Cascading MPC for Decoupling Safety and Planning

Dhimiter Pikuli

February 3, 2025

1 Introduction

In this paper we develop a crowd navigation control scheme based on MPC (Model Predictive Control). The basic Newtonian motion equations are used as the model. Firstly, the dynamics are defined in 1D in Section 2.1 then extended to 2D in Section 2.2. A simple MPC approach is described in Section 3 with the sole objective to follow a reference path, before introducing additional constraints to the problem in Section 4 so the agent avoids collisions. Finally, in Section 5 a new MPC scheme is introduced that can decouple the planning horizon from the safety horizon. The idea between the novel approach is to utilize it as a tool for further analysis as it remains impractical in application due to the increased running complexity.

2 Dynamics

As mentioned previously we assume that the model behind the control of the agent is based on Newtonian motion. The position $\vec{p}$ is predicted using up to its second derivative $\ddot{\vec{p}}$, however we simplify the problem which gets increasingly demanding in computation, by using the first derivative as control $\dot{\vec{p}}$.

2.1 Dynamics in 1D

The model in one dimension is

$$x_{k+1} = x_k + \dot{x}_k T + \frac{1}{2} T^2 \ddot{x}_k \quad \text{and} \quad \dot{x}_{k+1} = \dot{x}_k + T\ddot{x}_k,$$

where k represent the current step and T the time step used to discretize time. In order to use velocity $\dot{x}$ as control, acceleration $\ddot{x}$ is rewritten as

$$\dot{x}_{k+1} = \dot{x}_k + T\ddot{x}_k \Leftrightarrow \ddot{x}_k = \frac{\dot{x}_{k+1} - \dot{x}_k}{T}. \tag{1}$$

After the substitution we write the dynamics in matrix form as

$$\begin{bmatrix} x_{k+1} \\ \dot{x}_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_k \\ \dot{x}_k \end{bmatrix} + \begin{bmatrix} \frac{1}{2}T^2 \\ T \end{bmatrix} \frac{\dot{x}_{k+1} - \dot{x}_k}{T} \Leftrightarrow$$

$$\begin{bmatrix} x_{k+1} \\ \dot{x}_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_k \\ \dot{x}_k \end{bmatrix} + \begin{bmatrix} \frac{1}{2}T \\ 1 \end{bmatrix} \dot{x}_{k+1} - \begin{bmatrix} \frac{1}{2}T \\ 1 \end{bmatrix} \dot{x}_k = \begin{bmatrix} 1 & \frac{1}{2}T \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_k \\ \dot{x}_k \end{bmatrix} + \begin{bmatrix} \frac{1}{2}T \\ 1 \end{bmatrix} \dot{x}_{k+1}.$$

Applying this function iteratively i-times results in matrices le

$$\begin{bmatrix} 1 & \frac{1}{2}T \\ 0 & 0 \end{bmatrix}^i \begin{bmatrix} \frac{1}{2}T \\ 1 \end{bmatrix} = \begin{bmatrix} T \\ 0 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} 1 & \frac{1}{2}T \\ 0 & 0 \end{bmatrix}^i = \begin{bmatrix} 1 & \frac{1}{2}T \\ 0 & 0 \end{bmatrix}.$$

For simplicity in the following notation we assume that the current time-step has index zero and that the horizon for which MPC predicts the future position is N resulting in the following prediction for position and velocity at step N

$$\begin{bmatrix} x_N \\ \dot{x}_N \end{bmatrix} = \begin{bmatrix} 1 & \frac{1}{2}T \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_0 \\ \dot{x}_0 \end{bmatrix} + \sum_{k=1}^{N-1} \begin{bmatrix} T \\ 0 \end{bmatrix} \dot{x}_{N-k} + \begin{bmatrix} \frac{1}{2}T \\ 1 \end{bmatrix} \dot{x}_N \tag{2}$$

