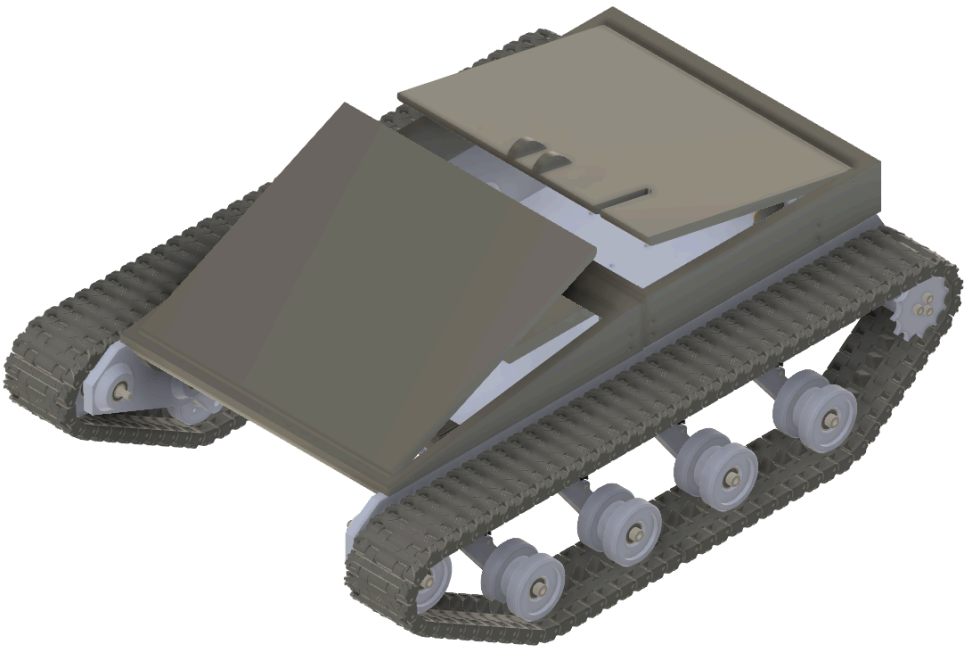
JIF

JONAS NICKLAS, ILIAN ODENBACH, UND FELIX SCHREIBER

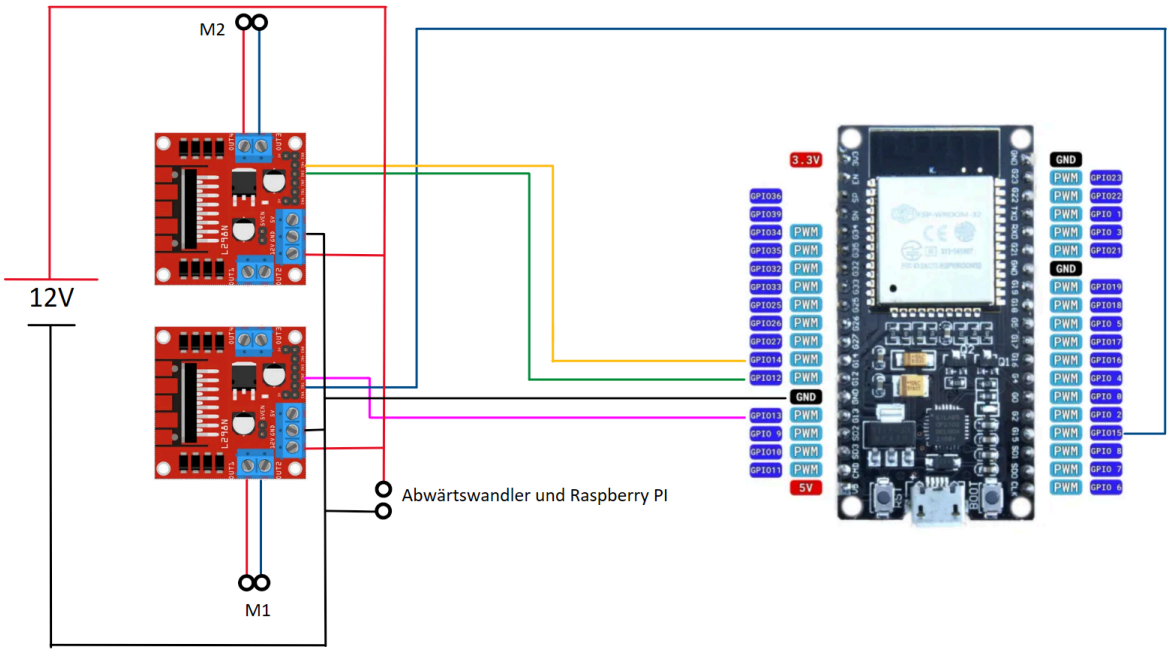
Inhaltsverzeichnis

1. Roboter

1.1. **Fahrgestell und Karosserie.** Das Fahrgestell ist nicht von uns designt worden, eben so wenig wie der Kettenantrieb. Dies war auch nicht notwendig, da Dejan von HowToMechatronics eine wunderbare Basis für unser Vorhaben bereits geschaffen hatte. [1] Auf dieser Basis wurde eine Karosserie für unsere Zwecke in Fusion360 erstellt mit zwei aufklappbaren Deckeln, die die Wartung an dem Roboter erleichtern. Um die elektronischen Komponenten auf dem Fahrgestell zu befestigen, wurden Adapterstücke konstruiert, da die Löcher des Fahrgestells für andere Komponenten designt wurden. Ebenso wurden stärkere Federn in den Dämpfern eingebaut die 10 Newton in etwa maximal Belastung haben. Der Schlitz auf der Oberseite des Hinterendeckels ist für die Kabelführung der Webcam.



1.2. **Hardware.** Der Roboter wird von zwei 37mm 12V DC-Motoren mit 66rpm betrieben, welche angesteuert werden über zwei L298N H-Brücken, wobei jeder Motor mit einem der Motoren verkabelt ist. Die Steuerung der H-Brücken hat ein ESP32 development board, welches seriell mit einem Raspberry Pi 3B+ verbunden ist, inne. Die Stromversorgung übernimmt ein 12V 5000mAh LiPo-Akku, der die Motortreiber direkt versorgt, parallel dazu geschaltet ist ein Abwärtswandler, welcher die 12V auf 5V konvertiert und so den Raspberry Pi mit Strom versorgt. Man stellt sich sicherlich die Frage warum zwei H-Brücken verbaut wurden, wenn man an einem 2 Motoren anschließen kann, aber eine H-Brücke