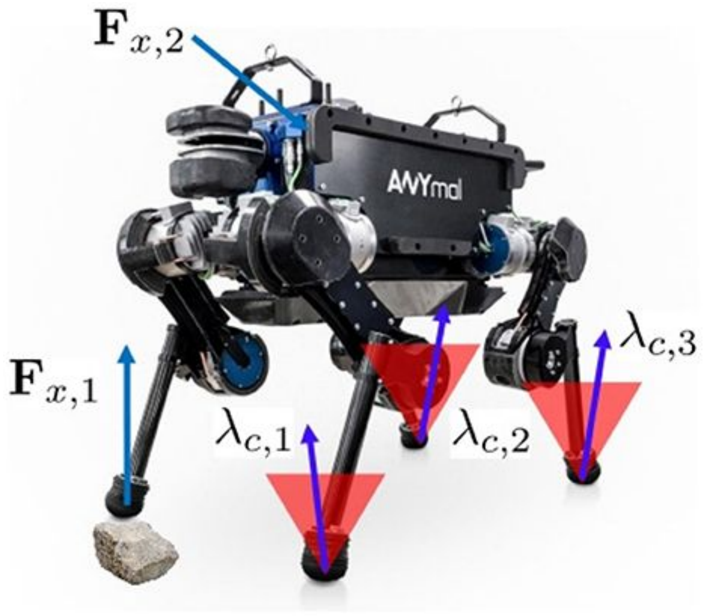
Convex Optimization in Legged Robots

Prathamesh Saraf[1] , Mustafa Shaikh[1] , Myron Phan[2]

1. Electrical and Computer Engineering (MS)
2. Mechanical And Aerospace Engineering (MS)
University of California - San Diego

Presentation flow

- Introduction and Motivation
- Prior Works
  - ZMP (LP)
  - LQR (QP)
- Linear MPC (QP)
- Non-linear MPC (QCQP)
- SOCP
- Applications
  - Jumping
  - Landing



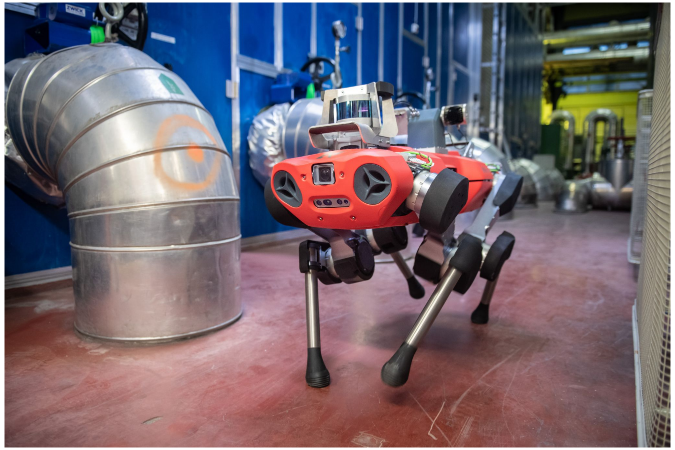
Myron Phan Mustafa Shaikh Prathamesh Saraf

Introduction

- Legged robots can be deployed in applications, like search and rescue operations, manufacturing, and military.
- This work focuses applications of convex optimization in stability and trajectory planning for legged robots
- Formulate the control problems as optimization problems with -
  - Objective function:
    - Energy consumption, stability of the system.
  - Constraints: physical limitations of the system:
    - Dynamics of the system, input/output, hardware limitations

Societal Impact

- Search and rescue during natural disasters
- Home assistance:
  - Assist people with mobility impairments,
  - Providing support and help with daily tasks
- Entertainment:
  - Dancing robots
  - Interactive experiences.
- Agriculture:
  - Harvest crops
- Military:
  - Surveillance and reconnaissance purposes
  - Search and rescue



Myron Phan Mustafa Shaikh Prathamesh Saraf

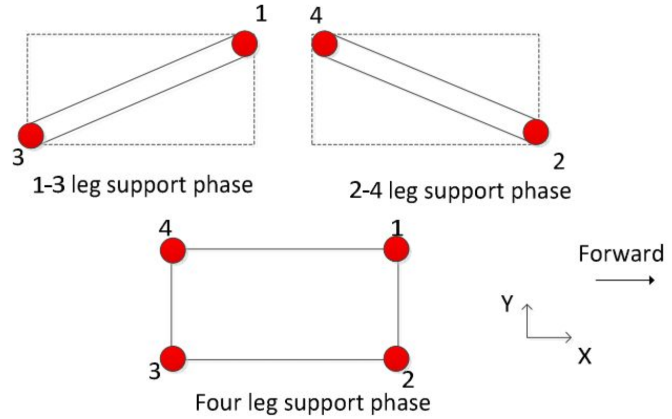
Zero Moment Point

$$Xzmp = Xcom - \frac{Zcom}{g}\ddot{X}com$$

$$Yzmp = Ycom - \frac{Zcom}{g}\ddot{Y}com$$

$$\min \|(\boldsymbol{x}_{\mathrm{zmp}} - \boldsymbol{x}_{\mathrm{cp}}), (\boldsymbol{y}_{\mathrm{zmp}} - \boldsymbol{y}_{\mathrm{cp}})\|$$

$$0 = z\ddot{x} - (x - x_{zmp})(\ddot{z} + g)$$



$$x_{zmp} = \left(\sum_{i=1}^{n} m_i(\ddot{z}_i + g)x_i - \sum_{i=1}^{n} m_i \ddot{x}_i z_i - \sum_{i=1}^{n} I_{iy}\ddot{\Omega}_{iy} \right) / \sum_{i=1}^{n} m_i(\ddot{z}_i + g)$$

$$y_{zmp} = \left(\sum_{i=1}^{n} m_i(\ddot{z}_i + g)y_i - \sum_{i=1}^{n} m_i \ddot{y}_i z_i - \sum_{i=1}^{n} I_{ix}\ddot{\Omega}_{ix} \right) / \sum_{i=1}^{n} m_i(\ddot{z}_i + g)$$

