

# Festo Robotino Operation Manual

## Introduction:

Festo Robotino 3 is a mobile robot platform for research and education. With its wide range of drives, sensors and interfaces, Robotino® can be used very flexibly. It also supports a wide range of programming languages such as Python, C++, MATLAB, LabView as well as ROS. This manual aims to familiarise students regarding the operation and programming of Robotino.

## List of sensors and accessories:

- URG Hokuyo 2D laser scanner x1
- IR distance sensors x9
- Bumper sensor x10
- Gyroscope x1
- Optional sensors: The Robotino can also be equipped with a camera. This can be done by connecting any USB-enabled camera to one of the USB ports in the Robotino
- Wheel encoder x3

## List of drives:

- Omni-directional wheels with motors x3  
These wheels allow for smooth linear and angular movement in any direction.

## List of interfaces:

**Note:** This list includes the interfaces that are most likely to be used by the students. Additional interfaces exist which have not been listed in this section

- TP-Link TL-WDN4800 USB Router: This USB router enables wireless communication between the user and the Robotino and **must not be removed** from the robot.
- Ethernet port
- Digital inputs x8
- Digital outputs x8
- Analog inputs x8
- 24V relays x2
- 24V power supply x5
- VGA display output x1
- USB port x6 (One of which is taken up by the TP-Link router)

## Assembly and connection

The I/O interface is mounted directly on the main PCB in the control unit.



The following connections are included:

Connection	Typ
DI1 ... DI8	Digital inputs, 24 V, protected against overload
DQ1 ... DQ8	Digital outputs, 24 V, short-circuit, max. 1 A
AI1 ... AI8	Analogue inputs, 0 - 10 V, 50 Hz
NO1 ... NC2	Relay, 24 V
+/-	Power supply, 24 V, max. 3 A total

Overview of I/O interfaces. Source: Festo Infoportal

### Wireless Communication:

An integral part of communication with the Robotino is the wireless network that it hosts. Communication can occur only when the host PC is connected to the WiFi network of the Robotino.

**Wifi Network name:** Robotino.300.796

**Network Password:** robotino

**IP address of robotino:** 172.26.1.1

### Robotino's Operating System (OS) Credentials:

**Note:** Please do not connect and interact with the OS unless you're 100 percent sure about what you're doing!

Robotino runs on **Ubuntu 12.04 LTS**. The credentials for the OS are as follows

User	Password
robotino	robotino
root	dorp6

It is very unlikely that you will need to interface directly with the OS within the Robotino. It is also not recommended to change any settings since it can lead to failure in the robot. However, these credentials might be useful for troubleshooting.

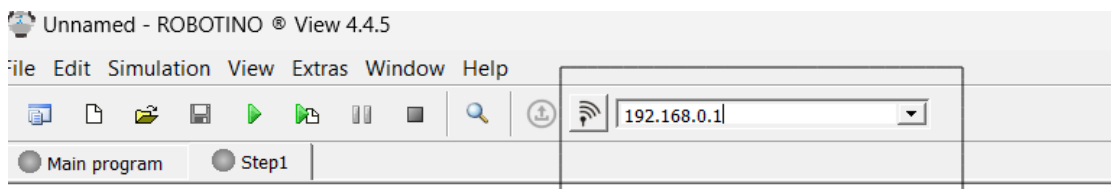
### Connecting to Robotino:

The 4 major ways of connecting to Robotino are as follows:

1. **Windows/Linux:** Connecting to Robotino via Robotino View
2. **Ubuntu/Linux:** Connecting to Robotino via ROS
3. **VNC wireless display:** Connecting to the OS of Robotino directly using VNC
4. **VGA wired display:** Connecting to the OS of Robotino directly using VGA

### Connecting to Robotino via Robotino View in Windows/Linux:

1. Install Robotino View in your laptop. Download link can be found [here](#).
2. Connect to the wifi network of the Robotino using the credentials provided previously
3. Open Robotino View
4. In the box at the top of the screen, change the IP address to the one provided previously in the manual



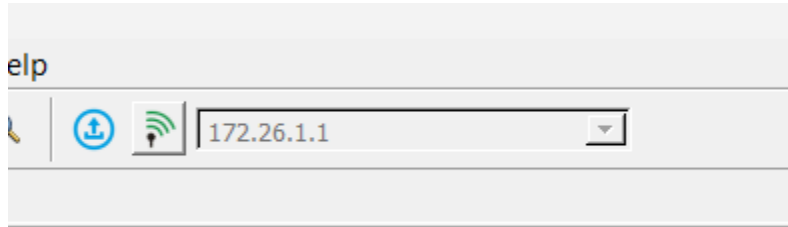
Step 4: Part 1



Step 4: Part 2

5. Press the “Ping” button which is directly to the left of the box, to connect to the Robotino

6. If the connection is successful, the box will get greyed out and the “Upload” button next to the box will change from, grey to blue



