게임 밸런싱을 위한 효과적인 캐릭터 조절 알고리즘

An Efficient Algorithm for Character Adjustments in Game Balancing

현혜정, 김태식
계명대학교 미디어아트대학 게임 · 모바일콘텐츠학과

HyeJung Hyun(hyunhj@kmu.ac.kr), Taesik Kim(tskim@kmu.ac.kr)

요약

게임에서 밸런스 조절은 그 게임의 성공 여부에 큰 영향을 미치는 중요한 요인으로 자리 잡고 있다. 실제 게임 밸런싱이 정확하게 무엇을 의미하고 어떻게 해야 하는지 구현하기가 매우 어려운 실정이다. 무엇을 위해 게임 밸런싱이 필요하고 게임 중에 게이머를 위한 것인지 게임 자체의 운영을 위한 것인지, 얼마만큼 레벨을 조절할 때 정확하게 밸런싱이 되었다고 해야 할지, 그리고 실제 밸런싱 때 플레이어의 경험과는 어떻게 연계하여 레벨을 조절해서 밸런스를 유지시켜야 할지 등 매우 어려운 문제가 내제되어 있기 때문이다. 이러한 문제에 대한 답변은 매우 주관적이며 게임의 특성에 따라 다르게 표현된다. 따라서 본 연구에서는 객관적 지표를 제공할 수 있는 로그 함수를 이용한 레벨 조절 방안을 제안하고자 한다. 제안된 알고리즘을 사용한 결과 레벨 조절을 수행한 결과와 그렇지 않은 결과가 통계적으로 유의한 차이가 있음을 확인할 수 있었다.

■ 중심어 : | 게임 밸런스 | 게임 알고리즘 |

Abstract

A balanced game is one where the main determining factor for the success of the player is the skill level of that player. Random events can occur, but a better player should be more successful than a poor one unless the player has an unusually long run of bad luck. Balancing a game is a very difficult to define a purpose, and to implement process. The possible ways are somewhat subjective and depend on the nature of the game. This paper provides new algorithm for character adjustment using log function with some nature-based ideas. The algorithm shows not only how we decide the exact point of character adjustment but also how much we have to modify the parameters for changing abilities of the character. This paper shows the experimental results of the algorithm and shows differences between normal game playing and playing with character adjustment.

■ keyword : | Game Balance | Game Algorithm |

I. 서 론

게임에서 밸런스 조절은 그 게임의 성공 여부에 큰 영향을 미치는 중요한 요인으로 자리 잡고 있다. 게임 개발자로부터 게임을 운영하는 운영자, 게이머에 이르기 까지 실제적으로 민감하게 작용할 수 있는 요소를

접수번호 : #071030-003
접수일자 : 2007년 10월 30일
심사완료일 : 2007년 12월 21일
교신저자 : 현혜정, e-mail : hyunhj@kmu.ac.kr

가지고 있는데 경우에 따라서는 잘 개발된 게임이라 할지라도 밸런스 문제로 인해 사용자들의 외면을 당할 수도 있으며 운영자에게는 수익에 직접 영향을 줄 수 있기 때문이다. 게임은 컴퓨터와의 1대1 게임이나 다수를 상대로 하는 게임에서나 항상 상대라는 개념이 존재하며 상대와의 Action을 통해 승패를 결정지어야 하기 때문에 어느 쪽이든 상대적인 우월성 혹은 우위의 능력을 가지도록 할 수밖에 없다. 이를 얼마만큼 조절하여 게이머들이 항상 도전을 위한 긴장과 게임적인 요소에서의 흥미를 동시에 유지하면서 장시간 게임을 하도록 유도하는 것이 게임 개발자나 운영자들이 추구하는 성공의 요인이 될 수 있다[1][2]. 상대에 비해 너무 차이가 나는 능력을 가지거나 너무 어렵거나 쉬운 플레이 조건을 제공한다면 그 게임에 대해 쉽게 흥미를 잃어 가면서 그 게임은 차츰 소멸해 가는 과정을 거치게 될 것이다. 이러한 문제는 상대와의 능력 차이를 최소화 하는 것으로 어느 정도 해결 될 수 있는데 플레이 중에 여러 인자들을 분석하여 적절하게 캐릭터의 능력을 향상시키거나 저하시키게 한다. 이러한 것을 게임 밸런싱 이라고 하는데 실제 게임 밸런싱 이 정확하게 무엇을 의미하고 어떻게 해야 하는지 구현하기가 매우 어려운 실정이다. 무엇을 위해 게임 밸런싱이 필요하고 게임 중에 게이머를 위한 것인지 게임 자체의 운영을 위한 것인지, 얼마만큼 레벨을 조절할 때 정확하게 밸런싱이 되었다고 해야 할지, 그리고 실제 밸런싱 때 플레이어의 경험과는 어떻게 연계하여 레벨을 조절해서 밸런스를 유지시켜야 할지 등 매우 어려운 문제가 내제되어 있기 때문이다[3][4].

