American College of Sofia

Sharo

Forest Patrol Quadruped

Momchil Kolev & Kaloyan Dimitrov

David Yordanov

9 Apr. 2023

Table of Contents

Abstract

Sharo is a 3D-printed quadruped robot whose job is to patrol the forests in order to combat illegal logging and wildfires. Sharo can traverse difficult terrains, detect fires, and catch perpetrators with face detection. It is equipped with a GPS for navigation and to alert authorities of illegal forest activities, IMU for attitude estimation, and an ultrasonic sensor for collision avoidance. Due to the use of hobby electronics and 3D printing, it is affordable, making it ideal for deployment at scale. This also makes it more accessible for educational institutions from less developed regions as an engaging introduction to legged robotics. It can also be used as an affordable R&D platform for scientists and engineers. We demonstrate this capability by implementing highly optimized inverse kinematics and a novel approach to defining motion paths based on Bézier curves.

Fig. 1 Sharo

Introduction

Forest loss is a major driver of climate change, accounting for around 10% of global warming (“Learn the effects of deforestation | WWF”). Illegal logging is rampant, and we even observed such an incident in a park in Bulgaria’s capital Sofia. This problem is further exacerbated by the fact that forest lands are extremely large, and patrolling them to combat illegal logging is practically impossible. Instead of that, we propose that robots be deployed in order to monitor forests and alert the authorities of any such incidents, as well as detect wildfires.

Forests, however, are one of the most challenging environments for robotics and man-made machinery as a whole. The terrain is uneven, while soil characteristics can greatly vary even across small distances. Additionally, sticks and rocks of varying sizes can easily hinder the operations of any machine tasked to traverse such a terrain. Small-wheeled robots in particular, can be practically unusable in these conditions, while bigger ones, as well as those using tracks, risk damaging the environment. Object avoidance is also not a complete solution as, for example, fallen leaves can obscure obstacles.

Drones and other flying apparatus, on the other hand, also cannot be used for ground observation in forests due to the limited visibility caused by the thick tree canopies. A possible solution is for drones to be deployed in the forest instead of over it, but navigating between tree branches is an even bigger challenge. Moreover, drones are especially fragile and fault-intolerant. They also tend to generate high levels of noise and are thus disruptive to the animals, making them completely unsuitable for forest patrol.

If we look at nature, however, thousands of species live exclusively in the forest, many of which are quadrupeds. This is the reason we decided to explore quadruped robotics for the task of forest patrol. Quadrupeds operate on the ground and generate little noise, which makes them a much more suitable solution than any flying apparatus. However, quadrupeds are much better

suited for the challenging terrain of forests compared to wheeled vehicles as the legs can be used to adaptively maintain balance and also overcome higher obstacles. Furthermore, even if the robot is knocked off-balance, it can easily recover. Overall, quadruped robots have much more robust locomotion, which is a requirement for effectively traversing forest terrain and are much more fault-tolerant.

Finally, in order for such a solution to be viable, low cost is paramount because adequately covering large areas such as forests requires more than one robot. Hence, we made affordability a main priority even at the cost of non-crucial features like athletics.

Background Research

Many robotic quadrupeds are currently being designed, tested, and some are even entering the industry in fields such as construction, oil and gas, emergency response, and warehouse automation. However, the earliest concepts date back to the 19th century. In 1870, Pafnuty Chebyshev came up with the lambda linkage, a mechanism that converts rotational motion to translational motion and is able to walk on flat terrain. The concept of the mechanical horse was also developed in 1893. Both of these designs, however, were not capable of independent leg control. Hutchinson developed a locomotive version with independent leg control in 1940, while the first autonomous quadruped robot was constructed in the 1960s (Biswal and Mohanty). After these initial innovations, many versions were developed at various institutes with less substantial novel achievements.

Generally, there are three types of leg designs: prismatic, articulated, and redundant articulated. Prismatic legs consist of a rotating joint and a linear prismatic actuator. On the other hand, articulated legs consist of two rotating joints and are thus more biomimetic. Articulated leg designs could further be subdivided into mammal-type and sprawling-type designs. Mammal-type refers to a vertical configuration of both leg segments, whereas the first leg segment is horizontal in sprawling types. Finally, redundant articulated legs consist of 3 leg segments and 3 rotating joints. Each design has advantages and disadvantages and should be chosen based on the intended application of the robot. The prismatic leg is simple to implement but has insufficient terrain adaptability. Redundant articulated legs boast the best performance characteristic but have a complex design and are thus more expensive and susceptible to failure due to the higher number of breaking points. Articulated legs present a middle-ground alternative, allowing for sufficient adaptability and performance while being much simpler, and

thus more resistant. Because of this, they are most commonly used today (Zhong et al.). For these reasons, we chose them for our project.

Some of the most recent advancements in quadruped robotics were made by the Boston Dynamics Corporation, which developed BigDog, LS3, LittleDog, and, most recently, Spot. In 2011, their Cheetah tethered model reached 45km/h, and in 2013, their Wildcat model became the fastest untethered robotic quadruped. Spot is their current feature-rich iteration which is already entering fields like construction and remote inspection of hazardous environments (Biswal and Mohanty).

