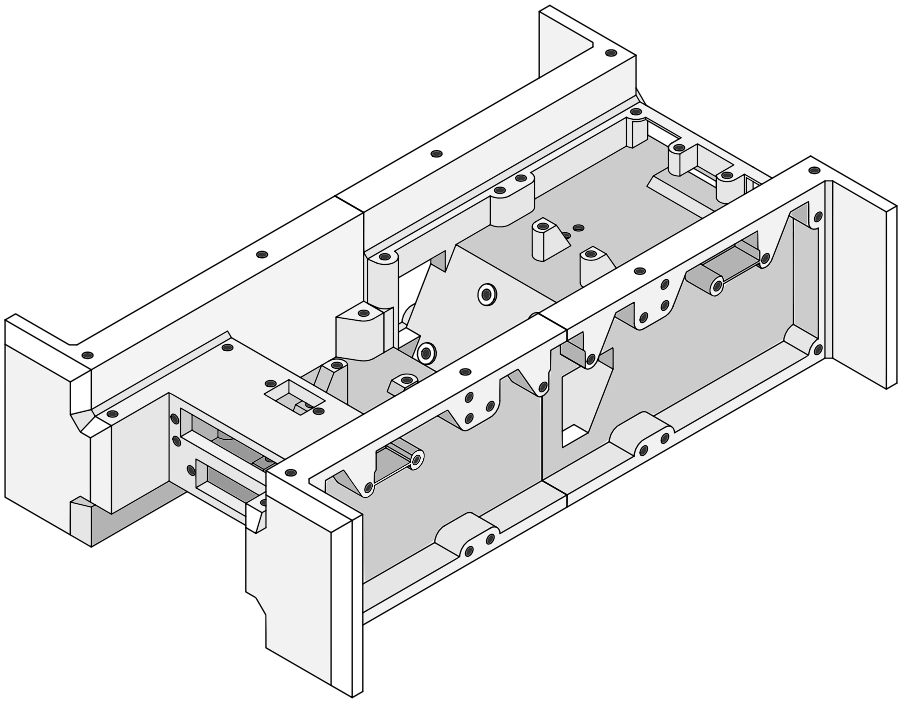
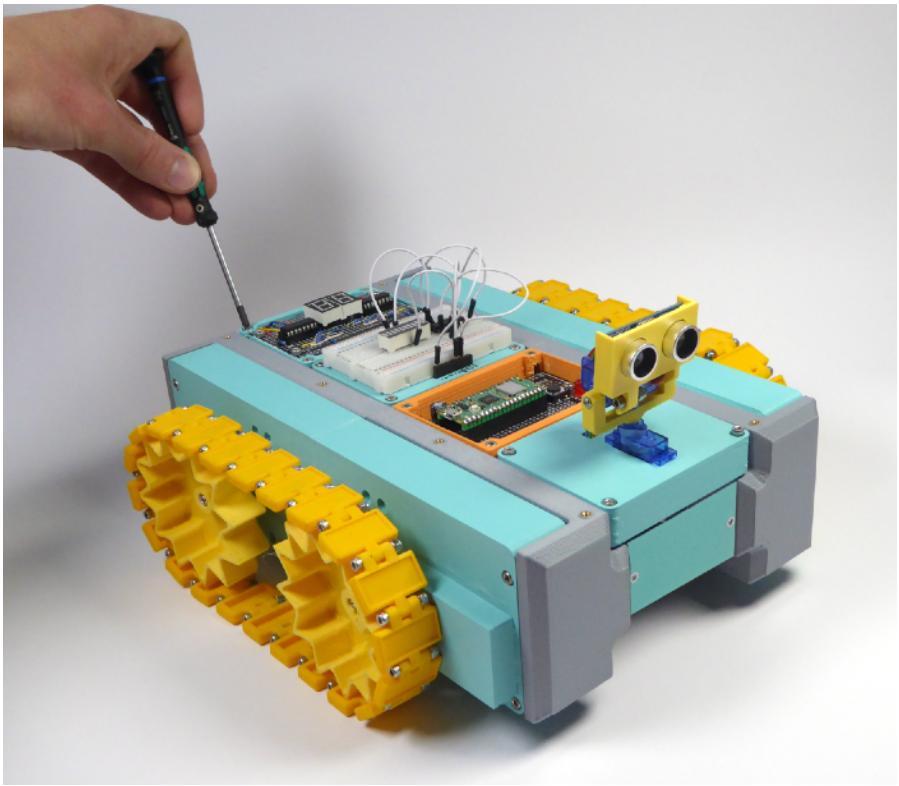
# unibot

by Emil Altmann





# TABLE OF CONTENTS:

# MOTIVATION

For the past seven years, I have worked as a supervisor in a Lego and robotics workshop at my high school. During this time, I have gained valuable insight into the growing demand for educational robotics. In addition, my study of computer science allowed me to maintain and upgrade Arduino-based robotic vehicles. From these experiences, I have curated a list of goals that includes both successful concepts and lessons learned from less effective designs. Using this list as a foundation, I have begun to develop my initial design concepts for a modular driving robot. A primary goal of mine is to minimize the complexity of repair and maintenance, recognizing that wear and tear is inevitable - especially in educational settings where diverse and inexperienced users interact with these robots.

My robot design consists of a central main frame that acts as the core chassis and houses the battery connections and wiring. In the standard configuration, there are eight module bays. Each bay has a nine-pin DB9 connector for power and five data connections. In addition, the main controller bay has a 25-pin DB25 connector for interfacing with other modules.
The normal bays - three at the top and two at the bottom - position the DB9 connector at the rear and provide four screw points for secure mounting. These modules are 90mm wide, 70mm long and 20mm deep. These specific dimensions have been chosen to accommodate common components such as breadboards and PCBs used in prototyping, which typically measure around 85mm x 55mm.
The controller bay is 40mm deep to accommodate additional circuitry such as power regulators. In addition, the side motor bays are significantly larger at 80mm wide, 273mm long and 30mm deep. They also feature three parallel connectors. These larger dimensions and multiple connectors facilitate the installation of one to three normal size modules.

# List of requirements for the diploma robot:

**WHAT THE ROBOT SHOULD DO:**
- drive
- bluetooth communication
- line detection
- find its orientation and position
- range detection
- be interactive
- not break and be easy to repair

**WHAT THE ROBOT SHOULD HAVE:**
- differential steering
- three infrared line sensors
- Intertial Measurement Unit (IMU)
- pan (and tilt) gimbal with TOF or ultrasonic sensor
- LED lights/displays/bars
- power tool battery

**WHAT THE ROBOT MIGHT HAVE:**
- stereo vision cameras
- light detection and ranging (LiDAR) sensor
- Radio frequency identification (RFID) reader
- Camera
- mecanum wheels
- tracked drive
- color sensors for line detection
- global navigation satellite system (GNSS), e.g. GPS
- robot arm (2+ axis manipulator)
- bumper sensors

**WHAT ARE THE DESIGN RULES OF THE ROBOT:**

- be modular / hot-swappable components
- only use “off the shelf components“
- 3D print easily and quickly
- no irreversible connections / fasteners / assemblies
- minimal component diversity
- only rugged connectors
- PCBs should be doable in Perma-Proto board layouts

**LIMITS AND NO GO'S:**

- should not be larger than 200 x 300 x 100 mm
- should be quiet
- problems should be solvable without disassembly
- no rats nest of wires
- no loose or easy to loosen components.
-

**CONCLUSIONS FOR THE TOP AND BOTTOM BAYS:**

- fit standard 51x81mm PCBs (min 60mm x 90mm)
- 20mm height
- open at front and top

**CONCLUSIONS FOR THE MOTOR BAY:**

- height of the complete robot (min 80mm)
- depth 30mm and length 250mm

# REV0

Revision 0 was never intended to be an object. Rather, it consisted of the many shortlived ideas for standards, connectors, interfaces and features. Many of these were abandoned due to technical limitations. Nevertheless, I would like to describe some of these initial ideas and show their potential and shortcomings.

First, I would have liked to use a smaller connector with either all ports or a fixed preselection. This would have had the advantage of eliminating the interposer. Also, the modules would be more deterministic. So it would be easy to replace two modules. Unfortunately, even with a relatively small connector, such as a DuPont-style connector, there would not be enough room for all the connections. Also, after talking to some electronics engineers about my plans, they recommended that I use a D-Subminiature (D-Sub or DB) style connector. These connectors were commonly used in older PC electronics such as VGA cables, serial ports, or printer ports. Their big advantage is ruggedness. They are designed for many mating cycles and can handle 5A of current per pin connection.

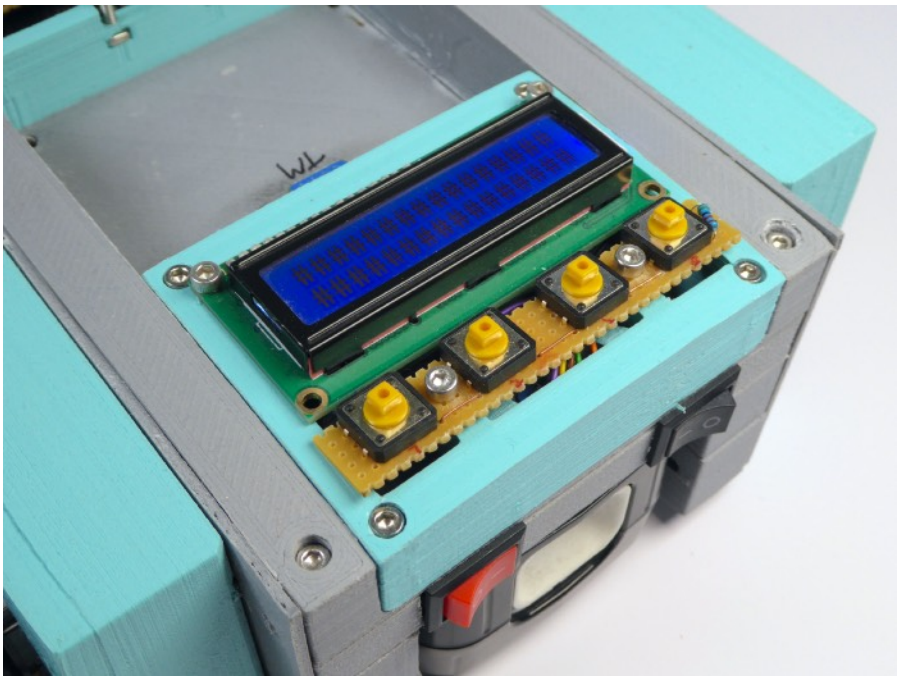
The second important decision was the size of the modules and the positioning of the connectors and screws. In the beginning, the normal bays were 90mm x 60mm x 20mm with four screws set 10mm from each corner. The two main constraints on module size were the size of the connector and the size of a Perma-Proto board. The connector required a minimum height of 18mm. The Perma-Proto board is a standard electronic prototyping board from adafruit that measures 51mm x 81mm. After some layout testing, I found that

the 10mm depth for the screws and the 60mm length conflicted with the connector and the prototyping board. So I optimized the usable space by pushing the screws further out and increasing the length to 70mm.

For the motor modules, I started with a simple scaled version of the normal module. The dimensions were 200mm x 80mm x 30mm. They resulted from the total space needed for the inner chassis. But I realized that only four screws wouldn't be enough to secure the module. So I increased their number to 8. This resulted in a screw spacing very close to the normal modules. So I redesigned the motor modules to be as compatible as possible with the normal modules. But I planned only one central connector. In the first meeting with my professor Oliver Vogt, he convinced me to use three connectors on each side. This resulted in even better inter-compatibility between the motor modules and the normal modules. The only limitation is that the three connectors are parallel. So any signal to and from one will be seen at the others.

In my job of upgrading the robots for the Institute of Computer Science, I had already researched the best power source for this type of robot. Usually two types of batteries are used. On the one hand, there are commercial, specially designed ones like those used by LEGO. On the other hand, many hobbyist robot kits use batteries designed for remote-controlled model cars. The former was out of the question because there is usually no way to get these batteries individually. There is also no documentation available. The main argument against the second type is safety. Many of them

have no protection. So when things go wrong, they really go wrong. And the chargers are flimsy at best.
So after some discussion and research, I decided to go with power tool batteries. Because most of them are really well designed, especially in terms of mechanical and electrical robustness. Also, modern power tools are a bit like robots because they use powerful motors and electronics, as well as microcontrollers. The final plus is the chargers: they charge the batteries very quickly and safely, and always balance the packs. All in all, the result is very adaptable, safe, robust and easy to use batteries.



**The back of the first revision as used in my school.**


TB
TM
TF

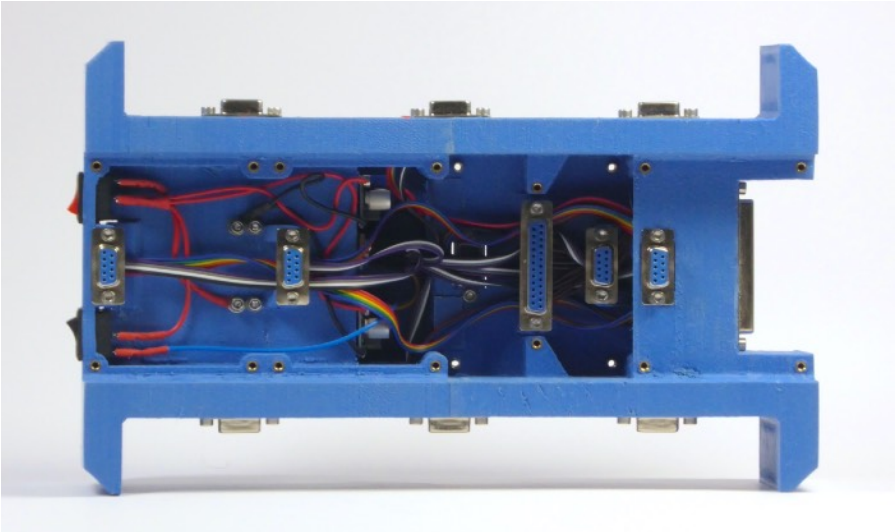

R

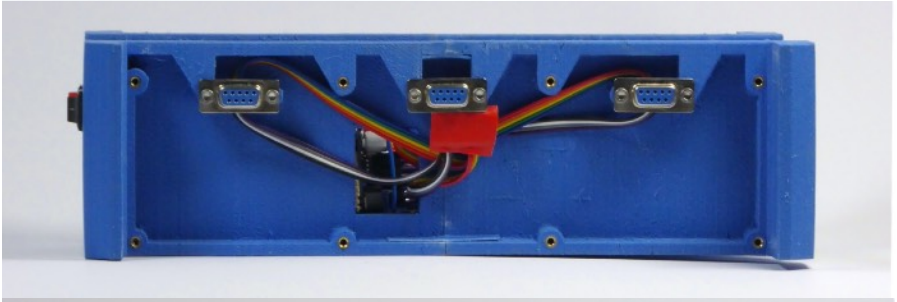

BM
BF

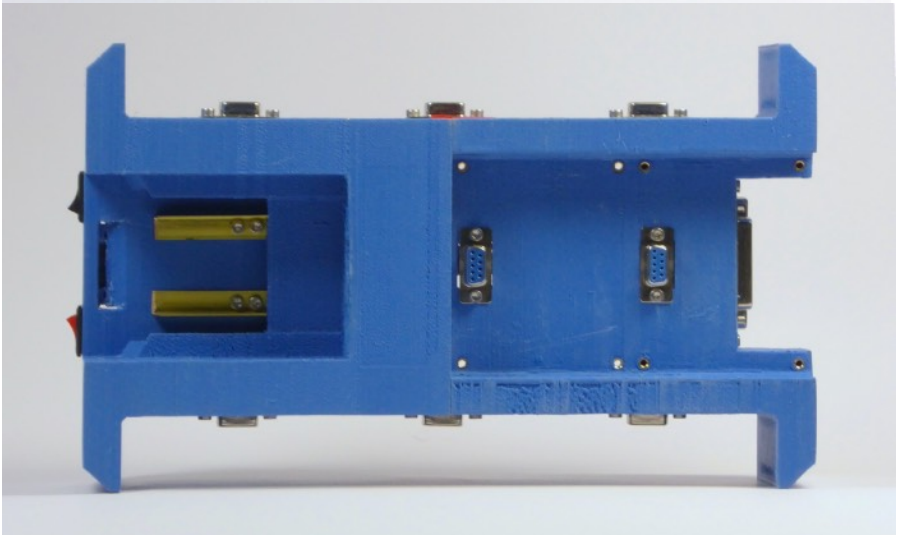
# REV1

The first revision was quickly designed and printed. The reason for this was because my robotics workshop at the Georg-Christoph-Lichtenberg-School started in the first week of the semester after the Easter holidays. My students are very interested in my work and we decided to test the rev1 together. After sketching the model, I cut it into many pieces and 3D printed it. After an intense week I had a mechanically and electronically working robot. Over the next few weeks we added some modules designed by the students and one student even wrote an android app for the remote control.

When I designed the parts, I didn't have 3D printing in mind. So many parts are not printable without support structures. Also, the cable management was a total mess. This was improved a bit by covering all the internals with panels. Luckily there were no bugs, but if there had been, it would have been a nightmare to fix. So I decided for the second revision that all electronics should be exposed for easy repair. I also planned to have the three connectors on each motor bay and the two power buttons on the back. I retrofitted them for rev1, with the red one being the main power switch and the black one just switching the battery voltage to the modules. So you can either turn off the whole robot or just the power to the motors. The last thing I had to correct was the distance between the modules and their bays. For rev2 I added 0.5mm clearance around each module.

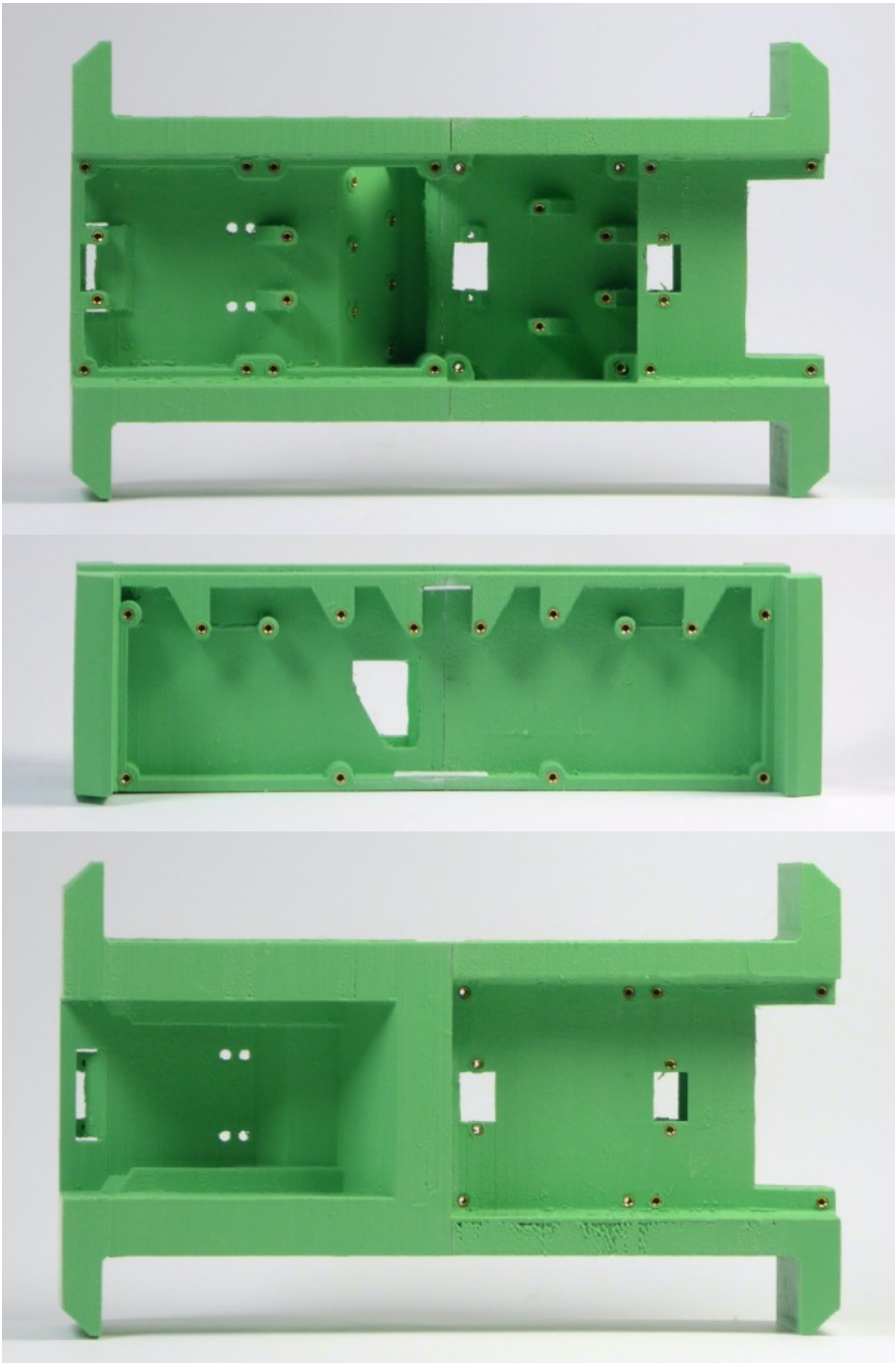






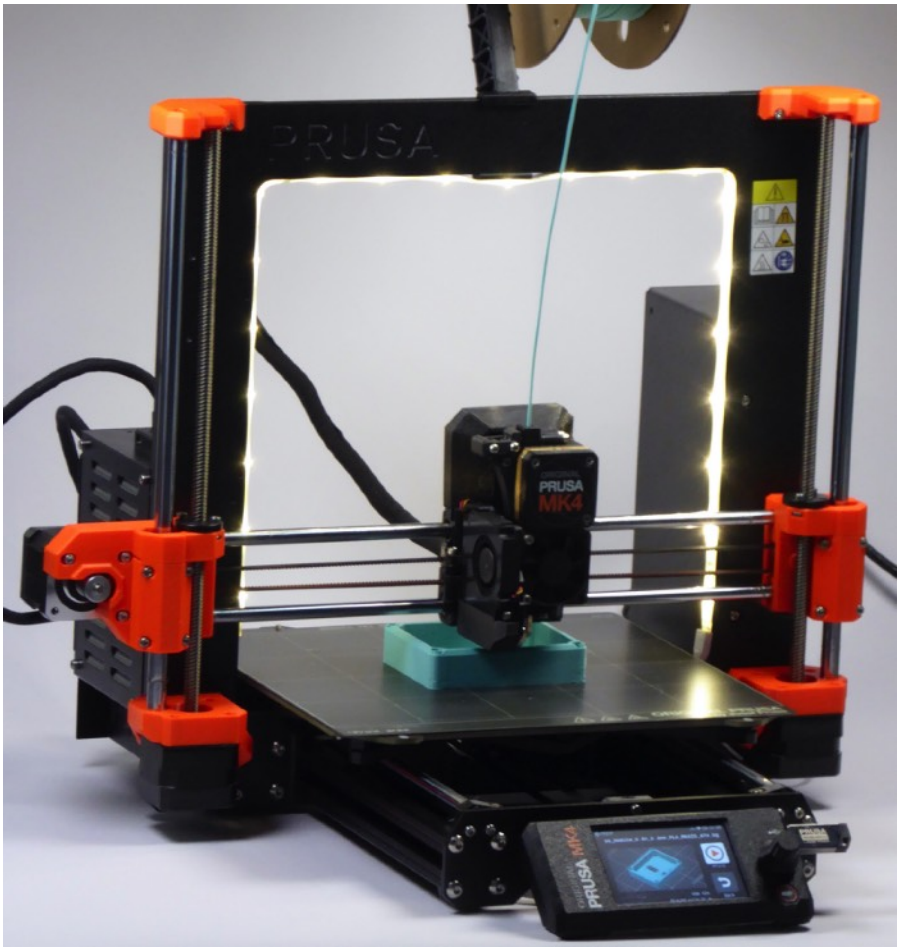
# REV2

After the first revision, I scrapped the entire 3D sketch. I had enough changes to start over. My main goal from this revision on was to get the best 3D printability. So with each step, I thought about reducing material usage, optimizing features for printing, and eliminating anything that wasn't absolutely necessary. Funnily enough, after printing, I realized that I had removed a little too much in the controller bay. Also, I wanted to print the body in two parts, each standing on its own end. For this reason, it took a lot of time to optimize all the overhangs for printing, and I even had to add support structures directly into my design. In the engine bays, there are two support columns for the outer connectors that would otherwise be printed into thin air. There were only three features left in this design with very wide bridges. A modern 3D printer can bridge a gap if there is a wall on two opposite sides of the structure. The slicer software then uses special parameters for the bridge between these two walls. These features include the back of the battery connector, the back of the interposer connector, and the middle dividing wall are located.



