Journal of KIIT. Vol. 22, No. 3, pp. 71-78, Mar. 31, 2024. pISSN 1598-8619, eISSN 2093-7571
http://dx.doi.org/10.14801/jkiit.2024.22.3.71

통합유도조종에서 실시간 모델 예측 제어를 위한 FPGA 기반 PD-IPM 가속화 기법

김대연*, 이헌철**, 최원석***[1], 정보라***[2], 조영기***[3]

Accelerated PD-IPM on FPGA for Real-Time Model Predictive Control in Integrated Guidance and Control Systems

Daeyeon Kim*, Heonchoel Lee**, Wonseok Choi***[1], Bora Jeong***[2], Youngki Cho***[3]

이 연구는 금오공과대학교 대학 학술연구비로 지원되었음(2022년)

요 약

본 논문에서는 자동화된 시스템, 특히 군사 분야에서의 효과적인 기동 능력을 확보하기 위한 핵심 요소로써 통합 유도 및 제어 시스템을 다룬다. 최근 연구에서는 MPC(Model Predictive Control) 및 볼록 최적화에 관한 관심이 높아지고 있다. 그러나 볼록 최적화의 뛰어난 성능에도 불구하고 연산 요구량이 많아 임베디드 시스템에서의 실시간성 보장이 어려운 문제가 있다. 따라서 PD-IPM(Primal-Dual Interior-Point Method)을 이용한 볼록 최적화에 대한 FPGA 기반의 가속화 방법을 제안하고, 적용하여 시스템의 실시간성을 향상시키는 것에 중점을 두었다. 실험 결과에서는 제안된 알고리즘이 목표물 요격을 성공적으로 수행하면서도 전체 수행시간이 약 33% 단축되었다.

Abstract

This paper addresses the integrated guidance and control system as a key element to secure effective maneuverability, especially in the military domain. Recent research has shown a growing interest in Model Predictive Control(MPC) and convex optimization. However, despite the excellent performance of convex optimization, there is a challenge in ensuring real-time capability in embedded systems due to high computational requirements. Therefore, the paper proposes a method to accelerate convex optimization using Primal-Dual Interior-Point Method(PD-IPM) on FPGA-based platforms and emphasizes enhancing the real-time capability of the system. In the experimental results, the proposed algorithm successfully performs target tracking while reducing the overall execution time by approximately 33%.

Keywords

convex optimization, FPGA, intergrated guidance and control, probabilistic filter, parallelization

* 금오공과대학교 IT융복합공학과 석사과정
- ORCID: https://orcid.org/0009-0009-0558-6876
** 금오공과대학교 IT융복합공학과 부교수(교신저자)
- ORCID: https://orcid.org/0000-0003-2962-3474
*** LIG넥스원
- ORCID[1]: https://orcid.org/0009-0000-2663-4421
- ORCID[2]: https://orcid.org/0009-0009-7777-7954
- ORCID[3]: https://orcid.org/0009-0009-3608-2531

• Received: Jan. 29, 2024, Revised: Mar. 19, 2024, Accepted: Mar. 22, 2024

• Corresponding Author: Heoncheol Lee
Dept. of IT Convergence Engineering, School of Electronic Engineering
Kumoh National Institute of Technology, Korea
Tel.: +82-54-478-7476, Email: hclee@kumoh.ac.kr

Ⅰ. 서 론

현대 자율 시스템에서 효과적인 기동 능력을 확보하기 위해서는 유도 및 제어 시스템, 특히 IGC(Integrated Guidance and Control)가 핵심 요소이다[1][2]. 최근까지 많은 분야에서 통합 유도 및 제어 시스템의 중요성이 주목받아 왔지만, 특히 군사 분야에서는 불확실성과 비선형성의 존재와 제어 루프 간의 교차 연결 관계와 같은 연구 과제가 여전히 존재한다[3].

이에 대안으로, 최근 연구에서는 MPC(Model Predictive Control)에 대한 연구들이 두드러지게 등장하고 있다[4]. MPC는 다음 Time step에 대한 시스템 동작을 예측하며, 이를 기반으로 최적의 제어 입력을 선택함으로써 시스템 성능을 개선하는 방법으로 주목받고 있다[5]. 특히, 다양한 자율 시스템에서 MPC의 적용은 미래 예측과 제어 입력 최적화를 결합함으로써 시스템의 성능과 안정성을 동시에 고려하는 효과적인 방법을 제시하고 있다[6].