Zero Moment Point

Advantages:

1. Simple, intuitive, continuous monitoring, faster adaptation,
2. Versatile: Can be applied to any type of legged robots

Limitations:

1. Simplified Model and Dynamic Environment:
   a. Rigid body assumption with a fixed center of mass and flat ground
   b. Static environment, fixed ground and no unexpected disturbances
2. Limited Robustness:
   a. relies on robot's position and velocity,
   b. the robot's dynamics and control inputs.
3. Additional low level whole body controller to compute the joint angles and inputs

Linear Quadratic Regulator

$$J = \int_0^\infty (x^T Q x + u^T R u) dt$$

$$A^T P + PA - PBR^{-1} B^T P + Q = 0$$

$$K = R^{-1} B^T P$$

$$\tau = \tau_0 - Kx.$$

Q, R are PSD matrices

where, J = cost function, τ = input joint torques

Sean Mason, et al, “Full Dynamics LQR Control of a Humanoid Robot: An Experimental Study on Balancing and Squatting” 2014 14th IEEE-RAS International Conference on Humanoid Robots (Humanoids)

Linear Quadratic Regulator

Advantages:

1. Easy Implementation and tuning
2. More robust to disturbances in the environment than ZMP constraint.

Limitations:

1. Limited to Linear Systems
2. Computationally intensive for large and complex legged robots which reduces real-time performance and adaptability to sudden changes
3. Better suited for unconstrained systems

Model Predictive Control

- Uses the system dynamics, for predicting future steps based on the current state
- Better candidate for complex systems like legged robots

$$
\begin{aligned}
\min_{\mathbf{x},\mathbf{u}} \quad & \sum_{i=0}^{k-1} ||\mathbf{x}_{i+1} - \mathbf{x}_{i+1,\mathrm{ref}}||_{\mathbf{Q}_i} + ||\mathbf{u}_i||_{\mathbf{R}_i} \\
\text{subject to} \quad & \mathbf{x}_{i+1} = \mathbf{A}_i \mathbf{x}_i + \mathbf{B}_i \mathbf{u}_i, i = 0 \dots k-1 \\
& \underline{\mathbf{c}}_i \leq \mathbf{C}_i \mathbf{u}_i \leq \overline{\mathbf{c}}_i, i = 0 \dots k-1 \\
& \mathbf{D}_i \mathbf{u}_i = 0, i = 0 \dots k-1
\end{aligned}
$$

$$
\begin{gathered}
f_{\min} \leq f_z \leq f_{\max} \\
-\mu f_z \leq f_x \leq \mu f_z \\
-\mu f_z \leq f_y \leq \mu f_z
\end{gathered}
$$

Prior works in MPC

- Linear system dynamics
- Linear constraints
- Quadratic cost function
- Infeasible 20 years back
- But a top choice now
- z : centre of pressure

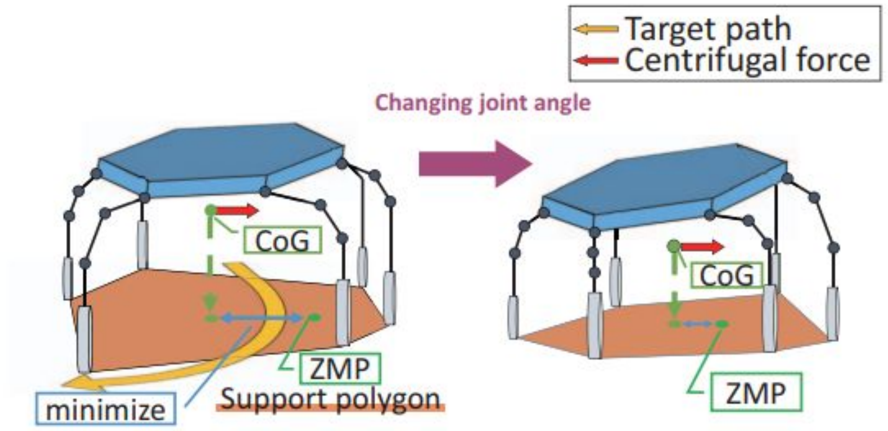
$$\min_{\dddot{x}_k, \dddot{x}_{k+1}, \ldots} \sum_{i=k}^{\infty} \frac{1}{2} Q \left(z_{i+1} - z_{i+1}^{ref} \right)^2 + \frac{1}{2} R \, \dddot{x}_i^2$$

