'93 한국자동제어학술회의논문집 (1993.10.20~22)

다수 무인운반차 시스템을 위한 관리제어기 설계 기법

°이정훈*, 최명환**, 이범희*, 김정덕***, 박 현***
*서울대학교 제어계측공학과, **강원대학교 제어계측공학과, ***㈜삼성항공

Supervisory Controller Design Technique for Multiple-AGV Systems

°J.H. Lee*, M.H. Choi**, J.D.Kim***, H. Park***
*Dept. of Control & Instrumentation Eng., Seoul National University
**Dept. of Control & Instrumentation Eng., Kangwon National University
***Samsung Aerospace

A B S T R A C T

A supervisory controller design technique for multiple-AGV systems is presented in this paper. The guidepath is represented in the form of a network, and its modifications are easily tested. The network has two-layered structure, where the path sets between each two nodes are made in advance using the K-shortest path algorithm. Occupation times for all links are stored in link-occupation table, and are updated after the dispatching time. Dispatching and scheduling for each AGV are optimized in terms of minimum-time objectives. In all times, the paths are guaranteed to be conflict-free and deadlock-free. The simplicity and flexibility on this control scheme make the supervisory controller suitable for real applications.

1. 서론

다수대의 무인운반차가 동시에 주행하는 궤도(Guidepath)주행식 무인운반차 시스템(AGVS)에서, 무인운반차는 미리 설치된 궤도를 이탈하지 못한다. 따라서 이러한 시스템에서의 관리제어기(Supervisory Controller)는, 주어진 궤도에 대한 경로지도(Path-map)를 작성·저장하고 이를 통하여 각 무인운반차의 배차(Dispatch)와 스케쥴링(Scheduling)을 수행하는 것을 그 기능으로 한다[1].

이러한 시스템을 실제 플랜트에 적용하기 위해서는, 관리제어기의 제어기법이 단순성(Simplicity)과 유연성(Flexibility)을 지녀야 한다[2]. 제어기법의 단순성이란, 실제의 일반적인 플랜트의 궤도에서 사용되는 단순화된 구조를 이용하여, 안정되고 무인운반차 상호간의 충돌(Conflict)과 막힘(Deadlock)이 없는 최적경로(Optimal Path)로 각 무인운반차를 주행시키는 것을 의미한다. 한편, 관리제어기의 유연성이란 궤도의 수정이나 첨삭에 대한 처리, 각각의 상황에 대한 무인운반차의 배차 및 스케쥴링이 유연하고 효율적으로 행해져야 함을 뜻한다.

현재 많은 플랜트에서 사용되는 고정패턴(Fixed Pattern) 주행방식은, 무인운반차 상호간에 연계된 조건부 고정패턴을 미리 설정한 뒤 각 무인운반차의 현재위치와 상태에 따라 조건부배차하는 배차계획법이다[3,4]. 이 방식은 그러나 유연생산체계(FMS)가 지니는 유연성을 효율적으로 활용할 수 없고, 기존 궤도가 수정되는 경우 모든 고정패턴을 다시 작성하여야 하는 단점을 지니며, 주행경로의 최적화에 한계를 갖는다. 한편, 이론적인 방향에서의 접근들은 상당수 이러한 단점을 극복하고 있다. 이 경우 문제의 핵심은 최적경로의 생성과 충돌·막힘현상의 제거이다. 몇몇의 FMS 시뮬레이터들은 요구노드(Request Node)로부터 최단거리에 위치한 대기중인 무인운반차를 호출하여 최단거리경로로 주행시키는 방법을 사용하고 있다[5,6]. Broadbent(1985)와 Sen(1990), Wang(1991)은 무인운반차의 스케쥴링 기법을 보다 발전시켜, 각 노드의 점유시간을 시간표형태(Timetable)로 저장하여 두고서 매번의 스케쥴링 이후 이를 갱신(Update)하는 방식을 사용하였다[1,7,8]. 이를 통하여, 최단거리경로에 다른 무인운반차가 존재하는 경우는 차순의 최단거리경로(Second-shortest Path)를 탐색하도록 할 수 있다. 이때 매회의 경로탐색을 온라인(On-line)으로 수행하는 것은 소요시간면에 있어서 큰 부담이며, 따라서 Wang은, 최단거리경로는 오프라인(Off-line)으로 미리 찾아 저장하여 두고 차순최단거리경로만을 온라인으로 탐색한다[1]. 다른 한편으로, Kim and Tanchoco[9], Egbelu and Tanchoco[10] 등은 무인운반차가 정속도로 운동함을 전제하고서, 시간윈도우그래프(Time Window Graph) 등을 이용한 최적경로 스케쥴링 기법을 제안하고 있다. 그러나, 이들 이론적인 접근들은 실제의 많은 상용화된 플랜트에의 응용으로부터 상당부분 유리되어 있다. 실제 시스템의 경우, 무인운반차의 궤도상의 정확한 위치는 관리제어기가 파악하지 못하며, 단지 궤도상의 지정된 지점을 통과하는 순간에 한하여 무인운반차의 위치가 확인된다. 따라서, 주행시의 위치오차로 인한 충돌을 막기 위해, 궤도를 여러개의 분선(Segment)으로 나누어 하나의 분선에 둘 이상의 무인운반차가 동시에 진입할 수 없도록 제어하는 경우가 대부분이다[11]. 이러한 시스템에서는 무인운반차의 정확한 위치의 파악을 가정하는 기존의 관리제어이론이 비효율적이거나 무력할 수 있으며, 분선화된 궤도 구조를 이용하는, 보다 단순하면서도 효율적인 관리제어기법이 요구된다.

