가중치 완전 탐색 알고리즘을 이용한 유도탄 측추력기 반비례 제어 시스템 설계

2024-2 AJOU SOFTCON

팀 원 정 유 민 지도교수 박 종 호 지도교수 임 재 성

서론

- 유도탄의 측추력기는 급격한 기동을 위해 다양한 알고리즘으로 제어된다.
- 기존에는 그리디 알고리즘을 이용하여 측추력기를 제어하였으나, 미사일의 자세값이 정확하지 않고 심하게 떨린다는 단점이 있었다.
- 본 논문에서는 모든 측추력기가 만들어 낼 수 있는 모멘트 조합을 고려하는 완전 탐색 알고리즘을 제안한다. 제안한 알고리즘을 이용하여 측추력기 제어 로직을 구현하고, 이를 그리디 알고리즘과 비교 분석한다.
- 나아가 제안한 완전 탐색 알고리즘의 과한 연산량 해소를 위해 측추력기마다 가중치를 부여한 후 특정 반비례 그래프보다 작아지면 측추력기를 바로 선택하는 가중치 완전탐색 알고리즘을 제안한다.

연구 환경

1. 유도탄 모델

- 본 연구에서는 PAC-3 MSE 유도탄을 기반으로 한 모델을 사용하였다.
- 그림 1은 본 연구에서 사용한 유도탄 모델을 나타낸다.

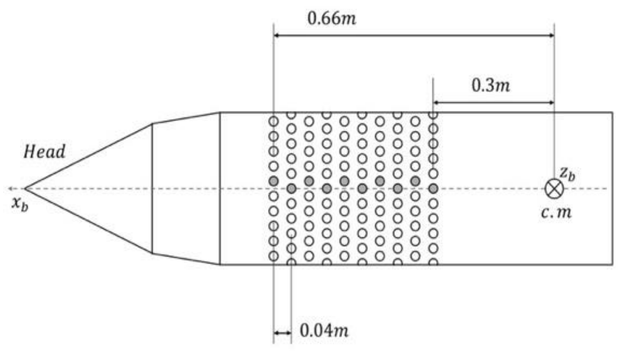

그림 1. PAC-3 MSE 유도탄 모델 형상

2. 시뮬레이션 환경

- 연구에 사용된 시뮬레이션은 MATLAB R2023b를 이용하여 그림 2의 다이어그램을 따라 구현하였다.

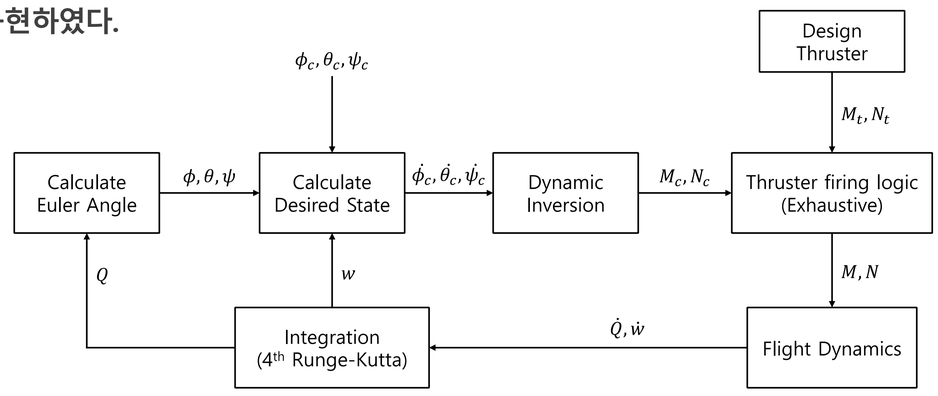

그림 2. 제어 시뮬레이터 다이어그램

- 그림 2에서 ϕ,θ,ψ는 Euler 각도이고 ϕ_c,θ_c,ψ_c 는 Euler 각도 명령이다. M, N은 Pitch, Yaw 방향 회전 모멘트이며, M_c, N_c는 모멘트 명령을, M_t, N_t는 측추력기가 만들어내는 모멘트를 의미한다. 또한, Q는 쿼터니언이고, w는 각속도이다.

