오브젝트 풀링을 이용한 FPS 디펜스 게임 개발

임원규[O], 안성옥[*], 김수균[*]
[O*]배재대학교 게임공학과
email: kimsk@pcu.ac.kr[*]

Development of FPS Defense Game Using Object Pooling

Wongyu Lim[O], Syoungog An[*], Soo Kyun Kim[*]
[O*]Dept of Game engineering Paichai University

● 요 약 ●

게임엔진을 이용한 FPS 디펜스 게임은 유니티3D 엔진을 사용하여 개발 하였으며 1인칭 시점으로 제한시간동안 몰려오는 적군을 막아내며 목표물을 지키는 게임이다. 많은 오브젝트를 관리하기 위해서 오브젝트 풀링을 사용하여 오브젝트가 생성-제거의 반복시 메모리에 부담을 주게되는 것을 씬 시작시 가용할 오브젝트를 불러온 뒤에 필요시에만 사용 하는 방법으로 메모리의 부담을 적게 하였고 플레이 기록을 랭킹으로 하여 사용자 간에 경쟁심을 유발 할 수 있도록 하였다.

키워드: Object Pooling, FPS, Game Engine

Ⅰ. 서론

1인칭 디펜스 게임은 1인칭의 시점으로 몰려오는 적군을 막아내는 게임이다. 이 게임의 규칙은 제한 시간동안 정해진 자원을 가지고 더 많은 적군을 물리치며 정해진 목표물을 방어하는 것이 이 게임의 목표이다. 자원이 한정되어있고 자원을 효과적으로 활용해 몰려오는 적을 물리쳐야하기 때문에 플레이어에게 좀 더 전략적이고 난이도 있는 플레이를 요구할 수 있다. 플랫폼을 PC로 계획하게 된 이유는 Fig. 1과 같이 PC게임의 대다수가 1인칭 게임으로 차지하고 있으며 모바일에는 FPS장르가 조작에 큰 어려움과 수많은 UI로 게임 몰입에 방해하기 때문에 PC를 기반으로 개발하게 되었다.

Fig. 1. Concurrent Users

Ⅱ. 본론

1. 게임 구현

게임은 메인화면과 게임화면이 있으며 메인화면에서는 우주선이 우주공간을 날아다니는 연출을 주었으며 게임시작 버튼을 누를시 바로 게임을 플레이 할 수 있다. 게임 시작 시 로딩화면을 통해 게임씬을 불러오도록 하였으며 로딩간 우주선이 불시착 하는 연출을 주도록 하였다. 로딩이 완료 되면 각 위치에서 적군이 목표를 향해 몰려오며 플레이어는 체력이 0이 되지 않도록 제한 시간동안 적군으로부터 목표를 지켜야 한다. 만약 제한 시간 이전에 플레이어의 체력이 0이 되거나 목표의 체력이 0이 되면 게임에서 패배하게 되고 메인메뉴로 돌아가며 제한시간동안 목표 달성 시 승리하게 되며 점수를 랭킹에 기록하고 메인메뉴로 돌아가도록 한다. 랭킹은 PlayerPrefs[1]를 사용해 파일로 저장하였고 게임 종료 후 다시 실행하여도 기록을 유지할 수 있도록 하였다.

2. 오브젝트 풀링

게임 오브젝트를 생성-제거를 주기적으로 할 시에 할당 오브젝트가 많을 경우 메모리에 큰 부담을 주게 된다. 그렇기 때문에 Fig.2처럼 오브젝트 풀링(ObjectPooling)을 사용하여 게임 씬을 시작 할 때 가용할 오브젝트를 미리 생성하고 필요할 때 마다 오브젝트를 꺼내서 사용한다. 만약 더 필요시엔 오브젝트를 추가로 생성하고 비사용시에는 비활성 한 뒤 오브젝트 풀에 넣어 메모리 관리에 좀 더 효율적으로 사용하였다.

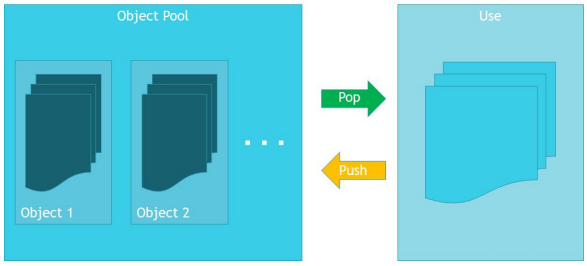

Fig. 2. Object Pooling

3. 싱글턴 패턴

싱글턴 패턴은 스크립트를 최초 한번만 인스턴스를 할당하고 재사용시 이 인스턴스[2]를 사용하는 디자인 패턴이다. 이 패턴을 게임에도 적용시켜 게임씬에 1개만 존재하는 오브젝트에 적용시켰으며 최초에 생성한 인스턴스를 재생성 하지 않고 재사용함으로서 메모리의 낭비를 방지할수 있다. 또한 static 변수로 전역에서 사용하기 때문에 다른 오브젝트에서도 데이터를 쉽게 참조 할 수 있다(Fig. 3).

```
private static Object instance;
public static Object objInstance{
    get
    {
        if(instance==null)
        instance = this;
    }
    return instance;
}
```

Fig. 3. Singleton Pattern

4. 로딩화면

게임 씬을 불러올 때 한 씬에 많은 양의 데이터가 있다면 씬을 불러오는 동안 사용자는 씬이 멈춰있는 화면을 하염없이 바라볼 뿐이다. 이를 보완하기 위해 Fig.4와 같이 LoadAsync를 쓰레드로 사용하여 씬을 불러온 후에 씬 로딩이 완료되면 해당 씬으로 교체한다[3]. 이를 통해 사용자가 로딩이 끝난다면 게임이 시작된다는 것을 직관적으로 알 수 있고 로딩 진행 현황을 표시해 로딩이 얼마나 진행 되었는지 알 수 있도록 하였다.

```
IEnumerator LoadScene{
  yield return null;
  AsyncOperation oper = LoadSceneAsync();
  oper.allowSceneActivation = false;
  while(!oper.isDone){
    if(oper.progress==1.0f)
      oper.allowSceneActivation = true;
  }
}
```

Fig. 4. LoadAsync

5. 랭킹

게임을 단순히 게임을 플레이만 하는데 목표를 두면 유저들이 금방 지루함을 느낄 수 있다. 때문에 플레이어들의 경쟁심을 유발할 수 있도록 랭킹 시스템을 넣어서 플레이어들의 기록을 통해 순위를 매겨서 게임의 플레이타임을 좀 더 늘릴 수 있도록 하였다. 기록된 랭킹은 프로그램이 종료 되더라도 랭킹 기록이 지워지지 않고 다시 불러 올수 있도록 PlayerPrefs를 사용해 기록을 저장하고 불러오도록 하였다.

Ⅲ. 결론

디펜스 게임을 1인칭으로 플레이 함으로서 적군을 좀 더 살감나게 물리치는 효과를 얻을 수 있고 다양한 몬스터를 상대로 다양한 전투를 플레이 할 수 있다. 또한 랭킹을 통해 사용자간 경쟁심을 유발하여 유저 vs 유저간 전투로 방향성을 노릴 수도 있다. 그리고 게임의 최적된 플레이를 위해 오브젝트 풀링,싱글턴 패턴,로딩화면을 사용하여 가용되는 메모리에 부담을 적게 하도록 하였다.

REFERENCES

[1] https://docs.unity3d.com/ScriptReference/index.html
[2] http://www.devkorea.co.kr/
[3] https://store.steampowered.com/stats