한국시뮬레이션학회 논문지
Vol. 17, No. 2, pp. 91-103 (2008. 6)

자재 취급 시스템을 위한 다중 에이전트 기반의 교착상태에 자유로운 AGV 시뮬레이터 개발

이재용[1] · 서윤호[1†]

Development of Multi-agent Based Deadlock-Free AGV Simulator for Material Handling System

Jae-Yong Lee · Yoon-Ho Seo

ABSTRACT

In order to simulate the behavior of automated manufacturing systems, the performance of material handling systems should be measured dynamically. Multi-Agent technology could be well adapted for the development of simulator for distributed and intelligent manufacture systems. A multi-agent system is composed of one coordination agent and multiple application agents. Issues in AGVS simulator can be classified by the set-up and operating problems. Decisions on the number of vehicles, bi- or uni-directional guide-path, etc. are fallen into the set-up problem category, while deadlock free algorithm and conflict resolution are in operating problem. In this paper, a multi-agent based deadlock-free simulator for automated guided vehicle system (AGVS) are proposed through the use of multi-agent technologies and the development of deadlock-free algorithm. In this AGVS simulator proposed, well-known Floyd algorithm is used to create AGVS Guide path, through which AGVS move. Also, AGVs avoid vehicle conflict and deadlock using check path algorithm. And Moving vehicle agents are operated in real-time control by coordination agent. AGV position is dynamically calculated based on the concept of rolling time horizon. Simulator receives and presents operating information of vehicle in AGVS Gannt chart. The performance of the proposed algorithm and developed simulator based on multi-agent are validated through set of experiments.

Key words : Automated guided vehicles, Multi-agent, Simulator, Deadlock-free, Check path algorithm, Real-time control

요 약

자동화 제조 시스템에 시뮬레이션 기법을 사용하기 위해서는 자재취급 시스템의 성과를 동적으로 측정해야만 한다. 다중 에이전트 기술은 제조시스템을 지능화 시키고, 분산처리 된 시뮬레이터 개발에 적합하다. 다중 에이전트 시스템은 하나의 조정 에이전트와 다수의 응용 에이전트들로 구성된다. AGVS 시뮬레이터에 있어 이슈는 운반차량의 대수 결정, 양단방향 흐름에서의 이동경로 결정등과 같은 set-up문제와 운영문제로 구분 지을 수 있다. 본 논문에서는 다중 에이전트 기술을 사용하고, 실시간 교착상태 해법 알고리즘이 포함된 시뮬레이터를 소개한다. 시뮬레이터는 잘 알려진 프로이드(Floyd) 알고리즘을 사용하여 AGVS의 최단 이동경로를 구성된다. 움직이고 있는 차량 에이전트는 조정 에이전트에 의해 실시간 제어로 작동되고, AGV는 경로확인 알고리즘을 사용하여 충돌과 교착상태를 피한다. AGV의 위치는 수평시간 계획법에 근거하여 동적으로 재계산된다. 충돌해소 알고리즘은 어떠한 배치 형태에서라도, 그리고 큰 규모의 문제에서도 AGVS의 운영 중의 주행문제 해결을 보장한다. 시뮬레이터는 AGV들의 AGVS 칸트차트를 통하여 작동정보를 받고, 표현한다. 제안된 알고리즘의 성과와 다중 에이전트 기술을 사용하여 개발한 시뮬레이터는 실험을 통하여 검증된다.

주요어 : AGV, 다중 에이전트, 시뮬레이터, 교착상태에 자유로움, 경로확인 알고리즘, 실시간 제어

* 본 연구는 한국과학재단의 출연금으로 수행한 특정기초연구 지원사업의 연구결과로 수행되었음.(R01-2006-000-10941-0)
2008년 5월 26일 접수, 2008년 6월 11일 채택
[1] 고려대학교 산업시스템정보공학과
주 저 자 : 이재용
교신저자 : 서윤호
E-mail; yoonhoseo@korea.ac.kr

1. 서 론

1.1 연구의 배경

Automated Guided Vehicle(AGV)은 자재 취급 장치로서 중요한 역할을 담당하고 있다. AGV는 특정 작업장에서 처리된 자재를 적재하여 적재한 자재를 필요로 하는 작업장으로 이동하여 하역시키는 임무를 수행한다. 이러한 AGV System (AGVS)에서 주행상의 문제가 발생된다면 시스템의 자재 수급에 문제가 발생되어 전체 공정의 운영에 차질이 생겨나고, 공정을 원상태로 회복시키는데 막대한 비용이 발생된다. 이러한 주행상의 문제를 예측하고, 해결하기 위해서 다양한 방법들이 연구되어지고 있으며, 그 중 패키지 형태의 시뮬레이터가 널리 사용된다[1].

기존의 패키지형태의 시뮬레이터는 프로그램 설치에 따르는 비용이 많이 들고, 시뮬레이션 시키고자 하는 배치형태의 구성을 위해 많은 시간이 소요된다. 또한, 구성한 배치형태에 적합한 주행규칙에 대한 프로그램을 다시 만들어야 하는 전문성과 시뮬레이션 모델링 구성을 위한 시간이 많이 소요된다. 그러므로 AGVS를 쉽게 모델링하고 이를 시뮬레이션 할 수 있는 전용 AGVS를 위한 시뮬레이터 개발이 필요하다. 제안하는 시뮬레이터는 상용 시뮬레이터의 장점을 살리기 위하여, 3차원 그래픽으로 유명한 OpenGL을 사용하여 주행의 현실성을 살리고, 패키지 형태에서와 같이 시스템 파라메타들의 조절, 줌 등과 같은 기능을 추가 했다. 패키지 형태의 시뮬레이터의 단점을 보완하여, 양방향의 주행흐름경로 환경에서 조감도 형태에 근거한 2차원의 배치에 관한 네트워크 정보와 시스템 파라메타가 주어진다면 비전문가들도 별도의 프로그래밍이 필요 없이 시뮬레이션 모델링 및 연구를 수행할 수 있다. 또한, 전용 시뮬레이터를 실현을 위해 배치형태, AGV의 대수 선택등과 같은 다양한 시스템 파라메타 환경이 주어지더라도 주행문제 없이 시뮬레이션 시킬 수 있는 주행문제 해결 알고리즘이 필요하다.

AGV의 주행문제들을 해결하기 위한 방법은 두 가지 종류로 구분할 수 있다[2]. 그 첫 번째 방법은 시스템을 운영시키기 전에 AGVS에서 충돌이 일어나지 않도록 미리 흐름경로를 구성, AGV의 경로할당, 주행 속력결정과 그에 따르는 노드의 도착과 떠나는 시간을 선 계획하는 것이다. Langevin et al.[3]는 AGV에게 충돌이 없는 이동경로를 계획하기 위해 정수 계획법을 사용하여 AGV의 이동경로를 최소화시키는 이동경로를 찾아내었다. Rajotia et al.[4]는 같은 시간대에 예약되지 않은 경로를 각 AGV에 할당하는 선 계획하는 알고리즘을 제안하였다. KO and EGBELU[5]는 branch and bound를 사용하여 AGV들의 Travel time을 최소화 하는 일 방향 흐름가능 AGV 이동경로를 선 계획하였고, Seo and Lee[6]는 Tabu 탐색을 사용하여 물류비를 최소화 하는 일 방향 흐름가능 AGV 이동경로를 설계하였다. 이들의 연구에서 제시하는 주행문제해결 방법은 예측 못한 돌발요인으로 인해 AGV가 계획대로 이행되지 않을 때에는 주행문제에 대해서 시스템이 유연하게 대응할 수 없는 문제점을 가진다.

