Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Computational Robotics Lab
Prof. Stelian Coros

Semester Thesis

A Model Predictive Controller for Quadrupedal Locomotion

Spring Term 2021

Supervised by:
Dongho Kang
Stelian Coros

Author:
Luohong Wu

Contents

Abstract **iii**

Symbols **v**

1 Introduction **1**

2 Model Predictive Control **2**

2.1 Problem Statement 2

2.2 Linearization and Discretization of Dynamics 4

3 Computationally Efficient MPC Formulation **6**

3.1 Eliminate Swing Legs 6

3.2 Substitue Robot's States 6

3.3 Final Formulation 8

4 Results and Conclusions **9**

4.1 Set Up the Robot 9

4.2 Experiment Results 9

4.3 Comparison 9

4.4 Conclusion 10

4.5 Future Work 10

Bibliography **11**

Abstract

The goal of this semester project is to efficiently produce robust locomotion for a commercial quadrupedal robot platform Unitree A1 by using a model-based controller, more specifically a model predictive controller. The main contribution of this work is twofold: first, we set up the framework for controlling A1 based on the Robot Operating System. The robot's states are updated in our simulation platform. Second, we applied a model predictive control approach for the A1 robot model. The model predictive control approach can perform more robust locomotion tasks as it finds an optimal system input to track command trajectories in a long time horizon while taking into account the system's dynamics. Finally, we proposed a more computationally efficient MPC formulation by effectively reducing the number of optimization variables and constraints that results in real-time execution on A1 robot's onboard PC with limited computational power. Our formulation is 6 times faster than the previous MPC formulation. With our formulation, the MPC-based controller can be run in real-time with a time horizon of 1.2 seconds on A1, which is impossible for the previous formulation.

Symbols

Symbols

ϕ, θ, ψ	roll, pitch and yaw angle
x, y, z	x, y, z coordinate
$\boldsymbol{\Theta}$	orientation
$\boldsymbol{p}$	position
$\boldsymbol{\omega}$	angular velocity
$\dot{\boldsymbol{p}}$	linear velocity
$\boldsymbol{f}$	ground reaction force
$\boldsymbol{x}$	robot's states
$\boldsymbol{u}$	control inputs
$\boldsymbol{g}$	gravitational acceleration
$\boldsymbol{R}, \boldsymbol{W}$	weight matrix
$\boldsymbol{A}, \boldsymbol{B}$	dynamics matrix
$\boldsymbol{P}, \boldsymbol{Q}, \boldsymbol{G}$	derived dynamics matrix
μ	friction coefficient
T	time horizon
dt	time step size
N	discretized time horizon

Acronyms and Abbreviations

MPC	Model Predictive Control
QP	Quadratic Programming

Chapter 1

Introduction

Locomotion of legged robots is a challenging problem because a legged robot is a floating-base system with high degrees of freedom and complex dynamics. In addition, the system is under-actuated because the base coordinates can not be control directly. In order to control the robot's base, contact constraints with the environment need to be taken into account, which means simple methods such as inverse kinematics will fail. For producing robust locomotion for legged robots, many models have been proposed to produce robust locomotion for quadrupedal robots. One of the state-of-the-art models is hierarchical operational space control. [1]. In this model, different tasks and constraints such as equations of motion and motion tasks are formulated into prioritized quadratic problems. These problems are either solved by standard QP solver [2] or using an analytical way by null-space projection [1]. However, it's not intuitive to determine the priorities for different tasks under different scenarios. In addition, it's also complicated to consider inequality constraints when they are broken easily. Another state-of-the-art model is Model Predictive Control(MPC). Given a system's inputs, MPC simulates the system's dynamics for some time steps and optimizes the inputs to minimize the objective function such as tracking error, etc. Finally, MPC executes only the inputs of the first-time step in order for feedback control. Due to its advantage of robustness and simple formulation, now it has also been applied on robot locomotion control [3, 4]. The robust performance of this MPC-based controller has been illustrated by experiments[3]. However, the controller is computationally expensive and can not be used in real-time if the simulation time horizon is too long.
In this work, we proposed and implemented a more computationally efficient formulation of MPC for Unitree A1 based on condensing technology [5]. The main contribution of this work is twofold: first, we developed a software for controlling A1 [6] based on the Robot Operating System [7]. Second, we applied a model predictive control approach for the A1 robot model. The model predictive control approach can perform more robust locomotion tasks as it finds an optimal system input to track command trajectories in a long time horizon while taking into account the system's dynamics. In the end, a more computationally efficient MPC formulation was proposed, which reduced the problem size and improved the speed more than 6 times. The proposed formulation can be executed in real-time on the onboard PC which has limited computational power.

Chapter 2

Model Predictive Control

In order to produce robust locomotion for Unitree A1 (Figure 2.1) the task is divided into two steps. The first step is trajectory planning. The trajectory is kinematic level, which means it includes only the robot's states. In order to compute locations of each footstep, a linear combination of the corresponding hip locations, a Raibert heuristic, and a velocity-based feedback term from the capture point is used [8]. The second step is trajectory tracking using MPC, which is the focus of this work. After the ground reaction forces are computed, the Jacobian transpose method was used to transfer the forces to motor torques. For all swing legs, the simple inverse kinematics method is used because there is no force for swing legs.