$$z_k^{min} \leq z_k \leq z_k^{max}$$

$$z_k = \left[\begin{array}{ccc} 1 & 0 & h_{CoM}/g \end{array} \right] \hat{x}_k .$$

$$z = x - \frac{h_{CoM}}{g} \ddot{x},$$

Non-Linear MPC - QCQP



$$J = \phi(\boldsymbol{x}(t+T)) + \int_t^{t+T} L(\boldsymbol{x}(\tau), \boldsymbol{u}(\tau))\mathrm{d}\tau,$$

$$\phi(\boldsymbol{x}(t)) = Q_{1\mathrm{f}}\frac{1}{S(\boldsymbol{x}(t))} + Q_{2\mathrm{f}}e^2(\boldsymbol{x}(t)) + Q_{3\mathrm{f}}(h_{\mathrm{ref}} - h_{\mathrm{b}}(\boldsymbol{x}(t)))^2,$$

$$L(\boldsymbol{x}(t), \boldsymbol{u}(t)) = Q_1\frac{1}{S(\boldsymbol{x}(t))} + Q_2 e^2(\boldsymbol{x}(t)) + Q_3(h_{\mathrm{ref}} - h_{\mathrm{b}}(\boldsymbol{x}(t)))^2 + \boldsymbol{u}(t)^T \boldsymbol{R}\boldsymbol{u}(t).$$

$$e = \sqrt{(x_{\mathrm{ZMP}} - x_{\mathrm{CoG}})^2 + (y_{\mathrm{ZMP}} - y_{\mathrm{CoG}})^2}.$$

Non-Linear MPC - QCQP

- ZMP + MPC
- Non-linear constraints
- Quadratic Cost function
- High accuracy
- Slow computation
- Infeasible for real time applications

$$\begin{aligned} \text{minimize} \quad & J(\boldsymbol{x}(t), \boldsymbol{u}(t)), \\ \text{subject to} \quad & \dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t), \boldsymbol{u}(t)), \\ & \boldsymbol{x}(t) = \boldsymbol{x}_0, \\ & \boldsymbol{C}(\boldsymbol{x}(t), \boldsymbol{u}(t)) = 0, \end{aligned}$$

$$\boldsymbol{C}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{\epsilon}(t)) = \boldsymbol{\eta}(\boldsymbol{x}(t), \boldsymbol{u}(t)) + \boldsymbol{\epsilon}^2(t) = 0,$$

$$\boldsymbol{\eta}(\boldsymbol{x}(t)) \leq \boldsymbol{0}.$$

Model Predictive Control

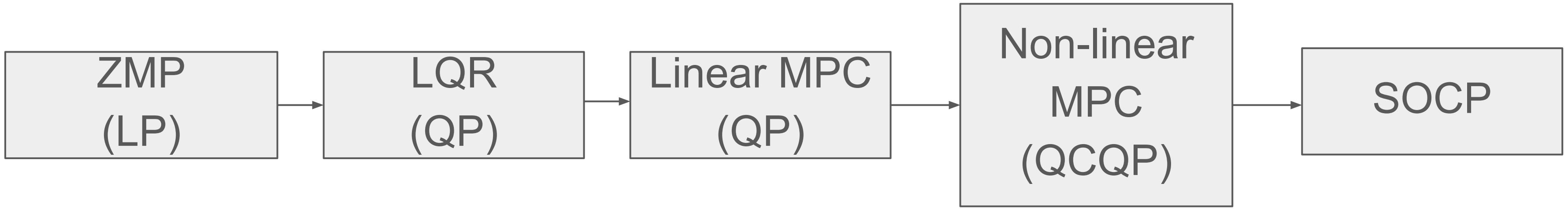
Advantages:

1. Optimal control, more robust to disturbances than LQR control.
2. Works for complex nonlinear systems making it realistic
3. Single QP required instead of multiple SISO controllers
4. But the look ahead horizon control helps the robot pre plan its behaviour

Limitations:

1. Computational intensive due to nonlinear dynamics
2. Hardware limitations: runs at 50Hz which is insufficient for online optimization

Evolution of Stability



Myron Phan Mustafa Shaikh Prathamesh Saraf

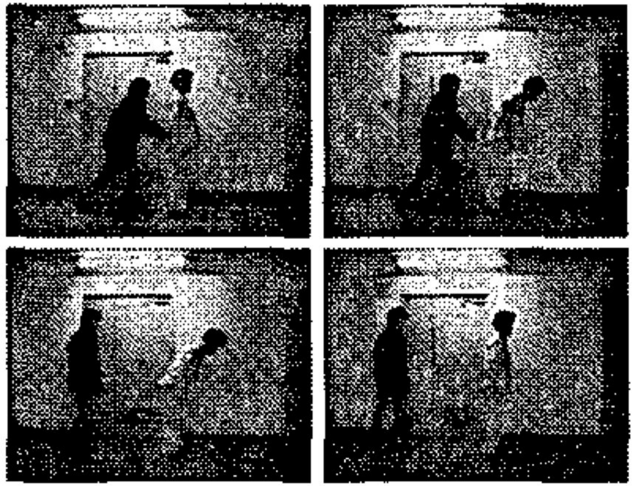
Active Balancing

- Correction of a robot's motion to prevent the robot from tipping over while being supported by one or more legs
- Input reference trajectory, which may NOT be feasible , is assumed to be given
- Objective is to correct the actual motion in real time, so as to maintain balance in the presence of external disturbances while closely following the reference trajectory
- <u>All active balancing problems track the error between the planned trajectory and posture, and the actual one</u>

