Journal of Next-generation Convergence Information Services Technology Vol.10, No.4, August (2021), pp.423-432
ISSN 2384-101X(p) 2672-1163(e) *http://dx.doi.org/10.29056/jncist.2021.08.07*

유니티 에디터의 익스텐션 활용을 통한 던전의 절차적 생성 개발 파이프라인 효율화에 대한 연구

A Study on the Efficient of Dungeon Procedural Generation Development Pipeline by Using Unity Editor Extension

오영욱[1], 김정윤[2]*

Young Wook Oh[1], Jung Yoon Kim[2]*

요 약

게임의 개발비용은 하드웨어의 발전과 게임사간의 경쟁으로 인해 증가하고 있다. 절차적 생성은 이러한 게임의 개발비용 증가에 대한 대안 중에 하나로 학계와 산업계로부터 주목받고 있다. 이미 여러 게임에서 아이템이나 레벨 등이 콘텐츠들의 생성에 절차적 생성을 이용하고 있으며, 게임의 플레이랑 상관이 없는 환경 애셋의 생성에도 사용되고 있다. 이 중 게임의 레벨은 게임 플레이의 주요 무대가 되는 장소이며, 로그라이크 장르에서는 절차적으로 게임 레벨이 생성되는 것이 중요한 요소중 하나이다. 이러한 레벨의 절차적 생성은 대부분 게임의 플레이 코드에서 실행된다. 따라서 개발의 중간과정을 확인하기 위해서는 게임을 플레이 해야만 하며, 그에 따른 실행 시간이 필요하다. 또한 실행했을 때 생성된 던전의 형태가 원하는 형태나 테스트에 필요한 형태가 아니라면 반복해서 실행할 필요가 있어 개발에 필요한 시간이 증가한다. 유니티 엔진의 확장기능을 이용하여 에디터에서 절차적 생성을 실행한 후 테스트만 따로 실행할 수 있다면 이러한 시간 손실을 줄일 수 있을 것이다.

핵심어 : 유니티 엔진, 에디터 확장, 개발툴, 개발 효율화, 절차적 생성

Abstract

Gaming development costs are increasing because of hardware development and competition among gaming companies. Procedural generation is the attention of academia and industry as one of the alternatives to increasing game development costs. In many games, items and maps are already using procedural generation to create content, and they are also used to create environmental assets that are not related to game play. Among them, the level of the game is a place that serves as the main stage for game play, and In the rogue-like game genre, procedural generation level is one of the important factors.. Procedural generation of game levels is usually executed in the code of game logic. Therefore, in order to check the intermediate development process, the game must be played, and the run time is necessary

1 Department of Game Engneering, Gachon University, Seongnam, Korea [Graduate Student]
e-mail: ohyoungwook@gachon.ac.kr
2 Department of Game Engineering, Gachon University, Seongnam, Korea [Professor]
e-mail: kjyoon@gachon.ac.kr (Corresponding author)
* 본 연구는 문화체육관광부 및 한국콘텐츠진흥원의 연구개발지원사업으로 수행되었음(과제번호: R2020040243)
* 본 논문은 2021년도 차세대컨버전스정보서비스학회 춘계학술대회에서 발표한 논문을 수정 및 보완한 것입니다.

Received(June 12, 2021), Review Result(1st: June 30, 2021, 2nd: July 20, 2021), Accepted(August 13, 2021), Published(August 31, 2021)

accordingly. In addition, if the shape of the dungeon created is not the desired shape or the shape necessary for testing, it must be carried out several times, which increases the time necessary for development. If you can run only tests separately after executing procedural generation in the editor using the Unity engine extension function, this time loss can be reduced.

Keyword : Unity Engine, Editor Extension, Develop Tool, Development Efficiency

1. 서론

AAA급 게임의 개발 인원은 점차 증가하여 1,000명에 가까워지고 있으며, 가정용 게임기의 성능 향상에 따라 개발 난이도 역시 증가하고 있다. 높은 그래픽 품질을 지원하는 그래픽카드를 활용하기 위한 사실적인 비주얼을 위해서 많은 개발자가 필요하며, 그에 따라 게임 레벨 등의 콘텐츠 역시 많이 필요로 하고 있다. 이러한 개발비용의 증가는 게임제작사가 게임을 DLC 등으로 나누어 팔거나, 모바일 게임 등에서 확률형 아이템 비즈니스모델을 사용하여 큰 지출을 유도하는 형식을 선택하게 만들면서 게이머에게 비판받는 상황을 만들고 있다.

