ROBOCUPJUNIOR RESCUE LINE 2024

TEAM DESCRIPTION PAPER

ZTKa BROS 3000

General information for this template

- This template contains a suggested structure for your Team Description Paper. Please look into the official rubrics posted on the official RoboCup website and the community website to see which areas of your TDP will be scored.
- This document is supposed to be between 5 and 10 **pages** long (from Abstract to Conclusion). Please **keep the formatting** (font size and type, margins, line spacing, etc.) and number figures and tables.
- Use diagrams, flow charts, etc. throughout this document to better **illustrate your work**
- Submit the TDP as a **PDF file**.

Abstract

- For RoboCupJunior Rescue Line we prepared a robot that can follow the line, avoid obstacles, climb ramps and save victims in the evacuation zone. Our robot uses some very innovative methods and complex algorithms for line following and victim rescue.

1. Introduction

a. Team

- Our team contains two family and team members. Oldest member is Maksim who has lot of experience in robotic competition while Oskar is relatively new in this category but he has love for 3D design and printing. Maksim's role is to define and program robot functionality and Oskar's role is to design mechanical and functional components of robot and print on 3D printer. Oskar is 6th grade of Primary school Dubovac in Karlovac and Maksim is 1st grade of First Private Gymnasium in Zagreb Croatia. Both of them are regular members at community of technical culture Karlovac which provides them knowledge and place to develop something cool.

2. Project Planning

a. Overall Project Plan

- Our main goal for this competition is to test our robot and see did we manage to build something what can be competition to others. Master plan for building custom robot is to use previous experience form Fischertehnik components and all lessons learned on other competitions to build custom made robot which will have ability to cross and solve any track.
- Milestones:
    a. Design undercarriage with optimal size
    b. Test and define which microcontroller will be used
    c. Test and define types of sensors needed for resolving different sets of problems
    d. Design robotic arm for ball catching
    e. Learn image processing algorithms and build program for camera
    f. Build v1 of robot and define points for improvement
    g. Build v2 of robot and test robot stability on Seesaws
    h. Build v3 of robot and test line following with all obstacles

i. Build v4 and test robot in Evacuation Zone and Victims rescue

- As we have experience with Fischertehnik robot and different competitions we have good vision what robot needs to do and what are all downsides of Fischertechnik system. Idea is to all that experience transfer to now custom robot.

b. Integration Plan

- Trough components testing we concluded that I2C and UART communication will provide necessary communication channel for our components with enough throughput and speed. For main board we choose ProS3 ESP32-S3 microcontroller board as provides more than enough processing and memory capabilities for our project even if we change plan to expand our robot with few more components regardless initial plan. Second part in project where we used lot of time to make good analyze and plan is power supply as this is crucial part for robot functionality. Main voltages that we will be using is 9V, 5V and 3.3V and design covers around 5A of max continuous output. While defining types and amount of sensor used in this project main concern was will we will have.

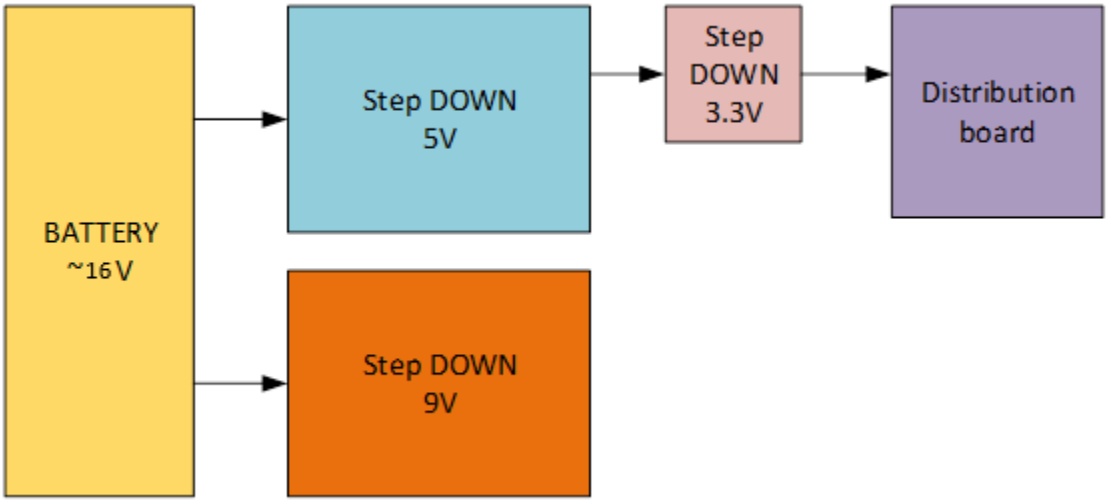

Figure 1 : Power supply diagram

- As mentioned before main communication will be over I2C protocol and for that we used I2C multiplexer. With this approach we overcome three most common problems:
  a. pin limitation
  b. different voltage per component
  c. duplicate address when used 2 or more of the same component

- As Multiplexer board can be connected in chain it is possible to have up to 64 components at one I2C address which provide us lot of space for additional sensors.

