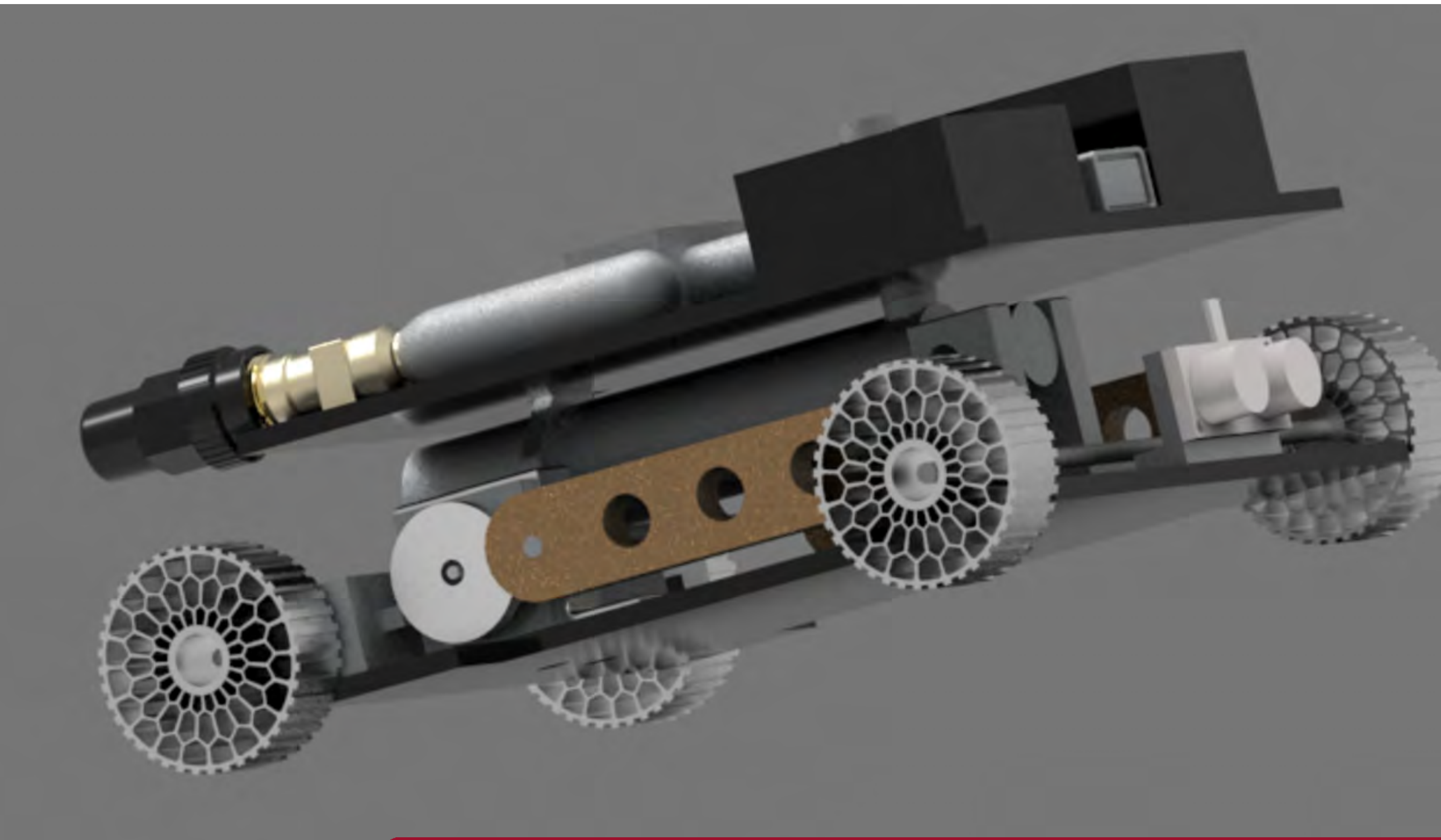
EXAMENSARBETE INOM MASKINTEKNIK,
GRUNDNIVÅ, 15 HP
STOCKHOLM, SVERIGE 2021

Pneumatic jumping car
Pneumatiskt hoppande bil

FELIX EKMAN

SOFIA HANSSON



KTH
SKOLAN FÖR INDUSTRIELL TEKNIK OCH MANAGEMENT

Pneumatic jumping car

FELIX EKMAN AND SOFIA HANSSON

Bachelor's Thesis at ITM
Supervisor: Nihad Subasic
Examiner: Nihad Subasic

TRITA-ITM-EX 2021:43

Abstract

As new challenges presents themselves continuously in today's society the need for autonomous driving solutions is ever growing and with it: the need for creative solutions. This thesis investigates the pneumatic solution to making a car jump. The analysis of theoretical and testing results have demonstrated a correlation between the weight of the car and height of the jump. With the selected configuration it has been proven unattainable to make the car jump. Further research should look into the optimization of the pneumatic system, specifically the mass flow rate throughout the whole system.

Keywords— Mechatronics, Pneumatic, CO_2, Arduino, Robotics

Referat

Pneumatisk hoppande bil

I dagens samhälle presenteras nya utmaningar dagligen och behovet av autonoma lösningar i fordon ökar ständigt. I takt med det ökar behovet av kreativa lösningar. Detta examensarbete undersöker pneumatiska lösningen bakom att få en bil att hoppa. Analysen av teoritiska och experimentiella resultat visar en korrelation mellan bilens vikt och den möjliga hopphöjden. Med den valda konfigurationen är det inte möjligt att utföra hoppet och åtgärder tas upp i disskusionen. Vidare forskning bör undersöka optimeringen av det pneumatiska systemet, specifikt på hur det maximala massflödet kan uppnås i systemet.

Keywords— Mekatronik, Pneumatik, CO_2, Arduino, Robotik

Acknowledgements

We would like to express our sincere gratitude to our advisors Prakhar Joshi, Sai Kaushik Abburu and Paolo Ascia for providing their invaluable guidance, comments and suggestions throughout the project.

We would also like to thank Tomas Östberg for assistance with his expertise in welding, Nihad Subasic for his comments and suggestions throughout the project and Staffan Qvarnström for guiding us in the Mechatronic's laboratory at KTH.

Contents

Contents

List of Figures

Abbreviations

CAD Computer aided design. 5, 16–18

DC Direct current. 3, 4, 14, 15, 18

EM Electrical motors. 4

IDE Integrated development environment. 3, 15

PLA Polylactic acid. 5, 16

PWM Pulse width modulation. 3, 4

Nomenclature

J Joules. 20

K kelvin. 20

MPa megapascal. 5, 10

N Newton. 10, 11

Nm newton meter. 14, 27

PSI pounds per square inch. 10

V volts. 3, 14, 15

$^\circ$ degrees. 12, 14, 23

$^\circ C$ degrees Celsius. 10

bar bar. 10, 11, 20, 23

cm centimeter. 20, 22, 25

g gram. 1, 10, 20, 25

kPa kilopascal. 10

kg kilo gram. 18, 22, 25, 27

mm millimeter. 9, 18, 20

mol mole. 20

Chapter 1

Introduction

This chapter includes the background, purpose and the scope of the thesis.

1.1 Background

Autonomous robots first appeared in 1948 with Elmer and Elsie [25] and have ever since exceedingly evolved in the fields of spaceflights, private households, delivery services and military. To the present day its' use has mainly been for indoor navigation and outdoor navigation through air however, there is an increased desirability for more autonomous robots to navigate outdoors.

With many new challenges emerging such as that of rapidly spreading epidemics, natural disasters and explosions, it has become apparent that there is a need to minimize exposure to harm for workers in dangerous situations. Many of these situations often require movements in complex environments which is one of the unsolved problems to ground vehicles.

1.2 Purpose

The purpose of the thesis is to build a fully functional robot car that can launch itself over obstacles in order to facilitate the movements of ground vehicles. The thesis investigates a pneumatic solution to formulate the jump of a four-wheeled car, including the technical solutions to the design, manufacturing and assembly of the complete car. The research questions this report investigates are:

What is the maximum height the car can reach?

How many jumps are feasible with a 16 g CO_2 cartridge?

1.3 Scope

The thesis focuses on the configuration of a pneumatic solution to making a four wheeled autonomous car jump within a time period of three months. A secondary focus involves the autonomous driving mechanisms. The resources of the study consist of existing available materials at the department of Mechatronics as well as a budget of 1000 SEK.

Chapter 2

Theory

This chapter includes the theory of all necessary components.

2.1 Microcontroller

To control the electronics in the study and interpret the physical world with analog inputs, a microcontroller is required. For this study an arduino uno was chosen for this purpose as they are inexpensive and easy to use. The arduino that was chosen has 6 analog input pins and 14 digital in- and outputs of which 6 provide Pulse width modulation (PWM) [5].

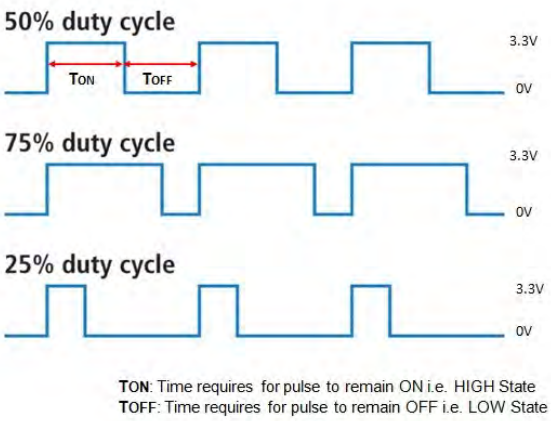


Figure 2.1. Illustration of PWM [24].

