Journal of Korea Robotics Society (2020) 15(2):100-106
https://doi.org/10.7746/jkros.2020.15.2.100 ISSN: 1975-6291 / eISSN: 2287-3961

양팔 로봇을 위한 부분적 비동기 작업 계획

Partially Asynchronous Task Planning for Dual Arm Manipulators

정 성 엽[1] · 황 면 중†
Seong Youb Chung[1], Myun Joong Hwang†

Abstract: In the agricultural field, interests in research using robots for fruit harvesting are continuously increasing. Dual arm manipulators are promising because of its abilities like task-distribution and role-sharing. To operate it efficiently, the task sequence must be planned adequately. In our previous study, a collision-free path planning method based on a genetic algorithm is proposed for dual arm manipulators doing tasks cooperatively. However, in order to simplify the complicated collision-check problem, the movement between tasks of two robots should be synchronized, and thus there is a problem that the robots must wait and resume their movement. In this paper, we propose a heuristic algorithm that can reduce the total time of the optimal solution obtained by using the previously proposed genetic algorithm. It iteratively desynchronizes the task sequence of two robots and reduces the waiting time. For evaluation, the proposed algorithm is applied to the same work as the previous study. As a result, we can obtain a faster solution having 22.57 s than that of the previous study having 24.081 s. It will be further studied to apply the proposed algorithm to the fruit harvesting.

Keywords: Two Robot Arms, Fruit Harvesting, Task Planning, Collision-free, Asynchronous Movement

1. 서 론

로봇을 농업에 이용하기 위한 연구가 최근 활발하게 진행되고 있다. 특히 과수 수확에 로봇을 이용하고자 하는 연구가 다수 진행되고 있다. 미국의 스타트업 기업인 어번던트 로보틱스(abundant robotics)[1]는 농업용 로봇을 이용하여 사과 수확을 자동화하려는 시도를 하였으며, Tanigaki[2] 등은 4자유도 로봇을 개발하여 말단(end-effector)에 진공 파이프를 부착하여 체리를 수확하는 로봇을 개발하였다. Van Henten[3] 등은 온실 내에서 오이를 자동으로 수확하기 위한 로봇의 기구학적 구조에 관한 최적 설계 방법을 제안하였다. Nguyen[4] 등은 과수원에서 로봇이 사과를 수확하기 위한 모션과 계층적 경로 계획을 위한 체계를 제안하였으며, 시뮬레이션과 실험을 통해 가능성을 검증하였다. 농업 분야에서 생산성을 높이기 위해 양팔 로봇(dual arm manipulators)을 이용한 연구도 꾸준히 진행되고 있다. Zhao[5] 등은 비구성 환경에서 로봇을 이용한 토마토 수확 작업의 효율을 향상시키기 위해 모듈 개념의 양팔 로봇 시스템을 제안하였다. 3자유도인 각 로봇은 다른 형태의 말단을 가지고 있으며, 협동을 통해 토마토 수확을 효과적으로 수행할 수 있음을 증명하였다. Davidsona[6] 등은 사과 수확의 전체 시간을 줄이기 위해 양팔 로봇 시스템을 도입하였다. 이 연구에서는 두 팔의 역할을 구분하여 하나의 로봇이 사과를 채집(picking)해서 떨어뜨리면 다른 로봇은 떨어진 사과를 잡아서(catching) 저장박스까지 이동하는 데 걸리는 시간을 줄일 수 있었다. 이와 같이 로봇은 노동 집약적인 과수 수확 분야에서 아주 유용하게 사용될 수 있으며, 특히 양팔 로봇을 이용할 경우 작업의 효율면에서 장점을 가짐을 알 수 있다. 그러나 산업용 로봇에서의 양팔 로봇을 이용하는 기술적 수준까지 도달하기 위해서는 더 많은 연구가 필요하다. 양팔 로봇을 효과적으로 이용하기 위해서는 작업 계획이 필수적이며 특히 같은 작업 공간을 이용하기 때문에 로봇 간 충돌을 고려해야만 한다.

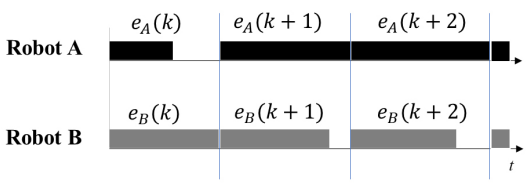
본 연구팀은 이전 연구에서 같은 공간에서 작업하는 양팔

Received : Feb. 14. 2020; Revised : Mar. 11. 2020; Accepted : Mar. 11. 2020
※ The research was supported by a grant from the 2018 program for visiting professors overseas in Korea National University of Transportation
1. Professor, Mechanical Engineering, Korea National Univ. of Transportation, Chungju, Korea (sychung@ut.ac.kr)
† Associate Professor, Corresponding author: Mechanical Engineering, Korea National Univ. of Transportation, Chungju, Korea (mjhwang@ut.ac.kr)