Previous works

Kudoh et al. - QP that minimizes joint acceleration

- Balancing of a stationary humanoid robot subject to impact from in front or behind (future work of this is for robot walking)
- Control of the ZMP
- Maintains balance against impacts to the robot; previous approaches either used a full unstable reference trajectory to compute a stable one, that can manipulate large perturbations to realize global trajectory OR small perturbations in real time, but this couldn't generate optimal motion over a longer time horizon
- This paper does both
- QP approach controls acceleration of COM and works well against large perturbations



Myron Phan Mustafa Shaikh Prathamesh Saraf

UC San Diego

A bit more on dynamics in active balancing

- Most active balancing methods use ZMP (zero moment point) or COM (centre of mass)
- Active balancing requires simultaneous consideration of both ZMP and COM constraints since if the projected COM moves outside support polygon, robot will tip over
- Many ways to formulate these constraints - e.g. COM as equality constraint i.e. fixed to a predetermined point in the support polygon - disadvantage is that it restricts the solution space vs having COM in the support polygon

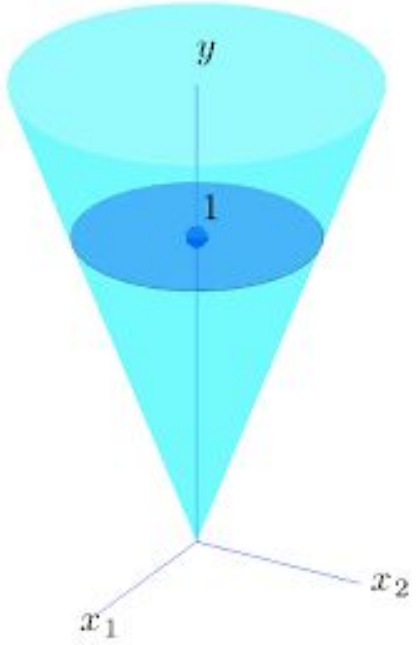
Second Order Cone Programs (SOCP)

The *second-order cone* in $\mathbf{R}^{p+1}$ is defined as

$$\mathbf{K}_p := \left\{(x, y) \in \mathbf{R}^{p+1} \;:\; \|x\|_2 \leq y\right\}.$$

This set is convex, since it is the intersection of (an infinite number of) half-spaces:

$$\mathbf{K}_p = \bigcap_{u \,:\, \|u\|_2 \leq 1} \left\{(x, y) \in \mathbf{R}^{p+1} \;:\; x^T u \leq y\right\}.$$



Myron Phan Mustafa Shaikh Prathamesh Saraf

SOCPs

- Objective is convex and constraints define a convex set - the constraint is of the exact form of the second order cone definition so its equivalent to requiring the solution to remain within the second order cone

$$\begin{aligned} &\min_x f^T x \\ &\text{subject to} \\ ||A_i x + r_i|| \le l_i^T x + m_i,\ i = 1, \dots, N \quad &\text{———} \quad ||x||_2 \le y \\ &Dx = g \end{aligned}$$

- Due to generality of SOCP, many types of convex problems can be reformulated as SOCPs e.g. QCQP using rotated cones, problems with sums of norms, problems with hyperbolic constraints, robust least squares (I.e. when there is uncertainty in the data) and many more!*

Applications of second-Order cone programming - Miguel Soma Lobo a32,Lieven Vandenberghe b,, Stephen Boyd c73,Herve Lebret

SOCPs for Active Balancing

- The main objective is to minimize the joints acceleration tracking error

$$\min_{q} ||\ddot{q} - \ddot{q_{ref}}||$$

- First use standard change of variables with epigraph formulation

$$\min t$$
$$\text{subject to} ||\ddot{q} - \ddot{q_{ref}}|| \leq t$$

- Still not SOCP since we don't have f'x in objective - new optimization variable which is a vector containing all the quantities in our objective function and constraints, and use selection matrices to pick out the ones we want

$$\mathbf{x} = (\tau, \ddot{q}, \mathcal{F}_1, \ldots, \mathcal{F}_m, \boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_m, \nu) \in \Re^{2m+N+1} \qquad f = \begin{bmatrix} 1 & 0 & 0 & \ldots & 0 \end{bmatrix}$$

Myron Phan   Mustafa Shaikh   Prathamesh Saraf

SOCPs for Active Balancing

- We now have conic constraints expressed in terms of selection matrices that select out the appropriate variable from the x vector we saw earlier

- Linear objective function

- Key is that this is a general form into which many types of COM/ZMP constraints can be expressed

