Model Predictive Control for Object Avoidance with UAVs: Literature Review

Kelvin Cui

Aerospace Intelligent Mechatronics Lab

Toronto Metropolitan University

Papers Reviewed:

- Constrained Model-Based Predictive Control For Obstacle Avoidance *(Robotica 2017)*
- Model Predictive Control for Aerial Collision Avoidance in dynamic environments *(MED 2018)*
- Nonlinear MPC for Collision Avoidance and Control of UAVs with Dynamic Obstacles *(R-AL 2020)*
- Nonlinear Model Predictive Control-Based Guidance Algorithm for Quadrotor Trajectory Tracking with Obstacle Avoidance *(JSSC 2021)*

Constrained Model-Based Predictive Control For Obstacle Avoidance (Robotica 2017)

Eric Boivin, Andre Desbiens, Eric Gagnon

- This paper uses a MPC to control a fixed-wing autonomous airplane.
- Obstacle avoidance is achieved through MPC constraints.
  - Obstacles are assumed to be ellipsoidal or fit within ellipsoidal bounds.
    - Shape and location of obstacles are known and fixed.
  - Collision avoidance is achieved with a hard constraint on path intersection.

Constrained Model-Based Predictive Control For Obstacle Avoidance (Robotica 2017)

UAV Modeling (Fixed Wing)

$$\Delta \boldsymbol{X}(k+1) = A\Delta \boldsymbol{X}(k) + B\Delta \boldsymbol{U}(k)$$
$$\Delta \boldsymbol{Y}(k) = C\Delta \boldsymbol{X}(k)$$

$$\boldsymbol{U}(k) = \begin{bmatrix} u_\rho(k) & u_\psi(k) & u_z(k) \end{bmatrix}^T$$

$$\boldsymbol{Y}(k) = \begin{bmatrix} y_\rho(k) & y_\psi(k) & y_z(k) \end{bmatrix}^T$$

$$\boldsymbol{N}(k) = \begin{bmatrix} n_x(k) \\ n_y(k) \\ n_z(k) \end{bmatrix} = \begin{bmatrix} n_x(k-1) + T_s y_\rho(k)\cos[y_\psi(k)] \\ n_y(k-1) + T_s y_\rho(k)\sin[y_\psi(k)] \\ y_z(k) \end{bmatrix}$$

- **X** is the state vector, and **k** is the current iteration
- **U** is the command vector, where **deltaU(k) = U(k) - U(k-1):**
  - **u_rho** is the velocity set point,
  - **u_phi** is the heading set point,
  - **u_z** is the altitude set point,
- The output vector is **Y**:
  - **y_rho** is the velocity in the flat earth horizontal plane,
  - **y_phi** is the heading,
  - **Y_z** is the altitude.
- **N** is the position vector in an xyz frame, with **k** being the current iteration, and **T_s** being the sample time
  - This position can be deduced from the output vector, as shown

Aerospace Intelligent Mechatronics Lab

Toronto Metropolitan University

Constrained Model-Based Predictive Control For Obstacle Avoidance (Robotica 2017)

Scenario Definition

- **P** contains all of the target waypoints, defined by a x,y,z coordinates
- **M** contains all of the x,y,z coordinates of the center of each obstacle
- **W** contains the ellipsoid semi-axis of each obstacle
  - **f_m** (>= 1) is a multiplicative safety factor
  - **f_a** (>= 0) is a additive safety factor
- **delta_j** represents the predicted distance between the vehicle and the jth target
  - **k** is the current time, while **tao** is a future time between 1 and the prediction horizon.

$$\boldsymbol{P} = \begin{bmatrix} \boldsymbol{P}_1 \\ \boldsymbol{P}_2 \\ \vdots \\ \boldsymbol{P}_p \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} p_{x1} & p_{y1} & p_{z1} \end{bmatrix}^T \\ \begin{bmatrix} p_{x2} & p_{y2} & p_{z2} \end{bmatrix}^T \\ \vdots \\ \begin{bmatrix} p_{xp} & p_{yp} & p_{zp} \end{bmatrix}^T \end{bmatrix}$$

$$\boldsymbol{M}(k) = \begin{bmatrix} \boldsymbol{M}_1 & \boldsymbol{M}_2 & \cdots & \boldsymbol{M}_m \end{bmatrix}^T$$

$$\boldsymbol{W}(k) = \begin{bmatrix} \boldsymbol{W}_1 & \boldsymbol{W}_2 & \cdots & \boldsymbol{W}_m \end{bmatrix}^T$$

$$\boldsymbol{W}_i = f_m \begin{bmatrix} w_{xi} & 0 & 0 \\ 0 & w_{yi} & 0 \\ 0 & 0 & w_{zi} \end{bmatrix} + f_a$$

$$\delta_j(k+\tau/k) = \left| \hat{\boldsymbol{N}}(k+\tau/k) - \boldsymbol{P}_j \right| = \sqrt{\left[\hat{\boldsymbol{N}}(k+\tau/k) - \boldsymbol{P}_j\right]^T \left[\hat{\boldsymbol{N}}(k+\tau/k) - \boldsymbol{P}_j\right]}$$

Aerospace Intelligent Mechatronics Lab

Toronto Metropolitan University

Constrained Model-Based Predictive Control For Obstacle Avoidance (Robotica 2017)

