논문 99-24-5B-10 한국통신학회논문지 '99-5 Vol.24 No.5B

UML을 이용한 자율축구로봇의 동작특성과 스케줄링 가능성 분석

정회원 이 재 익*, 강 순 주*, 서 대 화*

UML based modeling for the behavior and schedulablity analysis of autonomous soccer robots

Jae-Ick Lee*, Soon-Ju Kang*, Dae-Wha Seo* ***Regular Members***

요 약

실시간 내장형 시스템(real-time embedded system)의 반응 동작(reactive behavior)을 정확하게 분석하기 위해서 상태차트(state charts)에 바탕을 둔 객체 지향 모델링 방법론들이 많이 사용되고 있다. 하지만 이들 방법론들은 경성 실시간 시스템(hard real-time system)이라면 반드시 고려해야할 스케줄링 가능성분석(schedulability analysis)에 필요한 시간에 대한 정보와 시간 제약(timing constraints)을 정확히 나타내지 못하는 문제점이 있다. 본 논문에서는 최근 OMG(Object Management Group)에서 객체지향개발의 기준으로 채택된 UML(Unified Modeling Language) ver1.1을 사용하여 실시간 내장형 시스템의 사례인 자율 축구로봇의 반응 동작과 시스템의 시제 동작(temporal behavior)을 모델링하여, 모델링된 시스템에서의 스케줄링 가능성 분석 방안을 제안한다.

ABSTRACT

As real-time embedded systems become more complex, the reactive behavior of the real-time systems is hard to implement using ad-hoc techniques. State machine based approaches are naturally suited for modeling reactive behavior of real-time systems, thus several object-oriented development methodologies are attractive as a potential development methodology for real-time systems. However these methodologies do not provide any mechanisms to specify and enforce timing constraints that are required for schedulability analysis of hard real-time systems. In this paper, we show UML, adapted as standard object-oriented development by OMG, based modeling for autonomous soccer robot as a case study of a hard real-time system, and show how the proposed real time scheduling theory may be applied to the UML based model.

Ⅰ. 서 론

실시간 시스템 개발에 있어 모델링 방법론은 구현에 들어가기 전에 시스템에 대한 정확한 분석을 바탕으로 시스템의 적정성 여부와 효과적인 설계를 유도해 낼 수 있으므로 매우 중요한 의미를 가지고 있다. 이 중 한정된 자원을 가지고 주어진 시간 내에 적절한 응답을 해야하는 실시간 내장형 시스템 모델링은 시스템의 복잡한 반응 동작(reactive behavior)에 대한 엄밀한 분석이 요구되기 때문에, 비정형적인 방법으로는 설계의 적정성 여부를 정밀하게 판단하지 못한다. 복잡한 실시간 내장형 시스템의 반응동작의 정확한 모델링을 위해서는 상태머신(state-machine)에 바탕을 둔 방법이 적합하고, 상태머신의 발전된 형태인 상태차트(statecharts)[7]에 바탕을 둔 UML[1], ROOM[11], OMT[9], OCT-OPUS[2], OOSE[8]등이 실시간 내장형 시스템 개발에 많이 사용되어 지고 있다. 이들 객체 지향 개발방법론들은 분석에서 설계, 구현으로의 과정이 연속적으로 이어

* 경북대학교 전자공학과(Jilee@rtlab.kyungbook.ac.kr)
논문번호 : 99054-0218, 접수일자: 1999년 2월 18일

진다. 따라서 기존의 실시간 시스템의 주된 개발방법이었던 구조적(structured)기법중심의 개발 방법보다 소프트웨어의 개발기간을 단축할 수 있고, 재사용성, 소프트웨어 유지보수등에서 많은 장점을 가지고 있다[6]. 하지만 이런 장점이 있는 객체 지향 개발방법도 경성 실시간 시스템이라면 반드시 요구되어지는 스케줄링 가능성 분석에 필요한 시간제약(timing constraints)과 시제행동(temporal behavior)에 대한 명확한 표현에 대한 지원이 이루어져야만 한다. 시간정보 표현에 대한 측면에서 ROOM이나 OMT방법은 시간제약(timing constraints)에 대해 명확한 표현을 제공하지 못하는 문제점을 가지고 있다.

본 논문에서는 실시간 시스템 모델링에 있어 UML의 이용과 동시에, M.Sakena[10]가 제안한 디자인 가이드라인을 바탕으로 UML의 확장 메커니즘인 정형(stereo type)과 시퀀스다이어그램을 사용하여, UML기반 모델에 스케줄링 가능성분석을 제안한다.

이에 대한 사례로 로봇 축구를 위한 자율주행로봇(autonomous soccer robots)의 동작원리를 UML을 사용하여 모델링하고, 모델링된 시스템에 실시간 시스템 스케줄링 알고리즘을 적용하여 스케줄링 가능성 분석을 시도한다. 그리고 객체기반 내장형 시스템 시뮬레이션 툴을 사용하여 모델을 검증해보았다.

본 논문은 서론에 이어 2장에서는 관련 연구분야에 대해 기술하고, 3장에서는 UML기반 실시간 시스템 모델링에서의 스케줄링 가능성 분석과정에 대하여 알아본다. 4장에서는 실례로 UML을 이용하여 실시간 내장형 시스템인 자율 축구로봇을 모델링한다. 5장에서 모델링된 자율 축구로봇 모델에 스케줄링 가능성 분석과 모델링된 시스템의 시뮬레이션을 통하여 모델을 검증한 후 마지막으로 6장에서 결론을 맺고, 향후 연구방향을 제시한다.

Ⅱ. 관련연구동향

경성 실시간 시스템 개발에 있어 시간에 대한 제약조건의 처리는 가장 중요한 부분이다. 따라서 경성 실시간 시스템 설계에서는 설계와 구현단계 이전에 시스템의 각 태스크들이 마감시한(deadline)을 만족하는가를 분석하는 스케줄링 가능성 분석이 반드시 이루어져야 한다. 분석이 이루어지기 위해서는 모델링 단계에서 시간에 대한 제약(timing constraints)과 시스템의 시간에 대한 행동에 대한 표기 등이 필요하게 된다. 실시간 객체지향 개발방법으로 많이 사용되고 있는 ROOM이나 OMT등은 시간에 따른 시스템의 행동을 묘사하기 위해 메시지 시퀀스 차트를 사용하고 있다[12]. 하지만 메시지 시퀀스 차트 그 자체만으로는 시간제약에 대한 표기법과 시스템의 시제행동에 대한 명확한 표현이 힘들다. 이에 대해 시스템의 모델링단계에서 시간에 대한 표기법을 가진 변형된 메시지 시퀀스 차트에 대한 연구가 이루어지고 있다.

