Ceres-NMPC: Efficient Constrained Nonlinear Least Squares Solver for Nonlinear Model Predictive Control

Jun-Gee Hong
Department of Electrical and Computer Engineering
Inha University
Incheon, Republic of Korea
wnsrl970219@gmail.com

Kwang-Ki K. Kim
Department of Electrical and Computer Engineering
Inha University
Incheon, Republic of Korea
kwangki.kim@inha.ac.kr

***Abstract*—This paper presents Ceres-NMPC, an NMPC solver that leverages the off-the-shelf Ceres Solver, a popular tool in computer vision and robotics for large-scale nonlinear least squares problems. Although Ceres Solver is widely used in applications like simultaneous localization and mapping, its built-in support is limited to box constraints, which complicates its use for optimal control tasks involving complex dynamic and safety constraints. Ceres-NMPC overcomes this limitation by integrating the augmented Lagrangian method, transforming constrained problems into an unconstrained format to enable efficient solutions for nonlinear least squares-based optimal control problems. The solver's performance is demonstrated through diverse scenarios, highlighting its capability in handling complex, nonlinear systems effectively.**

***Index Terms*—Nonlinear model predictive control, Constrained trajectory optimization, Constrained nonlinear least squares, Augmented Lagrangian.**

I. Introduction

Model Predictive Control (MPC) is a control strategy that predicts future states based on the current system state and a dynamic model, solving the optimal control problem in real-time over the prediction horizon within a fixed control rate, so controller can update control input whenever new information is acquired. Unlike traditional classical control strategies that rely solely on current state information, MPC can apply complex control objectives and various constraints. This flexibility has demonstrated its effectiveness in diverse industries, including underactuated highly dynamical systems such as robotics and aerospace launch vehicles [1] [2].

Nonlinear Model Predictive Control (NMPC) typically stats from transforming the continuous-time, infinite-dimensional optimal control problem over the prediction horizon into a discrete, finite-dimensional nonlinear optimization problem (NLP). This optimization problem can be solved using state-of-the-art nonlinear solvers. High performance solvers enabled extensive research in MPC across various fields. There are numerous application of NMPC using SOTA solvers such as IPOPT [3] or SNOPT [4], which handle large-scale non-convex optimization problems [5].

Nevertheless, developing nonlinear optimization solvers specialized for optimal control remains an active research area. This is because leveraging the special structure of optimal control problems allows for numerically stable and fast computations [6]. Furthermore, application-specific solvers need to design algorithms which can satisfies requirements of that application and characteristics of problem's linearity or convexity. Furthermore, these days, developing solvers that could run in embedded system that has limited computational resources is an open problem which has significant research attention [7], [8].

The solver for optimization problems in NMPC discussed in this paper considers real-time performance which is essential in Robotics control application which is common to use sensor-based reactive control that must quickly respond to unpredictable environmental changes or disturbances.

In this study, we propose the Ceres-NMPC algorithm, a solver for nonlinear predictive control that employs the Augmented Lagrangian Method which can handle constrained nonlinear problem, and the Ceres Solver which was originally developed for a general-purpose nonlinear least squares solver. Ceres-NMPC leverages the strengths of the Ceres Solver [9], such as numerical stable algorithms for large-scale problems, while addressing its limitation of handling equality constraints by integrating the Augmented Lagrangian Method. This combination enables one to easily formulate optimal control problems. We demonstrate the validity and scalability of the Ceres-NMPC solver by solving various scenarios of constrained nonlinear optimal control problems.

The structure of this paper is as follows. In Chapter 1, we introduce existing method for formulating nonlinear optimal control, and briefly introduce Ceres Solver. Chapter 2 describes a theoretical background of Ceres-NMPC. It contains all ingredient for Ceres-NMPC: method to handle constrained nonlinear least squares problem, and formulating optimal control problem that our algorithm can address. In Chapter 3, we explain the overall methodology and algorithm of Ceres-

NMPC. In Chapter 4, we validates the proposed algorithm through various scenarios. Finally, Chapter 5 presents the conclusions and future research directions.

A. Related Work

As mentioned in the introduction, optimal control in optimization approach starts by transforming the continuous optimal control problem into a finite-dimensional nonlinear optimization problem. Common transcription methods include the Single-Shooting Method, the Multiple-Shooting Method, and the Collocation Method [6].

When an optimal control problem is converted into a finite-dimensional nonlinear optimization problem, it is often referred to as a dynamic optimization problem. There are generally two main approaches to solving this dynamic optimization problems. The first approach involves approximating the nonlinear dynamic optimization problem with a linear-quadratic regulator (LQR) problem and solving it iteratively. Methods such as Differential Dynamic Programming (DDP) and Iterative-LQR (iLQR) fall into this category [10]. While these methods have the advantage of working well in practice, they inherit the fundamental inconvenience of handling constraints from LQR. Consequently, various algorithms are emerging today as a research topic to address this limitation [11].

The second approach, known as a direct method, treats nonlinear dynamic optimization problems in the same manner as general optimization problems. Since Ceres-NMPC belongs to this category, we next examine how nonlinear optimization problems are solved using direct methods.

There are three big families in direct methods. they are Sequential Quadratic Programming (SQP), Interior Point Methods, and the Augmented Lagrangian Method.

First, Sequential Quadratic Programming approximates the original problem as a quadratic programming (QP) problem and solves it iteratively. A well-known commercial solver using this method is SNOPT [4]. Many nonlinear Model Predictive Control (MPC) problems have shown successful results using SQP-based solvers, mainly because techniques like Real-Time Iteration (RTI) and warm starting align well with the real-time requirements of MPC. Additionally, by employing advanced embedded quadratic problem solvers [12], these methods are widely utilized in the field of embedded nonlinear MPC.

Second, there is the Interior Point Method, which is a type of Newton method applied to the modified nonlinear Karush–Kuhn–Tucker (KKT) system. It has the advantages of speed and effectively handling hard constraints. Furthermore, it is easy to implement using a general-purpose optimizer like IPOPT [3]. However, it has been found to be less effective for warm starting, which is often required for MPC [13].

Lastly, the Augmented Lagrangian Method, used by the Ceres-NMPC algorithm, repeatedly optimizes the augmented Lagrangian of the original problem. The Augmented Lagrangian Method replaces various constraints in the original problem with slack variables, allowing the inner optimization loops to be solved using solvers that handle simpler constraints (e.g., box constraints). This opens the door to a wide range of possible solvers. Moreover, since the inner optimization repeatedly solves a similar problem, it is also effective for warm starting. As a result, the use of the Augmented Lagrangian Method in nonlinear optimal control is becoming increasingly common [14], [15].

Next, we introduce the Ceres Solver. Ceres solver, developed by Google, is an open-source nonlinear least squares solver designed to efficiently handle large-scale problems and incorporate Lie group representations of rotational dynamics. With built-in support for sparse linear algebra operations and automatic differentiation, Ceres provides flexibility and convenience in formulating problems. Currently, Ceres is widely used in the robotics field as a back-end optimization framework for Pose Graph Optimization in Simultaneous Localization and Mapping (SLAM).