인공지능을 연구하는 사람들은 좀 더 자연스러운 방법으로 캐릭터의 능력을 향상시키는 알고리즘을 개발하여 밸런싱에 적용하기위한 연구를 진행하고 있다[5][6]. 게임 밸런싱은 알고리즘을 적용하는 방법론에 따라 static balance와 dynamic balance 방법이 있다. static balance 방법은 게임의 rule과 상호간 어떻게 interactive하는가를 고려하는 것으로 실시간 전략게임에서 unit 간의 상호 strength가 그 요소가 될 수 있다. Dynamic balance는 게임 플레이어가 진행 중인 게임의 상태 혹은 능력 범주 내에서 아주 미세하게 레벨을 조정시키는 방법인데 플레이어가 레벨 조정을 인식하지 못하도록 해야 한다[7]. 게임을 디자인 할 때 Rule 이나 캐릭터의 능력이 조정 가능하도록 하되 게임의 승패에 직접 영향을 미친다거나 플레이어의 의도와는 다르게 balance 조절을 진행시키지 않아야 한다. 또한 게임 전체적으로 볼 때 밸런싱은 일정한 규칙과 방법 그리고 크기에 따라 진행시켜야 한다. 플레이들은 자신이 직접 100% 콘트롤하는 게임에서 더 흥미를 가지게 될 것이며 경우와 게임을 하는 장소와 환경에 따라 확연히 달라지는 캐릭터 능력과 게임 규칙에 대해서는 순식간에 흥미를 잃게 될 것이다. 일부 게임 운영자는 일정한 규칙 없이 필요에 따라 임의로 레벨을조정 하는 것으로 알려져 있는데 정확한 알고리즘이나 수학적 모델 없이 직관적으로 밸런싱을 하는것 보다 자동화된 알고리즘에 따라 처리하게 된다면 더 효과적인 게임 운영이 가능할 것이다[8].

본 논문은 캐릭터의 능력을 조절하면서 밸런싱을 유도하는 방법을 제안한다. 제시하는 방법의 수학적 근거와 신뢰도를 위해 먼저 게임 플레이 환경 즉, 시뮬레이션 모델을 구현하였으며 이 시뮬레이션을 통해 도출되는 레벨 혹은 능력신장과 플레이 시간과의 관계는 일반적인 학습곡선 형태와 같게 하였다. 이렇게 구축된 조건하에서 실제 게임을 시뮬레이션을 통해 시험하였으며 이 과정에서 자동으로 캐릭터의 능력을 조절하는 시점을 발견하고 또한 조절하는 방법을 제시하였으며 이를 통해 밸런싱이 없는 게임 플레이와 자동으로 밸런싱을 한 게임과의 결과를 비교 분석 하였다. 이를 통해 능력 조절을 통한 게임 밸런싱의 새로운 방법뿐만 아니라 다양한 알고리즘 개발시 적용할 수 있는 사전 검증 실험으로도 활용할 수 있는 가능성을 제시하였다.

II. 밸런싱을 위한 게임 환경 구축

1. 게임 환경

본 연구에서는 먼저 실제로 게임을 하는 것과 같은 시뮬레이션환경을 구축해야 한다. 그 이유는 밸런싱을 위한 캐릭터 능력조절을 할 때 실제와 같은 환경과 같

이 일관성 있게 이루어져야 하기 때문이다. 이를 위해 먼저 일반화와 정규화 된 기준데이터가 있어야 한다. 본 논문에서도 캐릭터의 능력을 조절 하고 평가까지 이루어지는 과정에서 일관성 있게 같은 환경이 적용되어 특정 레벨에서의 평균 소요시간 도출과 밸런싱이 되어야 한다. 이것은 다수의 플레이어들이 같은 환경으로 구축된 환경에서 게임을 하게하여 각 레벨별로 소요되는 시간의 평균값을 구하면 된다. 이 평균값은 랜덤하게 나타나는 것이 아니라 인간들이 반복되는 학습을 통해 자연스럽게 축적되는 능력과 쉬운 게임은 빠른 시간 내에 해결하고 어려운 문제는 일반적으로 기하급수(exponential series) 형태로 나타난다는 일반적인 현상과 유사한 결과를 나타낼 수 있는 모델이 되어야 한다. 또한 평균 소요시간을 찾기 위해 게임에 사용되었던 캐릭터들의 특성 또한 같은 조건으로 계속 사용되어져야 한다. 이러한 환경 하 에서 다음 [그림 1]과 같은 과정을 거쳐 평균소요시간을 산출 할 수 있다. 게임에 소요되는 시간개념은 실제 시스템 시간을 측정하는 방법과 각 캐릭터들의 액션을 시간 단위로 보는 방법과 일정 점수에 도달하여 한 게임이 종료되는 시점을 시간의 단위 개념으로 볼 수 있다. 본 연구에서는 향후 인공지능기법을 적용할 수 있는 이론적인 접근 방법을 모색하기 위해 플레이어들의 경험에 바탕을 둔 개념에 가장 근접한 일정점수 도달에 따른 게임 종료되는 시점을 하나의 시간 단위로 간주 하였다.

