도로면 크랙실링 자동화 장비의 최적 경로계획 알고리즘 개발

Development of an Optimal Trajectory Planning Algorithm for Automated Pavement Crack Sealer

유 현 석* 이 정 호** 김 영 석***
Yoo, Hyun-Seok Lee, Jeong-Ho Kim, Young-Suk

요 약

도로면 크랙실링 공법은 예방적 차원에서 도로면에 발생된 크랙을 초기에 효과적으로 보수할 수 있는 공법으로 국내외에서는 1990년대 초반부터 기존 도로면 크랙실링 작업의 생산성, 안전성 및 품질의 균일성 확보를 목적으로 크랙실링 자동화 장비의 개발을 위한 지속적인 연구개발 노력을 수행해 왔다. 도로면 크랙실링 자동화 장비를 개발함에 있어 특히 경로계획은 주어진 작업 영역 내에서 개발 장비로 하여금 실링될 크랙 네트워크를 시간 · 효과적으로 횡단할 수 있도록 하는 운항 정보를 제공하게 되므로 이는 개발 장비의 성능을 결정 짓는 매우 중요한 연구주제라 할 수 있다. 본 연구의 목적은 작업 영역 내 경로계획 데이터의 효과적인 모델링을 통해 크랙실링 자동화 장비의 최적 경로계획 알고리즘을 개발하는 것으로써, 경로 집합 전체를 완전 탐색하는 2단계 트리 알고리즘과 크랙의 순열만을 탐색하는 1단계 트리 알고리즘을 개발하였으며, 알고리즘의 성능 측정 및 분석을 통해 최적 경로계획 알고리즘의 적용 범위와 그에 따른 성능 향상 정도를 평가하였다. 이 연구의 결과는 도로면 크랙실링 자동화 장비의 생산성 향상에 크게 기여할 수 있을 것으로 기대된다.

키워드 : 크랙 실링, 비전 알고리즘, 최적 경로 계획, 건설 자동화

1. 서론

1.1 연구의 배경 및 목적

도로면 크랙 실링(crack sealing) 공법은 도로면에 발생된 크랙을 초기에 효과적으로 보수할 수 있는 공법으로써, 크랙의 발전을 지연시켜 후속 크랙을 방지하고 방수기능으로 도로 하부구조를 보호하여 동결피해를 방지할 수 있는 예방적 차원의 보수공법이다(이정호 외 2004). 선진 외국에서는 크랙실링 공법의 이점 및 도로면 유지보수 공사의 위험요소를 인식하여 1990년대 초반부터 현재에 이르기까지 크랙실링 자동화 장비의 개발을 위한 연구를 진행하여 왔으며, 국내에서도 2001년부터 도로면 크랙실링 자동화 장비(APCS; Automated Pavement Crack Sealer)가 개발되어 2004년 프로토타입(prototype) 장비의 현장실험이 성공적으로 완료된 바 있다.

현재까지 개발된 크랙실링 자동화 장비의 머신비전 시스템(machine vision system)은 크랙 탐지 및 모델링 모듈과 경로계획 모듈로 구성되어 있는데, 이 가운데 크랙 탐지 및 모델링 모듈은 크랙 네트워크의 골격(spine)을 정확하게 탐지하고 모델링하기 위한 모듈로서 크랙실링 자동화 장비의 정확성과 품질을 좌우하는 중요한 모듈이다. 크랙 탐지 및 모델링 모듈은 지속적으로 인식률 개선을 위한 연구개발을 통해 균열 완전자동 인식률이 95.5%(유현석 외 2004. 11)에 이르렀으며, 이를 보완하기 위한 반자동화 매뉴얼 맵핑(manual mapping) 기능과 매뉴얼 에디팅(manual editing) 기능이 함께 구성되어, 어떠한 경우에도 크랙의 탐지 및 모델링이 가능하도록 설계되어 있다. 이에 반해 경로계획 알고리즘은 크랙실링 자동화 장비의 매니퓰레이터

* 일반회원, 인하대학교 건축공학과 박사과정, hsyoo.cm@inha.ac.kr
** 일반회원, 인하대학교 원가관리센터 선임연구원, 공학박사, inhacmr@hotmail.com
*** 종신회원, 인하대학교 건축공학과 교수(교신저자), youngsuk@inha.ac.kr

(manipulator)가 크랙 탐지 및 모델링 단계에서 디지털 좌표로 인식된 크랙 네트워크를 어떠한 순서로 이동하여 실링할 것인지를 결정하는 알고리즘으로서 크랙실링 자동화 장비의 속도와 생산성을 좌우하는 모듈이라 할 수 있다.

현재까지 크랙실링 자동화 장비의 경로계획은 Kim 과 Haas(1998)의 그리디 경로계획 알고리즘(greedy path planning algorithm)이 대표적인 경로계획 알고리즘이었다. 그리디 알고리즘은 구현이 간단할 뿐만 아니라 빠르게 동작하고, 매우 효과적인 결과(efficient path)를 도출하지만 알고리즘의 특성상 항상 최적의 결과(shortest path)를 보장하는 알고리즘은 될 수 없으며, 도로면 영상에 촬영된 크랙의 수가 증가할수록 최적 결과 값과의 오차(idle distance)가 점점 더 크게 나타나는 문제점이 있었다.

크랙실링 자동화 장비의 경로계획은 컴퓨터 공학의 그래프 이론(graph theory) 측면에서 "모든 경로를 탐색해 보기 전까지는 최적의 결과를 알 수 없는" 문제에 속한다. 이러한 문제는 크랙의 수가 적으면 해결이 가능하지만 크랙의 수가 증가할수록 경로의 수도 기하급수적으로 증가하기 때문에 막대한 시간 비용이 드는 문제로 분류된다. 그러나 실제 크랙실링 장비를 이용하여 현장 실험을 수행한 결과, 도로면 영상에 존재하는 크랙은 최대 8~9개 정도이며, 대부분의 경우 1~4개 정도의 크랙이 발견된다. 이러한 결과는 이론적으로 일정 범위 내에서 모든 경로에 대한 탐색이 가능하고, 항상 최적의 결과를 보장하는 최적 경로계획 알고리즘의 개발이 가능함을 의미한다. 또한 최근 컴퓨팅 기술의 발전에 힘입어 최적 경로계획 알고리즘이 소화할 수 있는 범위도 더 증가하고 있다.

본 연구의 목적은 효과적인 경로계획 데이터 모델링을 통해 크랙실링 자동화 장비의 최적 경로계획 알고리즘을 개발하는 것이며, 개발된 알고리즘의 성능 측정 및 분석을 통해 최적 알고리즘의 적용 범위와 기존 알고리즘 대비 성능 향상 정도를 평가하는 것이다. 이 연구의 결과는 크랙실링 자동화 장비의 생산성을 한층 증진시킬 수 있을 것으로 기대된다.

1.2 연구의 범위 및 방법

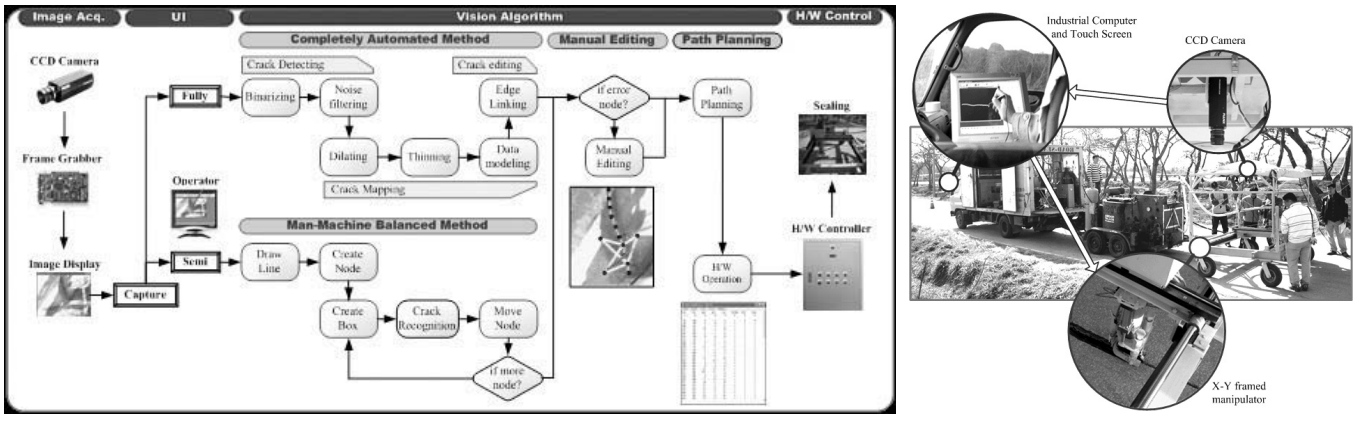
본 연구의 범위는 크랙 실링 자동화 장비의 제어를 위한 머신비전 알고리즘 가운데 기존 경로계획 알고리즘의 문제점을 분석하고 새로운 경로계획 알고리즘을 제안하여 성능을 비교 · 검토하는 것으로서, 연구의 방법은 다음과 같다.

