기본 동작들과 클래스 상속에 기초한 4족 동물의 다양한 '보행' 표현

Representation of 'Walk' for Quadruped Animal Based on Primitive Action and Class Inherit

이인균, 박종희
경북대학교 전자공학과 정보통신학전공

In-Kyun Lee(cooky8884@ee.knu.ac.kr), Jong-Hee Park(jhpark@ee.knu.ac.kr)

요약

본 논문에서 4족(quadruped) 동물들의 여러 가지 인스턴스(instance)들에 대한 '보행(walk)'의 모델링을 간단하게 하기 위한 방법이다. 최초의 클래스 계층에서 정교하게 모델화 된 4족 동물 사용에 의해 최초의 클래스 계층에서 정의 되어 지지 않은 새로운 하위 클래스의 인스턴스에 대해 쉽게 확장하는 '보행'의 모델을 제안한다.

이 방법을 얻기 위해서 분석된 walk의 패턴과 이전에 조사한 클래스 계층에서 연구되어진 4족 동물의 유사한 구조를 따라 분류하고 적용한다. 그리고 상위 클래스(super class)에서 상속되는 동작(action)에 대한 방법을 제안한다. 본 논문은 4족 동물의 특징들을 구체화함으로써 4족 동물의 'walk'를 모델화 하고 필요한 요인들을 정의하였다. 또한 '보행'의 파라메타들에 관한 도메인들을 사용하고 4족 동물의 전형적인 인스턴스들인 말과 소를 모델을 적용한다.

■ 중심어 : | 4족 동물 | 보행 | 기본 동작 | 클래스 상속 |

Abstract

In this paper, we propose a method for simplifying the modeling of 'walk' for various instances of quadruped and easily extending the model of 'walk' for the instance of new subordinate class which is not defined in the original class hierarchy by using the sophisticatedly modeled 'walk' of quadruped.

To achieve this method, we apply the analyzed pattern of walk and classification according to the structural similarity of quadruped studied in the previous researches to the class hierarchy and propose a method for inheriting the actions of super class.

This paper model the 'walk' of quadruped by concertizing the characteristics of quadruped and defining the necessary factors and appropriate domains in terms of parameters of 'walk' and apply the model to the horse and cow, typical instances of quadruped.

■ keyword : | Quadruoed | Walk | Primitive Action | Class Inherit |

접수번호 : #060314-002
접수일자 : 2006년 03월 14일

심사완료일 : 2006년 05월 01일
교신저자 : 이인균, e-mail : cooky8884@ee.knu.ac.kr

I. 서 론

현실세계를 반영하는 가상세계에 존재하는 객체들은 다양한 이동능력을 가진다. 여기에서 말하는 이동은 하나의 객체가 물리적 매개체를 통해 의도적으로 진행 방향을 조절하는 과정을 의미한다. 객체가 외부의 도움 없이 스스로 자신의 진로를 제어할 수 있을 때 자율적인 이동 능력을 갖추었다고 할 수 있다. 인간이나 동물은 이동하기 위해 특정한 동작을 수행해야 한다. 생물학적 객체는 일련의 근육을 함께 사용함으로써 행동을 취한다. 그 결과 생물체는 걷거나, 기거나, 뛰는 이동 동작을 수행한다[1]. 현실세계의 이러한 움직임을 가상세계에 적용하기 위해서는 필수적으로 효율적인 모델링 기법이 필요하다. 이러한 이동에 대한 개념을 이용하고 분석하고 적용한 사례에 대한 연구는 단일모델에 대해 보다 사실적이고 복잡한 모델링 기법이 주를 이루고 있다. 특히 애니메이션 분야, 게임, 그래픽스 등에서 이러한 양상을 볼 수 있는데 실사와 같은 묘사와 이를 위해 복잡한 과정과 연산을 통한 연구가 진행되어오고 있다[2].

본 논문에서는 이러한 기존 연구가 지향하고 있는 사실성 보다는 다양성과 간소함에 있어서 효과적인 모델링 기법에 대해 논하고자 한다. 이를 위하여 4족 동물의 'Walk'의 모델링을 통해 하위계층 인스턴스의 '보행(Walk)'의 모델링을 간소화하고, 새로운 종류의 하위계층의 인스턴스가 추가된 경우 이에 대해 쉽게 '보행'의 모델을 확장할 수 있는 방법을 제시한다. 이를 위해 본 논문은 기존의 연구에서 도입되었던 '보행'의 패턴분석 및 4족 동물의 구조적 유사성에 의한 분류를 계층관계에 적용하고 일반화하며, 각 계층이 가지는 'Walk'의 파라미터와 이를 구성하는 요인(factor)의 도메인 및 기본 동작의 연속성 사이의 상속 방법을 제시한다. 이러한 표현은 하위계층 인스턴스의 모델링의 간소화와 추가된 인스턴스의 확장성이라는 장점을 가진다.

이러한 표현을 위해서 본 논문은 로봇공학에서 연구되고 분류된 '보행'의 패턴과 구조적 분류를 이용한다. 기존 연구의 목적은 동적이고, 역학적으로 효율적인 동작의 생성이다[3]. 이러한 목적으로 4족 동물을 구조적으로 유사한 분류인 달리지 못하는 동물(non-cursorial), 달릴 수 있는 동물(cursorial) 그리고 기어 다니는 동물(reptile)로 분류하고[3], 각각이 가지는 걷는 모양(gait)의 패턴을 분석하고 해석한다[4].

우리는 이렇게 분석된 내용을 사용하여 4족 동물이며 달릴 수 있는 동물의 보행이라는 일반적인 종류가 아닌 실제적인 인스턴스를 표현하고자 한다. 즉, 달릴 수 있는 동물의 보행이 아닌 말이나 소의 보행을 표현하거나, 기어 다니는 동물 계열인 악어, 도마뱀 같이 동일한 계열의 종류를 다양하고 효율적으로 표현하는 것이다. 기존의 연구에서 분석된 다리의 수에 의한 분류와 구조적인 유사성의 분류를 계층관계에 적용하고 각각을 계층관계를 구성하는 하나의 클래스로 사용한다. 이러한 계층관계의 상속성 때문에 유사한 패턴이나 분류의 4족 동물의 보행을 구현함에 있어서 상위클래스의 구현과 특징을 그대로 사용할 수 있는 장점을 가진다[5][6].

