MPC-Controlled Quadrotor with Inverted Pendulum

Thomas Stephen Felix
ROBO
University of Pennsylvania
Philadelphia, USA
stfelix@seas.upenn.edu

Khaled Alzamel
ESE
University of Pennsylvania
Philadelphia, USA
kalzamel@seas.upenn.edu

Jeyanth Siddharth Seeka Chitti
MEAM
University of Pennsylvania
Philadelphia, USA
sid99@seas.upenn.edu

***Abstract*—We address and aim to solve the control problem of an inverted pendulum mounted on top of a quadrotor. This classical control problem is nonlinear due to the dynamical nature of the quadrotor coupled with the nonlinear pendulum dynamics model. We have used Model Predictive Control (MPC) to eliminate the error between the current trajectory and the desired trajectory generated by Direct Collocation. The system is subjected to multiple objects in the environment, and one main objective is to avoid obstacles with these objects by finding a collision-free path using RRT*.**

***Index Terms*—MPC, quadrotor, inverted pendulum, direct collocation, RRT*.**

Github Repository

I. Introduction

Development of flight control strategies has become a focus for many researchers in the last decade due to the increasing functionality and usage areas of quadrotors including search and rescue operations, crop monitoring, and surveillance of dangerous areas. Augmenting an inverted pendulum system to a quadrotor becomes more interesting as this can be further extended to other applications such as balancing payloads on drones or bipedal robots.

In this paper, we develop a control strategy to maintain the pendulum's vertical position on top of the quadrotor while hovering and when the quadrotor is following a pre-defined nonlinear curved trajectory. The solution to this control problem suggests that this strategy could be applied to accomplish other tasks.

This control problem involves a nonlinear unstable system because the nonlinear dynamics of the quadrotor coupled with the pendulum dynamics by nature increases the complexity of the problem.

Although earlier work presented quadrotor control strategies for hovering and curved trajectory [1], the environment where the quadrotor was operating was assumed to be obstacle-free. In our case, we have tested our control strategy in an environment where obstacles exist and the quadrotor autonomously chooses an optimal path that avoids collision with these obstacles.

In Section II, we briefly explain the visualization of the quad-rotor and the simulation of the dynamics. Section III demonstrates the use of Direct Collocation to generate a reference trajectory to allow for smooth quadrotor movement and explains how MPC is used to track and eliminate any difference between a measured trajectory and the reference trajectory. We also briefly explain some additional work we did along the directions of implementing CLF-QP for the quadrotor, and LQR for the inverted pendulum to control a decoupled system. Section IV will present how we utilized the RRT* algorithm [4] to avoid obstacles placed in the environment. In Section V, we discuss the results of our experiments. Section VI talks about different future directions that we can take to continue and enhance the project. In Section VII, we conclude and provide a summary of our study.

II. Dynamics & Visualization

A. IPQ (Inverted Pendulum and Quad-rotor) Visualization

The visualization of the combined quadrotor and inverted pendulum system is accomplished using Drake's ***MultibodyPlant*** class. The pendulum is welded 0.03 m above the frame of the quadrotor. At each time step, we calculate the position and orientation of the quadrotor as well as the pendulum. We make the assumption that there is no rotation about the z-axis of the inverted pendulum.

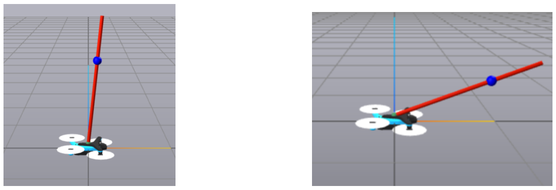
B. Simulation Dynamics

The combined dynamics of the quadrotor and pendulum are modeled to be similar to what is described in [3]. The simulated model can accept as input the body-z force and the three axis moments, or the torque generated by each propeller. They are related by the following equation :

$$\begin{bmatrix} f_z \\ \tau_x \\ \tau_y \\ \tau_z \end{bmatrix} = \begin{bmatrix} K_f & K_f & K_f & K_f \\ 0 & K_f \cdot l & 0 & -K_f \cdot l \\ -K_f \cdot l & 0 & K_f \cdot l & 0 \\ K_m & -K_m & K_m & -K_m \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix}$$

In the above equation, K_f and K_m are the force and moment constants. l is the distance between the propeller and the center of the base.

The dynamics of the system are used to simulate how the movement of the quadrotor will affect the relative position of the inverted pendulum at every timestep. Figure 1 illustrates the system's response to nominal torque inputs and highlights the effect of a slight deviation in the pendulum's initial position.



