FIELD AND SERVICE ROBOTICS

HOMEWORK 4

Student:
Paolo Maisto
P38000191

Anno Accademico 2023/2024

The entire folder also include the videos is loaded in the following github repository:
https://github.com/PaoloMaisto/HW_FSR_2024/tree/main/HW_4/FSR_HOMEWORK_4_202

Answer 1:

When a body is partially or completely immersed in a fluid (in the case of underwater robotics, so we consider water as fluid), it is not only subject to gravity but also to another force called buoyancy, which is a hydrostatic effect that is not a function of the relative motion between the body and the fluid, as stated by Archimedes' principle. Summarizing, the forces to which it is subject are the following:

- The force of gravity $\boldsymbol{f_g^b}$ acting at the center of mass and directed downwards;
- The buoyant force $\boldsymbol{f_b^b}$ acting at the center of buoyancy and directed upwards.

$$f_g^b = R_b^T \begin{bmatrix} 0 \\ 0 \\ w \end{bmatrix} = R_b^T \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} \qquad f_b^b = -R_b^T \begin{bmatrix} 0 \\ 0 \\ b \end{bmatrix} = -R^T \begin{bmatrix} 0 \\ 0 \\ \rho \Delta g \end{bmatrix} \tag{1}$$

As the formulas show, while the first depends on the weight of the submerged body, the second depends on $\boldsymbol{b}$, where $\boldsymbol{\rho}$ is the density of the fluid, $\boldsymbol{\Delta}$ is the volume of the submerged part of the body, while $\boldsymbol{g} = [0\ 0\ g]^T$ is the acceleration due to gravity expressed in vector form. These two forces are often considered together as a single contribution and are defined as restoring forces $\boldsymbol{g_{rb}^b}$. Therefore, due to gravity and buoyancy, the wrench expressed in the body-fixed frame is as follows:

$$g_{rb}^b = -\begin{bmatrix} f_g^b + f_b^b \\ S(r_c^b) f_g^b + S(r_b^b) f_b^b \end{bmatrix} \tag{2}$$

where $\boldsymbol{r_c^b}$ is the position of the center of mass expressed in the body frame, whereas $\boldsymbol{r_b^b}$ is the position of the center of buoyancy expressed in the body frame. From this formula, it can be deduced that if the two centers where the forces are applied are not along the same vertical line, this causes a torque that leads the robot to rotate due to this mechanical issue or due to geometric construction or constraint. Obviously, if they are aligned along the same vertical line, the robot can only move up or down according to gravity and buoyancy.

Different from the aerial where we have only the gravity, here we have also the buoyancy. We don't consider the buoyancy in the aerial beacuase of the density of the air is lighter of the density of the water and also the density of the robot to respect the density of the air were not comparable, becuase the density of the robot were much higer of the density of the air. Instead, in underwater robots the density of the robot is comparable to the density of the water. And since the density of the robot is comparable to the density of the water the added effect appeared.

Answer 2:

a. The statement is **FALSE**. The added mass effect should not be understood as merely an extra load (mass and inertia) on the structure. Instead, it represents the reaction force generated by the surrounding fluid particles, which accelerate due to the body's movement. This reaction force manifests as additional inertia from the fluid around the body, accelerating along with it. Consequently, when incorporating the added mass effect into a dynamic model, the matrices used lack the symmetrical and positive definite properties characteristic of actual inertia matrices.

b. The statement is **TRUE**. The added mass effect in underwater robotics must be considered, differently from other cases including aerial robotics, since the density of water is comparable to the density of an underwater robot. Unlike in other cases, such as aerial robots, where the density of air is much lighter than the density of water and the robot itself, as a result of not being the two comparable densities this effect is neglected.

c. The statement is **TRUE**. Since from the viscosity of the fluid induces dissipative drag and lift forces on the body, which contribute to the damping effect. That is, in mechanical systems, damping reduces the amplitude of oscillations by dissipating energy. For instance, in a pendulum, the air's viscosity prevents persistent oscillations, leading to asymptotic stability as the pendulum eventually comes to a stop. In the same way the stability proof involves $\dot{V}$, which depends on $\boldsymbol{D_{RB}}$, which is the damping matrix which models both the drag and the lift effect and it's positive definite; so the larger the damping matrix $\boldsymbol{D_{RB}}$ the faster the system's convergence to stability.

d. The statement is **FALSE**. Although the Ocean current is usually considered both constant and irrotational, this assumption is valid only if the ocean current is expressed in the world frame.
Despite adaptive and integral controllers, commonly used in underwater robotics, benefit from the ocean current being constant and irrotational, because they can asymptotically compensate the constant disturbance terms. This compensation is most effective when the disturbance is seen as constant in a reference frame. For this reason, it is preferable to express the ocean current in the fixed world frame.

