M20 „Programmierung eines Roboters (NXT)“

Aufgaben

0. Leiten Sie während der Vorbereitung die Formeln zur Berechnung der nicht gegebenen Größen in Abb. 4 und in Abb. 5 her.

1. Schreiben Sie ein Programm um,
   a) den NXT geradeaus fahren zu lassen und vor einem Hindernis zu stoppen;
   b) Zufallszahlen *x*, *y* sowie das Ergebnis *x*·*y* auf dem LCD-Display des NXT auszugeben;
   c) den NXT eine Sinuskurve fahren zu lassen.

2. Vervollständigen Sie den „Lückenquelltext“, um den NXT ein Einparkmanöver vollziehen zu lassen.

Zusatz: Programmieren sie (a) einen „tanzenden“ Roboter, (b) einen Roboter, der immer fährt und versucht, Hindernissen auszuweichen oder (c) entwerfen Sie ein Programm nach eigenen Vorstellungen.

Literatur

Bei Interesse LabVIEW-Handbuch oder LabVIEW-Demo herunterladen und ansehen.

Zubehör

LEGO© NXT, PC mit der Software LabVIEW for LEGO MINDSTORMS 2010

Schwerpunkte zur Vorbereitung

- Variablentypen in Programmiersprachen (integer, boolean, ...)
- Logische Funktionen (and, or, ...)
- Schleifen und andere Strukturkontrollen in Programmiersprachen (if, if ... then, if ... then ... else, goto, for, while, ...)
- LabVIEW

Allgemein

In der Experimentalphysik trifft man sehr häufig auf die Notwendigkeit, Daten elektronisch zu erfassen und die Steuerung von Experimenten zu automatisieren. So werden beispielsweise in der Robotik – in loser Verbindung zur Mechanik – komplexe Bewegungsabläufe automatisiert. Zum Handwerkszeug des/r Experimentalphysikers/in sollte daher eine grundlegende Kenntnis von Programmiersprachen und Schnittstelleneigenschaften gehören. In diesem Versuch sollen spielerisch erste Erfahrungen mit der Programmierung von Bewegungsabläufen vermittelt werden. Dazu wird ein Lego® NXT-Modul verwendet, in dem über ein Firmware-Upgrade die Möglichkeit geschaffen wurde, komplexe Programme unter der Verwendung der Programmiersprache LabVIEW zu schreiben.
Beim Lego® NXT handelt es sich um einen Mikrocontroller-gesteuerten Roboterbaukasten. Zu diesem gehören drei Schrittmotoren, zwei Tastsensoren (Schalter), ein RGB-Farbsensor sowie ein Ultraschall Sensor. Der im Praktikum genutzte Aufbau (siehe Abb. 1) ist ein einachsiger Fahrzeugroboter mit zwei angetriebenen Rädern und einem Nachlaufrad. Die zentrale Steuereinheit enthält ein LCD-Display, eine USB Schnittstelle sowie drei Outputs und vier Inputs.

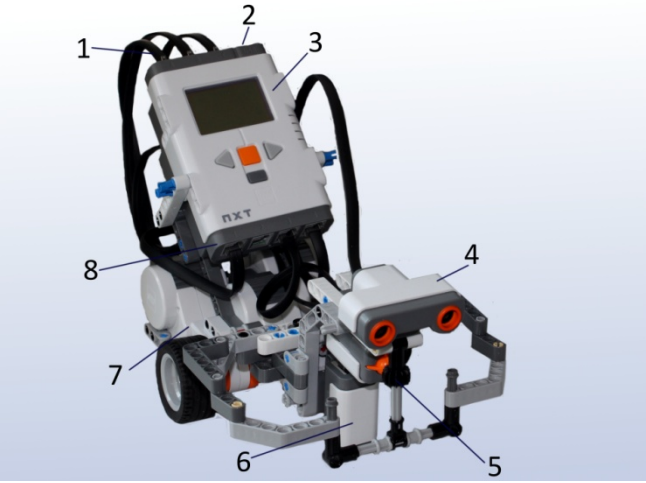


Abb. 1 NXT „Tribot“
1 Outputs A, B, C
2 USB-Schnittstelle
3 NXT Modul mit LCD-Display und Bedienelementen
4 Ultraschall Sensor
5 Berührungssensor (Tastschalter)
6 RGB-Farbsensor
7 Schrittmotor
8 Inputs 1, 2, 3, 4

LabVIEW (Laboratory Virtual Instrument Engineering Workbench)

