물리엔진을 이용한 효과적인 하이브리드 시뮬레이션 방법론

이완복*, 유석호**

An Efficient Hybrid Simulation Methodology Using the Game Physics Engine

Wan-Bok Lee*, Seuc-Ho Ryu**

요 약 사람이 만든 대부분의 시스템들은 하이브리드 시스템으로 모델링될 수 있다. 하이브리드 시스템 모델은 의사결정을 담당하는 상위레벨 모델과 기계적 제어를 담당하는 하위레벨 모델로 구성되기 때문에, 그 구조가 복잡하기 마련이며 모델을 해석할 때 모델의 종류에 따라 서로 다른 해석 기법이 요구된다. 상위레벨 모델은 FSM과 같은 이산사건시스템의 해석기법으로 풀이할 수 있으나, 하위레벨은 연속시간 모델로 구성되기 때문에 공학적 수치해석 기법이 동원되어야 한다. 개발자가 이 두 레벨의 모델을 함께 개발하는 방식에는 어려움이 많기 때문에 본 연구에서는 게임 분야에서 널리 사용되어지고 검증되어진 물리엔진 미들웨어를 이용하여 하위레벨의 모델링 및 해석 과정에 적용할 수 있는 방법을 제시한다.

주제어 : 시뮬레이션, 하이브리드 시스템, 물리 엔진, DEVS

Abstract Most of the man-made systems can be modeled as a hybrid system which consists of both the high-level and the low-level component model. High level model is responsible for decision-making and the low-level one takes control of the mechanical component parts. Since the two models requires different interpretation method according to their type, analysis of a hybrid system becomes a difficult job. For the Analysis of the high-level model, methods for discrete event system models such as FSM can be used. On the contrary, numerical analysis techniques are required for the low-level continuous-time system model. Since it becomes a difficult thing for a modeller specifies and develops both the two-level models altogether, we propose an efficient hybrid simulation method which employs a game physics engine that has been widely and successfuly used in the area of game industry.

Key Words : Simulation, Hybrid system, Physics engine, DEVS

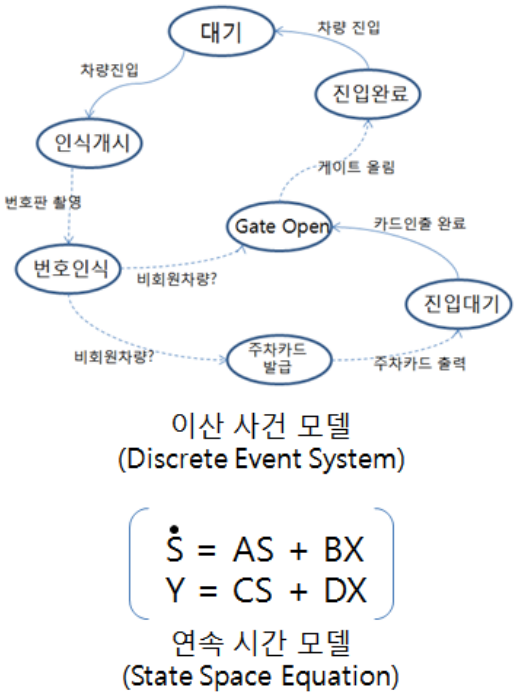
1. 서론

컴퓨터 시뮬레이션은 시스템을 구현하기 이전에 시스템의 특성 검증, 성능 평가 등의 목적으로 많이 활용되고 있다. 특히, 시뮬레이션은 계산량이 매우 많으며, 오랜 시간이 소요되기 때문에 효과적인 모델링 및 시뮬레이션 방법이 요구된다. 병렬 분산 시뮬레이션[3], 컴파일드(Compiled) 시뮬레이션[4] 등은 모두 이러한 방법들에 속한다고 볼 수 있다.

