Journal of The Korea Society of Computer and Information
Vol. 23 No. 8, pp. 31-36, August 2018

www.ksci.re.kr
https://doi.org/10.9708/jksci.2018.23.08.031

Design of 2.5D Survival Game using Inventory

Soo Kyun Kim*, Hong-Rae Kim*, Won Joo Lee**

Abstract

The survival game is characterized by the ability to survive until the item is collected and the game is completed at the specified time, and the inventory function to store the item is the core of the game. Typical survival games include 'Don't Starve', 'H1Z1', and 'Lust'. The purpose of this paper is to design a 2.5D survival game that can be enjoyed by the smart device using Unity 3D engine. Because it is designed as a mobile platform, designing light inventory function using two lists rather than existing inventory function makes it easier to design than existing inventory and light design suitable for mobile. In general, it is characterized by designing a mobile game so that it does not depend on the place of the survival game.

▸Keyword: List, Inventory, 2.5D Game, Unity3D, Joystick

I. Introduction

2017년 모바일 게임 시장[1]은 전년대비 24.3% 성장하였고, 모바일게임을 즐기는 인구는 약 수억 명에 달한다. 이미 많은 게임 사용자를 가지고 있는 모바일 게임 시장에는 많은 모바일용 게임이 시장에 선보여지고 있다. 모바일 플랫폼의 장점은 PC보다 적은 비용, 적은 시간으로 제작 할 수 있고, 휴대성이 좋기 때문에 장소에 구애를 적게 받고 게임을 즐길 수 있다는 점이다. 서바이벌 게임의 장점을 몇 가지의 게임으로 예를 들면 '돈스타브(Don't Starve)[2]'라는 게임의 경우에는 '굶지 마'라는 명확하면서도 단순한 게임목표로 플레이어들을 집중시킨다. H1Z1[3]라는 게임은 '배틀 로얄'이라는 100여명의 플레이어가 한 서버에 동시에 접속하여 그 중 상위권 플레이어가 보상을 획득하는 방식을 도입하여 플레이어의 경쟁을 유도한다.

본 논문은 모바일 플랫폼을 이용한 2.5D 서바이벌 게임 개발을 목표로 하며, 유니티3D 엔진[4-7]과 NGUI[8-10]를 이용하여 모바일 서바이벌 게임을 설계하는 방법에 대해 소개한다. 본론에서는 리스트를 이용한 인벤토리 구현과 모바일에 적합한 조이스틱[11] 설계, 게임 디자인 등에 대해 구체적으로 설명한다.

II. System Overview

본 절은 게임의 구성에 대해 설명한다. Fig. 1은 게임 구성을 표현한 게임 구성도이며, 메인 화면에서 게임시작, 옵션, 도움말 및 게임 종료로 구성된다.

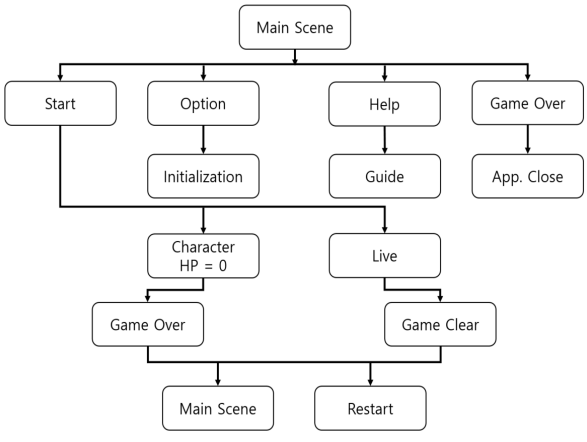

Fig. 1. Flowchart

• First Author: Soo Kyun Kim, Corresponding Author: Won Joo Lee
*Soo Kyun Kim (kimsk@pcu.ac.kr), Dept. of Game Engineering, Pai Chai University
*Hong-Rae Kim (minlog@empal.com), Dept. of Game Engineering, Pai Chai University
**Won Joo Lee (wonjoo2@inhatc.ac.kr), Department of Computer Science, Inha Technical College
• Received: 2018. 07. 26, Revised: 2018. 08. 05, Accepted: 2018. 08. 07.
• This work was supported by the research grant of Pai Chai University in 2018.