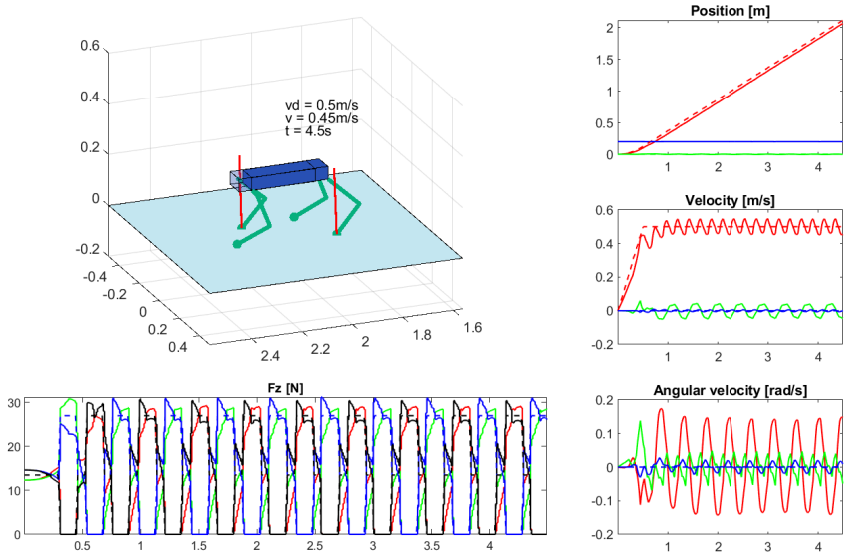
Answer 3:

The provided matlab code is designed to simulate a quadruped robot and manage its control using Model Predictive Control (MPC). It also enables the simulation of various gaits, such as trot, bound, pacing, gallop, trot run, and crawl. The goal is to implement the quadratic function using the QP solver `qpSWIFT`. Specifically, the matrices supplied to `qpSWIFT` as input represent equality constraints (to ensure dynamic consistency and contact constraints) and inequality constraints (to prevent sliding contact, enforce torque limits and manage the swing leg task) that the controller must adhere to. The output, `zval`, includes the desired accelerations and ground reaction forces.

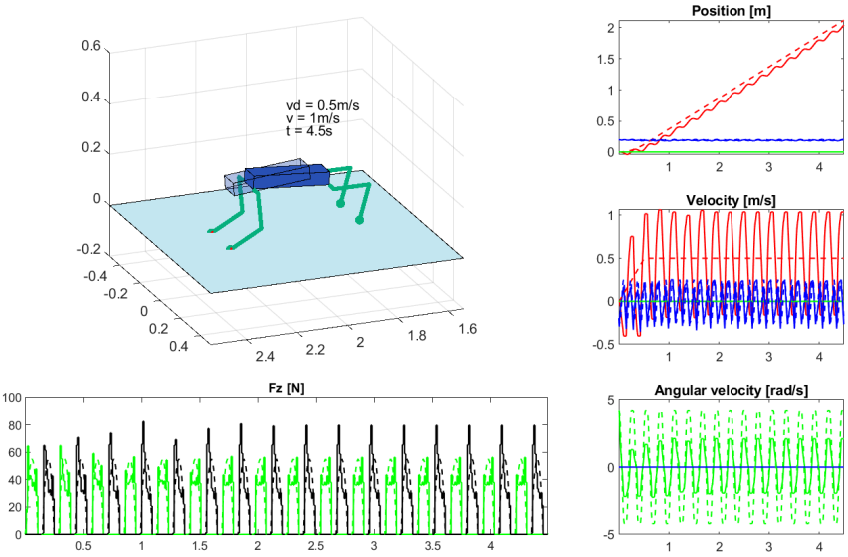
Several simulations were carried out by modifying various parameters: first the parameters in the file *main.m* such as the gait and the desired velocity; then the physical parameters in the file *get_params.m*, as the friction coefficient and the mass of the robot.

Below are presented the various simulations. All simulations performed and reported below have been recorded and uploaded to the *Recordings* folder.

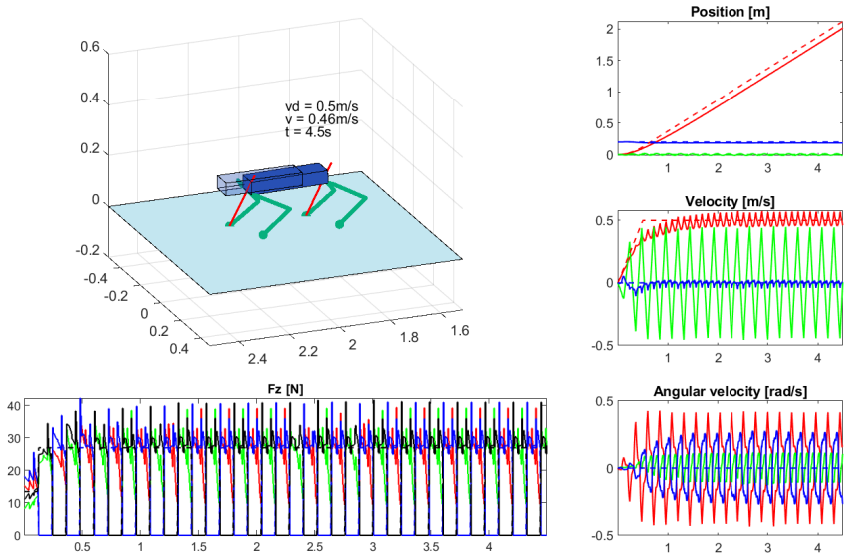
From the next page there are the first six simulations, one for each gait, in which the parameters are not changed; these simulations will then be taken as a reference for the results obtained later in which the parameters will be changed:



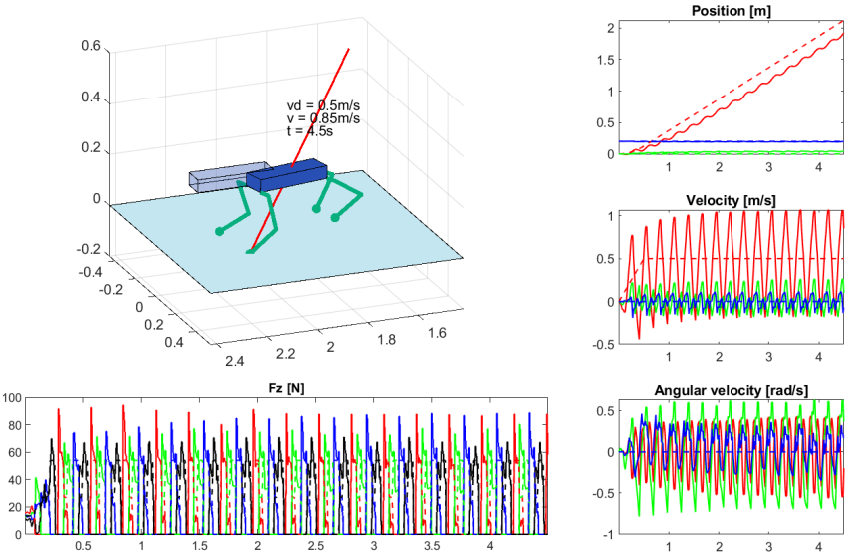
(a) Gait 0 - trot



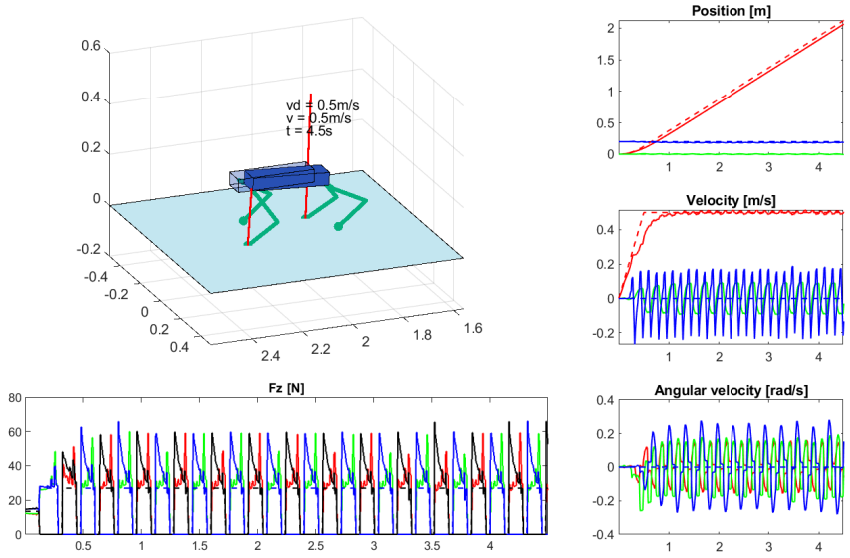
(b) Gait 1 - bound



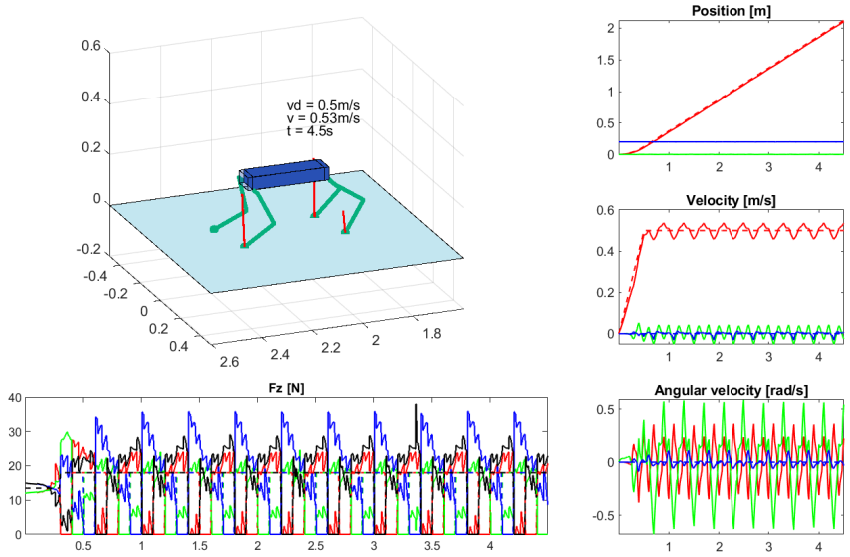
(c) Gait 2 - pacing



(d) Gait 3 - gallop



(e) Gait 4 - trot run



(f) Gait 5 - crawl

Figure 1: Results obtained by modifying only gait.