2.1 Problem Statement

For making the problem simpler, the robot is simplified as a lump with 4 legs, which is shown in figure 2.2. Moreover, the robot's orientation is expressed as a vector of X-Y-Z Euler angles $\boldsymbol{\Theta} = [\theta \quad \psi \quad \phi]^T$, where θ is the pitch, ψ is the yaw and ϕ is the roll. The robot's position is expressed as $\boldsymbol{p} = [x \quad y \quad z]^T$. In addition, the robot's linear velocity and angular velocity is $\boldsymbol{v} = \dot{\boldsymbol{p}}$ and

$$\boldsymbol{\omega} = \begin{bmatrix} \cos(\theta)cos(\psi) & 0 & -sin(\psi) \\ 0 & 1 & 0 \\ \cos(\theta)sin(\psi) & 0 & cos(\psi) \end{bmatrix} \dot{\boldsymbol{\Theta}} = \boldsymbol{R_r}\dot{\boldsymbol{\Theta}} \tag{2.1}$$

In this case, the robot's state can be expressed as $\boldsymbol{x} = [\boldsymbol{\Theta}^T \quad \boldsymbol{p}^T \quad \boldsymbol{\omega}^T \quad \dot{\boldsymbol{p}}^T]^T$. The robot has 4 legs, the ground reaction force of the j-th leg is expressed as $\boldsymbol{f_j} = [x \quad y \quad z]^T$ and the robot's ground reaction forces at $t = i$ are

$$\boldsymbol{u_i} = [\boldsymbol{f_1}^T \quad \boldsymbol{f_2}^T \quad \boldsymbol{f_3}^T \quad \boldsymbol{f_4}^T]^T. \tag{2.2}$$

Assume that the time horizon is N. Then the kinematic level trajectory is $\boldsymbol{x_{ref}} = [\boldsymbol{x_1'}^T \quad \boldsymbol{x_2'}^T \quad \dots \quad \boldsymbol{x_N'}^T]^T$,where $\boldsymbol{x_i'} = [\boldsymbol{\Theta_i'^T} \quad \boldsymbol{p_i'^T} \quad \boldsymbol{\omega_i'^T} \quad \dot{\boldsymbol{p}}_{\boldsymbol{i}}'^{\boldsymbol{T}}]^{\boldsymbol{T}}$. The goal is to track this trajectory accurately, which can be expressed as

$$\min_{\boldsymbol{x},\boldsymbol{u}} \quad \sum_{i=1}^{N} \|\boldsymbol{x_i} - \boldsymbol{x_i'}\|_{\boldsymbol{R}}^2 \tag{2.3}$$

The main idea of MPC is optimizing constrained optimization variables $\boldsymbol{x}$ and $\boldsymbol{u}$ to

Figure 2.1: Unitree A1

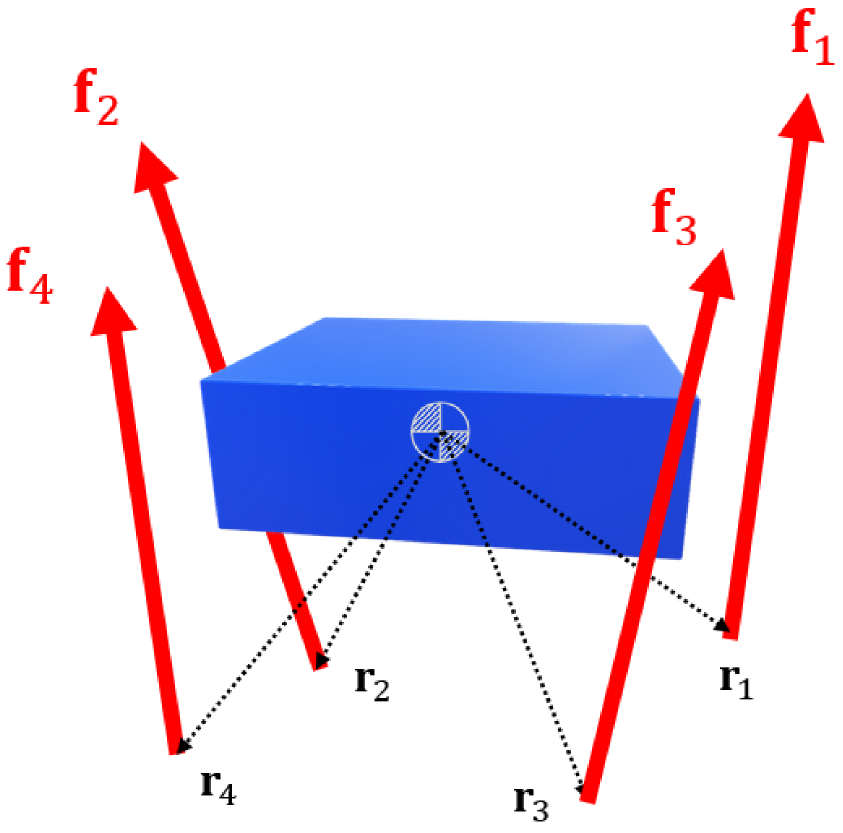


Figure 2.2: model A1 as a legged lump

minimize an objective function within N time steps, where

$$\boldsymbol{x} = [\boldsymbol{x_1}^T \quad \boldsymbol{x_2}^T \quad \dots \quad \boldsymbol{x_N}^T]^T \tag{2.4}$$

