논문 2022-17-42

실시간 탄도 궤적 목표물 추적을 위한 GPU 기반 병렬적 입자군집최적화 기법

(Parallelized Particle Swarm Optimization with GPU for Real-Time Ballistic Target Tracking)

한 윤 호, 이 헌 철*, 권 혁 훈, 최 원 석, 정 보 라
(Yunho Han, Heoncheol Lee, Hyeokhoon Gwon, Wonseok Choi, and Bora Jeong)

Abstract : This paper addresses the problem of real-time tracking a high-speed ballistic target. Particle filters can be considered to overcome the nonlinearity in motion and measurement models in the ballistic target. However, it is difficult to apply particle filters to real-time systems because particle filters generally require much computation time. This paper proposes an accelerated particle filter using graphics processing unit (GPU) for real-time ballistic target tracking. The real-time performance of the proposed method was tested and analyzed on a widely-used embedded system. The comparison results with the conventional particle filter on CPU (central processing unit) showed that the proposed method improved the real-time performance by reducing computation time significantly.

Keywords : Ballistic target tracking, Graphics processing unit, Particle swarm optimization, Real-time systems

Ⅰ. 서 론

탄도 궤적을 가진 목표물 추적 및 요격의 성능은 얼마나 정확하게 목표물을 추적할 수 있는지에 따라 판단할 수 있다. 목표물의 위치, 각도 등의 상태를 정밀하게 추정할 수 있는 알고리즘을 채택하여 표적을 추적하여야 한다. 일반적으로 목표물의 상태 측정 시 발생하는 모델 노이즈는 수학적 단순성을 고려할 때 가우스 분포를 가진다고 가정된다. 그러나 레이돔 탐색기 (Seeker radome)와 섬광 (Scintillation) 등에 의하여 발생하는 측정 모델 노이즈는 비선형 및 비가우스 특성을 가지기 때문에 노이즈의 가우스 분포에 대한 가정은 적용되기 힘들다 [1, 2]. 비선형 및 비가우스 특성의 불확실성으로 인하여 기존의 몇몇 필터링 알고리즘들은 만족스럽지 못한 성능이 나타날 수 있다는 것으로 알려져 있다. 선형 칼만 필터 기반의 표적 추적 알고리즘들은 표적의 상태를 추정하는 도중 적절한 값에 수렴하지 않거나 심지어 값이 분기될 수 있다.

노이즈의 비선형 및 비가우스 특성으로 인하여 발생하는 문제들을 해결하기 위하여 Extended Kalman filter (EKF), Particle filter (PF), unscented Kalman filter (UKF) 등의 다양한 비선형 필터들이 목표 상태 추정을 위하여 적용되었다. 또한 최적화 기법도 비선형 및 비가우스 노이즈가 있는 환경에서 목표의 상태 추정을 위하여 적용될 수 있다. 그중 입자 군집 최적화 기법은 다양한 유형의 오류 분포를 처리할 수 있는 능력을 가지고 있어 목표물의 상태 추정에 적용하여 연구되고 있다. 또한 입자 군집 최적화 기법은 사용되는 입자들이 서로 정보를 교환하면서 최적점을 찾아 나아가기 때문에 몇 개의 입자들이 지역적 최소점 (local minimum)에 빠지더라도 전체적으로는 전역 최적점 (global optimum)을 수렴할 수 있다는 장점이 있다. 그러나 입자 군집 최적화 기법 (Particle Swarm Optimization)을 사용할 때 가장 큰 제한점은 최적의 값을 찾기 위해서는 입자의 수와 알고리즘의 계산 반복 횟수에 비례하여 성능이 나타난다는 점이다. 따라서 실시간 시스템에 입자 군집 최적화 기법을 적용하기 위해서는 계산 문제에 대한 해결책이 마련되어야 한다는 제약점이 있다.

본 논문에서는 실시간 탄도 목표물의 정밀한 추적 및 요격을 위하여 비선형 및 비가우스 노이즈 환경에서 상태 추정이 가능하도록 입자 군집 최적화를 사용한다. 그러나 입자 군집 최적화 기법을 사용하여 목표물의 상태를 정밀하게 추정하기 위해서는 많은 수의 입자들과 알고리즘의 반복 횟수가 필요하다. 입자 군집 최적화 기법은 샘플링 기반의 알고리즘으로써, 사용되는 입자 수와 반복 횟수가 커질수록 알고리즘 실행 시 소요되는 시간도 늘어난다. 따라서 실시간성을 갖도록 하기 위해서는 입자 군집 최적화 기법의 가속화가 필수적이다. 본 논문에서 입자 군집 최적화 중 계산 시간이 많이 소요되는 부분을 식별한 후, 그 부분에 대하여 GPU와 CUDA를 사용한 병렬화를 적용하였다. 또한 탄도 목표물 요격체에서 입자 군집 최적화 기법을 사용하여 상태

*Corresponding Author (hclee@kumoh.ac.kr)
Received: Oct. 14, 2022, Revised: Nov. 15, 2022, Accepted: Nov. 26, 2022.
Y. Han: Department of IT Convergence Engineering, Kumoh National Institute of Technology (M.S. Candidate)
H. Lee: Department of IT Convergence Engineering, Kumoh National Institute of Technology (Assist. Prof.)
H. Gwon: PGM R&D Lab, LIGNEX1 (Chief Research Engineer)
W. Choi: PGM R&D Lab, LIGNEX1 (Chief Research Engineer)
B. Jeong: PGM R&D Lab, LIGNEX1 (Research Engineer)
※ 이 연구는 광역방어 특화연구센터 프로그램(UD200043CD)의 일환으로 국방과학연구소와 방위사업청의 지원으로 수행되었음.

ⓒ IEMEK J. Embed. Sys. Appl. 2022 Dec. 17(6) 355-365
ISSN : 1975-5066
http://dx.doi.org/10.14372/IEMEK.2022.17.6.355

추정을 한다는 가정하에 PC환경이 아닌 온보드 환경에 적합하도록 NVIDIA Jetson Xavier를 사용하여 가속화가 진행되었다. 따라서 임베디드 환경에서 입자 군집 최적화를 CPU와 GPU의 상호설계하여 실시간성을 향상시키며 탄도 목표물 상태 추정에 성공하였다.

표 1은 그중 대상 추적, 객체 추적, 모션 추적 등의 다양한 목표 상태 추적 연구에 입자 군집 최적화 기법이 사용된다는 것을 보여준다. 미사일 응용에 사용되는 관련 연구도 발견된다. GPU를 이용한 가속화 방법은 여러 분야에서 적용할수 있다 [3]. 입자 군집 최적화 기법의 가속화는 연구들은 실시간 임무 완수를 위하여 GPU를 CUDA와 함께 사용하여 수행되며, 입자 군집 최적화 기법의 가속화 연구는 대부분 모션 추적 연구에서 진행되었다. 입자 군집 최적화 기법은 목표물의 상태 추적이 아닌 검출 등 다른 방면에서도 가속화 연구가 진행되고 있다 [4, 5]. GPU를 사용하여 입자 군집 최적화를 가속화한 연구들은 대부분 사용된 입자들이 가지고 있는 정보의 에러를 계산하는 부분이나 비용함수를 계산하는 부분을 가속화 하였다. 본 논문에서의 입자 군집 최적화 기법은 탄도 목표물을 추적하는 목적에 적용되어 다른 연구들과 어플리케이션이 차이가 있다. 때문에 다른 연구들과 달리 알고리즘 중 연산시간이 가장 많이 소요된 입자들이 어느 지점으로 움직여야 하는지 판단하는 부분과 우도함수 계산부분을 가속화 하였다.

본 논문의 나머지 부분은 다음과 같이 구성되어 있다. 우선 입자 군집 최적화 기법과 최적화 기법을 사용하여 탄도 궤적 목표물 추적과 입자 군집 최적화 기법이 실시간성을 가지기에 제한점에 대하여 설명한다. 다음으로 입자 군집 최적화 알고리즘의 블록 단위로 프로파일링하여 각 부분마다 소요되는 연산시간을 구한 후 각 입자가 가지는 정보를 업데이트하는 부분과 입자들이 가지는 정보를 종합하여 최적의 값을 찾는 부분에 대한 병렬화 방법이 제안된다. 마지막으로 임베디드 시스템에서 제안된 방법의 결과를 병렬화하지 않은 알고리즘과 제안된 방법을 사용한 알고리즘의 결과를 비교한 후 결과들에 대한 결론을 제시한다.