Step 6: Successful connection

### Connecting to Robotino using ROS in Ubuntu/Linux:

**Note:** Last confirmed to work in Ubuntu 20.04 with ROS Noetic. This section assumes that the student has prior knowledge of ROS

1. Go to the Github Repo for the ROS node. Link can be found [here](#).
2. Download/clone the ROS robotino repo to your laptop.
3. Follow the instructions given on the README.md file within the git repo.
4. **Important:** For Robotino 3, the following variables must be set in the terminal:  

```
export ROBOT=rto-3  
export ROBOT_ENV=sample
```
5. Run the command “roslaunch rto\_bringup robot.launch” . The robot should be connected successfully. Refer the figure below for the correct output of the command.

```

/home/rahul/catkin_robotino_ws/src/rto_core/rto_bringup/launch/robot.launch http://localhost...
NODES
/
  robot_state_publisher (robot_state_publisher/robot_state_publisher)
  rto_drive (rto_node/rto_node)
  rto_laser (rto_node/rto_laserrangefinder_node)
  rto_odom (rto_node/rto_odometry_node)

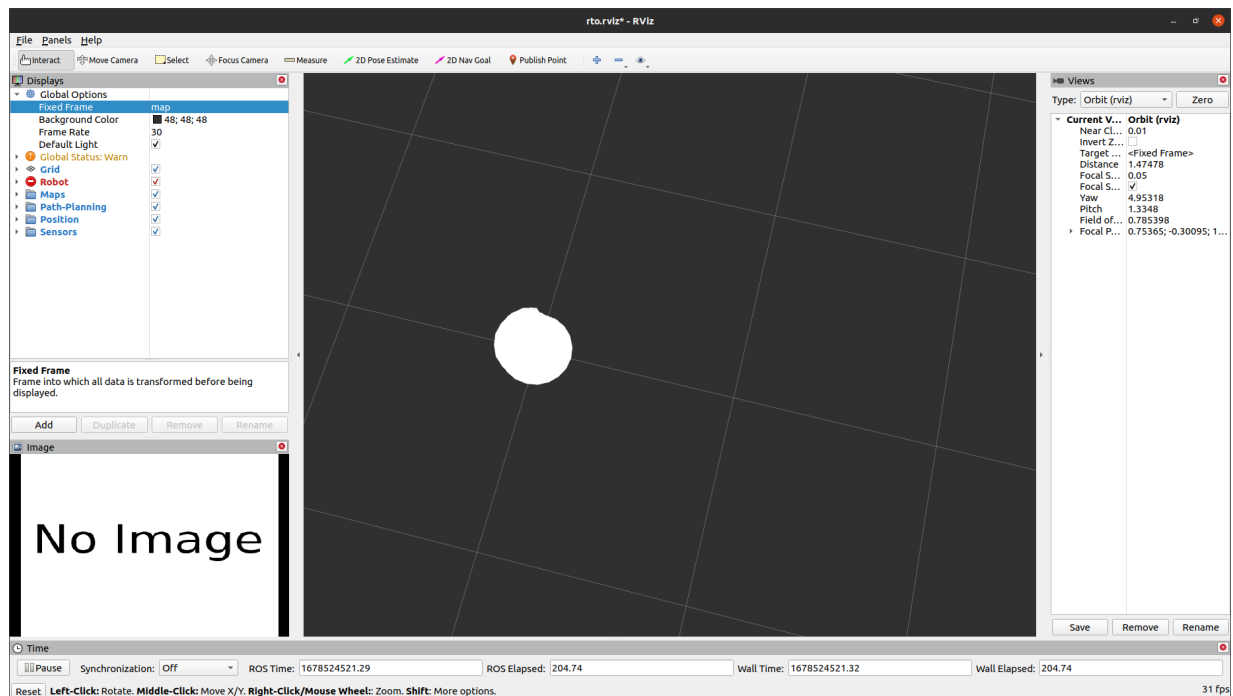
auto-starting new master
process[roscout-1]: started with pid [5659]
ROS_MASTER_URI=http://localhost:11311

setting /run_id to 5ff4c6b6-bfe8-11ed-b2d8-291dcd29d295
process[roscout-1]: started with pid [5669]
started core service [/roscout]
process[rto_drive-2]: started with pid [5676]
process[rto_odom-3]: started with pid [5677]
process[rto_laser-4]: started with pid [5678]
process[robot_state_publisher-5]: started with pid [5679]
[ WARN] [1678524028.825967348]: The root link base_link has an inertia specified in the URDF, but KDL does not support a root link with an inertia. As a workaround, you can add an extra dummy link to your URDF.
[ INFO] [1678524028.866038583]: Odometry connected to RTO.
[ INFO] [1678524028.896524109]: LaserRangeFinder0 connected to RTO.
[ INFO] [1678524028.896615761]: RTONode connected to RTO.

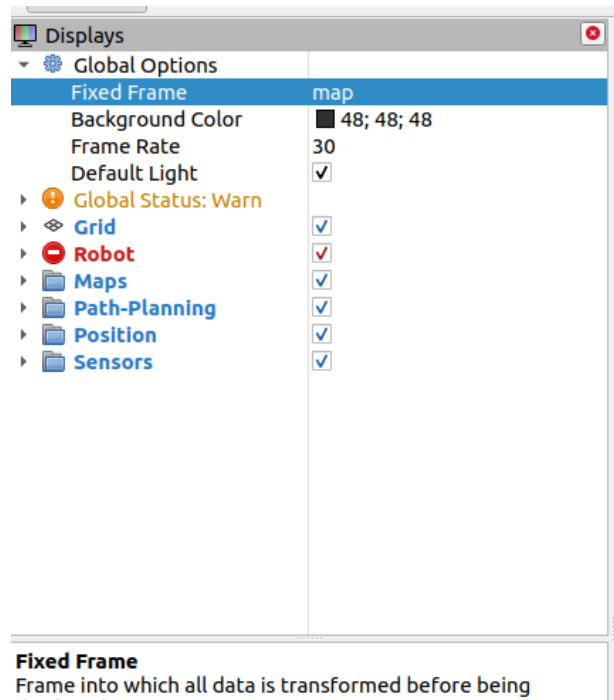
```