and $\boldsymbol{u} = [\boldsymbol{u_0}^T \quad \boldsymbol{u_1}^T \quad \dots \quad \boldsymbol{u_{N-1}}^T]^T$. In this case, the objective function is a weighted tracking error and control cost,$\|\boldsymbol{x} - \boldsymbol{x_{ref}}\|^2_{\boldsymbol{R}} + \|\boldsymbol{u}\|^2_{\boldsymbol{W}}$, where $\boldsymbol{R}$ and $\boldsymbol{W}$ are weight matrices. In addition, there are some inequality constrains

$$\mathrm{I}(\boldsymbol{x}, \boldsymbol{u}) > \mathrm{Values_{In}} \tag{2.5}$$

and equality constrains

$$\mathrm{E}(\boldsymbol{x}, \boldsymbol{u}) = \mathrm{Values_{Eq}} \tag{2.6}$$

The inequality constrains include a) friction cones and b) input limits, and the equality constrains include a) ground reaction forces of swing legs are always **0** and b) the robot's dynamics. This can be formulated as a constrained optimization problem (2.1), where μ is the friction coefficient.

$$\begin{aligned}
\min_{\boldsymbol{x},\boldsymbol{u}} \quad & \|\boldsymbol{x} - \boldsymbol{x_{ref}}\|^2_{\boldsymbol{R}} + \|\boldsymbol{u}\|^2_{\boldsymbol{W}} \\
\text{s.t.} \quad & -\mu f_p^y < f_p^x < \mu f_p^y \quad \forall p \in \{\text{stance legs}\} \\
& -\mu f_p^y < f_p^z < \mu f_p^y \quad \forall p \in \{\text{stance legs}\} \\
& \text{minLimits} < \boldsymbol{x_i} < \text{maxLimits} \\
& \text{minLimits} < \boldsymbol{u_i} < \text{maxLimits} \\
& \boldsymbol{f_q} = \mathbf{0} \quad \forall q \in \{\text{swing legs}\} \\
& \boldsymbol{x_{i+1}} = \mathrm{D}(\boldsymbol{x_i}, \boldsymbol{u_i}) \quad \forall \quad 0 \le i \le N-1
\end{aligned} \tag{2.7}$$

2.2 Linearization and Discretization of Dynamics

The dynamics in problem (2.1) is nonlinear and can not be solved quickly. In order to solve problem 2.1 efficiently, the problem is formulated as a Quadratic Programming(QP) problem by linearizing the dynamics of the robot. Firstly, from (2.1), it is known that

$$\dot{\boldsymbol{\Theta}} = \boldsymbol{R_r}^{-1}\boldsymbol{\omega} \tag{2.8}$$

If the roll angle and pitch angle is small, (2.8) can be approximated as

$$\dot{\boldsymbol{\Theta}} \approx \boldsymbol{R_y}(\psi)\boldsymbol{\omega} \tag{2.9}$$

Under the assumptions that a) the tracking error is small and b) the roll and pitch angles are small, the , the dynamics of the robot can be linearized in the following way [3]:

$$\begin{aligned}
\frac{d}{dx}\begin{bmatrix}\boldsymbol{\Theta}\\ \boldsymbol{p}\\ \boldsymbol{\omega}\\ \dot{\boldsymbol{p}}\end{bmatrix} = & \begin{bmatrix}\mathbf{0}_3 & \mathbf{0}_3 & \boldsymbol{R_y} & \mathbf{0}_3\\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{1}_3\\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3\\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3\end{bmatrix}\begin{bmatrix}\boldsymbol{\Theta}\\ \boldsymbol{p}\\ \boldsymbol{\omega}\\ \dot{\boldsymbol{p}}\end{bmatrix} + \\
& \begin{bmatrix}\mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3\\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{1}_3\\ \boldsymbol{I}^{-1}[\boldsymbol{r_1}]_x & \boldsymbol{I}^{-1}[\boldsymbol{r_2}]_x & \boldsymbol{I}^{-1}[\boldsymbol{r_3}]_x & \boldsymbol{I}^{-1}[\boldsymbol{r_4}]_x\\ \mathbf{1}_3/m & \mathbf{1}_3/m & \mathbf{1}_3/m & \mathbf{1}_3/m\end{bmatrix}\begin{bmatrix}\boldsymbol{f_1}\\ \boldsymbol{f_2}\\ \boldsymbol{f_3}\\ \boldsymbol{f_4}\end{bmatrix} + \begin{bmatrix}\mathbf{0}_3\\ \mathbf{0}_3\\ \mathbf{0}_3\\ \boldsymbol{g}\end{bmatrix}
\end{aligned} \tag{2.10}$$