Bei LabVIEW handelt es sich um eine Programmierumgebung, mit der komplexe Programme zur Steuerung von Geräten und zur Erfassung von Messwerten effizient erstellt werden können. Dank einer großen Auswahl vorgefertigter Funktionen und eines gut durchdachten Schnittstellenmanagement vereinfacht LabVIEW die Arbeit des Programmierers erheblich.
Die Umgebung eines VI´s (Virtual Instrument) von LabVIEW besteht aus zwei Teilen: dem sogenannten Frontpanel und dem dazugehörigen Blockdiagramm. Auf dem Frontpanel wird die Bedienoberfläche des Programms mit Ein- und Ausgabemöglichkeiten erstellt, während im Blockdiagramm das auszuführende Programm grafisch programmiert wird.
Zur anschaulichen Erläuterung verwenden wir das einfache Programm in Abb. 2. Auf dem Frontpanel sind die Eingabeobjekte A, B, OK und Stopp sowie das Ausgabeobjekt C angeordnet. Im Blockdiagramm sind die Ein- und Ausgabeelemente auf Variable (Terminals) abgebildet. Für das auszuführende Programm fungieren die Eingabeterminals dabei als Datenquellen (Daten werden eingelesen) und das Ausgabeterminal als Datensenke (Daten werden ausgegeben).
Die Datentypen werden, wie im Blockdiagramm ersichtlich, farblich unterschiedlich markiert. So sind der Typ Boolean grün (┈┈), der Typ Integer blau (——), der Typ Double orange (——) und der Typ String pink (〰〰) gekennzeichnet. Diese Grundtypen unterscheiden sich nicht nur in der Art der Daten, sondern auch in ihrer Länge in Bits. Dabei werden die Zahlenformate Integer und Double von LabVIEW noch weiter unterteilt. Arrays eines Datentyps werden durch dickere Drähte angezeigt (z.B. ═══ 1D-Array Double).

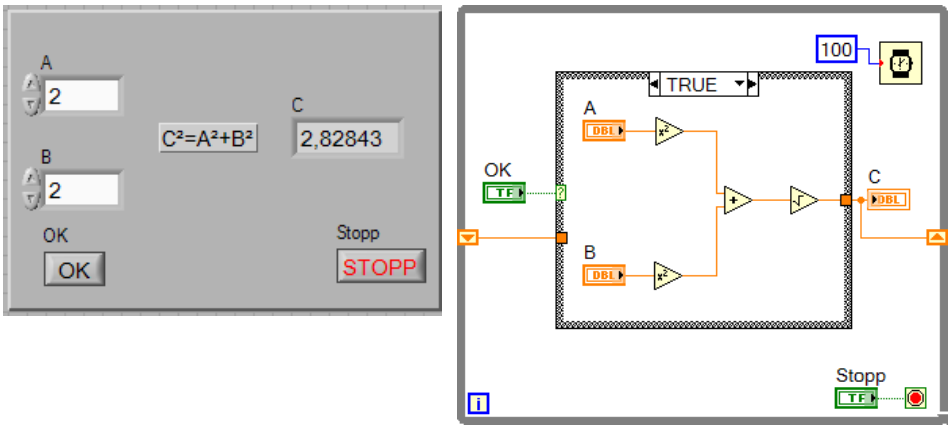
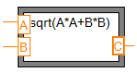


Abb. 2 Frontpannel und Blockdiagramm in LabVIEW

Um Operationen mit Daten durchzuführen, werden diese in sogenannte Knoten (z.B.) zusammengeführt. Diese Knoten dienen der Verarbeitung von Daten. Zum einen müssen Werte an einen Knoten übergeben werden, zum anderen geben Knoten Werte entsprechend der Funktion zurück. Als Knoten kann auch ein eigenes VI (Sub-VI) dienen. Daraus ergibt sich der Vorteil, dass Programmteile, die an mehreren Stellen des Hauptprogramms ausgeführt werden sollen, nur einmal geschrieben werden müssen. Um ein eigenes VI mit Daten-Ein- und Ausgängen zu belegen, müssen die Ein- und Ausgangsobjekte des Frontpanels mit den Anschlüssen des VI´s verknüpft werden. Die Auswahl zur Anzeige der Anschlüsse erscheint bei Rechtklick auf das Symbol des VI oben rechts im Frontpanelfenster.
Wie andere Programmiersprachen auch stellt LabVIEW diverse Strukturen zur Steuerung des Programmablaufs zur Verfügung. In Tabelle 1 sind einige grundlegende Typen aufgeführt.

	Case-Struktur: Abhängig von der Eingangsvariablen wird ein Fall (Case) ausgeführt (vergleichbar mit if()-elseif()-else in textbasierten Sprachen).
	For-Schleife: Wird so oft wiederholt, wie am N-Eingang angegeben. Der Anschluss i liefert den Wert des Schleifenzählers zurück (0 bis N-1).
a)  b) 	While-Schleife: Führt die Schleife solange aus (a) bis die Schleifenbedingung wahr ist oder (b) solange die Schleifenbedingung wahr ist. Der Inhalt der While-Schleife wird ausgeführt, bevor die Schleifenbedingung überprüft wird (vergleichbar mit do{} while() in textbasierten Sprachen).
	Formelknoten: Führt darin formulierte Berechnungen oder Vergleiche aus. Dient nicht direkt der Steuerung des Programmablaufs sondern der Übersicht, da er bei komplexen Berechnungen die Verdrahtung vieler Knoten ersetzt.

