TogetheROS.Bot Environment Setup

TogetheROS.Bot supports installation on the RDK with Ubuntu 20.04/22.04 system. Installing through DEB packages on Ubuntu system is simple and recommended for users who want to experience it initially.

1.Environment Setup

Before installing tros.b, it is recommended to upgrade the RDK system image to the latest version.

If the image has already been installed, you can upgrade it by running the commands `sudo apt update` and `sudo apt upgrade`.

Friendly Reminder:

- **If you have installed system version 1.x, you need to upgrade to version 2.x.**
- **For instructions on how to check your system version number, please see [FAQs](**https://d-robotics.github.io/rdk_doc/FAQ/applications_and_examples)。 **

2. Installing TogetheROS.Bot

2.1 Installing and Upgrading TogetheROS.Bot via apt

Prerequisites

- The Environment Setup has been completed.
- The RDK system has been installed.
- The RDK can access the internet normally.
- The RDK can be accessed remotely via SSH.

Note:

- **The 2.x version of tros.b only supports 2.x system images. The 1.x version of tros.b (https://developer.d-robotics.cc/api/v1/fileData/TogetherROS/index.html) only supports 1.x systems.**
- **If you are using a 1.x system image, you need to upgrade your system (https://d-robotics.github.io/rdk_doc/Robot_development/quick_start/preparation) to version 2.x.**
- **For instructions on how to check your system and tros.b version numbers, please see the FAQs (https://d-robotics.github.io/rdk_doc/FAQ/applications_and_examples). ****

Dependencies	1.x tros.b	2.x tros.b
1.x System Image	√	x
2.x System Image	x	√

Installing tros.b

Note: The RDK IP used here is 192.168.2.59. You need to replace it with your own RDK IP during installation.

Log in to RDK:

```
ssh root@192.168.2.59
```

Install the tros.b package:

- Foxy
- Humble

```
sudo apt update
sudo apt install tros-humble
```

Note: If you encounter the error `E: Unmet dependencies. Try 'apt --fix-broken install' with no packages (or specify a solution).' after running the installation command, please execute the command` **apt --fix-broken install` to install the related dependencies before installing tros.b.**

After the installation is complete, check the files in the /opt directory

```
root@ubuntu:/userdata# ls /opt/
hobot  tros
```

The tros.b is installed in the /opt directory.

Upgrade tros.b

Taking RDK installation as an example, the upgrade method for x86 Ubuntu is the same as that for RDK.

Login to RDK:

```
ssh root@192.168.2.59
```

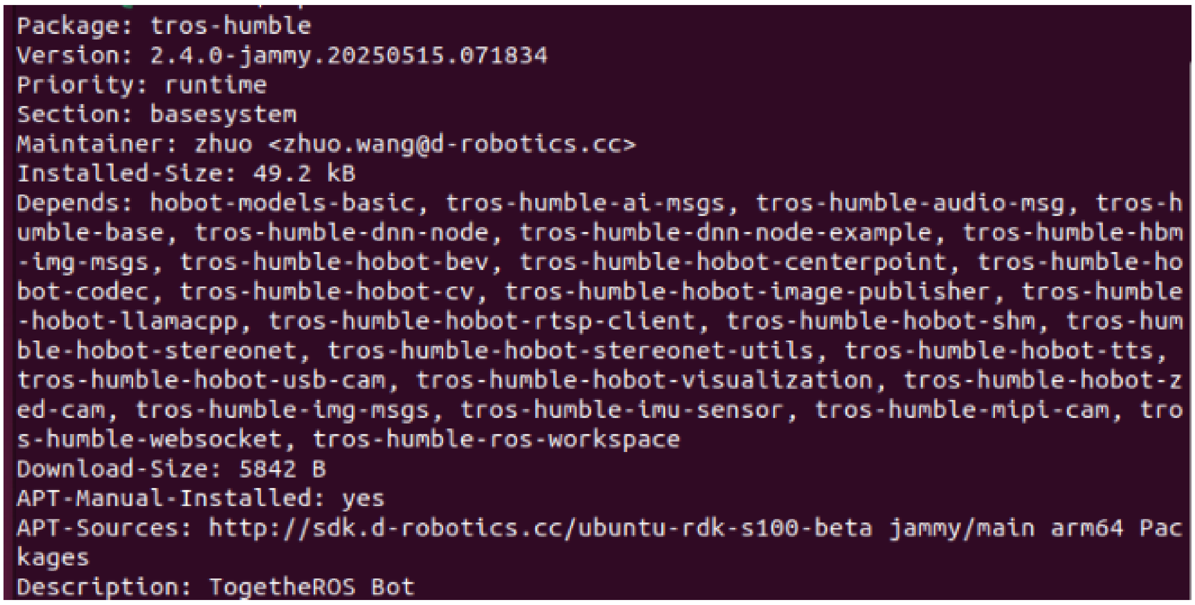
Upgrade tros.b deb package:

```
sudo apt update
sudo apt upgrade
```