Incremental pose estimation problem, which is the subproblem of SLAM, involves estimating current pose(state) using history of sensor measurements and kinematic model(dynamics). It was often formulated as a Moving Horizon Estimation or Sliding Window Smoothing problem, this formulation aligns closely with that of MPC, making them a dual problems to each other [6]. Although there have been successful applications of the Ceres Solver to pose estimation problems in SLAM [16], to the best of the authors' knowledge, there have been no reported cases of using it to solve control problems to date.

II. Theoretical Background

A. Nonlinear Least Square

In this chapter, we explain the Levenberg-Marquardt algorithm [17], which can be interpreted as a type of trust-region method for solving nonlinear least squares problems. This algorithm is one of the methods employed by the Ceres Solver to address nonlinear least squares optimization.

Consider the following equation which represents a nonlinear least squares problem without constraints:

$$\min_{\boldsymbol{x}} r_1(\boldsymbol{x})^2 + \cdots + r_m(\boldsymbol{x})^2 = \|\boldsymbol{r}(\boldsymbol{x})\|_2^2 \tag{1}$$

Here, $\boldsymbol{x} \in \mathbb{R}^n$ is a vector of parameters, and each $r_i : \mathbb{R}^n \to \mathbb{R}$ is a residual function. In Ceres terminology, each r_i is called a "Residual", the parameter vector $\boldsymbol{x}$ is called "Parameter Blocks" and the collection of all r_i is referred to as a "Residual Block". The vector function $\boldsymbol{r} : \mathbb{R}^n \to \mathbb{R}^m$ combines all these residuals.

A standard approach to solving nonlinear least squares problems is the Gauss-Newton (GN) method, which iteratively solves a linearized version of the original problem. At the k-th iteration, it considers the following problem:

$$\Delta \boldsymbol{x}^k = \arg\min_{\Delta \boldsymbol{x}} \|\boldsymbol{r}(\boldsymbol{x}^k) + D\boldsymbol{r}(\boldsymbol{x}^k)(\Delta \boldsymbol{x})\|_2^2 \tag{2}$$

Here, $D\boldsymbol{r}$ is the Jacobian matrix of the residual vector $\boldsymbol{r}$ evaluated at $\boldsymbol{x}^k$. This approximates the Hessian of the original optimization problem in Equation (1) as $D\boldsymbol{r}(\boldsymbol{x})^\top D\boldsymbol{r}(\boldsymbol{x})$.

Trust-region methods improve the numerical stability and convergence of the Gauss-Newton method. They do this by repeatedly solving the following trust-region subproblem:

$$\begin{aligned} \Delta \boldsymbol{x}^k = \arg\min_{\Delta \boldsymbol{x}} &\|\boldsymbol{r}(\boldsymbol{x}^k) + D\boldsymbol{r}(\boldsymbol{x}^k)(\Delta \boldsymbol{x})\|_2^2 \\ \text{s.t. } &\|\Delta \boldsymbol{x}\| \leq \alpha^k \end{aligned} \tag{3}$$

This ensures that the step $\Delta \boldsymbol{x}^k$ remains within the trust region of radius α^k.

Furthermore, the size of the trust region is adjusted based on the step quality ratio:

$$\begin{aligned} &\rho = \frac{\|\boldsymbol{r}(\boldsymbol{x}^k + \Delta \boldsymbol{x}^k)\|_2^2 - \|\boldsymbol{r}(\boldsymbol{x}^k)\|_2^2}{\|\boldsymbol{r}(\boldsymbol{x}^k) + D\boldsymbol{r}(\boldsymbol{x}^k)\Delta \boldsymbol{x}^k\|_2^2 - \|\boldsymbol{r}(\boldsymbol{x}^k\|_2^2} \\ &\text{if} \quad \rho > \epsilon \quad \text{then} \quad \boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \Delta \boldsymbol{x}^k \\ &\text{if} \quad \rho > \gamma_1 \quad \text{then} \quad \alpha^{k+1} = 2\alpha^k \\ &\text{if} \quad \rho > \gamma_2 \quad \text{then} \quad \alpha^{k+1} = 0.5\alpha^k \end{aligned} \tag{4}$$

Here, ρ represents how well the step $\Delta \boldsymbol{x}^k$ optimizes the original residual function compared to the linearized residual function. If ρ is small, it indicates that the linear approximation within the current trust region is valid, so the trust region can be expanded and the step accepted. Conversely, if ρ is large, the trust region is reduced until the evaluation ratio meets the desired criteria.

If there exists a Lagrange multiplier $\boldsymbol{\lambda}^k \geq \boldsymbol{0}$ such that $|\Delta \boldsymbol{x}^k| \leq \alpha^k$ and the following conditions hold, then the solution $\Delta \boldsymbol{x}^k$ to Equation (3) is a global minimizer:

$$\begin{aligned} &\big(D\boldsymbol{r}(\boldsymbol{x}^k)^\top D\boldsymbol{r}(\boldsymbol{x}^k) + \boldsymbol{\lambda}^k \mathbf{I}\big)\Delta \boldsymbol{x}^k = -D\boldsymbol{r}(\boldsymbol{x}^k)^\top \boldsymbol{r}(\boldsymbol{x}^k) \\ &\boldsymbol{\lambda}^k(\alpha^k - \|\Delta \boldsymbol{x}^k\|) = 0 \\ &D\boldsymbol{r}(\boldsymbol{x}^k)^\top D\boldsymbol{r}(\boldsymbol{x}^k) + \boldsymbol{\lambda}^k \mathbf{I} \succeq 0 \end{aligned} \tag{5}$$

The last condition in Equation (5) is automatically satisfied since adding a positive diagonal matrix to the positive semidefinite Gauss-Newton Hessian preserves positive semidefiniteness. Practically, since the optimization in Equation (3) is itself an approximation, exact solutions to Equation (5) are often replaced by effective approximation methods [18].

Moreover, Ceres Solver can handle box-constrained problems using a projection operator ($\mathcal{P}$) and line search. The search direction and step update are given by:

$$\begin{aligned} &\boldsymbol{s}^k = \mathcal{P}_{\mathbf{B}}(\boldsymbol{x}^k + \Delta \boldsymbol{x}^k) - \boldsymbol{x}^k \\ &\mathcal{P}_{\mathbf{B}} \quad \text{is projection operator on Box set} \quad \mathbf{B} \\ &\mathbf{B} = \{\boldsymbol{x} | \quad \underline{\boldsymbol{x}} \leq \boldsymbol{x} \leq \overline{\boldsymbol{x}}\} \\ &\boldsymbol{x}^{k+1} \leftarrow \boldsymbol{x}^k + \alpha \boldsymbol{s}^k \end{aligned} \tag{6}$$

Here, the step size $\alpha \in [0, 1]$ is determined through a line search. Since the box constraint $\mathbf{B}$ is a convex set, it ensures that $\boldsymbol{x}^{k+1} \in \mathbf{B}$ is satisfied.