시뮬레이션 속도 문제 이외에도 사람들이 만든 시스템(Man made systems)은 논리적인 제어를 담당하는 부분과 기계적인 장치에 해당하는 물리적인 부품장치들로 구성되어 있기 때문에 모델링이 매우 어려운 문제점이 있다. 예를 들어 [그림 1]에서 보이는 바와 같이 주차장 출입통제 시스템의 경우, 차량이 진입할 경우 차량번호를 인식한 후 해당차량이 회원의 차량인지 외부차량인지

*공주대학교 게임디자인학과
**공주대학교 게임디자인학과 (교신저자)
논문접수: 2012년 10월 27일, 1차 수정을 거쳐, 심사완료: 2012년 11월 20일

판단하고 출입카드를 발급해야 할 대상인지를 판단해주는 의사결정 모델이 탑재되어 있다. 이러한 부분은 시스템 상태와 그 천이 관계를 나타내는 이벤트를 기본 요소로 하여 모델링하는 FSM(Finite State Machine) 모델과 번호인식과정을 처리하는 신경회로망회로나 유전자 알고리즘 모델로 구성할 수 있다. 상위레벨의 모델에서 게이트를 열어주기로 판단하였다면, 하위레벨의 모델에서는 전기모터나 유압 액추에이터를 이용하여 게이트를 물리적으로 들어 올리게 된다. 이때 소요되는 시간이나 게이트의 괘적 등은 힘과 질량이 작용하는 운동역학 방정식[5]을 풀어서 계산할 수 있다.



[그림 1] 하이브리드 시스템 모델의 예: 주차 차단기

이와 같이 간단한 시스템이라도 사람이 만든 시스템은 의사결정을 담당하는 상위레벨의 모델과 단순히 기계적 조작을 담당하는 하위레벨의 모델로 구성되어지는 것이 일반적이다. 특히 요즘에는 가정에서 사용하는 세탁기, 오디오를 비롯하여 심지어 어린이 장난감에 이르기까지 마이크로프로세서가 광범위하게 적용되기 때문에 위와 같은 하이브리드 시스템의 예를 쉽게 접할 수 있다.

그러나 하이브리드 시스템을 구성하는 두 계층의 모델은 해석과정에서 서로 상이한 풀이 방법이 동원되어야 한다. 상위모델은 이벤트에 의해 각 객체들이 영향을 주고 받으면서 자신의 상태를 변화하기 때문에 이산사건 시스템 모델로 수월하게 표현되고 해석될 수 있는 반면, 하위 레벨의 모터 제어나 액추에이터 제어와 관련된 부분은 기존 Runge-Kutta Method, FEM(Finite Element Method), FDM(Finite Differential Method) 등과 같이 State Space Equation을 풀이하는 수치해석기법이 요구된다. 이 두 가지 풀이 기법은 모델의 표현에서부터 그 해석 접근 방식이 매우 다르기 때문에 사람이 만든 시스템들의 해석에는 DES(Discrete Event System) 모델해석과 연속시간(Continuous Time Model) 모델 해석 기법을 상호 잘 연동시켜야 하는 어려움이 있다.

이러한 복잡성으로 인하여 하이브리드 시스템의 모델링 및 시뮬레이션 방법론에 대하여서는 기존에 진행되어진 관련 연구가 적은 편이다. [1][2]에서는 연속시간 모델 시뮬레이터로 MATLAB을, 그리고 이산사건 시뮬레이터로 DEVSim++를 사용하여 하이브리드 시뮬레이션을 성공적으로 수행한 예에 대해서 밝힌바 있다. 그러나, [1]의 방법에서는 연속시간 모델에 대하여 모델러가 미적분 방정식을 작성하고, 그것을 수치해석 라이브러리를 이용하여 Solver를 작성해주어야 하는 불편함이 있다. 이러한 방식은 모델러로 하여금 매우 어려운 작업을 요구하는 것일 뿐만 아니라, 작성한 모델의 정확성에 대해 검증하기 어려운 문제점을 남기게 된다.