The idea of patrolling forests with four-legged robots has been demonstrated before in Italy by the University of Pisa, but for that purpose, they've been using Anybotics' Anymal, which even though is packed with functionalities, costs $150,000 (twice as expensive as Boston Dynamics' Spot Base Kit). Cheaper variations with similar applications are MIT's Mini Cheetah, Weilan's AlphaDog, and Unitree's Go1, but they are all too expensive for the average consumer or mass scale operations.

Even though quadruped robotics have come a long way, they are still largely inaccessible outside the industry. They remain expensive due to the costly parts used to produce them, which in turn were chosen because of the high payload and agility requirements set for these projects.

Companies like Petoi with their robot Bittle and XGO offer budget quadrupeds but at the cost of a smaller size and less powerful motors.

In our attempt to develop a more affordable and easily deployable alternative that is still capable of operating in varying conditions, we decided to focus on reducing the cost of the actuators and producing the corpus, as well as using hobby-grade electronics instead of the more capable yet expensive industrial alternatives.

Due to our choice of the articulated leg design, we only require rotary actuators. Most fluid-powered actuators involve a mechanism to convert linear motion to rotary, which introduces unneeded complexity and weight to the system ("All About Rotary Actuators - Types, Applications, and Uses"). Hence, electric motors are a better choice. Conventional AC and DC motors, although more powerful, have no positional control ("Types of Electric Motors | automate.org"). Motor encoders, along with a control algorithm like PID can be used to implement such functionality, but that would also unnecessarily increase the complexity and cost. On the other hand, both stepper and servo motors provide the required precision positioning. Stepper motors are less efficient, slower, and provide less torque (Tang).

Procedure

We started working on the first prototype more than a year ago. It was based on the SpotMicro design, and for the electronics, we decided to use an Arduino Uno and 9kg servos and implemented our own kinematics calculations. At first, while testing on a test bench, the motion seemed promising, but we faced a multitude of problems when we placed the robot on the ground. The motors were not powerful enough, so on each step, Sharo collapsed on the floor. The motion was also very slow and jittery because of the poorly optimized calculations and low processing power of the microcontroller, so we scrapped the idea and went in a completely new direction. We began working on our own mechanical design to overcome the limited leg motion of the SpotMicro. We also decided to use the IKPy inverse kinematics library in order to focus more on programming robust locomotion. We tried a variety of leg designs and came up with a novel approach to defining motion that greatly simplified the leg motion code. After achieving a stable gait, we moved on to integrating various sensors, and we added control and communication. The IKPy library, although great for prototyping, also proved slow especially after adding the sensors and communication, so we returned to the original inverse kinematics calculations but with further improvements.

Raspberry Pi 4	“0.96 OLED screen
DFROBOT DC5535 35kg servos x12	Buck Converter x2
Servo Driver (PCA9865)	Li-ion / LiPo batteries
Raspberry Pi Camera	PMod NAV 10-DOF IMU
HC-SR04 Ultrasonic Sensor	3D Printed Parts
Adafruit GPS Module	Hardware (bolts, nuts, bearing)

Fig. 2 Parts and Materials

Mechanical Design

There are two most common quadruped leg designs. First, there is the standard Spot-looking design, which incorporates a servo on each joint. The issue with it, is that it has limited freedom of motion, and the servo motors add additional weight to the legs. In the second design, these problems are eliminated by moving all of the servos inside the body, but another issue occurs. The many complex joints lead to backlash, which causes the robot to wobble. We combined the simplicity of the first with the idea of the second, and many prototypes later, we finalized our own version.

Each leg consists of three 35kg servos allowing for a 3-dimensional movement. The so-called shoulder is in charge of moving the legs left and right, while the motion of the other two - forward, backward, up, and down. It ends with a soft TPU ball for better grip and cushioning. Everything is 3D printed, which allowed us to easily prototype with different designs.