표 1. 목표 추적에 적용된 입자 군집 최적화 관련 연구들
Table 1. Related works to PSO for target tracking

Related Works	GPU-based parallelization	Parallelization part	Missile Application
[6], [7], [8], [9]	X	-	X
[10], [11]	X	-	O
[12]	O	calculation of the error	X
[13]	O	calculation of the cost func	X
Ours	O	predicted measurement / associated likelihood func	O

Ⅱ. 탄도 궤적 목표물 추적

1. 표본 추출 기반 탄도 궤적 목표물 추적

특정 대상을 추적하는 알고리즘의 주 목표는 실제 추적하는 대상의 상태를 실시간으로 추정하는 것이다. 본 연구의 대상인 탄도 미사일의 추적 알고리즘의 성능을 평가하기 위해서는 먼저 탄도 미사일 궤적의 모사가 요구된다. 대기권 밖에서의 비행과 달리 대기권으로의 재돌입 한 후에는 중력과 더불어 항력 등의 공기역학적 힘이 탄도 미사일의 경로를 결정하는 데 중요한 영향력을 미친다. 본 연구에서는 탄도 미사일을 점 질량으로 가정하였으며, 3차원 데카르트 좌표계에서 기술하였다. 다음은 중력 및 공기역학적 힘을 포함한 3차원 비선형 운동 방정식을 나타낸다.

(1), (2), (3)의 함수에서 x, y, z는 위치를 나타내며 V는 속도 나타내고, γ와 ψ는 각각 비행경로각과 헤딩각을 의미한다. 또한, m은 질량, g는 중력 상수를 나타내며, T, D, L은 각각 추력, 항력, 양력을 나타낸다. 공기역학적 힘은 공기 밀도 ρ, 항력 계수 C_D, 양력 계수 C_L 그리고 기준 넓이 S로 이루어져 있다. 그리고, ϕ는 양력을 발생 방향을 나타낸다. 본 연구에서 고려하는 재진입 단계에서는 일반적으로 추진기관의 연소가 끝난 상태이기 때문에 추력은 0으로 설정하였으며, 질량도 일정하다고 가정하였다. 또한, 대기권 내에서 일반적으로 탄도 미사일의 기동이 미미하고 항력에 비하여 상대적으로 영향이 작기 때문에 양력을 0으로 가정하였다.

$$\begin{aligned}\dot{x} &= V\cos\gamma\cos\psi \\ \dot{y} &= V\cos\gamma\sin\psi \\ \dot{z} &= -V\sin\gamma\end{aligned} \tag{1}$$

$$\begin{aligned}\dot{V} &= \frac{(T-D-mg\sin\gamma)}{m} \\ \dot{\gamma} &= \frac{(L\cos\phi - mg\cos\gamma)}{mV} \\ \dot{\psi} &= \frac{L\sin\phi}{mV\cos\gamma}\end{aligned} \tag{2}$$

$$\begin{aligned}D &= \frac{1}{2}\rho V^2 \cdot C_D \cdot S \\ L &= \frac{1}{2}\rho V^2 \cdot C_L \cdot S\end{aligned} \tag{3}$$

2. 목표물의 움직임 및 측정 모델

목표물 추적은 기준 운동 모델을 기반으로 추정 알고리즘을 설계하므로 몇 가지 모델을 고려할 수 있다. 본 논문에서는 잘 알려져 있는 Singer 모델을 기반으로 연구되었다 [14, 15]. Singer 모델은 목표물의 가속도가 평균이 0인, 1차 정적 마르코프 프로세스라고 가정한다. 연속적인 시간에서 Singer 모델의 상태 공간 표현은 (4)와 (5)의 식으로 나타낼 수 있다. (4)의 식에서 구할 수 있는 x는 추적 목표물의 상태를 의미한다. (4)와 (5)의 식에서 w는 평균이 0이며, 시정수 τ로 얻어지는 백색 가우스 노이즈이다. 그리고 F와 G 행렬의 I_3와 τ는 3차 항등 행렬과 기동 상수이다. w의 이산 시간 방정식은 (6)과 (7)의 식으로 나타난다. 여기서 Φ_k와 $\triangle t$는 상태 전이 행렬과 샘플링 시간 간격을 의미한다.

$$\dot{x} = Fx + Gw \tag{4}$$

$$F = \begin{bmatrix} 0_3 & I_3 & 0_3 \\ 0_3 & 0_3 & I_3 \\ 0_3 & 0_3 & -\frac{I_3}{\tau} \end{bmatrix}, \qquad G = \begin{bmatrix} 0_3 \\ 0_3 \\ I_3 \end{bmatrix} \tag{5}$$

$$x_k = \Phi_{k-1} x_{k-1} + w_{k-1}, \qquad w_{k-1} \sim N(0,\ Q_k) \tag{6}$$

$$\Phi_k \simeq I + F\Delta t \tag{7}$$

$$Q_k \simeq S_w Q_0 = S_w \begin{bmatrix} \frac{\Delta t^5}{20} I_3 & \frac{\Delta t^4}{8} I_3 & \frac{\Delta t^3}{6} I_3 \\ \frac{\Delta t^4}{8} I_3 & \frac{\Delta t^3}{3} I_3 & \frac{\Delta t^2}{2} I_3 \\ \frac{\Delta t^3}{6} I_3 & \frac{\Delta t^2}{2} I_3 & \Delta t I_3 \end{bmatrix} \tag{8}$$

(6)의 공분산 Q_k은 전력 스펙트럼 밀도 (Power Spectral Density)인 S_w와 백색 노이즈 jerk 모델인 Q_0로 구성된다. 일정 시간 동안 가속이 증가하는 크기는 해당 시간 동안의 jerk의 적분을 나타낸다.

(4)의 식에서 상태 변수 x는 다음 (9)의 식처럼 데카르트 좌표계에서 위치, 속도 및 가속도를 나타내는 벡터인 P, V 그리고 A를 사용하여 정의한다. (10)의 식에서 $[x, y, z]$은 데카르트 좌표계에서 목표물의 위치를 의미한다.

$$x = [P^T\ V^T\ A^T]^T \tag{9}$$

$$P = [x\ y\ z]^T \tag{10}$$

목표물의 측정값인 고도, 방위각, 그리고 거리는 탄도탄 탐지 레이더에 의하여 측정된다고 가정한다. 이 측정값들은 표적과 레이더의 상대 위치에 따라서 얻을 수 있다. (11)의 식에서의 아래 첨자 r는 목표 대상과 레이더 사이의 상대적 위치를 나타내고, m은 레이더 위치를 의미한다. 따라서 두 개의 베어링 각도 z_θ와 z_ψ, 상대 거리 z_R는 레이더 측정 오차 등을 포함하여 다음 식 (12)와 같이 나타낼 수 있다. 레이더 측정 오차는 가우시안 잡음인 레이더 수신기 잡음 n_θ, n_ψ 그리고 n_R와 비가우시안 잡음인 글린트 (Glint) 잡음 $n_{G,\theta}$와 $n_{G,\psi}$로 구성된다 [16].

$$[x_r\ y_r\ z_r]^T = [x\ y\ z]^T - [x_m\ y_m\ z_m]^T \tag{11}$$

$$\begin{bmatrix} z_\theta \\ z_\psi \\ z_R \end{bmatrix} = \begin{bmatrix} \tan^{-1}\left(\frac{z_r}{\sqrt{x_r^2 + y_r^2}}\right) + n_{G,\theta} + n_\theta \\ \tan^{-1}\left(\frac{y_r}{x_r}\right) + n_{G,\psi} + n_\psi \\ \sqrt{x_r^2 + y_r^2 + z_r^2} + n_R \end{bmatrix} \tag{12}$$

Ⅲ. 문제점 기술

1. 목표물 추적 문제

탄도 미사일과 같은 고속의 목표물 경우 추적 알고리즘의 업데이트율과 추정 정확도가 매우 중요한 부분이다. 성공적인 요격을 위한 정밀 유도 및 제어는 얼마나 정확히 표적을 추적할 수 있는지에 크게 영향을 받는다. 본 연구에서는 입자 군집 최적화 알고리즘이 목표 추적의 높은 추정 정확도

