이산사건 형식론을 이용한 전자전 조우 모델링 연구

윤 기 천

국방과학연구소

Ⅰ. 서 론

1-1 전자전 조우 시뮬레이션

전술항공기의 임무 계획시 적의 대공방어 체계로 밀집되어 있는 방공망을 통과하여 임무를 완수하기 위하여는 다음과 같은 사전의 임무계획이 필요하게 된다.

첫째, 적의 전자전투서열(Electronic Order of Battle, EOB)에 근거하여 조우될 위협체계 (레이더, 유도탄, 요격기 및 대공포 등)의 성능과 탑재된 전자전 장비의 성능을 고려하여 적의 레이더에 가장 적게 노출되고, 또한 노출되더라도 ECM으로 처리 가능한 최적의 비행경로를 선정하여야 한다. 이 최적의 경로는 항공기의 피격률이 최소가 되는 경로이여야 하며, 임무 수행이 원활히 추진될 수 있는 경로이어야만 한다.

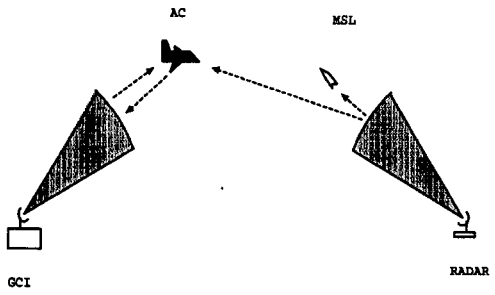
둘째, 굴곡이 심한 한국의 지형특성을 최대한 이용하여야 하며, 또한 아군 항공기의 성능과 기종 및 전술개념이 종합적으로 고려되어야 한다.

셋째, 항공기의 이륙시점에서 임무 완수 후 귀환까지의 경로 중 어느 지점에서 어떤 환경이 주어지며, 어떤 상태로 작전임무가 수행되는 지와 전체 과정 중에 주의해야 될 것들은 무엇인지를 알려주는 전술요구사항이 도출되어야 한다.

넷째, 전체적인 일련의 임무수행 과정을 화면 상에서 시뮬레이션 시킴으로써 작전을 수행할 조종사가 미리 알고 출격할 수 있는 예비훈련 및 숙지과정이 성공적인 임무완수를 위하여 반드시 필요하게 된다.

이상과 같은 전자전의 동적조우 시뮬레이션 연구는 전자전 장비의 운영뿐만 아니라 작전의 성공률을 향상시키는 수단이 되므로 매우 중요한 연구라 할 수 있겠다. 선진 각국에서는 적의 대공방어 레이더 및 무기의 제원, 배치형태, 위치 등의 위협정보와 아군의 침투기종, 탑재 운용되는 전자전 체계 제원 등의 전술정보가 입력자료로 사용되어 항공 작전 계획시 적의 대공방어 능력 및 위험도를 분석하고, 아군 전자전 능력을 감안한 최적 침투계획을 도출하며, 임무 성공률 및 작전시 고려 사항 등을 출력시켜 활용함으로써 전자전 장비의 효율을 증대시키고 항공작전 계획을 과학화 하고 있으며, 최근 걸프전에서 활용하여 그 효과를 입증한 바 있다.

공군전력을 기본으로 하는 미국은 현재 AFEWES (Air Force EW Center)에서 개발한 IMOM (Improved Many-On-Many)라는 임무계획장비를 운용하고 있으며, 이 분야에 대하여 1970년대 초부터 연구를 시작하여 많은 기술을 축적하였고, 컴퓨터에 의한 위



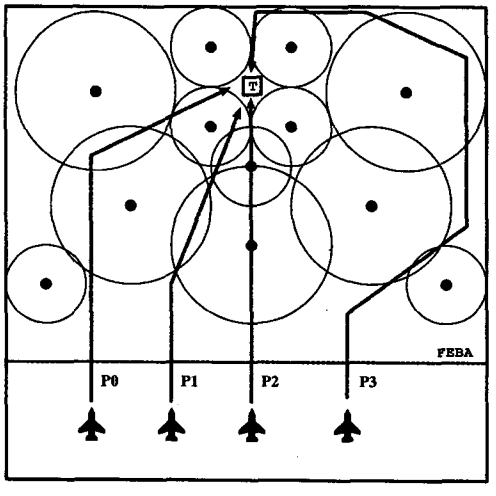
[그림 1] 항공기와 레이다의 일대일 조우모형

협정보처리, 전술기의 최적침투경로 선정 및 Cruise 미사일 비행경로 선정 등의 실용화를 완료하였으며 계속적으로 연구를 추진하고 있는 추세이다.

그러나 Wargame 시뮬레이션에 관련한 연구결과가 전문 학술지에 부분적으로는 발표되지만 [KB91][AG92][FS89][MT94][NEB94], 자세한 내용이 실리지 않는 이유는 기술보호상의 문제도 있으나 마땅한 모델링 이론과 관련 체계의 연구가 이직도 미흡하기 때문인 것으로 판단된다.