kann maximal 2 Ampere abgeben, was sich dann auch auf beide Motoren aufteilen würde was zu einem Leistungsverlust führt, da jedem Motor nur 1 Ampere zugewiesen werden kann. Um diese Problematik zu lösen wurden zwei H-Brücken verbaut und so können jedem Motor 2 Ampere zugewiesen werden. Leider ist aber immer noch ein Spannungsabfall zu bemerken. So wird ein zukünftiger Schritt sein ein Motortreiber zu verbauen, welcher eine noch höhere Stromstärke stemmen kann.



1.3. **micro-ROS.** Um den ESP32 vom ROS2-Netzwerk aus anzusteuern, wurde micro-ROS verwendet. Dazu wurde zuerst ein neues Workspace erstellt, welches für das Flashen und dem micro-ROS-Agent zuständig sein wird. Anschließend noch ein src Ordner in dem micro-ROS Workspace erstellen, wo dann das micro_ros_setup Package hinein geklont wurde. [2] Nachdem ersten Build ist nun ein firmware Ordner zu sehen, in dem Verzeichnis ,"…/microros_ws/firmware/freertos_apps/apps", wurde dann die ros_esp32cam_diffdrive App geklont. [3] Nach dem Konfigurieren und dem Builden konnte der ESP32 auch schon geflasht werden. Nun fehlte nur noch der micro-ROS-Agent. Nach dessen Builden und Starten konnte man unter den Topics auch unser /cmd_vel finden. Somit war der erste Teil der Basis geschaffen.

Quellen

1. howtomechatronics: Fully: 3D Printed Tank- Tracked Robot Platform, https://howtomechatronics.com/projects/fully-3d-printed-tank-tracked-robot-platform/
2. micro-ROS: First micro-ROS Application on FreeRTOS, https://micro.ros.org/docs/tutorials/core/first_application_rtos/freertos
3. Reinbert: ros_esp32cam_diffdrive, https://github.com/Reinbert/ros_esp32cam_diffdrive