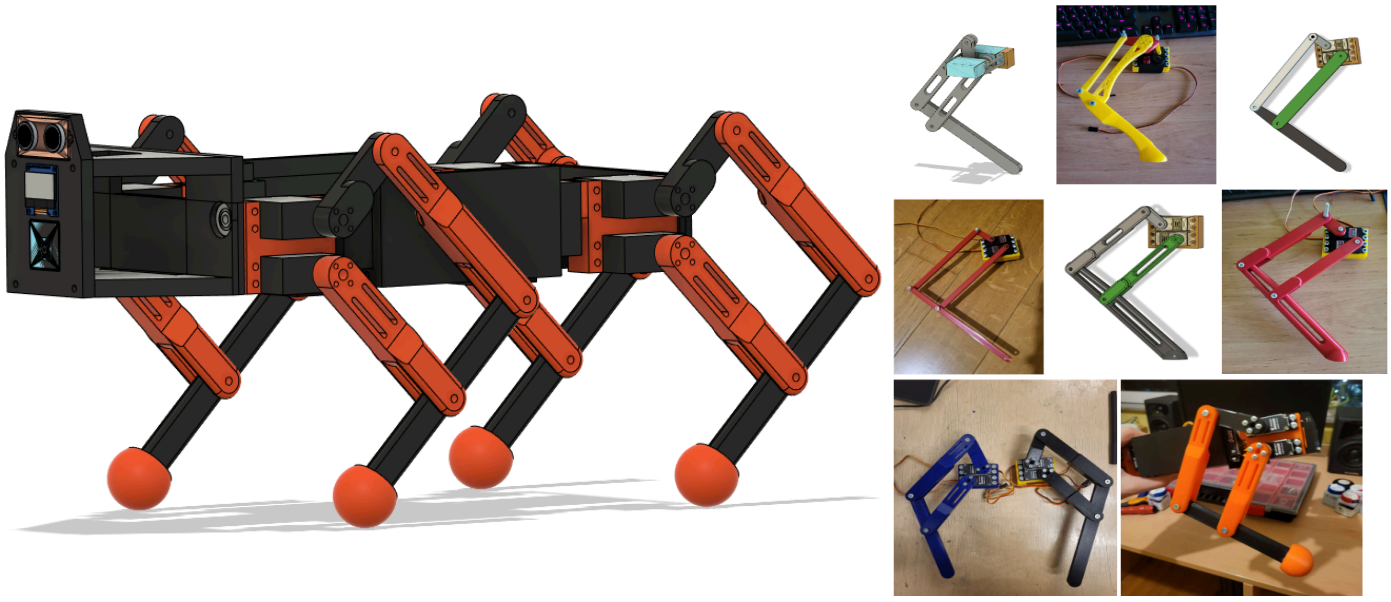

Fig. 3 Leg design evolution

Electronics

The main processing unit was chosen to be the Raspberry Pi 4 as it is an accessible computing platform with sufficient I/O. For face and fire detection, the Raspberry Pi camera is used since it is easily integrated with the computer board. The PMod NAV 10-DOF IMU is used for attitude estimation. The HC-SR04 ultrasonic sensor is used for crash avoidance because of its low price and availability. Since there are only 2 PWM channels on the Raspberry Pi and the generation of the signal also takes up significant resources, we use a 16-channel servo driver board, the Adafruit PCA9685, which is powered by a 2S LiPo or Li-Ion battery. When fully charged, the battery has a voltage of around 8.4V, while the operational voltage of the servos is 5-7.4V. Hence, we use a buck converter to lower the battery voltage to 6V. For the Raspberry and the sensors, which require constant 5V, we use a separate buck converter to lower it for the desired voltage.

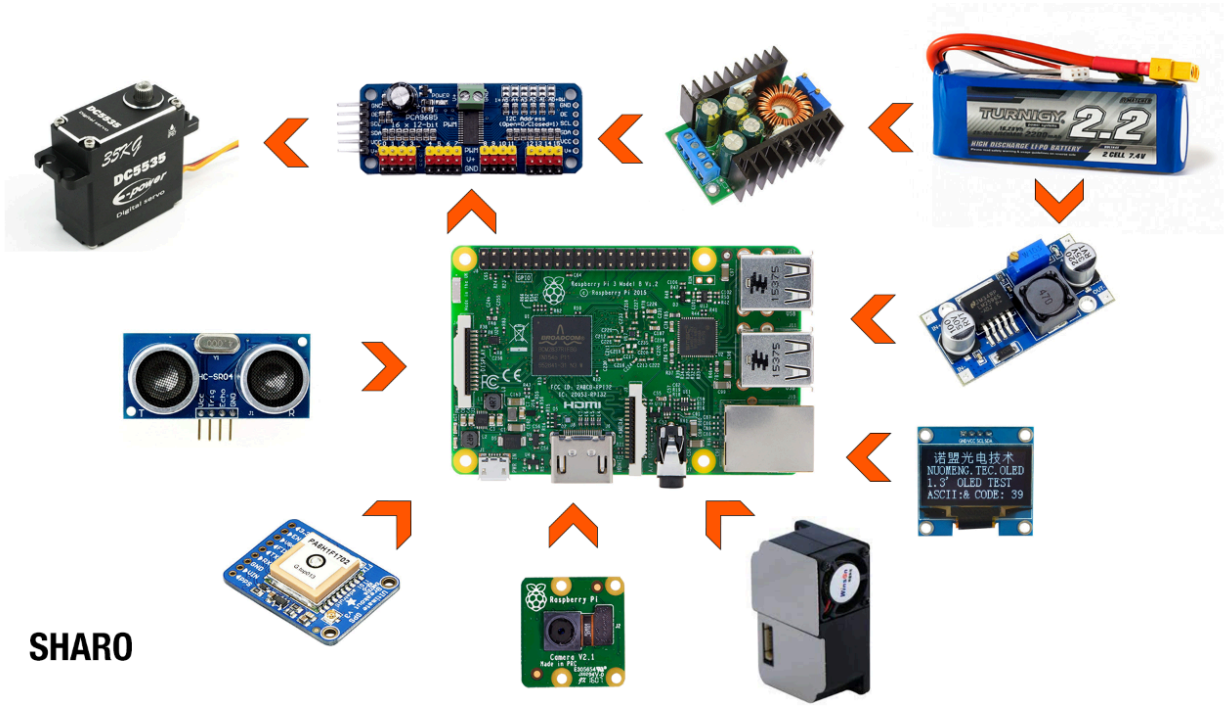

Fig. 4 Electronics

Communication and Control

We have implemented a simple client-server architecture. The Raspberry Pi on the robot hosts a local TCP server that accepts commands from a client program running on another device on the same network(the operator console). When a command is received by the server, it is passed to the controller. The controller, in turn, is responsible for coordinating with the PCA9685 servo driver using the I2C protocol and thus controlling the servos. Both the server and the controller involve some sort of I/O operations, so we use the *mutliprocessing* library in Python to allow easy communication between the two processes without interfering with each other. We also use a separate process to monitor all the sensors, and send the acquired data to the controller and TCP server, which is also responsible for sending that data back to the operator console.