Check the current tros.b version of Foxy

- Humble

  Open a terminal and type `apt show tros-humble` in the command line.


```
Package: tros-humble
Version: 2.4.0-jammy.20250515.071834
Priority: runtime
Section: basesystem
Maintainer: zhuo <zhuo.wang@d-robotics.cc>
Installed-Size: 49.2 kB
Depends: hobot-models-basic, tros-humble-ai-msgs, tros-humble-audio-msg, tros-h
umble-base, tros-humble-dnn-node, tros-humble-dnn-node-example, tros-humble-hbm
-img-msgs, tros-humble-hobot-bev, tros-humble-hobot-centerpoint, tros-humble-ho
bot-codec, tros-humble-hobot-cv, tros-humble-hobot-image-publisher, tros-humble
-hobot-llamacpp, tros-humble-hobot-rtsp-client, tros-humble-hobot-shm, tros-hum
ble-hobot-stereonet, tros-humble-hobot-stereonet-utils, tros-humble-hobot-tts,
tros-humble-hobot-usb-cam, tros-humble-hobot-visualization, tros-humble-hobot-z
ed-cam, tros-humble-img-msgs, tros-humble-imu-sensor, tros-humble-mipi-cam, tro
s-humble-websocket, tros-humble-ros-workspace
Download-Size: 5842 B
APT-Manual-Installed: yes
APT-Sources: http://sdk.d-robotics.cc/ubuntu-rdk-s100-beta jammy/main arm64 Pac
kages
Description: TogetheROS Bot
```

You can see that the current tros.b version has been upgraded to 2.4.0.

Note:

- The version number `version` displayed in the query is the actual version of `tros.b` installed. This example uses version `2.4.0`.
- For detailed release information on `tros.b`, please see [Release History].(https://d-robotics.github.io/rdk_doc/Robot_development/quick_start/changelog)。

2.2Install from source code

Prerequisites:

- The development machine can access the D-Robotics organization on GitHub.
- Docker is installed on the development machine.

Compiling tros.b

1 Using Dockerfiles

This part of the operation is all done on the development machine.

- Foxy
- Humble

```
## Create a directory
cd  /mnt/data/kairui.wang/test
mkdir -p cc_ws/tros_ws/src
## Obtain the Docker for cross-compilation
wget http://archive.d-
robotics.cc/TogetheROS/cross_compile_docker/pc_tros_ubuntu22.04_v1.0.0.tar.gz
## Load the Docker image
sudo docker load --input pc_tros_ubuntu22.04_v1.0.0.tar.gz
## Check the corresponding image ID for pc_tros
sudo docker images
## Launch Docker and mount the directory
sudo docker run -it --entrypoint="/bin/bash" -v PC local directory: Docker
directory imageID, here is an example using:
sudo docker run -it --entrypoint="/bin/bash" -v
/mnt/data/kairui.wang/test:/mnt/test 4cbdb9d61e19
```

2 Obtaining the tros.b Source Code

This part of the operation is all completed within the Docker container on the development machine.

Here, we take the /mnt/test directory in the Docker container as an example.