Figures 1a to 1f reports the results of the simulations conducted with various gaits, all utilizing the nominal parameters. The selected gait determines the leg movement patterns and which legs move in unison.

a. **Gait=0 - trot**, the legs on the same diagonal move simultaneously, with either two or four legs in contact with the ground at any given time.

b. **Gait=1 - bound**, either no legs or two legs touch the ground at once. In this case, the robot alternates between moving its front legs together and its hind legs together.

c. **Gait=2 - pacing**, also involves moving two legs at a time, but here, the legs on the same side of the body move together. There is a brief moment when all four legs touch the ground, but usually, two legs are in the air while the other two remain on the ground.

d. **Gait=3 - gallop**, is perhaps the most intricate and dynamic gait, where periodically one or two legs make ground contact. This gait is highly asymmetrical, with the robot having zero, one, or two feet touching the ground at different moments.

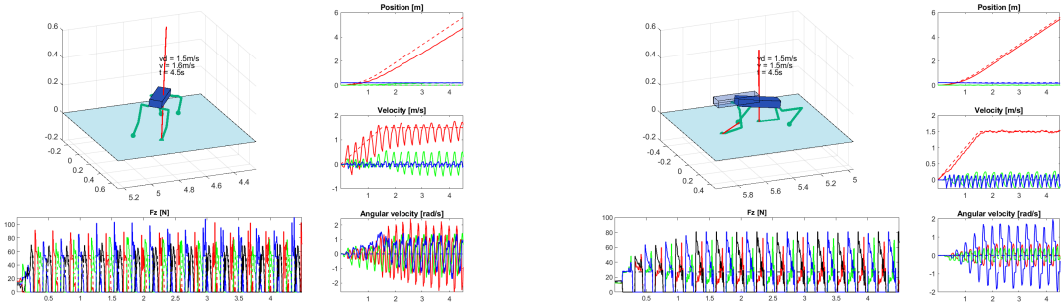
e. **Gait=4 - trot run**, in the same way as the trot (Gait=0), but the robot does not wait for the swing legs to land before moving the other legs, leading to either zero or two legs on the ground and creating a more dynamic motion.

f. **Gait=5 - crawl**, is the only gait offering static stability, as it maintains at least three legs on the ground at all times, forming a stable triangular support polygon.

Figures 1a to 1f and their corresponding recordings demonstrate that with standard parameters, all the gaits perform well, achieving velocities that generally align with the target speed and maintaining minimal position errors, primarily due to the initial acceleration phase. From these preliminary simulations, it is evident that the robot's movement is driven by a limit cycle. Consequently, even with a constant desired velocity, there are oscillations around the target value.

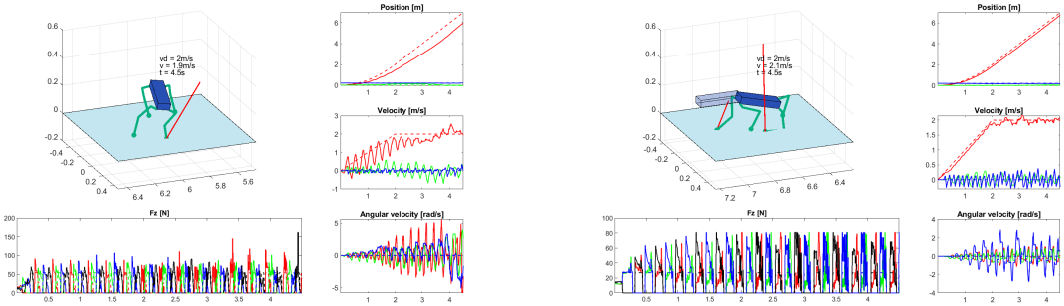
Furthermore, the pacing gait shows the highest angular velocity along the x-axis, while the bound gait displays the highest angular velocity along the y-axis.

Analyzing the ground reaction forces reveals that having more legs in contact with the ground results in lower individual forces, as the load is distributed across more points. This explains why the gallop simulation exhibits the highest reaction forces. Additionally, there is a relationship between the peak velocity reached during the robot's oscillations and the ground reaction forces. Despite having the same number of legs touching the ground, the bound gait experiences higher forces compared to pacing, and its velocity fluctuates more around the desired 0.5 m/s, indicating greater acceleration and deceleration, thus necessitating higher ground reaction forces.



(a) Gait 3 - gallop (b) Gait 4 - trot run

Figure 2: Results obtained by modifying only the desired velocity, $v_d = 0.5m/s \cdot 3$, so 3 times the nominal value.

