Récapitulatif de la méthode MPC 22/02/2020

State vector

The reference velocity $\dot{q}^\star$ that is sent to the robot is expressed in its local frame. It has 6 dimensions: 3 for the linear velocity and 3 for the angular one.

$$\dot{q}_l^\star = [\dot{x}_l^\star \;\; \dot{y}_l^\star \;\; \dot{z}_l^\star \;\; \dot{\phi}^\star \;\; \dot{\theta}^\star \;\; \dot{\psi}^\star]^T \tag{1}$$

The velocity vector of the robot is:

$$\dot{q}_l = [\dot{x}_l \;\; \dot{y}_l \;\; \dot{z}_l \;\; \dot{\phi} \;\; \dot{\theta} \;\; \dot{\psi}]^T \tag{2}$$

At the start each iteration of the MPC, the current position and orientation of the robot defines a new frame in which the solver will work. This frame is at ground level with the x axis pointing forwards (x axis of the local frame), the y axis pointing to the left (y axis of the local frame) and the z axis point upwards. Instead of working in pitch, roll and yaw, the solver will work in terms of rotation around the x, y and z axes of this new frame. As this solver frame and the local frame are initially superposed, we have for the initial conditions of the solving process:

$$q_0 = [x \;\; y \;\; z \;\; \theta_x \;\; \theta_y \;\; \theta_z] = [0 \;\; 0 \;\; z \;\; \theta_x \;\; \theta_y \;\; 0] \tag{3}$$

$$\dot{q}_0 = [\dot{x} \;\; \dot{y} \;\; \dot{z} \;\; \dot{\theta}_x \;\; \dot{\theta}_y \;\; \dot{\theta}_z] = [\dot{x}_l \;\; \dot{y}_l \;\; \dot{z}_l \;\; \dot{\phi} \;\; \dot{\theta} \;\; \dot{\psi}] \tag{4}$$

The state vector of the robot and the reference state vector are then:

$$X = \begin{bmatrix} q \\ \dot{q} \end{bmatrix} \qquad X^\star = \begin{bmatrix} q^\star \\ \dot{q}^\star \end{bmatrix} \tag{5}$$

The reference velocity is supposed constant over the prediction horizon in the local frame of the robot which means we need to rotate it accordingly to get the reference velocity in solver frame.

For time step k of the prediction horizon, the reference velocity vector is defined as follows:

$$\dot{q}_k^\star = \begin{bmatrix} R_z(\Delta t \cdot k \cdot \dot{\phi}^\star) \\ R_z(\Delta t \cdot k \cdot \dot{\phi}^\star) \end{bmatrix} \cdot \dot{q}_0^\star = \begin{bmatrix} R_z(\Delta t \cdot k \cdot \dot{\phi}^\star) \\ R_z(\Delta t \cdot k \cdot \dot{\phi}^\star) \end{bmatrix} \cdot \dot{q}_l^\star \tag{6}$$

with $R_z(\phi)$ the 3 by 3 rotation matrix by an angle ϕ along the vertical axis. There is no rotation along x and y axes due to the assumption that the trunk is almost horizontal.

To get the reference position vector, the following iterative equation can be use:

$$q_{k+1}^\star = q_k^\star + \begin{bmatrix} R_z(\Delta t \cdot k \cdot \dot{\phi}^\star) \\ R_z(\Delta t \cdot k \cdot \dot{\phi}^\star) \end{bmatrix} \cdot \dot{q}_0^\star \tag{7}$$

The solver will work around the reference trajectory so we define the optimization state vector as follows:

$$\mathcal{X}_k = X_k - X_k^\star \tag{8}$$

Solver matrices

Goal: create the matrices that are used by standard QP solvers. Those solvers tries to find the vector $\mathbb{X}$ that minimizes $\frac{1}{2}\mathbb{X}^T.P.\mathbb{X} + \mathbb{X}^T.Q$ under constraints $M.\mathbb{X} = N$ and $L.\mathbb{X} \leq K$.

The evolution of the state vector of the robot over time can be described as follows:

$$x(k+1) = A(k)x(k) + B(k)f(k) + g \tag{9}$$

Matrices A et B depends on k and $g = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ \ -9.81 \cdot dt\ 0\ 0\ 0]^T$

The contact forces vector $f(k) = f_k$ always include the forces applied on the four feet even if some of them are not touching the ground. In that case we will set the problem in such a way that forces for such feet are not considered at all in the solving process.

$$f_k = [f_{k,0}^T \quad f_{k,1}^T \quad f_{k,2}^T \quad f_{k,3}^T]^T \tag{10}$$

$$f_{k,i} = \begin{bmatrix} f_{k,i}^x \\ f_{k,i}^y \\ f_{k,i}^z \end{bmatrix} \tag{11}$$

with $f_{k,i}^x$, $f_{k,i}^y$ and $f_{k,i}^z$ the components along the x, y and z axes of the solver frame for the i-th foothold at time step k.