M.sakena등은 실시간 내장형 시스템의 CASE 툴 제작에 대한 가이드 라인을 제시하는 논문에서 ROOM으로 모델링된 시스템의 스케줄링 가능성 분석을 위해 시간제약을 나타내는 트랜잭션기반의 메시지 시퀀스 차트형태로 나타내었다[10]. 그리고 적용사례로 자동 주행장치를 모델링하고, 이에 대한 스케줄링 가능성분석을 하였다. 또한 slomka등은 MSC와 SDL(Specification and Description Language)기반의 전화통신 시스템 모델에 스케줄링 알고리즘 적용을 위하여 시간제약에 대한 정보를 가진 변형된 MSC를 제안하여 성능 측정에 대한 연구를 하였다[13].

Ⅲ. UML모델에 기반을 둔 스케줄링 가능성 분석 절차

UML기반의 모델은 모델의 각 특성을 명확히 나타낼 수 있는 다수의 다이어그램으로 구성되어진다. 모델링시 우선 시스템의 요구사항으로부터 usecase 다이어그램을 구성하고, usecase를 바탕으로 한 시나리오를 시간적인 흐름에 따른 시퀀스 다이어그램으로 나타낸다. 시퀀스 다이어그램에 나타난 객체들은 시스템의 정적인 구조를 나타내는 클래스 다이어그램의 클래스들을 정의하는데 사용되어지고, 복잡한 동작특성을 가진 객체들은 상태다이어그램으로 객체내부의 상태를 명확한 형태로 나타낸다. 모델링된 시스템의 스케줄링 가능성분석은 시퀀스 다이어그램에서의 시간에 대한 정보와 상태다이어그램, 클래스 다이어그램에 나타난 객체에 대한 정보를 바탕으로 이루어진다.

1. 트랜잭션(transaction)의 정의

시퀀스 다이어그램에서는 메시지 패턴과 이벤트 ID를 이용하여 다양한 시제 표현이 가능하다. 시퀀스 다이어그램에서 이벤트ID로 구분되어 시간적인 특성을 나타내는 부분을 트랜잭션으로 명한다. 트랜

잭션은 외부에서 발생한 이벤트에 의해 발생되어 이에 대한 적절한 반응을 함으로써 마치게 된다. 트랜잭션은 연속된 이벤트로 구성되고, 주기 또는 비주기적으로 도달하는 트랜잭션의 최소 도달시간(minimum int-erarrival time)과 마감시한(end-to-end deadline)을 가진다.

2.스케줄링 가능성분석의 과정

모델링된 시스템이 올려질 운용체제는 선점 우선순위(preemptive priority) 스케줄링을 하고, 단일 프로세서이며, 이벤트 우선 순위 스케줄링을 한다고 가정한다.

1)시퀀스 다이어그램에서 시간제약이 있는 부분을 트랜잭션으로 정의하고 트랜잭션의 활동주기와 마감시한을 구한다.

2)각 트랜잭션의 실행시간을 계산한다. 트랜잭 션은 연속된 이벤트로 구성된다. 이벤트를 수신 한 객체는 수신된 이벤트에 대한 프로세싱을 수 행하게 된다. 수신된 이벤트에 대한 객체내의 프로세싱을 가상 태스크(virtual task)라 한다 [10]. 가상 태스크는 이벤트를 수신할 때 수 행되어, 이벤트에 관련된 프로세싱을 수행한 후 이에 대한 응답이나 이벤트를 발생시키면 끝나 게 된다. 트랜잭션은 연속된 이벤트로 구성된다. 따라서 트랜잭션을 구성하는 이벤트에 의한 가 상 태스크들의 실행시간을 합하여 트랜잭션의 실행시간을 구한다.

3)메시지에 우선 순위를 부여한다.

시스템의 행동과 시간제약은 트랜잭션의 형태로 나타나기 때문에 각 트랜잭션에다가 우선 순위를 부여한다. 그 후 메시지에게 그 메시지가 속한 제일 높은 우선 순위의 트랜잭션과 같은 우선 순위를 부여한다. 스케줄링이 메시지의 우선 순 위에 의해 결정됨으로, 메시지의 우선 순위는 응 답시간에 직접적으로 영향을 미친다. 따라서 트 랜잭션의 마감시한을 고려한 우선 순위 부여가 필요하게 된다. 일반적으로 DM(deadline monotonic)또는 RM(rate monotonic)스케 줄링 알고리즘[5]이 사용된다. DM은 마감시한 이 짧은 태스크에게 높은 우선 순위를 부여하는 방법이고, RM은 발생빈도 수가 높은 태스크에 게 높은 우선 순위를 부여하는 방법이다.

4)객체를 쓰레드에 지정

모델에 스케줄링 알고리즘을 적용시키는 데 있어 제일 큰 문제점은 상태 머신(state machine) 의 RTC(run-to-completion)프로세싱에 의한 블록킹이다 [11]. 이는 같은 쓰레드내의 높은 우선순위의 가상 태스크가 낮은 우선 순위의 가상 태스크의 수행으로 블록킹되어, 우선 순위가 높 은 트랜잭션이 마감시한을 놓치게 되는 경우이 다. RTC프로세싱은 쓰레드내의 선점(pree- mption)으로 인한 내부적인 동시성문제의 해결 방식인 상호배제 메커니즘의 사용으로 인한 신뢰 성문제나 복잡성 문제 때문에 사용하는 방식이 다. RTC프로세싱의 장점은 단순함에 있고, 단점 은 긴 블록킹 시간으로 인해 마감시한을 넘기는 것이다. 상태 머신의 실행은 RTC프로세싱으로 이루어지기 때문에 RTC블록킹은 피할 수 없다. 하지만 M.Saksena[10]가 제시한 디자인 가 이드라인에 따라 객체 내에서 마감시한이 짧은 이벤트의 상한 블록킹 시간을 명시한 후, 객체내 의 태스크의 실행시간이 그 태스크보다 높은 우 선 순위를 가진 태스크의 상한 블록킹 시간을 넘 지 않도록 함으로써 최소화시킬 수 있다. 이 디 자인 가이드 라인은 객체들을 그룹화 하는데 있 어 일종의 제약을 제공해주는데, 각각의 객체가 서로의 상한 블록킹 시간을 만족시킬 수 있는 것 들끼리 하나의 쓰레드로 모을 수 있게된다.

