http://dx.doi.org/10.7735/ksmte.2016.25.6.504

J. Korean Soc. Manuf. Technol. Eng.
ISSN 2508-5107(Online) / ISSN 2508-5093(Print)

Special Issue : Next generation Industrial Robot

# 요청한 작업 경로에 따른 매니퓰레이터의 기구학적 변수 선정을 위한 군집 지능 기반 최적 설계

이준우[a*]

# Swarm Intelligence-based Optimal Design for Selecting the Kinematic Parameters of a Manipulator According to the Desired Task Space Trajectory

Joonwoo Lee[a*]

[a] *Department of Electrical Engineering, Kyungpook National University (KNU), 80, Daehak-ro, Buk-gu, Daegu 41566, Korea*

**ARTICLE INFO**

*Article history:*
Received 18 October 2016
Revised 21 November 2016
Accepted 24 November 2016

*Keywords:*
Manipulator design
Kinematic parameters optimization
Swarm intelligence
Ant colony optimization (ACO)
Particle swarm optimization (PSO)

**ABSTRACT**

Robots are widely utilized in many fields, and various demands need customized robots. This study proposes an optimal design method based on swarm intelligence for selecting the kinematic parameter of a manipulator according to the task space trajectory desired by the user. The optimal design method is dealt with herein as an optimization problem. This study is based on swarm intelligence-based optimization algorithms (i.e., ant colony optimization (ACO) and particle swarm optimization algorithms) to determine the optimal kinematic parameters of the manipulator. The former is used to select the optimal kinematic parameter values, whereas the latter is utilized to solve the inverse kinematic problem when the ACO determines the parameter values. This study solves a design problem with the PUMA 560 when the desired task space trajectory is given and discusses its results in the simulation part to verify the performance of the proposed design.

## 1. 서 론

로봇은 현재 다양한 분야에서 폭넓게 활용되고 있으며, 이미 우리의 생활 속 깊이, 가까운 곳에서 어렵지 않게 접할 수 있게 되었다. 이러한 폭넓은 활용 및 응용으로 인해, 다양한 형태의 로봇 개발에 대한 요구가 증가하고 있다. 또한 각종 로봇 부품들에 대한 제조 단가가 저렴해짐에 따라, 사용자 요구 중심의 주문 제작 로봇의 필요성도 날로 증가하고 있다. 제조업에 있어서도, 과거에는 소품종 대량생산 중심의 제조공정이 중요하였지만, 최근에는 소비자의 요구에 맞춘 다품종 소량생산 중심의 제조공정의 중요성이 커지고 있다. 아울러 이러한 제조공정을 로봇을 활용하여 자동화 하고자 하는 움직임이 함께 일어나면서, 사용자 요구 중심의 주문 제작 로봇의 필요성이 증대되고 있다.

본 논문은 이러한 산업환경 변화에 대비하는 사용자가 요구하는 작업 공간(task-space) 내의 경로(trajectory)를 추종할 수 있는 매니퓰레이터(manipulator)의 설계에 있어, 기구학적인 변수(kinematic parameters)들의 최적 선정을 위한 설계 기법에 대해 논하고자 한다. 본 논문은 이러한 매니퓰레이터의 기구학적인 최적 변수 선정 문제를 최적화 문제로 정의하고, 이를 군집 지능(SI: swarm intelligence)에 기반한 최적화 알고리즘들을 도입하여 해결하고자 한다.

본 논문에서의 매니퓰레이터 최적 설계의 방법을 요약하면 다음

* Corresponding author. Tel.: +82-53-950-5602
Fax: +82-53-950-6600
*E-mail address:* jwl@knu.ac.kr (Joonwoo Lee).

과 같다. 먼저, 사용자가 필요로 하는 작업의 경로를 작업 공간에서 경유점들의 집합으로 생성하고, 이 들 각각의 경유점들에서의 작업 공간 상의 목표점 및 목표 접근 각도와 실제 도달점 및 도달 각도 사이의 차이, 즉, 도달성 에러(reachability error)를 최소화하며, 각 경유점에서의 조작성(manipulability)은 최대가 되도록 하는 매니퓰레이터의 기구학적 변수들의 값들을 선정한다. 이러한 최적 변수 선정은 조합 최적화(combinatorial optimization) 문제 풀이에 적합한 개미 군집 최적화(ACO: ant colony optimization) 알고리즘을 통해 문제를 해결하고자 한다.