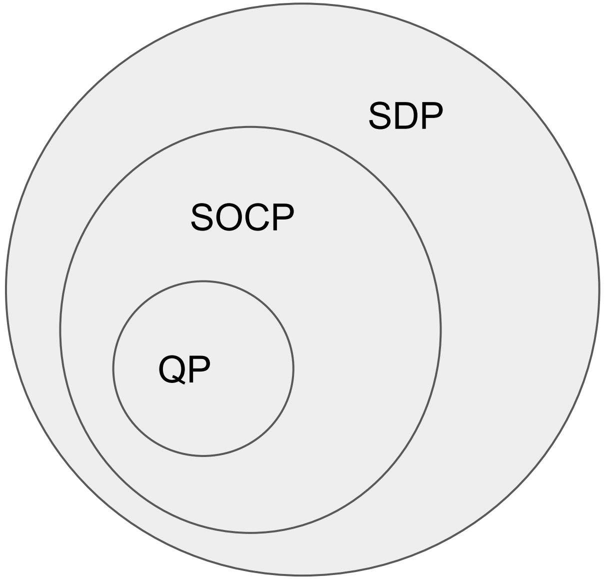
$$\min f^T \mathbf{x}$$

subject to

$$\|S_{\ddot{q}}\mathbf{x} - \ddot{q}_{\text{ref}}\| \leq S_{\tau}\mathbf{x}$$
$$A_{\text{static}} S_{\ddot{q}}\mathbf{x} \leq b_{\text{static}}$$
$$\left(A^{\mathcal{S}} A_{\text{zmp}} + b^{\mathcal{S}} b_{\text{zmp}}\right) \mathcal{F}_o \leq 0$$
$$\| \left(A_{\text{zmp}} - p^d_{\text{zmp}} b_{\text{zmp}}\right) \mathcal{F}_o \| \leq \epsilon_{\text{zmp}} b_{\text{zmp}} \mathcal{F}_o$$
$$\|P_{xy} \text{Ad}^*_{T_j} (S_{\mathcal{F}}\mathbf{x})_j\| \leq (S_{\boldsymbol{\lambda}}\mathbf{x})_j \qquad \forall j \in [1, m]$$
$$\| \left(P_{mz} - \left[p^d_{\text{zmp}}\right] P_{xy}\right) \mathcal{F}_o \| \leq S_{\nu}\mathbf{x} - \epsilon_{\text{zmp}} \mathbf{1}_{1\times m} S_{\boldsymbol{\lambda}}\mathbf{x}$$
$$\frac{1}{\mu}\mathbf{1}_{1\times m} S_{\boldsymbol{\lambda}}\mathbf{x} + \frac{1}{\mu_r} S_{\nu}\mathbf{x} \leq P_z \mathcal{F}_o$$
$$A_{\dot{V}_{\text{consts}}} S_{\ddot{q}}\mathbf{x} = \dot{V}_{\text{consts}} - b_{\dot{V}_{\text{consts}}}$$
$$\mathcal{F}_o = H S_{\mathcal{F}}\mathbf{x}$$

Why SOCPs?

- SDP is intersection of affine set and cone of PSD matrices.
- 2nd order cone can be embedded into PSD cone since we can construct a PSD matrix that represents a second order cone constraint
- But SDP solvers are much slower per iteration than SOCP solvers, so this is generally not advisable
- Robust and efficient interior point algorithms exist for SOCP - converge in 5-50 iterations regardless of the problem dimension*



Applications of second-Order cone programming - Miguel Soma Lobo a32,Lieven Vandenberghe b,, Stephen Boyd c73,Herve Lebret

Future work

- For the paper on posture correction under impact, future work related to maintaining posture control with a moving robot i.e. walking
- For the SOCP paper, the authors recognized the need to improve the computational aspects of the optimization
- exploring alternative convex optimization formulations for active balancing

UC San Diego

Future work - Relaxations

- Convexifying an objective or constraints can involve relaxing a non convex constraint
- Global optimality not guaranteed except under specific conditions; formulated in some domains such as aerospace; e.g. lower bound constraints on rocket thrust can be relaxed and global optimal still guaranteed
- Similar research must be carried out for legged robots - this will enable optimal solutions to be quickly found even under relaxed constraints in non-convex problems

$$
\begin{aligned}
\min_{u,t_f}\ & m(t_f, x(t_f)) + \zeta \int_0^{t_f} \ell(g_1(u(t)))\,\mathrm{d}t \\
\text{s.t. } & \dot{x}(t) = A(t)x(t) + B(t)u(t) + E(t)w(t), \\
& \rho_{\min} \le g_1(u(t)),\ g_0(u(t)) \le \rho_{\max},
\end{aligned}
$$

$$\Big\downarrow \quad \text{relaxation}$$

$$
\begin{aligned}
\min_{\sigma,u,t_f}\ & m(t_f, x(t_f)) + \zeta \int_0^{t_f} \ell(\sigma(t))\,\mathrm{d}t \\
\text{s.t. } & \dot{x}(t) = A(t)x(t) + B(t)u(t) + E(t)w(t), \\
& \rho_{\min} \le \sigma(t),\ g_0(u(t)) \le \rho_{\max}, \\
& g_1(u(t)) \le \sigma(t), \\
& x(0) = x_0,\ b(t_f, x(t_f)) = 0.
\end{aligned}
$$

$$\Big\downarrow$$

$$
\begin{aligned}
m_{\mathrm{LCvx}} &= \begin{bmatrix} \nabla_x m[t_f] \\ \nabla_t m[t_f] + \zeta \ell(\sigma(t_f)) \end{bmatrix} \in \mathbb{R}^{n+1}, \\
B_{\mathrm{LCvx}} &= \begin{bmatrix} \nabla_x b[t_f]^\top \\ \nabla_t b[t_f]^\top \end{bmatrix} \in \mathbb{R}^{(n+1)\times n_b}.
\end{aligned}
\longrightarrow
$$