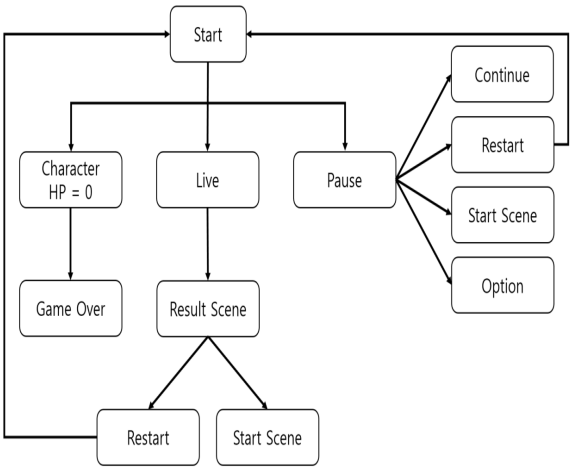

Fig. 2. Play Scene

Fig. 2에서는 게임시작 버튼을 눌러 게임이 시작되면 캐릭터의 HP가 0이 되어 패배하거나 목표 시간까지 생존하여 승리가 결정되면 결과 화면이 뜨게 되고 게임을 다시 시작하거나 메인 화면으로 돌아가도록 구성한다.

III. Game Design

본 절은 모바일용 서바이벌 게임 설계를 위해 모바일에 적합한 UI(조이스틱, 버튼, 인벤토리 등)[12]를 설계하는 방법에 대해 소개한다. 특히 리스트를 이용한 인벤토리[13] 제작 방법과 방향성 벡터를 사용한 조이스 구현 및 NGUI(Next-Generation UI)의 스크롤 뷰를 사용한 조합 창 등을 설계하는 방법을 설명한다.

3.1. Implement inventory using a list

서바이벌 장르의 게임은 재료를 습득하고 조합하여 생존을 해야 하는 게임이기 때문에 아이템을 보관하고 삭제하는 기능인 인벤토리를 구현하는 것은 필수적이다.

본 설계에서는 2개의 리스트를 사용하여 가벼운 인벤토리 기능을 구현하고, 인벤토리를 표현하는 이미지는 NGUI 기능을 사용한다. Fig. 3은 인벤토리를 구현하는데 쓰이는 2가지의 리스트와 획득한 아이템을 인벤토리 창과 리스트에 추가해 주는 슈도코드 보여준다. 모든 아이템을 관리하는 리스트는 게임 내에 존재하는 모든 아이템들을 리스트에 추가하여 관리해 주는 기능을 하고 플레이어가 습득한 아이템 리스트에서는 플레이어가 습득한 아이템의 추가와 삭제, 조합 기능을 담당한다.

```
public List<ItemScript>
 모든 아이템을 관리하는 리스트 =
    new List<ItemScript> ();
    public List<string>
  플레이어가 습득한 아이템 리스트 =
 new List<string> ();
public void SetInfo (string 이미지 이름)
{
  인벤토리에 추가될 이미지이름 = 이미지 이름;
  아이템 이름 = 이미지 이름;
}
private void 인벤토리에 아이템 추가()
{
  GameObject 인벤토리에 추가될 오브젝트 =
  NGUITools.AddChild(부모객체, 자식객체);

  아이템 이미지 스크립트 itemScript=
  gObjItem.GetComponent
  <아이템 이미지 스크립트>();
 for(int i=0; i<게임에 존재하는 아이템 개수;i++){
   if(게임에 존재하는 아이템 [i]=="아이템 이름")
   itemScript.SetInfo(게임에 존재하는 아이템[i];
  }
   플레이어가 가지고 있는
   아이템 정보.추가(itemScript);
}
```

Fig. 3. Inventory Pseudocode

Fig. 4의 인벤토리 이미지 부분을 보게 되면 제거(Delete)라는 버튼이 있는데, 이 버튼이 아이템 삭제를 담당한다. 제거 버튼을 클릭 후 삭제를 원하는 아이템을 터치하게 되면 삭제된다. 삭제 기능의 경우는 삭제 버튼이 눌렸을 때와 안 눌렸을 경우를 나타내는 조건을 만들고, 눌렸을 경우 Fig. 3에서 만들었던 플레이어가 습득한 아이템 리스트에서 해당하는 리스트 정보를 삭제해주고 인벤토리에 추가된 오브젝트도 삭제 해준다.