5)각 트랜잭션의 블록킹 타임을 계산한다.

RTC 블록킹에 의한 블록킹 시간은 다음의 식으로 계산 가능하다.

$B_i = \max_j C_j$ (B_i =가상태스크 VT_i의 RTC 블록킹 시간, C_j=가상 태스크 VT_j의 실행시간)

6)각 트랜잭션의 응답시간을 분석한다.

각 트랜잭션의 블록킹 시간이 계산되면 응답시간

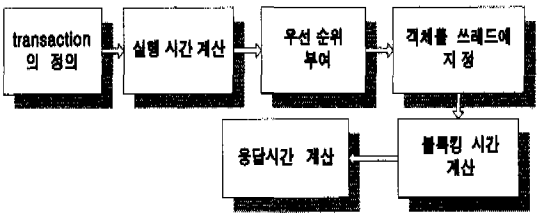


그림 1. 트랜잭션 단위의 스케줄링 가능성 분석 절차

분석은 실시간 스케줄링 이론에 의해 계산이 가능하다. 응답시간은 응답시간R을 구하는 식

$R = \sum_{VTj \in hp(i)} \left(\lceil \frac{Ri}{Tj} \rceil * Cj \right) + Ci + Bi$ (Ri, Ci, Bi는 VT_i의 응답시간, 실행시간, 블록킹 시 간, hp(i)는 VT_i 보다 높은 우선 순위를 가진 VT들의 집합)을 이용한다. 첫 번째 항

$\sum_{VTj \in hp(i)} \left(\lceil \frac{Ri}{Tj} \rceil * Cj \right)$는 높은 우선 순위에 의

한 방해이고, 그 다음은 태스크의 실 행시간 그리고 마지막은 낮은 우선 순위의 태스크에 의한 블록킹 시간이다. 그림1은 트랜잭 션 단위의 스케줄링 가능성분석의 절차를 보여준 다.

Ⅳ. 자율 축구 로봇의 UML기반 모델링

앞에서 제안한 UML모델기반의 경성 실시간 시스템 스케줄 가능성 분석절차의 적용사례로, 로봇 축구대회에 사용되는 자율 축구로봇을 모델링 하였다. 본 논문에서는 빠르게 움직이면서 서로 협력하여 변화하는 환경에서 축구경기를 진행하게 되는 로봇 축구경기에 사용될 이동로봇의 동작원리에 대한 객체지향 모델링을 목표로 한다. 로봇 축구대회는 인공지능영역과 로봇 제어등 많은 영역의 기술을 테스트할 수 있고, 여기에서의 기술은 군사분야, 우주탐사, 원자력 발전소 제어등 중요한 분야에 응용될 수 있다. 변화하는 환경에서, 로봇에 내장된 한정된 자원을 가지고 주어진 시간 안에 상황에 적합한 목표를 수행해야 로봇은 실시간 내장형 시스템의 특징을 잘 나타내고 있어 본 논문에서 실시간 시스템의 모델링 사례로 들었다. 본 논문에서는 모델링된 시스템의 소프트웨어 아키텍처는 시스템의 동작원리를 명확하게 나타내고 복잡함을 피하기 위해 추상화되었다. 그리고 시스템을 효과적으로 동작하게 하는 하드웨어나 소프트웨어설계보다는 UML 객체지향 개발방법으로 실시간 시스템을 모델링하고, 모델링된 시스템에 스케줄링가능성분석을 시도해 보는 것에 주안을 두고 있다.

1. 자율 축구 로봇

로봇이 위치한 환경은 좁은 구역 안에서 공과 다른 로봇 그리고 로봇 자신이 모두 빠르게 움직이고 있는 동적인 환경이다. 따라서 로봇은 동적인 외부환경의 변화에 대응하려면 한정된 시간 내에 외부환경에 대한 입력을 바탕으로 다음 행동을 결정하여 외부로 반응을 해야한다. 이런 환경은 로봇의 행동 결정이나 각 디바이스에 대하여 실시간적인 제어가 필요하게 된다. 따라서 로봇이 주어진 짧은 시간 안에 적절한 반응을 하지 못하여 장애물에 충돌하거나 골을 허용하게 되면 목표수행이 실패하게됨으로 축구로봇에 대한 제어 시스템 설계에서는 경성 실시간 시스템 설계에 바탕을 두어야 한다. 로봇은 부착된 카메라로부터 주변 상황에 대한 정보를 얻는다. 입수한 정보를 바탕으로 로봇 스스로가 다음 행동을 결정한 후, 두 개의 바퀴를 구동하여 목표를 수행한다. 목표수행과정에서 카메라로부터 미처 인지하지 못한 장애물을 소나 센서가 감지하면 그에 따른 회피동작을 한 후 목표를 계속 수행한다. 로봇 축구대회에서 로봇은 공격수, 수비수, 골키퍼로 각 역할 분담이 되고, 각 역할에 따라 목표가 다르게 설정된다. 본 논문에서는 공격수를 모델링하였다. 공격수는 기본적으로 로봇이 공을 찾으면 공을 향해가서 상대편의 골을 앞에 위치시킨 다음 상대편의 골로 다가가서 골과의 거리가 가까울 때 공을 찬다. [3,4]

2. UML기반의 축구로봇 모델링

1) 축구 로봇의 usecase 다이어그램

operator는 로봇을 직접 초기화시키거나, 부착된 센서를 조정하는 액터이다. 그리고 operator는 일반적으로 로봇을 초기화시키는 명령을 보낸다. 로봇은 기본적으로 초기화된 후 자율적으로 축구게임을 진행하게된다. 그림 2는 로봇축구모델의 usecase 다이어그램이다.