본 연구에서 구축한 게임 환경은 컴퓨터와 1대1로 대전하는 액션게임으로 각 각의 캐릭터마다 타격최소 거리, 타격최대거리, 그리고 상대방에게 벌점을 가하는 점수로 구성된다. 타격의 최소 거리를 반영한 것은 게임에서 사용할 수 있는 무기의 특성이 적과 떨어진 거리와의 관계를 반영했기 때문이다.

본 연구의 실험대상이 되는 게임은 대전게임으로 1:1로 상대방과 적절한 액션을 취해 격투를 하는 것이며 적군의 타격이 성공했을 경우 아군 캐릭터가 점수를 잃게 되고 반대의 경우 적군이 점수를 잃게 된다. 게임 실행 시 얻을 수 있는 보너스 점수는 고려하지 않았으며 그 이유는 전체 게임 플레이어가 이 게임을 실행하면서 소요되는 시간 값을 평균화 하여 초기 게임 환경을 구축하기 때문에 보너스 개념은 평균 시간 계산에 거의 영향을 미치지 않은 요소가 되기 때문이다. 아군과 적군 캐릭터는 상황에 따라 적절히 사용할 수 있는 각각의 액션을 가지고 있으며 게임 중 레벨 (단계)이 향상될 때마다 캐릭터는 점점 강한 액션을 사용하게 하였다. 레벨은 총 10단계로 구성되어 있으며 매 레벨마다 아군 캐릭터가 적군에게 이기면 다음 레벨로 진행하고 그렇지 않을 경우 계속 싸우게 된다.

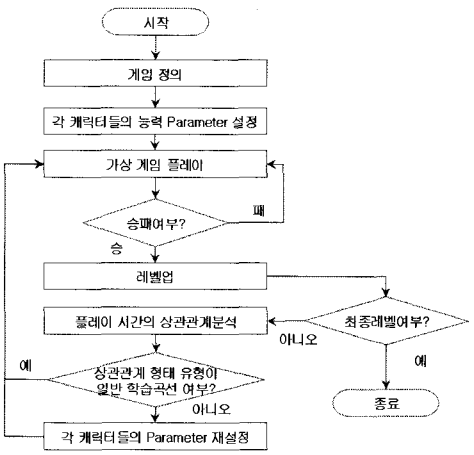

그림 1. 평균소요시간 산출을 위한 시뮬레이션 단계

2. 평균 소요 시간 산출

밸런싱을 하기 위해서는 먼저 기준이 되는 값 즉 각 레벨에서의 평균 플레이 시간이 도출되어져야 한다. 게임 실행 시 아군과 상대 캐릭터가 서로 타격권 범주에 진입할 때 격투가 일어나고 각 캐릭터가 사용한 액션(무기)의 세기에 따라 승패가 갈라지게 된다. 각 캐릭터가 가지고 있는 최소 타격 거리 이상 최대 타격 거리 이내에 상대방이 존재하여야 하고 상대가 타격을 가하더라도 피할 경우는 비긴 것으로 한다. 일반적으로 게임에서 두 캐릭터 혹은 오브젝트가 충돌하거나 근접하는 것을 판단할 때 detection 알고리즘을 이용하거나 의사결정 알고리즘을 적용한다. 본 연구에서는 서로의 캐릭터가 액션을 하는 시점을 판단하기 위해 detection 알고리즘을 바탕으로 최소거리 타격 개념을 추가하였다. 본 시뮬레이션에서 적용한 액션 발생 판단은 다음 식에 따

른다.

캐릭터 N의 위치 및 최소 타격 거리, 최대 타격 거리를 $N(x, y, ra_1, ra_2)$라고 정의할 때 두 캐릭터 $A(a_x, a_y, ra_1, ra_2)$, $B(b_x, b_y, rb_1, rb_2)$ 사이의 공격가능성을 다음과 같이 계산 한다.

A가 공격 가능한 경우

$$(ra_1 + rb_2) < ||A - B|| < (ra_2 + rb_2)$$
$$= (ra_1 + rb_2) < \sqrt{(a_x - b_x)^2 + (a_y - b_y)^2} \quad (1)$$
$$\wedge \sqrt{(a_x - b_x)^2 + (a_y - b_y)^2} < (ra_2 + rb_2)$$

B가 공격 가능한 경우

$$(rb_1 + ra_2) < ||A - B|| < (ra_2 + rb_2)$$
$$= (rb_1 + ra_2) < \sqrt{(a_x - b_x)^2 + (a_y - b_y)^2} \quad (2)$$
$$\wedge \sqrt{(a_x - b_x)^2 + (a_y - b_y)^2} < (ra_2 + rb_2)$$