A separate process is responsible for managing the Raspberry Pi camera. It hosts its own UDP server, to which the operator console connects. By separating the communication into two separate independent channels we have not only better redundancy in the case of failure, but also reduced latency. Currently, computer vision is done on the operator console in order to not strain the Raspberry Pi computational resources unnecessarily.

A great challenge we faced was integrating the IMU. Although the raw data from the accelerometer, gyroscope, and magnetometer is easy to obtain programmatically, using it to produce accurate attitude estimates proved exceptionally difficult. Initially, we attempted to use the Madgwick Orientation Filter implementation provided by the *ahrs* library, but the data was unreliable and enormous drifting occurred. This was mainly due to the poor documentation of the PMod NAV, and in particular to its reference frame and sensor orientation. However, even after figuring that out, the drift only slightly decreased. We then attempted to use the Extended Kalman Filter, but had similar results. In the end, the sensor’s low rate proved to be the key

issue. We solved this by estimating the attitude much more frequently than we were receiving data from the IMU. By duplicating the samples from the sensor, calibrating the gyroscope on each power up, and switching to the Mahony Orientation Filter(a supposedly less accurate estimator for lower powered processing units), the drift was eliminated, but two other issues surfaced. Firstly, the convergence time was unacceptably high, as the estimator would need a few seconds to reach and settle on the correct orientation. This was greatly improved by increasing the proportional parameter of the algorithm, which, however, increased the amplitude of the oscillations around the estimation. These oscillations remain unresolved for the moment as they do not affect the performance of Sharo in a critical way and we plan to fix them by applying a low pass filter to the data from the IMU. The second issue was a high overshoot, where after starting to tilt the robot in a certain direction the estimation would jump 10°-15° in the opposite direction of the motion. We managed to largely mitigate this problem by decreasing the integral parameter of the Mahony filter.

Although usable, the data from the IMU is still unreliable to be used for self-balancing and remains a major obstacle, on which we are actively working to overcome.

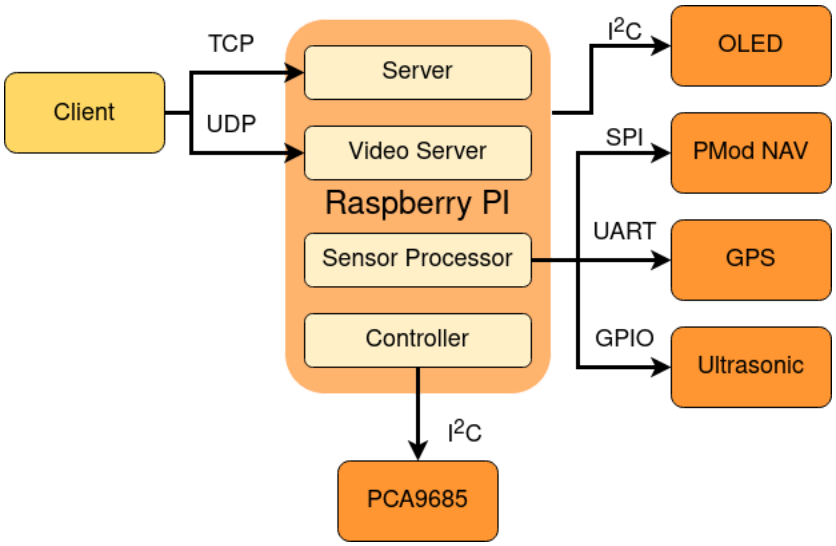

Fig. 5 System architecture

Kinematics

In general, kinematics is an essential topic in the robotics field and is a subject of active research. It is the study of multi-degree of freedom systems called kinematic chains, which consist of rigid links and joints that provide pure rotation or translation. A quadruped leg fits this model exceptionally well, as it consists of two links and two joints.

We describe the leg using the URDF format, which allows us to apply tried and tested algorithms for inverse kinematics – that is the process of calculating desired joint(servo) angles based on a position in space that we want our leg to reach. For each leg, we define a 3-dimensional coordinate system with an origin where the leg is connected to the body, allowing us to program specific positions of the robot intuitively instead of trying to manually tune servo angles. Initially, we implemented our own inverse kinematics algorithm on the Arduino platform, but found it to be poorly optimized and the Arduino itself unable to handle such volume of computation. We then tried the ESP32, which also did not meet our needs. Consequently, we switched to the Raspberry Pi and the Python programming language, as well as the IKPy library for inverse kinematics (Manceron). Currently, the positions we have programmed are sit, lie, and stand.

However, using the same approach for complex movement routines like walking is impractical as it would involve manually setting the coordinates of many points while attempting to maintain balance and later adapt to uneven terrain. To solve this issue, we propose a novel approach to describing motion in robot kinematics – Bézier curves.