Copyright ⓒKROS

로봇의 충돌 없는 최적 작업 시퀀스(sequence)를 구하는 방법을 제안하였다[7,8]. 두 로봇이 같은 작업 공간에서 이동하는 경우에 작업 시퀀스를 구할 때 로봇 간에 충돌할 수 있는 상황을 고려해야한다. 작업 대상이 N개라 하면, 두 로봇이 작업을 끝내고 이동할 때, 어디에서 충돌이 발생할지 조사를 해야 한다. 로봇의 이동 경로에서 충돌을 검사하는 과정은 수학적으로 많은 시간을 필요로 한다. 따라서, 충돌을 조사하며 두 로봇의 작업 시퀀스를 구하는 것은 계산 복잡도로 인해 사실상 불가능하다. Yoon[7,8] 등은 쉐도우 공간이라는 개념을 이용하여 충돌을 쉽게 판별할 수 있는 방법을 제안하였고, 모든 이동 경로의 조합에 대해 사전에 충돌 조사를 시행하여 충돌 행렬에 저장하고 작업 시퀀스를 구할 때는 충돌 여부를 충돌 행렬의 해당 요소를 간단히 확인하는 방법으로 충돌 검사 시간을 최소화할 수 있었다. [Fig. 1]은 로봇의 작업 시퀀스는 일부를 간트 차트(Gantt chart)로 나타낸 것이다. 여기서 $e_\pi(k)$는 로봇 π가 k 번째 또는 k 단계의 작업을 위해 이동하는 이벤트를 의미하며, 작업 시퀀스는 로봇 A와B의 이벤트 시퀀스로 생각할 수 있다. 작업 시퀀스를 구할 때, 충돌 검사를 단순화하기 위해, [Fig. 1]과 같이 두 로봇이 작업을 끝낸 후 동시에 이동하는 것으로 가정하였다. 따라서, 기존 연구의 작업 시퀀스대로 작업을 하기 위해서는 필연적으로 하나의 로봇은 대기해야만 한다. 만일, 충돌을 고려하지 않는다면, 대기 시간을 모두 없앨 수 있을 것이고 작업 종료 시간을 단축할 수 있을 것이다. 그러나, 현실적으로 로봇 이동 시 충돌은 존재할 수밖에 없기 때문에 작업 시퀀스의 모든 단계에서 대기시간을 없앨 수는 없다. 하지만, 일부라도 대기시간을 줄일 수 있다면, 전체 시간을 단축할 수 있는 효과가 있다. 만일, [Fig. 1]의 k 번째 단계에서 로봇 B의 $e_B(k)$가 $k+1$ 단계의 로봇 A의 $e_A(k+1)$과 충돌이 없다면, $k+1$ 단계의 로봇 A의 작업을 대기시간 없이 바로 시작할 수 있다. 전체 시퀀스에 이와 같은 상황을 검사하여 로봇의 대기시간을 없앤다면 전체 작업 시간을 단축할 수 있다.

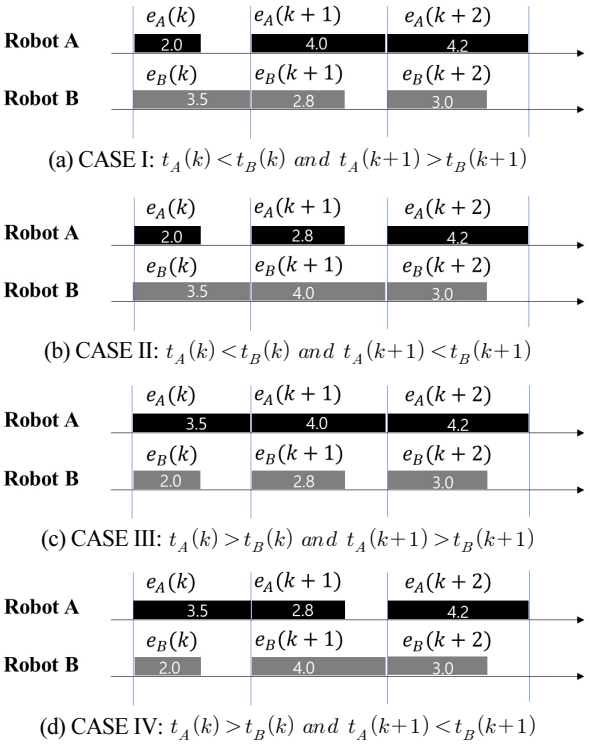
본 논문은 기존의 동기화된 최적 작업 시퀀스의 부분적인 비동기화를 통해 기존 해의 성능을 향상하는 방법을 제안하고자 한다. 2장에서는 작업시간 단축이 가능한 경우에 대해 검토하고, 작업 시간을 단축할 수 있는 부분적 비동기 알고리즘에 대해 기술하도록 한다. 3장에서는 기존 논문에서 제안한 유전자 알고리즘의 검증을 위해 이용하였던 작업에 대해 본 논문에서 제안한 방법을 동일하게 적용하고 기존 방법의 성능과 비교하여 알고리즘의 유용성을 검증한다. 마지막으로 4장에서는 결론과 향후 계획에 대해서 기술한다.



[Fig. 1] Example of a task sequence for two robots

2. 부분적 비동기 알고리즘

양팔 로봇의 작업 시퀀스에서 로봇 π의 k 번째 이동을 $e_\pi(k)$라 하고, 이때 걸리는 시간을 $t_\pi(k)$라 하면, $t_A(k)$와 $t_B(k)$의 크기와 $t_A(k+1)$와 $t_B(k+1)$의 크기에 따라, [Fig. 2]와 같이 4가지 경우로 작업 시퀀스를 분류할 수 있다. [Fig. 2]는 로봇의 이동을 간트 차트로 표현한 것으로 그림의 막대에 표시된 값은 시간을 의미한다. 여기서 [Fig. 2(a)]의 경우, A 로봇의 이동 $e_A(k+1)$을 바로 시작하면, $k+2$번째 시작 시간을 앞당길 수 있다. 단, 로봇 A가 $e_A(k+1)$을 수행하는 동안 로봇 B의 $e_B(k)$과 충돌이 없는 경우 가능하다. [Fig. 2(b)]의 경우는 로봇 A의 $e_A(k)$와 $e_A(k+1)$에 필요한 시간이 로봇 B의 $e_B(k)$와 $e_B(k+1)$에 필요한 시간보다 짧기 때문에 $e_A(k+1)$을 먼저 시작한다고 시간이 단축되지 않는다. [Fig. 2(c)]도 [Fig. 2(b)]와 비슷한 경우로 시간을 단축시킬 수 없다. [Fig. 2(d)]의 경우는 [Fig. 2(a)]의 경우와 마찬가지로 $e_B(k+1)$을 $e_A(k)$가 끝난 후 바로 시작하면, 시간 단축 효과를 볼 수 있다. 이러한 과정을 $k=1$부터 시퀀스가 끝날 때까지 반복하면, 전체 작업