PWM is necessary to vary DC voltage and is done by sending out current as pulses and by varying the length of the on- and off-time, the average current can be changed [17]. See Figure 2.1. This will be used when controlling the electrical motors with the microcontroller. The arduino uno has a recommended input voltage of 7 to 12 V that is suitable for the demands of our circuit size [5]. The code uploaded to the Arduino board is written in the Arduino Integrated development environment (IDE) and is based on C and C++ [15].

2.2 Electrical motors

The Electrical motors (EM) are used to convert electrical energy to mechanical energy. Most commonly it operates through the interaction between the motor's magnetic field and its electric current in the wire winding which generates force in the form of torque [13].

2.2.1 DC motors

Brushed DC motors mainly consist of the stators, rotors and brushes whilst the brushless DC motors mainly consist of stators, rotors and a permanent magnet [17]. When the electrical current is applied to a brushed DC motor, the stator magnets (north and south) stay fixed while the coil rotates 180°, the soft contacts, also known as brushes, make sure it only rotates in one direction. Brushless DC motors develop their torque by alternating the polarity of the rotating magnets and the stationary magnets on the stators. There are multiple benefits of using brush-less DC motors over a brushed DC motor. One of the benefits of using a brush-less DC motor is that they do not need the brushes, which wear down quickly over time [20], and therefore have an increased efficiency (producing more torque per watt). Another benefit is their high torque to weight ratio which would be advantageous to constructions in need for lighter weights. DC motors typically operate between 6 Volts to 700 Volts[14].

2.2.2 Stepper motors

Stepper motors converts pulsing electrical current into precise one-step movements. They are examples of brush-less motors however, compared to DC motors, they have their permanent magnets on the inside that rotate whilst the coils are on the outside and static. Another difference to basic DC motors is that the stepper motor can stay still at any given angle; holding torque [6].

2.3 Motor Driver

The motor driver consist of an electronic circuit that alternates between the polarity of a voltage applied to a load by activating and deactivating transistors [26]. H-bridge uses electromotive force to control the current through the motor. This allows the motors to rotate in both ways depending on where the current is applied which in turn allows speed control via PWM. The difference in applied current in the two wheels determines the direction of the car[27].

A dual H-bridge motor driver allows for speed and direction control of two DC motors at the same time or one stepper motor and can operate between 5V and 35V with a maximum current up to 2A [10].

An arduino motor shield is a dual full-bridge driver designed with two motor channels in H-bridge allowing DC motors, solenoids, relays and stepper motors to be controlled [4]. This allows two DC motors to be driven independently.

Key parameters to take into account when choosing an H-bridge:

- Stall torque: Maximum torque required to prevent rotation
- Stall current: Maximum current required to prevent rotation
- Rotational speed [rad/s]

2.4 Additive manufacturing

3D printing is a manufacturing process where the print-head deposits thermoplastic that quickly sets in successive layers of a model. 3D printing is used for making models and prototypes of CAD systems. All 3D-printed parts are made with Polylactic acid (PLA) which has a density of 1250 kg/m^3 and tensile strength between 55 to 72 MPa [19]. The strength of the 3D printed item depends on several variables such as infill percentage, extruding temperature and build speed [9]. Since the total weight of the car determines the height of the car's jump it is important to keep the infill percentage as low as possible.

2.5 Ultrasonic sensors

An ultrasonic sensor measures the distance to the obstacle. The sensor emits a sound wave and with the speed of sound in air and the time it takes for the sound to be received, the distance can be calculated [18].

2.6 Pneumatic cylinder

A pneumatic actuator is a device that translates a source of static power into useful output powers, converting energy to useful motions. Pneumatic actuators utilizes the power of compressed gas and transforms it into force in a linear motion by using a piston [3].The two most common pneumatic actuators can be subdivided based on their operating principles: single-acting and double-acting. A double-acting cylinder creates a pushing and pulling motion by using two ports for compressed air to enter and exit, while a single-acting cylinder uses one port for the compressed air to enter and an internal spring to return the piston to its' initial position [7]. Some pneumatic cylinders come with a safety exhaust orifice in case of failure in the pneumatic system.

Pneumatic cylinders come in multiple different shapes depending on the application. Important factors to take into consideration when choosing a pneumatic cylinder is the required force, speed and air consumption determined by the key parameters of stroke size, bore size, rod diameter and system pressure.

The theoretical force [8]of the actuators can be calculated using the pistons source area (A) and the supplied air pressure (P) as followed:

$$F = AP \tag{2.1}$$

The actual force applied to the load will usually be less than the calculated force and therefore a safety factor of 5 or 6 on supplied pressure should be taken into consideration [8].

In order to predict the true load of the cylinder it is important to include the force loss due to sliding friction which in turn must include the friction factor:

$$F_f = (P_f - P_r)A + F_{lc} \tag{2.2}$$

Where P_f is front pressure, P_r is rear pressure and F_{lc} is the weight of the car.

Important to make note of that the speed can concurrently be affected by multiple other factors such as the size of ports, tubing and rate of flow of the control valves. There are many factors such as system contamination, corrosion, minor leaks and wear that will affect the system's air pressure and flow.

2.7 Solenoid valve

A solenoid valve uses magnetic force to operate and converts electrical signals to pneumatic functions. The magnetic force displaces the valve body against the return spring, consequently letting the flow of compressed air through the valve [23]. In our case it is used to actuate a pneumatic cylinder.

A solenoid has two normal states:

- N/C (Normally closed), the valve remains closed when not actuated with a current

- N/O (Normally open), the valve remains open when not actuated with a current

There are multiple different valve types, the difference is mainly dependant on the number of ports and amount of states.

2.8 Pressure regulator

A pressure regulator works with a spring actuated valve that acts as a diaphragm. The pressure flow inside the regulator activates the diaphragm, as the increased pressure reaches the set pressure, the valve plug closes by compressing the spring [12].

Factors to take into consideration when selecting a pressure regulator:

- operating pressure rate
- nature of medium
- inlet pressure that is going into the regulator
- set pressure
- flow rate required

Chapter 3

Demonstrator

This chapter includes the choice of components, design and assembly of each subsystem, and the analysis of the research questions.

3.1 Subsystems

This section goes through the choice of components, schematics of the system and testing of each subsystem.

3.1.1 Pneumatic system

One of the solutions to a jumping mechanism for the car is to use a pneumatic cylinder, transforming compressed gas into linear motion applied on the piston. However, pneumatic cylinders are often operated using large pressure reservoirs. In order to create a smaller, lighter and more suitable source of CO_2 for the cylinder our solution to the jumping mechanism takes inspiration from the design of a gas operated gun [16] by using a CO_2 cartridges as the source to a pneumatic cylinder.

Gas operated guns fire projectiles using pneumatic potential energy within compressed air that is, prior to the use, mechanically pressurized and stored within the gun. When firing the gun the compressed air is released through valves. CO_2 powered compressed guns use detachable pressure reservoirs, such as powerlet cartridges, that are filled with already pressurized CO_2 gas and are usually for one time use.

This solution was found to be promising and the decision to proceed with this was made.

Design iteration I

A first iteration of the pneumatic system was proposed in order to propel the car, operating using the arduino uno to control the solenoid valve. The assembly consisted of a pneumatic cylinder, a solenoid valve, a pressure valve, a CO_2 cartridge, fittings, a pressure hose and pneumatic tubing to connect the components, see Figure 3.1.

The selection of a pneumatic cylinder was based on the size of the bore, by maximising the bore area one concurrently maximises the force of the pneumatic cylinder as they are proportional to each other, see equation 2.1. As the pressure was unknown at the time, it was kept as a constant. With the given budget a single acting cylinder with a bore size of 25 mm and a stroke length of 50 mm was procured, see component specifications in table

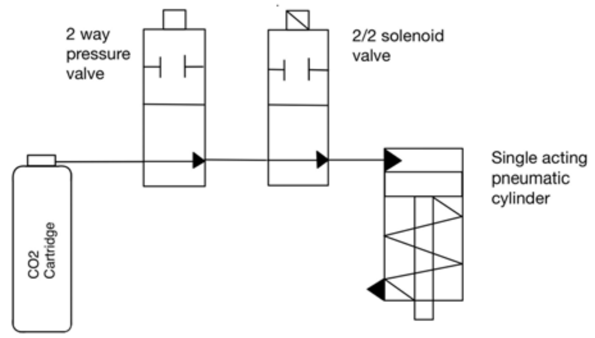


Figure 3.1. Pneumatic diagram of the first design iteration. Made in *Notability*

Figure 3.2. Pneumatic diagram of the first design iteration. Made in *Notability*

in appendix A. Compatible pneumatic plug-in connectors were simultaneously procured for connecting components to the pneumatic tubing.

The optimal CO_2 pressure for a pneumatic system, with respect to the CO_2 consumption, is 1.42 MPa [11], a 16 g CO_2 cartridge was selected for which the general standard at 20°C the gas reaches a pressure of 852.8 PSI which in turn would allow for the optimal pressure to be obtained when regulated. A N/C direct operated 2 port solenoid valve was selected, as it was available at the mechatronic's laboratory's inventory.