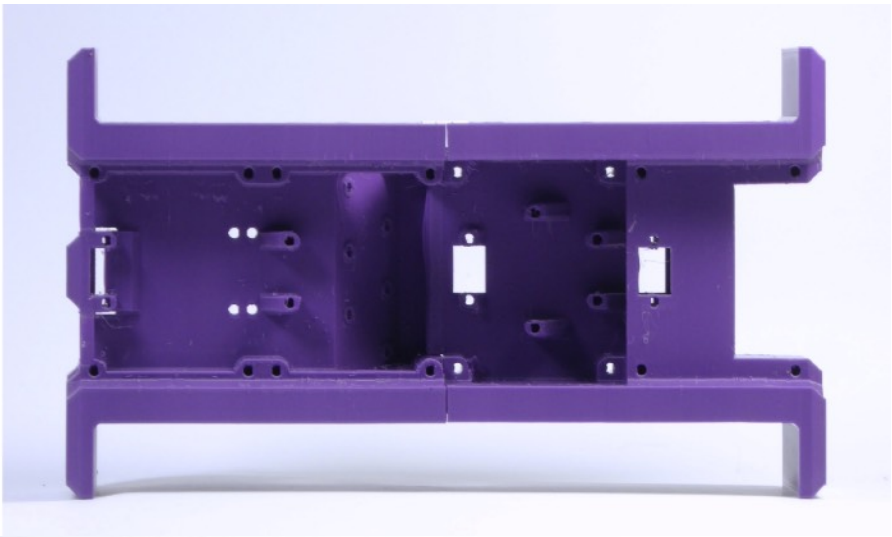
# REV3

The third revision was the next step in printability. I angled the back of the battery compartment. This made the overhang flat enough to be easily 3D printed. Another trick I tried in order to increase the 3D printability, was the inclusion of false bottoms. In the bridging part of the interposer connection, where the 3D printer has to print 60mm in thin air, I simply added two 0.2mm surfaces with a slight offset. This gave the 3D printer a much better chance to bridge the gap and create a cleaner surface. I also corrected the few oversights in the second revision. These were the wrongly placed spacers for the controller bay connector, the missing screw columns for the controller module, and the missing clearance in the motor bays. Funnily enough, I did not realize until after the print run that I had an odd corner in the back of the motor bays since the second revision. Another change was the inclusion of threaded inserts for all screw positions. Previously, I had left a few positions where nuts were used for fastening. But to improve the serviceability, I replaced them all with inserts. This eliminated the frustration of losing nuts inside the chassis. The last change in this revision was the reinforcement of the upper front connector, because in the second revision the flexibility of the plastic made it unnecessarily difficult to connect the modules properly.



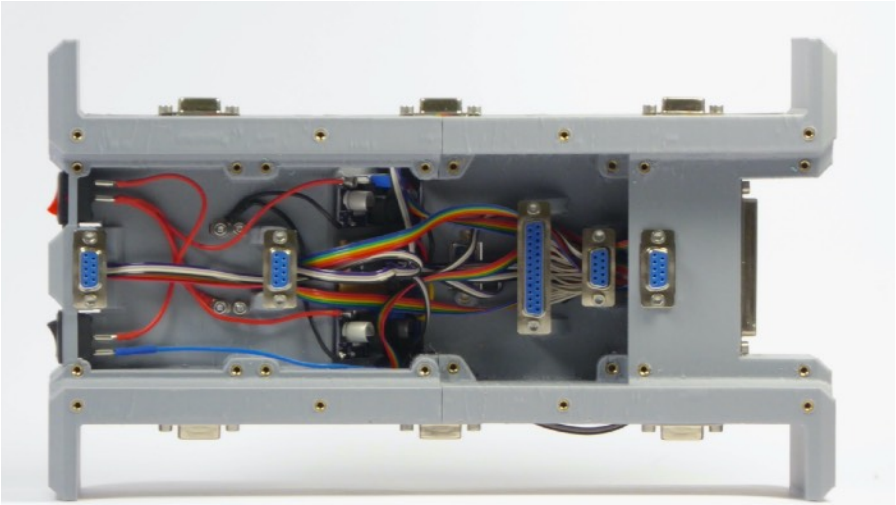

**Printing with my PRUSA MK4 3D Printer.**

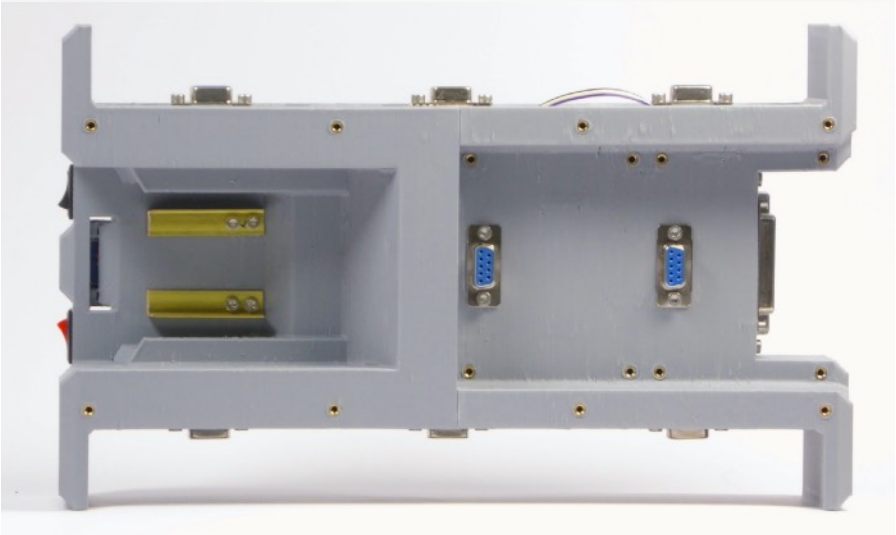
Until the third revision, I printed all the chassis and modules on my workhorse Ultimaker 2 (released in 2015) using a 0.8mm nozzle, 0.16mm layer height, and Polymaker's Polyterra PLA filament. At this point, the printer had about 3000 hours of printing and had extruded over 4km of filament. Unfortunately, its performance degraded as the size of the 3D prints increased. But it was able to print the two halves of the chassis in about 14 hours each. To go further and get faster and better results, I upgraded halfway through to an original Prusa i3 mk4. This 3D printer - released in 2023 - was able to print the same files in less than half the time with exceptional quality and reliability.



# REV4

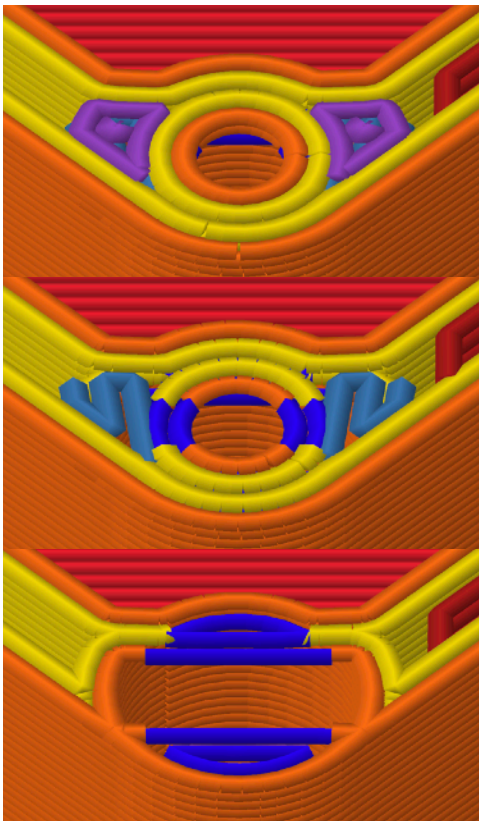
The fourth revision was the first major 3D print on my new Prusa i3 mk4 3D printer. I used the box standard configuration with a 0.8mm nozzle and 0.4mm build height. This cut the print time per part by a factor of three, from about 15 hours to 5 hours, with no loss in quality. I didn't tweak the slicer profile, but other than some stringing, I was happy with the results.
The biggest change I made in this revision was the motor bays. After discussion with my professor, I changed the design of the motor modules so that they are exactly three times as wide as a normal one. This makes it possible to connect three normal modules on each side, however, due to electrical restrictions, these modules are still connected in parallel. For this reason, their function is slightly reduced compared to the top or bottom position. Because of this extension I had to push out the corners of the frame. But on the positive side, this increased their use as bumpers, especially in the back, where the two switches were completely exposed before.
I also tweaked a lot of small details for easier 3D printing and increased structural integrity.





# REV5

The fifth and final iteration of the chassis included some minor improvements for 3D printability. I also added eight treaded inserts to the top and bottom of the robot for future expansion. I also added two more treaded inserts in each motor bay for adding adapters between the 80 millimeter motor module height and the 70 millimeter normal module height.



**Slight differences make printing easier:**

Sometimes a little more complex geometry helps the slicer and therefore the printing process. In this case, I added two steps below the hole inset. The blue lines show where the slicer detects and optimizes for bridging.

# MODULES

As the chassis continued to be modified, many of the modules had only one revision. In the beginning, the modules were just boxes with the right dimensions and connectors and screws in the right places. But as time went on, I implemented some design features that were carried over to all the final designs.
One of the first decisions was to always have one side fully exposed for ease of repair and understanding. I also implemented a way to catch the mounting screws. In the beginning I left an 8mm hole for a locknut, but unfortunately this resulted in a very thin corner wall. Fortunately, there are extra slim nut drivers available, which allowed me to reduce the hole size to 6mm. With this redesign, I also added a radius at each corner and a chamfer at the bottom. This made it easier to install the modules. Also, with my new 3D printer, I was finally able to redesign and test the printability of overhangs and stepped holes. In this case, I tried to modify the features so that the slicer software could find a better way to 3D print them. I think this design perfectly illustrates design for manufacturing. I made a lot of decisions in the direction of 3D printing. Most of them aren't too obvious, like the wall thickness and avoiding overhangs. But even the rounded corners and chamfered bottoms are designed for 3D printing. 3D printers are 2.5D machines. Therefore, they make visual steps in height, but can make arbitrary shapes in any plane. For this reason, a chamfer will always look cleaner if it is oriented horizontally. Also, smooth contours around the perimeter of the object allow for slower acceleration. This results in a cleaner and faster print.

In conjunction with the release of the other files, I also added some template modules and technical drawings.
These templates consist of a finished version of each module type - normal, motor, single motor, controller - one with an open top and one with an open back. Each is designed for 3D printing with a 0.8mm nozzle or 0.4mm nozzle positioned with the closed side facing the print bed.

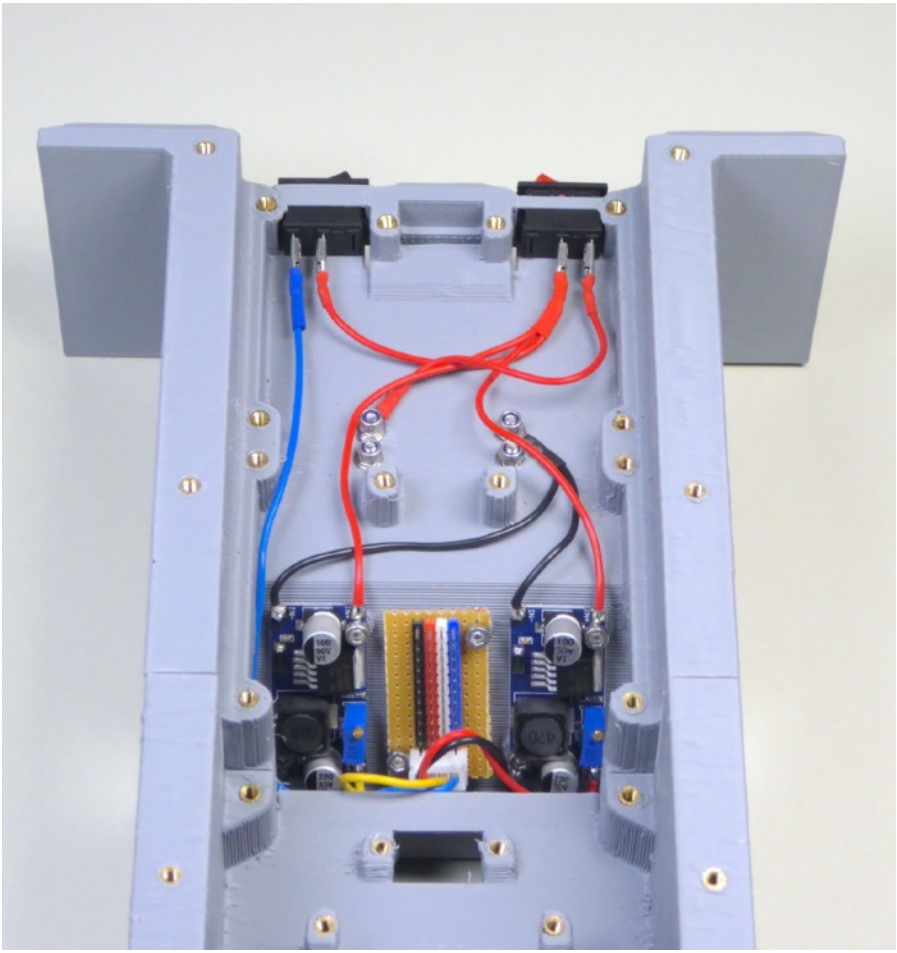
**A pitch and yaw controllable ultrasonic distance sensor module.**

# ASSEMBLY

The main assembly of the chassis consists of two steps. First the mechanical assembly and second the electrical assembly.
I always start by fitting all 4mm holes with a m3 x 6mm x 4mm melting insert. These can be easily installed by heating them with a soldering iron while pressing them lightly into the hole. After that, the two chassis parts can be glued together. Because of the print orientation and the risk of layer separation, four plastic taps can be glued in place.

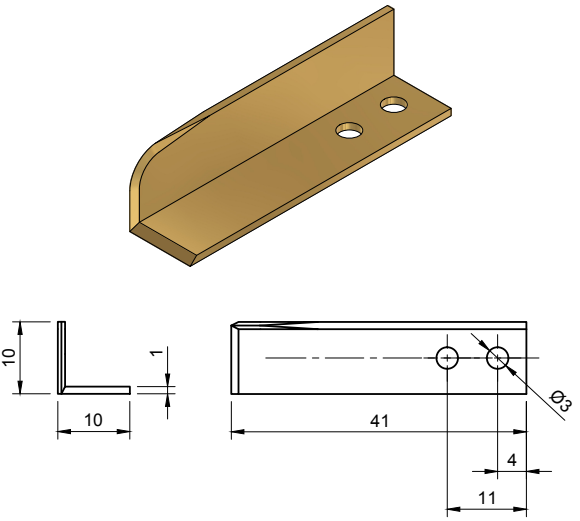
**Some brass m3 threaded inserts.**

**The complete chassis wiring harness**
In the middle is the connector to the interposer with all the connectors to the normal bays.
Left and right are the two motor bay harnesses.
Bottom left is the whirring between the controller bay and the interposer.
Bottom right is the power connector to the controller bay.



**The robot's power infrastructure.**

The second part of the assembly is the creation and installation of the wiring harness. It's designed to be as modular as possible. For example, the side connectors have a common connector for ease of installation and repair.
I also had to design the battery contact for the Makita BL1415N power tool batteries. This contact can be made from a simple 10mm x 10mm brass angle. The two mounting screws also serve as the power terminals.
The wire from the positive terminal is switched on by the red switch. After that, the two buck regulators - one 3.3V and one 5V - are powered, as well as the black switch. This switch is like a safe stop because it disconnects the battery voltage from the modules. This allows you to turn off the robot's motors without turning off the microprocessor.




**Front and top view of the right battery contact.**

# WHY 3D PRINTING?

I asked myself this question early on, and the reasons are many. The most obvious reason is that 3D modeling and design should be done with the primary manufacturing processes in mind. To fulfill this design-for-manufacturing strategy, I choose 3D printing for manufacturing, not just prototyping. Over five years of continuous product design, I have accumulated a lot of knowledge about how to design fast and reliable 3D printable objects. My focus has always been on practicality. So in prototyping, the process of bringing objects to reality should always be as fast or faster than the actual design cycles. So a requirement for all my models was that the 3D printer could produce the object in less than a day. If not, that production time would introduce unnecessary delays into the iteration process. I also never wanted to replicate another manufacturing process with 3D printing. So if I chose 3D printing as my process, the resulting objects should always be recognizable as having been produced by that process.
Another reason for 3D printing is accessibility. Of course, with the increasing capabilities of manufacturing services, especially in the realm of subtractive manufacturing, such as milling or turning, the availability of relatively cheap and effortless production has increased in recent years. But in the context of open source manufacturing, 3D printing is one of the few with cheap and technically mature machines, although there are many open source projects for other manufacturing processes.

In injection molding, for example, Precious Plastic has made great strides in domestic plastic recycling and injection molding. Unfortunately, injection molding's greatest strength is

also its greatest weakness: molds. Although resin 3D printing has added the ability to make reasonably inexpensive injection molds, for most complex or larger shapes, a professional metal mold is required. As a result, profitability is highly dependent on volume.
In the area of milling and routing, PrintNC is one of the more mature projects. Unlike injection molding, I have only had tangential contact with this subject. But from my point of view, the main problems are stiffness, noise and materials. Real milling machines are beasts in terms of rigidity. They have to be heavy and strong because of the cutting forces. So a lot of hobby and low-end milling machines are just routers. So their use is often limited to wood, plastic and non-ferrous metals. They also work in a 2.5d fashion. Hence their shortcomings compared to industrial 3 or more axis milling machines. Regarding noise, while modern 3D printers are quiet enough to work next to in an office, mills or routers are quite the opposite. This results in reduced applicability. The final shortcoming of milling is material sources. Since this manufacturing process is highly industrialized, so are the material suppliers. Often it is not possible to purchase the required materials in reasonable quantities.
In the context of my project, I chose 3D printing because of its availability, familiarity, and flexibility. Since I wanted my work to be in the public domain, it was imperative that it be reproducible with minimal barriers. In addition, many users already have access to 3D printers. So there is often little or no cost to using them. Finally, because I want users to be able to modify and adapt my robotic platform to their use case, 3D printing is one of the most flexible manufacturing technologies available.

# INTERVIEW:

As already mentioned, I worked as an assistant to Benjamin Herwig in the Intelligent Embedded Systems group before writing my diploma thesis. I maintained and improved the robot vehicles used there. I asked Benjamin the following five questions about the lectures:

1st - How are the lectures in Robotics and Embedded Systems structured?

Basically, we offer two lectures with self-propelled robots, but these can in no way be described as robotics. The lectures always consist of a theoretical and a practical part. In the practical part, most of the topics covered in the theoretical lecture are put into practice, where the students themselves enter into a creative process, accompanied by teachers and assistants.

2 - What are some examples of tasks that the students should solve with the robots?

The tasks to be worked on range from the absolute basics to the control of complex peripheral electronics. For example, in the beginning it is only a matter of making a light emitting diode flash, while later in a course camera and internal measurement data are also recorded, transmitted and evaluated on another computer system. Above all, students should get a feel for how the complex process chain of data processing and artificial intelligence can be viewed, from the real environmental data surrounding a computer system to the knowledge and actions derived from it.

3 - What are the technical requirements for the robots?

On the one hand, the robots must be technically robust, but at the same time they must be built using technology that allows students to understand the technology well and quickly, despite its complexity. For example, it is important not to use microcontroller platforms that are too complex.

4 - Are there also didactic requirements that can improve teaching?

Related to the last point, it is important to be able to explain the technology in a simple and efficient way, so that students can quickly get their first practical experience, which will increase their motivation to learn.

5 - What features and capabilities are missing in the robots used so far?

