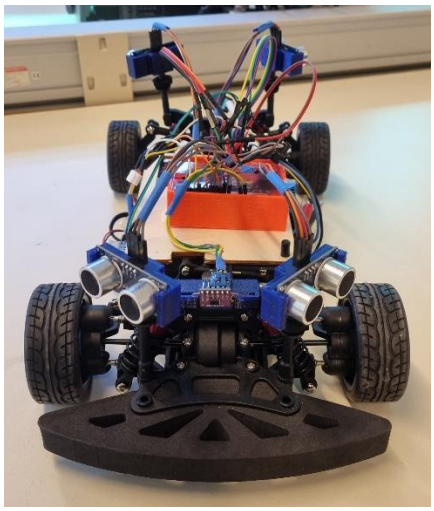
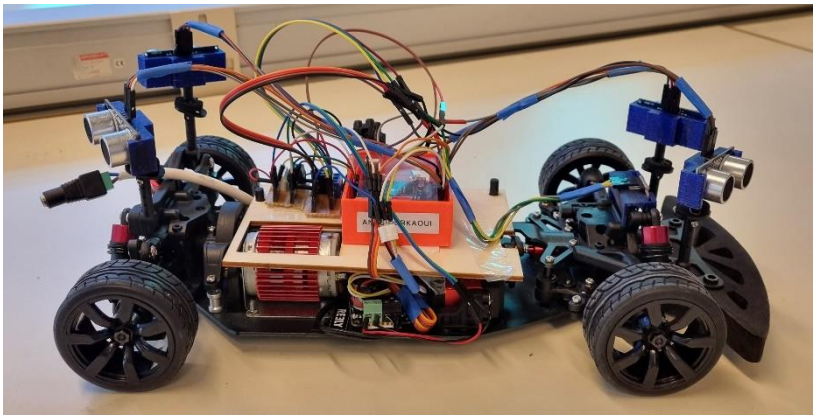
AutoRCX





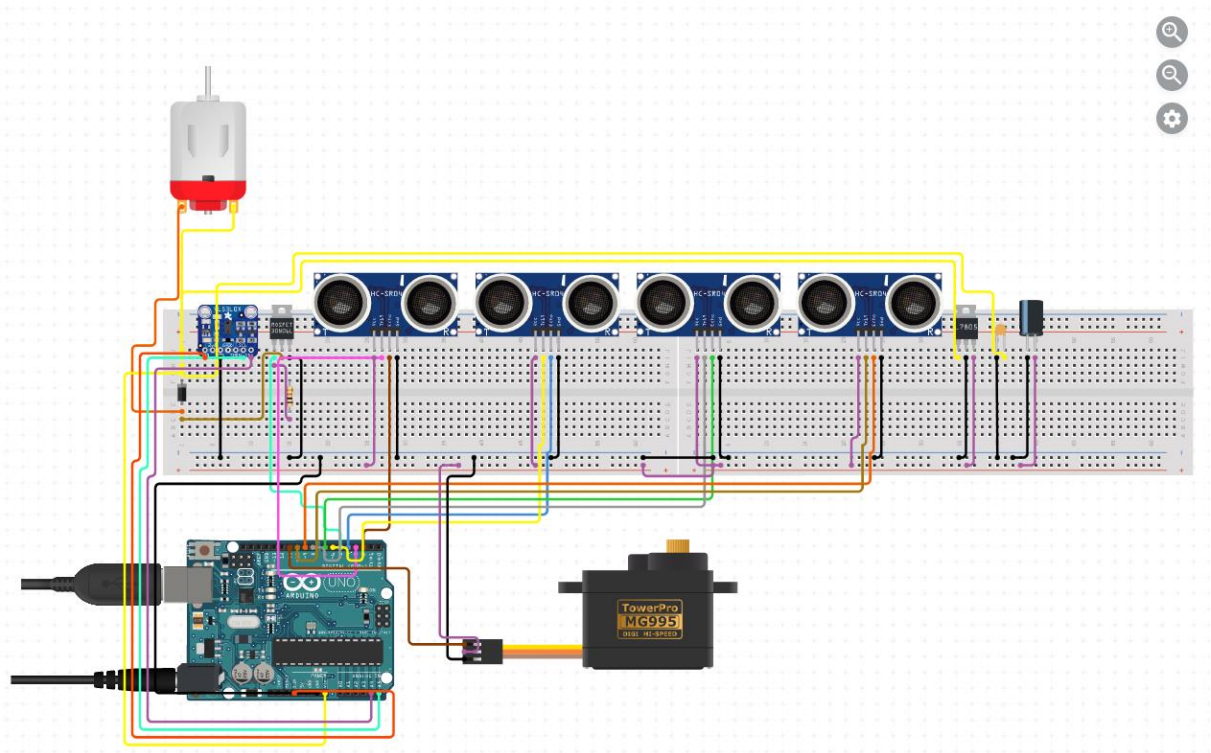
Introduction:

AutoRCX is an exciting project aimed at developing an autonomous RC car capable of navigating towards a destination while avoiding obstacles. By leveraging advanced technologies such as the Nvidia Jetson Nano, ultrasonic sensors, an Arduino UNO card, a laser sensor, and a camera, AutoRCX aims to demonstrate the potential of autonomous driving in a scaled-down remote-controlled car.