[Fig. 2] Four cases of adjacent events (k, $k+1$) for two robots according to task time

시간을 줄일 수 있다.

만일 조립 시퀀스가 시간 단축으로 시작 시간 등이 변경된다면, 변경된 시간 정보를 저장하기 위한 변수가 필요하다. 이를 위해 본 논문에서는 로봇 π의 대기시간에 해당하는 $d_\pi(k)$를 새롭게 정의하였다. [Fig. 2(a)]에서 로봇 A의 경우 k와 $k+1$ 단계의 대기시간 $d_A(k)$와 $d_A(k+1)$는 각각 1.5 s와 0.0 s가 되며, 로봇 B의 경우 $d_B(k)$와 $d_B(k+1)$는 각각 0.0 s와 1.2 s가 된다. 만일 [Fig. 2(a)]에서 A 로봇이 이동 $e_A(k+1)$을 바로 시작하여 [Fig. 3]과 같이 작업 시퀀스의 시간 정보가 변경되었다면, 대기 시간 정보를 업데이트하여 변경 사항을 저장할 수 있다. 즉, [Fig. 3]의 로봇 A의 $d_A(k)$와 $d_A(k+1)$는 각각 0.0 s와 0.3 s로 변경되며, 로봇 B의 $d_B(k)$와 $d_B(k+1)$는 각각 0.0 s와 0.0 s가 된다. [Fig. 3]에서 보듯이, $k+1$ 번째 단계에서 두 로봇의 동기화는 부분적으로 풀리게 된다.

유전자 알고리즘으로부터 구한 유사 최적해는 모든 단계에서 각 로봇의 이동이 동기화되어 있다. 따라서 초기 대기시간 $d_A(k)$와 $d_B(k)$는 $t_A(k)$와 $t_B(k)$ 중 긴 시간에 의해 식 (1)과 (2)와 같이 결정된다.

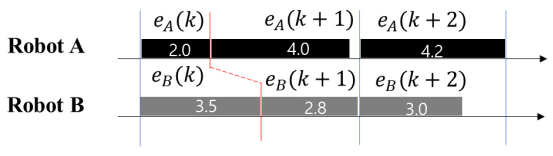
$$d_A(k) = \max(t_A(k), t_B(k)) - t_A(k) \tag{1}$$

$$d_B(k) = \max(t_A(k), t_B(k)) - t_B(k) \tag{2}$$

[Fig. 2]의 4가지 경우에서 로봇 A와 B의 k와 $k+1$ 단계에서 대기시간을 살펴보면 [Table 1]과 같다. 여기서 대기시간을 이용하여 식 (3)의 좌변을 계산하여 보면, [Table 2]와 같이 시간을 단축할 수 있는 경우는 그 값이 0 보다 크고, 그렇지 않은 경우는 0이 됨을 확인할 수 있다. 이러한 성질을 이용하여 시간을 단축할 수 있는 조건으로 식 (3)을 이용할 수 있다.

$$d_A(k)d_B(k+1) + d_B(k)d_A(k+1) > 0 \tag{3}$$

식 (3)을 만족하는 경우 시간 단축에 대한 정보를 업데이트하기 위하여 k 단계의 대기시간 $d_A(k)$와 $d_B(k)$과 $k+1$ 단계의 대기시간 $d_A(k+1)$와 $d_B(k+1)$를 변경하여야 한다. 수학적 전개를 위해 $st_\pi(k)$를 로봇 π의 k 단계의 이벤트 시작 시간으로 정의하였다. $st_\pi(k)$는 식 (4)와 같이 $k-1$ 단계까지의 이벤트 시간 t_π과 대기시간 d_π을 모두 합하여 구한다.



[Fig. 3] Time-modification of the task sequence of [Fig. 2(a)]

[Table 1] Delay times for four cases of [Fig. 2]

CASE	$d_A(k)$	$d_B(k)$	$d_A(k+1)$	$d_B(k+1)$
I	1.5	0.0	0.0	1.2
II	1.5	0.0	0.0	0.0
III	0.0	0.0	1.5	1.2
IV	0.0	1.2	1.5	0.0

[Table 2] Conditions using delay times

CASE	$d_A(k)\,d_B(k+1)$	$d_B(k)\,d_A(k+1)$	Eqn. (3)
I	1.8	0.0	1.8
II	0.0	0.0	0.0
III	0.0	0.0	0.0
IV	0.0	1.8	1.8

$$st_\pi(k) = \sum_{i=1}^{k-1}[t_\pi(i) + d_\pi(i)] \tag{4}$$

우선 [Fig. 2]의 (a)나 (d)의 경우와 같이 k 단계에서 시간을 단축할 수 있을 때, 로봇 A와 B의 k 단계의 대기시간은 모두 식 (5)와 같이 0이된다. 이 값의 영향으로 인하여 $k+1$ 단계의 대기시간 역시 변경된다.