A theoretical force was calculated, using equation 2.1, to 294.52 N, assuming the pressure inside the cylinder to be the maximum operating pressure of the pneumatic cylinder (6 bar, 1 bar = 100 kPa). The true load of the cylinder was calculated, using equation 2.2, to be 263.621 N using a front pressure of 6 bar, rear pressure of 1 bar (Atmospheric air pressure) and the mass of the car. With the estimated true load calculated with the parameters above, one could expect the car to reach a reasonable height during launch.

This design was intended to be tested on by quickly opening the pressure valve, ahead of an excessive pressure build up between the valve and the solenoid, before releasing it by actuating the solenoid. Due to the use of a 2 port solenoid, the exhaust of the pressure had to be done manually which would be disadvantageous as it would prevent the vehicle to be fully autonomous.

While experimenting with the pneumatic cylinder (see Figure 3.2) without a regulator there was a pressure build up that caused the pneumatic tubing to burst, as a result of this, a second design iteration was developed.

Design iteration II

For the same reason as to why the first design iteration was developed, the second iteration was developed with an additional focus on including a pressure regulator. This assembly consists of a pressure regulator, CO_2 cartridge, relevant fittings, an additional gauge, a solenoid, pneumatic cylinder and pneumatic tubing, see Figure 3.3.

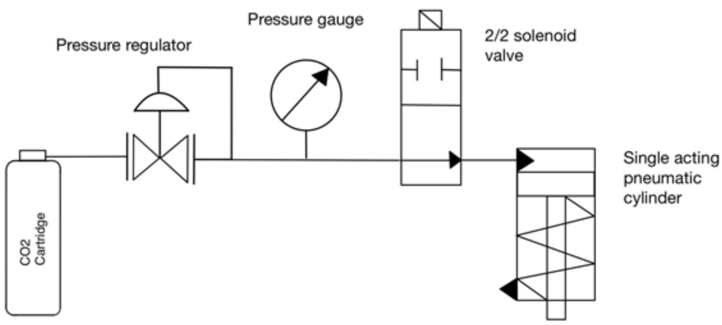


Figure 3.3. Pneumatic diagram of the second design iteration. Made in *Notability*

All components procured for design iteration I were kept except for the pressure valve, this was replaced by a pressure regulator. The pressure regulator was selected with respect to the specifications of the previously selected components, the geometrical constraints of the chassis and the budget constraints. The procured pressure regulator had multiple constraints including an operational pressure range between 0-2 bar. The maximum operating pressure of the regulator sets the maximum operating pressure of the whole system.

The theoretical force was calculated to 98.175 N with an operating pressure of 2 bar using equation 2.1. The estimated true load was calculated as 67.3 N, using equation 2.2, which is ca. three times the weight of the car.

Figure 3.4. Pneumatic diagram of the second design iteration. Taken by authors.

During the experimental phase of design iteration II an additional gauge was implemented to ensure the correct pressure value within the system, see Figure 3.4. However, the gauge was not included in the final assembly.

3.1.2 Motorized levers

The car was positioned at an appropriate angle in preparation for the jump with a pair of levers powered by a stepper motor. The levers were attached to the ends of an axle, which in turn was connected to the stepper motor using gears. The levers were designed with lightening holes to reduce the weight of the car. The complete subsystem can be seen in Figure 3.5.

Via gears, power was transferred to the axle and by utilizing the gear ratio the output torque was increased.

$$Gearratio = \frac{z_1}{z_2} = \frac{M_2}{M_1} \tag{3.1}$$

Index 1 and 2 denotes the in- respectively output gear. z denotes the number of teeth on the gear and M the torque. With equation 3.1 the gear ratio was calculated to 2.18; the factor the output torque will be increased by.

The stepper motor is a Tamagawa TS3214N61; a universal stepper motor with six wires. The motor has 200 steps per full revolution which is 1.8 ° per step. This stepper motor was selected as it was available for free in the mechatronics laboratories inventory. There were no available data sheets as it was an outdated product, so the performance of the motor needed to be tested experimentally. As high torque was critical to raise the car, the rotational speed was held as low as possible to increase torque [21]. The torque was increased further by ignoring the center taps and the motor was used as a bipolar stepper motor. This enabled the motor to have a higher torque since the entire coil would be energized instead of half.

To determine the necessary output torque of the motor during the lift, a free body diagram was drawn which is displayed in Figure 3.6. From the diagram, equation 3.2, 3.3, 3.4 and 3.5 were identified.

$$x_1 = L_1 cos(\alpha) - h_1 sin(sin\alpha) \tag{3.2}$$

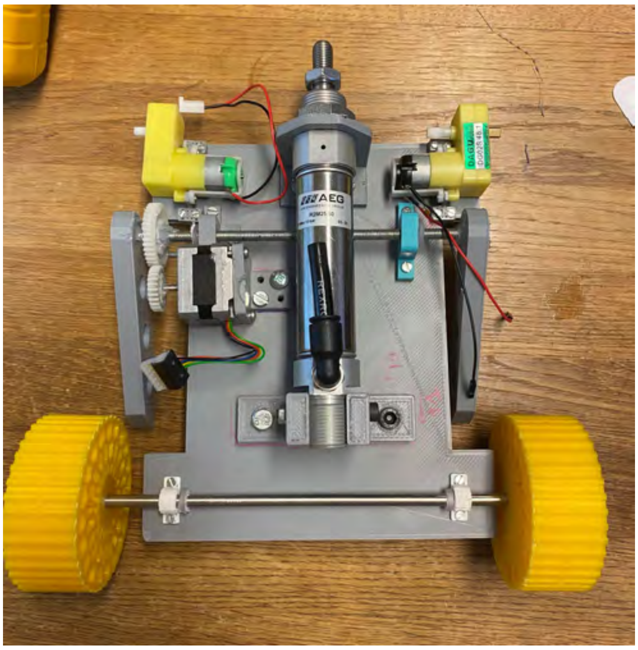


Figure 3.5. First floor assembly with the motorized levers for positioning the car at an angle. Taken by authors.

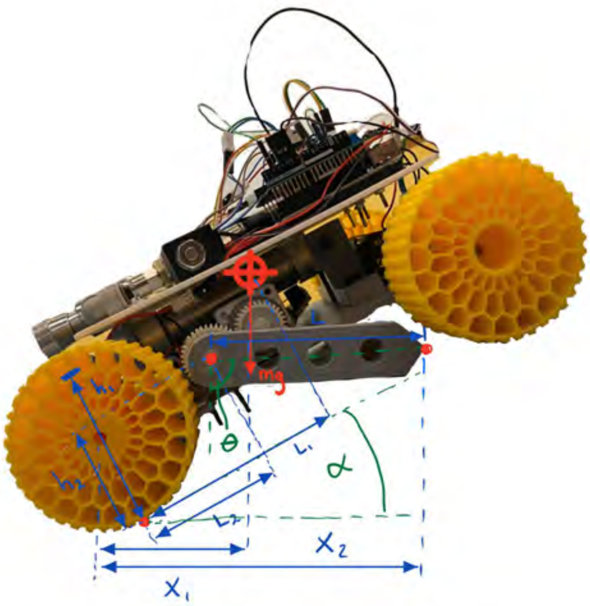


Figure 3.6. Free body diagram of the car seen from the side. Made in *Notability*

$$\theta = cos^{-}1(\frac{h_1cos(\alpha) + L_2sin(\alpha)}{L}) \tag{3.3}$$

$$x_2 = (L_2 - (h_2tan(\alpha)))cos(\alpha) + Lsin(\theta) \tag{3.4}$$

$$T = \frac{mgx_1L_1}{sin(\theta)x_2} \tag{3.5}$$

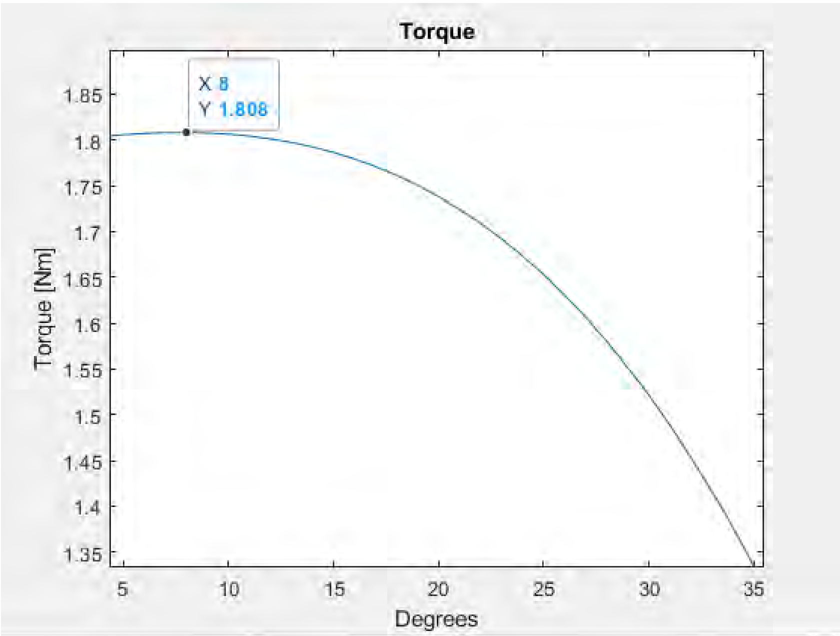


Figure 3.7. Torque plot. Made in *MATLAB*

According to equation 3.5, the torque necessary to raise the car will vary with the angle of the car. MATLAB was used to solve the equation 3.5 with the tilt angle varying between 0 to 90°. From the calculation the conclusion was drawn that the maximum torque needed was 1.8 Nm and that occurs at an angle of 8°. The resulting graph from the calculations can be seen in Figure 3.7. Experiments on the subsystem showed that the torque from the stepper motor was insufficient to either lift the car or hold it at an angled position.