Tab. 1 Auswahl von Strukturen in LabVIEW

Um Daten innerhalb einer Schleife von einer Iteration zur nächsten zu übertragen, werden Schieberegister genutzt ( ).Zur Umwandlung des Ausgangs einer Schleife in ein Schieberegister muss diese Option nach Rechtsklick auf den Ausgang ausgewählt werden. In unserem Beispiel wird die zuletzt berechnete Größe von C gespeichert, wenn der Wert von OK = False ist (Abb. 3).

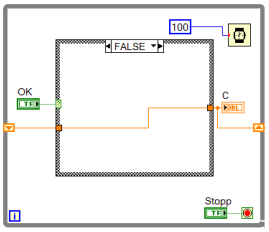


Abb. 3 Darstellung des Blockdiagramms für OK = False

Durch Lokale Variablen (C lesen, C schreiben) können innerhalb eines VI´s die Werte eines Objektes abgerufen oder verändert werden.

Für alle Funktionen von LabVIEW existiert unter „Hilfe - Kontexthilfe einschalten“ eine kurze Beschreibung des Verhaltens dieser Funktion. Gerade beim Einstieg in diese Sprache sollte man diese Hilfe bei der Auswahl der passenden Funktion nutzen.

Zu Aufgabe 0

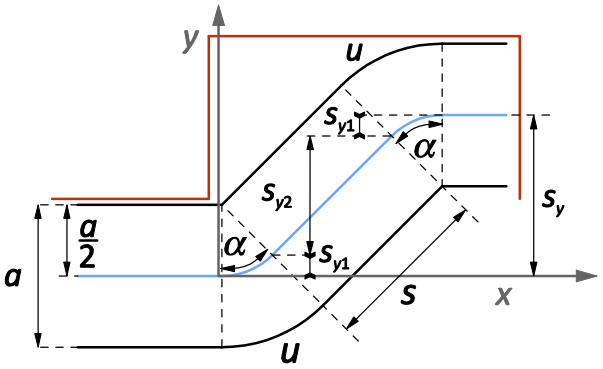


Abb. 4 Schema der zu fahrenden Strecke bei Aufgabe 2.
Braun: Parklücke
Schwarz: Bahn der Räder des NXT
Blau: Bahn des Achsmittelpunktes
Gegeben: α, s_y, a,

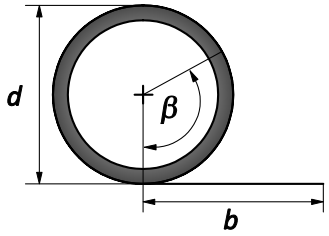


Abb. 5 Antriebsrad des NXT
Gegeben: β, d

Zu Aufgabe 1

Die zum Steuern und Lesen der Aus- und Eingänge des NXT nötigen VI´s befinden sich im Funktionsmenu NXT Robotics – NXT I/O.
Bei der Annäherung an ein Hindernis liefert der Ultraschallsensor Abstandsinformationen, um den Punkt zu bestimmen, an dem der Roboter anhalten soll.
Bei der Ausgabe der Zufallszahlen sollen die Werte auf dem Display in unterschiedlichen Reihen ausgegeben werden. Erfahrenere Programmierer/innen dürfen natürlich auch die Form $x*y = z$ in einer Reihe wählen.
Um eine Sinuskurve zu fahren, ist die Leistung der Motoren zu modulieren gemäß $P = P_0+x\sin(t)$, wobei $P_0 = 0{,}5P_{max}$ und die Modulation der Leistung $x = 0{,}1P_{max}$ betragen soll. In einer Schleife ist die notwendige Schrittweite für t zu erzeugen.

Zu Aufgabe 2

Der „Lückentext“ wird zu Beginn des Praktikums zur Verfügung gestellt und besprochen.
Ziel ist es, das „Fahrzeug“ in eine Parklücke fahren zu lassen (Abb. 4). Dabei soll der NXT geradeaus fahren, bis er eine reflektierende Marke erreicht. Diese Marke ist so angebracht, dass sich die Antriebsachse des NXT am Anfang der Parklücke befindet. Daraufhin soll er eine Drehung um 45° vollziehen, in die Parklücke fahren und anschließend eine Drehung in die entgegengesetzte Richtung machen.
Zum Ende des Parkmanövers soll er sich der hinteren Wand der Parklücke nähern.

Zusatz

In der Zusatzaufgabe können Sie ein eigenes vollständiges Programm erarbeiten. Vor Beginn der Programmierung sollte eine Beschreibung der Aufgaben des NXT erstellt werden.