두 번째 종류의 연구방법은 첫 번째 연구의 한계를 해결하기 위해서 AGV들이 주행 중에 들리게 되는 경로에서 AGV의 이동 중에 실시간으로 주행문제를 발견하고, 충돌을 피하는 우회경로 혹은 AGV의 속력을 재할당하여 실시간으로 주행문제를 해결하는 것이다. Thomas와 Wenger[7]는 AGV들의 주행 정보에 대해 간주하지 않은 상태에서 각각의 AGV들의 최단 경로를 탐색한 후에 각 AGV의 속력을 제어하여 실시간으로 움직이는 AGV의 행동을 계획하였다. Fanti[8]는 같은 시간대에 분할된 Zone내에 한 대의 vehicle이 있어야 한다는 생각으로 Zone 제어 기술을 사용한 실시간으로 경로 할당 알고리즘을 제안하였다. Samia and Castagna[2]는 AGV의 실시간으로 주행문제 발생을 예측하고서, 주행문제가 발생되는 AGV들에 우선노드 점유권을 줌으로써 충돌을 해소하는 RVRAA알고리즘을 제안하였다.

본 논문에서는 앞의 두 종류의 방법이 적용된 AGV의 주행문제 해결 알고리즘이 내장된 AGVS 시뮬레이터를 개발하는데 그 목표를 두었다. 어떠한 시스템 파라메타가 주어진 AGVS 주행 환경에서도 주행문제를 해결할 수 있는 알고리즘을 개발하기 위하여 다중 에이전트 기법을 사용하였다. 에이전트는 실시간으로 AGVS의 주행문제를 판단하여, 해결 알고리즘을 기반으로 에이전트 간의 통신을 통해 실시간으로 주행문제를 해결할 수 있다.

본 논문은 다음과 같이 구성되어 있다. 2장에서는 문제와 용어의 정의를 기술하였으며, 3장에서는 다중 에이전트 방법론을 사용하여 주행문제를 해결하는 방법을 기술하였다. 4장에서는 본 연구를 통해 개발된 다중 에이전트 기반의 양방향 AGV 시뮬레이터에 대한 설명을 했고, 5장에서는 개발한 AGVS전용 시뮬레이터를 사용하여 제시하는 주행문제 해결 알고리즘을 실험을 통해 검증을 하였으며, 6장에서는 본 연구의 결론 및 추후연구에 대해 기술하였다.

2. 문제기술과 용어의 정의

2.1 문제기술

본 논문에서는 양방향 주행가능 경로를 가지는 AGVS에서 기간 내의 작업 목록을 수행할 수 있는 시스템 파라메타에 유연한 시뮬레이터를 개발하는데 그 목적이 있다. 시뮬레이터에서의 모든 AGV들은 적재지점에서 하역지점을 주행함으로써 작업 목록을 수행한다. 본 연구에서 다루는 AGVS는 그림 1과 같은 전체 작업장에 각각의 작업장(이하 S_w)S_1～S_4의 위치가 할당되어져 있으며 이들 간에 양방향 주행 가능한 통로가 연결되어 있고, 다양한 속도를 가지는 다수의 AGV가 표 1과 같이 주어진 작업의 순서에 따라 작업장을 이동하며 단위화물 M을 옮기는 작업을 수행하는 시스템으로 정의된다.

AGV들이 작업수행을 위한 주행을 할 때에 아크 혹은 아크들의 교차지점인 노드에서 진행경로가 반대방향인 AGV간의 정면충돌, 진행방향이 같은 AGV들 중 빠른 속력의 AGV와 느린 속력의 AGV간의 후면충돌, 특정 시점, 공간에 다수의 AGV들의 진입으로 인한 교착상태와 같은 문제가 발생할 수 있다. 이러한 문제들을 다중 에이전트를 이용하여 실시간으로 주행문제를 발견하고, AGV에게 경로와 속력을 재 할당 시키는 본 논문에서 제시하는 방법을 통해 주행문제를 해결하고, 이를 전용 시뮬레이터를 사용하여 AGVS를 검증한다.

2.2 용어정의

2.2.1 AGVS 칸트차트

Kim and Tanchoco[9]는 AGVS내의 각 AGV의 계획된 이동경로를 Time window 그래프를 이용해 AGV들 간의 충돌방지 알고리즘의 검증을 시각화했다. AGVS 운영 시 본 논문을 위해 개발된 AGVS전용 시뮬레이터에서는 사용자가 실시간으로 AGVS운영 정보를 확인할 수 있는 상태창이 실행된다. 이를 AGVS 칸트차트라고 하고, 본 논문에서는 이를 사용하여 알고리즘을 검증한다. AGVS 칸트차트에는 시간 척도에 따른 행위계획, AGV의 현 주행상황을 bar형태 위에 나타낼 수 있고, 칸트차트로 표현되는 AGV의 행위계획과 상태는 다음과 같다.

- 머무름은 AGV가 화물을 적재, 하역과 같은 작업 수행을 위해 혹은 AGV가 작업을 완료하고, 다음 작업을 Dispatching받기 위해서 작업장에서 머무르는 행위이다. 칸트차트에서의 상태는 bar위에 Unload, Load로 표기되고, 시간이 지남에 따라 bar가 좌측으로 이동한다.
- 대기는 AGV가 주행 중 충돌해결을 위해 주행경로에서 잠시 멈추는 행위이다. 칸트차트에서는 bar가 시간의 흐름에도 좌측으로 이동하지 않는 상태로 있다.
- 주행은 AGV가 작업 목록을 수행을 위하여 적재지점에서 하역지점을 오고가는 행위를 말한다. 칸트차트에서 계획된 행위계획이 bar에 그대로인 상태로, 시간이 지나면서 bar들이 좌측으로 이동한다.

시간축의 원점은 AGVS내에서 각 AGV의 현재 주행시점을 나타낸다고 할 때, 칸트차트에서 앞의 표현들을 통해서 AGVS의 모든 지점에서 발생하는 주행 문제와 AGV의 움직임을 그림 2와 같이 표현 할 수 있다. 또한 칸트차트에서 모든 AGV들의 계획된 경로와 각 노드의 연결 관계, 각 AGV의 도착시간을 확인 할 수 있다. 즉,

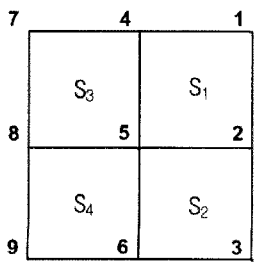

그림 1. 양방향 AGV주행가능 흐름경로 네트워크

표 1. AGV 작업 리스트의 예

Job	Load station	Unload station	Unit_L
1	1	9	M
2	2	8	M
3	3	7	M
4	4	6	M
5	5	1	M
6	6	4	M
…	…	…	M

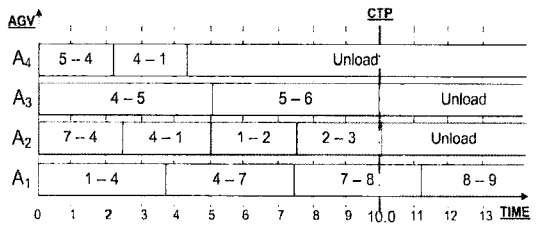

그림 2. AGVS 칸트차트

AGVS의 모든 계획된 경로에 대하여 다음을 가지고, 이는 주행문제 해결을 위한 입력 데이터가 된다.

- 이전노드, 진입시간
- 다음노드, 진입시간
- 이전노드 → 다음노드의 형태의 경로, 아크에서 머무르는 시간

2.2.2 Checked Time Path

본 연구에서는 time-window상에 나타난 AGV들의 주행문제를 실시간으로 예측하기 위해 검토되는 도착시간이 할당된 선 계획 경로를 확인 된 시간경로(Checked Time Path, 이하 CTP)라 정의하였다. 이를 위하여 충돌을 판단할 수 있는 임의의 시간범위의 주행문제 예측시간구간이 사용된다. 본 논문에서의 구간을 그림 2에서 0～10초 이내로 정하였고, 그 시간 구간 안에 포함된 각 AGV들의 이동경로가 각 AGV들의 CTP가 된다. 본 논문에서는 CTP를 사용하여 예측이 가능한 시간 내에서의 문제발생 혹은 정상작동을 실시간으로 예측할 수 있게 개발되었다. CTP를 사용한 AGV의 주행문제에 대한 해결 방법은 3절에서 자세하게 기술하였다.