3.1.3 Control of the car

Circuit

The circuit consisted of a microcontroller, two DC motors, one stepper motor, two motor drivers, a solenoid, a transistor, a resistor and an ultrasonic sensor. The configuration can be seen in Figure 3.8. The two DC motors were controlled by the arduino motor shield and designed as in-wheel motors to the back wheels of the car. The motors were powered with the 12 V input from the external batteries. The stepper motor that powers the levers was driven with the L298N motor driver which was also powered by 12 V. The solenoid valve that controls the flow of carbon dioxide in the pneumatic system was controlled by the Arduino Uno in a circuit consisting of a transistor, resistor and a diode. The ultrasonic sensor was used for detecting obstacles, powered directly from the 5V pin on the arduino and it communicated with it via a digital pin.

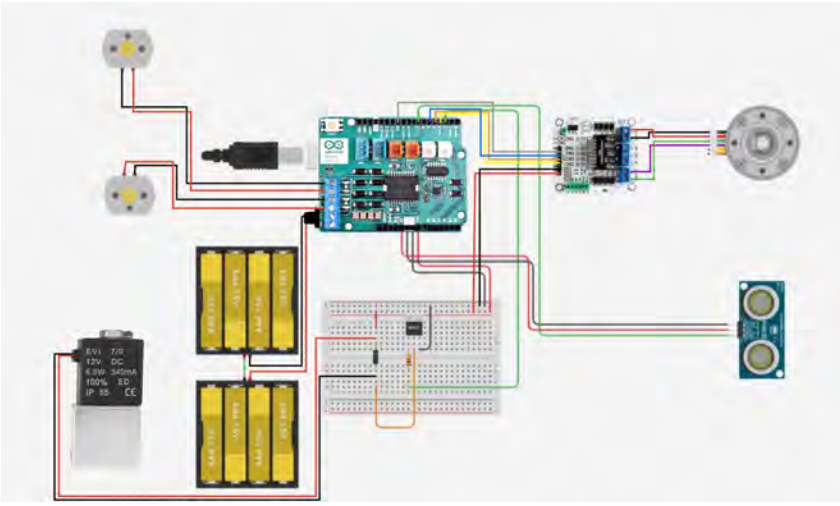


Figure 3.8. The circuit. Made in *Tinkercad* and edited in *Notability*

The complete circuit was powered by eight serial connected 1.5 V AAA batteries which equals to 12 V. This was suitable as there was a 12 V requirement for actuating the solenoid valve and other electrical components. AAA batteries are common and could easily be replaced as they were placed in a plastic mounting attached to the chassis.

Programming

The code controlling the car was written in the arduino IDE. Initially individual codes were written and tested for subparts of the system. These included the solenoid valve, the DC motors and the stepper motor. They were later integrated together to form the final version of the program. The code is written with a "switch case" control structure which allows different codes to be executed depending on various conditions. This supports future implementation of wireless control of the car. When executing the main program, the car drives forward at a constant speed until it reaches an obstacle at where it stops. The program continues by raising the car to an angle from which it launches by turning on the solenoid valve, allowing air to pass through to the pneumatic cylinder. For safety reasons the program breaks after one iteration and has to be manually be restarted, allowing the user to inspect the vehicle.

3.2 System assembly

3.2.1 Computer aided design

The construction of the design has gone through two computer aided design iterations as we progressed due to changes in our component specifications as well as optimization of our technical solutions. All three dimensional computer aided design models of the car were produced in Solid Edge 2020 [22].

At first the design consisted only of the essential parts required for the car to drive, as the study proceeded additional parts could be added to the model. The procured components such as the electrical and pneumatic components were incorporated to the model along with housings to each one of them. All 3D-printed parts were made using PLA (polylactic acid).

Design iteration I

The preliminary design iteration focused on assembling the rotating stands, the pneumatic cylinder, the wheels and the chassis in order to visualize the solution, see Figure 3.9.

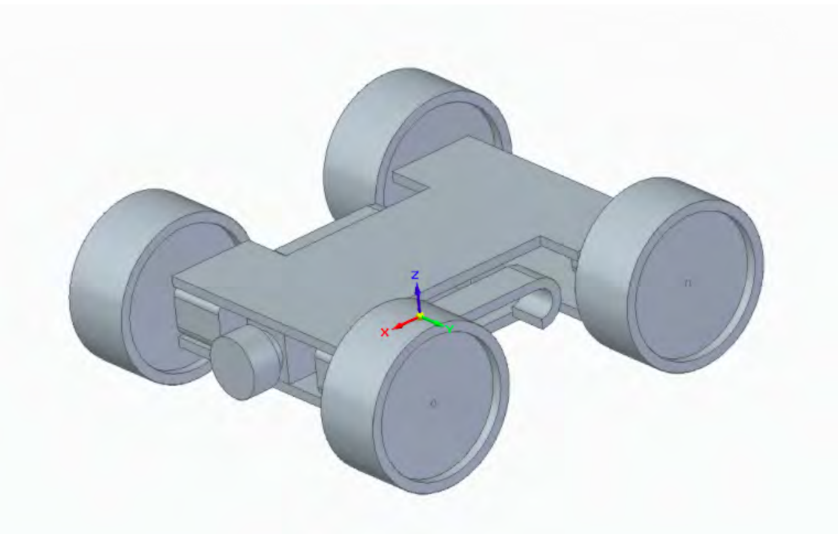


Figure 3.9. First CAD iteration. Made in *Solid Edge 2020*

The construction of this design was created with the intentions to, with the help of the levers, position itself at an angle before launch and subsequently land on either side of the car. See Figure 3.10 for a visual representation of the intended movement. The change of angle would allow adjustments to the projectile motion depending on the height of the obstacles.

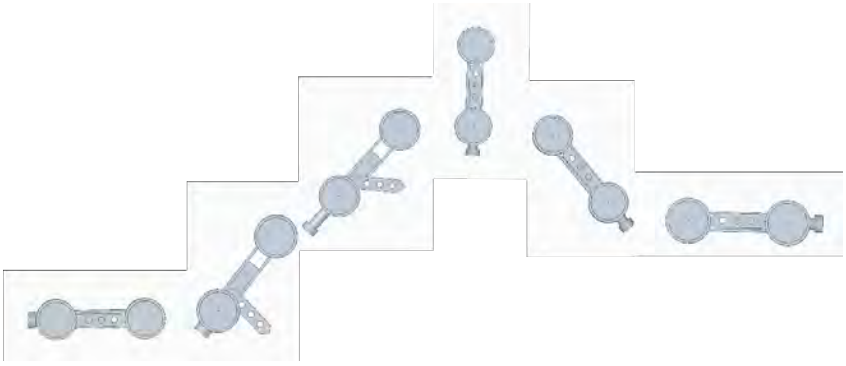


Figure 3.10. The intended movement of the car with design iteration I. Made in *Adobe Illustrator*

Design iteration II

After having received all the necessary component specifications, another design solution was needed in order to accommodate for all components in the assembly. The second design iteration, see Figure 3.11, featured two levels. The assumption was made that by positioning the heavier components further in the back, the center of mass would be positioned favourably with regards to the desired movements. The disadvantage of having two levels would come with the inability of landing on both sides, this had to be taken into account in the analysis of the launch.

