



(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2017년11월09일
(11) 등록번호 10-1794776
(24) 등록일자 2017년11월01일

(51) 국제특허분류(Int. Cl.)

G06F 17/17 (2006.01)

(52) CPC특허분류

G06F 17/17 (2013.01)

(21) 출원번호 10-2016-0067617

(22) 출원일자 2016년05월31일

심사청구일자 2016년05월31일

(56) 선행기술조사문헌

KR101523848 B1*

KR101072448 B1*

*는 심사관에 의하여 인용된 문헌

(73) 특허권자

성균관대학교산학협력단

경기도 수원시 장안구 서부로 2066 (천천동, 성균관대학교내)

동아대학교산학협력단

부산광역시 사하구 낙동대로550번길 37, 동아대학교 내 (하단동)

조선대학교산학협력단

광주광역시 동구 필문대로 309 (서석동)

(72) 발명자

정상용

서울특별시 서초구 신반포로23길 41, 106동 308호

이진환

경기도 수원시 팔달구 화산로 57, 142동 403호

(뒷면에 계속)

(74) 대리인

심경식, 홍성욱

전체 청구항 수 : 총 18 항

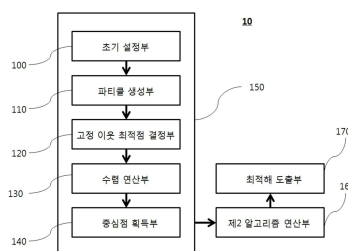
심사관 : 홍경아

(54) 발명의 명칭 최적해 도출 장치 및 최적해 도출 방법

(57) 요약

본 발명의 최적해 도출 장치는, 대상 함수에 대한 d 차원의 최적해에 대응하는 d 차원의 변수를 위치 값 및 속도 값으로 갖는 복수의 파티클(particle)을 복수 회 수렴시킴으로써 각각의 위치 값을 갖는 복수의 중심점 (centroid)을 획득하는 제1 알고리즘 연산부; 상기 복수의 중심점 각각에 대응하고, 각각의 위치 값을 갖는 복수의 후보점을 획득하는 제2 알고리즘 연산부; 및 상기 복수의 후보점 중 상기 대상 함수에 대한 최소 비용을 갖는 후보점의 위치 값을 최적해로서 도출하는 최적해 도출부를 포함한다.

대표도 - 도1



(72) 발명자

송준영

경기도 수원시 장안구 화산로259번길 38, 205호

김종욱

부산광역시 사하구 제석로79번길 11, 103동 403호

김용재

광주광역시 북구 천변우로79번길 25, 202동 901호

이 발명을 지원한 국가연구개발사업

과제고유번호 1415140194

부처명 산업통상자원부

연구관리전문기관 한국산업기술평가관리원

연구사업명 그린카등수송시스템산업핵심기술개발

연구과제명 (P)HEV용 구동모터의 통합형 최적설계 및 FMEA기반 설계검증 플랫폼 개발

기 여 율 1/1

주관기관 성균관대학교산학협력단

연구기간 2015.06.01 ~ 2016.05.31

공지예외적용 : 있음

명세서

청구범위

청구항 1

대상 함수에 대한 d 차원의 최적해에 대응하는 d 차원의 변수를 위치 값 및 속도 값으로 갖는 복수의 파티클 (particle)을 복수 회 수렴시킴으로써 각각의 위치 값을 갖는 복수의 중심점(centroid)을 획득하는 제1 알고리즘 연산부;

상기 복수의 중심점 각각에 대응하고, 각각의 위치 값을 갖는 복수의 후보점을 획득하는 제2 알고리즘 연산부; 및

상기 복수의 후보점 중 상기 대상 함수에 대한 최소 비용을 갖는 후보점의 위치 값을 최적해로서 도출하는 최적해 도출부를 포함하고,

상기 제1 알고리즘 연산부는

상기 복수의 파티클 중 하나의 파티클을 고정 이웃 최적점(fixed neighbor best point)으로 결정하는 고정 이웃 최적점 결정부;

상기 복수의 파티클을 상기 고정 이웃 최적점으로 수렴시키는 수렴 연산부; 및

상기 고정 이웃 최적점과 근접한 적어도 2 개의 파티클 및 상기 고정 이웃 최적점에 대한 중심점을 획득하는 중심점 획득부를 포함하는,

최적해 도출 장치.

청구항 2

삭제

청구항 3

제1 항에 있어서,

상기 제1 알고리즘 연산부는

파티클의 개수와 계수 값을 설정하는 초기 설정부; 및

임의의 위치 값과 속도 값을 갖는 상기 복수의 파티클을 생성하는 파티클 생성부를 더 포함하고,

상기 파티클 생성부, 상기 고정 이웃 최적점 결정부, 상기 수렴 연산부, 및 상기 중심점 획득부를 반복적으로 이용함으로써 상기 복수의 중심점을 획득하는,

최적해 도출 장치.

청구항 4

제3 항에 있어서,

상기 초기 설정부는 반경(radius) 값을 더 설정하고,

상기 중심점 획득부는 상기 고정 이웃 최적점을 기준으로 상기 반경 값 내에 상기 적어도 2 개의 파티클이 포함되는 경우, 상기 고정 이웃 최적점과 상기 적어도 2 개의 파티클이 구성하는 도형에 대한 상기 중심점을 획득하는,

최적해 도출 장치.

청구항 5

제1 항에 있어서,

상기 고정 이웃 최적점 결정부는 상기 복수의 파티클 각각에 대해 상기 대상 함수에 대해 산출된 비용 값들을 이용해서 최소 비용을 갖는 파티클을 상기 고정 이웃 최적점으로 결정하는,

최적해 도출 장치.

청구항 6

제5 항에 있어서,

상기 고정 이웃 최적점 결정부는 상기 최소 비용이 임계 값을 초과하는 경우, 상기 최소 비용을 갖는 파티클을 후보 이웃 최적점으로 우선적으로 설정하고,

상기 복수의 파티클을 상기 후보 이웃 최적점으로 이동시키면서 발견된 더 낮은 비용 값을 갖는 파티클을 상기 후보 이웃 최적점으로 재설정하고,

재설정된 상기 후보 이웃 최적점의 비용 값이 상기 임계 값 이하가 되는 경우, 상기 후보 이웃 최적점을 상기 고정 이웃 최적점으로 결정하는,

최적해 도출 장치.

청구항 7

제1 항에 있어서,

상기 수렴 연산부는 상기 고정 이웃 최적점의 위치 값을 반영하여 상기 복수의 파티클 각각의 위치 값 및 속도 값을 반복 업데이트시키고,

상기 반복 업데이트 과정에서 상기 고정 이웃 최적점의 값은 고정된 위치 값인,

최적해 도출 장치.

청구항 8

제3 항에 있어서,

상기 제1 알고리즘 연산부는 기존의 중심점에 대해서 일정 간격 이상 이격된 새로운 고정 이웃 최적점을 발견하지 못하는 경우, 제1 알고리즘을 종료하는,

최적해 도출 장치.

청구항 9

제1 항에 있어서,

상기 제1 알고리즘 연산부는 제1 알고리즘으로서 수정된 PSO(Particle Swarm Optimization) 알고리즘을 이용하는,

최적해 도출 장치.

청구항 10

제1 항에 있어서,

상기 제2 알고리즘 연산부는 제2 알고리즘으로서 MADS(Mesh Adaptive Direct Search) 알고리즘을 이용하는,

최적해 도출 장치.

청구항 11

대상 함수에 대한 d 차원의 최적해에 대응하는 d 차원의 변수를 위치 값 및 속도 값으로 갖는 복수의 파티클을 복수 회 수렴시킴으로써 각각의 위치 값을 갖는 복수의 중심점을 획득하는 제1 알고리즘 연산 단계;

상기 복수의 중심점 각각에 대응하고, 각각의 위치 값을 갖는 복수의 후보점을 획득하는 제2 알고리즘 연산 단계; 및

상기 복수의 후보점 중 상기 대상 함수에 대한 최소 비용을 갖는 후보점의 위치 값을 최적해로서 도출하는 최적해 도출 단계를 포함하고,

상기 제1 알고리즘 연산 단계는

상기 복수의 파티클 중 하나의 파티클을 고정 이웃 최적점으로 결정하는 고정 이웃 최적점 결정 단계;

상기 복수의 파티클을 상기 고정 이웃 최적점으로 수렴시키는 수렴 연산 단계; 및

상기 고정 이웃 최적점과 근접한 적어도 2 개의 파티클 및 상기 고정 이웃 최적점에 대한 중심점을 획득하는 중심점 획득 단계를 포함하는,

최적해 도출 방법.

청구항 12

삭제

청구항 13

제11 항에 있어서,

상기 제1 알고리즘 연산 단계는

파티클의 개수와 계수 값을 설정하는 초기 설정 단계;

임의의 위치 값과 속도 값을 갖는 상기 복수의 파티클을 생성하는 파티클 생성 단계; 및

상기 파티클 생성 단계, 상기 고정 이웃 최적점 결정 단계, 상기 수렴 연산 단계, 및 상기 중심점 획득 단계를 반복적으로 수행함으로써 상기 복수의 중심점을 획득하는 복수의 중심점 획득 단계를 더 포함하는,

최적해 도출 방법.