Next we build the vector representation of the horizon for $X = \begin{bmatrix} x_1, & \dots & x_N \end{bmatrix}^\intercal$, $\dot{X} = \begin{bmatrix} \dot{x}_1, & \dots & \dot{x}_N \end{bmatrix}^\intercal$

$$\begin{aligned} X &= x_0 + \frac{1}{2}T\dot{x}_0 + M_{xv}\dot{X}, \\ \text{where} \quad M_{xv} &= \begin{bmatrix} 1/2 & 0 & \dots & 0 \\ 1 & 1/2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 1 & \dots & 1 & 1/2 \end{bmatrix} T, \\ \text{trivially} \quad \dot{X} &= \dot{X}. \end{aligned}$$

2.2 Dynamics in 2D

In 2D we go from using $x, \dot{x}, \ddot{x}$ to $\vec{p} = \begin{bmatrix} x & y \end{bmatrix}^\intercal, \vec{\dot{p}} = \begin{bmatrix} \dot{x}, & \dot{y} \end{bmatrix}^\intercal, \vec{\ddot{p}} = \begin{bmatrix} \ddot{x}, & \ddot{y} \end{bmatrix}^\intercal$ while also transforming the horizon vectors to

$$P = \begin{bmatrix} x_1 \\ \vdots \\ x_N \\ y_1 \\ \vdots \\ y_N \end{bmatrix} \in \mathbb{R}^{2N}, \dot{P} = \begin{bmatrix} \dot{x}_1 \\ \vdots \\ \dot{x}_N \\ \dot{y}_1 \\ \vdots \\ \dot{y}_N \end{bmatrix}, P_0 = \begin{bmatrix} x_0 \\ \vdots \\ x_0 \\ y_0 \\ \vdots \\ y_0 \end{bmatrix}, \dot{P}_0 = \begin{bmatrix} \dot{x}_0 \\ \vdots \\ \dot{x}_0 \\ \dot{y}_0 \\ \vdots \\ \dot{y}_0 \end{bmatrix}.$$

Building on the dynamics for 1D and keeping in mind the symmetry between both dimensions x and y, the prediction for the horizon predictions is

$$\begin{aligned} P = P_0 + \frac{1}{2}T\dot{P}_0 + M_{pv}\dot{P} \quad \text{where} \quad M_{pv} &= \begin{bmatrix} M_{xv} & 0 \\ 0 & M_{yv} \end{bmatrix} \\ \text{and} \quad M_{xv} &= M_{yv}. \end{aligned} \tag{3}$$

3 MPC

After defining our model we can implement a naive MPC without any safety guarantees or other limiting properties. It is necessary to define a cost function for which to optimize control. In this case, we assume a reference trajectory P^{ref} with the same length as the control horizon N. This reference trajectory can be as simple as a straight line to a goal position or it can be learned based on an abstraction of the current state of the environment which we call a (partial) observation. Similarly, it is possible to directly specify reference velocities $\dot{P}^{ref}$ instead of positions.

3.1 Cost Function

For reference positions P^{ref} the cost function takes the form

$$\frac{1}{2}\left|\left|P - P^{ref}\right|\right|,$$

which we need to minimize for the control input $\dot{P}$

$$\begin{aligned}
&\min_{\dot{P}} \frac{1}{2}\left(P_0 - P^{ref} + \frac{1}{2}T\dot{P}_0 + M_{pv}\dot{P}\right)^\intercal \left(P_0 - P^{ref} + \frac{1}{2}T\dot{P}_0 + M_{pv}\dot{P}\right) \\
\Leftrightarrow &\min_{\dot{P}} \frac{1}{2}\left(P_0 - P^{ref} + \frac{1}{2}T\dot{P}_0\right)^\intercal M_{pv}\dot{X} + \frac{1}{2}\dot{X}^\intercal M_{pv}\left(P_0 - P^{ref} + \frac{1}{2}T\dot{P}_0\right) + \frac{1}{2}\dot{P}^\intercal M_{pv}^\intercal M_{pv}\dot{P} \\
\Leftrightarrow &\min_{\dot{P}} \left(P_0 - P^{ref} + \frac{1}{2}T\dot{P}_0\right)^\intercal M_{pv}\dot{P} + \frac{1}{2}\dot{P}^\intercal M_{pv}^\intercal M_{pv}\dot{P}.
\end{aligned}$$

We solve this problem formulation with a QP-solver with quadratic and linear component Q and v respectively

$$\begin{aligned}
Q &= M_{pv}^\intercal M_{pv} \\
v &= \left(P_0 - P^{ref} + \frac{1}{2}T\dot{P}_0\right)^\intercal M_{pv} \\
\Rightarrow &\min_{\dot{P}} \frac{1}{2}\dot{P}^\intercal Q\dot{P} + v\dot{P}.
\end{aligned}$$

In order for the QP solver to find the optimal solution it is necessary to somehow limit control which we can do naively by limiting velocity to a desired range, e.g. $-b \leq \dot{P} \leq b$.