본 논문의 구성은 다음과 같다. 먼저, 동작에 대한 정의와 객체들을 공통성과 일반화에 기초하여 움직임에 관한 계층관계를 구성한다. 다음으로 계층관계를 통한 상속성에 대한 정의와 그 예를 들고 '보행'의 패턴을 나누고 기본 동작에 대하여 형식화한다. 이를 통해 4족 동물의 특징의 구체화와 필요한 파라미터를 고려하여 요인을 구성하고 그 도메인을 결정하여 4족 동물의 '보행'을 표현하고 실제 인스턴스인 말과 소의 '보행'에 이 방법을 적용해 본다.

II. 관련 연구

2.1 애니메이션

기존의 동작 생성을 위한 일반적인 기법에는 키프레이밍 방식이 있다. 이 방식은 동작을 이루는 장면을 묘사하면 중간 단계의 연결된 자세를 보간하여 동작을 생성하는 것이다[1]. 이러한 동작의 형태 자체를 중요하게 묘사하는 방식은 동작 형태를 세부적으로 일일이 지정해야 하는 하위 레벨의 작업 중심으로 많은 작업량과 전문성을 요구한다. 또한 동작의 시간적 속성 부여나 사용자와의 상호작용이 어렵다[1].

위의 키프레이밍 방식을 보완하여 동작의 시간적 속성

부여와 보다 효율적인 사용자와의 상호작용을 통해 동작을 생성하고자 하는 기법이 아이콘을 이용한 방식이다[1][7]. 이 방식은 인체 동작에 영향을 미치는 요소들인 각 신체 부위의 높이, 각도, 위치 등을 조절할 수 있는 아이콘을 제공하고 이들의 조작을 통해서 동작을 생성한다. 따라서 상호작용을 통한 동작 생성이 가능하지만 단지 몇 가지의 동작 생성에 제한되어져 있어 다양한 동작 형태의 생성에는 적합하지 않다.

이 밖에 인간의 움직임을 만들어내는 가장 자연스러운 방법으로 모션 캡처(Capture)가 있다. 이는 3차원 컴퓨터 애니메이션의 생성을 위해 가장 많이 사용되는 기술이다. 이 방법은 인간의 움직임을 손으로 그리는 것이 아니라 직접 캡처하여 움직임의 정보를 3차원으로 저장하며, 움직임을 추적하기 위해 마커(Marker) 또는 트랙커(Tracker)라 불리는 센서를 사용한다. 이 시스템은 제작과 운용에 많은 비용이 요구되며 대부분이 매우 제한된 범위의 상황만을 한정된 시간을 통해 체험할 수 있는 형태를 띄고 있다[8].

2.2 로봇공학

로봇공학에서는 움직임을 표현함에 있어서 역학적으로 안정성과 실물과의 유사성에 그 기반을 둔다. 이러한 대표적인 연구에는 보행자세의 패턴을 분석하고 규정하며, 여러 가지 요인과 운동학 및 역 운동학을 고려하여 수식을 도출한다[3][4].

2.3 시각 스크립트

시각 스크립트는 실생활에서 손짓을 이용한 제스춰로 의미를 전달하는 것처럼 동작의 종류와 방향, 경로 등을 지시하기 위해서 마우스 드래깅으로 궤적을 입력한다. 시각 스크립트를 이용하면 동작의 의미적 기술이 가능하고 동작 경로 등의 공간적 속성을 시각적으로 부여할 수 있다. 또한 동작 대상이 되는 신체 부위나 드래깅 속도 등의 제약 조건을 이용하여 동일한 형태의 시각 스크립트로도 다양한 인체 동작을 표현 할 수 있다. 이것은 시각 스크립트를 이용하여 시각적으로 동작을 직접 표현하므로 기존의 키프레임 방식처럼 각 동작을 구성하는 장면을 일일이 묘사하거나 텍스트 스크립트를 이용하여 동작의 개념을 지정된 용어로 표현하는 작업이 필요 없다. 따라서 보다 직관적으로 동작을 생성하고 제어할 수 있는 장점이 있지만 인체의 동작 생성과 보다 수동적이고 정적인 표현이라는 단점이 있다. 또한 스크립트의 생성을 통한 장면 연출이기 때문에 인스턴스의 생성은 불가능하며 주어진 대상의 동작만 스크립트의 재사용과 조합으로 표현한다[9]. 이러한 단점으로 인해 가상세계의 수많은 객체의 움직임을 표현하는 것은 그 한계가 있다.

그러므로 본 논문에서는 객체의 움직임을 표현하기 위한 기본적인 동작을 정의하고 그 동작들의 조합으로 객체들의 움직임을 표현한다. 그러기 위해서 객체들을 다리의 수에 의한 분류와 유사성에 따른 분류를 이용하여 계층구조로 분류하고 이를 바탕으로 계층 구조의 상속성을 이용하여 각 클래스의 동작을 표현한다.

Ⅲ. 동작(Action)

3.1 동작의 구조

동작은 효과를 발생시키기 위한 움직임의 연속된 동작이며 동작은 전제조건(precondition), 절차(procedure), 결과(effect)로 구성된다. [그림 1]은 다리를 가지는 동물의 동작인 'Walk'의 세 가지 부분(part)를 나타낸다[10].