(a) t = 0 (b) t = 1.2

Fig. 1: Simulated quadrotor with inverted pendulum using nominal torque inputs to maintain hovering position and a slight deviation in pendulum start position

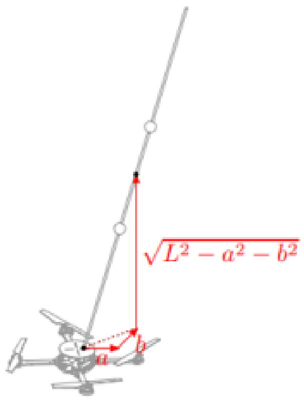


Fig. 2: 3D offset of the pendulum COM in reference to the quadrotor. Image taken from [3]

III. Control

To achieve control, we leverage an MPC framework that works in conjunction with a pre-planned reference trajectory obtained by Direct Collocation. The reference solution, generated offline, provides a dynamically feasible state and input sequence for the system.

We first define the state vector of the combined Inverted Pendulum and Quadrotor system. The quadrotor state consists of the position of the quadrotor $\mathbf{p} = [p_x, p_y, p_z]^T$, the linear velocity of the quadrotor $\mathbf{v} = [v_x, v_y, v_z]^T$, the Euler angles $\mathbf{q} = [\phi, \theta, \psi]^T$ and the angular velocity $\boldsymbol{\omega} = [\omega_x, \omega_y, \omega_z]^T$. The pendulum state is tracked by two variables a and b. The 3D offset of the pendulum COM with respect to the base attached to the quadrotor is given by $[a, b, \zeta]$ and they are related by the equation

$$\zeta^2 = L^2 - a^2 - b^2$$

where L is the distance of the COM on the pendulum from the base.

Finally we define the state x as follows :

$$x = \begin{bmatrix} \mathbf{p} & \mathbf{q} & a & b & \mathbf{v} & \boldsymbol{\omega} & \dot{a} & \dot{b} \end{bmatrix}^T$$

A. MPC

During online execution, the MPC controller continuously solves a finite-horizon optimal control problem at each discrete time step, with the initial condition being the current measured state. The controller's cost function penalizes deviations between the current state and the desired state of the system while also regularizing the control input to avoid actuator saturation. The cost function is given as:

$$\begin{aligned} J = \sum_{k=0}^{N-1} &[(x_k - x_d)^\top Q (x_k - x_d) \\ &+ (u_k - u_d)^\top R (u_k - u_d)] \\ &+ (x_N - x_d)^\top Q_f (x_N - x_d) \end{aligned} \tag{1}$$

The optimization problem can be setup as

$$\min_{x,u} \quad J \tag{2}$$

$$x_0 = \tilde{x} \tag{3}$$

$$x_{k+1} = A x_k + B u_k \tag{4}$$

$$u_{min} \leq u_k \leq u_{max} \tag{5}$$

Here, A and B refer to the discretized dynamics of the non-linear system at (x_d, u_d) at every timestep. This is calculated using ***Sympy*** to find the Jacobians of A and B.

$$A = \left. \frac{\partial f(x,u)}{\partial x} \right|_{x_d, u_d}$$

$$B = \left. \frac{\partial f(x,u)}{\partial u} \right|_{x_d, u_d}$$

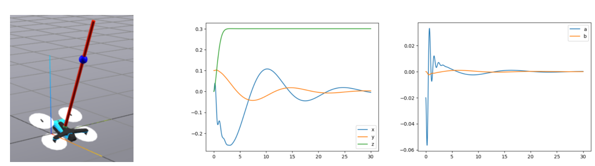
These calculations can be found in the github link attached to this paper if required.

The variables x, u track the error of the current state, input to the required state and input.

$$\tilde{x} = x - x_d \quad \tilde{u} = u - u_d$$

We tune the value of Q, Q_f, N and t where N is the number of receding horizon steps and t refers to discrete time sampling period. We prioritize position of the quadrotor and relative position and velocities of the pendulum by specifying the following Q matrix.

$$Q = diag(100, 100, 40, 20, 20, 20, 40, 40, 1, 1, 1, 1, 1, 1, 100, 100)$$



(a) Non equilibrium start position (b) Position of the quadrotor over time (c) Position of the pendulum over time

Fig. 3: Tracking position of quadrotor and pendulum over time while attaining a position (0, 0, 0.3) and stabilizing the pendulum

As seen in figure 3 when the target state (x_d, u_d) was set to a fixed value, the controller initially required a significant amount of time to converge to the desired state. After tuning, the controller employed a long prediction horizon (N=20, t=0.1) to effectively plan how to utilize the pendulum dynamics to guide the quadrotor toward the target state. The system gradually inched toward the goal position while ensuring the pendulum maintained an upright orientation throughout the process.