Moreover, with dt being the time step size, equation 2.2 can be discretized in the following way:

$$\begin{aligned}\boldsymbol{x}_{i+1} = &\begin{bmatrix}\mathbf{1}_3 & \mathbf{0}_3 & \boldsymbol{R_y} & \mathbf{0}_3\\ \mathbf{0}_3 & \mathbf{1}_3 & \mathbf{0}_3 & \mathbf{1}_3\\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{1}_3 & \mathbf{0}_3\\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{1}_3\end{bmatrix}\boldsymbol{x}_i +\\ &\begin{bmatrix}\mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3\\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{1}_3\\ \boldsymbol{I}^{-1}[\boldsymbol{r_1}]_x & \boldsymbol{I}^{-1}[\boldsymbol{r_2}]_x & \boldsymbol{I}^{-1}[\boldsymbol{r_3}]_x & \boldsymbol{I}^{-1}[\boldsymbol{r_4}]_x\\ \mathbf{1}_3/m & \mathbf{1}_3/m & \mathbf{1}_3/m & \mathbf{1}_3/m\end{bmatrix}\boldsymbol{u}_i + \begin{bmatrix}\mathbf{0}_3\\ \mathbf{0}_3\\ \mathbf{0}_3\\ \boldsymbol{g}\end{bmatrix} dt\end{aligned} \tag{2.11}$$

For convenience, define:

$$\begin{aligned}\boldsymbol{A}_i &= \begin{bmatrix}\mathbf{1}_3 & \mathbf{0}_3 & \boldsymbol{R_y} & \mathbf{0}_3\\ \mathbf{0}_3 & \mathbf{1}_3 & \mathbf{0}_3 & \mathbf{1}_3\\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{1}_3 & \mathbf{0}_3\\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{1}_3\end{bmatrix}\\ \boldsymbol{B}_i &= \begin{bmatrix}\mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3\\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{1}_3\\ \boldsymbol{I}^{-1}[\boldsymbol{r_1}]_x & \boldsymbol{I}^{-1}[\boldsymbol{r_2}]_x & \boldsymbol{I}^{-1}[\boldsymbol{r_3}]_x & \boldsymbol{I}^{-1}[\boldsymbol{r_4}]_x\\ \mathbf{1}_3/m & \mathbf{1}_3/m & \mathbf{1}_3/m & \mathbf{1}_3/m\end{bmatrix} \quad \hat{\boldsymbol{g}} = \begin{bmatrix}\mathbf{0}_3\\ \mathbf{0}_3\\ \mathbf{0}_3\\ \boldsymbol{g}\end{bmatrix}\end{aligned} \tag{2.12}$$

In this case, $\boldsymbol{x}_{i+1} = \boldsymbol{A}_i\boldsymbol{x}_i + \boldsymbol{B}_i\boldsymbol{u}_i + \hat{\boldsymbol{g}} \cdot dt$.
By linearization and discretization, the complex original problem 2.1 can not be transferred to the following simple QP problem, which can be solved fast.

$$\begin{aligned}\min_{\boldsymbol{x},\boldsymbol{u}} \quad & \|\boldsymbol{x} - \boldsymbol{x_{ref}}\|_{\boldsymbol{R}}^2 + \|\boldsymbol{u}\|_{\boldsymbol{W}}^2\\ \text{s.t.} \quad & -\mu f_p^y < f_p^x < \mu f_p^y \quad \forall p \in \{\text{stance legs}\}\\ & -\mu f_p^y < f_p^z < \mu f_p^y \quad \forall p \in \{\text{stance legs}\}\\ & \text{minLimits} < \boldsymbol{x_i} < \text{maxLimits}\\ & \text{minLimits} < \boldsymbol{u_i} < \text{maxLimits}\\ & \boldsymbol{f_q} = \mathbf{0} \quad \forall q \in \{\text{swing legs}\}\\ & \boldsymbol{x_{i+1}} = \boldsymbol{A}_i\boldsymbol{x}_i + \boldsymbol{B}_i\boldsymbol{u}_i + \hat{\boldsymbol{g}} \cdot dt \quad \forall \quad 0 \le i \le N-1\end{aligned} \tag{2.13}$$

Chapter 3

Computationally Efficient MPC Formulation

With the used QP solver, the computational complexity of a QP problem is proportional to the number of optimization variables, which is also the size of the problem. In order to speed up the optimization problem (2.5), two methods are induced to reduce the problem size.

3.1 Eliminate Swing Legs

For all swing legs, the ground reaction force is constrained to $\mathbf{0}_3$. So there is no need to do any optimization for swing legs, and they can be eliminated from the optimization variables. More specifically, at time $t = i$, let n_i be the number of stance legs, $\boldsymbol{u}_i$ is changed from $\boldsymbol{u_i} = [\boldsymbol{f_1}^T \quad \boldsymbol{f_2}^T \quad \boldsymbol{f_3}^T \quad \boldsymbol{f_4}^T]^T$ to $\boldsymbol{u_i} = [\boldsymbol{f_1}^T \quad \dots \quad \boldsymbol{f}_{n_i}{}^T]^T$. By this method, the number of optimization variables is reduced from $N(12+4)$ to $12N + \sum_{i=1}^{N} n_i$, where $N = \frac{T}{dt}$ is the discretized time horizon. In addition, the equality constrains of swing legs are also eliminated from the problem, which is $\boldsymbol{f_q} = \mathbf{0} \quad \forall q \in \{\text{swing legs}\}$.