청구항 14

제13 항에 있어서,

상기 초기 설정 단계에서, 반경 값을 더 설정하고,

상기 중심점 획득 단계에서, 상기 고정 이웃 최적점을 기준으로 상기 반경 값 내에 상기 적어도 2 개의 파티클이 포함되는 경우, 상기 고정 이웃 최적점과 상기 적어도 2 개의 파티클이 구성하는 도형에 대한 상기 중심점을 획득하는,

최적해 도출 방법.

청구항 15

제11 항에 있어서,

상기 고정 이웃 최적점 결정 단계에서, 상기 복수의 파티클 각각에 대해 상기 대상 함수에 대해 산출된 비용 값을 이용해서 최소 비용을 갖는 파티클을 상기 고정 이웃 최적점으로 결정하는,

최적해 도출 방법.

청구항 16

제15 항에 있어서,

상기 고정 이웃 최적점 결정 단계에서,

상기 최소 비용이 임계 값을 초과하는 경우, 상기 최소 비용을 갖는 파티클을 후보 이웃 최적점으로 우선적으로 설정하고,

상기 복수의 파티클을 상기 후보 이웃 최적점으로 이동시키면서 발견된 더 낮은 비용 값을 갖는 파티클을 상기 후보 이웃 최적점으로 재설정하고,

재설정된 상기 후보 이웃 최적점의 비용 값이 상기 임계 값 이하가 되는 경우, 상기 후보 이웃 최적점을 상기 고정 이웃 최적점으로 결정하는,

최적해 도출 방법.

청구항 17

제11 항에 있어서,

상기 수렴 연산 단계에서, 상기 고정 이웃 최적점의 위치 값을 반영하여 상기 복수의 파티클 각각의 위치 값 및 속도 값을 반복 업데이트시키고,

상기 반복 업데이트 과정에서 상기 고정 이웃 최적점의 값은 고정된 위치 값인,

최적해 도출 방법.

청구항 18

제13 항에 있어서,

상기 제1 알고리즘 연산 단계에서, 기존의 중심점에 대해서 일정 간격 이상 이격된 새로운 고정 이웃 최적점을 발견하지 못하는 경우, 제1 알고리즘을 종료하는,

최적해 도출 방법.

청구항 19

제11 항에 있어서,

상기 제1 알고리즘 연산 단계에서, 제1 알고리즘으로서 수정된 PSO 알고리즘을 이용하는,

최적해 도출 방법.

청구항 20

제11 항에 있어서,

상기 제2 알고리즘 연산 단계에서, 제2 알고리즘으로서 MADS 알고리즘을 이용하는,

최적해 도출 방법.

발명의 설명

기술 분야

[0001] 본 발명은 최적해 도출 장치 및 최적해 도출 방법에 관한 것이다.

배경 기술

[0002] PSO(Particle Swarm Optimization) 알고리즘은 전역 최적화(global optimization) 문제를 해결하기 위한 알고리즘이다. PSO 알고리즘은 사회적 행동의 시뮬레이션(simulation of social behavior)을 위해 J. Kennedy에 의해 최초로 제안되었으며, 최적화 알고리즘으로서는 1995년에 처음으로 제안되었다.

[0003] PSO 알고리즘은 각 에이전트(agent)의 메모리 및 통신 능력에 의해서 멀티모드 비용 함수(multimodal cost functions)에 대한 장점을 갖고 있다. 그러나 PSO 알고리즘은 수렴(convergence)에 있어서 국부 최적화 방법들(local optimization methods)보다 더 긴 계산 시간이 필요하다는 단점이 있다.

[0004] 이러한 PSO 알고리즘의 단점을 개선하기 위해서 여러가지 수정 버전이 제안되고 있으며, 논문 "A particle swarm optimization algorithm with novel expected fitness evaluation for robust optimization problems"은 RPSO(Robust PSO) 알고리즘을 제안한다.

선행기술문헌

비특허문헌

- [0005] (비특허문헌 0001) F. Luan, J.-H. Choi, and H.-K. Jung, "A particle swarm optimization algorithm with novel expected fitness evaluation for robust optimization problems," IEEE Trans. Magn., vol. 48, no. 2, pp. 331?334, Feb. 2012.

발명의 내용

해결하려는 과제

- [0006] 해결하고자 하는 기술적 과제는, 기존 PSO 알고리즘보다 함수 호출 횟수를 감소시키고 계산 시간을 최소화하여 최적해를 도출하는 최적해 도출 장치 및 최적해 도출 방법을 제공하는 데 있다.

과제의 해결 수단

- [0007] 본 발명의 한 실시예에 따른 최적해 도출 장치는, 대상 함수에 대한 d 차원의 최적해에 대응하는 d 차원의 변수를 위치 값 및 속도 값으로 갖는 복수의 파티클(particle)을 복수 회 수렴시킴으로써 각각의 위치 값을 갖는 복수의 중심점(centroid)을 획득하는 제1 알고리즘 연산부; 상기 복수의 중심점 각각에 대응하고, 각각의 위치 값을 갖는 복수의 후보점을 획득하는 제2 알고리즘 연산부; 및 상기 복수의 후보점 중 상기 대상 함수에 대한 최소 비용을 갖는 후보점의 위치 값을 최적해로서 도출하는 최적해 도출부를 포함한다.
- [0008] 상기 제1 알고리즘 연산부는 상기 복수의 파티클 중 하나의 파티클을 고정 이웃 최적점(fixed neighbor best point)으로 결정하는 고정 이웃 최적점 결정부; 상기 복수의 파티클을 상기 고정 이웃 최적점으로 수렴시키는 수렴 연산부; 및 상기 고정 이웃 최적점과 근접한 적어도 2 개의 파티클 및 상기 고정 이웃 최적점에 대한 중심점을 획득하는 중심점 획득부를 포함할 수 있다.
- [0009] 상기 제1 알고리즘 연산부는 파티클의 개수와 계수 값을 설정하는 초기 설정부; 및 임의의 위치 값과 속도 값을 갖는 상기 복수의 파티클을 생성하는 파티클 생성부를 더 포함하고, 상기 파티클 생성부, 상기 고정 이웃 최적점 결정부, 상기 수렴 연산부, 및 상기 중심점 획득부를 반복적으로 이용함으로써 상기 복수의 중심점을 획득할 수 있다.
- [0010] 상기 초기 설정부는 반경(radius) 값을 더 설정하고, 상기 중심점 획득부는 상기 고정 이웃 최적점을 기준으로 상기 반경 값 내에 상기 적어도 2 개의 파티클이 포함되는 경우, 상기 고정 이웃 최적점과 상기 적어도 2 개의 파티클이 구성하는 도형에 대한 상기 중심점을 획득할 수 있다.
- [0011] 상기 고정 이웃 최적점 결정부는 상기 복수의 파티클 각각에 대해 상기 대상 함수에 대해 산출된 비용 값들을 이용해서 최소 비용을 갖는 파티클을 상기 고정 이웃 최적점으로 결정할 수 있다.
- [0012] 상기 고정 이웃 최적점 결정부는 상기 최소 비용이 임계 값을 초과하는 경우, 상기 최소 비용을 갖는 파티클을 후보 이웃 최적점으로 우선적으로 설정하고, 상기 복수의 파티클을 상기 후보 이웃 최적점으로 이동시키면서 발견된 더 낮은 비용 값을 갖는 파티클을 상기 후보 이웃 최적점으로 재설정하고, 재설정된 상기 후보 이웃 최적점의 비용 값이 상기 임계 값 이하가 되는 경우, 상기 후보 이웃 최적점을 상기 고정 이웃 최적점으로 결정할 수 있다.
- [0013] 상기 수렴 연산부는 상기 고정 이웃 최적점의 위치 값을 반영하여 상기 복수의 파티클 각각의 위치 값 및 속도 값을 반복 업데이트시키고, 상기 반복 업데이트 과정에서 상기 고정 이웃 최적점의 값은 고정된 위치 값일 수 있다.
- [0014] 상기 제1 알고리즘 연산부는 기존의 중심점에 대해서 일정 간격 이상 이격된 새로운 고정 이웃 최적점을 발견하지 못하는 경우, 제1 알고리즘을 종료할 수 있다.
- [0015] 상기 제1 알고리즘 연산부는 제1 알고리즘으로서 수정된 PSO(Particle Swarm Optimization) 알고리즘을 이용할 수 있다.
- [0016] 상기 제2 알고리즘 연산부는 제2 알고리즘으로서 MADS(Mesh Adaptive Direct Search) 알고리즘을 이용할 수 있

다.