또한 본 문제에서는 임의의 변수 값들의 조합에 의해서 구성된 매니퓰레이터들이 목표 경유점에 도달할 수 있는지의 여부를 판단하기 위해, 관절 공간 상에서 만들어지는 여러 조합들을 작업 공간의 실제 도달점으로 변환하여 그 차이를 확인해야 하는 부분이 존재하게 된다. 이러한 변환의 문제를 역기구학(inverse kinematic) 문제라 하는데, 이는 매번 닫힌 형태(closed-form)의 해를 찾을 수는 없기에 수치적인 해법으로의 접근이 필요하다. 이를 위해 본 논문에서는 연속 최적화(continuous optimization) 문제 풀이에 적합한 입자 군집 최적화(PSO: particle swarm optimization) 알고리즘을 사용한다. PSO 알고리즘은 ACO 알고리즘에 의해 구성된 후보 매니퓰레이터를 가지고, 작업 공간 상에 주어진 각 경유점에서의 도달성 에러는 최소가 되도록 하고, 관절 공간(joint-space) 상에서는 이웃하는 점들 사이의 관절 각도 변화량이 최소가 되도록 하는(이는 매니퓰레이터의 작업 공간상의 경로 추정을 부드럽게 함) 관절 경유점들을 찾고, 이를 바탕으로 ACO 알고리즘이 스스로 구성한 후보 매니퓰레이터의 적합성 판단 시에 활용할 수 있는 정보를 제공한다. 이러한 일련의 과정을 통해, 사용자가 요청한 작업 경로에 따른 매니퓰레이터의 기구학적인 변수를 선정하여, 사용자가 원하는 작업이 가능한 최적의 매니퓰레이터를 설계한다.

이후의 본 논문은 다음과 같이 구성된다. 다음 절에서는 본 논문에서 다룰 최적화 문제를 풀기 위해 필요한 배경 지식들을 정리하며, 3절에서는 본 논문에서 제안하는 군집 지능 기반 매니퓰레이터의 기구학적 설계 방법을 소개한다. 4절에서는 제안한 군집 지능 기반 최적 설계법의 타당성 검증을 위한 시뮬레이션을 수행하고, 그 결과에 대해 논하며, 마지막으로 결론을 맺는다.

## 2. 배경 지식

### 2.1 매니퓰레이터의 기구학적인 변수 설계

기구학(kinematics)이라 함은 어떤 시스템 본체에 가해지는 힘이나 그 자체의 무게 등은 고려하지 않고, 시스템 본체 혹은 그 시스템의 동작에 대해 다루는 역학의 한 부류 학문이다. 직렬연결(serial-link) 매니퓰레이터는 일련의 기계적인 연결부(link)들과 관절(joint)들로 구성되어 있는 로봇 팔(robot-arm)로써, 한쪽 끝은 베이스에 고정되어 있고 나머지 다른 한쪽 끝에는 작업 수행을 위한 말단장치부(end-effector) 혹은 도구(tool)가 부착되어 있는 형태이다[1].

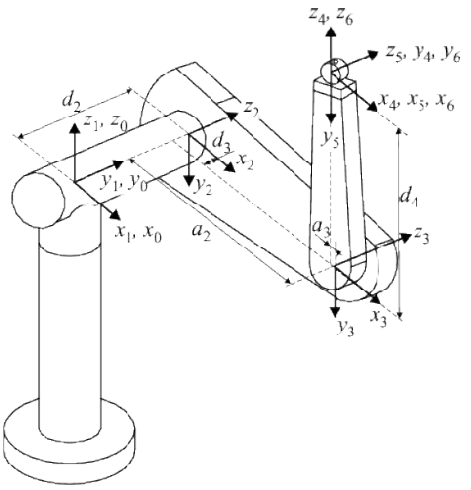



**Fig. 1 Kinematic parameters and frame assignments for the PUMA 560 manipulator[2]**

**Table 1 Denavit-Hartenberg parameters: their physical meaning, symbol, and formal definition[1]**

| | | | |
|---|---|---|---|
| Joint angle | $\theta_j$ | The angle between the $x_{j-1}$ and $x_j$ axes about the $z_{j-1}$ axis | Revolute joint variable |
| Link offset | $d_j$ | The distance from the origin of frame $j$-1 to the $x_j$ axis along the $z_{j-1}$ axis | Prismatic joint variable |
| Link length | $a_j$ | The distance between the $z_{j-1}$ and $z_j$ axes along the $x_j$ axis; for intersecting axes is parallel to $\hat{z}_{j-1} \times \hat{z}_j$ | Constant |
| Link twist | $\alpha_j$ | The angle from the $z_{j-1}$ axis to the $z_j$ axis about the $x_j$ axis | Constant |
| Joint type | $\sigma_j$ | $\sigma$ =0 for a revolute joint, $\sigma$=1 for a prismatic joint | Constant |

Fig. 1은 대표적인 직렬연결 매니퓰레이터인 UNIMATE사의 PUMA 560이며, 매니퓰레이터의 기구학적인 변수들과 좌표계를 나타내고 있다. PUMA 560은 6개의 회전관절(revolute joint)로 구성되어 있는 선구자격인 로봇 팔로써, 현재의 대부분 매니퓰레이터들은 이러한 PUMA 560의 형태를 하고 있는 경우가 많다. 이러한 직렬연결 매니퓰레이터를 체계적으로 표현하기 위해 1955년에 Denavit와 Hartenberg에 의해 제안된 표기법을 사용한다. Table 1은 Denavit-Hartenberg 변수들과 그들의 물리적인 의미, 기호, 공식적인 정의들을 보여준다. Table 2는 PUMA 560의 Denavit-Hartenberg 표기법으로 나타낸 것이다.