B. Constrained Nonlinear Least Square : Augmented Lagrangian

In this chapter, we describe how to solve constrained nonlinear least squares problems using the Augmented Lagrangian Method [19]. To begin, in addition to Equation (1), we consider the following problem with constraints:

$$\begin{aligned} \min_{\boldsymbol{x}} \ &\|\boldsymbol{r}(\boldsymbol{x})\|^2 \\ \text{s.t. } &\boldsymbol{c}(\boldsymbol{x}) = 0 \\ &\boldsymbol{x} \in \mathbb{C} \end{aligned} \tag{7}$$

Here, $\boldsymbol{c}$ represents a set of p equality constraints, defined as $\boldsymbol{c} : \mathbb{R}^n \to \mathbb{R}^p$. The set $\mathbb{C}$ is a convex set, and the constraints that the Ceres Solver can handle are of the form $\mathbb{C} = \{\boldsymbol{x} \mid \underline{\boldsymbol{x}} \leq \boldsymbol{x} \leq \overline{\boldsymbol{x}}\}$ which is box constraint.

The Lagrangian for defining the first-order necessary conditions for the above problem is as follows:

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = \|\boldsymbol{r}(\boldsymbol{x})\|^2 + \boldsymbol{\lambda}^\top \boldsymbol{c}(\boldsymbol{x})$$

The corresponding first-order optimality conditions are as follows:

$$\begin{aligned} &\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = \|\boldsymbol{r}(\boldsymbol{x})\|^2 + \boldsymbol{\lambda}^\top \boldsymbol{c}(\boldsymbol{x}) \\ &\mathcal{P}_{\mathbb{C}}[\boldsymbol{x}^* - \nabla_{\boldsymbol{x}} \mathcal{L}(\boldsymbol{x}^*, \boldsymbol{\lambda}^*)] - \boldsymbol{x}^* = 0 \\ &\nabla_{\boldsymbol{\lambda}} \mathcal{L} = \boldsymbol{c}(\boldsymbol{x}^*) = 0 \\ &\nabla_{\boldsymbol{x}} \mathcal{L} = 2D\boldsymbol{r}(\boldsymbol{x})^\top \boldsymbol{r}(\boldsymbol{x}) + D\boldsymbol{c}(\boldsymbol{x})^\top \boldsymbol{\lambda} = 0 \end{aligned} \tag{8}$$

Here, $\boldsymbol{x}^*$ and $\boldsymbol{\lambda}^*$ are the values of the primal and dual variables that satisfy the first-order optimality conditions. The Augmented Lagrangian corresponding to the above problem is as follows:

$$\mathcal{L}_A(\boldsymbol{x}, \boldsymbol{\lambda}, \mu) = \|\boldsymbol{r}(\boldsymbol{x})\|^2 + \boldsymbol{c}(\boldsymbol{x})^\top \boldsymbol{\lambda} + \mu \|\boldsymbol{c}(\boldsymbol{x})\|^2$$

Here, μ is the parameter of the quadratic penalty function $\|\boldsymbol{c}(\boldsymbol{x})\|^2$, which determines the weight of the cost function that increases as the equality constraints are violated. The Augmented Lagrangian Method iteratively optimizes the defined Augmented Lagrangian $\mathcal{L}_A$ to find $\boldsymbol{x}$ and $\boldsymbol{\lambda}$ that satisfy the first-order optimality condition in Equation (8).

The Augmented Lagrangian Method consists of two main loops: the outer loop and the inner loop. The outer loop updates the dual variables $\boldsymbol{\lambda}$ and the penalty parameter μ, while the inner loop performs the optimization of the Augmented Lagrangian to optimize the primal variables $\boldsymbol{x}$ and provide the initial guess for the next optimization problem.

The sequence $(\boldsymbol{x}^t, \boldsymbol{\lambda}^t, \mu^t)$ represents the t-th iteration of the Augmented Lagrangian sequence $(\boldsymbol{x}, \boldsymbol{\lambda}, \mu)$. The optimization in the $(t+1)$-th outer loop iteration uses the results from the t-th iteration, $(\boldsymbol{x}, \boldsymbol{\lambda}, \mu)$ as the initial guess to solve the following Augmented Lagrangian optimization problem:

$$\begin{aligned}
\boldsymbol{x}^{t+1} &= \arg\min_{\boldsymbol{x}\in\mathbb{C}} \mathcal{L}_A(\boldsymbol{x}; \boldsymbol{\lambda}^t, \mu^t) \\
&= \arg\min_{\boldsymbol{x}\in\mathbb{C}} \|\boldsymbol{r}(\boldsymbol{x})\|^2 + \boldsymbol{c}(\boldsymbol{x})^\top \boldsymbol{\lambda}^t + \mu^t \|\boldsymbol{c}(\boldsymbol{x})\|^2 \\
&= \arg\min_{\boldsymbol{x}\in\mathbb{C}} \|\boldsymbol{r}(\boldsymbol{x})\|^2 + \mu^t \|\boldsymbol{c}(\boldsymbol{x}) + \frac{1}{2\mu^t}\boldsymbol{\lambda}^t\|^2 - \frac{1}{2\mu}\|\boldsymbol{\lambda}^t\|^2 \\
&= \arg\min_{\boldsymbol{x}\in\mathbb{C}} \left\| \begin{bmatrix} \boldsymbol{r}(\boldsymbol{x}) \\ \sqrt{\mu^t}\boldsymbol{c}(\boldsymbol{x}) + \frac{1}{2\sqrt{\mu^t}}\boldsymbol{\lambda}^t \end{bmatrix} \right\|^2
\end{aligned} \tag{9}$$

$\boldsymbol{x}^{t+1}$ and $\boldsymbol{\lambda}^t$ must satisfy the first-order optimality conditions of the Augmented Lagrangian, as given below.

$$\begin{aligned}
&\mathcal{P}_{\mathbb{C}}[\boldsymbol{x}^{t+1} - \nabla_{\boldsymbol{x}} \mathcal{L}_A(\boldsymbol{x}^{t+1}, \boldsymbol{\lambda}^t, \mu^t)] - \boldsymbol{x}^{t+1} = 0 \\
&\nabla_{\boldsymbol{x}} \mathcal{L}_A = \\
&2D\boldsymbol{r}(\boldsymbol{x}^{t+1})^\top \boldsymbol{r}(\boldsymbol{x}^{t+1}) + D\boldsymbol{c}(\boldsymbol{x}^{t+1})^\top (\boldsymbol{\lambda}^t + 2\mu^t \boldsymbol{c}(\boldsymbol{x}^{t+1}))
\end{aligned} \tag{10}$$

By comparing the stationary condition, which is the first-order optimality condition of Equation (10) and Equation (8), we observe the following:

$$\boldsymbol{\lambda}^{t+1} = \boldsymbol{\lambda}^t + 2\mu^t \boldsymbol{c}(\boldsymbol{x}^{t+1}) \tag{11}$$

The update in Equation (11) aligns with the direction of $\boldsymbol{\lambda}$ that satisfies the first-order optimality condition (8) of the original optimization problem.

The penalty parameter μ^t is updated based on the following conditions:

$$\begin{aligned}
\mu^{t+1} &= \mu^t \qquad \text{if} \|\boldsymbol{c}(\boldsymbol{x}^{t+1})\|_\infty \leq \beta \|\boldsymbol{c}(\boldsymbol{x}^t)\|_\infty \\
\mu^{t+1} &= \alpha \mu^t \qquad \text{otherwise}
\end{aligned} \tag{12}$$

This principle implies that if the violation of the equality constraint $\boldsymbol{c}$ does not decrease by at least a factor of β compared to the previous iteration, the associated penalty parameter μ is increased by α. Here, α and β are hyperparameters, and generally, a larger α and a smaller β result in slower convergence.