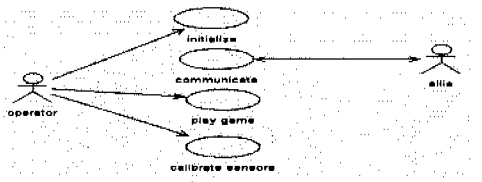


그림 2. 로봇축구모델의 usecase 다이어그램

2) 축구로봇의 클래스 모델과 상태 다이어그램

축구로봇은 Vision Module, Model Manager, Strategy Planner, Communicator, Motor Controller, Motor, Sensor의 클래스로 구성된다. 전체적인 구성에 대해 설명하면 Vision Module은 카메라를 통해 얻은 시각정보를 이미지 프로세싱하여 로봇 주위에 대한 정보를 Model Manager에게 전달하고, Model Manager는 얻은 정보를 바탕으로 로봇이 가지고 있는 맵을 재구성하고, 공과 상대편 골에 대한 변화된 정보를 Strategy Planner에게 전달한다. Strategy Planner는 Model Manager로부터의 정보를 가지고 다음 행동을 결정하여 Motor Controller에게 명령을 전달하게 되고 Motor Controller는 명령을 컨트롤 시그널로 변화하여 Motor에게 전달하여 모터를 구동시킨다. 그림3는 로봇 축구의 클래스 다이어그램이다.

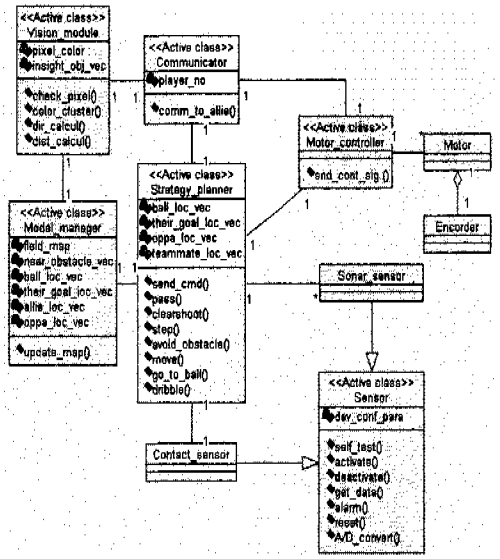


그림 3. 로봇의 클래스 다이어그램

Vision Module은 100ms주기로 카메라로부터 이미지의 RGB 컬러 픽셀을 이미지 프로세싱하여 필드 내에 있는 중요한 물체의 위치에 대한 정보를 얻는다. 중요한 물체는 로봇에게 정해진 임무에 따라 달라지게 되는데 공격수인 경우 상대방의 골, 공이 중요한 개체가 되고, 골키퍼 같은 경우에는 우리편 골대, 상대편 로봇, 공이 중요한 물체가 된다. Vision Module은 필드내의 중요한 물체의 한 부분인가를 알기 위해 각각의 픽셀 컬러를 체크한다. 그 후 color clustering과정을 거쳐 인식 가능한 물체로 매핑 한다. 각각의 color cluster의 크기와 위치는 실제 필드내 물체의 방향과 거리계산에 사용된다. 이미지 프로세싱을 마치게 되면 Vision Module은 카메라에 잡힌 각각의 물체에 대해 벡터를 구성할 수 있다. 벡터는 로봇 현재 위치로부터의 물체에 대한 거리와 방향으로 구성된다. 그림4는 Vision Module의 상태 다이어그램이다.

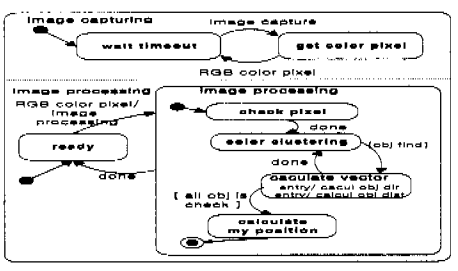


그림 4. Vision Module의 상태 다이어그램

Image capturing 상태와 image processing 상태는 동시에 수행되는 상태이다. Model Man- ager는 100ms를 주기로 Vision Module의 출력인 필드내의 물체의 위치정보를 현재 환경에 대한 맵으로 전환시킨다. Model Manager는 필드의 맵과 근처에 있는 물체의 위치벡터를 포함하고 있다. 위치 벡터는 4개의 요소, 물체에 대한 거리, 방향, 거리의 변화, 물체가 향하는 방향으로 구성된다. 거리의 변화와 물체가 향하는 방향은 움직이는 물체의 위치를 예측할 때 사용되고, 이전에 인식된 물체의 위치에 대한 정보와 현재의 물체의 위치정보를 이용하여 계산된다. 그림5는 Model Manager의 상태 다이어그램이다.

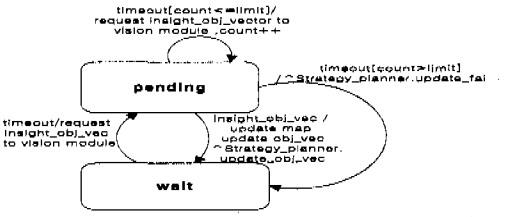


그림 5. Model Manager의 상태 다이어그램

그림6에서 Model Manager가 필드에 대한 정보를 갱신하면 Strategy Planner에게 정보가 갱신되었다는 것을 알려주고, Strategy Planner는 Model Manager로부터 중요한 물체에 대한 갱신된 정보를 얻는다. Strategy Planner는 갱신된

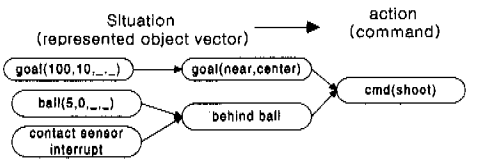


그림 6. Strategy Planner의 다음 행동 결정 과정

정보를 가지고 로봇의 역할에 따른 전략을 이용하여 다음 행동을 결정하게 되는 역할을 한다. 그리고 다음 행동이 결정되면 Strategy Planner는 motor controller에게 다음 행동에 해당하는 명령을 내린다.

그림7은 Strategy Planner의 상태 다이어그램이다. 로봇이 공에게 다가갔다면 공을 몰고 상대편의 골로 향하는 명령을 내리게 된다. 그리고 공이 여전히 로봇의 앞에 위치하고 있고 골과의 거리가 가까우면 골을 향해 슛을 한다. 슛한 후 골이 실패하면 다

시 공을 향해 다가가게 되고 성공하면 원래의 위치로 돌아오게 된다. object_vector에 공의 거리의 변화와 공이 움직이는 방향에 대한 정보가 있으면 이 정보를 이용하여 공의 이동코스를 계산하여 그 지점으로 이동하여 공을 가로챈다. 만일 이동 중에 로봇의 진행방향에 장애물이 나타나면 회피 알고리즘을 적용하여 장애물을 회피하게 된다. 장애물을 회피하게 되면 갱신된 object vector를 참고하여 다시 행동을 결정하게된다.