Step 5: Terminal output on successful connection

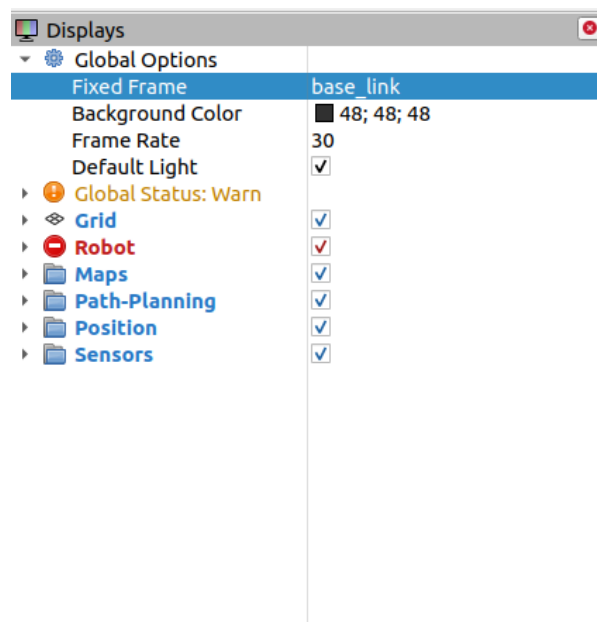
6. To visualise the sensor readings, you can use “roslaunch rto\_bringup rviz.launch”



