게임 캐릭터를 위한 자동동작생성기의 개발과 응용

요 약

게임과 캐릭터 애니메이션의 발전으로 캐릭터의 동작을 사실적으로 표현하는 기술은 다양한 분야에서 요구되었다. 이에 따라 사실적인 캐릭터 동작을 생성하는 다양한 방법에 대한 연구가 활발히 진행되어 왔다. 캐릭터의 동작을 생성하는 방법은 숙달된 사용자가 캐릭터의 움직임을 지정하는 방식과 동역학 등의 물리적 법칙을 활용하여 동작을 시뮬레이션하는 방법, 모션 캡쳐 등과 같은 입력 장치를 이용하여 배우의 움직임을 측정하여 데이터로 만드는 방법 등이 있다. 이러한 기법들을 이용하여 생성된 동작들은 각각의 방법마다 나름의 장점이 있지만, 제어가 힘들다는 공통적인 단점을 가지고 있다. 사용자에게 가장 편리한 제어를 제공하기 위해서는 사실적인 동작을 고수준 매개변수를 이용하여 생성할 수 있어야 하며, 이러한 동작의 변형 역시 고수준 매개 변수를 통해 이루어져야 한다. 본 논문은 고수준 매개변수(high-level parameter)를 이용하여 캐릭터의 동작을 효율적으로 제어할 수 있는 자동화된 도구의 개발 방법을 제안하며, 이러한 자동 동작 생성 도구를 활용하여 실제 게임을 제작하는 데에 필요한 기술을 소개한다.

ABSTRACT

As game and character animation industries are growing, techniques for reproducing realistic character behaviors have been required in various fields. Therefore, intensive researches have been performed in order to find various methods for realistic character animation. The most common approaches to character animation involves tedious user input method, simulation with physical laws based on dynamics, and measurement of actors' behaviors with input devices such as motion capture system. These approaches have their own advantages, but they all have common disadvantage in character control. In order to provide users with convenient control, the realistic animation must be generated with high-level parameters, and the modification should also be made with high-level parameters. In this paper we propose techniques for developing an automated character animation tool which operates with high-level parameters, and introduce techniques for developing actual games by utilizing this tool.

키워드

자동동작 생성기, 캐릭터 애니메이션, 고수준 파라미터

Ⅰ. 서 론

게임과 캐릭터 애니메이션의 발전으로 캐릭터의 동작을 사실적으로 표현하는 기술은 다양한 분야에서 요구되고 있다. 이에 따라 사실적인 캐릭터 동작을 생성하는 다양한 방법에 대한 연구가 활발히 진행되어 왔다. 캐릭터의 동작을 생성하는 방법은 숙달된 사용자가 움직임을 지정하는 방식과 동역학 등의 물리적 법칙을 활용하여 동작을 시뮬레이션하는 방법, 모션 캡쳐 등과 같은 입력 장치를 이용하여 배우의 움직임을 측정하여 데이터로 만드는 방법 등이 있다.

모든 관절동작을 사용자가 직접 입력하는 방법은

생성할 수 있는 동작에 제약이 없는 반면, 사용자에게 많은 작업시간을 요구하며, 생성된 동작의 사실성도 매우 낮다. 물리 시뮬레이션 등을 이용한 동역학적인 계산 방법은 사실적인 동작을 생성하지만 역학 모델링 및 동작제어의 문제로 인하여 상호작용적 환경이나 게임과 같은 실시간 환경에 적용하기가 거의 불가능하다. 현재 가장 일반적으로 사용되고 있는 모션 캡처 기법은 센서로 획득된 배우의 움직임 데이터를 기초로 동작을 생성하는 방법이다. 이 방법은 매우 사실감 있는 동작이 생성이 가능하지만 현실에서는 불가능한 움직임이나 모션 캡처가 어려운 동작에는 적용하기 힘들다.

캐릭터의 동작을 생성하는 다양한 기법들은 사실성과 제어성의 두 가지 목표를 적절히 절충할 수밖에 없다. 동작에 대한 제어 능력이 뛰어난 기법들은 일반적으로 생성되는 동작의 사실성이 떨어지고, 동작의 사실성이 뛰어난 기법들은 일반적으로 제어 능력이 낮다.