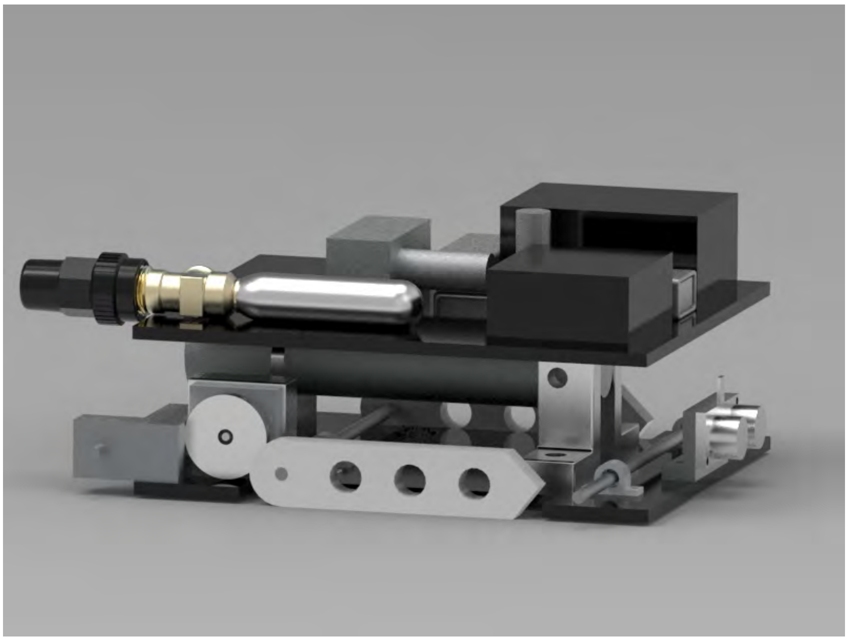


Figure 3.11. Second CAD iteration. Made in *Solid Edge 2020*

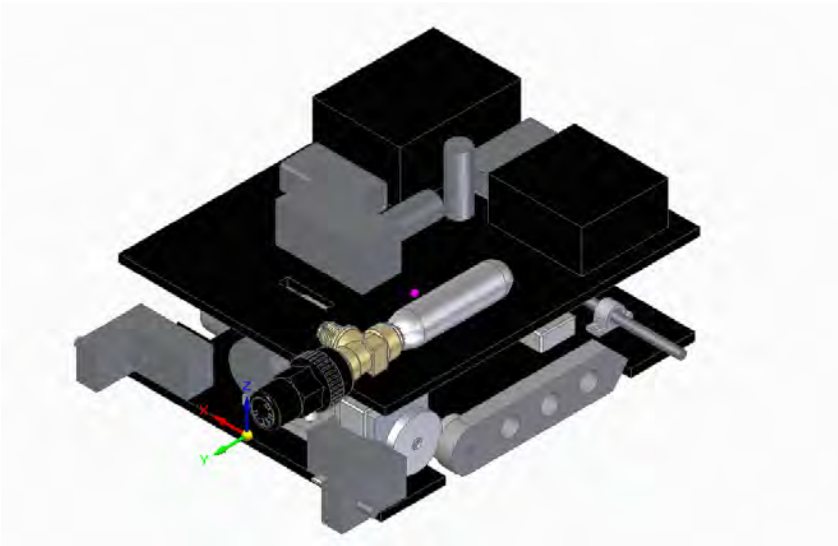


Figure 3.12. Positioning of the coordinate system with respect to the car. Made in *Solid Edge 2020*

The second design iteration included the ultrasonic sensors, DC motors and the front and rear axle, all were procured from the department of mechatronic's inventory. For the manufacturing of the wheels, an already existing design from grabcad [2] was used. Due to the design not being applicable in Solid Edge it has been excluded in this CAD iteration.

The Solid Edge software provides tools to identify the center of mass and total weight of the car by assigning material and weight to the components in the computer aided design. For simplification of the model, the components are considered to have homogeneous masses. The center of mass was found at these coordinates with respect to the positioned coordinate system (see Figure 3.12):

x: -17.32 mm

y: -90.15 mm

z: 43.10 mm

The acquired total weight of the computer aided design added up to 1.952 kg, whereas the physical weight added up to a weight of 1.99 kg.

3.2.2 Assembly

The complete assembly of the car follows the assembly of the second computer aided design iteration with additional mountings, nuts and screws. All mountings were designed to prevent undesired movements and secured with bolted joints. To prevent axial movements in the rear and front axle,adhesive rings were tightly fitted on the axle on both sides of each wheel axle mounting. To facilitate the connections of jumper wires between the two levels of the assembly holes were made in the chassis using a drill press.

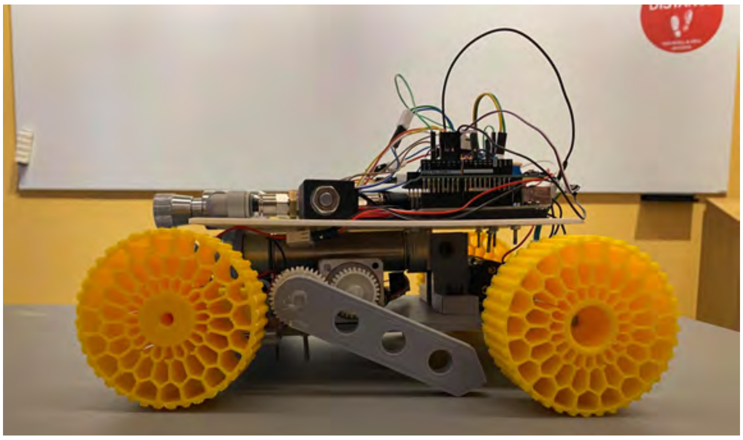


Figure 3.13. The complete assembly. Taken by authors

3.3 Analysis of research questions

The thesis investigates two research questions:

1. How many jumps are feasible with a 16 g CO_2 cartridge?
2. What is the maximum height the car can reach?

3.3.1 Research question I

A hypothesis of how many times the system can actuate the pneumatic cylinder can be made with the following equation:

$$N_p = \frac{V_{CO_2}}{V_{act}} \tag{3.6}$$

N_p: Number of pneumatic actuations, V_{CO_2}: The total volume of CO_2 and V_{act}: CO_2 consumption per actuation. The pneumatic system will operate at 2 bar which was deduced from testing, this in turn gave the maximal velocity of the pneumatic cylinder. The total volume of CO_2 at 2 bar can be calculated with the ideal gas law:

$$pV = nRT \tag{3.7}$$

Where p is the pressure of the gas, V the volume of the gas, n the number of moles, R the ideal gas constant and T the temperature.

The number of moles (n) was determined using the following formula:

$$n = \frac{m}{M} \tag{3.8}$$

Where m is the mass of the gas and M is the molar mass $[g/mol]$. With a molar mass for CO_2 at 44.01 $[g/mol]$ and a mass of 16 g, the number of moles will be 0.36. At room temperature, 298.15K, R = 8.3145J/$(molK)$, n= 0.36 moles the total volume of the CO_2 cartridge at 2 bar would be 4.46 dm^3.

The air consumption per actuation was estimated by calculating the internal volume of the cylinder and the tubes. The internal volume of the pneumatic cylinder can be determined using equation 3.9.

$$Volume = \frac{\pi}{4} b^2 s \tag{3.9}$$

Where b is the bore length and s is the stroke length, the parameters for the selected cylinder can be found in the component specification in appendix A and the volume of the tube was estimated using the inner diameter of 6 mm and a length of 6 cm. The total volume gives us the air consumption of $0.026dm^3$ per actuation. The total volume of CO_2 in the cartridge divided by the consumption gives a theoretical number of 172 actuations per CO_2 cartridge.

3.3.2 Research question II

To reach the maximal height, the car is launched in a projectile motion. A simulation of the projectile motion was made in Acumen[1], see appendix E. To strengthen the hypothesis on the achievable height of the car during launch, the following calculations were made based on the systems in Figure 3.14.

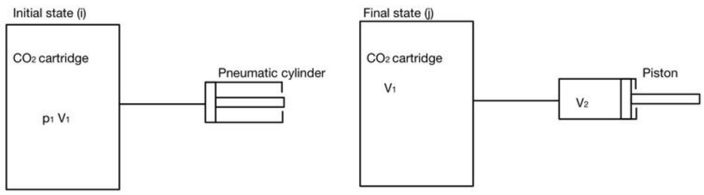


Figure 3.14. The pneumatic system in the initial and final state. Made in *Notability*

Initially, the law of conservation of mechanical energy in the system was used:

$$\pm L = -\int_i^j pdV + \frac{1}{2}\left(c_j^2 - c_i^2\right) + g\left(z_j - z_i\right) - \frac{1}{2}\omega^2\left(r_j^2 - r_i^2\right) + L_{\omega,i-j} \tag{3.10}$$

Where $\pm L$ is external work, p is the pressure, V is the volume, c is the gas velocity, z is the system height, r is the rotational radius and ω is the rotational speed. Indices i and j indicate the initial state respectively the final state. $L_{w,i-j}$ is dissipative work and assumed to be zero as the losses are assumed to be zero. The following was concluded:

$$c_j^2 = c_i^2 = 0 \Rightarrow \frac{1}{2}\left(c_j^2 - c_i^2\right) = 0 \tag{3.11}$$

$$z_j = z_i = z \Rightarrow g(z - z) = 0 \tag{3.12}$$

$$r_j^2 - r_i^2 = r^2 \Rightarrow \frac{1}{2}\omega^2\left(r_j^2 - r_i^2\right) = \frac{1}{2}\omega^2\left(r^2 - r^2\right) = 0 \tag{3.13}$$

Equation 3.11,3.12, 3.13 are substituted in equation 3.10. Therefore, the equation 3.10 can be written as:

$$\pm L = -\int_i pdV \tag{3.14}$$

Assuming ideal behavior of the CO_2, the ideal gas law can be applied:

$$pV = nRT \Rightarrow p = \frac{nRT}{V} \tag{3.15}$$

Where p is the pressure of the gas, V the volume of the gas, n the number of moles, R the ideal gas constant and T the temperature. The only component that does work is the piston, thus the piston work is:

$$\pm L = -\int_i^j \frac{nRT}{V} dV \tag{3.16}$$

Integrating equation 3.16 the following is obtained:

$$L = \int_i^j \frac{nRT}{V} dV = nRT \int_i^j \frac{1}{V} dV = nRT \int_{V_1}^{V=V_1+V_2} \frac{1}{V} dV = nRT \left(\log\left(V_1 + V_2\right) - \log\left(V_1\right)\right) \tag{3.17}$$

