

Modell 11: Malroboter

Ziele und Einordnung

Überblick

Der ferngesteuerte Buggy aus Modell 10 wird zum Malroboter umgebaut.

Das Zeichnen einfacher geometrischer Figuren kann automatisiert werden: Dazu ist eine Zeitmessung von Drehungen (= Winkel) und Geradeausfahrten (= Seitenlänge) erforderlich.

Hinweis: Die Aufgaben lassen sich am zuverlässigsten mit einer USB-Verbindung zum fischertechnik BT Smart Controller programmieren, da bei einer Bluetooth-Verbindung übertragungsbedingte Verzögerungen auftreten.



Themen

Wie kann man den Buggy um einen vorgegebenen Winkel drehen? Wie kann der Buggy eine vorgegebene Strecke abfahren?

Lernziel

- Repräsentation von Winkeln und Strecken durch Zeiten
- Messung von Zeiten mit Scratch

Zeitaufwand

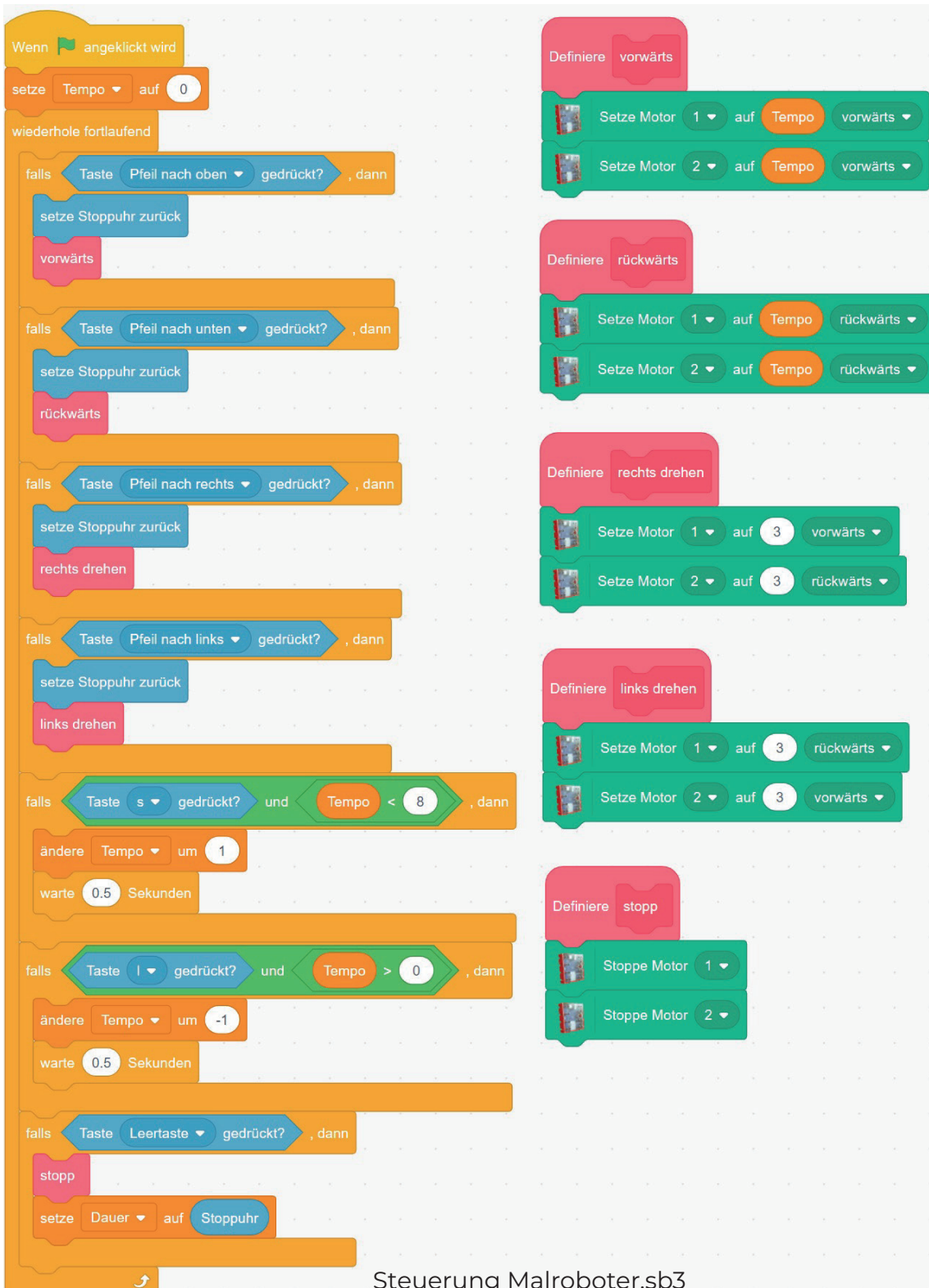
Der Aufbau des Malroboters benötigt ca. eine halbe Stunde. Sollte der Buggy aus Modell 10 noch aufgebaut sein, kann dieser umgebaut und die Bauzeit somit verkürzt werden. Die erste Aufgabe „Steuerung Malroboter“ ist eine direkte Anwendung der im Modell 10 (Buggy) programmierten Fernsteuerung des Buggy. Aufgabe „Messung Malroboter“ erfordert mehrere Messungen, um die für das Drehen um die erforderlichen Winkel (120°, 90°) und das Abfahren einer bestimmten Strecke benötigten Zeiten zu bestimmen. Aufgabe 3 ist eine direkte Anwendung der Messergebnisse und nicht schwierig zu programmieren. Für die Lösung der Aufgaben sollte eine Schulstunde ausreichen.

