Proc. of Korea Information Science Society Conference, 24(2):587-590, 1997.

에너지 제한 조건을 이용한 캐릭터 애니메이션 기법

박재권 o 강영민 김성수 조환규
부산대학교 전자계산학과 그래픽스 응용 연구실

Character Animation Techniques Using Energy Constraints

Jae-Kwon Park o Young-Min Kang Sung-Soo Kim Hwan-Gue Cho
Graphics Application Lab., Dept. of Computer Science, Pusan National University
{*jkpark, ymkang, sskim, hgcho*}*@pearl.cs.pusan.ac.kr*

요약

컴퓨터 애니메이션을 구현하기 위한 다양한 기법들이 제안되었으나, 현재까지의 애니메이션 기법들은 애니메이터에게 많은 노력을 요구한다. 본 논문에서는 애니메이터의 작업을 최소화하기 위하여 에너지 파라미터만으로 걸음걸이 애니메이션을 구현하는 기법을 제시한다. 이를 위하여 본 논문에서는 3차원 공간내에서 목표지점까지 걸음을 걷는 객체를 생성하며, 이 객체의 동작은 제어 프로그램에 따라 결정된다. 본 논문에서 제시하는 기법은 유전자 프로그래밍을 이용하여, 객체에 주어지는 에너지 파라미터에 가장 적합한 제어 프로그램을 자동적으로 생성하는 것으로, 에너지 파라미터를 설정하는 최소의 작업만으로 에너지 상태에 따른 다양한 동작을 구현할 수 있는 방법을 제공한다.

1. 서론

컴퓨터 애니메이션은 거의 모든 영상매체에서 예전에는 불가능했던 다양한 표현들을 만들어 내고 있으며, 그 중요성은 날로 증대되고 있다. 애니메이션이 다양한 영상매체에서 대량으로 생산되는 상황에서 더욱 현실성 있는 영상을 제공하고, 자동화된 절차를 제공할 수 있는 애니메이션 기법에 대한 연구와 구현이 요구되고 있다.

컴퓨터 애니메이션을 구현하는 다수의 기법들이 이미 제안되었으며, 이러한 기법들은 크게 두 가지 영역으로 나눌 수 있다. 이 두 가지 영역 가운데 하나는 애니메이션에 대한 애니메이터의 제어를 강화하여 원하는 동작과 일치하도록 하는 방법이며, 또다른 영역은 애니메이션의 자동화를 통하여 애니메이터의 노력을 최소화하는 방법이다. 제어의 측면이 강조된 기법은 애니메이션 생성을 위해 많은 작업을 필요로 하고, 자동화의 측면이 강조된 기법은 캐릭터의 동작을 원하는 대로 제어하기가 어렵다.

본 논문은 이러한 두 가지 애니메이션 모델중 후자에 속하는 자동화된 애니메이션을 구현하기 위한 기법을 제시하며, 이 기법은 애니메이션 구현과정을 자동화하면서도 에너지 제한조건을 이용하여 동작을 쉽게 제어할 수 있는 방법을 제공한다. 이 기법의 구현과 실험을 위해, 본 논문에서는 제어 프로그램에 의해 동작하는 네 개의 다리를 가진 가상적인 생물을 구현하였다. 이 가상 생물의 동작을 결정하는 제어 프로그램은 유전자로 표현되며, 본 논문이 제시하는 기법은 유전자 프로그래밍을 통해 이 유전자를 적합한 값으로 변화시켜 제어 프로그램을 자동적으로 생성한다.

2. 애니메이션 기법 일반

지금까지의 애니메이션 기법들은 다음과 같이 동작 정보를 이용하는 방법과, 동작 최적화를 이용한 자동화 방법의 두 가지로 크게 나눌 수 있다.

2.1 동작 정보를 이용하는 방법

동작 정보를 이용하는 기법의 가장 대표적인 것이 키프레임 기법이다. 이 방식은 애니메이션의 모든 프레임들 가운데서 특정한 프레임을 지정하고 지정된 프레임에 대한 객체의 회전, 이동등과 같은 변환을 저장한 뒤, 키프레임 사이의 프레임들은 보간을 통해 생성하는 방법이다. 이러한 키프레임 기법은 애니메이터의 의도와 일치하는 동작을 생성할 수 있다는 장점이 있지만, 애니메이션의 제작과정이 지나치게 복잡하고, 많은 시간을 소비한다. 그리고 사람이나 동물의 행동을 센서를 통해 입력받아 애니메이션을 구현하기 위한 동작 정보를 추출하는 방법도 있으나, 그러한 기법 역시 구현 장치의 비용이 크다는 한계를 가질 수 밖에 없다.