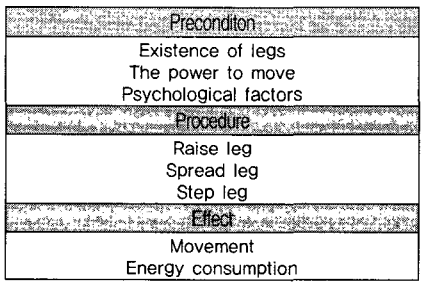


그림 1. 'walk'의 구조

[그림 1]에서와 같이 다리를 가지는 동물의 '보행'은 움직임에 필요한 전제조건을 다리가 존재하는가, 움직임에 필요한 힘이 있는가, 그리고 움직임이 필요한 이유와 목적 등을 나타내는 심리적인 요인들이 있는 가로 표현하고, 움직임을 나타내는 절차적인 부분으로 다리를 들고,

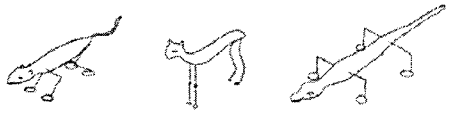
뻗고, 내딛는 순서를 가진다. 그리고 이에 대한 결과로서 위치의 이동과 에너지 소비의 효과를 발생시킨다. 본 논문에서는 가상세계의 동물의 움직임을 모델링하기 위해서 객체들의 속성과 동작 등을 나타내주는 지식베이스인 온톨로지를 사용하고, 이를 통해서 계층관계를 구성하여 상속성을 적용한 모델링을 하고자 한다.

3.2 클래스 계층 구조(Class Hierarchy)

본 논문에서는 분류를 위하여 동작들의 유사성에 따른 집합으로 묶어 주고 그것을 대상으로 하는 계층관계를 구성한다. 다양한 생물학적 분류 기준에서 우리는 행동에 초점을 맞추어 분류를 적용할 것이다.

이 클래스 계층에서는 상위 클래스는 하위클래스의 일반적인 특성을 포함하고 있는 클래스이다.

다리가 있는 동물은 다리의 존재 유무로 분류되고, 그 하위 클래스는 다리의 수로 두발 동물(biped), 4족 동물(quadruped), 육각류의 동물(hexapod)로 분류된다. 그리고 4족 동물의 하위 클래스는 [그림 2]와 같이 다리의 모양에 따라 달리지 못하는 동물, 달릴 수 있는 동물, 기어다니는 동물로 분류된다.



(a) Non-Cursorial (b) Cursorial (c) Reptilian

그림 2. 4족 동물의 자세 모양

이렇게 세분화된 클래스 계층 구조는 하위 클래스로 내려갈수록 개별적 특이성을 추가하여 쉽게 최종적인 인스턴스를 생성할 수 있다.

개별적 특이성은 각각의 클래스가 가지고 있는 그들만의 특정한 속성이다. 다리가 존재하는 동물들의 속성에는 움직일 수 있는 속성을 가진다.

이런 움직임에 관한 속성은 아래 [그림 3]과 같이 분류할 수 있다.

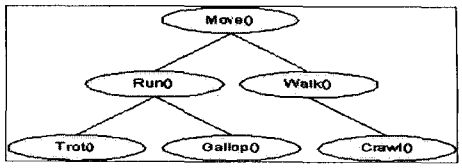


그림 3. 동물의 움직임과 관련된 속성의 분류

이와 같은 클래스 계층 구조는 거주의 형태가 동일하고 구조적인 형태의 유사성의 분류를 기준으로 한다. 이런 클래스 계층 분류는 구조적 유사성이 행동의 유사성을 가져다주는 장점이 있다. 이러한 분류에 의해 클래스에 따라 가지는 클래스 계층 구조는 [그림 4]와 같이 나타낼 수 있다.

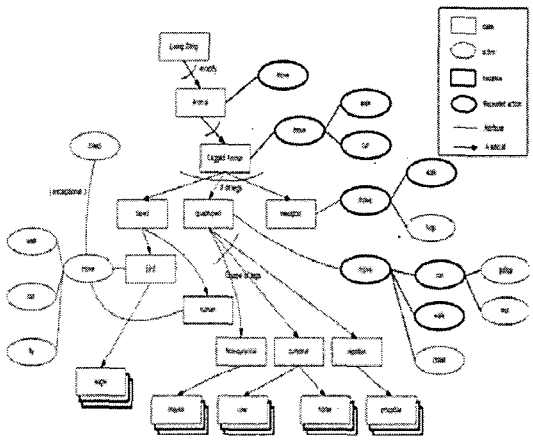


그림 4. '보행'의 움직임과 관련된 클래스 계층 구조

각각의 클래스가 가지고 있는 그들만의 특정한 속성은 자신을 더 구체화하는 값으로 작용하며, 이는 일정한 범위내의 값의 집합으로 나타낸다.

즉, 도메인이라는 것은 특정 인스턴스들을 생성할 수 있게 하는 요인 또는 변수 값들의 집합이다. 일반적으로 그 값들은 벡터의 형태로 모델링 된다. 예를 들어, (크기, 방향) or (v1, v2, ..., vn). 이렇게 특이성을 지닌 클래스들은 하위클래스로 내려가면서 이 도메인의 범위가 더 축소되고 구체화되게 된다[12]. 예로, 말이라는 클래스를 생성하기 위해서 우리가 상위클래스로부터 구체화시켜온 다리길이의 범위가 [100,150] 사이라고 가정하자. 우

리가 원하는 인스턴스는 단지 한 마리의 말이 아닌 다양한 종류의 말의 인스턴스이다. 즉, 같은 말이라 할지라도 다리 길이로 인해 키의 차이도 있을 것이며, 어린 말과 다 성장한 말의 인스턴스도 존재할 것이다. 우리는 이러한 인스턴스의 생성을 위해 단지 최종적으로 구체화한 범위 내에서 다리의 길이를 선택하고 인스턴스를 생성시키면 되는 간편하고 효율적인 방법을 사용하게 된다.

Ⅳ. 4족 동물의 움직임의 모델링

동작을 설명하고자 할 때 현실세계에서는 두뇌의 판단과 신경계의 작용으로 근육과 관절을 움직여 행동을 한다. 하지만 가상 세계에서는 어떠한 움직임에 관한 함수로서 움직임을 표현하며 그 움직임에 대한 변수들이 존재한다. 즉, 객체를 움직이는 것은 변수를 포함한 함수로 표현 될 수 있다. 이러한 변수들에 의한 작용으로 움직임을 조절하고 관절을 움직이며 스스로를 제어해 나가는 것이다. 본 논문에서는 실제적으로 변수를 포함한 함수로서 기본 동작을 규정하고, 이를 통해서 동작의 상속이 어떻게 이루어지는가를 살펴보고 '보행'의 패턴을 규정한다.