In particular, they lack robustness and ease of maintenance and repair. There is also a lack of modularization of the robot components used, which cannot all be used in each course offered.

# SCENARIOS AND USE CASES

Due to the technical nature of the robots. It's mandatory that the supervisor has knowledge of electronics and computer science.
The following is an approximation of the cost and labor for the robot chassis and modules. Unfortunately, in many educational contexts there is a very tight budget for materials and labor. So, as mentioned before, my main focus was to make the assembly and maintenance as easy as possible.
Therefore a lot of the production work is done by the 3D printer. For the chassis, this means about ten hours of printing 500 grams of PLA. This is split into two prints due to volume limitations and the fact that 3D printers in German schools can only be used under supervision.
In addition, each normal module takes about an hour to print.
The time for the electrical construction depends on the skill of the builder. But for the chassis, the time is between four and eight hours, and one hour per module.

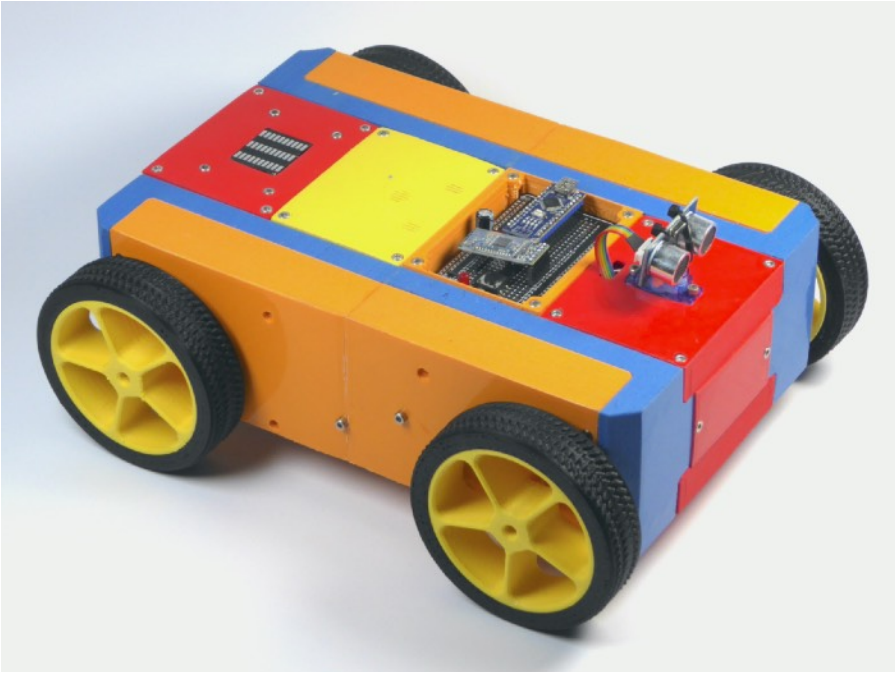
The material costs for the chassis are as follows:
- 20€ for 3D printing
- 10€ for m3 hardware
- 15€ for 12x DB9, 2x DB25 and 1x DB37 connectors
- 5€ for DC regulators
- 5€ for switches and other small electrical parts
- 5€ for 1.5 meter flat ribbon cable

For the modules, the cost is very dependent on the included electronics, but there are approximately:
- 10€ for 3D printing
- 5€ for m3 hardware
- 5€ for DB9 connector and cables
- + cost of motors, sensors and microcontroller

There are many different ways to configure, program and extend the robots. Therefore there are unlimited application scenarios. Below I will describe a selection of them:

# LOW LEVEL PROGRAMMING

The configuration consists of the following modules:
- Arduino nano controller module
- triple tcrt5000 sensor module
- single servo single SRF05 ultrasonic sensor module
- triple bar graph display module.
- double 7-segment LED display module
- dual dc motor module

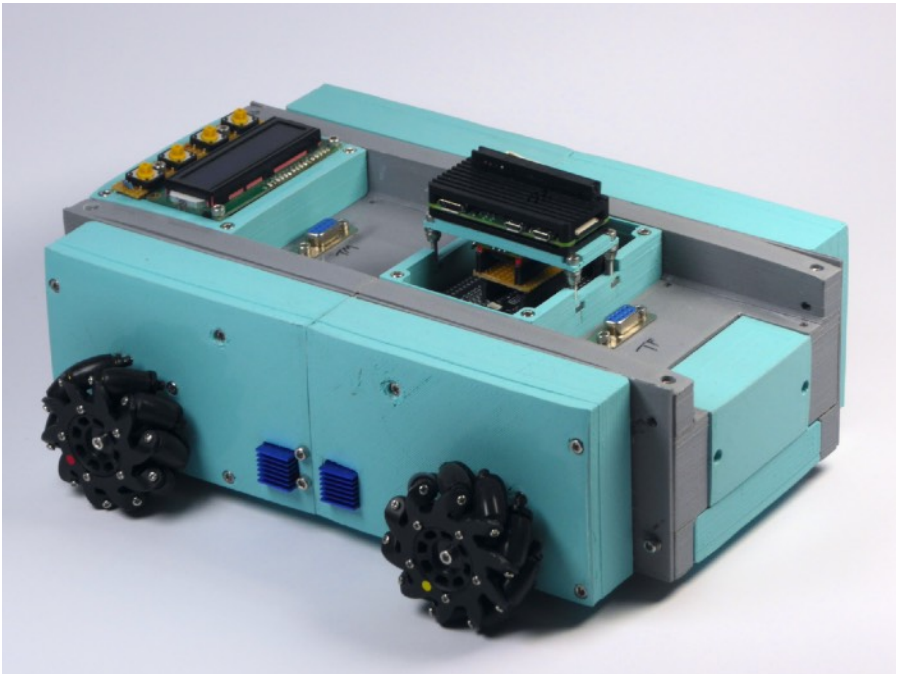


This robot could be used at the university for teaching computer science and mechatronics students. The Arduino nano microcontroller is very simple and low power. That's why it's very good for programming at the lowest level, because a student is fully able to understand the hardware used.
In addition, the two display modules both use the 74hc595 shift register. They combine a basic concept of serial data transmission and LED display technology.

The servo-controlled ultrasonic sensor and the triple line sensor are the two main sensor modules for autonomous driving. Many modern cars use ultrasonic distance sensors for obstacle detection. For example, in combination with the servo motor, the robot can scan its environment to detect walls. Since the ultrasonic sensor returns a digital signal, the length of the signal determines the measured distance. Therefore, the programmer must measure time, which is a fundamental concept in hardware programming.
In addition, the triple line sensor is the standard for line-following robots. These use three black and white sensors side by side. In normal operation, the robot tries to keep the black line under the center sensor and corrects when one of the side sensors detects it. Also, the sensors return an analog value, so the programmer must read and interpret them correctly.

# RASPBERRY PI ROBOT

- Raspberry Pi zero w controller module
- 1602 LCD display module
- triple color sensor module
- environment sensor module
- quad stepper motor module



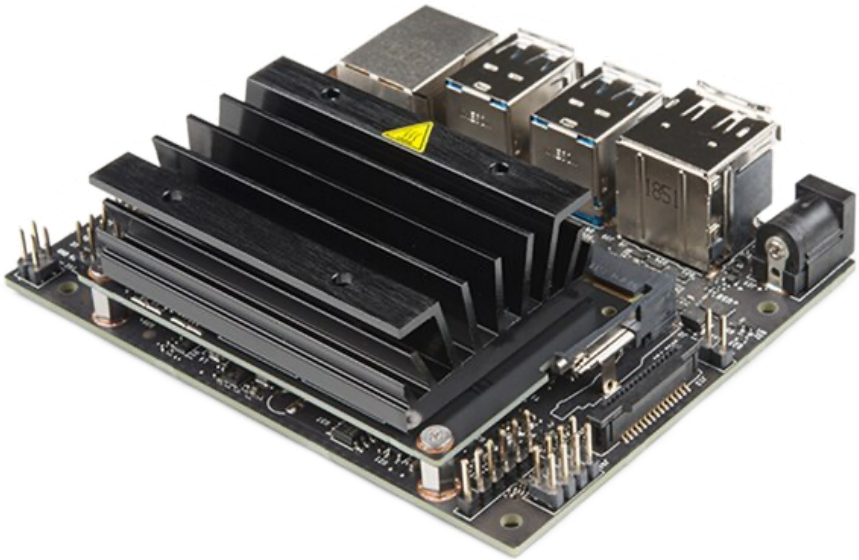
This robot allows a higher level of abstraction because the main controller is a Raspberry Pi zero single-board computer running Linux. So it's possible to program the robot using higher level languages like Java or Python. It's even possible to program the robot using graphical programming languages. So it could be used in a school context.
The function of the modules is abstracted. This means that you can easily call a function to display text on the display or collect color information. Another nice feature of the motor modules are the mecanum wheels. With them you can not only drive in tank mode - forwards, backwards and turn on the spot - but you can also drive sideways without turning.

I used this configuration, specifically revision one of the robot, with my students in the high school workshop. One of my students worked on programming an Android app remote and Java software for the robot. Two others built a robot arm and a camera arm, both of which can be controlled by the app.

# RESEARCH IN AUTONOMOUS DRIVING

This configuration is speculative due to the high cost of high-performance components for autonomous driving.

- NVIDIA Jetson Nano single board computer
- LiDAR Sensor
- Short Wave Radar Sensor
- Intel RealSense stereo vision camera



**NVIDIA Jetson Nano** by SparkFun Electronics from Boulder, USA - NVIDIA Jetson Nano Developer Kit,
CC BY 2.0,
https://commons.wikimedia.org/w/index.php?curid=81141809

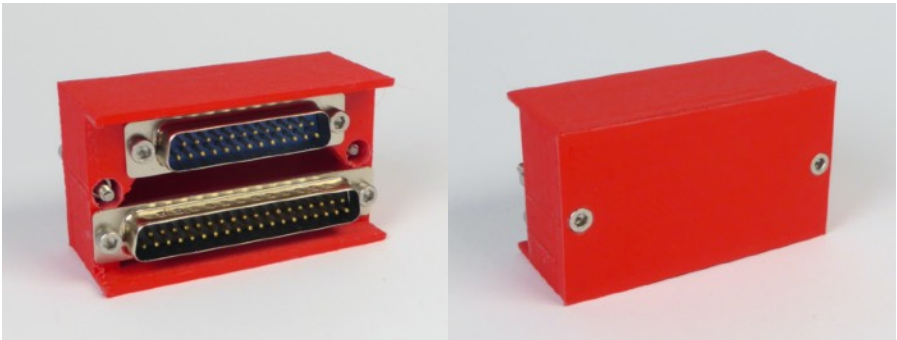
With this scenario, I would like to illustrate another use case for the robot. Currently, the automotive industry is doing a lot of research on artificial intelligence for autonomous driving. The main focus in this area is software development. As a result, many researchers have shifted their knowledge and focus away from the hardware side. In this context, they could use my tested platform for their driving studies. One possible configuration would be an NVIDIA Jetson Nano as the main controller. This single-board computer is more like a high-performance laptop than a Raspberry Pi. In particular, its dedicated graphics chip makes it state of the art for embedded machine learning applications, and for autonomous driving this powerful processor needs good and reliable sensors. The robot could be equipped with a short-wave radar, which is standard in today's cars. This sensor is commonly used for vehicle tracking and collision detection. In addition, an Intel RealSense stereo vision camera can be used for lane following and sign and signal recognition. Finally, a Light Detection And Ranging (LiDAR) sensor can be used to map the environment, which is necessary for situational awareness in slow and precise maneuvers such as parking.

# FUTURE WORK

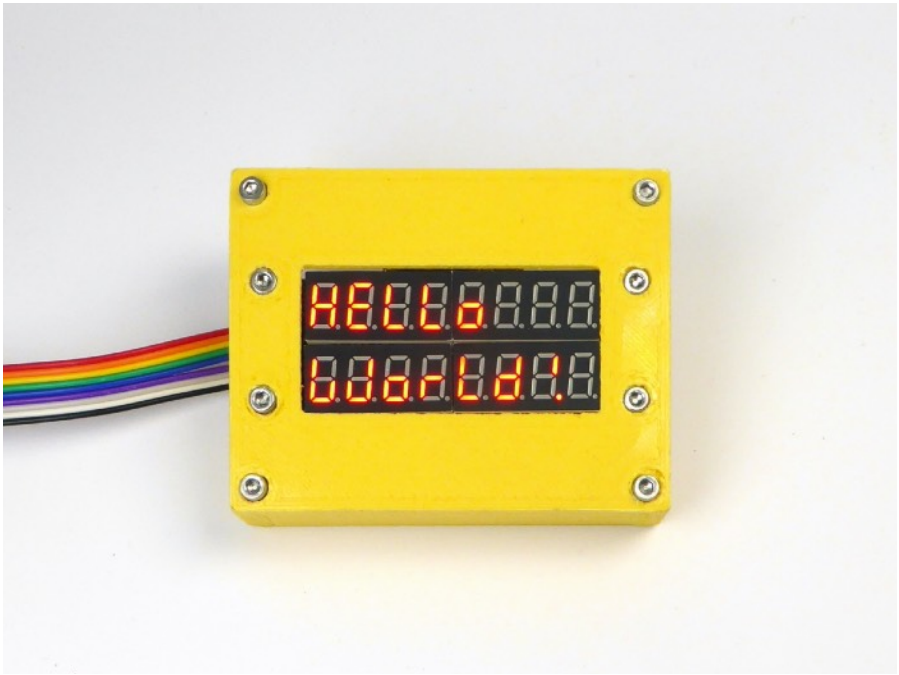
**Networking:**
The DB9 connector from the controller bay currently has no use for its five data pins, but they could be used in the future for a robot-wide network, simulating the current state of the art in automobiles, where most components have their own processor and communicate via one or more networks. This could be easily implemented by changing the electronics of the modules and adding a microcontroller to each one.

**Hot-Plugging:**
Currently, modules can only be swapped while the robot is powered off and if they are electrically compatible. So in most cases you have to change or modify the interposer as well. If you don't want to use a system-wide network as discussed above, you could upgrade the interposer and the modules.
The modules would need protected input/output drivers, so the design would need to include fuses and voltage protection diodes. In addition, each output device would need some sort of bus driver, optionally optically isolated for maximum rigidity.
Changes to the interposer would be the need to include a reconfigurable logic chip. In this application, a Field Programmable Gate Array (FPGA) would be most appropriate. This type of logic chip has an internal structure that allows you to route any input to any output. In addition, there are two unused pins on the DB37 connector of the interposer that would be used for the programming interface between the controller module and the smart interposer.

**Other chassis:**
The standardization of the chassis and modules makes it possible to make smaller and larger versions of it.
One thing I would like to develop would be two smaller versions of the chassis. The goal would be to keep the same modules, but in one case shorten the chassis by a quarter. This would result in double size motor bays instead of triple size ones. I could also use a hardwired configuration, resulting in fewer cables and no need for the interposer.
The second step after this modification would be to reduce the size even further. Because of the battery compartment, I would probably have to add an upright bay in the front. Also, I would either keep a double sized motor bay or reduce it to a single motor bay.
The last option would be to increase the overall size. This would probably require me to rethink the data connections. It would also increase the weight of the robot. So all these facts should be considered.



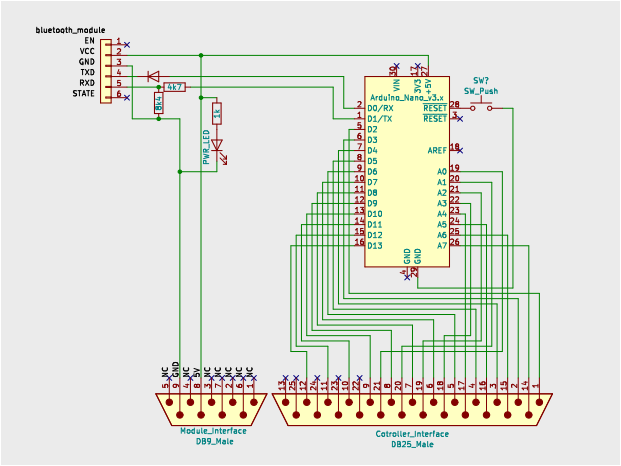
**A dumb interposer as used in all current designs of my robot. The configuration is changed by soldering patches between the two connectors.**



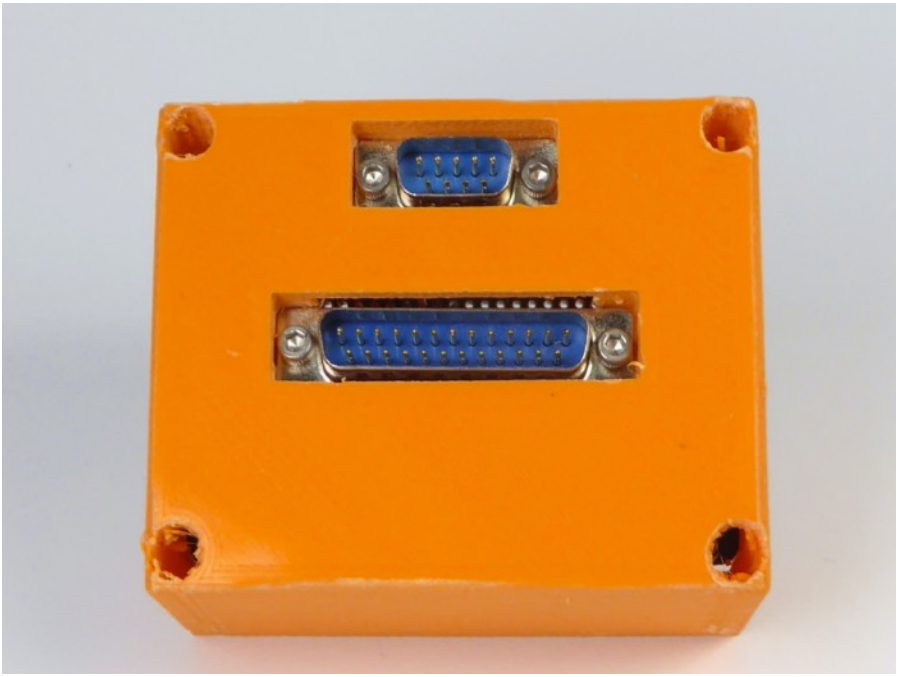
**The obligatory “Hello World!“ program on a seven-segment display.**

# CONTROLLER MODULES

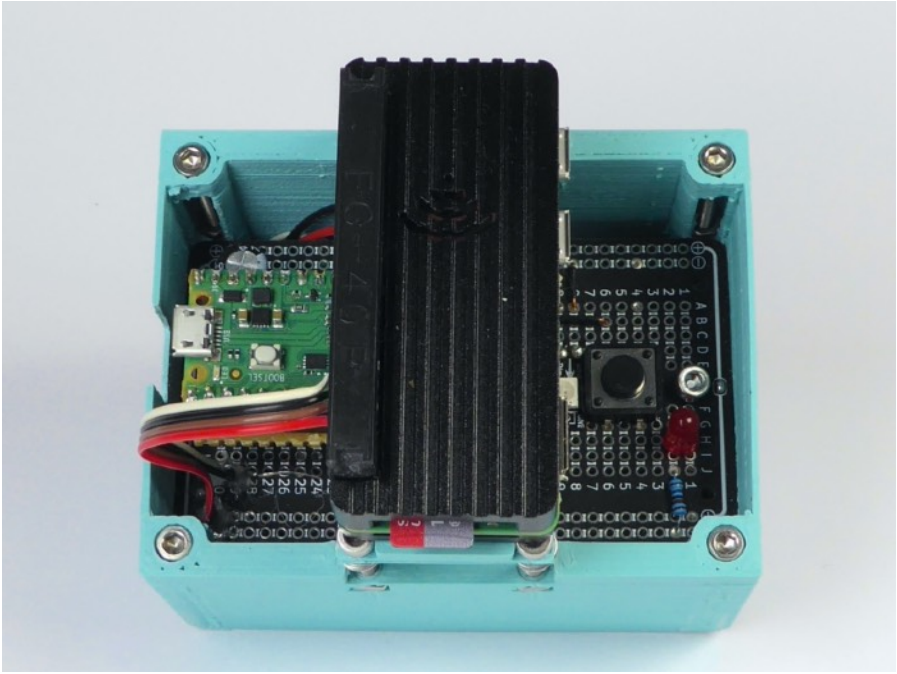
# ARDUINO NANO CONTROLLER MODULE

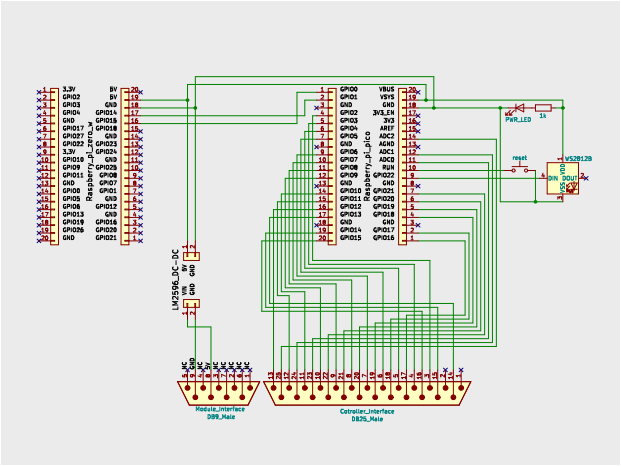


This controller is based on an Arduino Nano board. It also has a Bluetooth module for wireless communication. Its strength lies in the simplicity of the microprocessor, making it an excellent choice for learning about hardware and low-power programming.



