Perception-Aware Model Predictive Control

Sumukh Porwal
Worcester Polytechnic Institute
Email: sporwal@wpi.edu

***Abstract*—I formulated a perception-aware model predictive control (MPC) framework for quadrotors, which seamlessly integrates control and planning to achieve both action and perception objectives. This framework employs numerical optimization to generate trajectories that adhere to system dynamics and respect the platform's control input constraints. Concurrently, it optimizes perception objectives by enhancing the visibility of a point of interest and reducing its velocity within the image plane, ensuring robust and reliable sensing.**

Balancing perception and action objectives is inherently challenging due to potential conflicts between their respective requirements. For instance, tracking a reference trajectory necessitates quadrotor rotation to align thrust with the desired acceleration direction, whereas the perception objective may prioritize minimizing such rotations to maximize point-of-interest visibility. To address these challenges, our approach utilizes a model-based optimization framework that incorporates both perception and action objectives while coupling them through system dynamics.

The proposed perception-aware MPC operates in a receding-horizon manner by iteratively solving a nonlinear optimization problem, ensuring dynamic adaptability and optimal performance.

I. Introduction

Advancements in perception algorithms, the availability of affordable cameras, and the enhanced computational capabilities of compact computers have established vision-based perception as the standard for onboard sensing in micro aerial vehicles. Cameras offer several advantages over other sensors, including reduced weight, cost, size, power consumption, and an extensive field of view. Despite these benefits, vision-based perception has notable limitations: it can be inconsistent and its accuracy is highly sensitive to environmental factors (e.g., texture distribution, lighting conditions) and the robot's motion (e.g., motion blur, camera orientation, and distance from the observed scene).

These constraints make it impractical to entirely replace motion-capture systems with onboard vision, as camera motion can degrade estimation quality and impose strict limits on a robot's agility. However, perception can be enhanced through deliberate planning of robot motion that accounts for the constraints and needs of onboard vision. For instance, when navigating a narrow gap while relying on an onboard camera for localization, it is essential to maintain continuous visibility of the gap. Similarly, exploring an unfamiliar environment necessitates orienting the camera toward texture-rich regions.

To fully exploit the agility of autonomous quadrotors, it is imperative to create a synergy between perception and action by treating them as a unified problem.

II. Problem Formulation

The necessity of coupling perception and action becomes evident when considering safety requirements, which rely on accurate and reliable perception. The performance of vision-based perception is highly susceptible to degradation caused by camera motion. On one hand, excessive motion can prevent the extraction of sufficiently accurate information from images, resulting in issues such as motion blur or insufficient texture in the scene. On the other hand, perception quality can significantly improve when its constraints and requirements are explicitly addressed, such as ensuring the visibility of highly textured regions and minimizing motion blur. This calls for a synergistic approach to perception and action.

Let x and u represent the state and input vectors of a robot, respectively, with its dynamics governed by the differential equation $\dot{x} = f(x, u)$. Let z denote the state vector of the perception system (e.g., the projection of 3D points onto the image plane) and σ a vector of parameters characterizing the perception system (e.g., the camera's focal length or field of view). The perception state z is coupled with the robot state x and input u through the system dynamics, expressed as $z = f_p(x, u, \sigma)$.

Action objectives can be quantified using an action cost $L_a(x, u)$, while perception objectives can be represented by a perception cost $L_p(z)$. The joint optimization problem that integrates perception and action can then be formulated as:

$$\min_{u} \int_{t_0}^{t_f} \left[L_a(x, u) + L_p(z)\right] dt, \tag{1}$$

subject to the constraints:

$$r(x, u, z) = 0, \tag{2}$$

$$h(x, u, z) \leq 0, \tag{3}$$

where $r(x, u, z)$ and $h(x, u, z)$ denote equality and inequality constraints, respectively, that must be satisfied for both perception and action. These constraints ensure the feasibility of the solution while addressing the intertwined requirements of perception and action.

III. Environment Setup

In this section, I describe the simulation environment and tools used to model, simulate, and solve the optimal control problem for the quadrotor system. Specifically, I leverage the **CasADi** framework for symbolic computation and the **Acados** solver for efficient real-time optimization.

