From the first steps with Rover to autonomous driving (parallel parking)

Science
Technology
Engineering
Math

From the first steps with Rover to autonomous driving (parallel parking)

Science
Technology
Engineering
Math

Abstract

Starting with simple commands to let the Rover move we will try to show how a simple model of autonomous driving can be realized.

Step 1

Make the TI-Innovator Rover move FORWARD and BACKWARD, until a certain limit.

Step 2

Include a further ultrasonic sensor to measure the length of possible parking spots

Step 3

Let the Rover find the first suitable parking spot

Step 4

Let the Rover parking parallel

Science
Technology
Engineering
Math

Step 1 (idea and realization Hans Martin Hilbig)

Make the TI-Innovator Rover move FORWARD, until a certain limit.

Dependent on the distance of the ranger of the rover to a wall the following action is expected.

a) The wall is more than 30cm away:
   The Rover stops and the red RGB LED is on

b) The wall is more 13cm and less then 30cm away:
   The Rover goes forward until the distance is less than 13 cm.

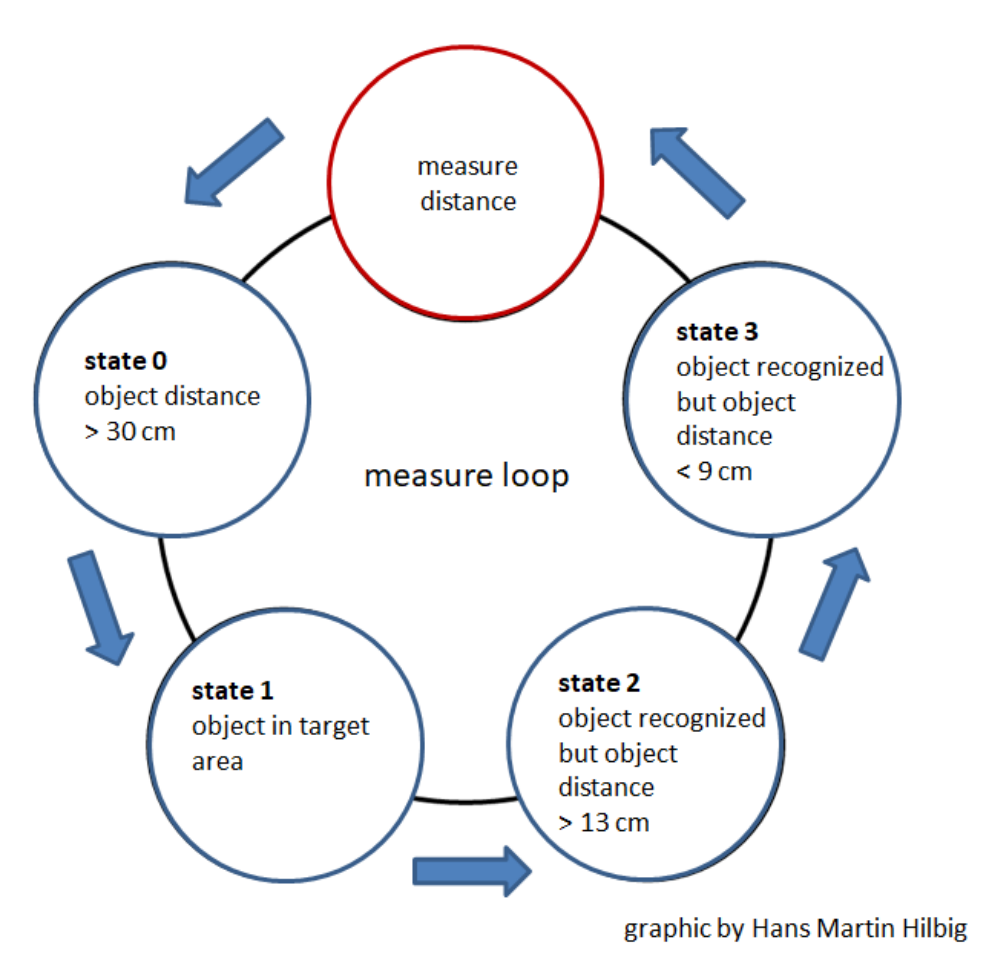
c) The wall is less than 13cm and more than 9cm away:
   The Rover stops and the green RGB LED is on

d) The wall is less than 9cm away:
   The Rover goes backward until the distance is more than 9 cm.