3. 다중 에이전트를 이용한 주행문제해결

3.1 AGVS에 다중 에이전트 기술을 적용

다중 에이전트 시스템의 최소단위 구성은 시스템의 조정자역할을 하는 하나의 조정 에이전트와 시스템의 분산된 특정임무를 수행하는 둘 이상의 응용 에이전트들로 이루어진다[10]. 본 논문의 시뮬레이터에서는 AGVS의 운영을 관리하기 위하여 Tapio et al.[11]가 제시한 manAge라는 AGV 에이전트 아키텍처를 적용하였다. 본 논문을 위해 개발한 시뮬레이터에 적용되는 각 에이전트들이 수행해야 하는 일들은 다음과 같이 정리된다.

① 시스템의 작업 목록을 수행해 내야한다.
② 계산을 통하여 충돌을 예측한다.
③ AGV의 속력, 위치를 제어해야 한다.
④ 이동경로를 통해 AGV의 진행을 제어해야 한다.
⑤ 입력 데이터처리를 통해 의사결정을 내린다.

3.2 다중 에이전트 아키텍처

본 연구에서 적용된 다중 에이전트 시스템은 그림 3과 같이 하나의 시스템 조정 에이전트(Coordinate Agent; 이하 CA)와 그 외 다수의 응용 에이전트인 시스템의 문제발견 에이전트(Problem Detecting Agent; 이하 PDA), 문제해결 에이전트(Problem Solving Agent; 이하 PSA), 다수의 AGV 운영에이전트(Moving Agent; 이하 MA)들로 구성된다. 시뮬레이터를 구동시키기 위한 시스템 파라미터를 기반으로 각 에이전트들은 3.1절에서 소개한 에이전트들이 수행해야 하는 일들을 적절히 임무 분산시켜 에이전트간의 메시지교환과 AGVS 운영 데이터에서의 데이터 갱신을 통해서, AGVS는 원활한 임무 수행을 한다.

3.2.1 AGV의 운영 에이전트

MA의 각 개체를 n으로 나타내고, N은 AGVS의 모든 AGV의 대수를 나타낸다고 할 때, MA는 각각의 n에 할당된 AGV개체(AGV of n; 이하A_n)를, 운영데이터 갱신 모듈로부터 전달 받은 A_n의 주행, 적재, 경로에서의 대기, 하역동작과 같은 동작명령을 실행키는 응용 에이전트이다. 이 외에도 각 n에 할당된 A_n의 주행 중에 거치게 되는 노드들을 나열한 이동경로, 주행속도, 이동경로들을 진입하는 시간정보와 같이 미래에 실행 될 것으로 계획되어진 정보인 선 계획 데이터(preplanned data; 이하 p-data)와 A_n들의 동작, 고장, 작업완료로 인한 대기상태와 같은 현재 주행 상태정보, 현재 진입한 노드와 다음 노드, 두 노드의 도착시간을 포함하는 실시간 정보인 실시간 데이터(real-time data; 이하 rt-data)를 A_n과 주고받는다. 시스템 시간이 0초일 때 예제에서 만들어지는 rt-data와 p-data의 자료구조는 표 2와 같다. 또한 n은 A_n이 동작수행 후 주고받은 갱신된 rt-data(renewed real-time data; 이하 Rrt-data)와 p-data를 PDA와 CA로 정보를 전달하는 역할을 한다.

시스템 파라메타로 입력된 N의 개수와 같은 모든 n들은 다음의 알고리즘 구조로 동작한다.

Step 0. Set N = 0; n=0; 운영데이터 갱신모듈로부터 시스템 운영데이터를 받는다.

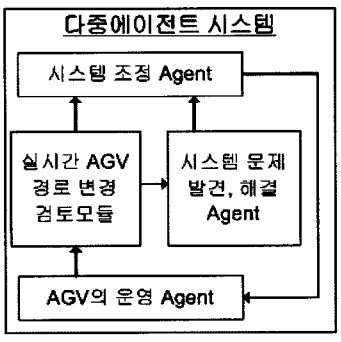

그림 3. 다중에이전트 아키텍처

표 2. p-data와 rt-data의 자료구조

AGV	p-data (이동경로, 노드도착시간, AGV의 속력)				
A_1	1	4	7	8	…
	0	3.75	7.5	11.25	…
	V_1	V_1	V_1	V_1	…
A_2	7	4	1	2	…
	0	2.5	5	7.5	…
	V_2	V_2	V_2	V_2	…
A_3	4	5	6	Unload	…
	0	5	10	-	…
	V_3	V_3	V_3	-	…
A_4	5	4	1	Unload	…
	0	2.14	4.28	-	…
	V_4	V_4	V_4	-	…

AGV	rt-data	AGV 현재 상태 [진입노드 , 다음 진입노드]						
A_1	운영	[1,4]	A_2	운영	[7,4]	A_3	운영	[4,5]	A_4	운영	[5,4]

Step 1. n = n + 1, n은 A_n에 운영데이터를 보내고, 동작 명령대로 A_n을 실행 시킨다.

1a. if n의 rt-data현재 상태 = 동작, Step 2로 간다.

1b. if n의 rt-data현재 상태 = 대기, Step 4로 간다.

1c. if n의 rt-data현재 상태 = 고장, AGVS에서 n과 A_n을 제외시킨다.

Step 2. A_n으로부터 n의 Rrt-data와 p-data를 업데이트 한다.

Step 3. 진입노드가 목적지 노드인지를 검토한다.

3a. if n의 진입노드 = n의 목적지, n의 현재 상태 = 대기 Step 4로 간다.

Step 4. 모든 n의 실행여부를 검토한다.

4a. if n < N, n = n+1, Step 1으로 간다.

4b. if n = N, PDA호출판단 모듈로 Rrt-data, p-data를 보내고, PDA호출여부를 판단한다.

3.2.2 시스템 문제발견 에이전트

PDA는 각 n에서 전달받은 CTP에 속하는 p-data와 Rrt-data를 수집, 처리하여서 AGV간의 주행 중 발생하는 문제(detected data; 이하 d-data)를 탐지하고, 이를 PSA로 전달하는 역할을 하는 응용 에이전트이다. 또한 시스템의 운영을 동기화시키는 시간단위인 △T마다 PDA호출판단 모듈이 실행되어서, 모든 A_n중에서 현재 진입노드가 변하는 순간에 호출된다. $P_I(A_n)$, $P_J(A_n)$는 A_n의 계획된 주행경로 노드 I, J를 나타내고, $T_I(A_n)$, $T_J(A_n)$는 A_n이

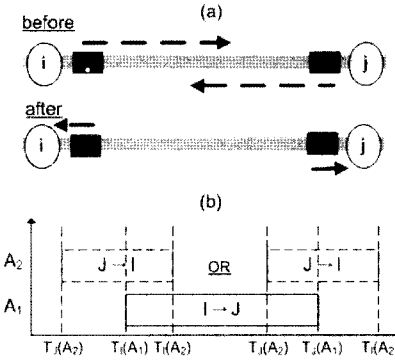

그림 4. 정면충돌의 상황

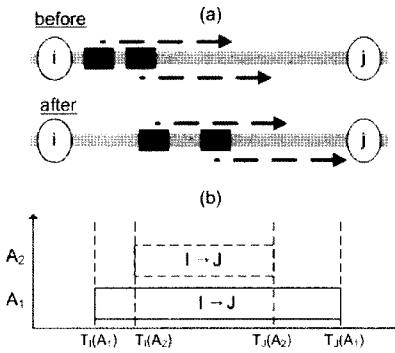

그림 5. 후면충돌의 상황

노드 I, J에 진입하는 시간이라 할 때에, 주행상태 혹은 작업완료로 인한 대기상태인 A_n들은 그림 4(a), 5(a)와 같이 노드 I, J들로 구성된 경로[이전노드, 다음노드]의 노드 진입시간을 비교하여 주행문제를 발견 할 수 있다. 다음은 A_n들의 CTP범위 안에 속한 p-data와 Rrt-data를 사용하여 PDA에서 발견해 낼 수 있는 주행문제들이다.