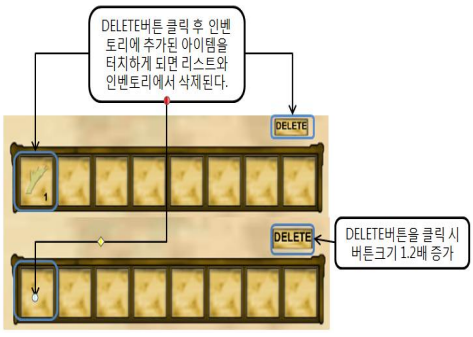

Fig. 4. Delete Function

3.2. Combination Function

인벤토리가 구현되어 플레이어가 재료를 획득 할 수 있게 되면 획득한 재료를 이용하여 게임에 필요한 아이템을 조합 할 수 있는 조합 기능을 구현해야 한다. 플레이어 화면에 표시될 조합 창은 NGUI의 스크롤 뷰(ScrollView)를 사용하여 Fig. 5와 같이 터치 드래그를 사용하여 밑에 숨겨진 조합 목록을 확인 할 수 있도록 설계한다.

Fig. 6은 플레이어 화면에 표시될 조합 창을 구현한 모습이다. 조합 기능을 구현하는 방법은 Fig. 7의 슈도코드와 같이, 생성(Create) 버튼을 눌렀을 경우 조합에 필요한 아이템 개수가 만족한다면 원래 가지고 있던 아이템 개수에서 조합에 필요한 아이템

개수를 빼주고, 이후로는 3.1의 아이템을 추가해 주는 부분과 동일하게 인벤토리 창에 조합 아이템 이미지를 추가해주고 플레이어가 가지고 있는 아이템 리스트에 조합 아이템 정보를 저장하게 된다.

Fig. 5. Scroll View using Combination Window

Fig. 6. Combination Window

```
void 조합할 아이템 ()
{
      if (조합에 필요한 아이템 개수) {
         가지고 있던 아이템 개수 -=
            조합에 필요한 아이템 개수;

GameObject 인벤토리에 추가될 오브젝트 =
  NGUITools.AddChild(부모객체, 자식객체);

   아이템 이미지 스크립트 itemScript=
              gObjItem.GetComponent

<아이템 이미지 스크립트>();
 for(int i=0; i<게임에 존재하는 아이템 개수;i++){
 if(게임에 존재하는 아이템 [i]=="아이템 이름")
   itemScript.SetInfo(게임에 존재하는 아이템[i];
  }
플레이어가 가지고 있는
아이템 정보.추가(itemScript);
}
```

Fig. 7. Pseudocode of Combination Window

조합 기능이 추가되어 인벤토리에 조합 아이템을 추가 하는 것이 가능해지면 조합 아이템을 Fig. 8처럼 터치 드래그 하여 게임 상의 임의의 좌표에 설치 할 수 있도록 기능을 추가해야 한다.

Fig. 8. Touch Item and Install

설치 기능 같은 경우에는 조합 아이템을 드래그 했을 경우 조합 아이템이 터치한 좌표로 계속 따라 다녀야 한다. 그 부분은 Fig. 9에서 조합 아이템이 드래그 될 경우 NGUI카메라의 스크린 좌표를 기준으로 터치 좌표를 받아와서 드래그 된 조합 아이템의 위치 값에 터치 좌표를 대입 해주어 화면에 터치 드래그 하는 위치대로 표시해 주게 된다. 드래그가 끝났을 경우에는 게임 상에 조합아이템을 생성시켜 주고 생성될 위치 값은 플레이어의 좌표 값을 이용하게 된다.

```
void 아이템이 드래그 될 경우(Vector2 delta)
{
      if (드래그 된 아이템이 조합 아이템일 경우)
   {
       Ray ray = NGUI카메라의 스크린 좌표를 기준
                 (터치좌표);
        Vector3 currentPos = NGUI 화면상의
                             터치좌표 (0);

      조합 아이템의 드래그 된 위치 값 =
     new Vector3 (NGUI화면상의 터치좌표.x,
        NGUI화면상의 터치좌표.y, 0);
  }

void 드래그가 끝났을 경우()
{
       if (조합 아이템일 경우) {
         조합 아이템 위치 값 = new Vector3
       (캐릭터 위치 값.x, 캐릭터 위치 값.y,
        캐릭터 위치 값.z);

   플레이어가 가지고 있는 아이템 리스트 . 삭제
   (조합 아이템);

오브젝트 삭제 (드래그 된 조합 아이템);

    }
}
```

Fig. 9. Pseudocode of Combination Window

마지막으로 설치가 완료된 조합 아이템을 인벤토리에서 지워주게 되고 플레이어가 가지고 있는 아이템 리스트 에서도 삭제하게 된다.