A. CasADi for Modeling and Symbolic Computation

CasADi is an open-source tool for numerical optimization, symbolic computation, and automatic differentiation, which is particularly suited for control and robotics applications. It provides a flexible interface for creating mathematical models and solving optimization problems. For our quadrotor simulation, CasADi plays a central role in defining the system dynamics, constraints, and cost functions. The following steps outline how CasADi is utilized:

1) System Dynamics Modeling: The dynamics of the quadrotor are defined as a set of ordinary differential equations (ODEs), which govern the evolution of the system's state variables over time. The quadrotor dynamics, as described in Section 2, are implemented symbolically using CasADi. Specifically:

- State vector $\mathbf{x}$: Position $\mathbf{p_{WB}}$, velocity $\mathbf{v_{WB}}$, orientation quaternion $\mathbf{q_{WB}}$, and angular velocity $\boldsymbol{\Omega_B}$.
- Input vector $\mathbf{u}$: Thrust vector $\mathbf{c}$ and body angular velocity inputs $\boldsymbol{\Omega_B}$.

The dynamics are formulated as:

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u})$$

where f is a function that encodes the quadrotor's translational and rotational dynamics. CasADi's symbolic capabilities enable a clean representation of f, including quaternion-based rotations and external forces (e.g., gravity).

2) Cost Function Definition: To solve an optimal control problem (OCP), I need to define a cost function that encodes the desired objectives for the quadrotor. CasADi is used to symbolically represent the cost function, which typically takes the form:

$$J = \int_{t_0}^{t_f} \ell(\mathbf{x}(t), \mathbf{u}(t))\, dt + \phi(\mathbf{x}(t_f))$$

where ℓ is the running cost, penalizing deviations in state and control inputs, and ϕ is the terminal cost. For our application:

- The running cost ℓ penalizes deviations from a desired trajectory and control effort:

$$\ell(\mathbf{x}, \mathbf{u}) = \|\mathbf{x} - \mathbf{x}_{\text{ref}}\|_Q^2 + \|\mathbf{u} - \mathbf{u}_{\text{ref}}\|_R^2,$$

  where Q and R are weight matrices.
- The terminal cost ϕ ensures the quadrotor reaches a target state at the end of the horizon.

CasADi's automatic differentiation capabilities facilitate the computation of gradients and Hessians required for optimization solvers.

3) Constraints Definition: In addition to the dynamics, I impose state and input constraints to ensure realistic behavior:

- State constraints: Bounds on position, velocity, and orientation.
- Input constraints: Limits on thrust and angular velocity inputs.

CasADi enables these constraints to be defined symbolically and incorporated seamlessly into the optimization problem.

B. Acados for Real-Time Optimal Control

Acados is an open-source software package that provides fast and reliable solvers for optimal control problems (OCPs). It is specifically designed for real-time applications and is well-suited for embedded systems, robotics, and aerospace applications. Acados leverages high-performance optimization algorithms, such as Sequential Quadratic Programming (SQP) and Interior Point Methods (IPM), to solve nonlinear OCPs efficiently.

1) Integration with CasADi: Acados integrates tightly with CasADi, enabling the symbolic models (dynamics, costs, and constraints) defined in CasADi to be exported and used as inputs to the Acados solver. This workflow significantly simplifies the transition from problem formulation to real-time optimization. The steps include:

- Symbolic modeling of the system dynamics, cost function, and constraints in CasADi.
- Exporting the model in a format compatible with Acados.
- Configuring the solver settings in Acados, including:
  - Discretization method: I use direct multiple shooting with a time horizon discretized into N intervals.
  - Solver algorithm: SQP-based solver for faster convergence.
  - Tolerances and maximum iterations: To balance accuracy and computational efficiency.
- Solving the optimal control problem using Acados.

2) Simulation Workflow: The simulation workflow combines CasADi and Acados as follows:

- Define the quadrotor's dynamics, cost function, and constraints symbolically using CasADi.
- Discretize the problem over a finite time horizon using multiple shooting.
- Initialize the state and input trajectories.
- Solve the OCP using Acados to compute the optimal control inputs.
- Simulate the system forward using the optimal control inputs and update the state.
- Repeat the process in a receding horizon fashion for Model Predictive Control (MPC).

This framework allows for efficient trajectory planning and control of the quadrotor in both simulation and real-time scenarios.

C. Simulation Results and Visualization

The combined use of CasADi and Acados enables accurate simulation of the quadrotor dynamics while solving for optimal trajectories in real-time. The simulation results are visualized using Python, where:

- The state evolution (position, velocity, orientation) is plotted over time.
- Control inputs (thrust and angular velocities) are visualized to ensure feasibility.
- 3D trajectories of the quadrotor are displayed for qualitative evaluation.