또한, 최근 연구에서는 볼록 최적화에 관한 관심이 높아지고 있다[7]. 볼록 최적화는 다양하고 복잡한 최적화 문제를 다루는 데에 유용하며, 특히 다양한 제약 조건 간의 균형을 맞추어야 하는 통합 유도 및 제어 설계에서 필수적이라고 할 수 있다.

하지만 볼록 최적화는 뛰어난 성능에도 불구하고 연산 요구량이 많아 발사체에 탑재되어야 하는 환경에서 여러 제약 조건이 발생하는, 즉 임베디드 시스템에서의 실시간성 보장이 어렵다[8]. 따라서 결정론적인 동작과 정확한 타이밍에 예측하도록 보장되어야 하는 탄도 미사일 분야에 IGC가 적용되기 위해서는 실시간성을 보장하는 것이 매우 중요하다.

이에 본 논문에서는 볼록 최적화의 한 방법인 PD-IPM(Primal-Dual Interior Point Method)를 MPC에 적용할 때 발생하는 실시간성 보장 문제를 해결하기 위한 FPGA(Field-Programmable Gate Array) 기반 병렬 및 가속화 방법을 제시하고, 통합 유도 및 조종 알고리즘 시뮬레이션 결과와 함께 수행시간 측면에서의 분석과 검증을 시행하였다.

2장에서는 PD-IPM을 이용한 통합 유도 및 조종 문제에서의 실시간성 보장 문제가 정의되고, 3장에서 제안하는 기법의 전체 구조 및 병렬 처리 기법의 소개와 수행시간 프로파일링을 시행하였으며, 4장에서는 통합 유도 및 조종 알고리즘의 시뮬레이션 결과와 제안된 방법의 수행시간 비교 및 자원 사용량에 대한 분석이 이루어진다.

Ⅱ. 문제점 기술

IGC 문제에서의 유도탄 종말 호밍 유도 기하 및 교전 기하, 시스템 특성 방정식, 시야각과 가속도를 포함한 제약 조건은 [9]에서 정의되었다. 결론적으로 MPC 기반 IGC의 최적화 문제는 다음과 같이 정의된다.

$$\min_{u_0,\cdots,u_{N-1}} \sum_{k=0}^{N-1}(x_k^T Q_k x_k + u_k^T R_k u_k) + x_N^T Q_N x_N \qquad (1)$$

여기서 식 (1)의 해는 표 1의 볼록 최적화에 주로 사용되는 PD-IPM을 이용하여 얻어지는데, 이때 식 (2)에 나타난 행렬 간의 연산을 포함하는 과정이 제시된 알고리즘에 의해 비용이 수렴할 때까지 반복된다[10]. Ψ와 $\tilde{A}$, $\tilde{B}$는 약 600×600의 행렬로 매우 크기가 큰 행렬이라고 할 수 있는데, 반복마다 대규모의 행렬 연산이 반복되고, 이는 통합 유도 및 제어 시스템의 수행시간이 늘어나는 것의 주요 원인이 된다.

$$\Psi = \begin{bmatrix} R_0 & & & \\ & Q_1 & & \\ & & \ddots & \\ & & & R_{N-1} \\ & & & & Q_N \end{bmatrix} \qquad (2)$$

$$\tilde{A} = \begin{bmatrix} B-I & & & \\ A & B-I & & \\ & \ddots & \ddots & \\ & & A & B-I \end{bmatrix}, \tilde{B} = \begin{bmatrix} -Ax_0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

$$\tilde{C} = \begin{bmatrix} 0 & C_1 & & \\ & \ddots & \ddots & \\ & & 0 & C_{N-1} \\ & & & 0 \quad C_N \end{bmatrix}, \tilde{D} = \begin{bmatrix} d \\ d \\ \vdots \\ d \end{bmatrix}$$