본 논문은 사실적인 동작을 자동으로 생성하면서도 캐릭터의 감정이나 연령, 성별과 같은 고수준의 개념적 상태 제어를 효율적으로 수행할 수 있는 자동동작 생성기를 개발하고 이를 게임환경에 실제로 활용하는 기술을 소개한다. 동작 생성 도구 개발 기술은 기존 기법에 비해 편리하고 직관적인 방법인 고수준 개념 매개 변수 제어를 통해 동작을 생성하며, 고수준 매개 변수는 연령, 성별, 몸무게, 키, 피로도 등과 같은 다양한 상태를 제어하며, 동작은 실시간으로 생성되어 게임에 적용된다.

본 논문은 이러한 자동동작 생성 도구의 개발과 함께 이를 활용하여 실제 게임에 적용했을 때에 나타나는 문제들도 다룬다. 본 논문은 제안된 캐릭터 동작 생성 도구의 결과와 오픈 소스 그래픽 렌더링 엔진인 오우거(OGRE)를 연동하여 실제 게임을 제작하는 데에 필요한 실무적인 문제의 해결책도 같이 제시한다.

II. 자동동작 생성기법

본 논문에서 소개되는 자동동작 생성기를 통해 생성하고자 하는 캐릭터는 인간형 구조를 가지고 게임과 같은 실시간 환경에서 동작하는 캐릭터이다. 이러한 캐릭터의 움직임을 제어하기 위하여 17개의 관절로 이루어진 캐릭터 모델을 사용하였으며, 각 관절은 표 1과 같이 계층적으로 연결되어 있다. 고수준 매개변수 제어를 통해 직관적으로 캐릭터를 제어하기 위하여 캐릭터의 연령 및 성별, 정신 상태와 같은 다양한 매개함수에 대하여 체형요소, 관절동작이 어떻게 변화하는가를 고려했으며, 이를 바탕으로 계층적인 네트워크를 구축하였다. 이 계층화된 네트워크를 통해 다양한 수준에서의 매개함수 제어가 가능하게 된다.

표 1. 매개함수 계층 구조
Table. 1 Hierarchy of parametric functions

매개함수 계층	
매개함수	내용
고수준 매개함수	나이, 성별, 피로도 등과 같이 운동에 영향을 미치는 고수준의 개념적 제어를 가능하게 하는 매개 함수
중수준 매개함수	팔길이, 팔굽힘, 보폭, 보행 속도 등과 같이 고수준 제어와 저수준 동작 제어의 중간적 개념의 제어 매개함수
저수준 매개함수	동작에 직접적인 변형을 가하는 원시 정의 매개 함수

세 개의 수준으로 계층화되어 있는 매개함수들은 크게 두 종류로 나뉠 수 있는데, 저수준 매개함수로 구성되는 원시 매개함수와, 나머지 신체 동작 매개함수로 구분할 수 있다.

원시 매개함수란 캐릭터의 체격과 모션에 직접적으로 영향을 주는 매개함수로, 매개함수 계층구조의 최하위 계층을 이룬다. 원시 매개함수 구성요소는 관절(17관절) 선택, 기본 변환(회전, 확대축소, 평행이동, 제어점), 영향범위(x축, y축, z축, xyz축 모두, 시간축, 속도) 및 보행 사이클(cycle) 당 프레임 수, 점프의 높이, 점프 체공시간 등이다.

신체동작 매개함수는 모두 21종의 신체 매개함수와 23종의 동작 매개함수로 구성된다. 이러한 매개함수들 사이의 연결을 구성하고 이러한 연결구조는 각각의 매개함수 변위가 계층적으로 연결되어 있는 매개함수에 차례로 전달되어 캐릭터의 다양한 동작을 생성한다.

이러한 매개함수의 매개변수를 제어하여 캐릭터의 동작을 자동적으로 생성할 수 있는 도구의 모습이 그림 1에 나타나 있다. 화면 오른쪽에 나타나 있는

제어 변수들은 나이, 성별, 신장, 체중, 보폭, 속도, 주행여부, 피로도 등을 제어할 수 있게 한다.