1-2 연구 개요

본 연구에서는 여러 대의 침투항공기가 다수의 다종 레이더 체계로 밀집된 방공망을 통과할 때 발생되는 사건들을 대상 시나리오로 선정하였다. [그림 1]과 같이 항공기와 레이더의 일대일 대응은 간단한 최소 대응 모델로 고려될 수 있으며, [그림 2]와 같이 다수의 항공기와 여러 대의 레이더 체계와의 대응은 복잡한 상호관계의 정립 및 체계적인 해석을 필요로 한다. 이러한 동적 조우 상황을 시뮬레이션 하기 위하여는 각 객체의 모델링을 위한 체계적인 형식론 및 편리한 시뮬레이션 환경을 필요로 한다.



[그림 2] 다중 조우 모형

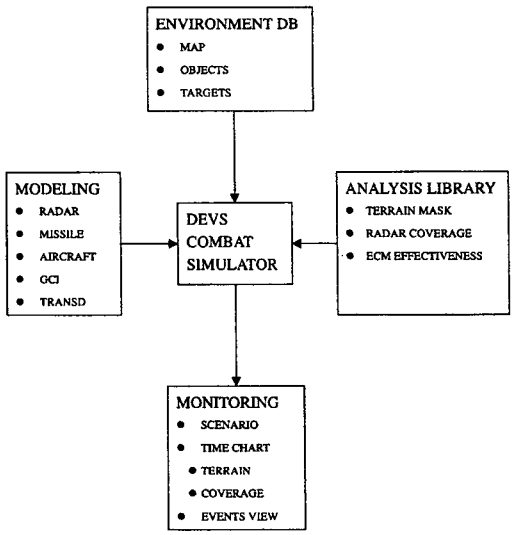
본 연구에서는 동적 조우 환경을 이산사건 시스템의 형식론인 DEVS (Discrete Event Systems Specification)로 모델링하고 이산사건 시뮬레이터인 DEVSim++을 이용하여 시뮬레이션 하였다. 전형적인 시나리오에서 실험한 결과 조우 상태 및 레이더의 상태 천이 등을 정확히 도출할 수 있었으며 임무 수행에 필요한 전자전 효과를 수치적으로 도출할 수 있었다. 본 연구결과는 전자전 분야의 동적 조우상황 분석 및 임무 계획, 예측 및 훈련분야에 활용될 수 있을 것이다.

Ⅱ. 시뮬레이션 체계 구성

2-1 체계 구성

전자전의 조우상황은 실제적으로 매우 복잡하지만 다음과 같이 간략한 모델로 구성할 수 있다. 즉, 탐지거리, 추적거리, 교전거리, 안테나 비임 형태 등 여러 종류의 레이더를 지닌 지대공 유도탄(SAM) 및 대공포(AAA) 체계가 지역적으로 널리 배치되어 있고, 이러한 대공방어체계를 ECM 장비를 가진 여러 종류의 전투기들이 방어망을 돌파하는 [그림 2]와 같은 전형적인 시나리오로 간주할 수 있다. 이러한 전자전 시나리오를 시뮬레이션하기 위한 시스템의 구성은 [그림 3]과 같이 Environmental Database, Modeling, Analysis Library, Monitoring 및 DEVS Combat Simulator로 구성하였다.

여기서 지형 DB는 수치고도(DTED) 자료를 사용하였고, Object DB는 ECM 조우환경을 구성하고 있는 위협(Radar와 Missile), 표적, 항공기 등을 표현하고 있다. 모델링은 DEVS 형식론에 입각하여 ECM 환경의 구성 요소인 Radar, Missile, Aircraft, GCI, TRANSD를 표현하고 있다. 이들의 각각의 기능은



[그림 3] 시스템 구성도

다음 장에서 자세히 설명하기로 한다.

Analysis Library는 ECM 환경분석에서 요구되는 표적지역의 지형차폐효과 분석, 레이더 탐지거리 분석 및 ECM 효과분석 프로그램들로 구성된다.

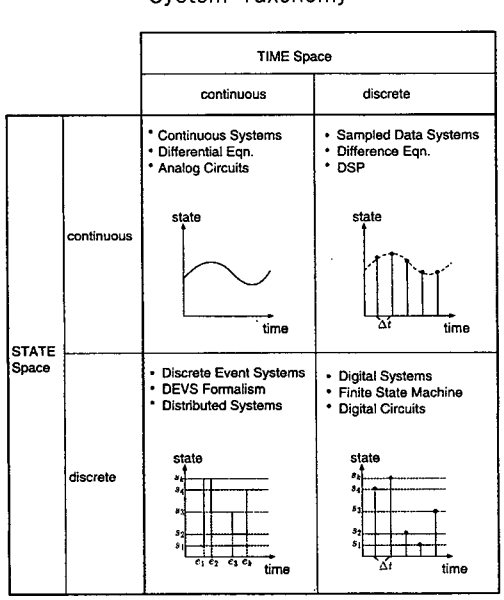
Monitoring은 지형차폐 및 탐지범위 분석내용을 확인할 수 있으며, 시뮬레이션이 진행됨에 따라 발생되는 모델들의 동작 및 각종 event들을 화면으로 표시해 주며 시뮬레이션이 완료되면 각종 event들을 시간상으로 도시해 주는 Time Chart 자료를 표시하여 준다.