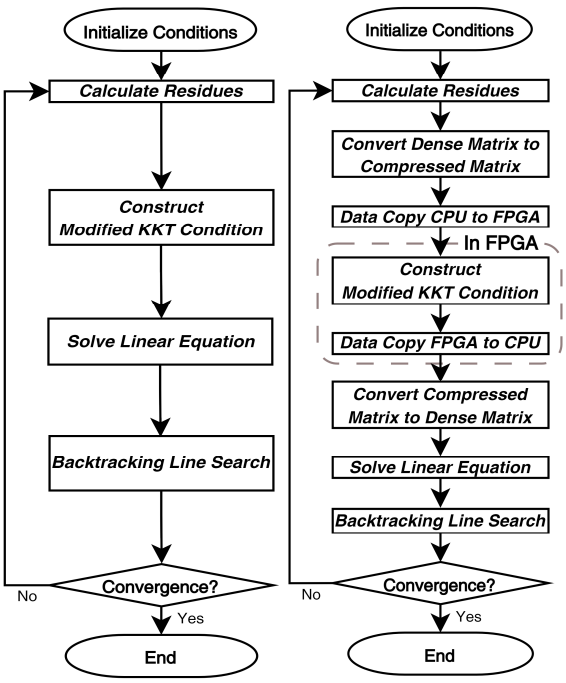
이전에는 이러한 문제를 해결하기 위해 GPU (Graphic Processing Unit) 기반 병렬 처리 방법이 연구되었다[11].

가속화를 위한 다양한 라이브러리 및 하드웨어가 개발되어 있어 접근성이 뛰어나다는 장점이 있지만, 저수준 연산에 특화된 GPU의 특성과 발사체에 탑재되어야 하는 환경은 시스템의 발열과 같은 제약 조건들이 심각하게 고려되어야 하는데, 이 경우 GPU를 기반으로 한 시스템 구성은 적합하지 않을 수 있다[12]. 그러므로 요구되는 제약 조건에 부합하는 통합 유도 및 제어 시스템을 설계해야 하는 상황에서 FPGA를 기반으로 한 가속화 방법을 설계함으로써 실시간성을 높여야 할 필요성이 있다.

III. 제안하는 기법

3.1 전체 구조

그림 1은 기존 및 제안된 알고리즘의 전체적인 흐름을 시각화한 것이다. 기존 알고리즘의 경우 연산이 CPU에서만 수행되지만, 제안된 알고리즘은 연산이 서로 다른 프로세서에서 동시에 수행된다.



(a) 기존 알고리즘 (b) 제안된 방법
(a) Baseline (b) Proposed method

그림 1. 기존 및 제안된 방법의 흐름도
Fig. 1. Flowcharts of baseline method and proposed method

특히 Construct modified KKT condition 과정에서 발생하는 대규모 행렬을 희소 행렬로 변환하고, 변환된 희소 행렬 연산 처리를 FPGA에서 수행함으로써 압축된 행렬에 대한 병렬 처리가 가능하다. 따라서 제안된 방법을 적용하는 경우 수행시간 측면에서 기존 연산 시스템과 비교했을 때 효율적인 연산이 이루어진다.

FPGA는 하드웨어 가속기의 역할을 하므로 설계된 하드웨어에 적합한 데이터로 변환하는 과정을 수행한다. 이는 적절한 병렬 연산 구조를 가지는 하드웨어와 해당 시스템에 알맞은 데이터로의 변환을 통해 알고리즘의 수행 속도를 높이고, 전체적으로 성능이 향상될 수 있다.

표 1. Primal-Dual 내부점 방법
Table 1. Primal-Dual interior point method

Algorithm 1: Primal-Dual interior point method

Choose: $\alpha \in (0,1), y^0 > 0, u^0 > 0, v^0 > 0, \epsilon > 0$

K: Maximum iteration: $f_0(y) = y^T \Psi y$

while:

$-f(y^k)^T u^k > \epsilon$ or $(\|r_{prim}\|_2^2 + \|r_{dual}\|_2^2)^{\frac{1}{2}} > \epsilon$

// Find update direction

compute: $f(y^k) = \widetilde{C} y^k - \widetilde{D}$

$E = \nabla^2 f_0(y) + \sum_{i=1} u_i \nabla^2 f_i(y)$

$F = -diag(u) \nabla f(y)^T$

$G = -diag(f(y))$

solve:

$$\begin{bmatrix} E & \nabla f(y) & \widetilde{A}^T \\ F & G & 0 \\ \widetilde{A} & 0 & 0 \end{bmatrix} \begin{bmatrix} \triangle y^k \\ \triangle u^k \\ \triangle v^k \end{bmatrix} = - \begin{bmatrix} r_{dual} \\ r_{cent} \\ r_{prim} \end{bmatrix}$$