Let's consider a case with only 3 time steps in the prediction horizon.

Since we are optimizing both the state vector X of the robot to have it close to $X^\star$ and the contacts forces f which are the output of the MPC, the solver vector $\mathbb{X}$ is then:

$$\mathbb{X} = [\mathcal{X}_1 \quad \mathcal{X}_2 \quad \mathcal{X}_3 \quad f_0 \quad f_1 \quad f_2]^T \tag{12}$$

Matrix M is defined as follows:

$$M = \begin{bmatrix} -I_{12} & 0_{12} & 0_{12} & B_0 & 0_{12} & 0_{12} \\ A_1 & -I_{12} & 0_{12} & 0_{12} & B_1 & 0_{12} \\ 0_{12} & A_2 & -I_{12} & 0_{12} & 0_{12} & B_2 \\ 0_{12} & 0_{12} & 0_{12} & E_0 & 0_{12} & 0_{12} \\ 0_{12} & 0_{12} & 0_{12} & 0_{12} & E_1 & 0_{12} \\ 0_{12} & 0_{12} & 0_{12} & 0_{12} & 0_{12} & E_2 \end{bmatrix} \tag{13}$$

A, B and E have a size of 12 by 12.

Matrix N is defined as follows:

$$N = \begin{bmatrix} -g \\ -g \\ -g \\ 0_{12\times1} \\ 0_{12\times1} \\ 0_{12\times1} \end{bmatrix} + \begin{bmatrix} -A_0X_0 \\ 0_{12\times1} \\ 0_{12\times1} \\ 0_{12\times1} \\ 0_{12\times1} \\ 0_{12\times1} \end{bmatrix} + \begin{bmatrix} I_{12} & 0_{12} & 0_{12} & 0_{12} & 0_{12} & 0_{12} \\ -A_1 & I_{12} & 0_{12} & 0_{12} & 0_{12} & 0_{12} \\ 0_{12} & -A_2 & I_{12} & 0_{12} & 0_{12} & 0_{12} \end{bmatrix} \cdot \begin{bmatrix} X_1^\star \\ X_2^\star \\ X_3^\star \\ 0_{12\times1} \\ 0_{12\times1} \\ 0_{12\times1} \end{bmatrix} \tag{14}$$

Matrix A_k for time step k is defined as follows:

$$A = \begin{bmatrix} I_3 & 0_3 & \Delta t \cdot I_3 & 0_3 \\ 0_3 & I_3 & 0_3 & \Delta t \cdot I_3 \\ 0_3 & 0_3 & I_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & I_3 \end{bmatrix} \tag{15}$$

The inertia matrix of the robot in solver frame rotated according to the orientation of the robot at time step k is:

$$\mathcal{I}_k = R_z(\Delta t \cdot k \cdot \dot{\phi}^\star) \cdot \mathcal{I} \tag{16}$$

Matrix B_k for time step k is defined as follows:

$$B = \begin{bmatrix} 0_3 & 0_3 & 0_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & 0_3 \\ \Delta t/m \cdot I_3 & \Delta t/m \cdot I_3 & \Delta t/m \cdot I_3 & \Delta t/m \cdot I_3 \\ \Delta t \cdot \mathcal{I}_k^{-1} \cdot [r_{k,0}]_\times & \Delta t \cdot \mathcal{I}_k^{-1} \cdot [r_{k,1}]_\times & \Delta t \cdot \mathcal{I}_k^{-1} \cdot [r_{k,2}]_\times & \Delta t \cdot \mathcal{I}_k^{-1} \cdot [r_{k,3}]_\times \end{bmatrix} \tag{17}$$

with $r_{k,i}$ the vector expressed in solver frame going from the position of the robot at time step k to the position of the i-th foothold and $[r_{k,i}]_\times$ the associated skew-symmetric matrix.

