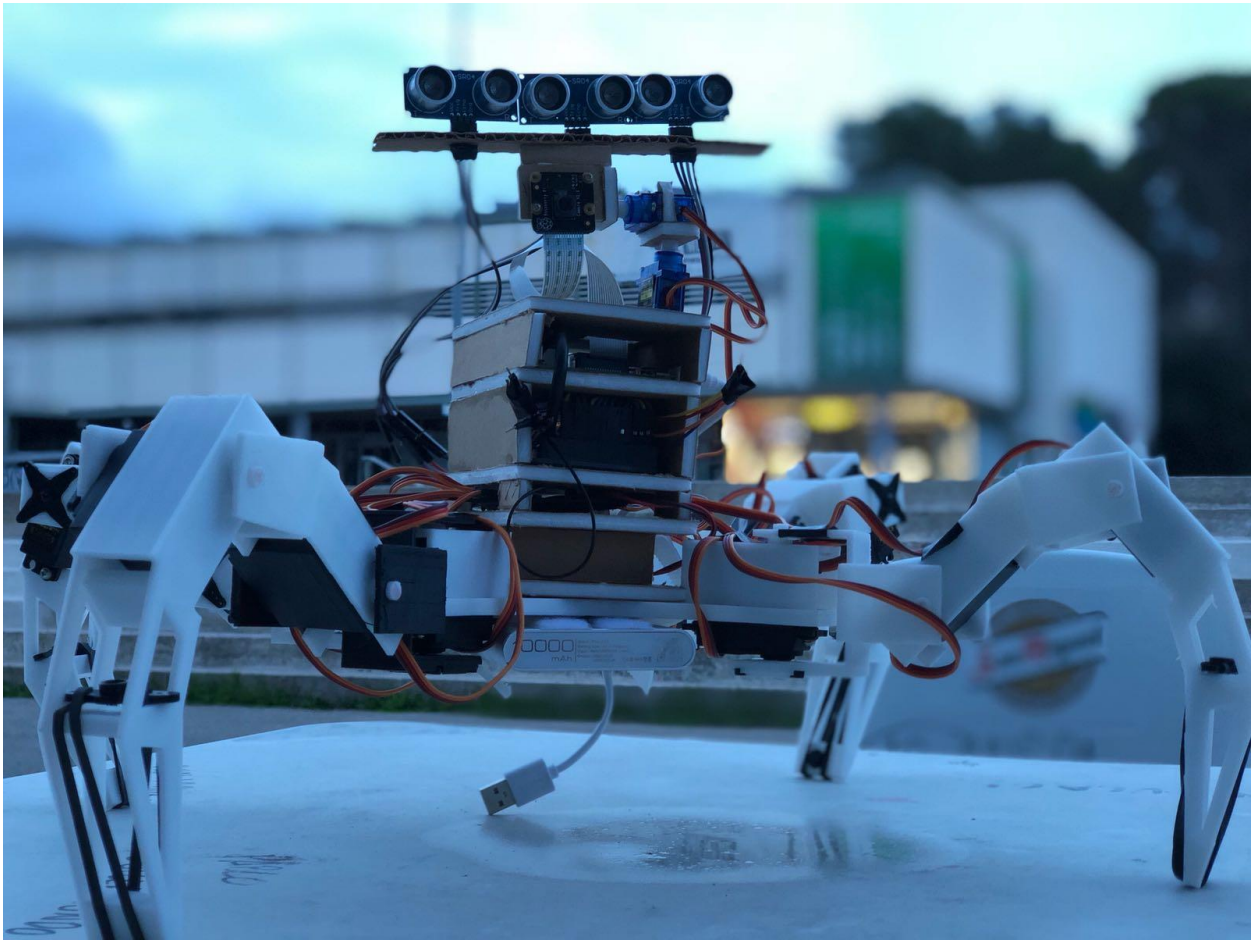


Quadruped Robot

22/07/2018

Edgar Gracia Larrosa - edgar.gracia@e-campus.uab.cat

(Edu)