3.2 Substitue Robot's States

Since the robot's initial state $\boldsymbol{x}_0$ and dynamics $\boldsymbol{x_{i+1}} = \boldsymbol{A}_i\boldsymbol{x}_i + \boldsymbol{B}_i\boldsymbol{u}_i + \hat{\boldsymbol{g}} \cdot dt$ is known, the robot's state $\boldsymbol{x}$ can be represented using only control inputs $\boldsymbol{u}$, which means $\boldsymbol{x} = \boldsymbol{x}(\boldsymbol{u})$. For convenience, we define $\bar{\boldsymbol{g}} = \hat{\boldsymbol{g}} \cdot dt$. By induction, the following equations can be derived:

$$
\begin{aligned}
&\boldsymbol{x}_1 = \boldsymbol{A}_0\boldsymbol{x}_0 + \boldsymbol{B}_0\boldsymbol{u}_0 + \bar{\boldsymbol{g}} \\
&\boldsymbol{x}_2 = \boldsymbol{A}_1\boldsymbol{A}_0\boldsymbol{x}_0 + \boldsymbol{A}_1\boldsymbol{B}_0\boldsymbol{u}_0 + \boldsymbol{A}_1\bar{\boldsymbol{g}} + \boldsymbol{B}_1\boldsymbol{u}_1 + \bar{\boldsymbol{g}} \\
&\boldsymbol{x}_3 = \boldsymbol{A}_2\boldsymbol{A}_1\boldsymbol{A}_0\boldsymbol{x}_0 + \boldsymbol{A}_2\boldsymbol{A}_1\boldsymbol{B}_0\boldsymbol{u}_0 + \boldsymbol{A}_2\boldsymbol{A}_1\bar{\boldsymbol{g}} + \boldsymbol{A}_2\boldsymbol{B}_1\boldsymbol{u}_1 + \boldsymbol{A}_2\bar{\boldsymbol{g}} + \boldsymbol{B}_2\boldsymbol{u}_2 + \bar{\boldsymbol{g}} \\
&\dots\dots \\
&\boldsymbol{x}_N = (\prod_{i=0}^{N-1} \boldsymbol{A}_i)\boldsymbol{x}_0 + \sum_{k=0}^{N-1} \{(\prod_{i=k+1}^{N-1} \boldsymbol{A}_i)\boldsymbol{B}_k\boldsymbol{u}_k\} + \sum_{i=1}^{N-1} ((\prod_{k=i}^{N-1} \boldsymbol{A}_k)\bar{\boldsymbol{g}}) + \bar{\boldsymbol{g}}
\end{aligned}
\tag{3.1}
$$

The final result of substitution is:

$$\boldsymbol{x}(\boldsymbol{u}) = \begin{bmatrix} \boldsymbol{A}_0\boldsymbol{x}_0 \\ \boldsymbol{A}_1\boldsymbol{A}_0\boldsymbol{x}_0 \\ \boldsymbol{A}_2\boldsymbol{A}_1\boldsymbol{A}_0\boldsymbol{x}_0 \\ \vdots \\ \vdots \\ (\prod_{i=0}^{N-1}\boldsymbol{A}_i)\boldsymbol{x}_0 \end{bmatrix} + \begin{bmatrix} \boldsymbol{B}_0 & & & & & \\ \boldsymbol{A}_1\boldsymbol{B}_0 & \boldsymbol{B}_1 & & & & \\ \boldsymbol{A}_2\boldsymbol{A}_1\boldsymbol{B}_0 & \boldsymbol{A}_2\boldsymbol{B}_1 & \boldsymbol{B}_2 & & & \\ \vdots & \vdots & \vdots & \ddots & & \\ \vdots & \vdots & \vdots & \vdots & \ddots & \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ (\prod_{i=1}^{N-1}\boldsymbol{A}_i)\boldsymbol{B}_0 & (\prod_{i=2}^{N-1}\boldsymbol{A}_i)\boldsymbol{B}_1 & \dots & \dots & \boldsymbol{A}_{N-1}\boldsymbol{B}_{N-2} & \boldsymbol{B}_{N-1} \end{bmatrix} \begin{bmatrix} \boldsymbol{u}_0 \\ \boldsymbol{u}_1 \\ \boldsymbol{u}_2 \\ \vdots \\ \vdots \\ \boldsymbol{u}_{N-1} \end{bmatrix}$$

$$+ \begin{bmatrix} \bar{\boldsymbol{g}} \\ \boldsymbol{A}_1\bar{\boldsymbol{g}} + \bar{\boldsymbol{g}} \\ \boldsymbol{A}_2(\boldsymbol{A}_1\bar{\boldsymbol{g}} + \bar{\boldsymbol{g}}) + \bar{\boldsymbol{g}} \\ \vdots \\ \vdots \\ \sum_{i=1}^{N-1}((\prod_{k=i}^{N-1}\boldsymbol{A}_k)\bar{\boldsymbol{g}}) + \bar{\boldsymbol{g}} \end{bmatrix} \tag{3.2}$$