When using reference velocities the cost function is $1/2||\dot{P} - \dot{P}^{ref}||$ and the resulting QP formulation is

$$\begin{aligned}
&\min_{\dot{P}} \frac{1}{2}\left(\dot{P} - \dot{P}^{ref}\right)^\intercal \left(\dot{P} - \dot{P}^{ref}\right) \\
\Leftrightarrow &\min_{\dot{P}} \frac{1}{2}\dot{P}^\intercal\dot{P} - \dot{P}^{ref\intercal}\dot{P} \\
\Rightarrow &Q = I_N \quad \text{and} \quad v = -\dot{P}^{ref\intercal}.
\end{aligned}$$

3.2 Stability

For better stability and smoother navigation it helps to add another weighted component in the cost function, e.g.

$$\left|\left|P - P^{ref}\right|\right| + c\left|\left|\dot{P}\right|\right|$$

with constant c for tuning the importance of the stability component. This change transforms the quadratic component into $Q = M_{pv}^{\mathsf{T}} M_{pv} + cI_N$.

4 Constraints

The advantage of using MPC is the possibility to explicitly define constraints on the system. For navigation, we have three main constraint: avoiding collisions with moving and static obstacles, in case of a collision the agent has zero velocity (passive safety) and finally the agent should be limited to viable control inputs. All these types of constraints can be integrated in the QP-solver as inequality or equality constraints after projecting them into a linear space. This projection often results in more strict constraints which can potentially affect the efficiency of the solution.

4.1 Static Obstacle Avoidance

The MPC scheme should be able to constrain the agent position when necessary thus defining regions of the environment that the agent cannot access, e.g. walls. This can be extended to any obstacles that are approximated as polygons. In this case, simple linear constraints can be computed to divide the plane into two sub-planes, one designated to an obstacle and the other to the agent as a viable space to move in. We represent the constraint with the equation $ax + by + c$ as an inequality. Putting x and y in one vector and taking into the consideration the whole future horizon P the constraint looks like the following

$$M_{aj}P + c_j < 0 \Leftrightarrow M_{aj}P < -c_j \quad \text{where} \quad M_{aj} = \begin{bmatrix} I_N a_j & | & I_N b_j \end{bmatrix},$$

where j indexes the constraint. Replacing with our model for the future Equation (3) gives constraint matrix and vector G_j and b_j for static obstacles

$$\begin{aligned} M_{aj}(P_0 + \frac{1}{2}T\dot{P}_0 + M_{pv}\dot{P}) < -c_j \Leftrightarrow & M_{aj}M_{pv}\dot{P} < -M_{aj}(P_0 + \frac{1}{2}T\dot{P}_0) - c_j, \\ \Rightarrow & G_j = M_{aj}M_{pv} \quad \text{and} \quad b_j = -M_{aj}(P_0 + \frac{1}{2}T\dot{P}) - c_j. \end{aligned}$$

In this case the unit is distance, so a and b have no unit and c is also a distance (m).

4.2 Moving Obstacle Avoidance

We approximate each moving member of the crowd as a circle with predefined radius. The radius is used to define distance d between the agent $\vec{p}$ and a crowd member $\vec{m}_k$ where k is used to index the crowd

$$||\vec{p} - \vec{m}_k|| \geq d,$$

as proposed in (Bohorquez et al., 2016). In this formulation, we assume the same distance d for each member of the crowd. Since QP-solvers allow only for linear constraints we linearize by rewriting

$$\hat{u}_k^\intercal(\vec{p} - \vec{m}_k) \geq d \quad \text{with} \quad \hat{u}_k^\intercal = \frac{\vec{p} - \vec{m}_k}{||\vec{p} - \vec{m}_k||}.$$