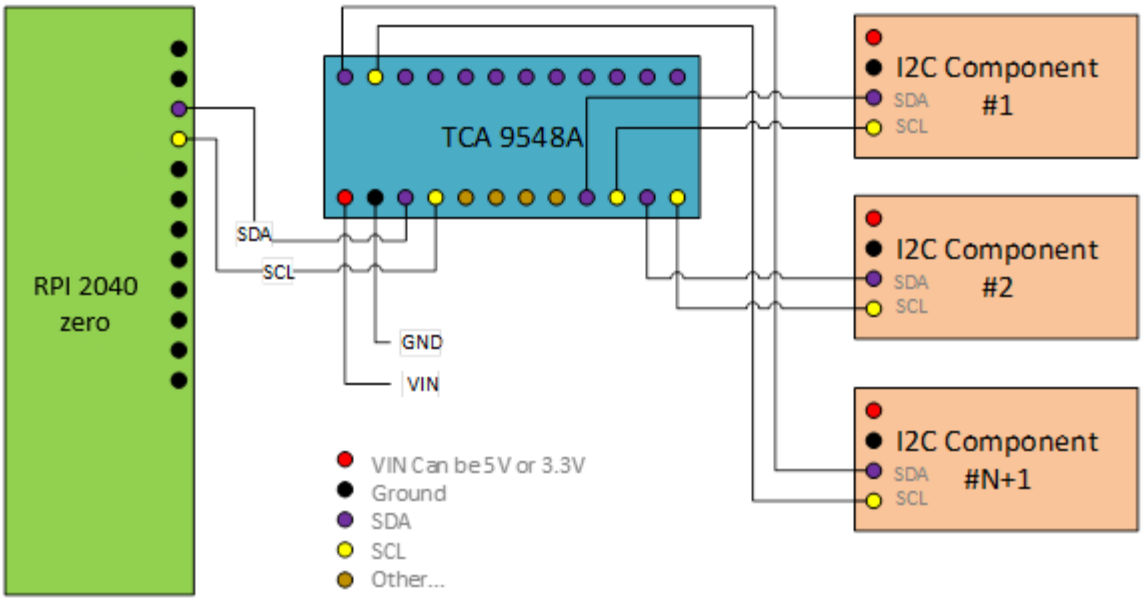

Figure 2 : Multiplexer communication diagram

- For communication between camera's and camera controller and main controller UART is used as a protocol of two-way communication. Two-way communication is needed so that camera can send current ball position so that ESP32 microcontroller can control DC motors and servos to move robot and arm near ball so that arm can catch it.