이러한 비즈니스모델들은 게임은 무료로 시작할 수 있게 한 후, 부분 유료화 등을 통해 이용자가 꾸준히 비용을 지급하게 하는 것을 기반으로 동작하는 경우가 많으며, 이러한 게임들은 이용자들이 최대한 오랫동안 게임에 머물면서 최대한 돈을 많이 쓰게 하는 고객 총 가치를 중요하게 여기고 있다 [1].

많은 게임이용자는 자신이 즐기는 게임을 오래 플레이하기를 원하지만, 이용자가 게임에 익숙해질수록 콘텐츠 소모 속도는 빨라지고 더 소모할 콘텐츠가 없다고 느끼면 이러한 이용자들은 소셜 네트워크 서비스 등을 통해 불만을 표시하지만, 게임 개발자가 콘텐츠를 제작하는 데는 한계가 있다. 이러한 한계점은 앞서 언급한 개발비용의 증가로 더욱 명징하게 나타나고 있다.

절차적 생성은 이러한 게임 개발의 비용 증가에 대한 대안으로 학계와 산업계에서 주목을 받고 있다. 절차적 생성은 캐릭터, 장비, 레벨, 퀘스트 등 게임의 다양한 분야에 적용되고 있으며, 오래 즐길 수 있는 게임들에서 특히 선택되고 있다. 또한 이러한 게임의 플레이에 직접적으로 사용되는 콘텐츠뿐만 아니라, 게임의 배경으로 쓰이는 소품이나 맵에서 움직이는 행인 등의 환경에서도 적극적으로 활용되고 있다 [2].

이러한 절차적 생성은 대부분 난수를 이용하여 게임 플레이 코드에서 동작하게 된다. 절차적 생성을 이용한 게임 레벨 생성의 경우 대부분 처음 게임을 시작할때 게임을 생성하게 되는데, 현대 게임 엔진의 경우 대부분 에디터에서 플레이를 지원하지만, 레벨의 생성은 에디터에서 게임의 플레이로 넘어간 후 확인을 할 수 있는 경우가 대부분이다. 이러한 절차적 생성의 결과물을 확인하기 위해 개발사 내부에서 하우스툴을 제작하는 경우도 있으나, 이러한 하우스 툴의 경우 보안등을 이유로 공개되지 않기 때문에 어떤 식으로 사용되는지는 외부에서는 알기 힘들다.

게임 레벨의 절차적 생성뿐만 아니라, 게임 데이터를 처리하는 방식에 따라 게임의 특정 환경을

실시간으로 테스트할 수 없고, 게임을 처음부터 실행하여 환경을 개발자나 테스터가 직접 구성해야 하는 경우도 많다. 이러한 개발 방식은 게임 디자이너와 QA 직군이 개발과 테스트를 진행하기 위해 재현시간이 필요하다는 문제점이 있으며, 재현이 복잡하거나 쉽지 않은 경우 많은 시간을 소비하여 개발 기간을 증가시킨다.

유니티 엔진과 언리얼 엔진4는 현업에서 인기 있는 엔진이며, 게임 엔진의 에디터에 필요한 기능을 추가할 수 있는 확장기능을 제공하고 있다. 하지만 실사용 사례와 개발 노하우의 공유는 저조하다.

본 연구에서는 절차적 맵 생성기능을 유니티 엔진의 에디터 확장기능에서 실행할 수 있게 하여 게임 개발 파이프라인을 개선하여 플레이와 로딩에 소모되는 시간을 줄이며, 이를 통해 개발 효율을 증가시킬 수 있을 것으로 기대한다.