From the first steps with Rover to autonomous driving (parallel parking)

Science
Technology
Engineering
Math

State-Transition-Programming



From the first steps with Rover to autonomous driving (parallel parking)

Science
Technology
Engineering
Math

This is a part of code:

Local variables for the distance and the state

Global while-loop to stop the program with pressing any key

Defining and finding the state

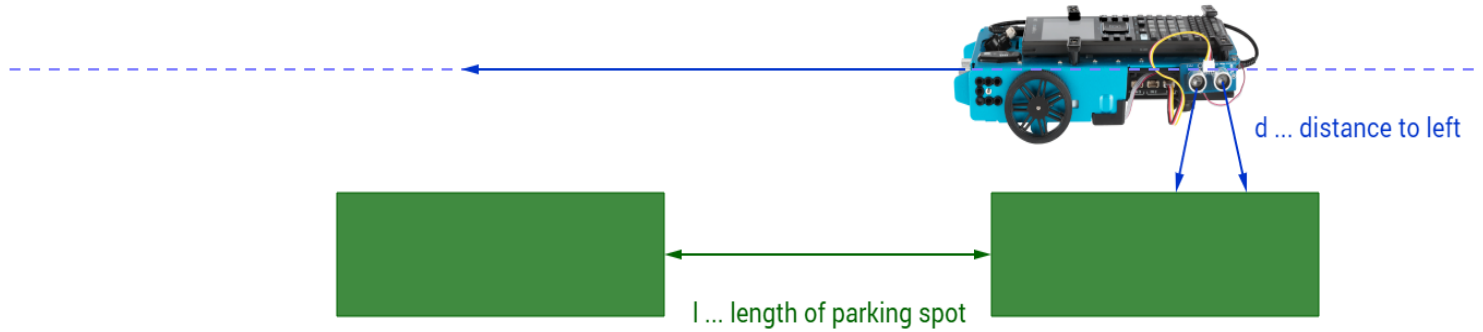
Do some experiments with the program.

```
* testing
Define testing()=
Prgm
© d … distance, s … state
Local d,s
Send "CONNECT RV"
s:=0
While getKey="[ ]"
   Send "READ RV.RANGER"
   Get d
©  DispAt 1, "distance: ", d
   If d > 0.3 Then
     Send "SET RV.COLOR.RED ON TIME 0.1"
     If s=0 Then
        Send "RV STOP "
        s:=0
     EndIf
     ElseIf d ≥0.09 and d≤0.13 Then
        If s>1 Then
           Send "RV STOP "
           s:=1
        EndIf
        Send "SET RV.COLOR.GREEN ON TIME 0.1"
```

Step 2

Include a further ultrasonic sensor to measure the length of possible parking spots.

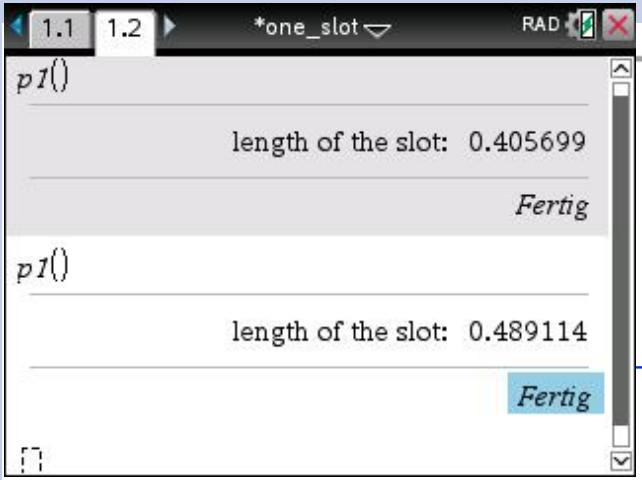
The Rover should stand by shown.