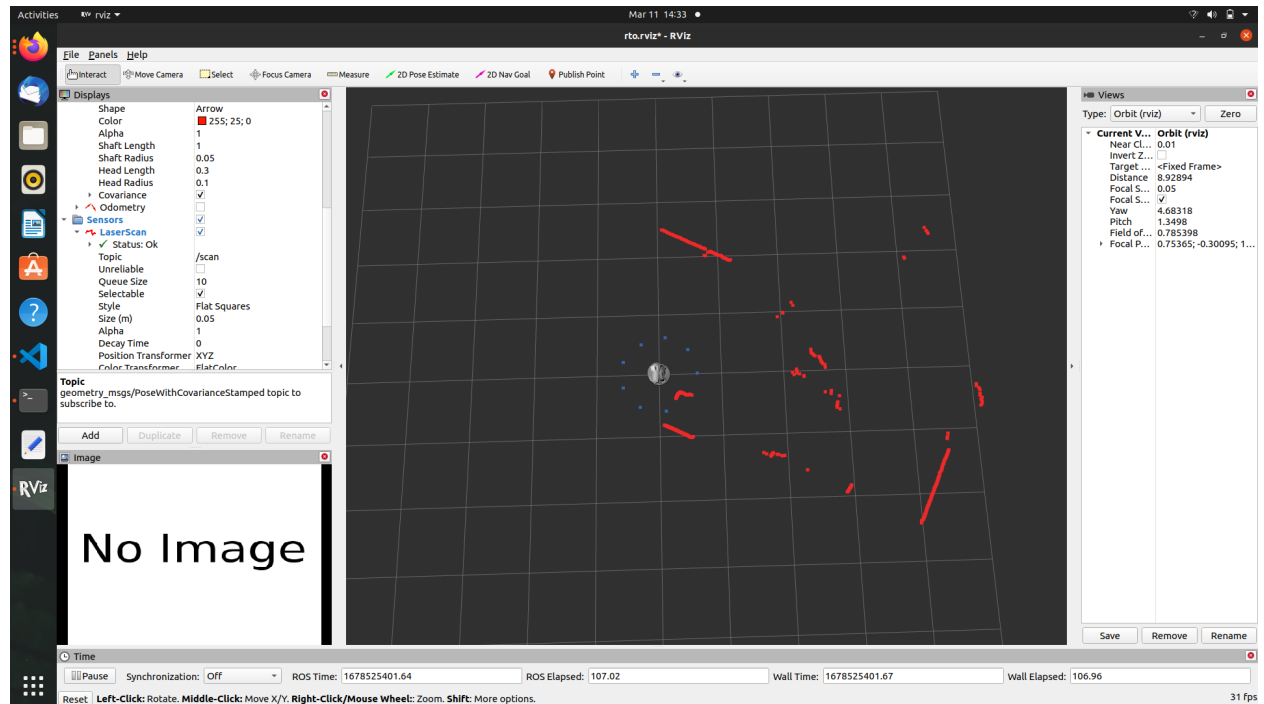
7. Initially, RViz might act erroneously, to solve this, under “Global options”, change fixed frame from “map” to “base\_link”. RViz should now work normally.