2. 기존 연구

게임 개발의 효율을 개선하기 위한 연구는 국내에서 많이 찾아볼 수 있으나, 게임 개발 툴에 대한 내용은 파이프라인 개선의 효과보다 개발의 적용에 집중한 연구가 많다. 2017년부터 gamasutra에 연재되고 있는 Classic Tool Retrospective [3]에서 게임 개발에 사용된 툴의 개발자를 인터뷰하여 소개하고 있다. 이 시리즈는 idSoft에서 둠을 비롯해 각종 게임에 사용되어 개발 효율성을 크게 개선한 TED5를 시작으로 초기의 언리얼 에디터, 데우스엑스의 개발툴, 겜브리오 엔진 등을 당시 개발에 참여한 개발자들과 직접 인터뷰하여 의도와 효과를 설명하고 있다. 2020년에 국내에도 번역되어 출간된 게임 엔진 블랙북은 idSoft의 Wolfenstein 3D의 개발 과정과 배경을 을 다루고 있다. 이 책에서는 사내에서 사용하였던 TED5 툴에 관련하여 “도구 개발 시간을 절약했을 뿐만 아니라 준비 시간도 단축할 수 있었으며 디자이너들이 몇 분이면 하나의 레벨을 만들 수 있었다.”라고 언급하였다 [4].

권익현의 온라인 게임 인터페이스 템플릿(저작툴) 제작에 관한 연구 [5]에서는 게임 인터페이스의 템플릿 툴을 만들어 게임 엔진에 붙일 수 있는 엔진과 툴을 제작하여 개발자들에게 제공하는 연구를 하였다. 이 연구의 저작툴은 온라인 게임에서 기본적으로 쓰이는 템플릿들을 툴에서 제공하며, 사용자들에게 설문을 통해 해당 저작 툴의 효용성에 관해 연구하였다. 나현숙, 정상혁, 정주홍의 스텔스 게임 레벨 디자인 툴의 개선 [6]에서는 J. Templay의 스텔스 게임의 레벨 작성에 관한 연구를 바탕으로 추가 기능을 통해 개선사항을 제공하고 있다. 또한 이러한 개선사항을 유니티 에디터의 확장기능을 통해 적들의 경로나 시야 등을 에디터에서 시각적으로 표시해주어 게임 디자이너의 레벨 생산성을 올릴 수 있도록 돕고 있다.

3. 연구 방법

3.1 기존 게임의 절차적 콘텐츠의 생성 사례

게임 콘텐츠의 절차적 생성은 게임 내부의 콘텐츠 종류를 가리지 않고 연구되며 실제 게임에 적용되고 있다. 1980년에 탄생한 로그(Rogue)는 BSD 유닉스에 기본 탑재되어 있기도 했지만, 절차적으로 생성되는 던전으로 끝없이 재미를 느낄 수 있는 점에 많은 프로그래머의 사랑을 받았다. 이러한 로그의 주요한 특징은 후에 넷핵, ADOM등 수많은 파생작을 만들어냈으며, 이후 풍래의 시렌 시리즈나 디아블로 시리즈, 이상한 던전 시리즈 등이 로그의 시스템에 영향을 받아 절차적 생성 던전을 차용하고 있다. 이러한 게임들을 로그라이크(Rogue-Like)라고 부르며 해당 장르의 주요한 특징 중 하나는 절차적 생성 콘텐츠이다.

넥슨의 야생의 땅: 듀랑고 에서도 절차적 생성을 적극적으로 활용하였으며, 노 맨즈 스카이(No Man’s Sky)나 마인크래프트(Minecraft), 테라리아(Terraria) 등의 게임들도 레벨의 생성에는 절차적 생성을 활용하였다.

게임의 무대가 되는 레벨뿐만 아니라, 캐릭터의 얼굴, 아이템 등에서도 절차적 생성은 적극적으로 활용되고 있다. 디비전 시리즈의 경우 아이템의 능력치는 대부분 난수를 이용하여 생성되며, 디아블로 시리즈 역시 아이템에 붙는 특성과 수치들이 난수를 이용하여 생성된다. 이렇게 생성되는 아이템의 수치는 비즈니스모델에 사용되기도 하여 문제가 되기도 한다.

임무의 자동 생성은 많은 온라인 게임에서 연구되고 있으며, 실제로 적용된 게임들도 많으나 이용자 대부분은 무의미하다고 느끼는 부분이 많아 앞으로의 연구가 필요하다. 주로 규칙을 이용해 절차적으로 생성되는 일일 퀘스트에 많이 사용된다.

