캐릭터 동작 애니메이션 제어를 위한 에이전트 시스템

(An Agent-based System for Character Motion Animation Control)

김 기 현 † 김 상 욱 ††

(Kihyun Kim) (Sangwook Kim)

요 약 사용자가 하나의 캐릭터 이상을 애니메이션 하기를 원할 때 캐릭터들 사이에 충돌과 같은 기대되지 않은 동작 애니메이션을 생성할 수 있다. 그러므로, 이러한 문제가 적절한 제어 메커니즘을 이용하여 해결되어야 한다. 본 논문은 사용자의 의도를 반영한 애니메이션 시나리오을 표현하기 위해 캐릭터의 동작 애니메이션을 제어하는 에이전트 기반 시스템을 제안한다. 이 시스템은 3차원 공간상에서 캐릭터가 움직이는 경로에 따라 캐릭터들간의 충돌을 회피하고 동작의 형태를 조정하는 방법을 제공한다. 에이전트는 동작을 동기화하기 위해 다른 에이전트와 통신한다. 에이전트는 캐릭터의 동작을 조정하는 여러 지능적인 에이전트로 확장되어진다. 에이전트 시스템은 의도된 동작 애니메이션 뿐만아니라 전체 캐릭터 애니메이션에 대한 동작의 스케쥴링을 가능하게 한다. 에이전트들의 정보를 전달하고 에이전트들의 현 상태를 추론하는 방법으로써 에이전트의 대화를 위한 페트리넷 분석을 이용하여 오토마타 모델을 디자인한다. 에이전트 기술을 이용하여 캐릭터의 동작을 제어하기 위한 에이전트 시스템을 구현한다. 인체 모델 캐릭터의 동작을 제어하는 예를 보이고, 동작 제어의 가능성을 보인다.

Abstract When user wants to animate more than one character, some unexpected motion animation like a collision between characters may occur. Therefore, this problem must be resolved using a proper control mechanism. This paper proposes an agent-based system that controls the motion animation of the character for representing animation scenario reflecting user's intention. This system provides a method that coordinates a type of motion and avoids collision between characters according to the moving path of a character in three-dimensional space. Agent communicates with others for motion synchronization. Agent is extended into several intelligent agents that coordinate character's motion. Agent system enables not only an intended motion animation, but also the scheduling of motion to an entire character animation. It designs automata model using Petri-net analysis tool for the agent's interaction as a method that passes the agent's information and infers the current state of agents. We implement this agent system to control the motion of character using agent technology and show an example of controlling the motion of human character model to prove the possibility of motion control.

1. 서 론

에이전트란 일반적으로 사용자를 대신하여 주어진 작업을 수행하는 독립적 컴퓨터 프로그램이라고 정의 할 수 있다[1][2]. 에이전트 기술은 다양한 분야에서 적용되고 있다. 특히, 게임 캐릭터와 주변 환경 사이의 자동화된 상호작용, 아바타 어플리케이션, 스토리보드 디자인을 위한 3차원 그래픽 에이전트, 만화, 광고를 애니메이션하는 곳에 적용될 수 있다.

캐릭터 애니메이션을 표현하기 위해서 여러 가지 세부적인 기술적 알고리즘이 적용된다. 그 대표적인 예로 키프레임, 운동학, 역운동학 등과 같은 방식에서는 인체 모델 캐릭터와 같은 대상체를 이용하여 각 신체 부위의

† 학생회원 : 경북대학교 컴퓨터과학과
kimkh@woorisol.knu.ac.kr
†† 비 회 원 : 경북대학교 컴퓨터과학과 교수
swkim@cs.knu.ac.kr
논문접수 : 2001년 2월 19일
심사완료 : 2001년 7월 25일