- [0017] 본 발명의 한 실시예에 따른 최적해 도출 방법은, 대상 함수에 대한 d 차원의 최적해에 대응하는 d 차원의 변수를 위치 값 및 속도 값으로 갖는 복수의 파티클을 복수 회 수렴시킴으로써 각각의 위치 값을 갖는 복수의 중심점을 획득하는 제1 알고리즘 연산 단계; 상기 복수의 중심점 각각에 대응하고, 각각의 위치 값을 갖는 복수의 후보점을 획득하는 제2 알고리즘 연산 단계; 및 상기 복수의 후보점 중 상기 대상 함수에 대한 최소 비용을 갖는 후보점의 위치 값을 최적해로서 도출하는 최적해 도출 단계를 포함한다.
- [0018] 상기 제1 알고리즘 연산 단계는 상기 복수의 파티클 중 하나의 파티클을 고정 이웃 최적점으로 결정하는 고정 이웃 최적점 결정 단계; 상기 복수의 파티클을 상기 고정 이웃 최적점으로 수렴시키는 수렴 연산 단계; 및 상기 고정 이웃 최적점과 근접한 적어도 2 개의 파티클 및 상기 고정 이웃 최적점에 대한 중심점을 획득하는 중심점 획득 단계를 포함할 수 있다.
- [0019] 상기 제1 알고리즘 연산 단계는 파티클의 개수와 계수 값을 설정하는 초기 설정 단계; 임의의 위치 값과 속도 값을 갖는 상기 복수의 파티클을 생성하는 파티클 생성 단계; 및 상기 파티클 생성 단계, 상기 고정 이웃 최적점 결정 단계, 상기 수렴 연산 단계, 및 상기 중심점 획득 단계를 반복적으로 수행함으로써 상기 복수의 중심점을 획득하는 복수의 중심점 획득 단계를 더 포함할 수 있다.
- [0020] 상기 최적해 도출 방법은 상기 초기 설정 단계에서, 반경 값을 더 설정하고, 상기 중심점 획득 단계에서, 상기 고정 이웃 최적점을 기준으로 상기 반경 값 내에 상기 적어도 2 개의 파티클이 포함되는 경우, 상기 고정 이웃 최적점과 상기 적어도 2 개의 파티클이 구성하는 도형에 대한 상기 중심점을 획득할 수 있다.
- [0021] 상기 최적해 도출 방법은 상기 고정 이웃 최적점 결정 단계에서, 상기 복수의 파티클 각각에 대해 상기 대상 함수에 대해 산출된 비용 값들을 이용해서 최소 비용을 갖는 파티클을 상기 고정 이웃 최적점으로 결정할 수 있다.
- [0022] 상기 최적해 도출 방법은 상기 고정 이웃 최적점 결정 단계에서, 상기 최소 비용이 임계 값을 초과하는 경우, 상기 최소 비용을 갖는 파티클을 후보 이웃 최적점으로 우선적으로 설정하고, 상기 복수의 파티클을 상기 후보 이웃 최적점으로 이동시키면서 발견된 더 낮은 비용 값을 갖는 파티클을 상기 후보 이웃 최적점으로 재설정하고, 재설정된 상기 후보 이웃 최적점의 비용 값이 상기 임계 값 이하가 되는 경우, 상기 후보 이웃 최적점을 상기 고정 이웃 최적점으로 결정할 수 있다.
- [0023] 상기 최적해 도출 방법은 상기 수렴 연산 단계에서, 상기 고정 이웃 최적점의 위치 값을 반영하여 상기 복수의 파티클 각각의 위치 값 및 속도 값을 반복 업데이트시키고, 상기 반복 업데이트 과정에서 상기 고정 이웃 최적점의 값은 고정된 위치 값일 수 있다.
- [0024] 상기 최적해 도출 방법은 상기 제1 알고리즘 연산 단계에서, 기존의 중심점에 대해서 일정 간격 이상 이격된 새로운 고정 이웃 최적점을 발견하지 못하는 경우, 제1 알고리즘을 종료할 수 있다.
- [0025] 상기 최적해 도출 방법은 상기 제1 알고리즘 연산 단계에서, 제1 알고리즘으로서 수정된 PSO 알고리즘을 이용할 수 있다.
- [0026] 상기 최적해 도출 방법은 상기 제2 알고리즘 연산 단계에서, 제2 알고리즘으로서 MADS 알고리즘을 이용할 수 있다.

발명의 효과

- [0027] 본 발명에 따른 최적해 도출 장치 및 최적해 도출 방법은 기존 PSO 알고리즘보다 함수 호출 횟수를 감소시키고 계산 시간을 최소화하여 최적해를 도출할 수 있다.

도면의 간단한 설명

- [0028] 도 1은 본 발명의 한 실시예에 따른 최적해 도출 장치를 설명하기 위한 도면이다.
- 도 2는 파티클 생성부에 의해 생성된 복수의 파티클을 설명하기 위한 도면이다.
- 도 3은 고정 이웃 최적점 결정부에 의한 후보 이웃 최적점의 결정을 설명하기 위한 도면이다.
- 도 4는 고정 이웃 최적점 결정부에 의한 고정 이웃 최적점의 결정과 수렴 연산부에 의한 복수의 파티클의 수렴 과정을 설명하기 위한 도면이다.

도 5는 중심점 획득부에 의한 중심점의 획득 과정을 설명하기 위한 도면이다.

도 6 및 7은 복수의 중심점의 획득 과정을 설명하기 위한 도면이다.

도 8 및 9는 예시적인 대상 함수의 최적해를 도출하는 데 있어서, 기존 PSO 알고리즘, RPSO 알고리즘, 및 본 발명의 한 실시예에 따른 최적해 도출 방법의 성능 비교를 설명하기 위한 도면이다.

도 10 내지 16은 본 발명의 한 실시예에 따른 최적해 도출 방법을 IPMSM의 최적 디자인 설계에 이용하는 케이스를 설명하기 위한 도면이다.

발명을 실시하기 위한 구체적인 내용

- [0029] 이하, 첨부한 도면을 참고로 하여 본 발명의 여러 실시 예들에 대하여 본 발명이 속하는 기술 분야에서 통상의 지식을 가진 자가 용이하게 실시할 수 있도록 상세히 설명한다. 본 발명은 여러 가지 상이한 형태로 구현될 수 있으며 여기에서 설명하는 실시 예들에 한정되지 않는다.
- [0030] 본 발명을 명확하게 설명하기 위해서 설명과 관계없는 부분은 생략하였으며, 명세서 전체를 통하여 동일 또는 유사한 구성요소에 대해서는 동일한 참조 부호를 붙이도록 한다. 따라서 앞서 설명한 참조 부호는 다른 도면에서도 사용할 수 있다.
- [0031] 또한, 도면에서 나타난 각 구성의 크기 및 두께는 설명의 편의를 위해 임의로 나타내었으므로, 본 발명이 반드시 도시된 바에 한정되지 않는다. 도면에서 여러 층 및 영역을 명확하게 표현하기 위하여 두께를 과장되게 나타낼 수 있다.
- [0032] 도 1은 본 발명의 한 실시예에 따른 최적해 도출 장치를 설명하기 위한 도면이다.
- [0033] 도 1을 참조하면, 본 발명의 한 실시예에 따른 최적해 도출 장치(10)는 제1 알고리즘 연산부(150), 제2 알고리즘 연산부(160), 및 최적해 도출부(170)를 포함한다.
- [0034] 제1 알고리즘 연산부(150)는 초기 설정부(100), 파티클 생성부(110), 고정 이웃 최적점 결정부(120), 수렴 연산부(130), 중심점 획득부(140)를 포함할 수 있다.
- [0035] 최적해 도출 장치(10)는 적어도 하나의 프로세서와 메모리로 구성되는 범용 컴퓨팅 장치일 수 있다. 컴퓨팅 장치는 휴대성에 따라 데스크탑(desktop), 랩탑(laptop), 스마트폰(smart phone), 태블릿(tablet) 등으로 구분되기도 하고, 연산 능력에 따라 퍼스널 컴퓨터(personal computer)와 서버(server)로 구분되기도 하고, 분담된 역할에 따라 데이터베이스 서버, 연산 서버 등으로 구분되기도 하지만, 적어도 본 발명의 한 실시예에 따른 최적해 도출 알고리즘을 프로그램으로 구동할 수 있는 연산 능력을 갖춘 컴퓨팅 장치는 모두 본 발명의 한 실시예에 따른 최적해 도출 장치(10)가 될 수 있다. 즉, 최적해 도출 장치(10)는 하드디스크(HDD), SSD(Solid State Disk), 이동식 저장매체 등의 저장매체에 기록된 최적해 도출 프로그램을 연산할 수 있는 컴퓨팅 장치일 수 있다.
- [0036] 본 실시예에서는 최적해 도출 장치(10)를 복수의 기능부로 구분하고 있지만, 이는 설명의 편의를 위한 것으로서 당업자라면 복수의 기능부를 통합하여 하나의 기능부로 구성하거나, 하나의 기능부를 분할하여 복수의 기능부로 구성할 수도 있다.
- [0037] 또한 각각의 기능부는 하드웨어적으로 구분되거나, 또는 하드웨어는 공유하되 소프트웨어적으로 구분될 수 있으며, 이를 어떻게 구현하는 지는 당업자의 선택에 따라 달라질 수 있다.
- [0038] 본 발명의 한 실시예에 따른 최적해 도출 장치(10)는 대상 함수에 대한 d 차원의 최적해에 대응하는 d 차원의 변수를 위치 값 및 속도 값으로 갖는 복수의 파티클(particle)을 복수 회 수렴시킴으로써 각각의 위치 값을 갖는 복수의 중심점(centroid)을 획득하는 제1 알고리즘 연산부(150); 복수의 중심점 각각에 대응하고, 각각의 위치 값을 갖는 복수의 후보점을 획득하는 제2 알고리즘 연산부(160); 및 복수의 후보점 중 대상 함수에 대한 최소 비용을 갖는 후보점의 위치 값을 최적해로서 도출하는 최적해 도출부(170)를 포함한다.
- [0039] 제1 알고리즘 연산부(150)는 제1 알고리즘으로서 수정된 PSO 알고리즘을 이용할 수 있다. 이하에서 제1 알고리즘 연산부(150)를 이용하여 상세히 설명될 제1 알고리즘은 ePSO(explorative Particle Swarm Optimization)로 명명될 수 있다.
- [0040] 제2 알고리즘 연산부(160)는 제2 알고리즘으로서 MADS(Mesh Adaptive Direct Search) 알고리즘을 이용할 수 있

다. MADS의 구체적인 알고리즘에 대해서는 후술한다.