$\theta = \min(1, \min(-u_k^i / \triangle u_k^i : \triangle u_k^i < 0))$

// Backtracking line search to find theta

while:

$\| r(y^+, u^+, v^+) \| > (1 - a\theta) \| r(y, u, v) \|$

$y^+ = y^k + \theta \triangle y^k, u^+ = u^k + \theta \triangle u^k, v^+ = v^k + \theta \triangle v^k$

compute:

$f^+ = \widetilde{C} y^+ - \widetilde{D}, r_{dual}^+, r_{prim}^+, r_{cent}^+$

$\theta = \alpha \theta$

end

// Primal-Dual Update

$(y^{k+1}, u^{k+1}, v^{k+1})$

$:= (y^k + \theta \triangle y^k, u^k + \theta \triangle u^k, v^k + \theta \triangle v^k)$

end

3.2 연산 시간 프로파일링

그림 2는 알고리즘의 각 부분에서 시간이 얼마나 소요되었는지를 보여준다. ARM Cortex-A53 프로세서 및 FPGA를 탑재한 Xilinx UltraScale+ MPSoC ZCU104 Evaluation Kit 개발보드 기준 CPU만 사용하였을 때의 수행시간을 측정하였다. 결과적으로 연산 시간을 가장 많이 차지하는 부분은 Solve lineqr equation 부분으로 8957ms가, 다음으로 Construct modified KKT Condition 부분에서는 3922ms가 소요되었다. 이러한 연산 시간 프로파일링을 통해 특정 부분에서 연산 시간 소모가 많이 일어나고 있음을 확인할 수 있다. 이때 가장 많은 시간이 소요되는 부분에 대한 연산을 병렬로 수행 가능한지 확인 후 병렬 처리를 통해 성능을 높일 수 있는지 생각해볼 수 있는데, 이종 프로세서 간 데이터 교환에 필수적인 지연 등을 심각하게 고려하여야 한다.

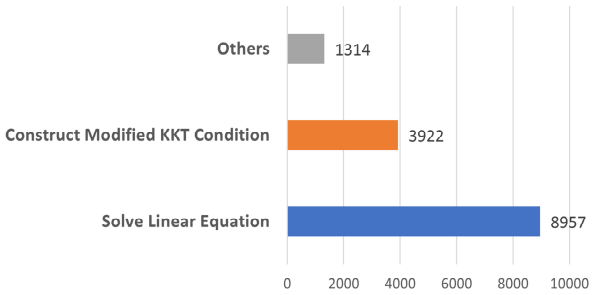


그림 2. PD-IPM 알고리즘 수행시간 측정 결과 (ms)
Fig. 2. Evaluation results of PD-IPM algorithm execution time (ms)

3.3 대규모 희소 행렬 계산 병렬화

대규모 행렬의 경우 각 원소에 접근하여 곱한 후 다시 더하는 행렬 곱의 특성상 매우 많은 연산이 필요한데, 행렬이 희소성을 가지는 경우 연산을 압축하여 수행한다면 연산 횟수를 크게 줄일 수 있다. 일반적으로 CSC(Compressed Sparse Column), CSR (Compressed Sparse Row)와 같은 희소 행렬로 변환하는 것과 같이 행렬 자체에 대한 변환이 시도되었으나[13]. 연산을 수행하는 하드웨어를 직접 구성하는 경우 연산 과정을 행렬 구조에 맞게 설계하는 방법을 생각해볼 수 있다. 그러므로 특정 패턴이 반복적으로 나타나는 행렬이 희소성을 가진다면 해당 패턴에 해당하는 연산만 수행하는 하드웨어를 구성할 수 있는데, 이를 통해 0이 아닌 수를 가지는 원소에 대한 계산만 수행함으로써 불필요한 연산을 회피할 수 있다.

PD-IPM 알고리즘에서 Modified KKT condition은 다음 식 (3)과 같이 얻어진다.

$$S=\frac{1}{2}\left(\begin{bmatrix}2P & -A^TB & C^T\\ C & 0 & \end{bmatrix}+\begin{bmatrix}2P & -A^TB & C^T\\ C & 0 & \end{bmatrix}^T\right) \quad (3)$$

여기서 P는 covariance, A는 inequality constraint를, C는 equality constraint, B는 equality constraint에서 계산된 행렬을 의미한다. 여기서 A와 B의 행렬 원소 구조는 알고리즘의 초기화 이후 수렴하기 전까지 같으므로, 각 행렬의 행렬 곱 결과 행렬도 iteration에 따라 항상 같은 특성을 가지며, 본 연구에서는 그림 3과 같이 특정한 패턴이 발견되는 희소 행렬에 대한 곱 연산을 간소화하고, 이를 병렬로 수행하는 IP(Intellectual Property)를 설계하여 알고리즘의 실시간성을 높였다.

