Using ROS For Linux

Using ROS for Linux

Robot Operating System (ROS) provides libraries and tools to help software developers create robot applications. It provides hardware abstraction, device drivers, libraries, visualizers, message-passing, package management, and more. This document shows how to install **arena_camera**, LUCID's ROS driver.

System Requirements

- Ubuntu 16.04 or Ubuntu 18.04
- ROS Kinetic for Ubuntu 16.04 or ROS Melodic for Ubuntu 18.04
- Arena SDK for Linux x64 v0.1.26 or higher
- Driver file: arena_camera_ros found at https://github.com/lucidvisionlabs/arena_camera_ros /releases/ (https://github.com/lucidvisionlabs/arena_camera_ros/releases/)
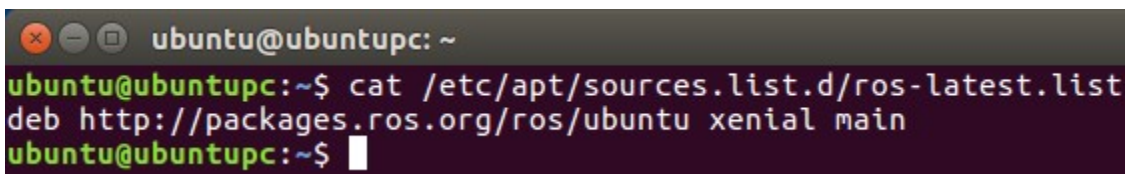
Table of Contents

Initial ROS Setup

1. Set up your system to acquire software from packages.ros.org.

```
$ sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) \
    main" > /etc/apt/sources.list.d/ros-latest.list'
```

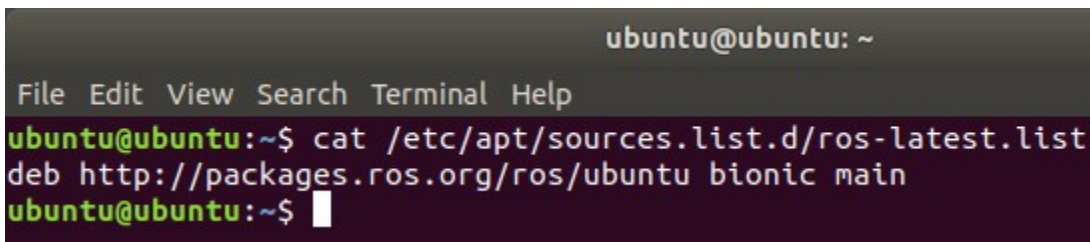You can confirm the new list file is set by checking the contents of `ros-latest.list`:

On Ubunto 16.04 (xenial):

(https://thinklucid.com/wp-content/uploads/2019/11/ROS-Ubuntu-16-image1.jpg)

On Ubuntu 18.04 (bionic):

(https://thinklucid.com/wp-content/uploads/2019/11/ROS-Ubuntu-18-image2.jpg)

2. Use apt-key to install the Open Robotics key to your list of trusted keys.

```
$ sudo apt-key adv --keyserver 'hkp://keyserver.ubuntu.com:80' --recv-key C1C
F6E31E6BADE8868B172B4F42ED6FBAB17C654
```

3. Install ROS Desktop. The following commands will install ROS onto the system.

On Ubuntu 16.04 (xenial):

```
$ sudo apt-get update
$ sudo apt-get install ros-kinetic-desktop-full
```

On Ubuntu 18.04 (bionic):

```
$ sudo apt-get update
$ sudo apt-get install ros-melodic-desktop-full
```

4. Initialize rosdep. This will setup system dependencies for ROS.

```
$ sudo apt-get install python-rosdep
$ sudo rosdep init
$ rosdep update
```

5. Setup ROS environment variables.

   On Ubuntu 16.04 (xenial):

```
$ echo "source /opt/ros/kinetic/setup.bash" >> ~/.bashrc
$ source ~/.bashrc
```

   On Ubuntu 18.04 (bionic):

```
$ echo "source /opt/ros/melodic/setup.bash" >> ~/.bashrc
$ source ~/.bashrc
```

6. Install ROS package workspace dependencies. This will allow you to create and manage your own ROS workspaces, including the ROS workspace used by arena_camera.

```
$ sudo apt-get install python-rosinstall python-rosinstall-generator python-w
stool build-essential
```

Arena SDK Setup

1. Download and install Arena SDK. Arena SDK for Linux is available on the Downloads page at https://thinklucid.com/downloads-hub/ (https://thinklucid.com/downloads-hub/). The Arena SDK dependencies are listed in the README file included with the SDK.