From Equation (11), it can be observed that if the sequence $\boldsymbol{\lambda}^t$ converges, it implies $\boldsymbol{c}(\boldsymbol{x}) = \mathbf{0}$, which ultimately corresponds to finding the optimal variable $\boldsymbol{x}$ that satisfies the equality constraints. Therefore, the iterative method terminates when $\|\boldsymbol{c}(\boldsymbol{x}^t)\|_\infty < \epsilon$.

C. Optimal Control Problem for Nonlinear Model Predictive Control

The following is the problem formulation for Model Predictive Control (MPC) considered in this paper, which is a dynamic optimization problem discretized into N steps over the prediction horizon.

$$\begin{aligned}
V(\hat{\mathbf{x}}_0) = \min_{\mathbf{x}_{1:N}, \mathbf{u}_{1:N-1}} & \sum_{k=0}^{N-1} l_k(\mathbf{x}_k, \mathbf{u}_k) + l_N(\mathbf{x}_N) && \text{(13a)} \\
\text{s.t.} \quad & \mathbf{x}_0 - \hat{\mathbf{x}}_0 = \mathbf{0} && \text{(13b)} \\
& \mathbf{x}_{k+1} - \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) = \mathbf{0}, && \text{(13c)} \\
& \mathbf{h}_N(\mathbf{x}_N) = \mathbf{0} && \text{(13d)} \\
& \underline{\mathbf{x}} \leq \mathbf{x}_k \ \ \leq \bar{\mathbf{x}}, && \text{(13e)} \\
& \underline{\mathbf{u}} \leq \mathbf{u}_k \ \ \leq \bar{\mathbf{u}} && \text{(13f)} \\
& \underline{\mathbf{x}} \leq \mathbf{x}_N \ \ \leq \bar{\mathbf{x}}, && \text{(13g)} \\
& k = 0, 1, 2, \cdots N-1
\end{aligned}$$

In the optimization problem, the decision variables $\mathbf{x}_k \in \mathbb{R}^n$ and $\mathbf{u}_k \in \mathbb{R}^m$ represent the state variables and control variables, respectively.

The term $l_k(\mathbf{x}_k, \mathbf{u}_k)$ denotes the stage cost function at each time step, while $l_N(\mathbf{x}_N)$ is the terminal cost function. Equation (13b) represents an equality constraint that fixes the initial condition of the state variable $\hat{\mathbf{x}}_0$. Equation (13c) is the discretized dynamic equation. Equations (13d) and (13e) represent constraints at the terminal time, and the last two equations define stage-wise inequality constraints on the state and control variables. It is assumed that all functions are at least twice differentiable.

The primary objective of many Model Predictive Control (MPC) problems is to stabilize the system by reducing errors between the system state and a stable equilibrium or a reference trajectory, while satisfying all constraints. In this context, the above problem formulation can accommodate actuator output constraints and the system's maximum and minimum state constraints, thus addressing many practical control problems. Furthermore, the inclusion of terminal constraints (13d) and (13e) ensures the stability of MPC [20].

Nonlinear Model Predictive Control (NMPC) solves a new optimization problem (13) at every time step t_0 with a different initial condition $\hat{\mathbf{x}}_0$ or its estimate (full state) within the sampling time. The first control input $\mathbf{u}_0$ from the solution is applied to the system during the sampling period, providing feedback control.

When performing optimization at t_0, the process of exploring around the trajectory of the optimal state and control variables obtained in the previous time step is known as warm-starting. Approximate solutions are often computed to guarantee real-time feasibility [8].

III. Ceres-NMPC Algorithm

In this chapter, we examine the structure of the Ceres-NMPC algorithm. First, we redefine the optimal control problem discussed earlier into a constrained nonlinear least squares form that can be solved by Ceres-NMPC. Then, we describe the process of solving it using the Augmented Lagrangian Method. Additionally, we explore the characteristics of the problems addressed by the Ceres Solver within the Augmented Lagrangian Method.

In this study, the problem (13) is formulated using the Multiple-Shooting method for discretization.

A. Reformulation of the Nonlinear Model Predictive Control Problem into a Least Squares Form

To utilize Ceres, a nonlinear least squares solver, the objective function of the optimization problem (13a) must be reformulated into a sum-of-squares form of nonlinear functions. This is achieved using a kernel function ρ.

$$\min_{\mathbf{x}_{1:N},\mathbf{u}_{1:N-1}} \sum_{k=0}^{N-1} \rho\left(\|l_k(\mathbf{x}_k,\mathbf{u}_k)\|^2\right) + \rho\left(\|l_N(\mathbf{x}_N)\|^2\right) \tag{14}$$

Assuming that the functions l_k and l_N are positive, the original optimization problem can be reformulated using the kernel function $\rho(x) = \sqrt{x}$.

Ceres supports a variety of kernel functions, which it internally refers to as loss functions.

The following is an example of an optimal control problem with a quadratic cost, which is frequently used in control problems. In this case, the cost function is a linear combination of the squared state variables and the squared control variables.

$$\begin{aligned}\min_{\mathbf{x}_{1:N},\mathbf{u}_{1:N-1}} & \sum_{k=0}^{N-1} \left(\mathbf{x}_k^\top \mathbf{Q}_k \mathbf{x}_k + \mathbf{u}_k^\top \mathbf{R}_k \mathbf{u}_k\right) + \mathbf{x}_N^\top \mathbf{Q}_N \mathbf{x}_N \\ &= \sum_{k=0}^{N} \left\| \begin{bmatrix} \mathbf{S}_k & \mathbf{0} \\ \mathbf{0} & \mathbf{W}_k \end{bmatrix} \begin{bmatrix} \mathbf{x}_k \\ \mathbf{u}_k \end{bmatrix} \right\|^2 \end{aligned} \tag{15}$$

Here, the weight matrices $\mathbf{Q}_k \succeq 0$ and $\mathbf{R}_k \succ 0$, and $\mathbf{S}_k$, $\mathbf{W}_k$ can be expressed as $\mathbf{Q}_k^{1/2}$ and $\mathbf{R}_k^{1/2}$, respectively. These can be computed using Singular Value Decomposition (SVD) or Cholesky Decomposition.

Thus, the original nonlinear dynamic optimization problem becomes a constrained nonlinear least squares problem with a squared cost function and the constraints given by 13b-13g. The solution method for this problem will be explained based on Equation (15) for simplicity.

B. Augmented Lagrangian for Optimal control problem

Next, to solve the constrained nonlinear least squares problem with the objective function given by Equation (15), the Augmented Lagrangian Method described earlier is employed.