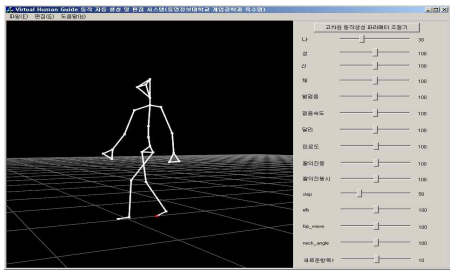


그림 1. 고수준 제어를 통한 동작 생성기
Fig. 1 Motion Generation Tool with High Level Control

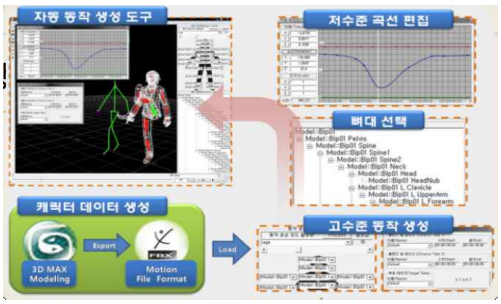


그림 2. 자동동작 생성기의 기능
Fig. 2 Functions of Automatic Motion Generation Tool

그림 1과 같은 초보적인 인터페이스를 통해 자동 동작 생성 기술의 유효성을 검증해 본 뒤 이를 실제 게임 제작 환경에서 사용할 수 있는 도구로 완성하기 위하여 그림 2와 같은 기능을 제공하는 저작 도구 개발을 수행하였다. 그림에서 확인할 수 있는 바와 같이 개발된 저작도구는 개념적 매개변수를 이용한 고수준 동작 제어 기능과 함께, 생성된 도구의 세부적인 요소를 수정할 수 있는 저수준 제어를 같이 제공하고 있다.

기본적인 동작 생성 방법은 매개함수들 사이의 연결을 통해 변위를 전달하는 방식이며, 연결 정보에는 그 매개함수가 받아들일 값의 범위, 연결된 매개함수에 변위를 전달할 때의 가중치 및 전달 함수의 종류 등이 포함되며, 이들 정보가 체형요소, 관절동작의 자유도를 결정한다.

본 논문에서 소개되는 자동 동작 생성 도구는 고정된 구조의 뼈대에만 적용되는 것이 아니라, 다양한 뼈대 구조에 적용될 수 있도록 개발되었다. 게임 등에서 사용되는 캐릭터들을 각각 서로 다른 뼈대 구조를 가지고 있는데, 본 도구는 이렇게 서로 다른 뼈대 구조를 가진 캐릭터들에 대해서 동일한 방식으로 동작 생성 기술을 적용할 수 있다. 그림 3은 본 논문에서 소개하는 자동 동작 생성 도구를 이용하여 서로 다른 종류의 골격을 가진 캐릭터의 동작을 생성한 결과를 보이고 있다. 그림 3의 첫 행은 인체 골격과 다른 구조를 갖는 괴물 캐릭터의 동작을 생성한 결과이며, 두 번째 행은 17개 관절로 이루어진 기본 골격의 캐릭터 동작을 생성한 결과이다.

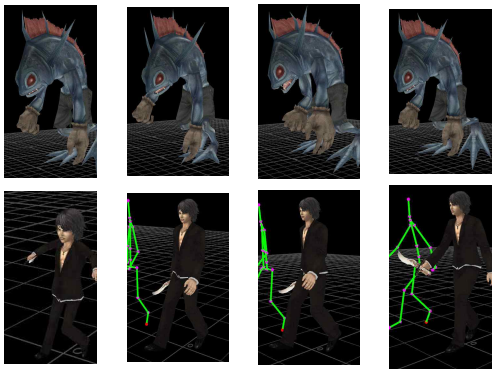


그림 3. 자동 동작 생성 결과 가시화
Fig. 1 Visualization of Automatic Motion Generation

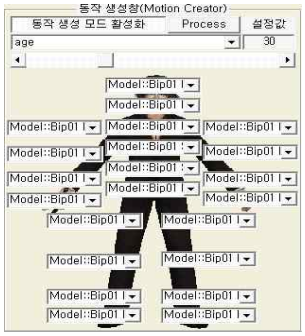


그림 4. 동작 제어 창
Fig. 4 Motion Control Window

그림 4는 캐릭터의 동작을 생성할 때 적용되는 고수준 매개 변수를 각각의 관절에 지정할 수 있도록

하는 인터페이스를 보이고 있다. 이 인터페이스는 고수준 파라미터를 각각의 뼈대에 독립적으로 설정할 수 있도록 해 주며, 이러한 골격 구조 곳곳에 서로 다른 고수준 파라미터를 설정할 수 있는 기능은 생성할 수 있는 동작의 다양성을 더욱 증대시킨다.