4.1 기본 동작(Primitive Action)

상위클래스에서 하위클래스로 구체화시켜 나가면서 가장 기본적으로 상위클래스가 가지는 동작에 대한 요인에 대한 범위와 집합도 구체화되어 나감을 3.2절에서 설명하였다. 이 절에서는 [그림 2]와 같이 다리의 형태를 규정하고[6][13] 이들에 공통적으로 적용할 수 있도록 다리의 각도와 길이를 요인으로 하는 함수를 식(1)와 같이 규정한다.

4족 동물의 클래스가 가지는 특이성은 대표적으로 다리의 수를 들 수 있다. 이를 통해 상위클래스에서 상속받을 기본 동작에 대한 함수가 네 다리로 확장되게 되며, 하위클래스인 달리지 못하는 동물, 달릴 수 있는 동물, 기어 다니는 동물의 세 클래스가 [그림 3]과 같은 자세로 링크의 길이와 초기각도가 세팅되어진다[6]. 이러한 기본 동작에 대한 함수는 [그림 5]를 통해서 계산해 볼 수 있고 보폭, 길이, 각도 및 시간에 따른 함수를 통한 속도와 각속도의 식은 다음과 같다.

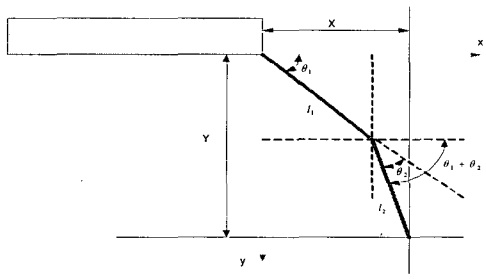


그림 5. 4족 동물의 다리 모델링

여기서 l_1은 다리의 상부 길이, l_2는 다리의 하부 길이를 나타내고, Θ_1, Θ_2는 상하부 관절의 움직임에 대한 각도를 나타낸다.

$$\begin{bmatrix} X \\ Y \end{bmatrix} = l_1 \begin{bmatrix} \cos\Theta_1 \\ \sin\Theta_1 \end{bmatrix} + l_2 \begin{bmatrix} \cos(\Theta_1+\Theta_2) \\ \sin(\Theta_1+\Theta_2) \end{bmatrix} \quad (1)$$

(1) 단, X : 다리를 뻗었을 때의 보폭,
Y : 각도에 따라 계산된 다리의 직선 길이

따라서, 다리의 길이는 상부링크와 하부링크 두 부분으로 나누고 각각이 이루는 각도와 길이에 대한 보폭과 다리의 직선 길이를 계산할 수 있다. 이것으로써 속도의 개념을 정의 할 수 있는데 그 식은 아래 (2)와 같다.

$$\dot{X} = -l_1\dot{\Theta}_1\sin\Theta_1 - l_2(\dot{\Theta}_1+\dot{\Theta}_2)\sin(\Theta_1+\Theta_2)$$
$$\dot{Y} = l_1\dot{\Theta}_1\cos\Theta_1 + l_2(\dot{\Theta}_1+\dot{\Theta}_2)\cos(\Theta_1+\Theta_2) \quad (2)$$

X, Y 각각의 변위의 시간에 따른 미분값을 정의하고 정리하면 아래 (3)와 같다.

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \end{bmatrix} = \begin{bmatrix} -l_1\sin\Theta_1 - l_2\sin(\Theta_1+\Theta_2) & -l_2\sin(\Theta_1+\Theta_2) \\ l_1\cos\Theta_1 + l_2\cos(\Theta_1+\Theta_2) & l_2\cos(\Theta_1+\Theta_2) \end{bmatrix} \begin{bmatrix} \dot{\Theta} \\ \dot{\Theta} \end{bmatrix} \quad (3)$$

여기서, 자코비안 행렬을 식(4)과 같이 정의하고,

$$j=\begin{bmatrix} -l_1\sin\Theta_1-l_2\sin(\Theta_1+\Theta_2)-l_2\sin(\Theta_1+\Theta_2) \\ l_1\cos\Theta_1+l_2\cos(\Theta_1+\Theta_2)-l_2\cos(\Theta_1+\Theta_2) \end{bmatrix} \quad (4)$$

속도와 각속도를 아래의 식(5)으로 계산할 수 있다.

$$\dot{S}=J\dot{\Theta}$$
$$\dot{\Theta}=J^{-1}\dot{S} \quad (5)$$

본 논문에서는 위의 식을 이용하여 클래스가 가지는 도메인 내의 범위에서 값들을 통해 보폭과 각도를 계산하여 움직임을 표현하고자 한다. 위의 식(1)같은 경우는 일반적으로 지면이 평평한 경우를 대상으로 하지만 다양한 환경이 존재하는 가상세계에서는 오르막과 내리막인 경우도 고려할 사항이다. 이에 대해 본 논문은 오르막(또는 내리막)의 경사라는 요인을 더 고려하여 생각해 본다. [그림 6]의 경우 오르막을 올라가는 4족 동물의 형태를 나타내고 있다.