For convenience, define:

$$\boldsymbol{P} = \begin{bmatrix} \boldsymbol{A}_0\boldsymbol{x}_0 \\ \boldsymbol{A}_1\boldsymbol{A}_0\boldsymbol{x}_0 \\ \boldsymbol{A}_2\boldsymbol{A}_1\boldsymbol{A}_0\boldsymbol{x}_0 \\ \vdots \\ \vdots \\ (\prod_{i=0}^{N-1}\boldsymbol{A}_i)\boldsymbol{x}_0 \end{bmatrix}$$

$$\boldsymbol{G} = \begin{bmatrix} \bar{\boldsymbol{g}} \\ \boldsymbol{A}_1\bar{\boldsymbol{g}} + \bar{\boldsymbol{g}} \\ \boldsymbol{A}_2(\boldsymbol{A}_1\bar{\boldsymbol{g}} + \bar{\boldsymbol{g}}) + \bar{\boldsymbol{g}} \\ \vdots \\ \vdots \\ \sum_{i=1}^{N-1}((\prod_{k=i}^{N-1}\boldsymbol{A}_k)\bar{\boldsymbol{g}}) + \bar{\boldsymbol{g}} \end{bmatrix}$$

$$\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{B}_0 & & & & & \\ \boldsymbol{A}_1\boldsymbol{B}_0 & \boldsymbol{B}_1 & & & & \\ \boldsymbol{A}_2\boldsymbol{A}_1\boldsymbol{B}_0 & \boldsymbol{A}_2\boldsymbol{B}_1 & \boldsymbol{B}_2 & & & \\ \vdots & \vdots & \vdots & \ddots & & \\ \vdots & \vdots & \vdots & \vdots & \ddots & \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ (\prod_{i=1}^{N-1}\boldsymbol{A}_i)\boldsymbol{B}_0 & (\prod_{i=2}^{N-1}\boldsymbol{A}_i)\boldsymbol{B}_1 & \dots & \dots & \boldsymbol{A}_{N-1}\boldsymbol{B}_{N-2} & \boldsymbol{B}_{N-1} \end{bmatrix}$$

(3.3)

By substitution, the objective function is change from

$$\min_{\boldsymbol{x},\boldsymbol{u}} \quad \|\boldsymbol{x} - \boldsymbol{x_{ref}}\|^2_{\boldsymbol{R}} + \|\boldsymbol{u}\|^2_{\boldsymbol{W}} \tag{3.4}$$

to

$$\min_{\boldsymbol{u}} \quad \|\boldsymbol{x}(\boldsymbol{u}) - \boldsymbol{x_{ref}}\|^2_{\boldsymbol{R}} + \|\boldsymbol{u}\|^2_{\boldsymbol{W}} = \|\boldsymbol{P} + \boldsymbol{Q}\boldsymbol{u} + \boldsymbol{G} - \boldsymbol{x_{ref}}\|^2_{\boldsymbol{R}} + \|\boldsymbol{u}\|^2_{\boldsymbol{W}} \tag{3.5}$$

Compared to the original optimization problem, the number of optimization variables is reduced from $N(12+4)$ to $4N$, where $N = \frac{T}{t}$ is the discretized time horizon.

3.3 Final Formulation

By eliminating swing legs and substituting x using u, the new formulation of MPC is shown in equation 3.6.

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{u}} \quad & = \|\boldsymbol{P} + \boldsymbol{Q}\boldsymbol{u} + \boldsymbol{G} - \boldsymbol{x_{ref}}\|^2_{\boldsymbol{R}} + \|\boldsymbol{u}\|^2_{\boldsymbol{W}} \\ \text{s.t.} \quad & -\mu f_p^y < f_p^x < \mu f_p^y \quad \forall p \in \{\text{stance legs}\} \\ & -\mu f_p^y < f_p^z < \mu f_p^y \quad \forall p \in \{\text{stance legs}\} \\ & \text{minLimits} < \boldsymbol{u_i} < \text{maxLimits} \end{aligned} \tag{3.6}$$

It is worth mentioning that the minimal ground reaction force should be larger than 0, which can produce more balanced force distribution compared to a minimal value of 0. Compared to the original optimization problem (2.5), the number of optimization variables is reduced from $N(12+4)$ to only $\sum_{i=0}^{N-1} n_i$, where $N = \frac{T}{t}$ is the discretized time horizon and n_i is the number of stance legs when time $t = i$. The new QP problem should be solved faster than the original problem.

Chapter 4

Results and Conclusions

4.1 Set Up the Robot

In order to test MPC on the robot, a software based on Robot Operating System was implemented. The structure of software is shown in figure 2.3. Given the user software and A1's low level software, the implemented software can communicate robot' states and commands between the other two software [9].

4.2 Experiment Results

By solving the QP problem (2.5), the MPC can be applied to control the robot. When the time horizon is 0.8 s, it worked quite well. The demo video is available here: https://youtu.be/qMX67qDd-2k. However, once the time horizon was increased to 1.0 s, due to computational complexity, the frame rate drops dramatically to only 12 FPS, which is much smaller than 30 FPS. The video is available here: https://youtu.be/P4xwSANzpwk.
Applying the new MPC to control the robot with the same time horizon T=1.0 s, the frame rate is improved from 12 FPS to 30 FPS, which means now the MPC-based controller can be used in the real robot. The video of the demo is available here: https://youtu.be/u5-EJJdgFO4.