III. 자동 동작 생성 도구의 게임 활용

개발된 자동 동작생성 도구는 게임 개발의 효율성을 높이기 위한 도구로 개발되었다. 이러한 목표를 효과적으로 달성할 수 있는지를 검증하기 위하여 자동 동작생성 도구를 실제 게임 제작 환경에 적용하여 보았다. 개발된 동작생성 도구는 동작 데이터의 범용성을 위하여 FBX 파일 형식을 사용하였다. 따라서 FBX 파일 형식을 게임 데이터로 사용하는 경우에는 즉각적으로 결과를 활용할 수 있다.

실제 게임 제작에서는 캐릭터의 기하정보와 애니메이션 정보를 다루기 위하여 매우 다양한 파일 형식을 사용한다. 따라서 생성된 캐릭터 동작을 목표 환경에 맞게 변환할 필요가 있다. 자동 동작생성 도구 자체를 변경하여 목표 환경에 적합한 동작 데이터를 생성하는 방법이 있지만, 가능한 목표 환경의 수는 매우 다양하며, 새로운 동작 데이터 형식이 나올 때마다 이를 반영해야 한다는 문제가 있다. 이러한 문제점 때문에 본 논문의 실험은 자동 동작 생성 엔진에서 나온 결과 동작 데이터를 실시간으로 목표 환경에 맞춰 변환하는 방법을 사용하였다. 이러한 작업을 수행하는 소프트웨어 모듈을 '실시간 상호 변환 모듈'이라고 하자. 이러한 방식은 게임 환경에서 실시간 상호 변환 모듈이라는 단계를 한 번 더 거쳐야 하기 때문에 계산 효율이 떨어질 수 있다. 하지만, 실시간 상호 변환 모듈의 수정만으로 자동 동작생성 도구의 결과를 다양한 환경에 적용할 수 있다는 장점을 가진다. 본 논문에서는 FBX 파일로 생성되는 동작 데이터를 공개 소스 게임 엔진인 Ogre3D에 적용하는 상호 변환 모듈을 개발하여 게임 환경에 적용하였다.

본 논문의 연구에서 사용된 상호 변환 모듈의 핵심적인 기능은 동작생성 도구에서 사용하는 FBX 파일 형식에서 뼈대의 움직임에 의해 변형된 정점들을 실시간에 목표 환경인 Ogre3D의 메쉬(mesh) 정보로 변환하는 것이다. 변환 대상은 정점의 위치뿐만 아니라 법선 벡터, UV 텍스처 좌표 정보가 하나의 조합을 이루어야 하며, 각각의 다각형을 이루는 정점들의 인덱스 정보도 변환이 필요하다.

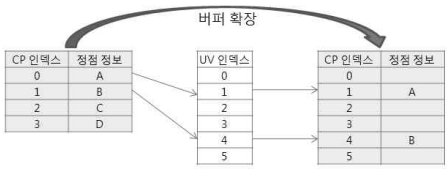


그림 5. PV 저장방식의 CP 방식으로의 변환
Fig. 5 Conversion from PV to CP Method

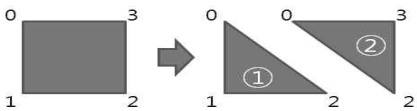


그림 6. 사각형을 두 개의 삼각형으로 분할하기
Fig. 6 Dividing a Quad into two Triangles

FBX 파일 형식에서 정점이 저장되는 방식은 크게 두 가지로 나뉜다. 첫 번째 방법은 제어점마다 법선 벡터와 uv 텍스처 좌표가 1 대 1로 대응하는 방식으로 By-Control-Point (CP) 방식이라고 부른다. 두 번째 방식은 uv 텍스처 좌표가 제어점에 대응하는 것이 아니라 폴리곤에 대응한다. 그래서 제어점과 uv 텍스처 좌표의 수가 동일하지 않으며, 제어점의 인덱스와 uv 텍스처의 인덱스가 달라서 각각 인덱스 버퍼를 따로 둔다. 이러한 방식을 FPB 파일 형식에서는 By-Polygon-Vertex(PV) 방식이라고 부른다