여기서 $U_1 = (ra_1 + rb_2)$ 이고 $U_2 = (rb_1 + ra_2)$ 이라고 할 때 만약 $U_1 > U_2$ 조건하에서 $U_1 < ||A-B|| < (ra_2 + rb_2)$ 영역에 두 캐릭터가 위치할 경우 A, B 두 캐릭터가 동시에 공격할 수 있게 되며 $U_2 < ||A-B|| < U_1$ 영역일 경우는 A캐릭터만 공격권을 갖게 된다. 또한 $U_1 < U_2$ 조건하에서 $U_2 < ||A-B|| < (ra_2 + rb_2)$ 영역에 두 캐릭터가 위치할 경우는 A, B 두 캐릭터가 동시에 공격할 수 있게 되며 $U_1 < ||A-B|| < U_2$ 영역에 두 캐릭터가 위치할 경우 B캐릭터만 공격권을 갖게 된다. 상호 공격을 하는 캐릭터들이 어떤 액션을 사용했는지 보면서 상대적으로 센 액션을 사용한 캐릭터가 상대에게 점수를 잃게 한다.

제시된 환경과 방법에 따라 총 3,000명의 플레이어가 10단계 레벨을 통과하면서 소요된 시간을 시뮬레이션을 통해 산출하였으며 표와 그래프로 나타내면 아래와 같다.

표 1. 3,000명을 대상으로 한 각 레벨별 평균소요 값

level	평균시간 (게임횟수)	level	평균시간 (게임횟수)
1	1.69	6	27.59
2	1.87	7	37.68
3	4.17	8	93.35
4	13.38	9	135.96
5	21.40	10	260.23

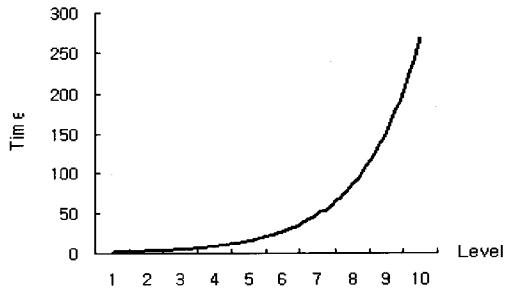

그림 2. 3,000명을 대상으로 한 각 레벨별 평균 값 그래프

III. 게임 밸런싱

게임 중 각 캐릭터들에 영향을 미치는 파라미터 값을 랜덤으로 조절시키는 것은 게임 전체의 일정한 패턴을 무너뜨리는 결과를 가져오게 된다. 연구에 따르면 한 번에 하나의 파라미터만 조절한 후 나타나는 결과를 분석하여 feedback 하는 방법이 무난하다고 알려져 있다. 그러나 파라미터가 많을 경우 어떤 파라미터를 선택할 것이며 또 얼마만큼의 값을 조절해야 하는지는 아직까지 연구 중에 있으며 아직 까지는 개략적으로 그 게임의 특성에 맞도록 운영자가 직관적으로 선택하여 조절하도록 하고 있다. 그러나 이러한 방법은 많은 문제점을 야기 시키기 때문에 인공지능기법을 이용하여 자동화 할 수 있는 방법에 관심을 가지고 있다[5]. 그러나 많은 연구에도 불구하고 실제적으로 어떤 능력에 관련된 파라미터 값을 언제, 얼마만큼 변경 시키느냐 하는 것은 가장 어려운 문제 중의 하나가 될 것이며 더구나 정확한 효과를 검증하기가 어려운 실정이다. 그러나 다양

한 실험과 인공지능 등의 방법을 이용하면 어느 정도 그러한 문제점을 해소할 수 있는데 게임의 종류와 특성에 따라 달라지는 환경을 고려하여 게임에 반영할 수 있는 여러 방법이 개발되어져야 한다[9][10].

본 논문에서는 캐릭터의 능력 조절을 위해 인간이 반복적인 플레이를 통해 축적하는 경험적 지식을 어느 정도 반영하면서 일정한 법칙에 따라 순차적으로 캐릭터를 조절하는 방법을 제안한다. 위 Ⅱ.2절에 언급한 3,000명의 게임 플레이에서의 각 레벨별 평균 소요 시간을 기준치로 설정한 후 새로운 플레이어가 플레이를 할 때 소요되는 시간을 분석하여 평균값보다 너무 많은 시간을 소요하고 있는 플레이어에게는 캐릭터의 능력을 증가시키고 반대로 너무 빨리 진행하는 플레이어에게는 캐릭터의 능력을 다소 감소시키는 방법을 취하게 된다. 제안하는 방법의 수행 절차는 다음과 같다.