본 논문은 궤도 방식의 일반적인 무인운반차 시스템을 대상 플랜트로 하며, 사용하는 유도방식에 의존하지 않는다. 궤도는 많은 수의 짧은 분선들로 이루어지며, 이러한 분선들을 링크(Link)로 정의하고 인접한 분선과 분선의 경계를 노드(Node)로 정의하였다. 노드는 궤도상에 장착된 물리적인 표지체일 수도 있고, 개념적으로만 존재하는 가상의 구획표지일 수도 있다. 무인운반차가 노드를 통과하는 때에 관리제어기와 각 무인운반차 간의 통신이 이루어지며, 이 순간에 무인운반차는 통과중인 노드의 번호를 관리제어기로 송신한다. 무인운반차의 주행속도는 궤도의 각 링크에 부여된 속도계수(Velocity Parameter)에 의해 결정된다.

본 논문에서는, 분선화된 궤도 구조를 이용하는 실제 무인운반차 시스템에서 다수대의 무인운반차를 운용할 때의 효율적인 관리제어 기법을 제안한다. 제안하는 관리제어기는 궤도의 설계와 수정, 삽입/삭제에 유연하게 대처한다. 모든 무인운반차는 출발지점으로부터 목적지점까지 최단시간경로로 주행하며, 무인운반차 상호간의 충돌(Conflict)이나 막힘현상(Deadlock or Gridlock)은 일어나지 않는다. 시스템의 궤도가 복잡하여 대체경로가 다수개 존재할 때, 제안되는 관리제어기법은 보다 높은 효율을 보인다. 많은 연산이 미리 오프라인으로 수행되므로 실시간의 관리제어가 가능하며, 관리제어의 알고리즘이 충분히 단순하므로 실용화에 적합하다.

3절에서는 관리제어환경의 구성기법을 제안하였다. 궤도를 네트워크로서 표현하며, 이의 수정과 삽입/삭제가 유연하게 수행될 수 있도록 하여 관리제어기의 유연성을 최대한 확보하도록 하였다. 또한, 무인운반차의 경로는 다수의 대체경로까지를 오프라인으로 탐색하도록 하여 실시간의 안정한 관리제어가 가능하도록 하였으며, 이때 네트워크를 중층적 구조로 표현함으로써 대체경로 탐색을 보다 용이하게 수행하였다. 4절에서는 관리제어기의 제어기법을 제안하였다. 3절에서 제안한 경로시간 표현을 이용하여 궤도상의 모든 링크의 경유시간과 점유상태를 미리 예견할 수 있도록 하였으며, 이들 상태정보를 링크점유시간표에 저장하였다. 이 링크점유시간표를 사용하여, 무인운반차가 순환로를 갖지 않는 최단시간경로(Minimum-time Path)로 주행하도록 하였다. 또한, 스케쥴링 시에 무인운반차 상호간의 충돌과 막힘현상이 발생하지 않는 경로만을 선택하도록 함으로써, 최단시간경로 스케쥴링 문제(Minimum-time Path Scheduling Problem)를 해결하였다. 한편, 이러한 전 과정은 제어기법상의 단순성으로 인하여 실용화가 용이하다.

2. 문제의 설정과 접근방법

무인운반차 시스템의 주행궤도를 네트워크로서 표현하는 것은 일반적으로 행해지는 방법이며[1,7,8,10,12,13,14], 이를 그림 1에 예시하였다.

주행궤도는 다수의 노드와 링크로 구성된다. 링크는 궤도를 구성하는 가장 기본적인 요소로, 노드와 노드의 연결선이 된다. 무인운반차는 링크를 추종하여 주행하며, 정상적인 주행 상태에서는 링크상에 정지하지 않는다. 노드는 링크와 링크가 만나는

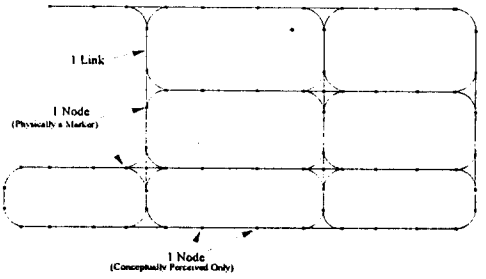

그림 1. 주행궤도의 네트워크표현

점으로, 무인운반차가 정지하거나 출발할 수 있는 위치이다. 따라서, 무인운반차의 주행경로(Path)는 링크들과 경유노드들의 연결체로서 이루어지며, 출발노드와 경유노드 그리고 종착노드 번호들의 수열(Sequence)로서 표현된다. 스테이션(Station)은 무인운반차가 화물을 이적재하거나 배터리 충전 등의 작업을 행하는 노드로, 스테이션도 노드의 일종으로 간주된다.