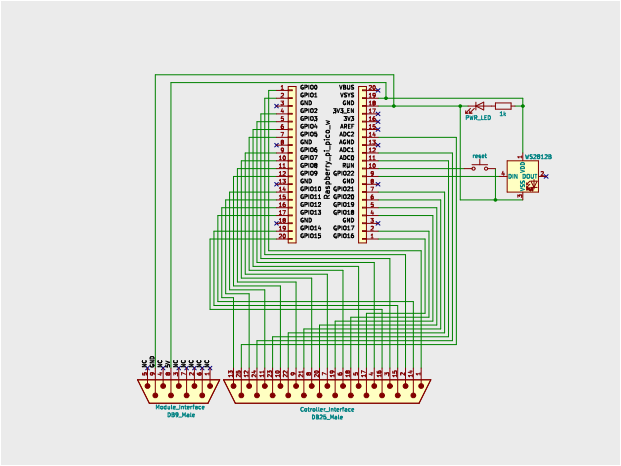
# RASPBERRY PI ZERO W CONTROLLER MODULE



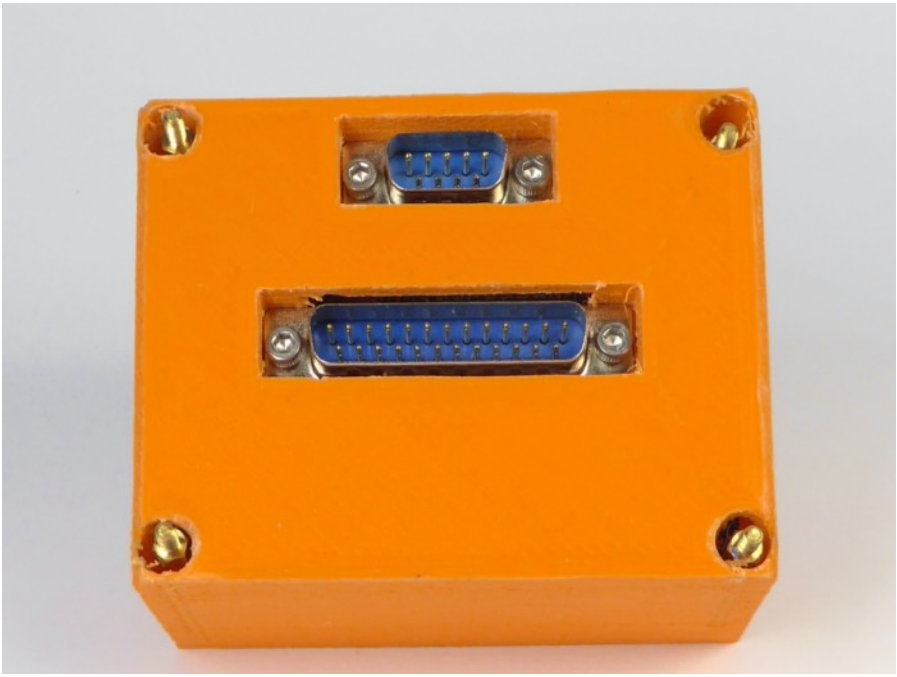


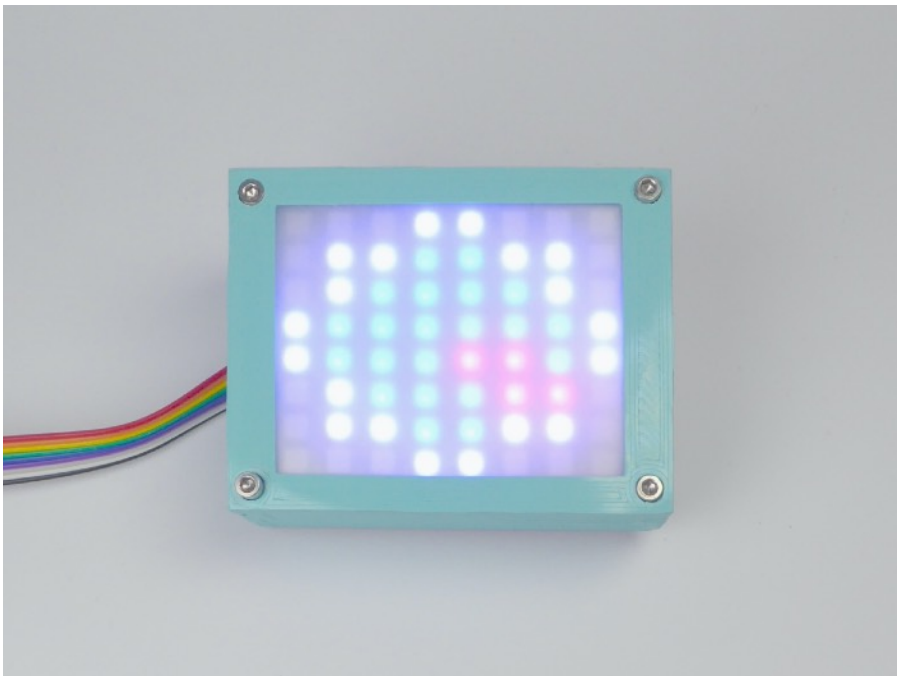
This controller is based on the Raspberry Pi Zero W, a single board computer. The "W" in its name stands for "wireless", as it has both WiFi and Bluetooth capabilities. It also runs Linux, which makes it suitable for various applications. For example, you can use this board to create a wireless remote control for the robot.
The second microcontroller in the module is the Raspberry Pi Pico. It communicates with the Zero via a serial UART connection and handles all connections to and from the modules.

# RASPBERRY PI PICO W CONTROLLER MODULE



This controller uses a Raspberry Pi Pico microcontroller. The RP2040 chip, developed by the Raspberry Pi Foundation, powers the Pico. You can program it using either C/C++ through the Arduino IDE or MicroPython. This makes it an excellent choice for getting started with textual hardware programming.

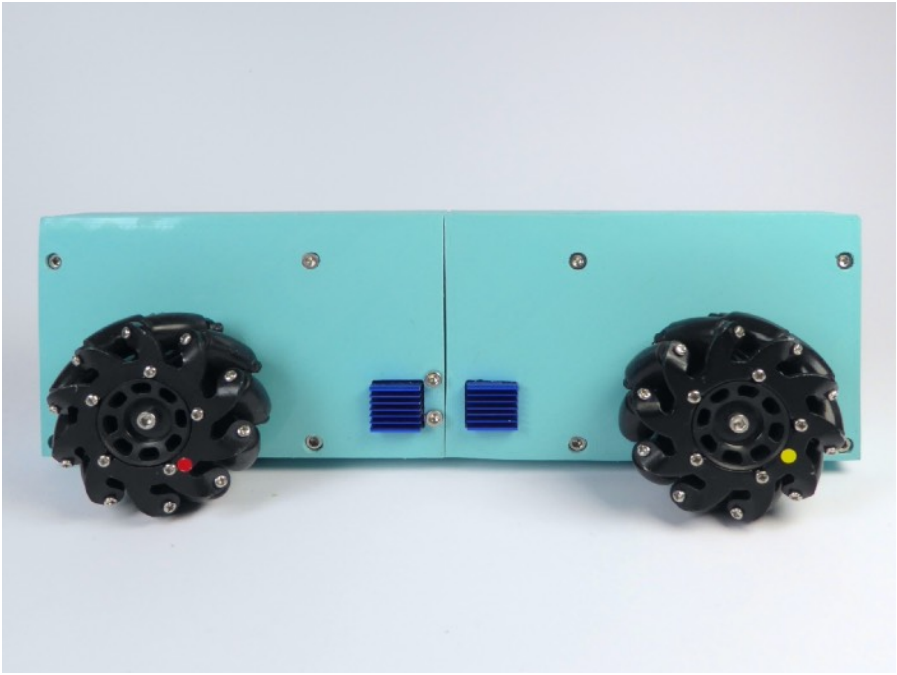


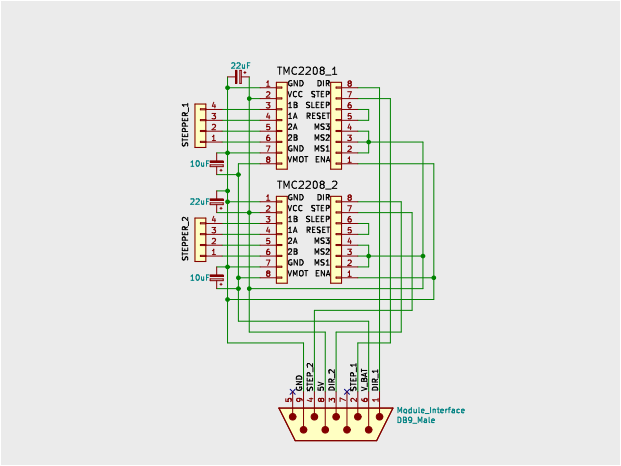


**A compass drawn on an 8x8 RGB LED matrix.**

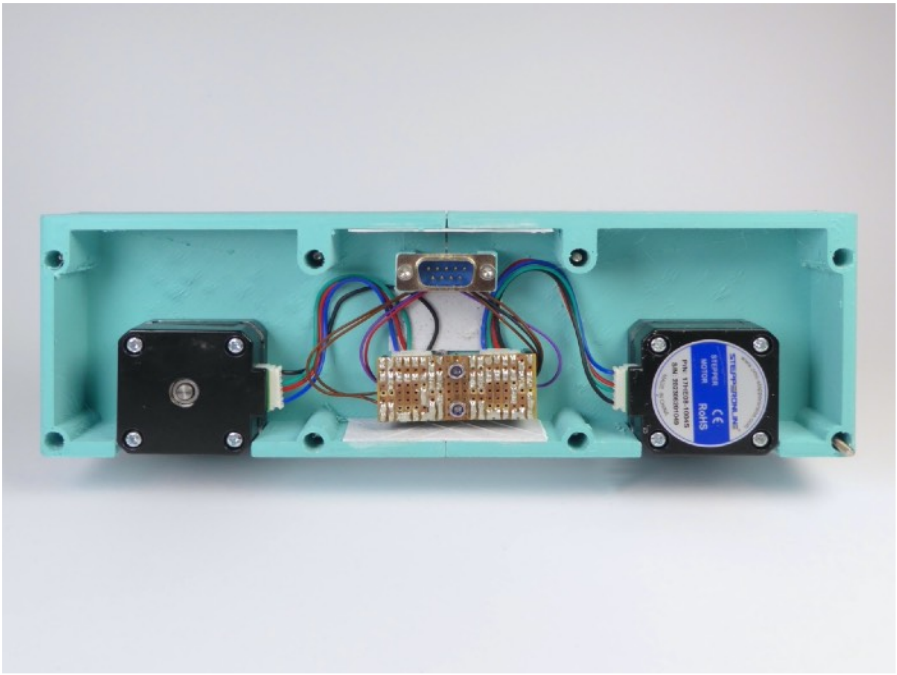
# MOTOR MODULES

# DOUBLE STEPPER MOTOR MODULE

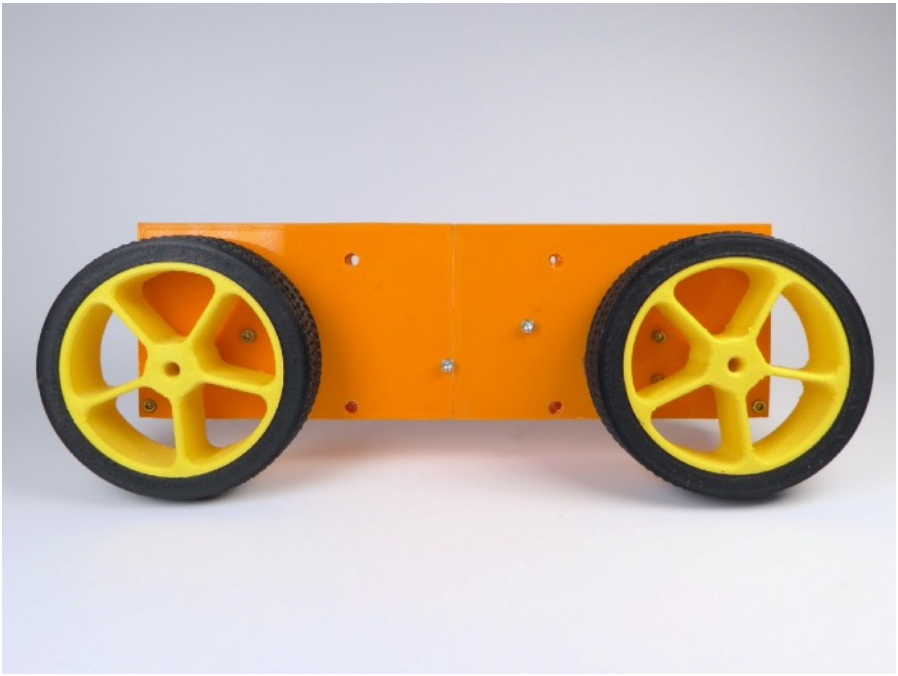


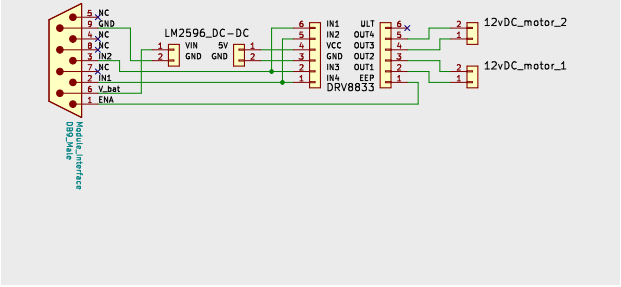


This module uses two independently controllable stepper motors. With the addition of the mecanum wheels, a four-wheeled robot can move in any direction without turning. In addition, the tmc2208 stepper drivers used in this module are necessary for nearly silent rotation.
This model was designed before the major redesign of the motor modules. However, updated model files will be released.

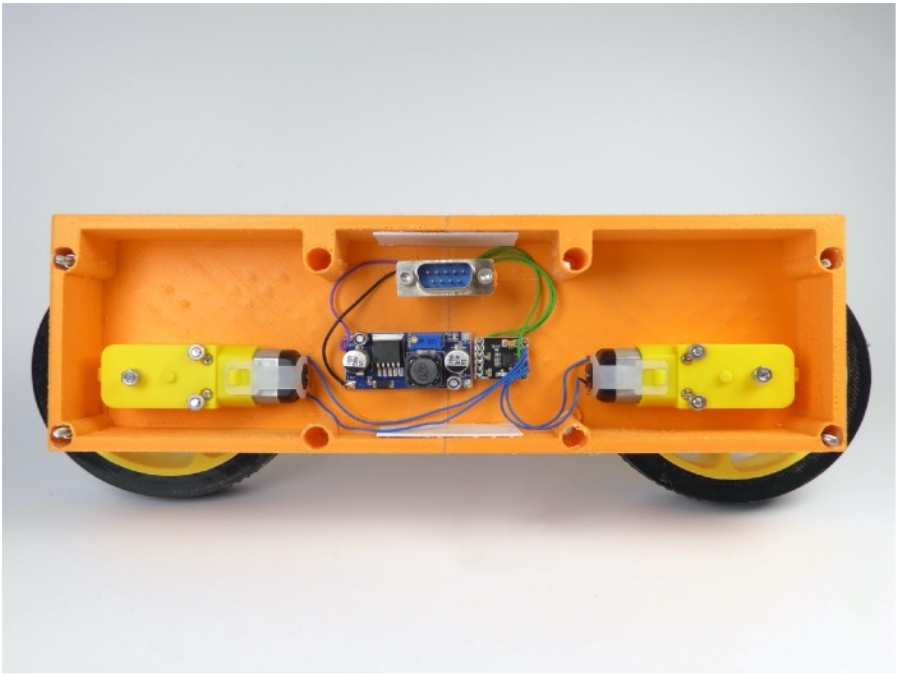


# DOUBLE LIGHTWEIGHT MOTOR MODULE

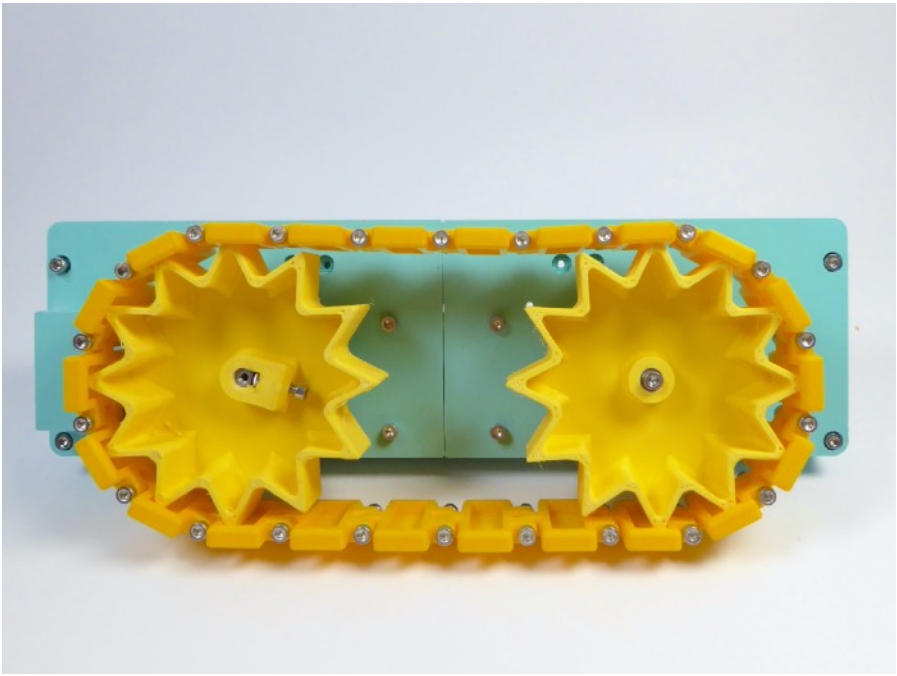


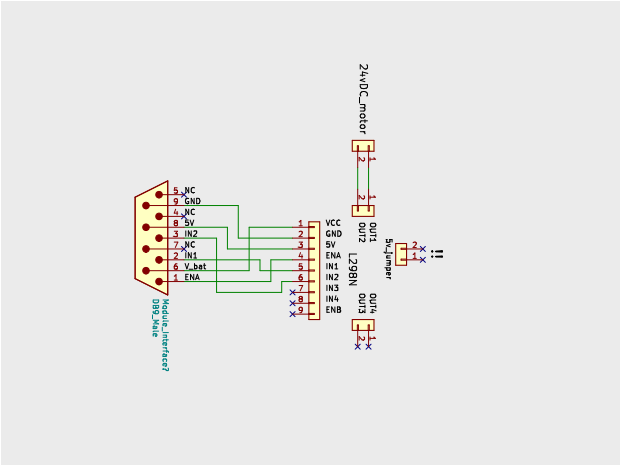


This motor module is one of the simplest. It uses two cheap but effective DC motors controlled by a drv8833 H-bridge driver. Both motors are connected in parallel. This allows a tank-like differential drive.

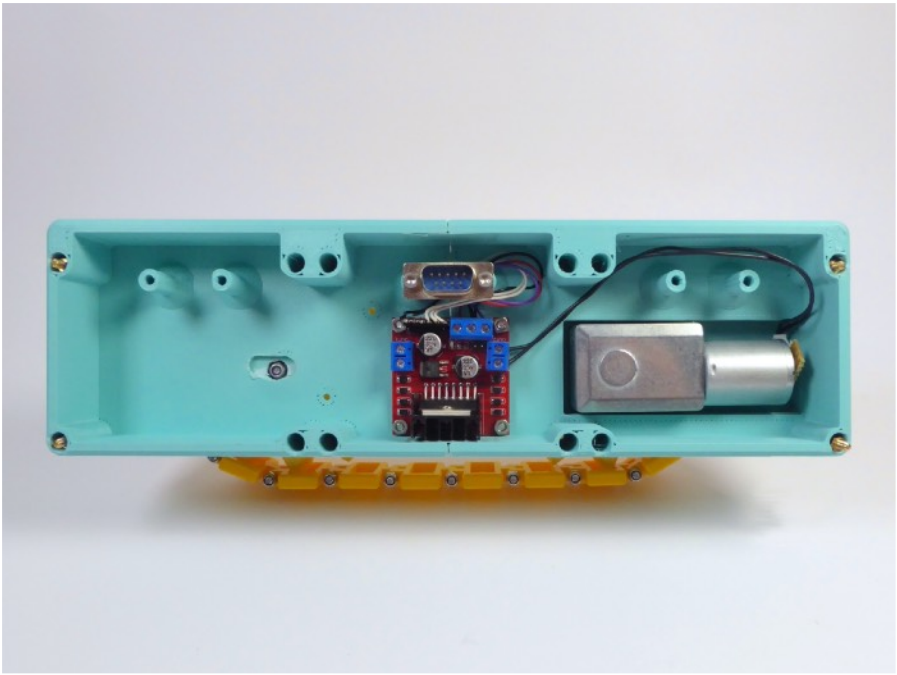


## SINGLE MOTOR TRACK MODULE

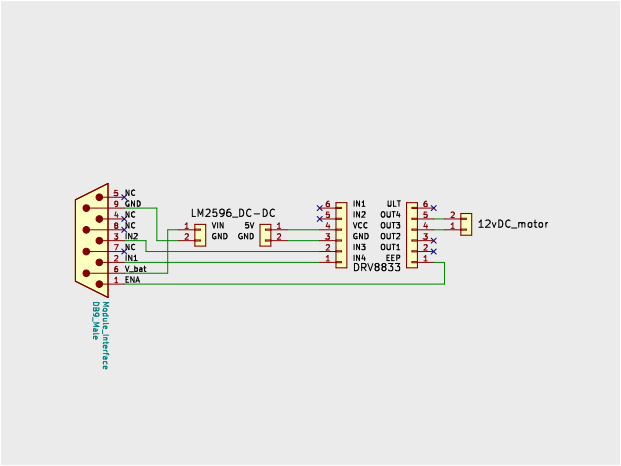




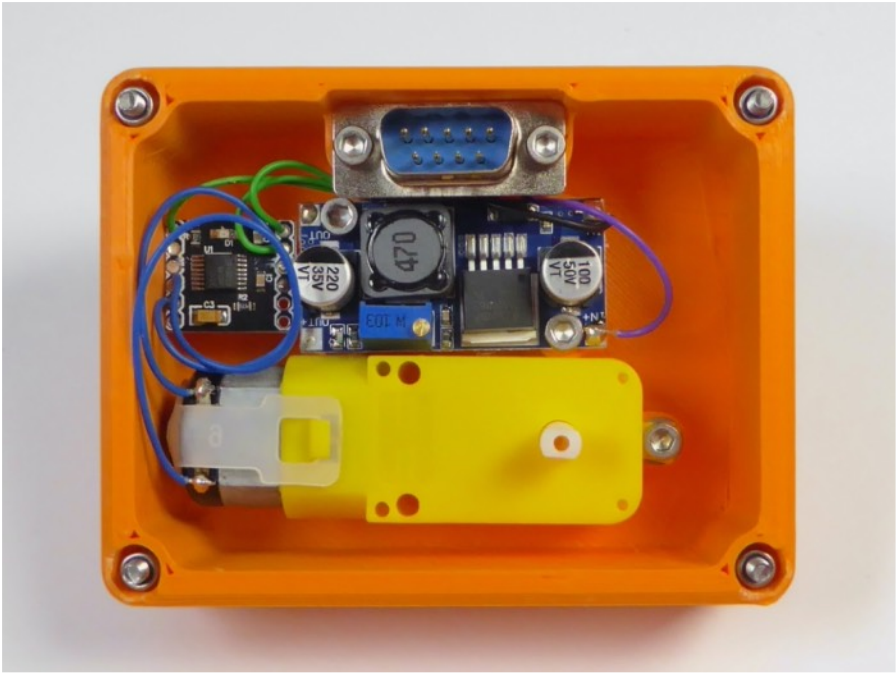
This module uses a DC motor with a worm gear drive. Therefore, it has a lot of power at low speed, which is perfect for a tracked vehicle. In addition, the motor driver used is one of the most common and robust. All in all, this module implements a very solid drive train.