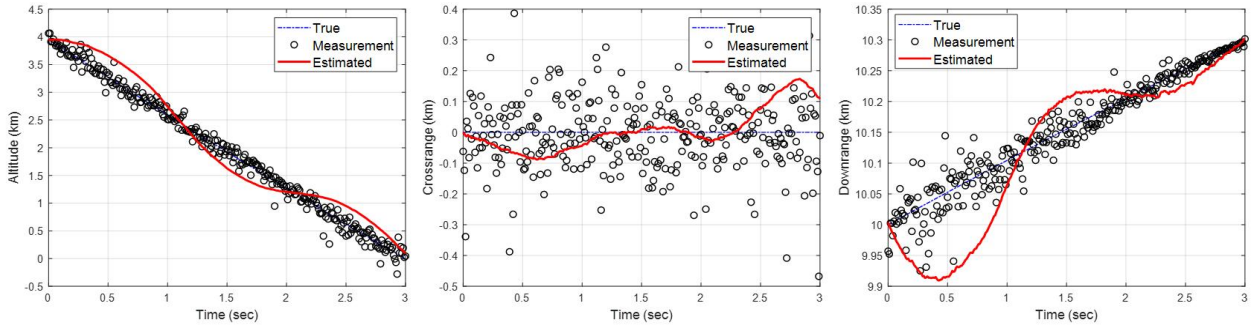


그림 1. 100개의 입자를 사용한 입자 군집 최적화 탄도 목표물 추적 결과
Fig. 1. Result of the ballistic target tracking with PSO using 100 particles

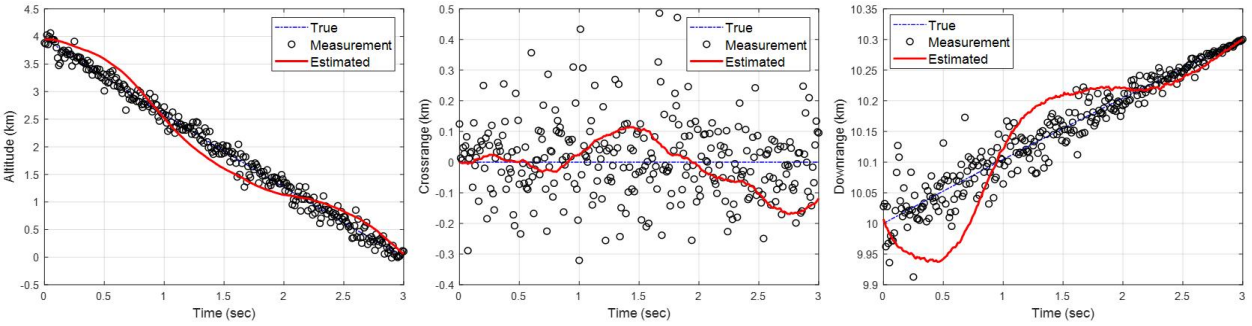


그림 2. 200개의 입자를 사용한 입자 군집 최적화 탄도 목표물 추적 결과
Fig. 2. Result of the ballistic target tracking with PSO using 300 particles

를 위하여 사용되었다. 탄도 궤적을 가지는 목표물을 추적하기 위하여 입자 군집 최적화를 적용하였을 때, 충분한 수의 입자를 사용하여야 높은 정확도를 가지는 추정을 할 수 있다. 입자 군집 최적화에서 충분하지 않은 수의 입자들을 사용하였을 때의 목표물 추적 결과는 그림 1과 그림 2에 나타난다. 그림 1은 탄도 목표물 추적에 100개의 입자를 사용한 입자 군집 최적화 결과이며 그림 2는 200개의 입자를 사용하였을 때이다. 두 그림 모두 파란 선으로 나타난 목표물의 실제 상태를 정확하게 추정하는데 실패하였다. 그림 1보다 그림 2에서 목표물의 실제 값과 추정 값 간 차이가 좀 더 작았다. 따라서 목표물의 정확한 상태 추정을 위하여 보다 많은 수의 입자가 필요하다는 것을 알 수 있다.

2. 실시간성 문제

탄도 궤적 목표물을 요격하기 위해서는 대상 목표물을 요격할 물체가 실시간으로 상태를 추정할 수 있어야 한다. 그러나 입자 군집 최적화 알고리즘 특성상 정확한 추정을 위하여 계산 시에 시간이 많이 소요된다는 커다란 제한점이 있다. 입자 군집 최적화 알고리즘은 입자들이 가지는 속도, 가속도 그리고 위치 정보를 계산한 후 그 중의 최적값을 찾아 움직이며 최적점을 찾아내는 알고리즘이다. 입자들이 가지는 정보를 계산하는 부분이 한 사이클마다 반복해서 진행되며, 각각의 입자마다 가지는 정보를 계산하기 때문에 알고리즘 수행 시 많은 계산시간이 소요된다. CPU를 이용하여 입자 군집 최적화 알고리즘을 진행하게 되면 반복 횟수만큼 순차적으로 계산이 수행된다. 정확한 목표 상태 추정을 위하여 입자 수가 증가할수록 동일한 계산을 더 많이 반복하여야 하므로 알고리즘 수행 중 계산시간 또한 증가한다. 목표물의 상태 추정을 위하여 입자 군집 최적화 기법을 적용하여 고속으로 움직이는 탄도 목표물을 추적하여 요격할 때, 이러한 제한점으로 인하여 실시간으로 변경되는 목표물의 상태를 제대로 추적할 수 없으므로 요격에 실패할 수 있다. 따라서 고속으로 움직이는 탄도 목표물에 대하여, 추적 시 실시간으로 목표물의 상태 추정할 수 있어야 하므로 이 문제를 해결하기 위하여 입자 군집 최적화 기법에 대한 GPU 기반 가속화 방법을 제안한다.

탄도 목표물을 추적하여 요격하기 위한 물체는 목표물 상태 추정을 위한 장치가 탑재되어야 한다. 해당 물체에 PC 환경을 탑재하기는 전력, 발열 등의 문제로 인하여 제한이 된다. 때문에, 주로 목표물 상태 추정의 계산을 진행할 수 있는 보드 등을 사용하게 된다. 이를 고려하여 본 논문에서의 실험은 임베디드 시스템에서 CPU와 GPU를 상호 설계하여 병렬화 및 가속화를 진행하였다. 탄도 목표물의 실시간 추적 및 요격을 위하여 입자 군집 최적화 기법을 온보드 환경에서 가속화하여 실시간성을 향상하였다.

Ⅳ. 제안하는 기법

1. 데이터 획득

우선 CPU에서의 전체 알고리즘의 흐름도는 그림 3에 나타난다. 먼저 앞서 설명한 식들을 근거로 하여 추적하기 위한 탄도 목표물의 모델을 만들고 추적 시 발생하는 노이즈를 생성한다. 그 후 생성된 탄도 목표물에 대한 상태를 추정하기 위하여 입자 군집 최적화 기법을 적용하여 목표를 추적할 수 있도록 한다. 입자 군집 최적화 기법의 알고리즘은 먼저 입자들을 생성하고 값을 초기화한다. 생성된 입자들에 최적값을 찾고자 하는 공간에 랜덤한 값들을 부여한다. 입자들이 가지는 값들은 그 지점에서의 위치, 속도 그리고 가속도의 정보를 의미한다. 이 정보를 사용하여 목표물에 대한 상태를 추정한다. 다음 추정한 결과값을 우도 함수에 적용하여 가장 가능성 있는 추정치를 구한다. 각각의 입자가 가지고 있는 정보 중 가장 최적점으로 판단되는 값을 국소 최적점 (Local optimal point)이라 하고, 모든 입자가 가지고 있는 정보 중에서 가장 최적점으로 판단되는 값을 전역 최적점 (Global optimal point)이라 한다. 우도함수로 구한 추정치와 전역 최적점, 국소 최적점 그리고 입자의 정보를 사용하여 입자 정보에 대하여 업데이트를 진행한다.

이 과정이 각각의 입자마다 적용되며 임의의 반복 횟수만큼 반복하며 최적점을 찾아간다. 입자 군집 최적화 알고리즘이 완료되면 탄도 목표물의 실제값과 비교하여 얼마나 정밀하게 목표물의 상태를 추정하였는지 확인한다.

