A Comparing Study of MPC and Control Barrier Functions Algorithms for Multi-Agent Systems in the presence of Obstacles

João Miguel Carvalho Pina Neves

Thesis to obtain the Master of Science Degree in

Information Systems and Computer Engineering

Supervisors: Daniel Silvestre
Carlos Silvestre

Examination Committee

Chairperson: Prof. José Carlos Alves Pereira Monteiro
Supervisor: Daniel Silvestre
Member of the Committee: Prof. Bruno João Nogueira Guerreiro

May 2022

1

Abstact

1.1 Introduction

1.1.1 Abstract

This document serves as an abstract for the thesis Robust MPC algorithms for **A Comparing Study of MPC and Control Barrier Functions Algorithms for Multi-Agent Systems in the presence of Obstacles**. The thesis aims to study the common methods of formation control and measure their accuracy, computation time, among other measures.

1.1.2 Motivation of the Dissertation

A formation control problem is defined by a multi-agent system which must fulfill some objective, while acting cooperatively and avoiding undesired effects such as collisions between agents. In the literature, a variety of approaches are taken depending on the individual agents specific sensing capability and level of communication between agents.

In this document, multiple types of formation control will be studied, as well as the solutions to formation control, namely, the use of Control Lyapunov Functions and Control Barriers Functions, and MPC. Notably, the use of MPC is well studied, due to its wide adoption in industrial settings, where it provides high performance and stability through the optimisation. MPC can, however, also be naturally applied in formation control systems, where the optimisation can be used to minimise the error in formations and thus achieve a desired objective. While MPC has many attractive qualities in many different areas of study, it also has its drawbacks. MPC optimisations are computationally-heavy for certain non-convex systems.

Recently, there has been a growing interest [1–3] in the use of Control Barrier Functions in combination with MPC in order to achieve a system that both guarantees the safety by using the Lyapunov Function's stability and benefits from MPC's favourable optimization qualities.

To this end, in this dissertation we aim to do a comparative work between the different methods in order to ascertain their unique advantages and disadvantages. The different methods will be simulated and then their outputs will be measured, namely, the simulation time, error of the formation, distance travelled, and actuation of the system.

We will then draw conclusions based on the output we receive from our measurements.

An implementation of the formation control, as well as the videos of the simulations, are also provided for future study in our GitHub Repository.

1.2 Formation Control Background

We can describe a discrete time system with N-agents with the following equations:

$$\begin{cases} x_i^{(k+1)} = f_i(x_i^{(k)}, u_i^{(k)}), \\ y_i^{(k)} = g_i(x_1^{(k)}, ..., x_N^{(k)}), \quad i = 1, ..., N \\ z_i^{(k)} = h_i(x_i^{(k)}), \end{cases} \tag{1.1}$$

where the position of agent i is denoted as $x_i^{(k)} \in \mathbb{R}^{n_i}$ specifies the position of agent i at the control instant k, $u_i^{(k)} \in \mathbb{R}^{p_i}$ denotes the actuation, $y_i^{(k)} \in \mathbb{R}^{h_i^{(k)}}$ denotes the measurement, and output is denoted by $z_i \in \mathbb{R}^r$. where $x_i^{(k)}$ specifies the position of agent i at the control instant k.

A physical formation control system that works in discrete time will have an underlying system in continuous time in order to manage the physical elements of the system. The discrete time system will provide an objective to the continuous time system which will then control the actions of the agent. With this formulation, we define the 3 main types of formation control:

1.2.1 Position-based control

Agent i senses their position x_i with respect to a global coordinate system. Each agent i actively controls x_i in order to achieve a global formation. To achieve a desired formation, each agent i requires only their individual desired position x_i^*, and each agent can then act independently to arrive at their desired position. Measurements y_i contain some absolute variables that are sensed with respect to a global coordinate system. The constraint (**??**) is given as:

$$F(z) := z = F(z^*). \tag{1.2}$$

where $z = [z_0 ... z_N]$. Agents actively control z_i. Communication is necessary in the event that collisions between agents are possible. In conclusion, this type of formation control requires complex sensors to actively sense the agent's position in regards to a global coordinate system, but also requires less communication between agents.

1.2.1.A Control Lyapunov Functions

The application of Lyapunov Functions to formation control problems stems from the use of Control Lyapunov Functions (CLF) and Control Barrier Functions (CBF). These use the attractive qualities of Lyapunov Functions, namely, Lyapunov Stability, to formation control problems.