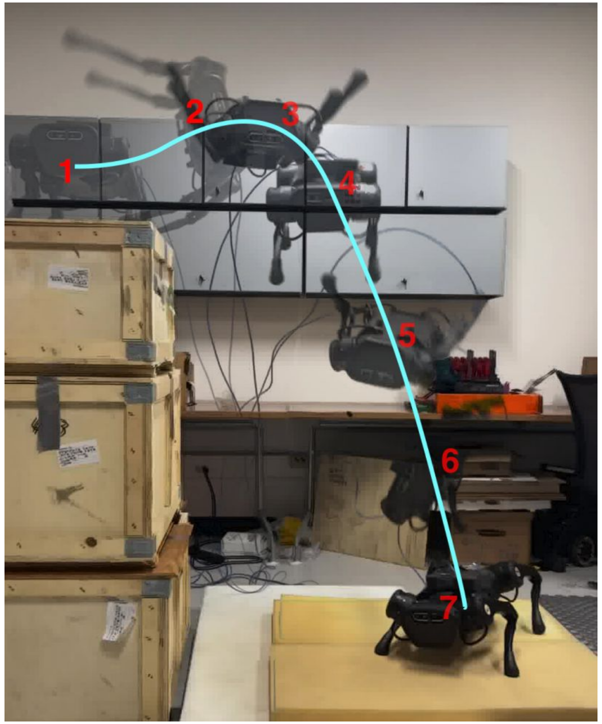
#1: A,B totally controllable
#2: Vector m and columns of B must be linearly independent

Danylo Malyuta and Taylor P. Reynolds and Michael Szmuk and Thomas Lew and Riccardo Bonalli and Marco Pavone and Behcet Acikmese, 2021 arXiv 2106.09125, Convex Optimization for Trajectory Generation

Contact-timing and Trajectory Optimization for 3D Jumping on Quadruped Robots

Summary / Relevance:

- Jumping trajectory optimization through optimized contact scheduling
  - No manual tuning of contact times
  - Timing and actuator effort are coupled



Myron Phan Mustafa Shaikh Prathamesh Saraf

UC San Diego

Contact Timing Optimization Formulation

- Upper and lower bound total time
- Time decision variable embedded in rotation propagation
- Generates reference trajectory and optimal timings

$$\underset{\boldsymbol{x},\boldsymbol{f},\boldsymbol{e}_R}{\text{minimize}} \sum_{k=1}^{N^c} \epsilon_\Omega \boldsymbol{\Omega}_k^T \boldsymbol{\Omega}_k + \epsilon_f \boldsymbol{f}_k^T \boldsymbol{f}_k + \epsilon_R \boldsymbol{e}_{R_k}^T \boldsymbol{e}_{R_k}$$

$$\boldsymbol{x} := \left[\boldsymbol{p}; \dot{\boldsymbol{p}}; \boldsymbol{\Omega}; \dot{\boldsymbol{\Omega}}; \boldsymbol{R}\right] \qquad \boldsymbol{e}_{R_k} = log(\boldsymbol{R}_{ref,k}^T.\boldsymbol{R}_k)^\vee$$

$$\text{s.t. } [\boldsymbol{R},\boldsymbol{p},\boldsymbol{p}_f^s](k=1) = [\boldsymbol{R}_0,\boldsymbol{p}_0,\boldsymbol{p}_{f,0}^s], \quad \textit{initial states} \tag{2a}$$

$$[\boldsymbol{\Omega},\dot{\boldsymbol{\Omega}},\dot{\boldsymbol{p}}](k=1) = \boldsymbol{0}, \quad \textit{initial states} \tag{2b}$$

$$[\boldsymbol{R},\boldsymbol{p},\boldsymbol{p}_f^s](k=N^c) = [\boldsymbol{R}_g,\boldsymbol{p}_g,\boldsymbol{p}_{f,g}^s], \textit{final states} \tag{2c}$$

$$|\boldsymbol{R}(k)[\boldsymbol{p}_f^s(k) - \boldsymbol{p}(k)] - \bar{\boldsymbol{p}}_f^s(k)| \leq \mathbf{r}, \textit{foot position} \tag{2d}$$

$$|\boldsymbol{f}_k^{s,x}/\boldsymbol{f}_k^{s,z}| \leq \mu, |\boldsymbol{f}_k^{s,y}/\boldsymbol{f}_k^{s,z}| \leq \mu, \quad \textit{friction cone} \tag{2e}$$

$$\boldsymbol{f}_{k,min}^s \leq \boldsymbol{f}_k^{s,z} \leq \boldsymbol{f}_{k,max}^s, \quad \textit{GRF limits} \tag{2f}$$

$$\boldsymbol{p}_{k,min} \leq \boldsymbol{p}_k \leq \boldsymbol{p}_{k,max}, \quad \textit{CoM in contact phases} \tag{2g}$$

$$\gamma(\boldsymbol{x}_k, \boldsymbol{x}_{k+1}, \boldsymbol{f}(k), \boldsymbol{p}_f^s(k)) = 0, \quad \textit{discrete dynamic} \tag{2h}$$