2. 계산 시간 프로파일링

입자 군집 최적화 알고리즘을 병렬화하기 전, 알고리즘 중 어느 부분에서 계산 시간이 많이 소요되는지 식별하는 작업이 병렬화를 하기 전 선행되어야 한다. 입자 군집 최적화를 사용하여 탄도 목표물을 추적하는 알고리즘을 보여주는 그림 3의 흐름도에서 각 부분마다 소요되는 계산 시간을 그림 4에서 보여주고 있다. 알고리즘 각각의 부분마다 소요되는 시간을 식별하기 위하여 입자 군집 최적화 알고리즘의 입자는 1000개로 정의하였고 한 입자가 최적점을 향하여 이

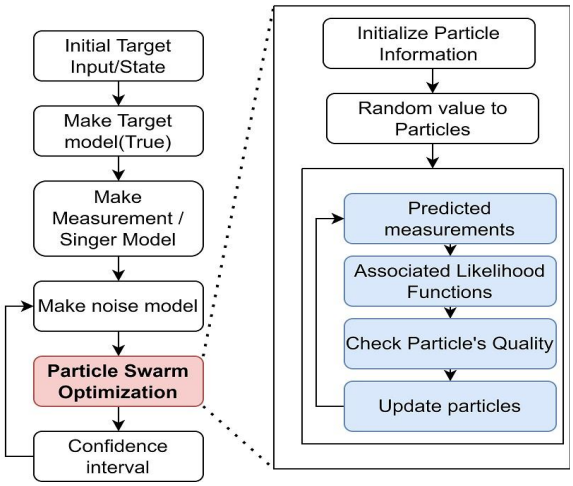


그림 3. 목표물 추적 알고리즘의 흐름도
Fig. 3. Flowchart of target tracking algorithm

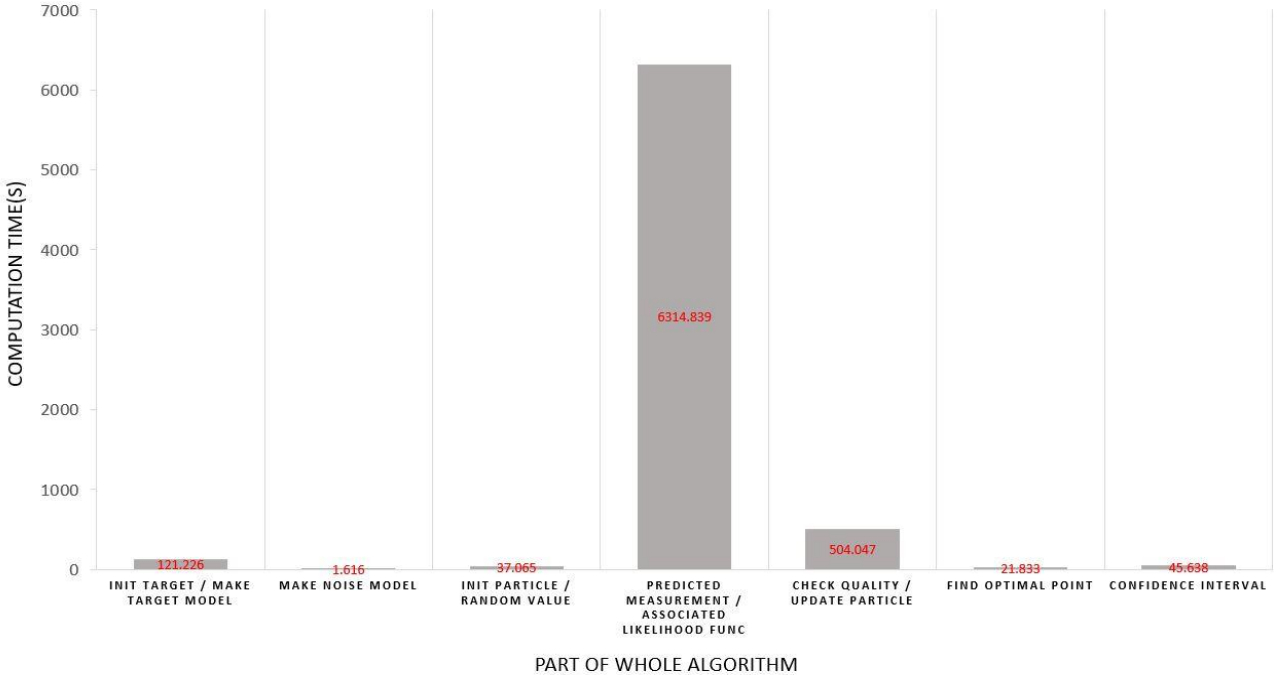


그림 4. 입자 군집 최적화 기법을 적용한 목표물 추적 알고리즘의 각 파트별 수행시간
Fig. 4. Computation time of the Target tracking algorithm using Particle Swarm Optimization

동하는 횟수는 10번으로 정의하였다. 그림 4에서 추적 대상이 되는 탄도 목표 모델 생성 부분과 노이즈 모델을 생성하는 부분들은 목표물을 추적하는 알고리즘에서 소요되는 시간에 비하여 매우 적은 시간을 필요로 한다. 목표물의 상태를 추정하는 입자 군집 최적화 부분에서 시간이 많이 소요되는 것을 알 수 있다. 특히 시간이 가장 많이 소요되는 부분은 각 입자들이 가지는 정보를 사용하여 입자들이 어느 지점으로 움직여야 하는지 판단하는 부분과 우도함수를 계산하는 부분으로 식별되었다. 이 부분의 계산 시간은 약 6314.839초 소요되었으며 이 부분에 대하여 GPU를 사용한 병렬화를 진행하고 결과적으로 CPU에 비하여 실시간성을 가질 수 있도록 가속화 하였다.

3. 병렬적 입자 군집 최적화 기법

3.1 목표의 대한 상태 추정 부분 병렬화

그림 4에서 제일 시간이 오래 걸리는 부분이라고 식별된 부분 중 먼저 입자들이 가지고 있는 정보들을 사용하여 목표의 상태라고 예측되는 측정값을 구하는 부분을 병렬화한다. 데카르트 좌표계 안에서 목표의 위치와 상태를 추정하기 위하여 목표의 거리와 각도와 목표의 회전 각도를 다음의 식을 사용하여 구한다.

$$Range[n] = \sqrt{x_{p1}^2[n] + x_{p2}^2[n] + x_{p3}^2[n]} \tag{13}$$

$$\theta[n] = -atan(x_{p3}[n] / \sqrt{x_{p1}^2[n] + x_{p2}^2[n]}) * \frac{180}{\pi} \tag{14}$$

$$\psi[n] = atan(x_{p2}[n] / x_{p1}[n]) * \frac{180}{\pi} \tag{15}$$

(13), (14) 그리고 (15)의 식에서 거리, 각도, 회전 각도를 추정하기 위하여 사용되는 x_{p1}, x_{p2}, x_{p3}은 후에 서술할 입자 군집 최적화 알고리즘에서 최적점을 찾아가기 위하여 알고리즘 안에서 구해지는 입자의 위치 정보이다. 목표 상태의 추정을 위하여 알고리즘상의 입자 위치 정보는 행이 3개이고 열은 입자 군집 최적화에 사용되는 입자의 개수 (n)만큼의 크기를 가지는 행렬로 얻어진다. (13)~(15)의 식은 입자의 개수인 n번 만큼 반복되며 각각 열이 n개인 행렬로 나타난다. 입자의 개수만큼 위의 식들이 반복되면서 목표 상태를 추정하기 때문에 입자가 많을수록 정확도가 높은 상태를 추정할 수 있지만, 계산 또한 많이 반복되기 때문에 병렬화를 진행하여 가속화 하였다.

CUDA 커널들을 정의하기 전, 커널들에 사용될 스레드와 블록들에 대하여 정의하여야 한다 [17]. 스레드의 크기는 1024로 정의하였으며, 스레드들을 포함하는 블록은 (N × 입자의 수)로 정의된다. 커널에 들어갈 데이터 행렬들의 행의 크기에 따라 N을 1, 3, 9로 3개를 정의하였다. 커널에서 입력받은 데이터들의 주소를 찾기 위한 id는 아래 (16)의 식을 사용하여 정의된다. (16)의 식에서 $blockIdx$는 GPU 메모리 블록의 x축 차원의 개수이며, $blockDim$는 블록의 행의 개수, 그리고 $threadIdx$는 스레드의 수를 의미한다. 본 연구에서 한 블록이 가질 수 있는 스레드의 수는 1024개로 정의하였다.

$$id = blockIdx * blockDim * threadIdx \tag{16}$$

CUDA를 사용하여 GPU 상에서 병렬 계산하기 위하여 (13)~(15)의 식에 사용되는 함수들을 커널로 정의하여야 한다.