위치값을 설정해서 동작 애니메이션을 표현하거나 하나의 캐릭터 대상만을 중점적으로 디자인하는 문제점이 있다[3]. 그림 1과 같이 3차원 공간상에서 여러 캐릭터간의 동작 애니메이션을 실행할 경우 각 캐릭터간의 충돌 문제나 이동 경로에 따른 자연스러운 애니메이션 연출을 표현하기 위한 제어 기법이 부족하다. 인체 캐릭터 모델의 동작 변이를 위해 시간, 공간적 제약 조건을 이용하여 동작 데이타를 조작하도록 허용하는 동작 언어 해석기를 사용하는 경우 반복적 데이타, 자연스러운 동작 변이를 가능하게 한다[4]. 그러나, 여러 캐릭터의 동작 애니메이션에 대한 동작 변이를 스케쥴링 할 경우는 많은 동작 상황과 동작을 스케쥴링 해야하는 단점을 가지고 있고 캐릭터간의 상호작용적인 면[5]을 통한 다양한 동작 애니메이션의 제어와 표현력이 부족하다. 에이전트 기술을 적용한 인체 모델 캐릭터 에이전트의 동작 제어에 관한 연구에서 실시간 인체 모델 캐릭터 에이전트 제어를 위해 스크립트 언어를 이용하는 IMPROV시스템은 계층화된, 동작간의 전이를 가능하게 하기 위해 절차적 기술을 사용한다. 동작 애니메이션의 디자인은 텍스트 스크립트 언어를 이용하여 동작을 정의하고 사용자 관점의 환경을 제공한다[6]. 그러나, 동작의 전체 이동경로에 대한 정보의 표현력과 캐릭터간의 동작 종류에 따른 동작 제어의 다양성을 제공하지 못한다. 에이전트 객체와의 상호작용을 모델링하고 제어하기 위해 스마트 즉, 객체 자신이 가능한 상호 작용을 묘사하기 위한 능력을 가지는 개념을 이용하여 스크립트된 명령어를 통해 미리 정의된 계획을 사용하여 캐릭터 에이전트의 동작 애니메이션을 디자인한 경우 각 에이전트가 제어 모듈을 가지고 객체의 행동을 해석하는 하나 이상의 에이전트를 동기화 할 수 있는 방법을 제안한다. 행동 정의를 위한 키워드의 정의와 미리 계획된 절차를 사용하므로 가상 환경의 변이에 따른 제어와 다른 캐릭터 에이전트와의 상호 작용을 인식하기 위한 모듈성이 부족하다[7]. 캐릭터 에이전트의 동작 애니메이션을 생성하고 제어하기 위해 L 시스템과 같은 생성 규칙을 적용한 시스템의 경우 센서에 기반하여 에이전트간 내부 통신을 통한 동작 애니메이션 제어를 가능하게 한다. 또한, 동작 애니메이션 제어를 위한 생성 규칙을 사용자가 편집 할 수 있다. 그러나, 여러 캐릭터 에이전트간 대화를 통한 캐릭터 동작간 거리의 제어와 우선 순위에 따른 다양한 동작 시나리오의 표현이 어렵다[8]. 지능적, 자율성의 에이전트 속성을 제공하는 메커니즘의 경우 에이전트 자원을 관리하기 위해 구조적인 방법을 제시한다. 동적인 환경 변화에 대응할 수 있는 방안을 제공한다. 에이전트간 동작의 제어보다는 동작 생성의 제약 조건에 관심을 두고 있어 여러 캐릭터 에이전트간의 대화를 통한 제어 관계가 결여되어 있다[9].

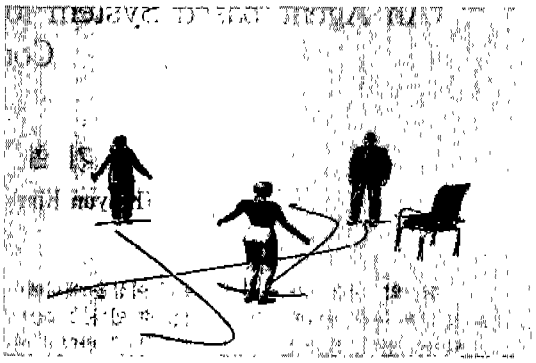

그림 1 캐릭터간의 동작 시나리오

본 논문은 캐릭터의 동작에 대한 사실적 묘사에 중점을 두는 것이 아니라, 그림 1과 같이 여러 동작 시나리오에 따라 캐릭터 각각에 대한 애니메이션 동작을 사용자 의도에 따른 동작 시나리오 형태로 전체적인 의미로 해석하여 표현하고 제어하기 위해선 단순히 하나의 캐릭터 동작 제어에 치중하는 것이 아닌 다른 여러 캐릭터들 간에는 상호 동기적으로 캐릭터 동작 애니메이션을 제어할 수 있는 요소가 필요하다. 여기서는 이러한 의도로 3차원 공간상에서 캐릭터간의 동작 표현을 원활히 제어하고 지원하기 위해 에이전트 기반 기술을 이용한 여러 캐릭터의 애니메이션 시나리오에 따른 동작 제어에 적용해 본다. 그 대상으로 인체 모델 캐릭터의 동작 애니메이션에 따른 제어에 에이전트 기술을 이용한다.

이러한 각 캐릭터간의 동작 표현 제어가 캐릭터 상호간의 의사 전달을 위한 통신과 언어학적 측면, 캐릭터의 동작 애니메이션을 위한 조정 서비스와 같은 에이전트 속성을 이용하여 여러 캐릭터간의 다양한 동작 제어가 가능하다.

본 논문의 제 2절에서 캐릭터 애니메이션 제어를 위한 에이전트 시스템에서 필요로 하는 에이전트의 형태와 속성을 정의한다. 제 3절에서는 캐릭터 동작 에니메이션 제어를 위한 에이전트 시스템 프레임워크와 특징을 설명한다. 제 4절에서는 에이전트간의 대화 규칙을 표현하기 위한 페트리 넷 모델을 설명하고 그 예를 보인다. 제 5절에서는 에이전트 시스템을 이용한 여러 인체 모델 캐릭터의 동작 애니메이션 제어에 대한 구현 예를 보인다. 결론과 향후 연구 방향을 제 6절에서 제시한다.

2. 에이전트 형태와 속성

본 절에서는 개별 에이전트가 문제와 해결을 위한 지식을 공유하고 다른 에이전트와의 조정과정을 통해 추론해야 하는 일반적인 에이전트가 가져야 할 특징과 속성을 제외한 애니메이션을 디자인 할 경우 각 캐릭터간의 동작 애니메이션 제어를 위해 요구되는 에이전트가 가져야할 형태와 속성에 대해 언급한다. 에이전트는 쓰레드 형태로 각각 실행되며 동작 애니메이션에 관련된 지식, 목표, 계획을 조정하는 지능적 에이전트를 형성한다. 3차원 공간상에 존재하는 캐릭터(인간과 같은 형태의 동물 등), 정적인 주변 환경 객체가 에이전트의 개체로 분류된다. 여러 에이전트들은 동일 3차원 공간상에 동시에 존재하고 실행된다. 에이전트는 캐릭터와 주변의 환경 객체간에 정보를 교환하여 이동 경로에 따른 동작 표현의 내부적 제어를 가능하게 한다.