- 정면충돌은 A_1이 경로[$P_I(A_1)$, $P_J(A_1)$], A_2는 경로[$P_J(A_2)$, $P_I(A_2)$]가 그림 4(a)와 같이 계획되어있고, 각 경로의 노드진입시간이 (1)을 만족되면 예측된다.

$$T_J(A_2)<T_I(A_1)<T_I(A_2) \text{ or } T_J(A_2)<T_J(A_1)<T_I(A_2) \quad (1)$$

- 후면충돌은 그림 5(a)와 같은 주행상황에서 A_1은 경로[$P_I(A_1)$, $P_J(A_1)$], A_2는 경로[$P_I(A_2)$, $P_J(A_2)$]로 계획되어 있고, 각 경로의 구성노드의 진입시간이 (2)를 만족하면 충돌발생이 예측된다.

$$T_I(A_2)<T_I(A_1) \text{ and } T_J(A_1)<T_J(A_2) \quad (2)$$

또한 정면, 후면충돌은 그림 4(b), 5(b)와 같이 AGVS

칸트차트로 표현시킬 수 있다.

- 교착상태는 특정 공간에서 다수의 AGV들에 의해 발생되는 정면충돌, 후면충돌들이 복잡하게 얽힌 상태이다. 예를 들어 2.1절의 예에서 A_1, A_2는 아크(1,4)에서 (1)을 만족하므로 정면충돌이 예측된다. 마찬가지로 A_3, A_4는 아크(4,5)에서 (1)을 만족함으로 정면충돌이 예측된다. 각 A_n의 CTP는 노드4를 포함하여 만들어 지는 공간 S_1, S_2에서 4대의 A_n이 정면충돌이 복잡하게 얽혀있어 서로의 진행경로를 모두 막아버리는 교착상태가 예측된다.

3.2.3 시스템 문제해결 에이전트

각 A_n은 플로이드 알고리즘을 사용하여 처음 발생된 최단경로를 주행하지만, 충돌을 고려하지 않은 경로계획이었기 때문에 주행 중에는 충돌문제가 생겨난다. 이러한 주행 중의 충돌문제를 PDA는 실시간으로 주행문제를 예측하여 d-data를 발생 시켜 PSA를 호출하게 된다. PSA는 예측된 d-data를 해결할 수 있는 우회경로와 경로를 지날 때의 속력을 재할당하여 A_n의 이동경로, 진입노드의 도착시간을 재계산하여 갱신된 p-data(Renewed p-data; 이하 Rp-data)를 CA에게 보내는 역할을 하는 응용 에이전트이다. PSA에서는 d-data의 문제해결을 위해 우회경로와 속력을 재 계획하기 위해서 다음의 4단계를 거친다.

1단계. MA의 주행문제 정보의 추출

A_n들이 그림 2의 AGVS 칸트차트와 같이 주행하면, 4대의 A_n들은 S_1, S_2에서 교착상태가 발생된다. 교착상태는 다수의 정면충돌과 후면충돌들의 발생으로 나누어 질 수 있는데, 표 3은 교착상태의 주행문제를 다수의 충돌들로 구분한 d-data의 자료구조를 나타낸다. 표 3에서와 같이 충돌이 발생되는 두 A_n들에게는 충돌유형, 충돌경로, 충돌시간에 모두 영향을 받을 때에 충돌이 발생한다. 표 3의 d-data를 사용해서 표 4에서와 같이 수평시간 계획법을 적용하기 위해 충돌시간의 선후관계를 따져 먼저 발생되는 충돌을 A_n에 할당시킬 수 있다. 예를 들어 A_4는 아크(1,4)와 아크(4,5)에서 정면충돌의 발생이 예측되었지만, 그림 2의 칸트차트를 통해 경로[5,4]에서의 충돌이 경로[4,1]에서의 충돌보다 먼저 일어남을 확인할 수 있다. 따라서 A_4를 포함한 충돌이 발생되는 모든 A_n은 표 4와 같은 충돌 파라메타들을 할당시킬 수 있다.

표 3. 충돌에 의한 충돌파라메타

충돌되는 AGV들	충돌 유형	충돌 아크	충돌 시간
A_1 : A_2	정면충돌	(1,4)	0~5
A_1 : A_4	정면충돌	(1,4)	0~4.28
A_3 : A_4	정면충돌	(4,5)	0~5

표 4. 각 AGV들에 할당된 충돌파라메타

충돌AGV	충돌유형	충돌경로	충돌시간
A_1	정면충돌	[1,4]	0~5
A_2	정면충돌	[4,1]	0~5
A_3	정면충돌	[4,5]	0~5
A_4	정면충돌	[5,4]	0~5

2단계. 데이터의 그룹화

PDA에서 전달받은 d-data를 통하여 이전단계에서 A_1, A_2는 0~5의 충돌시간 사이에 양방향 흐름가능 아크(1,4)에서 정면충돌이 일어난다는 것을 알았다. A_1을 아크(1,4)에서 [$P_1(A_1)$, $P_4(A_1)$]로 진행 할 것을 계획하면, A_2의 경로[$P_4(A_2)$, $P_1(A_2)$]는 A_1의 진행에 방해가 되지 않게 하기 위해 그림 6(a)와 같이 우회경로 경로 4→5→2→1를 계획해야 한다. 아크(4,5)에서 정면충돌이 일어날 것으로 예상되는 A_3, A_4중, A_3에 경로[$P_4(A_3)$, $P_5(A_3)$]를 계획하면, 그림 6(b)에서와 같이 A_4는 우회경로 경로 5→8→7→4 혹은 경로 5→2→1→4를 계획해야만 한다. 충돌을 해결하기 위한 우회경로는 공통적으로 2차원의 S_w의 주위 모든 노드를 지나게 된다. 즉, 하나의 충돌을 해결하기 위해서, 하나의 우회경로를 선택 계획하는 것은 A_n에 곧 하나의 S_w을 할당하는 것과 같다. 이와 같이 A_n의 충돌해결을 위해서 우회경로 선택을 위한 S_w들의 모임(Set of S_w for nth AGV; 이하 SA_n)은 2가지의 후보 S_w를 만들어 낸다. 예제의 두 충돌에 대해서 그림 6(b)와 같이 SA_n={S_1, S_3}이 형성되고, 그림 6(c)와 같이 격자형태가 아닌 S_w들로 구성된 설비배치 형태에서도 아크(3,4)에서 발생되는 충돌의 해결을 위해 우회경로를 선택하기 위한 S_w인 S_1 혹은 S_2에 의해 SA_n={S_1, S_2}를 형성한다. 다음 단계의 우회경로 통합과정에서 우회경로가 커지는 것을 방지하기 위해서, 충돌의 우회경로 선택을 위한 S_w는 T_f순간에 발

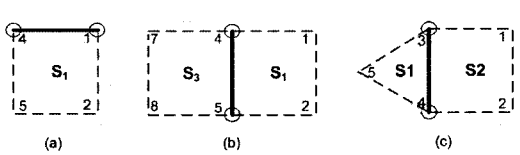

그림 6. 공간의 형성

생된 d-data의 충돌경로를 가장 많이 포함하는 SA_n의 요소들 중 하나를 충돌의 해결을 위한 S_w로 선택한다. 예를 들어서 예제에서, T_t=0인 순간에 발생될 것으로 예상된 충돌은 SA_1={S_1}, SA_2={S_1}, SA_3={S_1, S_3}, SA_4={S_1, S_3}를 형성하고, 충돌 경로를 포함하는 아크(1,2), 아크(4,5)를 가장 많이 포함 하는 S_1을 우회경로를 할당을 위한 S_w로 선택하겠다. S_1이 선택되면 A_1～A_4는 충돌 예상시간 0～5사이에 S_1주위 경로를 지나게 될 것을 계획한다. S_1주위 경로 아크에 A_1을 계획된 경로[P_1, P_4]로 이동시키면, A_2는 경로[P_4, P_1]는 우회경로 경로 4→5→2→1를 계획하고, A_3, A_4역시 A_1, A_2의 일 방향 흐름경로에 영향을 받아서 A_3에는 경로[P_4, P_5]를 A_4는 경로[P_5, P_4]의 우회경로 경로 5→2→9→4를 계획한다. 즉 충돌 예상시간인 0～5 사이에 S_1의 주위를 이동할 것으로 계획되어져 있는 A_1～A_4들은 같은 일 방향 흐름에서 이동경로를 재 계획한다. 각 충돌의 우회경로를 위해 결정된 S_w에서 충돌 예상시간에 A_n들은 일 방향 흐름으로 이동경로를 재 계획한다.

