Ballista – ein Traum von Alexander

Hannan Javed Mahadik, Mechatronik
Otto-von-Guericke-Universität Magdeburg

Abstract - Robots and machine are taking over a lot of manual work, earlier done by men, and making it better by being not only quicker in doing the same task, but also in a much more efficient way.

This paper is heavily based on the Ballista (Catapult) used by many great rulers including Alexander the Great in a bid to win wars. It is an engineering marvel built to launch projectiles across great distances inflicting quite some damage. Depending on the needs, a catapult can be modified and used to destroy or create an opening in mountain ranges or just simply as a war weapon. We have created a miniature, albeit working prototype of the same using the Lego Mindstorms NXT kit and programming of the same using MATLAB.

Thanks to the programming kit made by students of RWTH

Aachen, which is available for public use, programming of our Ballista / Catapult model was made much easier and efficient.

Keywords – Autonomous, Ballista, MATLAB, Lego Mindstorms NXT kit, Boolean logic

I. INTRODUCTION

One of the main reasons for building an autonomous model of the great Ballista is naturally efficiency. The Ballista should be able to move on its own, and additionally be astute enough to sense when there subsists an obstruction in its path and then turn accordingly in either the left or right direction, unless asked to halt by the user incharge. Naturally, to obviate any accidents due to the shortcomings of the technology we have available today, the final kineticism, i.e. of launching the projectile, was to be made manual by using a simple and reliable button.

Engendering an aperture in a mountain range may not sound as important as it actually is. Doing so can greatly cut down the transportation time of humans, which is especially important when lives are at stake as immediate medical attention may not be available in that segment of society anteriorly disunited from cities/towns due to the mountain range. Also, in reducing the transportation time, and the distance travelled, will help financially, as it will not only reduce the costs incurred by utilizing lesser amount of petrol, but it is also sustainable as the environment won't be as damaged.

This was the main aim of our project as we tried to engineer a sustainable and useful machine.

II. PREVIEW

A. *Use of Boolean Logic*

The prototype engineered heavily relies on simple Boolean logic to decide whether it is possible to keep moving forward or if it requires to turn in either direction due to an obstruction or hinderance in its path. Utilizing either 0 or 1, signals are sent to the NXT Motors and processed accordingly. As the model only needed to decide if an obstruction subsisted or not, Boolean was not only logical, but withal efficient.

The turning of the ballista was made possible by simply having one motor run in one direction, and the other in the opposite, ensuring that the result is quick and desirable.

B. *Basic Programming*

MATLAB was to be used as the primary software to program the 'brain' of the Ballista.

The functions were to be independently written, tested, and then concatenated to ascertain stability and efficiency.

III. BODY

A. *Brainstorming*

At the very beginning, the group reviewed the current literature available to come up with a fairly different, and exciting idea.
Out of the many impressive models, the Segway Robot Model was the one that seemed the most impressive as it had to balance itself on its 2 wheels, and also while it was something the general audience could easily relate to. Using this Segway model, the initial idea was

DOI: 10.24352/UB.OVGU-2019-044

to build a prototype that might help disabled people by implementing a similar mechanism into automatic wheelchairs, making them 'smart' and hence eliminating the risk of them tipping over and causing unnecessary accidents.

Figure 1: The original Segway Model [1]

However, this was not to be chosen due to a lot of technical as well as mechanical issues which proved to be a deciding factor in picking out the final project idea.

B. *Metamorphosis*

So as to not have the efforts put in go to waste, the Segway model was dismantled and converted into something different, which was then to be used as the main project. The 'body' of the Segway rider was used as the main motor responsible for the 'arm' of the Ballista, while the 'head' of the same, which was basically an Ultrasound

Sensor, was used to detect obstacles as shown in the adjoining figure(2).