1. 한 레벨에서 아군 캐릭터가 이길 때 까지 반복해서 게임
- *1.1 어느 캐릭터 중 하나의 점수가 게임종료 조건이 될 때 까지 다음과정을 수행*
  - *양 캐릭터의 타격가능여부 결정*
  - *캐릭터 액션 선택*
  - *액션 및 판정*
- *1.2 승리 판정*
  - *만약 아군 캐릭터가 졌을 경우 능력 조정 대상인지 확인*
    - *상향조정 대상일 경우 레벨조정계수를 산출하여 캐릭터의 능력 증가*
    - *하향조정 대상일 경우 레벨조정계수를 산출하여 캐릭터의 능력 감소*

1. 밸런싱 시점 판단

밸런싱을 위해 캐릭터의 능력을 조절해 주어야하는 시점인 critical point는 어떤 레벨에서 한 플레이어가 전체 플레이어들의 평균시간 보다 더 플레이하거나 적게 플레이 하는 허용 임계치를 주어 그 수치를 벗어나는 경우를 밸런싱 시점으로 판단 한다. Ⅱ.2절에서 구한 각 레벨별 평균값 중 레벨 k평균값 $A(L_k)$는 다음 식(3) 과 같이 정의 할 수 있다. 레벨 k 에서 임의의 플레이어 p가 현재 플레이하고 있는 시간을 $L_k(t_p)$라고 했을 때 평균시간 값과의 차이 $\Delta(L_k)$ 는 식(4) 와 같이 나타낼 수 있다.

$$A(L_k)=\frac{\sum_{p=1}^{n} L_k(t_p)}{n} \tag{3}$$

$$\Delta(L_k)=L_k(t_p)-A(L_k) \tag{4}$$

λ 을 critical point라고 할 때 식(3)과 식(4)을 이용해서 캐릭터의 능력치를 조절할 시점을 찾을 수 있는데 식(5)은 상향 조절 시점을 식(6)은 하향조절 시점을 나타내고 있다.

$$\Delta(L_k)\geq A(L_k)\cdot\lambda \tag{5}$$

$$\Delta(L_{k-1})<A(L_{k-1})\wedge\Delta(L_k)\geq A(L_k)\cdot(1-\lambda) \tag{6}$$

캐릭터 능력치를 하향할 경우는 플레이어가 평균치에 비해 빠른 속도로 레벨을 향상 시키고 있을 때 필요한 것으로 현재 레벨에서의 플레이 시간을 이용해서 판단하기는 어렵다. 따라서 식(6)과 같이 이전 레벨에서 얼마만큼 플레이를 했는지 그리고 현재 레벨에서 플레이하고 있는 상태를 값을 함께 이용해서 레벨조정 시점을 판단할 수 있다.

2. 캐릭터 능력 조절 계수

게임 진행 중에 캐릭터의 능력 조절이 필요할 경우 어느 정도의 능력을 증가 혹은 감소시켜야 할지 계산하여야 한다. 식(5) 와 식(6)에 의해 캐릭터의 조절 시점으로 판정되었을 때 조절 계수 즉 조절해야하는 값을 계산 하여야 하는데 캐릭터의 능력이 액션과 연관되어 있다면 어떤 액션에 얼마만큼의 능력을 가감 시켜야하는지 결정하여야 한다. 본 논문에서는 플레이어의 시간 개념을 실제 플레이한 회수로 간주하고 게임 시 많이 사용하였던 액션의 경험 축적도 일부 반영하여 조절 대상 액션과 조절계수를 산출하였다.

인공지능의 신경망모델에서 학습요소와 학습활동이 능력 혹은 지능이 신장되는 것을 나타내는 학습곡선은

인간들의 일반적인 학습곡선의 형태와 차이가 없음을 보여주고 있으며 그 형태도 특수한 경우를 제외하고는 [그림 3]과 같은 log 형태로 나타나는 것이 일반적이다.