In the MPC scheme it is necessary to avoid collision with the crowd by predicting the future positions of the crowd which we write as

$$\hat{u}_{k,i}^\intercal(\vec{p}_i - \vec{m}_{k,i}) \geq d, \quad \forall i, k$$

where i indexes the position of the agent and the crowd member k at step i. Using the dynamics Equation (3) we can write the constraint in matrix form

$$\hat{U}_k \left(P_0 + \frac{1}{2} T \dot{P}_0 + M_{pv} \dot{P} - M_k \right) \geq d$$

$$\text{where} \quad \hat{U}_k = \begin{bmatrix} I_N \hat{u}_k^x & | & I_N \hat{u}_k^y \end{bmatrix} \quad \text{and} \quad M_k = \begin{bmatrix} m_{k,1}^x \\ \vdots \\ m_{k,N}^x \\ m_{k,1}^y \\ \vdots \\ m_{k,N}^y \end{bmatrix}.$$

Which is rewritten in order to derive constraint matrices and vectors G_k, b_k for each member of the crowd k

$$\begin{aligned} \hat{U}_k \left(P_0 + \frac{1}{2} T \dot{P}_0 - M_k \right) + \hat{U}_k M_{pv} \dot{P} &\geq d \\ \hat{U}_k M_{pv} \dot{P} &\geq d - \hat{U}_k \left(P_0 + \frac{1}{2} T \dot{P}_0 - M_k \right) \\ -\hat{U}_k M_{pv} \dot{P} &\leq -d + \hat{U}_k \left(-(M_k - P_0) + \frac{1}{2} T \dot{P}_0 \right) \\ \Rightarrow G_k = -\hat{U}_k M_{pv} \quad \text{and} \quad b_k &= -d + \hat{U}_k \left(-(M_k - P_0) + \frac{1}{2} T \dot{P}_0 \right). \end{aligned}$$

4.3 Passive Safety

Since it is impossible to ensure complete safety due to the limited future horizon but fundamentally also because predicting crowd behaviour perfectly is unattainable, we aim for passive safety. For passive safety it suffices that in case of a collision the agent is at rest. This means that at the end of each control trajectory the velocity should be equal to zero. In this way if the problem ever becomes infeasible for the current step the last computed trajectory serves as a safety net in order to bring the agent to a stop without any collisions. In case of a collision after the agent has stopped the damage is only relative to velocity of the moving obstacle. Mathematically, this condition is expressed simply

$$\vec{\dot{p}}_N = 0.$$

We can simplify our control by hard setting the last control value $\vec{p}_N$ equal to 0 thus also reducing the size of the problem.

$$\dot{P} = \begin{bmatrix} x_1, & \dots & x_{N-1}, & y_1, & \dots & y_{N-1} \end{bmatrix}^\intercal \quad \text{and} \quad x_N = y_N = 0 \tag{4}$$

With new dynamics

$$M_{xv} = \begin{bmatrix} 1/2 & 0 & \dots & 0 \\ 1 & 1/2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & 1/2 \\ 1 & \dots & \dots & 1 \end{bmatrix} T \in \mathbb{R}^{N \times N-1}, \quad \text{and} \quad M_{pv} = \begin{bmatrix} M_{xv} & 0 \\ 0 & M_{yv} \end{bmatrix} \in \mathbb{R}^{2N \times 2(N-1)}.$$

In the following we use this notation.

4.4 Limiting Acceleration and Velocity

Acceleration and velocity are limited in our modeling in their absolute value which requires a quadratic representation when using Cartesian coordinates. This would result in a non-linear constraint. In order to define constraints for acceleration and velocity, the conditions need to be linearized. We do this by fitting the circle that describes the acceleration and velocity limit with a regular polygon as in Figure 1. The distance from origin O to A represents the