1) 비전 알고리즘의 개념과 프로세스를 설명하고 비전 알고리즘 내에서 경로계획 알고리즘의 역할과 중요성을 분석한다.
2) 선행 개발된 경로계획 알고리즘의 성능과 문제점을 분석한다.
3) 이미지 영상 내에 존재하는 크랙 네트워크에 대한 효과적인 경로 모델링 방법과 데이터 구조(data structure)를 구현한다.
4) 기존 그리디 알고리즘과 제안된 알고리즘을 프로그래밍 언어로 각각 구현하여 성능 및 장단점을 분석하고 크랙실링 자동화 장비에서의 효과적인 적용방법을 제시한다.

2. 경로계획 알고리즘의 역할과 선행 알고리즘의 고찰

2.1 머신비전 알고리즘의 구성과 경로계획 알고리즘의 역할

선행 연구에서 비전 알고리즘은 그림 1의 ①과 같이 영상획득에서부터 노이즈 제거, 크랙탐지 및 맵핑, 경로계획, 크랙 실링까지 5단계로 이루어져 있다. 첫 번째 단계인 영상 획득 단계는 CCD 카메라를 크랙실링 장비의 상단에 설치하여 도로면의 영상을 획득하는 단계로서, 640×480 해상도의 8bit Gray 비트맵(bitmap) 영상을 사용한다. CCD 카메라에 의해 촬영되는 아날



① 머신 비전 알고리즘의 구성　　② 하드웨어 구성

그림 1. 크랙실링 자동화 장비(APCS)의 구성

로그 영상은 프레임 그래버(frame grabber)를 통해 디지털로 변환되어 메모리에 저장되며, 사용자의 초기 영역 설정에 따라 영상 처리(image processing)가 이루어진다.

두 번째 단계는 크랙 탐지(crack detecting) 단계로서, CCD 카메라에서 획득된 도로면 영상을 이진화하여 영상에서 크랙을 구분하고 지능적으로 노이즈를 제거하는 프로세스를 말한다. 선행 연구(유현석, 2004)에서는 신경망 학습 기법을 이용하여 성공적으로 도로면 영상에서 노이즈 객체를 제거하였다.

세 번째 단계인 크랙 맵핑 알고리즘은 자동화 장비의 실런트 분사장치가 정확히 크랙의 중심을 따라 이동하게 하기위한 알고리즘으로서 완전자동화(full automatic mapping) 방식과 반자동화 방식(manual mapping)으로 나눌 수 있다. 완전자동화 방식은 노이즈가 완전히 제거된 상태에서 세선화 알고리즘(thinning algorithm)을 통해 크랙의 골격(skeleton)만을 남기는 방식이다. 이 알고리즘은 노이즈가 완전히 제거되었을 때만 정상적으로 작동한다는 단점이 있으나 반자동화 방식에 속하는 매뉴얼 맵핑 방식은 노이즈 잔존이나 크랙의 손상 여부에 크게 영향을 받지 않고 비교적 정확한 중심선을 추출할 수 있는 장점이 있다.

네 번째 단계인 경로계획은 맵핑과정을 통해 추출된 여러 개의 크랙에 대하여 자동화 장비의 실런트 분사장치의 이동순서를 정하는 알고리즘이다. 실런트 분사장치는 원점(좌상단 끝점)을 기준으로 영상 내의 크랙 중심선을 따라 한 번씩 이동하게 되는데, 만약 여러 개의 크랙이 존재한다면 크랙의 실링 순서에 따라 크랙간 이동거리(idle distance)는 상당한 차이를 보인다. 경로계획 알고리즘의 목적은 수많은 크랙 네트워크 중 크랙 간 이동거리가 가장 짧은 네트워크를 탐지하는 것이다.

경로계획을 마친 후 프로그램은 크랙 맵핑 데이터와 경로 데이터를 조합하여 텍스트 형태의 하드웨어 명령어를 출력한다. 하드웨어 명령어는 프로토콜(protocol)을 통해 장비의 컨트롤 드라이버와 통신하게 되며 실런트 분사장치는 하드웨어 명령어에 따라 작동한다.

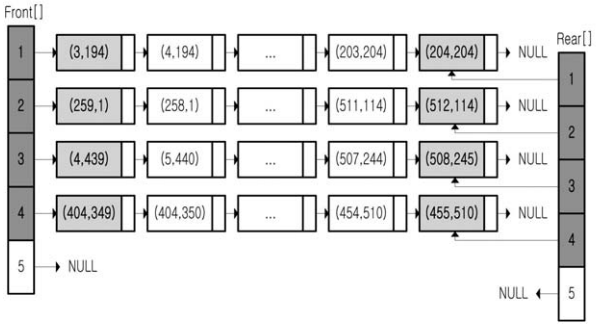


그림 2. 크랙 네트워크 맵핑 데이터의 저장구조

경로에 대한 맵핑 데이터는 일반적으로 배열(array)형태나 연결 리스트(linked list)와 같은 데이터 구조를 이용한다. 맵핑 데이터 가운데 경로 계획에서 필요한 데이터는 경로수(path number, 이하 N)와 각 경로의 양 끝점(front, rear)의 X, Y 좌표이다. 그림 2는 연결 리스트로 저장된 맵핑 데이터의 구조이다. 이 논문에서는 배열과 연결리스트가 혼합된 구조를 사용하였다. 각 경로의 처음점(front[])과 끝점(rear[])은 배열 포인터(array pointer)에 의해 참조되고, 내부의 경로데이터는 스트링(string) 형태의 연결리스트 구조로 연결되어 있다. 그림 2에서 각 경로의 처음점과 끝점의 좌표가 경로계획에서 사용되는 데이터이다. 큐(queue)형태로 저장된 n개의 경로 데이터를 기준으로 경로계획에 필요한 정보는 각 경로의 시작점과 종점의 (x, y) 좌표이다. 이 좌표는 큐의 양쪽 배열 포인터가 가리키는 값이 된다. 예를 들어 그림 2에서 경로 3의 시작점은 front[3]→(4, 439), 종점은 rear[3]→(508, 245)의 좌표를 가지게 된다. 크랙 영상에서 흔히 발견되는 나뭇가지 형상의 크랙은 스트링 형태의 선형 데이터로 각각 분할되어 저장되어야 한다. 경로계획 알고리즘은 다음과 같은 원칙이 있다.

① 경로계획의 시작점은 영상의 좌상단 끝점에 위치하고 좌표는 (0,0)이다. 시작점은 자동화 장비의 실런트 분사장치가 작업명령을 대기하는 위치이다.

② 모든 크랙 객체는 한 번씩 실링된다. 다시 말해, 한번 실링된 크랙 객체는 다음 이동경로 대상에서 제외된다.

③ 어떤 경로계획이든 크랙의 실링거리의 합은 같으므로 크랙간 이동거리(idle distance)가 가장 짧은 경로계획이 가장 우수한 경로계획이다.

④ 모든 크랙 객체에 대하여 실링을 종료한 직후 자동화 장치가 다음 작업영역으로 이동하기 때문에 분사장치가 원점으로 복귀하는 거리는 크랙간 이동거리에 포함하지 않는다.

경로계획 알고리즘은 자동화 장비의 작업속도 및 생산성과 직접적인 연관을 가진다. 자동화 장비의 실런트 분사장치의 이동속도가 평균 150mm/sec, 작업 영역이 2m×1.5m, 영상영역이 640×480 화소라고 할 때, 하나의 화소당 크기는 3.125mm이며, 최적의 경로계획과 최악의 경로계획의 이동거리차가 평균 800화소라고 한다면 실제 실런트 분사장치의 이동거리의 차는 2,500mm이며, 1번 실링할 때마다 16.6sec의 작업시간이 지연되는 결과를 초래한다. 국내업체의 1일 평균 크랙실링 작업량이 1.4km 정도임을 감안할 경우 경로계획 알고리즘만으로 일일 최대 2시간의 생산성 차이가 발생하게 된다.