- [0041] 제1 알고리즘 연산부(150)와 제2 알고리즘 연산부(160)를 거쳐 최종적으로 선택된 복수의 후보점은 최적해 도출부(170)를 통해서 하나의 후보점만이 남게되고, 이러한 하나의 후보점이 대상 함수에 대한 최적해로 선택된다.
- [0042] 도 2는 파티클 생성부에 의해 생성된 복수의 파티클을 설명하기 위한 도면이고, 도 3은 고정 이웃 최적점 결정부에 의한 후보 이웃 최적점의 결정을 설명하기 위한 도면이고, 도 4는 고정 이웃 최적점 결정부에 의한 고정 이웃 최적점의 결정과 수렴 연산부에 의한 복수의 파티클의 수렴 과정을 설명하기 위한 도면이고, 도 5는 중심점 획득부에 의한 중심점의 획득 과정을 설명하기 위한 도면이고, 도 6 및 7은 복수의 중심점의 획득 과정을 설명하기 위한 도면이다.
- [0043] 이하에서 도 1의 각 기능부를 설명함에 있어서, 도 2 내지 7을 참조하여 설명한다.
- [0044] 초기 설정부(100)는 파티클(particle)의 개수와 계수 값을 설정할 수 있다. 복수의 계수 값은 후술하는 수학적식에서 이용하는 w , c_1 , c_2 를 포함할 수 있다. w 는 관성 중량(inertia weight)이고, c_1 및 c_2 는 가속 계수(acceleration coefficients)이다. 이러한 복수의 계수 값은 사용자로부터 입력 받은 값을 이용할 수도 있다.
- [0045] 한 실시예에 따른, 초기 설정부(100)는 반경(radius) 값을 더 설정할 수 있다. 반경 값에 대해서는 도 5를 참조하여 후술한다.
- [0046] 파티클 생성부(110)는 임의의 위치 값과 속도 값을 갖는 복수의 파티클을 생성한다. 이때 복수의 파티클의 개수는 초기 설정부(100)에서 미리 정한 파티클의 개수에 대응할 수 있다.
- [0047] 현재 단계에서, 파티클 각각은 임의의 위치 값과 속도 값을 갖는다. 각 파티클의 위치 값은 최적해 도출 장치(10)에서 구하고자 하는 최적해일 수도 있고, 최적해 부근의 값일 수도 있고, 최적해와 전혀 상관없는 값일 수도 있다. 따라서 본 발명의 실시예에서는 이러한 파티클의 위치 값을 어떻게 효율적으로 최적해에 근사시키는지에 대한 방법을 제시한다.
- [0048] 파티클의 속도 값은 파티클의 다음 위치 값과 현재 위치 값의 차이를 나타낸다. 즉, 수학적식을 이용하여 후술하는 바와 같이, 파티클의 현재 위치 값에 속도 값을 더하면, 파티클의 다음 위치 값이 도출되고, 이러한 위치 값을 업데이트할 때 파티클이 이동하였다고 표현한다.
- [0049] 도 2를 참조하면, 임의의 위치 값을 갖는 복수의 파티클이 탐색 공간(20)에 퍼져있음을 확인할 수 있다. 도 2 내지 7은 설명의 용이성을 위하여 복수의 파티클 및 그 이동 방향을 그림으로 도시하였으나, 이는 설명을 위한 것으로서 각 파티클은 데이터로서만 존재한다.
- [0050] 대상 함수에 대하여 구하고자 하는 최적해가 d 차원인 경우, 각 파티클은 d 차원의 변수를 위치 값 및 속도 값으로 갖는다. 각 파티클이 d 차원을 갖는다고 할 때, i 번째 파티클의 위치 값을 아래 수학적식 1과 같이 벡터 값 X_i 로 나타낼 수 있다. 또한 i 번째 파티클의 속도 값을 아래 수학적식 2와 같이 벡터 값 V_i 로 나타낼 수 있다.
- [0051] [수학적식 1]
- [0052]
$$X_i = (x_i^1, x_i^2, x_i^3, \dots, x_i^d)$$
- [0053] [수학적식 2]
- [0054]
$$V_i = (v_i^1, v_i^2, v_i^3, \dots, v_i^d)$$
- [0055] 고정 이웃 최적점 결정부(120)는 복수의 파티클 중 하나의 파티클을 고정 이웃 최적점(fixed neighbor best point)으로 결정한다.
- [0056] 고정 이웃 최적점(fixed neighbor best point)이란 본 발명에서 도입한 위치 값으로서, 복수의 파티클 각각에 대해 대상 함수에 대해 산출된 비용 값들 중 최소 비용을 갖는 파티클이 이러한 고정 이웃 최적점으로 결정될 수 있다. 대상 함수로부터 도출하고자 하는 최적해에 근접할수록 파티클의 비용은 낮게 계산될 수 있다.
- [0057] 각각의 파티클의 위치 값을 직접 대상 함수에 대입하여 그 결과 값에 따라 각 파티클의 비용을 결정할 수 있다. 또는 각각의 알고리즘에 적합하도록 대상 함수에 대응하는 비용 함수를 작성하고, 이러한 비용 함수에 각각의 파티클의 위치 값을 대입함으로써 각 파티클의 비용을 획득할 수도 있다.
- [0058] 이러한 대상 함수 또는 비용 함수는 각각의 현실적인 케이스마다 개별적으로 제공될 필요가 있다. 도 8 내지

16을 참조하여 두 가지 예시적인 최적화 케이스를 후술한다.

[0059] 한 실시예로서, 고정 이웃 최적점 결정부(120)는 최소 비용이 임계 값을 초과하는 경우, 최소 비용을 갖는 파티클을 후보 이웃 최적점(candidate neighbor best point)으로 우선적으로 설정하고, 복수의 파티클을 후보 이웃 최적점으로 이동시키면서 발견된 더 낮은 비용 값을 갖는 파티클을 후보 이웃 최적점으로 재설정하고, 재설정된 후보 이웃 최적점의 비용 값이 임계 값 이하가 되는 경우, 후보 이웃 최적점을 고정 이웃 최적점으로 결정할 수 있다.

[0060] 즉, 고정 이웃 최적점 결정부(120)는 파티클 생성부(110)가 생성한 복수의 파티클 중 하나의 파티클을 단순히 고정 이웃 최적점으로 결정할 수도 있으나, 이러한 고정 이웃 최적점의 비용이 임계 값을 초과하는 경우 적절한 고정 이웃 최적점을 선택하기 위하여 적절한 알고리즘이 추가될 수 있다. 이때 임계 값을 결국 허용 오차 범위로 나타낼 수 있는데, 본 발명의 실시예에서 임계 값은 기존 PSO보다 허용 오차 범위가 더 클 수 있다.

[0061] 도 3을 참조하면, 파티클(P1)이 복수의 파티클 중 가장 낮은 비용을 가지고 있으나, 그 비용이 임계 값을 초과할 수 있다. 이때 파티클(P1)은 후보 이웃 최적점으로 우선적으로 설정될 수 있다. 파티클(P1)을 제외한 다른 복수의 파티클은 아래 수학식 3 및 4에 따라 파티클(P1)을 향해 이동(수렴)할 수 있다.

[0062] [수학식 3]

$$[0063] \quad \mathbf{v}_{i(n+1)}^d = \mathbf{w} * \mathbf{v}_{i(n)}^d + \mathbf{c}_1 * \text{rand}_1^d * (\mathbf{pBest}_i^d - \mathbf{x}_{i(n)}^d) + \mathbf{c}_2 * \text{rand}_2^d * (\mathbf{cnBest}_i^d - \mathbf{x}_{i(n)}^d)$$

[0064] [수학식 4]

$$[0065] \quad \mathbf{x}_{i(n+1)}^d = \mathbf{x}_{i(n)}^d + \mathbf{v}_{i(n+1)}^d$$

[0066] $\mathbf{w}$, $\mathbf{c}_1$, $\mathbf{c}_2$ 는 초기 설정부(100)에서 미리 설정한 값을 이용할 수 있고, rand_1^d 및 rand_2^d 는 n 번째 단계에서 $n+1$ 번째 단계로 넘어갈 때 마다 0 내지 1 사이의 값 중 임의로 선택될 수 있다. $\mathbf{pBest}_i^d$ 는 i 번째 파티클이 거처온 위치 값 중 최소 비용을 갖는 위치 값을 의미하며, $\mathbf{cnBest}_i^d$ 는 현재 후보 이웃 최적점으로 설정된 파티클(P1)의 위치 값을 의미한다. $\mathbf{pBest}_i^d$ 는 업데이트될 수 있다.