These results confirm the efficacy of the proposed environment setup in achieving accurate and optimal control of the quadrotor system.

IV. Methodology

A. Nomenclature

This study utilizes a world frame $\mathcal{W}$ with an orthonormal basis $\{x_{\mathcal{W}}, y_{\mathcal{W}}, z_{\mathcal{W}}\}$. The quadrotor's body frame $\mathcal{B}$ is also orthonormal, with a basis $\{x_{\mathcal{B}}, y_{\mathcal{B}}, z_{\mathcal{B}}\}$. Additionally, the robot is equipped with a camera whose reference frame $\mathcal{C}$ has a basis $\{x_{\mathcal{C}}, y_{\mathcal{C}}, z_{\mathcal{C}}\}$. Figure 1 illustrates these reference frames.

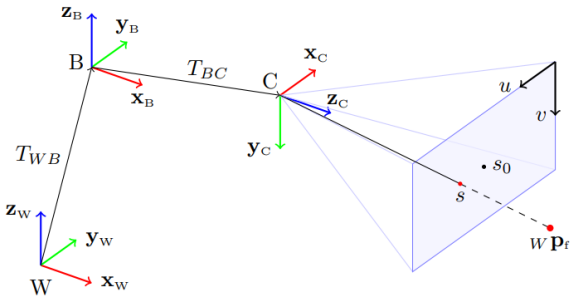

Fig. 1. A schematics representing the world frame W, the body frame B and the camera frame C [1]

Vectors are denoted in bold, with a prefix indicating the frame of reference and a suffix specifying the origin and endpoint. For instance, ${}_{\mathcal{W}}\mathbf{p}_{\mathcal{WB}}$ represents the position of the body frame $\mathcal{B}$ relative to the world frame $\mathcal{W}$, expressed in $\mathcal{W}$. If a vector lacks a prefix, it is assumed to be expressed in the first frame mentioned in the suffix.

Quaternions are used to describe orientations, with a quaternion $\mathbf{q} = (q_w, q_x, q_y, q_z)^\top$ evolving over time as $\dot{\mathbf{q}} = \frac{1}{2}\mathbf{\Lambda}(\boldsymbol{\omega}) \cdot \mathbf{q}$. The skew-symmetric matrix $\mathbf{\Lambda}(\boldsymbol{\omega})$ for a vector $\boldsymbol{\omega} = (\omega_x, \omega_y, \omega_z)^\top$ is defined as:

$$\mathbf{\Lambda}(\boldsymbol{\omega}) = \begin{bmatrix} 0 & -\omega_x & -\omega_y & -\omega_z \\ \omega_x & 0 & \omega_z & -\omega_y \\ \omega_y & -\omega_z & 0 & \omega_x \\ \omega_z & \omega_y & -\omega_x & 0 \end{bmatrix}.$$

Quaternion-vector multiplication is represented by $\otimes$, denoting a rotation of vector $\mathbf{v}$ by the rotation induced by $\mathbf{q}$.

B. Quadrotor Dynamics

The position and orientation of the quadrotor's body frame $\mathcal{B}$ relative to the world frame $\mathcal{W}$ are given by $\mathbf{p}_{\mathcal{WB}} = (p_x, p_y, p_z)^\top$ and $\mathbf{q}_{\mathcal{WB}} = (q_w, q_x, q_y, q_z)^\top$, respectively. Linear velocity is represented by $\mathbf{v}_{\mathcal{WB}} = (v_x, v_y, v_z)^\top$ in $\mathcal{W}$, while angular velocity $\mathbf{\Omega}_{\mathcal{B}} = (\omega_x, \omega_y, \omega_z)^\top$ is expressed in $\mathcal{B}$. The thrust vector $\mathbf{c} = (0, 0, c)^\top$ is mass-normalized, where $c = \frac{f_1+f_2+f_3+f_4}{m}$, f_i denotes the thrust of the i-th motor, and m is the quadrotor mass.

The quadrotor's dynamics are:

$$\dot{\mathbf{p}}_{\mathcal{WB}} = \mathbf{v}_{\mathcal{WB}}, \quad \dot{\mathbf{v}}_{\mathcal{WB}} = \mathbf{g}_{\mathcal{W}} + \mathbf{q}_{\mathcal{WB}} \otimes \mathbf{c}, \quad \dot{\mathbf{q}}_{\mathcal{WB}} = \frac{1}{2}\mathbf{\Lambda}(\mathbf{\Omega}_{\mathcal{B}}) \cdot \mathbf{q}_{\mathcal{WB}}$$

where $\mathbf{g}_{\mathcal{W}} = (0, 0, -g)^\top$ is the gravity vector with $g = 9.81$ m/s^2.