The primary objective of AutoRCX is to create a fully autonomous RC car that can operate independently, making informed decisions based on sensor data and providing a seamless and immersive experience. With the integration of powerful computational capabilities offered by the Nvidia Jetson Nano board, the car can process sensor inputs, analyze the environment, and execute intelligent maneuvers.

AutoRCX incorporates various cutting-edge components to achieve its goals. Ultrasonic sensors provide the car with proximity data, enabling it to detect obstacles in its path. The laser sensor enhances the precision of distance measurements, contributing to accurate obstacle avoidance. The 720p camera enables real-time image capture for object recognition and lane tracking, empowering the car with vision-based decision-making capabilities.

Electrical diagram:



Scope statement:

Objectives :

- Develop an autonomous RC car capable of operating without direct human intervention.
- Enable the car to navigate towards a specified destination by processing sensor data and making informed decisions.
- Implement obstacle detection and collision avoidance mechanisms using ultrasonic sensors and a laser sensor.
- Utilize a 720p camera to capture real-time images for object recognition and lane tracking.
- Integrate the Nvidia Jetson Nano board for powerful computing and image processing capabilities.
- Implement GPS and compass functionality for accurate location determination and orientation.
- Create a feedback mechanism to provide real-time updates on the car's location, progress, and actions.
- Document the entire development process, including hardware setup, software implementation, and testing procedures.

Estimated time spent on the project:

- Time on-campus: 144h
- Time off-campus: 56h
- Total estimated time: 200h
- Cost as an engineer (starting with a gross annual salary of 38 000€ for 1600 hours of work): 4750 €

Budget:

- Nvidia Jetson Nano board: 227€
- Ultrasonic sensors: 24€
- Laser sensor: 5€
- Camera : 32,46€
- DC motor controller : 12,80€
- GPS receiver : 16€
- Compass : 13,90€
- Battery : 40€
- Arduino UNO: 29,40€
- Power converter : 17€
- RPLIDAR : 113€
- Car frame : 120€
- DC motor : 19€
- Wheels and tires : 18€

⇨ Total : 687,56€

Risk and Challenges :

- Integration and compatibility issues between hardware components and software libraries.
- Ensuring reliable and accurate sensor data processing for precise decision-making.
- Optimizing the machine learning model for object recognition and minimizing latency.
- Passing information from Nvidia Jetson Nano card to the Arduino UNO card and eventually to the car components.
- Balancing power consumption and battery life for extended operation.
- Dealing with unpredictable environmental factors that may affect sensor accuracy.

Algorithmic functioning:

The algorithmic functioning of the AutoRCX involves a series of steps and decision-making processes to enable the car to navigate towards a destination while avoiding obstacles. Here's a high-level overview of the algorithm:

1. Initialize :

- Power on the system and initialize all components.
- Calibrate sensors (e.g., compass) for accurate readings.

2. Read Sensor Data :

- Continuously read data from ultrasonic sensors, laser sensor, camera, GPS receiver, and compass.

3. Object Detection and Tracking :

- Process camera data using a machine learning model to detect objects and identify their positions relative to the car.
- Track detected objects over time to determine their movement patterns.

4. Path Planning :

- Determine the optimal path to the destination based on the current location, detected objects, and available road information.
- Consider factors such as obstacle avoidance, traffic rules, and road conditions.

5. Obstacle Detection and Collision Avoidance :

- Use ultrasonic sensors and laser sensor data to detect obstacles in the car's path.
- Analyze the sensor readings and adjust the car's trajectory to avoid collisions.

6. Localization and Orientation :

- Utilize GPS data to determine the car's current location.
- Use compass data to determine the car's orientation with respect to the destination.

7. Motor Control :

- Send control signals to the motor controllers to adjust the car's movements accordingly.

8. Feedback and Monitoring :

- Provide real-time updates to the user interface about the car's location, progress, and actions.
- Continuously monitor sensor data and system performance for any anomalies or critical situations.