At the t-th iteration of the Augmented Lagrangian Method, the optimization involves solving the following nonlinear least squares problem:

$$\begin{aligned}\min_{\mathbf{x}_{1:N},\mathbf{u}_{1:N-1}} \quad & \sum_{k=0}^{N} \left\| \begin{bmatrix} \mathbf{S}_k & \mathbf{0} \\ \mathbf{0} & \mathbf{W}_k \end{bmatrix} \begin{bmatrix} \mathbf{x}_k \\ \mathbf{u}_k \end{bmatrix} \right\|^2 \\ & + \left\| \sqrt{\mu}\boldsymbol{g}_0(\mathbf{x}_0) + \frac{1}{2\sqrt{\mu}}\boldsymbol{\lambda}_0 \right\|^2 \\ & + \sum_{k=0}^{N-1} \left\| \sqrt{\mu}\boldsymbol{g}_{k+1}(\mathbf{x}_k,\mathbf{u}_k,\mathbf{x}_{k+1}) + \frac{1}{2\sqrt{\mu}}\boldsymbol{\lambda}_{k+1} \right\|^2 \\ & + \left\| \sqrt{\mu}\boldsymbol{g}_{N+1}(\mathbf{x}_N) + \frac{1}{2\sqrt{\mu}}\boldsymbol{\lambda}_{N+1} \right\|^2 \\ \text{s.t.} \quad & \underline{\mathbf{x}} \leq \mathbf{x}_k \ \leq \overline{\mathbf{x}}, \\ & \underline{\mathbf{u}} \leq \mathbf{u}_k \ \leq \overline{\mathbf{u}} \end{aligned} \tag{16}$$

Here, $\boldsymbol{g}$ represents the equality constraints from Equation (13). For simplicity, the outer loop index t of the Augmented Lagrangian Method is omitted.

By reformulating the problem using a new decision variable vector $\mathbf{z} = [\mathbf{x}_0, \mathbf{u}_0, \mathbf{x}_1, \mathbf{u}_1, \cdots, \mathbf{x}_{N-1}, \mathbf{u}_{N-1}, \mathbf{x}_N]$, which stacks the state and control variables alternately across all time steps, the problem is expressed as follows:

$$\begin{aligned}\min_{\mathbf{z}} & \left\| \begin{bmatrix} \mathbf{H}\mathbf{z} \\ \boldsymbol{\Phi}(\mathbf{z};\boldsymbol{\lambda},\mu) \end{bmatrix} \right\|^2 \\ \text{s.t.} & \ \underline{\mathbf{z}} \leq \mathbf{z} \leq \overline{\mathbf{z}} \end{aligned} \tag{17}$$

$$\mathbf{H} = \begin{bmatrix} \mathbf{S}_0 & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{W}_0 & \cdots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \ddots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{W}_{N-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{S}_N \end{bmatrix}, \underline{\mathbf{z}} = \begin{bmatrix} \underline{\mathbf{x}} \\ \underline{\mathbf{u}} \\ \vdots \\ \underline{\mathbf{u}} \\ \underline{\mathbf{x}} \end{bmatrix}, \underline{\mathbf{z}} = \begin{bmatrix} \underline{\mathbf{x}} \\ \underline{\mathbf{u}} \\ \vdots \\ \underline{\mathbf{u}} \\ \underline{\mathbf{x}} \end{bmatrix},$$

$$\boldsymbol{\Phi}(\mathbf{z};\boldsymbol{\lambda},\mu) = \begin{bmatrix} \sqrt{\mu}\boldsymbol{g}_0(\mathbf{x}_0) + \frac{1}{2\sqrt{\mu}}\boldsymbol{\lambda}_0 \\ \sqrt{\mu}\boldsymbol{g}_1(\mathbf{x}_0,\mathbf{u}_0,\mathbf{x}_1) + \frac{1}{2\sqrt{\mu}}\boldsymbol{\lambda}_1 \\ \vdots \\ \sqrt{\mu}\boldsymbol{g}_{k+1}(\mathbf{x}_k,\mathbf{u}_k,\mathbf{x}_{k+1}) + \frac{1}{2\sqrt{\mu}}\boldsymbol{\lambda}_{k+1} \\ \vdots \\ \sqrt{\mu}\boldsymbol{g}_{N+1}(\mathbf{x}_N) + \frac{1}{2\sqrt{\mu}}\boldsymbol{\lambda}_{N+1} \end{bmatrix}$$

$\mathbf{z} \in \mathbb{R}^{N(n+1)+Nm}, \boldsymbol{\lambda} \in \mathbb{R}^{n(N+2)}, \mu \in \mathbb{R}$

Here, the function $\Phi(\mathbf{z};\boldsymbol{\lambda},\mu) : \mathbb{R}^{N(n+m)+N} \to \mathbb{R}^{N+1}$ is a function of $\mathbf{z}$ parameterized by $\boldsymbol{\lambda}$ and μ.

The Ceres solver can solve the problem (17). By updating the dual variable λ and the penalty parameter μ as described in II-B, the constrained nonlinear least squares problem, i.e., the given optimal control problem, can be solved. The overview of the Ceres-NMPC algorithm is shown in Figure 1.

If the cost function and kernel function are as described in Equation (14), the problem (17) can be modified and solved

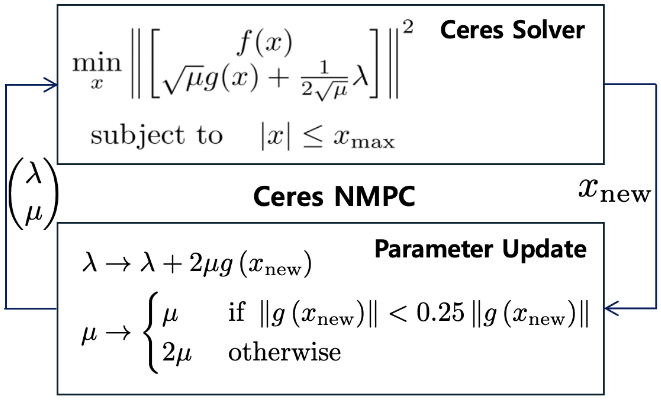

Ceres Solver
$\min_{x} \left\| \begin{bmatrix} f(x) \\ \sqrt{\mu} g(x) + \frac{1}{2\sqrt{\mu}}\lambda \end{bmatrix} \right\|^2$
subject to $|x| \leq x_{\max}$
$\begin{pmatrix} \lambda \\ \mu \end{pmatrix}$
Ceres NMPC
x_{new}
Parameter Update
$\lambda \rightarrow \lambda + 2\mu g\left(x_{\text{new}}\right)$
$\mu \rightarrow \begin{cases} \mu & \text{if } \left\|g\left(x_{\text{new}}\right)\right\| < 0.25 \left\|g\left(x_{\text{new}}\right)\right\| \\ 2\mu & \text{otherwise} \end{cases}$