DEVS Combat Simulator는 DEVS 모델들을 사용하여 임의로 선택한 표적에 대하여 각각의 동작들을 시뮬레이션하고 그 내용을 Monitor로 출력하게 된다.

Ⅲ. 이산사건 시스템

3-1 시스템 분류

시스템은 일반적으로 [그림 4]와 같이 시간영역과 상태공간에 따라 네가지 형태 즉, 연속시스템(continuous system), 이산 시간 시스템(discrete time system), 디지털 시스템 (digital system) 및 이산 사건 시스템 (discrete event system) 으로 분류할 수 있다 [TGKim-1992]. 여기서 이산사건 시스템이란 시스템의 상태변수가 임의의 시간에서 순간적으로 변하며 한 상태에 머무르는 시간이 불규칙적인 시스템을 말한다. 여기서 상태 공간은 유한개의 이산적인 값으로 구성되나 시간영역은 연속적인 값으로 구성된다. 이산 사건 시스템의 예로는 multiprocessor 시스템, 컴퓨터 통신 네트웍, 교통제어 시스템, wargame 등을 들 수 있다. Continuous system의 수학적 모델링에 미분방정식을 사용하는 것과 같이 Discrete event system을 수학적으로 모델링하기 위한 형식론으로서 Temporal logic, Timed Petri nets [PET81] 및 DEVS formalism [ZEI84] 등이 있으며,



[그림 4] 시스템 구분

이 중에 계층적 구조를 가지고 모듈화된 모델로 모델링하는 형식론이 DEVS 형식론이다. Zeigler에 의해 1984년에 창시된 DEVS (Discrete Event System Specification) 형식론은 이산사건 시스템을 모듈로 나누어서 이들을 계층적으로 모델링 할 수 있는 방법을 제시한다. 즉 DEVS 형식론은 atomic 모델들과 coupled 모델들을 명세함으로써 효율적이고 형식적(formal)인 모델링 할 수 있게 한다. Atomic 모델은 모델의 가장 작은 단위이며, 이들을 component로 가지면서 그들의 연결명세 (coupling scheme) 를 명세하는 모델이 coupled 모델이다.

객체 지향 모델링 시뮬레이션 환경인 DEVSim++은 DEVS 형식론을 C++ 언어로 구현한 것이다. 객체 지향형 프로그래밍 paradigm과 이산사건 모델링과의 호환성은 매우 잘 적용된다. 이러한 객체지향 언어에서의 객체(Object, An instance of class)는 data structure와 이를 다루기 위한 operation을 함께 가지고 있다. 객체지향형 언어의 특징으로는 information hiding, data abstraction, dynamic binding, inheritance 등이 있으며, 이러한 특징들은 기존의 순차적 언어에 비해 많은 장점을 가지게 된다. Information hiding과 data abstraction은 프로그램의 신뢰성을 높이며, dynamic binding은 기존 code를 수정할 필요 없이 새로운 class를 추가할 수 있게 함으로써 프로그램의 유연성을 높이며, inheritance는 code를 효율적으로 재 사용할 수 있게끔 한다. 객체는 실제 시스템을 바로 표현할 수 있으므로 프로그래밍 시스템에서의 객체와 실제 현상의 모델 요소는 일대일로 잘 대응될 수 있다. 따라서 객체지향언어의 하나인 C++은 이산사건 시스템의 모델링 및 시뮬레이션에 적합하며, 사실상 C++은 이산사건 시뮬레이션을 위해서 개발된 언어이다 [Str1991].

3-2 이산사건 형식론 (DEVS Formalism)

B. P. Zeigler에 의해 1984년에 제안된 DEVS 형식론은 집합이론에 기반을 두고 있으며, 이산사건 시스템을 계층적으로 표현하기 위하여 atomic 모델 및 coupled 모델의 두가지 모델 class를 사용한다 [Zei84][Zei90].

Atomic 모델은 더 이상 분해할 수 없는 시스템 component를 표현하는 모델로서 다음과 같이 구성된다.

$M=\langle X,\ S,\ Y,\ \delta_{int},\ \delta_{ext},\ \lambda,\ t_a \rangle$

여기서

X : 입력사건 집합(input events set), finite set

Y : 출력사건 집합(output events set), finite set

S : 상태변수 집합(sequential states set), finite set

δ_{int} : 내부변이 함수(external transition function)

δ_{ext} : 외부변이 함수(external transition function)

λ : 출력함수(output function),

t_a : 시간전진 함수(time advanced function)

위의 항들에는 다음과 같은 제약조건들이 있다.

(1) Atomic 모델의 전체 상태변수 집합은 다음과 같이 정의된다.