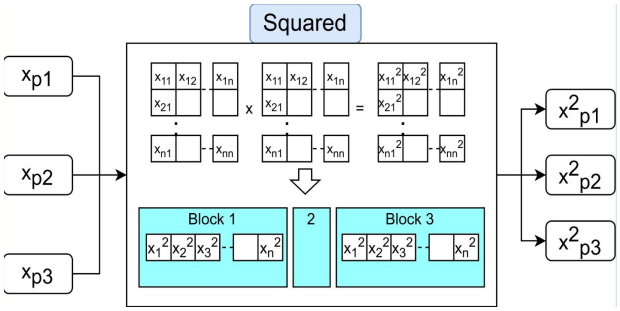


그림 5. 거리, 각도, 회전 각도를 구하기 위한 입자들의 위치정보 제곱 커널
Fig. 5. Square kernel of positional information of the particles to obtain Range, Theta, and Psi

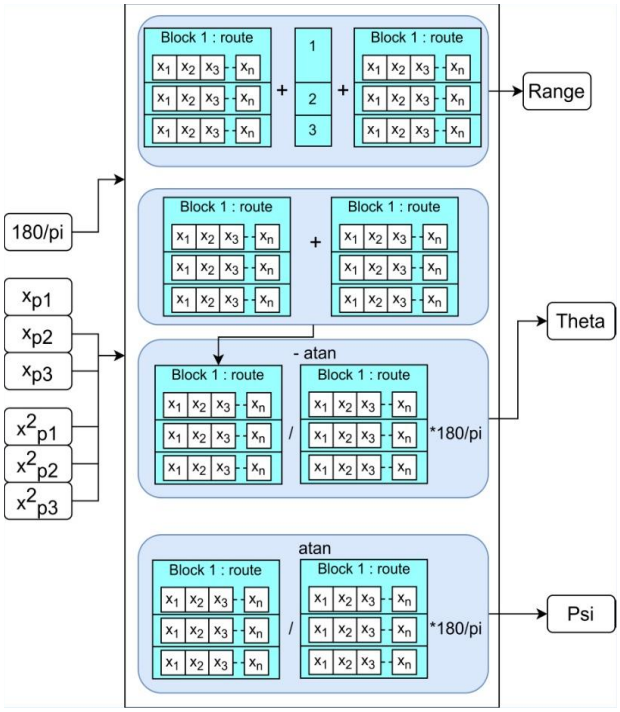


그림 6. 거리, 각도, 회전 각도를 계산하는 커널
Fig. 6. Kernels for obtaining Range, Theta, and Psi

알고리즘 상의 입자의 위치 정보 값이 본래의 값과 제곱 값이 사용되기 때문에 위치 정보 값을 제곱해주는 커널을 정의한다. (13), (14), (15)의 수식들을 각각 커널로 정의하여 CUDA를 사용하여 병렬 계산할 수 있게 하였다. 먼저 그림 5와 같이 x_p 행렬들의 제곱을 구하는 커널을 정의하여 입자 정보의 제곱을 병렬 계산하여 구한다.

그 후 거리, 각도, 회전 각도를 구하는 (13)~(15)의 식들은 서로의 결과에 영향을 끼치지 않아 독립적으로 연산이 가능하다. 그림 6과 같이 거리, 각도, 회전 각도를 구하는 커널들을 동시에 병렬 수행하여 계산시간을 단축한다.

3.2 입자 정보 업데이트 부분 병렬화

두 번째 부분은 입자가 가지고 있는 정보를 업데이트하는 부분에서 계산시간이 많이 소요된다. 이 부분을 크게 두 부분으로 나누면 입자가 가지고 있는 정보를 업데이트하는 부분과 업데이트하기 전 업데이트에 사용되는 값을 구하는 부분으로 구분한다. 먼저 입자 군집 최적화 기법에서 입자가 가지고 있는 정보를 업데이트하는 부분은 아래의 식으로 구할 수 있다.

$$P_{a+1} = kai*((c*eps*(OP_g - P_x)) + (c*eps(OP_l - P_x))) - (1-kai)*P_v \quad (17)$$

$$P_{v+1} = P_v + P_a \quad (18)$$

$$P_{x+1} = P_x + P_v \quad (19)$$

(17)의 식에서 입자 군집 최적화 알고리즘 내의 하나의 입자가 가지고 있는 위치 정보 P_x, 속도 정보P_v, 국소 최적점 OP_l, 모든 입자들이 가지는 정보 중 전역 최적점 OP_g를 사용하여, 다음 사이클에 하나의 입자가 가질 가속도P_{a+1} 정보를 구할 수 있다. (18)과 (19)의 식 또한 하나의 입자가 가지는 속도, 가속도, 위치 정보를 사용하여 다음 사이클에 하나의 입자가 가질 속도P_{v+1}와 위치P_{x+1}정보를 구한다. 식 (13)~(15)에서 사용되는 입자의 위치 정보는 식 (17)~(19)을 사용하여 업데이트 한다.

입자가 가지고 있는 정보들을 위의 식들을 적용하여 업데이트하기 전, 한 사이클에서 (13)~(15)의 식을 사용하여 목표의 상태를 추정한 후 추정한 결과값을 우도 함수에 적용하여 가장 가능성 있는 추정치를 구하여 다음 사이클의 입자 정보 업데이트에 사용한다.

$$Range_A = \frac{1}{\sqrt{2\pi}*sig_R} * \exp\left(\frac{-(mea_R - Range)^2}{(2*sig_R)^2}\right) \quad (20)$$

거리의 대한 추정치$Range_A$는 (20)의 식을 사용하여 구한다. sig_R는 추적할 목표물에 대하여 추정한 거리 데이터에서 발생할 수 있는 노이즈이며, mea_R은 목표물에 대하여 추정한 거리 측정값이다. 본 논문에서는 sig_R를 1로 설정하였다. 또한 (20)과 (21)~(24) 식에서 사용되는 $Range$는 (13)의 식을 사용하여 얻은 목표의 거리 추정치이다.

$$temp_1 = (1-ep)*(1/\sqrt{2\pi*(sig_{t,p}^2 + (sig_{g1}/Range^2))}) \quad (21)$$

$$temp_2 = \exp\left(\frac{-(mea_{t,p} - \theta)^2}{2((sig_{t,p} + sig_{g1})^2/Range^2)}\right) \quad (22)$$

$$temp_3 = (ep)*(1/\sqrt{2\pi*(sig_{t,p}^2 + (sig_{g2}/Range^2))}) \quad (23)$$

$$temp_4 = \exp\left(\frac{-(mea_{t,p} - \theta)^2}{2((sig_{t,p} + sig_{g2})^2/Range^2)}\right) \quad (24)$$

각도와 회전 각도의 추정치는 (21)~(24)의 식들을 사용하여 구할 수 있다. 각도의 측정 노이즈 sig_t와 회전 각도의 측정 노이즈 sig_p는 0.1로 설정하였고 추정 시 생길 수 있는 노이즈 sig_{g1}과 sig_{g2}는 각각 0.5와 1로 설정하였다. 노이즈 값들과 각각의 측정값들을 사용하여 각도와 회전 각도에 대한 추정치는 위의 식에서 구할 수 있는 $temp$들의 합으로 나

