Lektion 4: Der Rover soll fahren! **Übung 3: Zusätze, COLOR und Timing**

In der dritten Übung für Lektion 4 geht es um die Drehung um einen bestimmten Winkel, den Einfluss der Zeit (das richtige „Timing“) sowie um die Verwendung der RGB-LED (COLOR LED) auf dem Rover.

Lernziele:

- Verwendung des Befehls **TO ANGLE**
- Verwendung des Befehls **RV.COLOR**
- Abstimmung des Timings auf dem Taschenrechner und dem Rover

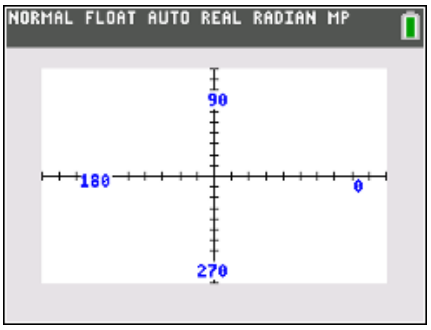
---

Diese Übung behandelt zwei Befehle und ein wesentliches Merkmal des Rovers:

- den Befehl **TO ANGLE**, der sich deutlich unterscheidet von LEFT und RIGHT
- das Einschalten der eingebauten RGB-LED durch **RV.COLOR** (bezeichnet mit ‘Color’, befindet sich ivorne links über der Ladezustandsanzeige)
- die Abstimmung der Bewegung des Rovers mit dem Programmablauf durch den Befehl **Wait**

Der Befehl TO ANGLE

Der Befehl **Send(“RV TO ANGLE** *<Zahl>***”)** wird verwendet, um den Rover um einen bestimmten Winkel zu drehen. Mathematisch gesehen entspricht dabei dem Winkel 0° die Richtung der positiven x-Achse (auch genannt „Ost“). Durch eine Drehung im Gegenuhrzeigersinn kommt man dann zu 90° (positive y-Achse, „Nord“), 180° (negative x-Achse, „West“) und 270° (negative y-Achse, „Süd“) (s. Bild rechts).

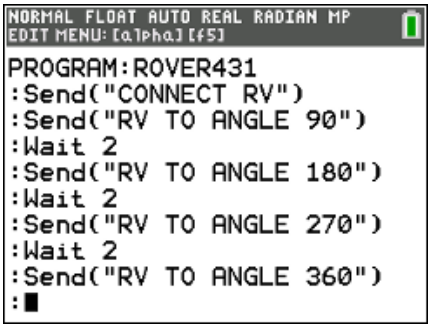


Drehrichtungen

Egal wie der Rover steht, wird seine Ausrichtung beim Empfang des Befehls **Send(**“**CONNECT RV**”) auf 0° festgelegt. Mit dem Befehl **Send(“RV TO ANGLE 0”)** wird der Rover angewiesen, sich aus jeder beliebigen Stellung wieder in diese anfängliche Ausrichtung zu drehen.

Die Voreinstellung für das Winkelmaß ist **DEGREES,** aber man kann die Winkeldrehung auch in **RADIANS** oder **GRADS** angeben (auswählbar im Menü **RV Settings…**).

Ein kleines Programm:




```
Send(“RV TO ANGLE 90”)
Wait 2
Send(“RV TO ANGLE 180”)
Wait 2
Send(“RV TO ANGLE 270”)
Wait 2
Send(“RV TO ANGLE 360”)
```

Verhält sich der Rover wie erwartet?

Programmablauf und Bewegung des Rovers synchronisieren

Das Programm auf dem Taschenrechner ist meistens abgelaufen („Fertig“), bevor der Rover seine Bewegungen abgeschlossen hat. Das liegt daran, dass die Fahrbefehle, die im TI-Innovator™ Hub gespeichert sind, weniger Zeit für ihre Durchführung benötigen wie der Rover für ihre Ausführung. Die Fahrbefehle werden in einer Art „Warteschlange“ gespeichert und ausgeführt, wenn der Rover dazu bereit ist.