이에 본 연구에서는 연속시간 모델에 대해서 게임분야에서 널리 사용되어지고 있는 물리엔진 미들웨어를 이용하여 연동함으로써 손쉽게 하이브리드 시스템 모델을 구축하고 시뮬레이션 할 수 있는 방안을 구축하고자 한다. 게임 물리엔진을 이용할 경우 매우 정교하게 물리 모델을 표현하고 해석하는 것은 어려울 수 있지만, 일반적으로 접하는 많은 경우의 물리 현상에 대해 그 모델을 쉽게 표현하고 해석할 수 있으며, 사용자 편의성이 매우 뛰어나다는 장점을 가진다.

2. 관련연구

2.1 게임 물리 엔진

게임 물리 엔진은 게임 월드 속에서 요구되는 각종 물리현상의 연산을 쉽게 할 수 있도록 물리 연산 기능에 대한 편의를 제공하고 그 연산을 가속화시켜주는 미들웨어

이다. 물리엔진이 지원하는 주요기능으로서는 강체동역학, 충돌감지, 중력장의 영향, ragdoll 시뮬레이션, cloth 시뮬레이션, 파티클 효과 등이 있으며, 사용자와의 상호연동을 높이고 몰입감을 증진시키기 위해서 최근에 출시되는 3D 게임에서는 꼭 필요한 요소가 되어가고 있다. 예를 들어, 물리엔진이 적용되지 않은 게임에서 특정 물체가 과도한 힘을 받아 파괴되는 장면을 연출할 시, 미리 정해진 파괴 애니메이션에 따라 표현되기 때문에, 외부 힘이 가격된 위치와 그 세기에 관련없이 파괴되는 장면을 보이기 때문에 현실감이 떨어지게 되며, 게이머들의 몰입도를 저하시키게 된다. 반면에 물리엔진이 적용되면, 힘을 받은 부위를 중심으로 파괴가 진행되는 애니메이션을 보여줄 수 있기 때문에 사용도의 몰입도를 높이고, 더욱 생동감 있는 장면 연출이 가능해진다.

주요 물리엔진으로는 <표 1>.에서 보이듯이 상용제품으로 NVIDEA사의 PhysX와 Havok사의 제품이 있다. PhysX는 SDK 개발에 역점을 두어 현재 PC 플랫폼 게임에서 가장 많이 채택되고 있으며, 반면에 Havok 제품은 강력한 기능과 성능으로 인해 콘솔 플랫폼에서 가장 많이 선호되고 있다.

〈표 1〉 주요 게임 물리 엔진의 비교

물리엔진명	제조사	구분	적용된 게임 수 (2006년 ~ 2009년)	특징
PhysX	NVIDEA	상용	200	GPU/PPU 지원
Havok	Havok	상용	181	Intel에 흡수합병
ODE	공개 프로젝트	공개	37	Open source
Newton		공개	12	Free, but closed source
Bullet	공개 프로젝	공개	7	Open source

3D 공간에서 물리엔진을 이용하면, 3차원 공간 상에서의 충돌처리, 객체간 거리 및 상호작용 연산을 매우 쉽게 할 수 있다. 폴리곤 모델에 콜라이더(Collider)를 지정해주면, 콜라이더간 충돌이 있을 경우에 콜백함수를 통해 객체간 충돌여부를 3차원 공간에서 파악할 수 있으며, 중력장과 물리 머티어리얼을 이용하면, 객체간 작용하는 마찰력과 중력장에 의한 역학관계를 매우 쉽게 반영할 수 있는 잇점이 있다.

2.2 DEVS 모델링 및 시뮬레이션

시뮬레이션 연구 분야에서는 모델을 수학적으로 명세할 수 있으며, 그 동작성에 대해 검증할 수 있는 모델링 방법에 대해 연구되어진 바 있다[6][7][8]. 특히, 상위레벨의 의사결정을 모델링하는 이산사건 모델에서는 많은 독립적인 랜덤변수들이 사용되어지며, 그 결과 또한 랜덤하게 얻어지기 때문에 모델의 정확성을 따져보는 것이 매우 어렵다.