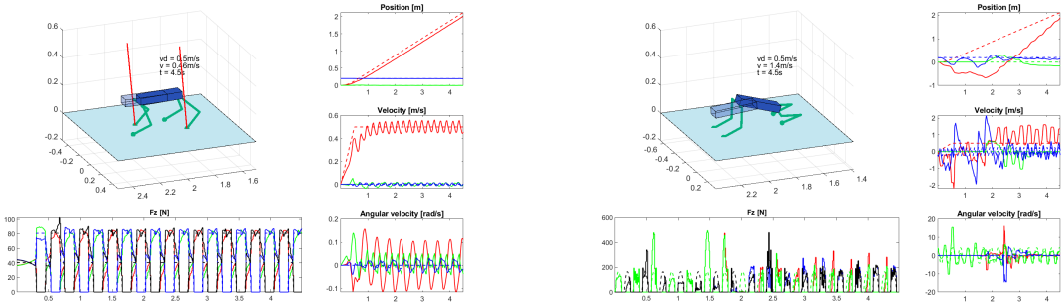


(a) Gait 3 - gallop (b) Gait 4 - trot run

Figure 3: Results obtained by modifying only the desired velocity, $v_d = 0.5m/s \cdot 4$, so 4 times the nominal value.

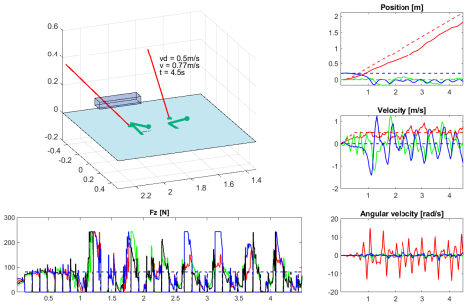
The graphs shown above show the behavior of the robot increasing the desired velocity to three times the nominal one before and after four times the nominal one with the gait 3 and 4. In the first situation we could define ourselves as a limit situation, because in some gaits the robot can move although with some difficulty, but it is possible to notice that with gait 3 it has many difficulties, because with this gait we notice that after a while the robot although it does not fall, it begins however to rotate along the vertical axis. In the same way in the second simulation of the gait 4 - trot run the robot touches the ground first with the rear elbows instead of the rear feet.

Therefore we can say that by increasing the desired speed the robot begins to have several problems, in which you have a maximum desired velocity limit according to the various gait.

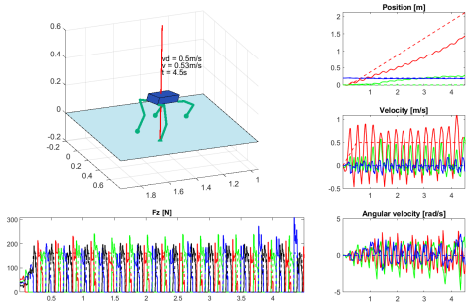


(a) Gait 0 - trot

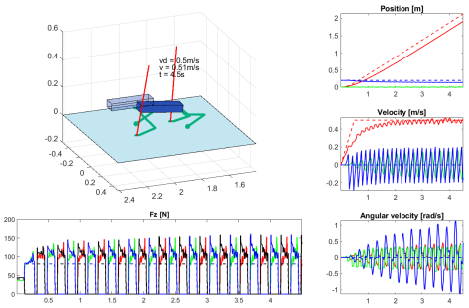
(b) Gait 1 - bound



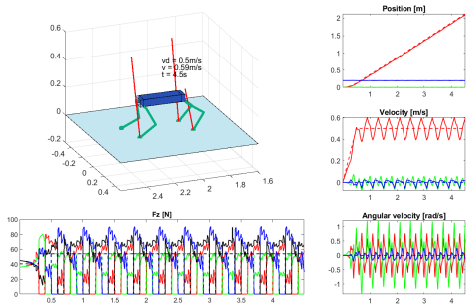
(c) Gait 2 - pacing



(d) Gait 3 - gallop



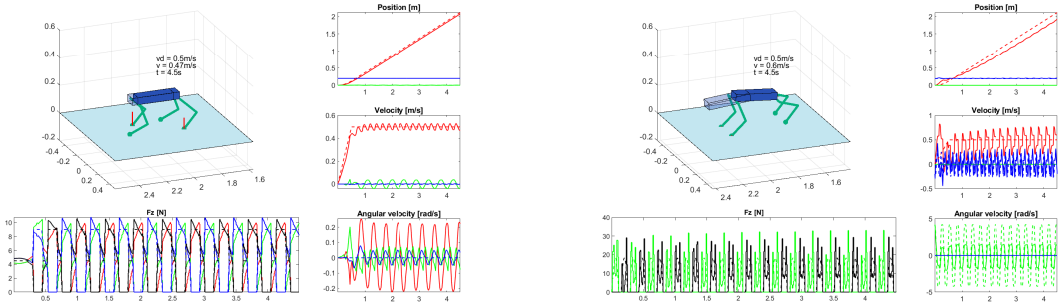
(e) Gait 4 - trot run



(f) Gait 5 - crawl

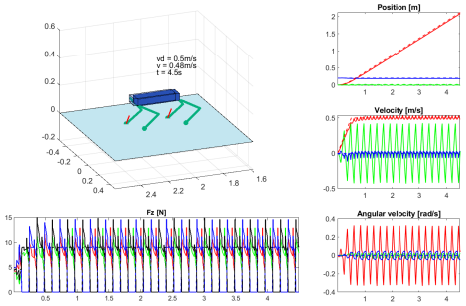
Figure 4: Results obtained by modifying only the mass, $m = 5.5kg \cdot 3$, so 3 times the nominal value.