고려 대상인 무인운반차 시스템은 다음의 특성을 지니도록 만들어진다.

- 모든 궤도는 양방향(Bidirectional)이다.
- 각 노드와 링크에는 고유식별 번호를 부여한다.
- 중앙제어기는 무인운반차가 어느 링크 혹은 노드에 위치하는지만을 알 수 있으며, 링크상에서의 정확한 위치는 알려지지 않는다.
- 무인운반차 상호간의 충돌을 방지하기 위해, 하나의 링크에는 한 대의 무인운반차만이 존재할 수 있도록 제어한다.

본 논문에서는 이상의 시스템을 위한 관리제어환경 설계기법과 관리제어기 설계 기법을 제안한다.

3. 주행경로집합의 설정

(1) 주행궤도의 네트워크표현

① 궤도의 수학적 표현

무인운반차 시스템의 주행궤도를 네트워크로서 표현할 때, 각 노드와 링크는 네트워크를 이루는 그래프(Graph)의 요소들이 된다[14].

각 노드에 1부터 n까지의 고유식별번호를 부여하면 노드공간(Node Space) N은 $N = (1, 2, 3, ..., n)$으로 표현된다. 노드 i로부터 노드 j로 향한 링크 $l(i, j)$가 존재할 때 $l(i, j)$는 링크공간(Link Space) L의 원소가 되며, 노드 i는 노드 j로 인접한다(Adjacent to)고 하고, 노드 j는 노드 i로부터 인접한다(Adjacent from)고 한다[16]. 노드 i와 노드 j 사이의 궤도가 양방향이면, 링크는 $l(i, j)$와 $l(j, i)$의 2개가 양 노드 사이에 존재하게 된다. 한편, 이 링크의 물리적인 길이를 경로길이(Path Length) $d(i, j)$로 정의한다. 그리하면, 주어진 네트워크는 노드공간 N과 링크공간 L로서 이루어지는 그래프 G로 표현된다[15].

$$G = (1, 2, 3, ..., n ; l(i, j)), \quad l(i, j) \in L \quad (3.1)$$

노드 p로부터 도달가능(Accesible)한 노드 q까지의 경로 $P:(p, q)$는 링크 $l(p, p_1)$, $l(p_1, p_2)$, ... , $l(p_m, q)$들의 순서

열(Ordered Sequence)로써 이루어지며, 경로 $P:(p,q)$의 길이는 아래와 같이 정의된다[15].

$$d(P) = d(p,p_1) + \ldots + d(p_m,q) \qquad (3.2)$$

주행거리가 아닌 주행시간을 최소화하기 위해서는 경로길이 외에 경로의 통과에 소요되는 시간을 계산하여야 한다. 이는 관리제어 기법에만 주목하는 상당수의 기존 연구에서 결여된 것으로, 본 논문에서는 경로를 통과하는 시간을 가리키는 수학적인 표현으로서 경로시간(Path Time) 개념을 제시한다. 경로시간은 출발과 정지, 가/감속, 그리고 회전에 소요되는 시간을 포함하는 개념이다. 링크 $l(i,j)$의 경로시간값 $\tau(i,j)$를 다음과 같이 정의한다.

$$\tau(i,j) \equiv (C_{ij}V)^{-1}d(i,j) \quad , \quad 0 < C_{ij} \le 1 \qquad (3.3)$$

여기서 C_{ij}는 링크 $l(i,j)$에 부여된 속도계수로 회전구간 등의 감속구간에서는 $C_{ij} < 1$ 이며, V는 무인운반차의 최고주행속도로서 상수값으로 주어진다. 각 링크 상에서의 무인운반차의 주행속도는 무인운반차의 최고주행속도와 각 경유링크에 부여된 속도계수에 의하여 결정되도록 하였다.

경로 $P:(p,q)$의 경로시간 $\tau(P)$는 주행중의 가속과 감속, 회전에 소요되는 시간을 고려하여야 하며, 다음과 같이 정의한다.

$$\tau(P) \equiv \sum_{i,j}(C_{ij}V)^{-1}d(i,j) + \sum_m \alpha_m + \sum_n \beta_n + \sum_p \gamma_p + \sum_r \eta_r \qquad (3.4)$$

여기서, α_m은 가속비용, β_n은 감속비용이며 γ_p은 회전비용, η_r는 경로주행중의 일시정지시간이다. α_m, β_n은 무인운반차의 주행속도의 변화로 인한 시간오차를 보정하기 위한 값으로서, 가/감속이 그림 2에서 보이는 사다리꼴의 속도 프로파일에 의