Zeigler에 의해 개발된 DEVS 모델은 이산사건 시스템을 모듈러하고도 계층적으로 모델링 할 수 있는 방법을 제시하고 있는데, 수학적으로 모델을 명세할 수 있기 때문에 모델의 정확성을 따져보기가 용이한 장점이 있다[7]. 또한 DEVS 모델을 시뮬레이션할 수 있는 도구들이 Devsim++를 비롯하여 몇 가지가 이미 개발 되어 진 바 있다[9][10][11].

DEVS 모델에서는 모델을 계층적으로 분해할 수 있도록 원자 모델(Atomic Model)과 결합 모델(Coupled Model)의 두 가지 클래스로 나누어서 모델을 표현하고 있다. 원자 모델은 더 이상 나눌 수 없는 모델로서 각 구성요소의 동적 특성을 시간명세 상태전이시스템(timed state transition system) 레벨에서 기술한다. 원자 모델 AM의 수학적 정의는 다음과 같다.

$$
\begin{aligned}
AM &= \langle X, S, Y, \delta_{int}, \delta_{ext}, \lambda, ta \rangle \\
X &: \text{입력사건집합} \\
S &: \text{상태집합} \\
Y &: \text{출력사건집합} \\
\delta_{int} &: S \to S : \text{내부상태전이함수} \\
\delta_{ext} &: Q \times X \to S : \text{외부상태전이함수} \\
\lambda &: S \to Y : \text{출력함수} \\
ta &: S \to Real : \text{시간전진함수} \\
Q &= \{(s,e) \mid s \in S, 0 \le e \le ta(s)\} \\
&: \text{total state of AM}
\end{aligned}
$$

결합 모델 CM은 구성요소 모델들 간의 연결 관계를 표현하는 모델로서 구성요소 모델들을 조립하여 더 큰 단위의 모델을 만들 수 있게 한다.

DEVS 모델에 대한 자세한 내용은 [6][7]에서 참조할 수 있다.

$$CM = \langle X, Y, \mathrm{M}_i, \mathrm{EIC}, \mathrm{EOC}, \mathrm{IC}, \mathrm{SEL} \rangle$$

X : 입력사건집합

Y : 출력사건집합

M_i : DEVS컴포넌트집합

$EIC \subseteq X \times \bigcup_i X_i$: 외부입력관계

$EOC \subseteq \bigcup_i Y_i \times Y$: 외부출력관계

$IC \subseteq \bigcup_i Y_i \times \bigcup_i X_i$: 내부입출력관계

$SEL : 2^{M_i} - \Phi \rightarrow \mathrm{M}_i$

3. 하이브리드 시뮬레이션 방법론

앞부분에서 이미 하이브리드 시스템 모델은 의사결정을 표현하는 상위레벨의 DES 모델과 기계적 제어를 표현하는 하위레벨의 연속시간 모델로 구성 된다는 것을 설명하였다. 이 두 모델은 사용하는 형식론이 전혀 다르며, 두 모델을 해석하는 Solver의 형태도 전혀 다르기 때문에 하이브리드 시뮬레이션을 진행하기 위해서는 서로 다른 두 종류의 모델들 간 연동 메카니즘을 정의하는 것이 필요하다.

3.1 레벨별 모델의 명세

강체의 비행 궤도, 충돌 여부의 판단과 같이 하위레벨의 모델은 3차원 공간 안에서 작용하는 힘과 물체의 물리량에 대해 미적분 방정식을 계산하여야 하므로 매우 복잡하고 모델링하기 어려운 문제점이 있다. 하지만, 물리엔진을 적용하면, 기본 파라메터 설정을 통하여 현실세계에서 실현되는 대분분의 물리현상에 대해 표현이 가능하다. <표 2>는 최근 많이 이용되고 있는 Unity3D 게임엔진에 실장되어 있는 PhysX 물리엔진에서 지정할 수 있는 주요 파라메터들이다. 물론 LOD(Level Of Detail)가 매우 높은 모델을 표현하기 위해서는 이러한 물리엔진에서 제공하는 파라메터만을 이용해서는 정확히 표현할 수 없으므로 적용하기 어려운 문제점이 있다. 하지만, 많은 경우에 있어서는 하위레벨 모델에 대해 매우 수월하게 표현하고 물리 계산 결과를 얻을 수 있다.