Combining the Control Lyapunov Functions and Control Lyapunov Barriers, we obtain a control system in the following Quadratic Programming (QP) formulation:

$$\begin{aligned} \min_{(u,\delta) \in \mathbb{R}^{m+1}} \quad & \frac{1}{2}\|u\|^2 + \frac{1}{2}\kappa\delta^2 \\ \text{s.t.} \quad & L_p V(x) + L_j V(x) u + \gamma(V(x)) \leq \delta \quad \textbf{(CLF)}, \\ & L_p h(x) + L_j h(x) u + \alpha(h(x)) \geq 0 \quad \textbf{(CBF)} \end{aligned}$$

the CBF constraint guarantees that $u^* \in K_{CBF}(x)$ keeps the system trajectories invariant with respect to the safe set $int(C)$. The relaxation variable δ in the CLF constraints softens the stabilisation objective, maintaining the necessary feasibility of the QP.

1.2.2 Model Predictive Control

The application of MPC to the formation control problem is straightforward. According to [4], the MPC control problem can be characterised as follows. At decision instant k, the controller samples the state of the system $x(k)$. The following optimisation problem is then solved to find the control action.

$$\begin{aligned} \min_{X(k),U(k)} \quad & J(X(k), U(k)) \\ \text{s.t.} \quad & x^{(k+i+1|k)} = f(x^{(k+i|k)}, u^{(k+i|k)}) \quad (i = 0, ..., K-1), \\ & G(X(k), U(k)) \leq 0, \\ & x^{(k|k)} = x^{(k)} \end{aligned} \tag{1.4}$$

where

$$\begin{aligned} X(k) &= \{x^{(k+1|k)}, ..., x^{(k+K|k)}\} \\ U(k) &= \{u^{(k|k)}, ..., u^{(k+K-1|k)}\} \end{aligned}$$

for control horizon K. The same notation for x and u is used for equation (1.1), with the addition of the prediction notation. That is, $x^{(k+1|k)}$ is the predicted state (or position) of an agent at control instant $k+1$, based on the information at instant k.

1.2.2.A Collision Avoidance

Collision Avoidance in this system is achieved by the method detailed in [5]. In chapter 4.5.3 of the book, a method to achieve linear constraints from non-linear obstacles is presented. The method relies on creating constraints that define the planes tangent to the obstacle.

A simplified version of the proposed algorithm is used in this thesis:

Algorithm 1.1: Obstacle Avoidance Strategy

```
begin
    Solve optimisation problem without obstacle constraints
    Determine planes tangent to obstacle facing each point in the trajectory
    Solve optimisation problem with tangent planes as linear constraints
```

- **Solve optimisation problem without obstacle constraints**, saving the agent's positions in each iteration.
- **Determine planes tangent to obstacle facing each point in the trajectory**, using the saved positions. This requires a calculation to discover the tangent line which passes through the unsafe set's limits.
- **Solve optimisation problem with tangent planes as linear constraints**, which leads to a system where safety is assured.

This strategy incurs a time penalty, since it requires the problem to be solved without obstacle constraints, and then with obstacle constraints. These constraints slow down computation considerably, as is shown in the later results chapter.

1.3 Experimental Measurements

The following measures were taken in runtime to assess the viability of each method of formation control:

Time: The time of execution will be measured.

Formation Error: The current difference between the agent's positions and the desire3Experimentald formation will be measured.

Distance Travelled: The agent's total travel distance will be measured.

Energy Spent: The actuation will be measured at each iteration. This is meant as an approximation to battery power spent.

The regularly used formations place the agents in a virtual circle around a point. The point is then moved, causing the formation to follow it.

The formations tested either move in a straight line or move in a "sine wave" pattern. Certain formations will also shift the agent's positions at chosen intervals. The velocity of the formation will also be variable. The trajectory that the formation took will be illustrated along with the final results. The color of each agent's trajectory will be different to better distinguish them.

1.4 Results

The following chapter displays the aggregated results for the simulations with no obstacles, 1 obstacle, and 5 obstacles.