MPC Formation : Cost & Non-Linear Inequality Constraints

$$\Phi = [\Delta\boldsymbol{U}(k : k + h_c - 1)]^T \boldsymbol{\lambda} [\Delta\boldsymbol{U}(k : k + h_c - 1)] + \sum_{\tau=1}^{h_p} \delta_j(k + \tau/k)$$

$$\Delta\boldsymbol{U}_{min}(k : k + h_c - 1) \leq \Delta\boldsymbol{U}(k : k + h_c - 1) \leq \Delta\boldsymbol{U}_{max}(k : k + h_c - 1)$$

$$\boldsymbol{U}_{min}(k : k + h_c - 1) \leq \boldsymbol{U}(k : k + h_c - 1) \leq \boldsymbol{U}_{max}(k : k + h_c - 1)$$

$$\boldsymbol{N}_{min}(k + 1 : k + h_p) \leq \hat{\boldsymbol{N}}(k + 1 : k + h_p/k) \leq \boldsymbol{N}_{max}(k + 1 : k + h_p)$$

$$0 \leq \left| \boldsymbol{N}_R(k) - \hat{\boldsymbol{N}}(k + \tau/k) \right| \leq h_s \quad \text{for } \tau \in [1, h_p]$$

$$\boldsymbol{y}_{s-min}(k + 1 : k + h_p) \leq \hat{\boldsymbol{y}}_s(k + 1 : k + h_p/k) \leq \boldsymbol{y}_{s-max}(k + 1 : k + h_p)$$

- The cost function **Phi** minimizes both the distance to the jth target, while weighing the command input
  - Command inputs and rates are constrained for safe operation of autopilot with **U_min**, **deltaU_min, U_max** and **deltaU_max**
- The predicted position **N** is constrained over the prediction horizon
  - Also constrained over the sensor horizon relative to the current measured position
- The predicted velocity **y_s** is bounded over the prediction horizon as well, defined by:

$$\hat{y}_s(k + \tau/k) = \sqrt{(\hat{y}_\rho(k + \tau/k))^2 + \left(\frac{\hat{y}_z(k+\tau/k) - \hat{y}_z(k+\tau-1/k)}{T_s}\right)^2}$$

Aerospace Intelligent Mechatronics Lab

Toronto Metropolitan University

Constrained Model-Based Predictive Control For Obstacle Avoidance (Robotica 2017)

MPC Formation : Obstacle Avoidance

- The point **L(k + tao/k)** exists on the line between **N(k + tao - 1/k)** and **N(k+tao/k)** if 0 <= **n(k+tao/k)** <= 1.
- The surface of the ellipsoid of the obstacle is defined by the equation:

$$\left[\boldsymbol{E}_i - \boldsymbol{M}_i\right]^T \boldsymbol{W}_i^{-2} \left[\boldsymbol{E}_i - \boldsymbol{M}_i\right] = 1$$

- If **E_i** = **L(k+tao/k)**, then there is an intersection between the path and the obstacle.
    - This can be calculated with **n_1** and **n_2**.
        - If both are complex, then the line does not intersect. Otherwise, the line intersects the surface of the ellipsoid.
    - A binary variable is used to represent intersections, and is constrained to zero.

$$\begin{aligned}\boldsymbol{L}(k+\tau/k) =& \hat{\mathbf{N}}(k+\tau-1/k) \\ &+ \eta(k+\tau/k)\begin{bmatrix} \hat{\mathbf{N}}(k+\tau/k) \\ -\hat{\mathbf{N}}(k+\tau-1/k) \end{bmatrix}\end{aligned}$$

$$\eta_{1i}(k+\tau/k) = \frac{-b_i(k+\tau/k) + \sqrt{\varphi(k+\tau/k)}}{2a_i(k+\tau/k)}$$

$$\eta_{2i}(k+\tau/k) = \frac{-b_i(k+\tau/k) - \sqrt{\varphi(k+\tau/k)}}{2a_i(k+\tau/k)}$$

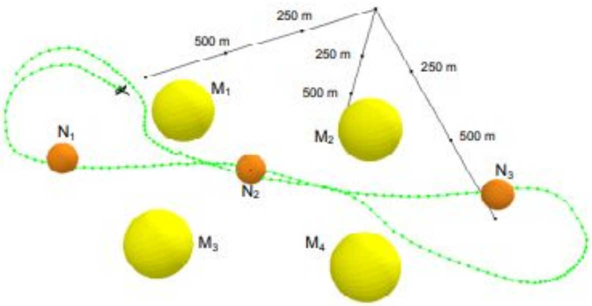
$$\begin{aligned}\varphi(k+\tau/k) &= b_i^2(k+\tau/k) \\ &\quad - 4a_i(k+\tau/k)c_i(k+\tau/k) \\ \hat{\mathcal{M}} &= \hat{\mathbf{N}}(k+\tau-1/k) - \boldsymbol{M}_i \\ a_i(k+\tau/k) &= \hat{\mathcal{N}}^T \boldsymbol{W}_i^{-2} \hat{\mathcal{N}} \\ b_i(k+\tau/k) &= 2\hat{\mathcal{N}}^T \boldsymbol{W}_i^{-2} \hat{\mathcal{M}} \\ c_i(k+\tau/k) &= \hat{\mathcal{M}}^T \boldsymbol{W}_i^{-2} \hat{\mathcal{M}} - 1\end{aligned}$$