3.3. Design of Joystick using Directional Vector

Fig. 10과 같이, 게임에서는 3D 공간에서의 캐릭터 움직임을 방향성 벡터를 이용한 조이스틱으로 구현한다. Fig. 11은 조이스틱이 눌렸을 경우 방향성 벡터를 구하는 방법을 나타내 주는 슈도코드이다. 본 논문에서 방향성 벡터를 구하는 방법으로 움직인 후의 조이스틱 센터의 위치 값 x와 y값에서 초기 조이스틱 센터의 x와 y위치 값을 빼주는 방법으로 방향성 벡터를 측정한다. 움직일 타겟이 있을 경우 구해진 방향성 벡터, 타겟의 이동속력, Time.deltaTime을 곱한 값을 움직일 캐릭터의 위치 값에 더해주는 방식으로 캐릭터 움직임을 구현한다.

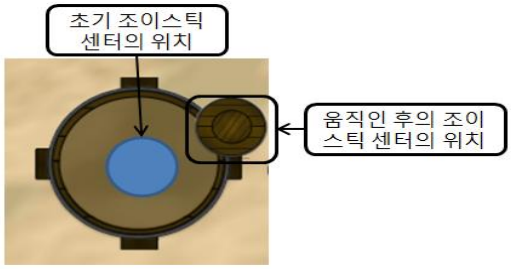

Fig. 10. Joystick

```
if(조이스틱이 눌렸을 경우) {
  방향성 벡터=(new Vector3(움직인 후 조이스틱 센터의
x위치 값-초기 조이스틱 센터의 x위치 값,0,
움직인 후 조이스틱 센터의 y위치 값-초기 조이스틱 센터
의 y위치 값)).nomalized;

if(움직일 타겟이 있을 경우){
      움직일      타겟의      좌표      +=방향성      벡터
       *Time.deltaTime*타겟의 이동속력;
   }
}
```

Fig. 11. Pseudocode of Vector

IV. Experimental Results

본 논문에서 설계한 게임은 유니티3D 5.3.4f1[14] 버전으로 플랫폼은 안드로이드에서 개발하였다. 게임 디자인은 3D Max 2016, 포토샵 CC를 사용하여 제작하였고, 프로그램 소스코드는 C#[15]을 이용하였다. Fig. 12는 시작화면을 나타내며, 게임의 제목과 시작버튼, 도움말, 옵션 및 나가기 버튼으로 구성되어 있다.

Fig. 12. Start Scene

Fig. 12는 게임시작 버튼을 누른 직후의 화면으로 게임의 스토리가 뜨고 탈출시작이라는 버튼을 누르게 되면 설정된 시간이 흐르면서 게임이 시작된다.

Fig. 13. Game Play Scene

Fig. 13은 아이템 획득후의 인벤토리와 조합 창을 터치 했을 경우의 장면이다. 아이템을 획득할 경우 Fig. 14의 인벤토리처럼 아이템 이미지와 개수가 표시되고 왼쪽의 조합 창을 터치했을 경우에는 조합에 필요한 재료의 개수와 생성 버튼이 나타나게 된다.

Fig. 14. Inventory and Combination Window

Fig. 15는 조합한 아이템을 드래그 하여 게임 상에 설치하는 장면이다. 아이템을 조합할 경우 인벤토리에 조합한 아이템이 추가되고 추가된 아이템을 게임 상에 드래그 할 경우 드래그한 위치로 아이템이 설치된다.

Fig. 15. Combination Item

Fig. 16은 생존에 필요한 아이템을 수집하는 장면이다. 돌이나 나무 같은 재료를 수집하기 위해서는 조합목록에 있는 도끼나 곡괭이를 조합해야하고 인벤토리에 해당 아이템이 존재한다면 Fig. 16과 같이 재료를 수집하는 것이 가능하다.