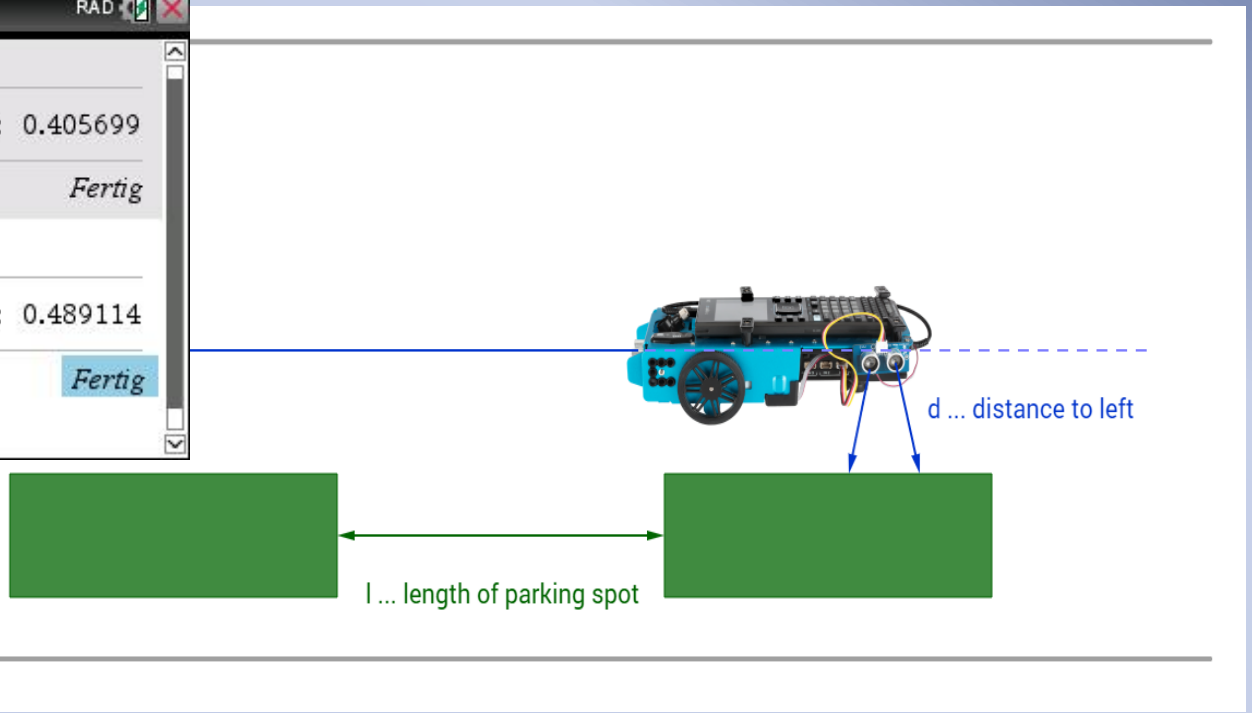


Step 2

Include a further ultrasonic sensor to measure the length of possible parking spots.

The Rover should stand by shown.





