Robi-Design

tieStAPistics

Software Vorteile unseres Roboters

- Kann langsam schneller fahren (auch während Kurvenfahrten)
- Kann sachte langsamer werden (auch während Kurvenfahrten)
- Kann Kurven fahren
- Erkennt wenn er irgendwo anstößt („touch“)
- Mehrere Bausteine können zusammen zu einer Fahrt verbunden werden
- Alle Fahrstrecken geben wir in „cm“ an. Den Raddurchmesser und PI etc. speichern wir in einem Baustein den wir „Konstanten“ nennen. Wenn wir andere Reifendurchmesser verwenden würden, müssen wir nur an einer Stelle eine Änderung machen, da alle Werte in „cm“ sind.
- Alle Winkel geben wir in Grad bezogen auf die Startposition an. Das heißt wir haben z.B. eine Nulllinie, eine 90° Linie, eine -12° Linie etc. Und wir können den Roboter zu jedem Zeitpunkt auf diesen virtuellen Linien fahren lassen. Die Linien beziehen sich auf den Tisch und ändern sich nicht wenn sich der Robi dreht.
- Zum Fahren verwenden wir den Gyrosensor
- Für alle eigenen Bausteine haben wir ein „Test“-Programm erstellt, um Fehler zu finden und den Baustein zu verbessern.
- Außerdem schreiben wir in einer Datei alle wichtigen Werte mit, auch um Fehler zu finden.
- Wir haben alle Aufgaben in einem EV3-Programm
- … und durch eine „Benutzeroberfläche“ kann man verschiedene Unterprogramme auswählen
- Wir können Fahrbewegungen (Punkte) in Dateien aufzeichnen, anzeigen lassen und später wieder abfahren lassen. Für die FLL-Aufgaben verwenden wir das aber nicht.

Hardware Vorteile unseres Roboters

- Wir verwenden die größtmöglichen LEGO Reifen, da wir damit besonders schnell fahren können
- Er ist relativ schwer und die Reifen drehen sich deswegen nicht durch
- Man kommt relativ einfach an den Akkuladestecker
- Es gibt eine Schutz-Abdeckung
- Wir verwenden 4 große Motoren, da diese am meisten Leistung haben
- Wir können an 3 Seiten Aufsätze befestigen und mit einem Verschlussmechanismus sichern
- Wir wollten eigentlich auch die Fahrmotoren für einen Kettenantrieb verwenden (hat aber nicht wirklich funktioniert). Deswegen gibt es auf der Unterseite auch Adapter
- Wir haben vorne zwei Kugeln, um auch schwerere Aufsätze zu verwenden
- Die Startrampe ist sehr wichtig, um den Gyrosensor exakt auszurichten

Vorteile unserer Aufsätze

- Obwohl wir eigentlich dachten, die Aufgabe mit Fluchtgeschwindigkeit wäre besonders schwer, hatten wir diese Aufgabe als erstes gelöst. Wir verwenden 6 Federn, die wir in einem Gehäuse zusammenhalten. Die Lösung mit den Federn ist sehr Modell schonend, da nicht unnötig viel Kraft auf die Matte abgegeben wird.
- Für die Raumfahrt verwenden wir eine Kette wie ein Fliesband um die Fahrzeuge auf die Rampe zu stellen und den Trigger auszulösen.
- Die anderen Aufsätze konnten recht einfach bleiben, da wir mit der Software sehr weiche und flüssige Fahrbewegungen programmiert haben.
- Wir haben unseren Roboter und die Aufsätze mit dem (Lego Digital Designer – LDD) digitalisiert und haben somit eine Aufbauanleitung falls etwas kaputt gehen sollte.

Was wir nicht hinbekommen haben

- Wir hatten sehr viele Versuche um über die Krater zu fahren, da wir dachten dass das eine einfache Aufgabe sein könnte. Aber leider hat nichts so funktioniert wie wir es gedacht hatten. Einen Rover über den Krater zu werfen, hat zwar Prinzipiell funktioniert, aber wir haben die Aufgabe dann doch weggelassen.

Warum verwenden wir nur den Gyro Sensor?

Letztes Jahr haben wir viel mehr Sensoren verwendet. Allerdings gab es während des Wettbewerbs Probleme mit dem **Ultraschallsensor**. Wir vermuten, dass das damit zusammenhängt, dass auch andere Teams den Ultraschallsensor verwenden, und deswegen die Sensoren sich gegenseitig stören. Oftmals waren die Abstände ungenau.

Beim **Farbsensor** kommt es sehr darauf an, wie man Ihn verwendet. Es gibt zwei Modi: Farben erkennen und schwarz/weiß erkennen. Zum Linienerkennen war der schwarz/weiß Modus besser, aber um den Roboter nach einer Linie auszurichten, braucht es im Verhältnis zum Gyrosensor relativ viel Zeit.

Den **Touchsensor** benötigt eine Kraft aus genau einer Richtung. Wir sehen keinen sinnvollen Verwendungszweck.

Aber neben dem Gyro Sensor lesen wir oft auch die **Motoren** aus. Die Werte, die die Motoren liefern, verwenden wir in unseren Berechnungen, wie weit denn der Roboter schon gefahren ist. Wir können so z.B. auch deinen „touch“ erkennen.

Wir haben sehr viel Entwicklungsarbeit verwendet, den **Gyrosensor** möglichst bei allen Fahrbewegungen zu verwenden. Der Gyrosensor hat vor allem beim Einschalten einige Probleme, wenn beim Roboter starten am Sensor gewackelt wird. Im Internet gibt es einige Tricks um per Software einen Reset des Sensors auszulösen. Mit unserem Sensor (relativ neu) hat keine Lösung geholfen. Das Problem ist, dass der Gyrosensor manchmal kontinuierlich hoch zählt, obwohl der Roboter nicht bewegt wird. Die einzige Lösung ist, den Stecker kurz zu ziehen, oder den Roboter neu zu booten. Zur Sicherheit haben wir vor jedem Start eines Programmes eine „Detection“ eingebaut, ob der Sensor gerade hoch zählt. Das System bleibt dann stehen, und Piepst („düt düt düt ...“).