B. Direction Collocation

To better use the dynamics of the system to achieve a target state, we use direct collocation to plan the states and corresponding inputs required for a dynamically feasible path between a start and an endpoint. The objective is to enable the optimizer to determine intermediate states that leverage the system's dynamics to reach the desired target state. For example, the pendulum can be slightly tipped forward to initiate movement toward the target configuration, while also ensuring that it can stabilize in an upright position upon reaching the target.

We define the optimization problem as follows :

$$\min_{u} \sum_{i=0}^{N-2} \frac{\Delta t}{2}(u_i^T u_i + u_{i+1}^T u_{i+1}) \tag{6}$$

$$x_0 = x_i \tag{7}$$

$$x_{N-1} = x_f \tag{8}$$

$$u_{N-2,i} = m\frac{g}{4} \; \forall i \in [0,3] \tag{9}$$

$$f(x_c, u_c) = \left.\frac{dx_s(t)}{dt}\right|_{t_c} \tag{10}$$

$$u_{min} \leq u_{i,j} \leq u_{max} \; \forall i \in [0, N-2] \; \forall j \in [0,3] \tag{11}$$

In the optimization framework described above, equations (7) and (8) define the initial state (x_i) and the final target state (x_f) of the system. Equation (9) enforces that the final torque input corresponds to a hovering configuration. Constraint (10) ensures that the dynamics computed at each knot point are consistent with those predicted by the spline formed between x_k and x_{k+1}. Finally, equation (11) imposes torque limits, ensuring the inputs remain within the allowable range for each propeller.

To initialize the optimization, the hovering torque is used as the initial guess for the control input u at each time step. For the system states, we uniformly sample between the start and final configurations as an initial guess.

The number of knot points required generally depends on the distance between the initial and final configurations. There exists a trade-off: increasing the number of knot points improves the optimizer's ability to generate a reliable trajectory that the controller can track but at the cost of increased computation time.

Once we determine the knot points from the optimizer, we create a target state trajectory $x_d(t)$ and target input trajectory $u_d(t)$. These target trajectories are then fed to the MPC controller for trajectory tracking rather than goal tracking. Thus the combined system will try to use the system dynamics to move towards the target configuration.

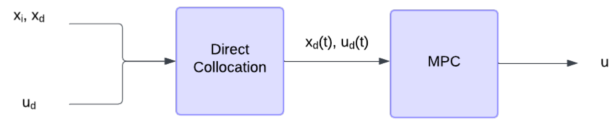
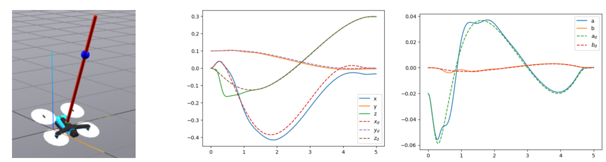


Fig. 4: Direct collection feed a target state and input trajectory for MPC trajectory tracking



(a) Non equilibrium start position (b) Quadrotor position over time (c) Pendulum position over time

Fig. 5: Using MPC to track desired position of quadrotor and pendulum generated thorugh direct collocation over time while attaining a position (0, 0, 0.3) and stabilizing the pendulum

The results shown in Figure 5 are promising. Significant time was spent tuning parameters such as the number of knot points, the weighting matrices for the MPC cost function (Q and R). The quadrotor is able to attain the target positions in relatively shorter interval while maintaining the pendulum in an upright position.

C. Additional Work: CLF-QP + LQR

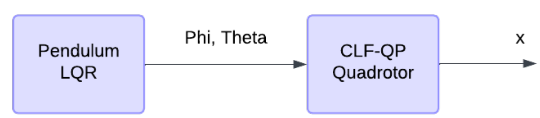


Fig. 6: CLF-QP + LQR

In [3], the paper talks about decoupling the system into control for the quadotor and control of the inverted pendulum. Two methods were discussed for the control of the quadrotor, namely feedback linearization and CLF-QP. Similarly, two methods were discussed for the control of the inverted pendulum, feedback linearization and LQR. Inspired, we set out to implement the system described in figure 6.

We were ultimately unable to replicate the results from the paper. The behavior we observed was akin to the controller not providing enough torque to compensate for the tilt in the pendulum. Further work was not continued in this direction such as implementing control with feedback linearization or tuning the LQR and QP weighting matrices Q and R.