Bézier curves are extensively used in computer graphics, in both CAD/CAM applications as well as creative software suites like Adobe Photoshop. Bézier curves are very flexible and allow us to define smooth splines using only a few points. This is the reason they are also used

for various file formats, including SVG for vector graphics and TTF/OTF for fonts. The curve is then evaluated at m points and we apply the same inverse kinematics for each point. This also allows us to control the speed of the motion by varying m. Mathematically, Bézier curves can be defined as follows:

$$Bézier(n,\ t) = \sum_{i=0}^{n} \binom{n}{i} \cdot (1 - t)^{n-i} \cdot t^{i} \cdot P_{i}$$, where n is the order of the curve(the number of control points) P_i is the position of the i-th control point, $\binom{n}{i}$ is the binomial term and $(1 - t)^{n-i} \cdot t^{i}$ is the polynomial term (Barnhart). Lines can be described with a first-order Bézier curve, while curves can be described with a second or third-order curve. Furthermore, this is a generalized form for any number of dimensions, and the transition from 2D-only kinematics to 3D is effortless. Bézier curves have other convenient properties making them ideal for describing the locomotion of a robot. For example, the first and the last control point coincide with the beginning and end of the curve. By providing an intuitive definition of start and end points, as well as enabling multiple curves to be seamlessly joined into a single path, this feature facilitates the description of more complex movements with ease.

Another advantage of Bézier curves is the pseudo-torque control they can provide only through a positional interface. We used this property to achieve a very stable flat terrain gait. The spline of each leg's motion consists of a line pushing the robot forward when there is contact with ground and a second-degree curve to move back to the forward position while other legs push. Initially, we started with a horizontal line, but we faced issues with Sharo tilting and eventually falling over. The solution was to tilt the motion line, so that more force is applied by the leg at one end of the line than at the other to compensate for the tilting. Hence, we have demonstrated how the force of a positional actuator can be controlled.

The flexibility of Bézier curves, however, allows this approach to be applied to a much more general problem in contemporary robotics: haptics (Seminara et al.). By changing the shape of a Bézier curve, precise forces can be applied with smooth transitions, instead of instantaneously applying a force, for instance. Combined with haptic sensors, feedback loop systems can be designed with predictable behavior. If properly implemented, such a system could be capable of carrying extremely delicate and brittle items and even withstanding disturbances like a sudden crash by increasing the gripping force enough to hold onto the item, but not to break it.

This is made possible by another crucial property of Bézier curves – to computer generate them easily. Bézier curves are only controlled by a few points, each with 3 coordinates, changing which induces rather predictable changes in the curve's shape. This property allows feedback loop systems to be developed with the help of these splines. Although not explored in this project, we believe that various control systems could be improved with the help of Bézier curves, and this to be a curious topic for further inquiry.

The Bézier curve approach to locomotion has the enormous benefit of allowing fast iteration over possible walking routines. We were thus able to define two gaits for Sharo. The first one is ideal for flat terrain easily maintaining balance and stable body orientation, while the second one is more robust and ideal for difficult terrain with the tradeoff of not being able to maintain stable body orientation. We can also use the IMU to detect disturbances during the flat terrain gait and switch to the robust gait in an attempt to maintain balance. Furthermore, we were then able to add variable linear and steering speed to these routines in just a single day.

Another challenge we encountered was that the IKPy library, although easy to use, is also poorly optimized and unable to run real-time inverse kinematics. We initially solved this issue

with precomputation, but this approach is very limiting as it dramatically increases the boot up time and does not allow dynamic walking. We identified the reason for this issue to be that the inverse kinematics algorithm used by the IKPy library is not suitable for our problem.

There are two distinct methods for solving inverse kinematics problems – analytical and iterative. Analytical solutions are precise and computationally inexpensive, but struggle with more complex kinematics chains. Moreover, since they produce all possible solutions to the derived system of equations, an additional problem arises: How do we determine the best possible solution? On the other hand, iterative solutions produce good approximations but can require a long time in order to converge, and are thus computationally expensive (Nilsson).

Hence, both approaches are viable and should be chosen on a per-case basis, depending on the specific task at hand. Iterative approaches are more general and are more easily applied to complex kinematics chains with many joints and links, whereas analytical approaches are best suited for simpler kinematics chains. Generally, if the system at hand has too few degrees of freedom, the analytical approach is unable to produce a solution, whereas too many degrees of freedom result in infinitely many solutions. As already stated, Sharo's leg design consists of only two joints and two links(femur and tibia, respectively the upper and lower part of the leg) and thus the analytical approach is preferable.

Taking the 3D coordinate system described above, the inverse kinematics problem can be described as such: How can two connected lines(the first with the length of the femur and the second – the tibia) starting from the origin, p. A, reach a given target point, C. We can break down this problem in 3D space into two problems in 2D space. Firstly, we can only look at the ZY plane. We need to calculate the angle, θ, between the y-axis and the target.

$$\theta = arctan(\frac{C_z}{C_y})$$

This is guaranteed to have exactly one solution.