2.2 기존 경로계획 알고리즘과 문제점

2.2.1 그리디 경로계획 알고리즘

Kim과 Haas(1998)의 그리디 경로계획 알고리즘은 작업 영역 내에 다양한 크랙 네트워크가 존재할 경우 기준점에서 가장 가까운 크랙 네트워크를 순차적으로 방문하도록 경로계획을 수립하는 방식이다. 그리디 경로계획 알고리즘의 프로세스는 다음과 같다.

① 방문하지 않은 크랙 중에서 가장 가까운 끝점을 탐색한다.

② 가장 가까운 끝점으로 매니퓰레이터를 이동하여 반대쪽 끝점까지 크랙을 실링하고, 작업이 완료되면 해당 크랙을 방문한 크랙으로 기록한다.

③ 매니퓰레이터가 이동한 반대쪽 끝점을 탐색 기준점으로 설정한다.

④ 탐색 기준점에서 방문하지 않은 크랙이 있는지 확인하고, 방문하지 않은 크랙이 있으면 단계 1로 복귀한다. 더 이상 방문하지 않은 크랙이 없으면 매니퓰레이터를 홈포인트로 이동하고 크랙실링을 종료한다.

그리디 경로계획 알고리즘에서 첫 기준점은 원점이 되고 가장 가까운 크랙 네트워크의 한점으로 이동하면 다음 기준점은 해당 크랙 네트워크의 반대쪽 끝점이 된다. 그리디 알고리즘은 이 같은 과정을 반복하여 수행함으로써 크랙간의 경로를 계획하는 방법을 말한다. 다음 그림 3은 4개의 크랙 네트워크가 탐지된 영상으로써, 그리디 알고리즘의 단계별 진행 과정은 다음과 같다.

① 원점을 기준으로 4개의 크랙 네트워크의 양 끝점, 즉 8개의 끝점까지의 거리를 계산하여 가장 가까운 끝점 0F로 단말장치를 이동시킨다.

② 0F에서 0R까지 크랙을 실링하고 0R을 기준점으로 정한 다음, 나머지 점들까지의 거리를 측정하여 가장 가까운 끝점 1F로 말단장치를 이동시킨다.

③ 1F에서 1R까지 크랙을 실링하고 1R를 기준점으로 정한 다음, 나머지 점들까지의 거리를 측정하여 가장 가까운 끝점 2R로 말단장치를 이동시키다.

④ 2R에서 2F까지 크랙을 실링하고 2F를 기준점으로 정한 다음, 나머지 점들까지의 거리를 측정하여 가장 가까운 끝점 3F로 말단장치를 이동시킨다.

⑤ 3F에서 3R까지 크랙을 실링하고 말단장치를 원점으로 복귀한다.

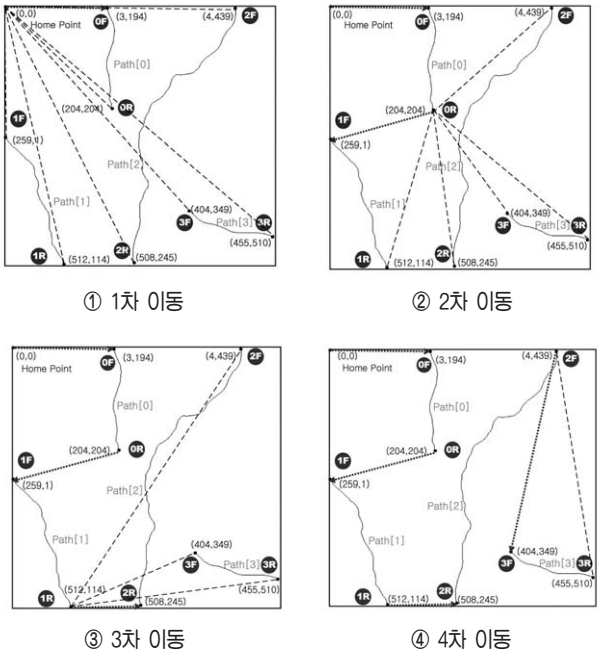


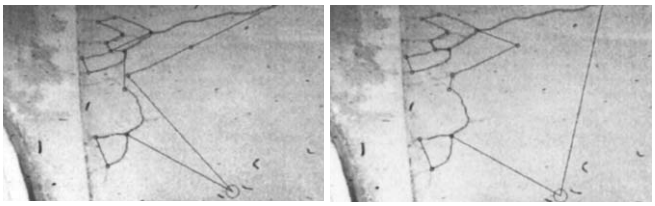
그림 3. 그리디 알고리즘의 진행과정

Kim과 Haas(1998)의 그리디 경로계획 알고리즘은 매우 빠른 시간 안에 좋은 경로(good path)를 찾을 수 있는 장점이 있다. 실제 한 장의 영상 내에 10개 이상의 크랙이 있더라도 계산에 소요되는 시간은 3ms 내외 정도로 가장 간단하고 빠른 속도를 가진 알고리즘이다. 그러나 그림 3과 같이 계산된 그리디 경로계획 알고리즘의 결과는 최단거리의 이동경로와는 상당한 차이가 있다. 그리디 알고리즘으로 계산된 이동거리는 945 화소인데 반해 실제 최단 경로(Home→1F→2R→0F→3F)의 이동거리는 882 화소로서 63화소(7.1%)만큼의 차이가 있다. 또한 그리디 알고리즘은 크랙의 수 n이 증가할수록 이러한 오차는 점점 커지는 경향을 보인다.

2.2.2 Simulated Annealing을 이용한 경로계획

Mathurin과 Velinsky(2000)는 기존 그리디 알고리즘의 문제점을 개선하기 위해 메타 휴리스틱(meta-heuristic) 알고리즘인 Simulated Annealing 기법(SAa)을 이용한 경로계획 알고리즘을 개발하였다. 이 알고리즘은 크랙 경로의 순열(permutation)을 생성하고 이웃 경로를 랜덤하게 선택한 다음, 확률 함수를 이용하여 더 짧은 경로를 찾아내는 알고리즘이다. 이 알고리즘은 대량의 계산을 피하기 위해 볼쯔만 확률 분포(Boltzmann probability distribution)를 사용하며 온도 감소함수인 알파(α) 팩터로 0.88을 사용하였다. Feng 외(2005)의 논문

에 소개된 바에 의하면 Simulated Annealing 기법을 이용한 경로계획 알고리즘이 항상 최단 경로계획을 보장하지는 않지만 다음 그림 4의 경우와 같이 기존 그리디 알고리즘보다 최대 15% 이상 오차(idle distance)를 줄일 수 있는 것으로 알려져 있다.



① 그리디 알고리즘 ② Simulated Annealing

그림 4. Simulated Annealing 결과(Feng 외 2005)

3. 경로계획 알고리즘의 설계

3.1 최단 경로계획 알고리즘의 고찰

크랙실링 장비의 경로계획은 순환 외판원 문제(TSP; Traveling Salesman Problem, 이하 TSP)와 밀접한 관련이 있다. TSP는 알고리즘 학문에 있어서 대표적인 NP-Class[1] (nondeterministic polynomial time complete) 문제로서 그림 5와 같이 외판원이 n개의 지점을 모두 방문(중복으로 방문하지 않는다)한 뒤 처음 출발지로 돌아오기까지 소요되는 비용이 가장 적은 경로를 찾아내는 문제를 말하며, 최단거리를 보장하는 경로를 찾기 위해서는 모든 경로를 탐색해 보는 방법 이외에는 해결방법이 없는 문제를 말한다.