[0067] 수학식 3 및 4를 참조하면, i 번째 파티클은 n 번째 단계에서 위치 값 $\mathbf{x}_{i(n)}^d$ 및 속도 값 $\mathbf{v}_{i(n)}^d$ 을 갖고 있으나, $n+1$ 번째 단계에서 위치 값 $\mathbf{x}_{i(n+1)}^d$ 및 속도 값 $\mathbf{v}_{i(n+1)}^d$ 을 갖도록 업데이트된다. i 번째 파티클은 업데이트 됨에 따라 파티클(P1)을 향해 그 위치가 점차적으로 수렴하게 된다.

[0068] 상술한 과정을 통해 복수의 파티클은 파티클(P1)으로 이동하게 되는데, 이러한 과정에서 특정한 파티클의 위치 값에 대한 비용이 파티클(P1)의 비용보다 낮음을 발견할 수도 있다. 이러한 경우, 발견된 특정한 파티클이 파티클(P1)을 대신하여 후보 이웃 최적점으로 재설정될 수 있다. 즉, $\mathbf{cnBest}_i^d$ 가 업데이트될 수 있다.

[0069] 이때 재설정된 후보 이웃 최적점의 비용이 여전히 임계 값을 초과하는 경우, 상술한 과정이 되풀이될 수 있다. 만약 후보 이웃 최적점의 비용이 임계 값 이하가 되는 경우, 고정 이웃 최적점 결정부(120)는 이러한 후보 이웃 최적점을 고정 이웃 최적점으로 결정할 수 있다.

[0070] 고정 이웃 최적점은, 후술하는 바와 같이, 중심점을 획득하기까지 그 값이 고정되는 점에서, 유동적인 값을 갖는 후보 이웃 최적점과 차이가 있다.

[0071] 고정 이웃 최적점이 결정되면, 수렴 연산부(130)는 복수의 파티클을 고정 이웃 최적점(P2)을 향해서 이동시킨다(도 4 참조). 수렴 연산부(130)는 아래 수학식 5 및 6에 따라 i 번째 파티클을 고정 이웃 최적점(P2)의 정보인 $\mathbf{fnBest}_i^d$ 를 반영하여 위치 값 및 속도 값을 반복 업데이트시킬 수 있다.

[0072] [수학식 5]

$$[0073] \quad \mathbf{v}_{i(n+1)}^d = \mathbf{w} * \mathbf{v}_{i(n)}^d + \mathbf{c}_1 * \text{rand}_1^d * (\mathbf{pBest}_i^d - \mathbf{x}_{i(n)}^d) + \mathbf{c}_2 * \text{rand}_2^d * (\mathbf{fnBest}_i^d - \mathbf{x}_{i(n)}^d)$$

- [0074] [수학식 6]
- [0075]
$$\mathbf{x}_{i(n+1)}^d = \mathbf{x}_{i(n)}^d + \mathbf{v}_{i(n+1)}^d$$
- [0076] 수학식 5 및 6은 상술한 수학식 3 및 4와 고정 이웃 최적점(P2)의 위치 값을 나타내는 fnBest_i^d 를 제외하고 동일하므로, 별도로 설명하지는 않는다. 상술한 반복 업데이트 과정에도, fnBest_i^d 은 중심점을 획득하기까지 고정되어 업데이트되지 않는다.
- [0077] 중심점 획득부(140)는 고정 이웃 최적점(P2)과 근접한 적어도 2 개의 파티클(P3, P4) 및 고정 이웃 최적점에 대한 중심점(centroid)(CP1)을 획득한다(도 5 참조).
- [0078] 한 실시예로서, 중심점 획득부(140)는 고정 이웃 최적점(P2)을 기준으로 반경 값(501) 내에 적어도 2 개의 파티클(P3, P4)이 포함되는 경우, 고정 이웃 최적점(P2)과 적어도 2 개의 파티클(P3, P4)이 구성하는 도형에 대한 중심점(CP1)을 획득할 수 있다. 도 5를 참조하면, 3 개의 파티클(P2, P3, P4)이 구성하는 도형은 삼각형이 될 것이고, 이러한 삼각형에 대한 중심점(CP1)이 획득가능하다. 삼각형의 중심점(CP1)은 삼각형의 외심, 내심, 무게 중심, 수심 등이 될 수 있고, 이는 당업자가 적절히 선택하여 설계 가능하다.
- [0079] 제1 알고리즘 연산부(150)는 파티클 생성부(110), 고정 이웃 최적점 결정부(120), 수렴 연산부(130) 및 중심점 획득부(140)를 반복적으로 이용하여 복수의 중심점을 획득한다.
- [0080] 예를 들어, 도 6과 같이 중심점(CP1)은 데이터로서 메모리에 기억해 두고, 파티클 생성부(110)를 이용해서 새로운 복수의 파티클이 생성가능하다. 한 실시예에서, 파티클 생성부(110)는 중심점(CP1)과 무관하게 임의의 위치 값과 속도 값을 갖는 복수의 파티클을 생성할 수 있다. 다른 실시예에서, 파티클 생성부(110)는 중심점(CP1)으로부터 일정 간격 이상을 갖는 위치 값을 갖는 복수의 파티클을 생성할 수도 있다.
- [0081] 생성된 복수의 파티클에 대해서, 고정 이웃 최적점 결정부(120), 수렴 연산부(130) 및 중심점 획득부(140)를 이용하면 새로운 중심점(CP2)이 획득가능하다. 제1 알고리즘 연산부(150)는 상술한 과정을 반복하여 새로운 중심점(CP3) 또한 획득가능하다(도 7 참조).
- [0082] 제1 알고리즘 연산부(150)는 기존의 중심점(CP1, CP2, CP3)에 대해서 일정 간격 이상 이격된 새로운 고정 이웃 최적점을 발견하지 못하는 경우, 제1 알고리즘을 종료할 수 있다.
- [0083] 제2 알고리즘 연산부(160)는 제2 알고리즘을 이용하여, 복수의 중심점(CP1, CP2, CP3) 각각에 대응하는 복수의 후보점을 획득할 수 있다. 이때 제2 알고리즘은 MADS(Mesh Adaptive Direct Search)와 같이 잘 알려진 알고리즘일 수 있다.
- [0084] MADS 알고리즘은 일반적으로 하나의 중심점을 기준으로 제1 크기의 메쉬(Mesh)를 생성하고, 메쉬 크기에 맞춰 일정 방향으로 폴(Poll)을 뺀 다음, 폴이 위치한 부분의 비용이 기존 중심점의 비용보다 낮은지 확인하고, 비용이 낮으면 해당 폴이 위치한 위치로 중심점을 이동시킨다. 이러한 과정을 반복하다가 중심점이 이동될 여지가 없다면, 메쉬의 크기를 제1 크기보다 더 작은 제2 크기로 줄이고 상술한 과정을 반복하여, 최종적으로 중심점에 대응하는 후보점을 획득한다.
- [0085] 최적해 도출부(170)는 상술한 과정을 통해서 획득된 복수의 후보점 중 최소 비용을 갖는 후보점의 위치 값을 최적해로서 도출한다. 따라서 본 발명에 따른 최적해 도출 장치(10)는 제1 알고리즘 및 제2 알고리즘을 이용하여 주어진 대상 함수에 대한 최적해를 도출할 수 있다.
- [0086] 도 8 및 9는 예시적인 대상 함수의 최적해를 도출하는 데 있어서, 기존 PSO 알고리즘, RPSO 알고리즘, 및 본 발명의 한 실시예에 따른 최적해 도출 방법의 성능 비교를 설명하기 위한 도면이다.
- [0087] 본 발명의 실시예에 따른 최적해 도출 방법의 성능을 증명하기 위해, 가우시안 기초 함수(Gaussian basis function)가 사용되었다. 가우시안 기초 함수의 일반적인 정의는 다음 수학식 7과 같다.
- [0088] [수학식 7]

[0089]

$$f(x) = \sum_{i=1}^m \beta_i e^{-\sum_{j=1}^d \frac{(x_j - c_{ij})^2}{2\sigma_i^2}}$$

[0090] β 는 진폭(amplitude)이고, c 는 센터(center) 값이고, σ 는 함수의 폭을 나타낸다. m 은 가우시안 기초 함수의 전체 개수를 나타낸다.

[0091] 수학식 7에 기초하여 아래 수학식 8과 같은 예시적인 대상 함수를 생성하였다. 수학식 8은 이차원 함수이며, 그 래픽적으로 도시하면 도 8과 같다.

[0092] [수학식 8]

$$f(x_1, x_2) = 0.7e^{\frac{(x_1-1)^2+(x_2-1)^2}{-0.18}} + 0.75e^{\frac{(x_1-1)^2+(x_2-3)^2}{-0.32}} + e^{\frac{(x_1-3)^2+(x_2-1)^2}{-2}} + 1.2e^{\frac{(x_1-3)^2+(x_2-4)^2}{-0.32}} + e^{\frac{(x_1-5)^2+(x_2-2)^2}{-0.72}}$$

[0094] 수학식 8에 따른 대상 함수는 (x_1, x_2) 가 (3, 4)일 때 최대 값을 가지며, 따라서 (3, 4)에서 최소 비용을 갖는다고 할 수 있다. (x_1, x_2) 는 (1, 1), (1, 3), (3, 1), 및 (5, 2)에서 비교적 낮은 비용을 갖게 되나, 최소 비용을 갖지는 않는다.