In the above graphs the mass has been increased by multiplying the nominal mass by 3: the result is a significant rise in reaction forces with the ground, along with a slight overall increase in position and speed errors. It is observed that the simulations with gait 1, 2 and 3 present problems, in fact the first two lead the robot to fall, while the third the robot can walk but rotates around the vertical axis, instead in the other simulations the robot can walk.

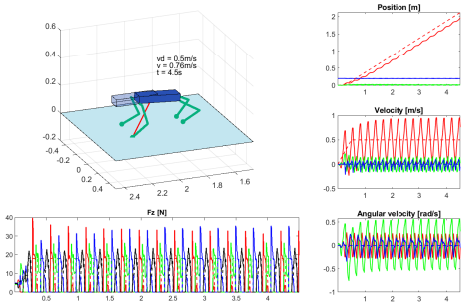


(a) Gait 0 - trot

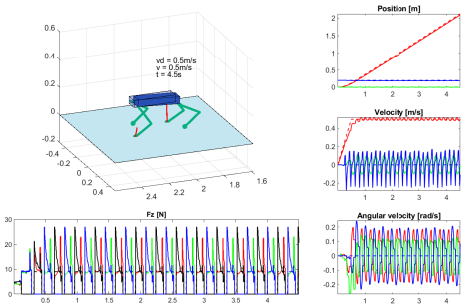
(b) Gait 1 - bound



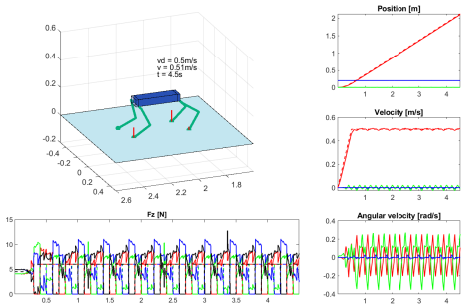
(c) Gait 2 - pacing



(d) Gait 3 - gallop



(e) Gait 4 - trot run



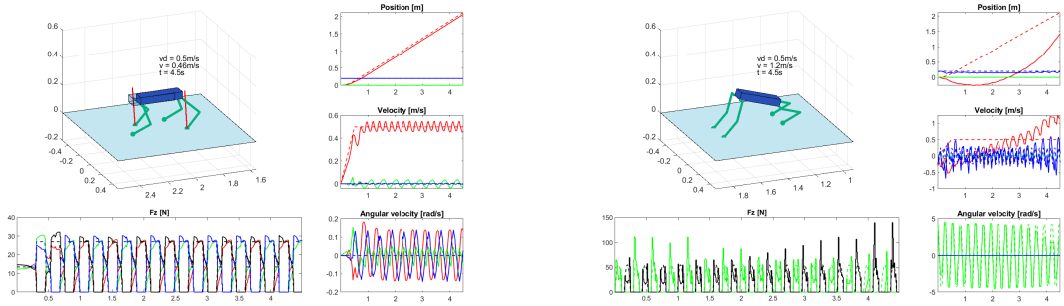
(f) Gait 5 - crawl

Figure 5: Results obtained by modifying only the mass, $m = 5.5kg/3$, so one third of the nominal value.

The figures above and their respective videos demonstrate a significant finding applicable across all gait choices for the robot: reducing the robot's mass and not changing the other parameters leads to several advantages by substantially reducing errors in both velocity and position. This is because a robot with a lighter mass is easier to accelerate to the desired velocity, and with less mass the force of gravity acting on it is decreased, making it easier to maintain its upright position and resulting in smaller reaction forces with the ground.

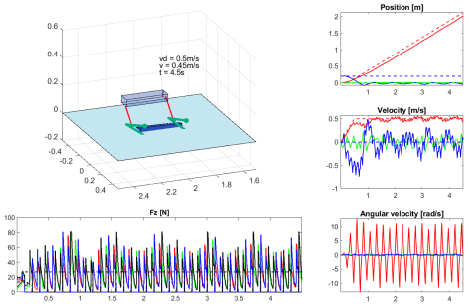
In essence, many of the issues and peculiar behaviors observed in the figures and videos reported can be effectively addressed by opting for a relatively lower mass. However, it's acknowledged that implementing this solution may not be universally feasible in practical scenarios.

However, this is not the case for the gait bound where there is an initial acceleration that leads to a higher position error than for conditions of nominal parameters.