Fig. 1: Algorithm structure

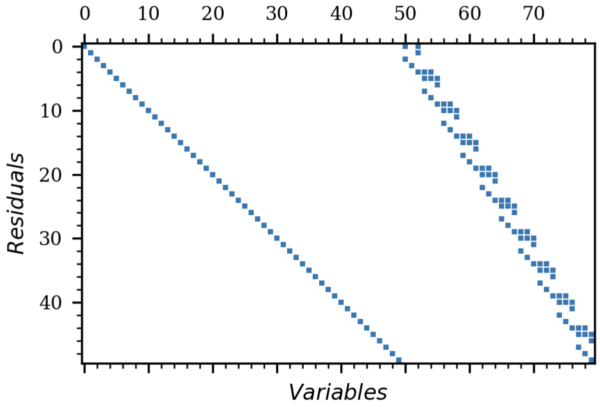


Fig. 2: Sparse Jacobian block of overall residual

as follows:

$$\min_{z} \quad \rho\left(\|\mathbf{l}(\mathbf{x}_k, \mathbf{u}_k)\|^2\right) + \rho\left(\|l_N(\mathbf{x}_N)\|^2\right) + \|\mathbf{\Phi}(\mathbf{z}; \boldsymbol{\lambda}, \mu)\|^2$$

$$\mathbf{l}(\mathbf{x}_k, \mathbf{u}_k) = \begin{bmatrix} l_0(\mathbf{x}_0, \mathbf{u}_0) \\ l_1(\mathbf{x}_1, \mathbf{u}_1) \\ \vdots \\ l_N(\mathbf{x}_N) \end{bmatrix} \tag{18}$$

C. Jacobian for Augmented Lagrangian

The Ceres solver internally requires the residuals and the Jacobian matrix of Equation (17).

Figure (2) illustrates the transpose of the Jacobian matrix ($\mathbf{J}^\top$) for the optimal control problem being solved. Ceres approximates the Hessian as ($\mathbf{J}^\top \mathbf{J}$), and the operation ($\mathbf{J}^\top \mathbf{J}$) on the Jacobian, shown in Figure (2), reveals a sparse band-structured block diagonal matrix.

The Ceres solver supports both direct factorization methods, such as SPARSE NORMAL CHOLESKY (using Cholesky decomposition), and iterative methods, such as Conjugate Gradient, to effectively solve linear systems involving sparse matrices.

IV. Simulation Results

In this chapter, the performance and convergence of the Ceres-NMPC algorithm are evaluated by solving various optimal control problems.

For all problems, the control trajectory is implemented using zero-order hold, and integration is performed using the Forward Euler method. The initial guesses for both state variables and control variables are set to zero.

A. Cartpole Swing Up Control

The dynamics of the cart-pole system, assumed to be a point mass, are as follows:

$$\begin{aligned} \ddot{\theta} &= \frac{1}{l\left(M + m\sin^2\theta\right)}\Bigg(- F\cos\theta - ml\dot{\theta}^2\cos\theta\sin\theta \\ &\qquad - (M+m)g\sin\theta\Bigg) \\ \ddot{p} &= \frac{1}{M + m\sin^2\theta}\left(F + m\sin\theta\left(l\dot{\theta}^2 + g\cos\theta\right)\right) \end{aligned} \tag{19}$$

In the state-space equations above, p represents the position of the cart, and θ is the angle of the pole with respect to the positive x-axis (positive p-direction). F denotes the horizontal force applied to the cart. M and m are the masses of the cart and the pole, respectively, and l is the length of the pole. The gravitational acceleration is $g = -9.81$.

The state variables are defined as $\boldsymbol{x} = [\theta, \dot{\theta}, p, \dot{p}]^\top$, and the control variable is $u = F$. The pole starts in the downward position with an initial condition $\boldsymbol{x}_0 = [0, 0, 0, 0]^\top$, and the objective is to bring it to the upright position $\boldsymbol{x}_N = [\pi, 0, p_r, 0]^\top$. To achieve this, the cost function for the optimal control problem is formulated as follows:

$$\min_{\boldsymbol{u}, \boldsymbol{x}} \sum_{k=0}^{N} 12\left(\cos(\theta)_k + 1\right)^2 + 0.1\dot{\theta}_k^2 + 6(p_k - p_r)^2 + 0.1\dot{p}_k^2$$

Here, p_r is the desired position. The parameters for the above problem are summarized in Table (I).

Figure (3) shows the open-loop trajectory for the first MPC iteration, demonstrating that a trajectory satisfying the given constraints and achieving the control objective has been generated.

B. Point to Point Control and Reference Trajectory Tracking Control of Mobile robot

The kinematic model of the mobile robot is simplified using the unicycle kinematics, and its dynamics are as follows:

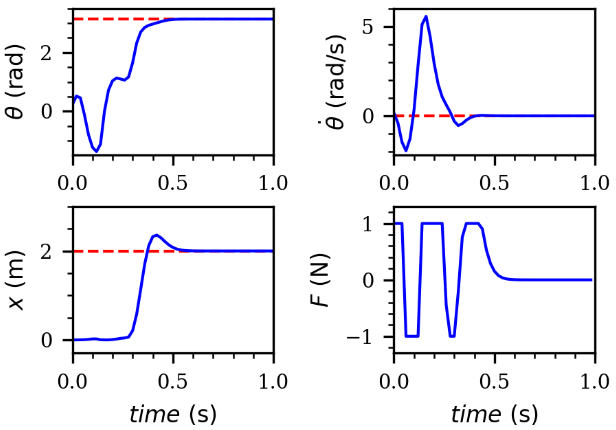


Fig. 3: Simulation results for scenario IV-A

Parameter	Value	Parameter	Value
M (kg)	1	m (kg)	0.3
l (m)	0.5	Δt (sec)	0.5
F_{max} (N)	3	F_{min} (N)	3
p_{max} (m)	3	p_{min} (m)	0
p_r (m)	2	N	50

TABLE I: Paramters for scenario IV-A

$$\begin{aligned}\dot{p_x} &= v\cos\theta \\ \dot{p_y} &= v\sin\theta \\ \dot{\theta} &= w\end{aligned} \tag{20}$$

Here, the state variables are $\boldsymbol{x} = [p_x, p_y, \theta]^\top$, representing the position and orientation of the mobile robot in the plane. The control inputs are $\boldsymbol{u} = [v, w]^\top$, where v is the velocity and w is the angular velocity in the robot's local coordinate frame.

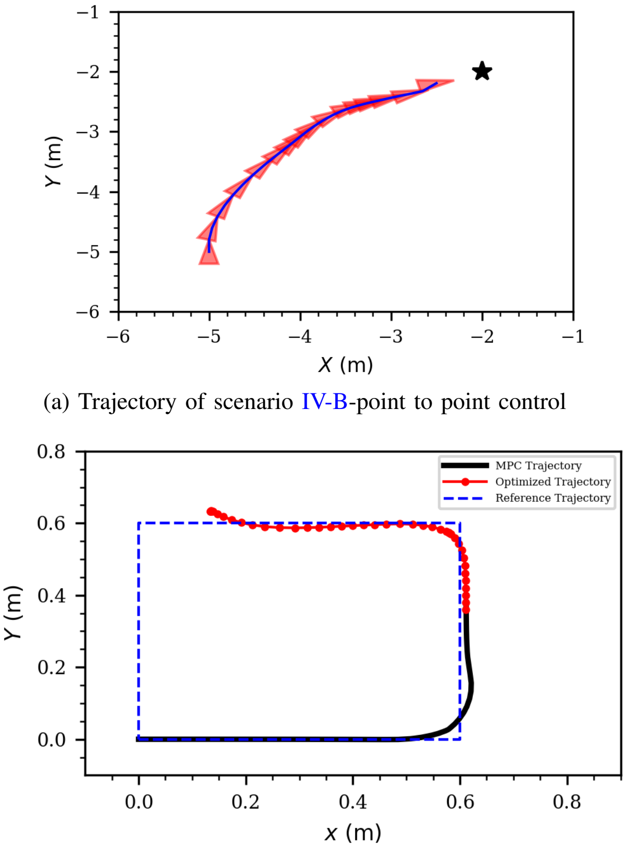
The cost functions for the two optimal control problems are as follows:

$$\min_{\boldsymbol{u},\boldsymbol{x}} \sum_{k=1}^{N} \left\| \begin{bmatrix} 100 & 0 & 0 \\ 0 & 100 & 0 \\ 0 & 0 & 1 \end{bmatrix} [\boldsymbol{x}_k - \boldsymbol{x}_k^r] \right\|^2$$