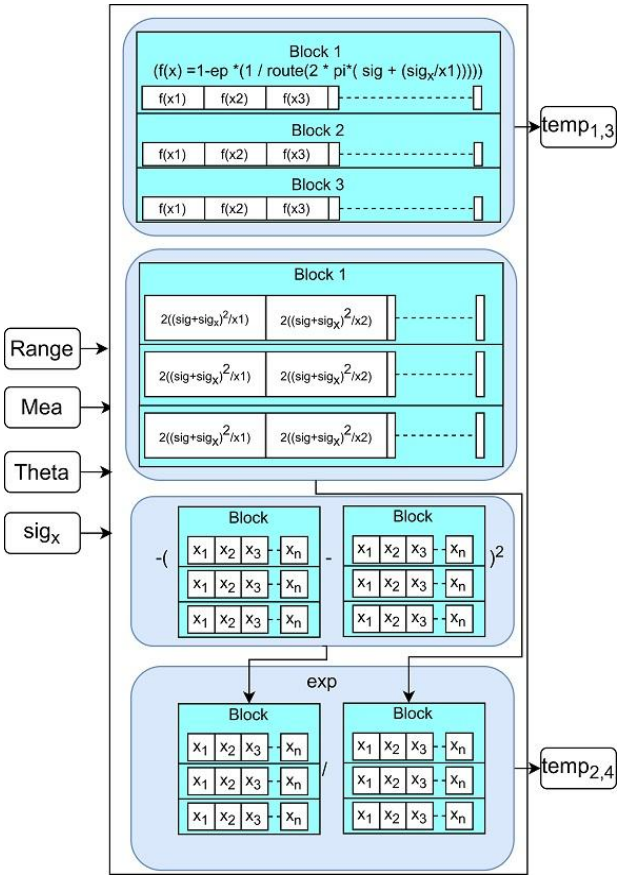


그림 7. 우도함수를 계산하기 위한 커널들
Fig. 7. Kernels for calculating the likelihood functions

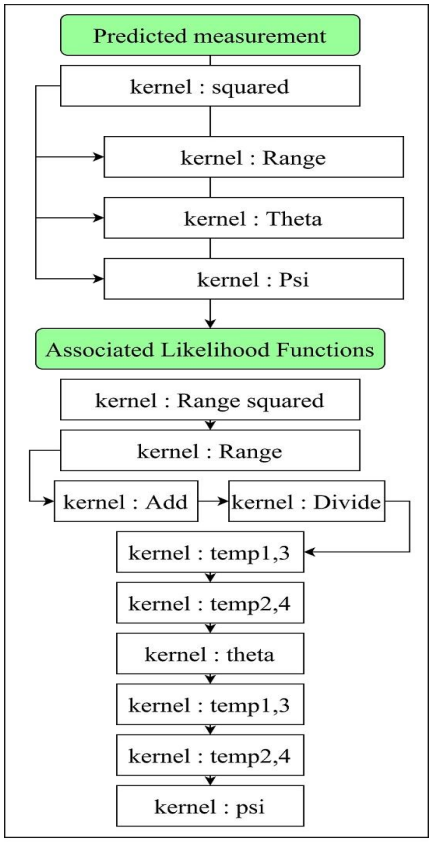


그림 8. 입자 군집 최적화에 사용되는 CUDA 커널 흐름도
Fig. 8. Flowchart of CUDA kernels in PSO

타낼 수 있다. 이러한 과정을 각각의 입자마다 반복하여 입자가 가지는 추정치를 구하여야 하므로 입자의 정보를 업데이트하는 과정보다 이 과정에서 많은 계산 시간이 소요된다.

(20)~(24)의 연산을 병렬 계산을 위하여 CUDA 커널로 정의할 때, 공통적인 연산들은 같은 커널을 사용할 수 있다. 그림 7에서 나타나듯이, (22)와 (24)의 식에서 $temp_2$과 $temp4$를 구하는 연산은 사용되는 노이즈의 값만 다르기 때문에, 노이즈 값만 다르게 하여 병렬 계산한다. 마찬가지로 (21)과 (23)의 식 또한 같은 형태를 띠고 있어 커널에 들어가는 값만 차이를 주면 되어 동일한 커널을 사용할 수 있다.

그림 8에서 CUDA를 사용하여 병렬 계산을 진행할 때, 사용되는 커널들의 전체적인 흐름도를 보여준다. 먼저 목표에 대한 상태 추정 부분을 병렬 계산한다. 여기서 얻어지는 거리 정보에 대한 추정을 사용하여, 입자들이 가진 정보를 업데이트 하기 전 단계인 가장 가능성 있는 추정치를 구하는 우도 함수 부분을 병렬 계산하여 가속화한다.

V. 실험 결과

1. 탄도 궤적 목표물 추적 결과

논문에서 입자 군집 최적화 알고리즘 중 목표의 상태 추정 부분과 추정치를 구하는 부분을 병렬화하여 고속 표적 추적을 위한 GPU로 입자 군집 최적화 기법을 가속화 한다. 제안된 가속화 된 알고리즘의 추적 결과는 탄도 표적 추적 시나리오에서 평가한다. 수치적 시뮬레이션을 위하여 식 (1)과 (2)의 동적 모델을 사용하여 실제 기준 궤적을 생성한다. 비산물의 공기역학적 항력과 중량은 [18]을 참고하여 설정하였다. 샘플링 간격은 Δt = 0.01초로 설정하였으며 총 시뮬레이션 시간은 3초가 소요되었다. 레이더 수신기 소음 모델의 표준 편차 n_R,n_θ,n_ψ는 각각 1m, 0.1deg, 0.1deg이며 글린트 노이즈$n_{G,\theta}$, $n_{G,\psi}$는 (25)의 식과 같은 가우시안 분포를 따른다.

$$p=(1-\epsilon)p_{G_1}+\epsilon p_{G_2} \tag{25}$$

위의 식에서 ϵ는 글린트 확률을 의미하며, P_{G_1}은 $p_{G_1}\sim N(0,\ 0.1^2)$, P_{G_2}는 $p_{G_2}\sim N(0,\ 1^2)$를 따르는 가우스 모델이다. 추적 모션 모델은 식 (5)를 따르는 Singer 모델이며 측정 모델은 식 (12)를 사용하여 얻는다. 레이더의 위치는 지상에 고정되어 있는 것으로 가정하고, 탄도 표적은 중력 및 공기역학적 항력을 고려하여 고속으로 움직인다고 가정 한다. 따라서 탄도 표적의 속도는 시뮬레이션 시간에 따라서 달라진다.

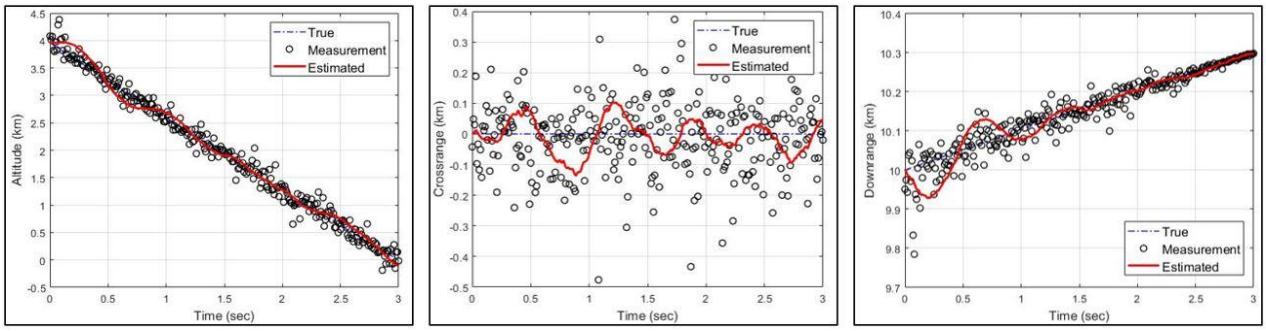


그림 9. 한 입자의 이동 횟수가 5일 때의 레이더 측정 및 트루 모델과 추정한 목표 결과 범위 비교
Fig. 9. When the number of epoch used in PSO was 5, Estimated target model compared to calculated by radar measurements and true position.

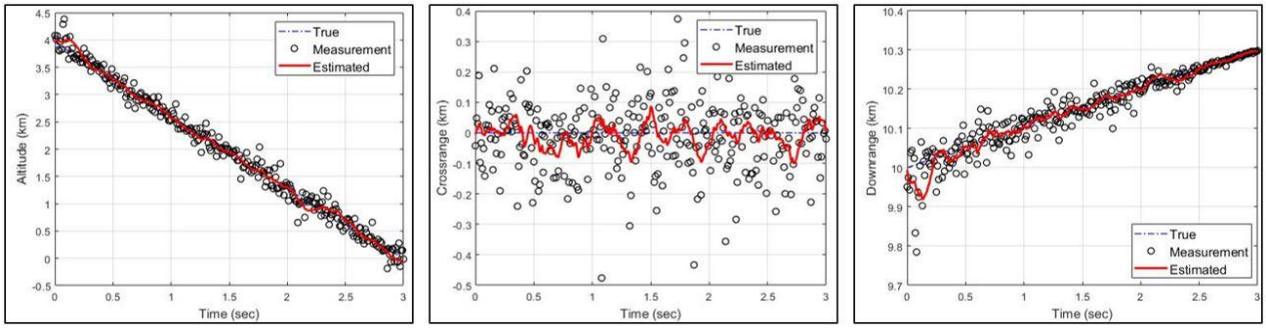


그림 10. 한 입자의 이동 횟수가 10일 때의 레이더 측정 및 트루 모델과 추정한 목표 결과 범위 비교
Fig. 10. When the number of epoch used in PSO was 10, Estimated target model compared to calculated by radar measurements and true position.