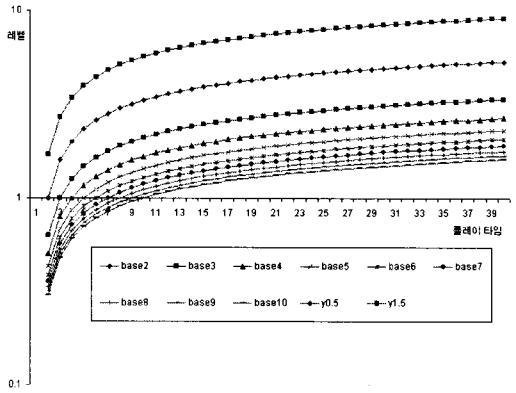

그림 3. 로그함수의 base 별로 차이가 나는 기울기

게임에서 플레이어가 레벨을 높여가는 과정에서도 학습효과가 반영된다고 볼 수 있다. 적 캐릭터에게 이기기까지 반복되는 액션을 통해 터득한 경험을 통해 다음 단계에서는 보다 나은 전략을 구현할 수 있기 때문이다. 게임 중 밸런싱을 위해 캐릭터의 능력을 임의로 조절 할 경우 자연적으로 인간이 취득하는 과정과 능력치 향상과 매우 비슷하게 해야 할 것이다. 게임 전체에 일정하게 미세하게 적용해야하며 인간의 그것과 같은 패턴의 일정한 법칙에 따라야 하고 일반 플레이어보다 너무 많은 능력이 차이가 나는 경우와 약간의 차이가 나는 경우 등도 그에 따른 조절치도 달라야 할 것이다. 이러한 복합적인 요소들을 일부 반영할 수 있는 방법으로 log 함수를 적용하였다. [그림 3]의 log 그래프는 base의 변화에 따라 다양하게 달라지는 것을 보여주고 있는데 게임 특성과 플레이 상태에 따라 적절한 형태의 log 함수를 선택하여 조절계수로 활용하면 된다. 캐릭터 조절 시 가장 고려해야할 사항중의 하나로는 플레이어로부터 인위적인 변경을 인지하지 못하게 하거나 최소화 하여야 한다. 그리고 평균시간보다 너무 많이 차이가 나는 플레이어에게는 빠른 속도로 그 레벨을 벗어날 수 있도록 해야 하며 조금 밖에 차이가 나지 않는 플레이어에게는 많이 차이가 나는 플레이어에 비해 더 적은 능력치를 부여해야 할 것이다. [그림 3]은 log 함수의 base에 따라 나나나는 기울기의 차이를 보여주고 있는데 이 기울기는 학습효과의 상승 개념으로 적용할 수 있고 플레이했던 시간을 접목시키면 완만하면서도 일정한 법칙에 따라 조절이 가능한 수치를 얻을 수 있다. 또한 log 함수의 base 선택은 현 레벨에서 소요되고 있는 시간과 연계하여 평균치에서 멀어져 있는 정도에 따라 얼마나 빨리 레벨을 회복해야하는지 판단하는데 이용된다. 따라서 능력조절계수 δ는 식(7)과 같이 구할 수 있다.

$$\delta = log\,(playtime)\,/\,log(base) \qquad (7)$$

δ 는 아군 캐릭터의 액션에 적용되어 능력을 신장시키게 된다. 캐릭터의 능력은 만약 무기일 경우 폭발력, 사정거리 확대, 정확성 등을 확장시키면 될 것이며 게임의 종류와 특성에 따라 다양하게 활용할 수 있다.

3. 밸런싱 실험

게임 밸런싱의 목적중 하나는 플레이어들로 하여금 적당한 유도를 통해 지속적으로 게임을 즐길 수 있도록 하는 것이다[11]. 플레이어가 밸런싱에 대해 어떻게 인지하는지는 실제 환경에서 실험을 하여야 하나 밸런싱 알고리즘이 얼마나 잘 구성되었는지, 또한 어느 정도 효과를 발생시켰는지는 특정한 장르를 이용하여 시뮬레이션을 통해 간접적으로 확인 할 수 있다. 게임 중 캐릭터들의 능력치를 조절해 주지 않았을 경우와 조절을 통해 밸런싱 한 것과의 결과적인 차이는 먼저 총 플레이한 횟수가 될 것이다. 또한 평균 소요 시간보다 얼마만큼 더 많이 했는지 혹은 덜 했는지도 매우 중요한 차이점이 된다. 밸런싱의 목표를 플레이어가 소요하는 시간을 평균소요시간과 가장 근접하게 유도하는 것이라 했을 때 각 레벨별로 기준치인 평균 소요 시간과 플레이했던 시간과의 차이의 합을 최소화 하도록 유도하는 것이 좋은 알고리즘이라고 할 수 있을 것이다. 따라서 본 연구에서 제시한 알고리즘을 통해 도출된 결과와 조절이 없이 진행된 플레이어의 결과를 밸런싱이라는 관

점에서 비교해 볼 때 평균소요시간과의 차이 합을 비교 분석하는 것이 최선의 방법이라 할 수 있겠다. 따라서 도출된 결과의 목적함수는 다음과 같이 표현 될 수 있다.

각 레벨별로 평균 플레이 시간을 산출할 때 게임한 플레이어 수를 n으로, 각 플레이어가 레벨 i에서 플레이한 횟수를 $L(t_i)$라고 했을 때 평균 소요 시간 $A(L_k)$는 다음과 같이 구해진다.

$$A(L_k)=\frac{\sum_{i=1}^{n} L(t_i)}{n} \qquad k=1\ to\ no.\ of\ levels \qquad (8)$$

이 값을 이용하여 레벨 k에서 실제 플레이한 횟수와 평균 소요시간의 차이 $\Delta(L_k)$를 다음과 같이 구한다.

$$\Delta(L_k)=L(t_k)-A(L_k) \qquad (9)$$

따라서 플레이어 P가 레벨 l개로 구성된 게임 전체에서 발생한 차이시간의 합은 다음과 같이 계산되어진다.

$$P=\sum_{k=1}^{l}\Delta(L_k) \qquad (10)$$

이 식을 게임을 하는 플레이어에 적용하면 되고 목적함수는 다음과 같이 된다.

$$\min imize\left(\sum_{k=1}^{l}\Delta(L_k)\right) \qquad (11)$$

구축된 게임 환경과 제안한 알고리즘을 이용하여 총 10,000명의 플레이어가 10개 그룹으로 1,000명씩 나누어 각각 10단계 레벨을 통과하는 모의실험을 하였다. 첫 실험은 밸런싱을 하지 않고 게임을 하였으며 두 번째 실험은 본 연구에서 제안한 조절알고리즘을 적용하여 밸런싱을 하였다. 밸런싱을 위한 임계치는 평균값을 100%으로 간주 할 때 190%로 하였다. [표 2]는 실험을 통해 도출된 식(10)값의 1000명에 대한 평균을 보여 준다.