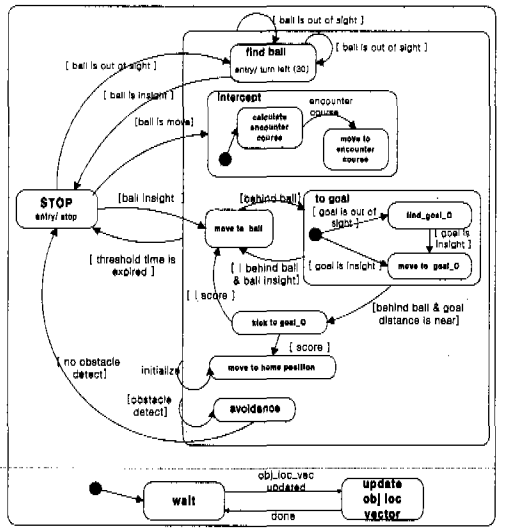


그림 7. Strategy Planner의 상태 다이어그램

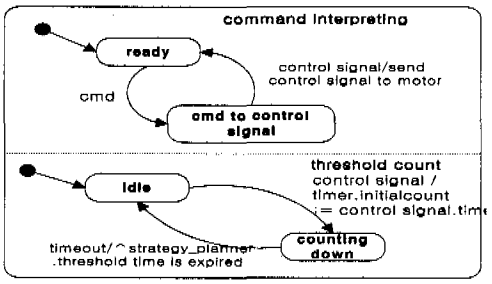


그림 8. Motor Controller의 상태 다이어그램

그리고 로봇이 카메라로부터의 입력에서 공을 찾지 못할 경우 로봇은 공을 찾기 위해 30° 씩 회전하게 된다. 이 행동은 공을 찾을 때까지 계속된다. 그리고 각 행동에 대한 명령은 그 행동에 대한 특정 조건이 만족되면 끝나거나, 행동에 대한 threshold time이 경과하게 되면 그 명령은 끝나게 된다. Motor Controller는 Strategy Planner로부터 받은 명령을 컨트롤 신호로 변환하여 왼쪽과 오른쪽의 Motor에 전달하게 된다. Motor Controller는 정확한 움직임을 위하여 명령을 모터의 구동시간으로 변환시킨다. 그림8은 Motor Controller의 상태다이어그램이다.

Contact Sensor와 Sonar Sensor는 디지털 센서로써 Sensor라는 부모 클래스로부터 기능을 상속받는다. Contact Sensor는 로봇의 그립에 위치하여 로봇이 공의 뒤에 위치하였는가를 확인해주고, Sonar Sensor는 장애물을 감지한다.

3) 시나리오의 시퀀스 다이어그램

시간에 따른 객체간의 메시지 전달을 나타내는 시퀀스 다이어그램은 use case에 따른 각각의 시나리오에 따라 작성되었다. 로봇의 use case에 대한 각각의 시나리오를 나타내면 그림9와 같다. initialize use case에서는 operator가 host computer를 통해 로봇의 초기화를 명령한다는 시나리오가 존재한다. 그림10은 initialize 시나리오의 시퀀스 다이어그램이다.

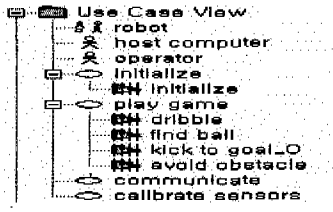


그림 9. use cases와 시나리오

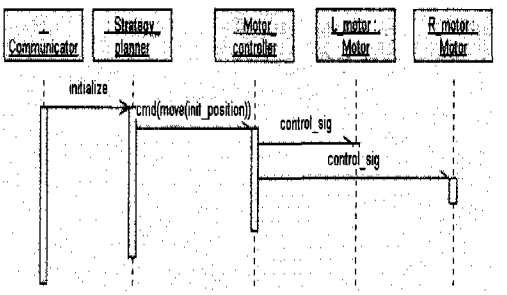


그림 10. Initialize 시나리오의 시퀀스 다이어그램

play game use case의 시나리오는 4가지가 존재한다. 공을 드리블하여 상대방의 골로 향하는 경우, 공을 시야에서 놓친 경우, 상대방의 골에 충분한 거리로 다가갔을 경우 상대방의 골로 슛을 하는 경우, 주행중 장애물을 회피하는 경우이다. 먼저 로봇이

취하는 가장 기본적인 행동으로 공에게 가서 공을 드리블하여 상대편의 골로 향하는 것이다. 이 시나리오에 해당하는 시퀀스 다이어그램은 그림11과 같다. 그림11의 시퀀스 다이어그램에는 3개의 트랜잭션이 나타난다. A ~ B 구간은 update model 트랜잭션으로 Model Manager가 주기적으로 Vision Module에게 물체에 대한 새로운 벡터를 요청하여 object vector와 맵을 갱신한 후 Strategy Planner에게 갱신 사실을 알려주는 것이다. 따라서 update model 트랜잭션은 timeout 이벤트, request insight_object_vector 이벤트, insight_object 이벤트로 구성된다. 주기는 100ms이고 마감시한 역시 100ms이다. B ~ C 구간은 go to ball 트랜잭션으로 Strategy Planner가 Model Manager로부터 object vector가 갱신되었다는 신호를 수신하면 갱신된 object_vector를 Model Manager로부터 수신한다. 로봇이 공을 가지고 있지 않다는 사실을 알고 공을 향해 움직이라는 다음 행동을 결정하여 Motor Controller에게 공의 위치로 움직이라는 명령을 보내게 된다. 이 구간의 마감시한은 100ms이다. D ~ E구간은 dribble 트랜잭션구간으로 Strategy Planner가 로봇이 공의 뒤에 있다는 정보를 가지면 공을 몰고 상대편 골로 향해 움직이라는 명령을 내린다. 이 구간의 마감시한은 100ms이다. 로봇이 이동 중에 로봇의 주행경로에 존재하는 장애물을 감지하면 장애물을 회피하게된다.