1.4.1 No Obstacles

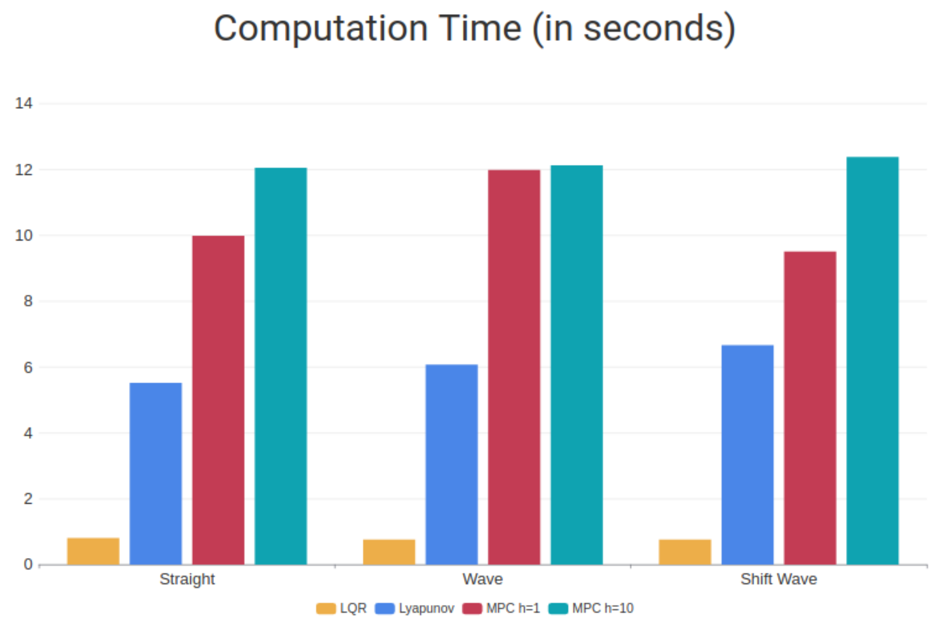

Figure 1.1: Chart of time for each computation

The LQR method, used as a baseline, clearly shows a much faster computation time. It should be noted that despite having a much larger horizon, the MPC with $h = 10$ does not run much slower than the $h = 1$ version. The greedy approach from the Lyapunov Function method also shows a clear advantage for execution time compared to the others.

Figure 1.2: Chart of error for each computation

The noticeable data from this chart shows the advantage of using MPC with a high horizon. The average error is much lower than the low horizon version and the Lyapunov Function method.

It should also be noted how close the average error is between the Lyapunov method and the MPC with $h = 1$ one is. This can be explained by both methods arriving at the same stable local minimum, making their execution very similar.

1.4.2 1 Obstacle

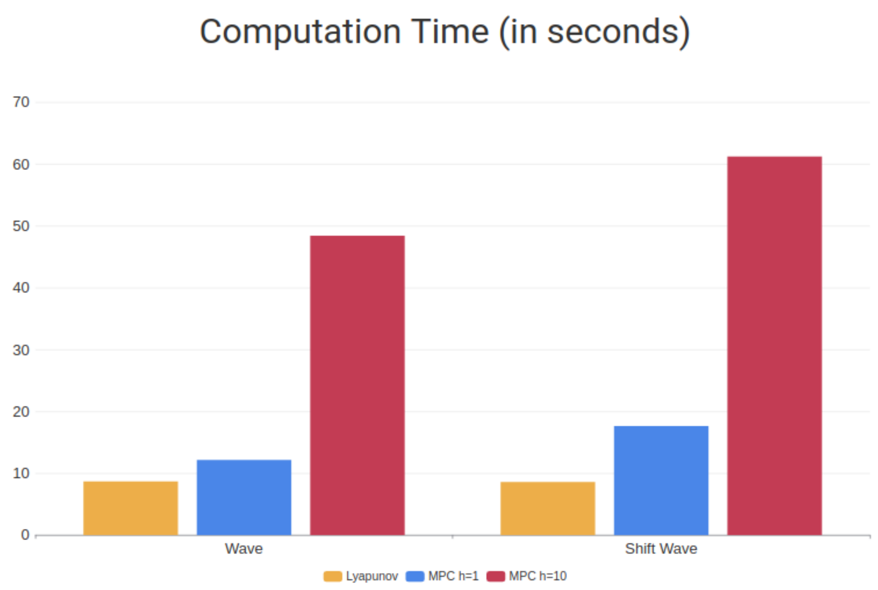

Figure 1.3: Chart of time for each computation

MPC with $h = 10$ already shows a much higher computation time with the added obstacle.

Figure 1.4: Chart of error for each computation

It should be noted that the MPC with $h = 10$ method has a higher error than in the version without obstacles. This is expected in this simulation given that the agents can't arrive at their optimal position if their optimal position is inside an obstacle.