$$\boldsymbol{R}_{k+1} = \boldsymbol{R}_k exp(\widehat{\boldsymbol{\Omega}_k T_i/N_i}), \quad \textit{SO(3) manifold} \tag{2i}$$

$$\sum_{i=1}^{n_p} T_i \in [T_{min}, T_{max}], \; N^c = \sum_{i=1}^{n_p} N_i \tag{2j}$$

$$\text{for} \quad k = 1, 2, \ldots N^c$$

Myron Phan Mustafa Shaikh Prathamesh Saraf

Trajectory Optimization Formulation

- Error term for tracking and final configuration
- Actuator effort
- Reference trajectory and optimal timings from last problem are used here

$$J = \sum_{h=1}^{N-1} \epsilon_q (\boldsymbol{q}_h - \boldsymbol{q}_{ref,h})^T (\boldsymbol{q}_h - \boldsymbol{q}_{ref,h}) + \epsilon_\tau \boldsymbol{\tau}_h^T \boldsymbol{\tau}_h$$
$$+ \epsilon_N (\boldsymbol{q}_N - \boldsymbol{q}_N^d)^T (\boldsymbol{q}_N - \boldsymbol{q}_N^d),$$

- Full-body dynamics constraints (3)
- Initial configuration: $\boldsymbol{q}(h=0) = \boldsymbol{q}_0, \dot{\boldsymbol{q}}(h=0) = \boldsymbol{0}$.
- Pre-landing configuration: $\boldsymbol{q}_{j,h} = \boldsymbol{q}_{h,N}^d, \dot{\boldsymbol{q}}_{j,h} = \boldsymbol{0}$ ($h \in [(N-a) : N]$), where a is a jumping task-based selection ($a < N$).
- Final configuration: $\boldsymbol{q}_h(h=N) = \boldsymbol{q}_N^d$.
- Joint angle constraints: $\boldsymbol{q}_{j,min} \le \boldsymbol{q}_{j,h} \le \boldsymbol{q}_{j,max}$
- Joint velocity constraints: $|\dot{\boldsymbol{q}}_{j,h}| \le \dot{\boldsymbol{q}}_{j,max}$
- Joint torque constraints: $|\boldsymbol{\tau}_h| \le \boldsymbol{\tau}_{max}$
- Friction cone limits: $|\boldsymbol{F}_h^x / \boldsymbol{F}_h^z| \le \mu$, $|\boldsymbol{F}_h^y / \boldsymbol{F}_h^z| \le \mu$
- Minimum GRF: $\boldsymbol{F}_h^z \ge \boldsymbol{F}_{min}^z$

UC San Diego

Advantages, Limitations and unanswered questions

Advantages:

- No need to manually tune contact timings

Disadvantages:

- Simplified Model
- Offline

Unanswered Questions / future works:

- Implement vision

Double Barrel Roll

Myron Phan Mustafa Shaikh Prathamesh Saraf

UC San Diego

An Optimal Motion Planning Framework for Quadruped Jumping

Summary / Relevance:

- Heuristic based optimization, works well on very nonlinear objectives
- Convergence quality is superior to gradient based methods

Myron Phan Mustafa Shaikh Prathamesh Saraf

Optimization Formulation

$$\min_{\boldsymbol{x}} \quad \boldsymbol{Fitness}(\boldsymbol{x})$$

$$\text{s.t.} \quad \begin{cases} \boldsymbol{x}(t_{k+1}) = \boldsymbol{x}(t_k) + \Delta t \dot{\boldsymbol{x}}(t_k) \\ \dot{\boldsymbol{x}}_{k+1} = f(\boldsymbol{u}_k, \boldsymbol{x}_k, \boldsymbol{p}_k) \\ \boldsymbol{x}_k \in \mathbb{X}, k = 1, 2, \cdots, N \\ \boldsymbol{u}_k \in \mathbb{U}, k = 1, 2, \cdots, N-1 \\ \boldsymbol{x}(t_0) = \boldsymbol{x}_0, \boldsymbol{x}(t_{end}) = \boldsymbol{x}_{end}, \end{cases}$$

Myron Phan Mustafa Shaikh Prathamesh Saraf

UC San Diego

Optimization Variables

- Jumping trajectories can be realized by force profiles of each feet.
  - Represent each force profile as a time varying polynomial
  - Optimize the coefficients

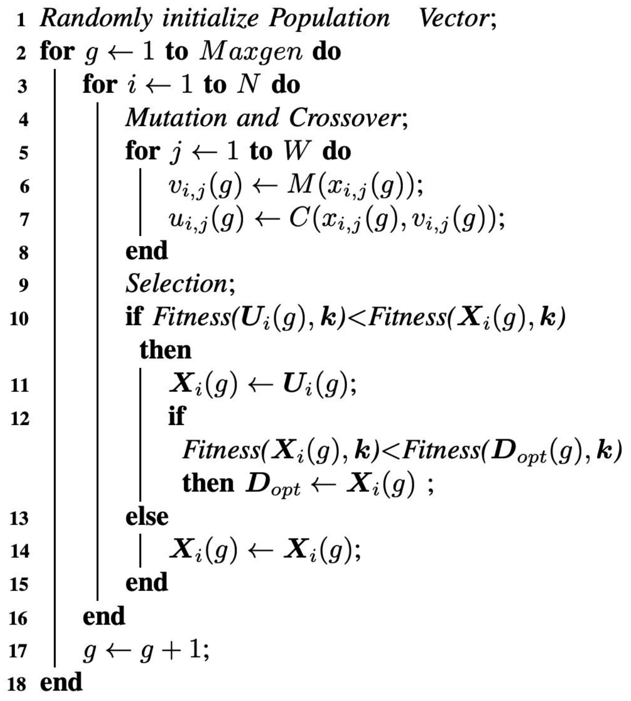
$$\boldsymbol{f}_i = \begin{cases} \boldsymbol{\eta}_1 \boldsymbol{\lambda}_1 [\boldsymbol{t} \quad 1]^T & \boldsymbol{t} \in [0, T_1] \\ \boldsymbol{\eta}_2 \boldsymbol{\lambda}_2 [\boldsymbol{t}^2 \quad \boldsymbol{t} \quad 1]^T & \boldsymbol{t} \in [T_1, T_2] \\ 0 & \boldsymbol{t} \in [T_2, T_3] \end{cases}$$