궤적 및 상태 추정의 결과는 그림 10에 나타난다. 입자 군집 최적화 알고리즘에 사용되는 입자 수와 입자의 이동횟수가 너무 적으면 수렴이 잘되지 않거나 에러 바운드가 크게 튀는 현상이 발생한다. 목표물 추적 알고리즘이 잘 수렴하도록 입자 군집 최적화 알고리즘의 입자는 1000개로 정의하였고 한 입자가 최적점을 향하여 이동하는 횟수는 10번으로 정의하였다. 초반 부분의 에러 바운드가 튀는 현상이 있지만 이것은 입자 군집 최적화 기법의 특성상 처음에는 랜덤한 값이 들어가기 때문이고 반복이 진행될수록 최적의 값을 찾아가기 때문에 후에는 목표 상태 추정이 제대로 동작하는 것을 알 수 있다. 비교를 위하여 한 입자의 이동횟수를 5번으로 조정하였을 때의 하향 범위 비교와 고도 범위 비교 결과를 그림 9에 나타내었다. 그림 9, 10의 왼쪽 그림은 트루 모델과 추정된 고도 범위를 (Altitude) 비교한 그림이며 중간 그림은 교차 범위 (Cross-range), 오른쪽 그림은 목표 하향 범위 (Down-range)의 결과 비교를 보여준다.

2. 병렬화 및 가속화 결과

본 논문에서의 병렬화 및 가속화 실험은 NVIDIA Jetson Xavier 환경에서 진행되었으며, CPU는 6-core NVIDIA Carmel ARM v8.2 64-bit GPU는 NVIDIA Volta architecture with 384 NVIDIA CUDA cores and 48 Tensor cores 사양을 가진다. 실험에서는 CPU만을 사용한 병렬화 하지 않은 입자 군집 최적화 기법과 CPU와 GPU를 함께 사용하여 병렬화

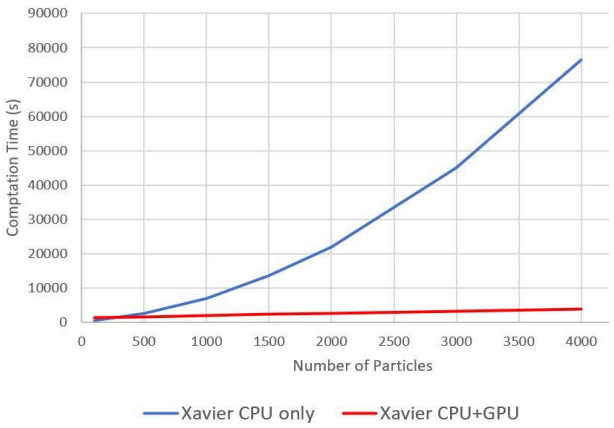


그림 11. 전체 목표 추적 알고리즘의 계산 시간 비교
Fig. 11. Computation time of Whole target tracking algorithm

된 입자 군집 최적화 알고리즘을 비교한다.

그림 11에서는 목표 추적 알고리즘 전체 수행시간 비교가 나타나 있으며 그림 12과 표 2 에서는 그 중 입자 군집 최적화 알고리즘만의 CPU와 GPU 사용 간의 수행시간 비교를 나타낸다. 목표 추적을 위한 알고리즘은 300번의 반복을 진행하여 수행되었으므로, 표 2의 계산시간은 그중 1번의 반복에서 입자 군집 최적화 기법이 수행된 시간을 의미한다. 그림 11, 12과 표 2에서 입자 군집 최적화 기법에 사용된 입자 수가 100, 500개 즉, 입자 수가 적을 때는 CPU만을

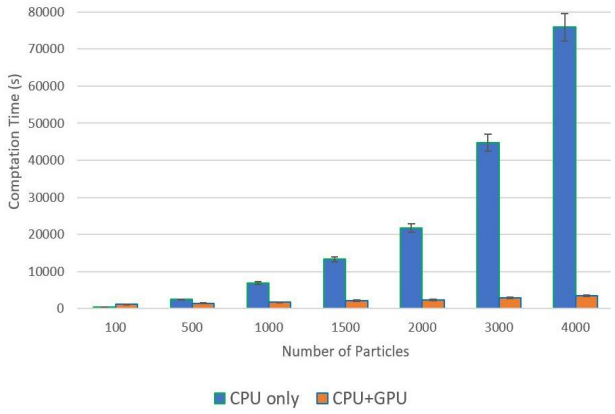


그림 12. 목표 추적 알고리즘 중 입자 군집 최적화 기법의 계산 시간 비교
Fig. 12. Computation time of Particle Swarm Optimization

표 2. 한 step 당 입자 군집 최적화 기법의 계산 시간 비교
Table 2. Computation time comparison of Particle Swarm Optimization per one step

Number of Particles	only CPU	CPU+GPU
100	1.2031	3.8219
500	8.1129	4.7291
1000	22.8975	5.4873
1500	44.4302	7.0893
2000	72.3309	7.7308
3000	149.1618	9.5788
4000	252.9826	11.4243

사용했을 때 계산시간이 더 적게 소요되거나 비슷하게 소요된다. 하지만 입자 수가 1000개 이상으로 사용되었을 때에는 CPU만을 사용하여 알고리즘을 진행하였을 때보다 GPU를 사용하여 앞서 설명한 연산시간이 많이 걸리는 부분을 병렬화하여 진행한 부분이 더 시간이 적게 걸리는 것을 알 수 있다. 병렬 계산을 위하여 CUDA를 사용하면 필연적으로 오버헤드가 발생한다. 오버헤드 시간은 CUDA 사용을 위한 CUDA 초기화, 선언된 커널들을 불러오는 시간 등을 포함하게 된다. 그림 11, 12 그리고 표 2에서 GPU를 사용하여 계산 시간을 진행하였을 때 오버헤드 시간까지 포함한 결과이다. 오버헤드 시간은 입자 군집 최적화 기법의 입자 수와 관계없이 평균적으로 97초 정도 소요되었다.

표 3과 그림 13에서는 입자 군집 최적화 기법을 입자 수를 1000개로 설정하여 진행된 탄도 목표 추적 알고리즘의 전체 계산 시간을 파트별로 보여준다. 앞서 설명한 계산 시간이 많이 소요된다고 식별된 목표에 대한 상태 추정 부분과 입자 정보 업데이트 부분을 GPU를 사용하여 병렬화하였을 때, 해당 파트들이 CPU만을 사용하였을 때보다 계산시간이 현저히 줄어 가속화가 성공하였다는 것을 알 수 있다. CUDA 사용을 위한 오버헤드를 포함하더라도 전체 계산 시간은 GPU를 사용하여 병렬화한 방법이 계산 시간이 적게 소요되었다.

표 3. 목표 추적 알고리즘의 파트별 계산 소요 시간
Table 3. Computation time for each part of the target tracking algorithm.

Part	Computation time(s)	
	Only CPU	CPU + GPU
CUDA init delay	-	94.559
Others	746.68	819.849
predict measurement/ associated likelyhood func	6314.839	1011.639
Total	7061.519	1926.047
Without CUDA init delay	7061.519	1831.488

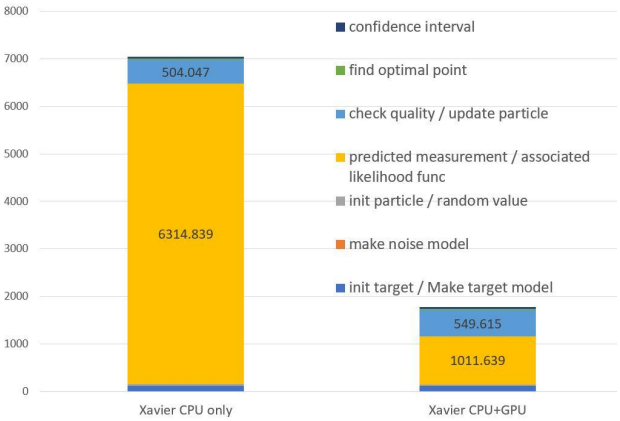


그림 13. 목표 추적 알고리즘의 파트별 계산 소요 시간
Fig. 13. Computation time for each part of the target tracking algorithm.