Step 7: Initial value



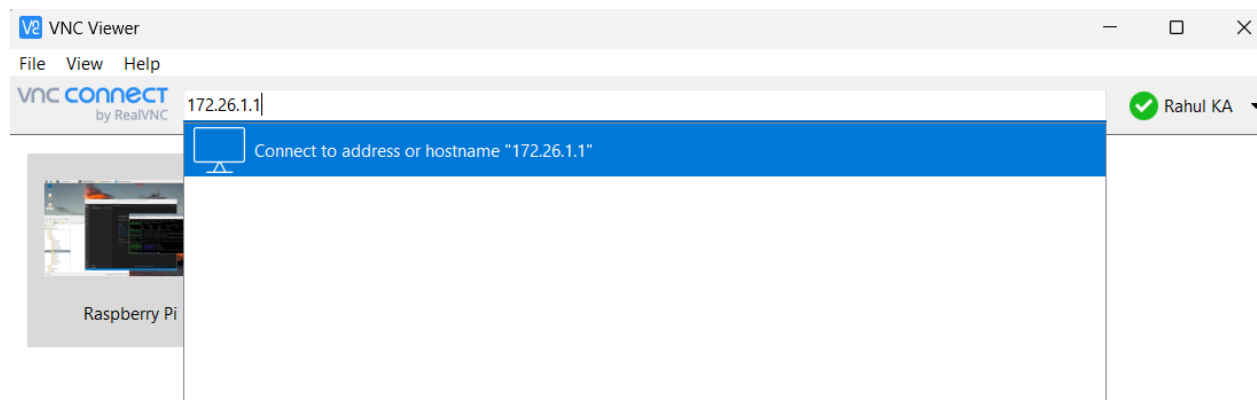
Step 7: Correct Value



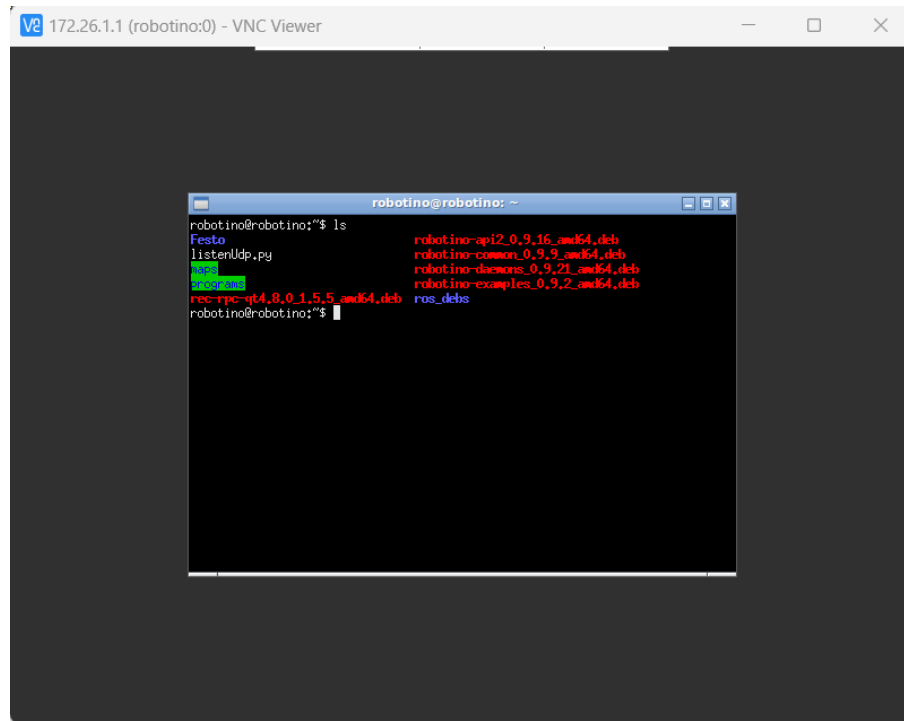
Step 7: Proper working of RViz

### Connecting to Robotino using VNC:

1. Download any VNC viewer of your choice (RealVNC, TightVNC, etc.) in your laptop.
2. Connect to the WiFi network of Robotino
3. Add a new session, and in the IP address field enter "172.26.1.1"
4. The OS of the Robotino should now show up
5. You can now use the terminal to manipulate the OS inside the Robotino. The terminal uses the Ubuntu operating system credentials listed previously.



Step 3: Connecting in RealVNC



The image shows a VNC Viewer window titled '172.26.1.1 (robotino:0) - VNC Viewer'. Inside the window is a terminal window titled 'robotino@robotino: ~'. The terminal shows the command 'ls' being executed, resulting in a list of files and directories. The output is as follows:

```
robotino@robotino:~$ ls
Festo          robotino-api2_0.9.16_amd64.deb
listenUdp.py   robotino-common_0.9.9_amd64.deb
rec-rpc-qt4_8.0.1-5_amd64.deb robotino-dawears_0.9.21_amd64.deb
ros_debs       robotino-examples_0.9.2_amd64.deb
```

VNC output

### Connecting using VGA:

1. Connect a USB keyboard and USB mouse to the Robotino using the USB ports.
2. Using a VGA connector, connect the Robotino to a display monitor.
3. The output will be similar to that of the VNC method
4. An advantage of using VGA is that it let's you connect to the Robotino even if the WiFi network is down
5. Using VGA, it is also possible to directly access the BIOS of the controller inside the Robotino.
6. However, it is recommended that VGA connection only be performed as a last resort, since it is much easier to diagnose any issues in the Robotino using VNC

### Programming using Festo:

- It is recommended to use either ROS or Robotino View to write programs for the Robotino. However, it is possible to use REST APIs in C++, MATLAB and Python to connect to the Robotino and control it.
- More info on various programming APIs for the Robotino can be found here:  
<https://wiki.openrobotino.org/index.php?title=Coding>
- The document below contains the documentation for writing programs in Robotino View:  
[https://doc.openrobotino.org/download/RobotinoView/RobotinoView2\\_EN.pdf](https://doc.openrobotino.org/download/RobotinoView/RobotinoView2_EN.pdf)



**References and useful links:**

These links may provide more insight into the Robotino system. Open Robotino will probably have most of the answers to your questions. The Festo Infoportal is also a very good resource

:

1. Open Robotino page for Robotino 3:  
<https://wiki.openrobotino.org/index.php?title=Robotino3>
2. Official Festo Info Portal for Robotino:  
<https://ip.festo-didactic.com/InfoPortal/Robotino3/Overview/EN/index.html>
3. Software downloads page for Robotino (including Robotino View):  
<https://wiki.openrobotino.org/index.php?title=Downloads>
4. OpenRobotino Forums: <https://forum.openrobotino.org/>
5. GitHub page for Robotino ROS repository by dietriro: [https://github.com/dietriro/rto\\_core](https://github.com/dietriro/rto_core)
6. OpenRobotino ROS page : <https://wiki.openrobotino.org/index.php?title=ROS>
7. Documentation for C++ REST API for Robotino:  
[https://doc.openrobotino.org/download/RobotinoAPI2/rec\\_robotino\\_api2/](https://doc.openrobotino.org/download/RobotinoAPI2/rec_robotino_api2/)