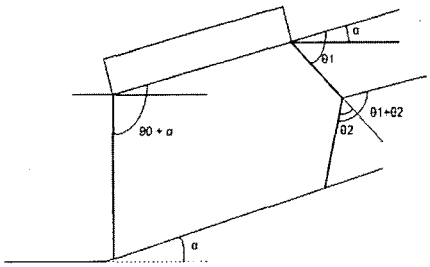


그림 6. 오르막일 경우 자세의 형태

[그림 6]에서 한쪽다리를 이동을 위해 굽힐 때 나머지 다리가 균형을 잡기위해 버티는 각도는 경사각 α에 자세를 똑바로 하기 위해 직각으로 버티는 각도 90를 합한 각도가 된다. 또한 식(1)에서 경사각을 고려하면 보폭과 각도에 의한 다리의 길이는 다음 식(6)와 같다.

$$\begin{bmatrix} X' \\ Y' \end{bmatrix}=l_1\begin{bmatrix} \cos(\Theta_1-\alpha) \\ \sin(\Theta_1-\alpha) \end{bmatrix}+l_2\begin{bmatrix} \cos(\Theta_1+\Theta_2-\alpha) \\ \sin(\Theta_1+\Theta_2-\alpha) \end{bmatrix} \quad (6)$$

여기서 오르막일 경우를 고려하기 때문에 각 관절의 각도에서 경사각 α만큼 빼준 것이 새로운 보폭이 된다. 만약 내리막이라면 경사각의 부호가 바뀌게 됨을 알 수 있다.

4.2 보조와 '보행(Walk)'의 패턴분석

각 클래스가 가지는 움직임에 관한 속성은 [그림 3]과 같이 분류할 수 있다.

각각의 속성은 다리의 위치에 따라 지면에 내딛고 드는 순서를 가지고 있으며 이를 이용하여 최종적으로 네 발로 기다(), 걷다(), 달리다()의 형태로 나타낼 수 있다. 이러한 순차적인 모습은 [그림 7]과 같다[3][4]. 여기서, LF는 왼쪽 앞발 (Left Fore), LH는 왼쪽 뒷발(Left Hind,) RF는 오른쪽 앞발(Right Fore), RH는 오른쪽 뒷발(Right Hind)을 나타낸다. 그림에서 보듯이 각각의 자세에 대한 막대의 형태는 주어진 시간동안 지면에 내딛고 있는 다리의 시간이다. 즉, 시간동안 어느 다리가 지면에 닫고 떨어져 있는가를 3가지의 상으로 나타낸다. 예를 들어, 주어진 시간 t에 기다()의 파라미터로 아래와 같이 추가된다.

$\overline{(\text{Crawl Sequence})}_{T=t}$ = (LF:up, LH:up, RF:down, RH:up)

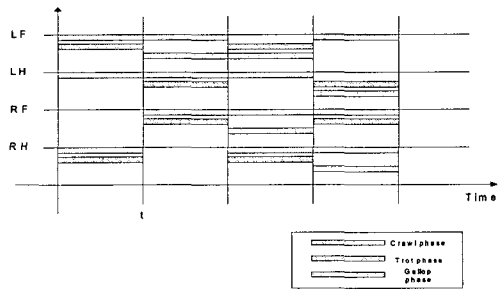


그림 7. 기다(), 걷다(), 달리다()의 순차적 모습

4.2.1 보조와 관련된 요인들

위의 [그림 7]의 경우 다리를 들고 내딛는 동작이 정적인 모습을 볼 수 있다. 반드시 각각의 다리들이 들고 내딛는 동작이 시간동안 정확히 달라야만 한다. 하지만 실세계의 동작은 다리를 들고 내딛는 동작이 교차하면서 바뀌는 모습이 일반적이다. 이를 위해 기존 연구에서는

몇 가지 요인으로 정의하고 이를 이용하여 보다 더 사실적인 모습의 자세를 만들어 내었다[3][4]. [그림 8]에서는 이러한 요인을 이용하여 자세의 순서를 규정하고 있다[3].

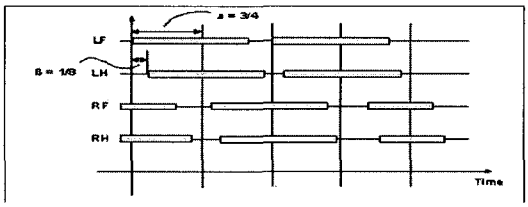


그림 8. 일반적인 보조의 모습

[그림 8]에서 보듯이 기존 연구에서는 필수 요인(duty factor) (α)와 위상차(phase difference) (β)를 정의하고 있다. 필수 요인은 한 주기 동안 다리가 지면에 닫고 있는 비율로 정의하며 다음과 같다.

$$duty\ factor = \frac{rate\ of\ standing\ period}{tatal\ period\ of\ one\ stride\ motion} \quad (7)$$
$$(0 < \text{duty factor}(\alpha) < 1)$$

이를 통해서 각각의 보조를 $\alpha_{crawl} >= 0.75$, $\alpha_{trot} = 0.5$, $\alpha_{gallop} < 0.5$ 로 정의한다[3]. 따라서 식(1)는 다음과 같이 변환된다.

$$X'' = \alpha l_1 \cos\Theta_1 + \alpha l_2 \cos(\Theta_1 + \Theta_2) \quad (8)$$

4.3 예제 : 주어진 지형에 따른 인스턴스의 움직임의 표현

움직임을 상속 받기위해서는 상위클래스가 가지는 기본 모양(primitive shape)과 기본 동작에 대한 식이 필요하다. 식(1)에서 보듯이 기본 모양으로 다리의 상하부 관절의 길이와 이루는 각도를 상속받고, 움직임이 이루어지는 절차를 상속받게 된다. 절차는 움직임을 구성하는 차례에 대한 각각의 도메인을 상속 받는 것을 의미한다. 가장 일반적인 의미의 '보행'을 살펴보면 [그림 9]와 같다[12].

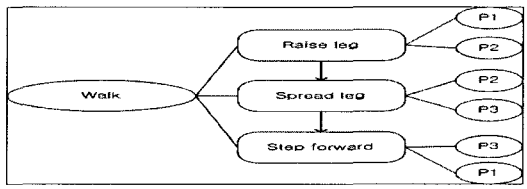


그림 9. '보행'의 절차의 구성부분

'보행'은 다리를 들고, 뻗고, 내딛는 동작으로 구성되며, 이는 또한 해당되는 도메인을 가진다. [표 1]에서는 말의 걷다()의 상속 받는 과정을 나타내고 있다.