제안기법

$Input: M_c, N_c, i_u$

$Output: M, N, i_u, Selected\_thruster$

```
procedure EXHAUSTED ALGORITHM(M_c, N_c, i_u)
    if time_step is a multiple of 0.1/dt then
        for all (N_t, M_t) in TM do
            diff_TM ← sqrt((N_c − N_t)^2 + (M_c − M_t)^2)
        end for
        find index such that diff_TM[index] = min(diff_TM)
        Selected_thruster ← index
        i_u ← i_u − Selected_thruster
        M, N ← TM(index)
        Count = 0
    else if Count < tb/dt then
        M, N ← TM(index)
        Count ← Count + 1
    else
        M, N ← 0
    end if
    return M, N, i_u, Selected_thruster
end procedure
```

표 1. 완전 탐색 알고리즘 기반 측추력기 선정 의사코드

- 본 논문에서 제안하는 완전 탐색 알고리즘은 측추력기의 모든 모멘트 조합을 탐색한다. 이후 탐색 결과 명령값과 가장 유사한 모멘트를 만드는 측추력기 조합을 찾아 선택하는 방법이다.
- 표 1은 완전 탐색 알고리즘을 이용한 측추력기 선정 과정을 보여준다.
- 의사코드에서 i_u는 가용 측추력기를, TM은 측추력기가 생성할 수 있는 모든 모멘트 조합을 의미한다.

결과 및 분석

- 시뮬레이션을 이용하여 4가지 경우의 알고리즘을 비교했다. 측추력기를 최대 2개까지 사용하는 그리디 알고리즘 Greedy 2와 완전 탐색 알고리즘 Exhaustive 2, 3개까지 사용하는 Greedy 3와 Exhaustive 3이다.
- 그림 3은 4가지 알고리즘의 시간에 따른 자세각 추종 성능을 나타낸다. 모든 경우 자세각 명령값 45도를 준수하게 추종하고 있다.
- 또한 완전 탐색 알고리즘에 비해 그리디 알고리즘의 오차가 크다. (그림 4)

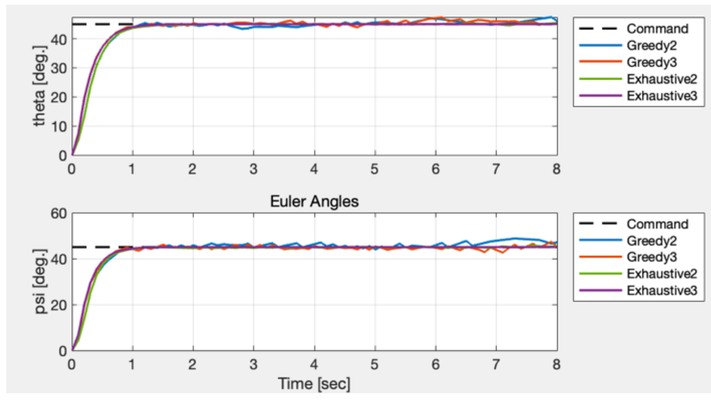

그림 3. 알고리즘별 자세각 제어 결과 (0~8s)

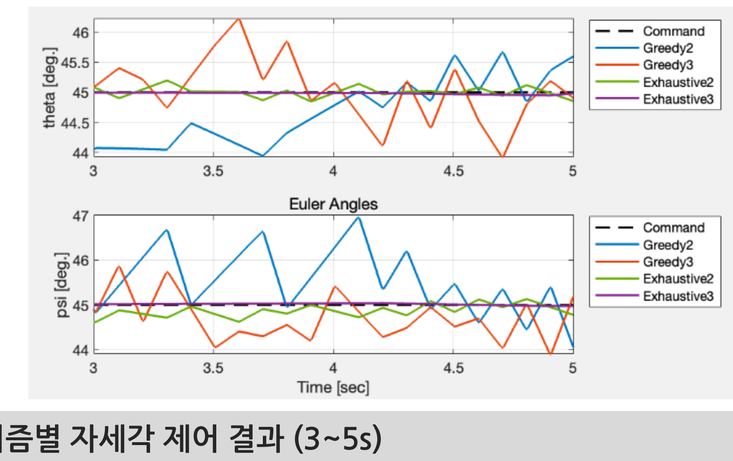

그림 4. 알고리즘별 자세각 제어 결과 (3~5s)

- 그리디 알고리즘은 모멘트 계산값이 명령값과 유사해지면 더 이상 제어하지 않는 반면, 완전 탐색 알고리즘은 모든 모멘트 조합을 고려하여 아주 작은 모멘트도 생성할 수 있다.
- 따라서 완전 탐색 알고리즘을 사용했을 때 유도탄이 더 안정적인 비행이 가능하다고 볼 수 있다. 그러나 작은 모멘트에도 측추력기를 사용하기 때문에 유한한 개수의 측추력기를 과도하게 사용할 가능성도 존재한다.
- 이를 방지하기 위해 측추력기에 가중치를 부여한 가중치 완전 탐색 알고리즘을 제안한다.
- 가중치는 측추력기를 다회성으로 가정한 후 일반 완전탐색 알고리즘을 이용하여 어느 측추력기가 가장 많이 사용되는지를 기준으로 하여 계산하였다.