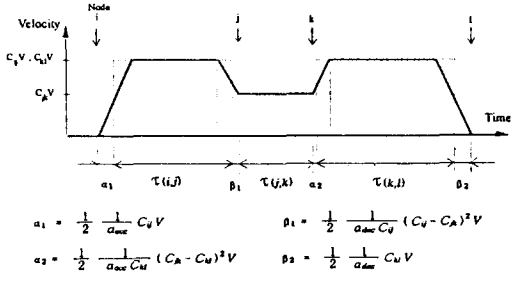

그림 2. 사다리꼴의 속도 프로파일과 가속비용/감속비용

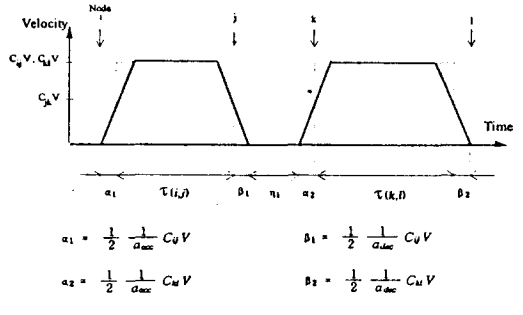

그림 3. 일시정지시간이 존재하는 경우의
사다리꼴의 속도 프로파일과 가속비용/감속비용

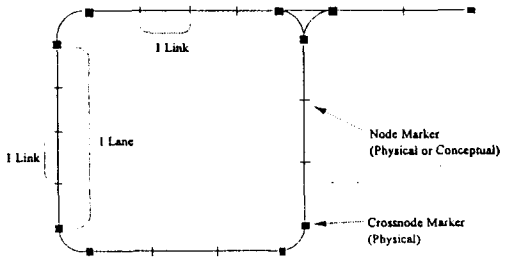

그림 4. 궤도의 중층구조 네트워크표현

해 이루어지는 경우 α_m과 β_n은 그림의 α_1,α_2와 β_1,β_2로 주어진다. 만일 무인운반차가 경로상에서 정지한 후 재출발한다면 이때의 속도 프로파일과 가/감속비용은 그림 3과 같이 주어진다. 한편, γ_p은 무인반송차의 회전능력에 의해 결정된다. 이러한 경로시간 표현을 통하여, 주어진 시간에서의 무인운반차의 궤도상 위치를 계산할 수 있다.

② 궤도표현의 저장과 수정

궤도를 표현하는 네트워크는 노드들과 링크들로 구성된다. 분선화된 궤도 구조에서는 궤도를 짧은 길이의 많은 링크들로 분할할 것이 요구된다. 그러나, 노드집합의 모든 노드들이 경로집합의 생성시에도 필요한 것은 아니므로, 경로집합의 생성에 반드시 필요한 노드들을 교점노드(Crossnode)로 정의하여 별도로 관리한다[6]. 3개 이상의 링크가 연결된 노드와 서로다른 링크형태(Link Type)의 링크가 연결된 노드가 교점노드가 되며, 두 인접한 교점노드간의 경로는 레인(Lane)으로 정의된다(그림 4). 이는 경로집합의 생성시에 알고리즘의 적용환경을 개선함과 동시에 계산시간의 단축효과를 얻기 위한 것이다.

이를 위하여, 궤도 모델을 그래픽 에디터 상에서 작성할 때 그래픽 에디터가 이 궤도를 다음 5개의 자료화일(Data File)로 만들어 저장하도록 하였다.

- 교점노드 화일 : 교점노드의 좌표와 스테이션 이름을 저장.
- 레인 화일 : 레인의 시작교점노드, 종료교점노드, 레인형태, 속도계수, 길이 등을 저장.
- 노드 화일 : 노드의 좌표를 저장.
- 링크 화일 : 링크의 시작노드, 종료노드, 속도계수, 길이 등을 저장.
- 교점노드-노드 맵핑 화일 : 교점노드-레인과 해당귀속노드를 저장.

5개의 자료화일들로 분할하는 이유는, 비록 각 화일간의 자료의 중복은 허용하더라도 이를 실행화일에서 입력받을 때에 하나 혹은 두개의 자료화일만을 입력하여 처리하도록 함으로써, 궤도의 크기가 큰 경우의 실시간 처리를 용이하게 하기 위한 것이다. 또한, 궤도의 수정시에는 에디터상에서 자료화일의 해당 부분만을 수정하도록 하여, 궤도변경에 대한 유연성을 보장하도록 하였다.