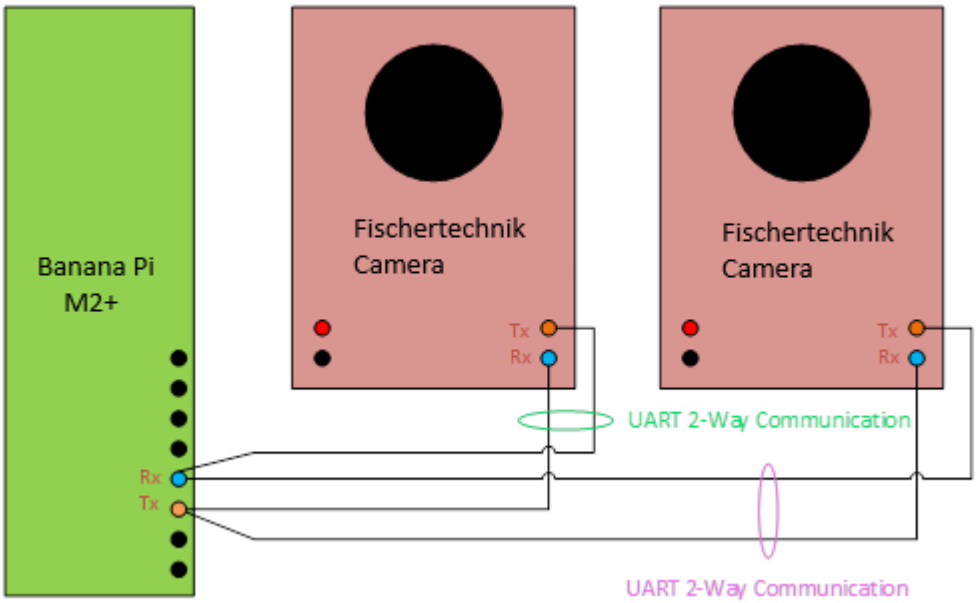

Figure 3 : UART communication diagram

3. Hardware

- List of components and quantity used for custom robot development

Category	Type	Quantity	Communication
Microcontroller	UM ProS3 ESP32-S3	1	I2C, UART, Analog

Sensor	Primoroni VL53L5CX 8x8 Time of Flight	2	I2C
Sensor	Pololu VL53L0X Time-of-Flight	2	I2C
Controller	Adafruit 16-Channel 12-bit PWM/Servo Driver	1	I2C
Controller	BDC motor controller, 4x3.6A	1	PWM
Sensor	Adafruit 9-DOF Absolute Orientation IMU BNO055	1	I2C
Multiplexer	TCA9548A I2C Multiplexer	2	I2C
Camera controller	Banana Pi M2+	1	UART
Camera	Fischertechnik SN259 Camera	2	USB/UART
Motor	DC motor 1:100	4	-
Motor	Servo motor HS-40	5	-
Motor	Servo motor RD-5606HB-300	1	-
Battery	Li-Ion 3.6V	4	-
Power	5V, 15A Step-Down Voltage Regulator	1	-
Power	3.3V, 6.5A Step-Down Voltage Regulator	1	-
Power	Buck Converter with LED display 1.25 - 37 V	1	-
Other	Distribution board for power and I2C	1	-

- At start of project we used IR sensor array for line following but in process of development that approach is not used any more as we switched to line following using camera.

a. Mechanical Design and Manufacturing

- When we started discussion around robot look, one thing over which we agreed that will be “car like design” (undercarriage with four wheels) as we used that design in Fischertehnik components and was always useful. As we are learning about CAD applications and using 3D printer in our Community of technical culture Karlovac logical approach was to design and print mechanical parts by ourselves. For undercarriage design we used for starting point idea is undercarriage which we won on internal online competition at RoboCupJunior Croatia in time of Covid lockdown. As we already have lots of mechanical parts from

Fischertehnik, on undercarriage we design same montage slots so that we can use Fischertehnik parts for fast testing and prototyping. We spend lot of time watching YouTube videos to analyze ideas from other custom robot's people building around world.

