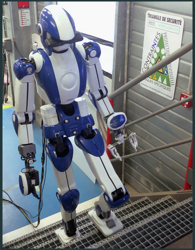
Ideas and software for the locomotion of homemade robots

Stéphane Caron

June 20, 2022



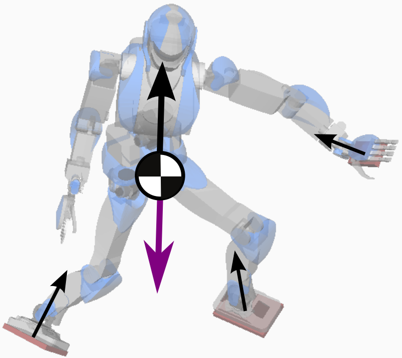
The Good

the Quad

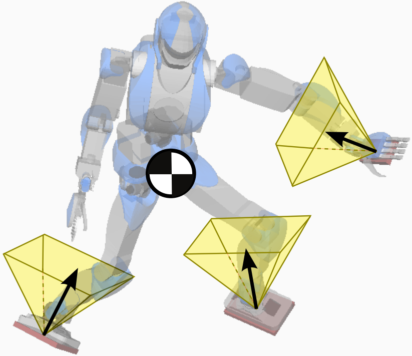
and the Upkie

Two ideas from locomotion

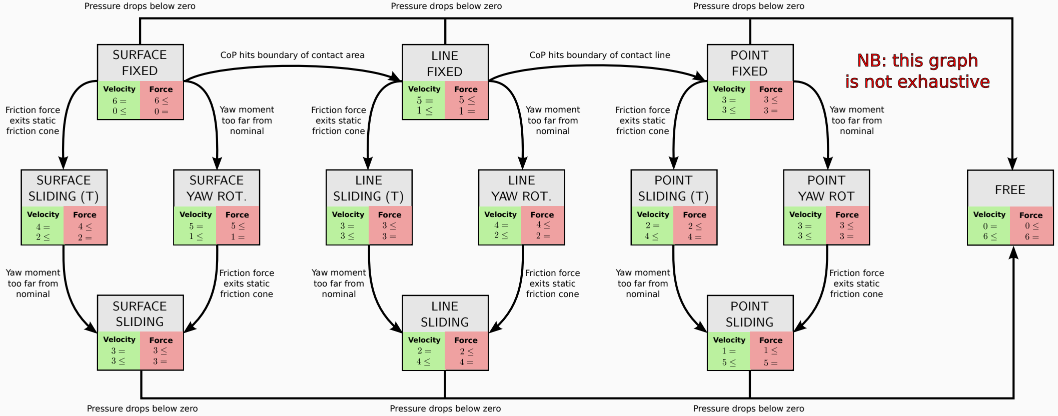
Contacts



Friction cones



Contact modes



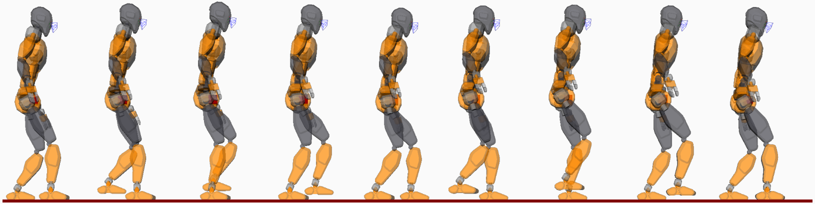
Contact stability

Contact stability $:=$ maintaining a given contact mode

Idea #1

All our motions alternate contact-stable phases

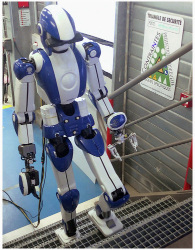
Idea #1



DSP	SSP for right foot swing	DSP	SSP for left foot swing	DSP

Idea #1

Can you think of a counter-example?



Surface

Point

Wheel

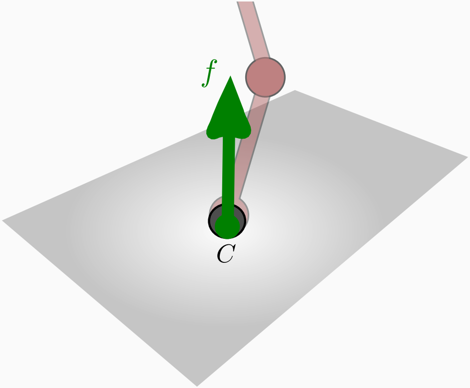
The Quad: fixed point contact

1. **End-effector velocity:**

$$^{G}v_C = 0$$

2. **Friction cone:**

$$\|f^{xy}\| \leq \mu f^z \qquad f^z \geq 0$$



G: ground frame, C: contact frame (no superscript $\Rightarrow$ that frame)