2.1 에이전트 형태

(1) 인터페이스 에이전트

- 사용자에 의해서 받아들여진 행동을 관찰하고 모니터한다.
- 각 캐릭터 에이전트간의 상태를 보여주기 위한 인터페이스
- 각 에이전트 파라미터를 보여주며 에이전트가 가지는 현 동작 타입 정보를 나타낸다.

실시간으로 각 에이전트의 초기화와 사용자의 요구조건을 만족하기 위해 에이전트 시스템과 메세지를 교환[10]함으로써 캐릭터의 동작 형태에 따른 애니메이션 제어를 요구하게 된다.

(2) 정보 에이전트

에이전트를 위해 요구된 동작 형태와 주변 환경에 대한 자원 정보를 전달하는 역할. 현재의 캐릭터 에이전트가 의사소통을 하기 원하는 다른 캐릭터 에이전트에게 관한 속성 정보와 메세지 타입을 가진다. 요청된 모든 자원 목록을 유지하고 에이전트가 어떤 자원을 요구할때마다 관련된 자원 목록을 생성한다. 이러한 요구성을 그룹으로 등록된 각 캐릭터 에이전트에게 전달한다. 에이전트 리스트에 그 자원 정보를 저장하고 탐색할 수 있다.

(3) 그룹 에이전트

에이전트가 서로 의사소통을 원하는 대상을 그룹화하여 각각의 쓰레드 형태로 관리함으로써 서로간의 정보를 교환한다.

(4) 캐릭터 에이전트

캐릭터의 대상을 움직이는 객체와 정적인 객체로 구분한다.

- 능동적 에이전트: 현재 동작 애니메이션의 중심이 되는 캐릭터 에이전트. 능동적 에이전트에 대해서 두가지 주요 기능을 가지고 있다. 첫째, 관련된 그룹 객체에 메세지를 연속적으로 보내게 되고 둘째, 그룹에 등록한 모든 에이전트가 이들 메세지를 받도록 그 그룹에 대해 메세지를 알리게 된다. 이들 두 함수는 메시지를 보내는 시간과 받는 시간을 가지고 메시지안에 타임스탬프를 가지고 등록하게 된다.
- 수동적 에이전트: 능동적 에이전트와 상호 의사 전달을 하여 능동적 에이전트의 특별한 동작 요구에 반응하여 자신의 동작 애니메이션의 진행속도와 시간을 조절하는 에이전트

(5) 환경 에이전트

3차원 공간상에 있는 캐릭터 에이전트에 대한 주변 객체와의 현 상태에 따른 상호 관계를 모니터링하고 상호 작용을 하기 원하는 대상에 대한 메시지 처리를 담당한다.

(6) 탐색 에이전트

각 에이전트 대상(인체모델, 동물, 정적인 객체)에 따라 적용되는 동작 형태와 행동 타입을 지정, 여기서는 키프레임과 역운동학을 적용하여 동작 형태를 정의한다. 동작의 종류는 데이타 베이스화하여 인덱싱 형태로 호출한다.

2.2 에이전트 속성 타입

(1) 동작 정보: 각 캐릭터 에이전트 대상에 따른 동적인 동작정보

- 에이전트 식별자: 에이전트 대상에 따라 주어지는 식별자
- 위치 정보: 에이전트의 공간적 정보
- 시간 정보: 각 캐릭터 에이전트 동작 프레임수에 따른 시간 정보

(2) 동작 속성: 각 캐릭터 에이전트 동작 형태를 정의하고 디자인(예: 달리다. 걷다. 흔들다. 등)한다.

- 대화 리스트: 에이전트 안에 있는 각 에이전트의 현 상태를 나타내는 대화 메시지 형태의 목록들. 여기서 대화는 어떤 목표를 달성하기 위한 에이전트의 계획을 의미한다. 즉, 다른 에이전트와의 상호작용에 근거하여 최종 동작 애니메이션 계획이 다른 캐릭터 에이전트와의 동작 형태와 조건에 따라 결정된다.
- 자원 리스트: 각 캐릭터 에이전트가 표현할 동작의 형태에 관한 자원목록(예: 달리다, 걷다. 뛰다 등)

(3) 메세지 형태 : 대화에 따른 에이전트사이의 메세지 타입을 정의한다.

-그룹: 현 캐릭터의 동작 애니메이션에 대한 정보와 제어를 전달하기 위한 다른 캐릭터를 그룹화하여 서로 통신한다.
-일대일 통신: 능동적 캐릭터 에이전트가 직접 정보를 전달하여 다른 수동적 캐릭터의 동작 제어를 가능하게 하기 위한 인터페이스
-메시지 라우터: 각 에이전트로 전달되는 대화의 메시지를 처리하고 운영하기 위한 메시지 라우터 역할

(4) 에이전트 목록: 현 능동적 캐릭터 에이전트가 자신의 동작 애니메이션 형태를 전달하고자 하는 다른 수동적 에이전트의 정보를 가지고 있다.