$$d_A(k) = d_B(k) = 0.0 \tag{5}$$

$k+1$ 단계의 대기시간 $d_A(k+1)$와 $d_B(k+1)$는 로봇 A와 B의 $k+1$ 단계의 작업 시작시간 $st_A(k+1)$와 $st_B(k+1)$에 $k+1$ 단계의 이동시간 $t_A(k+1)$와 $t_B(k+1)$을 각각 더한 후, 큰 값을 기준으로 식 (6)과 (7)과 같이 구할 수 있다.

$$d_A(k+1) = \max(st_A(k+1) + t_A(k+1), st_B(k+1) + t_B(k+1)) - (st_A(k+1) + t_A(k+1)) \tag{6}$$

$$d_B(k+1) = \max(st_A(k+1) + t_A(k+1), st_B(k+1) + t_B(k+1)) - (st_B(k+1) + t_B(k+1)) \tag{7}$$

예를 들어, [Fig. 2(a)]가 시간 단축에 의해 [Fig. 3]과 같이 되었다면, 식 (5)에 의해, $d_A(k)$와 $d_B(k)$는 모두 0이 된다. 그리고, $st_A(k)$와 $st_B(k)$를 모두 0으로 가정할 때, $st_A(k+1) + t_A(k+1)$와 $st_B(k+1) + t_B(k+1)$는 각각 2.0 s + 4.0 s = 6.0 s와 3.5 s + 2.8 s = 6.3 s로 계산되고, 따라서 식 (6)과 (7)에 의해 $d_A(k+1)$는 max(6.0 s, 6.3 s) − 6.0 s = 0.3 s로, $d_A(k+1)$는 max(6.0 s, 6.3 s) − 6.3 s = 0.0 s로 각각 계산된다. 유전자 알고리즘을 통해 구한 유사 최적해에 이와 같은 알고리즘을 반복적으로 적용하여 전체 작업 시간을 단축할 수 있다. 이와 같은 과정을 알고리즘 형태로 나타내면 아래와 같다.

STEP 1. 유전자 알고리즘을 통해 결과가 좋은 순으로 유사 최적해 $\{(seq_A, seq_B)_i\}$을 N_S 선택함

STEP 2. 선택된 해 i에 대해 로봇 $d_A(k)$와 $d_B(k)$를 구함

STEP 3. 선택된 해 i에 대해, $k=1$부터 아래의 과정을 수행함

A. $d_A(k)d_B(k+1)+d_B(k)d_A(k+1)$를 계산하여 0보다 큰 경우 B로 이동하고, 그렇지 않으면 C로 이동함

B. 식 (5), (6), (7)을 이용하여 로봇 A와 B의 k 번째 및 $k+1$ 번째 대기 시간 $d_A(k), d_B(k), d_A(k+1), d_B(k+1)$을 구함

C. k가 작업 시퀀스의 길이 M 보다 작은 경우 k에 1을 더한 후 A로 이동하고 그렇지 않으면, STEP 4로 진행함

STEP 4. 작업 시퀀스 i에 대해 작업 종료 시간 $END_t(i)$을 $\max(st_A(M)+t_A(M), st_B(M)+t_B(M))$ 값으로 저장

STEP 5. i가 N_s 보다 작으면 i에 1을 더한 후 STEP 2로 이동하고 그렇지 않으면 STEP 6로 이동함

STEP 6. 구한 $END_t(i)$ 중 가장 작은 값으로 최적해를 구함

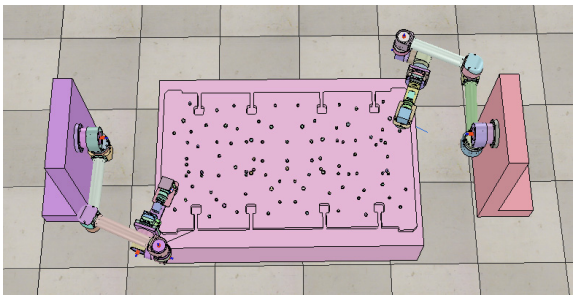
3. 시뮬레이션 및 결과

2장에서 제안한 알고리즘을 검증하기 위하여 기존 연구에서 사용하였던[7] 동일한 시스템과 문제를 이용하였다. [Fig. 4]는 양팔 로봇 시스템이 대면적 위의 물체를 놓기 위해 준비하고 있는 상태를 나타내며 왼쪽에 있는 로봇이 A, 오른쪽에 있는 로봇이 B이다. [Fig. 5]는 102개의 작업점을 도식적으로 보여주는 것으로 O와 X 점은 각각 로봇 A와 로봇 B가 접근해서 수행할 수 있는 작업점의 위치를 나타낸다. 따라서, [Fig. 5]의 중심부에 O와 X가 중첩되어 있는 부분은 로봇 A와 B가 모두 작업 가능한 지점을 나타낸다. [Fig. 5]의 x축과 y축의 값은 mm 단위를 의미한다.