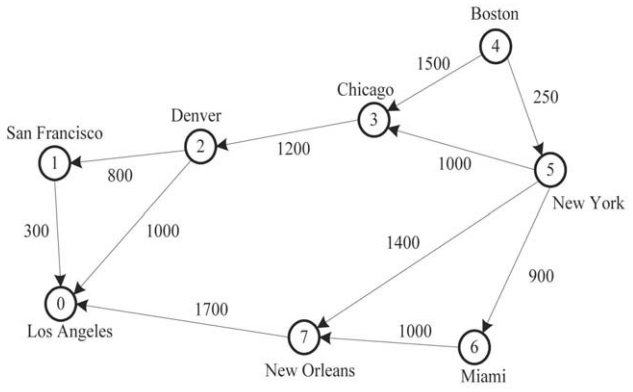


그림 5. 순환외판원 문제(Horowitz 외 1993)

1) 1971년 스테판 쿡(Stephen Cook)은 시간 안에 풀 수 있는 문제들의 집합을 P-Class라고 하고, 시간 안에 풀 수 없는 문제의 집합을 Not-P Class라고 할 때, 해결책을 찾아내는데 아무리 많은 시간이 소요되더라도 시간 안에 풀 수 있는 해결책이 있고, 이를 검증하는 것이 가능한 속성을 가진 부류의 문제를 NP-Class라고 정의하였다. (Decker, 1996)

TSP는 n개의 지점을 모두 방문해야 하므로 시간 복잡도[2]는 O(n!)이 되며, n의 값이 작을 때는 프로시저의 동작이 완료될 때까지 많은 시간이 걸리지 않지만 n값이 증가하면 할수록 이전 단계의 n배가 소요되므로 n값이 어느 정도 커지면 엄청난 시간이 걸리게 된다. 예를 들어 지점의 개수가 3개일 때 걸리는 시간을 1ms라고 한다면 20개 지점의 최단 이동거리를 찾기 위해서는 무려 684만년이 걸리게 되며, 이러한 문제 때문에 TSP는 NP-Class 문제로 분류된다.

앞서 언급한 바와 같이 TSP는 모든 경로를 검색하여 그 경로를 검색한 뒤 최적의 경로를 찾는 방법 외에는 개발된 알고리즘이 없다. 방문지의 수 n의 값이 클 경우 TSP의 시간비용이 매우 크기 때문에 흔히 Dijkstra(1959), Kruskal(1956), Prim(1957), Sollin(1962)과 같은 그리디 알고리즘(greedy algorithm)을 사용한다. 그리디 알고리즘이란 여행 경로를 완성하기 위하여 방문하지 않은 정점 중 최소 비용의 가중치를 갖는 것을 다음번 간선으로 선택하는 방법을 말하는데, 그리디 알고리즘은 상당히 좋은(good) 해결책을 찾을 수 있지만 반드시 최적(optimal)의 경로를 찾을 수 있는 것은 아니며, 특히 n이 커질수록 최적 경로를 찾을 수 있는 확률은 점점 더 낮아지게 된다.

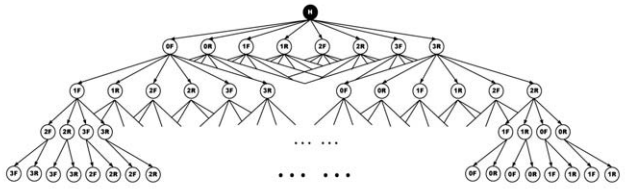


그림 6. 크랙실링 자동화 장비의 경로계획

크랙실링 자동화 장비의 경로계획 알고리즘은 실런트 분사장치가 크랙으로 이동하고 크랙을 실링하는 경로 순서를 계획하는 것을 말한다. 영상에서 존재하는 크랙의 수를 n이라고 할 때, 가능한 경로계획의 수는 $2^n \times n!$개가 된다. 다시 말해 영상에 8개의 크랙이 존재할 경우 10,321,920개의 경로계획이 가능하다. 최적의 경로계획이라 함은 이 가운데 이동거리가 가장 짧은 경로계획을 의미하며, 반드시 하나 이상의 최적 경로계획이 존재한다.

크랙실링 장비의 경로계획은 TSP 문제의 경로계획과 상당히

2) 점근 표기법(asymptotic notation)은 어떤 함수의 증가율을 다른 함수와의 비교로 표현하는 방법이다. 알고리즘의 시간복잡도를 단순화할 때나 무한급수의 뒷부분을 간소화할 때 쓰이며, 소프트웨어 분야에서는 함수의 시간 복잡도를 정의하는데 있어 최악의 경우일 때의 시간복잡도(worst-case computing time)를 표현하는 big O notation이 널리 사용된다. 실제 알고리즘의 연산시간 측정은 일반적으로 알고리즘이 구현된 시스템에서 지원하는 Clock()함수를 이용하여 millisecond 단위로 측정한다.

유사하며 한 단계 더 복잡한 문제이다. TSP 문제가 O(n!)의 시간 복잡도를 가지는데 반해 크랙실링 장비의 경로계획은 O(n!×2^n)의 시간 복잡도를 가진다. 순환 외판원의 문제는 단순히 지점을 방문하는 문제이지만 크랙실링 장비에서의 하나의 크랙(n)에 대하여 두 개의 끝점(vertex)을 가지기 때문에 TSP에 비해 2^n배 만큼 더 복잡한 문제가 된다. 그림 6은 영상 내에서 4개의 크랙이 존재할 때를 직관적인 트리구조로 모델링한 것으로 총 384개의 경로가 존재하게 됨을 알 수 있다. 그러나 그림 6과 같은 트리 구조 데이터 모델은 크랙의 수 n이 증가함에 따라 트리의 크기가 기하급수적으로 증가한다는 단점이 있다. 트리의 크기는 트리의 모든 노드의 개수(TN)에 노드의 크기(8 byte)를 곱한 값으로 표현할 수 있고 n에 따른 TN은 다음 식 1과 같다. 이에 따라 전체 트리를 모델링할 경우 크랙의 수 n이 8이라면 전체 트리의 크기는 136Mbytes, n이 9라면 전체 트리의 크기는 2.4 Gbytes에 달하게 되고, 이는 일반적인 컴퓨터의 메모리 용량을 초과하는 결과가 된다.

$$\begin{aligned} TN &= 1+2n+2n\times 2(n-1)+\ldots+2^n\times n! \\ &= \frac{2^n\times n!}{2^n\times n!}+\frac{2^n\times n!}{2^{n-1}\times(n-1)!}+\ldots+\frac{2^n\times n!}{1} \\ &= \sum_{i=0}^{n}\frac{2^n\times n!}{2^i\times i!} \end{aligned} \qquad \text{(식 1)}$$

크랙실링 장비의 경로계획은 TSP와 마찬가지로 NP-Class의 문제이며, 모든 경로를 방문해야만 최적의 경로를 알 수 있는 문제이다. 그러나 크랙실링 장비 경로계획의 시간 복잡도가 O(n!×2^n)이라면 이 문제를 O(n!)의 문제와 O(2^n)의 문제로 구분하여 생각하는 것이 문제 해결의 실마리가 될 수 있다. 다시 말해, 먼저 어떤 순서로 크랙을 방문할 것인가(O(n!))하는 문제와 각 크랙의 어떤 점으로 먼저 진입할 것인가(O(2^n))하는 문제로 구분하여 경로의 수를 생각할 수 있으며, 이는 크랙실링 장비 경로계획을 효과적으로 풀어나가기 위한 중요한 단서가 된다.

3.2 최단경로 탐색을 위한 2단계 트리 알고리즘

최적 경로계획 알고리즘은 원점(0,0)을 기준으로 n개의 경로를 지나가는 모든 경우의 수를 데이터 구조(data structure)로 모델링하고 연산을 통해 크랙 간 이동 거리(idle distance)가 가장 짧은 경로를 선택하는 방법이다. 본 연구에서는 모든 경로의 경우의 수를 모델링하기 위한 방법으로는 2단계의 트리 구조를 사용한다. 원점을 기준으로 모든 경로를 한 번씩 지나가는 순서의 집합을 경로집합(표 1 참조)이라고 한다면, 1단계 트리는 크랙의 양 끝점(front[rear])을 고려하지 않고 경로집합 내의 각 경로의 순서에 대한 경우의 수를 결정한다. 2단계 트리는 1단계 트리에서 구해진 경로 순서에서 각 경로의 양 끝점(front[rear]) 중 진입할 점의 순서에 대한 경우의 수를 결정한다. 이러한 두 단계의 트리 구조로부터 n개의 경로에 대한 모든 경우의 수를 구할 수 있으며, 이 후 3단계로 크랙 경로 간의 이동거리를 연산하게 된다.

1단계 트리는 크랙의 개수(n)를 입력 값으로 한다. 예를 들어 그림 7에서 경로의 개수는 4개이다. 이 입력값을 기준으로 원점(0)을 포함한 5개의 원소(경로)를 가지는 경로집합({0, 1, 2, 3, 4})을 생성한다. 1단계 트리의 목적은 경로집합 내 원소들의 순서에 대한 순열 조합을 구하는 것이다. 먼저, 원점에서 특정 경로를 선택하여 이동하는 과정을 트리 구조로 모델링하면 그림 7 (a)와 같이 원점(0)은 최상위 부모 노드(parent node, root node)로 표현하고 원점에서 이동 가능한 4개의 경로는 자식 노드(child node)로 표현한다. 다시 말해 출발점이 부모 노드이고 이동 가능 경로를 자식 노드로 표현한다.