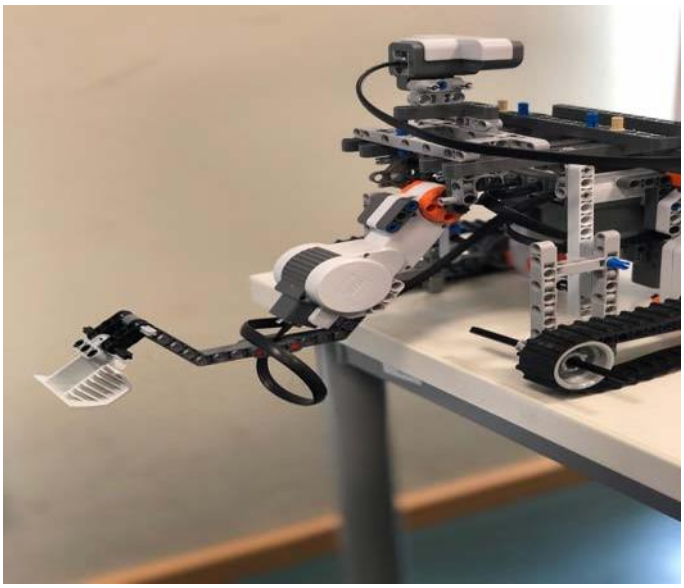

Figure 2: The 'Segway Body' used to build the arm

C. *Building the Ballista*

As mentioned earlier, the 'head' of the Segway rider, an Ultrasound sensor, was attached to the top of the Ballista model, so as to identify any obstacle that may lie in the path of the prototype. The sensor sends signals to the robot according to the distance at which the hinderance stood, ensuring that the Ballista did not simply try to knock the obstacle away, but instead stop and turn in a different direction in a uniformly manner.

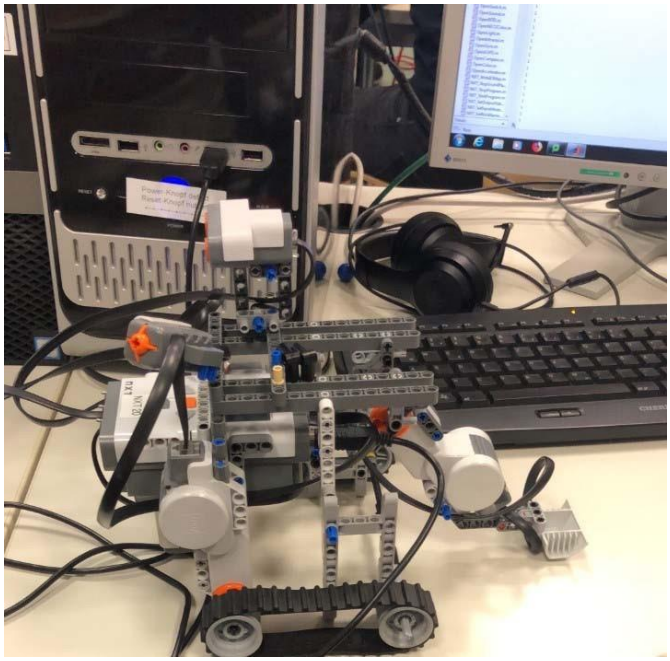

Figure 3: First build

An arm made with a simple Lego pieces were attached to the motor, with the motor programmed to run at full power to ensure proper projectile motion, having adequate speed ensured that the projectile travels the required distance quickly, and has desired impact/effect. A

touch sensor was attached to the prototype to send a signal to the Lego Mindstorms brick to activate the catapult arm.

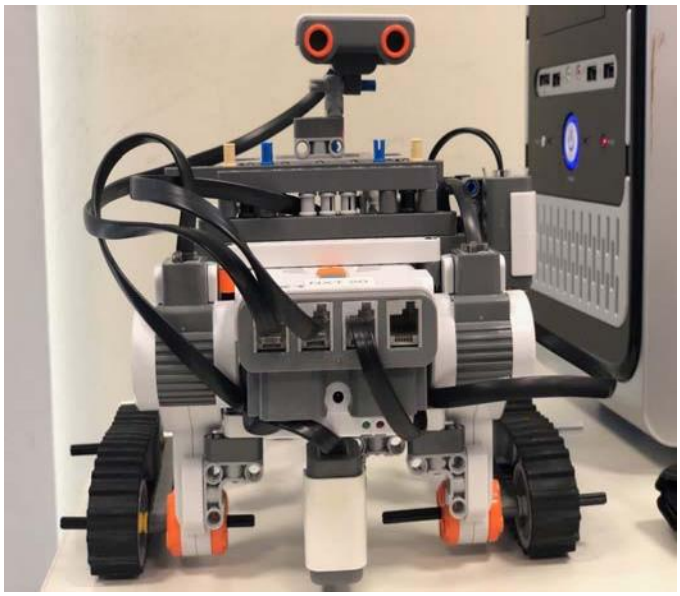

Figure 4: The Ultrasound sensor placed at the top of the model

Originally, four wheels were attached to either side of the Lego Mindstorms NXT brick. This however, affected the way the

Ballista's movement and stability. To overcome this issue, caterpillar tread tracks were used as replacement. This proved to be a vital decision as the prototype was now able to even tread through rougher terrains without running the risk of it tipping over.