3. 동작 제어를 위한 에이전트 프레임워크

본 절에서는 캐릭터의 동작 애니메이션 제어를 위한 에이전트 시스템 프레임워크(MOCAF)에 대해서 설명한다. 이러한 에이전트 프레임워크가 기존 에이전트 이론 방법에 따라 캐릭터 애니메이션의 동작 제어에 어떻게 적용될 수 있는지를 묘사한다.

3.1 에이전트 프레임워크의 특징과 개념모델

에이전트 프레임워크란 단일 에이전트가 해결하기 어려운 복잡한 문제를 여러 에이전트들이 상호 협력하여 해결할 수 있도록 모든 에이전트간의 상호 연결을 책임지는 하부구조이다. 캐릭터 에이전트 시스템은 에이전트 생성과 처리, 각 캐릭터의 동작 애니메이션 형태를 결정하고 에이전트 타입과 속성, 이름, 에이전트가 동록하는 그룹 정보를 생성, 관리한다.

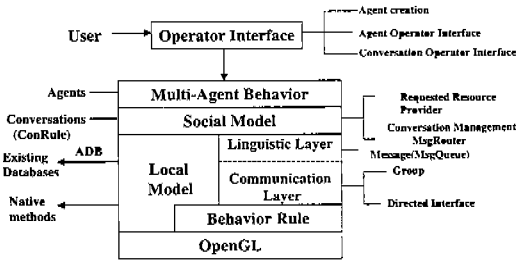

그림 2 프레임워크와 클래스들의 형태

그림 2는 계층화된 구조로 서비스를 분할함으로써 각 캐릭터의 동작 애니메이션을 위한 통신과 상호작용, 조정 등을 제공하게 된다. 그림 2는 여러 계층을 구성하는 구조와 클래스를 보여주고 있다.

(1) Opeator 인터페이스는 사용자에게 에이전트에 대한 모든 필요한 정보를 표시한다. 즉, 에이전트의 생성, 모니터링, 대화 상태를 표시하는 역할을 한다. 캐릭터 에이전트가 존재하는 환경정보와 동적인 변화 상태를 보여준다.

(2) 로컬 모델은 사용자가 디자인하는 애니메이션 동작 형태와 정보를 구성하는 역할을 하게 된다. 그림 3은 에이전트의 동작을 계획하기 위한 지식베이스와 제어단위를 나타낸다.

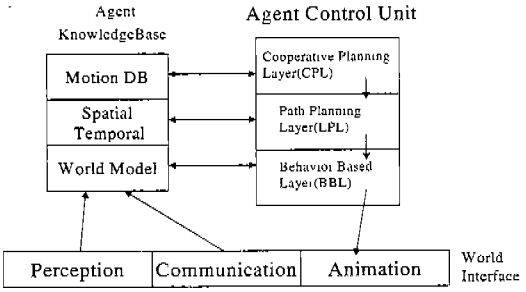

그림 3 로컬 모델

(3) 통신 계층은 에이전트의 행동을 조정하고 서로 협력하기 위한 정보 교환을 가능하게 한다. 프레임워크 상에서 그들의 통신언어(서로 상호작용하기 위해)에 의해 사용되는 여러 통신 모드를 나타낸다. 통신 모드는 직접 및 간접 형태로 각 캐릭터 에이전트간의 메시지 전송을 담당한다. 에이전트의 통신이 보장될 때 에이전트간의 동작 애니메이션 생성 규정은 대화 규칙에 따라 상호 조정된 협동적인 양식에서 서로 상호작용하기 위해 제어된다.

(4) 에이전트의 사회적 모델은 통신계층상에서 확립되어진다. 각 에이전트간의 언어적, 조정지원을 제공하는 서비스가 확립되면 에이전트를 생성하기 위해 가용한 모든 필요한 컴포넌트들을 가지게 된다. 에이전트의 Social 행동은 일관적 해결책을 야기하기 위해 시스템의 다른 에이전트와 통신하거나 상호작용, 조정하기 위한 방법을 제안하는 의미를 가지고 있다. 이것은 에이전트 시스템을 위한 프레임워크의 구성에서 가장 중요한 면중의 하나이다.

(5) 언어학적 레이어[11]는 에이전트의 내부 자료구조와 알고리즘에 독립되는 메시지 기반 인터페이스를 제공하는 에이전트 독립적인 시멘틱을 제공한다.

이러한 캐릭터 에이전트 시스템 프레임워크는 에이전트 기술을 기반으로 여러 캐릭터의 동작 생성과 제어에 효율적으로 사용되어질 수 있다.