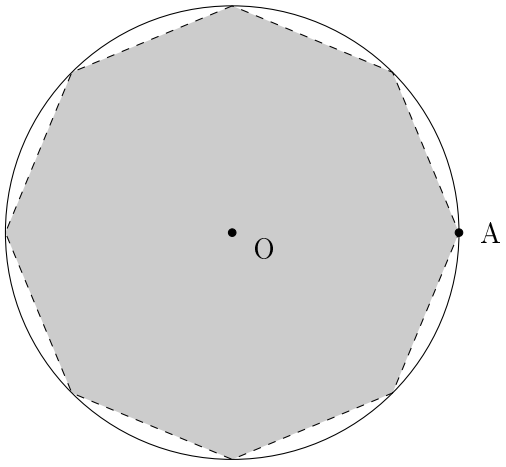

Figure 1: Linearizing constraints means also applying a harsher constraint that fit inside the non-linear constraints.

maximum viable value for acceleration a_{max} or velocity v_{max}. In order to apply the octagon as a constraint all its edges represent a linear constraint. Points of the octagon are generated starting from point $A = \begin{bmatrix} 1, & 0 \end{bmatrix}^\intercal$, where constant c relates to the maximal value that the norm can take for the given constraint. The other points are generated by multiplying with a rotation matrix, e.g. for an octagon

$$\begin{bmatrix} \cos \pi/4 & -\sin \pi/4 \\ \sin \pi/4 & \cos \pi/4 \end{bmatrix}.$$

With the generated points then it is possible to compute the edges (lines) of the polygon for each two given vertices

$$m = \frac{y_2 - y_1}{x_2 - x_1} \quad \text{and} \quad b = y_1 - mx_1.$$

For the four edges above $y < mx + b$ meaning $y - mx - b < 0$ while for the four edges below $y > mx + b$ meaning $y - mx - b > 0$ or even $-y + mx + b < 0$. We build the constraints for the entire horizon for each edge l

$$\begin{aligned} M_{vl}\dot{P} - b_{vl} < 0 \Leftrightarrow M_{vl}\dot{P} < b_{vl} \\ \text{where} \quad G_{vl} = M_{vl} = \begin{bmatrix} -I_N m_l & | & I \end{bmatrix} \end{aligned} \tag{5}$$

In this case the unit is $[m/s]$ which means that m does not have a unit and b has unit $[m/s]$.

To limit acceleration we make use of Equation (1). In this case, we also need to account for $\vec{p}_N$ which is explicitly 0 (as a reminder $\dot{P}$ was redefined in Equation (4))

$$\begin{aligned} & M_{al}\ddot{P} - \hat{b}_{al} < 0 \Leftrightarrow M_{al}\ddot{P} < \hat{b}_{al} \Leftrightarrow M_{al}\frac{\dot{P}_{1:N} - \dot{P}_{0:N-1}}{T} < \hat{b}_{al} \\ \Leftrightarrow & M_{al}\begin{bmatrix} \dot{x}_1 - \dot{x}_0 \\ \vdots \\ \dot{x}_N - \dot{x}_{N-1} \\ \dot{y}_1 - \dot{y}_0 \\ \vdots \\ \dot{y}_N - \dot{y}_{N-1} \end{bmatrix}\frac{1}{T} < \hat{b}_{al} \overset{\vec{p}_N = 0}{\Longleftrightarrow} M_{al}\begin{bmatrix} \dot{x}_1 - \dot{x}_0 \\ \vdots \\ -\dot{x}_{N-1} \\ \dot{y}_1 - \dot{y}_0 \\ \vdots \\ -\dot{y}_{N-1} \end{bmatrix}\frac{1}{T} < \hat{b}_{al} \\ \Leftrightarrow & \frac{M_{al}}{T}\begin{bmatrix} \dot{x}_1 \\ \vdots \\ -\dot{x}_{N-1} \\ \dot{y}_1 \\ \vdots \\ -\dot{y}_{N-1} \end{bmatrix} - \frac{M_{al}}{T}\begin{bmatrix} \dot{x}_0 \\ 0 \\ \vdots \\ \dot{y}_0 \\ 0 \\ \vdots \end{bmatrix} < \hat{b}_{al} \\ \Leftrightarrow & M_{al}\begin{bmatrix} D & 0 \\ 0 & D \end{bmatrix}\dot{P} < \hat{b}_{al} + \frac{M_{al}}{T}\begin{bmatrix} \dot{x}_0, & 0, & \dots & \dot{y}_0, & 0, & \dots \end{bmatrix}^\intercal \\ \text{where} \quad & D = \begin{bmatrix} 1 & 0 & \dots & 0 \\ -1 & 1 & \ddots & \vdots \\ 0 & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & 1 \\ 0 & \dots & 0 & -1 \end{bmatrix}\frac{1}{T} \in \mathbb{R}^{N \times N-1}. \end{aligned} \tag{6}$$