이러한 형태의 애니메이션으로 분류할 수 있는 기법으로는 여러가지 걸음걸이의 공통된 성질을 추출하여 걸음걸이를 합성하는 파라미터화된 걸음 합성법이 제시되었고[4], 캐릭터 동작의 감정을 표현하기 위한 방법으로 두 개의 극단적인 동작, 예를 들면 아주 지친 모습으로 걷는 사람과 아주 활기차게 걷는 사람의 동작 정보를 획득하여, 이들로부터 개별 팔이나 다리의 움직임 변수를 푸리에 변환을 통하여 재구성하고, 이 변수들을 적당히 보간(interpolation)하거나 외삽(extrapolation)함으로써 다양한 동작을 구성하는 방법이 제시되어 있다[3]. 이 방식의 단점은 표현하려는 동작의 양극단적인 동작에 대한 정보가 있어야 하고, 이로부터 푸리에 변환이 이루어져야 한다는 것이다. 그러나 이와 같이 rotoscopy에 의한 행동 정보를 추출해 이용하는 방법들은 많은 작업 비용과 수작업을 요구한다는 한계를 가진다.

2.2 동작 최적화를 사용하는 방법

이 영역에 속하는 기법의 대표적인 예로는 시공간 제한조건을 이용한 기법이 있다. 이 방식은 어떤 시공간상의 제한조건(예를 들어, 동작과정 중의 사용 에너지)을 가장 최적으로 만족시키는 일련의 동작이 가장 자연스러운 애니메이션 동작이 될 것이라는 가정을 하는 것이다[1]. 그러나 수많은 비선형 방정식으로 표시된 최적화문제를 해결하는 것은 상당한 계산량이 필요하므로, 이를 대신하기 위한 근사 방법인 Sequential Quadratic Programming과 같은 방법론이 제시되어 있다. 그리고 최근에는 이러한 최적화 문제를 해결하기 위한 새로운 패러다임으로서 유전자 프로그래밍이 제시되어 활발히 사용되고 있다[2]. 그러나 이러한 자동화 제어 기법들은 애니메이션 제작 절차를 자동화한다는 장점이 있는 반면, 동작에 대한 제어 능력이 행동을 명시하는 기법에 비해 매우 부족하다. 그리고 물고기의 근육과 뼈대가 외부 자극에 반응하는 시뮬레이션 방식의 기법도 제시되었으나[5], 이런 기법 역시 낮은 통제 능력이라는 한계를 가질 수 밖에 없었다.

2.3 새로운 에너지 제한 기법의 개요

앞에서 살펴본 바와 같이 현재까지의 애니메이션 기법들은 자동화와 제어라는 두 가지 측면중 하나에 비중을 두어 애니메이션을 생성한다. 그러나 본 논문에서 제시하는 기법은 에너지 제한조건을 이용하여 가상 생물의 동작을 다양하게 제어하면서도, 자동적으로 애니메이션을 생성하는 것이다.

이 기법은 가상 생물의 동작을 실질적으로 제어하는 제어 프로그램과, 동작 제어의 수단을 제공하는 에너지 제한조건을 기본으로 한다. 이 기법을 이용한 애니메이션을 구현하기 위해 애니메이터가 수행해야 하는 작업은 생성된 가상 생물이 어떤 제한조건에서 동작할 것이가를 명시하는 것뿐이다. 이 기법은 애니메이터의 다른 작업이나, 미리 추출된 동작 정보의 도움없이 자동적으로 제한조건을 만족하는 제어 프로그램을 생성해내며, 이를 위해 유전자 프로그래밍 기법을 사용한다. 이 기법은 애니메이션 자동화와 동작제어라는 두 측면을 모두 고려하여 효율적인 제어와 애니메이션 제작 절차 자동화를 이룰 수 있는 기법이다.