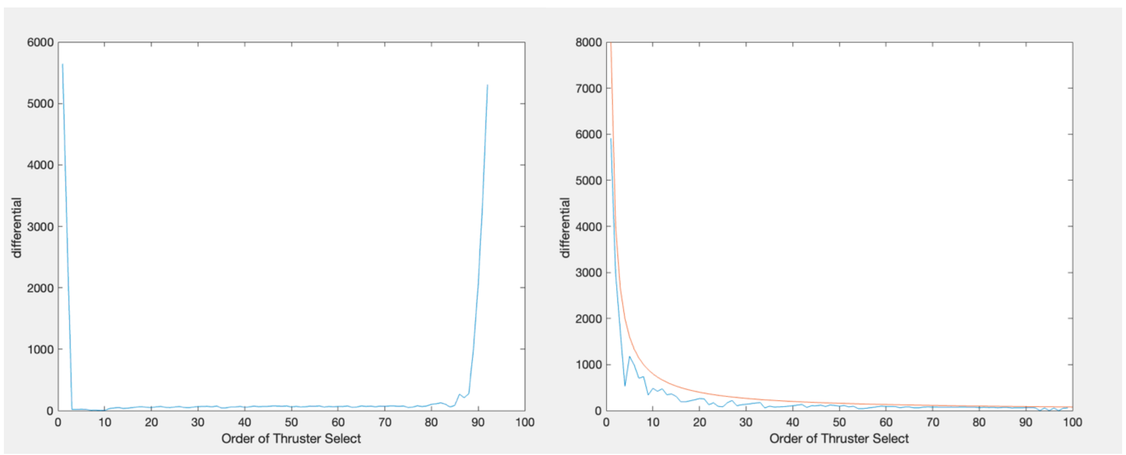

그림 5. 반비례 함수를 기준으로 한 differential (왼: 완전탐색 | 오: 가중치 완전탐색)

- 그림 5의 differential은 모멘트 명령값과 측추력기가 실제 생성한 모멘트의 차이 값이다.
- 반비례 함수보다 더 작은 differential 값을 생성하는 측추력기를 바로 선택하였다.
- 빨간색이 기준이 된 반비례 그래프이고 파란색이 실제 생성된 differential 값이다.
- 왼쪽의 기본 완전탐색의 경우 추종된 미사일의 자세를 유지하기 위해 0에 가까운 모멘트를 만들어내는 것을 확인할 수 있지만, 오른쪽의 반비례 가중치 완전탐색을 활용한 결과 이러한 불필요한 측추력기 사용이 사라져 더 효율적인 제어가 가능했다.

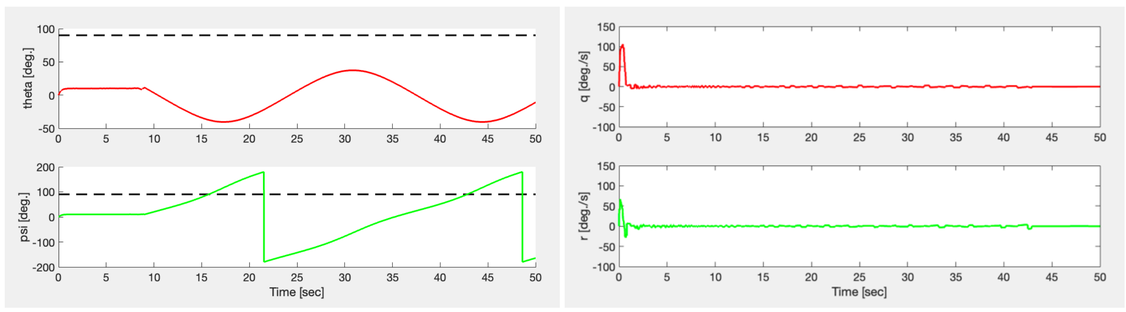

그림 5. 자세제어 결과 (왼: 완전탐색 | 오: 가중치 완전탐색)

- 그림 6의 자세제어 결과를 보면 완전탐색은 9초까지만 제어가 되고 진동하는 반면, 가중치 완전탐색은 50초 이후까지도 제어가 가능하다.
- 이를 통해 반비례 가중치 완전탐색 알고리즘이 기본 알고리즘에 비해 더 효율적으로 측추력기를 선택S하며 이를 통해 더 긴 시간 제어가 가능하다는 사실을 확인하였다.