Die Lösung der Experimentieraufgabe ist etwas kniffliger; sie benötigt mindestens eine eigene Schulstunde. Die Lösung kann unterstützt werden, indem Zwischenergebnisse (wie die „Malstrategie“ für das „Haus vom Nikolaus“ oder die Festlegung der benötigten Strecken und Winkel) gemeinsam zusammengetragen werden.

Lösungen und Hinweise

Programmieraufgaben Modell 11: Malroboter

Lösungsvorschlag Aufgabe Zeitmessung:



The image shows a Scratch script for controlling a Malroboter. The main script is on the left, and the function definitions are on the right.

Main Script:

- Wenn **angeklickt wird**
- setze **Tempo** auf **0**
- wiederhole fortlaufend
 - falls **Taste Pfeil nach oben gedrückt?**, dann
 - setze **Stoppuhr zurück**
 - vorwärts**
 - falls **Taste Pfeil nach unten gedrückt?**, dann
 - setze **Stoppuhr zurück**
 - rückwärts**
 - falls **Taste Pfeil nach rechts gedrückt?**, dann
 - setze **Stoppuhr zurück**
 - rechts drehen**
 - falls **Taste Pfeil nach links gedrückt?**, dann
 - setze **Stoppuhr zurück**
 - links drehen**
 - falls **Taste s gedrückt?** und **Tempo < 8**, dann
 - ändere **Tempo** um **1**
 - warte **0.5** Sekunden
 - falls **Taste | gedrückt?** und **Tempo > 0**, dann
 - ändere **Tempo** um **-1**
 - warte **0.5** Sekunden
 - falls **Taste Leertaste gedrückt?**, dann
 - stopp**
 - setze **Dauer** auf **Stoppuhr**

Function Definitions:

- Definiere vorwärts:**
 - Setze Motor 1 auf **Tempo** **vorwärts**
 - Setze Motor 2 auf **Tempo** **vorwärts**
- Definiere rückwärts:**
 - Setze Motor 1 auf **Tempo** **rückwärts**
 - Setze Motor 2 auf **Tempo** **rückwärts**
- Definiere rechts drehen:**
 - Setze Motor 1 auf **3** **vorwärts**
 - Setze Motor 2 auf **3** **rückwärts**
- Definiere links drehen:**
 - Setze Motor 1 auf **3** **rückwärts**
 - Setze Motor 2 auf **3** **vorwärts**
- Definiere stopp:**
 - Stoppe Motor 1
 - Stoppe Motor 2

Steuerung Malroboter.sb3

Lösungen und Hinweise

Programmieraufgaben Modell 11: Malroboter

Lösungsvorschlag Aufgabe Malroboter (Dreieck):