표 1. 걷다()의 상속과정

Class	Attributes
Legged Animal	•primitive shape : length = [1,10], angle = [0,180] •procedure : raise = [0,90], spread = [0,180], step = [0,180]
Quadruped	•length = [1,10], angle = [0,180] •raise = [0,90], spread = [0,180], step = [0,180] •number of legs = 4 •sequence of trot gait
Cursorial	•length = [1,5], angle = [170,180] •raise = [0,45], spread = [0,150], step = [0,170] •number of legs •sequence of trot gait
Horse	•length = [1,3], angle = [170,175] •raise = [0,45], spread = [0,150], step = [0,170] •number of legs •sequence of trot gait
Instance of horse	•length = 1, angle = 175 •raise = 40, spread = 145, step = 155 •number of legs •sequence of trot gait

3.2절에서 설명한 것처럼 구체화과정은 도메인의 범위의 축소과정과 자신만의 특이성을 덧붙여 나가는 과정이다. 생성된 최종적인 인스턴스는 식(1)와 식(6) 및 식(7)에 의해 오르막과 내리막인 지형에 적용하여 각각의 보폭과 보조의 패턴에 따른 duty factor를 적용한다. 이러한 순서로 동작에 대한 순서도는 [그림 10]과 같다.

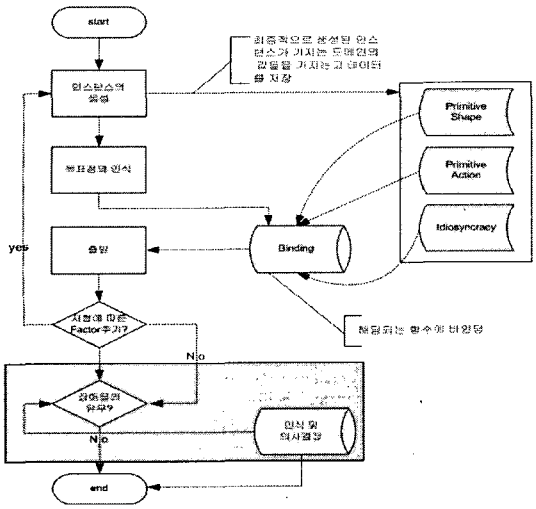


그림 10. 전체 프로세스에 관련된 순서도

다음으로는 지금까지 논하였던 모든 사항을 토대로 한 전체 이동에 대한 알고리즘을 살펴보겠다. 기존 논문에서 소개되었던 좇아가기 기반의 충돌회피 [알고리즘 1]을 응용한 것으로 환경 내에서 속도 변환 행동을 수행하는 알고리즘으로 다음과 같다.

알고리즘 1. 기본적인 보행()

```
function movement_work

   while(goal_position) /*목표점에 도달할때까지*/

   if collision occur estmated then  /*충돌이 있다면*/
      FindEmptyLocation(direction, speed) /*빈공간으로 피함*/
   end if

   if Obstacle_distace <= Critical_distance then  /*장애물이 존재한다면*/
      DecreaseAngularVelocity() /*속도를 감소시킴*/
      FindNewPath(goal_position, current_position)/*새로운 경로를 찾음*/
   end if

      /*상속받은  primitive shape를 이용하여 다음 위치를 예측*/
      project the position in the future using the primitive shape(length, angle)

   end while  /*목표지점에 도착하면 완료*/
end function
```

다음은 예제로 모든 말의 인스턴스의 확장된 알고리즘이다.

알고리즘 2. 확장된 요인을 이용한 말의 이동

```
function movement_walk_trot_of_a_horse
    initial duty_factor then
    if a gait is crawl()
       duty_factor = more than 0.75
    else if a gait is trot()
       duty_factor =  0.5
    else if a gait is gallop()
       duty_factor = less than 0.5
    end if

    if collision occur estimate then /*만약 충돌이 있다면*/
      FindEmptyLocation (direction, speed) /*빈공간으로 피함*/
    end if

    if obstacle_distance <= Critical_distance then /*만약 장애물이 존재한다면*/
      DecreaseAngularVelocity() /*속도를 감소시킴 */
      DecreaseDutyFactor () /*du쇼 factor 값을 감소시킴*/
      FindNewPath(goal_position, current_position) /*새로운 경롤를 찾음*/
    end if

    /*상속 받은 primitive shape를 이용하여다음 위치를 예측
      primitive action 과 primitive shape 및 해당 gait의 시퀀스를 이용 */
    project the ppsition in the future using _
    the  primitive  action(primitive  shae(length,angle),  sequence  of
       crawl)*duty factor

    end while /*목표점에 도달하면 완료*/
  end fuction
```

V. 구현 및 실험결과

앞에서 제시한 이론들을 바탕으로 가상환경에서 4족 동물의 움직임에 대한 구현을 할 수 있었다. 여기에 등장하는 4족 동물은 자신만의 고유한 요인들과 움직임에 관련된 요인들을 통해서 앞서 제시한 3가지 행동 양상을 표현한다. 시뮬레이션 시스템은 윈도우 환경에서 Visual C++ 6.0과 OpenGL API를 응용하여 제작하였으며, 세밀한 그래픽보다는 자세를 다양한 측면에서 확인할 수 있도록 3차원 그래픽으로 구현하였으며, 텍스쳐링 기법은 생략하였다.

그림 11. 시뮬레이션 화면(달리다()의 경우)

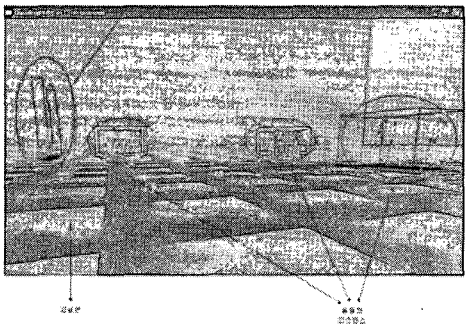


그림 12. 게임 엔진에 올린 시뮬레이션 화면

[그림 11]에서는 하나의 4족 동물 인스턴스의 기다(), 걷다(), 달리다()의 동작을 보여주는 화면이다. [그림 12]는 기존에 공개된 게임 엔진[14]에 생성된 각각의 인스턴스를 올린 화면이다. 이는 하나의 환경에 여러 인스턴스가 공존하는 모습을 보여주는 것이며 여기에는 장애물과 서로간의 충돌에 대한 검출을 AABB 방법에 의해서 판정한다.