상위 레벨의 행위 모델은 DEVS 모델로 명세하면 여러 가지 잇점이 있다. DEVS 모델은 수학적인 명세가 가능하기 때문에 모델 검증에 유리하며, 기존에 개발되어진 다양한 실행 라이브러리와 도구들을 이용하여 시뮬레이션을 쉽게 적용할 수 있는 잇점이 있다. DEVS 모델은 그 구조상 계층적인 모델링이 가능하도록 Coupled Model과 Atomic Model이 있으므로, 모델이 복잡하고 대규모일수록 정교하게 모델링할 수 있는 잇점이 생긴다. DEVS 모델로 표현하기 효과적인 대상은 마이크로 프로세서에서 동작하는 여러 수학적인 계산 또는 알고리즘 연산 모듈 등이 될 수 있다.

〈표 2〉 강체동역학과 관계된 주요 파라메터들

Parameter	Description
velocity	The velocity vector of the rigidbody.
angularVelocity	The angular velocity vector of the rigidbody.
drag	The drag of the object.
angularDrag	The angular drag of the object.
mass	The mass of the rigidbody.
useGravity	Controls whether gravity affects this rigidbody.
centerOfMass	The center of mass relative to the transform's origin.
worldCenterOfMass	The center of mass of the rigidbody in world space
inertiaTensorRotation	The rotation of the inertia tensor.
inertiaTensor	The diagonal inertia tensor of mass relative to the center of mass.

3.2 시간 및 데이터 동기화

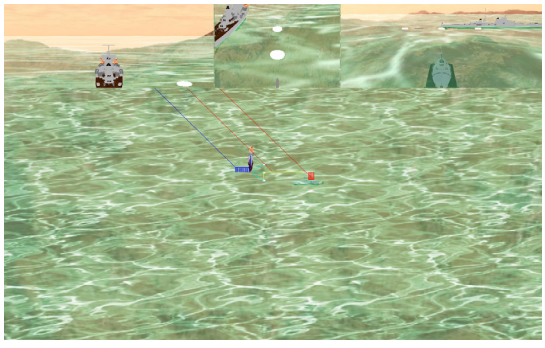
두 개 이상의 시뮬레이터가 연동되어 진행되려면 시간 동기화와 데이터 동기화가 이루어져야 한다[3][5].

먼저 시간 동기화를 맞추기 위해서는 연속 시뮬레이터의 시뮬레이션 진행 구간의 시작 시각과 끝마치는 시각이 이산사건 시뮬레이터의 사건 발생 시각과 동기화되어야 한다. 게임 물리엔진을 적용할 경우에는 연속 시뮬레이터의 시간은 실시간으로 진행되어지며, 반복적인 Game Loop의 실행과 더불어서 시뮬레이션 시간이 점진적으로 진행하게 된다. 반면에 이산사건 시뮬레이터에서는 가상적인 시뮬레이션 시간을 전진시키면서 발생된 이벤트들에 대해 그 영향 관계를 반영시켜주는 콜백 함수를 호출함으로써 처리하게 된다. 이 경우의 시간 동기화는 Game Loop 내에서 이산사건시뮬레이션의 시간 전진 여부를 제어함으로써 해결 할 수 있다. 즉 현재의 Game Loop 내의 진행 시간보다 가상 시뮬레이션 시간이 전진되도록 이산사건 시뮬레이터의 실행을 진행하고, 다음 발생 이벤트의 스케줄 시간을 Polling 하면서 Game Loop이 진행되도록 제어함으로써 해결 할 수 있다.