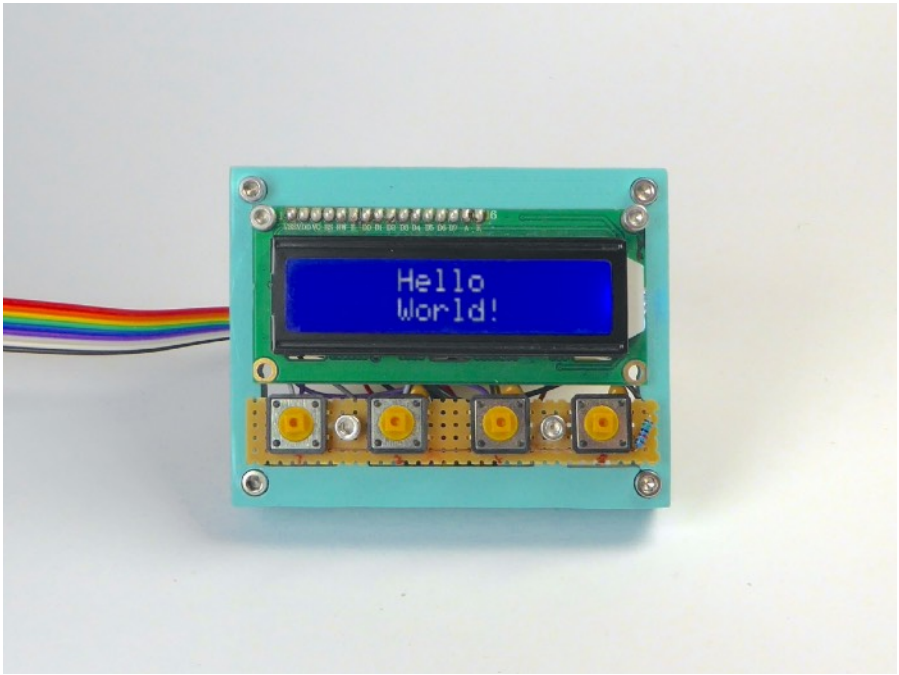


## SINGLE LIGHTWEIGHT MOTOR MODULE



This module is an example of a very compact single motor module. It's very similar to the double lightweight motor module. I also did not add a wheel to this module because it would be a great starting point for alternative methods of movement.

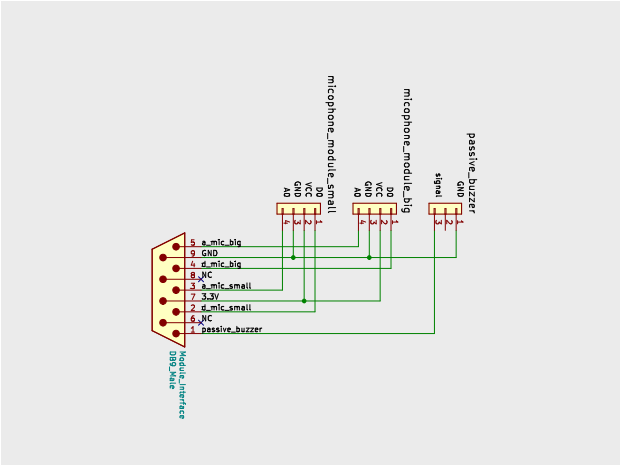




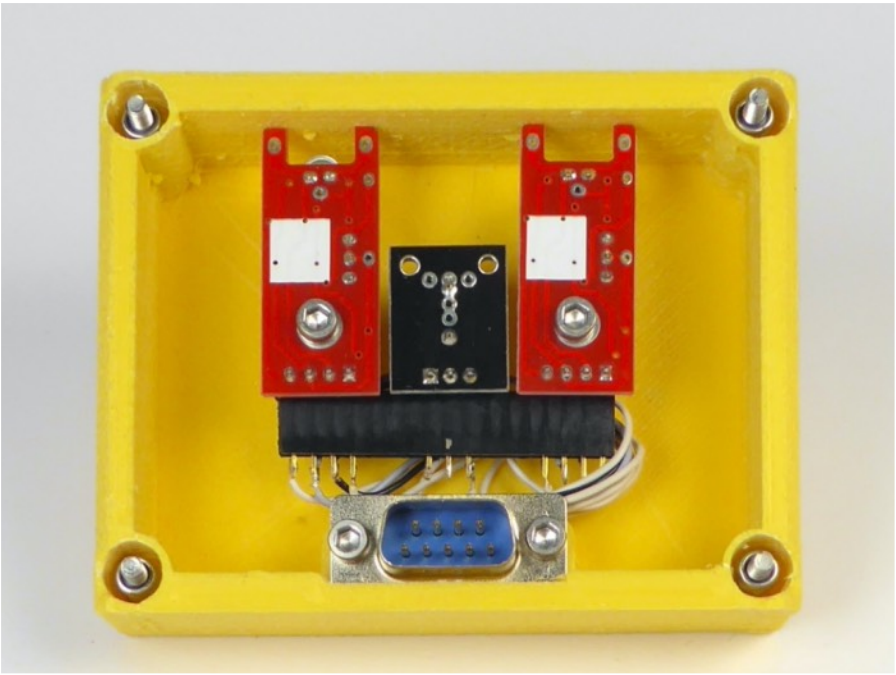

**Two classics combined with a two-line LCD and the “Hello World!“ program.**

# INTERFACE MODULES

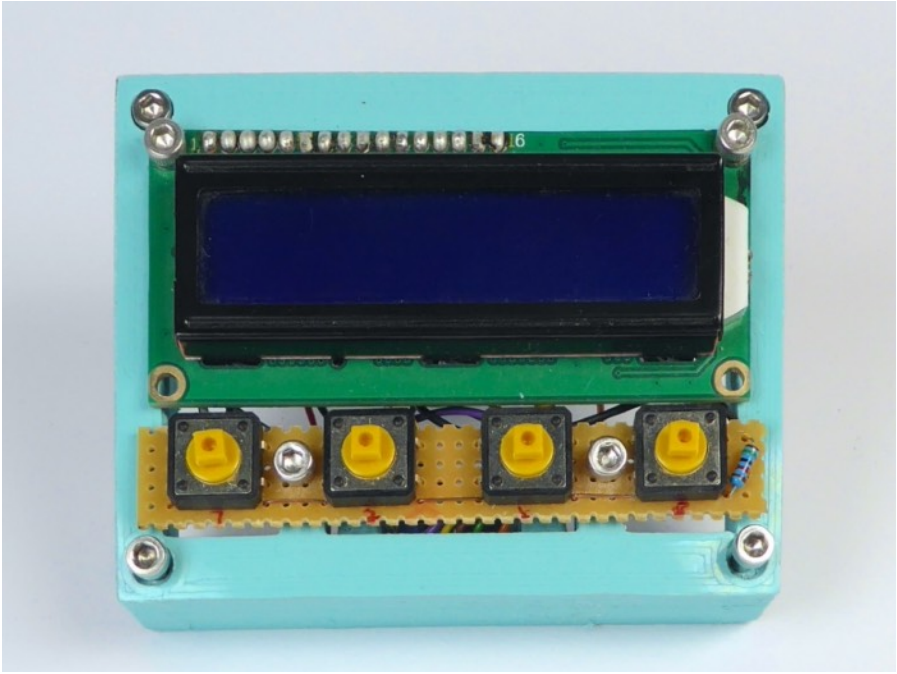
# AUDIO INTERFACE MODULE

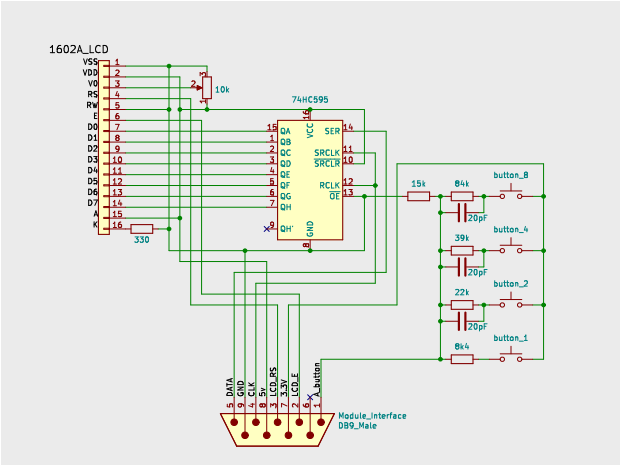


This module has two microphone modules and one speaker module. The microphone modules differ in their pitch sensitivity. In addition to their analog outputs, they have a digital output that is activated when a certain loudness threshold is reached. This makes it easy to react to clapping sounds, for example, by making the robot stop.

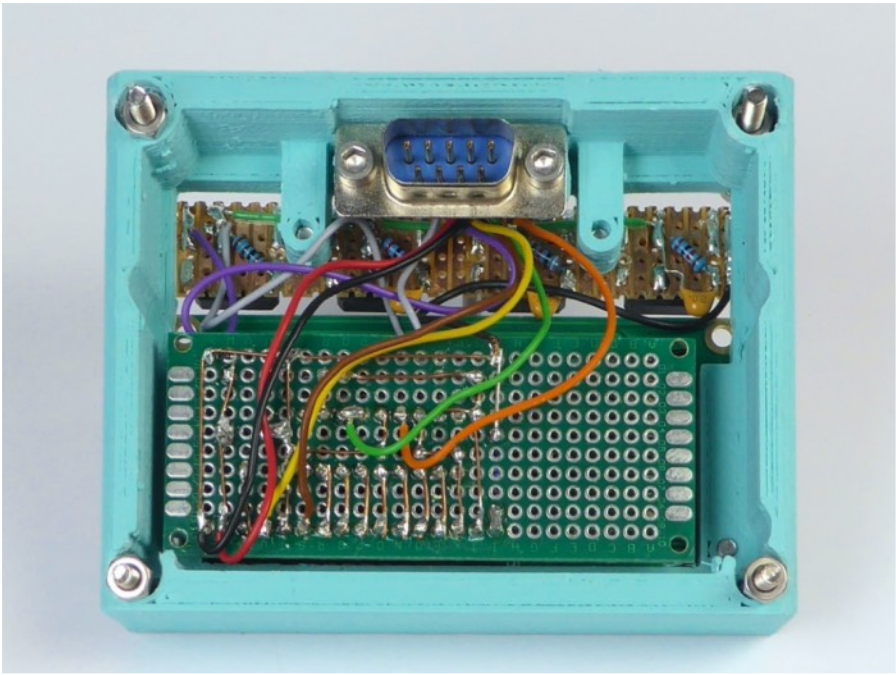


# LCD AND BUTTON INTERFACE MODULE

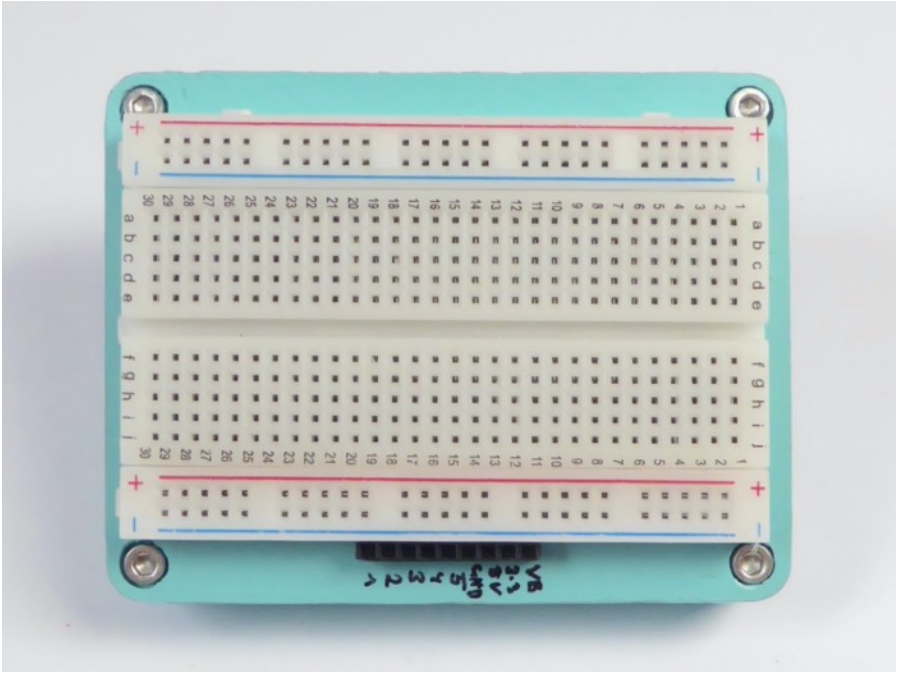




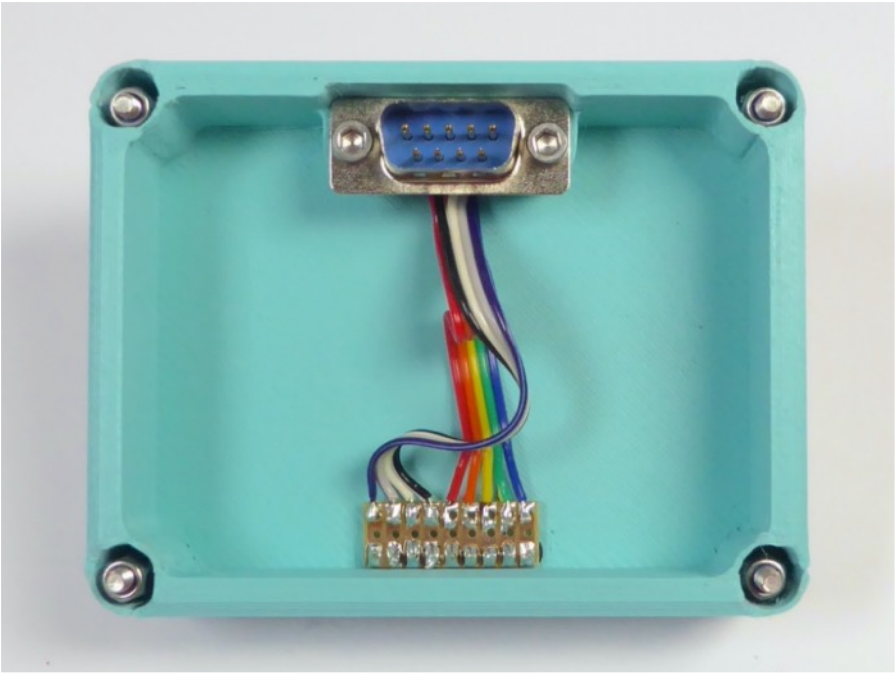
This module consists of a two-line, 16-character LCD display and a four-key keypad. The display can show any ASCII character as well as eight user-defined 5x8px graphics. In addition, the keypad can be used to record any single or combined input from the four keys.

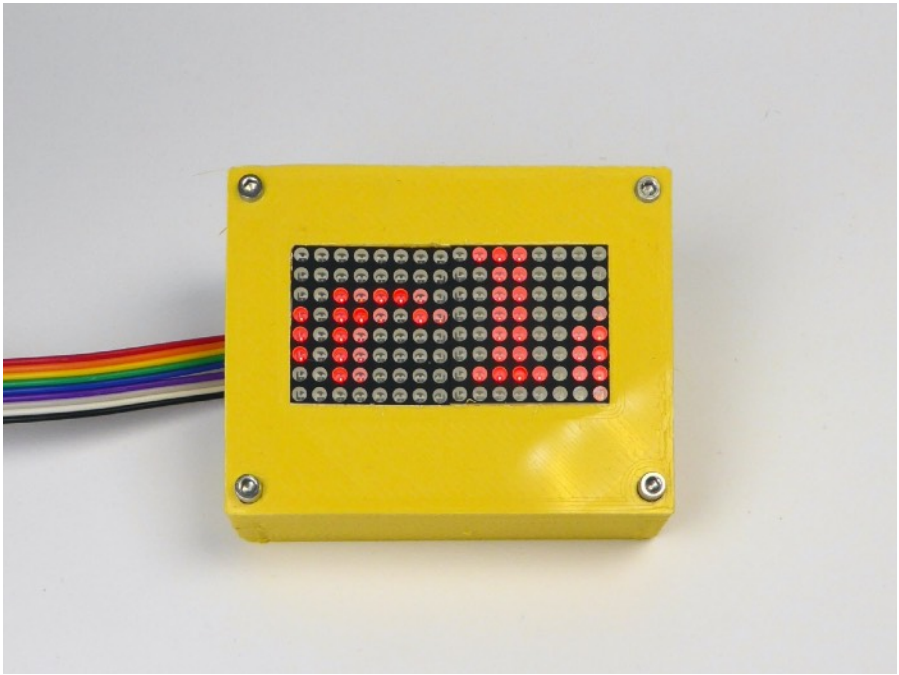


## BREADBOARD PROTOTYPING MODULE



This is the simplest module, but it is the starting point for many new ones. The breadboard allows solderless prototyping and the pin header provides all the connections to the chassis.