4.3 Comparison

In order to compare the two formulations, a more specific experiment was conducted. Under trotting pattern, the two MPC-based controller were applied with different

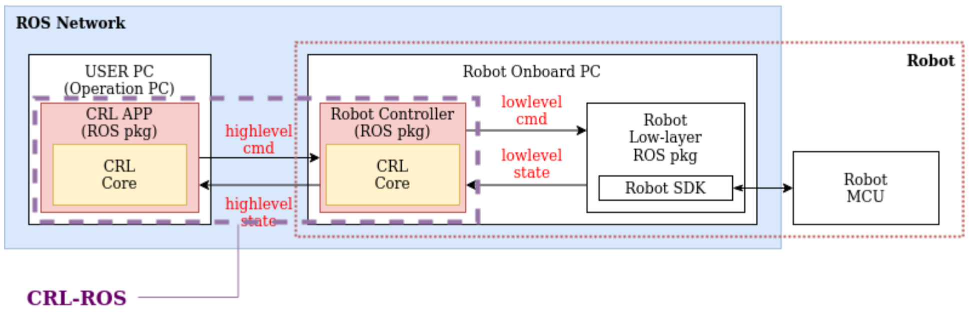


Figure 4.1: structure of the software

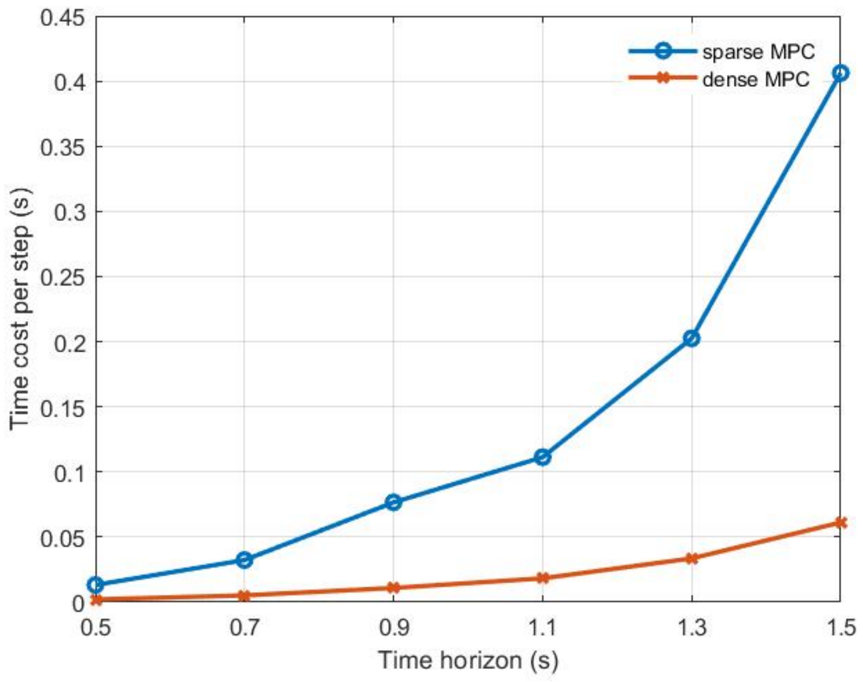


Figure 4.2: Comparison of sparse formulation and dense formulation

time horizons, and the time of one-step optimization was recorded. The result is shown in figure 4.1. From the chart, it can be seen that the new formulation is more than 6 times faster than the original formulation.

4.4 Conclusion

In this work, a framework based on ROS for controlling Unitree A1 was set up. And from experiments, it was verified that the MPC model can control the robot robustly. However, it's too computationally expensive to be used in the real robot due to the limited computational resource of the onboard PC. In order to solve this problem, there are 2 methods used in this work, a full formulation and a dense formulation respectively. And the later method proved to be beneficial by providing a speed up of more than 6 times. By the second method, it is possible now to run MPC with a time horizon of 1.2 seconds on the robot, which is enough for different periodic gait duration.

4.5 Future Work

The immediate next step is to apply the computationally efficient MPC on the real robot. In addition, other QP solvers can be tested for more speed up. In addition, a sub module for late or early contact should also be implemented, which is helpful for more producing robust locomotion [10].

Bibliography

[1] M. Hutter, H. Sommer, C. Gehring, M. Hoepflinger, M. Bloesch, and R. Siegwart, "Quadrupedal locomotion using hierarchical operational space control," *The International Journal of Robotics Research*, vol. 33, no. 8, pp. 1062–1077, 2014.

[2] A. Bjorck, *Numerical methods for least squares problems.* Philadelphia, PA: SIAM, 1996.

[3] J. Di Carlo, P. M. Wensing, B. Katz, G. Bledt, and S. Kim, "Dynamic locomotion in the mit cheetah 3 through convex model-predictive control," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 1–9.

[4] M. Rutschmann, B. Satzinger, M. Byl, and K. Byl, "Nonlinear model predictive control for rough-terrain robot hopping," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 1859–1864.