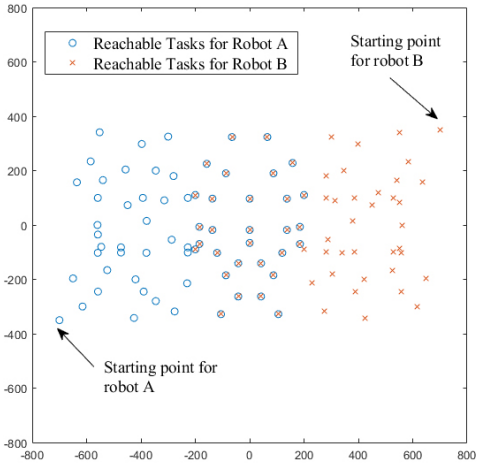
[Fig. 5]의 작업에 대해 기존 연구 결과에서 최적 작업 시퀀스를 그려보면 [Fig. 6]과 같고, 이동시간만 고려했을 때, 전체 시간은 24.081 s이다. 실제로는 작업 종료 시간은 각 지점에서 작업 시간과 이동 시간을 더해서 구해야 하지만 [Fig. 4]의 예와 같이 많은 응용에서 작업 시간은 거의 동일하기 때문에 본 논문에서는 작업 시퀀스의 성능 평가로 이동 시간만을 고려하였다.

[Fig. 6]에서 왼쪽과 오른쪽의 궤적은 각각 로봇 A와 B가 작업을 수행하며 이동한 경로를 나타내며, 숫자의 의미는 [Fig. 1]과 같이 이벤트를 간트 차트로 나타냈을 때, 로봇 π의 k 단계의 $e_\pi(k)$의 의미한다. 만일 왼쪽 궤적과 오른쪽 궤적에서 같은 숫자를 가진다고 할 때, 같은 시각에 로봇 A와 B는 해당 모서리의 시작점에서 같이 출발한다고 생각할 수 있다. 예를 들어 오른쪽 10번에 해당하는 모서리에 표시된 삼각형의 위치에 있는 로봇 B는 왼쪽 10번에 해당하는 모서리에 표시된 삼각형의 위치에 로봇 A가 작업을 마치고 도착했을 때, 동시에 다음 단계의 이동을 시작하는 것으로 생각할 수 있다. 본 논문은 이러한 대기시간을 체계적으로 줄이기 위한 방법에 관한 것이다.

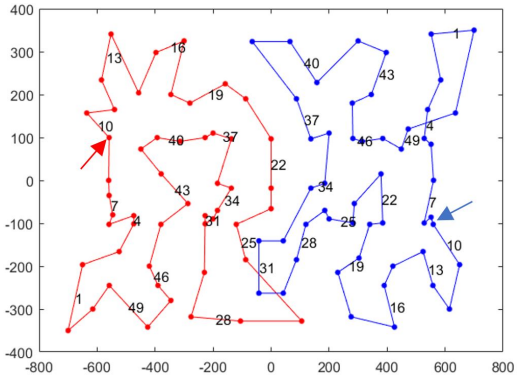
앞서 제안한 부분적 비동기 알고리즘이 효과가 있는 지 살펴보기 위해 기존의 최적 시퀀스에 적용해 보았다. 그 결과는 [Fig. 7]과 같으며, 전체 작업 종료 시간은 23.4 s로 0.7 s 정도 시간이 단축된 것을 확인할 수 있었다. [Fig. 7]의 숫자들은 해당 단계에서 식 (3)을 만족하여, 시간 단축이 이루어지고, 이에 따라 그 지점에서 로봇 A와 B가 비동기화가 이루어짐을 의미한다. 예를 들어 설명하기 위해 단계 2부터 4까지를 [Fig. 8]에



[Fig. 4] Configuration of two robotic arms



[Fig. 5] An example with 102 task points

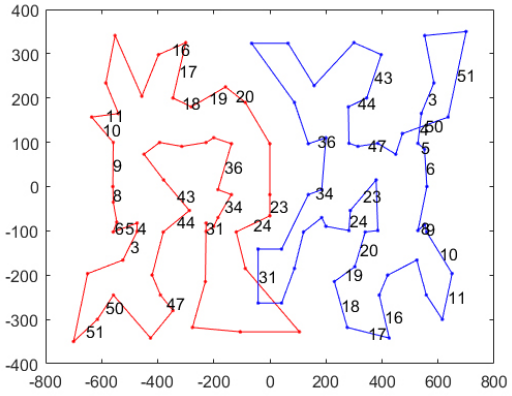


[Fig. 6] A near optimal sequence for two robots[7]

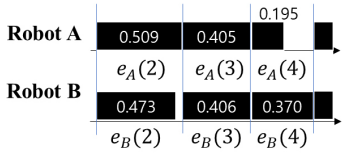
간트 차트로 알고리즘 적용 전과 후로 나타내어 표현하였다. [Fig. 8(a)]와 [Fig. 8(b)]를 보면, 3단계와 4단계에서 로봇 B의 이동이 앞당겨져 5번째 단계의 시작 시간이 0.033 s 만큼 줄어든 것을 확인할 수 있다.

기존의 유전자 알고리즘이 구한 최적 시퀀스는 각 단계에서 로봇 A와 B의 이동이 동기화되어 있기 때문에, 작업 순서의 배치에 따라 기존 알고리즘에서는 최적이 아니었지만 경우에 따라 본 논문에서 제안한 알고리즘에 의해 더 좋은 해가 될 수도 있다. 이것을 검증하기 위하여 본 논문에서는 기존 유전자 알고리즘을 반복하여 적용해 좋은 값을 가지는 최적해의 집합을 구하고, 이 집합의 모든 요소에 대해 제안한 알고리즘을 적용하여 최적해를 구하였다. 최적해의 집합은 총 80개의 요소를 가지도록 하였다.