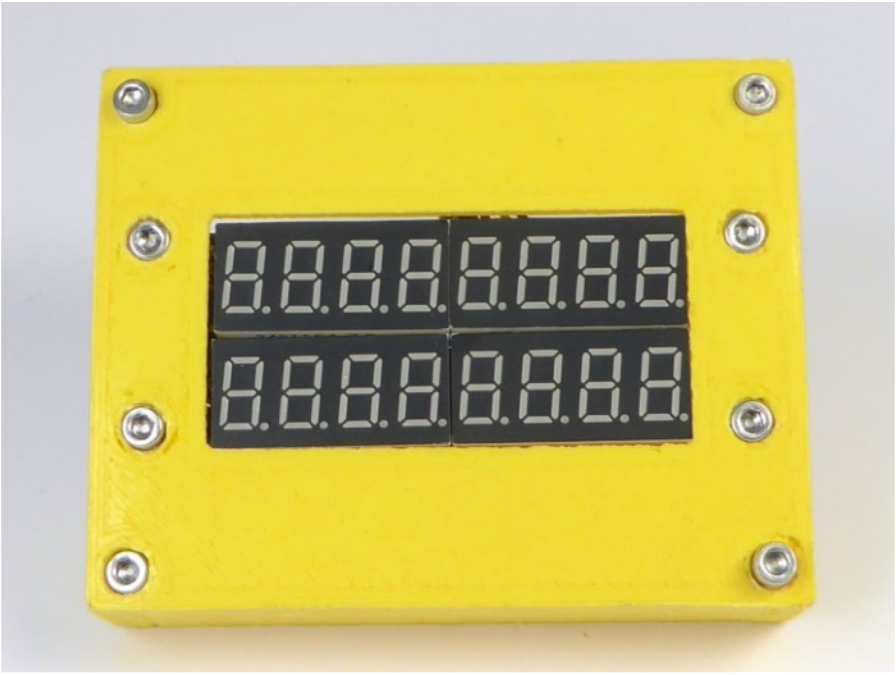


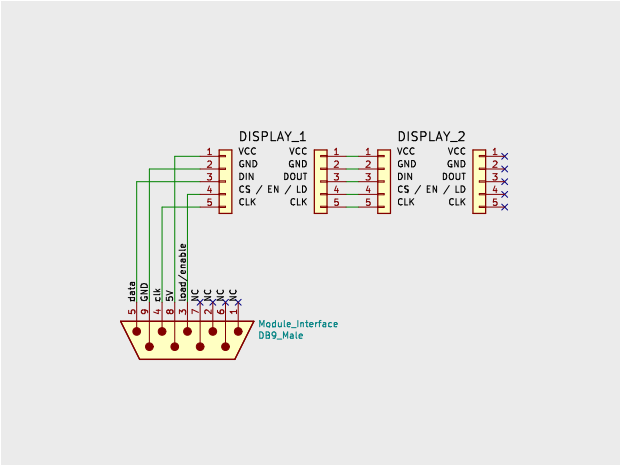


**An LED dot-matrix display that scrolls a message.**

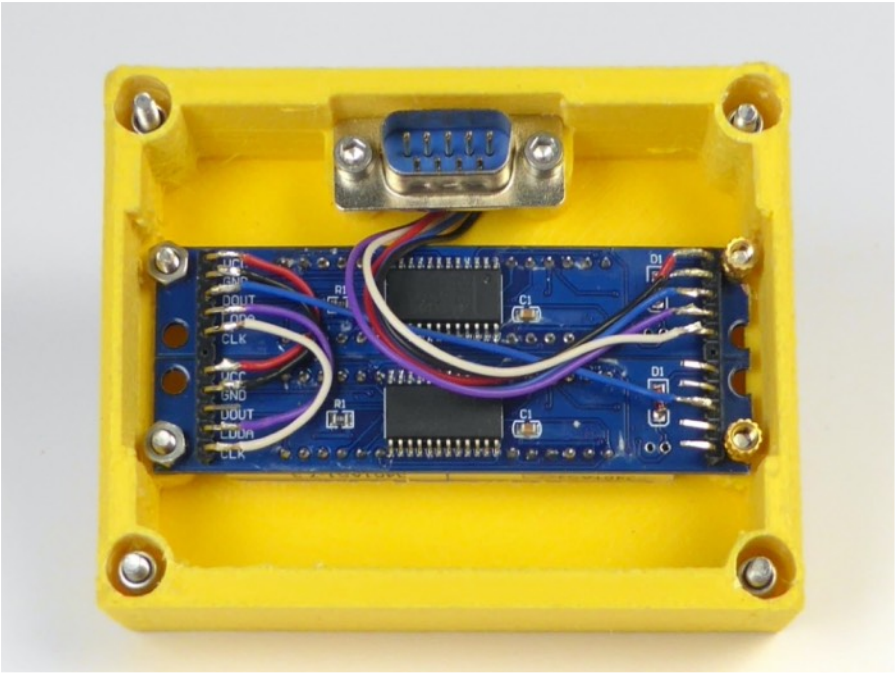
# DISPLAY MODULES

# 2X8 SEVEN SEGMENT LED DISPLAY MODULE

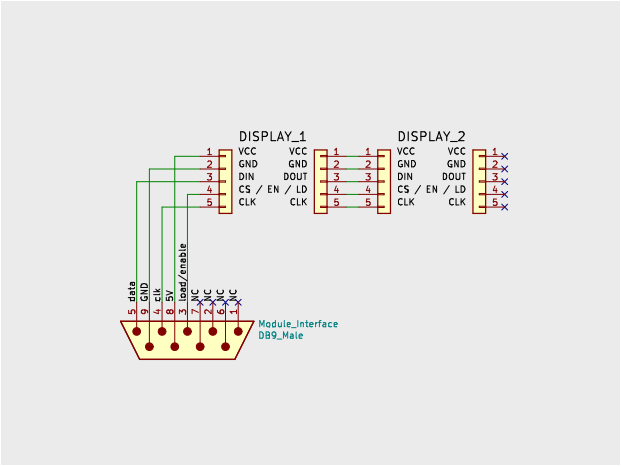




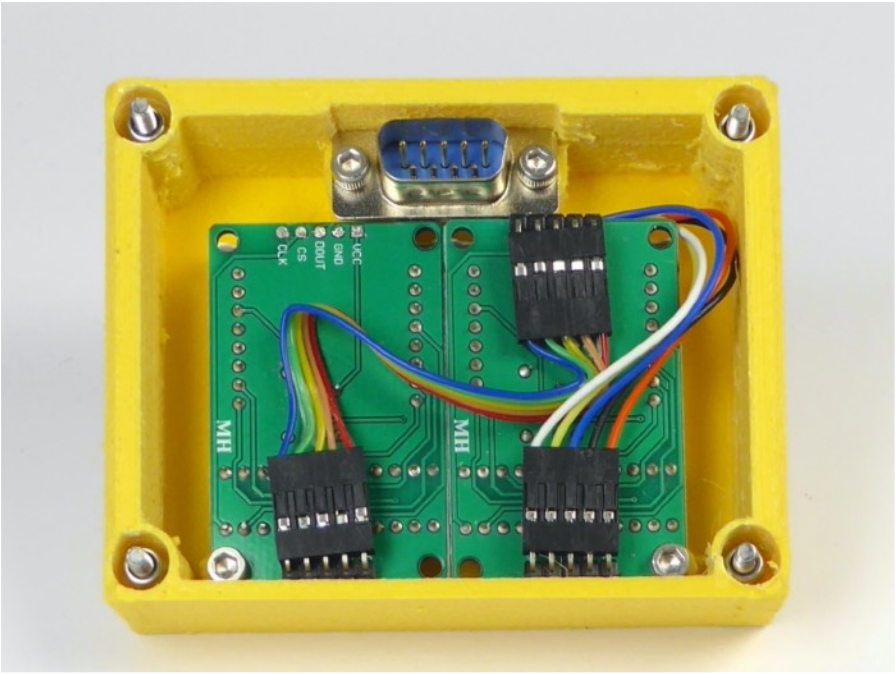
This module consists of two seven-segment displays. Each one is driven by a max7219 chip and has eight characters in width. The chip allows dimming of this display and has a decoder from decimal numbers to the seven segment display font. In addition, each segment can be individually addressed, allowing it to be used for any display.



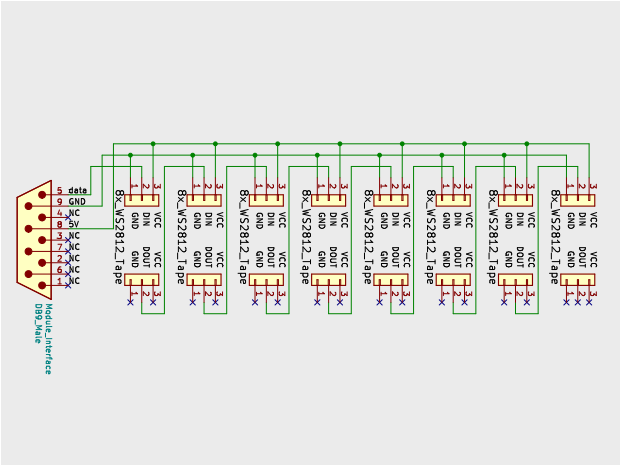
# 2X8X8 DOT MATRIX LED DISPLAY MODULE



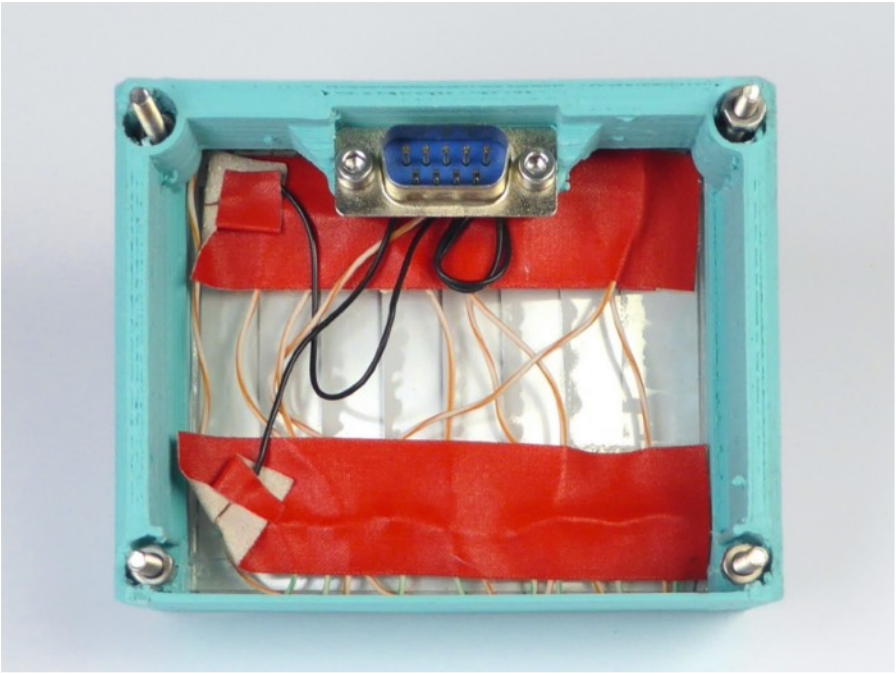
This module consists of two 8x8 dot matrix displays. Each is driven by a max7219 chip, which allows the display to be dimmed. In addition, each pixel can be individually addressed. For example, this module can be used for scrolling text as found on buses.



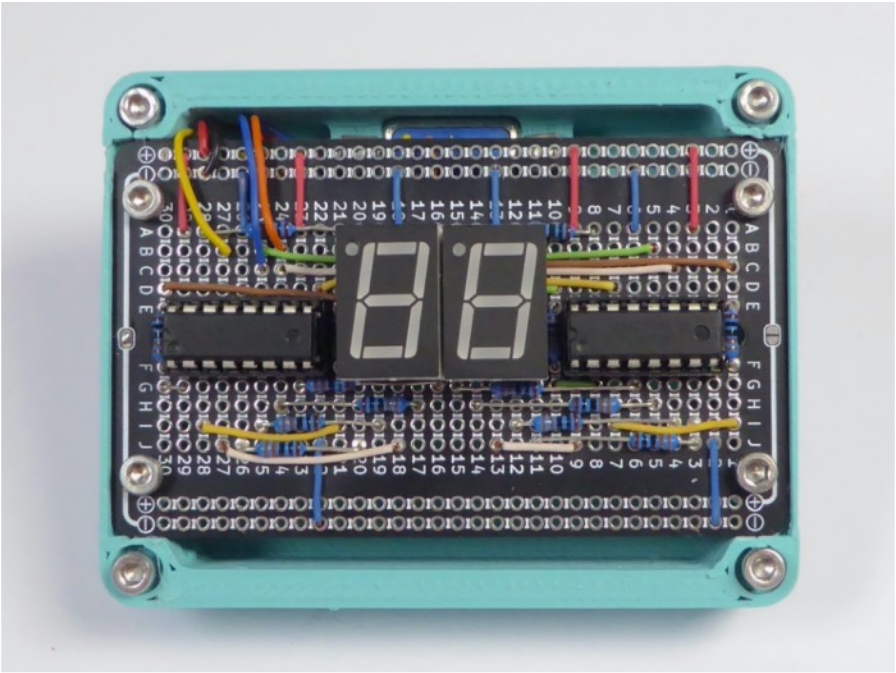
# 8X8 RGB LED DOT MATRIX DISPLAY MODULE

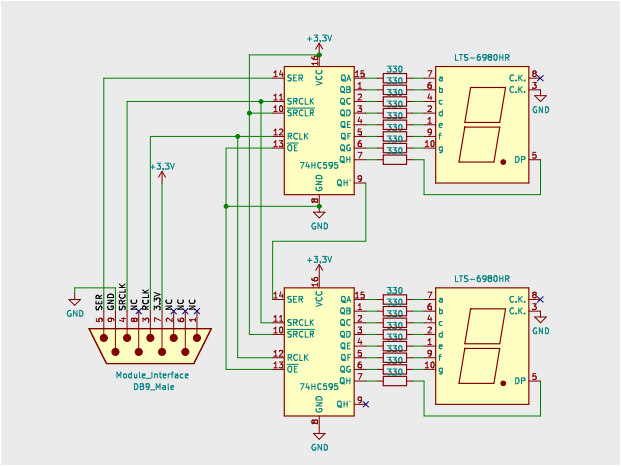


This module has eight rows of eight ws2812 RGB LEDs. These LEDs can be individually addressed and set to an 8-bit RGB value. This allows for over 16.6 million colors at high display brightness. Even with the LEDs dimmed to 30 percent, the display is clearly visible in direct sunlight.



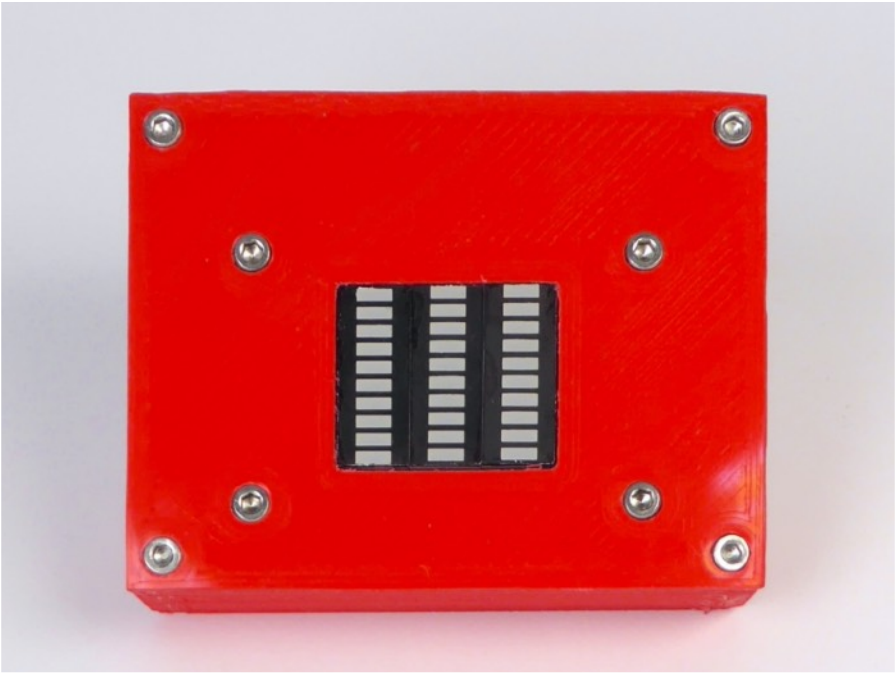
# DOUBLE SEVEN SEGMENT LED DISPLAY MODULE

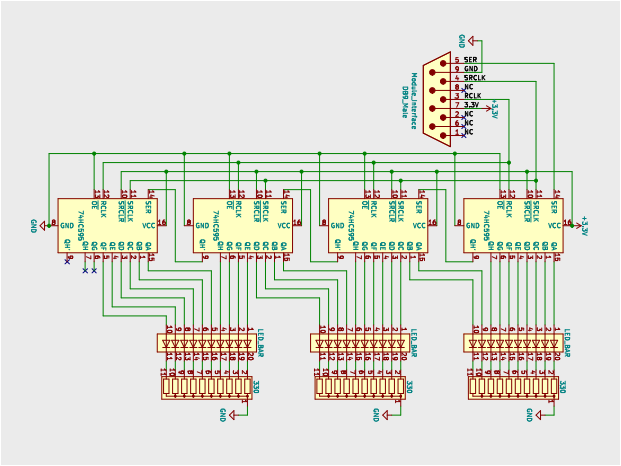




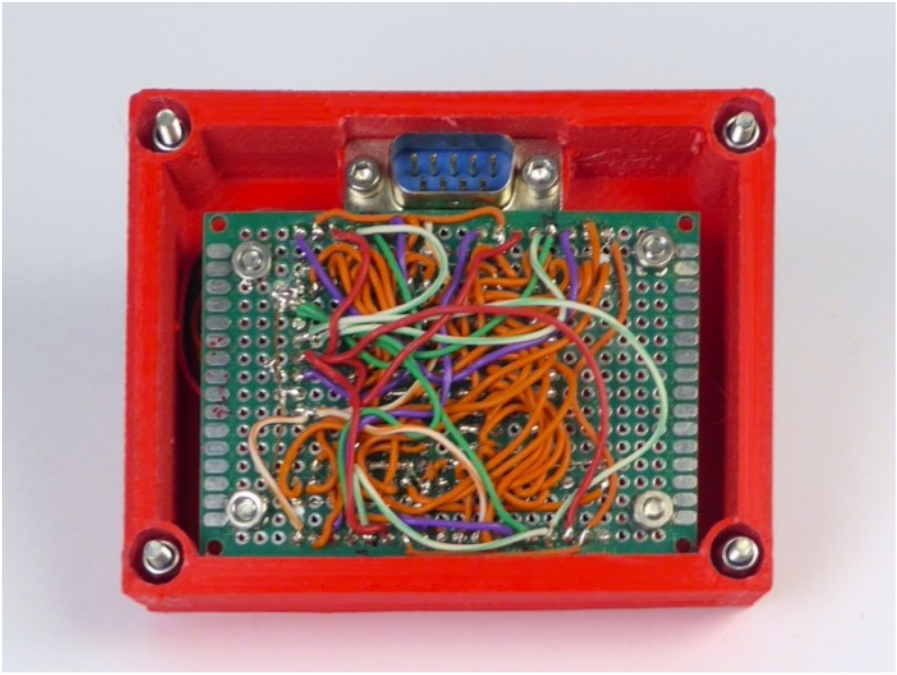
This module uses a classic approach to drive the display. It uses two 74HC595 8-bit shift registers. These shift registers are both fast and relatively easy to use. Their low-level nature makes them excellent for learning the basics of communication. So I decided to show the whole board to allow the user to understand the inner workings.

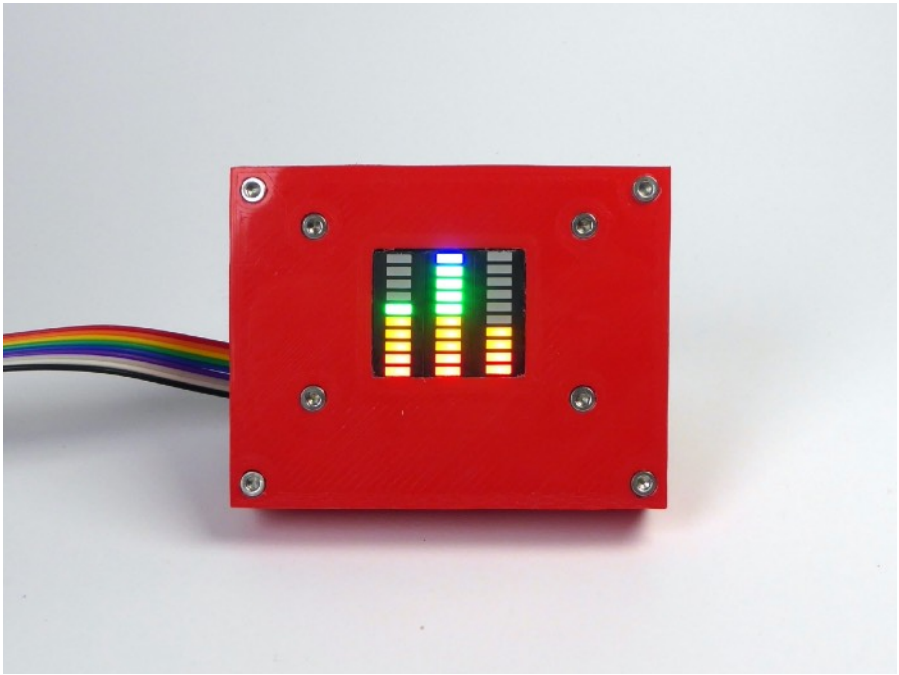
# TRIPLE BARGRAPH LED DISPLAY MODULE





This module uses four 74HC595 shift registers to drive three bargraph displays. It's very useful for learning low-level programming and is especially useful when combined with the TCRT5000 triple line sensor module. In this configuration, you can read the three light levels with the sensor module and display their intensity with the bargraph module.

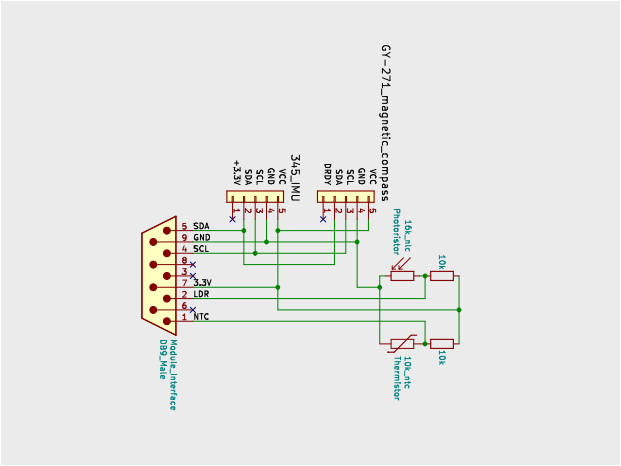




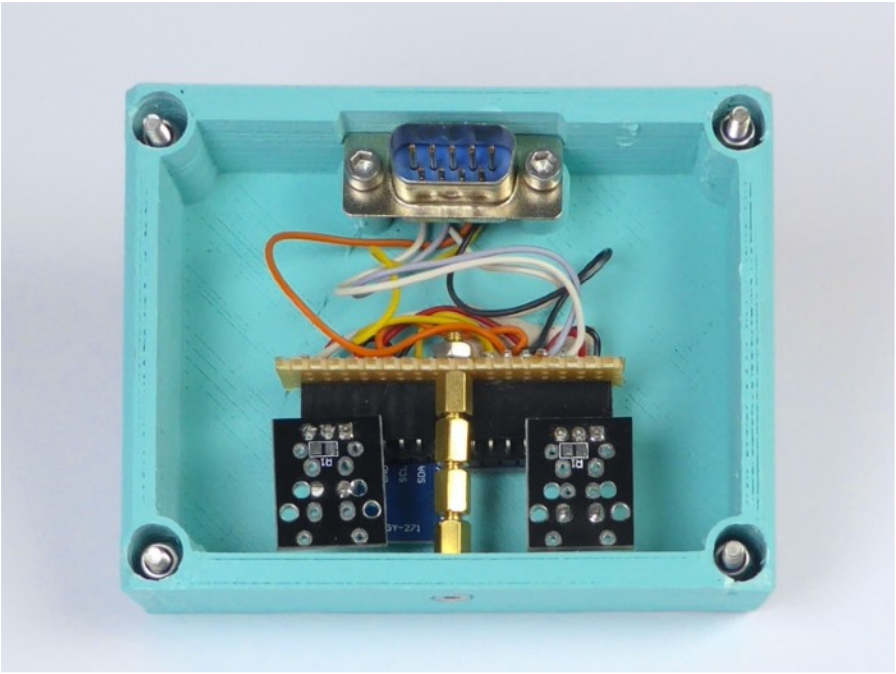
**An LED bargraph display that indicates signal strength.**

# SENSOR MODULES

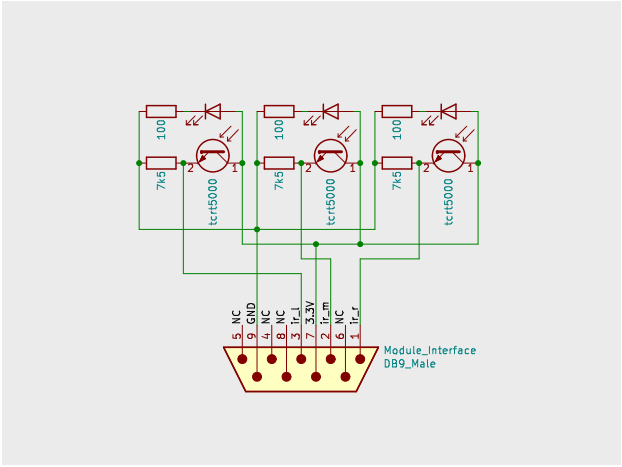
# ENVIRONMENT SENSOR MODULE



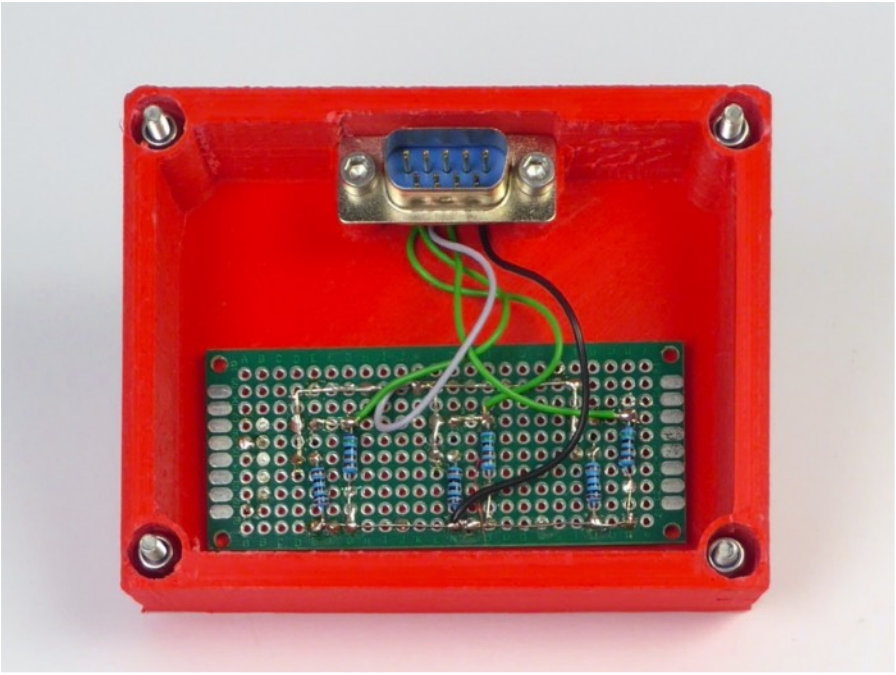
This module has four different environmental sensors. Two basic analog sensors, a temperature sensor, and a light sensor. It also has an inertial unit to measure gravity and external accelerations. The fourth module is a magnetic compass unit that measures the strength and direction of the magnetic field.