Ogre3D 환경에서는 FBX의 CP 방법으로 데이터 저장이 이루어진다. 또한 CP 방법은 대부분의 파일 형식에서 가장 기본적으로 사용되는 저장 방식이다. 따라서 실시간 상호 변환 모듈이 담당해야 하는 역할은 PV 방식으로 저장된 데이터를 CP 방식으로 변경하는 것이다. 상용 저작 도구를 이용하여 캐릭터를 제작하면 두 가지 방식 중 최적화된 방식으로 저장된다. 따라서 PV 방식으로 저장된 경우 CP 방식으로 변환하는 단계가 추가적으로 필요하게 된다. 매 프레임마다 이러한 변환이 일어나면 게임 진행의 효율성에 악영향을 끼치게 된다. 따라서 PV 방식의

FBX 파일은 게임 메인 루프가 아닌 로딩 단계에서 CP 방식으로 먼저 바꾼 후에 사용하여야 한다. PV 방식은 제어점의 수보다 UV 텍스처 좌표의 수가 많거나 같다. 이 사실을 이용하면 PV 방식에서 CP 방식으로의 변환은 제어점을 그림 5와 같이 UV의 인덱스에 대응시키는 것으로 쉽게 해결할 수 있다.

FBX에서의 다각형 하나는 세 개의 정점으로만 이루어진 것이 아니다. 즉 삼각형 또는 사각형이 사용된다. 하지만 Ogre3D 등은 일반적으로 삼각형 메쉬만을 지원하므로 FBX의 다각형을 삼각형으로 나누는 작업이 필요하다. 이를 위해서는 그림 6과 같이 사각형으로 표현된 면을 두 개의 삼각형으로 분할하고 새롭게 얻어진 면과 정점 인덱스 정보를 하드웨어 인덱스 버퍼에 저장한다. 이러한 방법들을 통하여 자동 동작생성 도구의 결과를 Ogre3D로 작성된 게임 환경에 효과적으로 통합할 수 있다.

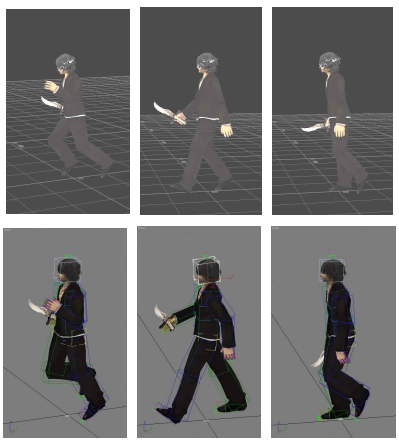
Ⅳ. 실험 결과

본 논문에서 소개된 자동 동작생성 도구를 이용하여 게임 환경의 캐릭터가 현재의 체력 등에 따라 동작을 변경하도록 하였다. 그림 7은 자동 동작 생성 엔진과 Ogre3D를 이용하여 3인칭 액션 게임을 제작한 결과 화면이다. 캐릭터의 체력 수치에 따라 자동 동작 생성 엔진에서 사용하는 고수준 매개변수의 하나인 '피로도' 수치가 변화하고 이에 따라 애니메이터의 개입없이 자동적으로 지친 동작을 생성할 수 있었다. 이러한 동작 생성은 자동 동작생성 도구를 이용하여 데이터로 생성될 수도 있고, 엔진 형태로 게임에 적용되어 게임의 흐름에 따라 실시간에 동작을 생성하게 할 수도 있다. 그림 7의 결과는 동작 엔진 형태로 게임에 포함되어 활용된 예이다.

그림 7. 피로도 변경을 통한 동작 제어
Fig. 7 Motion control with fatigue parameter

그림 8은 본 논문의 자동 동작생성 도구를 동작 엔진으로 활용하여 제작한 게임의 실행 화면이다. 그림에 나타나는 게임 캐릭터의 동작은 애니메이터가 아니라 동작생성 엔진에 의해 자동 생성되었다. 또한 캐릭터가 경험과 성장 등에 따라 캐릭터의 동작이 자연스럽게 변경되므로, 캐릭터의 동작만으로도 캐릭터의 피로도 등을 판단할 수 있다.