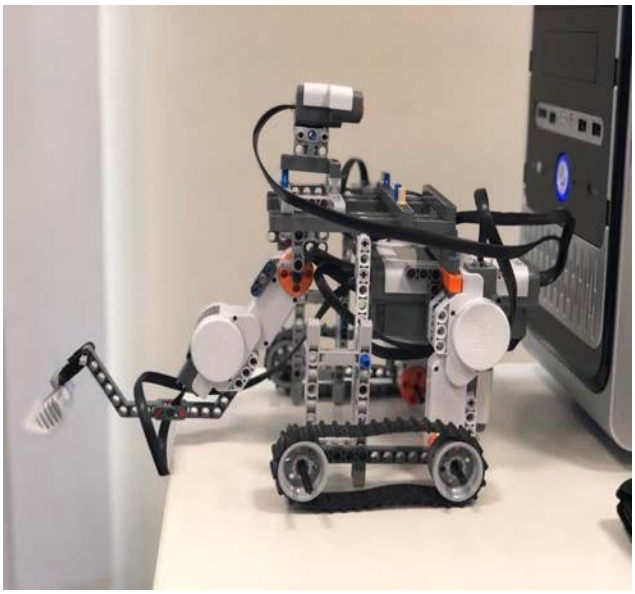

Figure 5: The Caterpillar tread tracks

The next hurdle was deciding how we could make the Ballista recognize edges. This was needed to ensure that the Ballista did not simply fall over, and damage or waste the resources used. This was done by simply relying on a light sensor, which was to be attached to the prototype in such a way that it pointed directly downwards. This was done to measure the reflectivity of the surface that the Ballista was to drive upon. In case of a cavity, the intensity of the light reflected back would not be as high as the one predefined as per the written code. This would then signal the Ballista not to move any further and instead to turn around in either direction and then proceed accordingly.

Therefore, to summarise, the main components of the model were:

- One Ultrasound Sensor
- One Touch Sensor
- One Light Sensor
- Three Motors
- Tread tracks

D. Programming

All of the required programming was done on MATLAB using the predefined NXT functions released by RWTH Aachen.