$$\boldsymbol{D}^*_{opt} := [\boldsymbol{T}_p, \quad \boldsymbol{x}(T_1), \quad \dot{\boldsymbol{x}}(T_3)] \in \mathbb{R}^{12}$$

Myron Phan Mustafa Shaikh Prathamesh Saraf

Differential Evolution Algorithm

- Similar to particle filter
  - Initialize population vector
  - Mutation
  - Survival of fittest
- Fitness Heuristic
  - Proportional to constraints violated



```
Randomly initialize Population Vector;
for $g \leftarrow 1$ to $Maxgen$ do
    for $i \leftarrow 1$ to $N$ do
        Mutation and Crossover;
        for $j \leftarrow 1$ to $W$ do
            $v_{i,j}(g) \leftarrow M(x_{i,j}(g))$;
            $u_{i,j}(g) \leftarrow C(x_{i,j}(g), v_{i,j}(g))$;
        end
        Selection;
        if Fitness($\boldsymbol{U}_i(g), \boldsymbol{k}$)<Fitness($\boldsymbol{X}_i(g), \boldsymbol{k}$) then
            $\boldsymbol{X}_i(g) \leftarrow \boldsymbol{U}_i(g)$;
            if Fitness($\boldsymbol{X}_i(g), \boldsymbol{k}$)<Fitness($\boldsymbol{D}_{opt}(g), \boldsymbol{k}$) then $\boldsymbol{D}_{opt} \leftarrow \boldsymbol{X}_i(g)$ ;
        else
            $\boldsymbol{X}_i(g) \leftarrow \boldsymbol{X}_i(g)$;
        end
    end
    $g \leftarrow g + 1$;
end
```

Myron Phan Mustafa Shaikh Prathamesh Saraf

Constraints

- Kinematic
- Kino-dynamic
  - Physics
  - Obstacles

$$\boldsymbol{x}(t_{k+1}) = \boldsymbol{x}(t_k) + \Delta t \dot{\boldsymbol{x}}(t_k)$$
$$\dot{\boldsymbol{x}}_{k+1} = f(\boldsymbol{u}_k, \boldsymbol{x}_k, \boldsymbol{p}_k)$$
$$\boldsymbol{x}_k \in \mathbb{X}, k = 1, 2, \cdots, N$$
$$\boldsymbol{u}_k \in \mathbb{U}, k = 1, 2, \cdots, N-1$$
$$\boldsymbol{x}(t_0) = \boldsymbol{x}_0, \boldsymbol{x}(t_{end}) = \boldsymbol{x}_{end},$$

Contact Force: $\boldsymbol{f}_{iz} > \boldsymbol{f}_{zmin}$.

Friction Cone: $|\boldsymbol{f}_{i,xy}/\boldsymbol{f}_{i,z}| < \mu$.

Joint Angle: $\boldsymbol{q}_{max} > \boldsymbol{q}_{ij} > \boldsymbol{q}_{min}$.

Joint Velocity: $|\dot{\boldsymbol{q}}_{ij}| < \dot{q}_{max}$.

Joint Torque: $|\boldsymbol{\tau}_{ij}| < \boldsymbol{\tau}_{max}$.

Joint Position: $\boldsymbol{z}_{ij} > \boldsymbol{z}_{min}, j \neq 2$.

Obstacle Avoidance: $\boldsymbol{O}_{az} > \boldsymbol{z}_{ij}(t_k) > \boldsymbol{O}_{bz}$.

UC San Diego

Advantages, Limitations and unanswered questions

Advantages:

- Robust against nonlinear objective functions

Disadvantages:

- Has to be pre-trained to run online

Unanswered Questions / future works:

- Doesn’t take landing accuracy into account

Myron Phan Mustafa Shaikh Prathamesh Saraf

UC San Diego

Conclusion

- Convex optimization plays an important role in the control and stability of legged robots
- ZMP -> LQR -> Linear MPC -> Non-linear MPC -> SOCP
- Linear MPC (QP) remain the most efficient till date
- Many optimization problems (mostly QPs in the current literature) can be formulated as SOCPs and solved using efficient algorithms
- Optimization techniques (least squares, heuristics) can be applied to jumping to optimize for trajectory, actuator effort, contact timings, etc.

Thank You!

Myron Phan Mustafa Shaikh Prathamesh Saraf