그림 8. 자동동작 생성 엔진의 게임 적용
Fig. 8 Application of Automatic Motion Generation to Game Environments



(a) 주행 (b) 보행 (c) 피로

그림 9. 자동생성결과(상)와 수작업(하) 비교
Fig. 9 Comparison of automatic (uppper) and manual (lower) works

자동 동작생성 도구 및 동작 엔진의 효율성을 검증하기 위하여 게임에서 나타나는 주인공 캐릭터의 동작을 애니메이터가 수작업을 통해 제작하여 소요되는 시간을 비교하기로 하였다. 그림 9의 (a)는 주행, (b)는 보행, (c)는 피로에 의해 느려진 걸음 걸이 등을 보이고 있다. 윗 행은 자동 동작생성 기법을 통

해 얻어진 것이며, 아래의 행은 애니메이터가 수작업으로 생성한 결과이다. 자동 동작생성을 통해 생성된 동작들은 실시간에 생성되기 때문에 사실상 거의 시간이 소요되지 않는다. 반면 애니메이터의 수작업을 통해 생성된 동작은 표 2와 같은 시간이 소요되었다. 또한 이러한 세 동작 이외의 다양한 중간 동작들을 생성하기 위해서는 다시 각각의 동작에 소요된 시간 정도가 더 필요하다.

표 2. 애니메이터 작업의 소요시간
Table. 2 Required Time for Manaul Animation

애니메이터의 작업소요 시간		
주행 동작 생성	보행 동작 생성	피로 동작 생성
5시간	5시간	3시간

V. 결 론

본 논문에서는 캐릭터의 다양한 동작을 자동 생성하면서 직관적으로 그 동작을 제어할 수 있는 도구의 개발과 이 도구를 게임 제작에 활용하는 방법에 대해 소개하였다. 현재 구현된 동작생성 기술은 게임 환경, 군중 시뮬레이션 등에서 요구되는 캐릭터의 동작을 효율적으로 생성하면서 효과적으로 제어할 수 있다.

현재의 연구는 보행과 주행 동작에 대해서 진행되었다. 캐릭터의 동작을 자동 생성하여 게임 등에 바로 적용하기 위해서는 보행 동작 이외의 다양한 동작에 대해서는 개념적 파라미터를 통한 고수준 제어가 가능해야 한다. 이러한 요구를 만족하기 위하여 향후 연구로서 다양한 동작에 대해 개념적 파라미터를 적용할 수 있는 방법을 연구할 계획이다.

개발된 도구가 주행과 보행 동작의 생성이라는 제한을 갖고 있지만, 그 발전 가능성은 매우 크다. 본 도구의 개발에 사용된 기술은 캐릭터 동작의 스타일(style)을 실시간에 동적으로 제어할 수 있는 모션 엔진으로서 게임 엔진에 통합될 수 있다. 이러한 형태의 활용 가능성을 검증하기 위하여 본 논문에서는 자동 동작생성 도구를 엔진 형태로 게임에 적용하였으며 이를 통해 개발된 시범 콘텐츠는 애니메이터의 노력이 없이 캐릭터의 상태에 적합한 동작을 효율적으로 생성할 수 있었다.

실험 결과는 이러한 동작 생성 엔진을 사용할 경우 필요한 동작을 애니메이터가 일일이 제작하는 것에 비해 월등한 제작 효율성을 가진다는 것을 보여주었다. 이러한 기술은 게임과 같은 실시간 환경에서 주인공 캐릭터의 자연스러운 동작 생성, 군중 시뮬레이션 환경에서 서로 중복되지 않는 자연스러운 동작 생성 등을 저비용으로 가능하게 할 것이다.

참고문헌

[1] Kazunori Hase, Kazuo Miyashita, Sooyol Ok, Yoshiki Arakawa, Human gait simulation with a neuromusculoskeletal model and evolutionary computation, Journal of Visualization and Computer Animation 14(2), pp 73-92, 2003

[2] Bindiganvavale R. and Badler N. I., Motion abstraction and mapping with spatial constraints. In Modeling and Motion capture Techniques for Virtual Environments, International Workshop, CAPTECH'98, pages 70-82, Nov. 1998.

[3] Bruderlin A. and Williams L.,Motion signal processing. In Robert Cook, editor, SIGGRAPH 95 Conference Proceedings, Annual conference Series, pages 97-104, August 1995.

[4] Cohen M. F., Interactive spacetime control for animation. Computer Graphics (Proceedings of SIGGRAPH 92), 26(2) pages 293-302, July 1992.

[5] Hodgins J. and Pollard N.,Adapting simulated behaviors for new characters. In Tumer Whitted, deitor, SIGGRAPH 97 Conference Prodeedings. pages 153-162, August 1997.

[6] 게임엔진데이터베이스, http://www.devmaster.net/engines/