Assuming no mechanical losses, all work is transformed into kinetic energy of the car. The kinetic energy then becomes potential energy:

$$L = E_K = E_P = m_c g h \tag{3.18}$$

Where E_K is the kinetic energy, E_P is the potential energy, m_c is the physical mass of the car, g is the gravitational acceleration and h is the height of the jump. By combining equation 3.17 and 3.18 the following expression for h is obtained:

$$h = \frac{L_L}{m_c g} = \frac{nRT \left(\log\left(V_1 + V_2\right) - \log\left(V_1\right)\right)}{m_c g} \tag{3.19}$$

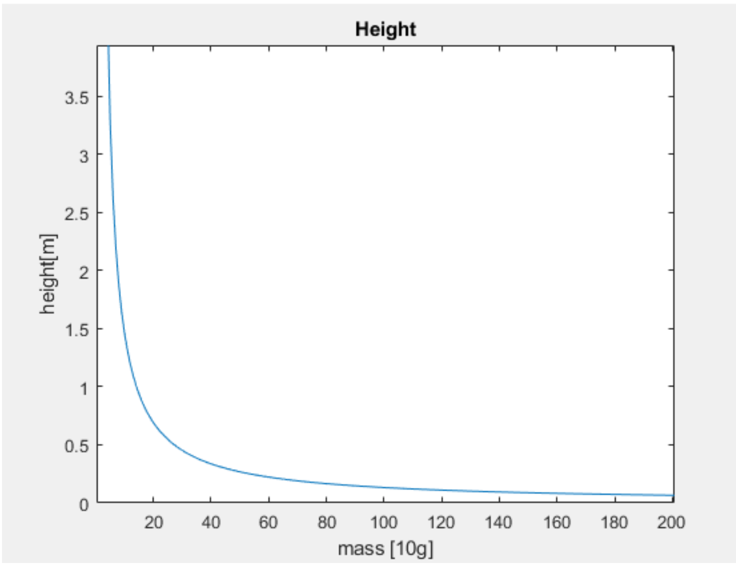


Figure 3.15. The correlation between the height of the jump and the weight of the car. Made in *MATLAB*

With equation 3.19 a correlation between the height of the jump and the weight of the car has been made visible through a graph in Figure 3.15. At the physical mass of 1.99 kg a theoretical height of 6.6 cm was determined.

3.3.3 Testing

To test our research questions two separate tests were conducted. During testing, necessary safety precautions were taken, such as the use of ear muffles, safety goggles and gloves.

Research question I

The number of times the car could jump was tested by actuating the piston as many times as possible with the car at a horizontal position until the CO_2 cartridge had been emptied. The pressure was set to 2 *bar* using the regulator and the solenoid had an actuation of 0.2 seconds.

The result was recorded and was used to determine the number of times the car can perform the jump using one CO_2 cartridge. The test used an arduino program that exclusively controlled the solenoid valve and terminated after one cycle. The test had to be restarted manually using a button for safety reasons and therefore leakage in the pneumatic tubing could be inspected between each actuation. Due to the lack of exhaust channel in our pneumatic system, the safety exhaust orifice of the cylinder was used to dispose of return pressure after each stroke.

Research question II

The maximum height achievable for the car was tested by placing the car at an 85° angle and actuating the pneumatic cylinder. The same arduino program as in the testing of research question one was used. The pressure was set to 2 *bar* and the solenoid had an actuation of 0.2 seconds. The test was conducted 10 times for a more accurate representation of the height and the results were recorded after each test.

Chapter 4

Result

4.1 Research question I

The test was conducted as described in the previous chapter. For the chosen pneumatic configuration the volume of CO_2 consumed per actuation was calculated. Using this, a theoretical number for the times the car could jump using $16g$ of CO_2 was calculated to 172. The number of actuations observed during testing was 136.

4.2 Research question II

The theoretical height of the jump was calculated to $6.6cm$ with the car's mass of $1.99kg$. The test was conducted 10 times to get an accurate result. The height observed at each test could not be verified due to the value being so trivial. Due to the insignificant result of the test, a conclusion that the car would not be able to jump with the current weight was drawn.

Chapter 5

Discussion

This study found that, with the selected configuration, there is a considerable importance on the weight of the car and performance of the components, as it prevented it from jumping and that the number of times the car could jump was 136 per CO_2 cartridge.

As the car was not able to jump at the weight of 1.99 kg an enabling factor would be to reduce the total weight of the car. In section 3.3.2 a correlation between the mass of the car and the height of the jump was presented and motivated with calculations. It showed that the relationship between the two was inversely proportional, meaning that by reducing the weight of the car by half, the height of the jump would double. The necessary components to perform the jump, including the components used for the positioning of the car, was heavier than anticipated and proved too heavy for the car to jump. There is little room to reduce the total weight of all 3D printed components, as they are produced in a lightweight PLA material. For the pneumatic system the current configuration consists only of the necessary components and the performance of the car would improve if there were more lightweight alternatives. If a larger budget was available, an alternative solution would be to exchange the components to better performing ones.

The theoretical force of the cylinder achieved was three times the weight of the car. In order to increase the force one could look into a configuration with either a regulator with a higher pressure capacity or a pneumatic solenoid. When choosing a pneumatic solenoid it is of importance to optimize the output mass flow rate as this would increase the force that the pneumatic cylinder exerts. Using a pneumatic solenoid would also allow for control of the exhaust of return pressure during actuation of the cylinder which in turn would increase safety during testing. From observing the derived mathematical expressions, the pressure in the cylinder would be a key factor in maximising the achievable height of the car. A pressure regulator with a higher maximum pressure would consequentially increase the height achievable of the car.

The motorized lever system, used for positioning the car at a desired angle, was neither able to lift the car or hold it at an angled position. Even though there was no data sheet for the used stepper motor, the result could have been anticipated as most of the stepper motors of the same size have a considerably lower maximal torque than the 1.8 Nm calculated in section 3.1.2. The subsystem could be improved by adding additional motors but as that would increase the weight, increasing the gear ratio is a better alternative. The point of rotation of the lever arm could also be lowered closer to the ground enabling the lever arm to be shorter which would increase the force exerted on the ground.

One of the limitations of this paper was the resources available with regards to the pneumatic system. It would be advised to look into potential external suppliers in advance.

With a theoretical number of 172 actuations one can conclude that the observed 135 actuations during testing are satisfactory. Due to lack of resources the test could only be done once, however, it is advised to repeat the tests for a more accurate result. Another factor of uncertainty should be taken into consideration due to the inconsistent values observed from the gauge of the pressure regulators during testing after having been set to a certain pressure.

An improvement to the second design iteration would be to implement a modified halo with a geometrical shape of an arc placed above the second level of the car. The halo could potentially allow for the car to land on both sides, as it would flip the car when it lands on the top side. The halo would also serve as protection to the second level components.

Further research should be done on how to maximise the mass flow rate within the pneumatic system.

Chapter 6

Conclusion

The final design iteration and assembly of the car is a functioning configuration, however, it would not be applicable for the use of overcoming obstacles. From the analysis of theoretical and testing results it is apparent that the weight of the car and the performance of the components are crucial factors when it comes to maximising the height achievable of the car during a jump.

Bibliography

[1] Acumen. *Acumen.* Accessed: 2021-03-25. URL: http://www.acumen-language.org/.

[2] Greg Adamson. *honeycomb wheel good2 sw16.SLDPRT*. grabcad. Format: stl. Accessed: 2021-03-01. 2014. URL: https://grabcad.com/library/honeycomb-wheel-high-impact-1.

[3] Hazem I Ali et al. “A review of pneumatic actuators (modeling and control)”. In: *Australian Journal of Basic and Applied Sciences* 3.2 (2009), pp. 440–454. ISSN: 1991-8178.

[4] *Arduino Motor Shield Rev3 — Arduino Official Store*. Accessed: 2021-03-05. URL: https://store.arduino.cc/usa/arduino-motor-shield-rev3.

[5] *Arduino Uno Rev3*. Accessed: 2021-03-19. URL: https://store.arduino.cc/arduino-uno-rev3.

[6] V.V. Athani. *Stepper Motors : Fundamentals, Applications And Design*. New Age International (P) Ltd., Publishers, 1997. ISBN: 9788122410068.

[7] Antony Barber. *Pneumatic handbook*. eng. 8th ed.. New York: Elsevier Science, 1997. ISBN: 1-281-03542-4.

[8] Antony Barber. “SECTION 7 - Actuators”. In: *Pneumatic Handbook*. Ed. by Antony Barber. Eighth Edition. Oxford: Butterworth-Heinemann, 1997, pp. 475–510. ISBN: 978-1-85617-249-3. DOI: 10.1016/B978-185617249-3/50008-8.

[9] F. Decuir, K. Phelan, and B. C. Hollins. “Mechanical Strength of 3-D Printed Filaments”. In: *2016 32nd Southern Biomedical Engineering Conference (SBEC)*. 2016, pp. 47–48. DOI: 10.1109/SBEC.2016.101.

[10] Dejan. *L298N Motor Driver - Arduino Interface, How It Works, Codes, Schematics*. en-US. Accessed: 2021-04-10. Aug. 2017. URL: https://howtomechatronics.com/tutorials/arduino/arduino-dc-motor-control-tutorial-l298n-pwm-h-bridge/.

[11] D.C. Doedens. “Optimal CO_2 pressure for a pneumatic system”. uuid: c70cc714-959f-48e1-96b1-fcdedddce371. MA thesis. Delft university of technology, 2015. URL: http://resolver.tudelft.nl/uuid:c70cc714-959f-48e1-96b1-fcdedddce371.