$Q=\{(s,\ e) \mid s \in S,\ 0 \le e \le t_a(s)\}$

(2) 외부변이 함수 $\delta_{ext}: Q \times X \rightarrow S$

(3) 내부변이 함수 $\delta_{int}: S \rightarrow S$

(4) 시간전진 함수 $t_a: S \rightarrow R$

(5) 출력함수(output function) $\lambda: S \rightarrow R$

입력사건 집합은 시스템의 모든 가능한 입력 사건들의 집합을 나타내며, 출력사건 집합은 시스템의 모든 출력 사건의 집합을 나타낸다. 상태변수 집합은 시스템이 가질 수 있는 모든 가능한 상태를 포함한다. 외부 변이함수에서는 외부에서 어떤 입력사건을 받았을 때 현 상태에서 다른 상태로의 변이를 명

세하며, 내부 변이함수는 내부 입력 사건이 없이 시스템이 시간전진 함수에서 정의한 시간이 지났을 때 내부적으로 상태 변수들이 변하는 모양을 표현하는 함수이다. 시간전진 함수란 외부 입력사건 또는 내부 변이가 일어났을 때 현 상태에서 얼마 동안 머문 후 내부 변이가 일어나는지를 지정하는 함수이다. 그러나, 시간 전진함수에서 지정한 시간이 아직 지나지 않은 상태에서 어떤 외부 사건이 발생하였을 때에는 그 시점을 기준으로 다른 새로운 시간을 지정해야만 한다. 출력함수에서는 어떤 상태에서 출력을 발생시키는지를 명세하는 함수이다.

Coupled 모델은 여러 개의 atomic 모델들 또는 coupled 모델들로서 구성된 복합적인 모델을 연결 상태 및 각 구성모델에 대한 정의로 표현하며, 다음과 같이 구성된다.

$DN = \{X,\ Y,\ M,\ EIC,\ EOC,\ IC,\ SELECT\}$

여기에서

X : 입력사건 집합(input events set), finite set

Y : 출력사건 집합(output events set), finite set

M : 모든 component 모델의 집합(set of all component models), finite set

EIC: 외부입력 연결 명세(external input coupling relation),

$EIC \subseteq DN.IN \times M.IN$

EOC: 외부출력 연결 명세(external output coupling relation)

$EOC \subseteq M.OUT \times DN.OUT$

IC : 내부 연결 명세(internal coupling relation),

$IC \subseteq M.OUT \times M.IN$

$SELECT$: 수행 우선순위 선택자(tie-breaking selector)

상기의 coupled 모델 명세에서 알 수 있듯이 한 coupled 모델은 다른 coupled 모델의 component가 될 수 있으므로 DEVS 형식론은 이산사건 시스템을 계층적으로 표현할 수 있다.

이상의 DEVS 형식론은 시뮬레이션 언어처럼 단지 모델을 명세하는데 사용되는 것 뿐만 아니라 연속 시스템에서의 미분 방정식과 같이 이산사건 시스템을 수학적으로 다룰 수 있는 기법들이 제공된다.

3-3 DEVSim++ Simulation Engine

DEVSim++은 C++ 언어로 작성된 Object-Oriented based DEVS용 시뮬레이션 소프트웨어이다 [Kim94]. DEVSim++은 시뮬레이션을 위한 다양한 기능들을 제공하며, 사용자는 다음과 같은 procedures들을 C++ 언어 및 DEVSim++ Library들을 이용하여 DEVS 형식론 및 시뮬레이션 과정들을 정의한다.

* external transition functions
* internal transition functions
* output functions
* time advanced functions
* coupled functions

DEVSim++과 유사한 상업용 및 학술용 시뮬레이션 언어로는 SIMSCRIPT, GPSS, SLAM 및 SIMAN 등이 있으나 현재 Obejct-Oriented DEVS용 언어는 DEVSim++이 유일한 소프트웨어이며, DEVSim++ 소프트웨어의 사용이 전 세계에 공개되어 있다(http://sim.kaist.ac.kr).

객체 지향 모델링 시뮬레이션 환경인 DEVSim++은 DEVS 형식론을 C++ 언어로 구현한 것으로 이는 DEVS 형식론의 명료한 의미론과 C++ 언어의 표현력과 속도를 장점으로 가지고 있다. 따라서 DEVSim++은 이산사건 시스템을 C++언어를 써서

DEVS 형식론 안에서 명세할 수 있다. DEVSim++ 환경에서 C++ 의 유연성을 가지고 모델링 할 수 있을 뿐만 아니라 기 개발된 모델은 새로운 모델을 개발하는데 재 사용될 수 있다. DEVS 형식론 안에서 개발 된 모델의 시뮬레이션은 추상화된 시뮬레이션 알고리즘을 가지고 있는데 이 추상화된 시뮬레이터는 역시 atomic 모델 부분과 coupled 모델를 위한 부분으로 구성되어 있다.

Ⅳ. DEVS를 이용한 동적 조우 모델링

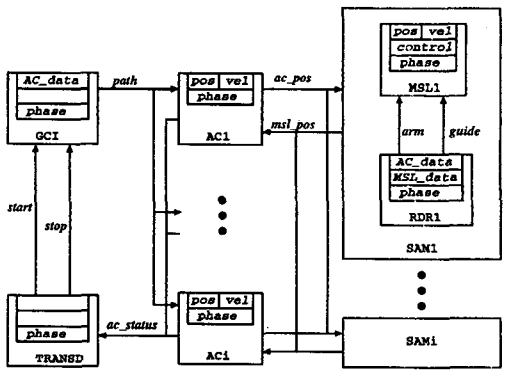
4-1 구성 요소들

본 연구의 대상 시나리오는 다수의 항공기가 밀집된 지대공 유도탄체계(SAM: Surface to Air Missile)로 구성된 방공망을 통과하는 것이다. 따라서, [그림 2]와 같이 배치된 방공망을 미리 계획된 경로로 항공기가 비행할 경우 어떤 현상이 언제 발생되는가를 Simulation한다. 시스템은 GCI, 항공기, SAM 및 TRANSD로 구성하였고, 연결구조는 [그림 5]와 같으며, 각각의 구성 요소별 개략적인 동작 기능은 다음과 같다.

