Students: Alessio Desogus, Aurélien Laissy & Théo Lemonnier

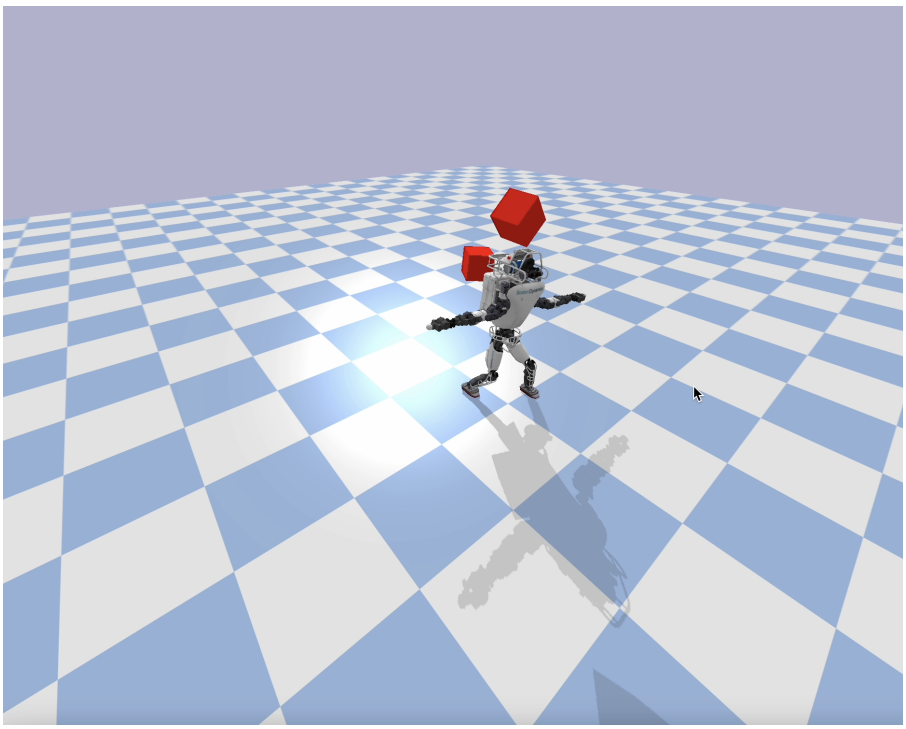


Figure 1: Bipedal Locomotion Control based on Divergent Component of Motion (DCM)

Questions

Question 1

Based on equation (9), which physical parameters will affect the rate of divergence of the DCM dynamics?

Solution

The rate of divergence of the Divergent Component of Motion (DCM) is influenced by several physical parameters, based on equation (9). These include:

- **Natural Frequency** (ω):
  The natural frequency is given by the equation:

$$\omega = \sqrt{\frac{g}{\Delta z}}$$

  where:
  - g is the gravitational acceleration constant.
  - Δz is the height difference between the Center of Mass (CoM) and the ground.

  Since the natural frequency depends on these values, any changes in gravity or the height of the model will affect the rate of divergence. Specifically:
  - A **lower CoM height** (Δz) results in a higher natural frequency ω, which makes the exponential term $\exp(\omega t)$ in equation (9) increase faster. This leads to faster divergence of the DCM dynamics. Therefore, to improve stability, the CoM should be kept higher.

- **DCM Offset** $(\xi_0 - \text{cop}_0)$:
  The initial difference between the DCM (ξ_0) and the Center of Pressure (cop_0) plays an important role in the dynamics. If the DCM is initially far from the CoP, the divergence will be faster because this difference is multiplied by the exponential term, which increases over time. Minimizing the DCM offset at foot placement is essential for maintaining balance and reducing divergence.

In conclusion, to control or limit the divergence of the DCM, both:

- Increasing the **initial height** of the DCM (Δz).
- Minimizing the **DCM offset** $(\xi_0 - \text{cop}_0)$.

are critical for ensuring stability.

Question 2

In the optimization-based DCM motion planning, how do we consider the stability of DCM movement?

Solution

In the optimization-based DCM motion planning, we consider the stability of the DCM movement via the minimization of a quadratic cost function which captures the physics of the system and allows the entire framework to be formulated as a Quadratic Programming (QP) problem.