The system's state vector $\mathbf{x}$ and input vector $\mathbf{u}$ are defined as:

$$\mathbf{x} = [\mathbf{p}_{\mathcal{WB}}, \mathbf{v}_{\mathcal{WB}}, \mathbf{q}_{\mathcal{WB}}]^\top, \quad \mathbf{u} = [\mathbf{c}, \mathbf{\Omega}_{\mathcal{B}}^\top]^\top.$$

C. Perception Objectives

Let ${}_{\mathcal{W}}\mathbf{p}_{\{}$ denote the 3D position of a landmark in $\mathcal{W}$. The camera's extrinsic parameters are described by $\mathbf{T}_{\mathcal{BC}} = [\mathbf{p}_{\mathcal{BC}}, \mathbf{q}_{\mathcal{BC}}]$. The landmark's coordinates in the camera frame $\mathcal{C}$ are computed as:

$${}_{\mathcal{C}}\mathbf{p}_f = (\mathbf{q}_{\mathcal{WB}} \otimes \mathbf{q}_{\mathcal{BC}})^{-1} \otimes ({}_{\mathcal{W}}\mathbf{p}_f - (\mathbf{q}_{\mathcal{WB}} \otimes \mathbf{p}_{\mathcal{BC}} + \mathbf{p}_{\mathcal{WB}})).$$

Projection onto the image plane using the pinhole camera model yields:

$$u = \frac{f_x {}_{\mathcal{C}}p_{fx}}{{}_{\mathcal{C}}p_{fz}}, \quad v = \frac{f_y {}_{\mathcal{C}}p_{fy}}{{}_{\mathcal{C}}p_{fz}},$$

where f_x and f_y are the focal lengths.

As previously mentioned, I aim to minimize the velocity of the landmark's projection onto the image plane. Assuming the landmark is static, differentiating the projection equations yields:

$$\dot{\mathbf{s}} = \begin{bmatrix} \dot{u} \\ \dot{v} \\ 0 \end{bmatrix} = \begin{bmatrix} 0 & -f_x/p_{fz}^2 & 0 \\ f_y/p_{fz}^2 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} ({}_{\mathbf{C}}\mathbf{p_f} \times {}_{\mathbf{C}}\dot{\mathbf{p}}_{\mathbf{f}}).$$

The term ${}_{\mathbf{C}}\dot{\mathbf{p}}_{\mathbf{f}}$ is computed as:

$${}_{\mathbf{C}}\dot{\mathbf{p}}_{\mathbf{f}} = -\frac{1}{2}\Lambda(\mathbf{\Omega}_{\boldsymbol{C}}) {}_{\mathbf{C}}\mathbf{p_f} - {}_{\mathbf{C}}\mathbf{v_{WC}},$$

where:

$${}_{\mathbf{C}}\mathbf{v_{WC}} = (\mathbf{q_{WB}}\mathbf{q_{BC}})^{-1} \circledast \left(\frac{1}{2}\Lambda(\mathbf{\Omega}_{\boldsymbol{B}})\mathbf{q_{WB}} \circledast \mathbf{p_{BC}} + \mathbf{v_{WB}}\right),$$

$$\mathbf{\Omega}_{\boldsymbol{C}} = \mathbf{q_{BC}}^{-1} \circledast \mathbf{\Omega}_{\boldsymbol{B}}.$$

D. Action Objectives

To achieve a desired trajectory, the quadrotor must account for two primary objectives:

- The control input vector $\mathbf{u}(t) \in \mathbb{R}^4$ consists of the thrust and angular velocity components. Specifically, the inputs include the collective thrust T and the angular velocities ω_x, ω_y, and ω_z along the three principal axes. These inputs directly influence the quadrotor's translational and rotational motions.
- The input vector $\mathbf{u}(t)$ must remain within the bounds of the actuator capabilities to ensure feasible operation.
- Due to the underactuated nature of the quadrotor, translational and rotational motions must be coupled to navigate effectively in 3D space. The quadrotor's orientation determines the direction of thrust, which in turn controls its position in the environment.

The trajectory planning must respect these constraints while ensuring task completion.