본 논문에서는 예측된 충돌의 충돌대상 A_n, 충돌발생 예상시간, 충돌회피를 위한 일 방향흐름의 경로 S_w들을 하나의 개념단위로 묶는 것을 '그룹화 한다.'라고 정의하겠다. 그룹화 된 일 방향 흐름경로 S_w를 지나는 다수의 A_n들 간에 속력의 차이로 인한 후면충돌이 일어나지 않게 하기 위해서 A_n들이 충돌 예상시간 범위 안에서 그룹의 일 방향 흐름경로 S_w를 지날 때에 모든 A_n들의 주행속력을 같게 계획한다. 그 결과 충돌발생 되는 A_n, 충돌예상시간, 충돌예상 공간, 동일속력들로 구성된 파라메타를 가지는 하나의 k번째 그룹G_k가 형성된다.

3단계. 그룹의 확대

이번 단계는 2단계에서 시스템 시간 T_t에 만들어진 G_k들을 충돌예상 시간과 충돌예상 S_w를 기준으로 다수의 G_k들을 서로 통합하는 과정을 수행한다. 그림 7에서 굵은 실선이 충돌아크를 나타낸다고 할 때, 확대 전 상황을 통합하는 과정은 표 5와 같다. 표 5의 Step 1에서 G_1, G_2에서 각각의 S_w들의 아크 중 아크(4,5)가 중첩되고, 충돌 예상시간 역시 6～6.42에 중첩이 된다. 이와 같은 상태일 때에 G_1, G_2은 하나의 확대 된 Step 2의 G_1을 형성하게 된다.

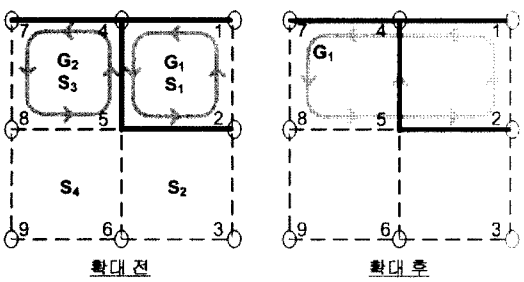

그림 7. 그룹의 확장

4단계. AGVS의 이동경로, 속력을 재 계획

공간 S_w에서 다수의 충돌들이 동시 다발적으로 발생하면, AGVS의 가장 큰 이슈인 교착상태가 발생된다. 교착상태는 AGVS에 AGV 대수가 많거나 혹은 특정 S_w의 AGV의 사용밀도가 편중되면 더욱 빈번하게 발생하게 된다. 이번단계는 확대 된 G_k안의 A_n들의 이동경로 할당법칙과 이로 인해 갱신되는 p-data(Renewed p-data; 이하 Rp-data)의 형성과정과 충돌회피를 위해 p-data가 갱신된 A_n에 의해 주행에 영향을 받게 되는 주행문제가 예측되지 않은 A_n들의 p-data의 갱신법칙을 소개한다.

확대 된 G_k안의 A_n들의 이동경로 할당법칙은 다음과 같다. 3단계를 거치면서 표 5의 확대 후 G_1에는 총 9대의 문제 발생된 A_n이 그림 7의 확대 후 그룹에 할당 된 S_w인 S_1, S_3위를 p-data대로 주행할 것으로 계획된다. 이때 0～9의 시간범위 동안에 시스템 내의 S_1, S_3의 흐름경로는 표 6과 같이 일 방향흐름 경로를 할당시킨다. 각 A_n들의 p-data를 검토하여 일 방향으로 할당된 S_w에서 중간노드의 플로이드 최단거리를 찾는다. 3단계를 거친 후 표 7에서와 같은 자료구조로 이미 G_1이 형성이 되어 그룹의 파라메타가 이미 존재한다. A_1은 V_1의 속력으로 1에서 9로

표 5. 확장의 단계

Step	G_k	그룹에 속한A_n	충돌 예상시간	충돌 공간S_w	G_k의 속력
1	G_1	A_1～A_8	0～6.42	S_1	V
	G_2	A_9	6～9	S_3	V
2	G_1	A_1～A_9	0～9	S_1 S_3	V

표 6. 일 방향의 흐름경로의 방향결정

SG의 외곽경로		
양방향 흐름경로	⇨	일방향 흐름경로
1↔4↔7 ↕ ↕ 2↔5↔8		1→4→7 ↑ ↓ 2←5←8
SG의 내부경로		
양방향 흐름경로	⇨	일방향 흐름경로
4↔5		5→4

가는 p-data를 가진다. 경로 중 노드1에서 7로 가는 이동경로는 S_w에 속한 경로에 속해있고, 노드 진입시간 또한 충돌예상 시간에 속해있다. 이 경로는 일방향성 경로 S_n에서의 최단거리 경로로 대체되고, 이 경로를 지날 때에는 G_1에 할당된 이동속력 V로 이동하게 된다. A_1이 노드1에서 9로 가기위해 계획 된 p-data는 표 7에서와 같이 Rp-data를 생성해낸다. 나머지 G_1에 속한 A_2~A_9들도 동일하게 Rp-data가 생성된다.

△T에 주행문제가 예측되지 않은 A_n들의 Rp-data의 생성법칙은 다음과 같다. 충돌이 예측되지 A_{10}은 G_1의 A_n에 속하지 않는다. A_{10}은 9에서 7로 이동하는 작업이 할당 되어있고, p-data는 표 7과 같다. 그 경로 중 8에서 7로 가는 경로는 G_1의 S_w에 속한 경로이고, 그 경로 노드의 진입시간은 G_1의 충돌예상 시간영역 0~9속에 있다. 따라서 A_{10}은 G_1의 일 방향 속성을 따른 노드8에서 7로 가는 최단경로가 할당되고, 그 경로를 지날 때에는 G_1의 파라메타의 속력 V로 주행할 것을 계획된다. 수평시간 계획법에 근거하여, PDA에서 순간에 실시간 주행문제 발생이 예측 될 때마다 주행문제를 해결할 수 있는 Rp-data를 생성하는 PSA의 4단계가 실행된다.

3.2.4 시스템 조정 에이전트

CA는 시스템시간 T_t를 관리하면서 시스템의 파라메타, DB에서 건네 오는 작업데이터, AGVS의 Rrt-data, Rp-data를 처리하는 역할을 한다. CA는 AGVS의 작업시작 모듈, 양방향흐름의 최단경로 생성모듈, 주행 중 모듈, 노드에서의 충돌감지 모듈들로 구분한다.