또한, Solve linear equation 부분은 numpy 라이브러리에 내장된 Fortran 기반 solver가 사용되었는데, 해당 solver는 일반적인 행렬 계산에 적용되기 위해 매트릭스의 특성을 분석하는 과정이 있지만 이러한 과정은 연산 수행시간에 유의미한 영향을 줄 수 있다. 이때 행렬의 특성이 일정한 것을 이용하여 적절한 solver를 사용하도록 재구성하여 매트릭스 특성을 분석하는 불필요한 과정이 제거된 solver를 사용하였다. 결론적으로 입력 행렬 특성이 일정하므로 저수준 연산이 가능하게 되어 수행시간이 단축될 수 있다.

IV. 결 과

4.1 환경 셋업

개발 환경은 3장과 같으며, CPU와 FPGA 간 데이터 통신에는 AXI Interface, 비트스트림 로드 및 라이브러리 관리가 쉬운 Petalinux 기반 PYNQ OS를 사용하였다[14].

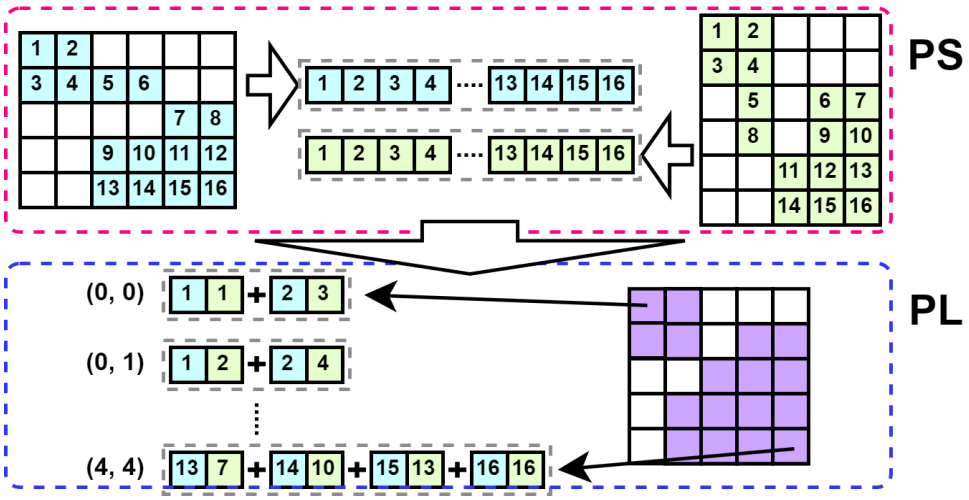


그림 3. 희소행렬 변환 및 연산 예시
Fig. 3. Example of sparse matrix transformation and operations

구현된 IP의 FPGA 자원 사용량은 표 2에 나타난 것과 같으며, LUT의 경우 개발 보드 기준 자원 사용량이 14%, LUTRAM은 5%, FF가 16%, BRAM이 1%, DSP가 11%, BUFG가 1%였으며, 제안된 알고리즘을 구현하고 가용한 자원이 약 80%이었다. 시뮬레이션을 위해 미사일과 표적의 속도는 Z_δ = 380 m/s, Z_δ = 380 m/s으로 가정하였다. 또한 미사일 초기 상태의 변수는 Υ_{m0} = 2 deg, Z_{m0} = 20 m. 부등식의 제약 조건 매개변수는 $a_{\max}$ = 10 G, $\sigma_{\max}$ = 5 deg, $\dot{\sigma}_{\max}$ = 20 deg/s로 설정하였다.

표 2. FPGA 자원 사용량
Table 2. FPGA resource utilization

Resource	Available	Utilization
LUT	230400	33092
LUTRAM	101760	4800
FF	460800	73240
BRAM	312	2
DSP	1728	186
BUFG	544	2

4.2 결과 및 분석

그림 4는 Matlab에서 도출된 통합 유도 및 제어 알고리즘의 결과 중 iteration에 따른 cost를 나타내는데, 32 iteration 끝에 cost가 수렴하는 것을 보여준다. 또한 그림 5는 목표물과 미사일 간의 상대적인 변위를 나타내며, 값이 감소하여 0에 수렴함에 따라 목표물의 요격에 성공하였다. 그림 6에서 (a)는 액추에이터의 명령 및 응답을, (b)는 유도탄의 가속도, (c)는 탐색기 시선각, (d)는 시선각속도를 나타낸다.