3.2 던전의 절차적 생성의 개발적 특성

로그라이크 던전 게임의 레벨 생성은 절차적 생성을 많이 사용하는 분야이다. 던전은 주로 방과 복도, 그리고 몬스터와 함정 등으로 구성되며, 많은 던전의 절차적 생성은 대부분 방을 기본으로 시작하여, 복도를 연결한 후, 몹과 함정, 오브젝트를 배치하는 순서로 진행된다.

로그의 소스코드가 공개되어있어 코드를 통해 던전 생성의 원리를 확인할 수 있다 [7]. 로그의 던전 생성 알고리즘은 [그림 1]의 게임화면에서 확인할 수 있듯이 3X3의 공간에 임의의 크기의 방을 배치한 후 복도를 잇는 형태로 던전을 생성한다. 던전은 시작과 끝이 존재하며, 배치된 몬스터들은 방을 배회한다.

[그림 1] 로그의 게임 화면

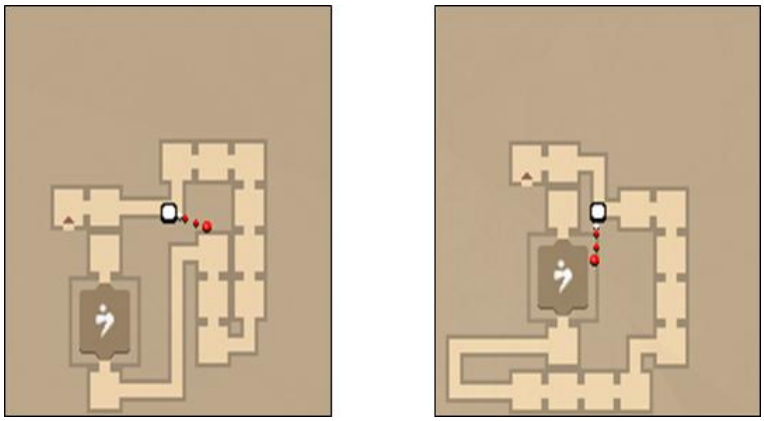
[Fig. 1] Game screen of Rogue

디아블로 1의 던전 생성 알고리즘은 리버스 엔지니어링을 통해 공개되었으며 [8], 그를 통해 알고리즘을 추측해볼 수 있다. 디아블로의 경우 스테이지별로 다른 레벨 생성 공식을 가지고 있으며, 타일 기반으로 자동으로 생성된다. [그림 2]에서 확인할 수 있는 첫 번째 스테이지인 성당의 경우 10X10방을 3개를 배치한 후 방들을 잇는 큰 복도와 재귀함수를 통해 나머지 던전의 구성요소들을 생성하며, 플레이를 위한 조건을 맞추지 못한다면 방을 새로 생성하는 형태이다.

[그림 2] 디아블로의 게임 화면

[Fig. 2] Game screen of Diablo

마비노기의 던전은 여신상 앞에 놓이는 아이템에 따라 던전의 형태가 달라지는 인스턴스 던전으로 아이템이 시드 역할을 한다. [그림 3]처럼 방과 방이 연결되며 가끔 복도가 방을 이어주기도 하지만 그 형태가 방의 위치를 차지하기 때문에 정사각형 방이 랜덤하게 배치되는 형태이다. 마비노기의 던전 생성 알고리즘이 공개되어있지는 않지만, 마비노기의 개발에 참여했던 개발자인 김용하의 2010년 NDC의 “그럴듯한 랜덤 생성 콘텐츠 만들기”에서 언급된 GRID 배열의 형태와 흡사한 것을 알 수 있다 [9].