(2) K개의 최단거리경로의 생성

무인운반차의 최단시간경로는 경로길이 $d(P)$가 아닌 경로시간 $\tau(P)$를 최소로 하는 경로이다. 만일 장애물이 궤도상에 존재하지 않는다면 최단시간경로는 최단거리경로(Shortest Path)와 일치하지만, 다수대의 무인운반차가 동시에 주행하는 시스템의 경우에는 반드시 일치하지 않을 수 있다. 따라서, 최단거리경로 상에 다른 무인운반차가 정지 혹은 이동중인 경우의 최단시간경로는 차순의 최단거리경로(2nd-shortest Path) 혹은 그 외의 최단거리경로일 수 있다[1]. 그러나, 궤도가 복잡한 무인운반차 시스템에서 매회의 최단거리경로와 차순최단거리경로, 기타의 경로를 온라인으로 탐색한다는 것은 상당한 시간을 요하는 일이므로, 노드의 수가 많은 경우 이들의 탐색을 오프라인으로 해둘 필요가 있다. 계산량의 관점에서 보면, 노드의 수가 N인 네트워크에서의 최단거리경로 탐색 알고리즘의 계산량은 Floyd-Warshall의 알고리즘의 경우 $O(N^3)$이며, 동일한 노드수의 네트워크에서의 K개의 최단거리경로(K-shortest Path) 탐색 알고리즘의 계산량은 Shier의 알고리즘의 경우 $O(KN^3)$이다[16,17,18]. 이들 탐색 알고리즘은 노드수의 제곱 혹은 세제곱에 비례하여 계산시간이 증가하므로, 자료화일에서의 노드 중 탐색에 유의미한 노드를 교점노드로 정의하고 이들만을 고려하여 탐색하도록 함으로써 탐색소요시간을 단축시킬 수 있다. 이를 위해, 오프라인의 경로탐색과정에서는 교점노드-레인층의 자료화일을 사용하여 탐색하도록 하였다.

본 논문에서는 K개의 최단거리경로 탐색 알고리즘을 사용하여, 최단거리경로와 동시에 최대 K-1개의 대체가능한 경로를 미리 오프라인으로 찾아 저장하여 두었다. K개의 최단거리경로 탐색 알고리즘은 다수최단거리경로 탐색 알고리즘이라고도 하는데, Generalized Floyd-Warshall 알고리즘[18], Shier의 알고리즘[16,17,18], Dreyfus의 알고리즘[19], Yen의 알고리즘[19] 등이 있으며, 여기에서는 계산효율이 다소 낮으나 메모리의 사용량이 적은 Shier의 알고리즘을 사용하였다. K개의 최단거리경로 문제에서는 단순한 최단거리경로 문제와 달리 순환경로(Cycle)가 문제의 해(Solution)에 나타나는데, 네트워크를 노드-링크가 아닌 교점노드-레인으로 표현하여 해를 구함으로써 순환경로의 발생을 최소화할 수 있다. 탐색된 경로는 교점노드의 순서열로서 화일로 저장되며, 이 때 경로의 물리적인 길이 $d(P)$가 최소인 경로부터 순서대로 정렬(Sorting)된다.

4. 최단시간경로 탐색

(1) 동적인 링크점유시간표 구성

무인운반차의 주행속도는 무인운반차의 최고주행속도와 각 경유링크에 부여된 속도계수에 의하여 유일하게 결정되도록 하였으므로, 무인운반차의 경로와 경로주행 출발시각으로써 무인운반차의 궤도상의 위치를 계산할 수 있다. 무인운반차의 위치를 계산하여 어느 링크가 언제 무인운반차에 의하여 점유되는지를 계산하고, 이 자료들을 링크점유시간표(Link Occupation Table)에 기록하여 관리한다. 관리제어기는 분산화된 경로 구조를 사용하며, 이미 점유된 링크에는 다른 무인운반차가 진입하지 못하도록 제어함으로써 충돌(Conflict)을 방지한다[11].

각 무인운반차의 주행에 의해 링크점유시간표가 매회 갱신되어야 하므로, 문제의 축소를 통하여 이를 해결한다[8]. 즉, 시스템에 V대의 무인운반차가 존재하는 경우, V-1대의 경로와 이로 인한 링크점유시간은 이미 결정된 것으로 하고, 나머지 한 대의 경로와 이의 링크점유시간을 계산한 뒤 링크점유시간표를 갱신한다. 이는 실제에 있어서는, 한대의 무인운반차의 스케쥴링시에 다른 무인운반차의 이미 결정된 경로를 변화시키지 않으며, 한번에 한 대의 무인운반차만을 배차한다는 의미를 지닌다.

무인운반차는 출발시와 가속구간진입시 그리고 일시정지후 재출발시에 가속이 이루어지며 반대의 경우 감속하게 되므로, 링크점유시간을 계산할 때에는 가속비용과 감속비용을 부가함으로써 이를 고려하였다. 또한, 무인운반차의 실제주행시에 발생하는 위치오차를 고려하기 위하여, 계산된 링크점유시간의 전후에 안전여유(Safety Allowance)를 주었다. 안전여유의 크기는 노드마커(Node Marker)의 존재여부에 따라 달라지는데, 궤도상의 모든 노드마다 마커가 물리적인 표지체로서 존재하는 경우에는 노드통과시에 실제의 무인운반차의 위치를 확인하여 링크점유시간표를 보정하여 줄 수 있으므로, 안전여유의 크기 ε이 고정된 상수로 정해진다. 반면에 물리적인 마커가 교점노드에만 존재하는 궤도의 경우에는, 노드는 개념적으로만 존재하는 것이므로 오차의 누적현상이 있을 수 있으며, 따라서 다음 교점노드까지의 주행에서 안전여유 역시 누적하여 ε, 2ε, 3ε, ... 로 주었다. 한편, 무인운반차가 교점노드를 통과하는 경우에는 교점노드와 연결(Connected)된 모든 링크들을 점유된 것으로 함으로써 교차점에서의 충돌을 방지하였다. 또한, 노드에 정지 혹은 정차대기(Staging)중인 무인운반차는 해당 노드와 연결된 모든 링크를 점유하고 있는 것으로 하였다. 정지중인 무인운반차의 점유시간은 점유시작시간으로부터 무한대시간까지로 간주하였다.