The first terms aim to minimize the distance between the CoP position at the next step and the nominal CoP position.

$$J_1 = \alpha_1 \|\text{cop}_x^T - \text{cop}_{x,\text{nom}}^T\|^2 + \alpha_2 \|\text{cop}_y^T - \text{cop}_{y,\text{nom}}^T\|^2$$

Where:

- cop_x^T and cop_y^T are the x and y coordinates of the CoP at the next step.
- $\text{cop}_{x,\text{nom}}^T$ and $\text{cop}_{y,\text{nom}}^T$ are the nominal or desired x and y coordinates of the CoP.
- α_1 and α_2 are positive weighting factors that penalize the deviations between the actual and desired CoP positions.

The second terms aim to minimize the distance between the DCM offset at the next step and the nominal DCM offset.

$$J_2 = \alpha_3 \|\gamma_{T,x} - \gamma_{\text{nom},x}\|^2 + \alpha_4 \|\gamma_{T,y} - \gamma_{\text{nom},y}\|^2$$

Where:

- $\gamma_{T,x}$ and $\gamma_{T,y}$ are the x and y components of the DCM offset at the next step.
- $\gamma_{\text{nom},x}$ and $\gamma_{\text{nom},y}$ are the nominal or desired DCM offsets.
- α_3 and α_4 are weighting factors that penalize the DCM offset deviation from the desired values.

The last term is the timing of the next step, represented by T, it also influences the stability of the DCM.

$$J = \alpha_5 \|\sigma - e^{\omega T_{\text{nom}}}\|^2$$

Where:

- σ is related to the step timing through the relation $\sigma = e^{\omega T}$.
- T_{nom} is the nominal or desired step timing.
- α_5 is a weighting factor that penalizes the deviation from the desired step timing.

The problem is formulated with both equality and inequality constraints. The equality constraint equation (Ax = b): $\gamma_T + cop_T + (cop_0 - \xi_0)\sigma = cop_0$ ensures the DCM trajectory satisfies both initial and final conditions, it describes the expression of the DCM. The inequality constraints ensure the solution remains within physically feasible bounds for step length and duration, we want to keep the CoP, exponential term and DCM offsets in a range that is physically feasible.

Question 3

You may notice in the code that we have a threshold for activating the use of the real DCM in optimization. If the error between the real DCM and the desired DCM is below a certain value, we ignore the error and use the desired DCM. But why does an error still exist between the real and desired DCM, even when there is no external push?

Solution

In normal walking, a small error between the real and desired DCM is natural and necessary because walking involves controlled instability, essentially "falling" forward and stabilizing with each step. The robot doesn't need to immediately correct small DCM deviations since they are part of the planned motion. However, when an external push occurs, the robot's balance is disrupted, and the desired DCM is no longer reliable. In this case, the real DCM must be used to accurately measure the robot's instability and react in real-time to prevent a fall, making it crucial for stabilization in such scenarios.

Question 4

What other balance recovery strategies can be used in addition to the stepping strategy? (Think about your balance recovery strategies, how do you recover from external disturbances in different situations?)

Solution

Three bio-inspired strategies have been predominantly used for balance recovery in bipedal robots: the **ankle strategy**, **hip strategy** and **stepping strategy** [1]. These strategies are directly inspired by our balance recovery behaviors observed in daily situations. The ankle strategy, which involves generating torque at the ankle joint, is typically used for small disturbances and is inspired by how we maintain balance during subtle perturbations while standing. When larger disturbances occur, we naturally engage the hip strategy by rapidly moving their hips to generate counter-momentum, particularly effective when standing on narrow surfaces where ankle torque alone is insufficient. Finally, the stepping strategy, which involves taking one or more steps to recover balance, is inspired by how we respond to significant perturbations that cannot be handled by ankle or hip strategies alone (which we explained in question 3).

In practice, we unconsciously combine these strategies based on the magnitude and direction of the disturbance. For instance, when stumbling on uneven pavement, we might first employ the ankle strategy for small adjustments, transition to the hip strategy if the disturbance is larger, and finally resort to the stepping strategy if the first two prove insufficient. Similarly, when being pushed while standing, we naturally progress through these strategies in order of increasing necessity, from ankle to hip to stepping, depending on the severity of the disturbance.

Question 5

What's the reason that we use the DCM equations in the optimization for balance recovery? Why we do not use the whole dynamic equation of the robot to solve the control problem?

