



#### Unit 5: Rover's Sensors

#### Skill Builder 1: Introducing Ranger

In this lesson, you will learn to read the distance from the front of the TI-Innovator Rover to an obstacle. This lesson monitors that distance and displays information in two different ways. Another sensor is on the bottom of Rover and it can 'see' colors.

#### Objectives:

- Read Rover's Ranger measurement
- Display it on the screen using **print()**
- Display it on the screen using **text\_at()**
- Determine the unit of measure

The two small cylinders on the front of Rover are not headlights. They are an Ultrasonic Ranger. A silent (to us) tone is sent out by one of the two sensors and the other one 'hears' the echo when the sound bounces off an obstacle. The internal software then calculates the distance to the obstacle using the speed of the sound and the time that it takes for the echo to return to the sensor:  $D = V * T$

All that happens in the blink of an eye!



1. Begin a new Python Rover Coding project. For this lesson, use the convenient 'press **esc** to exit' loop:

```
while get_key() != "esc":  
    block
```

You will find this command on several menus, but since you are using **Rover** commands, look in **menu > TI Rover > Commands**.

```
1.1 1.2 1.3 Unit5 Pyt...ors RAD 10/22  
*u5sb1.py  
# Rover Coding Unit 5 SB1  
#=====   
import ti_rover as rv  
from math import *  
import ti_plotlib as plt  
from ti_system import *  
from time import *  
#=====   
while get_key() != "esc":  
    block
```

2. Add three statements in the **while** block to

- a) read the distance in front of Rover:

```
dist = rv.ranger_measurement()
```

(Find this on **menu > TI Rover > Inputs**.)

- b) print it on the screen:

```
print("Distance= ",dist)
```

- c) wait before reading the next distance:

```
sleep(.5)
```

Run the program and move your hand toward and away from the front of Rover. Point Rover at a wall, the ceiling, and the floor. Watch the values that are displayed. Note: Rover does not move...yet.

Can you determine what distance unit (feet, meters, etc.) is being used?

```
1.1 1.2 1.3 Unit5 Pyt...ors RAD 13/24  
*u5sb1.py  
from time import *  
#=====   
while get_key() != "esc":  
    dist=rv.ranger_measurement()  
    print("Distance= ",dist)  
    sleep(.5)
```



# 10 Minutes of Code - Python

TI-NSPIRE™ CX II WITH THE TI-INNOVATOR™ ROVER

## UNIT 5: SKILL BUILDER 1

### STUDENT ACTIVITY

- Rather than have the printed numbers scroll down the screen, improve the display of the distance values by using the **text\_at()** function found in the **tiplotlib** module. If you worked through the previous TI-Innovator™ Hub lessons (Units 1, 2, and 3), you have used that command before. But here you *must* provide the (temporary) name of the module due to the way it is imported in the Rover template:

**import tiplotlib as plt**

This *requires* that any functions that reside in the **tiplotlib** module *must* be preceded by the 'alias' name **plt**. You will also use the **cls()** function in the module which clears the drawing screen.

- See the functions

**plt.cls()**

**plt.text\_at(...)**

found on **menu > TI PlotLib > Draw**.

Other functions found in this module are used for graphing and will be useful when you start using Rover's Coordinate System in the next unit.

- Turn the **print()** statement into a *#comment* by pressing **ctrl+T** on the **print()** statement.

Add the two **plt** functions

**plt.cls()**

**plt.text\_at(7, str(dist), "left")**

found on **menu > TI PlotLib > Draw**.

About **str(dist)** again: The **plt.text\_at()** function can only display "text" which is either a literal string "in quotes" or a variable containing a string. **dist** is a variable containing a number, so it needs to be converted to a string using the **str()** function found on:

**menu > Built Ins > Type**

Row 7 is the vertical center line of the 13-line layout of this screen.

You can sample the distance faster by using a smaller **sleep()** value.

You have just created a digital tape measure!

*In the next lesson, you will use this information to keep Rover from crashing into something.*

```

1.1 1.2 1.3 *Unit5 Pyt.ors RAD 5/24
# Rover Coding Unit 5 SB1
=====
import ti_rover as rv
from math import *
import tiplotlib as plt
from ti_system import *
from time import *
=====
while get_key() != "esc":
    dist=rv.ranger_measurement()
    print("Distance= ",dist)

```

```

1 Actions 5 Pyt.ors RAD 1/16
2 Run
1 color(red,green,blue)
2 cls()
3 show_plot()
4 scatter(x-list,y-list,"mark")
5 plot(x-list,y-list,"mark")
6 plot(x,y,"mark")
7 line(x1,y1,x2,y2,"mode")
8 lin_reg(x-list,y-list,"display")
9 pen("size","style")
A text_at(row,"text","align")

```

```

1.1 1.2 1.3 *Unit5 Pyt.ors RAD 16/26
# u5sb1.py
=====
while get_key() != "esc":
    dist=rv.ranger_measurement()
    # print("Distance= ",dist)
    plt.cls()
    plt.text_at(7,"dist","left")
    sleep(.5)

```