We define constraints as

$$G_{al} = M_{al}\begin{bmatrix} D & 0 \\ 0 & D \end{bmatrix} \quad \text{and} \quad b_{al} = \hat{b}_{al} + \frac{M_{al}}{T}\begin{bmatrix} \dot{x}_0, & 0, & \dots & \dot{y}_0, & 0, & \dots \end{bmatrix}^\intercal$$

In this case the unit is $[m/s]$ which means that m does not have a unit and b has unit $[m/s]$. In both cases the sign of the inequality have to be changed depending on the line of the polygon.

The QP problem with all the additional constraints can be formulated as

$$\begin{aligned} & \min_{\dot{P}} \dot{P}^{\intercal} Q \dot{P} + v\dot{P} \\ \text{s.t} \quad & G_k \dot{P} \leq b_k \quad \forall k, \\ & G_j \dot{P} \leq b_j \quad \forall j, \\ & G_{vl} \dot{P} \leq b_{vl} \quad \forall vl, \\ & G_{al} \dot{P} \leq b_{al} \quad \forall al. \end{aligned}$$

4.5 Problem Complexity & Dimensionality

The simulation used test the approach is setup with the following specifications

$$v_{MAX} = 3m/s,\ a_{MAX} = 1.5m/s^2,\ T = 0.1s. \tag{7}$$

This implies that the agent needs at least 2 seconds to stop at any time since the maximum velocity is $3m/s$ and the maximum acceleration is $1.5m/s^2$. This also means that in order for the MPC to navigate at maximum velocity, a horizon longer then 2 seconds is necessary. This motivates the use of $N = 21$ ($N = 20$ would not work because the maximum velocity would not be reachable for the same reason as $N = 1$ would mean that the MPC always predicts $0m/s$ velocity in order to ensure passive safety). Given these parameters it is possible to list all constraints and dimensions parametrizing the problem. The simulation running with a constant velocity crowd involves 6 crowd members where for each crowd member 21 collisions need to be avoided for the entire horizon, meaning 126 constraints. In this environment no special static obstacles are implemented apart from the four walls which results in 84 constraints. The acceleration and velocity constraint are linearized using an octagon which results in 168 and 160 constraints for acceleration and velocity respectively. This brings the total to 538 constraints with a control output equal to 40.

However, it is possible to lower the number of constraints by ignoring those that are irrelevant to the problem. For example, it is not necessary to consider potential crashes with a crowd member far from the agent and going further from it. Similarly, it is not necessary to consider walls far from the agent. Acceleration constraints cannot be removed whereas velocity constraints can be only potentially violated in the current direction that the agent is moving. On average only 320 constraints are taken in consideration in the QP.

5 Cascading MPC

In an attempt to decouple efficiency from safety we propose to implement a cascading stream of safety breaking trajectories instead of ensuring safety simply from the current start position of the agent. This implies two horizons, M for planning and N for safety. The equation representing this objective is

$$\min_{\dot{P}} \left|\left| P - P^{ref} \right|\right| \quad \text{and} \quad \vec{p_i} = \vec{b_1^i}, \vec{\dot{p}_i} = \vec{\dot{b}_1^i} \quad \forall i \in \{1, \ldots, M\}, \tag{8}$$

where P and $\dot{P}$ are defined the same as previously whereas B, $\dot{B}$ and elements b_j^i will represent the cascading trajectory. In this notation i represents the planning step between 1 and M

whereas j the breaking step between 1 and N. The Equation (8) indicates that the steps in the planning horizon P are identical to the first steps of all breaking trajectories. Section 5 represents three breaking trajectories indexed with $i = 1, 2, 3$ while the plan is represented by the black line going from the agent in green to the black cross representing the goal.