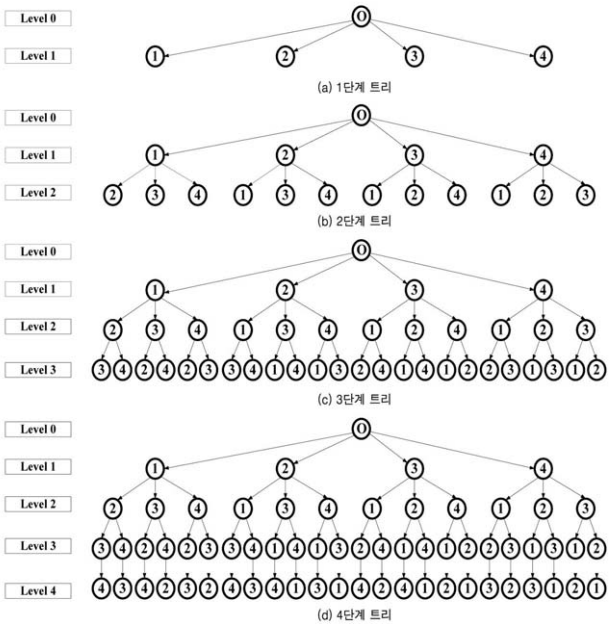


그림 7. 1단계 경로 트리

Level 2에서는 Level 1의 4개 노드가 부모가 되고, 각 노드에 Level 1에서 지나가지 않은 3개의 경로가 자식 노드로 표현된다. 여기에서 특정 노드의 자식 노드는 부모 노드의 형제 노드(sibling node)와 같음을 알 수 있다. 예들 들면 그림5 (b) Level 1 경로 1 노드를 부모로 하는 자식 노드는 원점과 경로 1을 제외한 나머지 경로 2, 경로 3, 경로 4이다. 마찬가지로 Level 1에서 경로1의 형제 노드는 경로2, 경로3, 경로4이다. 이와 같은 반복

적(iterative)인 방법으로 Level 3과 Level 4에서도 같은 방법으로 각 노드에 대한 트리를 구성을 할 수 있다. 1단계 트리의 구성이 완료되면 각각의 경로를 순회(traverse)하여 경로집합의 결과를 얻을 수 있으며, 첫 번째 트리의 경로집합 결과를 [경로순열]^{1} 형식으로 표 1에 나타내었다.

그림 7과 표 1의 결과를 바탕으로 크랙 경로의 순서에 대한 경우의 수를 일반화시켜 보면 표 2에서 볼 수 있듯이 n개의 경로를 원소로 하는 경로집합의 개수는 n!이 되고, 말단 노드(terminal node)의 개수와 같다.

표 1. 1단계 경로 트리의 경로집합

순열	경로집합^{1}
[1]^{1}	0 → 1 → 2 → 3 → 4
[2]^{1}	0 → 1 → 2 → 4 → 3
[3]^{1}	0 → 1 → 3 → 2 → 4
[4]^{1}	0 → 1 → 3 → 4 → 2
.....	
[21]^{1}	0 → 4 → 2 → 1 → 3
[22]^{1}	0 → 4 → 2 → 3 → 1
[23]^{1}	0 → 4 → 3 → 1 → 2
[24]^{1}	0 → 4 → 3 → 2 → 1

표 2 단계별 노드의 개수

Level(L)	자식노드의 개수(C)	노드의 개수(T)
0	n	1
1	n−1	n
2	n−2	n×(n−1)
3	n−3	n×(n−1)×(n−2)
...		
n−2	2	n×(n−1)×(n−2)×...×3
n−1	1	n×(n−1)×(n−2)×...×3×2
n	0	n×(n−1)×(n−2)×...×3×2×1 = n!
	C = n−L	T = n! / C! (C가 0일때 제외)

2단계 트리는 1단계 트리에서 얻어진 경로집합을 입력값으로 한다. 예를 들어 3번째 경로집합([3]^{1})의 경로(원소)는 {0, 1, 3, 2, 4}이다. 경로집합 내의 각 경로는 두 개의 끝점(front[n]와 rear[n], 이하 F[n]와 R[n])을 가지고 있다. 2단계 트리의 목적은 1단계 트리에서 주어진 경로집합 내 경로의 양 끝점에 대한 진입

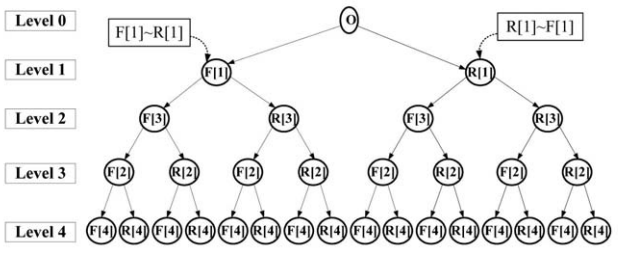


그림 8. 2단계 경로 트리

점을 결정하는 것이다.

1단계 트리에서 구해진 경로집합[3]^{1} = {0, 1, 3, 2, 4}를 기준으로 2단계 트리를 구성하면 그림 8과 같이 완전이진 트리(complete binary tree)의 형태를 가진다. 2단계 트리에서 각 노드는 진입점을 의미한다. 예를 들어 노드 F[1]은 그림 8에 점선으로 표시된 부분(F[1]~R[1])에서 알 수 있듯이 F[1]으로 진입해서 크랙 중심선을 따라 이동하고 R[1]에서 멈추는 것을 의미한다. 2단계 트리의 경로집합 결과 값을 [경로순열]^{1}[경로순열]^{2} 형식으로 표 3에 나타내었다.

표 3. 2단계 경로 트리의 경로집합

순열	경로집합^{2}
[3]^{1}[1]^{2}	0 → F[1] → F[3] → F[2] → F[4]
[3]^{1}[2]^{2}	0 → F[1] → F[3] → F[2] → R[4]
[3]^{1}[3]^{2}	0 → F[1] → F[3] → R[2] → F[4]
[3]^{1}[4]^{2}	0 → F[1] → F[3] → R[2] → R[4]
(생략)	
[3]^{1}[13]^{2}	0 → R[1] → R[3] → F[2] → F[4]
[3]^{1}[14]^{2}	0 → R[1] → R[3] → F[2] → R[4]
[3]^{1}[15]^{2}	0 → R[1] → R[3] → R[2] → F[4]
[3]^{1}[16]^{2}	0 → R[1] → R[3] → R[2] → R[4]

그림 8과 표 3의 결과를 바탕으로 진입점에 대한 경로의 순서에 대한 경우의 수를 일반화시켜 나타내면 표 4와 같이 진입점의 순서를 고려한 하나의 경로집합에서 발생할 수 있는 경우의 수는 2n이 되고, 1단계 트리와 같이 말단 노드의 개수와 같다. 이는 1단계 트리에서 24개의 경로집합 중 하나{0, 1, 3, 2, 4}에 대한 개수이므로 1단계의 모든 경로집합에 대한 경우의 수, 즉 n개의 경로에 대한 총 경로집합의 수는 $n!\times 2^{n}$개가 된다.

표 4. 단계별 노드의 개수

Level(L)	자식노드의 개수(C)	총 노드의 개수(T)
0	2	$1 = 2^{0}$
1	2	$2 = 2^{1}$
2	2	$4 = 2^{2}$
3	2	$8 = 2^{3}$
...		
n−2	2	2^{n-2}
n−1	2	2^{n-1}
n	2	2^{n}
	C = 2	$T = 2^{L}$

이동거리의 연산은 2단계 트리의 경로집합을 입력값으로 한다. 예를 들어, 표 3에서 경로집합[3]^{1}[14]^{2}는

경로집합[3]^{1}[14]^{2}
= {0 → R[1] → R[3] → F[2] → R[4]}
= {0 → R[1]~F[1] → R[3]~F[3] →F[2]~R[2] → R[4]~F[4] }

으로 나타낼 수 있고, (→) 구간이 非실링 거리이므로 크랙간 총 이동거리는 다음과 같다.

총 이동거리
= 원점(0)과 R[1] 사이의 직선거리
+ F[1]과 R[3] 사이의 직선거리
+ F[3]과 F[2] 사이의 직선거리
+ R[2]와 R[4] 사이의 직선거리

실제 프로그래밍 단계에서는 그림 9와 같이 연산 속도를 높이기 위하여 두 점 사이의 직선거리에 대한 값들을 미리 계산하여 2차원 배열에 저장하고 연산 과정에서 해당 값들을 불러서 사용하게 된다. 예를 들면 표 5에서 F[1]과 R[3] 사이의 거리를 계산할 때 배열 Distance[1][6]을 참조하여 287.1이라는 값을 얻는 것이 훨씬 빠른 연산속도를 보인다.