1. **End-effector velocity:**

$$ {}^{G}v_C = 0 $$

2. **Friction cone:**

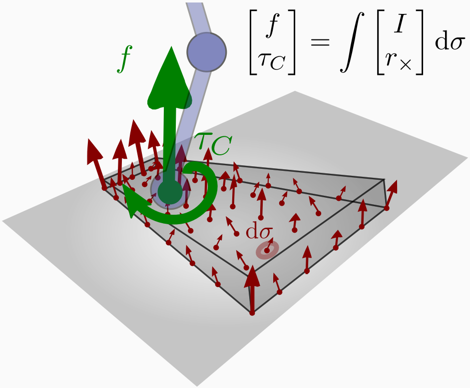
$$ \|f^{xy}\| \leq \mu f^z \qquad f^z \geq 0 $$

3. **Center of pressure area:**

$$ Ap_{cop} \leq b $$

4. **Yaw slippage interval:**

$$ |\tau^z_{cop} - \tau^z_{safe}| \leq \mu f^z d_{edge}(p_{cop}) $$



Derivation by Fourier-Motzkin: https://arxiv.org/pdf/1501.04719.pdf

The Upkie: rolling-without-slipping line contact

1. **End-effector velocity:**

$$ {}^{C}v_C^{yz} = 0 \qquad {}^{C}v_C^{x} = v_{cmd} $$

2. **Friction cone:**

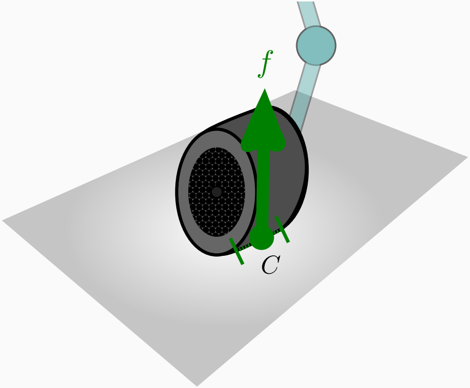
$$ \|f^{xy}\| \leq \mu f^z \qquad f^z \geq 0 $$

3. **Center of pressure interval:**

$$ |p_{cop}^y| \leq \ell_{wheel} $$

4. **Yaw slippage interval:**

$$ |\tau_{cop}^z - \tau_{safe}^z| \leq \mu f^z (\ell_{wheel} - |p_{cop}^y|) $$



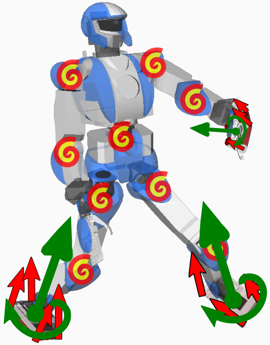
Derivation *e.g.* as limit of the surface condition.

How to enforce contact stability while moving?

Whole-body impedance control

Quadratic program on **torques** to track positions:

- Equations of motion
- *Contact stability constraints* $\leftarrow$
- Joint position limits (predictive)
- Joint velocity limits (predictive)
- Joint acceleration/torque limits



Widely reproduced, here is one: `https://arxiv.org/pdf/1607.08089.pdf`

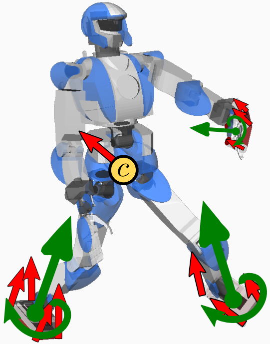
Whole-body admittance control

Quadratic program on **positions** to track forces:

- ~~Equations of motion~~
- CoP tracking at each contact surface
- Internal force tracking between contacts[1]

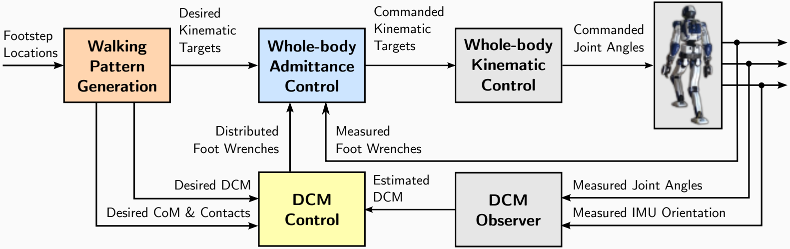
Quadratic program on **force** targets:

- Equations of motion
- Contact stability constraints



[1] General formulation in (21) of `https://doi.org/10.1109/IROS40897.2019.8968059`

Use in walking a position-controlled humanoid



Technical report: `https://arxiv.org/pdf/1809.07073.pdf`

Impedance or admittance?