표 7. 주행문제가 예측되지 않은 AGV들의 p-data처리

그룹화 된 파라메타	AGV	rt-data	p-data(경로,도착시간,속력) Rp-data(경로,도착시간,속력)				
G_k : G_1	A_1	운영	1	4	7	8	9
- G_k에 속한A_n		(1	V_1	V_1	V_1	V	
⇒ V_1 V_2 V_3 V_4 V_5 V_6 V_7 V_8 V_9		→4)	0	3.75	7.5	11.25	14.25
		운영	1	2	5	8	9
- 충돌예상 시간범위 ⇒ 0~9		(1	0	3	6	9	12.75
		→2)	V	V	V	V_1	
- G_k의 속력 ⇒ V	…	…					
- 충돌공간 S_w ⇒ S_1 S_3	A_{10}	운영	9	8	7		
		(3	0	3	6		
- S_w에 속한 노드 ⇒ 1 2 4 5 7 8		→6)	V_{10}	V_{10}			
		운영	9	8	5	4	7
		(3	0	2.5	5.5	8.5	11.5
		→6)	V_{10}	V	V	V	

3.2.4.1 AGVS의 작업시작 모듈

작업 시작모듈은 시스템의 입력정보생성 모듈로부터 시스템 파라메타와 작업리스트를 수용하며, AGVS의 꺼짐 상태에서 켜짐 상태로 전환 시에 한번 실행되는 모듈이다. 최초에 마지막으로 위치한 A_n의 위치를 읽어 들여서 n들에게 작업리스트의 작업을 Dispatching하고, n마다 Dispatching된 작업을 최단경로 생성모듈로 전달한다. 또한 모든 n의 rt-data의 현재 상태를 대기상태로 만든다. 그리고 AGVS의 운영 중에는 주행 중에 작업이 완료되어 Rrt-data의 현재 대기상태인 모든 n에게 새로운 작업을 Dispatching할 수 있도록 작업데이터를 전달한다.

3.2.4.2 최단경로 생성모듈

최단경로 생성모듈은 Dispatching받은 대기상태인 n에게 할당받은 작업을 수행할 수 있도록 이동경로를 적재지점에서 하역지점을 이동하는 최단경로를 만들어내는 역할을 한다. 이때 나머지 n들의 p-data를 감안하지 않고서 오로지 작업을 할당받은 n의 시작지점과 목적지점의 최단경로를 발생시킨다. 이때 플로이드 최단경로 생성 알고리즘을 사용하고, 경로생성을 받은 n의 현재 상태를 대기에서 동작 상태로 전환시킨다. 또한 V_n의 속력을 토대로 A_n들의 p-data, rt-data를 생성, 갱신시킨다.

3.2.4.3 주행 중 모듈

주행 중 모듈은 모든 n들 중에서 할당된 작업을 완료하여 Rrt-data가 현재 대기상태인 n에게 새로운 작업을 Dispatching하고, 작업이 Dispatching된 n에 최단경로 생성모듈로 할당받은 작업을 전달하는 역할을 한다.

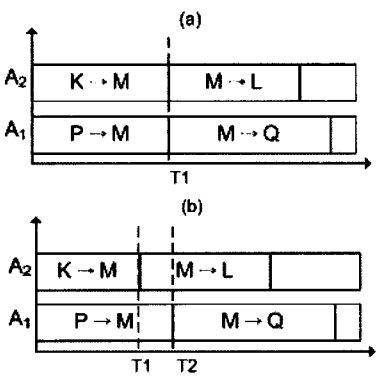

그림 8. 노드에서의 충돌해결

3.2.4.4 노드에서의 충돌감지 모듈

양방향 흐름가능 경로를 가지는 S_w에서 모든 A_n들은 각각의 최단경로를 주행 중 문제발생 될 것으로 예측 된 A_n의 문제경로만을 일 방향흐름 S_w에서 경로를 재 계획하였다. 이 경우 아크에서 충돌은 발생이 되지 않지만 일 방향흐름의 아크의 교차지점인 노드에서의 충돌은 처리하지 못한다. 충돌감지 모듈에서는 모든 n들의 rt-data를 고려해서 노드에서의 충돌을 발견하고, 동작명령을 할당하는데 그 과정은 다음과 같다.

Step1. 동작하는 모든 A_n의 Rrt-data를 검토하여 그림 8(a)와 같이 T_1시간에 노드M에서의 충돌을 예상한다.

Step2. 그림8(b)와 같이 T_1 - T_2의 시간동안 A_1에 주행 중 대기명령을 내리고, A_1의 p-data를 갱신한다.

Step3. 모든 A_n에게 주행, 대기, 하역, 적재와 같은 동작명령을 예약한다.

Step4. AGVS에서 처리를 위해 갱신된 Rp-data와 Rrt-data를 시스템의 운영을 위한 p-data, rt-data로 전환시키는 운영데이터 갱신 모듈로 정보를 전달한다.

4. 다중 에이전트 기반 양방향 AGV 시뮬레이터 개발

본 연구에서는 AGVS를 관할하는 조정 에이전트 및 다수의 응용 에이전트를 둔 다중 에이전트 시스템을 적용하여 AGVS전용 시뮬레이터에 적절한 임무 분산 처리가 가능하도록 설계하였고, VC++기반에서 다양한 그래픽이 가능한 OpenGL을 사용하여 시뮬레이터를 구현하였다.

4.1 다중 에이전트 기반의 AGVS 시뮬레이터의 구조

본 논문에서 제시하는 다중 에이전트 기술이 적용된 AGVS 시뮬레이터의 구조는 그림 9와 같다. 제시하는 AGVS 시뮬레이터는 크게 상단의 시스템의 입력정보 생성모듈, 중단의 다중 에이전트 시스템과 AGVS운영 데이터모듈, 하단의 시뮬레이터 엔진으로 구성되었다. 시뮬레

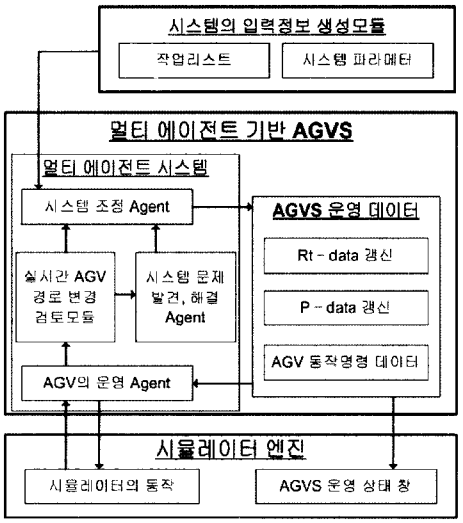

그림 9. AGVS 시뮬레이터의 구조

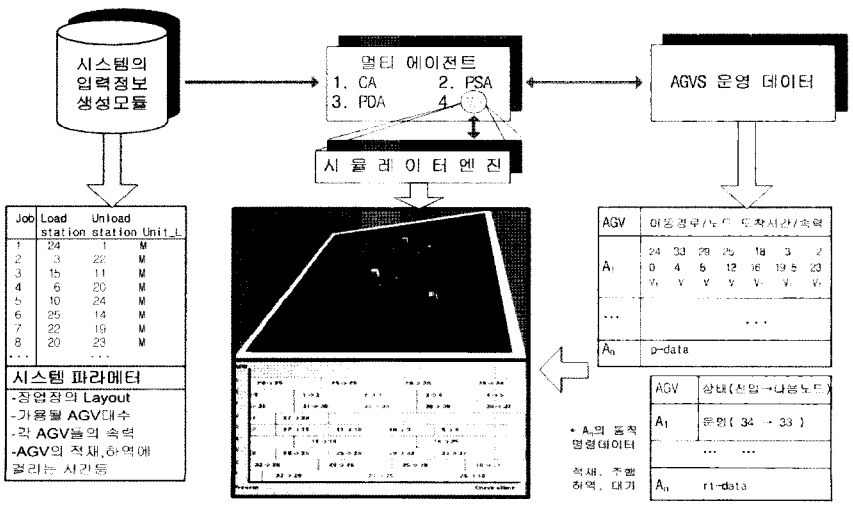

그림 10. 다중 에이전트에 근거한 AGVS 시뮬레이터의 개요

이터의 구조에 근거하여 본 논문에서 제시하는 다중 에이전트 시스템 기반의 AGVS 시뮬레이터는 그림10과 같은 개요로 작동된다.