4. 에이전트간 대화 제어 모델

각 캐릭터가 공간상에서 어떤 동작을 표현하는 행동을 제어하기 위해선 규칙 기반(예: 그림 1의 어떤 하나의 능동적 캐릭터가 어떤 형태의 동작 애니메이션을 할 경우 다른 나머지 캐릭터가 반응해야 할 동작 애니메이션 상태에 대한 규칙)형태를 가져야 한다. 이것은 각 캐릭터 에이전트간의 동작 생성 형태에 따른 확장성을 제공하기 위해 사용된다. 규칙 기반 형태로 각 캐릭터 에이전트간의 사회적 행동을 나타내고 관련된 에이전트와의 상호작용을 한다. 이러한 규칙성을 가진 대화를 통해 각 캐릭터 에이전트는 상호 동의된 규칙 관습에 따라 동작을 표현하기 위한 메시지를 교환하고, 현 상황에 따른 동작 상태를 변경하고 동작 애니메이션을 수행한다. 각 캐릭터 에이전트는 자신의 실행 쓰레드를 가지고 있으며, 동시적으로 다른 캐릭터 에이전트에 대한 정보를 가지고 실행되며 그들을 스케쥴하는 방법에 대하여 관여함이 없이 다른 캐릭터 에이전트와의 대화를 통해 동작 생성 시간과 형태를 제어 할 수 있다. 모든 캐릭터 에이전트간의 대화는 다른 캐릭터 에이전트와 문맥상에서 동시적으로 실행된다. 캐릭터 에이전트의 동작 정책을 지정하기 위해 대화 규칙을 사용한다. 대화는 다른 에이전트와의 상호작용에 기반한 어떤 목표를 달성하기 위한 에이전트의 계획이다. 즉, 다른 에이전트와의 상호작용에 근거하여 동작 애니메이션에 대한 각 캐릭터의 동작을 스케쥴하고 제어하기 위한 규칙을 나타낸다. 대화 규칙은 다른 에이전트의 동작 형태와 조건에 따라 오토마타 모델의 관점에서 시각화될 수 있다. 이것은 논리적 일관성을 제시하고 각 에이전트에 대한 동작 애니메이션의 일관된 분석을 허용한다.

4.1 동작 제어를 위한 제어 모델

에이전트의 동작 제어를 위한 제어 모델 기법을 설명한다. 동작의 모든 제어 상태를 표시하기 위해 에이전트 시스템에서 다른 에이전트와의 모든 가능한 상호 작용의 대화 규칙 상태를 유한 상태 모델로 표시한다. 이것은 여러 실행조건을 식별하고 대화의 각 규칙을 위한 동작 애니메이션 타입을 식별한다. 각각의 대화는 오토마타 모델에 의해 표현되어진다. 캐릭터 에이전트의 이동경로에 따른 여러 동작 애니메이션은 다시 세부적 동작으로 분류된다. 대화의 현 상태가 에이전트의 다음 동작 애니메이션에 영향을 준다. 계획의 최종 상태는 계획의 실행이 종료되어지는 상황을 묘사한다. 그러므로, 대화란 에이전트가 어떤 상황에서 무엇을 해야할지에 대한 규칙 기반 묘사를 나타낸다. 에이전트의 여러 행위는 규칙에 대한 실행 조건을 이용하여 식별한다. 이것은 각 대화를 위한 대화 규칙을 식별하도록 도움을 준다. 이러한 대화 모델을 분석하고 일관성을 위한 도구로 본 논문에서는 페트리넷 모델을 이용하여 캐릭터 동작 애니메이션 제어를 위한 기법으로 사용한다. 이것은 대화 규칙을 이용하여 에이전트의 계획을 시각화 할 수 있다는 장점이 있다. Speech-act 기반[12] 메시지를 통해 서로 상호작용하여 유한 상태 머쉰 형태로 디자인된다. 페트리넷은 정보 흐름의 추상적이고 형식적인 모델이다. Places와 Transitions 네트워크 형태로 4개의 튜플 PN= (P, T, IN, OUT)로 구성된다. P={p1, p2, p3 .. pn}은 Places(조건 혹은 자원)의 집합, T={t1, t2, t3 .. tn}은 transitions(각 에이전트에서의 모델 행동)의 집합을 나타낸다. 페트리넷은 순차적 실행, 병렬적, 동기화, 계층적인 모델링을 가능하게 한다. 이러한 페트리넷 모델을 이용하여 캐릭터의 각 동작 애니메이션에 따른 상태 변화를 표현 할 수 있다.

p1: state 'start' of character 1, p2: state 'reject' of character 1
p3: state 'reject' of character 1, p4: state 'start' of character 3
p5: state 'reject' of character 3, p6: state 'reject' of character 3
p7: state 'start' of character 2, p8: state 'reject' of character 2
p9: state 'reject' of character 2, p10: state 'accepted' of character 1
p11: state 'accepted' of character 2, p12: state 'accepted' of character 3

그림 4 Places

t1: character 1 starting for a static object and transmitting message of type 'propose execution' to character 3
t2: character 3 receives a message of type 'propose execution' from the character 1, it transmits a message of type 'reject' to character 1 and character 1 can't start or enter into waiting state.
t3: character 3 starting for a static object and transmitting message of type 'propose execution' to character 2
t4: character 2 receives a message of type 'propose execution' from the character 3, it transmits a message of type 'reject' to character 3 and character 3 can't start or enter into waiting state.
t5: character 2 starting for a static object and transmitting message of type 'propose execution' to character 3
t6: character 3 receives a message of type 'propose execution' from the character 2, it transmits a message of type 'reject' to character 2 and character 2 can't start or enter into waiting state.
t7: character 1 starting for a static object and transmitting message of type 'propose execution' to character 2
t8: character 1 receives a message of type 'propose execution' from the character 3, it transmits a message of type 'reject' to character 3 and character 3 can't start or enter into waiting state.
t9: character 3 starting for a static object and transmitting message of type 'propose execution' to character 1
t10: character 2 starting for a static object and transmitting message of type 'propose execution' to character 1
t11: character 2 receives a message of type 'propose execution' from the character 1, it transmits a message of type 'reject' to character 1 and character 1 can't start or enter into waiting state.
t12: character 1 receives a message of type 'propose execution' from the character 2, it transmits a message of type 'reject' to character 2 and character 2 can't start or enter into waiting state.
t13: character go to a static object.
t14: solution found, begin again.