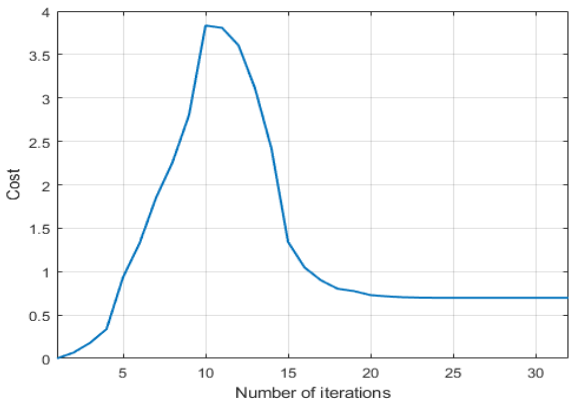


그림 4. 시뮬레이션 결과에서 비용 변화
Fig. 4. Cost variation in simulation results

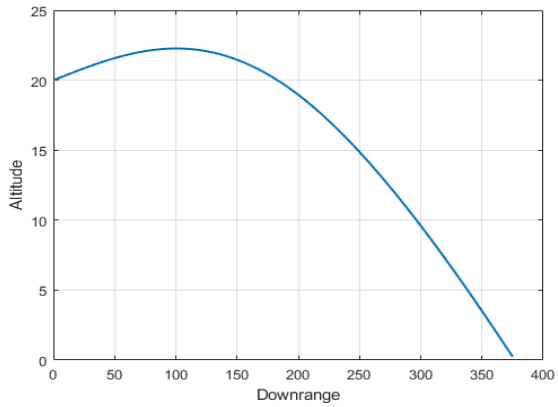
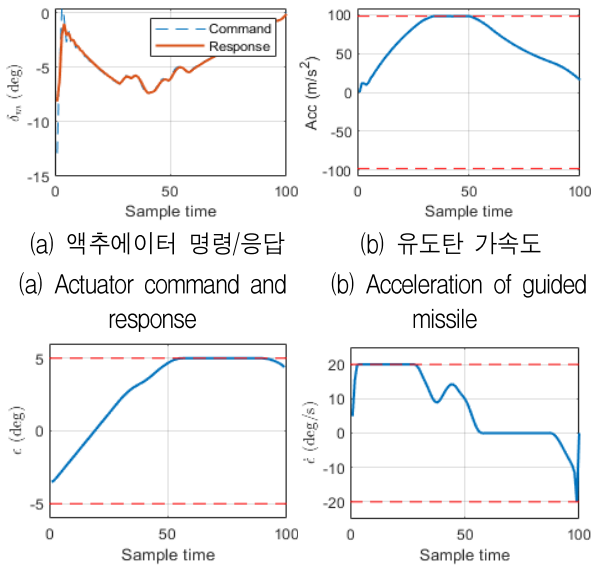


그림 5. 유도탄과 목표물의 상대 변위
Fig. 5. Relative displacement between guided missile and target



(a) 액추에이터 명령/응답 (b) 유도탄 가속도

(a) Actuator command and response (b) Acceleration of guided missile

(c) 탐색기 시선각 (d) 시선각속도

(c) Seeker look angle (d) Look angle rate

그림 6. 통합 유도 및 제어 수치 시뮬레이션 결과

Fig. 6. Integrated guidance and control numerical simulation results

그림 7에서 확인할 수 있듯이 기존 시스템 대비 전체 수행시간이 약 33% 단축된 결과를 보여주었다. 대규모의 희소 행렬 연산이 병렬로 수행됨에 따라 수행시간이 감소하였고, solver의 저수준 연산에 따른 선형 시스템 계산이 간소화되어 수행시간이 감소하였다.