* TRANSD: Simulation을 시작과 끝을 결정하고 항공기 상태를 모니터 한다. GCI의 동작을 시작시키고, Game이 완료되면 중지시킨다.
* GCI: 항공기의 비행을 관제하는 지상관제소로서 항공기의 비행궤도를 조종한다.
* Aircrafts: 각각의 출발지점에서 목표지점을 향하여 비행한다. GCI에서 보낸 비행 진행 message를 받아서 비행하며 또한 비행 중에 현 위치 정보를 각 레이더로 보낸다.
* SAMs: 레이더체계(Radar)와 유도탄(Missile)으로 구성되며 항공기를 탐색하고, 추적하며, 유도탄 사정거리 이내로 항공기가 진입시 유도탄을 항공기 위치로 비행시키고 근접 위치에서 폭발시킨다. 항공기가 격추되면 다시 탐색모드로 원 위치하나, 격추 실패 시에는 또 다시 유도탄을 발사시킨다.
* Radar: SAM의 한 구성요소이며, 탐색모드에서는 레이더 Beam을 회전시키고, 항공기가 추적거리 이내로 접근 시에는 추적모드로 전환한다. 추적모드에서는 레이더 beam을 항공기 위치로 고정시키며, 항공기를 계속 추적한다. 2대 이상의 항공기가 동시 진입시에는 가장 가까운 항공기를 추적한다. 항공기가 요격거리 이내로 접근하면 유도탄을 발사 시키고 유도탄을 항공기 위치로 유도시킨다.

  항공기와 유도탄의 거리가 파괴 유효반경 이내이면 유도탄을 폭발시킨다.
* Missile: SAM의 한 구성요소이며, 레이더의 유도 message를 받아서 항공기 위치로 비행하고 폭발시에 arm message를 각 항공기에 보내고 각 항공기는 이 message를 받아서 피격여부를 결정하게 된다.

System architecture 구현의 문제점은 항공기와 SAM 체계와의 M:N mapping 및 message link 관계이나 이는 DEVSim++의 coupling scheme이 보장하므로 쉽게 구현될 수 있었다. 구현상에서의 중요한



[그림 5] 모델들의 연결구조

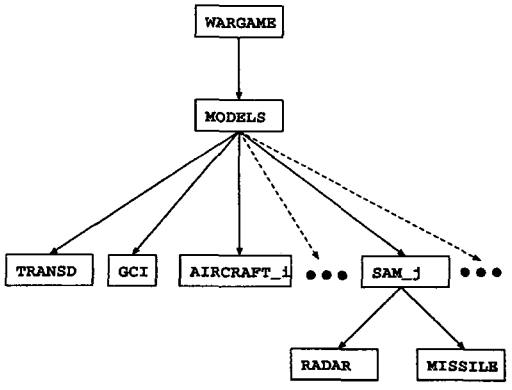
고려사항은 message의 정확한 전달을 확인하는 것이며 이는 trace 과정으로 확인하였다.

또한 Game의 진행사항을 쉽게 파악하고 해석해 볼 수 있기 위하여 Animation을 가미한 그래픽을 사용하여야 한다.

DEVSim++에서 제공하는 모든 기능을 충분히 활용하고, 각 Object에 대하여 정확하고 논리적인 Formalization을 규정하고, 그외의 mapping 및 link 관계는 DEVSim++의 Coupling Scheme을 따름으로써 복잡한 관계를 비교적 쉽게 구현할 수 있었다. 또한 그래픽은 Motif-1.2을 이용하였으며 모델링 프로그램과 link하여 Simulation 의 진행상황을 한 step 씩 바로 그래픽으로 표현하였다. 각 모델의 상황변수들을 별도 화일에 저장시켜 출력해석에 사용되도록 하였고, 입력변수도 별도 화일을 사용하여 사용자가 쉽게 변동할 수 있도록 하였다.

4-2 시스템 모델링

레이더 체계는 탐색, 추적, 유도탄 발사 및 유도 등의 상태집합과 레이더 위치(x,y,z), 탐지거리, 추적거리, 요격거리, 요격확률 및 ECM 취약성 등의 변수들 및 요격절차 등을 고려하여 모델링되며, 항공기 체계는 비행, 피격, 목표도달 등의 상태집합과 항공기 좌표, 속도, 비행경로 및 ECM 능력 등의 변수들로 모델링 된다.