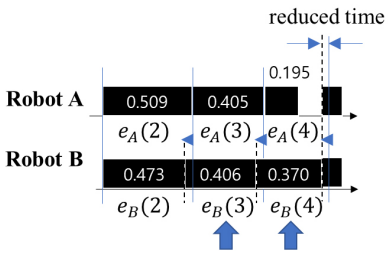
[Table 3]은 80개의 해에 대한 유전자 알고리즘의 결과를 전체 시간이 짧은 순으로 정렬한 것이다. [Table 3]의 두번째 열은 유전자 알고리즘의 결과로 얻은 해의 전체 시간을 나타내며, 세번째 열은 본 논문에서 제안한 부분적 비동기 알고리즘을 적용한 결과를 나타내고, 네번째 열은 두 시간의 차이를 나타낸다. [Table 4]는 부분적 비동기 알고리즘을 이용한 결과값 즉, 세번째 열의 값을 이용하여 [Table 3]를 다시 정렬한 결과이다. [Table 4]를 보면, 유전자 알고리즘으로 구한 최적의 해보다 다른 해의 결과가 더 좋게 나타나는 것을 확인할 수 있다. 유전자 알고리즘으로 정렬한 22번째 해가 부분적 비동기 알고리즘을 적용하였을 때, 가장 좋은 해가 됨을 알 수 있다. 따라서, 가장 좋은 해는 22.57 s의 최적 시간을 가지는 것으로, 기존 해가 24.332 s를 가지는 것에 비해 1.762 s 만큼 단축된 것을 확인할 수 있다. 22번째의 해를 [Fig. 7]과 같이 그래프로 나타내면 [Fig. 9]가 된다. [Fig. 9]에서 보면 총 51단계 중에 27개 부분에서 시간 단축이 이루어졌다는 것을 알 수 있다. 그리고, 오른쪽 루프에서 7단계와 43단계를 보면 궤적이 꼬여 있는 것을 확인할 수 있다. 이것은 로봇 B가 조금 돌아가더라도 대기시간을 줄여 전체 시간이 줄 수 있다는 것을 보여주는 예이다.



[Fig. 7] Result of the partially asynchronous algorithm for the task sequence of [Fig. 6]



(a) Events from 2 to 4 of the task sequence of [Fig. 6]



(b) Events from 2 to 4 of the task sequence of [Fig. 7]

[Fig. 8] Gantt chart representation of [Fig. 6] and [Fig. 7]

만일 작업 시퀀스에 [Fig. 10(a)]과 같은 상황이 있다고 가정하고, k 단계에서 부분적 비동기 알고리즘을 적용하면 [Fig. 10(b)]와 같이 된다. 여기에 다시 한 번 알고리즘을 적용하면, [Fig. 10(c)]로 시퀀스가 바뀌게 된다. 그러나, 이와 같은 상황에서는 $e_B(k)$가 끝나기 전에 $e_A(k+2)$가 시작되어 로봇 간에 충돌이 있을 수 있다. [Table 3]의 80개의 해에 [Fig. 10(a)]

[Table 3] Sorted sequences based on the total time of the genetic algorithm

No	Total Time of the Genetic Algorithm	Total Time of the Proposed Algorithm	Time difference
1	24.081	22.733	1.348
2	24.133	23.194	0.939
3	24.143	23.295	0.848
4	24.173	23.319	0.855
5	24.173	23.319	0.855
6	24.173	23.319	0.855
:	:	:	:
79	25.128	23.193	1.935
80	25.289	23.405	1.883

[Table 4] Sorted sequences based on the total time of proposed algorithm

No	Total Time of the Genetic Algorithm	Total Time of the Proposed Algorithm	Time difference
22	24.332	22.570	1.762
27	24.412	22.600	1.812
1	24.081	22.733	1.348
16	24.242	22.771	1.471
55	24.706	22.819	1.887
23	24.399	22.843	1.555
:	:	:	:
78	25.038	23.964	1.075
54	24.695	24.044	0.651

와 같은 경우를 체크해 봤을 때, 2개의 해에서 이러한 경우가 있는 것으로 확인되었다. 적은 수이기는 하지만 예외 상황을 처리하기 위해 본 논문에서는 [Fig. 10(d)]와 같이 대기시간을 변경하는 방법을 취하였다.

만일, $e_B(k)$와 $e_A(k+2)$가 충돌이 없다면, [Fig. 10(c)]로 시퀀스를 변경하면 시간을 더 단축시킬 수도 있을 것이다. 그러나, 이것은 또 다른 문제를 발생시킬 수 있다. 예를 들어 $e_B(k+2)$가 0.1 s이고, $e_A(k+2)$가 0.15 s라면, 이번에는 $e_B(k)$와 $e_A(k+3)$의 충돌 여부를 따져야 하고, $k+3$ 단계의 시간에 따라, $k+4$ 단계도 충돌 검사가 필요한 상황이 발생할 수 있다. 이것을 알고리즘을 너무 복잡하게 할 수 있다. 따라서 본 연구에서는 앞서 설명한 바와 같이 예외 상황으로 k 단계의 대기시간을 변경하여 [Fig. 10(d)]와 같이 처리하였다.