Ⅵ. 결 론

본 논문에서는 고속으로 움직이는 탄도 목표물을 추적하는 문제에 대하여 입자 군집 최적화 기법을 사용하였으며 실시간성을 가지기 위하여 GPU를 사용한 가속화를 진행하였다. 입자 군집 최적화 기법을 사용하여 목표로 하는 탄도 목표물의 움직임과 각도 등의 상태 추적을 하였다. 입자 군집 최적화 기법의 특성상 알고리즘의 반복이 진행되지 않은 초반에는 입자들이 최적의 값의 정보를 가지고 있지 않기 때문에 에러 바운드가 튀는 부분이 있으나, 알고리즘이 진행되어 반복이 진행되면서 입자들이 가지고 있는 전역, 국소 최적점의 정보를 따라 최적점을 찾아 움직이면서 상태 추적이 정상적으로 진행되었다.

탄도 목표물의 추적 알고리즘에서 입자 군집 최적화 기법에서 대부분의 시간이 소요되었으며, 특히 그 중 입자가 가지고 있는 정보를 이용한 목표에 대한 상태 추정 부분과 입자 정보 업데이트 부분에서 계산 시간이 많이 소요되는 것을 식별하였다. 입자 정보 업데이트 부분에서도 입자 정보를 업데이트 하기 전 업데이트에 필요한 값을 구하는 부분에서 시간이 많이 소요되었다. 계산 시간이 많이 소요된다고 식별된 두 부분을 CUDA를 사용하여 GPU를 이용한 병

렬화하였으며 CUDA 사용을 위하여 필연적으로 발생하는 오버헤드 시간을 고려하더라도 CPU만을 사용하여 알고리즘을 진행하였을 때보다 계산 시간이 줄어들었다. 탄도 목표물 추적 시 CPU만을 사용하여 목표물의 상태 추정을 하는 것보다 GPU를 사용한 본 논문에서 제안된 병렬화 된 입자군집 최적화 기법을 적용한 알고리즘이 계산 시간을 더 적게 소요하기 때문에 CPU만을 사용하였을 때보다 실시간성을 더 가지게 된다.

References

[1] G. M. Siouris, "Missile Guidance and Control Systems," Verlag, Berlin: Springer, pp. 113 - 119, 2004.

[2] G. A. Hewer, R. D. Martin, J. Zeh, "Robust Preprocessing for Kalman Filtering of Glint Noise," IEEE Transactions on Aerospace and Electronic Systems, Vol. 23, No. 1, pp. 120 - 128, 1987.

[3] G. Oh, H. Lee, H. Lee, "Hierarchical Correlation-based Anomaly Detection for Vision-based Mask Filter Inspection in Mask Production Lines," IEMEK J. Embed. Sys. Appl.., Vol. 16, No. 6, pp. 277 - 283, 2021 (in Korean).

[4] Y. Zhou, Y. Tan "GPU-based Parallel Particle Swarm Optimization", 2009 IEEE Congress on Evolutionary Computation, pp. 1493-1500, 2009.

[5] L. Mussi, S. Cagnoni, F. Daolio, "GPU-Based Road Sign Detection Using Particle Swarm Optimization", 2009 Ninth International Conference on Intelligent Systems Design and Applications. IEEE, pp. 152-157, 2009.

[6] A. Keshavarz-Mohammadiyan, H. Khaloozadeh, "PSO-PF Target Tracking in Range-based Wireless Sensor Networks with Distance-dependent Measurement Noise", 2015 23rd Iranian Conference on Electrical Engineering (ICEE), pp. 911-915, 2015.

[7] W. Ding, W. Fang "Target Tracking by Sequential Random Draft Particle Swarm Optimization Algorithm", 2018 IEEE International Smart Cities Conference (ISC2), 2018.

[8] S. S. Rayala, N. A. Kumar, "Particle Swarm Optimization for Robot Target Tracking Application", materialstoday PROCEEDINGS, Vol 33, No. 7, pp. 3600-3603, 2020.

[9] S. Yang, Q. Ma, W. Huang, "Particle Swarm Optimized Unscented Particle Filter for Target Tracking", 2009 2nd International Congress on Image and Signal Processing, pp. 1-5, 2009.

[10] Z. Cheng, L. Fan, Y. Zhang, "Multi-agent Decision Support System for Missile Defense Based on Improved PSO Algorithm" Journal of Systems Engineering and Electronics, Vol. 28, No.3, pp. 514-525. 2017.

[11] X. Zheng, Y. Gao, W. Jing, Y. Wang, "Multidisciplinary Integrated Design of Long-range Ballistic Missile Using PSO Algorithm" Journal of Systems Engineering and Electronics, Vol. 31, No. 2, pp. 335-349, 2020.

[12] J. G. dos Santos Júnior, J. P. S. do Monte Lima, "Particle Swarm Optimization for 3D Object Tracking in RGB-D Images", Computers & Graphics, Vol. 76, pp. 167-180, 2018.

[13] B. Rymut, B. Kwolek, "Real-time Multiview Human Pose Tracking Using Graphics Processing Unit-accelerated Particle Swarm Optimization", Concurrency and Computation: Practice and Experience, Vol. 27, pp. 1551-1563, 2014.

[14] R. A. Singer, "Estimating Optimal Tracking Filter Performance for Manned Maneuvering Targets," IEEE Transactions on Aerospace and Electronic Systems, Vol. 4, pp. 473-483, 1970.

[15] X. R. Li, V. P. Jilkov, "Survey of Maneuvering Target Tracking. Part I. Dynamic Models," IEEE Transactions on Aerospace and Electronic Systems, Vol. 39, No. 4, pp. 1333-1364, 2003.

[16] J. Kim, M. Tandale, P. K. Menon, "Particle Filter for Ballistic Target Tracking with Glint Noise," Journal of Guidance, Control, and Dynamics, Vol. 33, No. 6, pp. 1918 - 1921, 2010.

[17] H. M. Park, J. S. Kwon, T. H. Hwang, D. S. Kim, "Implementation of Integrated CPU-GPU for Efficient Uniform Memory Access Method and Verification System," IEMEK J. Embed. Sys. Appl.., Vol. 11, No. 2, pp. 57 - 65, 2016 (in Korean).

[18] D. C. Wright, D. Kadyshev, "An Analysis of the North Korean Nodong Missile," Science & Global Security, Vol. 4, No. 2, pp. 129-160, 1994.

Yunho Han (한 유 호)

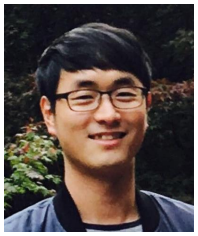
2021 Electronic Engineering from Kumoh National Institute of Technology (B.S.)

2021 ~ Department of IT Convergence Engineering, Kumoh National Institute of Technology (M.S. Candidate)

Field of Interests: Algorithm acceleration with GPU and FPGA, embedded system, Path Planning

Email: gksfbsgh@kumoh.ac.kr

Heoncheol Lee (이 헌 철)



2006 Electronic-Electrical Engineering and Computer Sciences from Kyungpook National University (B.S.)

2008 Electrical Engineering and Computer Sciences from Seoul National University (M.S.)

2013 Electrical Engineering and Computer Sciences from Seoul National University (Ph.D.)

2019~Department of IT Convergence Engineering, School of Electronic Engineering, Kumoh National Institute of Technology (Assist. Prof.)

Career:

2011 Researcher, ASRI, Seoul National University

2013 Senior Researcher, Agency for Defense Development

2019 Technical Adviser, LG Electronics

Field of Interests: Vision-based Anomaly Detection, SLAM, Path Planning, Algorithm Acceleration, Deep Learning

Email: hclee@kumoh.ac.kr

Hyeokhoon Gwon (권 혁 훈)

2002 Aerospace Engineering from KAIST University (B.S.)

2005 Aerospace Engineering from KAIST University (M.S.)

2020 Aerospace Engineering from KAIST University (Ph.D.)

Career: 2009 Researcher, LIGNEX1

Field of Interests: Convex optimization, optimal control, Guidance and autopilot design and nonlinear control

Email: hhkwon22@lignex1.com

Wonseok Choi (최 원 석)

2006 Electronic-Electrical Engineering from Hongik University (B.S.)

2017 Defense Engineering from Yonsei University (M.S.)

Career: 2007 Researcher, LIGNEX1

Field of Interests: FPGA, Filter, Sensor, Munition Design

Email: wonseok.choi@lignex1.com

Bora Jeong (정 보 라)

2020 Electronic-Electrical Engineering from Hanyang University (B.S.)

Career: 2020 Researcher, LIGNEX1

Field of Interests: FPGA, Filter, Sensor, Munition Design

Email: bora.jung@lignex1.com