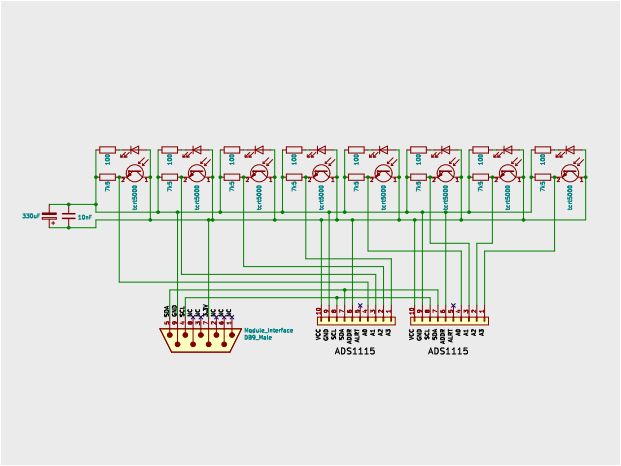
# TRIPLE TCRT5000 LINE SENSOR MODULE



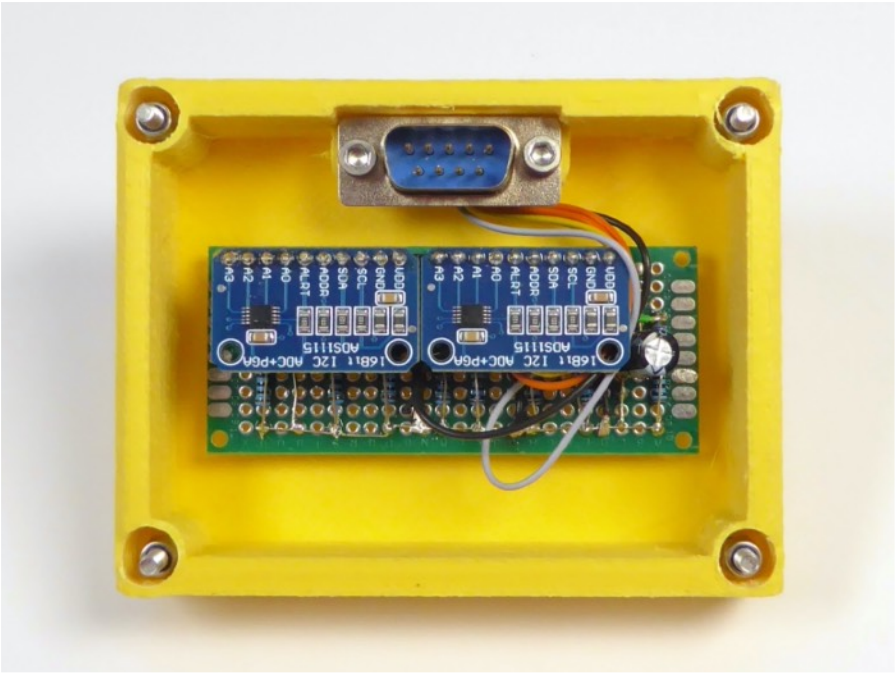
The most basic analog infrared line detection module. It measures the intensity of the reflection at three points below the module. To enable autonomous driving of a robot, you can use the middle sensor to detect the reflection of a black line. If one of the two outer sensors also detects this line, you must correct the steering to stay directly on the line.



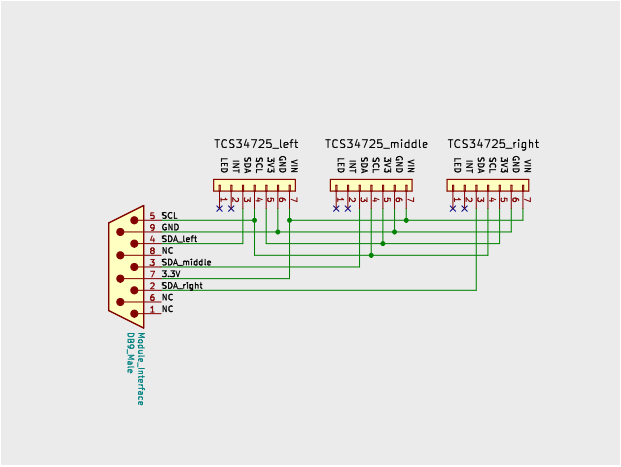
# OCTUPLE TCRT5000 LINE SENSOR MODULE



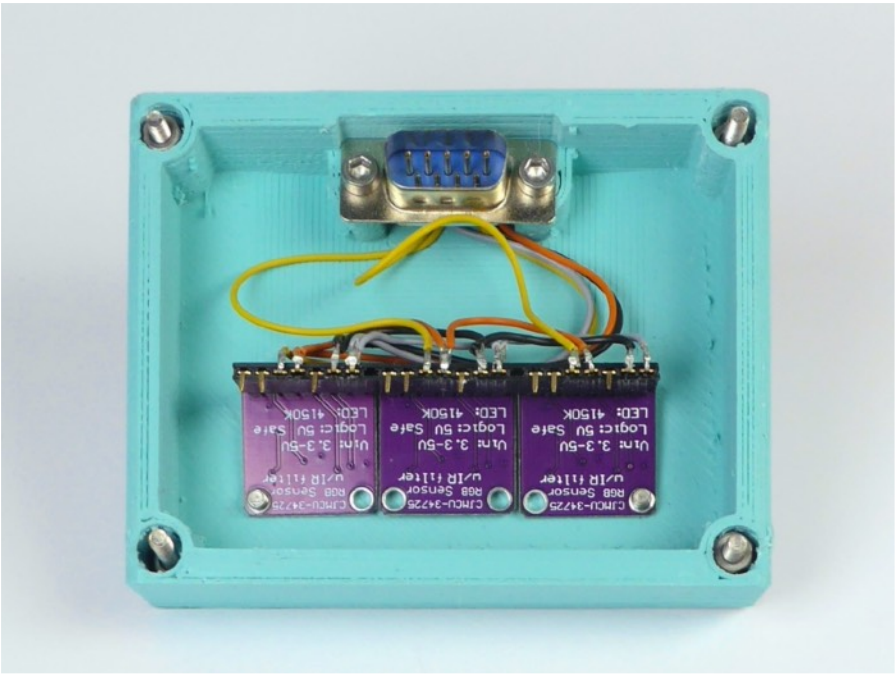
This module has eight analog infrared reflectance sensors. These sensors are read by two ADS1115 analog-to-digital converters. For example, you can use this module to detect the position and shape of a black line under the robot and use this information for autonomous driving.



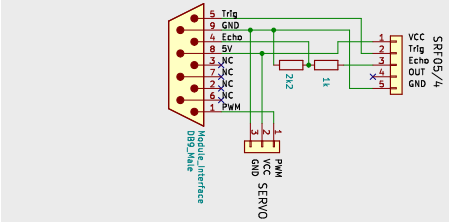
# TRIPLE COLOR SENSOR MODULE



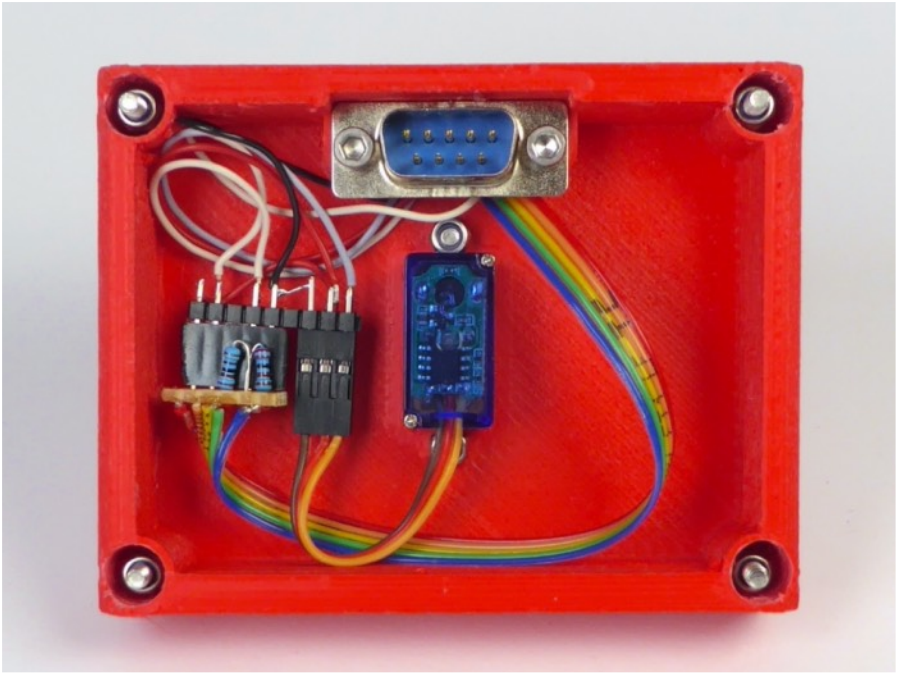
This is one of the advanced line detection sensor modules. In addition to detecting the total reflectance of the underlying surface, this module can also measure the corresponding RGB value. With this additional information, you can not only follow lines, but also use different colors for markers or choose alternative routes.



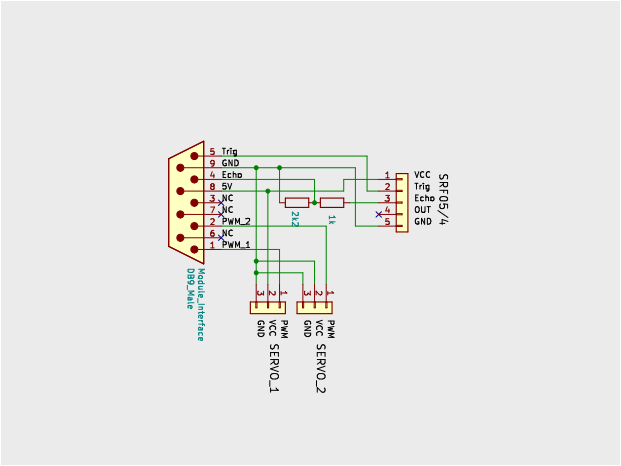
## SINGLE SERVO ULTRASONIC MODULE



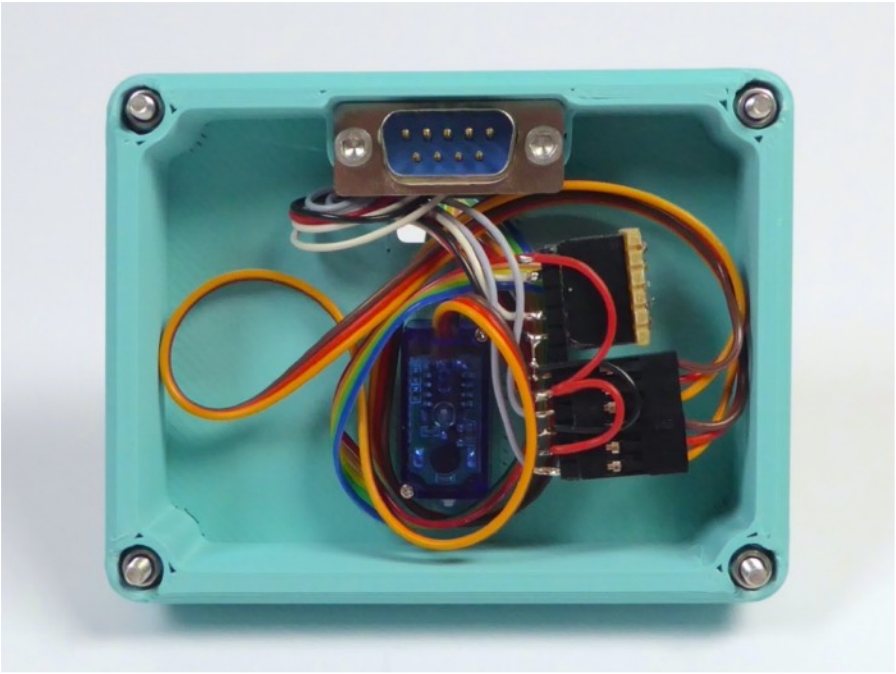
This module uses a very small and simple 9g servo motor to control the direction of the ultrasonic sensor. With this function, you can implement the most basic type of environmental mapping by continuously sweeping the sensor from left to right and measuring the distance to any object.



# DOUBLE SERVO ULTRASONIC MODULE

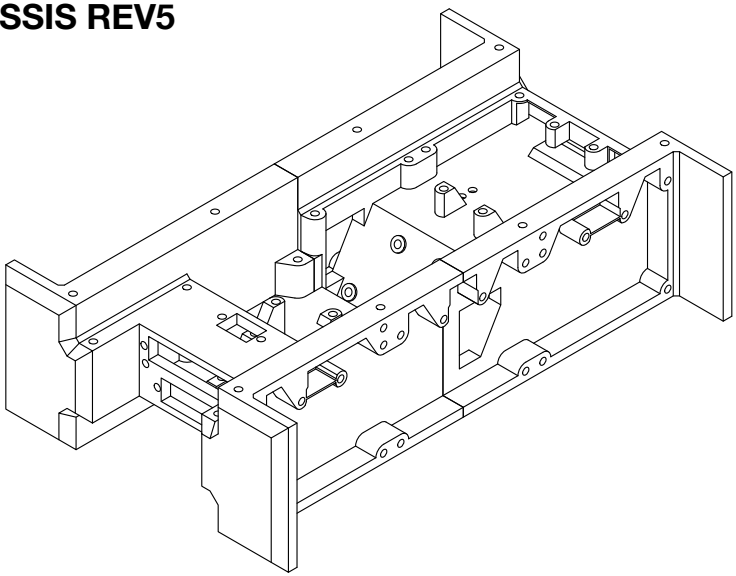


This is the advanced version of the single servo ultrasonic module. With the second servo, you can not only pan the sensor from left to right, but also tilt it. This makes it possible, for example, to search for stairs in front of the robot or to measure the head height above the robot.