3. 캐릭터 모델과 동작 제어

3.1 캐릭터 모델

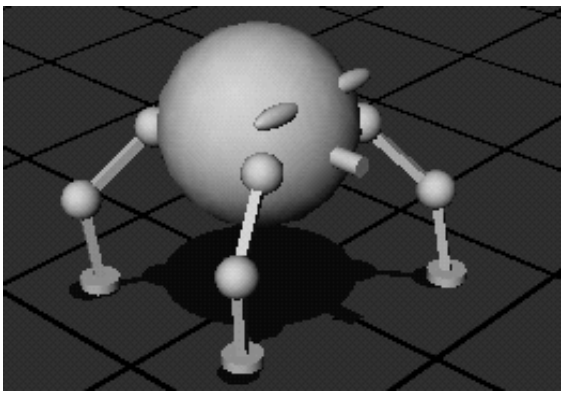

Figure 1: 캐릭터 모델

본 논문의 실험에 사용된 캐릭터 모델은 그림 1과 같이 구형의 몸통과 네 개의 다리를 가진 가상적인 생물이다. 이 가상 생물의 각 다리는 두 개의 관절을 가지고 있는데, 그 하나가 몸통과 다리를 연결하는 관절이며, 다른 하나는 무릎 관절이다. 각 관절의 회전 각도는 명시적으로 지정되는 것이 아니라 발과 몸통의 위치에 의해 결정된다. 즉, 시간에 대한 함수로 구현된 제어 프로그램에 따라 발과, 몸통의 위치가 결정되면 자동적으로 역역학(inverse kinematics)을 이용하여 각 관절의 회전 각도가 계산된다. 따라서 한 객체의 동작을 제어하기 위해서는 그 객체의 발과 몸통의 위치를 결정하는 제어 프로그램을 이용하여야 한다. 우리는 캐릭터의 동작을 제어하기 위한 방법으로 주어진 에너지 제한조건을 이용하여 이 제어 프로그램을 변경해 나간다.

3.2 애니메이션 생성 및 제어 방법

가상 생물의 동작을 결정하는 제어 프로그램은 시간에 대한 함수로 표현되며, 특정 시간에 해당하는 발의 위치와 몸통의 위치를 계산해 준다. 이 제어 프로그램에 사용된 각 계수들이 동일하면, 가상 생물은 동일한 동작을 하며, 다른 동작을 보이기 위해서는 이 제어 프로그램의 계수들을 변화시켜야 한다. 본 논문에서 제시하는 기법은 이 제어 프로그램의 계수들을 유전자로 이용하며, 에너지 제한조건을 만족하는 새로운 유전자를 생성함으로써 캐릭터의 동작을 제어한다.

애니메이터가 설정한 에너지 제한조건이 주어지면 다수의 가상 생물이 생성되고, 각 가상 생물들의 유전자들은 무작위 값으로 초기화된다. 이렇게 생성된 유전자들은 유전자 프로그래밍을 통하여 에너지 제한조건에 적합한 형태로 변화되도록 경쟁한다. 유전자가 무작위값으로 초기화된 가상 생물들은 서로 다른 다양한 행동을 하는데, 이중에서 불가능한 행동을 생성하는 유전자는 시스템에 의해서 자동적으로 제거된다. 가능한 행동을 하는 나머지 가상 생물들은 주어진 시간 동안 목표 지점까지 걸음걸이를 수행하게 된다. 주어진 시간이 지나거나, 생성된 가상 생물중 특정한 수가 목표를 달성하게 되면, 모든 가상 생물은 자신들의 행동에 대한 평가를 받으며, 평가를 통해 우수한 유전자와 열등한 유전자가 결정된다. 평가가 이루어지면 시스템은 다음 세대의 유전자를 생성하며, 다음 세대 유전자의 생성은 복제, 교배, 돌연변이의 세 가지 형태로 이루어진다.

3.3 동작 제어 기법

본 논문에서 구현한 가상 생물의 동작을 제어하는 가장 중요한 요소는 발의 위치와 몸통의 위치를 계산하는 역할을 하는 함수들이다. 이 함수들을 다양하게 정의하면 가상 생물의 동작을 다양하게 구현할 수가 있다. 이러한 함수들을 이용하여 동작 제어 프로그램을 만들며 우리가 제시하는 기법은 이 제어 프로그램을 자동적으로 생성하는 기법이다.