예외 상황을 수학적으로 나타내면 다음과 같다. 예외 상황 조건은 식 (8)과 같으며, 지연시간은 식 (9), (10) 및 (11)과 같다. 여기서 π는 로봇 A 또는 B를 나타내며, $\sim\pi$는 π와 반대되는 로봇을 의미한다. 예를 들어 π가 A면, $\sim\pi$는 B가 된다.

$$d_\pi(k) > t_\pi(k+1) \tag{8}$$

$$d_\pi(k) = s_{\sim\pi}(k+1) - s_\pi(k) - t_\pi(k) - t_\pi(k+1) \tag{9}$$

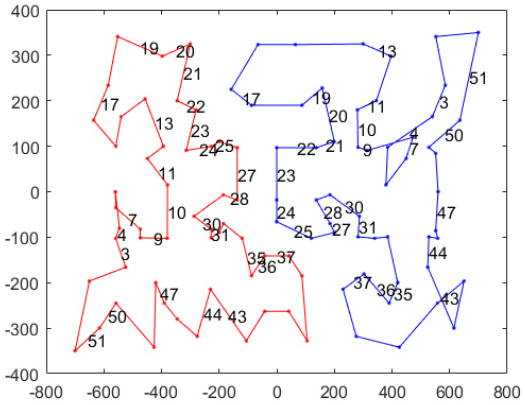
$$d_\pi(k+1) = t_{\sim\pi}(k+1) \tag{10}$$

$$d_{\sim\pi}(k) = d_{\sim\pi}(k+1) = 0 \tag{11}$$

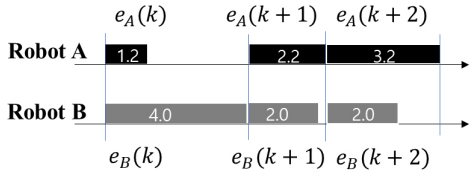
[Fig. 10(a)]와 같이 π가 A일 때를 예로 들어 설명하면, 식 (8)은 $e_A(k+1)$의 시간 $t_A(k+1)$가 로봇 A의 대기시간 $d_A(k)$보다 작아 $e_A(k+2)$의 시작이 $e_B(k)$가 끝나기 전에 시작할 수 있음을 나타내는 조건이다. 조건식 (8)은 시간 단축 조건식 (3)을 만족하는 경우에만 적용한다. 식 (9)는 로봇 A의 대기시간 $d_A(k)$를 구하는 것으로 $e_A(k+1)$가 끝나는 시간을 $e_B(k)$가 끝나는 시간 $st_B(k+1)$으로 맞추기 위한 수식이다. 즉, 이 값에 $st_A(k)$와 $e_A(k)$와 $e_A(k+1)$의 시간 $t_A(k)$와 $t_A(k+1)$을 차감하면 대기시간 $d_A(k)$를 구할 수 있다. 그리고, $d_A(k+1)$은 로봇 B의 시간 $t_B(k+1)$ 보다 작게만 설정하면 되며, 본 논문에서는 $t_B(k+1)$로 식 (10)과 같이 정하였다. 이제 로봇 B의 대기시간 $d_B(k)$와 $d_B(k+1)$은 [Fig. 10(d)]와 같이 모두 0으로 식 (11)과 같이 결정된다.

4. 결 론

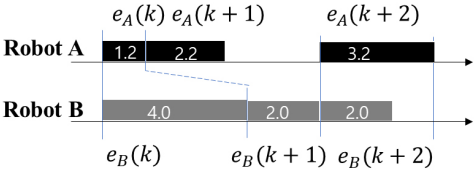
본 논문은 농업 분야 특히 과수 수확 분야에 이용될 수 있는 양팔 로봇을 위해 충돌이 없는 경로 계획에 관한 알고리즘을 제안한다. 본 연구팀은 이전 연구에서 유전자 알고리즘을 이용하여 양팔 로봇의 충돌 없는 작업 계획을 생성하는 효과적



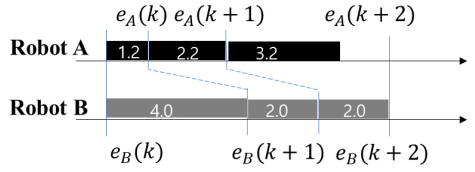
[Fig. 9] Best solution of the proposed algorithm



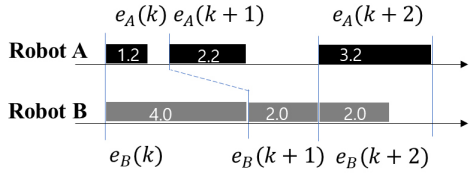
(a) Example of exceptional case



(b) Time-modification at k



(c) Time-modification at $k+1$



(d) New time-modification using non-zero $d_A(k)$ at k

[Fig. 10] Handling of exceptional case

인 방법을 제안한 바 있다[7]. 충돌 검사의 단순화 목적과 유전자를 조작하는 방법으로 인하여 두 로봇이 작업을 끝내고 이동하는 시각을 동기화 해야만 했다. 따라서, 이 방법을 통해 구한 두 로봇의 작업 시퀀스는 각 로봇이 대기하는 시간이 필연적으로 존재하게 된다. 본 논문은 기존 작업 시퀀스의 동기화된 움직임을 부분적으로 비동기화 하여 두 로봇의 대기시간을 줄일 수 있는 방법을 수학적으로 모델링하고 전체 시간을 줄일 수 있는 방법을 제안하였다. 이를 위해 작업 시퀀스를 이동시간

과 대기시간으로 새롭게 표현하였으며, 대기시간을 이용한 시간 단축 조건을 제시하고, 점진적으로 전체 시간을 단축시킬 수 있는 알고리즘을 제안하였다. 또한, 기존 유전자 알고리즘을 이용하여 구한 유사 최적해 80개에 대해 이 방법을 적용하였을 때, 0.651 s에서 1.887 s까지 시간 단축을 확인할 수 있었으며, 기존 최적해의 최고 성능 값 24.081 s 보다 1.511 s 빠른 최적해를 구하여 본 논문에서 제안한 알고리즘의 유효성을 검증하였다.