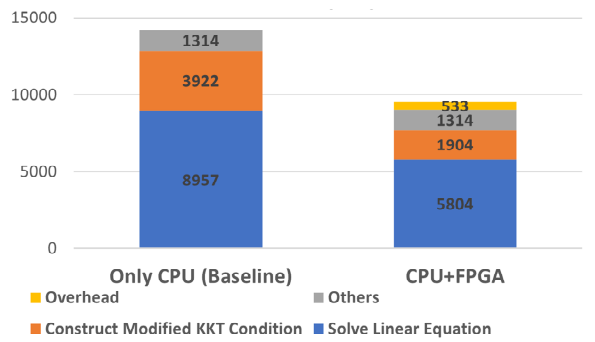


그림 7. 제안된 방법의 수행 시간 비교

Fig. 7. Comparison of execution time of proposed method

행렬 압축 및 FPGA로의 데이터 전송으로 인한 overhead가 발생하였는데, 다른 명령어 세트를 사용하는 프로세서 간 데이터 교환을 위한 지연이므로 이종 프로세서의 동시 설계를 통한 가속화 알고리즘 설계 시 필연적인 것으로 볼 수 있다. 결론적으로, 실험 결과 제안된 알고리즘의 실시간성이 향상되었으며, 제약 조건을 만족하면서 동작하였다.

V. 결 론

본 논문에서는 볼록 최적화의 한 방법인 PD-IPM을 MPC에 적용할 때 발생하는 실시간성 보장 문제를 해결하기 위한 FPGA 기반 병렬 및 가속화 방법을 제안하였다. 특히, 통합 유도 및 조종 알고리즘의 시뮬레이션 결과와 함께 수행시간 측면에서의 분석과 검증을 수행하여 제안된 방법의 효과를 확인하였다.

PD-IPM 알고리즘의 특성상 대규모 행렬 연산이 많아 시스템의 성능을 제약하고 있었는데, FPGA를 활용한 가속화 방법을 통해 이러한 제약을 극복하였다. 가속화된 PD-IPM의 전체 구조를 통해 병렬처리 및 희소 행렬 연산을 적용하여 알고리즘의 수행 속도를 높였으며, 이를 통해 기존 시스템에 비해 수행 속도가 약 33% 향상되었다. 실험 결과에서는 제안된 알고리즘이 목표물 요격을 성공적으로 수행하였으며, 제약 조건을 만족하는 동시에 전체 수행시간이 향상되었다.

종합적으로, 제안된 FPGA 기반의 가속화 방법은 실시간성을 보장하면서도 통합 유도 및 제어 시스템의 성능을 향상시키는 효과적인 방법으로 나타났다. 앞으로 더욱 높은 자원 효율성과 성능 향상을 위한 연구들이 계속되어야 할 것으로 보인다.

References

[1] H. Mingzhe and D. Guangren, "Integrated Guidance and Control of Homing Missiles Against Ground Fixed Targets", Chinese Journal of Aeronautics, Vol. 21, No. 2, pp. 162-168, Apr. 2008. https://doi.org/10.1016/S1000-9361(08)60021-7.

[2] J. H. Evers, J. R. Cloutier, C. F. Lin, W. R. Yueh, and Q. Wang, "Application of Integrated Guidance and Control Schemes to a Precision Guided Missile", in 1992 American Control Conference, Chicago, United States, pp. 3225-3230, Jun. 1992. https://doi.org/10.23919/ACC.1992.4792745.

[3] F. Santoso, M. A. Garratt, and S. G. Anavatti, "State-of-the-Art Integrated Guidance and Control Systems in Unmanned Vehicles: A Review", IEEE Systems Journal, Vol. 15, No. 3, pp. 3312-3323, Sep. 2021. https://doi.org/10.1109/JSYST.2020.3007428.

[4] S. Shamaghdari, S. K. Y. Nikravesh, and M. Haeri, "Integrated guidance and control of elastic flight vehicle based on robust MPC", International Journal of Robust and Nonlinear Control, Vol. 25, No. 15, pp. 2608-2630, Aug. 2015. https://doi.org/10.1002/rnc.3215.

[5] S. J. Qin and T. A. Badgwell, "A survey of industrial model predictive control technology", Control Engineering Practice, Vol. 11, No. 7, pp. 733-764, Dec. 2003. https://doi.org/10.1016/S0967-0661(02)00186-7.

[6] J. B. Rawlings, "Tutorial: model predictive control technology", in Proceedings of the 1999 American Control Conference (Cat. No. 99CH36251), Vol. 1, San Diego, CA, USA, pp. 662-676, Jun. 1999. https://doi.org/10.1109/ACC.1999.782911.