4.2 AGVS 시뮬레이터의 데이터 흐름도

본 연구를 위해서 개발한 AGVS 시뮬레이터는 그림 11과 같은 순서에 따라 구동된다. 그림 11에서 CA는 dispatching 된 작업을 기반으로 p-data와 rt-data를 최초에 발생시키고, PDA, PSA, MA와의 메시지 교환을 하면서 임무를 수행하고, 이를 통해 시스템이 유기적으로 연결되어 운영된다. 운영데이터 갱신모듈에서는 Rp-data와 Rrt-data를 p-data, rt-data로 갱신하고, 각 AGV들의 동작 명령을 할당한다. 그리고 갱신된 정보는 칸트차트를 갱신하며, MA에게 전달한다. MA는 전달받은 운영데이터의 AGV 동작 명령을 토대로 시뮬레이터 엔진의 AGV를 동작시키고, 시뮬레이터 엔진의 AGV를 동작시키고, rt-data와 p-data를 서로 주고받는다. PDA 호출판단 모듈에서는 모든 AGV들 중에서 rt-data의 경로가 변하는 순간에 PDA를 호출하여 CTP안에서 AGV의 주행문제를 판단한다. 주행상의 문제가 발견될 경우 PDA에서 주행문제 발견 데이터인 d-data를 생성한다. d-data가 발생되면, PSA 호출 판단모듈에서 PSA를 호출하여 문제를 해결 할 수 있는 우회경로, 속력을 재 할당한다. 시스템이 운영 중일 때에는 PSA를 통해 갱신된 Rrt-data와 Rp-data를 CA의 주행 중 모듈로 전달한다. CA에서는 AGVS의 전체시간을 관리하고, 작업 완료 된 AGV에게 는 새로운 작업을 할당시켜 AGV의 대기상태 시간을 최소화 한다.

5. 시뮬레이션 실험

다음의 가정을 바탕으로 시뮬레이터는 개발이 되었다.

- 모든 AGV들이 작업장에서의 화물을 적재, 하역하여 주행경로로 다시 이동하는데 걸리는 시간은 동일하다.
- 우선 작업완료 후 대기 중인 AGV에게 먼저 작업을 할당하는 dispatching rule을 가진다.
- 모든 작업장에서 적재, 하역하는 단위 화물의 양, AGV의 처리능력도 동일하다.
- AGV들에게 작업장이 중복할당 되지 않는다.

또한 시뮬레이션을 실험을 위해서 네트워크의 사이즈와 연결정보, 연결거리, AGV의 대수, 작업리스트, 각 AGV들의 주행 안전속력, 그룹에서의 후면충돌을 해결하기 위한 안전속력, CTP의 결정을 위한 시간범위와 같은 입력정보가 필요하다.

그림 12는 예제와 같은 배치형태에서 9대 AGV들의 주행 중 발생되는 교착상태를 시뮬레이터 엔진의 AGVS

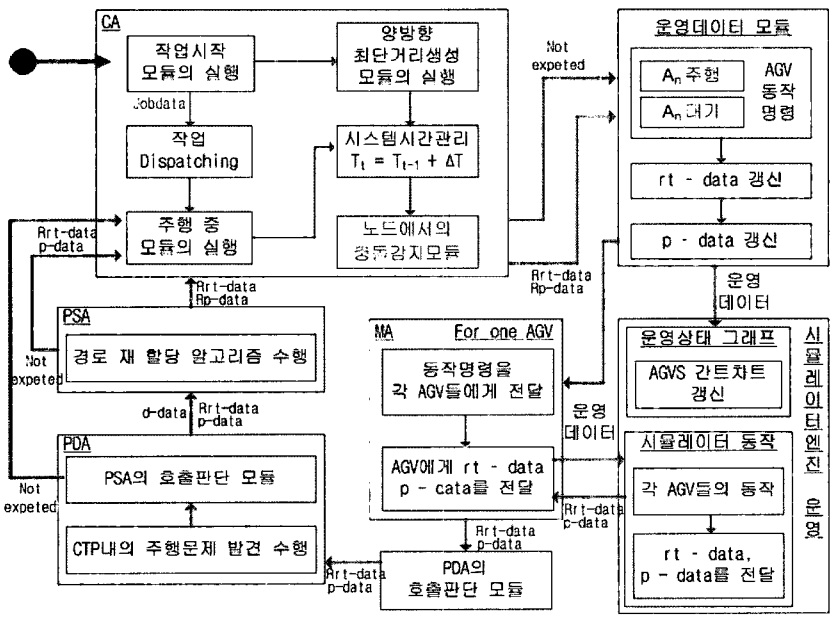

그림 11. 시뮬레이터의 순서도와 데이터 흐름도

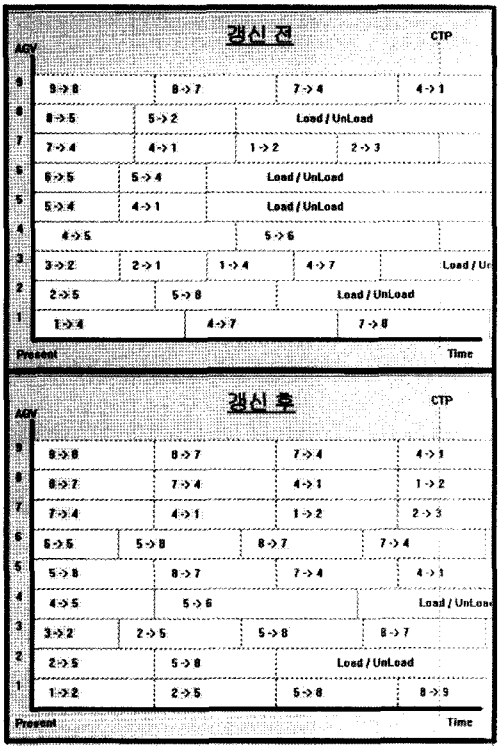

그림 12. 주행문제의 해결과정

표 8. AGVS의 d-data

충돌 AGV	충돌유형	충돌아크
A_1 : A4	정면충돌	(1, 4)
A_1 : A7	정면충돌	(1, 4)
A_1 : A9	정면충돌	(4, 7)
A_2 : A8	정면충돌	(2, 5)
A_3 : A7	정면충돌	(1, 4)
A_4 : A5	정면충돌	(4, 5)
A_4 : A6	정면충돌	(4, 5)

표 9. 수평시간계획에 근거한 AGV에 충돌할당

충돌AGV	충돌유형	충돌경로
A_1	정면충돌	[1, 4]
A_2	정면충돌	[2, 5]
A_3	정면충돌	[1, 4]
A_4	정면충돌	[4, 5]
A_5	정면충돌	[5, 4]
A_6	정면충돌	[5, 4]
A_7	정면충돌	[4, 1]
A_8	정면충돌	[5, 2]
A_9	정면충돌	[7, 4]

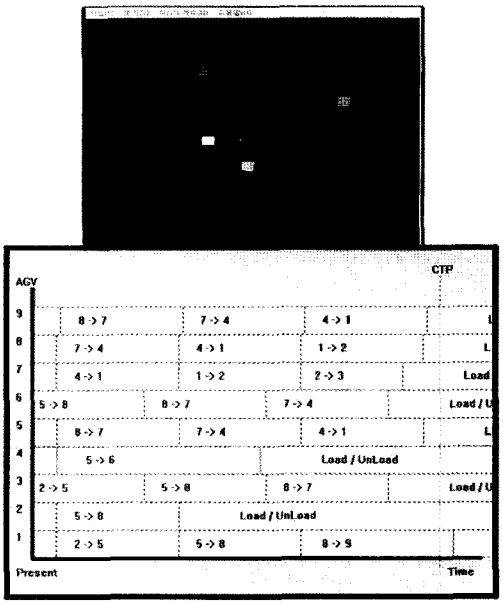

그림 13. 시뮬레이터의 작동

칸트차트에서 그 해결과정을 보여준다. 그림 12의 갱신 전 그림은 교착상태가 발생이 예측된 AGVS 칸트차트 이다. 운영되는 AGVS에서 PDA를 통해 발견한 표 8의 d-data를 토대로 각 AGV마다 수평 시간계획에 근거하여 우선적으로 해결해야 하는 주행문제를 표 9와 같이 할당할 수 있다. 이를 PSA를 통해 주행경로와 경로에서의 속력을 재 계획 시키면 그림 12의 갱신 후와 같이 교착상태가 해결된 갱신 된 AGVS 칸트차트가 출력된다.