[그림 6] 계층적 시스템 구성

시뮬레이션 결과(Performance Index)는 항공기의 표적 도달 가능성 및 도달 과정상의 항공기들 및 레이더 체계들의 상태변이 현상들이며 이들은 X-Window 상에 Animation을 가만한 Dynamic한 그래픽으로 표현하였다.

시스템의 구성 객체들은 [그림 6]과 같이 5개의 objects로 계층적인 구성이 되며, 항공기와 SAM은 사용자가 지정한 대수만큼 형성된다. SAM은 각각 레이더와 유도탄으로 구성된 Coupled Model로 구성된다.

각 Objects들은 DEVS 형식론으로 모델화 하였고, 한 예로서 Radar에 관한 DEVS 형식화와 각각의 세부 설명은 다음과 같다.

(1) Formal specifications for RADAR

* $FSM\_\{RADAR\} = \{X, S, Y, \delta_{int}, \delta_{ext}, \lambda\}$:
  레이더는 유한 개의 상태 수를 갖는 DEVS 형식론으로 명세한다.
* $X = \{ ac\_pos \ (from \ AC) \}$:
  입력 메시지는 항공기로부터 받는 항공기 위치 정보이다.
* $S = \{ READY, SEARCH, TRACK, GUIDE, ARM \}$:
  상태의 종류는 준비, 탐색, 추적, 미사일 유도 및 폭파이다.
* $Y = \{ rdr\_sig \ (to \ AC), msl\_flyto \ (to \ MSL), msl\_arm \ (to \ MSL) \}$:
  출력사건집합은 항공기로 보내는 레이더 신호, 미사일로 보내는 유도명령, 미사일로 보내는 폭발명령으로 구성된다.
* δ_{ext} $(READY, ?ac\_pos) = SEARCH$:
  외부변이 함수는 준비상태에서 항공기 위치 메시지가 도착하면 탐색 상태로 전환한다.
* δ_{ext} $(SEARCH, ?ac\_pos) = SEARCH$:

탐색상태에서 항공기 위치 메시지가 도착하면 탐색 상태로 계속 유지한다.

* δ_{ext} *(TRACK, ?ac_pos) = TRACK* :
추적상태에서 항공기 위치 메시지가 도착하면 추적 상태로 계속 유지한다.

* δ_{ext} *(GUIDE, ?ac_pos) = GUIDE* :
미사일 유도 상태에서 항공기 위치 메시지가 도착하면 미사일 유도 상태로 계속 유지한다.

* δ_{int} *(SEARCH, ac_in_track_range) = TRACK*;
내부변이 함수는 탐색모드에서 항공기의 위치가 추적가능거리 이내이면 상태를 추적 상태로 전환한다.

* δ_{int} *(TRACK, ac_out_track_range) = SEARCH* ;
추적모드에서 항공기의 위치가 추적가능거리 밖이면 상태를 탐색 상태로 전환한다.

* δ_{int} *(TRACK, ac_out_missile_range) = GUIDE*;
추적모드에서 항공기의 위치가 미사일 사거리 이내이면 상태를 미사일 유도 상태로 전환한다.

* δ_{int} *(GUIDE,|ac_pos - msl_pos| <= cep) = ARM*;
미사일 유도모드에서 항공기의 위치와 미사일 위치의 차이가 유효 파괴 반경 이내이면 상태를 미사일 폭파 상태로 전환한다.

* δ_{int} *(GUIDE, msl_pos>msl_max_range)=ARM (self-arm)*;
미사일 유도모드에서 미사일의 위치가 최대 사거리 밖이면 상태를 폭파(자폭) 상태로 전환한다.

* δ_{int} *(GUIDE, ac_pos>track_range)=ARM(self-arm)* ;
미사일 유도모드에서 항공기의 위치가 추적거리 밖이면 상태를 폭파(자폭) 상태로 전환한다.

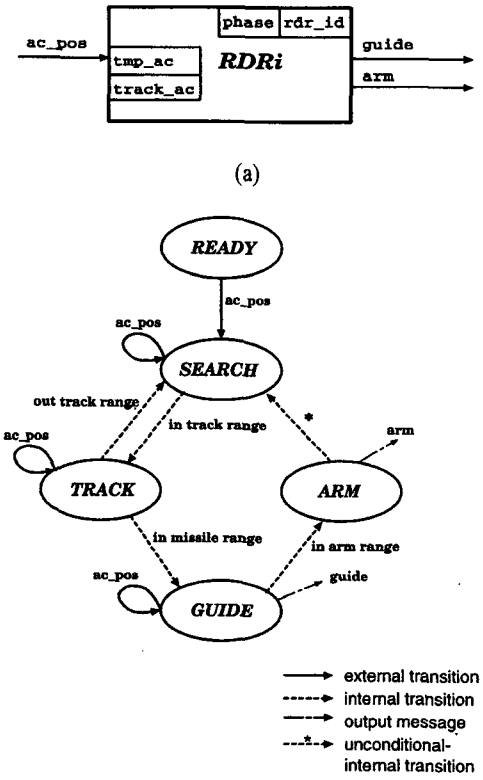
* δ_{int} *(ARM, -)= SEARCH* ;
미사일 폭파 모드 후에는 새로운 다음 표적 탐색을 위하여 무조건 탐색모드로 상태를 전환한다.