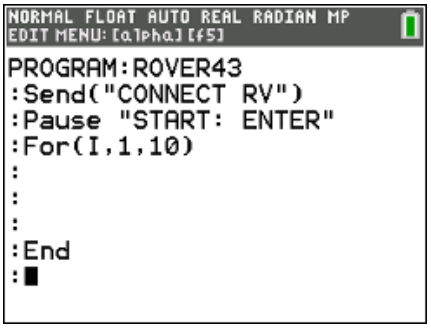
Das *Beispielprogramm* zeigt diese *Synchronisation* durch die Verwendung des Befehls **Wait**. Der Rover selbst bewegt sich zufallsgesteuert und schaltet dabei die eingebaute RGB-LED während der Bewegung mit dem Befehl **RV.COLOR** an. Innerhalb des Befehls **TO ANGLE** wird dabei **eval( )** verwendet, da der Winkel durch eine Berechnung entsteht.

Beispielprogramm

1. Das neue Programm ROVER43 anlegen.

2. Eine **For** – Schleife für die spätere Zufallsbewegung einfügen. Man sollte das zugehörige **End** nach ein paar Leerzeilen ebenfalls einfügen, um es später nicht zu vergessen.

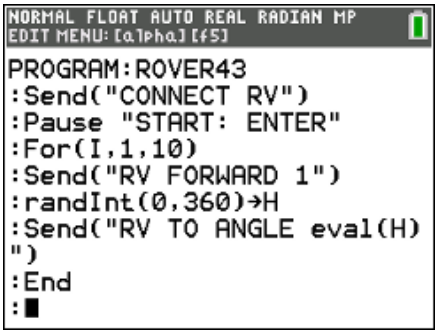
   Hinweis: Die Befehle **For(** und **End** befinden sich im Menü **CTL**.



3. In die Schleife kommt zunächst ein Befehl **FORWARD**.

4. Dann wird der Drehwinkel als eine Zufallszahl aus dem Bereich 0° … 360° erzeugt und in der Variablen H abgespeichert: **randInt(0,360) → H.**

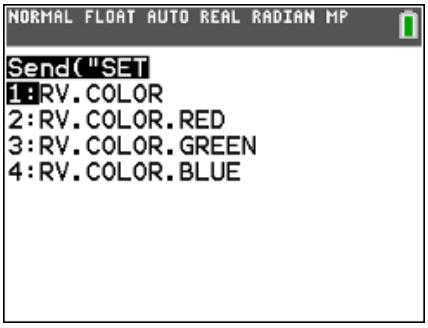
5. Jetzt kommt noch der Befehl **TO ANGLE** hinzu.



6. Lässt man nun das Programm laufen, so sollte man sehen:
   - Der Rover bewegt sich zufällig.
   - Das Programm meldet „Fertig“, während der Rover sich noch bewegt.

Der Befehl RV.COLOR

Da die RGB-LED des Hub durch den Rover verdeckt wird, wurde eine weitere RGB-LED oben auf dem Rover angebracht. Ihre Bezeichnung ist RV.COLOR. Sie kann genauso verwendet werden wie die RGB-LED auf dem Hub. Man kann sie über die vier Befehle steuern, die auf dem Bild rechts abgebildet sind. Man findet sie im Menü **prgm > Hub > Rover (RV)… > RV Color…**.

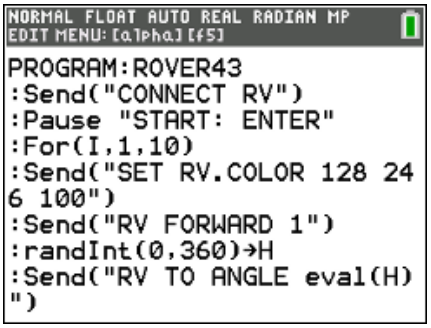


Send(“SET RV.COLOR 255 255 255”) erzeugt weißes Licht.

Einfügen des Befehls RV.COLOR in das Programm

7. Der Befehl **RV.COLOR** soll unmittelbar vor dem Befehl **FORWARD 1** in die **For(** – Schleife eingefügt werden. Die Farbwahl ist dabei beliebig.