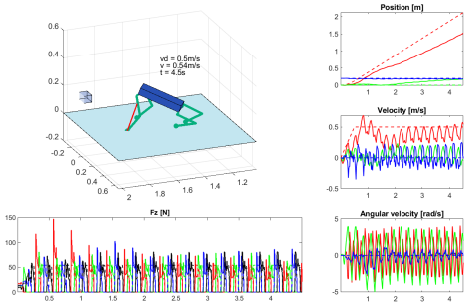


(a) Gait 0 - trot

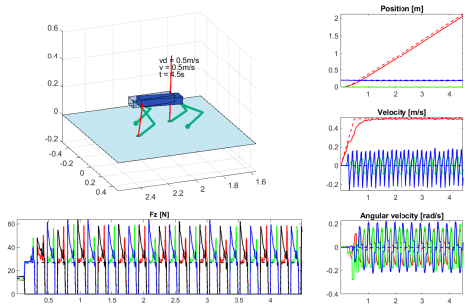
(b) Gait 1 - bound



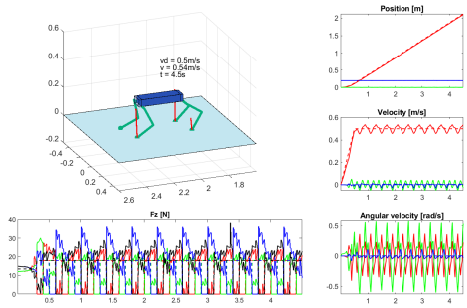
(c) Gait 2 - pacing



(d) Gait 3 - gallop



(e) Gait 4 - trot run

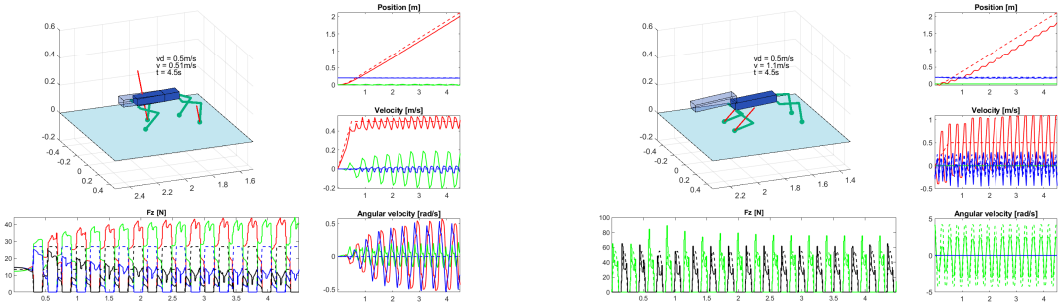


(f) Gait 5 - crawl

Figure 6: Results obtained by modifying only the friction coefficient, $\mu = 1/4$, so one quarter of the nominal value.

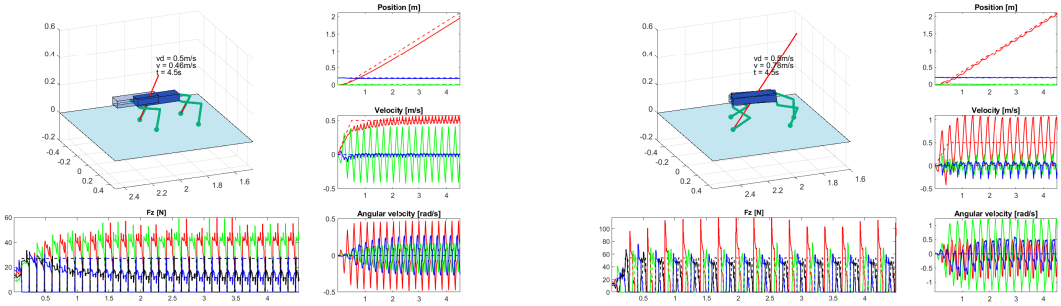
The plots above illustrate only the scenarios in which the coefficient of friction is reduced, as the texts conducted going to double it do not lead to significant effects. With lower friction, it becomes harder for the robot to accelerate because it must adhere to the no-slip constraint, which limits the horizontal forces it can exert on the ground, this constraint can even cause the robot to move backwards initially. Despite this, the robot eventually reaches a limit cycle around the desired speed.

From the simulations reported, where the friction coefficient has been divided by 4 with respect to its nominal value; it can therefore be deduced that decreasing the friction coefficient brings about various problems, as situations are created in which either the robot falls, like the case of gait=2, where this can be caused by the large oscillations that this type of gait presents along the y-axis; or in any case the robot manages to walk but still follows a completely unnatural motion, like the case of gait=3 (gallop).