3.3.1 발의 위치 결정

발의 위치는 시간에 따라 앞으로 전진한다. 발의 위치를 표현하기 위해 xz평면을 지면으로 고려하며, 높이는 y축의 좌표로 표현하였다. 그리고 전후의 방향은 z축의 방향으로 표현하였다. 3차원 공간에서 전진할 때, 발은 특정한 위치까지 올라갔다가 다시 내려 온다. 이러한 행동을 보이는 제어 프로그램을 구현하기 위해 발의 위치중 y축 좌표는 sine 곡선을 따라 올라갔다가 내려 오도록 하였으며, z축 좌표는 선형으로 증가하게 하였다. 따라서 걸음걸이를 수행할 때 i번째 발의 높이 $foot_{y_i}$와 발의 z축 좌표 $foot_{z_i}$는 다음과 같이 구할 수 있다.

$$\begin{aligned} foot_{y_i} &= A \cdot sin(t \cdot \pi / T) \\ foot_{z_i} &= t \cdot \frac{S}{T} \end{aligned} \quad (1)$$

이때, A는 발이 올라가는 최대 높이이며, T는 발을 한번 들었다 지면에 놓을때까지의 시간, t는 현재 시간, 그리고 S는 발의 보폭이다.

발의 위치는 항상 변하는 것이 아니라 들어서 움직일 때에만 변한다. 따라서 위의 함수들은 항상 적용되는 것이 아니라, 발의 움직임이 활성화되었을 때에만 적용된다. 그리고 각각의 발이 활성화되는 시간은 유전자의 형태로 저장된다. 발의 움직임이 활성화되는 순간에 그 다리에 해당하는 t의 값은 0이며, 외부 환경의 시간과 동일하게 증가한다. 이 t의 값이 T에 도달했을 때, 그 발의 움직임은 비활성화되어 다음 활성화 시간을 기다린다. 위의 함수에서 발의 높이를 결정하는 A와 보폭을 결정하는 S, 그리고 T는 유전자 프로그래밍을 통해 조정될 값들이다.

3.3.2 몸통의 위치 결정

몸통의 위치 결정은 xz평면상의 위치를 결정하는 함수와 높이를 결정하는 두 가지 함수로 구현하였다. xz평면상의 몸통 위치를 구하는 기본적인 방법은 네 다리 위치의 좌표 평균을 구하는 것이다. 그러나 하나의 다리가 들렸을 때 몸통의 중심은 나머지 세 다리로 지탱할 수 있게 이동해야 한다. 이를 표현하기 위해 네 발의 좌표평균에 몸통을 두다가, 하나의 다리가 들리는 순간에 나머지 세 발의 좌표평균으로 위치를 옮기는 방법을 생각할 수 있으나, 그것은 몸통 위치의 급격한 위치이동을 일으키므로 부자연스러운 동작을 보인다. 이를 보완하기 위해 구현한 위치 결정 함수는 하나의 다리가 높이 올라갈수록 지탱하는 세 발의 중심쪽으로 많이 이동하게 하는 것이다. 이러한 것들을 고려하여 몸통 중심 C를 구하기 위한 함수를 다음과 같이 구현하였다.

$$C = \frac{\sum_{i=1}^{N(legs)} (foot_i + \alpha \cdot (\varpi - foot_i) \cdot (1 - \frac{1}{foot_{y_i}+1}))}{N(legs)} \quad (2)$$

여기서 $N(legs)$는 다리의 갯수를, ϖ는 지탱하는 세 다리에 달린 발들의 좌표평균을 뜻하며, $foot_i$는 i번째 발의 위치, 그리고 $foot_{y_i}$는 i번째 발의 높이이다.

이 함수에서 α는 다리가 들렸을 때 지탱하는 다리의 중심 방향으로 이동하는 정도를 결정하는 요소로, 그 값은 구현된 가상 생물의 몸통 위치 결정에 영향을 미치게 된다. 이 값은 유전자 프로그래밍을 통하여 조정되는 값이다.