1.4.3 5 Obstacles

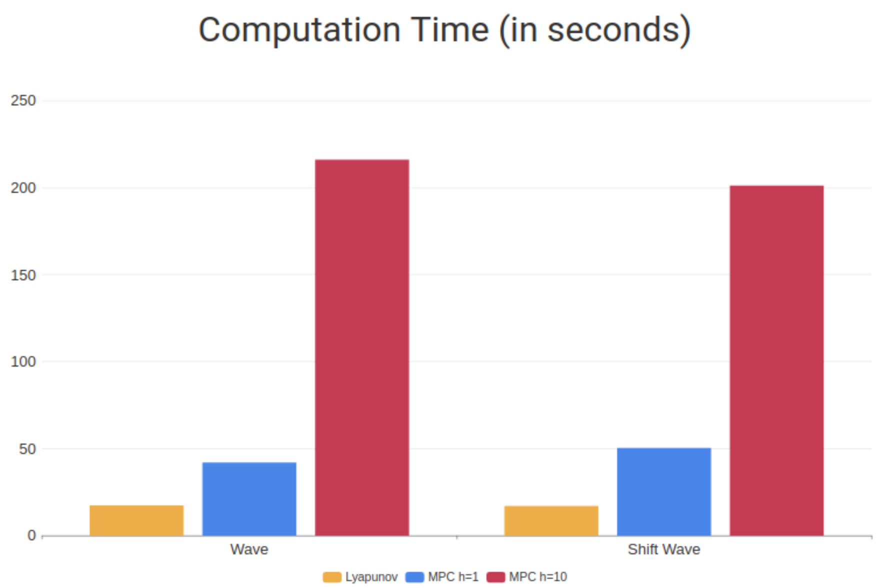

Figure 1.5: Chart of time for each computation

The increase in computation time required by the MPC with $h = 10$ method is noticeable. An increase in the number of obstacles leads to a much more computationally heavy optimization. To compare, it took 48.38s to complete the Wave formation simulation with one obstacle, while the time has increased to 215.99s with 5 obstacles, which is $4.47\times$ the time required before. The increase of time with the Lyapunov method was $2\times$, in comparison.

Figure 1.6: Chart of error for each computation

As before, the high average error from the MPC with $h = 10$ method can be explained by the optimal position being inside an obstacle.

1.5 Findings

In the previous section, we found some noticeable similarities as well as differences between the Lyapunov Function system and the MPC with $h = 1$. Both are prone to stabilize in local minimums instead of achieving a global minimum, which is expected for Lyapunov Functions, given that they have a greedy approach to optimization, unlike MPC with a high horizon which achieves a global minimum error quickly.

However, using MPC with a large horizon also drastically increases computation time, especially with the introduction of obstacles. Care should be taken when applying large horizons to MPC computations, given that they increase accuracy and allow the system to avoid local minimums, however, that may not be worth the increase in computation time.

The obstacle avoidance methods for Lyapunov Functions and MPC are different and as such produce different behaviours on the edge of the safe set of positions. CBFs produce somewhat erratic behaviour when agents are in the edge of the safe set of positions, while the MPC linear constraint method produces a much smoother trajectory, maintaining distance from the edge of the unsafe set.

It should be noted that despite being the one with the highest error on average, the CLF and CBF is consistently faster than the other methods.

It should also be noted that Lyapunov Functions have a higher energy usage on average, especially when the formation requires a sudden movement, expending energy to quickly correct the position.

Bibliography

[1] Z. Wu, F. Albalawi, Z. Zhang, J. Zhang, H. Durand, and P. D. Christofides, “Control lyapunov-barrier function-based model predictive control of nonlinear systems,” *Automatica*, vol. 109, p. 108508, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0005109819303693

[2] J. Zeng, B. Zhang, and K. Sreenath, “Safety-critical model predictive control with discrete-time control barrier function,” in *2021 American Control Conference (ACC)*, 2021, pp. 3882–3889.

[3] U. Rosolia and A. D. Ames, “Multi-rwu2019108508, 9137248, 9137248ate control design leveraging control barrier functions and model predictive control policies,” *IEEE Control Systems Letters*, vol. 5, no. 3, pp. 1007–1012, 2021.

[4] E. Camponogara, D. Jia, B. Krogh, and S. Talukdar, “Distributed model predictive control,” *IEEE Control Systems Magazine*, vol. 22, no. 1, pp. 44–52, 2002.

[5] A. Botelho, B. Parreira, P. N. Rosa, and J. M. Lemos, *Predictive Control for Spacecraft Rendezvous*. Springer, 2021.