* λ *(SEARCH or TRACK or GUIDE or ARM, -)= !rdr_sig* ;
출력함수는 탐색, 추적 및 폭파모드에서 레이더 신호를 출력시킨다.

* λ *(GUIDE, -) = !msl_flyto* ;
미사일 유도 모드에서는 미사일 유도 메시지를 출력시킨다.

* λ *(ARM, -) = !msl_arm* ;
미사일 폭파모드에서는 미사일 폭파 메시지를 출력시킨다.

이상과 같이 레이더의 상태 및 동적 특성에 관련



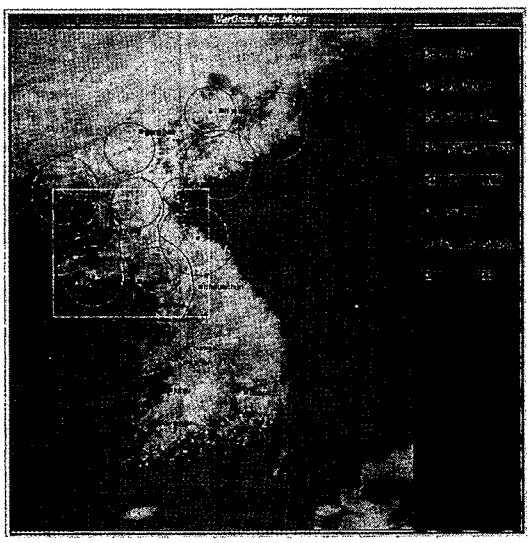
[그림 7] 레이더의 모델링

한 이산 사건들을 명료하게 명세할 수 있으며, 각 사건들의 입출력 관계와 상태의 흐름도(State diagram)는 [그림 7]과 같다.

4-3 동적 조우 시뮬레이션의 구현

본 연구에서의 중요 관심사항의 하나는 simulation 과정을 그래픽으로 동시에 표현시키는 것이며 이 문제는 DEVSim++ V1.0에 포함된 sim.step( )과정을 이용하였다. 그래픽 과정은 X 및 Motif Library를 사용하였으며 [그림 8]은 구현된 프로그램의 주 화면이다. 그래픽 화면의 주요 기능은 다음과 같이 설계 구현하였다.

* Wargame DB : 표적 위치, SAM site 위치 및 성능, 주요 도시 등의 Wargame area에 관련된 데이터를 관리한다.
* Display City : 지도 위에 주요 도시 및 표적들을 도시한다.
* Display Threats : 지도 위에 SAM sites 및 레이더/미사일의 coverage들을 도시한다.



[그림 8] 메인 화면

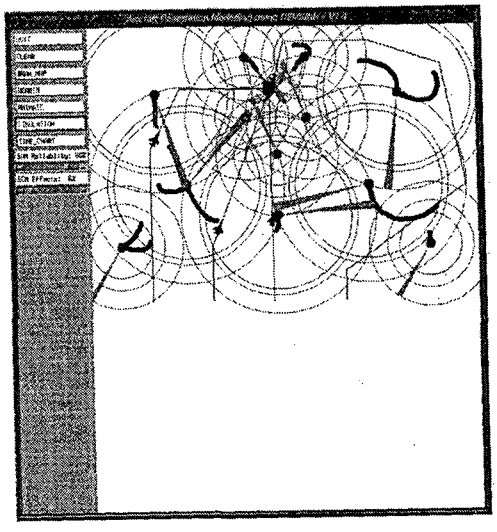
* Edit Aircraft Paths : 지도 위에 항공기의 비행 경로를 편집한다.
* Select Game Area: Wargame area를 선정한다.
* Map Analysis: Game Area의 지형에 대하여 등고선 분석, 3 차원 표시 등의 지형 분석 과정을 수행한다.
* Wargame Simulation: Simulation을 구동시키고 그 결과에 따른 Game 환경을 그래픽으로 Animate 하며, 각 항공기 위치별 위험도를 그래프로 도시한다.

각 객체 특성에 대한 입력 데이터는 다음과 같으며 사용자가 용이하게 변경 및 추가가 가능하게 그 구조를 설계하였다. 각 Object별 입력변수는 다음과 같다.

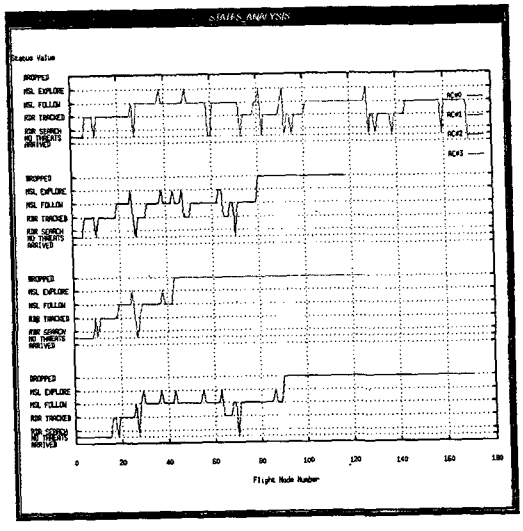
(1) SAM
- Total sites number
- Site location(x, y, x)
- Radar beam 특성(beam width, beam step, initial beam center)
- Acquisition Range (min, max)
- Track Range (min, max)
- Track time
- Missile 변수(velocity, range min, range max, lethal range)
- SAM system reliability