두 번째로 데이터 동기화를 만족시키기 위해서는 이산사건 데이터와 연속시간 데이터 간의 상호 변환이 필요하다. 비슷한 예로써 아날로그 회로와 디지털 회로가 상호 연동되는 경우, 디지털 회로의 신호는 D/A 컨버터를 통해서 아날로그 신호로 변환되어지며, 아날로그 회로의 신호는 A/D 컨버터에 의해 디지털 신호로 변환되어 상호 연동될 수 있다. 이러한 개념에서 A/E 변환기와 E/A 변환기가 제안된 바 있지만[2], 게임 물리 엔진을 적용할 경우에는 물리엔진에서 정해진 콜백함수를 호출해주기 때문에, 호출되는 콜백함수에서 입출력관계를 연계해줌으로써 데이터 동기화를 할 수 있다. 예를 들어 Unity3D 게임엔진에서는 OnCollisionEnter와 같은 콜백함수 안에서 이산사건시뮬레이터에서 인식 가능한 데이터 형태로 변환을 해주어 해결할 수 있다.

4. 적용 사례

앞에서 제안한 하이브리드 시뮬레이션 방법을 적용하여 대잠수함 예제에 적용하여 보았다. 이 예제에서는 군함과 잠수함이 등장하며, 잠수함이 발사한 어뢰에 대응하는 군함의 방어가 효과적으로 이루어졌는지 살펴볼 수 있도록 시뮬레이션하여 확인하였다. 잠수함은 수면 아래에서 인근 가까운 거리에 있는 군함에 대해 탐지할 수 있는 능력이 있다고 가정하였다. 일정 거리 이내에 군함이 있을 경우, 잠수함에서는 어뢰를 발사하게 되며, 발사된 어뢰는 군함을 향해 돌진하게 된다. 어뢰가 가까이 다가오게 되면, 군함에서도 어뢰를 인지하게 되며, 탈출을 시도하기 위해 기만기를 발사하고, 어뢰가 기만기를 추적하는 동안 군함은 진행방향을 바꾸어 나아가게 된다.



[그림 2] 시뮬레이션 화면

시뮬레이션은 Unity3D 게임 저작도구를 이용하여 개발하였다. 군함과 잠수함의 행위 모델은 DEVS로 모델링하였으며, 모델의 실행을 위해 시뮬레이션 라이브러리인 DEVSim++을 C#언어를 이용하여 Unity3D 상으로 이식하였다. 또한, 어뢰와 기만기, 그리고 어뢰와 군함의 충돌체크를 확인하기 위하여 각 객체마다 콜라이더를 입히고 Rigidbody 콤포넌트를 추가하였다.

시뮬레이션 결과를 효과적으로 보여주기 위해 시뮬레이션 화면을 [그림 2]에서 보이는 바와 같이 4등분 하였다. 가장 큰 윈도우에서는 모든 객체들을 한꺼번에 표시해주고, 주요 객체인 군함, 잠수함, 기만기를 위주로 확대하여 보여주는 화면을 두었다. 다중 화면은 게임저작도구에서 여러 카메라를 설정함으로써 구성할 수 있었다.

5. 결론

본 연구에서는 이산사건시스템 모델과 연속시간 모델로 구성되는 복잡한 하이브리드 시스템의 모델을 쉽게 구축하고 시뮬레이션 할 수 있는 방법을 제시하였다. 제시한 방법은 기존의 검증된 미들웨어를 활용해 하이브리드 시스템 모델을 구축하고 시뮬레이션하는 것이기 때문에 모델 구축이 어렵고 그 해석이 어려웠었던 기존의 문제점을 해결할 수 있었다.

그러나 현재 적용한 물리엔진 소스코드 레벨에서 완전히 공개된 것이 아니기 때문에 복잡하고 정교한 물리현상에 대해 표현 및 모델링할 수 없으며, 각종 제어 및 일반적인 수치해석 알고리즘을 적용하거나 확장할 수 없는 한계점이 있다. 그러므로 본 논문에서 제안한 방법이 모든 하이브리드 시뮬레이션 영역에서 적용될 수 있는 것은 아니다. 그렇지만 하이브리드 시뮬레이션 과정에서 빈번하게 사용되어지고 꼭 필요한, 충돌 처리 및 동역학 관계의 반영, 중력장의 효과 표현 등에는 매우 유용하게 적용될 수 있다. 더욱 정교한 물리 시뮬레이션이 필요하다면 연속시간 모델의 시뮬레이터를 이산사건 시뮬레이터와 직접적으로 연동함으로써 해결할 수 있다. 이 경우에는 이종 시뮬레이터간의 시간 동기화 및 데이터 동기화를 보장할 수 있는 메카니즘을 추가로 마련해주어야 한다.