Matrix E_k for time step k is defined as follows:

$$E_k = \begin{bmatrix} e_{k,0} & 0_3 & 0_3 & 0_3 \\ 0_3 & e_{k,1} & 0_3 & 0_3 \\ 0_3 & 0_3 & e_{k,2} & 0_3 \\ 0_3 & 0_3 & 0_3 & e_{k,3} \end{bmatrix} \tag{18}$$

$e_{k,i} = 0_3$ if the i-th foot is touching the ground during time step k, $e_{k,i} = I_3$ otherwise. In fact, if $e_{k,i} = I_3$ then with $M.X = N$ we are setting the constraint that $f_{k,i} = [0 \;\; 0 \;\; 0]^T$ (no reaction force since the foot is not touching the ground).

Matrix L is defined as follows:

$$L = \begin{bmatrix} 0_{20\times12} & 0_{20\times12} & 0_{20\times12} & F_\mu & 0_{20\times12} & 0_{20\times12} \\ 0_{20\times12} & 0_{20\times12} & 0_{20\times12} & 0_{20\times12} & F_\mu & 0_{20\times12} \\ 0_{20\times12} & 0_{20\times12} & 0_{20\times12} & 0_{20\times12} & 0_{20\times12} & F_\mu \end{bmatrix} \tag{19}$$

With:

$$F_\mu = \begin{bmatrix} C & 0_{5\times3} & 0_{5\times3} & 0_{5\times3} \\ 0_{5\times3} & C & 0_{5\times3} & 0_{5\times3} \\ 0_{5\times3} & 0_{5\times3} & C & 0_{5\times3} \\ 0_{5\times3} & 0_{5\times3} & 0_{5\times3} & C \end{bmatrix} \text{ and } C = \begin{bmatrix} 1 & 0 & -\mu \\ -1 & 0 & -\mu \\ 0 & 1 & -\mu \\ 0 & -1 & -\mu \\ 0 & 0 & -1 \end{bmatrix} \tag{20}$$

The K matrix is defined as $N = 0_{60\times1}$.

Matrix P in the cost function $\frac{1}{2}\mathbb{X}^T.P.\mathbb{X} + \mathbb{X}^T.Q$ is diagonal. That way the first term of the cost function looks like $\sum_{k=1}^{3} c_{k,X} \cdot (X_1 - X_1^\star)^2 + \sum_{k=0}^{2} c_{k,f} \cdot f_k^2$ with $c_{k,X}$ and $c_{k,f}$ with size $(12, 1)$. With $c_{k,X} > 0$ it push the solver into minimizing the error $X_k - X_k^\star$ to the reference trajectory that we want to follow while $c_{k,f}$ are small positive coefficients that act as a regularization of the force to get a solution with a norm as low as possible for the reaction force. In the 3 time steps example P has a size of 72 by 72. (12 x 3 lines/columns for $\mathcal{X}_{1,2,3}$ and 12 x 3 lines/columns for $f_{1,2,3}$).

Matrix Q involved in $\mathbb{X}^T.Q$ only contains zeroes since there is no reason to push $X_k - X_k^\star$ or f_k into being as negative/positive as possible. For instance if a coefficient of Q is positive then the solver will try to have the associated variable as negative as possible to have a high negative product between the coefficient and the variable which minimizes the cost. In the 3 time steps example Q will be of size 72 by 1.

Update of solver matrices

In the M matrix:

- A_k matrices are constant and all the same
- $[r_{k,i}]$ need to be updated at each iteration since the robot moves between each iteration
- $\mathcal{I}_k^{-1}$ need to be updated if the reference velocity vector is modified
- $E_{k,i}$ in the lower part of M need to be set to I_3 if the j-th foot was previously touching the ground and is not touching it anymore. And to 0_3 if it was not touching the ground and is now touching it.

In the N matrix:

- X_0 need to be updated at each iteration since the robot moves between each iteration
- $-g$ never changes
- $X_k^\star$ need to be updated if the reference velocity vector is modified

In the L matrix:

- Columns of F_μ matrices have to be set to zero or set back to non-zero values depending on the states of the contacts (just like the columns of B_k matrices)

The N matrix never change and is always full of zeros.

Output of the MPC

The desired reaction forces that need to be applied (by TSID or the real robot) are stored in f_0. Since the solver frame and local frame are initially superposed then f_0 directly contains the desired reaction forces in the local frame of the robot. To be used in TSID, these forces need to be rotated to be brought back in TSID's world frame.

The same applies for the next desired position of the robot that is stored in $\mathcal{X}_1$ and can be retrieved by adding $X_1^\star$ to $\mathcal{X}_1$. As the next position/orientation is expressed in the solver/local frame, it needs to be rotated to be brought back in TSID's world frame.