표 5. 두 지점간의 거리에 대한 비용인접 행렬

vertex		0	F[1]	R[1]	F[2]	R[2]	F[3]	R[3]	F[4]	R[4]
	dist.	[0][0]	[1][0]	[2][0]	[3][0]	[4][0]	[5][0]	[6][0]	[7][0]	[8][0]
0	[0][0]	0.0	446.0	341.2	340.4	575.2	245.1	339.1	248.0	529.7
F[1]	[1][0]	446.0	0.0	286.6	288.6	536.4	201.0	287.1	505.4	585.4
R[1]	[2][0]	341.2	286.6	0.0	2.2	288.1	216.5	2.2	257.4	307.4
F[2]	[3][0]	340.4	288.6	2.2	0.0	287.2	217.4	2.0	255.3	305.7
R[2]	[4][0]	575.2	536.4	288.1	287.2	0.0	504.5	289.2	371.2	118.0
F[3]	[5][0]	245.0	201.0	216.5	217.4	504.5	0.0	215.4	342.2	511.9
R[3]	[6][0]	339.1	287.1	2.2	2.0	289.2	215.4	0.0	255.5	307.6
F[4]	[7][0]	248.0	505.4	257.4	255.3	371.2	342.2	255.5	0.0	298.2
R[4]	[8][0]	529.7	585.4	307.4	305.7	118.0	511.9	307.6	298.2	0.0

경로집합[3]^1[14]^{2}에 대한 총 이동거리는 341.2+ 287.1+ 217.4+118.0=963.7(화소)가 된다. 이 결과는 총 384개의 결과값 중 42번째 결과 값이다. 최적경로 탐색 알고리즘은 n!×2n 개의 경로집합에 대해서 각각의 이동거리를 모두 계산하고 이 중 가장 작은 값을 가지는 집합을 선택한다. 실제로 가장 작은 값을 가지는 경로는 경로집합[21]^1[6]^2={0→F[4]→R[2]→R[1]→F[3]}이고 이동거리는 569.2(화소)이다. 2단계 트리 알고리즘은 모든 경로의 이동거리를 계산하기 때문에 항상 최단 경로를 보장한다.

2단계 트리 알고리즘의 실제 데이터 구조 구현모델은 그림 9와 같이 역방향 포인터를 가진 트리 구조이고, 말단 노드 하단에 별도의 포인터 큐를 생성하여 경로 데이터를 하나씩 로드할 수 있도록 설계되었다. 이는 상향으로 경로 데이터를 로드하는 방법이 하향 순회 방식보다 속도가 빠르고 구현이 간단하기 때문이다. 다음 그림 10은 2단계 경로계획 알고리즘의 순서도이고, 그림 11은 2단계 트리 알고리즘을 크랙의 수 n이 4인 임의의 영상에 대한 실행 결과를 텍스트 파일로 출력한 것으로써 원점이 H, 크랙 번호가 0~3으로 표현되어 있다.

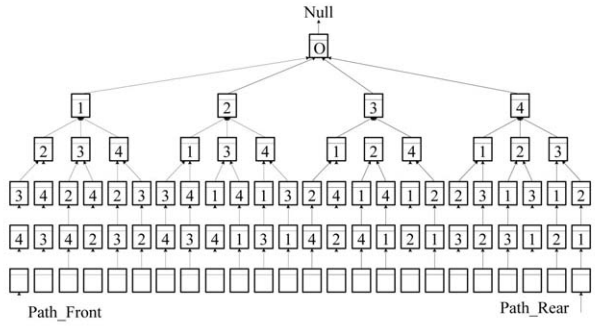


그림 9. 데이터 구조 구현모델

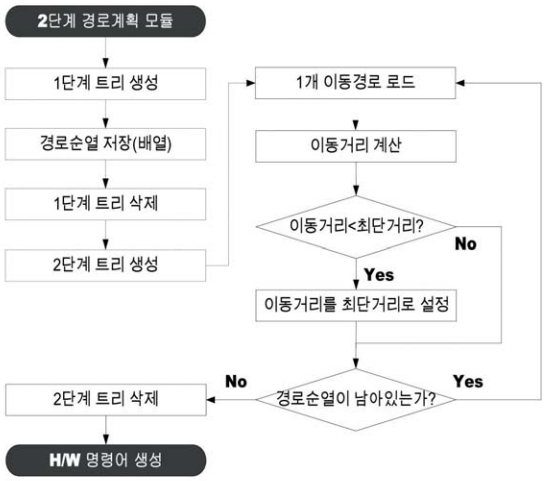


그림 10. 2단계 트리 알고리즘 순서도

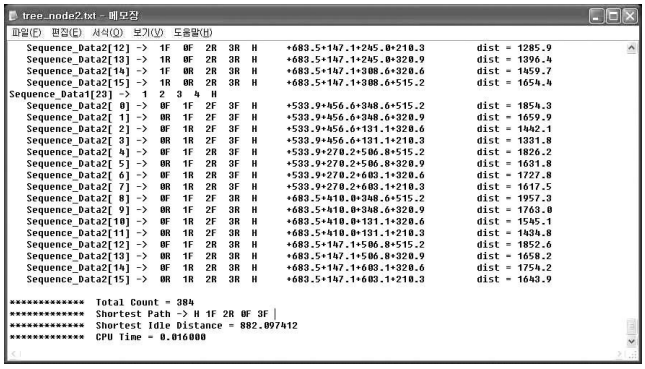


그림 11. 2단계 트리 실행결과

3.3 시간 비용 단축을 위한 1단계 트리 알고리즘

2단계 트리 알고리즘은 반드시 최단 경로를 보장하는 알고리즘이지만 매우 높은 시간 비용(time cost)을 요구한다. 실제 크랙의 수 n이 7일 때 연산시간은 3초를 초과하며, n이 8일 때는

72초를 초과한다. 본 연구에서는 2단계 트리 알고리즘의 이러한 문제점을 보완하기 위해 경로 순열에 대한 1단계 트리만을 이용하고 크랙의 진입점(front and rear)은 가까운 거리를 선택하는 1단계 트리 알고리즘을 개발하였다. 1단계 트리 알고리즘은 2번째 트리의 생성과 해체를 생략하기 때문에 전체 연산량에 있어 2단계 트리 알고리즘에 비해 $1/2^n$로 감소시킬 수 있다. 1단계 트리 경로계획 알고리즘의 순서도는 다음 그림 12와 같다.

4. 제안된 알고리즘의 성능분석

4.1 경로계획 알고리즘의 성능측정 환경

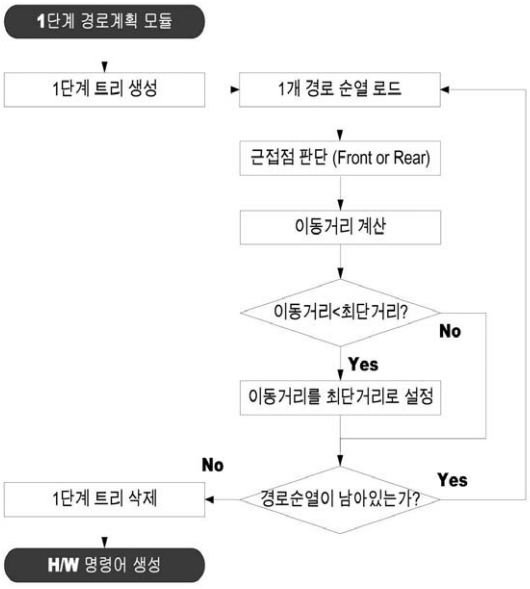


그림 12. 1단계 트리 알고리즘 순서도

성능 측정에 있어서 가장 중요한 항목은 연산시간과 크랙 간 이동거리이다. 이 두 가지 항목은 비전 알고리즘의 전체적인 프로세싱 타임과 직접적인 관계를 갖고 있다. 본 연구에서 연산시간 측정은 MFC(Microsoft Foundation Class)의 Clock() 함수를 이용하였다. Clock() 함수는 millisecond(1/1000 sec)단위로 측정된다. 연산시간의 측정범위는 경로계획 함수가 실행되는 시점부터 경로계획 결과가 텍스트로 출력되는 시점까지이다. 크랙간 이동거리 측정은 512×512 화소의 원시 이미지(Raw Image)를 기준으로 1개 화소의 크기를 1이라고 할 때, 두 점 사이의 거리는 두 점의 좌표차가 된다. 즉, A(x1, y1), B(x2, y2) 사이의 거리 D는 다음 식 2와 같으며 크랙간 이동거리 측정은 경로 내에 존재하는 크랙간 이동거리를 총 합산한 값이 된다.

$$Distance_{A,B} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \quad \text{(식 2)}$$

경로계획 알고리즘의 성능 측정에 사용된 데이터는 총 270장의 512×512 화소 원시 영상이고 크랙의 수가 1개인 영상부터 9개인 영상까지 각각 30장의 이미지가 성능측정에 사용되었다. 알고리즘의 성능 측정 환경은 표 6과 같다.