위의 함수는 몸통 중심의 x축과 z축 성분을 구하기 위한 함수이며, 높이에 해당하는 y축 성분은 몸통중심과 다리를 연결하는 관절(J)와 발의 위치($foot$)를 xz평면상에 투영했을 때의 거리에 따라 다시 결정되어야 한다. 이를 위하여 J와 $foot$를 xz평면에 투영했을 때의 좌표를 각각 $J_{(x,z)}$, $foot_{(x,z)}$라고 하고, 이 $J_{(x,z)}$, $foot_{(x,z)}$에 대한 함수 $H(J, foot)$를 정의하였다. 이 함수에 따라 생성되는 높이는 $foot_{(x,z)}$가 $J_{(x,z)}$에 가까울수록 높은 값을 가지고, 멀수록 낮은 값을 가진다. 즉, 이 $H(J, foot)$는 $\mid J_{(x,z)} - foot_{(x,z)} \mid = 0$일 때 최대값을 가지고, $\mid J_{(x,z)} - foot_{(x,z)} \mid$가 커질수록 감소하는 함수이다. 이 $H(J, foot)$함수를 이용하여 몸통의 높이를 결정하는 함수는 다음과 같이 구현하였다.

$$C_y = \frac{\sum_{i=1}^{N(legs)} \beta \cdot H_i(J_{(x,z)_i}, foot_{(x,z)_i})}{N(legs)} \quad (3)$$

여기서 β는 $H(J, foot)$함수의 값을 조절하는 함수로 이 역시 유전자 프로그래밍을 통하여 조정한다.

3.4 유전자 프로그래밍

앞에서 설명한 제어 프로그램의 각 함수들은 시간에 따라 특정한 동작을 생성한다. 이러한 함수에 따라 움직이는 객체의 동작을 제어하기 위해서는 A, S, T, α, β와 각 다리의 활성화 시간등의 값들을 조정하면 된다. 이러한 값들을 애니메이터가 직접 제어하여 애니메이션을 구현할 수도 있지만, 우리는 이러한 값들이 주어진 조건에 따라 자동적으로 생성되게 하였다. 이를 위해서 A, S, T, α, β와 다리 활성화 시간등의 값을 유전자로 표현하여 서로 다른 유전자를 가진 가상 생물들을 생성하고 서로 경쟁하게 하였다.

3.4.1 유전자 평가

생성된 N개의 가상 생물들은 한정된 시간동안 경쟁을 하며, 경쟁이 끝나면 주어진 제한조건에 따라 평가된다. 본 논문의 실험에서는 주어진 에너지를 최대한 사용하면서 가장 빠르게 걷는 객체를 우수한 유전자로 평가하였다. 평가방법은 다음과 같다.

전체적인 평가를 위해 두 가지 평가요소를 정의하였다. 첫째는 사용한 에너지(Eu)와 주어진 에너지(Eg)의 차이로 정의되는 에너지 적합도(Ef)이다. 둘째는 목표에 도달한 순위(Ao , $0 \leq Ao < N$)에 대한 함수로 표현되는 속도 평가(Af)이다. 속도 평가의 값은 도착 순위가 0일 경우 0의 값을 가지고, $N-1$일 경우 Eg의 값을 갖게 하였다. 이 두 가지 평가를 한 뒤에 종합적인 평가값 Tf는 Ef와 Af를 더한 값으로 정해지며, 그 값이 작을수록 우수한 유전자로 평가된다.

$$\begin{aligned} Tf &= Ef + Af \\ Ef &= Eg - Eu \\ Af &= \frac{Eg \cdot Ao}{N} \end{aligned} \quad (4)$$

3.4.2 유전자 재생산

평가가 끝나면 다음 세대 경쟁을 위한 새로운 유전자를 생성해야 한다. 이를 위해 사용되는 방법은 복제, 교배, 돌연변이의 세가지 방법이다. 복제는 현재의 유전자를 그대로 다음 세대에 전하는 것이다. 교배는 두 가지 형태로 구분되는데, 첫째는 두 개의 가상 생물을 선택하여 동일한 비로 혼합되게 하는 것이다. 이것은 두 가상 생물의 유전자를 평균하여 생성한다. 또다른 방법은 두 가상 생물을 선택한 뒤 $\rho : 1-\rho (0 < \rho < 1)$의 비로 섞는 것이다. 즉, 새로운 유전자 G_{new}는 선택된 두 유전자 G_1과 G_2를 이용해 $G_{new} = (\rho \cdot G_1 + (1-\rho) \cdot G_2)/2$의 식으로 구하였다. 다음 세대를 생성하는 방법중 나머지 하나가 돌연변이를 이용하는 방법이다. 이 방법은 ρ를 이용한 교배와 비슷하지만, ρ의 값을 0에서 1사이의 값으로 제한하지 않고 무작위값으로 설정하는 것이다.