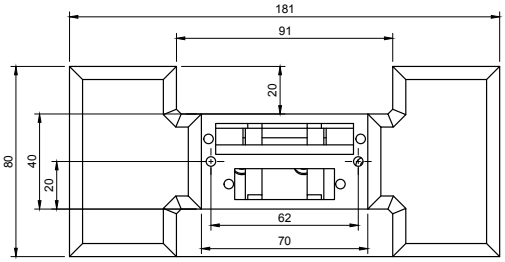


# TECHNICAL DRAWINGS

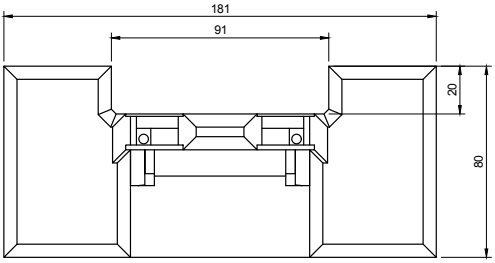
## CHASSIS REV5



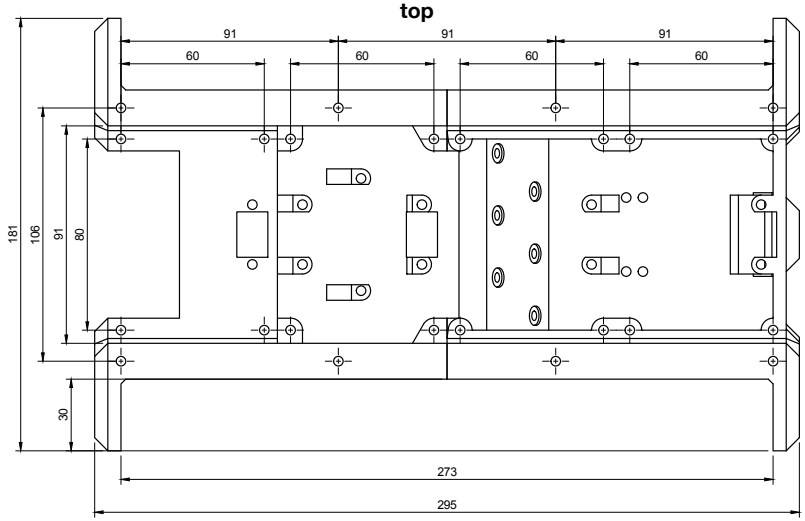
**front**



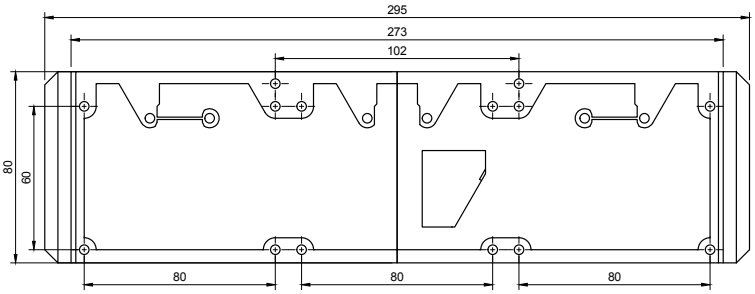

**back**



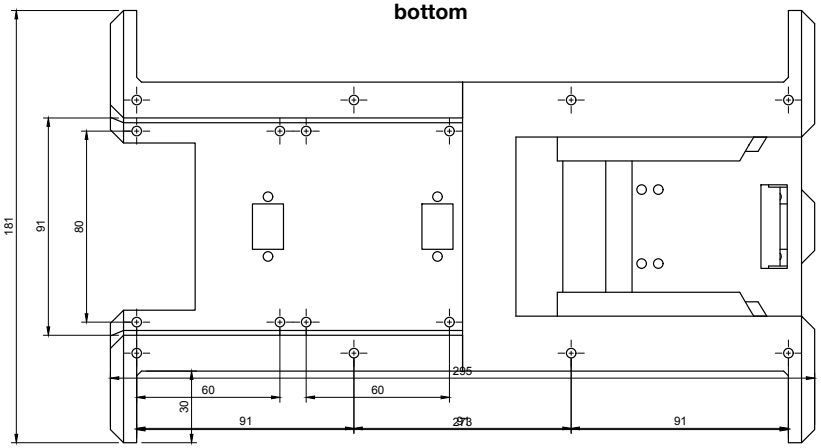
**top**



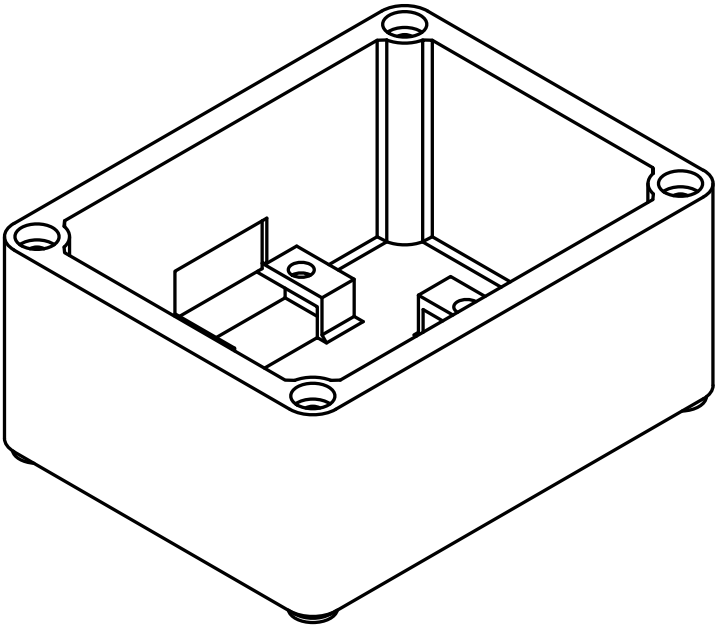

**side**



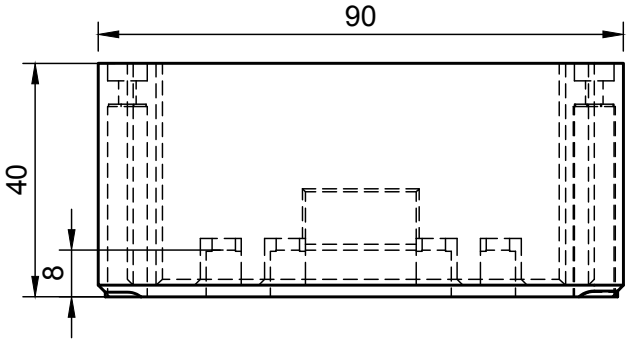

**bottom**



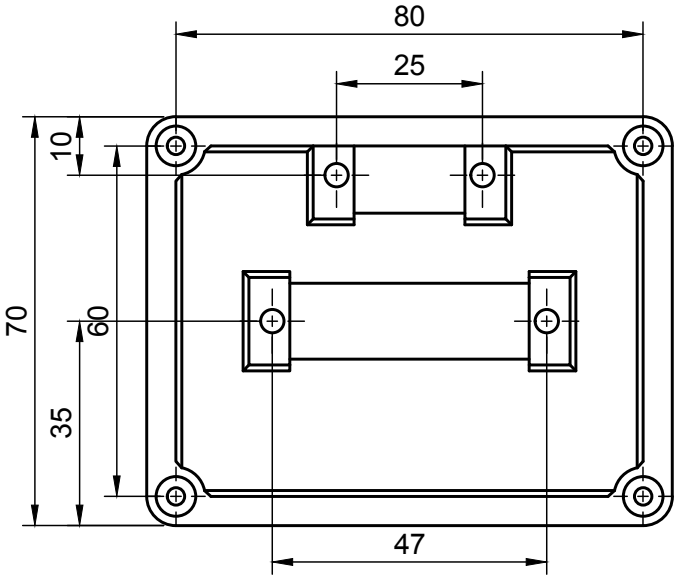
## CONTROLLER MODULE TEMPLATE



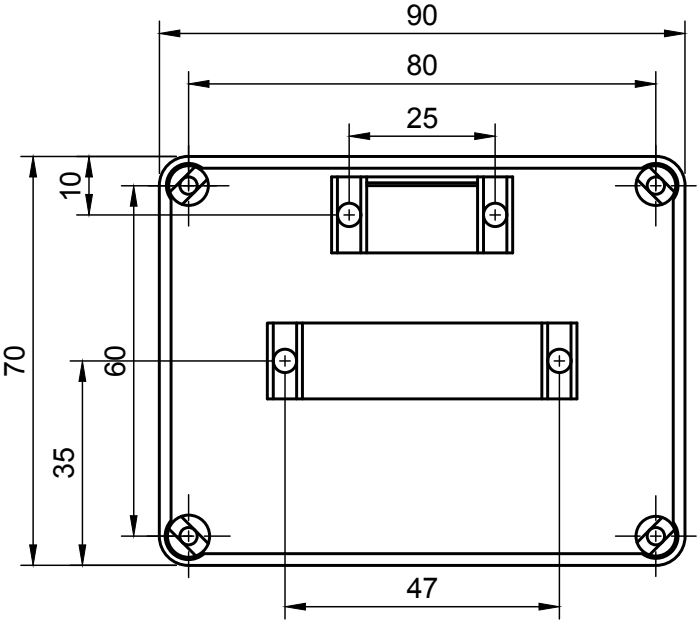
**front**



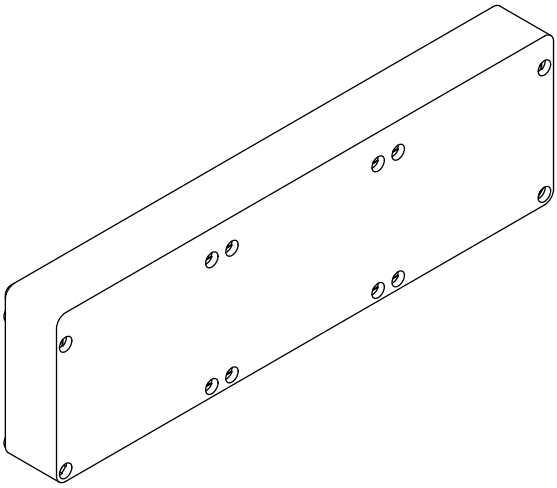
**top**



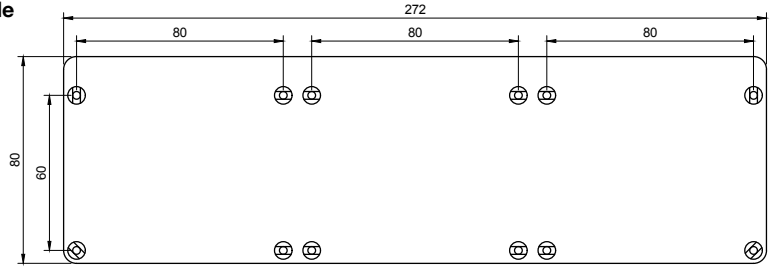

**bottom**



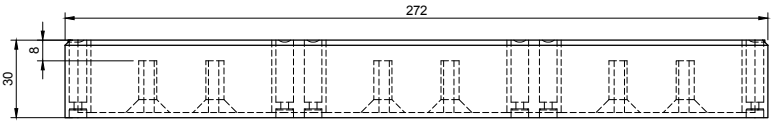
## TRIPLE MOTOR MODULE TEMPLATE



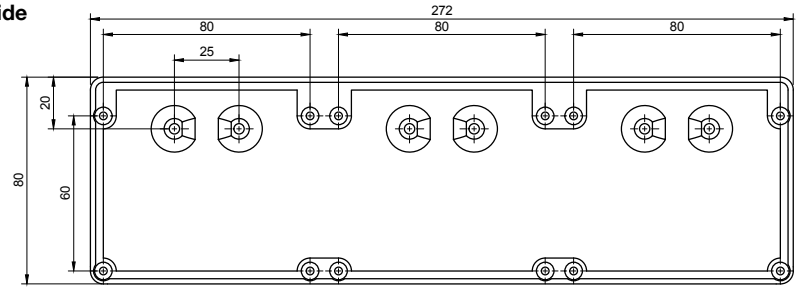
**outside**



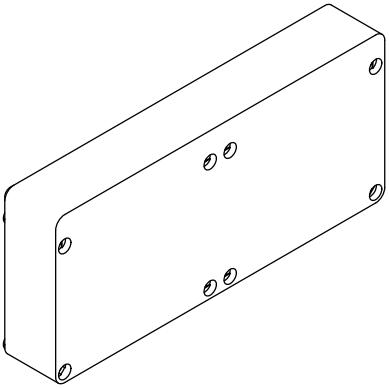

**top**



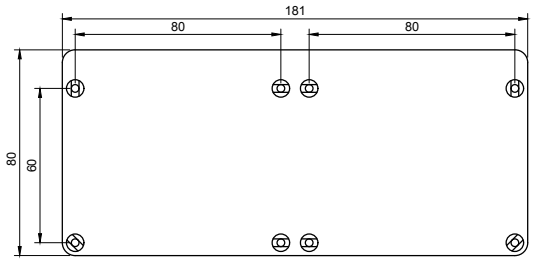

**inside**



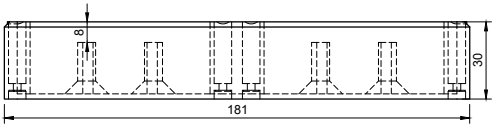
## DOUBLE MOTOR MODULE TEMPLATE



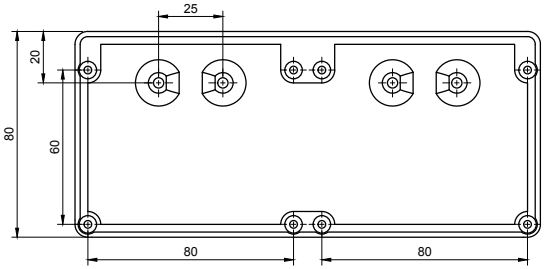
**outside**



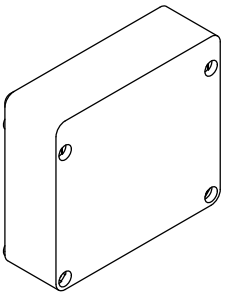

**top**



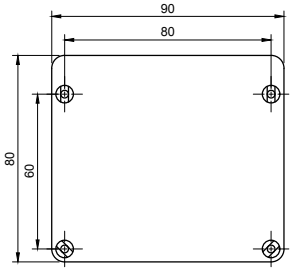

**inside**



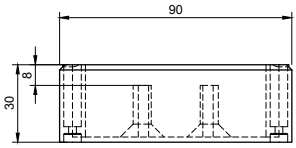
## SINGLE MOTOR MODULE TEMPLATE



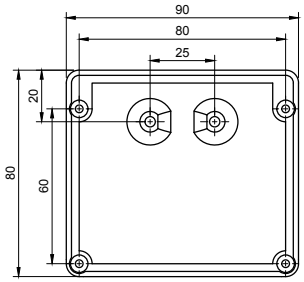
**outside**



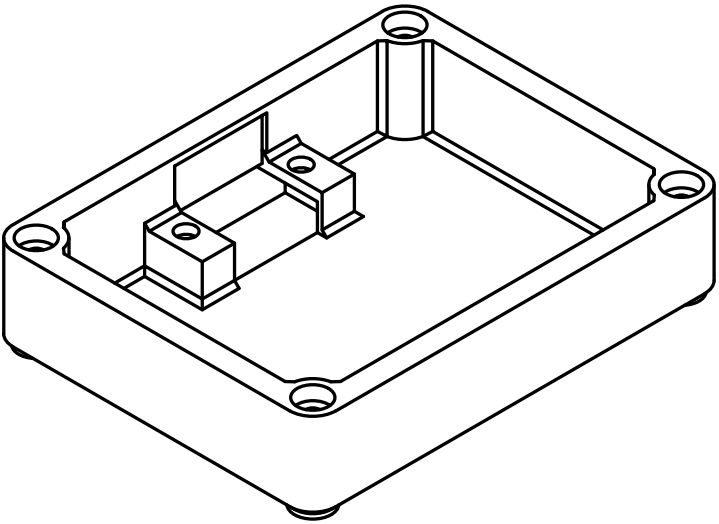

**top**

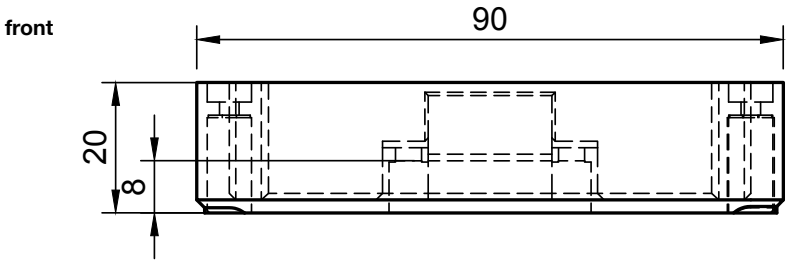


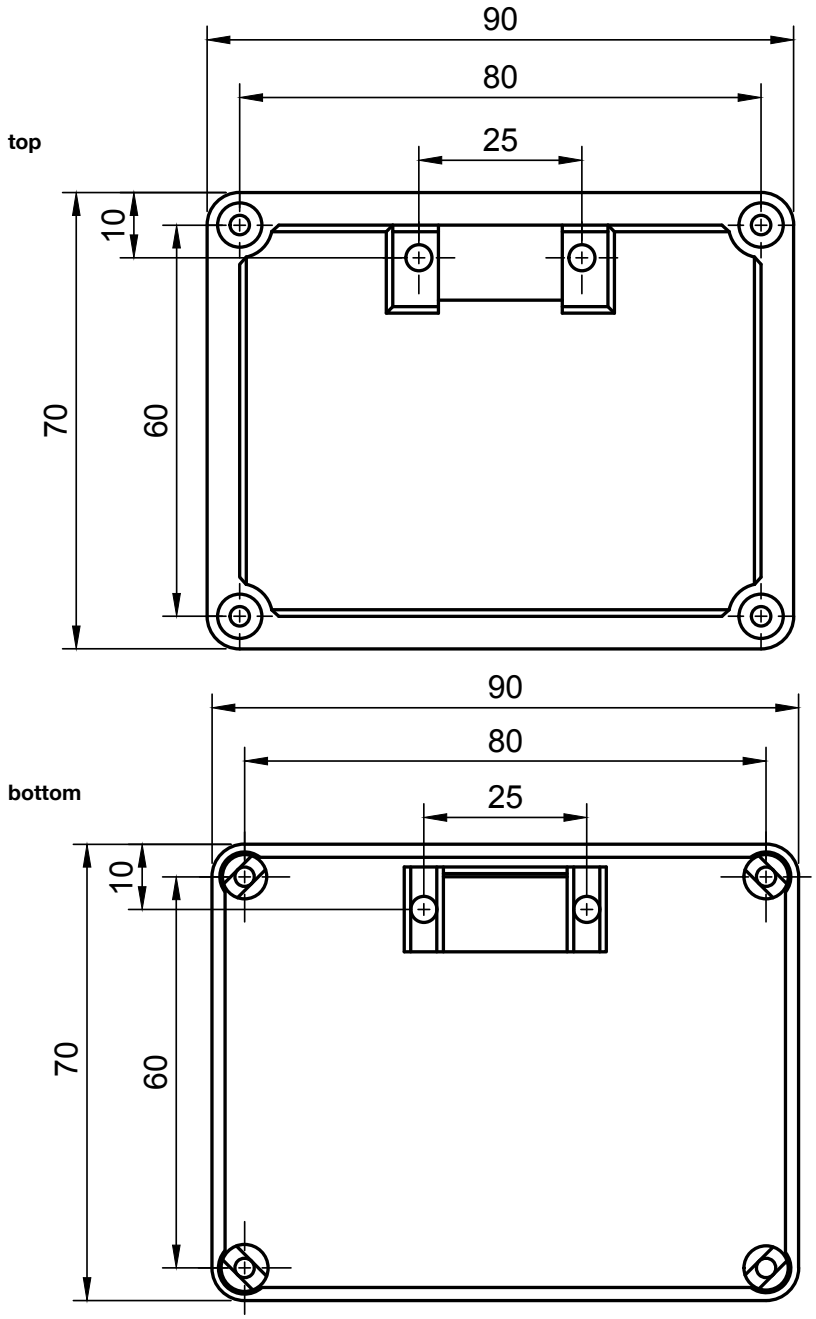

**inside**



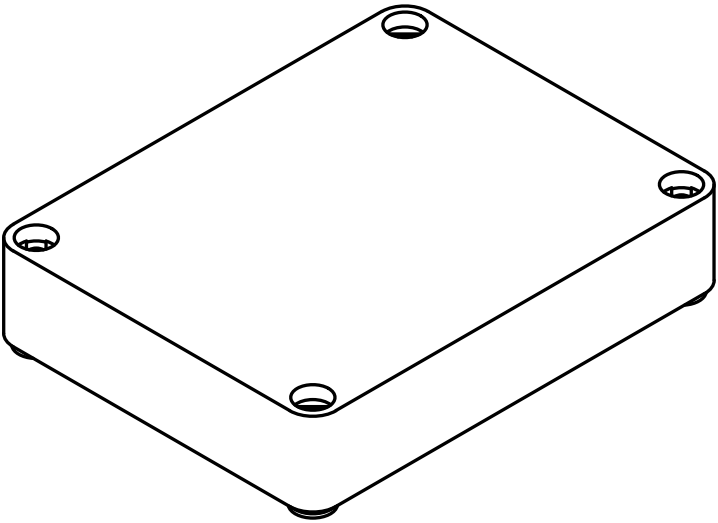
## UNIVERSAL MODULE TEMPLATE OPEN TOP

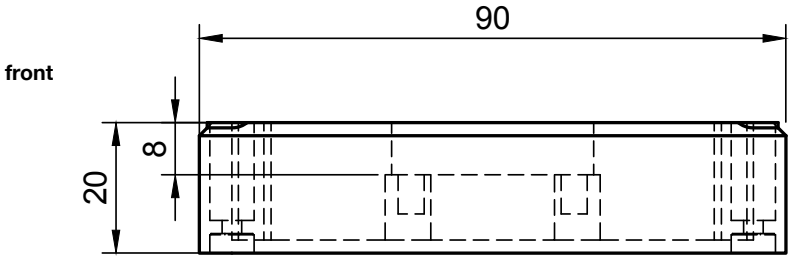


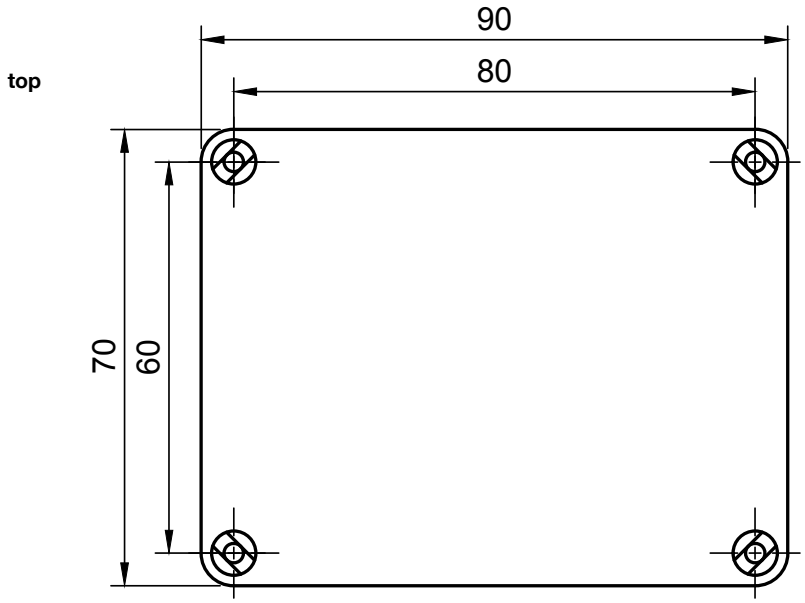


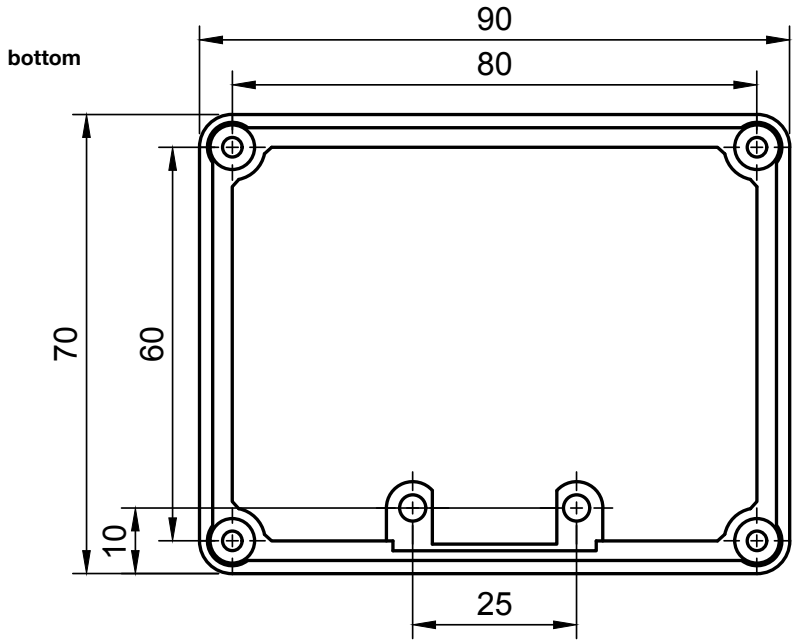

top
90
80
25
10
70
60
bottom
90
80
25
10
70
60

# UNIVERSAL MODULE TEMPLATE OPEN BOTTOM






top
90
80
70
60


bottom
90
80
70
60
10
25

# APPENDIX:

Wie bereits erwähnt, habe ich im Vorfeld meiner Diplomarbeit im Bereich Intelligente Eingebettete Systeme bei Benjamin Herwig als Hilfskraft gearbeitet. Dabei habe ich die dort eingesetzten Roboterfahrzeuge gewartet und verbessert. Im Folgenden habe ich Benjamin fünf Fragen zu den Vorlesungen gestellt:

1. Wie sind die Vorlesungen im Bereich Robotik und eingebettete Systeme aufgebaut?

Im Wesentlichen bieten wir zwei Vorlesungen mit selbstfahrenden Robotern an, die aber keinesfalls als Robotik bezeichnet werden können. Die Vorlesungen bestehen immer aus einem theoretischen und einem praktischen Teil. Im praktischen Teil werden die meisten der in der theoretischen Vorlesung behandelten Themen in die Praxis umgesetzt, wobei die Studierenden selbst in einen kreativen Prozess eintreten, der von Lehrenden und Hilfskräften begleitet wird.

2. Welche beispielhaften Aufgaben sollen die Studierenden mit den Robotern lösen?

Die zu bearbeitenden Aufgaben reichen von absoluten Grundlagen bis hin zur Ansteuerung komplexer Peripherieelektronik. Beispielsweise geht es anfangs nur darum, eine Leuchtdiode zum Blinken zu bringen, während später in einer Lehrveranstaltung auch Kamera- und Intertialmessdaten aufgenommen, übertragen und auf einem weiteren Rechnersystem ausgewertet werden. Die Studierenden sollen vor allem ein Gefühl dafür bekommen, wie die komplexe Prozesskette der Datenverarbeitung und auch der künstlichen Intelli-

genz von realen Umweltdaten, die ein Rechensystem umgeben, zu Wissen und daraus abgeleiteten Aktionen betrachtet werden kann.

3. Welche technischen Anforderungen ergeben sich daraus für die Roboter?

Die Roboter müssen einerseits technisch robust sein, gleichzeitig aber auch mit Techniken aufgebaut sein, die trotz aller Komplexität ein gutes bzw. schnelles Verständnis der Technik durch die Studierenden ermöglichen. Hier ist es z.B. wichtig, keine zu komplexen Mikrocontroller-Plattformen zu verwenden.

4. Gibt es auch didaktische Anforderungen, die die Lehre verbessern können?

Im Zusammenhang mit dem letzten Punkt ist es wichtig, dass die Technik einfach und effizient erklärt werden kann, so dass die Studierenden schnell zu ersten praktischen Erfolgserlebnissen geführt werden können, die die Studienmotivation verbessern.

5. Welche Funktionen und Möglichkeiten fehlen den bisherigen Robotern?

Insbesondere fehlt es an Robustheit und einfacher Wart- und Reparierbarkeit. Auch fehlt eine gewisse Modularisierung der verwendeten Roboterkomponenten, die nicht alle in jeder angebotenen Lehrveranstaltung eingesetzt werden (können).

# NOTE OF THANKS AND DISCLAIMER

I would like to thank my professor, supervisor and first examiner prof. Oliver Vogt and my second examiner Olaf Val for making this diploma possible. Furthermore I would like to thank my parents Marie and Alf for the love and support in my adventures.

I certify that all components of this work were performed between April and July 2024 by Nils Emil Altmann. I used DeepL write for correction purposes.

Kassel _____