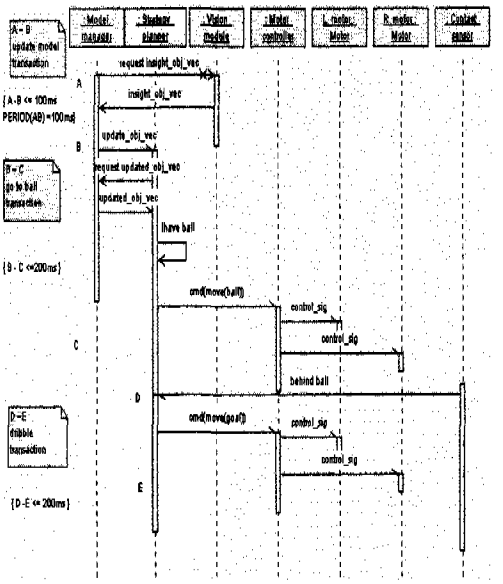


그림 11. 드리블 시나리오의 시퀀스 다이어그램

V. 스케줄링 가능성 분석 및 검증

이 장에서는 2장에서 소개하였던 스케줄링 가능성 분석과정을 이용하여 UML로 모델링된 축구로봇 모델에 실시간 스케줄링 알고리즘을 적용시켜, 스케줄링 가능성분석을 시도한다. 그리고 UML로 모델링된 축구로봇 모델의 시뮬레이션을 통해서 실제 각 트랜잭션이 마감시한을 만족하는지 확인해 본다.

1. 트랜잭션의 정의

시퀀스 다이어그램에서 시간에 대한 제약이 있는 부분을 트랜잭션으로 나타내고, 이를 표로 나타내면 표1과 같다.

표 1. 트랜잭션과 시간제약

트랜잭션	stimulus	response	주기	마감시한
image capture	image capture	RGB color pixel	100ms	70ms
image processing	RGB color pixel	insight obj vector	100ms	80ms
update model	time out	update map	100ms	100ms
go to ball	obj_vec update	control signal		200ms
dribble	have ball	control signal		200ms
avoidance	obs detected	control signal		150ms

표 2. 태스크의 실행시간

vision module	model manager	strategy planner	motor controller	other
$C_{RGBcolorpixel}$ =20ms	$C_{timeout}$ =2ms	$C_{updated\_obj\_vec}$=5ms	C_{cmd}=8ms	C*= 5ms
$C_{request\_insight\_obj\_vec}$=3ms	$C_{insight\_obj\_vec}$=3ms	C*=3ms	C*=3ms	
$Ci_{mage\_capture}$= 5ms	C*=2ms			
C*=3ms				

2. 각 트랜잭션의 실행시간계산

트랜잭션의 실행시간을 계산하기 위해서는 객체내의 이벤트에 의한 가상 태스크의 실행시간에 대한 정보가 필요하게 된다. 표2는 각 태스크에 대한 실행시간은 상태다이어그램을 참고하여 임의로 가정하여 나타낸 것이다. image capture 트랜잭션의 실행시간은 image capture 이벤트에 대한 태스크의 수행시간인 5ms이다. image processing 트랜잭션의 실행시간은 RGB color pixel 이벤트에 대한 태스크

수행시간인 20ms이다. 시퀀스 다이어그램을 참조하면 update model 트랜잭션은 총 4개의 이벤트로 구성되어 있다. 따라서 주기적인 timeout 이벤트에 의한 태스크, request insight_object_vector 이벤트에 의한 태스크, insight_object 이벤트에 의한 태스크, update_obj_vec 이벤트에 의한 태스크의 실행시간을 모두 합하여 실행시간을 계산하면 2+3+3+3이 되어 총 11ms가 된다. 같은 방식으로 go to ball 트랜잭션의 실행시간을 계산하면 28ms가 된다.

3. 메시지에 우선 순위 부여

트랜잭션에 우선 순위를 부여할 때는 DM(deadline monotonic)알고리즘에 근거하여 마감시한이 짧은 트랜잭션에 높은 우선 순위를 부여하였다.

4. 객체를 쓰레드에 지정

Vision Module은 짧은 마감시한을 가진 트랜잭션을 두 개 가지고 있다. image processing 태스크의 수행시간이 길기 때문에 다른 객체와 같은 쓰레드에 포함시킬 경우, 다른 트랜잭션의 마감시한에 영향을 줄 수 있다. 따라서 M.Saksena[10]가 제시된 설계 가이드라인에 따라 Vision Module은 하나의 쓰레드(쓰레드1)에 지정하고, 나머지 객체들은 다른 쓰레드(쓰레드2)에 지정하였다. 이렇게 지정하면 쓰레드1의 최대 RTC 블록킹 시간은 20ms가 되고, 쓰레드2의 최대 블록킹 시간은 8ms가 된다.

5. 블록킹 시간 계산

블록킹 시간은 앞장에서 제시되었던 식 $B_i = \max_j C_j$ (B_i =가상태스크 VT_i 의 RTC블록킹 시간, C_j=가상 태스크 VT_j의 실행시간)을 이용하여 구한다. image capture 트랜잭션은 쓰레드1의 최대 블록킹 시간인 20ms가 블럭킹 시간이 된다. image processing 트랜잭션은 5ms가 RTC 블록킹 시간이 된다. update model 트랜잭션의 블록킹 시간은 8ms, determine next action의 트랜잭션은 8ms, interpreter 트랜잭션은 8ms가 블록킹 시간이 된다.

6.스케줄링 가능성 분석

응답시간 역시 앞장에서 제시되었던 식 $R = \sum_{VTj \in hp(i)} \left(\lceil \frac{Ri}{Tj} \rceil * Cj \right) + Ci + Bi$ (Ri, Ci, Bi는 VT_i의 응답시간, 실행시간, 블록킹 시간, hp(i)는 VT_i 보다 높은 우선 순위를 가진 VT들의 집합)을 이용하여 구한다. update model 트랜잭션의 응답시간의 경우 image capture 트랜잭션, image processing 트랜잭션에 의한 방해, update model 트랜잭션의 자체 실행시간과 update model 트랜잭션보다 낮은 우선 순위의 태스크에 의한 RTC 블록킹 시간에 의해 응답시간

$R = \left(\lceil \frac{R}{100} \rceil * 5 \right) + \left(\lceil \frac{R}{100} \rceil * 20 \right) + 11 + 8 = 30$이 된다. 이런 식으로 계산하여 표로 나타내면 표3과 같고 모든 트랜잭션이 마감시한을 지키고 있음을 알 수있다. UML기반 축구 로봇 모델링의 스케줄링 가능성분석이 이루어짐으로써, 모델링된 시스템의 스케줄링 가능성분석에 필요한 각 변수들이 시퀀스 다이어그램의 시간제약에 관한 표현과 메시지의 시간적인 행동의 표현, 그리고 상태다이어그램에서 얻을 수 있음을 보여준다. 본 논문에서 사용한 응답시간 분석 식에서 필요한 변수들은 트랜잭션의 마감시한, 각 트랜잭션의 주기와 자체 실행시간, 낮은 우선 순위 태스크에 의한 블록킹 시간이다.