[7] S. Vaddi, "Moving mass actuated missile control using convex optimization techniques", in AIAA Guidance, Navigation, and Control Conference and Exhibit, Keystone, Colorado, pp. 6575, Aug. 2006. https://doi.org/10.2514/6.2006-6575.

[8] S. Ramadurgam and D. G. Perera, "An Efficient FPGA-Based Hardware Accelerator for Convex Optimization-Based SVM Classifier for Machine Learning on Embedded Platforms", Electronics, Vol. 10, No. 11, May 2021. https://doi.org/10.3390/electronics10111323.

[9] S. Lee, H. Lee, Y. Kim, J. Kim, and W. Choi, "GPU-Accelerated PD-IPM for Real-Time Model Predictive Control in Integrated Missile Guidance and Control Systems", Sensors, Vol. 22, No. 12, Jun. 2022. https://doi.org/10.3390/s22124512.

[10] S. P. Boyd and L. Vandenberghe, "Convex optimization", Cambridge university press, pp. 324-615, Mar. 2004.

[11] E. Smith, J. Gondzio, and J. Hall, "GPU Acceleration of the Matrix-Free Interior Point Method", in Parallel Processing and Applied Mathematics, Vol. 7203, pp. 681-689, Sep. 2012. https://doi.org/10.1007/978-3-642-31464-3_69.

[12] B. Wang, et al., "Computing in the air: An open airborne computing platform", IET Communications, Vol. 14, No. 15, pp. 2410-2419, Apr. 2020. https://doi.org/10.1049/iet-com.2019.0515.

[13] M. H. Mofrad, R. Melhem, Y. Ahmad, and M. Hammoud, "Multithreaded Layer-wise Training of Sparse Deep Neural Networks using Compressed Sparse Column", in 2019 IEEE High Performance Extreme Computing Conference (HPEC), Westin Hotel, Waltham, MA USA, pp. 1-6, Sep. 2019. https://doi.org/10.1109/HPEC.2019.8916494.

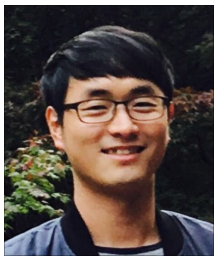
[14] AMBA and ARM, "Axi4-stream protocol specification", Arm Developer logo, pp. 1-56, Apr. 2021. https://developer.arm.com/documentation/ihi0051/b [accessed: Jan. 25, 2024]

저자소개

김 대 연 (Daeyeon Kim)

2023년 2월 : 금오공과대학교 전자공학부(공학사)
2023년 3월 ~ 현재 : 금오공과대학교 전자공학부 IT융복합공학과 석사과정
관심분야 : FPGA, 알고리즘 가속화, 컨벡스 최적화, 임베디드 시스템

이 헌 철 (Heoncheol Lee)



2006년 8월 : 경북대학교 전자전기컴퓨터학부(공학사)
2008년 8월 : 서울대학교 전기컴퓨터공학과(공학석사)
2013년 8월 : 서울대학교 전기컴퓨터공학과(공학박사)
2013년 9월 ~ 2019년 2월 : 국방과학연구소 선임연구원
2019년 3월 ~ 2023년 8월 : 금오공과대학교 전자공학부 IT융복합공학과 조교수
2023년 8월 ~ 현재 : 금오공과대학교 전자공학부 IT융복합공학과 부교수
관심분야 : SLAM, 자율주행, 인공지능, 알고리즘 가속화

최 원 석 (Wonseok Choi)

2006년 2월 : 홍익대학교 전기전자공학부(공학사)
2017년 2월 : 연세대학교 국방공학과(공학석사)
2007년 3월 ~ 현재 : LIG넥스원 수석연구원
관심분야 : FPGA, 필터, 센서 방위산업

정 보 라 (Bora Jeong)

2020년 2월 : 한양대학교 전기전자공학부(공학사)
2020년 3월 ~ 현재 : LIG넥스원 선임연구원
관심분야 : FPGA, 필터, 센서 방위산업

조 영 기 (Youngki Cho)

2000년 2월 : 인하대학교 자동차공학과(공학사)
2002년 2월 : 인하대학교 자동차공학과(공학석사)
2002년 3월 ~ 현재 : LIG넥스원 연구위원
관심분야 : 임베디드 시스템, 방위산업