표 6. 성능평가에 사용된 시스템 환경

CPU	Pentium 4 2.4Ghz(C) Northwood
RAM	DDR 512M (400MHz)
OS	Windows XP (SP1)

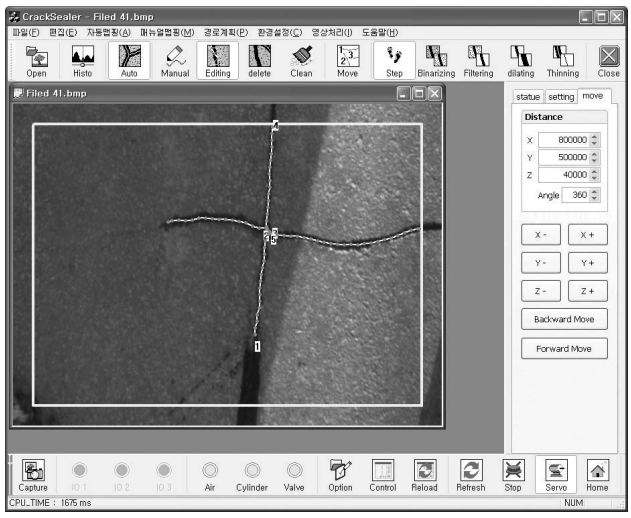


그림 13. APCS 크랙 네트워크 모델링 소프트웨어

본 연구에서 개발된 2단계 트리 및 1단계 트리 알고리즘은 그림 13의 크랙 인식 및 네트워크 모델링 소프트웨어에 통합되어 구현되었으며, 해당 소프트웨어는 Microsoft Visual C++ 6.0 기반으로 구현되었다. 본 소프트웨어에는 영상 획득, 크랙 인식, 크랙 네트워크 모델링, 경로계획, 모션 컨트롤 제어 등 크랙 실링 자동화 장비에 필요한 모든 기능이 통합 구현되어 있다.

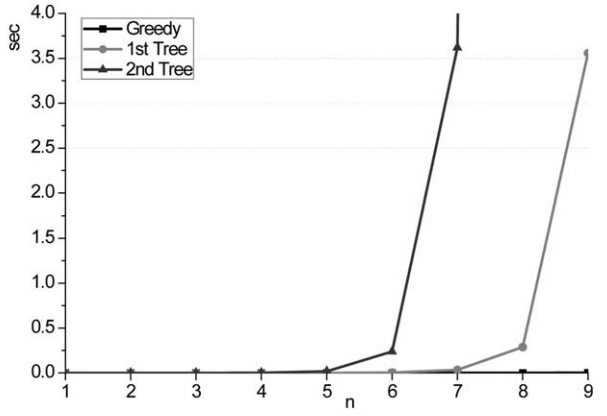
4.2 경로계획 알고리즘의 성능 측정 결과

기존 그리디 알고리즘과 본 연구에서 개발된 2단계 트리 알고리즘 및 1단계 트리 알고리즘을 구현하여 각 알고리즘의 연산시간과 이동거리를 측정한 결과(표 7), 그리디 알고리즘은 크랙의 수 n의 9까지 증가하더라도 연산시간이 최대 2ms이하였으나, 1차 트리 경로계획 알고리즘은 n이 9일 때 평균 3.55초가 걸려 1초를 초과하는 결과가 나타났고, 2차 트리 경로계획 알고리즘은 n이 7일 때 평균 3.61초가 걸려 1초를 초과하였다. 그림

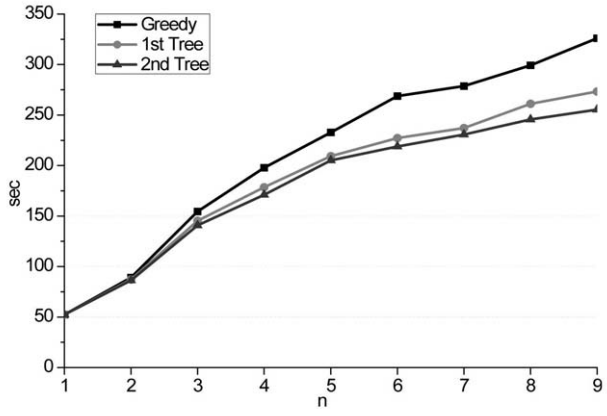
14의 ②에서 알 수 있듯이 항상 최적결과를 보장하는 2차 트리 경로계획 알고리즘에 비해 그리디 알고리즘은 평균 18.24% 더 먼 경로를 찾았으며, n이 증가할수록 오차가 증가하는 것으로 확인되었다. 이에 비해 1차 트리 경로계획 알고리즘은 2차 트리 경로계획 알고리즘에 비해 평균 4.03% 더 먼 경로를 찾았으며, 그리디 경로계획 알고리즘과 마찬가지로 n이 증가할수록 오차도 증가하였다.

표 7 경로계획 알고리즘의 성능측정 결과

n	연산시간 평균(단위:ms)			이동거리 평균(단위:pixel)		
	그리디	1차 트리	2차 트리	그리디	1차 트리	2차트리
1	0	0	0	52.38	52.38	52.38
2	0	0	0	89.08	87.09	86.28
3	0	0	0	154.51	145.26	140.68
4	0	0	2	197.86	178.53	171.03
5	0	0	18	232.76	209.23	205.03
6	0	4	239	268.83	227.12	218.9
7	0	33	3,618	278.63	237.02	230.7
8	0	289	72,843	299.14	261.06	245.67
9	2	3,557	1,767,980	325.94	273.21	255.38



① 경로계획 알고리즘 연산시간 측정결과

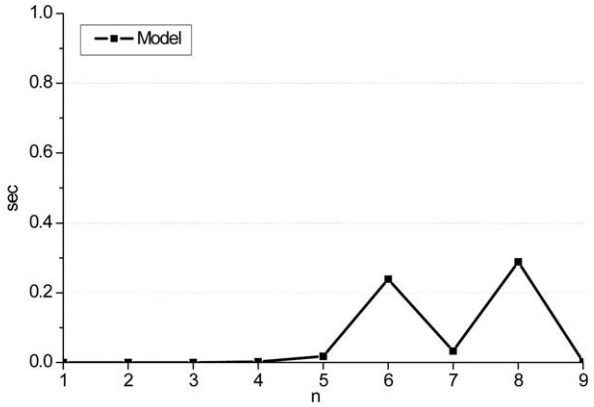


② 경로계획 알고리즘 이동거리 측정결과

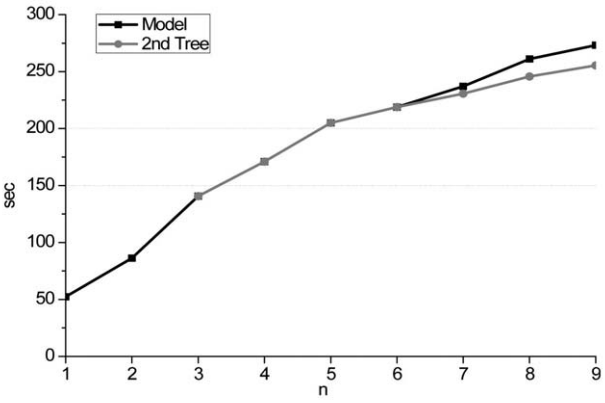
그림 14. 경로계획 알고리즘별 연산시간과 이동거리

4.3 크랙실링 장비의 경로계획 알고리즘 적용방법

앞서 언급한 바와 같이 크랙의 수 n에 따라 1차 트리 및 2차 트리 경로계획 알고리즘은 한계 적용범위가 있기 때문에 본 연구에서는 크랙실링 자동화 장비의 경로계획 알고리즘 적용에 있어 크랙의 수 n에 따라 적절한 경로계획 알고리즘이 실행될 수 있도록 하였다. 즉, 크랙의 수 n이 1이상 6이하인 경우 2차 트리 알고리즘이 실행되고, n이 7이상 8이하인 경우 1차 트리 알고리즘이 실행되며, n이 9이상인 경우 그리디 경로계획 알고리즘이 실행되도록 설정하였다. 그 결과 그림 15와 같이 경로계획 적용모델은 최대 0.3초 이내에 수행될 수 있으며, 이동거리의 적용결과는 최적 값에 비해 평균 5.7% 오차를 보였다.