4. 에너지 제한조건

본 논문이 제시하는 애니메이션 기법은 동작을 자동적으로 생성하는 것으로 객체의 행동을 에너지 제한조건으로 제어한다. 이렇게 객체의 동작을 에너지 제한조건으로 제어하기 위해서는 제어를 위한 에너지의 정의와 계산 방법을 구현하여야 한다. 본 논문의 실험을 위해 정의된 에너지는 몸통을 지탱하기 위하여 지면에 닿아 있는 다리들에 작용하는 토크와 공중에 떠있는 다리를 움직이는 데 필요한 에너지, 몸통의 위치를 높일 때 필요한 에너지, 그리고 몸통을 가속하는데 필요한 힘등을 고려하여 계산하였다. 매 시간 t에서 이러한 요소들이 계산되며, 객체가 소모하는 전체적인 에너지는 이 값들의 총합으로 표현된다.

4.1 에너지 계산의 요소들

지면에 붙어 있는 다리의 토크(τ)

에너지 계산의 요소중 첫번째는 지면에 닿아 있는 다리들에 가해지는 토크이다. 이 요소를 고려함으로써 가상 생물들은 에너지 제한조건에 따라 서로 다른 자세를 취하게 된다. 그

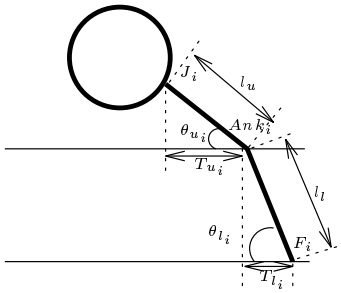

Figure 2: 다리에 걸리는 토크

림 2와 같이 몸통을 지탱할 때, 다리에 가해지는 토크는 상부의 다리에 가해지는 토크 T_u와 하부의 다리에 가해지는 토크 T_l의 합이다. i번째 다리에 힘 F_i가 가해지며 상부와 하부의 다리에도 각각 동일한 힘이 가해진다고 가정하고 i번째 다리에 걸리는 토크 τ_i와, T_{u_i}, T_{l_i}를 다음과 같이 계산하였다.

$$\begin{aligned} \tau_i &= T_{u_i} + T_{l_i} \\ T_{u_i} &= F_i \cdot cos\theta_{u_i} \cdot l_{u_i} \\ T_{l_i} &= F_i \cdot cos\theta_{l_i} \cdot l_{l_i} \end{aligned} \quad (5)$$

여기서 l_u는 상부 다리의 길이, l_l는 하부 다리의 길이이며, $foot_i$가 지면에서 들린 경우 ($foot_{y_i} > 0$)에는 τ_i를 0으로 한다.

$$\tau_i = F_i \cdot (cos\theta_{u_i} \cdot l_{u_i} + cos\theta_{l_i} \cdot l_{l_i}) \quad (6)$$

F_i를 구하는 방법은 몸통의 무게 $m_c \cdot g$를 지면에 닿은 각각의 다리들에 분배하여 구하는데, 분배하는 방법은 xz평면으로 투영한 발의 위치 $foot_{(x,z)_i}$가 xz평면에 투영한 몸통 중심 $C_{(x,z)}$와 가까울수록 많이 분배받고, 멀수록 적게 분배받도록 해야 한다. 그리고 지탱하는 세 다리의 발과, $C_{(x,z)}$의 거리가 각각 α, β, γ이고, 각 다리에 분배된 힘을 $F_\alpha, F_\beta, F_\gamma$라고 할 때, $F_\alpha + F_\beta + F_\gamma$의 값이 $m_c \cdot g$이 되어야 한다. 이러한 조건에 맞게 i번째 다리에 작용하는 힘 F_i를 구하는 방법을 다음과 같이 구현되었으며, 지면에서 들린 ($foot_{y_i} > 0$)경우에는 이 F_i의 값을 0으로 계산하고, 아래 식은 $foot_{y_i} = 0$인 경우에만 적용한다.

$$F_i = (\frac{1}{2} - \frac{D_i(foot_{(x,z)_i}, C_{(x,z)})}{2 \cdot \sum_{j=1}^{N(legs)} D_j(foot_{(x,z)_j}, C_{(x,z)})}) \cdot m_f \cdot g \quad (7)$$

여기서 $D_i(foot_i, C)$함수는 $foot_i$와 C 사이의 거리를 계산하는 함수이며, $foot_{y_i} > 0$일 때에는 그 값이 0이다.