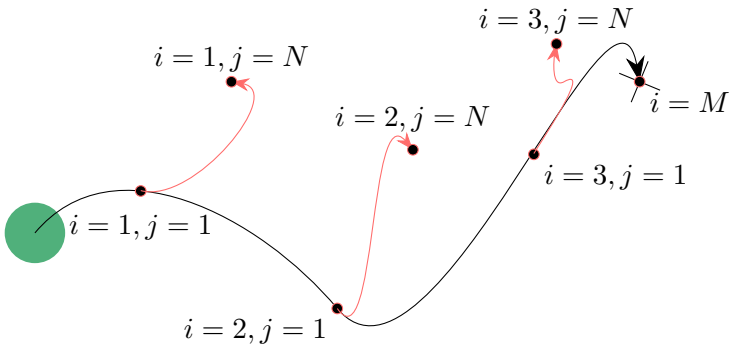

Figure 2: For the computed trajectory (black line) P at each time-stamp a breaking trajectory B^i is computed. In the image only three point are sampled from $\{1, \dots M\}$.

5.1 Changes in Dynamics

The target control computed will be linearized as

$$\dot{B}_x := \begin{bmatrix} \dot{bx}_1^1, & \dots & \dot{bx}_{N-1}^1, & \dot{bx}_1^2, & \dots & \dot{bx}_{N-1}^2, & \dots & \dot{bx}_1^M, & \dots & \dot{bx}_{N-1}^M \end{bmatrix}^\intercal \in \mathbb{R}^{M(N-1)},$$

where safety trajectories go up to $N-1$ as we specify again explicitly that the last control value for each breaking trajectory is zero, $\dot{b}_N^i = 0$, $\forall i \in \{1, \dots M\}$. For 2D we generalize $B = \begin{bmatrix} B_x, & B_y \end{bmatrix}^\intercal$ where B_y is defined analogously to B_x.

Inside breaking trajectories the dynamics remain unchanged (we still use M_{xv}) however the initial step b_1^i will be relative to all other previous steps b_1^j where $j < i$ (correlation ensured with matrix F, as seen in Equation (2) to compute x_N all previous $\dot{x}_{N-k}$ for $k < N-1$ are

multiplied by T)

$$B = B_0 + \frac{1}{2}T\dot{B}_0 + M_{b\dot{b}}\dot{B} \quad \text{where} \quad B_0 = \begin{bmatrix} bx_0 \\ \vdots \\ bx_0 \\ by_0 \\ \vdots \\ by_0 \end{bmatrix}, \dot{B}_0 = \begin{bmatrix} \dot{bx}_0 \\ \vdots \\ \dot{bx}_0 \\ \dot{by}_0 \\ \vdots \\ \dot{by}_0 \end{bmatrix}$$

$$M_{b\dot{b}} = \begin{bmatrix} M_{b\dot{b}x} & 0 \\ 0 & M_{b\dot{b}y} \end{bmatrix}, \quad M_{b\dot{b}x} = \begin{bmatrix} M_{xv} & 0 & \dots & 0 \\ F & M_{xv} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ F & \dots & F & M_{xv} \end{bmatrix} \in \mathbb{R}^{MN \times M(N-1)}$$

$$\text{and} \quad M_{xv} = \begin{bmatrix} 1/2 & 0 & \dots & 0 \\ 1 & 1/2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & 1/2 \\ 1 & \dots & \dots & 1 \end{bmatrix} T \in \mathbb{R}^{N \times N-1}, \quad F = \begin{bmatrix} 1 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & \dots & 0 \end{bmatrix} T \in \mathbb{R}^{N \times N-1}$$

because of symmetry in Cartesian space $\quad M_{xv} = M_{yv}, \quad M_{b\dot{b}x} = M_{b\dot{b}y}$.