(2) AIRCRAFT
- Total number of aircrafts
- number of node for each flight path of aircraft
- path node data(x,y,z,velocity)
- ECM effectiveness

(3) TARGET
- Target location(x,y,z)



[그림 9] Sample Scenario run



[그림 10] 시간별 상태천이(Time Chart)

4-4 실험 및 결과 분석

입력변수 화일을 sample scenario에 따라 작성하여 시뮬레이션을 수행하였다. Simulation 과정의 각 step에 대하여 모든 Object의 status를 출력화일에 저장하였다. [그림 9]는 시뮬레이션이 진행 중인 한 화면이다. Performance Index는 항공기의 목표지점까지의 생존률이며, 측정방법은 각 항공기의 상태변수를 기록하는 방안을 사용하였고, 이는 [그림 10]과 같이 Time chart로 도시되어 어느 지점에서 어떤 event가 발생되었는지를 표시하여 준다.

모두 4대의 항공기를 비행시키고 상대방으로 총 12대의 SAM을 배치시켰다. 시뮬레이션 수행 결과 SAM을 관통하여 비행한 1번 항공기는 추락하였고, 우회 비행한 나머지 항공기는 모두 목표지점에 도착하였다. 물론 입력변수의 SAM 성능 데이터에 따라 결과가 달라지며 이는 예상된 현상이었다. 레이더는 예상대로 항공기를 포착 추적하였고 유도탄을 발사시켰다.

V. 결 론

DEVS의 형식론과 DEVSim++의 지원 기능을 이용하여 기대 이상의 결과를 보일 수 있었다. 특히 Wargame 진행 상황을 그래픽으로 동시에 진행 및 시현함으로써 simulation 동작 상태를 직접 파악할 수 있었다. 복잡한 대공방어망을 통과할 때 언제 어느 레이더가 어떤 항공기를 대상으로 동작하며, 유도탄은 또한 언제 어떤 방향으로 어느 항공기를 향하여 비행하고 있는지를 그래픽 화면으로 직접 표현할 수 있었다.

보완될 내용은 통계적 출력 해석을 위한 Output Data 작성에 관한 연구, 지형정보를 고려한 3 차원 display 및 animation 기능이 추가되어야 한다. 또한 현재 미리 지정한 비행궤도 대신에 Simulation 과정 도중에 발생한 각종 events를 종합 이용한 autonomous flight path routing에 관한 연구가 필요 연구과제로 대두되었다.

본 연구결과는 적의 밀집된 방공망의 침투 가능성을 판정하고, 침투시 발생되는 사건을 용이하게

예측할 수 있으며, 입력 상황을 반전시키면 아군의 방공망 능력을 시뮬레이션 할 수 있게 되므로 침투 궤도 선정 및 분석, 방공망 구성 설계 및 평가, 레이더 체계 성능 요구사항 도출 등에 응용될 수 있겠다.

참고문헌

1. [Kim92] 김탁곤, "이산사건 시스템 모델링 시뮬레이션 기법," 전자공학회지, vol.~19, pp. 105-114, Jan., 1992.
2. [AG92] C. Alexopoulos and P.M. Griffin, "Path planning for a mobile robot," *IEEE trans. on SMC*, vol.22, no.2, pp. 318-322, Mar., 1992.
3. [FS] K.Fujimura and H.Samet, "A hierarchical strategy for path planning among moving obstacles," *IEEE Trans. on Robotics and Automation*, vol.~5, pp. 61-69, Feb., 1989.
4. [KB91] N.Kiryati and A.Bruckstein, "On navigating between friends and foes," *IEEE Trans. on PAMI*, vol.~13, no.~6, pp. 602-606, June, 1991.
5. [Kim94] T.Kim, *DEVSIM++ User's Manual. CORELAB.*, KAIST, 1994.
6. [MT] D.K. Mesecher and M.Teris, "Penetrator- a dynamic high fidelity ew encounter simulation," *Proceedings of the 1994 Summer Computer Simulation Conference*, pp. 349-354, June, 1994.
7. [Neb94] R.R. Nebiker, "Interactive wargaming (people-in-the-loop)," *Proceedings of the 1994 Summer Computer Simulation Conference*, pp. 538-543, June, 1994.
8. [Zei84] B.P. Zeigler, *Multifacetted Modeling and Discrete Event Simulation*, Academic Press, 1984.
9. [Zei90] B.P. Zeigler, *Object-oriented Simulation with Hierarchical, Modular Models: Intelligent Agents and Endomorphic systems*, Academic Press, 1990.
10. [Pet81] J. J. Peterson, *Petri Nets Theory and the Modeling of Systems*, Prentice-Hall, 1981.
11. [Str91] Bjarne Stroustrup, *The C++ Programming Language, 2nd Ed.*, Addison-Wesley, 1991.

≡ 필자소개 ≡

윤 기 천

국방과학연구소