[12] Rami El Golli et al. “Modelling of a pressure regulator”. In: *International Journal of Pressure Vessels and Piping* 84.4 (2007), p–234.

[13] Austin Hughes and Bill Drury. “Chapter One - Electric Motors – The Basics”. In: *Electric Motors and Drives (Fourth Edition)*. Ed. by Austin Hughes and Bill Drury. Fourth Edition. Boston: Newnes, 2013, pp. 1–38. ISBN: 978-0-08-098332-5. DOI: 10.1016/B978-0-08-098332-5.00001-2.

[14] Austin Hughes and Bill Drury. “Chapter Three - Conventional D.C. Motors”. In: *Electric Motors and Drives (Fourth Edition)*. Ed. by Austin Hughes and Bill Drury. Fourth Edition. Boston: Newnes, 2013, pp. 73–111. ISBN: 978-0-08-098332-5. DOI: 10.1016/B978-0-08-098332-5.00003-6.

[15] John M. Hughes. *Arduino Software*. Version 1.8.13. Accessed: 2021-04-19. URL: https://www.arduino.cc/.

[16] Dejan T Jevtić et al. “Modeling of gas parameters in the cylinder of the automatic gun during firing”. In: *Thermal Science* 00 (2020), pp. 152–152. DOI: 10.2298/TSCI200118152J.

[17] Hans Johansson. *Elektroteknik*. swe. Stockholm: Institutionen för maskinkonstruktion, Tekniska högsk., 2006.

[18] N Anju Latha, B Rama Murthy, and K Bharat Kumar. “Distance sensing with ultrasonic sensor and Arduino”. In: *International Journal of Advance Research, Ideas and Innovations in Technology* 2.5 (2016), pp. 1–5.

[19] Granta Design Limited. *CES EduPack software*. Version 2020. Cambridge,UK, 2009.

[20] EJ Moyer and U Chicago. “Basics on electric motors”. In: (2010).

[21] “Properties of Rotating Electrical Machines”. In: *Design of Rotating Electrical Machines*. John Wiley Sons, Ltd, 2013. Chap. 7, pp. 331–494. ISBN: 9781118701591. DOI: 10.1002/9781118701591.ch7.

[22] Siemens. *Solid edge*. Version 2020, universal edition. URL: https://solidedge.siemens.com/en/.

[23] Viktor Szente and Janos Vad. “Computational and experimental investigation on solenoid valve dynamics”. In: *2001 IEEE/ASME International Conference on Advanced Intelligent Mechatronics. Proceedings (Cat. No. 01TH8556)*. Vol. 1. IEEE. 2001, pp. 618–623. ISBN: 0780367367, 9780780367364.

[24] Binary Updates. *PWM Duty Cycle Pulse*. Accessed: 2021-03-27. URL: https://binaryupdates.com/pwm-in-lpc2148-arm7/pwm-duty-cycle-pulse/.

[25] W. Grey Walter. "An Imitation Of Life". In: *Scientific American* 182.5 (1950), pp. 42–45. ISSN: 00368733, 19467087.

[26] *What is Motor Driver: H-Bridge Topology and Direction control.* en. Accessed: 2021-05-20. URL: https://components101.com/articles/what-is-motor-driver-h-bridge-topology-and-direction-control.

[27] Feng-Kuang Wu, T.-J. Yeh, and Chun-Feng Huang. "Motor control and torque coordination of an electric vehicle actuated by two in-wheel motors". In: *Mechatronics* 23.1 (2013), pp. 46–60. ISSN: 0957-4158. DOI: 10.1016/j.mechatronics.2012.10.008.

Appendix A

Component specification table

Subsystem	Component	Quantity	Attribute	Value [-]	Total Weight [g]
Pneumatic system	Pneumatic Roundline Cylinder	1	Bore	25 [mm]	317
			Stroke	50 [mm]	
			Action	Single	
			Min. pressure	1 [bar]	
				10 [bar]	
	Solenoid	1	Normal state	N/C	166
			Port quantity	2	
	Pressure regulator	1	Min. pressure	0 [bar]	393
			Max. pressure	2 [bar]	
	Pneumatic tubing	-	Pressure range	0-10 [bar]	-
	CO2 cartridge	1	Pressure	60 [bar]	56.8
			Volume	0.15 [dmˆ3]	
Control of the car	L shape plastic gearbox with DC motor	2	Min. voltage	3 [V]	60
			Max. voltage	12 [V]	
			Rotation	CW/CCW	
	AAA battery	8	Voltage	1.5 [V]	92
			Capacity	1000 [mAh]	

Appendix B

Table of parameters

Parameter	Value	Unit
z_1	37	[-]
z_2	17	[-]
N_p	172	[-]
V_{CO_2}	0,00446	m^3
V_act	0.000026	m^3
p_1	60	bar
p_1	2	bar
R	8,3145	J/(mol*K)
M	44,01	g/mol
T	298,15	K
g	9,81	m/s^2
V_1	0,018183	m^3
V_2	0,000026	m^3
L_1	0,105	m
L_2	0,045	m
h_1	0,0781	m
h_2	0,05	m
b	25	mm
s	50	mm
m_c	1.99	kg

Appendix C

MATLAB code

```
%%%%%   Height as a function of mass    %%%%%

%Made by: Felix Ekman and Sofia Hansson
%Date: 2021-05-09
%KEX, MF133X, VT2021
%Task:
%Program for plotting the expected jump altitude with the mass
%of the car as a variable. The program plot the results where
%the y-axis is the estimated height of the jump and the x-axis
%is the mass of the car x10gram


close all, clear all, clc

m= 16; %g, mass CO2
M= 44.01; %g/mol, molar mass CO2
n=m/M; % number of moles
R=8.314; %J/mol*K
T=273.15+25; %K
V1= 0.01818315; %m^3
V2= 2.6*10^-5; %m^3
g=9.81; %m/s^2
L= n*R*T*(log(V1+V2)-log(V1));

for i=1:201 %for loop to get the height values for weights between
            %0-2000gram in steps of 10g

    mc= (i-1)/100; %mass of the car
    h(i)=L/(mc*g) %height of the jump
end

figure
plot(h)
title('Height')
xlabel('mass [10g]')
ylabel('height[m]')
```

```
%%%%%   Maximum torque at lever lift   %%%%%

%Made by: Felix Ekman and Sofia Hansson
%KEX, MF133X, VT2021
%Date: 2021-05-09
%Task:
%Program for calculating the maximum torque the stepper motor
%experiences as the levers are lowered between 0-90 degrees from
%the ground.
%The program then plots the results with the degrees on the y-axis
%and the torque on the x-axis

close all, clear all, clc

%%
m=1.99; %kg
g=9.81; %m/s^2
L=0.101; %m
L1=0.105; %m
L2=0.045; %m
h1= 0.0781; %m
h2=0.05; %m


for i = 1:91 %for loop that solves the equation for every degree
             %between 0-90 from the ground to it's vertical position

    a = (i-1)*(pi/180);
    x1(i)= L1 * cos(a) - h1*sin(a);
    O = acos((h2*cos(a)/(L)) + (L2*sin(a)/L));% O is theta
    x2(i)= (L2 - (h2 * tan(a)) ) * cos(a) + L*sin(O);
    T(i)= m*g.*x1(i)*L/(sin(O).*x2(i));

end

figure %Plot of the results
plot(T)
title('Torque')
xlabel('Degrees') %x-axis
ylabel('Torque [Nm]') %y-axis


fprintf("Maximum torque: %f Nm",max(T)) %prints the maximum torque in Nm



%%
```

Appendix D

Arduino code

```
//        Jumping robot
//        Made by: Felix Ekman & Sofia Hansson
//        KEX, MF133X, VT2021
//        Date: 2021-05-09
//        Task:
//        The program controls a jumping car. The program is written with a switch case control structure where the
//        different action of the car are separated into different cases. These cases are:
//        Case 0: DRIVE FORWARD
//        Case 1: DRIVE BACKWARD
//        Case 2: TURN RIGHT
//        Case 3: TURN LEFT
//        Case 4: LOWER THE LEVERS
//        Case 5: OPEN SOLENOID VALVE
//        The program is currently not using case 1-3 but they are included to simplify future work in controlling the vehicle.



// Declaring global varibles and including libraries:

#include <Stepper.h>      // Inludes the stepper library

int stepsPerRevolution = 200;
int solenoidPin = 2;      // This is the output pin on the Arduino used for the solenoid
int state =0;    // The state integer is initially set to 0 (DRIVE FORWARD)
int cm = 0;      // The integer with the value from the distance sensor