5.2 Objective

For the objective we need to follow the reference trajectory P^{ref} which relate to the first index of each breaking trajectory $b_1^i \ \forall i \in \{1, \dots M\}$. We accomplish this by filtering the irrelevant indexes in B

$$F_P = \begin{bmatrix} F_X & 0 \\ 0 & F_Y \end{bmatrix} \quad \text{where} \quad F_X = \begin{bmatrix} f & 0 & \dots & 0 \\ 0 & f & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & f \end{bmatrix} \in \mathbb{R}^{M \times M(N-1)},$$

$$F_X = F_Y, \quad f = \begin{bmatrix} 1, & 0, & \dots & 0 \end{bmatrix} \in \mathbb{R}^{1 \times (N-1)},$$

in this way the objective function can be written as

$$\frac{1}{2} \min_{\dot{B}} ||F_P B - P^{ref}|| \Leftrightarrow \frac{1}{2} \min_{\dot{P}} ||P - P^{ref}||.$$

The resulting QP is equivalent to the simple MPC formulation

$$Q = (F_P M_{b\dot{b}})^\intercal (F_P M_{b\dot{b}}) = M_{pv}^\intercal M_{pv}$$
$$v = \left(P_0 - P^{ref} + \frac{1}{2}\dot{P}_0 \right)^\intercal F_P M_{b\dot{b}} = \left(P_0 - P^{ref} + \frac{1}{2}\dot{P}_0 \right)^\intercal M_{pv}.$$

5.3 Constraints

5.3.1 Obstacle Avoidance

Collision avoidance constraints do not change at all as they are applied equally for each step. In this case however the crowd position must to be predicted for $M + N$ steps. The crowd position horizon vector for member k would look like the following in order to relate to the new control vector B

$$\begin{aligned} M_k = [&m^x_{k,1}, \quad \dots \quad m^x_{k,N}, \quad m^x_{k,2}, \quad \dots \quad m^x_{k,N+1}, \quad \dots \quad m^x_{k,M-N}, \quad \dots \quad m^x_{k,M+N}, \\ &m^y_{k,1}, \quad \dots \quad m^y_{k,N}, \quad m^y_{k,2}, \quad \dots \quad m^y_{k,N+1}, \quad \dots \quad m^y_{k,M-N}, \quad \dots \quad m^y_{k,M+N}]^\intercal \end{aligned}$$

5.3.2 Limiting Velocity and Acceleration

Velocity is limited the same as the original MPC Equation (5) because it is applied uniformly over the entire trajectory B, whereas acceleration needs a more careful consideration. The constraint introduced in Equation (6) needs to be updated in order to consider the dependency between the first step of each breaking trajectory while keeping the dependency inside the breaking trajectories. The resulting matrix resembles the structure of $M_{b\dot{b}}$ introduced previously

$$M_{al} \begin{bmatrix} D_B & 0 \\ 0 & D_B \end{bmatrix} \dot{B} < \hat{b}_{al} + \frac{M_{al}}{T} \begin{bmatrix} \dot{x}_0, & 0, & \dots & \dot{y}_0, & 0, & \dots \end{bmatrix}^\intercal$$

$$\text{where} \quad D_B = \begin{bmatrix} D & 0 & \dots & \dots & 0 \\ F_D & D & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & D & 0 \\ 0 & \dots & 0 & F_D & D \end{bmatrix} \in \mathbb{R}^{MN \times M(N-1)} \quad \text{and} \quad F_D = \begin{bmatrix} -1 & 0 & \dots & 0 \\ 0 & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & 0 \end{bmatrix},$$

with appropriate dimension for M_{al} and b_{al} in order to comply with the dimensionality of $\dot{B}$.

5.4 Problem Complexity & Dimensionality

Using the same specification as in Equation (7) and the same analysis, we can compute the number of constraints for a planning horizon of 5 and safety horizon 21. The number of wall constraints would be $4 \times 21 \times 5$. $8 \times 21 \times 5$ and $8 \times 20 \times 5$ for acceleration and velocity constraints respectively. Finally, for moving members of the crowd there are $6 \times 21 \times 5$ constraints bringing the total to 2690. After filtering out irrelevant constraints during execution on average the QP-solver takes into consideration 1570 constraints.

References

N. Bohorquez, A. Sherikov, D. Dimitrov, and P.-B. Wieber. Safe navigation strategies for a biped robot walking in a crowd. In *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pages 379–386, Cancun, Mexico, Nov. 2016. IEEE. ISBN 978-1-5090-4718-5. doi: 10.1109/HUMANOIDS.2016.7803304. URL `https://ieeexplore.ieee.org/document/7803304/`.