를 획득할 수 있으며 상대방 플레이어에게 죽거나 NPC 몬스터에게 죽으면 초기 캐릭터 생성위치에서 5초 후 리스폰된다. 획득한 점수가 30점이 넘고 킬 수와 데스 수 및 획득한 점수가 가장 많은 플레이어가 1등 플레이어가 되며 캐릭터 사이즈가 커지고 체력 및 스피드가 증가하는 버프 효과를 받게 된다. 제한 시간이 모두 지났을 때 1등한 캐릭터가 해당 스테이지에서 승리하게 되며 다른 플레이어들은 패배하게 된다.

**4.4 조작법**

W A S D : 이동

마우스 클릭 : 공격 / 스킬사용

1 2 3 4 : 스킬 교체

Spacebar : 점프

**4.5 연락처**

최규화 :

010-2557-6891

gusdl576@naver.com

이관구 :

010-9984-4779

[dufdif@gmail.com](mailto:dufdif@gmail.com),

유창민 :

010-3538-8395

min940913@naver.com