Aerospace Intelligent Mechatronics Lab

Toronto Metropolitan University

Constrained Model-Based Predictive Control For Obstacle Avoidance (Robotica 2017)

Implementation and Results



- MPC was implemented on an off-the-shelf UAV autopilot
  - Ground control station constraints guidance algorithms
  - MPC is implemented in C++ using NAG package, allowing for real time computation.
- A hardware-in-the-loop test was first conducted to minimize risk & complexity of outdoor flights
  - XPlane is used as a 3D viewer over UDP
- Real flights were then conducted, with simulated obstacles
  - Obstacle detection is only valid within simulated sensor range
  - *Plane was able to meet all targets and avoid all obstacles.*

Model Predictive Control for Aerial Collision Avoidance in dynamic environments (Journal: MED 2018)

Manuel Castillo-Lopez, Seyed Amin Sajadi-Alamdari, Jose Luis Sanchez-Lopez, Miguel A. Olivares-Mendez, Holger Voos

- A Non-linear MPC is used in order to generate a control command for a Drone UAV.
- Obstacle avoidance is achieved through soft constraints.
  - Increased distance to obstacles is achieved by propagating obstacle position using a constant velocity model throughout prediction horizon.

https://orbilu.uni.lu/bitstream/10993/37031/1/paper_med2018_final.pdf

Model Predictive Control for Aerial Collision Avoidance in dynamic environments (Journal: MED 2018)

UAV Modeling (Quadcopter)

$$\dot{x} = v_x cos(\psi) - v_y sin(\psi) \tag{2a}$$
$$\dot{y} = v_x sin(\psi) + v_y cos(\psi) \tag{2b}$$
$$\dot{z} = v_z \tag{2c}$$
$$\dot{\psi} = v_\psi \tag{2d}$$
$$\dot{v}_i = (-v_i + k_i u_i)/\tau_i, \quad i \in \{x, y, z, \psi\} \tag{2e}$$

- Drone state: $\mathbf{x} = [x \; y \; z \; \psi \; v_x \; v_y \; v_z \; v_\psi]^T$
  - **Phi** represents the yaw angle, while xyz are ENU frame
  - **U** is the control input: $\mathbf{u} = [u_x \; u_y \; u_z \; u_\psi]^T$
- A Non-linear motion model is defined with **2.**
  - Model parameters **k_i** and **t_i** are chosen using the classical step response method:

$$k_i = \frac{v_i(\infty)}{u_i(\infty)} \qquad \tau_i = \frac{3}{2}(t_{63} - t_{28})$$

Aerospace Intelligent Mechatronics Lab

Toronto Metropolitan University

Model Predictive Control for Aerial Collision Avoidance in dynamic environments (Journal: MED 2018)

MPC Trajectory Tracking

- Two cost terms are proposed:
  - **4** is used for trajectory tracking
    - Where **x*** is the reference state, and
      N is the prediction horizon
  - **5** aims for stability and energy efficiency

$$J^t = \frac{1}{2}\sum_{i=0}^{N-1} \|\mathbf{x}_i - \mathbf{x}_i^*\|_P^2 + \|\mathbf{x}_N - \mathbf{x}_N^*\|_Q^2 \qquad (4)$$

$$J^c = \frac{1}{2}\sum_{i=0}^{N-1} \|\mathbf{u}_i\|_R^2 \qquad (5)$$

Model Predictive Control for Aerial Collision Avoidance in dynamic environments (Journal: MED 2018)

MPC Collision Avoidance

- Soft constraints are used to guarantee feasible solutions in tight situations
  - **(6)** is used to avoid ellipsoidal obstacles
    - Ellipsoidal term is defined by:
      - $\xi(\mathbf{r}_i, \mathbf{r}_i^o) = \sqrt{(\mathbf{r}_i - \mathbf{r}_i^o)^T Q_i (\mathbf{r}_i - \mathbf{r}_i^o)}$
    - **Q_i** is a metric from ellipsoidal dimensions, where **R_w** is the world frame transform.
      - $Q_i = {}^O R_W^T \; M_i \; {}^O R_W \qquad M = diag(r_x^{-2}, r_y^{-2}, r_z^{-2})$
    - **S_i** is a slack term that relaxes the constraint with a sensitivity **theta^epsilon**, and is minimized by (**10).**
    - Obstacles are assumed to have constant velocity such that:
      - $\mathbf{r}_i^o = \mathbf{r}_{i-1}^o + \dot{\mathbf{r}}_{i-1}^o \cdot \Delta t$
  - **(11)** and **(12)** are used in a similar way to avoid walls and floor

$$\xi^2(\mathbf{r}_i, \mathbf{r}_i^o) + \theta^\xi s_i^\xi \geq 1 \tag{6}$$

$$J^\xi = \frac{1}{2} \sum_{i=0}^{N-1} \|\mathbf{s}_i^\xi\|_S^2 \tag{10}$$

$$\pi(\mathbf{r}_i) + \theta^\pi s_i^\pi \geq 0 \tag{11}$$

$$J^\pi = \frac{1}{2} \sum_{i=0}^{N-1} \|\mathbf{s}_i^\pi\|_T^2 \tag{12}$$

Aerospace Intelligent Mechatronics Lab

Toronto Metropolitan University

Model Predictive Control for Aerial Collision Avoidance in dynamic environments (Journal: MED 2018)