Table of Contents

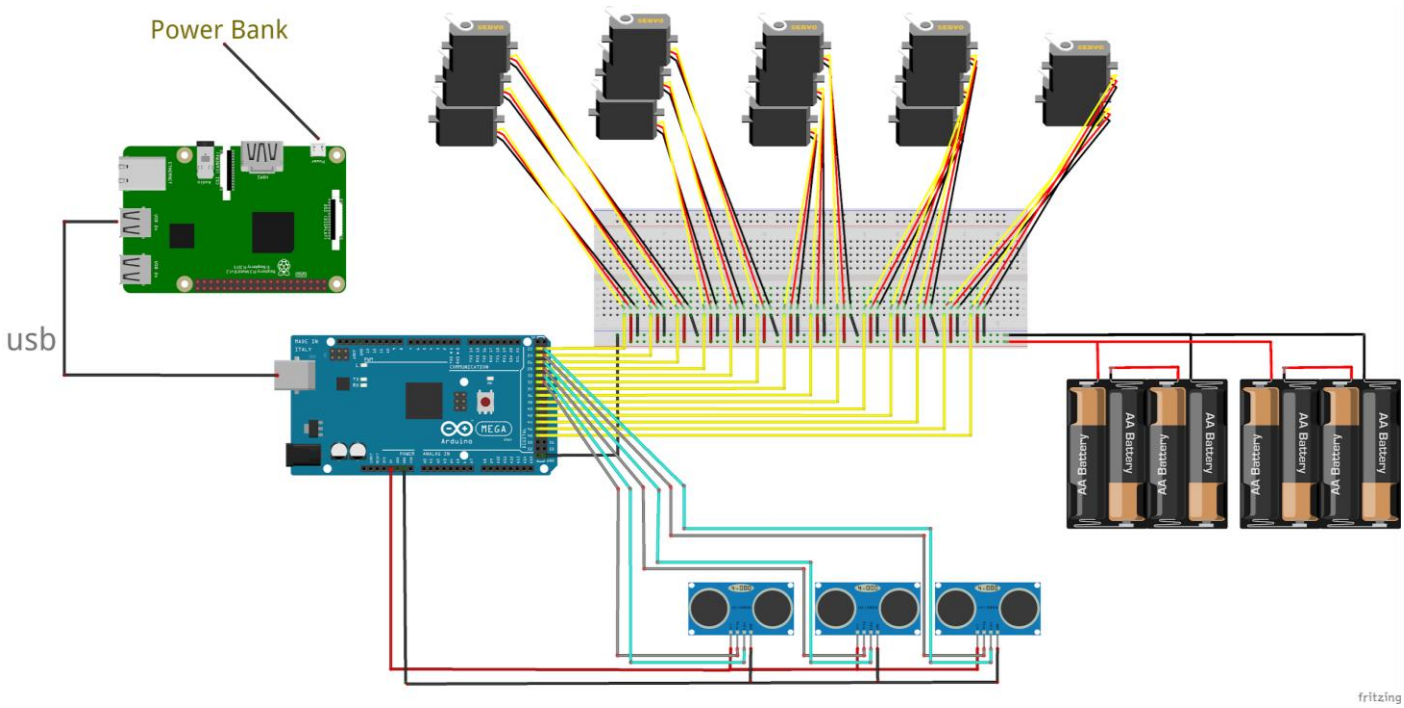
Project description

.....

Electronic components

1x Raspberry pi 3 B
1x Arduino Mega
1x Samsung MicroSD
2x Servo mini sg90
12x Servo MG995
3x HC-sr04
1x 20000 mAh Power bank
2x AA battery holder
8x rechargeable AA batteries
1x RPI NOIR Camera
1x USB type A - Micro
1x USB type A - B
4x Rubber band
headers pins 90º
Jumper cables
PCB
Foam Core and Cardboard

Scheme



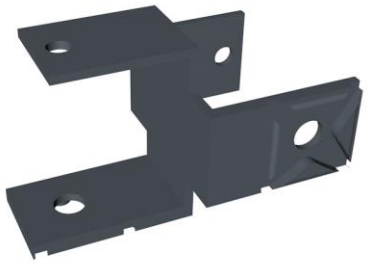
3D pieces

(https://github.com/Physical-computing-UAB/Quadruped/tree/master/Documents/3D%20parts)