본 논문에서는 PUMA 560의 연결부 오프셋 $d_j$, 연결부 길이 $a_j$,

**Table 2 Denavit-Hartenberg parameters of the PUMA 560[1]**

| $j$ | $\theta_j$ | $d$ | $a$ | $\alpha$ |
|---|---|---|---|---|
| 1 | $q_1$ | 0 | 0 | 1.571 |
| 2 | $q_2$ | 0 | 0.4318 | 0 |
| 3 | $q_3$ | 0.15 | 0.0203 | -1.571 |
| 4 | $q_4$ | 0.4318 | 0 | 1.571 |
| 5 | $q_5$ | 0 | 0 | -1.571 |
| 6 | $q_6$ | 0 | 0 | 0 |

연결부 뒤틀림 $\alpha_j$의 세 가지 변수를 변화시키면서 생성되는 후보 매니퓰레이터들 중 사용자가 요구하는 작업 공간 경로를 가장 좋은 성능으로 추종할 수 있는 매니퓰레이터를 설계하는 문제를 다루고자 한다.

## 2.2 개미 군집 최적화(ACO) 알고리즘

개미 군집 최적화 알고리즘[3]은 대표적인 조합 최적화 문제를 풀 수 있는 알고리즘으로, 개미의 먹이 수집 과정에서 영감을 얻어 개발된 알고리즘이다. 실제 개미들은 먹이를 찾아서 먹이를 찾은 곳으로부터 둥지로 돌아오는 길에 페르몬을 길 위에다 뿌리면서 둥지로 향한다. 이를 다른 개미들이 쫓아감으로써 여러 개미들이 먹이가 있는 곳에서부터 둥지까지 먹이를 운반할 가능성이 생긴다. 비록 페르몬은 일정 시간이 지나면 증발하지만, 점점 많은 개미들이 지난 간 곳, 즉, 최단 경로에는 그만큼 페르몬의 양이 많기 때문에 상대적으로 그 최단 경로를 따르는 개미 수는 점차 늘어난다. 또한 그 경로 위의 페르몬 양도 상대적으로 증가하면서 개미들은 먹이가 있는 곳에서부터 둥지까지 최적의 경로, 즉, 최단 경로를 찾게 되는 것이다.

**Algorithm 1** Ant Colony Optimization (ACO)

```
SetParameters(N_A, α, β, ρ);    // N_A: # of Ants
Initialize(T);    // Initialize pheromone trails
t ← 0;    // t: Iteration Counter
while Termination criteria do not meet do
    t ← t + 1;
  ▼▼ AntBasedSolutionConstruction Step ▼▼
    AntBasedSolutionConstruction();    // Roulette Selection with probabilities by Eq. (1)
  ▼▼ DaemonActions Step ▼▼
    {optional}
  ▼▼ PheromoneUpdate Step ▼▼
    PheromoneUpdate();    // Pheromone Update with Eq. (2)
end while
```

Algorithm 1은 이러한 원리에 기반한 일반적인 ACO 알고리즘의 전반적인 동작을 나타낸 것이다. 먼저, ACO 알고리즘의 변수들인 개미의 총 수 $N_A$, 페르몬 정보에 대한 가중치 변수 $\alpha$, 휴리스틱 정보에 대한 가중치 변수 $\beta$, 그리고 페르몬 증발 비율 변수 $\rho$를 선정한다. 다음으로 매 반복마다 개미들에 의해 얻은 페르몬 정보를 저장할 곳을 초기화 한 후, 이 정보들을 바탕으로 개미가 이동을 시작한다. 개미가 현재의 노드 $i$에서 주변의 다음 노드 $j$로 경로 이동을 하게 되는 확률은 다음의 식을 통해서 정해진다.

$$p\left(c_{ij}\middle|\mathbf{x}^p\right)=\frac{\left(\tau_{ij}\right)^{\alpha}\cdot\left(\eta_{ij}\right)^{\beta}}{\sum_{c_{il}\in N(\mathbf{x}^p)}\left(\tau_{il}\right)^{\alpha}\cdot\left(\eta_{il}\right)^{\beta}},\quad \forall c_{ij}\in\mathbf{N}(\mathbf{x}^p) \tag{1}$$

여기서 $c_{ij}$는 노드 $i$와 노드 $j$ 사이의 부분경로이며, $\tau_{ij}$는 부분경로 $c_{ij}$에 대한 페르몬 정보량이고, $\eta_{ij}$는 부분경로 $c_{ij}$에 대한 휴리스틱 정보량이다. 이러한 개미의 이동 과정을 ant-based solution construction step이라고 한다. 이러한 과정이 끝나면, 필요에 따라 추가적인 처리과정인 daemon actions step을 거치게 되고, 그 다음으로 이번 반복과정 중에 새롭게 생성된 정보들을 페르몬 정보 저장소에 반영하는 과정, 즉, pheromone update step을 수행하게 된다. 페르몬 정보 업데이트는 다음의 수식을 통해 실시된다.