MPC Optimal Control Problem & Implementation

$$
\begin{aligned}
\underset{X,U}{\text{minimize}} \quad & J = J^t + J^c + J^\xi + J^\pi \\
\text{subject to:} \quad & \mathbf{x}_0 = \bar{x}_0 \\
& \mathbf{x}_{i+1} = \Phi_i(\mathbf{x}_i, \mathbf{u}_i) && i = 0, \ldots, N-1 \\
& \xi(\mathbf{x}_i) + \theta^\xi s_i^\xi \geq 1 && i = 0, \ldots, N-1 \\
& \pi(\mathbf{x}_i) + \theta^\pi s_i^\pi \geq 0 && i = 0, \ldots, N-1 \\
& |\mathbf{u}_i| \leq \mathbf{u}_{max} && i = 0, \ldots, N-1
\end{aligned}
$$

- Combining previous equations, final form is a discrete nonlinear problem.
  - **u_max** introduces control limit constraints
- NMPC is implemented in C++ using the ACADO Toolkit.
  - Drone runs ROS, using V-REP as a SIL environment
- The implementation parameters are as follows:

Prediction horizon	4s
Discretization steps	20
Integrator type	Runge-Kutta 4
Maximum controls	1 m/s
Control Rate	20 Hz
Ellipsoidal obstacles sensitivity	0.15
Planar constraints sensitivity	0.25

Model Predictive Control for Aerial Collision Avoidance in dynamic environments (Journal: MED 2018)

Results - Street Crossing Scenario

- In a street crossing scenario, pedestrians are modeled with as a cylinder with infinite height, moving at a constant velocity.
- Two approaches were used:
    - The static approach samples the pedestrian position and assumes constant position throughout the prediction horizon
    - The dynamic approach propagates the pedestrian position with the constant velocity model.
- Dynamic approach was able to generate optimal paths that doesn't cross the future trajectory.

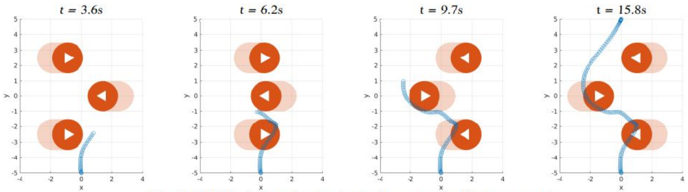


Fig. 7: UAV trajectory in street crossing scenario. Static approach.

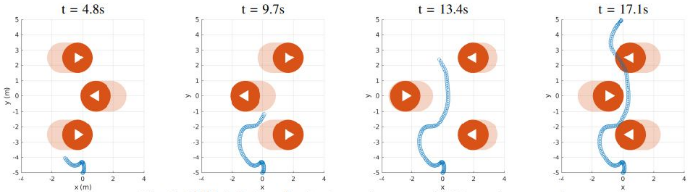
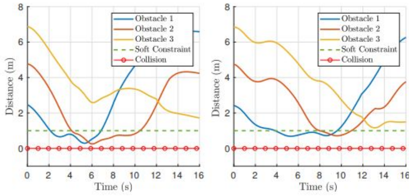


Fig. 8: UAV trajectory in street crossing scenario. Dynamic approach



(a) Static approach (b) Dynamic approach

Fig. 4: Distance to obstacles in street crossing scenario

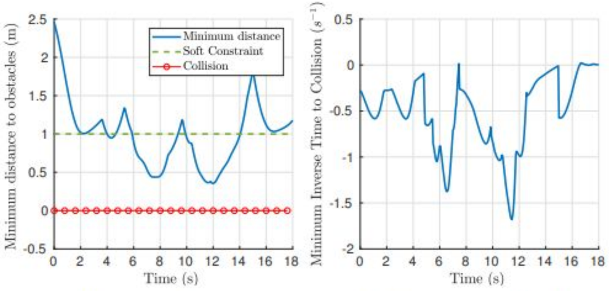
Aerospace Intelligent Mechatronics Lab

Toronto Metropolitan University

Model Predictive Control for Aerial Collision Avoidance in dynamic environments (Journal: MED 2018)

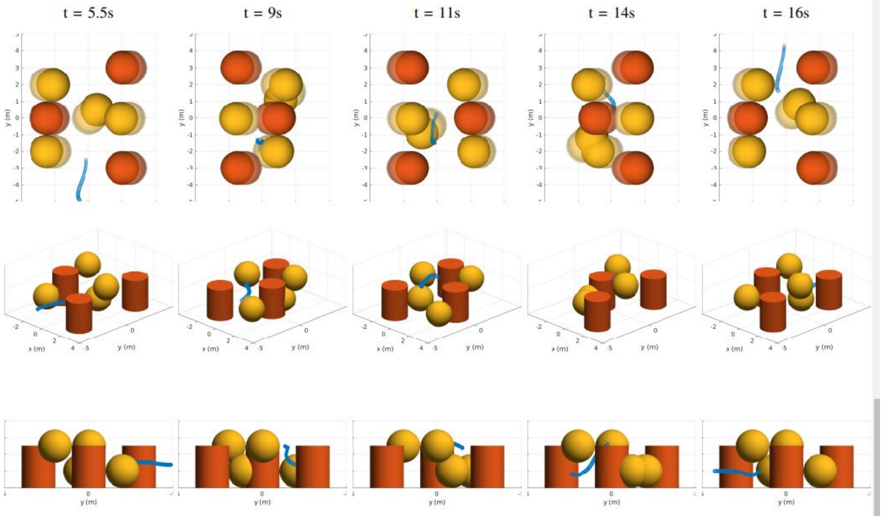
Results - Multiple Obstacle Scenario

- In the multiple object scenario, aerial robots are added in addition to the pedestrians.
- The UAV is able to avoid multiple collisions, despite soft constraint violation.