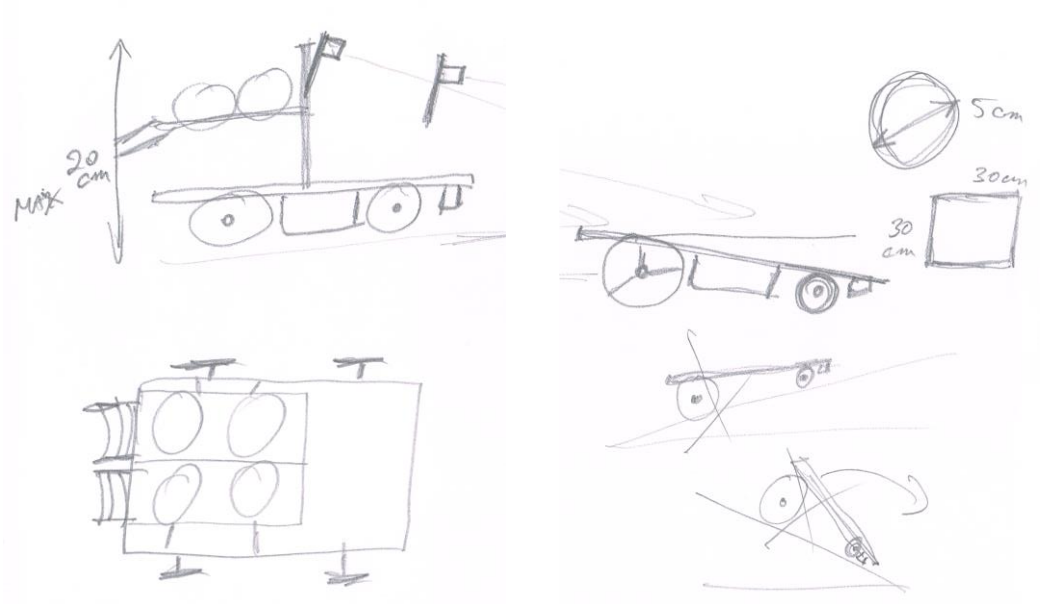

Figure 4 : Some of first drawing while idea discussion

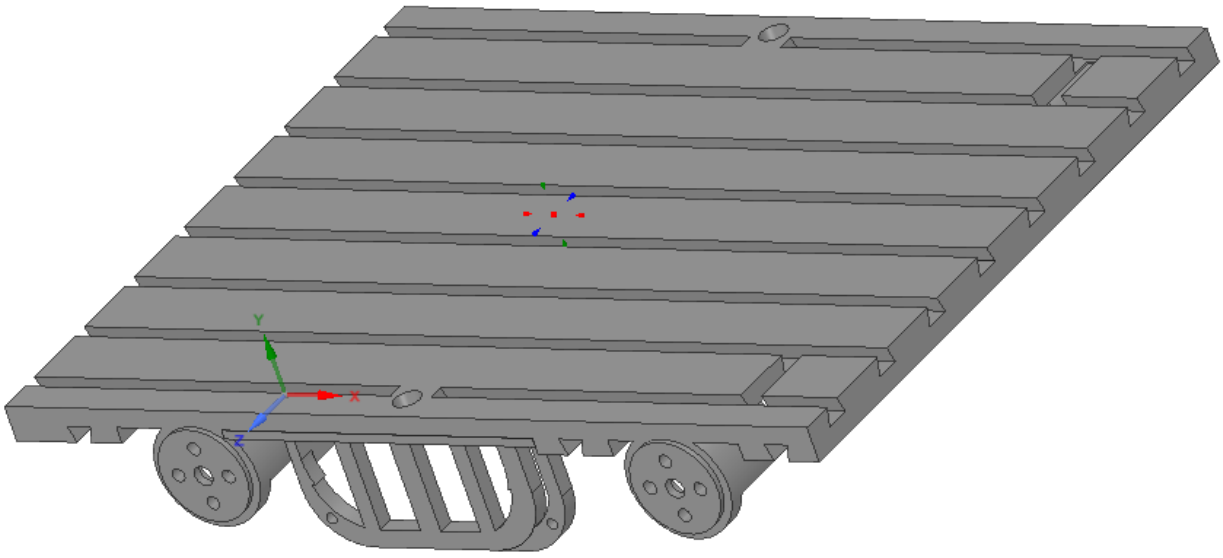

Figure 5 : Initial undercarriage design in CAD

- Second part that we agreed is that robot needs to have movable arm that can catch ball and put it down in collector bin (we agreed but without any idea how to achieve this even in theory 🤣).

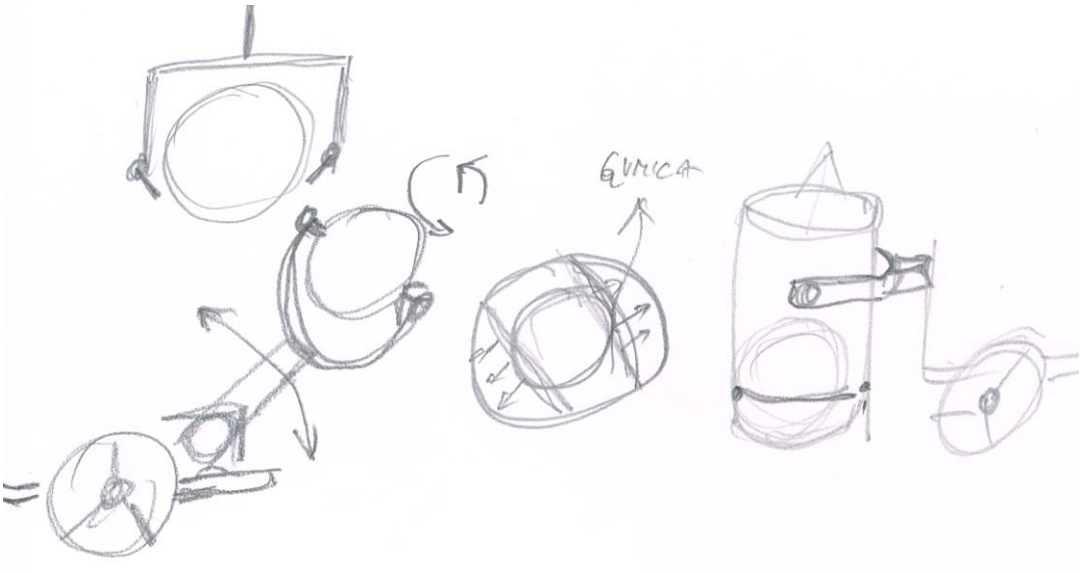

Figure 5 : One of ideas for ball catch mechanism

- Testing procedures for mechanical and design part is simple… design, try to print, fix issues which preventing successful print. Check is printed part fitting as planned is it thick and hard enough to hold load or pressure without breaking if not update design and print again. Some parts are done from 2-3 attempts and some needed 20-30 attempts like ball catching mechanism.