We can then look at the XY plane, where we need to determine the angle, φ, between the femur and the-x axis of the coordinate system, as well as the angle, α, between the femur and the tibia. The latter is easy using the law of cosines in △ABC, where A is the origin of the coordinate system(where the leg is attached to the body), B is the attachment point of the femur and the tibia, and C is the target point. Hence, AB is the femur and BC is the tibia.

$$\beta = arccos(\frac{AB^2+BC^2-AC^2}{2\times AB\times BC})$$

The former, however, is more difficult as calculating it requires the addition of α and ω, where α is the angle between AB and AC and ω is the angle between AC and the x-axis.

$$\beta = arccos(\frac{AB^2+BC^2-AC^2}{2\times AB\times BC})$$

There is always only one valid solution, which can be proven using two circles, centered at A and C, respectively. The first has a radius equal to the length of the *femur*, while the second one – the *tibia*. Consequently, a valid solution is any intersection between the two circles. There are three cases:

1. The circles are concentric: There are 0 solutions as the two circles have different radiuses $femur \neq tibia$. This case can easily be avoided by making sure that $C \neq A(C_x, C_y, C_z \neq 0)$.
2. The two circles touch at a single point($AC = femur + tibia \; or \; AC = femur - tibia$): This is the ideal case as only 1 solution is available.
3. The two circles intersect: This is the most common case in practice. Two solutions are available, but it is easy to determine which one to use as one of them is

outside the range of motion of the leg. $\beta = \beta'$, which can be proven by the fact that $\triangle ABC \cong \triangle AB'C$. Although $\varphi_1 \neq \varphi_2$, the use of the atan2 function in the code allows for one of the solutions to be eliminated by carefully choosing the signs of the supplied arguments to the function to eliminate the solution outside of the leg's range.

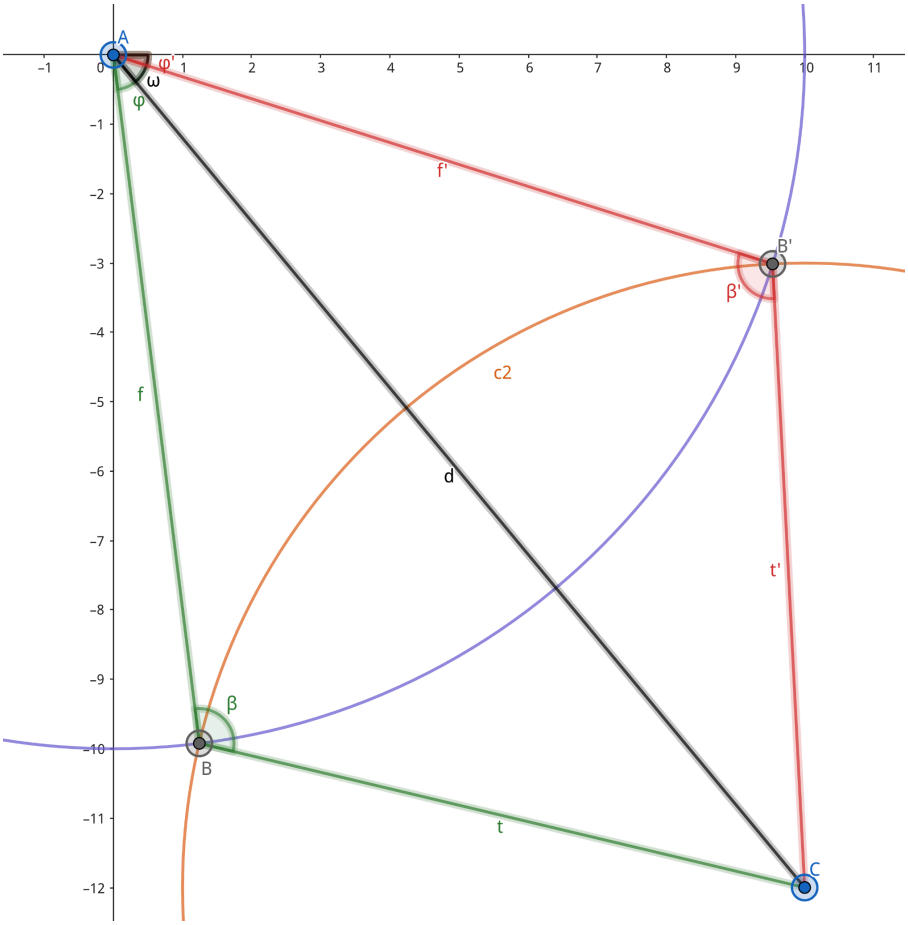

Fig. 6 General case of inverse kinematics(2 possible solutions)

We can summarize the algorithm with the following system of equations, which we have proven has a single solution, except for edge cases which can be easily avoided:

$$\theta = arctan(\frac{C_z}{C_y}) \quad (1)$$

$$\beta = arccos(\frac{AB^2+BC^2-AC^2}{2\times AB\times BC}) \quad (2)$$

$$\varphi = arccos(\frac{AB^2+AC^2-BC^2}{2\times AB\times AC}) + arctan(\frac{C_y}{C_x}) \quad (3)$$

On the other hand, the IKPy library, in an attempt to solve a greater number of problems, uses an iterative algorithm. Although useful for early prototyping and for verifying other implementations, it is less than optimal for a quadruped. The analytical solution proposed above is much faster and more precise.