(a) Minimum Distance (b) Minimum TTC^{-1}

Fig. 9: Risk variables in the multiple obstacle scenario



Aerospace Intelligent Mechatronics Lab

Toronto Metropolitan University

Nonlinear MPC for Collision Avoidance and Control of UAVs with Dynamic Obstacles (R-AL 2020)

Bjorn Lindqvist , Sina Sharif Mansouri, Ali-akbar Agha-Mohammadi and George Nikolakopoulos

- Obstacle avoidance achieved through two novel methods:
    - Obstacle trajectory classification for more accurate trajectory prediction
        - Three classes for obstacles are proposed (static, linear, projectile)
        - Classification is done on-line to quickly and accurately predict different obstacle trajectories.
    - Increasing radius of safety sphere throughout prediction horizon
        - Accounts for sensor and prediction error

Nonlinear MPC for Collision Avoidance and Control of UAVs with Dynamic Obstacles (R-AL 2020)

UAV Modeling (Quadcopter)

$$\dot{p}(t) = v(t)$$

$$\dot{v}(t) = R(\phi, \theta)\begin{bmatrix} 0 \\ 0 \\ T \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} - \begin{bmatrix} A_x & 0 & 0 \\ 0 & A_y & 0 \\ 0 & 0 & A_z \end{bmatrix} v(t)$$

$$\dot{\phi}(t) = {}^1/_{\tau_\phi}(K_\phi \phi_{\text{ref}}(t) - \phi(t)),$$

$$\dot{\theta}(t) = {}^1/_{\tau_\theta}(K_\theta \theta_{\text{ref}}(t) - \theta(t)),$$

- **p(t)** represents the position in xyz coordinates, with **v(t)** representing the linear velocity, viewed in a global fixed frame
- **R(phi, theta)** is a rotation matrix that describes the attitude in Euler form.
  - **Phi** is the roll angle along the X axis, while **Theta** is the pitch angle along the Y axis.
  - **T** is the total thrust
  - **A_x, A_y** and **A_z** are linear damping terms.
- **Phi_ref** and **Theta_ref** are the reference roll and pitch respectively.
  - The **K** and **Tao** values are gains and time constants respectively for the roll and pitch.

Nonlinear MPC for Collision Avoidance and Control of UAVs with Dynamic Obstacles (R-AL 2020)

Cost Function

- The state vector is: $x = [p, v, \phi, \theta]^\top$
- With input controls as: $u = [T, \phi_{\text{ref}}, \theta_{\text{ref}}]^\top$
- The forward euler method is used to predict using discrete time steps, allowing for the following equation to be used as the cost function:

$$J(\boldsymbol{x}_k, \boldsymbol{u}_k, u_{k-1|k}) = \sum_{j=0}^{N} \underbrace{\|x_{\text{ref}} - x_{k+j|k}\|^2_{Q_x}}_{\text{State cost}} + \underbrace{\|u_{\text{ref}} - u_{k+j|k}\|^2_{Q_u}}_{\text{Input cost}} + \underbrace{\|u_{k+j|k} - u_{k+j-1|k}\|^2_{Q_{\Delta u}}}_{\text{Input smoothness cost}},$$

- where **Q_x, Q_u** and **Q_deltau** are positive definite weight matrices.
- The reference input is kept at steady-state hovering to minimize input commands:

$$u_{\text{ref}} = [g, 0, 0]$$

Nonlinear MPC for Collision Avoidance and Control of UAVs with Dynamic Obstacles (R-AL 2020)

Obstacle Avoidance Constraints

$$h_{\text{sphere}}(p, \xi^{\text{obs}}) = [(r^{\text{obs}} + r_{\text{s}})^2 - (p_x - p_x^{\text{obs}})^2 - (p_y - p_y^{\text{obs}})^2 - (p_z - p_z^{\text{obs}})^2]_+ = 0, \quad (3)$$

$$\boldsymbol{h}_{\text{sphere}}(\boldsymbol{p}_k, \boldsymbol{\xi}^{\text{obs}}) = \bar{0}, \quad (4)$$

$$[\phi_{\text{ref},k+j-1|k} - \phi_{\text{ref},k+j|k} - \Delta\phi_{\max}]_+ = 0, \quad (5a)$$

$$[\phi_{\text{ref},k+j|k} - \phi_{\text{ref},k+j-1|k} - \Delta\phi_{\max}]_+ = 0. \quad (5b)$$

$$u_{\min} \leq u_{k+j|k} \leq u_{\max}. \quad (6)$$

- In order to follow the constraint formulation structure, the following function is defined: $[h]_+ = \max\{0, h\}$.
- **(3)** ensures that the drone position is outside of the position of the obstacle
    - **r_s** is a safety radius that linearly increases along the horizon **N**, accounting for measurement/prediction errors
    - **r^obs** and **p^obs** are the radius and centers of the obstacles respectively.
- **(4)** extends 3 to allow for **N** positions that describe obstacle trajectory at time k.
- **(5)** constrains the control delta for roll, and is extended for theta as well (pitch)
- **(6)** places hard bounds on possible control inputs that the low-level controller can handle.