1x Base.stl

2x Coxa.stl

2x Coxa-simetric.stl

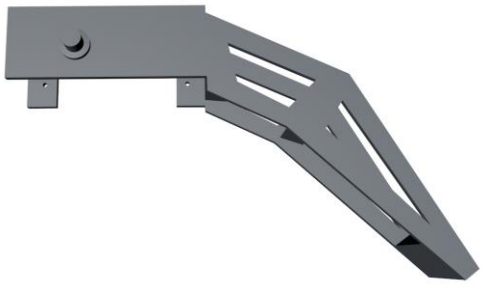


2x Femur.stl

2x Femur-simetric.stl

2x Tibia.stl

2x Tibia-simetric.stl



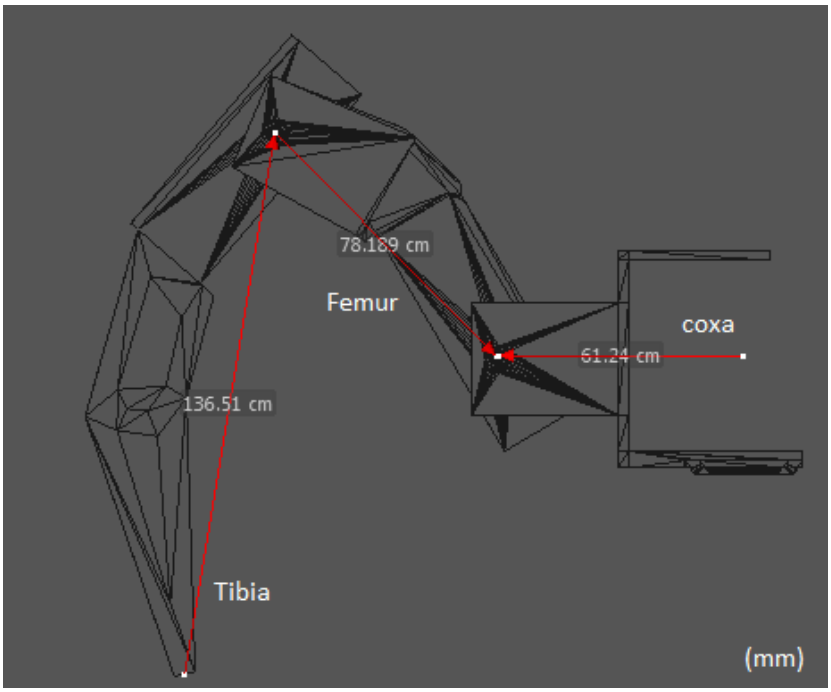
Hardware Issues

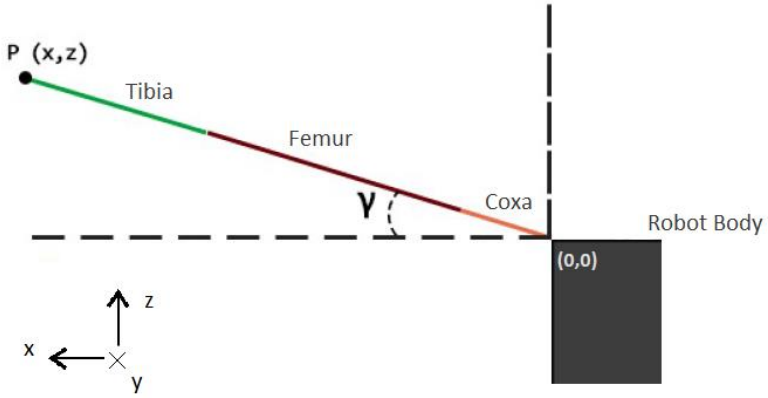
- Sometimes the coxae part broke. The solution found was fix it with glue. However, a better solution would be redesigned the part, improving the weak points.
- Batteries sometimes don't make contact with the battery holder. This causes that the robot does not have power to work. The solution is move a bit all the batteries or reweld the bad contact.
  This issue can be detected because, or the robot doesn't move (that means that both battery holders doesn't work) or the legs of the robot start to writhe (in that case one of the battery holders doesn't work correctly and the robot doesn't have enough current to move all the servos)
- The legs of the robot start to writhe. This is a symptom of a lack of power, caused by some problem with the battery holders or because the batteries have been discharged. (Sometimes when the batteries are not fully charged, if we turn on first the raspberry/arduino and then the power, we might have the same problem. We need to turn on first the battery power)