The main performance bottleneck of the analytical approach is its reliance on inverse trigonometric functions, which are also computationally heavy. However, in the case of a quadruped precise angles are not required for most tasks, so much faster approximations of *arccos* and *arctan* can be used. The implementations used by us are available in the Cg 3.1 Toolkit by Nvidia ("Cg Standard Library Documentation").

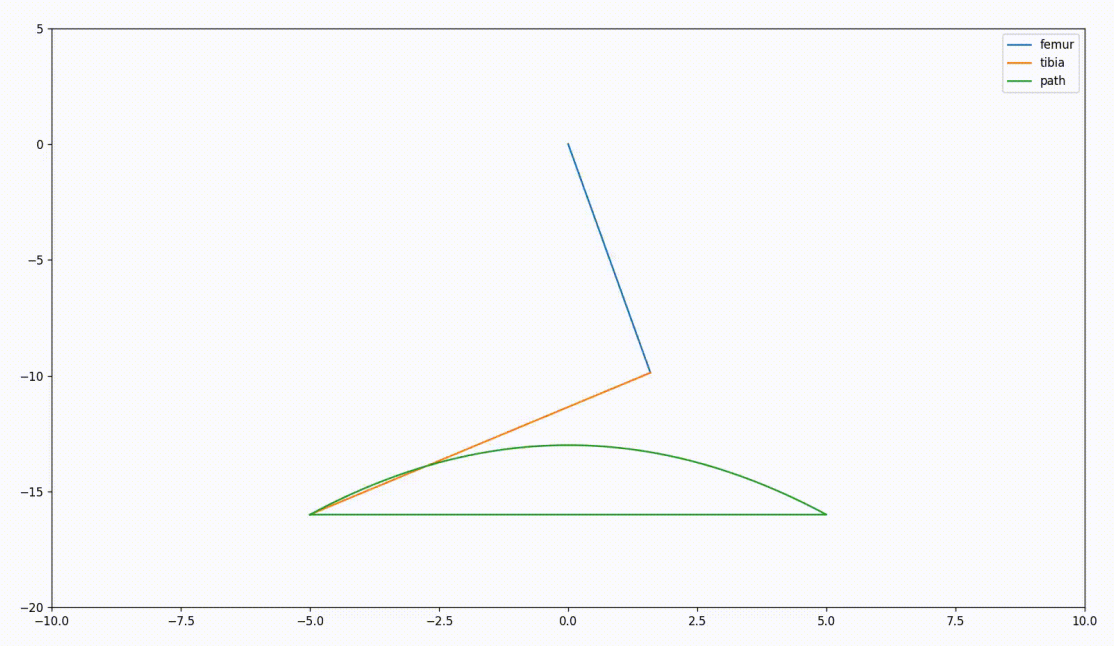

Fig. 7 2D simulation of a leg following a Bézier path with inverse kinematics

Applications

There are numerous potential practical applications for Sharo. Firstly, we began the project as an initiative to combat deforestation, and thus global warming. Sharo was initially meant as a forest patrol, capable of detecting illegal logging and wildfires. Recognizing that patrolling such large areas is infeasible for a single robot, we made affordability and thus deployment at scale a main priority.

While working on the project, however, we discovered many other possible applications. For example, Sharo could be used for educational purposes at all levels of schooling. Due to its affordable price, this robot dog would provide students of all ages with the opportunity to learn about the mechanics, kinematics, electronics and develop upon them with their own innovations. The skills and knowledge students gain from collaborating on these types of projects can be then applied to real-world situations. Additionally, this interactive approach for teaching is proven to be an effective way to get the students attention with activities, corresponding to their age. Sharo's versatility offers limitless possibilities for the curious to program it for a wide range of tasks, from basic movements to more complex behaviors. For instance, younger students can play along and learn the basics of robotics and computer programming, while highschool and university students can work on implementing their innovative ideas, such as we have with our idea to use Bézier curves to describe motion. Moreover, the cost-effectiveness of the project makes it more accessible for schools and universities with less financial opportunities from developing regions. Getting involved in such a project also introduces the students and professionals to an international community of like-minded individuals for collaborative learning.

Although not our initial goal, Sharo has the potential to also be used in industry. Currently, this space is dominated by the exceptionally capable offerings from the likes of Boston Dynamics and Anybotics. These companies have developed highly-capable robots to satisfy the high demands of industries such as construction. Sharo's plastic design and relatively low payload capacity make it unusable for such activities. However, there are many tasks with lower requirements, where the use of the more advanced and thus more expensive robots is simply unnecessary. From remote inspection, to warehouse management, there are a multitude of industry-critical roles that Sharo can take, and do so at the fraction of the cost of other options. Both smaller businesses and governments can greatly benefit from robotics without the need for a large investment. Furthermore, for the cost of one Boston Dynamics Spot, an entire swarm of Sharo robots can be deployed, which can be much more effective for certain tasks. Overall, we believe that the path to faster integration of robots in industry and society as a whole involves minimizing cost by correctly evaluating the requirements for particular tasks and thus choosing an appropriately priced option.