Nonlinear MPC for Collision Avoidance and Control of UAVs with Dynamic Obstacles (R-AL 2020)

Obstacle Trajectory Classification

- Obstacle trajectory is divided into three classes:
  - Static, without any movement $(\dot{p}^{obs} = 0)$.
  - Linear motion, where $\dot{p}^{\text{obs}}(t) = v^{obs}(t)$,
  - Projectile motion, represented by **(10)**.
    - **B** represents linear aerodynamic damping terms
    - Bouncing is simulated by applying a coefficient of restitution on the velocities to simulate energy loss
- Object motion is then discretized with forward euler, resulting in **(11)**.
- Classification is achieved by comparing the **M** last measured positions to the backwards prediction.
  - **(12)** is used to generate an error using each of the three classes
  - Lowest error class is chosen for future prediction.

$$\dot{p}^{\text{obs}}(t) = v^{obs}(t), \tag{10a}$$

$$\dot{v}^{\text{obs}}(t) = \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} - \begin{bmatrix} B_x & 0 & 0 \\ 0 & B_y & 0 \\ 0 & 0 & B_z \end{bmatrix} v^{obs}(t), \tag{10b}$$

$$\begin{aligned} x^{obs}_{k|k+n+1} &= \alpha_t(x^{obs}_{k|k+n}), \\ x^{obs}_{k|k+n-1} &= \beta_t(x^{obs}_{k|k+n}). \end{aligned} \tag{11}$$

$$e^{traj} = \sum_{j=1}^{M} | \, p_j^{prev} - p^{obs}_{k|k-j} \, | + | \, v_j^{prev} - v^{obs}_{k|k-j} \, | \, . \tag{12}$$

Aerospace Intelligent Mechatronics Lab

Toronto Metropolitan University

Nonlinear MPC for Collision Avoidance and Control of UAVs with Dynamic Obstacles (R-AL 2020)

MPC Formulation and Implementation

$$
\begin{aligned}
&\underset{\boldsymbol{u}_k, \boldsymbol{x}_k}{\text{Minimize}}\ J(\boldsymbol{x}_k, \boldsymbol{u}_k, u_{k-1|k}) && \text{(8a)}\\
&\text{s. t.: } x_{k+j+1|k} = \zeta(x_{k+j|k}, u_{k+j|k}),\\
&\quad j = 0, \ldots, N-1, && \text{(8b)}\\
&\quad u_{\min} \leq u_{k+j|k} \leq u_{\max},\ j = 0, \ldots, N, && \text{(8c)}\\
&\quad h^i_{\text{sphere}}(p_{k+j|k}, \xi_j^{\text{obs},i}) = 0,\ j = 0, \ldots, N, && \text{(8d)}\\
&\quad i = 1, \ldots, N_s && \text{(8e)}\\
&\quad \text{Constraints (5)}, j = 0,\ \ldots, N. && \text{(8f)}
\end{aligned}
$$

- The NMPC problem can be formulated with **(8)** assuming **N_s** spherical obstacles
  - Solved using the PANOC algorithm, leveraging the OpEn framework.
- Vicon Motion capture is used to track UAV and obstacle.
- Computation is completed on a remote laptop running ROS.

Nonlinear MPC for Collision Avoidance and Control of UAVs with Dynamic Obstacles (R-AL 2020)

Results

- Dynamic NMPC model is able to avoid thrown objects by predicting future trajectory.
  - Dynamic model is even able to avoid bouncing obstacle.
- Traditional methods are too slow to detect when objects enter collision sphere, resulting in collision.
- Can also accurately track and avoid multiple objects.
- Video demo: **https://youtu.be/vO3xjvMMNJ4**

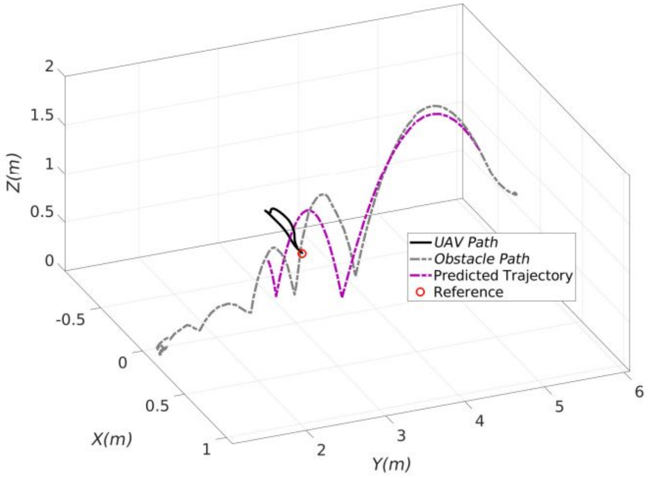


Fig. 6: Path of UAV and dynamic obstacle during experiment with bouncing ball.