First, in the point-to-point navigation problem, $\boldsymbol{x}_k^r$ is constant at $(-2, -2, 0)$ for all horizons N. Second, in the reference trajectory tracking problem, the reference trajectory $\boldsymbol{x}_k^r$ changes at each time step. In the reference trajectory problem, $\boldsymbol{x}_k^r$ can be set using a global planner or similar methods. Here, a rectangular reference path was used.

The quadratic cost function applies weights only to the state variables p_x and p_y to ensure position control. The remaining parameters are summarized in Table (II).

Figure (4a) shows the open-loop trajectory, illustrating the optimal path generated by the mobile robot toward the reference state $[-2, -2, 0]^\top$. Figure (4b) depicts the simulation of Model Predictive Control, demonstrating the robot effectively following an infeasible reference trajectory.



(a) Trajectory of scenario IV-B-point to point control

(b) Trajectory of scenario IV-B-Trajectory tracking control

Fig. 4: Results of scenario IV-B

Parameter	Value	Parameter	Value
v_{max} (m/s)	0.5	v_{min} (m/s)	0
w_{max} (N)	$-\pi/4$	w_{min} (N)	$\pi/4$
N	50	Δt(s)	0.5

TABLE II: Paramter of scenario IV-B

C. Pose to Pose control of Mobiel Robot

The dynamics are modeled using the bicycle kinematics model as follows:

$$\begin{aligned}\dot{p_x} &= v\cos\theta \\ \dot{p_y} &= v\sin\theta \\ \dot{\theta} &= \frac{v}{L}\tan\phi\end{aligned} \tag{21}$$

Here, the state variables are the same as in Section IV-B, and the control variables are $\boldsymbol{u} = [v, \phi]^\top$.

Parameter	Value	Parameter	Value
L (m)	0.1	Δ (sec)	0.1
$v_{\max}$ (m/s)	0.5	$v_{\min}$ (m/s)	0
$\phi_{\max}$ (N)	$-\pi/4$	$\phi_{\min}$ (N)	$\pi/4$
N	50		

TABLE III: Paramter of scenario IV-C

The cost function is defined as follows to ensure efficient and smooth control inputs:

$$\min \sum_{k=0}^{N-1} \|\boldsymbol{u}_k\|^2 + \gamma \sum_{k=0}^{N-2} \|\boldsymbol{u}_{k+1} - \boldsymbol{u}_k\|^2 \tag{22}$$

To ensure that the mobile robot reaches the desired pose, an equality constraint $\boldsymbol{x}_N = [0, 0.5, 0]$ was added at the terminal time. The remaining parameters are summarized in Table (III).

As shown in Figures (5a) and (5b), the mobile robot successfully reaches the specified position at the terminal time with smooth control inputs.

D. Experiment results

In all problems, the Ceres-NMPC algorithm satisfied the given constraints and converged to a local optimum. Figure (5c) illustrates the infinity norm of the residuals corresponding to the constraints ($\|g\|_\infty$) for each iteration of the Augmented Lagrangian Method during experiment (IV-C), showing convergence after the 60th iteration.

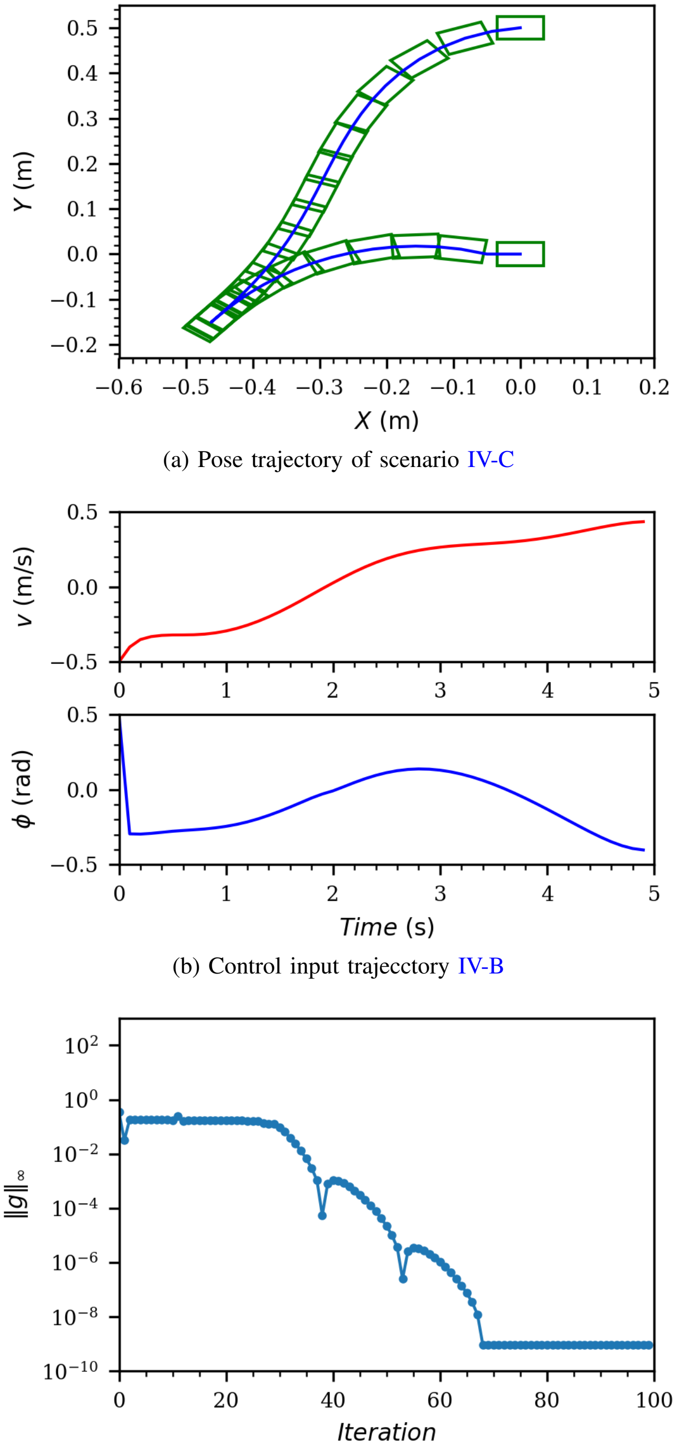
The convergence of the algorithm was evaluated by running the same problem (IV-C) 50 times with random initial guesses. Figure (6) depicts the time and iteration counts required to meet the termination conditions. On average, convergence was achieved in 2.4 seconds with 118 iterations of the Augmented Lagrangian Method.