그림 5 Transitions

그림 6은 3차원 공간상에 3명의 인체 모델 캐릭터와 1개의 정적 객체가 있는 경우 3명의 캐릭터가 1개의 정적 객체를 목표로 3차원 공간상에서 이동하며 동작 애니메이션을 실행할 경우 3명의 각 인체 모델 캐릭터 에

이전트들의 상호관계를 나타내는 동작 애니메이션 제어를 위한 페트리넷 모델의 시각적 구조 형태를 나타낸다. 그림 4와 5는 그림 6의 페트리넷 모델에 대한 Places와 Transitions을 나타낸다. 이러한 Places와 Transitions을 통해서 각 에이전트간의 대화를 분석하여 캐릭터간 동작 애니메이션을 제어하기 위한 규칙을 전달하고 생성하게 된다. 여기서 P1에서 P12는 각 캐릭터 에이전트의 조건에 따른 결과 상태를 나타내고, t1에서 t14는 각 에이전트가 취해야 할 동작 실행 조건 규칙을 의미한다. 그림 6에서 P1, P4, P7은 3명의 인체 모델 캐릭터 에이전트 각각의 동작 계획 초기상태로 동작 실행이 시작되는 상태이고 최종 목표 동작 애니메이션을 만족하면 최종상태 P10, P11, P12에 종료한다. 예를들면 CA1(캐릭터 에이전트 1번)은 t1과 t7을 통해 CA3와 CA2에게 자신이 먼저 동작을 실행할 것을 요구한다. CA3와 CA2는 t2와 t11를 통해 CA1의 동작 실행 요구조건을 수용할 것인지를 판단하여 제어 메시지를 전달한다. 판단의 조건 파라미터는 각 캐릭터 에이전트의 동작 우선순위, 이동간 거리, 속도, 동작시간 등에 의해 영향을 받는다. 여기서 만일 CA3가 t2를 통해 reject 메시지를 전달하면 CA1은 동작 우선순위가 낮거나 다른 캐릭터 에이전트가 실행중이므로 대기 상태로 들어간다. 대기 상태란 동작 애니메이션이 정지됨을 의미하는 것이 아니라 동작 시간과 이동간 거리 상태에 따라 동작 프레임의 속도를 늦추거나 빠르게 조정하는 상태를 의미한다. 마찬가지로 CA2가 동일한 메시지를 전달하면 같은 절차를 거쳐 동작 애니메이션을 조정하게 된다.

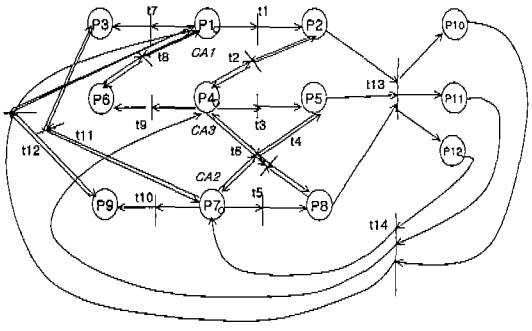

그림 6 3명의 인체모델 캐릭터에 대한 페트리넷 모델

이와같이 최종 상태에 도달하기 위해 다른 캐릭터 에이전트와 주기적으로 메시지를 전달하여 목표 요구 조건에 도달하면 CA1은 자신의 동작 애니메이션을 실행한다. 다른 캐릭터 에이전트도 동일한 방법으로 동작 애니메이션에 대한 해결 방안 조건을 탐색한다. 규칙 결정을 위한 상태정보(P)와 실행 조건 규칙(t)은 지식 베이스 형태로 저장된다. 그림 6의 페트리넷을 통해 대화 모델을 분석하기 위해 전체적인 문제를 단일화된 Petri- net의 형태로 분석하여 deadlock을 회피하고 Petri-net의 인접 행렬을 이용한 liveness와 safeness의 조건을 체크한다. 이것은 현재의 동작 애니메이션을 안정성 있고, 충돌회피를 보장하는 역할을 하게 된다. 에이전트들은 상호 관습에 따라 메시지를 교환 할 수 있고, 상태를 변경하고, 각 에이전트 동작 형태에 따라 동작 애니메이션을 수행한다.

5. 에이전트 시스템의 구현

본 시스템은 Visual C++ 6.0과 OpenGL을 사용하여 구현하였다.