$$\tau_{ij}\leftarrow\begin{cases}(1-\rho)\tau_{ij}+\rho\Delta\tau, & \text{if } \tau_{ij}\in\mathbf{x}_{good}\\ (1-\rho)\tau_{ij}, & \text{otherwise}\end{cases} \tag{2}$$

먼저 페르몬 증발 비율 변수 $\rho$를 통해 실제의 페로몬 증발처럼 이전의 정보들의 일부를 증발시키고, 이후 개미들이 생성한 경로들을 평가하여, 평가 순위가 높은 순서대로 $\Delta\tau$를 차등적으로 더해준다. 이러한 일련의 과정은 알고리즘의 최종 종료 조건을 만족할 때까지 반복하며, 최종적으로 최적의 해를 도출하게 된다.

## 2.3 입자 군집 최적화(PSO) 알고리즘

입자 군집 최적화 알고리즘[4,5]은 연속 최적화 문제를 풀기 위해 널리 사용되고 있는 알고리즘으로, 새 무리나 물고기 군집의 사회적 행동에 기반하여 개발된 알고리즘이다. PSO 알고리즘은 입자(particle)라 불리는 최적해 후보군과 이들 입자들의 위치와 속도에 관계된 아주 간단한 수식에 기반하여 탐색영역을 움직이는 이들의 이동에 기반하여 최적의 해를 도출한다.

**Algorithm 2** Particle Swarm Optimization (PSO)

```
SetParameters(N_P, n, w, φ_p, φ_g);    // N_P: # of Particles, n: Dimensions
Initialize(x);    // Initialize particles' position x
p ← Evaluate(x_1, ··· , x_{N_P});    // Evaluate initial position and set them as p_best of each particle
g_1 ← Best(p, 1)    // Select the best solution among p_best s as g_best
while Termination criteria do not meet do
    // Generate N_P new solutions
    for l = 1 to N_P do
        // A particle begins to move to a new position
        for i = 1 to n do
            r_p, r_g ~ U(0, 1);    // Determine parameters r_p, r_g
            v_l^i ← w v_l^i + φ_p r_p (p_l^i − x_l^i) + φ_g r_g (g^i − x_l^i);    // Update particle's velocity for each dimension
        end for
        x_l ← x_l + v_l;    // Move to a position with velocity v_l
        if f(x_l) < f(p_l) then
            p_l ← x_l;    // Update p_best
            if f(p_l) < f(g_1) then
                g_1 ← p_l;    // Update g_best
            end if
        end if
    end for
end while
```

Algorithm 2는 PSO 알고리즘의 수학적 모델과 해 탐색 과정을 보여준다. 먼저, PSO 알고리즘의 변수들인 총 입자의 수 $N_p$, 관성 가중치 $w$, 두개의 가속 상수 $\varphi_p$, $\varphi_g$를 선정한다. 다음으로 각각의 입자의 위치와 속도를 초기화하게 되는데, 시간 $k$일 때의 입자 $l$의 위치와 속도는 $\mathbf{x}_l(k)$과 $\mathbf{v}_l(k)$의 벡터로 표기한다. 또한 벡터 $\mathbf{p}_l(k)$는 입자 $l$의 현재시간 $k$까지 가장 좋은 위치를 저장하며, 벡터 $\mathbf{g}_1(k)$는 전체 입자들 중 가장 좋은 위치를 저장하는 벡터이다. 이들을 각각 "$p_{best}$"와 "$g_{best}$"라고 한다.

알고리즘 초기의 입자들의 위치는 탐색 영역 내에서 균일 분포 랜덤 수를 통해 초기화하며, 그 위치에서 출발을 한다. 입자들의 이동은 다음의 수식들을 통해 이루어진다. 먼저, 입자 $l$의 $i$번째 차원의 속도 $v_l^i(k+1)$는 다음 수식으로 결정된다.

$$v_l^i(k+1) = wv_l^i(k) + \varphi_p r_p \left(p_l^i(k) - x_l^i(k)\right) + \varphi_g r_g \left(g_1^i(k) - x_l^i(k)\right) \tag{3}$$

여기서 $r_p$, $r_g$는 0과 1사이의 균일 분포 랜덤 수이며, 관성 가중치 $w$는 입자들의 이동에 대한 관성의 정도를 결정한다. 식 (3)의 두 번째 항은 "인지(cognition)"부분이라 불리며, 입자들의 독립적인 행동을 표현하는 항이다. 세 번째 항은 "사회(social)"부분이라 하며, 입자들 사이의 협력을 표현하는 항이다. 이 두 개의 항에 곱해진 상수들 $\varphi_p$, $\varphi_g$는 각각 인지부분과 사회부분의 상대적인 영향력을 결정한다. 이렇게 식 (3)을 통해 얻어진 속도를 통해 입자 $l$은 다음 위치를 다음 수식을 통해 결정한다.