As with other nonlinear optimization methods, the convergence speed of the algorithm varied depending on the quality of the initial guess (warm start). Parameters determining the convergence of the Ceres-NMPC algorithm include the convergence criteria of the Ceres solver in the inner loop and the penalty parameter update parameter α and convergence threshold parameter β in the outer loop. These were manually tuned for optimal performance in this study.

V. Conclusion and Future Work

In this study, nonlinear Model Predictive Control (NMPC) was implemented using the open-source Ceres solver, which is tailored for nonlinear least squares problems. The experimental results so far indicate that the algorithm's convergence is sensitive to parameter settings.

The Augmented Lagrangian Method often encounters numerical instability (numerical ill-conditioning) as the penalty parameter increases. This can lead to tail convergence, where



(a) Pose trajectory of scenario IV-C

(b) Control input trajecctory IV-B

(c) Constraint violation indices : l_∞-norm

Fig. 5: Results of scenario IV-C

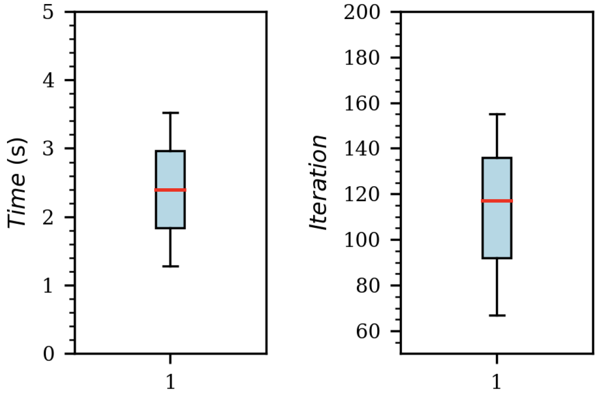


Fig. 6: Converge time and Iteration number of scenario IV-C with random initial guess

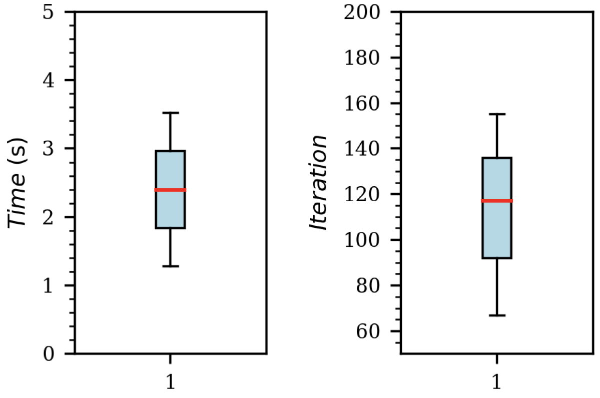


Fig. 7: Converge time and Iteration number of scenario 4b with warm start

the solution stagnates near the optimal value without fully converging. Such issues were addressed in [14] by projecting onto the constraints. In the future, we plan to adopt similar techniques to improve the convergence of the algorithm.

Additionally, we aim to utilize Ceres solver's capability of optimization on manifolds to solve optimal control problems involving rotational dynamics using Ceres-NMPC. Furthermore, introducing slack variables will enable the handling of not only box constraints but also general nonlinear constraints, which is another goal for future work.

References

[1] H. Nguyen, M. Kamel, K. Alexis, and R. Siegwart, "Model predictive control for micro aerial vehicles: A survey," in *2021 European Control Conference (ECC)*. IEEE, 2021, pp. 1556–1563.

[2] U. Eren, A. Prach, B. B. Koçer, S. V. Raković, E. Kayacan, and B. Açıkmeşe, "Model predictive control in aerospace systems: Current state and opportunities," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 7, pp. 1541–1566, 2017.

[3] A. Wachter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, no. 1, pp. 25—57, 2006.

[4] P. E. Gill, W. Murray, and M. A. Saunders, "SNOPT: An SQP algorithm for large-scale constrained optimization," *SIAM Journal on Optimization*, vol. 12, no. 4, pp. 979–1006, 2002.

[5] L. Biegler and V. Zavala, "Large-scale nonlinear programming using IPOPT: An integrating framework for enterprise-wide dynamic optimization," *Computers and Chemical Engineering*, vol. 33, no. 3, pp. 575–582, 2009.

[6] M. Diehl, H. J. Ferreau, N. Haverbeke, L. Magni, D. Raimondo, and F. Allgower, "Efficient numerical methods for nonlinear mpc and moving horizon estimation," 2009-01-01.

[7] K. Nguyen, S. Schoedel, A. Alavilli, B. Plancher, and Z. Manchester, "TinyMPC: Model-predictive control on resource-constrained microcontrollers," 2024.

[8] R. Verschueren, G. Frison, D. Kouzoupis, J. Frey, N. van Duijkeren, A. Zanelli, B. Novoselnik, T. Albin, R. Quirynen, and M. Diehl, "acados – a modular open-source framework for fast embedded optimal control," *Mathematical Programming Computation*, 2021.

[9] S. Agarwal, K. Mierle, and T. C. S. Team, "Ceres Solver." [Online]. Available: https://github.com/ceres-solver/ceres-solver

[10] M. Giftthaler, M. Neunert, M. Stäuble, J. Buchli, and M. Diehl, "A family of iterative gauss-newton shooting methods for nonlinear optimal control," 2017.

[11] M.-G. Kim and K.-K. K. Kim, "An extension of interior point differential dynamic programming for optimal control problems with second-order conic constraints," *The Transactions of The Korean Institute of Electrical Engineers*, vol. 71, no. 11, pp. 1666–1672, 2022.

[12] G. Frison and M. Diehl, "Hpipm: a high-performance quadratic programming framework for model predictive control," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 6563–6569, 2020, 21st IFAC World Congress.

[13] J. Gondzio and A. Grothey, "A new unblocking technique to warmstart interior point methods based on sensitivity analysis," *SIAM Journal on Optimization*, vol. 19, no. 3, pp. 1184–1210, 2008.

[14] T. A. Howell, B. E. Jackson, and Z. Manchester, "ALTRO: A fast solver for constrained trajectory optimization," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 7674–7679.

[15] W. Jallet, A. Bambade, N. Mansard, and J. Carpentier, "ProxNLP: A primal-dual augmented lagrangian solver for nonlinear programming in robotics and beyond," 2022.

[16] T. Qin, P. Li, and S. Shen, "VINS-Mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[17] C. Kanzow, N. Yamashita, and M. Fukushima, "Levenberg–Marquardt methods with strong local convergence properties for solving nonlinear equations with convex constraints," *Journal of Computational and Applied Mathematics*, vol. 172, no. 2, pp. 375–397, 2004.

[18] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2006.

[19] E. G. Birgin and J. M. Martínez, *Practical Augmented Lagrangian Methods for Constrained Optimization*, 2014.

[20] D. Mayne, J. Rawlings, C. Rao, and P. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.