Solution

The DCM equations are used in optimization for balance recovery for several key reasons related to simplicity, computational efficiency, and effectiveness in controlling balance dynamics, particularly during walking or push recovery. Below are the main reasons for using DCM equations instead of the full dynamic equations of the robot:

1. **Simplification of dynamics and computational efficiency:**
   The DCM equations simplify the full-body dynamics by focusing on the motion of the CoM relative to the CoP. The full dynamic equations would include all forces and torques acting on each joint and body segment, leading to unecessarily complex equations. Optimizing based on full dynamic equations would require solving complex systems in real time, which is computationally expensive. DCM equations are linear and can be solved efficiently using QP, making real-time control feasible for dynamic movements like walking and push recovery.
2. **DCM captures essential dynamics and focuses on stability control:** The DCM is derived from the LIPM, which assumes constant CoM height and negligible angular momentum. This simplification is sufficient to capture the key aspect of walking stability. The divergence of the DCM indicates instability in the system. By focusing on controlling the DCM, balance recovery can be achieved. The DCM gives clear control objectives, such as minimizing the DCM offset to maintain stability. It also directly relates to control factors like step timing and foot placement Eventually, since the CoM dynamics can be decomposed into stable and unstable components, with the unstable component represented by the DCM. Since instability is the main threat to balance, controlling the DCM suffices for maintaining stability.

Question 6

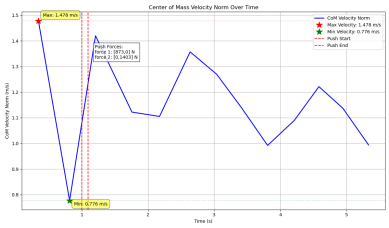
Adjust various parameters such as nominal parameters and the boundaries of the optimization variables, then report the maximum and minimum speeds at which the robot can achieve stable push recovery. During this process, you may need to modify the timing or magnitude of the external push to maintain stability. If any changes to the push are made, please provide detailed information on the push parameters, including its timing, magnitude, and direction.

Solution

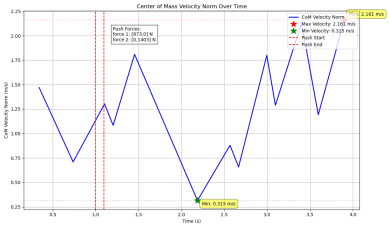
	MaxVel (m/s)	MinVel (m/s)	StepLen (m)	StepWid (m)	StepDur (s)	DeltaMin (s)	DeltaMax (s)
1	1.478	0.776	0.48	0.20	0.41	0.3	1.3
2	**2.161**	**0.315**	**0.48**	**0.20**	**0.4**	**0.3**	**1.3**
3	1.523	0.601	0.48	0.13	0.4	0.3	1.3
4	1.411	0.857	0.50	0.10	0.4	0.3	1.3
5	1.427	0.566	0.48	0.10	0.4	0.3	1.3
6	**1.570**	**0.858**	**0.50**	**0.10**	**0.38**	**0.3**	**0.3**

Table 1: Velocity and Step Parameters

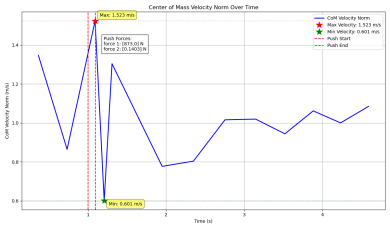
We ran multiple simulations and decided to keep some that are interesting to understand the nominal parameters impact on the maximum and minimum velocities of atlas. In this section we kept the same push for all the 6 tests displayed ie : Force1 = [873, 0, 0], and Force2 = [0, 1403, 0].