[5] J. L. Jerez, E. C. Kerrigan, and G. A. Constantinides, "A condensed and sparse qp formulation for predictive control," in *2011 50th IEEE Conference on Decision and Control and European Control Conference*, 2011, pp. 5217–5222.

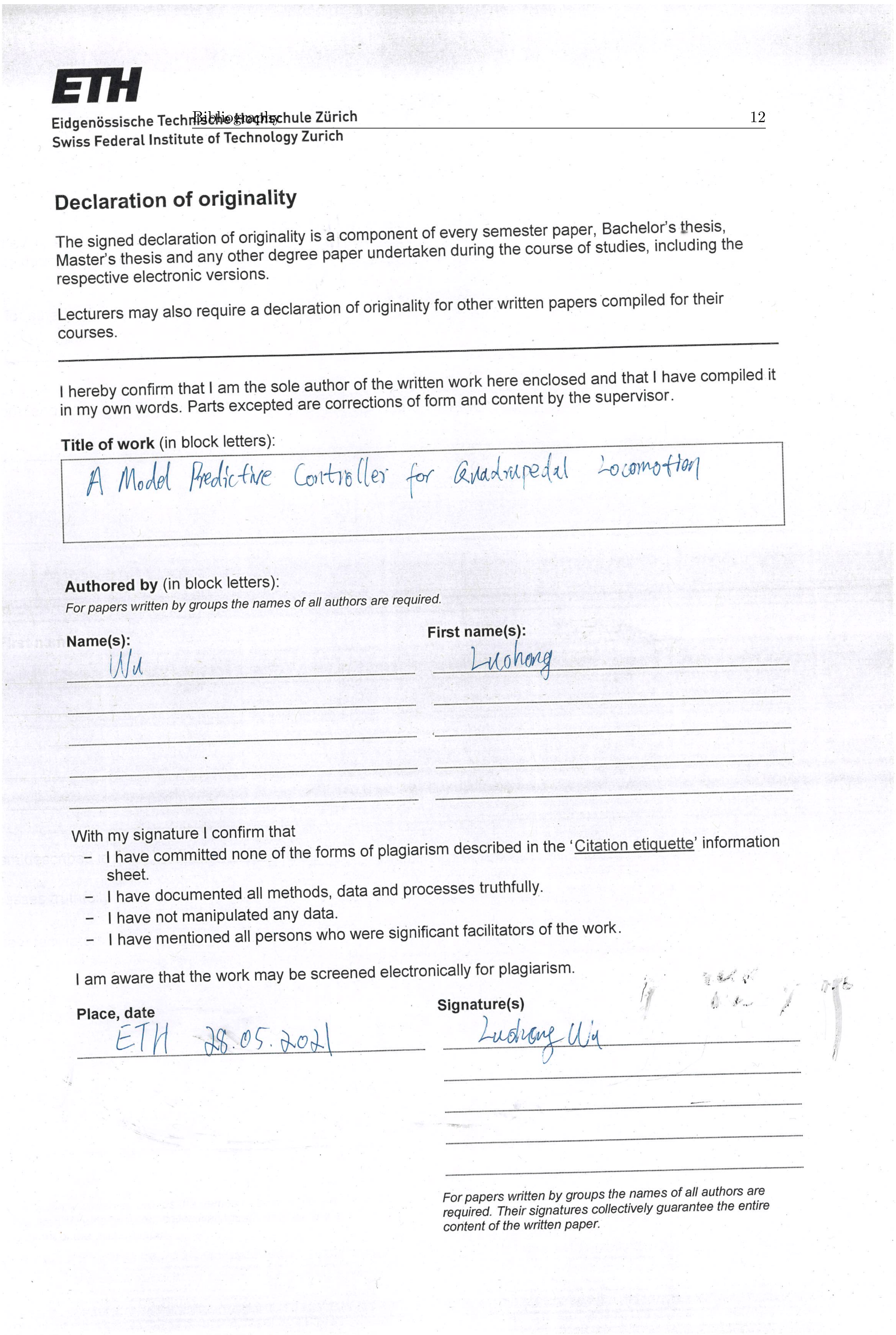
[6] Unitree Robotics, "A1 https://www.unitree.com/products/a1/."

[7] Stanford Artificial Intelligence Laboratory et al., "Robotic operating system."

[8] G. Bledt, M. J. Powell, B. Katz, J. Di Carlo, P. M. Wensing, and S. Kim, "Mit cheetah 3: Design and control of a robust, dynamic quadruped robot," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 2245–2252.

[9] Computational Robotics Lab, "Crl-ros https://gitlab.inf.ethz.ch/kangd/crl-ros/-/wikis/home."

[10] G. Bledt, P. M. Wensing, S. Ingersoll, and S. Kim, "Contact model fusion for event-based locomotion in unstructured terrains," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 4399–4406.



ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of work (in block letters):

A Model Predictive Controller for Quadrupedal Locomotion

Authored by (in block letters):
For papers written by groups the names of all authors are required.

Name(s):	**First name(s):**
Wu	Luohong

With my signature I confirm that
- I have committed none of the forms of plagiarism described in the '<u>Citation etiquette</u>' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date	**Signature(s)**
ETH 28.05.2021	Luohong Wu

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.