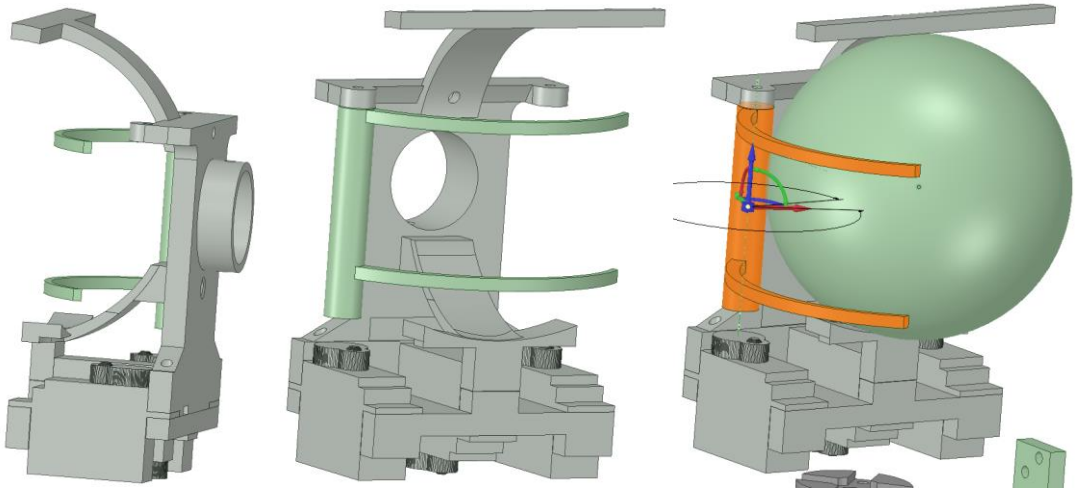

Figure 6 : Some of views of final for ball catch mechanism

- **Catching and sorting mechanism with camera is our innovation which we are very proud of because we spend lot of time and energy to accomplish this compact design with good functionality.**
- Main structure is undercarriage with DC motor montage section
- Power is distributed like noted in Figure 1 diagram while actuators are basically four DC motors with direct attached wheels (4x4 transmission)