9. Control using convolutional neural network model (Nvidia Jetson Nano board):

- Data Collection:

Data was collected using the driving simulator provided by Udacity. The simulator was run in the recording mode to capture camera images of the vehicle and the associated driving data, including steering angle, throttle, and speed. The data was saved in a CSV file containing the image paths and corresponding measurements.

- Data Preprocessing:

The collected images were converted to the appropriate format and preprocessed to fit the requirements of the CNN model.

Image preprocessing included pixel value normalization, cropping to remove irrelevant parts of the image, and resizing to the input dimensions expected by the model.

```
# Example code for image preprocessing

img = cv2.imread(os.path.join(DATA_IMG, log_content[index][i]).replace(" ", ""))

img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

# Apply preprocessing operations
```

- Model Construction:

A CNN model was built using the Keras library.

The model was designed with multiple convolutional layers to extract features from the input images.

Max pooling layers and fully connected layers were added to reduce dimensionality and perform classification on the images.

Example code for model construction

model = Sequential()

model.add(Lambda(lambda x: (x / 127.5) - 1., input_shape=(160, 320, 3)))

Add other layers to the model

- Model Training:

The model was trained on the collected data using a supervised learning procedure.

The data was split into training and validation sets to evaluate the model's performance.

Training was performed over multiple epochs, adjusting the model's weights to minimize loss and improve predictions.

- Autonomous Control Implementation:

A feedback loop controller was implemented to regulate the steering angle and acceleration of the vehicle.

The controller utilizes the model's predictions for steering angle and adjusts acceleration based on the current speed.

Encountered problems during the making of the project:

- There was not much space for the DC motor to be fitted properly, we had to cut a piece of the frame of the RC car.
- In order to hold the sensors in place, we 3D printed some holders instead of duct taping the sensors to the car frame.
- The Nvidia Jetson Nano having not too much storage capacity, we had to use a separate SD card to install ROS and all needed packages.
- We used a laser cut structure of Plexiglass to hold the Arduino UNO card and Nvidia Jetson Nano in place.
- The Nvidia Jetson Nano was bricked, and it needed to be reflashed.
- Compatibility with the Simulator: Ensuring compatibility and proper communication between the model, the driving script, and the simulator was tedious.
- Data Preprocessing: Image preprocessing was crucial to adapt the input data to the model's requirements.

- Model Training: Training the model posed challenges due to the complexity of the neural network and the volume of data to process.

Conclusion:

The AutoRCX project has reached its successful conclusion, resulting in the development of an autonomous car utilizing the Arduino Uno card. Throughout the project, significant progress has been made, demonstrating the potential of autonomous systems and robotics. Although there is more work to be done, the achievements so far lay a solid foundation for future enhancements.

Moving forward, the integration of the Nvidia card and the implementation of the Robot Operating System (ROS) will open up new possibilities for the AutoRCX project. The Nvidia card's computational power will enhance the car's capabilities, enabling more sophisticated sensor fusion, advanced object detection using the lidar and camera, and higher-level decision-making algorithms.

Furthermore, the design of a custom body shelter for the car will enhance its aesthetics and provide protection for the internal components. Careful consideration should be given to factors such as durability and ease of maintenance during the design process.

In conclusion, the AutoRCX project has achieved its primary objectives by successfully building an autonomous car using the Arduino Uno card. The future integration of the Nvidia card, lidar, camera, and ROS will unlock even greater potential for advanced autonomous functionality. The utilization of modules such as Bluetooth, GPS, and compass will further enhance the car's capabilities. With continued dedication and exploration, AutoRCX has the potential to make significant contributions to the field of autonomous systems and robotics.

Bibliography:

https://stackoverflow.com/questions/66554561/modulenotfounderror-no-module-named-rospkg

https://robotics.stackexchange.com/questions/24180/multiple-ros-installation-on-single-machine

https://www.circuito.io/static/reply/index.html?solutionId=645d439c368a2f002ecc33f1&solutionPath=storage.circuito.io

https://github.com/dusty-nv/jetson-inference#hello-ai-world

https://phoenixnap.com/kb/linux-format-disk

https://askubuntu.com/questions/21321/move-home-folder-to-second-drive

http://wiki.ros.org/melodic/Installation/Ubuntu

http://wiki.ros.org/fr/ROS/Tutorials/InstallingandConfiguringROSEnvironment

http://wiki.ros.org/ROS/Tutorials/UnderstandingNodes

https://developer.nvidia.com/embedded/learn/jetson-ai-certification-programs

https://dev.to/ivanorsolic/a-self-driving-rc-car-and-a-complete-guide-to-build-your-own-23el