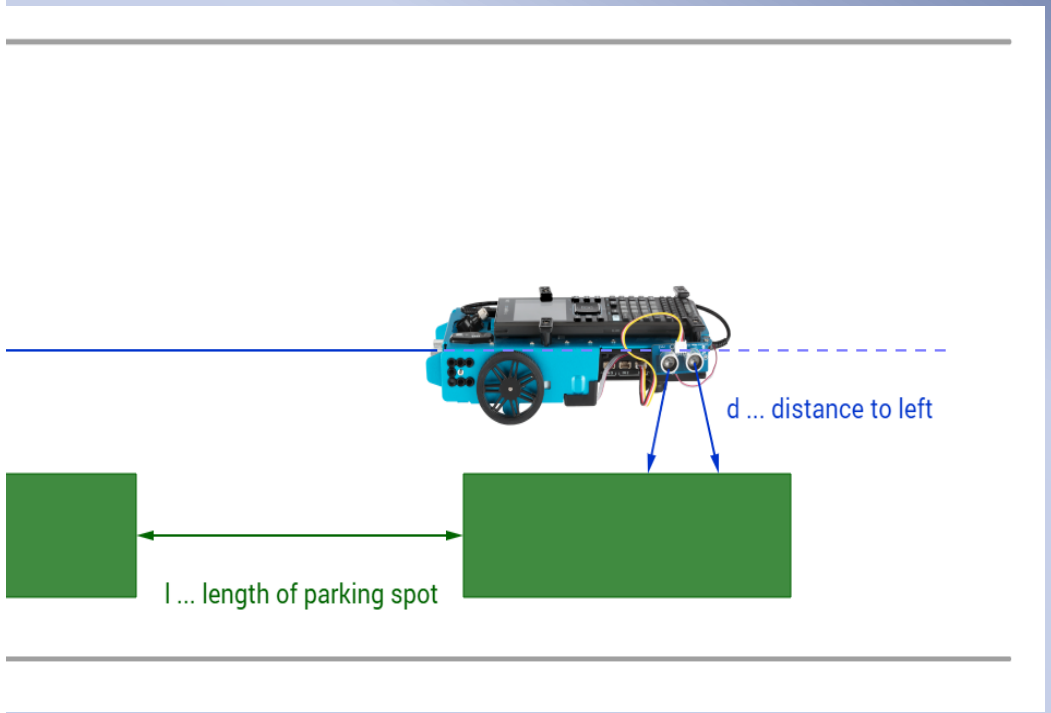
o

rking)

Science
Technology
Engineering
Math

a part of code

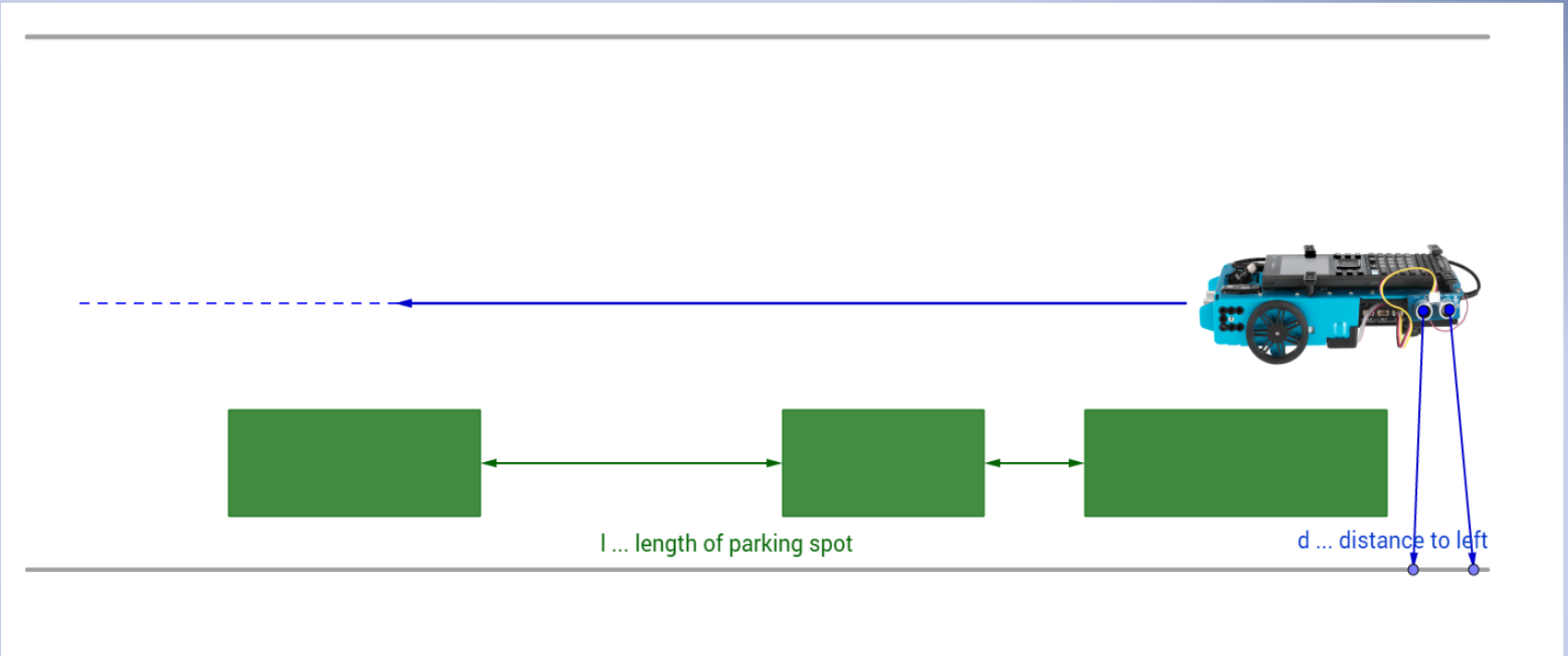
```
* p1
Define p1()=
Prgm
© var d … distance to the left
© var l  … length of the parking spot

Local d,l
d:=0: l:=0

Send "CONNECT RV"
Send "CONNECT RANGER 1 TO IN 1"
Wait 0.1

While d<0.3
    Send "RV FORWARD"
    Send "READ RANGER 1"
    Get d
EndWhile
Send "RV STOP CLEAR"

While d≥0.3
      Send "RV FORWARD"
      Send "READ RANGER 1"
      Get d
EndWhile
Send "READ RV.WAYPOINT.DISTANCE"
Get l
Disp "length of the slot:  ",l

Send "RV STOP CLEAR"
```