$$\mathbf{x}_l(k+1) = \mathbf{x}_l(k) + \mathbf{v}_l(k+1) \tag{4}$$

이러한 이동의 과정이 종료되며, 각 입자들은 평가를 통해 각자의 $p_{best}$를 갱신하고, 전체 입자들 중의 최고의 평가를 받은 입자를 $g_{best}$로 재선한다. 이러한 일련의 과정은 알고리즘의 최종 종료 조건을 만족할 때까지 반복하며, 최종적으로 최적의 해를 도출하게 된다.

## 3. 군집 지능 기반 매니퓰레이터의 기구학적 설계

### 3.1 매니퓰레이터의 기구학적 최적 설계 문제 정의

본 논문은 매니퓰레이터의 기구학적인 최적 변수 선정 문제를 최적화 문제로 정의하고, 이를 군집 지능에 기반한 최적화 알고리즘들을 도입하여 풀고자 한다. 이에 앞서 매니퓰레이터의 기구학적인 변수 선정 문제를 아래와 같은 최적화 문제로 정의한다.

For PSO algorithm:

$$\begin{aligned}\textbf{minimize}\ & (\text{Reachability Error in Task-Space}) \\ & -(\text{Reachability}) \\ & +(\text{Angular Variation in Joint-Space}) \\ & -(\text{Manipulability})\end{aligned} \tag{5}$$

For ACO algorithm:

$$\begin{aligned}\textbf{minimize}\ & \sum_{n_{Td}=1}^{N_{Td}} \{(\text{Reachability Error in Task-Space}) \\ & -(\text{Reachability}) \\ & +(\text{Angular Variation in Joint-Space}) \\ & -(\text{Manipulability})\}\end{aligned} \tag{6}$$

**Algorithm 3** Swarm-Intelligence-Based Optimal Design of Manipulator

```
SetParameters(e);   // e: Tolerance for reachability
SetParameters(N_A, α, β, ρ, N_P, n, w, φ_p, φ_g);  // For ACO, PSO
Initialize(T_a, T_d, T_α);   // Initialize pheromone trails
Calculate(H_a, H_d, H_α);   // Calculate heuristic information
▼▼ ACO Algorithm Part ▼▼
while Termination criteria do not meet do
    a, d, α ← SolutionConstruction(T_a, T_d, T_α, H_a, H_d, H_α); // Roulette Selection with probabilities by Eq. (1)
    ▼▼ PSO Algorithm Part ▼▼
    for n_Td = 1 to N_Td do
        Initialize(x);   // Initialize particles' position x
        p ← Evaluate(x_1, ··· , x_NP);   // Evaluate initial position and set them as p_best of each particle
        g_1 ← Best(p, 1)   // Select the best solution among p_best s as g_best
        while Termination criteria do not meet do
            // Generate N_P new solutions
            for l = 1 to N_P do
                // A particle begins to move to a new position
                for i = 1 to n do
                    r_p, r_g ~ U(0, 1);   // Determine parameters r_p, r_g
                    v_l^i ← w v_l^i + φ_p r_p (p_l^i − x_l^i) + φ_g r_g (g^i − x_l^i);   // Update particle's velocity for each dimension
                end for
                x_l ← x_l + v_l;   // Move to a position with velocity v_l
                if f(x_l) < f(p_l) then
                    p_l ← x_l;   // Update p_best
                    if f(p_l) < f(g_1) then
                        g_1 ← p_l;   // Update g_best
                    end if
                end if
            end for
        end while
    end for
    T_a, T_d, T_α ← PheromoneUpdate(T_a, T_d, T_α);   // Pheromone Update with Eq. (2)
end while
```

본 논문에서는 매니퓰레이터의 기구학적 변수 선정 문제를 두 종의 서로 다른 특징을 가진 최적화 알고리즘으로 메인 문제인 변수 최적화 문제 풀이와 서브 문제인 역기구학 문제 풀이의 이중 루프로 풀어내는 구조를 제안한다. PSO 알고리즘이 내부 루프의 역기구학 문제를 최적화 문제로써 수치해석법으로 접근하여 풀기 위해 식 (5)을 통해 찾은 후보해들을 평가한다. 사용자가 요구하는 작업 경로의 경유점 집합이 주어지면, PSO 알고리즘은 각 경유점에 대해, 그 경유점을 목표점으로 하여, 작업 공간상의 목표점 및 목표 접근 각도와 PSO 알고리즘에 의해 찾은 후보해의 도달점 및 도달 각도 사이의 차이(reachability error in task-space)를 최소화하고, 주어진 허용 오차(tolerance) 범위내에 도달점이 도달

했는지의 여부(reachability, 0은 미달, 1은 도달)를 최대화, 관절 공간 내에서 이전 도달점에서 현 도달점까지의 이동에 있어 각도의 변화량(angular variation in joint-space)은 최소화(이는 매니퓰레이터의 작업 공간상의 경로 추정을 부드럽게 함), 경유점에서의 조작성(manipulability)[6]은 최대가 되도록 하는 최적의 해를 찾는다. 식 (5)의 각 항은 정규화(normalization) 과정을 거쳐 각 항목의 값 분포가 0과 1 사이에 존재하도록 만든 후, 그 값들의 합(최소화 항목)과 차(최대화 항목)를 최소화하도록 최적화 문제를 정의한다.