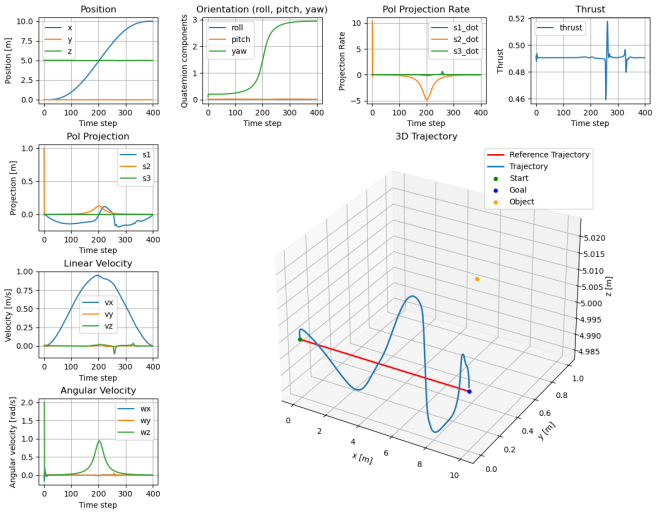

Fig. 2. Perception Aware MPC

V. Results

The results in figure 2 demonstrate the quadrotor's ability to follow the desired reference trajectory while ensuring the point of interest remains centered in the image plane. This is validated through a detailed analysis of the quadrotor states, thrust, and angular velocities.

A. Trajectory Tracking

The **3D Trajectory** plot shows the path traversed by the quadrotor (blue line) compared to the reference trajectory (red line). The quadrotor starts at the designated **Start** position (green dot) and successfully reaches the **Goal** position (blue dot) while keeping the point of interest (orange dot) in view. Despite the complexity of the trajectory, the quadrotor maintains effective control and minimizes deviations.

B. Quadrotor States

- **Position:** The position plots for x, y, and z components indicate smooth progression along the trajectory. The x-position increases over time, while y and z remain relatively constant, reflecting the desired motion.
- **Orientation (Roll, Pitch, Yaw):** The orientation quaternion components evolve consistently over time, showing that the quadrotor adjusts its attitude to stabilize and maintain focus on the point of interest.
- **Linear Velocity:** The velocity components (v_x, v_y, v_z) reflect the translational dynamics. A peak in v_x highlights forward motion, while v_y and v_z remain small, ensuring precise movement along the trajectory.
- **Angular Velocity:** The angular velocities (w_x, w_y, w_z) demonstrate active control, with notable adjustments occurring to keep the point of interest centered.

C. Point of Interest Tracking

The **Point of Interest (Pol) Projection** and **Projection Rate** plots validate the image-plane behavior:

- The projection components (s_1, s_2) remain close to their centered values, ensuring the point of interest stays near the center of the image.
- The projection rates ($\dot{s_1}$, $\dot{s_2}$) reflect dynamic adjustments in the quadrotor's motion, with significant control activity when corrections are needed.

D. Thrust and Control Inputs

The **Thrust** plot indicates that the control input remains within acceptable bounds, ensuring smooth vertical stability. Minimal variations in thrust reflect efficient control. The angular velocity adjustments complement the thrust to achieve the desired attitude and trajectory corrections.

E. Summary

The results clearly show that the quadrotor successfully:

1) Tracks the reference trajectory accurately.
2) Maintains the point of interest at the center of the image plane.
3) Ensures smooth transitions of position, velocity, and control inputs, demonstrating the effectiveness of the implemented control strategy.

The integrated plots highlight the coordinated behavior of position, velocity, orientation, thrust, and angular velocities, validating the robustness and precision of the proposed framework.

VI. Conclusion

In this project, I demonstrated a MPC for a quadrotor tasked with tracking a desired trajectory while ensuring a static point of interest remains centered in the image plane. Using a combination of CasADi and Acados for simulation, the results highlight the effectiveness of the proposed control strategy. The quadrotor successfully achieved accurate trajectory tracking, maintained stable position and orientation, and ensured minimal deviations in thrust and angular velocity throughout the flight.

The accompanying visualizations of position, velocity, orientation, and point of interest projections confirm the system's ability to handle the underactuated dynamics of the quadrotor.

References

[1] D. Falanga, P. Foehn, P. Lu, and D. Scaramuzza, "Pampc: Perception-aware model predictive control for quadrotors," 2018. [Online]. Available: https://arxiv.org/abs/1804.04811