지면에서 떨어진 다리의 위치 에너지 변화(ΔU_f)
두번째로 고려한 요소는 움직임이 활성화된 발을 들어 올리기 위해 필요한 에너지이다. 이 요소는 발의 이전 높이와 현재 높이의 차이로 계산하는데, 발의 높이차를 계산할 때 몸통 자체의 높이 변화로 일어난 차이는 제외한다.

$$\Delta U_f = m_f \cdot g \cdot (\Delta foot_y - \Delta C_y) \quad (8)$$

이때, m_f는 움직이고 있는 발의 질량, g는 중력 가속도이다. 이 요소는 값이 음수일 때에는 무시된다. 이 요소에는 회전관성(R_i)이 반영되어 곱해져야 한다. 발을 움직일 때, 발의 위치를 움직이는 다리의 질량중심으로 가정하고 다리와 몸의 연결부분을 회전중심 R_c라고 할 때, 회전 관성 R_i는 $| R_c - foot |$로 계산한다. 이 요소를 고려함으로써 가상 생물이 발을 움직이는 행동을 에너지 제한 요소로 제어할 수 있다.

몸통의 위치 에너지 변화량(ΔU_C)
가상 생물이 동작을 하면, 몸통의 위치가 변화한다. 이때 몸통의 높이를 높이는 동작은 위치에너지를 증가시키는 것으로 객체는 에너지를 소모한다. 이때 소모하는 에너지는 위치에너지 변화량 ΔU_c로 계산하며, 이 값 역시 음수일 때에는 무시한다. ΔU_c의 값은 다음과 같다.

$$\Delta U_c = m_c \cdot g \cdot \Delta H_c \quad (9)$$

여기서 ΔH_c는 몸통의 높이 변화, m_c는 몸통의 질량이다.

몸통을 가속하는 데 드는 힘(F_a)
몸통을 가속하는 힘 F_a는 몸통의 가속도 a_c에 질량을 곱한 $m \cdot a_c$로 구할 수 있다. 시간이 t일 때의 가속도는 t일 때의 속도와 $t-1$일 때 속도의 차이인 Δv_t로 구하며, Δv_t는 몸통의 위치 C를 이용하여 구한다.

$$\begin{aligned} F_a &= m_c \cdot \Delta v_t \\ \Delta v_t &= v_t - v_{t-1} \\ &= (C_t - C_{t-1}) - (C_{t-1} - C_{t-2}) \end{aligned} \quad (10)$$

C_t는 시간 t에서의 몸통 위치이다.

4.2 에너지 계산식

객체가 움직일 때 한 프레임마다 소비하는 에너지 E는 지탱하는 다리들에 걸리는 토크 T와 다리와 몸통을 움직이는데 드는 에너지와 힘 M의 합으로 구한다.

$$\begin{aligned} E &= T + M \\ T &= \sum_{i=1}^{N(legs)} \tau_i \\ M &= \sum_{i=1}^{N(legs)} \Delta U_{f_i} \cdot R_i + \Delta U_c + F_a \end{aligned} \quad (11)$$

한 객체가 사용한 에너지는 동작을 시작한 시간 t_s에서부터 목표를 달성한 시간 t_e까지 계산된 모든 E의 합, $\sum_{t=t_s}^{t_e} E_t$이다.