링크의 점유는 각 링크의 점유시작시간과 점유종료시간을 링크점유시간표에 기록하는 방식으로 저장된다. 안전여유의 존재와 교점노드에서의 링크점유 등으로 인해, 한순간에 점유되는 링크의 수는 시스템에 존재하는 무인운반차의 대수보다 많을 수 있다. 양방향 궤도의 경우 물리적으로 동일한 링크가 2개 존재하므로, 2개 모두 동일한 시간 동안 점유되는 것으로 하였다. 링크점유시간표는 한 링크당 최고 8회의 해당 링크 점유를 기록할 수 있도록 하였으며, 시간의 측정은 IBM PC의 타이머 인터럽트로 처리하였고, 10초마다 한번씩 링크점유 상황을 재구성(Refresh)하도록 하였다.

(2) 스케쥴링과 최단시간경로의 생성

무인운반차의 출발노드와 종착노드가 정해졌을 때, 주행경로와 주행시간을 결정하는 것을 스케쥴링이라 한다. 본 논문에서는 제안된 관리제어기가 순환경로를 갖지 않는 최단시간경로로 스케쥴링하도록 하였다. 전절에서 제안한 경로시간의 개념은 경로의 총 주행시간을 나타내므로, 최단시간경로는 출발노드로부터 종착노드까지의 경로들 중 경로의 경로시간값 $\tau(P)$가 최소인 경로이다. 모든 링크 $l(i,j)$의 속도계수 C_{ij}가 사용자에 의해 부여되고 무인운반차의 최고주행속도 V와 가/감속시의 가속

도 a_{acc}/a_{dec}, 회전비용 γ_p이 상수로 주어지므로, $\tau(P)$는 경로상의 링크들의 길이 $d(i,j)$와 경로주행중의 정지시간 η_r에 의하여 결정된다. 따라서, 주어진 출발노드로부터 종착노드까지의 경로집합의 각 원소 P에 대하여 각각의 $\tau(P)$를 구함으로써 최단시간경로를 발견할 수 있다.

경로집합에서 어느 한 경로가 선택되었을 때, 이 경로에 대한 $\tau(P)$의 값은 주행중의 정지시간 η_r에 의하여 결정된다. 무인운반차의 경로상에 주행중인 다른 무인운반차가 존재할 때 무인운반차는 주행경로 도중의 한 지점에서 일시정지할 수 있으며, 이 경우 $\eta_r>0$ 이다. 관리제어기는 K개의 경로로 이루어진 경로집합의 한 경로 P를 경로후보(Candidate Path)로서 택하여, 주행시작시간을 기준으로 링크점유시간표를 참조하여 타 무인운반차와의 충돌여부를 검사한다. 경로후보의 각 링크의 점유예정시간을 경로시간 표현을 사용하여 계산하였으며, 하나 이상의 링크에서 점유시간의 중복이 발견될 경우를 충돌로 간주하였다.

충돌에는 두 가지의 유형이 존재한다[8]. 하나는 전방충돌(Head-on Conflict)로, 그림 5-a에서와 같이 한 링크에서 진행방향이 반대인 두 무인운반차가 만나는 경우이다. 다른 하나는 교차충돌(Cross Conflict)로, 이는 다시 교차로(Junction) 통과시의 충돌(그림 5-b)과 주행경로 중간에 다른 무인운반차가 정지중인 경우의 충돌(그림 5-c), 그리고 경로의 종착노드에 다른 무인운반차가 정지중인 경우의 충돌(그림 5-d)로 분류될 수 있다. 무인운반차 시스템에서는 이들 외에도 추월충돌(Catching-up Conflict)이 발생할 수 있으나, 본 시스템의 경우 링크를 통과할 때의 속도가 속도계수값에 의해 규정되므로 추월충돌은 발생하지 않는다.

각각의 충돌의 유형에 따라 일시정지노드의 위치와 η_r의 값이 달리 결정된다. 전방충돌의 경우, 경로후보가 노드들의 수열 ($p_1, p_2, p_3, \ldots, p_i, p_j, \ldots, p_m$)로 표현될 때 이와 충돌하는 무인운반차의 경로가 ($q_1, q_2, p_3, \ldots, p_i, q_n \ldots$) 이면, 일시정지노드는 최초로 충돌하는 노드의 전 노드인 p_2로 결정되며 일시정지시간 η_r은 링크점유시간표를 참조함으로써 계산된다. 교차로 통과시의 교차충돌의 경우 역시 같은 방법으로 계산된다. 주행경로 도중의 한 노드에 다른 무인운반차가 정지중인 경우에는 $\eta_r = \infty$인 r이 존재하며, 따라서 이 경로후보는 기각되고 대체경로가 선택된다. 한편, 주행경로의 종착노드에 다른 무인운반차가 정지중인 경우에는 경로탐색의 해가 존재하지 않는다.