Yoon[7] 등은 충돌과 대기시간을 고려하지 않았을 때의 작업 시퀀스의 성능을 21.06 s로 보고하였다. 이 값은 실제 물리적으로 불가능한 값이지만 해의 성능을 비교하는 정보로 사용할 수 있다. 21.06 s을 전체 작업 시간의 가상의 하한 값이라고 할 때, 이전 연구에서 구한 최적 값 24.081 s와 차이는 3.021 s이다. 본 논문에서 제안한 알고리즘을 통해 구한 최고 성능 22.570 s는 기존 연구의 값 24.081 s을 1.511 s 단축시킨 것이고, 이 값을 시간 단축의 마진 3.021 s와 비교해 보면 성능 향상은 50%로 계산된다. 따라서 성능 향상 비율만을 고려할 때, 성능 향상 1.511 s는 유의미하다고 생각할 수 있다.

본 논문에서 제안한 알고리즘은 작업 시퀀스의 k와 $k+1$ 단계만을 고려하여 시간을 단축한 것이다. 만일 k 단계에서 앞의 몇 단계를 더 고려한다면 시간을 더 단축할 수도 있을 것이다. 이것에 대해서는 향후에 연구를 진행할 예정이다. 본 논문에서 대상으로 한 작업은 검사, 점 용접, 삽입 등과 같이 작업 위치에서 시간이 모두 동일한 경우만을 대상으로 하였는데, 과수 수확 같은 경우는 매 위치마다 작업 시간이 다를 수도 있다. 이것에 대해서는 시퀀스를 작업시간과 이동시간, 대기시간으로 표현하여 나타낼 수 있으며, 이렇게 표현된 작업 시퀀스의 최적화에 관한 연구는 진행 중에 있으며, 최종적으로는 실제적인 농업 분야의 작업에 본 논문에서 제안한 방법을 적용하여 유용성을 검증할 예정이다. 또한, 농업 분야의 응용과는 별도로 본 연구팀은 Yoon[9] 등이 보고한 바와 같이 기존 연구에서 제안한 방법이 3대 이상 로봇의 작업 계획으로 일반화가 가능한 지 연구를 진행하고 있다.

References

[1] T. Green, *SRI Spins off Abundant Robotics & Vacuum Robot Harvester*,[Online],https://www.roboticsbusinessreview.com/agriculture/sri-spins-off-abundant-robotics-vacuum-robot-harvester/, Accessed: Feb. 10, 2020.

[2] K. Tanigaki, T. Fujiura, A. Akase, and J. Imagawa, "Cherry-harvesting robot," *Computers and Electronics in Agriculture*, vol. 63, no. 1, pp. 65-72, August, 2008, DOI: 10.1016/j.compag.2008.01.018.

[3] E. J. Van Henten, D. A. Van't Slot, C. W. J. Hol, and L. G. Van Willigenburg, "Optimal manipulator design for a cucumber harvesting robot," *Computers and Electronics in Agriculture*, vol. 65, no. 2, pp. 247-257, March, 2009, DOI: 10.1016/j.compag.2008.11.004.

[4] T. T. Nguyen, E. Kayacan, J. De Baedemaeker, and W. Saeys, "Task and motion planning for apple harvesting robot," *IFAC Proceedings Volumes*, vol. 46, no. 18, pp. 247-252, August, 2013, DOI: 10.3182/20130828-2-SF-3019.00063.

[5] Y. Zhao, L. Gong, C. Liu, and Y. Huang, "Dual-arm robot design and testing for harvesting tomato in greenhouse," *IFAC-Papers Online*, vol. 49, no. 16, pp. 161-165, 2016, DOI: 10.1016/j.ifacol.2016.10.030.

[6] J. R. Davidson, C. J. Hohimer, C. Mo, and M. Karkee, "Dual robot coordination for apple harvesting," in *2017 ASABE Annual International Meeting, American Society of Agricultural and Biological Engineers*, pp. 1-9, 2017, DOI: 10.13031/aim.201700567.

[7] H. J. Yoon, S. Y. Chung, and M. J. Hwang, "Shadow space modeling and task planning for collision-free cooperation of dual manipulators for planar task," *International Journal of Control, Automation and Systems*, vol. 17, no. 4, pp. 995-1006, 2019, DOI: 10.1007/s12555-018-0236-1.

[8] H. J. Yoon, S. Y. Chung, and M. J. Hwang, "Shadow space modeling for task planning of dual manipulators," *2017 14th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, Jeju, Korea, pp. 749-750, 2017, DOI: 10.1109/URAI.2017.7992816.

[9] H. J. Yoon, M. J. Hwang, and S. Y. Chung, "A Collision Modeling Method for Multiple Manipulators Task Planning Using Shadow Space," *Journal of Institute of Control, Robotics and Systems*, vol. 25, no. 3, pp. 249-254, 2019, DOI: 10.5302/J.ICROS.2019.18.0127.

정 성 엽

1994 연세대학교 기계공학과(공학사)
1996 KAIST 기계공학과(공학석사)
2005 KAIST 기계공학과(공학박사)
2007 삼성중공업 메카트로닉스센터 책임연구원
2007~현재 한국교통대학교 기계공학전공 교수

관심분야: 양팔 로봇, 머신 비전, 지능 로봇

황 면 중

2001 KAIST 기계공학과(공학사)
2003 KAIST 기계공학과(공학석사)
2007 KAIST 기계공학과(공학박사)
2008~2009 Case Western Reserve University, Research Associate
2010~2013 삼성전자 생산기술연구소 책임연구원
2013~2015 한라대학교 기계자동차공학부 조교수
2015~현재 한국교통대학교 기계공학전공 부교수

관심분야: 양팔 로봇 시스템, 로봇 모션 및 제어