이렇게 각 경유점에 대해, ACO 알고리즘이 구성해 준 후보 매니퓰레이터를 가지고 PSO 알고리즘이 평가를 마치면, 경유점의 개수만큼 모아진 정보를 다시 식 (6)과 같이 합산하여 ACO 알고리즘이 최적의 매니퓰레이터 설계하는데 활용한다. 이때도 마찬가지로 식 (6)의 각 항은 0과 1의 범위의 값을 갖도록 정규화의 과정(엄밀히, 평균값 계산)을 거친 결과값을 사용한다.

### 3.2 군집 지능 기반 최적 설계 기법

앞서 언급했듯이, 본 논문은 매니퓰레이터의 기구학적인 최적 변수 선정 문제를 최적화 문제로 정의하고, 이를 군집 지능에 기반한 최적화 알고리즘들, 즉, ACO 알고리즘과 PSO 알고리즘으로 풀고자 한다. 최적 변수 선정 문제는 조합 최적화의 문제로 이를 풀기에 적합한 ACO 알고리즘을 통해 문제를 해결하며, 역기구학 문제는 연속 최적화 문제로써 이에 강점을 가지고 널리 활용되고 있는 PSO 알고리즘을 통해 문제를 풀고자 한다.

Algorithm 3은 본 논문에서 제안하는 군집 지능 기반 최적 설계 문제 해결을 위한 군집 지능 기반 최적화 알고리즘을 보여준다. 먼저, 역기구학 문제에서 도달성 허용오차 $e$를 선정하고, ACO와 PSO 알고리즘의 변수들을 선정한다. 또한 매번 ACO 알고리즘에 의해 찾아진 매니퓰레이터의 기구학적 변수들, 즉, Denavit-Hartenberg 변수들 $a$, $d$, $\alpha$에 대한 페로몬 정보 저장소 $\mathbf{T}_a$, $\mathbf{T}_d$, $\mathbf{T}_\alpha$를 초기화 한다. 그리고 ACO 알고리즘의 휴리스틱 정보로써, 본 논문에서는 일반적으로 널리 알려진 매니퓰레이터(본 논문에서는 PUMA 560을 사용함)의 기구학적 변수값들과의 차이 정보를 활용하며, 이 정보를 휴리스틱 정보 저장소 $\mathbf{H}_a$, $\mathbf{H}_d$, $\mathbf{H}_\alpha$에 저장하고 활용한다. 이때의 휴리스틱 정보 또한 같은 크기 기준을 가지고 정보 활용에 반영하고자 정보값이 0과 1사이의 범위를 갖도록 정규화 과정을 거친다. 이렇게 하여 ACO 알고리즘의 첫 단계인 후보해들을 생성한다. 이때에는 식 (1)과 같이 페르몬 정보와 휴리스틱 정보에 대해 지수 가중치를 두어 계산된 각 변수값들의 선택에 대한 확률을 활용하여 룰렛 선택을 실시한다. 이렇게 ACO 알고리즘에 의해 만들어진 최적 후보 매니퓰레이터의 정보를 바탕으로 PSO 알고리즘은 사용자가 요구한 작업 공간 상의 경로를 추종하기에 적합한 매니퓰레이터인지 여부를 평가하기 위한 역기구학 문제 풀이를 시작한다. 사용자가 요구한 작업 공간 상의 경유점들 각각에 대해 식 (5)의 평가함수 적용을 위한 정보들을 수집하고, 이를 바탕으로 역기구학 문제의 최적해를 찾는다. 이렇게 찾아진 모든 경유점에 대한 정보를 바탕으로 ACO 알고리즘은 자신이 찾은 후보 매니퓰레이터를 평가하고, 이를 페로몬 정보 저장소에 정보 갱신을 하게 된다. 이러한 일련의 과정은 최대 평가함수 호출 횟수(MaxNEFs: maximum number of function evaluations)를 기준으로 알고리즘의 종료 여부를 결정하게 되는데, PSO 알고리즘은 특별히 도달성 허용오차 범위 내에 도달점이 들어오게 되면, 비록 최대 평가함수 호출 횟수에 도달하지 않더라도 PSO 알고리즘의 도달성 여부 판단을 종료하는 조건을 추가한다.