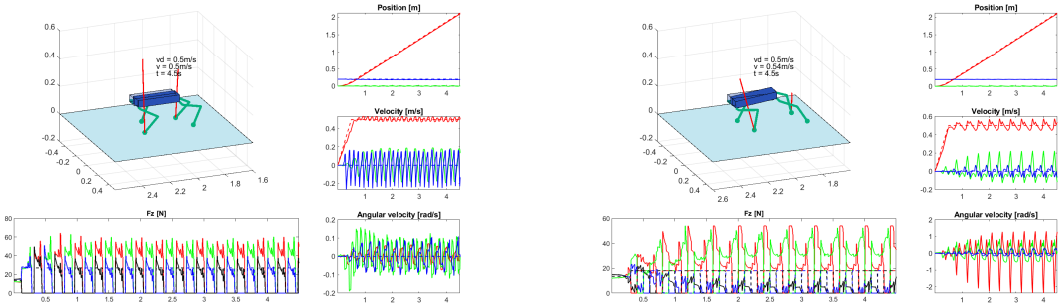
(a) Gait 0 - trot

(b) Gait 1 - bound



(c) Gait 2 - pacing

(d) Gait 3 - gallop



(e) Gait 4 - trot run

(f) Gait 5 - crawl

Figure 7: Results obtained by modifying only the k_{p_sw}, $k_{p\_sw} = 20$.

In the previous graphs, the value of K_{p_sw} is changed. Only the cases where K_{p_sw} is reduced are shown, as tests have indicated that increasing K_{p_sw} doesn't have a significant effect. This is understandable, since beyond a certain point, the swing leg task is already being followed with minimal error, and reducing the error further doesn't result in noticeable changes.

In all cases shown, it's observed that the robot struggles to stay upright, leading to either falling or having greater position and speed errors. Lowering K_{p_sw} leads to a larger error. However, this is not the case for the gallop gait (gait=3), where the errors are smaller compared to the nominal case; this could be due to the complex and asymmetrical nature of this gait. Additionally, with this parameter setting, the angular velocity in gallop gait simulation doubles in value.

Answer 4:

a. Without an actuator at point P, the system is not stable at $\theta = \frac{\pi}{2} + \epsilon$. As this leg ends up being an inverted pendulum, where you have an unstable equilibrium point at $\theta = \frac{\pi}{2}$, which corresponds to the position of the mass in vertical position. In this case, any perturbation ϵ will bring the mass to fall and then the system is unstable.

b. The expression to calculate the zero-moment point is the follows:

$$p_c^{x,y} - \frac{p_c^z}{p_c^z - g_0^z}(\ddot{p}_c^{x,y} - g_0^{x,y}) + \frac{1}{m(p_c^z - g_0^z)} S\dot{L}^{x,y} \tag{3}$$

Placing the origin of a reference frame at the point where the height h intersects the base HT, with the x-axis aligned horizontally to the right and the z-axis oriented vertically upwards, it is possible simplify the expression as follows:

$$p_z^x = p_c^x - \frac{p_c^z}{\ddot{p}_c^z - g_0^z}(\ddot{p}_c^x - g_0^x) - \frac{1}{m(\ddot{p}_c^z - g_0^z)} \dot{L}^y \tag{4}$$

because, in this case, the selection matrix for angular momentum is $S = -1$. Referring to the figure in the track it is possible to define the coordinates of p_c, that is the center of mass in the x-z plane, and its derivatives first and second as follows:

$$p_c^x = l\cos\theta$$
$$p_c^z = h + l\sin\theta$$
$$\dot{p}_c^x = -\dot{\theta} l\sin\theta$$
$$\dot{p}_c^z = \dot{\theta} l\cos\theta$$
$$\ddot{p}_c^x = -\ddot{\theta} l\sin\theta - \dot{\theta}^2 l\cos\theta$$
$$\ddot{p}_c^z = \ddot{\theta} l\cos\theta - \dot{\theta}^2 l\sin\theta$$

the angular momentum and its derivative, considering that we only have the torque around the y axis:

$$L^y = ml^2\dot{\theta}$$
$$\dot{L}^y = ml^2\ddot{\theta}$$

and finally the gravity acceleration:

$$g_0^x = 0$$
$$g_0^z = -g$$

Substituting in equation 4 we obtain:

$$p_z^x = l\cos\theta - \frac{h + l\sin\theta}{\ddot{\theta} l\cos\theta - \dot{\theta}^2 l\sin\theta + g}\Big(-\ddot{\theta} l\sin\theta - \dot{\theta}^2 l\cos\theta\Big) - \frac{l^2\ddot{\theta}}{\ddot{\theta} l\cos\theta - \dot{\theta}^2 l\sin\theta + g}$$

which in static conditions can be simplified, which leads us to obtain: $l\cos\theta$.

c. Assuming to have an actuator which ensures that $\dot{\theta} = \ddot{\theta} = 0$, the permissible values of the angular position (θ) are those that maintain the center of mass projected within the support polygon, which, in this scenario, corresponds to the base HT. Consequently, the ends of the base represent the maximum achievable values corresponding to left $-L_2$ (in point H) and right $+L_1$ (in point T), so:

$$-L_2 \leq l \cos\theta \leq L_1$$

from which you can get the constraints for θ:

$$\arccos\left(\frac{L_1}{l}\right) \leq \theta \leq \arccos\left(\frac{-L_2}{l}\right)$$