```

```
long readUltrasonicDistance(int triggerPin, int echoPin)  // Function that reads the distance from the ultrasonic distance sensor
{
  pinMode(triggerPin, OUTPUT);  // Initiate the trigger pin
  digitalWrite(triggerPin, LOW);  // Clear the trigger pin
  delayMicroseconds(2);

  digitalWrite(triggerPin, HIGH);        // Sets the trigger pin to HIGH state for 10 microseconds
  delayMicroseconds(10);
  digitalWrite(triggerPin, LOW);        // Clear the trigger pin
  pinMode(echoPin, INPUT);      // Reads the echo pin
  return pulseIn(echoPin, HIGH);  // Returns the sound wave travel time in microseconds
}


Stepper myStepper(stepsPerRevolution, 4, 5, 6, 10);      // Initialize the stepper library on pins 4,5,6,10

void setup() {          // The set up code

  myStepper.setSpeed(20);        // Set the stepper speed to 20, maximal 200

  Serial.begin(9600);          // Initialize the serial port:

      // Setup Channel A:
  pinMode(12, OUTPUT);        // Initiates Motor Channel A pin
  pinMode(9, OUTPUT);      // Initiates Brake Channel A pin

            // Setup Channel B
  pinMode(13, OUTPUT);      // Initiates Motor Channel B pin
  pinMode(8, OUTPUT);        // Initiates Brake Channel B pin

  pinMode(solenoidPin, OUTPUT);     // Sets the pin as an output used by the solenoid

}

void loop() {        // The repeating loop of the main code


    cm = 0.01356 * readUltrasonicDistance(7, 7);  // Measure the ping time in cm
    Serial.print(cm);             // Print the distance in cm in the serial monitor
    Serial.println("cm");
```

```
    delay(100); // Wait for 100 millisecond(s)


  switch(state) {
    case 0:     // CASE 0: DRIVE FORWARD, Drives the 2 dc motors forward
     // Channel A is the left motor and channel B is right

      // Motor A (left)
    digitalWrite(12, HIGH);  // Establishes forward direction of Channel A
    digitalWrite(9, LOW);   // Disengage the Brake for Channel A
    analogWrite(3, 100);     // Spins the motor on Channel A at 100/255 speed

    //Motor B (right)
    digitalWrite(13, HIGH);   //Establishes backward direction of Channel B
    digitalWrite(8, LOW);   //Disengage the Brake for Channel B
    analogWrite(11, 100);   //Spins the motor on Channel B at 100/255 speed

    delay(500);     //Drives for 500 ms

    digitalWrite(8, HIGH);   //Engage the Brake for Channel A
    digitalWrite(9, HIGH);  //Engage the Brake for Channel B

    if (cm > 50) {   // If statement controlling the next action. If the distance to the object
    // is less than 50 cm the program moves on to case 4 (LOWER THE LEVERS)
     // otherwise it continues to case 0.
      state=0;
    }
    else {
      state=4;
    }

    break;

    case 1:      // CASE 1: DRIVE BACKWARDS, Drives the 2 dc motors forward
     // Channel A is the left motor and channel B is right

    // Motor A (left)
    digitalWrite(12, LOW);   // Establishes backwards direction of Channel A
    digitalWrite(9, LOW);    // Disengage the Brake for Channel A
```

```
    analogWrite(3, 100);      // Spins the motor on Channel A at 100/255 speed

    // Motor B (right)
    digitalWrite(13, LOW);    // Establishes backward direction of Channel B
    digitalWrite(8, LOW);     // Disengage the Brake for Channel B
    analogWrite(11, 100);    // Spins the motor on Channel B at 100/255 speed

    delay(2000);          // Drives for 2000 ms

    digitalWrite(8, HIGH);   // Engage the Brake for Channel A
    digitalWrite(9, HIGH);    // Engage the Brake for Channel B
    state = 2;    // Continue to case 2
    break;

    case 2:      // CASE 2: TURN RIGHT, Drives the left motor forward and the right backwards

      // Motor A (left)
    digitalWrite(12, HIGH);    // Establishes forward direction of Channel A
    digitalWrite(9, LOW);      // Disengage the Brake for Channel A
    analogWrite(3, 200);      // Spins the motor on Channel A at 200/255 speed

     // Motor B (right)
    digitalWrite(13, LOW);    // Establishes backward direction of Channel B
    digitalWrite(8, LOW);     // Disengage the Brake for Channel B
    analogWrite(11, 200);     // Spins the motor on Channel B at 200/255 speed

    delay(2500);     // Drives for 2500 ms

    digitalWrite(8, HIGH);        // Engage the Brake for Channel A
    digitalWrite(9, HIGH);       // Engage the Brake for Channel B
    state = 3;                 // Continue to state 3
    break;

    case 3:      // CASE 3: TURN LEFT, Drives the right motor forward and the left backwards


     // Motor A (left)
```

```
    digitalWrite(12, LOW);       // Establishes backward direction of Channel A
    digitalWrite(9, LOW);        // Disengage the Brake for Channel A
    analogWrite(3, 200);          // Spins the motor on Channel A at 200/255 speed

     // Motor B (right)
    digitalWrite(13, HIGH);      // Establishes forward direction of Channel B
    digitalWrite(8, LOW);        // Disengage the Brake for Channel B
    analogWrite(11, 200);       // Spins the motor on Channel B at 200/255 speed

    delay(2500);     // Drives for 2500 ms

    digitalWrite(8, HIGH);      // Engage the Brake for Channel A
    digitalWrite(9, HIGH);       // Engage the Brake for Channel B
    state = 4;      // Continue to state 4
    break;

    case 4: // CASE 4: LOWER THE LEVERS, lowers the levers with the stepper motor to
                                                          // raise the car

    myStepper.step(60); // The stepper motor rotates 60 steps (1.8 degrees per step, 200 steps per revolution
    delay(1000);    // Waits for 1000 ms
    state=5;     // Continue to state 5
    break;

    case 5:  // CASE 5: OPEN SOLENOID VALVE, open the naturally off valve controlling the flow of gas
    digitalWrite(solenoidPin, HIGH);     // Switch Solenoid ON
    delay(200);     // Wait 100 ms
    digitalWrite(solenoidPin, LOW);      // Switch Solenoid OFF
    exit(0);  // "Exits" the loop and the program has to be manually restarted
  }

}
```

```
//        Solenoid Valve
```

```
//        Made by: Felix Ekman & Sofia Hansson
//        KEX, MF133X, VT2021
//        Date: 2021-05-09
//        Task:
//        The program actuates the solenoid valve controlling the gas flow of the jumping car. The program
//        actuates the solenoid valve once and then has to be manually restarted.

// Declaring global varibles:
int solenoidPin = 2; // This is the output pin on the Arduino used for the solenoid

void setup() {    // The set up code
  pinMode(solenoidPin, OUTPUT);   // Sets the pin as an output used by the solenoid
}

void loop() {     // The repeating loop of the main code

  digitalWrite(solenoidPin, HIGH);  // Switch Solenoid ON
  delay(200);      // Wait 100 ms
  digitalWrite(solenoidPin, LOW);     // Switch Solenoid OFF
  exit(0);    // "Exits" the loop and the program has to be manually restarted
}
```

Appendix E

Acumen simulation

```
//       Jump simulation
//       Done by: Felix Ekman & Sofia Hansson
//       Date: 2021-03-25
//       Course: KEX, MF133X, VT2021
//       Task: The program is a simulation of a jumping robot which is part of a bachelor th
//       at KTH. The program is used to visualize the projectile motion after the velocity i
//       y- and z-direction has been calculated. x,y,z and units of length in meters [m].

model Main(simulator) =
initially                                  //*   Initial condition   *
  y    = 0,                                //Postion horizontally
  y'   = 4.43,                             //Velocity horizontally
  y'' = 1,                                 //Acceleration horizontally

  z = 0,                                   //Position vertically
  z'= 7.67,                                //Velocity vertically
  z''=0,                                   //Acceleration vertically

  g=-9.82,                                 //Gravitational acceleration
  _3D=()
always                                     //*   Simulated movement   *
  simulator.endTime+ = 1.56,               //Simulation time
  y''=0,
  z''=g,
  if z<-0.1 then z'+=7.67 noelse,          //Repeats the jump


  _3D = (Box                               //3D object -  Chassi
          center=(0,y,z)                   //Centerpoint (x,y,z)
          size=(0.2,0.3,0.05)              //Size (x,y,z)
          color=red                        //Color
          rotation=(0,0,0),                //Orientation around axis

         Cylinder                          //3D object, Right back wheel
          center=(0.14,-0.125+y,z)         //Centerpoint (x,y,z)
          size=(0.08,0.05)                 //Size (x,y,z)
```

```
 color=black                    //Color
 rotation=(0,0,pi/2),           //Orientation around axis

Cylinder                        //3D object, left back wheel
 center=(-0.14,-0.125+y,z)      //Centerpoint (x,y,z)
 size=(0.08,0.05)               //Size (x,y,z)
 color=black                    //Colow
 rotation=(0,0,pi/2),           //Orientation around the axis

Cylinder                        //3D object, left front wheel
 center=(0.14,0.125+y,z)        //Centerpoint (x,y,z)
 size=(0.08,0.05)               //Size (x,y,z)
 color=black                    //Color
 rotation=(0,0,pi/2),           //Orientation around the axis

Cylinder                        //3D object, right front wheel
 center=(-0.14,0.125+y,z)       //Centerpoint (x,y,z)
 size=(0.08,0.05)               //Size (x,y,z)
 color=black                    //Color
 rotation=(0,0,pi/2),           //Orientation around the axis

Box                             //3D object, ground
 center=(0,4,-0.1)              //Centerpoint (x,y,z)
 size=(1,10,0.1)                //Size (x,y,z)
 color=white                    //Color
 rotation=(0,0,0),              //Orientation around axis

Box                             //3D object, wall
 center=(0,3.57,1.35)           //Centerpoint (x,y,z)
 size=(1,0.3,2.8)               //Size (x,y,z)
 color=red                      //Color
 rotation=(0,0,0))              //Orientation around axis
```

TRITA TRITA-ITM-EX 2021:43