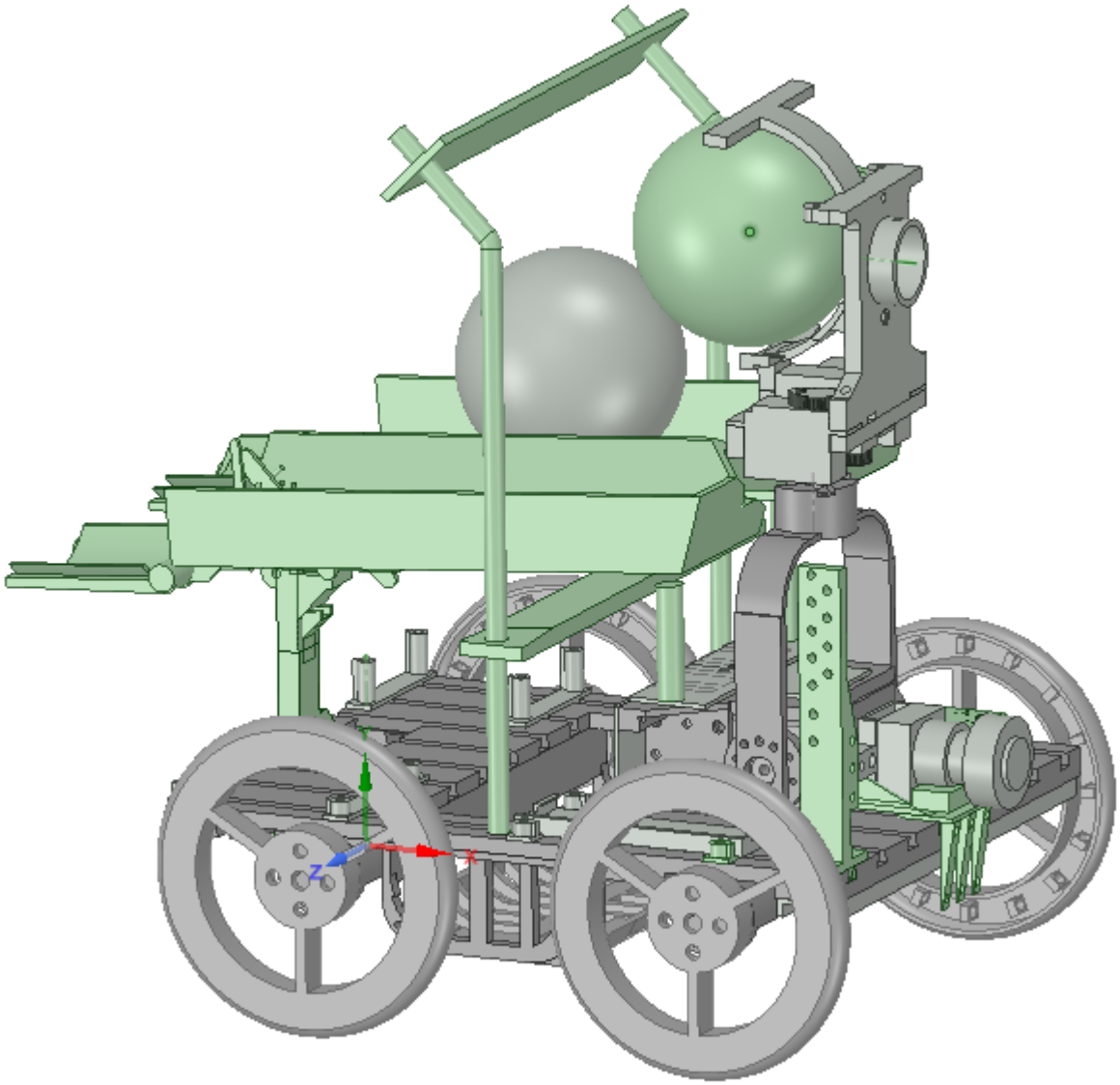

Figure 7 : Custom robot all sections visible view

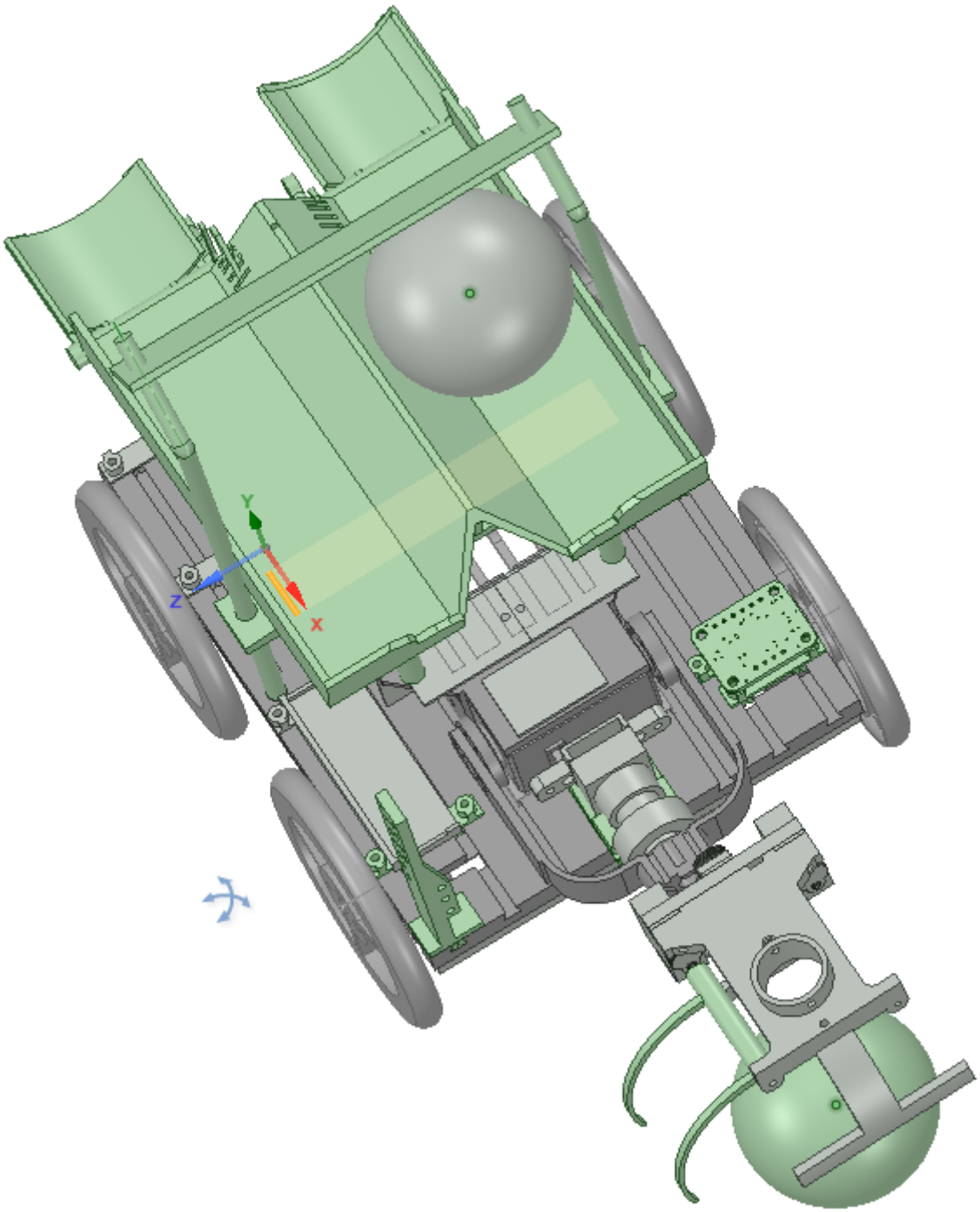

Figure 8 : Custom robot view from up

b. Electronic Design and Manufacturing

- As experienced competitor in robotic using Fischertechnik solutions we build list of problems which robot needs to solve. For each problem we make list which sensor can be used and what are advantages and disadvantages of that sensor.
- When we started with custom robot development the main idea is to use as much as possible commercially available components so that we are flexible about design change and that we build robot which can be rebuild by anyone.

- One of unusual sensor types that we use is time-of-flight with 8x8 matrix (VL53L5CX). Benefit of this type of distance sensor is ability to “visualize” object using algorithm and values from 64 points. We found that this sensor is useful even in evacuation zone to find out robot orientation and angle on wall or evacuation point.
- For main controller we choose ESP32 microcontroller and UM ProS3 board which has next features:
    - Dual 32bit Xtensa LX7 cores @ up to 240Mhz
    - RISC-V Ultra Low Power Co-processor
    - 2.4GHz Wifi - 802.11b/g/n
    - Bluetooth 5, BLE + Mesh
    - 16MB QSPI Flash
    - 8MB of extra QSPI PSRAM
    - 2x 700mA 3.3V LDO Regulators
    - LDO2 is user controlled & auto-shuts down in deep-sleep
    - Low power RGB LED
    - ULTRA LOW Deep Sleep Current
    - USB-C Connector with back-feed protection
    - USB ESD protection
    - Native USB + USB Serial JTAG
    - LiPo Battery Charging + PicoBlade connector
    - VBAT and 5V Sense Pins
    - 3D High Gain Antenna
    - STEMMA QT connector powered by LDO1