Joint control	Sensors	State observation	Whole-body control
Position	Force sensors	Lighter	Admittance
Torque	Joint torques	Heavier	Impedance

Impedance or admittance?

Have we tried everything?

New robots to revisit these ideas

Handle

ANYmal on Wheels

Ascento

Whole-body control of Ascento: `https://arxiv.org/pdf/2005.11431.pdf`

Impedance or admittance?

Joint control	Sensors	State observation	Whole-body control	Who?
Position	Force sensors	Lightest	Admittance	The Good
Torque	Joint torques	Heaviest	Impedance	The Quad
Position	Joint torques	Middle ground	Hybrid	The Upkie
⋮	⋮	⋮	⋮	⋮

Idea #2

Low-frequency whole-body control

Idea #2

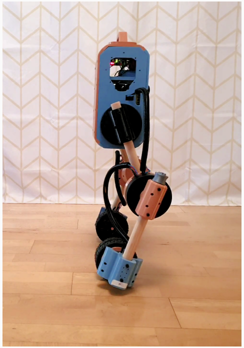
- Position control frequency $\ll$ torque control frequency
- Velocity control frequency $\ll$ torque control frequency
- Balancing is a low frequency task[2]

[2] Both theoretical and empirical evidence: `https://arxiv.org/pdf/1907.01805` $\leftarrow$ **remarkable!**

Software for homemade robots

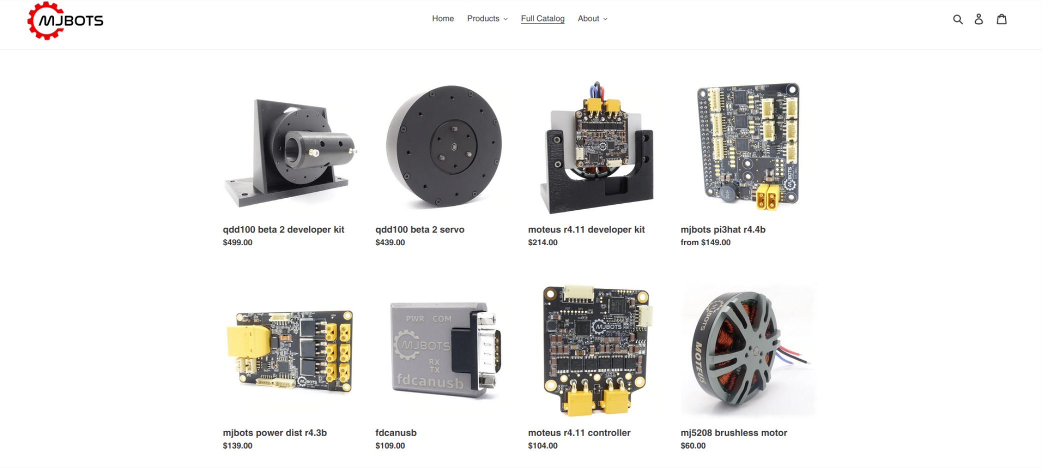
Upkie wheeled biped

- Joints: 6 (hips, knees, wheels)
- Total mass: 5.4 kg
- Print time: 33 h 14 min
- Knee torques: 2.0 Nm crouched
- Wheel torques: $0.2 + f(\alpha)$ Nm
- Autonomy: 3–4 h with 5.0 Ah battery
- Actuators + electronics: 2,400 €



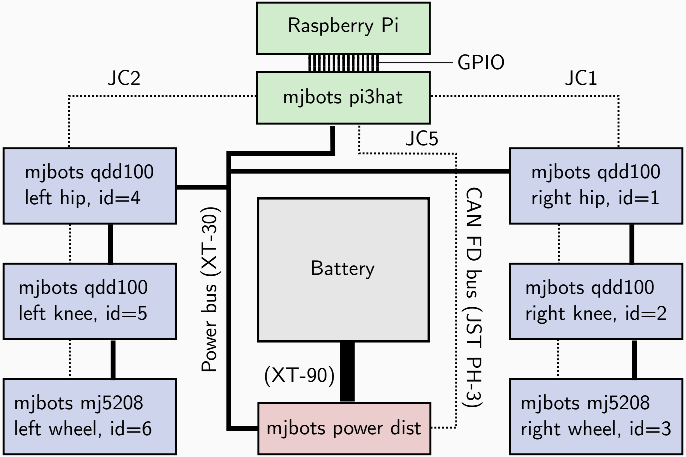
Project page: `https://hackaday.io/project/185729-upkie-homemade-wheeled-biped-robot`