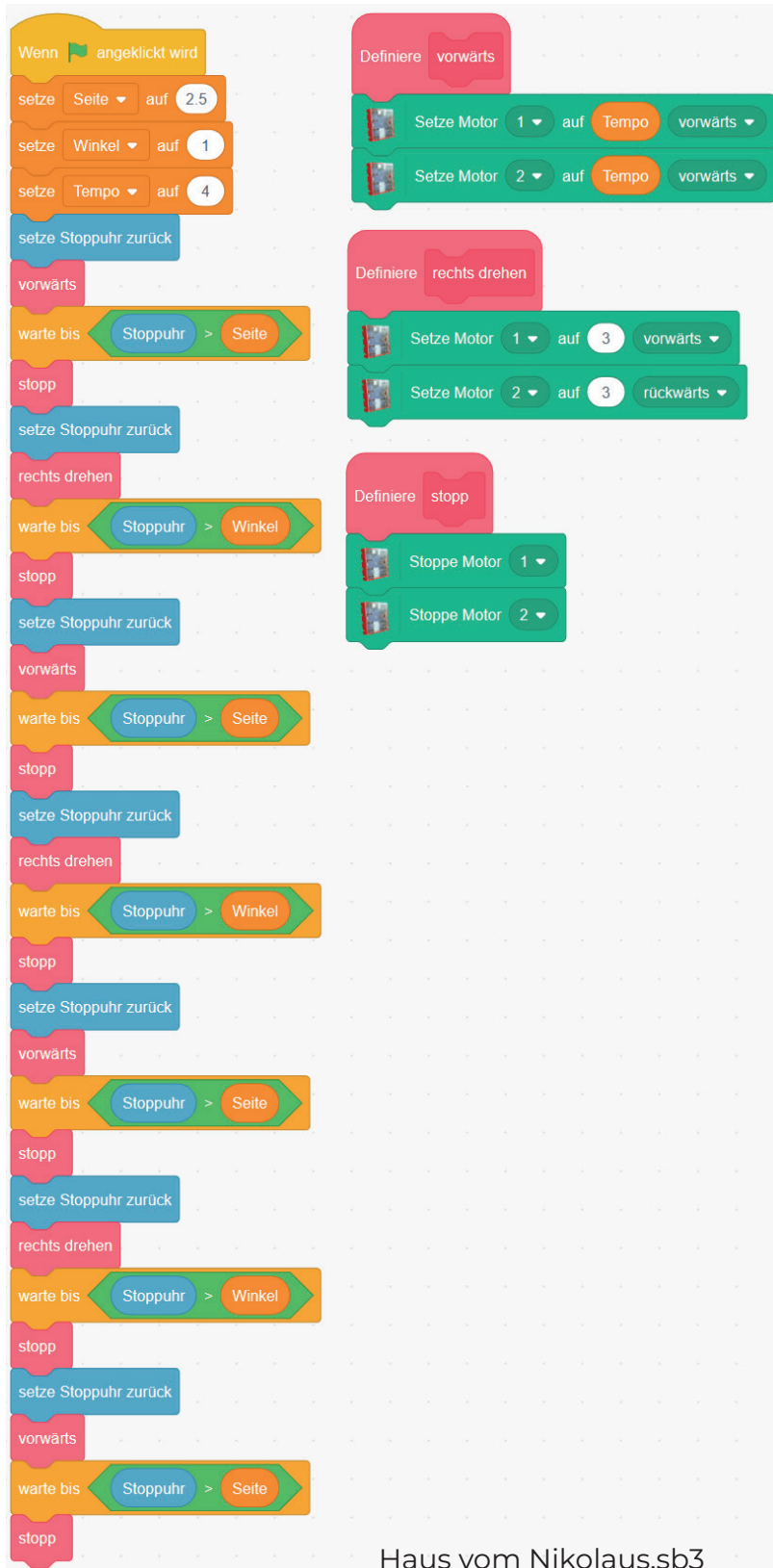


Messung Malroboter.sb3

Lösungen und Hinweise

Programmieraufgaben Modell 11: Malroboter

Lösungsvorschlag Aufgabe Malroboter (Quadrat):



Haus vom Nikolaus.sb3