밸런싱을 적용하지 않은 경우 10개 그룹 각 1000명은, 총 10,000명의 평균시간과의 차이 합의 평균은 466.67로 나타났으며 밸런싱을 한 경우 396.85로 나타났다.

표 2. 일반방법 및 밸런싱을 적용한 방법의 평균값

실험집단	$\sum_{i=1}^{l}\Delta(L_k)/1000$	
	일반방법	밸런싱적용
1	472.11	398.29
2	465.32	394.52
3	466.37	378.97
4	479.46	399.99
5	466.75	399.31
6	467.57	398.55
7	464.70	404.98
8	464.48	400.07
9	459.88	396.16
10	460.04	397.68
평균	466.67	396.85

Ⅳ. 분석 및 결론

[표 2]에서 제시된 결과와 같이 10,000명이 밸런싱없이 플레이한 결과와 목적 함수와의 차이의 평균값은 466.67이며 밸런싱한 결과와 목적 함수와의 차이의 평균은 396.85로 나타내고 있다. 즉 밸런싱을 적용한 경우 그렇지 않은 경우에 비하여 목적함수에 가까워짐을 알 수 있다. 이러한 결과 차이를 통계적으로 유의한지 확인하고자 정규검정을 수행하였다.

표 3. 기술통계량

	N	평균	표준편차	최소값	최대값	백분위수		
						25	50 중위수	75
일반 방법	10000	466.67	211.24	70.00	2332.00	333.00	432.00	537.00
밸런싱적용	10000	396.85	97.36	79.00	584.00	325.00	409.00	476.00

정규검정 수행을 위해서 정규분포를 따르는지 확인하기 위해 일표본 Kolmogorov-Smirnov 검정 결과[표 4]를 살펴보면 근사 유의확률 값이 0.00으로 유의수준 5%에서 유의하므로 일반 방법 적용 및 밸런싱 적용 결과가 모두 정규분포를 따르고 있다.

표 4. 일표본 Kolmogorov-Smirnov 검정

		일반방법	발련싱적용
N		10000	10000
정규 모수(a,b)	평균	466.67	396.84
	표준편차	211.24	97.36
최대극단차	절대값	.12	.06
	양수	.12	.05
	음수	-.08	-.06
Kolmogorov-Smirnov의 Z		12.33	6.13
근사 유의확률(양측)		.00	.00

a 검정 분포가 정규입니다.
b 데이터로부터 계산

다음은 일반방법 및 밸런싱 적용 결과를 빈도 분포 그래프를 나타낸 것이다. Kolmogorov-Smirnov의 Z는 관측 및 이론적 분포 함수간의 가장 차이를 나타내는 수로서 [표 4] Kolmogorov-Smirnov의 Z의 결과에 따라 일반방법 적용 결과 보다는 밸런싱 결과가 정규성이 강하다고 볼 수 있다. 또한 아래의 그래프[그림 3]을 살펴보면 밸런싱 결과가 일반방법적용 결과의 분포 보다는 그래프의 퍼짐정도가 약하게 되어 있다. 즉 가시적으로 확인할 수 있듯이 밸런싱 결과가 좀 더 정규 곡선에 가까워짐을 알 수 있다.

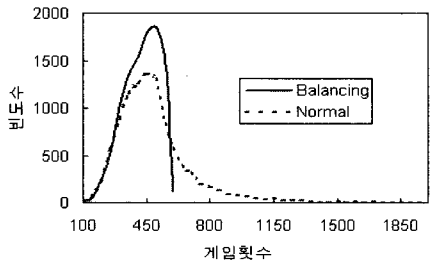

그림 3. N과 B 게임 회수의 평균빈도 그래프
N 일반 방법을 적용한 평균 값
B 밸런싱 적용한 평균 값

다음은 일반방법을 적용한 플레이 결과와 밸런싱 결과의 평균 차이가 유의함을 알아보고자 독립표본 검정을 실시하였다. 연구가설은 두 집단의 평균에는 차이가 있다고 가정하였다. 유의수준 5%를 기준으로 하여 두 집단 평균의 동일성에 대한 t-검정 결과를 [표 5]에 보여준다. 여기서 두 집단은 일반방법을 적용한 게임횟수에 대한 전체 레벨의 각 평균값과 밸런싱한 결과게임 횟수에 대한 전체 레벨의 각 평균값을 의미한다.