5.1 기본 구조

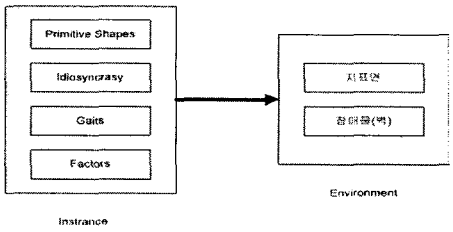


그림 13. 모델링 구조

[그림 13]은 전체적인 시스템의 구조를 나타낸 것이다. 최종적인 인스턴스는 공통적인 속성으로 가지는 기본 모양과 특이성(Idiosyncrasy), 걷는 모양(Gaits)와 그 외의 요인들을 사용하여 인스턴스를 생성해 낸다. 이렇게 생성된 인스턴스는 지표면과 장애물을 가지는 환경에 위치시키고 그들 상호간의 충돌과 회피를 통해 공존해 나가는 모습을 확인할 수 있다.

5.1.1 객체 계층 구조(Object Hierarchy)

본 게임의 엔진에 적용되는 객체(Object)의 계층구조는 다음과 같다[14].

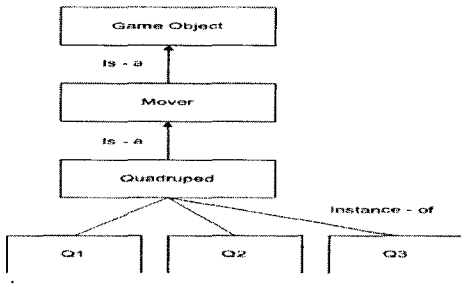


그림 14. 객체 계층구조

가장 일반적인 기본 클래스는 게임 객체 클래스이다. 이것은 모든 게임 내의 개체들을 포함하는 클래스이며 위치와 방향을 가진다.

움직임(mover) 클래스는 움직임에 관련된 벡터와 타이머 등의 기능을 가지는 클래스이다. 이 클래스는 특정한 하나의 캐릭터에 대한 기능을 포함한다. 최종적으로 4족 동물 클래스는 상위의 모든 클래스들을 상속받은 하나의 4족 동물의 클래스로서 Q1, Q2, Q3,... 등의 여러 가지 최종적인 인스턴스를 생성시키는 클래스가 된다. 이를 통해서 우리는 최종적인 인스턴스의 모습을 확인할 수 있다.

5.2 충돌검출

구현에서 여러 인스턴스들은 서로간의 충돌과 벽 같은 장애물에 충돌을 확인하고 거기에서 피해나가는 방법으로 AABB(Axis Aligned Bounding Box)를 이용한다. 이 방법은 서로가 충돌을 일으킬 가능성이 있다면 가장 빠르게 결정할 수 있는 방법을 제공한다[14].

AABB는 한 오브젝트에 대해 x,y,z축에 각각 평행한 최소 점과 최대 점을 구하여 연결시킨 6면체를 사용한 충돌체크 방식이다. 다시 말하면 오브젝트의 모든 정점을 조사하여 각 정점들의 세 개의 선분(x,y,z)의 최대, 최소 값을 구해서 정육면체를 만드는 것이다. 이렇게 되면 그 정육면체에 부딪히면 충돌이 되는 것이다. 울퉁불퉁한 물체에 AABB 방식을 쓰면 충돌체크가 어색하게 되나 집이나 정육면체에 가까운 물체에 AABB를 쓰면 사실적인 충돌체크가 된다.

5.3 구현 및 결과

5.3.1 걷다()

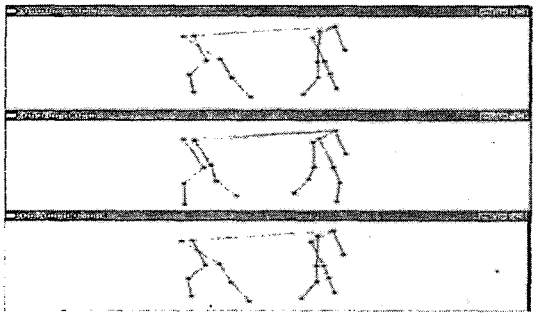


그림 15. 걷다()의 경우

대표적인 인스턴스를 생성시켜 거기에 3가지 타입의 보조에 대한 결과를 확인 할 수 있었으며 그 예로서 걷는다()의 경우는 [그림 15]와 같다.

위의 경우에서 보조의 필수 요인은 0.5, 걷는 모양의 타입은 걷다() 이며 각각의 보조에 해당하는 값은 4. 1절의 식(1)에 의해 보복이 결정된다. 이를 통해 해당되는 '보행'의 타입과 해당되는 요인을 이용하여 적절한 움직임에 대한 결과를 도출해 낼 수 있었다. 이렇게 생성된 각각의 인스턴스들을 환경을 가진 게임 엔진에 위치시켜 볼 수 있었다.

5.3.2 지형에 따른 움직임의 경우

가상환경에서의 지형은 대표적으로 오르막 및 내리막을 예로 들 수 있다. 대표적인 두 가지 형태의 지형을 이용하여 움직임을 적용하여 보았다. 그 결과는 아래 [그림 16]과 같다.

오르막과 내리막이 연속으로 있는 지형을 걷다()의 경우를 적용하여 보았다. 아래 그림처럼 오르막의 각도에 따른 보폭과 자세가 식(6)와 같이 결정되고 접촉면을 따라 이동을 하게 된다. 내리막도 마찬가지로 이와 동일하며 마지막 그림에서는 장애물이 있으면 달리다() 동작을 이용하여 넘어가는 모습을 보여주고 있다.