그림 7은 그림 6의 페트리 넷 모델을 이용하여 3명의 캐릭터 에이전트가 1개의 정적 객체로 향하는 동작 애니메이션을 할 경우 3차원 공간상에서 이동하는 경로에 따라 각 캐릭터 에이전트가 서로 상호작용하여 나타나는 결과의 예를 보여주고 있다. 여기서는 1번 캐릭터가 먼저 도착하고 3번, 2번 캐릭터가 도착하는 순서로 디자인하였다. 구체적으로 설명하면 사용자는 인터페이스상에서 초기 입력 조건으로 각 캐릭터 에이전트의 동작 애니메이션 우선순위와 각 캐릭터의 목표 지점에 도달하기 위한 거리를 이동 경로를 나타내는 선 모양 형태로 마우스를 이용하여 시각적 스케칭 기법을 사용함으로써 캐릭터들의 시작과 종료지점을 나타내고 이동시 동작해야 할 동작 형태도 정의하게 된다. 즉, 도착하는 우선순위가 1번 캐릭터가 첫 번째, 3번 캐릭터가 두 번째, 2번 캐릭터가 세 번째로 도착을 하는 형태로 조건을 입력하고, 각 캐릭터의 동작 형태는 1번과 2번 캐릭터는 "걷다", 3번 캐릭터는 "달리다"라는 동작 형태로 동작 애니메이션을 실행하는 조건을 인터페이스상에서 초기 상태에 입력한다. 이러한 초기 입력 상태를 기초로 그림 4와 5의 places와 transitions를 이용하여 각 캐릭터 에이전트간에 서로 대화를 통해 메시지를 전달하여 동작 애니메이션을 조정하게 된다. 예를들면 그림 6에서 CA3(캐릭터 에이전트 3번)은 t3와 t9을 통해 CA2와 CA1에게 자신이 동작을 실행할 것을 요구한다. CA2와 CA1은 t4와 t8을 통해 CA3의 동작 실행 요구조건을 수용할 것인지를 판단하여 제어 메시지를 전달한다. 다른 캐릭터 에이전트로부터 accept의 메시지가 도착하면 자신의 동작 애니메이션 즉, "달리다"라는 동작 애니메이션을 실행하게 된다. 그러나, reject라는 메시지가 도착하면 다른 캐릭터 에이전트 중 하나가 우선 순위가

높기 때문에 대기 상태에 들어간다. 다른 2번 캐릭터, 3번 캐릭터 에이전트도 동일한 형태로 메시지를 전달하여 최종 결과에 도달할 때까지 반복한다. 만일, 3차원 공간상에서 디자인한 이동 거리가 시간적인 요소로 판단할 때 1번 캐릭터가 30초, 3번 캐릭터가 20초, 2번 캐릭터가 10초 정도 안에 정적 객체에 도달한다고 가정할 경우 실제 우선 순위 조건은 1번, 3번, 2번 캐릭터이나 이동거리와 시간적인 요소에 의해 2번, 3번, 1번 캐릭터와 같이 사용자의 캐릭터 동작 우선 순위에 맞지 않는 동작 애니메이션이 실행될 수 있다. 그러므로, 에이전트 기술을 바탕으로 위의 그림 4와 5의 places와 transitions을 이용하여 사용자가 입력한 초기 조건을 만족하는 결과에 도달할 때까지 캐릭터 에이전트간 대화 메시지를 전달하여 우선순위와 같은 조건 등을 고려한 캐릭터간 동작 애니메이션을 지연시키거나 프레임간 속도를 조절하여 최종적으로 사용자가 원하는 형태인 1번, 3번, 2번 캐릭터가 도착하는 순서로 동작 애니메이션을 에이전트 프레임워크에 의해 제어하게 된다.

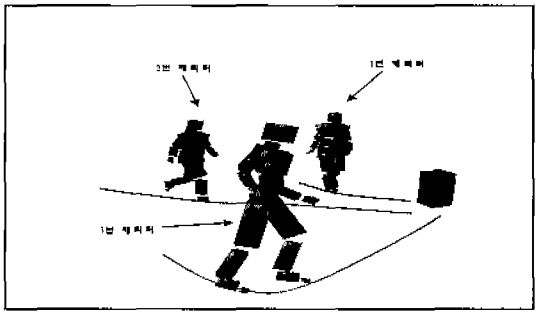

그림 7 3명의 캐릭터와 1개의 정적 오브젝트에 대한 동작 제어

따라서, 시간, 공간, 캐릭터 에이전트 동작 우선 순위 정보를 기반으로 이러한 애니메이션 과정이 대화 규칙을 통해 구성되며 페트리넷 모델을 기반으로 다른 캐릭터 에이전트간의 충돌이나 간섭없이 동작 애니메이션을 연출할 수 있다. 메시지에는 각 캐릭터의 동작의 특성, 프레임, 거리등의 정보가 포함되어 전달됨으로써 각 캐릭터 에이전트의 판단 요소로 사용되게 된다.

에이전트 기술을 적용한 캐릭터 동작 애니메이션 시스템과 기존의 동작 제어 즉, 에이전트 기술을 적용하지 않은 기법들을 캐릭터와의 상호작용성과 동작 제어 디자인 관점에서 여러 가지 항목들을 고려하여 비교 분석하면 표 1과 같다.

6. 결 론

본 논문은 3차원 공간상에서 여러 캐릭터간의 동작 애니메이션 제어를 에이전트 시스템을 이용하여 동작 제어가 가능함을 나타내는 디자인 원리와 방법들에 대해서 제안하였다. 이러한 기법을 이용하여 동적으로 동작이 변경되는 애니메이션 형태에 대한 각각의 캐릭터가 가지는 에이전트 사이의 의사교환과 조정, 협동 지식의 획득과 수정, 모델링을 하기 위한 유틸리티와 에이전트 서비스를 제공하게 된다.

제안한 에이전트 시스템의 특징은

(1) 일관성있고 동기적으로 사용자의 애니메이션 저작 의도를 반영하는 효과를 가지게 된다.