[0095] 도 9의 제1 테이블(TABLE I)에는 파티클의 개수가 36 개, w 가 0.6, c_1 이 1.5, c_2 가 1.5일 때, 기존 PSO(conventional PSO), RPSO, 및 본 실시예에 따른 최적해 도출 방법이 각각 독립적으로 10 번 수행되었을 때의 성능 측정 데이터가 기록되어 있다. 이러한 성능 측정 데이터는 함수 호출 횟수(Func call)와 최적해로 예상된 위치 값(Position (x_1, x_2))을 포함한다.

[0096] 본 실시예에 따른 최적해 도출 방법은, 기존 PSO 및 RPSO와 비교했을 때, 함수 호출 횟수가 더 적고, 최소 비용을 갖는 위치 값 (3, 4)을 잘 도출해냄을 확인할 수 있다. 즉, 더 빠르고, 더 정확함을 확인할 수 있다.

[0097] 기존 PSO, RPSO, 및 본 실시예의 최적해 도출 방법은 각각 155.8 회, 114.9 회, 및 21.2 회의 반복 평균 횟수(average number of iteration)를 갖는다.

[0098] 도 10 내지 16는 본 발명의 한 실시예에 따른 최적해 도출 방법을 IPMSM의 최적 디자인 설계에 이용하는 케이스를 설명하기 위한 도면이다.

[0099] IPMSM(Interior-buried Permanent Magnet Synchronous Machine)은 매입형 영구자석 동기전동기를 의미한다. IPMSM은 국부 포화가 극심하여 최적 설계가 매우 까다로운 모델이다. IPMSM에 본 발명에 따른 최적해 도출 알고리즘을 적용시킴으로써 토크 리플(torque ripple)을 최소화시킬 수 있음을 아래에서 상세히 설명한다.

[0100] 제약 조건(constraint condition)은 4 Nm 이상의 평균 토크이고, 목표 기능(objective function)은 3% 이하의 토크 리플이다. 대상 모터의 구체적인 설계 스펙은 도 10의 제2 테이블(TABLE II)에 요약되어 있다.

[0101] 도 11은 모터의 구성 및 6 개의 설계 변수에 대한 정의를 나타낸다. 마그넷 변수(magnet variables)인 x_2, x_3 , 및 x_4 는 로터 코어(rotor core)에 미치는 자기 포화 효과(magnetic saturation effect)를 고려하여 선택되었다. 마찬가지로, 치아 구조(teeth configuration)는 Back EMF(Back Electro Motive Force)의 THD(Total Harmonic Distortion)에 결정적인 영향을 미치므로, 변수 x_5 및 x_6 가 선택되었다.

[0102] 도 12의 제3 테이블(TABLE III)은 IPMSM의 최적 설계에 적용하기 위한 알고리즘 파라미터의 값들을 나타낸다.

[0103] 최적 설계 결과에 따르면, 93 회의 반복, 608 회의 함수 호출 횟수, 및 4 시간 22 분이 필요하였다. 도 13의 제4 테이블(TABLE IV)은 본 발명의 한 실시예에 따른 최적해 도출 방법을 통해 획득된 변수 값들을 나타낸다.

[0104] 도 14는 최적 설계된 모터의 등고선 선도(contour plot)를 나타낸다. 영구 자석(permanent magnet)의 브릿지(bridge) 및 센터 포스트(center post)는 적절한 수준으로 포화되었고, 고정자(stator)의 자속 밀도(flux density)는 1.6 T에 해당한다.

[0105] 도 15는 무부하 상태(no-load condition)에서 최적 설계 모델의 Back EMF 파형을 나타낸다. Back EMF의 THD는 2.35%로 계산되었다.

[0106] 도 16은 부하 상태(load condition)에서 토크 리플의 파형을 나타낸다.

[0107] 결과적으로, 평균 토크는 4 Nm이고 토크 리플은 2.93%으로 계산되므로, 제약 조건을 지키면서도 목표했던 설계 조건을 만족시켰다.

[0108] 지금까지 참조한 도면과 기재된 발명의 상세한 설명은 단지 본 발명의 예시적인 것으로서, 이는 단지 본 발명을 설명하기 위한 목적에서 사용된 것이지 의미 한정이나 특허청구범위에 기재된 본 발명의 범위를 제한하기 위하여 사용된 것은 아니다. 그러므로 본 기술 분야의 통상의 지식을 가진 자라면 이로부터 다양한 변형 및 균등한 타 실시 예가 가능하다는 점을 이해할 것이다. 따라서, 본 발명의 진정한 기술적 보호 범위는 첨부된 특허청구 범위의 기술적 사상에 의해 정해져야 할 것이다.

부호의 설명

[0109] 10: 최적해 도출 장치.