Nonlinear Model Predictive Control-Based Guidance Algorithm for Quadrotor Trajectory Tracking with Obstacle Avoidance (JSSC 2021)

Zhao ChunHui, Wang Dong, Hu JinWen, Pan Quan

- Trajectory Tracking performance is enhanced by adding stability constraints
  - Non-linear lyapunov stability constraints are added to increase tracking performance with disturbances compared to Lyapunov-based backstepping control.
- Obstacle avoidance is achieved with a simple switching cost function
  - Cost function activates when the drone is within a defined radius of a known, static target.

Nonlinear Model Predictive Control-Based Guidance Algorithm for Quadrotor Trajectory Tracking with Obstacle Avoidance (JSSC 2021)

UAV Modeling (Quadcopter)

- The position and yaw vector is represented by $\boldsymbol{\eta} = [x, y, z, \psi]^{\mathrm{T}}$
  - Where **xyz** are ENU frame, while **phi** represents the rotation about the **z** axis.
- The kinematics are then represented by: $\dot{\boldsymbol{\eta}} = \boldsymbol{R}(\psi)\boldsymbol{v}_b,$
  - Where

$$\boldsymbol{R}(\psi) = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 & 0 \\ \sin(\psi) & \cos(\psi) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- **v_b** is the body frame velocity.
  - The velocity response and reference velocity commands can be approximated by $\dot{\boldsymbol{v}}_b = \boldsymbol{S}\boldsymbol{v}_b + \boldsymbol{F}\boldsymbol{u},$
  - $\boldsymbol{u} = [u_x, u_y, u_z, u_\psi]^{\mathrm{T}}$ is the control input, while **S** and **F** are parameters based on the gains and time constants from their corresponding first-order transfer functions.
- As a result, the dynamic model is defined as:

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \boldsymbol{R}(\psi)\boldsymbol{v}_b \\ \boldsymbol{S}\boldsymbol{v}_b + \boldsymbol{F}\boldsymbol{u} \end{bmatrix} \triangleq \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}),$$

Nonlinear Model Predictive Control-Based Guidance Algorithm for Quadrotor Trajectory Tracking with Obstacle Avoidance (JSSC 2021)

NMPC Tracking

- The UAV is driven to follow a reference path $\mathcal{S}(\theta) = [x_{rr}(\theta), y_{rr}(\theta), z_{rr}(\theta)]^{\mathrm{T}}$
    - This is used to generate a time-parameterized reference trajectory: $\boldsymbol{\eta}_r(t) = [x_r(t), y_r(t), z_r(t), \psi_r(t)]^{\mathrm{T}}$
    - Theta is time-dependant, following the predetermined timing law: $\theta(t) = v_t t,$
        - **v_t** > 0 to denote the forward velocity of the drone.
    - Singularities in the reference trajectory are avoided by assuming that the reference trajectory and its derivatives are bounded.
- NMPC is designed so that the real position of the drone converges to the reference trajectory: $\lim_{t\to\infty} ||\boldsymbol{\eta}(t) - \boldsymbol{\eta}_r(t)|| = 0;$

Nonlinear Model Predictive Control-Based Guidance Algorithm for Quadrotor Trajectory Tracking with Obstacle Avoidance (JSSC 2021)

NMPC Obstacle Avoidance

- Obstacles are defined with a position **p_o** and a safety radius **r_s**.
- Switching behavior between trajectory tracking and obstacle avoidance is achieved through

$$J^{oa}(\boldsymbol{x}(t)) = \begin{cases} \lambda, & ||\boldsymbol{p}(t) - \boldsymbol{p}_o|| < r_s \quad \text{(obstacle avoidance)}, \\ 0, & ||\boldsymbol{p}(t) - \boldsymbol{p}_o|| \geq r_s \quad \text{(trajectory tracking)}, \end{cases} \tag{5}$$

  - However, due to the lack of continuous differentiability, **J^oa** is defined instead by

$$J^{oa}(\boldsymbol{x}(t)) \approx \rho(t) = \frac{\lambda}{1 + \mathrm{e}^{-k \cdot d(t)}},$$

    - Where $d(t) = r_s^2 - (\boldsymbol{p}(t) - \boldsymbol{p}_o)^{\mathrm{T}}(\boldsymbol{p}(t) - \boldsymbol{p}_o)$

Nonlinear Model Predictive Control-Based Guidance Algorithm for Quadrotor Trajectory Tracking with Obstacle Avoidance (JSSC 2021)