5. 애니메이션의 구현 결과 및 향후 연구과제

5.1 구현 환경및 결과

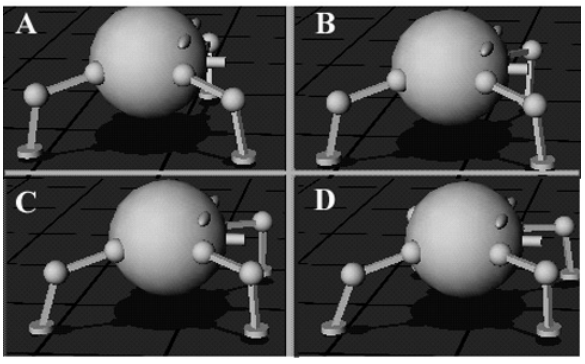

Figure 3: 낮은 에너지에서의 동작

본 논문의 실험은 SGI Indigo2에서 Open Inventor를 이용하여 C++로 구현되었다. 실험을 위해 구현된 가상 생물은 유전자 정보 및 관절의 각도등 약 50여 종의 멤버 데이터와

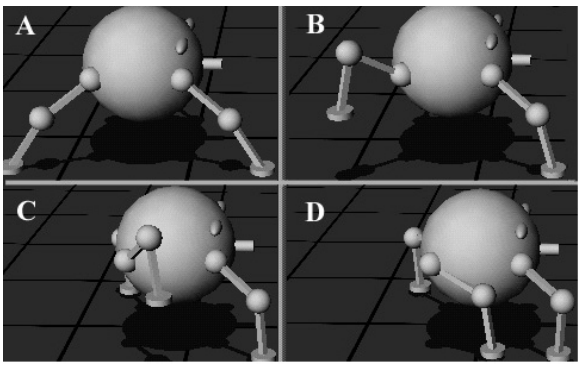

Figure 4: 높은 에너지에서의 동작

30여 종의 method들로 이루어졌으며, 하나의 시스템내에서 에너지 제한조건의 입력과 유전자 프로그래밍의 수행 및 생성된 유전자를 이용한 애니메이션 수행이 이루어진다. 본 실험에서는 유전자 프로그래밍을 위해 500개의 가상 생물을 생성하여 각각의 유전자를 임의로 설정한 뒤, 적합한 유전자로 진화되도록 구현하였다.

그림 3과 그림 4는 본 논문에서 제시한 기법을 이용한 실험의 결과이다. 이 그림에서 보듯이 낮은 에너지가 주어졌을 때 생성되는 유전자는 에너지 소모가 적은 동작을 생성하며, 높은 에너지가 주어진 경우는 큰 보폭과 높고 빠른 걸음으로 목표지점까지 이동함을 볼 수 있다.

5.2 결론 및 향후 연구과제

본 논문에서 제시하는 애니메이션 기법은 에너지 제한조건을 설정하는 최소의 작업을 통해 자동적으로 캐릭터의 동작을 생성하는 방법을 제공한다. 이 기법은 애니메이션의 구현을 자동화하면서도 높은 수준의 제어가 가능한 새로운 애니메이션 기법이다. 본 논문에서 제시된 기법은 그 실험 내용이 목표지점까지의 걸음걸이만을 제어하는 것이지만, 앞으로 더욱 다양한 제어 프로그램을 이용한다면, 다양한 동작에 대한 애니메이션을 효율적으로 제어하면서 그 구현 절차를 자동화할 수 있을 것이다. 그러나 본 논문에서 다룬 걸음걸이 제어 프로그램만으로는 도약과 같은 행동을 생성하거나 제어하는 것이 불가능하다. 이를 보완하고 걸음과 도약을 모두 포함할 수 있는 제어 프로그램에 대한 연구와, 그 밖의 다른 다양한 동작을 제어하고 자동화할 수 있는 방법에 대한 연구가 필요할 것이며, 에너지 제한조건과 함께 애니메이션을 더욱 효율적으로 제어할 수 있는 다양한 기법에 대한 연구도 필요할 것으로 보인다.

[1] M. Kass A. Witkin. Spacetime constraints. *Proc. of SIGGRAPH '88*, 22:159–168, 1988.

[2] J. K. Hahn L. Gritz. Genetic programming for aritic-ulated figure model. *The Journal of Visualization and Computer Animation*, 6:129–142, 1988.

[3] R. Takeuchi M. Unuma, K. Anjyo. Fourier principles for emotion-based human figure animation. *Proc. of SIGGRAPH '95*, pages 91–96, 1995.

[4] Michiel van de Panne. Parameterized gait synthesis. *IEEE Computer Graphics and Applications*, 16(2):40–49, 1996.

[5] D. Terzopoulos X. Tu. Artificial fishes: physics, locomotions, perception, behavior. *Proc. of SIGGRAPH '94*, pages 43–50, 1994.