100: 초기 설정부

110: 파티클 생성부

120: 고정 이웃 최적점 결정부

130: 수렴 연산부

140: 중심점 획득부

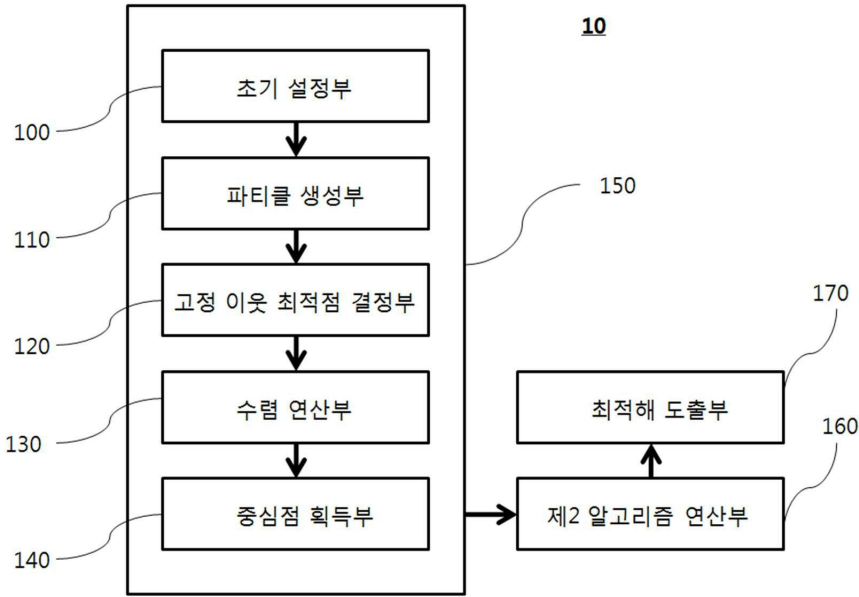
150: 제1 알고리즘 연산부

160: 제2 알고리즘 연산부

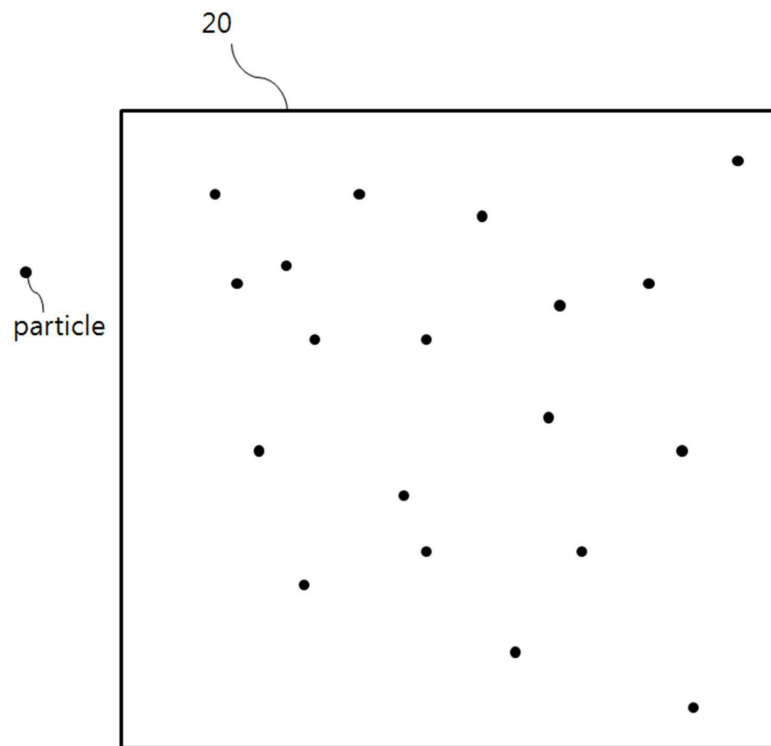
170: 최적해 도출부

도면

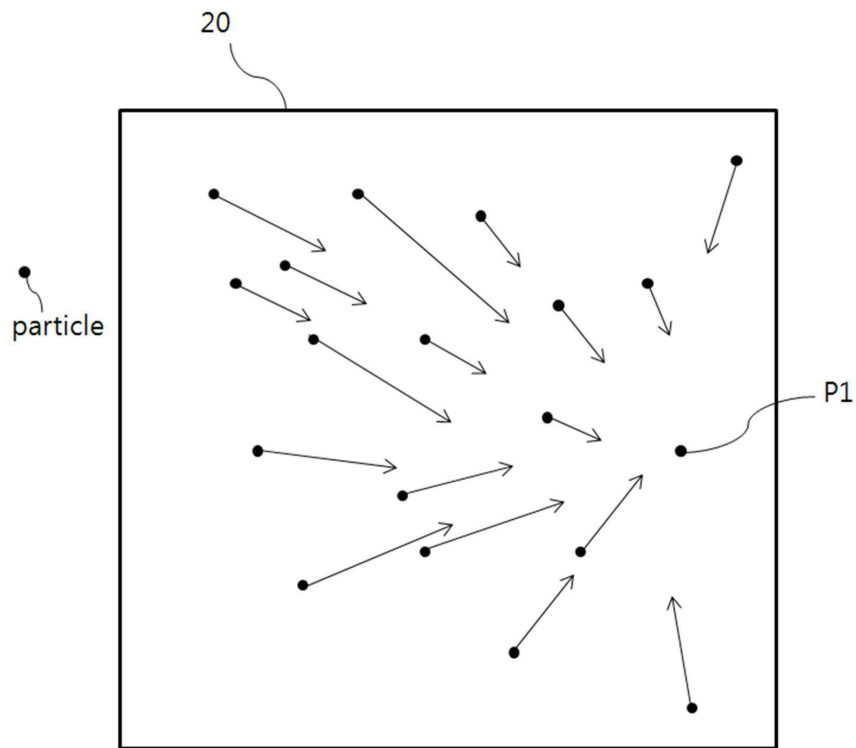
도면1



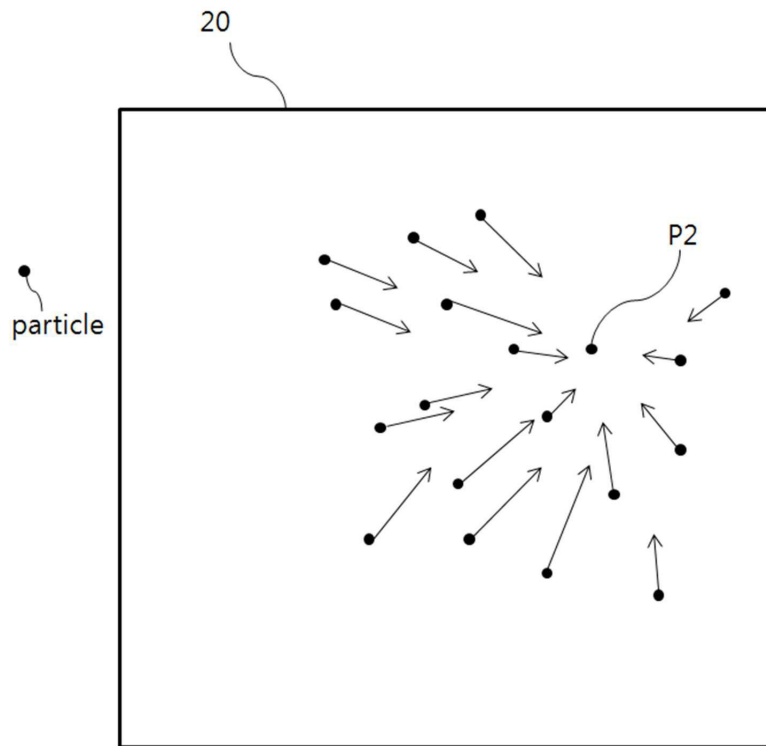
도면2



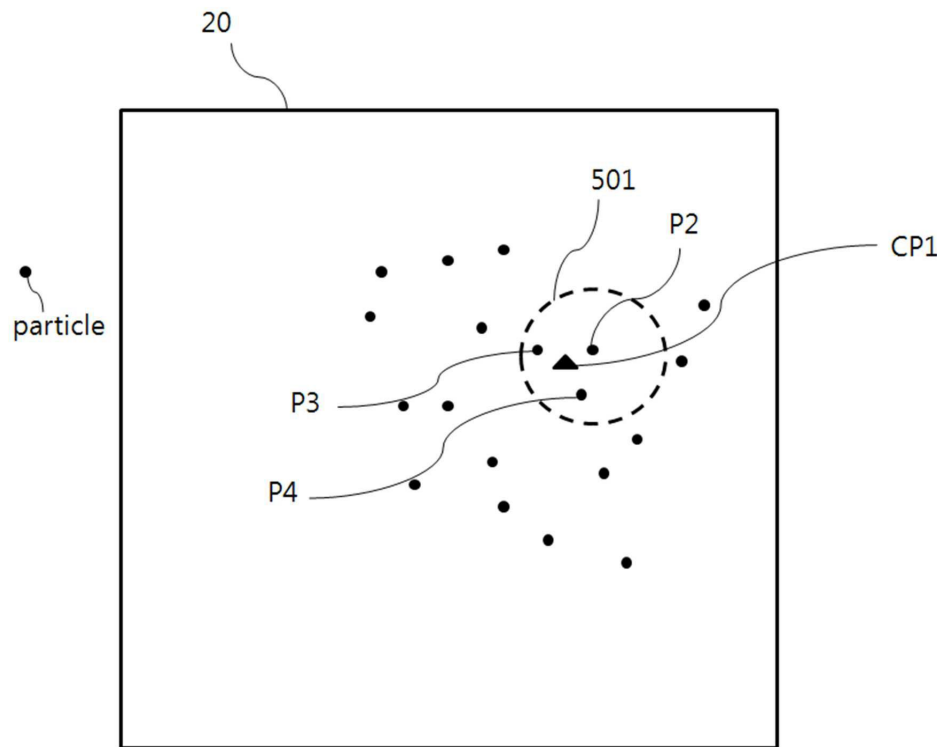
도면3



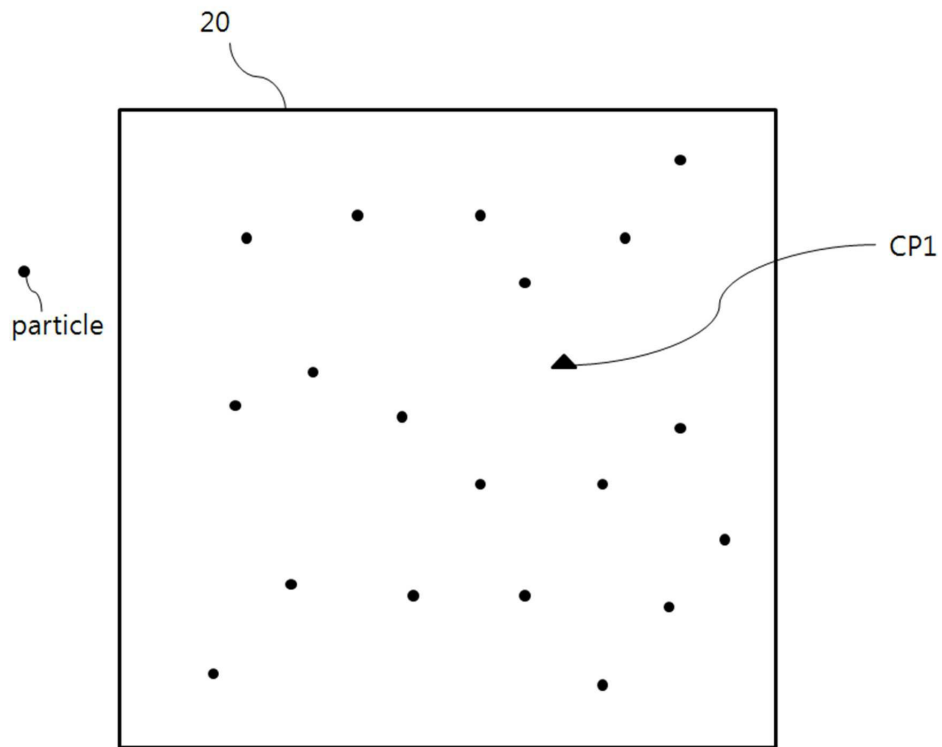
도면4



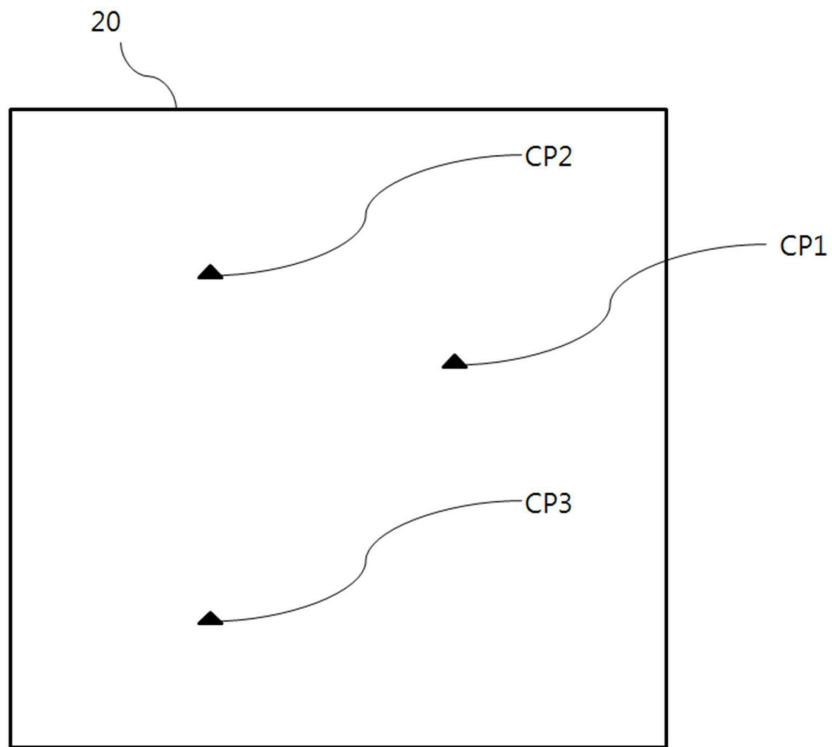
도면5



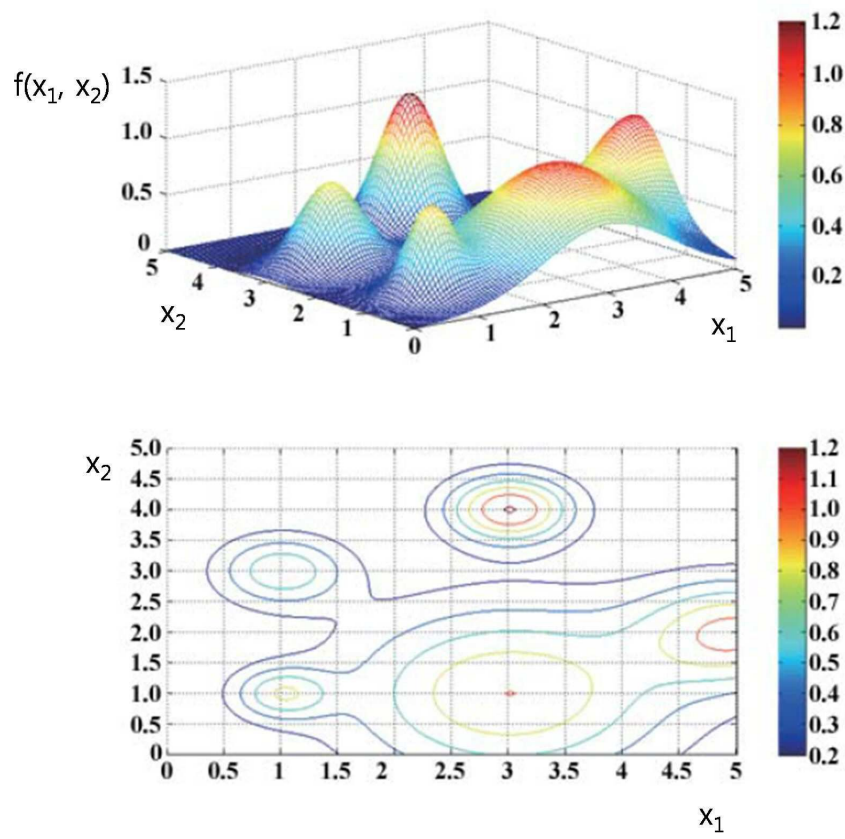
도면6



도면7



도면8



도면9

TABLE I
PERFORMANCE COMPARISON RESULT

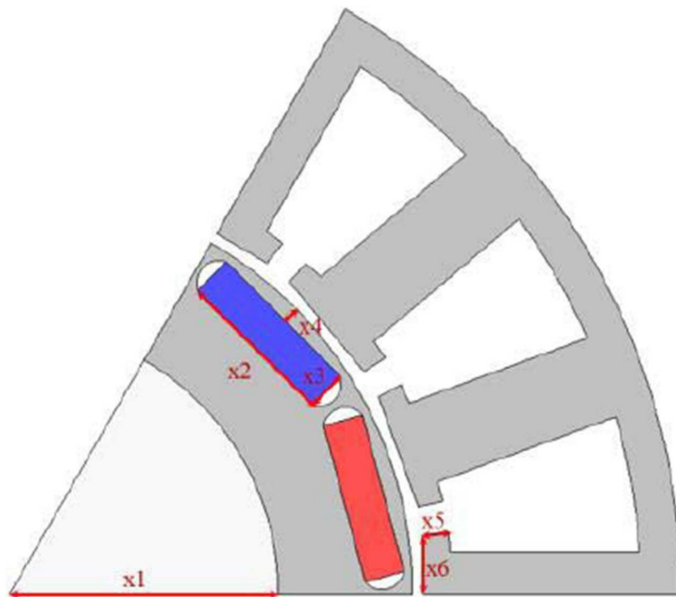
No. Run	Conventional PSO		RPSO		Proposed algorithm	
	Func call	Position (x_1, x_2)	Func call	Position (x_1, x_2)	Func call	Position (x_1, x_2)
1	5508	(3.0,1.0)	3528	(3.0,1.0)	1692	(3.0,1.0)
2	5688	(3.0,1.0)	3924	(3.0,1.0)	504	(3.0,4.0)
3	5616	(3.0,2.0)	4212	(3.0,2.0)	1080	(3.0,1.0)
4	5652	(3.0,1.0)	5364	(3.0,1.0)	1188	(3.0,1.0)