갱신 된 칸트차트에서 CTP의 주행계획을 보면 A_2, A_4의 rt-data 다음노드에 해당하는 노드5의 진입시간이 같음을 확인 할 수 있다. 이와 같은 상황에서 CA의 노드에서의 충돌감지 모듈들이 실행되어 예측된 노드에서의 충돌을 해결하기 위해 A_2혹은 A_4에게 p-data를 갱신시킨다. 다중 에이전트 기반의 양방향 주행가능 AGV 시뮬레이터 엔진을 통해 AGV들의 작업 수행 모습을 볼 수 있고, AGVS 칸트차트를 통해 운영 중의 문제들을 해결하는 과정과 AGVS가 동작될 때에 운영 정보를 확인 할 수 있다. 본 논문에서 그림 13과 같이 시뮬레이션 해 본 결과 교착상태가 발생되지 않고 할당된 작업을 모두 완료함을 확인할 수 있었고, 그 과정을 AGVS 칸트차트를 통해서 주행되는 AGV들의 현 상황을 모니터링 할 수 있었다.

개발한 시뮬레이터는 그림 14의 데이터 제너레이터를 사용해서 원하는 시스템 파라메타들을 입력한 후 그림 15

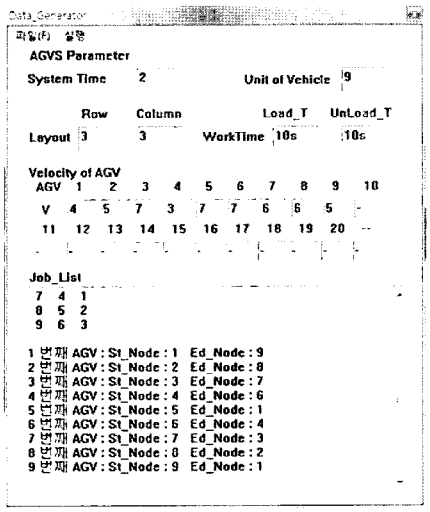

그림 14. 데이터 제너레이터

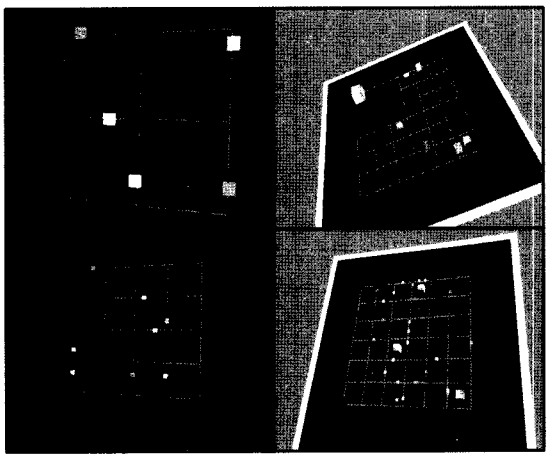

그림 15. 다양한 상황을 시뮬레이션 가능한 시뮬레이터

에서와 같이 여러 형태의 배치형태에서 AGV의 대수를 달리하여 실험을 할 수 있다. 그 결과 실시간 교착상태 해결 알고리즘에 의해서 주행문제 없이 할당된 작업을 완료하는 것을 관찰하였다. 실험 결과 AGV대수에 비해 주행공간이 협소한 상황의 시스템 파라메타가 주어진 상황에서는 거의 모든 AGV들이 정면충돌을 피하기 위해서 수평시간 계획법 의해 우회경로 재 할당이 반복적으로 시행되었다. 그 결과 전체 시스템의 흐름경로가 일 방향 주행가능 경로로 전환되었고, 속력이 다른 모든 AGV들이 후면 충돌을 피하기 위해서 동일 속력으로 계획되어져 이를 시뮬레이션을 통해 확인이 가능했다.

6. 결론 및 추후연구

본 논문에서는 다중에이전트 시스템을 AGVS에 도입하여 교착상태를 해결하는 양방향 주행가능 한 실시간 알고리즘을 제시했고, 시뮬레이터는 조절이 가능한 시스템 파라메타에 의존적이지 않고, 어떠한 시나리오에서도 손쉽게 시뮬레이션을 해 볼 수 있는 AGVS전용 시뮬레이터를 구현하였다.

본 연구에서 개발한 시뮬레이터는 교착상태 해결 알고리즘을 통해 주행흐름 경로가 일 방향으로 결정된 선 계획된 AGVS에서도 시뮬레이터 실험이 가능하다. 따라서 특정 주어진 시스템 파라메타 환경에서 일 방향 흐름경로 혹은 양방향 흐름 경로의 선택과 같은 대안결정 연구로 확장시킬 수 있다. 이러한 시스템을 웹에서 서버/클라이언트 구조로 시스템을 구축하여 사용자들이 손쉽게 공장을 설계할 수 있도록 하고, 이를 통해 효율적인 공정계획 수립을 위한 시스템으로 발전시킬 것이다.

감사의 글

이 논문은 2006년도 특정기초연구지원 사업으로 과학기술부의 지원에 의하여 연구되었음.

참 고 문 헌

1. Iris F.A. Vis(2006), "Survey of research in the design and control of AGV system", European Journal of Operational Research, Vol. 170, 2006, pp. 677-709.
2. Maza, S. and Castagna, P.(2005) "A performance-based structural policy for conflict-free routing of bi-directional automated guided vehicles" Computers in industry, Vol. 56, No. 7, pp. 719-733.
3. Correa, Langevin, Rousseau, L. M.(2004), "Dispatching and Conflict-Free Routing of Automated Guided Vehicles: A Hybrid Approach Combining Constraint Programming and Mixed Integer Programming", Lecture notes in computer science, Vol. 3011, pp. 370-379.
4. RAJOTIA, S. and SHANKER, K. and BATRA, J. L.(1998) "A semi-dynamic time window constrained routeing strategy in an AGV system", INT. J. Prod. Res, Vol. 36, No. 1, pp. 35-50.
5. K.-C.KO and P.J.EGBELU(2003) "Unidirectional AGV guidepath network design: a heuristic algorithm" INT. J. Prod. Res, Vol. 41, No. 10, pp. 2325-2343.

6. Seo, Yoonho and Lee, Chulung(2007) "Tabu search algorithm for flexible flow path design of unidirectional automated-guided vehicle systems" OR spectrum : Quantitative approaches in management, Vol. 29, No. 3, pp. 471-487.
7. H. Thomas and P. Wenger(1995) "On planning velocities and trajectories for multiple mobile robots", Proceedings of the Sixth Topical Meeting on Robotics and Remote Systems, Monterey, CA.
8. Fanti, Maria Pia(2002) "Event-based controller to avoid deadlock and collisions in zone-control AGVS" Inter national journal of production research, Vol. 40, No. 6, pp. 1453-1478.
9. Chang W. KIM and J.M.A. Tanchoco(1991) "Conflict-free shortest-time bidirectional AGV routing" INT. J. Prod. Res, Vol. 29, No. 12, pp. 2377-2391.
10. 조영임, 「인공지능시스템 : 7장 지능적 에이전트 시스템」, 홍릉과학출판, 2003.
11. Tapio Heikkila, Martin Kollingbaum, Paul Valckenaers, Geert-Jan Bluemink(2001), "An agent architecture for manufacturing control: manAge", Computers in Industry 46, pp. 315-331.

이 재 용 (99jaeyong@korea.ac.kr)

2006 고려대학교 전자공학과 학사
2006~현재 고려대학교 산업시스템 정보공학과 석사

관심분야 : SCM시뮬레이터, 휴리스틱

서 윤 호 (yoonhoseo@korea.ac.kr)

1984 고려대학교 산업공학과 학사
1990 미국 Pennsylvania State University. 산업공학과 석사
1993 미국 Pennsylvania State University. 산업공학과 박사
1993~2003 울산대학교 산업공학과 교수
2003~현재 고려대학교 산업시스템정보공학과 교수

관심분야 : 제조, 조립 및 물류 시스템의 VR적용 지능설계