유비쿼터스 컴퓨팅 환경과, 각종 센서들이 발전함에 따라 우리 생활 주변의 많은 물건들이 마이크로프로세서

와 디지털 센서 그리고 DC모터로 구동되어지고 있다. 이러한 시스템들은 모두 하이브리드 시스템으로 구축할 수 있기 때문에 효과적으로 하이브리드 시스템을 모델링하고 시뮬레이션 할 수 있는 방법론은 그 중요성이 더욱 커질 것으로 기대된다. 게임 산업계에서 주목받았던 물리엔진이 최근에는 영화와 애니메이션 분야에서 적극 활용되어지는 경향을 보이고 있으며, 그 연산을 가속화해줄 수 있는 하드웨어 플랫폼도 잇따라 등장하고 있다. 이러한 경향으로 볼 때 시뮬레이션 분야에서도 게임 물리엔진을 효과적으로 그리고 적극적으로 이용하는 시대가 곧 도래 할 것으로 기대된다.

참 고 문 헌

[1] 임성용, 김탁곤 (2001). 하이브리드 시스템 모델링 및 시뮬레이션 - 제1부: 모델링 및 시뮬레이션 방법론. 한국시뮬레이션학회 논문지, 10(3).

[2] 임성용, 김탁곤 (2001). 하이브리드 시스템 모델링 및 시뮬레이션 - 제2부: 시뮬레이터 연동 환경. 한국시뮬레이션학회 논문지, 10(3).

[3] R. M. Fujimoto (1990). Optimistic Approaches to Parallel Discrete Event Simulation. Trans. of The Society for Com-puter Simulation, 7(2), 153-191.

[4] D. M. Lewis (1991). A hierarchical compiled code event- driven logic simulator. IEEE Transactions on Computer-Aided Design, 10(6), 726-737.

[5] Francois E. Cellier and Ernesto Kofman (2006). Continuous System Simulation (first ed.). Springer, ISBN 978-0-387-26102-7.

[6] Bernard Zeigler (1976). Theory of Modeling and Simulation (first ed.). NY: Wiley Interscience.

[7] B. P Zeigler (1984). Multifacetted Modeling and Discrete Event Simulation. Orlando, FL: Academic Press.

[8] A. I. Concepcion, and B. F. Zeigler (1988), DEVS Formalism: A Framework for Hierarchical Model Development, IEEE Trans. on Software Engineering, 14(2).

[9] Bergero, Federico and Kofman, Ernesto (2011) PowerDEVS: a tool for hybrid system modeling and real-time simulation. San Diego: Society for Computer Simulation International.

[10] D. Baezner, G. Lomow, and B.W. Unger (1990). Sim++: The Transition to Distributed Simulation. In Proceedings of the SCS Multiconference on Distributed Simulation.

[11] DEVSim++ User's Manual. available by anonymous ftp to sim.kaist.ac.kr.

이 완 복

- 2004년: KAIST전자전산학과 전기 및 전자공학 전공(공학박사)
- 2007년: 現, 공주대학교 게임디자인학과 교수
- 관심분야 : 게임엔진, 시뮬레이션, 이산사건시스템
- E-Mail: wblee@kongju.ac.kr

유 석 호

- 1994년: 뉴욕공대 커뮤니케이션아트 졸업
- 2003년: 現, 공주대학교 게임디자인학과 교수
- 관심분야: 게임그래픽디자인, 멀티미디어
- E-Mail: seanryu@kongju.ac.kr