Actuators

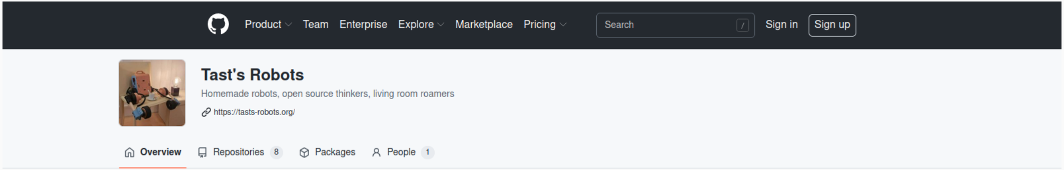


Store: `https://mjbots.com/`

Electronics



Software



Libraries for incremental buy-in (fear framework lock-in)

- *vulp:* real-time motion control
- *pink:* inverse kinematics, based on Pinocchio
- *upkie_locomotion:* locomotion agents and spines for Upkie
- *ltv-mpc:* linear time-variant model predictive control

Repositories: `https://github.com/tasts-robots`

Real-time motion control on Raspberry Pi

- Balance control in Python at low frequency
- Convert Python actions to joint commands in C++ at medium frequency
- Actuators run field-oriented control on-board at high frequency

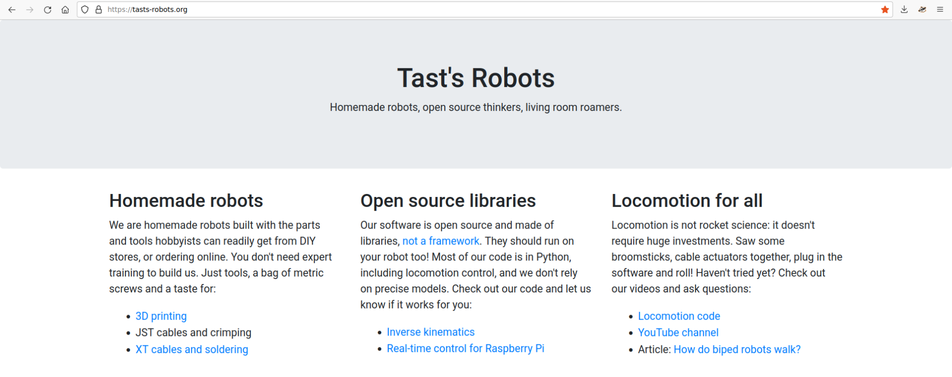
Try it out!

```
git clone https://github.com/tasts-robots/upkie_locomotion.git
cd upkie_locomotion
./tools/bazelisk run -c opt //agents/blue_balancer:bullet
```

Connect a USB controller to move the robot around. 🎮

Repository: `https://github.com/tasts-robots/vulp`

How can we share the fruits of homemade robots?



https://tasts-robots.org

Tast's Robots

Homemade robots, open source thinkers, living room roamers.

Homemade robots

We are homemade robots built with the parts and tools hobbyists can readily get from DIY stores, or ordering online. You don't need expert training to build us. Just tools, a bag of metric screws and a taste for:

- 3D printing
- JST cables and crimping
- XT cables and soldering

Open source libraries

Our software is open source and made of libraries, not a framework. They should run on your robot too! Most of our code is in Python, including locomotion control, and we don't rely on precise models. Check out our code and let us know if it works for you:

- Inverse kinematics
- Real-time control for Raspberry Pi

Locomotion for all

Locomotion is not rocket science: it doesn't require huge investments. Saw some broomsticks, cable actuators together, plug in the software and roll! Haven't tried yet? Check out our videos and ask questions:

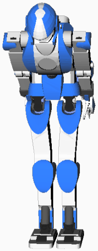
- Locomotion code
- YouTube channel
- Article: How do biped robots walk?

Reproducibility principles: `https://tasts-robots.org/` ← feedback welcome!

Takeaways

- All our motions alternate contact-stable phases
- Whole-body control to enforce contact stability
- **Low-frequency** whole-body control
- How can we share the fruits of homemade robots?
- Software libraries for incremental buy-in

Thank you for your attention!



This revision of the slides includes feedback from Vincent Padois and Grégoire Passault, thanks!