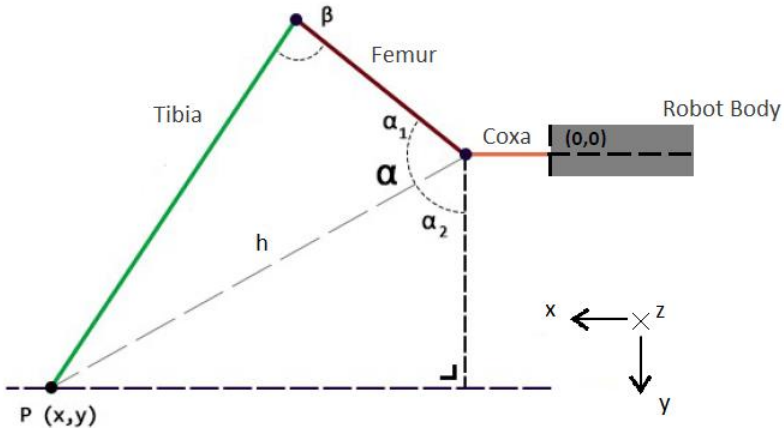
Inverse kinematics

The control software uses inverse kinematics techniques to move the legs. We use a coordinates system on each leg to determine the position that we want to reach. This coordinates are passed to the inverse kinematics functions that calculates the desired servo angles.

Next, we can see the equations used to solve it, the measures of our robot and the coordinates system used.


78.189 cm
Femur
coxa
61.24 cm
136.51 cm
Tibia
(mm)


P (x,z)
Tibia
Femur
γ
Coxa
Robot Body
(0,0)
z
x
y


β
Femur
Tibia
α1
Coxa
(0,0)
Robot Body
α
α2
h
x
z
y
P (x,y)

$$\gamma = \text{atan}\left(\frac{z}{x}\right)$$

$$h = \sqrt{x^2 + z^2} - Coxa$$

$$\alpha_0 = \text{atan}\left(\frac{h}{y}\right) \qquad \alpha_1 = \text{acos}\left(\frac{Femur^2 + h^2 + y^2 - Tibia^2}{2 * Femur * \sqrt{h^2 + y^2}}\right)$$

$$\alpha = \alpha_1 + \alpha_2$$

$$\beta = \text{acos}\left(\frac{Tibia^2 + Femur^2 - y^2 - h^2}{2 * Femur * Tibia}\right)$$

Raspberry configuration

Now we will explain all the necessary process to configure from scratch the raspberry and install all the necessary software.

First step is create a server that transmits the captured video by the camera.
We have to enable the camera on the raspberry configuration:
https://www.raspberrypi.org/documentation/configuration/camera.md
(Inside "raspi-config", the "Camera enable" option may be found inside "Interfacing Options")

Now, to stream the video we will use *pistreaming*, a server that transmits the camera video in real time:
https://github.com/waveform80/pistreaming

Next, we have to install the python library "pyserial", that will allow us establish a serial communication with Arduino through usb:
> sudo pip install pyserial

Then we have to copy the server that will control the robot and transmit information about

the sensors:

.....

Now that we have all the needed software installed, we have to configure the raspberry to run it on every boot:

.....

Finally, Raspberry will be connected with a client pc via wifi to control the robot and send information about the sensors and camera. For that reason we have turned the raspberry into a wifi access point.

We can configure it following the next tutorial:
https://www.raspberrypi.org/documentation/configuration/wireless/access-point.md

With this, every time we turn on the raspberry, it will create an access point which we can connect.