    - 27x GPIO including castellated headers
    - JTAG pins on the header
    - TinyPICO/ProS3 compatibility

4. Software

a. General software architecture

- One aspect of our custom robot is using fine tune and optimized program. Due the fact that I have experience in competitive programing trough different competitions in informatics I used all best practices and algorithms for robot program development.
- Programming language used is C++ as that language provides better execution and processing speeds. For IDE we used Arduino IDE with official libraries for ESP32 microcontroller.
- All Important program components and sensors are separated into custom made libraries with dependent classes and functions for easy reuse.

```
#include <MSS-Camera.h>
#include <MSS-PID.h>
#include <MSS-Motors.h>
#include <MSS-LineSensor16.h>
#include <MSS-EventTracker.h>
#include <MSS-TOF_8x8.h>
#include <MSS-RotationSensor.h>
#include <MSS-ObstacleAvoider.h>
#include <MSS-CameraLineSensor.h>
#include <MSS-TOF.h>
#include <Wire.h>
#include <MSS-Arm.h>

#define RESET_BUTTON_PIN  34

using namespace Settings;
```

- Camera programming was done using python and OpenCV which was easy to understand as I have experience in full feature Python programing. Main focus on camera programming is understand image processing and object tracking using build-in algorithms.
- List of used algorithms:

a. Convex hull & BFS – for ball detection using time-of-flight (No longer used)
b. Yolov8 object detection model – for detecting victims in evacuation zone

b. Innovative solutions

- Training a custom yolov8 model with our own data.
- We have spent a lot of time trying to get our model to work properly and tweaking hyperparameters so we get the best possible performance.