## 4. 군집 지능 기반 최적 설계 시뮬레이션 결과

이 장에서는 제안한 군집 지능 기반 최적 설계법의 검증을 위해 Fig. 2와 같은 사용자 요구 경로를 가지고 최적 매니퓰레이터 설계 시뮬레이션을 실시한다. 요구 경로의 총 경유점의 개수 $N_{Td}$는 21이며, 시작점에서의 좌표 및 각도는 $trans(2, -3, 1)\times trotx(\pi)$이며, 끝점에서는 $trans(1, 1, 2)\times trotx(\pi/2)$이다.

시뮬레이션 환경은 Intel® Core™ i7-5930K CPU @ 3.50 GHz와 32 GB의 램을 가진 컴퓨터에 Windows 10 기반으로 모든 시뮬레이터의 구현은 MATLAB 환경에서 구현하였다. 또한 매니퓰레이터를 구현하고, 시뮬레이션하기 위해 Peter Corke의 Robotics Toolbox for MATLAB Release 9[7]를 사용하였다. 각각의 알고리즘에 대한 변수 선정은 다음과 같이 이루어졌다;

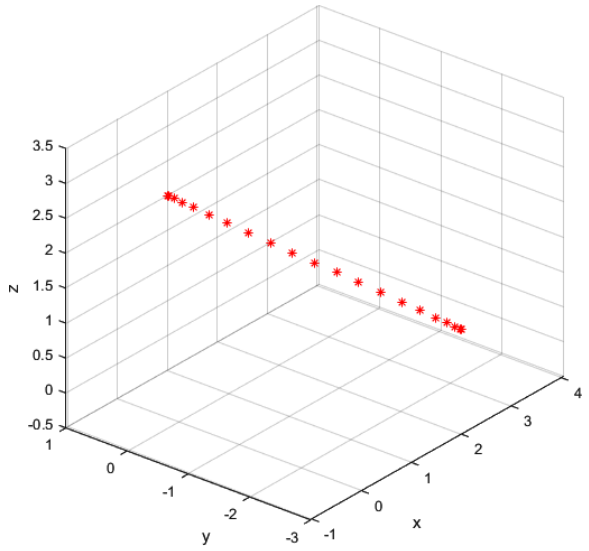
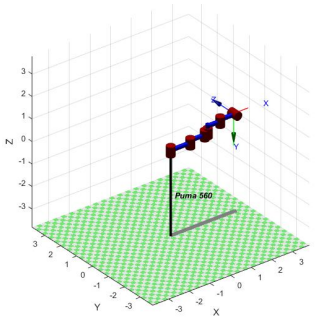
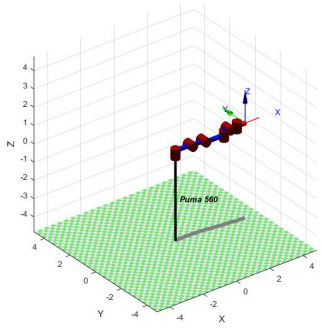


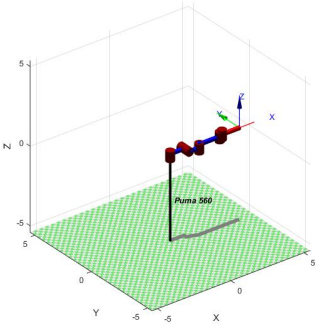

**Fig. 2 An example of the task-space trajectory desired by user**



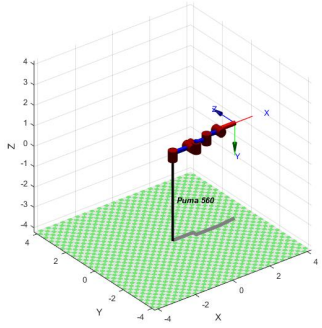

(a) The best at *NEFs_ACO*=900 (Evaluation value=0.0876)



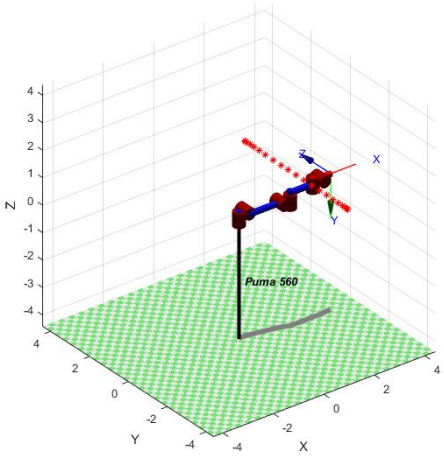

(b) The best at *NEFs_ACO*=2700 (Evaluation value=0.0448)




(c) The best at *NEFs_ACO*=4500 (Evaluation value=0.0158)




(d) The best at *NEFs_ACO*=6300 (Evaluation value=0.0019)




(e) Optimal design for the desired trajectory (*NEFs_ACO*=10000, Evaluation value=-0.0207)