① 경로계획 모델의 연산시간



② 경로계획 모델의 이동거리

그림 15. 크랙실링 자동화 장비의 경로계획 모델

5. 결 론

경로계획 알고리즘은 크랙실링 자동화 장비에서 자동으로 인식된 크랙 경로 정보를 이용하여 크랙 실링 자동화 장비의 말단 장치가 가장 짧은 이동경로를 찾아내어 신속하게 실링을 수행

할 수 있도록 하는 알고리즘으로써, 본 연구에서는 1단계 트리 알고리즘과 2단계 트리 알고리즘을 개발하여 기존 그리디 알고리즘과의 성능을 비교 분석하였으며, 본 연구를 통해 얻은 결론은 다음과 같다.

1) 크랙실링 자동화 장비의 머신비전 알고리즘 가운데 경로계획 알고리즘의 역할을 분석하고, 경로계획 알고리즘의 개념과 성능 차에 따른 생산성의 변화를 분석하였다.

2) 기존 그리디 경로계획 알고리즘을 분석한 결과, 매우 빠르고 효과적인 알고리즘이지만, 크랙의 수 n이 증가할수록 정확성이 저하되는 것으로 분석되었다.

3) 최적 경로계획 알고리즘의 이론적 고찰을 통해 크랙실링 자동화 장비의 경로계획 알고리즘은 경로순서에 관한 1단계 트리와 진입 순서에 관한 2단계 트리로 분할하여 모델링할 수 있음을 고찰하였다.

4) 경로 순서에 관한 1단계 트리 구조 모델링을 통해 경로 순열을 생성하였고, 다시 진입순서를 결정하기 위한 2단계 트리 구조를 모델링하여 크랙의 수 n에 대한 모든 이동거리를 산출하는 2단계 트리 경로계획 알고리즘을 개발하였다. 또한 2단계 트리 알고리즘의 시간 비용을 줄이기 위해 1단계 트리로 경로 순열을 생성하고 크랙의 진입순서는 가까운 위치를 선택하는 1단계 트리 경로계획 알고리즘을 개발하였다.

5) 기존 그리디 알고리즘과 본 연구에서 개발된 1단계 트리 알고리즘, 2단계 트리 알고리즘 등 각각의 알고리즘을 소프트웨어로 구현하였다. 본 연구에서 270장의 경로 데이터를 이용하여 각 알고리즘의 소요시간과 정확성을 테스트한 결과, 그리디 알고리즘이 가장 빠르지만 정확성(평균 18.24%의 오차)이 가장 낮았으며, 2단계 트리 알고리즘은 가장 느리지만 항상 최적의 경로를 탐색할 수 있었다. 1단계 트리 알고리즘은 2단계 트리 알고리즘보다는 빠르고, 오차는 평균 4.03%로 2단계 트리 알고리즘에 거의 근접하는 것으로 나타났다.

6) 크랙의 수 n에 따라 적절한 경로계획 알고리즘이 실행될 수 있도록 크랙의 수 n이 1이상 6이하인 경우 2차 트리 알고리즘이 실행되고, n이 7이상 8이하인 경우 1차 트리 알고리즘이 실행되며, n이 9이상인 경우 그리디 경로계획 알고리즘이 실행되도록 설정한 결과, 크랙실링 자동화 장비의 경로계획 적용 모델은 최대 0.3초 이내에 수행될 수 있으며, 이동거리의 적용결과는 최적 값에 평균 5.7% 오차가 있었다.

본 연구에서 개발된 1단계 트리 및 2단계 트리 경로계획 알고리즘은 크랙실링 자동화 장비의 생산성 향상에 기여할 수 있을 것으로 기대되며, 향후 유전자 알고리즘(genetic algorithm), 2-Opt 알고리즘과 같은 최적화 알고리즘(optimization algorithm)의 적용을 통해 시간 비용(time cost)의 성능을 개선할 수 있을 것으로 기대된다.

감사의 글

본 연구는 국토해양부 건설기술혁신사업의 연구비지원(06첨단융합C01)에 의해 수행 되었습니다.

참고문헌

김영석 (2004), “도로면 유지보수 자동화를 위한 원격조정 장비의 개발”, 건설교통부 한국건설교통기술평가원 최종보고서, p.100~114, 2004. 08

유현석, 이정호, 김영석, 김정렬(2004), “도로면 크랙실링 자동화를 위한 머신비전 알고리즘의 개발”, 한국건설관리학회 논문집, 제5권 제2호, pp.90~104, 2004. 04

유현석, 이정호, 김영석, 성낙원(2004), “신경망 학습기법을 이용한 도로면 크랙인식 알고리즘 개발에 관한 연구”, 2004 한국건설관리학회 학술발표대회 논문집, pp. 561~565, 2004. 11

이정호, 유현석, 김영석, 이준복, 조문영(2004), “도로면 크랙실링 자동화 로봇의 프로토타입 개발에 관한 연구”, 한국건설관리학회 논문집, 제5권, 제2호, pp. 162~171, 2004. 4

Dijkstra, E. W.(1959), “A Note on Two Problems in Connection with Graphs”, Numerische Mathematik Vol 1,, pp. 269~271

Feng, X., Mathurin, R., and Velinsky S. A. (2005), “Practical, Interactive, and Object-Oriented Machine Vision for Highway Crack Sealing”, Journal of Transportation Engineering, Vol. 131, No. 6, pp. 451~459, June, 2005.

Glover, F., and Punnen, A. P.(1997), “The traveling salesman problem : New solvable cases and linkages with the development of approximation algorithms”, J. Oper. Res. Soc., 48, pp. 502~510.

Graham, R. and Hell, P. “On the history of the minimum spanning tree problem”, “Annals of the History of Computing”, Vol 7. No. 1, 1985, pp.43~57

Horowitz, E., Sahni, S., and Anderson-Freed, S.(1993), “Fundamentals of Data Structures in C”, W. H. Freeman and Company, 1993.

Kim, Y. S.., Hass, C. T.(1998), "Path Planning for Machine Vision Assisted, Teleoperated Pavement Crack Sealer", Journal of Transportation Engineering, Vol. 124, No. 2, Mar/Apr, 1998.

Kirschke, K. R. and Velinsky, S. A. (1992), "Histogram-Based Approach for Automated Pavement-Crack Sensing",Journal of Transportation Engineering, Vol. 118, Issue 5, pp. 700-710, September/October 1992

Kruskal, J. B.(1956), "On the Shortest Spanning Tree of a Graph and the Traveling Salesman Problem", Proc. Amer. Math. Soc. 7, pp. 48~50

Mathurin R., Velinsky S. A(2000), "Simulated Annealing for the Optimal Trajectory Planning of an Automated Crack Sealing Machine", Proceedings ASME design technical conference, 2000

Prim, R. C.(1957), "Shortest Connection Networks and Some Generalizations", Bell Syst. Tech. J. 36, pp 1389~1401

Sollin, G.(1962), "Problemes de Recherche Operationelle Report", C. 41 S.E.G. Paris Le Trace des Canalizations, pp. 15~23

논문제출일: 2010.03.09
논문심사일: 2010.03.12
심사완료일: 2010.05.20

Abstract

During the last two decades, several tele-operated and machine-vision-assisted systems have been developed in construction and maintenance area such as pavement crack sealing, sewer pipe rehabilitation, and excavation. In developing such tele-operated and machine-vision-assisted systems, trajectory plans are very important tasks for optimal motions of robots whether their environments are structured or unstructured. This paper presents an optimal trajectory planning algorithm used for a machine-vision-assisted automatic pavement crack sealing system. In this paper, the performance of the proposed optimal trajectory planning algorithm is compared with the greedy trajectory plans which are used in previously developed pavement crack sealing systems. The comparison is based on computational cost versus overall gains in crack sealing efficiency. Finally, it is concluded that the proposed algorithm plays an important role in productivity improvement of the automatic pavement crack sealing system developed.

Keywords : *Crack Sealing, Machine Vision Algorithm, Trajectory Planning, Construction Automation*