NMPC Formation

$$
\begin{aligned}
\mathcal{P}_0(\boldsymbol{x}) : \min_{\widehat{\boldsymbol{u}}(\cdot)} J = \int_{t_k}^{t_k+T} & ||\widetilde{\boldsymbol{x}}(s;t_k)||_{\boldsymbol{Q}}^2 + J^{oa}(\widehat{\boldsymbol{x}}(s;t_k)) + ||\widehat{\boldsymbol{u}}(s;t_k)||_{\boldsymbol{R}}^2 \, ds \\
\text{s.t.} \quad & \dot{\widehat{\boldsymbol{x}}}(s;t_k) = \boldsymbol{f}(\widehat{\boldsymbol{x}}(s;t_k), \widehat{\boldsymbol{u}}(s;t_k)), \quad s \in [t_k, t_k+T], \\
& \widehat{\boldsymbol{x}}(t_k;t_k) = \boldsymbol{x}(t_k), \\
& ||\widehat{\boldsymbol{u}}(s;t_k)||_\infty \leq u_{\max}, \quad s \in [t_k, t_k+T], \\
& \frac{\partial V}{\partial \boldsymbol{x}} \boldsymbol{f}(\widehat{\boldsymbol{x}}(t_k;t_k), \widehat{\boldsymbol{u}}(t_k;\widehat{\boldsymbol{x}}(t_k;t_k))) \leq \frac{\partial V}{\partial \boldsymbol{x}} \boldsymbol{f}(\widehat{\boldsymbol{x}}(t_k;t_k), \boldsymbol{h}(\widehat{\boldsymbol{x}}(t_k;t_k))),
\end{aligned}
$$

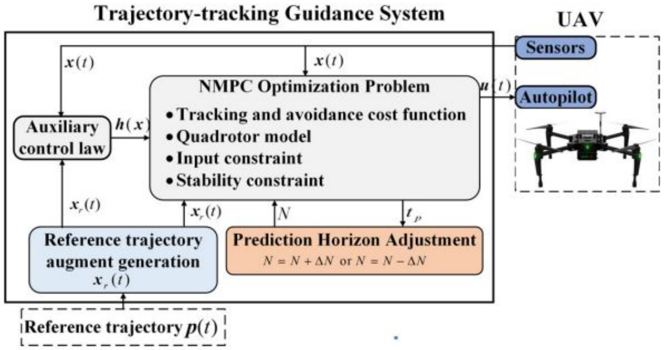
- The NMPC can now be formed as seen on the left.
    - The error state i $\widetilde{\boldsymbol{x}} = \widehat{\boldsymbol{x}} - \boldsymbol{x}_r$
    - **t_k** is the current time,
    - **T** is the prediction horizon,
    - **Q** and **R** are positive-definite matrices that define cost.
- This formation also considers Lyapunov Stability:
    - **h(x)** is the lyapunov-based nonlinear tracking control law
    - **V(x)** is the corresponding Lyapunov function.

Nonlinear Model Predictive Control-Based Guidance Algorithm for Quadrotor Trajectory Tracking with Obstacle Avoidance (JSSC 2021)

NMPC Implementation

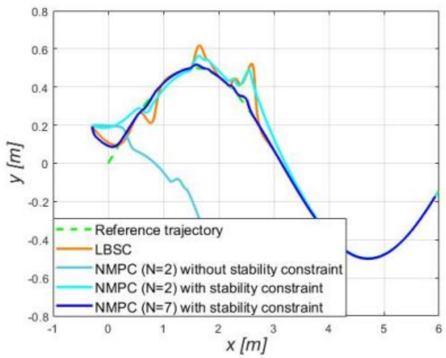
- The NMPC is converted to a discrete-time version as seen on the right.
  - **delta** is the sampling period.
  - **N** is the number of steps in the sampling period.
  - **c_in** accounts for both the input and stability constraint.

$$
\begin{aligned}
\mathcal{P}_1(\boldsymbol{x}) \quad & \min_{\widehat{\boldsymbol{u}}_i} J = \sum_{i=0}^{N} (||\widetilde{\boldsymbol{x}}_i||_{\boldsymbol{Q}}^2 + ||\widehat{\boldsymbol{u}}_i||_{\boldsymbol{R}}^2 + J^{oa}(\widehat{\boldsymbol{x}}_i))\delta \\
& \text{s.t.} \quad \widehat{\boldsymbol{x}}_{i+1} = \widehat{\boldsymbol{x}}_i + f(\widehat{\boldsymbol{x}}_i, \widehat{\boldsymbol{u}}_i)\delta, \\
& \qquad \widehat{\boldsymbol{x}}_0 = \boldsymbol{x}(t_k), \\
& \qquad \boldsymbol{c}_{\text{in}}(\widehat{\boldsymbol{x}}_i, \widehat{\boldsymbol{u}}_i) \leq \boldsymbol{0},
\end{aligned}
$$

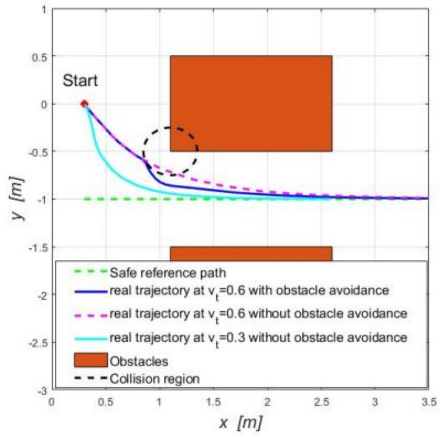


Nonlinear Model Predictive Control-Based Guidance Algorithm for Quadrotor Trajectory Tracking with Obstacle Avoidance (JSSC 2021)

NMPC Experimental Results



- Trajectory tracking was tested with and without the stability constraint (left)
    - Under wind gusting conditions, the stability constraint allows the drone to better track the given trajectory.
- For collision avoidance, a static obstacle was placed with a known position (right).
    - As v_t increased, the controller was able to avoid the collision region.



Aerospace Intelligent Mechatronics Lab

Toronto Metropolitan University