표 5. 독립표본 검정

		Levene의 등분산 검정		평균의 동일성에 대한 t-검정				
		F	유의확률	t	자유도	유의확률(양쪽)	평균차	차이 표준오차
게임횟수	등분산이 가정됨	1612.4	.00	30.02	19998	.00	69.82	2.326
	등분산이 가정되지 않음			30.02	14063.51	.00	69.82	2.326

[표 5]에서는 두 집단의 분산의 동질성을 먼저 알아야 한다. 분산의 동질성 여부는 Levene 의 등분산 검정, 즉 F 값을 이용한다. F 검정의 귀무가설은 두 분산의 차이가 없다는 것이다. 위에서는 유의도가 0.00 이므로 $P<0.05$ 에 해당하므로 집단의 분산이 같다는 가설은 기각할 수 있다. 따라서 등분산이 가정되지 않음을 알 수 있다. 등분산이 가정되지 않음에서 t 검정의 귀무가설은 두 집단의 차이가 없다는 것이다. t 검정의 유의도는 0.00 이다. 따라서 5% 유의수준에서 두 집단의 평균이 같다고 볼 수 없다. 두 집단은 결과적으로 평균적 차이가 유의미함을 알 수 있다.

이러한 결론을 통해 본 연구에서 제안한 밸런싱을 위한 알고리즘이 조절 없이 진행된 게임보다 비교적 효과적이라 할 수 있다. 본 연구에서는 임계치를 90%로 설정 하였을 때 전체적으로 약 15%가 개선된 것은 여러 긍정적인 의미를 내포하고 있으나 캐릭터에 미치는 다양한 파라미터들과의 상관관계 규명과 플레이어의 경험을 최대로 반영할 수 있는 방법과 유저들의 다양한 플레이 방법과 파라미터들의 연관성을 지능적으로 분석하여 반영하는 퍼지등 인공지능 기법을 적용하여 보다 나은 알고리즘을 개발할 필요가 있다. 제안된 알고리즘은 능력조정 계수치를 찾아내는 부분에서는 긍정적인 측면이 있으나 어떤 액션에 그 계수치를 적용해야 하는지에 대한 이론적인 근거는 약하다는 단점을 가지고 있다. 또한 단순한 항목으로 대치해서 밸런싱 테스트를 한 것 도 보완해야 할 점으로 향후 더욱 실제적인

환경을 구현하여 테스트해야 할 것이다. 플레이어들이 다양한 환경에서 사용했던 각종 액션(혹은 무기)이 점수 획득에 어떻게 기여했는지, 그러한 경험을 어떻게 효과적으로 관리하여 각 레벨별 평균시간 뿐만 아니라 레벨조절 요소로 사용되어야 할지 향후 연구 과제로 남아 있다.

참 고 문 헌

[1] 이면섭, "에너지개념을 도입한 대전형 액션 게임", 한국컴퓨터산업교육학회 논문집 Vol.7, No.3, pp.163-170, 2006.

[2] 장희동, "게임 메카닉스 시뮬레이션 방법에 관한 조사연구", 정보처리학회논문지A, 제12-A권, 제5호, pp.441-450, 2005.

[3] A. Gustavo, R. Geber, and Hugo Santana, "Automatic Computer Game Balancing: A Reinforcement Learning Approach," AAMAS'05, pp.1111-1112, 2005(7).

[4] G. Christian and J. Troel, "Spatial Principles of Level-Design in Multi-Player First-Person Shooters," NetGames, pp.158-169, 2003.

[5] E. L. John, "Research in human-level AI using computer games," Communication of the ACM, Vol.45, No.1, pp.32-35, 2002.

[6] S. Subhash, D. T. Csaba, and Z. Yunhong, "Selfish Load Balancing and Atomic Congestion Games," SPAA'04, pp.188-195, 2004.

[7] D. V. Bart , V. D. B. Bruno, and V. Tom, "Dynamic Microcell Assignment for Massively Multiplayer Online Gaming," NetGames'05, pp.1-7, 2005(10).

[8] D. Nicolas, E. N. Yee, and J. M. Robert, "Alone Together? Exploring the Social Dynamics of Massively Multiplayer Online Games," CHI 2006 Proceedings Games and Performances, pp.407-416, 2006(4).

[9] 손형률, 노창현, "MMORPG 사용자 시뮬레이션 개발", 게임&엔터테인먼트 논문지, Vol.1, No.1, pp.1-7, 2005.

[10] 김서영, 박태순, "MMORP 콘텐츠 분석툴", 한국콘텐츠학회 논문지, Vol.6, No.10, pp.81-88, 2006.

[11] 김미진, 김재준, "RPG 게임캐릭터의 정적 발란싱 디자인에 관한 연구", 한국콘텐츠학회논문지, Vol.5, No.5, pp.101-106, 2005.

저 자 소 개

현 혜 정(Hyejung Hyun) 정회원

- 1997년 2월 : 한림대학교 컴퓨터공학과(공학사)
- 2003년 2월 : 상명대학교 게임디자인학과(이학석사)
- 2007년 3월 ~ 현재 : 계명대학교 게임·모바일콘텐츠학과 초빙교수

<관심분야> : HCI, 디지털콘텐츠, 감성공학

김 태 식(Taesik Kim) 정회원

- 1987년 5월 : Minnesota State Univ., Moorhead(공학석사)
- 1992년 2월 : North Dakota State Univ.(공학박사)
- 1992년 ~ 현재 : 계명대학교 게임·모바일콘텐츠학과 교수

<관심분야> : 지능형게임, 인공지능, 카오스이론