Science
Technology
Engineering
Math

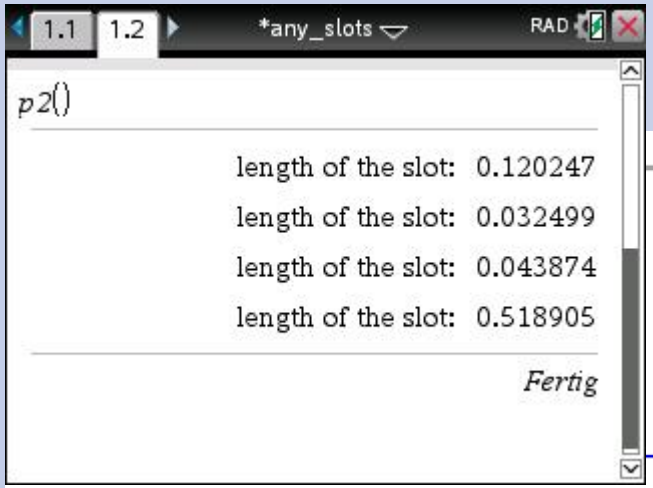
Step 3

Let the Rover find the first suitable parking spot



Step 3

Let the Rover find the first suitable parking spot



Science
Technology
Engineering
Math

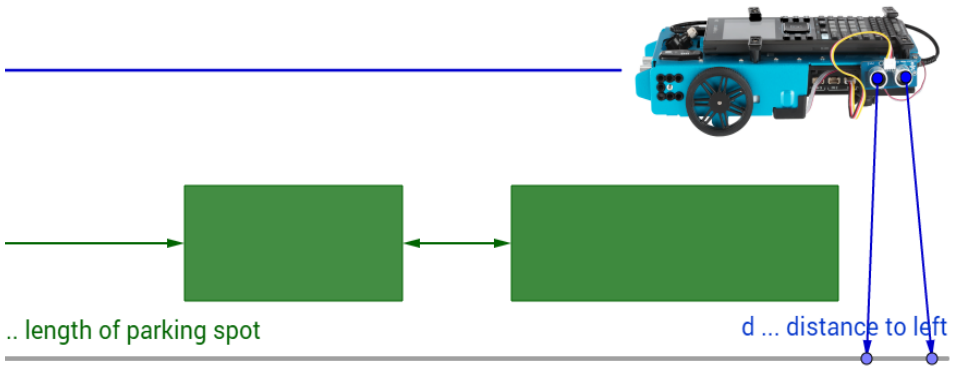
From the first steps with Rover to

king)

```
* p2
Define p2()=
Prgm
© var d … distance to the left
© var l … length of the parking slot

Local d,l
d:=0

Send "CONNECT RV"
Send "CONNECT RANGER 1 TO IN 1"
Wait 0.1

…

Send "READ RV.WAYPOINT.DISTANCE"
Get l
Disp "length of the slot: ",l

If l<0.5 Then
© recursive call
   p2()
Else
   Send "RV STOP CLEAR"
EndIf

EndPrgm
```

a part of code with recursive call

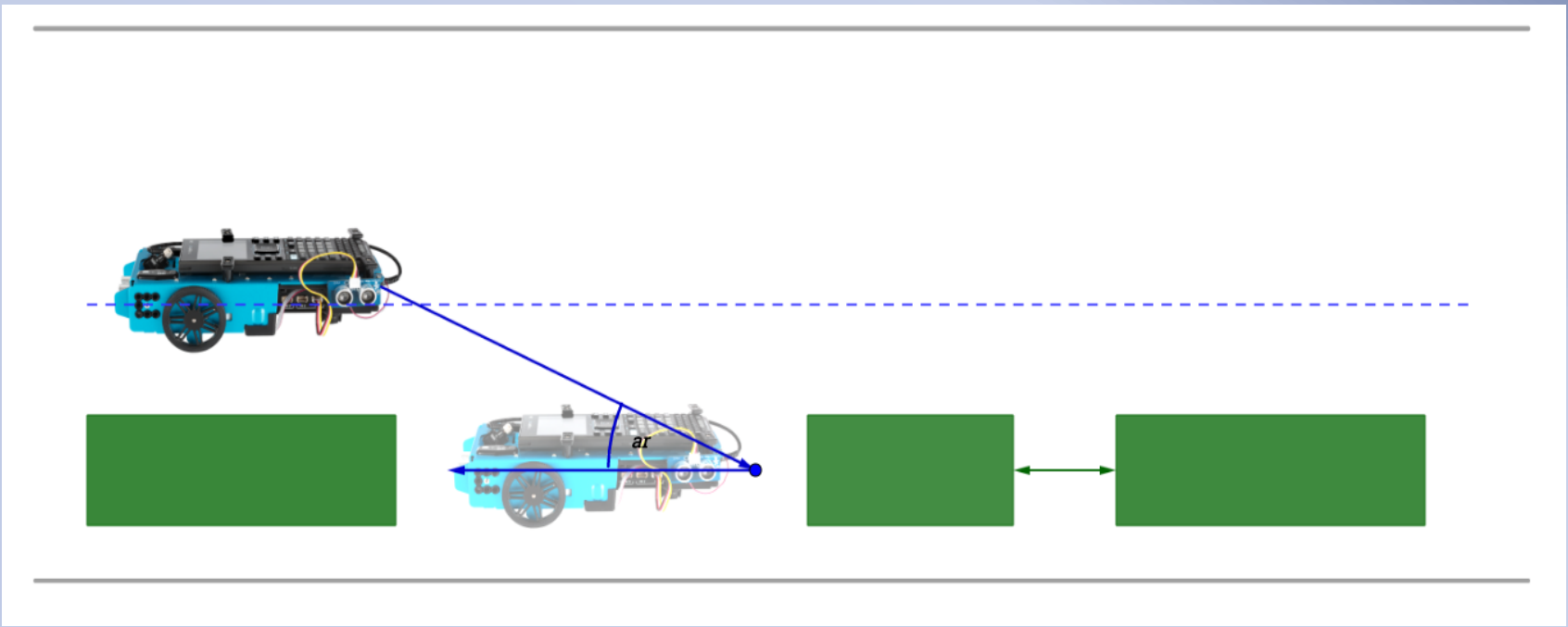
Idea by two 11th graders
B.Gerlach and K. Volkenand