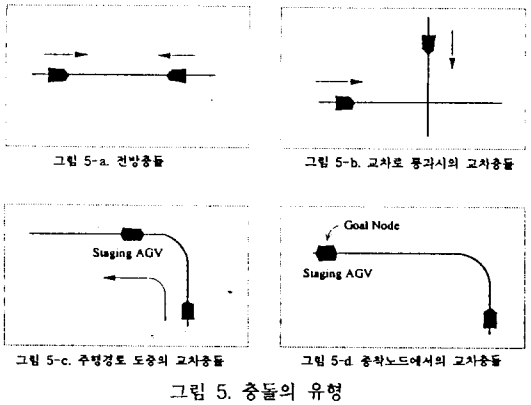

그림 5-a. 전방충돌
그림 5-b. 교차로 통과시의 교차충돌
그림 5-c. 주행경로 도중의 교차충돌
그림 5-d. 종착노드에서의 교차충돌
그림 5. 충돌의 유형

최단시간경로의 탐색은 경로집합의 K개 경로 중 $d(P)$가 최소인 경로 P_1으로부터 시작된다. 만일 이 최단거리경로의 총 일시정지시간이 $\sum_r \eta_r = 0$ 이면 최단거리경로는 최단시간경로와 동일한 경로를 가지며, 이 경로의 $\tau(P_1)$ 가 해당 경로집합 내에서 최소이므로 P_1는 최단시간경로이다. 만일 $\sum_r \eta_r > 0$ 이면, $d(P)$가 두번째로 작은 경로 P_2를 경로후보로 선정하여 이의 총 일시정지시간을 계산한다. P_2의 일시정지시간이 0인 경우는 $\min(\tau(P_1),\tau(P_2)) \le \tau(P_i), \forall i$ 이므로, $\min(\tau(P_1),\tau(P_2))$로써 최단시간경로를 구한다. P_2의 일시정지시간이 0보다 큰 경우에는 P_3에 대하여 위의 과정을 반복한다. 출발노드·종착노드가 결정되었을 때 이에 해당하는 경로집합의 원소의 수는 최대 K개이므로, 최단시간경로의 탐색은 이들 K개의 경로를 대상으로 최고 K회까지 이루어진다. 이 과정을 통해 얻어진 경로는 경로집합으로 주어진 K개의 경로들 중 최단시간경로이며, K가 충분히 큰 수이면 가능한 모든 경로들 중의 최단시간경로이다. 경로집합의 원소의 수를 유한으로 하는 이유는, 무한히 많은 경로를 탐색하지 않도록 함으로써 제어기법상의 안정성을 확보하기 위한 것이다.

한편, 일반적으로 4대 이상의 무인운반차가 격자형의 궤도에서 주행하는 시스템에서는 막힘현상이 발생하며, 이때 스케쥴링의 해가 명백히(Trivial) 존재함에도 불구하고 관리제어기는 해를 발견하지 못한다. 본 논문에서는 단순히 분선화된 경로 구조를 이용하는 것으로써 이를 해결하였다. 주행경로와 각 링크의 경유시간이 동시에 스케쥴링시에 결정되므로, 일단 스케쥴링이 이루어진 후에는 주행의 종료시점까지 다른 무인운반차의 주행으로 인한 방해를 받지 않으며 또한 방해를 주지 않는다. 따라서, 주행시의 막힘현상은 존재할 수 없다.

전 과정의 알고리즘 표현을 그림 6에 실었다.

(3) 무인운반차의 배차

다수대의 무인운반차와 다수개의 스테이션이 존재하는 시스템에서는, 무인운반차의 배차문제(Dispatching Problem)가 발생한다. 스테이션의 배차요구(Request)가 발생한 순간에 대기중(Ready)인 무인운반차가 유일하지 않은 경우에는 이들 대기중인 서버(Server)들 중 하나를 선택하여야 한다. 한편, 배차를 요구하는 스테이션이 다수개이고 배차가능한 무인운반차가 유일한 경우에는 어느 스테이션으로 배차할 것인지를 결정하여야 한다. 전자를 무인운반차의 선택(AGV Selection), 후자를 무인운반차의 파송(AGV Service)이라 한다[6]. 일반적으로, 이러한 배차문제에 적용되는 기법들은 해석적이라기보다는 발견적(Heuristic)이다[20].

무인운반차의 선택 규칙에 있어서는 최초로 대기상태에 들어간 서버의 선택, 가장 근접한 서버의 선택, 가장 오랜 시간 대기한 서버의 선택 등이 적용될 수 있다[8]. 그러나, 무인운반차 시스템의 생산성을 고려할 때는 오로지 최단시간에 스테이

션으로 도착할 수 있는 서버의 선택만이 의미를 갖는다. 본 논문에서는 경로시간 표현을 이용하여 이러한 서버를 선택할 수 있도록 하였다. 각 무인운반차 사이에 우선순위(Priority)의 차이가 없는 경우, 이러한 배차규칙으로 얻어지는 해는 배차문제의 최적해가 된다.