표 3. 자율 축구 로봇 제어 시스템의 스케줄 가능성분석

트랜잭션	주기	마감시한	우선순위	실행시간	블록킹	응답시간
Image capture	100ms	70ms	1	5	20	25
Image process	100ms	80ms	2	20	5	29
Update model	100ms	100ms	3	11	8	30
go to ball		200ms	5	28	8	60
dribble		200ms	5	25	8	57
avoidance		150ms	4	21	8	38

1)트랜잭션의 마감시한

트랜잭션의 마감시한은 시퀀스 다이어그램에서 timing marks를 이용해서 각 트랜잭션에 대한 마감시한이 나타난다.

2)각 트랜잭션의 주기

각 트랜잭션의 주기 또는 최소 도달시간은 시퀀스 다이어그램에서 각 트랜잭션마다 명시된 시간제약, PERIOD(EventID EventID')형태로 나타나거나, 트랜잭션을 유발시키는 메시지의 도달패턴의 정형확장으로 나타난다.

3)각 트랜잭션의 실행시간

각 트랜잭션의 실행시간 Ci는 시퀀스 다이어그램에서 트랜잭션을 구성하는 이벤트에 대한 정보와 상태 다이어그램을 참고하여 작성한 표2를 이용한

다. 트랜잭션을 구성하는 이벤트에 의해 각 객체 에서 수행되는 가상 태스크의 실행시간을 합하여 계산이 가능하다.

4)블록킹 시간

쓰레드에 객체들이 매핑 되었다면 표2를 참고하여 계산이 가능하다.

7. UML기반 축구로봇 모델의 시뮬레이션

(1)시뮬레이션 툴

상태차트에 바탕을 둔 객체지향 시뮬레이션 툴인 emultek사의 rapid 툴[14]을 사용하였다. rapid 툴은 다중 태스크(multi-tasking), 이벤트구동(event-driven)시스템, 그리고 내장형 시스템의 프로토타이핑(prototyping)의 시뮬레이션에 적합한 툴이다. 또한 UDO(User Define Object)를 사용하여 재사용성이 용이하고, 분석과정에서 나온 객체들을 바로 실행 가능한 객체로 만들기가 용이하기 때문에 분석 모델 동작에 대한 검증에 적합하다. 자율 축구로봇은 windows NT 운영체제에서 시뮬레이션 하였다.

(2)시뮬레이션의 구조 설계

시뮬레이션의 전체적인 구조를 UML의 컴포넌트다이어그램(component diagram)으로 나타내어 보면 그림12와 같다. player UDO는 환경으로부터 입수한 각 로봇과 골대, 그리고 공의 위치에 대한 정보를 가지고, 다음 행동을 결정하는 역할을 한다. player를 UDO로 구현함으로써 축구경기를 시뮬레이션 할 때 필요한 다른 로봇의 제작 시 재사용성이 증가한다. robocup RPD는 실제 시뮬레이션을 구동시키는 어플리케이션으로 각 UDO에서 받은 정보를 바탕으로 필드 상에 존재하는 각 로봇객체의 움직임과 변화하는 환경에 대한 시뮬레이션을 디스플레이 한다. 그림13은 각 컴포넌트들간의 관계를 보여준다.

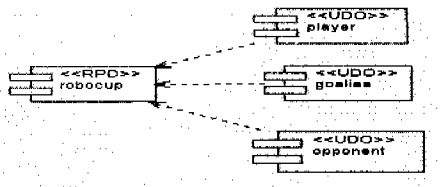


그림 12. 전체구조의 컴포넌트 다이어그램

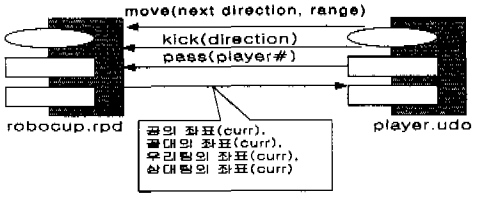


그림 13. 컴포넌트간의 관계

(3)player UDO와 RPD의 구성

player UDO는 자율이동로봇을 구성하는 객체들로 구성된다. 시뮬레이션을 위해서 구성객체들의 각각의 상태다이어그램들을 합쳐놓은 형태로 하여 모드 트리형식으로 나타내었다. 각 상태는 모드로 나타내어지고 활성객체(active object)의 상태를 동시모드(concurrent mode)로 나타내었다. 분석과정에서 나타난 객체내의 속성과 메쏘드는 상태의 행동 이벤트를 구성하게 된다. 실제 하드웨어를 구동하는 시간, 이미지를 프로세싱 하는 시간은 표2의 실행시간을 참고하여 시간지연을 주는 방식으로 구현하였다. robocup RPD 역시 필드 환경의 변화에 대한 상태를 모드 트리형식으로 나타내고 UDO 성질을 이용한다.

(4)최종 구현 모델

최종적으로 구현 된 모델은 그림14와 같다. 로봇1은 공을향해 이동한 다음 공을 잡은 다음에 다음 목표인 상대편 골을 향하여 이동한다. 이 과정중에 장애물인 상대편x를 만나면 일정거리이상 회피기동

표 4. 시뮬레이션 상에서의 응답시간

트랜잭션	주기	마감시한	우선순위	응답시간
image capture	100ms	70ms	1	40
image process	100ms	80ms	2	65
update model	100ms	100ms	3	75
go to ball		200ms	5	110
dribble		200ms	5	110
avoidance		150ms	4	102