[그림 3] 마비노기의 알비 던전의 지도

[Fig. 3] Map of Alby Dungeon in Mabinogi

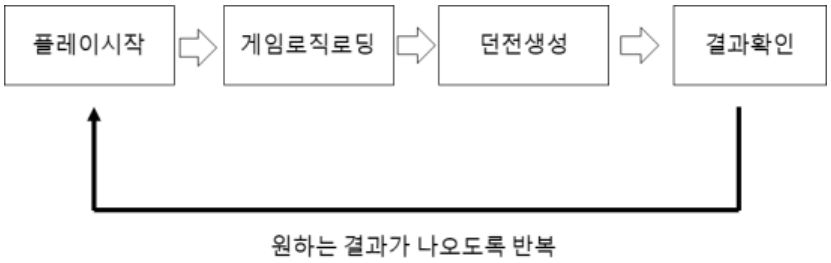
난수를 이용한 던전의 생성은 최근에는 언리얼 엔진이나 유니티 엔진의 애셋 거래시장을 통해 많이 판매가 되고 있으며, 이러한 도구를 통하여 게임 플레이에 레벨 생성을 추가하는 것은 물론, 이러한 도구들을 통해 생성한 후, 사람이 후가공하는 방식으로 처음부터 쌓아가는 것보다 개발 기간을 단축할 수 있다. 하지만 던전의 절차적 생성의 경우 규칙대로 생성하더라도, 개발 과정에 문제가 발생하는 경우가 많다. 목표로 하는 방이 연결되지 않거나, 복도가 꼬이는 등 클리어할 수 없는 던전이 생성될 수 있으며, 이에 대한 대책을 개발자가 미리 준비하지 않으면 이용자가 물리적으로 클리어할 수 없는 레벨과 만나게 된다. 이러한 문제를 미리 방지하기 위해 많은 테스트가 필요로 하지만, 게임 플레이 로직에서 절차적으로 레벨을 생성하게 할 경우, 랜덤 시드값을 따로 지정하여 플레이를 시작할 때 입력하거나 하지 않게 하면 재현이 쉽지 않다. 또한 테스트를 위해 플레이할 때마다 던전을 생성할 때 테스트할 때마다 던전을 생성하는 시간이 소모되어 개발 기간을 늘리는 상황을 가져온다.

3.3 유니티 에디터에서 던전 생성

앞서 언급한 플레이 로직에 절차적 생성 로직이 실행될 경우의 게임 개발 파이프라인은 [그림

4] 과 같다. 여기서 발생하는 시간 손실을 줄이기 위해 던전의 절차적 생성기능을 유니티 엔진의 에디터에서 실행할 수 있게 한다면, 던전을 게임 에디터에서 생성한 후, 유니티 에디터의 플레이 기능을 통해 같은 던전을 반복적으로 플레이할 수 있게 된다. 이를 통해 개선된 파이프라인의 형태는 [그림 5]와 같다. 이렇게 하면 게임 플레이 로직에서 던전을 생성하는 경우와 달리 게임 실행을 하면 던전의 생성을 건너뛰고 해당 던전을 플레이만 할 수 있기 때문에 개발 중 실행과 테스트에서 발생하는 시간의 손실을 줄일 수 있을 것이다.

또한 유니티 에디터에서 플레이할 경우 플레이 중간에 데이터가 바뀌어도, 플레이를 중지할 경우 에디터에서 생성한 상태로 돌아가기 때문에 이전에 생성된 던전의 형태가 유지된다는 장점이 있다.



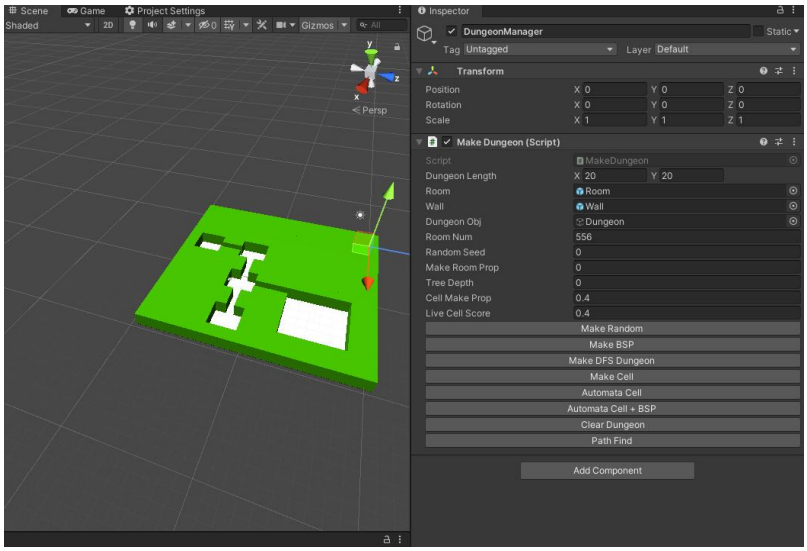
[그림 4] 플레이 로직에서 절차적 생성을 할 겨우의 게임 개발 파이프라인
[Fig. 4] Procedural game content development pipeline with create on play time