5	5472	(4.9,2.0)	5940	(3.2,1.1)	576	(3.0,4.0)
6	5436	(3.0,1.0)	3096	(3.0,1.0)	900	(3.0,4.0)
7	5580	(3.0,1.0)	3528	(3.0,1.0)	540	(3.0,4.0)
8	5508	(3.0,1.0)	4896	(3.0,1.0)	288	(3.0,3.9)
9	6372	(3.0,1.0)	1872	(3.0,1.0)	612	(3.0,4.0)
10	5616	(3.0,1.0)	5364	(3.0,1.0)	252	(3.0,4.0)

도면10

TABLE II
SPECIFICATION OF TARGET MOTOR

Specification	Value
Number of Poles	12
Number of Slots	18
Outer radius / Stack length	50[mm] / 20[mm]
Objective torque	4 [Nm]
Speed (rpm)	2000 [rpm]
Design objective	Less than 3% Torque Ripple to average torque

도면11



도면12

TABLE III
PARAMETERS OF OPTIMIZATION ALGORITHM

Specification	Value
Number of particles	60
ω	0.8
c_1, c_2	2
Max iteration number at MADS	100

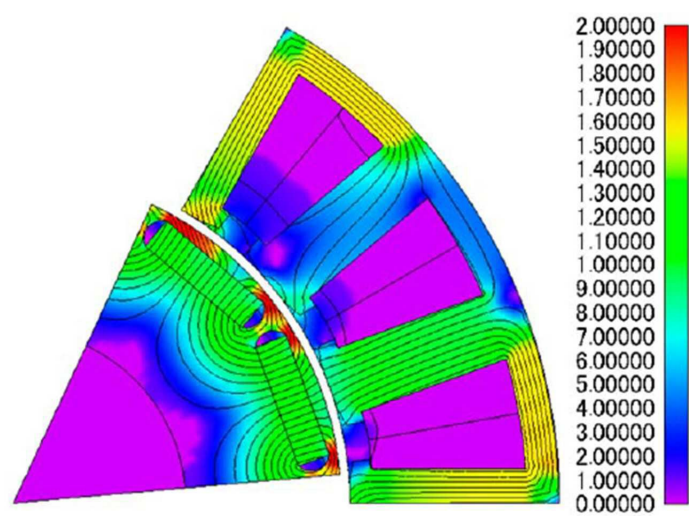
도면13

TABLE IV
VALUE OF OPTIMAL DESIGNED VARIABLES

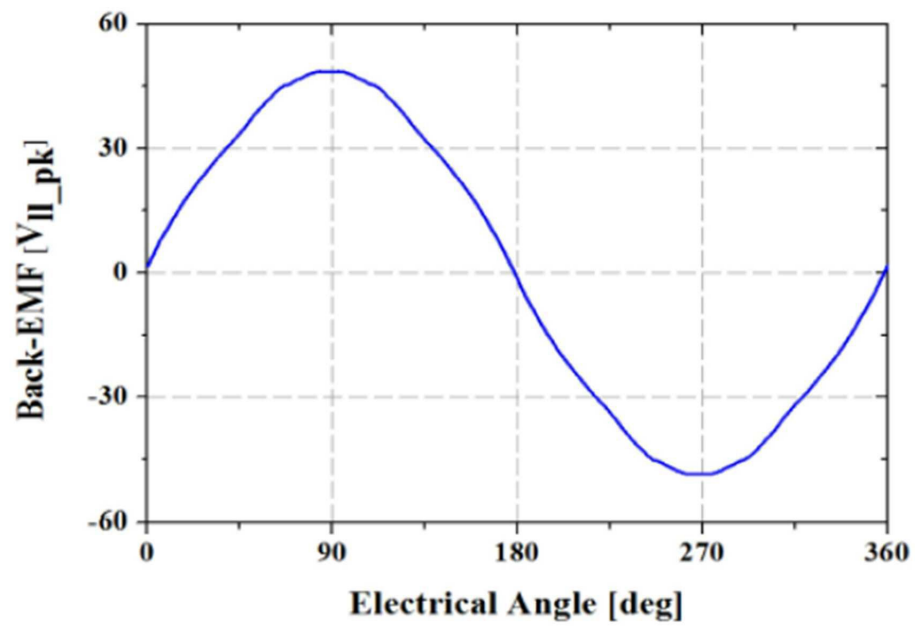
x_1	x_2	x_3	x_4	x_5	x_6
15.8	11.5625	3.0000	1.1875	1.6513	2.0132

(unit : mm)

도면14



도면15



도면16