Fig. 16. Correct items

Fig. 17은 본 논문에서 설계한 안드로이드 기반 생존게임을 스마트 폰에서 실행시킨 화면이다. 게임에 추가한 인벤토리 기능 및 조합기능, 조이스틱, 버튼 등 모든 기능이 정상 작동하는 것을 확인하였다.

Fig. 17. Execution on a Mobile

V. Conclusions

서바이벌 게임의 특징은 아이템(재료)을 모아 정해진 시간이나 목표를 완수 할 때까지 생존 하여야 하므로 아이템을 저장할 수 있는 인벤토리 기능이 핵심이라고 할 수 있다. 모바일 플랫폼으로 설계하기 때문에 기존에 쓰던 인벤토리 기능보다는 2개의 리스트를 이용한 간단하고 가벼운 인벤토리 기능을 모바일에 적합하게 설계한 것이 특징이다.

본 논문은 최근 게임시장에 인기 장르로 자리 잡은 서바이벌 게임을 유니티 3D 엔진을 사용하여 안드로이드 기반으로 개발하였고, 2개의 리스트만을 이용하여 가벼운 인벤토리 기능을 구현하였다. 또한 인벤토리뿐만 아니라 모바일에 적합하게 만든 조이스틱과 인벤토리의 기능을 이용해서 만든 조합기능 및 게임의 분위기에 맞는 사운드를 추가하였고 간단한 게임목표(살아남아라)로 플레이어의 도전의식을 이끌어 내어 게임의 완성도를 높였다.

REFERENCES

[1] 2017 Game Industry White Paper, Korea Creative Content Agency, 2017
[2] Don't Starve, https://namu.wiki/w/Don%27t%20Starve
[3] Just Survive: https://namu.wiki/w/H1Z1:%20Just%20Survive
[4] Creighton, "Unity 3D Game Development by Example Beginner's Guide", 2010
[5] Charles Bernardoff , NGUI for Unity, PACKT, 2014
[6] Will Goldstone, "Unity 3.x Game Development Essentials", Packt Publishing; 2 edition, December 20, 2011
[7] J. Lee, Absolute Class! Unity5, Wiki Books (2015).
[8] NGUI: http://www.tasharen.com/?page_id=140
[9] Sung Soo Kim, "A development research for User Interface using Unity 3D in real life", Paichai University Master's Thesis
[10] C. Pearson, Learning NGUI for Unity, Packt Publishing, December (2014).
[11] Joystick: https://ko.wikipedia.org/wiki/%EC%A1%B0%EC%9D%B4%EC%8A%A4%ED%8B%B1
[12] Bahn Kyoungjin, Hyo Kim, Kyungwon Lee, Hyunhee Kim, "A Study on Effect on Flow of Customized User Interface in Game", Society of Design Convergence, pp. 1-12, May 2007.
[13] Inventory: http://terms.naver.com/entry.nhn?docId=2028680&cid=42914&categoryId=42915
[14] Unity3D: https://unity3d.com/kr
[15] T. Dimes, C# Programming for Beginners: An Introduction and Step-by-Step Guide to Programming in C#, CreateSpace Independent Publishing Platform, January (2015).

Authors

Soo Kyun Kim received Ph.D. in Computer Science & Engineering Department of Korea University, Seoul, Korea, in 2006. He joined Telecommunication R&D center at Samsung Electronics Co., Ltd., from 2006 and 2008. He is now a professor at Department of Game Engineering at Paichai University, Korea. His research interests include multimedia, pattern recognition, image processing, mobile graphics, geometric modeling, and interactive computer graphics. He is a member of ACM, IEEE, IEEE CS, KACE, KMMS, KKITS and KIIT.

Hong-Rae Kim received B.S. degrees in Game Engineering from Paichai University.

Won Joo Lee received the B.S., M.S. and Ph.D. degrees in Computer Science and Engineering from Hanyang University, Korea, in 1989, 1991 and 2004, respectively. Dr. Lee joined the faculty of the Department of Computer Science at Inha Technical College, Incheon, Korea, in 2008, where he has served as the Director of the Department of Computer Science. He is currently a Professor in the Department of Computer Science, Inha Technical College. He has also served as the Vice-president of The Korean Society of Computer Information. He is interested in parallel computing, internet and mobile computing, and cloud computing.