**Fig. 3 The result of the manipulator design desired by user**

**Table 3 Denavit-Hartenberg parameters of the optimal manipulator designed by swarm intelligence-based optimizer**

| $j$ | $\theta_j$ | $d$ | $a$ | $\alpha$ |
|---|---|---|---|---|
| 1 | $q_1$ | 0.18 | 0.25 | 1.571 |
| 2 | $q_2$ | 0.24 | 1.2 | 0 |
| 3 | $q_3$ | 0.12 | 0.3 | -1.571 |
| 4 | $q_4$ | 0.3 | 0.9 | 0 |
| 5 | $q_5$ | 0.09 | 0.3 | -1.571 |
| 6 | $q_6$ | 0.05 | 0.45 | 0 |

$N_A$=30, $\alpha$=2, $\beta$=1, $\rho$=0.2, $N_p$=50, $w$=0.729, $\varphi_p$=$\varphi_g$=1.4945. 또한 군집 지능 기반 최적 설계 알고리즘에 관한 변수들의 선정은 다음과 같다.

$$e = 0.05,\ \ MaxNFEs\_ACO = 10^4,\ \ MaxNFEs\_PSO = 5\times10^4.$$

Fig. 3은 군집 지능 기반 최적 설계 알고리즘을 통해 찾아진 후보 매니퓰레이터 4종 및 최종적으로 주어진 사용자 요구 경로의 작업 수행에 적합한 최적 매니퓰레이터를 보여준다. Fig. 3에서 보여주듯이, 알고리즘이 순차적으로 진행됨에 따라 매니퓰레이터의 구조가 개선되어 감을 평가함수값 및 눈으로 확인할 수 있다. 작업 공간의 경로가 y축의 0를 기준으로 음의 방향쪽으로 기울어져 있으므로, 그 부분에 대한 도달성을 높이기 위해 차츰 매니퓰레이터의 3, 4번 관절에 연결부 오프셋 $d$가 추가되어감을 확인할 수 있다. 또한 작업 공간의 경로에 따라 상하좌우 앞뒤로의 움직임을 개선하기 위해 연결부 뒤틀림 $\alpha$가 개선되어가는 것도 확인할 수 있다. Table 3은 본 논문에서 제안한 최적 변수 선정 과정을 통해 얻어진 최종 매니퓰레이터의 기구학적 최적 변수 값들을 나타낸 표이다.

## 5. 결 론

본 논문은 다품종 소량생산 중심의 제조공정으로의 산업환경 변화에 대비하는 사용자가 요구하는 작업 공간에서의 경로를 추종하며 작업을 수행할 수 있는 매니퓰레이터의 설계에 있어, 기구학적인 변수들의 최적 선정을 위한 설계 기법에 대해 논하였다. 본 논문은 이러한 매니퓰레이터의 기구학적인 최적 변수 선정 문제를 조합 및 연속 최적화 문제로 정의하고, 이를 군집 지능(SI: swarm intelligence)에 기반한 최적화 알고리즘들, 즉, 최적 변수 선정을 위한 조합 최적화 문제 해결을 위한 ACO 알고리즘과 연속 최적화 문제인 역기구학 문제를 해결하기 위한 PSO 알고리즘을 도입하여 해결하는 이중 루프 최적 설계법을 제안하였다. 실제 요구 작업 경로 추종 문제를 갖고 검증 시뮬레이션을 실시하였고, 그 결과는 충분히 납득할 만한 설계 결과를 확인할 수 있었다.

추후 연구 과제로서 많은 시간이 소요되고 있는 현재의 알고리즘을 개선하는 부분과 기본적인 ACO, PSO 알고리즘보다 성능이 뛰어난 파생 알고리즘들을 도입하여 전체적인 알고리즘 성능 향상에 기여하는 부분이 남아있다.

## References

[1] Corke, P., 2011, Robotics, Vision and Control: Fundamental

Algorithms in MATLAB, Springer, Berlin.

[2] Lavalle, S. M., viewed 20 Apr. 2012, The Homogeneous Transformation Matrix, <http://planning.cs.uiuc.edu/node111.html>.

[3] Dorigo, M., Gambardella, L. M., 1997, Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem, IEEE Trans. Evol. Comput., 1:1 53-66.

[4] Kennedy, J., Eberhart, R., 1995, Particle Swarm Optimization, IEEE Int. Conf. Neural Netw., 1942-1948.

[5] Shi, Y., Eberhart, R., 1995, A Modified Particle Swarm Optimizer, IEEE World Congr. Evol. Comput., 5 69-73.

[6] Vahrenkamp, N., Asfour, T., Metta, G., Sandino, G., Dillmann, R., 2012, Manipulability Analysis, IEEE-RAS Int. Conf. Humanoid Robots, 568-573.

[7] Corke, P., n.d., Robotics Toolbox, viewed 27 Oct. 2012, <http://www.petercorke.com/Robotics_Toolbox.html>.