무인운반차의 파송 규칙에 있어서는 가장 먼저 일어난 배차요구의 선택(First-Come-First-Served), 가장 가까운 스테이션으로부터의 배차요구의 선택 등이 적용될 수 있다[8]. 여기서는 선착순 선택과 스테이션 우선순위에 의한 선택 규칙 중 사용자가 택일할 수 있도록 하였다.

5. 결 론

본 논문에서는 다수대의 무인운반차가 존재하는 시스템에서의 관리제어기 설계 기법을 제안하였다. 관리제어기의 실용화를 위한 조건으로 단순성과 유연성을 제시하였고, 이에 부응하는 관리제어환경 설계기법과 관리제어기법을 제안하였다. 제안된 관리제어환경은 시스템 궤도의 설계와 변경에 유연하게 대처할 수 있다. 제안된 관리제어기는, 무인운반차의 최적화된 스케쥴링과 배차를 단순한 제어기법을 사용하여 실시간에 수행할 수 있다.

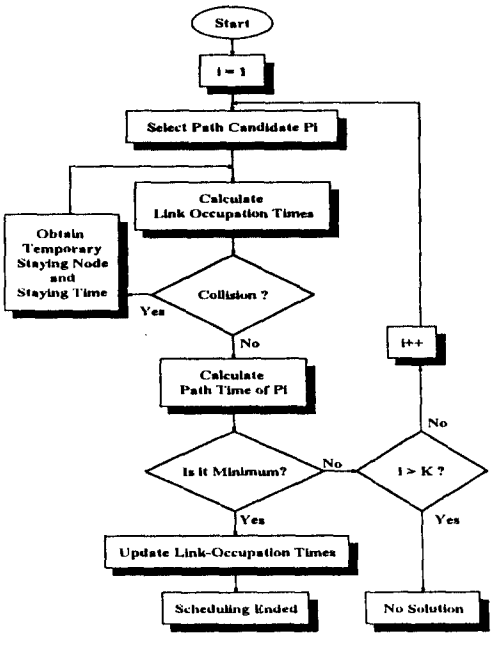

그림 6. 스케쥴링 알고리즘

< 참 고 문 헌 >

[1] C. S. Wang, "A Supervisory System for Free Ranging Automated Guided Vehicles," Imperial College, U. K., 1991.

[2] 神鋼電機株式會社 搬送機本部, "柔軟性でFMSに完全對應するコンピュータコントロール無人車システム," 無人搬送システム事例集, 流通研究社, 1992.

[3] 금성계전, 금성계전 AGV MINI-CART 사용설명서, 1989.

[4] "플렉시블 FA를 실현하는 무인반송 시스템," オートメイション紙, 日本 日刊工業新聞社, 월간 자동화기술 1991년10월호에서 재인용.

[5] AutoSimulations, *AUTOMOD II Rev. 1.1 Users Manual*, 1988.

[6] Deneb Robotics Incorporated, *QUEST Ver. 1.1 Users Manual*, 1993.

[7] A. Sen, "A Study of Free Ranging Automated Guided Vehicle Systems," Imperial College, U. K., 1990.

[8] S. P. Walker *et al.*, "Free-Ranging AGV and Scheduling System," *Automated Guided Vehicle Systems*, R. H. Hollier, ed., IFS(Publ.) Ltd., U. K., 1987.

[9] C. W. Kim, J. M. A. Tanchoco, "Conflict-free Shortest-time bidirectional AGV Routeing," Int. J. Prod. Res., Vol.29, No.12, 1991.

[10] P. J. Egbelu, J. M. A. Tanchoco, "Potentials for bi-directional guide-path for automated guided vehicle based systems," Int. J. Prod. Res., Vol.24, No.5, 1986.

[11] R. K. Miller, *Automated Guided Vehicles and Automated Manufacturing*, Society of Manufacturing Engineers, Michigan, 1987.

[12] O. Berman, M. R. Rahnama, "A Procedure For Dispatching Moving Mobile Servers," Networks, Vol.13, 1983.

[13] P. J. Egbelu, J. M. A. Tanchoco, "Characterization of automatic guided vehicle dispatching rules," Int. J. Prod. Res., Vol.22, No.3, 1984.

[14] M. A. Venkataramanan, K. A. Wilson, "A Branch-And-Bound Algorithm for Flow-Path Design of Automated Guided Vehicle Systems," Naval Research Logistics, 1991.

[15] G. F. Newell, *Traffic Flow on Transportation Networks*, MIT Press, 1980.

[16] D. R. Shier, "On Algorithms for Finding the K Shortest Paths in a Network," Networks, Vol.9, 1979.

[17] D. R. Shier, "Iterative Methods for Determining the K Shortest Paths in a Network," Networks, Vol.6, 1976.

[18] 강맹규, 네트워크와 알고리듬, 박영사, 1991.

[19] 박순달, 경영과학, 박영사, 1986.

[20] F. Taghaboni, J. M. A. Tanchoco, "A LISP-based controller for free-ranging automated guided vehicle systems,", Int. J. Prod. Res., Vol.26, No.2, 1988.