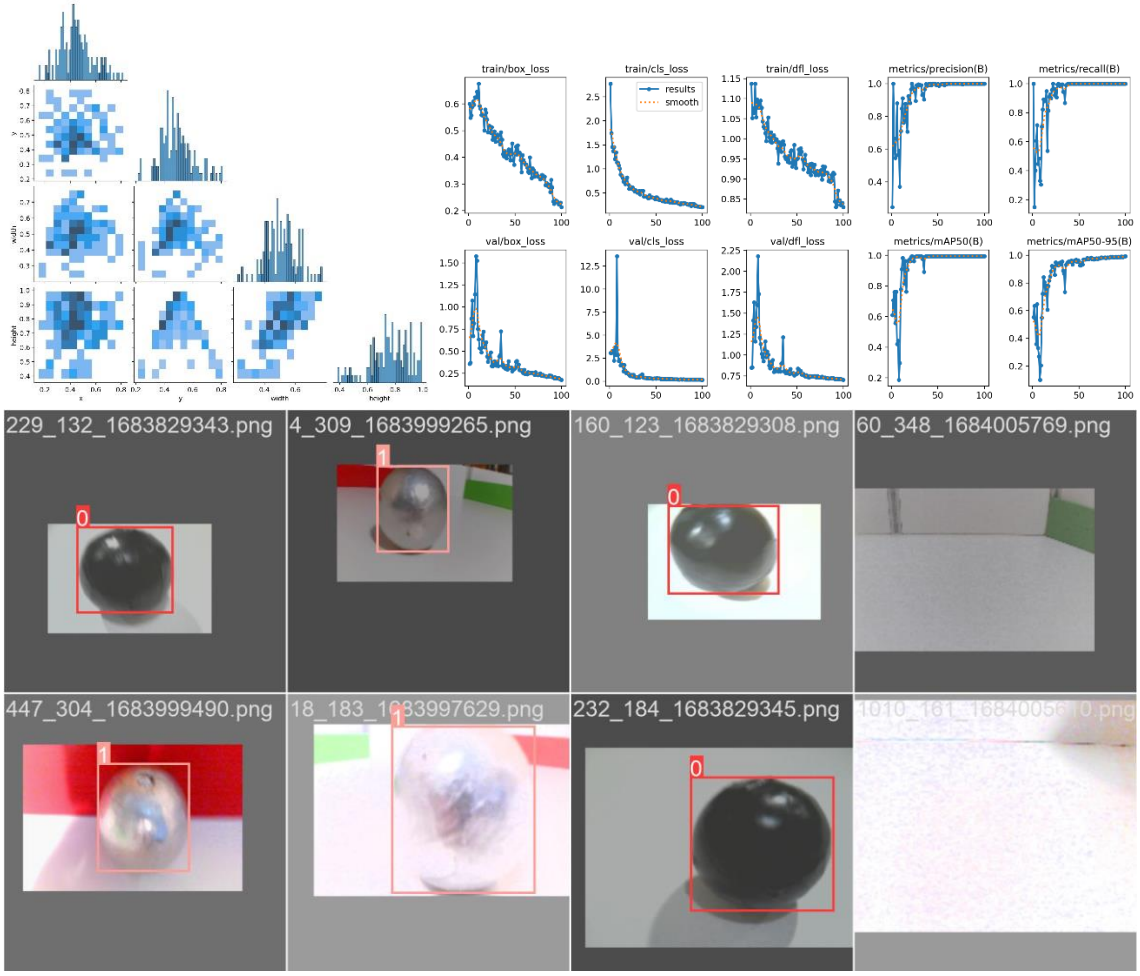

Figure 9 : Model performance diagrams and an example of a batch for model training

5. Performance evaluation

- Robot performance evaluation is done by good old run and log all errors for code fix. Fix bugs and run again, and again, and again…
- Yolov8 neural network uses DFL and CloU loss algorithms for bounding box regression which accurately shows performance of our model with each epoch.
- Part of evaluation is to invent tasks and obstacles that are not by guides of RoboCupJunior just to see how robot will solve that problem.

6. Conclusion

- As most of our journey of custom robot development is generally covered by this document one of things that can’t be transferred on paper is euphoria when you have good idea for some problem which you trying solve for days. Also, frustration when theory can’t be done in practical part of project and with even greater joy, happiness and proud when robot solves something without any error.

Appendix (optional)

- None

References

- None