- Foxy
- Humble

```
cd /mnt/test/cc_ws/tros_ws
## Obtain the configuration file
git clone https://github.com/D-Robotics/robot_dev_config.git -b develop
## Execute cd robot_dev_config and use the "git tag --list" command to view the
available release versions
## Use the "git reset --hard [tag number]" command to specify the release
version. For detailed instructions, refer to the "Compile Specific Version
tros.b" section on this page
## Pull the source code
vcs-import src < ./robot_dev_config/ros2_release.repos
```

The directory structure of the entire project is as follows:

```
├── cc_ws
|   ├── sysroot_docker
|   |   ├── etc
|   |   ├── lib -> usr/lib
|   |   ├── opt
|   |   └── usr
|   └── tros_ws
|       ├── robot_dev_config
|       └── src
```

The `tros_ws/robot_dev_config` path contains the configuration and script files needed for code fetching, compilation, and packaging. The `tros_ws/src` path stores the fetched code. The `sysroot_docker` path contains the header files and libraries required for cross-compilation, corresponding to the `/` directory of the RDK. For example, the path for the media library in `sysroot_docker` is `sysroot_docker/usr/lib/hbmedia/`, while the path in the RDK is `/usr/lib/hbmedia/`.

During compilation, the installation path of `sysroot_docker` is specified through the `CMAKE_SYSROOT` macro in the `robot_dev_config/aarch64_toolchainfile.cmake` compilation script.

3 Cross-Compilation

All of these operations are performed inside the docker on the development machine.

```
## Compile tros.b version X3 using build.sh
bash ./robot_dev_config/build.sh -p X3

## Compile tros.b version RDK Ultra using build.sh.
bash ./robot_dev_config/build.sh -p Rdkultra

## Compile tros.b version X5 using build.sh
bash ./robot_dev_config/build.sh -p X5

## Compile tros.b version S100 using build.sh
bash ./robot_dev_config/build.sh -p S100
```

After successful compilation, a message will prompt: N packages compiled and passed.

If using minimal_build.sh for minimal compilation, you can further compress the deployment package size by executing `./minimal_deploy.sh -d "install_path"`.

Install tros.b

Copy the compiled directory to the RDK and rename it as tros. Here, we place the deployment package in the /opt/tros directory to be consistent with the deb installation directory.

Compile the specified version tros.b

In the section **Compile**, in the step 2 **Obtain the Code**, the default is to fetch the latest version of tros.b source code. If you need to get a specific release version of the source code, you need to make the following modifications:

```
## Get the configuration file
git clone https://github.com/D-Robotics/robot_dev_config.git -b develop
cd robot_dev_config
## View available release versions
git tag --list
## Switch to the specified version number, here we take tros.b 2.0.0 as an
example
git reset --hard tros_2.0.0
cd ..
## Pull code
vcs-import src < ./robot_dev_config/ros2_release.repos
```

3.Using ROS2 package

Prerequisite: TogetheROS.Bot installed successfully

The interfaces of tros.b and the ROS2 Foxy/Humble version are fully compatible and can reuse the ROS2 rich tool package. Here we take the installation and use of the ROS2 foxy version ros-foxy-image-transport as an example to introduce how to use the ROS package in tros.b.

3.1Installing ROS2 package

3.1.1When adding the ROS apt source s.b, the ROS apt source is automatically added, so there is no need to add it manually.

Update apt repository

```
sudo apt update
```

3.1.2 Install packages

- Foxy
- Humble

```
sudo apt install ros-humble-image-transport
sudo apt install ros-humble-image-transport-plugins
```

3.2Using ROS2 package

- Foxy
- Humble

```
source /opt/tros/humble/setup.bash
ros2 run image_transport list_transports
```

The running result is as follows, showing the supported image formats by image_transport package.

```
root@ubuntu:/opt/tros# ros2 run image_transport list_transports
Declared transports:
image_transport/compressed
image_transport/compressedDepth
image_transport/raw
image_transport/theora

Details:
----------
"image_transport/compressed"
 - Provided by package: compressed_image_transport
 - Publisher:
      This plugin publishes a CompressedImage using either JPEG or PNG
compression.

 - Subscriber:
      This plugin decompresses a CompressedImage topic.

----------
"image_transport/compressedDepth"
 - Provided by package: compressed_depth_image_transport
 - Publisher:
      This plugin publishes a compressed depth images using PNG compression.
 - Subscriber:
      This plugin decodes a compressed depth images.

----------
"image_transport/raw"
 - Provided by package: image_transport
 - Publisher:
      This is the default publisher. It publishes the Image as-is on the base
topic.

 - Subscriber:
      This is the default pass-through subscriber for topics of type
sensor_msgs/Image.

----------
"image_transport/theora"
 - Provided by package: theora_image_transport
 - Publisher:
      This plugin publishes a video packet stream encoded using Theora.

 - Subscriber:
      This plugin decodes a video packet stream encoded using Theora.
```