8. Lässt man nun das Programm laufen, so sieht man, dass die LED sofort und dann dauerhaft leuchtet.



Nun soll die LED dazu gebracht werden nur dann zu leuchten, wenn der Rover sich vorwärts bewegt. Dazu muss der Rover warten, bis jede Teilbewegung vollendet ist und dann für die Drehung die LED ausschalten. Dazu muss ein **Wait**-Befehl dem Programm hinzugefügt werden.

Hinweis: An dieser Stelle hat man gut die Möglichkeit um zu experimentieren. Anders als die Fahrbefehle wird der Color-Befehl ausgeführt, sobald er beim Hub an der Reihe ist. Bewegung und die LED sind also nicht synchronisiert. Die Abstimmung obliegt dem Programmierer.

9. Wie lange dauert die Bewegung beim Befehl **FORWARD 1**? Etwa 1 Sekunde? Dann sollte man den Befehl **Wait 1** unmittelbar nach dem Befehl **FORWARD 1** einfügen. Den Befehl findet man im Menü **prgm** > **Hub**.

10. Lässt man nun das Programm laufen, so stellt man fest, dass die LED immer noch dauerhaft leuchtet. Sie muss also noch nach der Vorwärtsbewegung abgeschaltet werden.

```
NORMAL FLOAT AUTO REAL RADIAN MP
EDIT MENU: [alpha] [f5]
PROGRAM:ROVER43
:Send("CONNECT RV")
:Pause "START: ENTER"
:For(I,1,10)
:Send("SET RV.COLOR 128 24
6 100")
:Send("RV FORWARD 1")
:Wait 1
:randInt(0,360)→H
:Send("RV TO ANGLE eval(H)
```

11. Das geschieht durch den Befehl **Send("SET RV.COLOR 0 0 0")**.

12. Nun kann das Programm wieder getestet werden. Schaltet sich die LED zu den richtigen Zeiten ein und wieder aus?

```
NORMAL FLOAT AUTO REAL RADIAN MP
EDIT MENU: [alpha] [f5]
PROGRAM:ROVER43
:Send("CONNECT RV")
:Pause "START: ENTER"
:For(I,1,10)
:Send("SET RV.COLOR 128 24
6 100")
:Send("RV FORWARD 1")
:Wait 1
:Send("SET RV.COLOR 0 0 0"
)
```

Es müssen nun noch die Zeiten berücksichtigt werden, die der Rover zum Drehen benötigt, denn dann soll die LED ja nicht leuchten.

13. Ein weiterer **Wait** – Befehl wird *nach* dem Befehl **TO ANGLE** eingefügt. Er sollte lang genug gewählt werden, um jede Drehung bis hin zu 360° zu umfassen.

```
NORMAL FLOAT AUTO REAL RADIAN MP
EDIT MENU: [alpha] [f5]
PROGRAM:ROVER43
:Send("RV FORWARD 1")
:Wait 1
:Send("SET RV.COLOR 0 0 0"
)
:randInt(0,360)→H
:Send("RV TO ANGLE eval(H)
")
:Wait 3
:End
```

Erweiterung:
Nach jeder Drehung soll die Farbe wechseln! Hinweis: Verwende **eval()**.

Herausforderung: Man kann die Wartezeit bei der Drehung abhängig machen vom Drehwinkel, indem man die Zeiten bei der Verwendung von **TO ANGLE** misst.

Hinweis: Fahrbefehle werden sofort zum *TI-Innovator Hub* gesendet und dort in eine Warteschlange gestellt, bis der Rover sie durchführen kann.

Der Befehl **Wait** ist eine Anweisung an den *Taschenrechner* zu warten, während der Rover seine Fahrbefehle durchführt. Bei langen Wartezeiten kann es so aussehen, als würde der Rover pausieren während er doch nur auf den nächsten Befehl wartet.

Bei der Erweiterung können die Farben durchaus von der Schleifenvariablen I und dem Drehwinkel H abhängig gemacht werden. Der zulässige Wertebereich für die Farbvariablen r, g und b ist 0 … 255