Science
Technology
Engineering
Math

Step 4

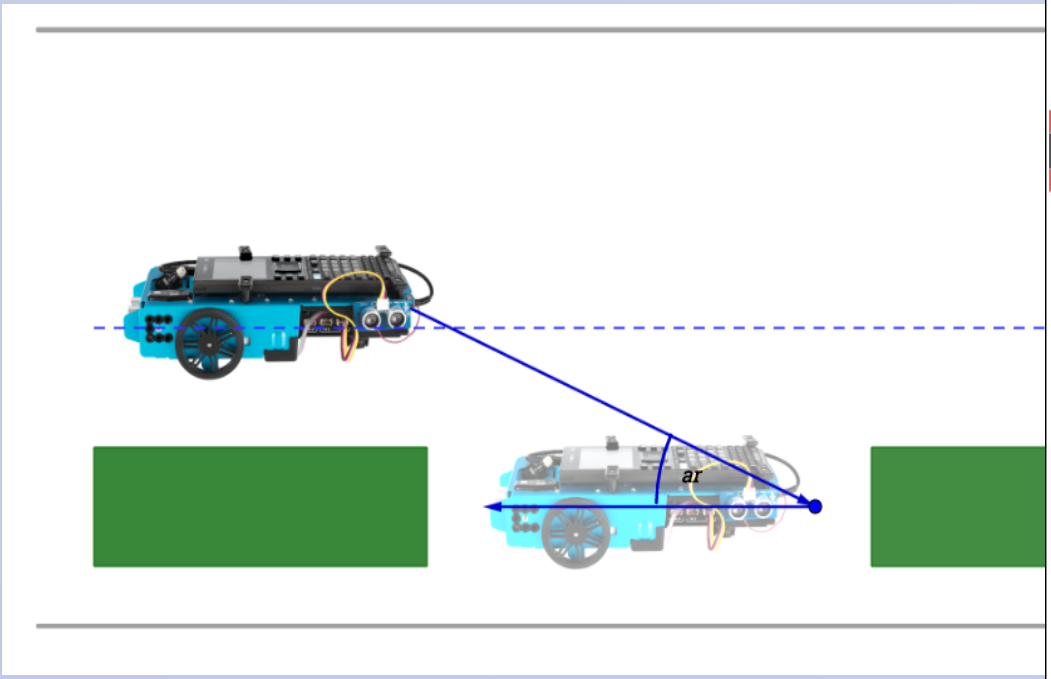
Let the Rover parking parallel



From the first steps with Rover to autonomous driving (parallel parking)

Science
Technology
Engineering

a part of code with Mathematic of backward parking



p3

```
© --- parking ---
  Disp "length of the last slot:  ",l
  lcorr:=l+d/2
  Wait 2
  Disp "corrected length: ",lcorr

  ar:=tan^-1((3·d)/(2·lcorr))·180/π

  Disp "angle right: ",ar

  back:=√(d^2+4/9·lcorr^2)

  If ar<15 Then
    Send "RV BACKWARD 0.1 M"
    backcorr:=back+0.04/d
  Else
    backcorr:=back+0.06/d
  EndIf
  Send "RV RIGHT EVAL(ar)"

  Send "RV BACKWARD EVAL(backcorr) M"
```

Science
Technology
Engineering
Math

Programming language: TI-BASIC

TI-BASIC is a programming language based of BASIC from Texas Instruments for programming of TI-Graphic and CAS-calculators and the Hub

View also:

TI-BASIC Developer (english):
http://tibasicdev.wikidot.com/starter-kit

TI Codes (TI-Nspire Technology)
https://education.ti.com/en/activities/ti-codes/nspire/10-minutes

TI Rover: important commands
TI-Innovator_Technology_Guidebook_EN_V_1_3 from page 29