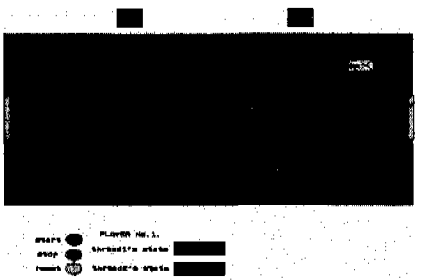


그림 14. 축구로봇의 동작화면

을 한다. 드리블 수행 중에 같은 팀 로봇2가 상대편 골의 적정 지점에 도달하면 그 로봇에게 공을 전달한다. 만일 로봇자신이 골의 적정지점까지 먼저 도달하면 스스로 골을 향해 슛을 한다. 어플리케이션 상에 로봇이 현재 수행하고 있는 트랜잭션이 표시되도록 하여 상황에 따른 로봇행동에 대해 확인하였다. 표4는 실제 어플리케이션 상에서 수행되는 각 트랜잭션의 응답시간을 확인하기 위해 툴에서 제공하는 time watch 객체를 이용하여 확인한 응답시간이다. 시스템의 부하를 고려하지 않은 상태인 표3의 응답시간분석에서 나타난 응답시간에 비해 응답시간의 지연이 있었지만 모두 마감시한을 만족함을 확인할 수 있었다.

Ⅵ. 결 론

본 논문에서는 최근 객체 지향 개발방법의 표준으로 채택된 UML에 스케줄링 알고리즘을 적용시켜 스케줄링 가능성분석이 이루어지는 방안에 대하여 제안하고 있다. 그 적용사례로 UML기반의 실시간 내장형 시스템 모델링의 전형적인 예인 로봇 축구대회를 위한 자율 축구 로봇을 모델링 하였고, 제시한 설계 가이드 라인을 바탕으로 모델링된 시스템에 실시간 스케줄링 알고리즘을 적용시켜 스케줄링 가능성 분석을 해보았다. 실시간 시스템 개발에 있어 모델링 방법론은 구현에 들어가기 전에 시스템에 대한 정확한 분석을 하고, 분석에 따라 시스템의 적정성 여부와 효과적인 설계를 유도해 낼 수 있으므로 상당히 중요한 의미를 가지고 있다. UML은 객체지향기술을 바탕으로 하고 있어, 분석에서 설계, 구현으로의 과정이 연속적으로 이어짐으로써 기존의 실시간 시스템의 주된 개발방법 이었던 구조적 기법중심의 개발 방법보다 소프트웨어의 개발기간을 단축할 수 있다. 이런 장점 외에 본 논문에서 제안하는 경성 실시간 시스템의 시간에 따른 행동을 UML의 확장 메커니즘인 정형(stereotype)과 시퀀스 다이어그램을 사용하여 표현함으로써 모델의 스케줄링 가능성 분석이 이루어질 수 있다. 따라서 본 논문에서 제안한 절차가 UML을 지원하는 실시간 시스템 개발 CASE 툴등에 적용되어질 경우 내장형 실시간 시스템 설계 및 분석의 효용성을 크게 향상시킬 수 있을 것이다. 앞으로는 본 논문의 연구 결과를 바탕으로 주기적인 메시지의 지터(jitter)에 관한 분석과 동적인 객체의 생성에 따른 우선 순위 반전에 대한 연구를 하고자 한다.

참 고 문 헌

[1] Unified Modeling Language for real-time design ver2.0, rational software, available at http://www.rational.com

[2] M.Awad, Object-oriented technology for real-time systems:pratical approach using OMT and fusion, Prentice Hall, 1996

[3] Wei-min chen, Autonomous soccer robots, Working notes of the AAAI 1997 fall symposium on model-directed autonomous systems

[4] Wei-min chen, Building integrated mobile robots for soccer competition, proceedings of 1998 international conference on robotics and automation

[5] A Burns, A.Wellings, Real-time systems and programming languages, Addison -Wesley,1996

[6] Keith Edwards, Real-time structured methods, Wiley and Sons

[7] David Harel, E.Gery, "Executable object modeling with statecharts", Proceedings of 18th Int. Conf. Soft. Eng, pp.246-257, March 1996

[8] Ivar Jacobson, Object-oriented software engineering: A use case driven approach, Addison-Wesley,1992

[9] James Rumbagh, Obct-oriented modeling and design. Prentice Hall, 1991

[10] M.saksena, "Guidelines for automated implementation of executable object oriented models for real-time embedded control systems," IEEE real-time system sympos ium,1997

[11] Bran Selic, Real-time object-oriented mode ling, p219, wiley,1994

[12] ITU-T, message sequence charts standard (z.120), International Telecommunication Union, 1994

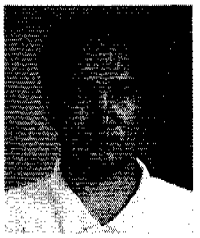
[13] Frank slomaka, MSC-based schedulability analysis

[14] Rapid user Manual, Emultek Ltd Press, 1996

이 재 익(Jae-ick Lee) 정회원

1997년 2월 : 경북대학교 전자공학과 졸업
1999년 2월 : 경북대학교 전자공학과 석사
1999년 3월~현재 : 경북대학교 전자공학과 실시간 시스템연구실 인턴연구원
<주관심 분야> 소프트웨어 공학, 객체지향 모델링
e-mail:jilee@rtlab.kyungpook.ac.kr

강 순 주(Soon-Ju Kang) 정회원



1983년 2월 : 경북대학교 전자공학과 (공학사)
1985년 2월 : 한국과학기술원 전산학과(공학석사)
1995년 2월 : 한국과학기술원 전산학과(공학박사)
1985년~1996년 8월 : 한국원자력연구소 핵인공지능연구실 선임연구원,전산정보 실 실장역임.
1996년 9월~현재:경북대학교 공과대학 조교수, IEEE, KISS, KIEE, KICS회원
<주관심 분야> 실시간 시스템설계, 지식처리형 소프트웨어 구조 및 소프트웨어 공학
e-mail:sjkang@ee.kyungpook.ac.kr

서 대 화(Dae-Dwa Seo) 정회원

1981년 2월 : 경북대학교 전자공학과(공학사)
1983년 2월 : 한국과학기술원 전산학과(공학석사)
1993년 2월 : 한국과학기술원 전산학과(공학박사)
1981년 3월~1995년 2월 : 한국전자통신연구소 근무
1998년 2월~1999년 2월 : University of California, Irvine 연구교수
1995년 3월~현재 : 경북대학교 공과대학 전자전기공학부 조교수
<주관심 분야> 병렬분산처리, 운영체제, 병렬컴퓨터 구조
e-mail:dwseo@ee.kyungpook.ac.kr