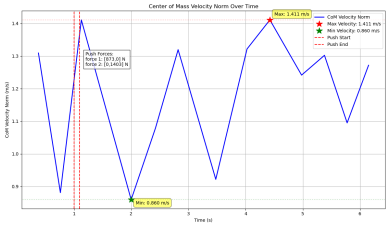
(a) Test 1



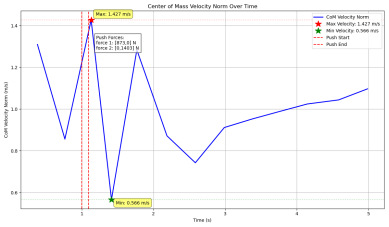
(b) Test 2 : fastest and slowest



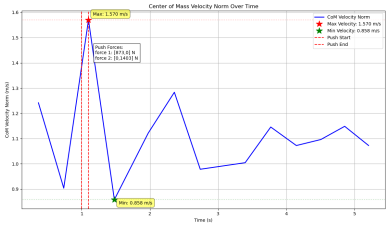
(c) Test 3



(d) Test 4

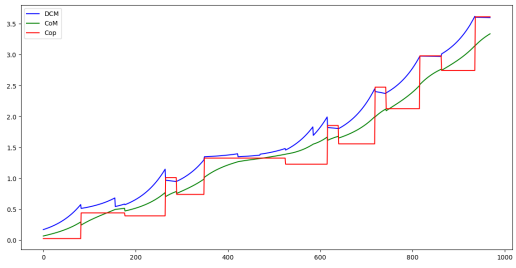


(e) Test 5

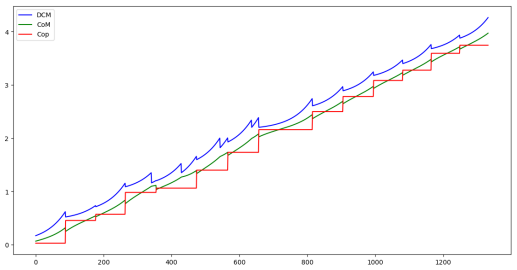


(f) Test 6 : most robust

Figure 2: Plots of the Norm of the COM velocity vs time



(a) Test 2: DCM, COM, COP



(b) Test 6: DCM, COM, COP

Figure 3: The x coordinate of DCM, COM, COP with respect to time

We can see the 6 plots of the norm of the COM Velocity Norm as well as when Atlas get hit by the blocks to see its resilience to disturbances. The videos of the experiment are available and named *initial_push_recovery.mov*, *fastest_and_slowest.mov* and *most_robust.mov*.

First we opted to detect the impact of the step duration. We observed that decreasing StepDur slightly by 0.01 (from test 1 to test 2) affected the min and max velocities a lot. In fact we see that the robot is almost about to stop when hit by the blocks (v =0.3 m/s) and then it accelerates to a velocity of 2.161 m/s which is the fastest encountered in our simulations. This may be because due to smaller ground contact time the robot is able to transmit the same power in less time to the ground when in contact.

Then we decided to decrease the step width between case 2 and 3, it gives us an increase in the minimum velocity and a decrease of the max velocity. Decreasing the step width increases the minimum velocity because the robot needs to move faster to stay balanced with a smaller support area. At the same time, it decreases the maximum velocity since the reduced lateral stability limits the robot's ability to take large steps, restricting its overall speed.

Then we wanted to see the impact between two different StepLen during test 4 and 5. We found that decreasing it would decrease the minimum velocity as well, ie the robot is closer to null velocity after getting hit when the step length is smaller.

Comparison between test 2 and test 6 (3): On one hand the COP (in x) of Test 2 is not non decreasing, therefore we can say that the robot center of pressure is sometimes going backward which is not very robust. On the other hand for Test 6, the COP is non decreasing, the steps are quite constant before

and after the push, during the push phase it stays non decreasing which seems quite robust.

(Bonus) Question 7

Adjust the parameters and increase the magnitude of the applied push to trigger a sequence of lateral recovery steps. The goal is to simulate a stronger disturbance that requires the system to take multiple corrective steps sideways to regain balance.

Solution

We tried various combinations of parameters without really finding any very satisfactory ones, the best reaction being two side steps (see the video named *side_steps.mov*). To tune the parameters we first increased the push magnitude in the lateral direction to make a single recovery step insufficient. We also increased the lateral step to allow the robot to take larger lateral steps and thus achieve faster recovery for the lateral push. Among other things, we also tried to reduce the step time to enable the robot to take multiple steps to the side more quickly, so as to improve its equilibrium in the face of large disturbances.

References

[1] Milad Shafiee-Ashtiani, Aghil Yousefi-Koma, Masoud Shariat-Panahi, and Majid Khadiv. Push recovery of a humanoid robot based on model predictive control and capture point. In *2016 4th International Conference on Robotics and Mechatronics (ICROM)*, pages 433–438, 2016.