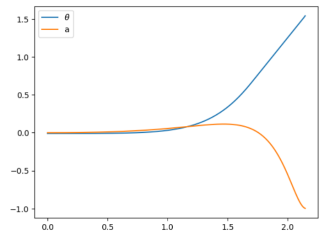


Fig. 7: Quadrotor starting with a slight tilt unable to compensate for deviations in pendulum

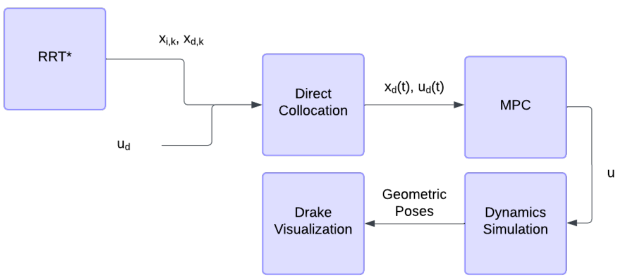
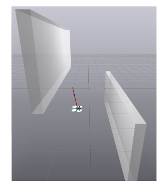
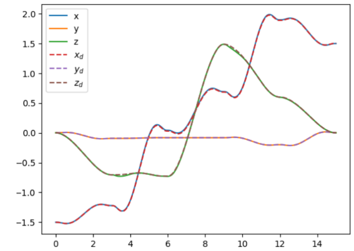


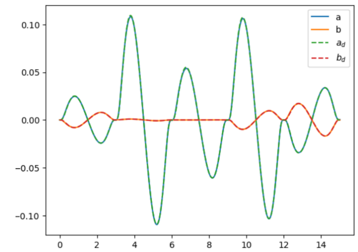
Fig. 8: Overall controller pipeline



(a) Simulation



(b) Position of quadrotor



(c) Relative position of pendulum

Fig. 9: 3D Quadrotor with inverted pendulum attempting obstacle avoidance

IV. Trajectory Planning

In addition to balancing the inverted pendulum, we aimed to design a system capable of moving from a start point to a goal point while avoiding obstacles. For this purpose, we implemented the RRT* algorithm as a global planner.

The RRT* algorithm was applied to a specific point on the quadrotor-pendulum system, corresponding to the mounting point of the pendulum on the quadrotor. However, since the pendulum can swing in any direction during motion, there is a risk that it could collide with obstacles even if the planned path for the mounting point avoids them. To account for this, we added a safety margin by padding the obstacles with a bounding box of 1.3 meters in the upward vertical and horizontal sides. This margin was determined based on the maximum distance between the tip of the pendulum and the closest point on the quadrotor when the pendulum is fully horizontal.

V. Results

By making the assumption that the quadrotor and inverted pendulum system tries to maintain its equilibrium point, we fed the intermediate points from the RRT* planner to direct collocation.

The results as we see in figure 9 were promising. The controller was successful in ensuring that the quadrotor system was able to balance the inverted pendulum while following the trajectory and avoid obstacles when doing so. However in order to track the trajectory well, a long prediction horizon was needed for MPC (N=20, t=0.1) which may not be feasible for real-time control.

VI. Future Works

Some possible avenues for future work include :

- Using non-linear MPC as opposed to discretized MPC. In addition, using specialized real-time MPC solvers to improve performance.
- Have a feedback loop between collocation and RRT to check if the path generated by collocation is indeed collision free.
- Further investigation on CLF-QP, to understand where it lacks and how we can improve on it to get better results.

VII. Conclusion

In this report, we presented a control strategy for an inverted pendulum mounted on a quadrotor operating in a 3D environment with obstacles and path planning. The proposed approach utilizes a pre-planned reference trajectory, generated using Direct Collocation, to ensure smooth system motion and minimize undesired oscillations. The RRT* algorithm was employed to identify collision-free paths by exploring all feasible routes around obstacles. Model Predictive Control (MPC) was then used to reduce deviations between the measured and reference states of the system.

We validated our approach through simulations in the Drake environment, where the quadrotor was able to successfully balance the inverted pendulum from the starting point to the goal point while avoiding obstacles by generating potential collision-free paths. However, solving MPC with a long prediction horizon may not suitable for real-time systems.

Further improvements, including faster real-time MPC trajectory tracking and dynamically feasible collision avoidance, can fully guarantee safe and reliable operation of the flying inverted pendulum system.

References

[1] M. Hehn and R. D'Andrea, "A flying inverted pendulum," 2011 IEEE International Conference on Robotics and Automation, Shanghai, China, 2011.

[2] T. A. Tamba, Y. Y. Nazaruddin and E. Juliastuti, "The Regulation of a Quadrotor UAV Carrying a Pendulum using Receding Horizon Control," 2020 12th International Conference on Information Technology and Electrical Engineering (ICITEE), Yogyakarta, Indonesia, 2020.

[3] X. Shi, Y. K. Nakka, "Nonlinear Controller Design for a Quadrotor with Inverted Pendulum," arXiv preprint, arXiv:2308.02741, 2023.

[4] Karaman S, Frazzoli E. Sampling-based algorithms for optimal motion planning. The International Journal of Robotics Research. 2011;30(7):846-894. doi:10.1177/0278364911406761