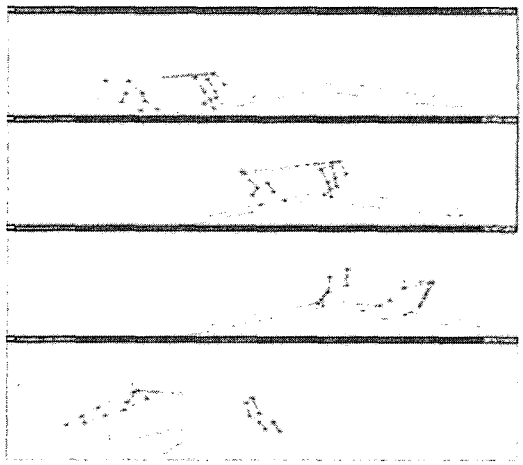


그림 16. 오르막과 내리막의 경우와 장애물을 넘는 경우

V. 결론 및 향후 과제

우리는 4족 동물의 '보행'을 모델링하여 이를 새로운 하위계층 관계로의 용이한 확장과 모델링을 간소화하여 표현하는 방법을 알아보았다. 이를 위해 기존의 연구에서 분석한 패턴과 분류방법을 이용하여 일반화하여 계층관계를 구성하고 각 계층이 가지는 파라미터와 기본 동작의 상속방법을 제시하였다. 이러한 방법을 통해 하위계층의 4족 동물의 '보행'의 표현을 쉽게 하고 새로운 하위계층의 인스턴스가 추가되었을 때 상위계층의 특성을 상속받아 용이하게 확장할 수 있었다. 4.3절에서 보듯이 말의 인스턴스를 생성함에 있어서 달릴 수 있는 동물의 특징을 동일하게 사용하고 하위계층 각각의 특징적인 값들의 조정에 의한 인스턴스의 생성이 가능했다. 결과적으로 새로운 개라는 클래스를 생성한다면 역시 기존의 계층관계에서 가지는 특징을 상속받고 개만의 특이성만 고려한다면 쉽게 표현할 수 있음을 알 수 있다.

본 논문에서는 4족 동물에 국한한 인스턴스의 생성과 움직임의 표현에 중점을 두었다. 가상세계의 다양한 객체들의 움직임을 표현하기 위해서는 다리의 개수가 없는 것부터 여러 개의 수준까지를 포괄 할 수 있는 모델링 기법이 필요할 것이다. 이를 위해서 본 논문에서 제시한 기본 동작과 상속 및 일반화 기법을 이용하고 공통된 속성

을 더 많이 규명하는 작업이 필요할 것이며, 이를 포함하는 계층관계를 구성한다면 수많은 객체를 표현 할 수 있는 용이한 방법론이 될 것이다.

참 고 문 헌

[1] A. J. Champandard 저, 이강훈 역, *인공지능 게임 프로그래밍 실전가이드:최신 AI기법을 적용한*, 에이콘 출판사, 2005.

[2] 이제희, "모션캡쳐의 과거, 현재, 그리고 미래", 한국정보과학회지, 제21권, 제7호, pp.39-43, 2003.

[3] I. Shimoyama, H. Miura, M. Horita, and Y. Hattori, "Analysis of Walk for Quadruped," Intelligent Robots and Systems '91. 'Intelligence for Mechanical Systems, Proceedings IROS '91. IEEE/RSJ International Workshop on, Vol.3, No.1, pp.1541-1544, Nov., 1991.

[4] K. INAGAKI and H. KOBAYASHI, "A Gait Transition for Quadruped Walking Machine," Intelligent Robots and Systems '93, IROS '93. Proceedings of the 1993 IEEE/RSJ International Conference on, Vol.1, No.1, pp.525-531, July, 1993.

[5] B. Meyer, "The many faces of inheritance: a taxonomy of taxonomy," IEEE Computer, Vol.29, No.5, pp.105-108, May, 1996.

[6] R. Mistry and G. Clapworthy, "Computer-Based Animation of a Multi-legged Articulated Body," Information Visualization, 2000. Proceedings. IEEE International Conference on, Vol.5, No.1, pp.315-317, July, 2000.

[7] S. Joung and J. Tanaka, "Icon-based Animation From The Object And Dynamic Models Based On Onit," Computer Human Interaction, 1998. Proceedings. 3rd Asia Pacific, Vol.3, No1, pp.473-474, July, 1998.

[8] 지세진, *Structured Causal Graph에 기반한 이벤트 전개 모델*, 경북대학교 석사 학위논문, 2001.

[9] M. S. Kim and E. T. Lee, "A visual interface for scripting virtual behaviors," Computer Human Interaction, 1998. Proceedings. 3rd Asia Pacific, Vol.3, No.1, pp.165-168, July, 1998.

[10] J. Park, "A logical simulation of discretionary events in spatio-temporal context," Tech. report #3, AIMM lab, Kyungpook Nat'l Univ., 2000.

[11] 권정우, *Juvenile Energy를 이용한 가상세계에서의 식물의 성장 모델링*, 경북대학교 석사 학위논문, 2003.

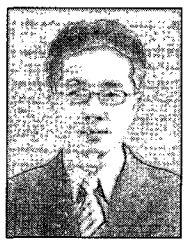
[12] J. Park, *"Actions,"* Tech. report #94, AIMM lab., Kyungpook Nat'l Univ., 2005.

[13] http://www.oricomtech.com/projects/legs.htm#Gait2

[14] http://home.planet.nl

저 자 소 개

이 인 균(In-Kyun Lee) 정회원



- 2002년 2월 : 영남대학교 기계공학부(공학사)
- 2005년 8월 : 경북대학교 전자공학과 정보통신전공(공학석사)
- 2005년 8월~현재 : (주)휴원 솔루션사업부 근무

<관심분야> : 인공지능, 인간과 컴퓨터 상호작용, 시뮬레이션 분야

박 종 희(Jong-Hee Park) 정회원

- 1979년 : 서울대학교(공학사)
- 1981년 : 한국과학원(공학석사)
- 1990년 : Univ. of Florida(공학박사)
- 현재 : 경북대학교 전자공학과 교수

<관심분야> : 멀티미디어 응용, Computer Aided Education, CAD/CAM, 지능형 정보시스템, 분산 데이터 처리 시스템