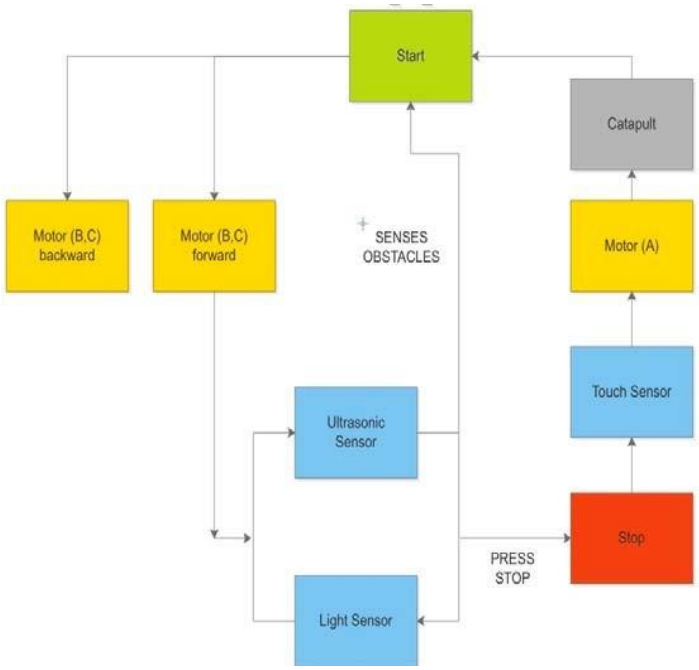

Figure 6: The Program Layout

The program was planned out to be written in parts as seen above to ensure efficiency, and also save time, as more than one person could simultaneously work on different functions. It is efficient because, in the case of an error, it is easier to identify and correct it when the code is separated in parts, as compared to debugging one big block of code.

```
OpenSwitch(SENSOR_2);
OpenUltrasonic(SENSOR_1);
OpenLight(SENSOR_3,'ACTIVE');

while 1
    % if switch is pressed he shoots with the catapult
    if GetSwitch(SENSOR_2) == 1
        handles.motorA = NXTMotor('A','Power',-100,'TachoLimit',100);
        handles.motorA.SendToNXT();
        pause(1);
        handles.motorA = NXTMotor('A','Power',10,'TachoLimit',100);
        handles.motorA.SendToNXT();
    end

    dist = GetUltrasonic(SENSOR_1);
    fprintf(['Distance:' num2str(dist) newline]);
    % drives forward until the distance is smaller than 15
    if dist < 20
        handles.motorA = NXTMotor('A','Power',-50,'TachoLimit',100);
        handles.motorA.SendToNXT();
        pause(0.3);
        handles.motorC = NXTMotor('C','Power',-50,'TachoLimit',475);
        handles.motorC.SendToNXT();
        handles.motorB = NXTMotor('B','Power',50,'TachoLimit',475);
        handles.motorB.SendToNXT();
        handles.motorB.WaitFor();
    end
    pause(0.2);

    x = GetLight(SENSOR_3);
    pause(0.1);
    if x < 600
        handles.motorC = NXTMotor('C','Power',-60,'TachoLimit',300);
        handles.motorC.SendToNXT();
        handles.motorB = NXTMotor('B','Power',-60,'TachoLimit',300);
        handles.motorB.SendToNXT();
        pause(2);
        handles.motorC = NXTMotor('C','Power',-50,'TachoLimit',475);
        handles.motorC.SendToNXT();
        handles.motorB = NXTMotor('B','Power',50,'TachoLimit',475);
        handles.motorB.SendToNXT();
    end
```

Figure 7: Snapshot of the Functions for the Sensors

The above attached image consists of the most important part or the 'heart' of the program. The 'while loop' makes sure that the program runs infinitely until the brick is manually shut down by the user.

The first if-else function reads the signal from the Touch sensor which relies on Boolean logic as its 'condition'. The Tacholimit ensures that the arm of the Ballista doesn't actually hit the body of the Robot and cause a malfunction. It is set to full power (i.e. 100) so that the Projectile takes flight with enough speed and power necessary to get the desired result upon impact. Thereafter, the power is set to 10, but in the opposite direction, which helps the arm to return to it's original position.

The following two functions are responsible with the movement of the robot. The Tacholimit here was set to 475 after carefully examining how many rotations the motor needed for the Ballista to rotate by 90 degrees. Albeit the Tacholimit in the next function is set to the value of 300, it does not provoke any change as the value is arbitrary and is only used to signal the prototype to move in the opposite direction and not over the brink of the surface in question.

IV. RESULT

After many runs and re-runs of the project, it is safe to conclude that the original goal set was achieved and the expectations met, if not exceeded. One thing which could be improved upon is the arm. A motor with higher power could ensure that the projectile travels with an even greater speed, and hence, have an even better impact.

Other than the aforementioned, the sensors could be further improved upon, technically, to reduce any errors. However, doing so would only make a minor difference in the functioning of the robot.

V. SUMMARY & CONCLUSION

The Ballista built by the ancient Romans was truly such an extraordinary piece of machinery. A weapon with great potential to be critically devastating in wars.

Using simple Boolean logic, the prototype built was able to traverse simple courses, that included few basic obstacles, large enough for the Ultrasound sensor to sense, and for the Ballista to then process and proceed accordingly. Also, the possibility of the Ballista traversing and falling over edges into pits or likewise was eliminated.

With a camera, instead of the ultrasound sensor, and with a few changes in the code, the Ballista could be built 'smarter'. It could be programmed to identify the type of obstacle and decide if it is an actual physical barrier of importance, or is it something that has no value, and could simply be driven across.

Of course, with the right implementation of technology, this prototype could be modified in infinite number of ways, and then recreated to a life-like scale. The possibilities are endless.

BIBLIOGRAPHY

1) FEIT OvGU (2019- February)[Online], Unterlagen zum LEGO Mindstorms Praktikum in der FEIT

2) [1] - http://www.nxtprogram.com/NXT2/segway/