(2) 확장성을 지원, 에이전트들의 등록정보를 일괄 관리함으로써 동작 애니메이션에 따른 동작제어를 이미 등록된 다른 에이전트와 협동작업을 통해서 스케줄 가능하다.

(3) 전체 애니메이션 동작을 각 에이전트의 부분적인

표 1 동작 제어 기법들의 비교

기법 / 항목	에이전트 기술 적용하지 않음		에이전트 기술 적용			
	키프레임, 운동학, 역운동학	3D 스튜디오, 마야	에이전트객체 (스마트)	규칙 기반 시스템	IMPROV 시스템	에이전트시스템 (MOCAF)
캐릭터간의 상호작용	–	–	가능	–	부분적지원	가능
세부적 동작 묘사	지원	지원	–	지원	–	–
동작 확장성 및 재사용성	–	–	–	–	–	가능
에이전트 구현 가능 여부	–	–	가능	가능	가능	가능
쓰레드 구조	–	–	–	–	–	가능
사용자의 의사 전달	정밀한 제어요구	정밀한 제어 요구	키워드정의 요구	가능	가능	가능

동작 애니메이션 형태로 나누어 에이전트간의 동시동작 애니메이션, 우선 순위에 따른 단일 동작 애니메이션을 가능하게 한다.

이러한 에이전트 프레임워크는 캐릭터의 에이전트를 정의하고, 동작 계획, 동작을 안내하기 위한 규칙, 에이전트간의 통신을 가능하게 함으로써 효율적인 동작 표현을 위한 애니메이션 어플리케이션에 적용되어 개발 사용될 수 있다. 그러나, 캐릭터 에이전트간의 동작 제어를 위한 규칙을 생성하기 위해 동작 상태와 변이과정이 테이블 형태로 지정되어 디자인되어 있어야 한다. 따라서 향후 사용자가 동작 시나리오에 따른 규칙을 생성하기 위한 페트리 넷 모델을 디자인하기 위한 도구의 개발이 필요하다.

참 고 문 헌

[1] Hyacinth S. Nwana., "Software Agents : An overview," The Knowledge Engineering Review, Vol. 11, No. 3, pp.1-46, Sept 1996.

[2] Hyacinth S. Nwana and Divine T. Ndumu, "A Perspective on Software Agents Research," The Knowledge Engineering Review. Vol. 14, No. 2, pp.1-18, 1999.

[3] N. I. Badler, K. H. Manoochehri and G. Walters, "Articulated Figure Positioning by Multiple Constraints," IEEE CG & A, Vol. 7, No. 6, pp.28-38, June 1987

[4] C. Rose, B. Guenter, B. Bodeheimer and M. F. Cohen, "Efficient Generation of Motion Transitions using SpaceTime Constraints," Computer Graphics (Proc. of SIGGRAPH' 96), pp.147-153, August 1996.

[5] Muller, J. P., "A Concceptual Model for Agent Interaction," Proceedings of the 2nd International Working Conference on Cooperative Knowledge Based Systems (CKBS-94), Deen, S.M.(ed.), Keele University:Dake Centre, pp.213-234, 1994.

[6] K. Perlin and A. Goldberg, "Improve: A System for Scripting Interactive Actors in Virtual World," Computer Graphics (Proc. of SIGGRAPH' 96), pp.205-216, August 1996.

[7] M. Kallman and D. Thalmann, "A Behavioral Interface to Simulate Agent-Object Interactions in Real Time," Proceedings of Computer Animation, pp.138-146, 1999.

[8] H. Noser and D. Thalmann, "A Rule-Based Interactive Behavioral Animation System for Humanoids," Transactions on Visualization and Computer Graphics, Vol. 5, No. 4, pp.281-307, 1999.

[9] Wagner da Silva, F., Garcia, L.M., Farias, R.C., Oliveira. A.A.F., "A Control Theory Approach for Real-time Animation of Artificial Agents," Proceedings XIII Brazilian Symposium on Computer Graphics and Image Processing, pp.211-218, 2000

[10] Chaib-draa, B., Moulin, B., Mandiau, R. and Millot, P., "Chapter 1- Trends in Distributed Artificial Intelligence," Foundations of Distributed Artificial Intelligence, G.M.P.O'Hare and N.R. Jenning s(eds.), pp.3-55, John Wiley & Sons Inc., 1996.

[11] Genesereth, M.R. and Ketchpel, S. P., "Software Agents," Communications of the ACM37(7), pp.48-53, 1994.

[12] Smith, I. A. and Cohen, P. R, "Towards a Semantics for a Speech Act Based Agent Communication Language," SRI technical note, 1995.

김 기 현

1997년 대구대학교 전자계산학과 졸업(학사). 1999년 경북대학교 컴퓨터과학과 졸업(석사). 2001년 경북대학교 컴퓨터과학과 박사 수료. 관심분야는 캐릭터 애니메이션, 에이전트 시스템, 인간과 컴퓨터 상호작용, 멀티미디어 시스템 등

김 상 욱

1979년 경북대학교에서 컴퓨터공학으로 학사 학위를 취득. 1981년 서울대학교에서 컴퓨터과학으로 석사 학위를 취득. 1989년 서울대학교에서 컴퓨터과학으로 박사 학위를 취득. 1988년 ~ 현재 경북대학교에서 컴퓨터과학과 교수로 재직중. 관심분야는 컴퓨터 언어, 객체중심 컴퓨팅, 시각언어, 멀티미디어와 지식처리임