[그림 5] 에디터에서 절차적 생성을 할 경우의 게임 개발 파이프라인
[Fig. 5] Procedural game content development pipeline with create on editor expansion

3.4 개발 적용

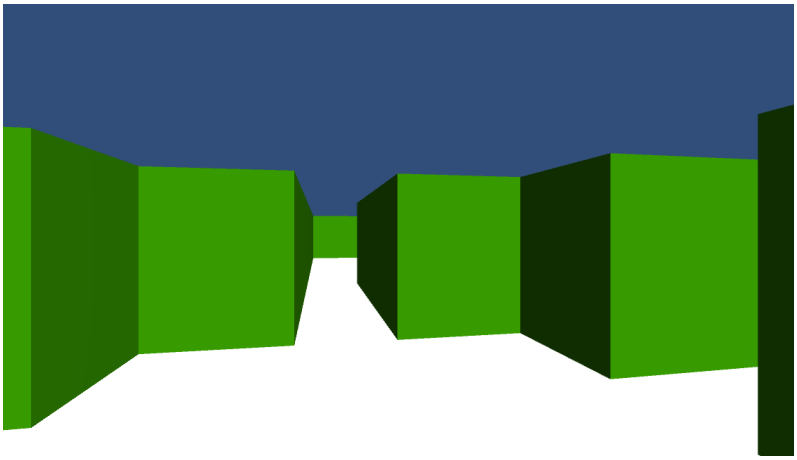
아래의 [그림 6]과 같은 형태로 에디터의 확장기능을 통해 일반적으로 RPG에서 사용하는 방과 복도의 형태의 던전을 생성할 수 있도록 버튼을 추가하였다. 개발자는 이 버튼을 통해 에디터에서 바로 던전을 생성할 수 있으며, 이렇게 생성한 던전을 개발자가 필요한 알고리즘들을 테스트 해 볼 수 있었다. 또한 한번 생성된 던전은 개발자가 지우지 않는 한 사라지지 않기 때문에 이 버튼

을 통해 에디터에서 바로 던전을 생성할 수 있다. 이렇게 생성한 던전을 통해 길 찾기 알고리즘을 테스트해볼 수 있으며, 한번 생성된 던전은 바로 게임 플레이 로직에서 생성했을 때와 달리, 플레이를 중단해도 없어지지 않기 때문에 [그림 7]과 같이 필요한 기능들을 테스트해볼 수 있다. 추가로 에디터에서 던전 생성에 사용하는 랜덤 함수의 시드값을 지정할 수 있기 때문에 테스트에 필요한 경우 시드값을 조정해서 다른 개발자나 테스터가 테스트한 시드값을 가져와서 던전을 재생성하는 것 역시 가능하게 하였다.



[그림 6] 유니티 에디터에 익스텐션을 적용한 화면

[Fig. 6] Screen with extension applied to Unity Editor



[그림 7] 생성된 던전을 엔진에서 플레이를 하여 실행해본 화면

[Fig. 7] Screen to run the generated dungeon on engine play

하지만 에디터에서 업데이트를 통해 계속 화면을 갱신하거나, 메모리 생성 및 해제를 빠뜨린 경우 유니티 에디터 자체가 크래시가 날 수 있으며, 앞서 언급한 메모리 초기화나 과도한 반복으로 인해 해당 GameObject를 선택할 경우 에디터가 매우 느려지는 문제가 발생할 수 있기 때문에 주의해서 작업할 필요가 있다.

4. 결론

본 연구에서는 던전을 유니티 에디터의 확장기능을 활용하여 절차적으로 생성할 수 있도록 하였다. 이를 위해 던전의 생성 코드를 게임의 플레이 로직이 아닌 에디터의 확장 코드에서 실행하도록 하였으며, 에디터에서 던전을 생성하지 않더라도 체크박스나 던전의 생성 여부를 체크하여 플레이할 때마다도 던전을 생성할 수 있게 하여, 던전의 생성을 게임 플레이에서도 사용할 수 있게 하였다.