Future Improvements

Although we currently have robust walking routines that are able to navigate flat and slightly rough terrain, they are after all predefined routines. Our primary goal moving forward with the project is to implement fully dynamic walking utilizing the IMU because one of the main design considerations when we came up with the Bézier curve approach to defining motion paths was its potential to greatly simplify the eventual implementation of adaptive gait.

We were also only able to test the fire detection model using handheld lighters and footage of wildfires. However, that footage was not captured with the Raspberry Pi camera, and we cannot be sure if the model will be reliable in practice. We have also begun integrating an air quality sensor, which can aid the detection of wildfires.

The current model only recognizes if there are faces in the frame. Combined with the GPS data, this is useful for alerting authorities, but integration with face recognition databases can also be added to make Sharo even more effective at combating all types of crime.

Additionally, we plan to add a Lidar for better SLAM and brushless motors for higher agility, which would be beneficial in even harder to traverse forest terrains. SLAM is also required for the effective use of robotics in industry, but remains an enormous challenge in the field of robotics. Sharo's IMU, camera, and LIDAR in the future will make it a perfect accessible platform for the development of novel algorithms for navigation and mapping, both indoors and outdoors.

Communication is currently a greatly limiting factor as we use the Raspberry Pi's builtin WiFi hotspot capabilities. In the future, we plan to integrate a 5G SIM card which would allow for low latency long range communication and control.

Conclusion

Overall, we successfully built our own affordable quadruped with an original mechanical design. We keep the cost down by relying on 3D printing and hobby electronics. We then demonstrated Sharo's capabilities by implementing simple yet robust motion routines using a novel approach with Bézier curves. We also implemented our own highly optimized inverse kinematics.

We described multiple potential practical applications of the project, including forest patrol, involving computer vision and multiple sensors.

We outlined current limitations and our planned solutions. We also charted the future direction of the project with concrete plans for improvements and how they will benefit our mission – using robotics to protect people and the environment, as well as to accelerate technological and scientific progress.

References

“All About Rotary Actuators - Types, Applications, and Uses.” *Thomasnet*, https://www.thomasnet.com/articles/machinery-tools-supplies/rotary-actuators/. Accessed 19 March 2023.

"AlphaDog | WEILAN." 阿尔法机器狗 | 蔚蓝, www.weilan.com/en/robots.html. Accessed 10 Apr. 2023.

"ANYmal - Autonomous Legged Robot - ANYbotics." *ANYbotics*, 27 Mar. 2023, www.anybotics.com/anymal-autonomous-legged-robot/. Accessed 10 Apr. 2023.

Barnhart, Ben. “The Birth of Bézier Curves & How It Shaped Graphic Design.” *Vectornator*, 28 October 2022, https://www.vectornator.io/blog/bezier-curves/. Accessed 19 March 2023.

Biswal, Priyaranjan, and Prases K. Mohanty. “Development of quadruped walking robots: A review.” *Ain Shams Engineering Journal*, vol. 12, no. 2, 2020, pp. 2017-2031. *ScienceDirect*, https://www.sciencedirect.com/science/article/pii/S2090447920302501.

“Cg Standard Library Documentation.” *NVIDIA*, https://developer.download.nvidia.com/cg/index_stdlib.html. Accessed 9 April 2023.

“Learn the effects of deforestation | WWF.” *WWF-UK*, https://www.wwf.org.uk/learn/effects-of/deforestation. Accessed 18 March 2023.

Manceron, Pierre. “IKPy.” *Zenodo*, 2022, https://doi.org10.5281/zenodo.6551158.

"Mini Cheetah is the First Four-legged Robot to Do a Backflip." *MIT News | Massachusetts Institute of Technology*, news.mit.edu/2019/mit-mini-cheetah-first-four-legged-robot-to-backflip-0304. Accessed 10 Apr. 2023.

Nilsson, Rickard. *Inverse Kinematics*. 2007. *DiVa Portal*, https://www.diva-portal.org/smash/get/diva2:1018821/FULLTEXT01.pdf. Accessed 9 April 2023.

Seminara, Lucia, et al. “Active Haptic Perception in Robots: A Review.” *Frontiers*, 1 July 2019, https://www.frontiersin.org/articles/10.3389/fnbot.2019.00053/full. Accessed 9 April 2023.

"Spot® - The Agile Mobile Robot." *Boston Dynamics*, www.bostondynamics.com/products/spot. Accessed 9 Apr. 2023.

Tang, Johann. “The Choice Between Servo Motors and Stepper Motors.” *Engineering Notes by Oriental Motor*, 30 November 2021, https://blog.orientalmotor.com/the-choice-between-servo-motors-and-stepper-motors. Accessed 19 March 2023.

“Types of Electric Motors | automate.org.” *Association for Advancing Automation*, 6 September 2016, https://www.automate.org/blogs/types-of-electric-motors. Accessed 19 March 2023.

"Unitree Go1." *UnitreeRobotics*, shop.unitree.com/products/unitreeyushutechnologydog-artificial-intelligence-companion-bionic-companion-intelligent-robot-go1-quadruped-robot-dog. Accessed 9 Apr. 2023.

Zhong, Yuhai, et al. “Analysis and research of quadruped robot’s legs: A comprehensive review.” *Sage*. *Sage Journals*, https://journals.sagepub.com/doi/full/10.1177/1729881419844148.