유니티 에디터의 확장기능을 통한 던전 생성의 경우 플레이 버튼을 누르지 않더라도 생성된 던전을 쉽게 확인해볼 수 있었으며, 벽 블록을 입체로 만드는 것으로 3D 형태의 던전으로도 활용할 수 있었다. 이렇게 생성한 던전은 게임 개발에 필요한 길 찾기나, 던전 생성이 잘못된 형태가 있는 경우의 던전을 반복적으로 테스트하는 데 도움을 받을 수 있었다.

또한 이렇게 에디터에서 생성한 던전을 개발자가 편집하는 것도 가능하였으며, 그렇게 편집한 던전을 실행하는 것 역시 가능하였다. 기존에 문제가 있는 경우를 찾기 위해 원하는 던전의 형태가 생성될 때까지 플레이 버튼을 눌러서 던전을 생성해 볼 경우 게임의 로딩이 필요하였으나, 에디터에서 던전을 생성할 경우, 게임의 로딩 과정을 생략하고 던전을 생성할 수 있었기 때문에, 반복적인 던전 생성에서 일어나는 게임 로딩의 시간 손실을 줄여서 필요한 기능을 테스트하는 시간을 단축할 수 있었다.

그러나 본 연구에서 구현한 기능들은 인공지능의 길 찾기와 실제 던전을 일인칭으로 확인 해볼 수 있는 한계가 있어서 좀 더 효과를 보기 위해서는 RPG에서 일반적으로 포함되는 몹들과 함정, 더 나아가 알고리즘에 열쇠와 잠긴문 등의 콘텐츠가 추가될 필요가 있다. 또한 전투 규칙 등을 적용하여 생성된 게임 레벨에서 반복적으로 플레이와 테스트를 진행할 수 있도록 하는 연구가 필요하며, 이를 통해 게임 디자이너가 던전의 밸런스를 수정할 수 있게 하여 절차적 생성 콘텐츠의 밸런싱에 대한 개발의 효율성에 대해 객관적인 수치를 얻을 수 있는 후속 연구가 필요하다.

References

[1] C. S. Lee, “[GameTech] Chang-soo Lee five-Rocks “Predict the future””, zdnet.co.kr, https://zdnet.co.kr/view/?no=20150319173504, (accessed February 5, 2021).

[2] Y. G. Cheong, B. C. Bae, “Procedural Content Generation for Games.”, Communications of the Korean Institute of Information Scientists and Engineers, vol. 31, no. 7, July 2013, pp 16-25.

[3] D. Lightbown, “Classic Tools Retrospective: John Romero talks about creating TEd, the tile editor that shipped over 30 games”, gamasutra.com, https://www.gamasutra.com/blogs/DavidLightbown/20170223/289955/Classic_Tools_Retrospective_John_Romero_talks_about_creating_TEd_the_tile_editor_that_shipped_over_30_games.php, (accessed February 5, 2021).

[4] F. Sanglar, “Team”, in Game Engine Black Book Wofelstein 3D, S. B. Lee, Eds., Seoul, Republic of Korea: HanbitMedia, 2021, pp. 102.

[5] S. D. Kim, “A study on the production for on-line games interface template tool”, Master’s thesis, Department of Business Administration, Graduate School of Business Administration, Dongguk University, Republic of Korea, 2006. [Online]. Available: http://dcollection.dgu.ac.kr/common/orgView/000000001678.

[6] H. S. Na, S. H. Jeong, J. H. Jeong, “Improving A Stealth Game Level Design Tool”, Journal of Korea Game Society, vol. 15 no. 4. February 2015, pp 29-38, doi: 10.7583/JKGS.2015.15.4.29.

[7] B. Ritzl, “The roguelike archive”, britzl.github.io, https://britzl.github.io/roguearchive, (accessed December 18, 2020).

[8] mewmew, AJenbo, “Devilution: Diablo devolved – magic behind the 1996 computer game”, github.com, https://github.com/diasurgical/devilution, (accessed December 19, 2020).

[9] Y. H. Kim, “Create plausible randomly generated content”, slideshare.net, https://www.slideshare.net/ysoya/ss-4373449, (accessed December 18, 2020).