> Note: These instructions will assume Arena SDK is installed at
> ~/**ArenaSDK_Linux_x64**:
>
> ```
> $ tar -xvf ArenaSDK_v0.1.26_Linux_x64.tar.gz -C ~
> ```
>
> ⓘ Please remember to go over the Getting Started steps discussed in the Arena SDK
> documentation and run the Arena_SDK.conf file in the Arena_SDK directory:
>
> ```
> $ cd ~/ArenaSDK_Linux_x64/
> $ sudo sh Arena_SDK.conf
> ```

2. Set up your **ARENA_ROOT** environment variable. This environment variable should be the path
   where you have installed Arena SDK.

```
$ echo "export ARENA_ROOT=~/ArenaSDK_Linux_x64" >> ~/.bashrc
$ source ~/.bashrc
```

You can confirm the variable is set with the typing echo $ARENA_ROOT in the terminal.

(https://thinklucid.com/wp-content/uploads/2019/11/arena_root_ros.jpg)

Creating a ROS workspace for the arena_camera ROS Driver

1. Set up your system to acquire software from packages.ros.org.

> Note: These instructions will assume catkin_ws is located at **~/catkin_ws**:
>
> ⓘ
> ```
> $ unzip arena_camera_ros-1.2.zip -d ~
> $ mv ~/arena_camera_ros-1.2/catkin_ws ~
> ```

2. Set up your **ARENA_CONFIG_ROOT** environment variable. This is also the path to the ROS

workspace used by arena_camera.

```
$ echo "export ARENA_CONFIG_ROOT=~/catkin_ws" >> ~/.bashrc
$ source ~/.bashrc
```

Building and Running the arena_camera ROS Driver

1. Navigate to your arena_camera ROS workspace

```
$ cd ~/catkin_ws
```

1. Copy the included **image_encoding.h** to your ROS include folder

On Ubuntu 16.04 (xenial):

```
$ sudo cp /opt/ros/kinetic/include/sensor_msgs/image_encodings.h /opt/ros/kin
etic/include/sensor_msgs/image_encodings.h.bak
$ sudo cp inc/image_encodings.h /opt/ros/kinetic/include/sensor_msgs/image_en
codings.h
```

On Ubuntu 18.04 (bionic):

```
$ sudo cp /opt/ros/melodic/include/sensor_msgs/image_encodings.h /opt/ros/mel
odic/include/sensor_msgs/image_encodings.h.bak
$ sudo cp inc/image_encodings.h /opt/ros/melodic/include/sensor_msgs/image_en
codings.h
```

> Note: A custom image_encoding.h is included to enable streaming support for
> LUCID's Helios camera.

3. Build arena_camera

```
$ catkin_make
```

The above command will build:

- **arena_camera:**   The Arena SDK camera code for ROS.

- **arena_camera_node:**   The ROS node for arena_camera. See
  http://wiki.ros.org/ROS/Tutorials/UnderstandingNodes
  (http://wiki.ros.org/ROS/Tutorials/UnderstandingNodes)
  for further information.

```
[ 95%] Building CXX object arena_camera/CMakeFiles/arena_camera.dir/src/arena_camera/binary_exposure_search.cpp.o
[ 96%] Building CXX object arena_camera/CMakeFiles/arena_camera.dir/src/arena_camera/encoding_conversions.cpp.o
[ 96%] Building CXX object arena_camera/CMakeFiles/arena_camera.dir/src/arena_camera/arena_camera.cpp.o
[ 97%] Building CXX object arena_camera/CMakeFiles/arena_camera.dir/src/arena_camera/arena_camera_node.cpp.o
[ 97%] Building CXX object arena_camera/CMakeFiles/arena_camera.dir/src/arena_camera/arena_camera_parameter.cpp.o
[ 98%] Linking CXX shared library /home/ubuntu/catkin_ws/devel/lib/libarena_camera.so
[ 98%] Built target arena_camera
Scanning dependencies of target arena_camera_node
[100%] Building CXX object arena_camera/CMakeFiles/arena_camera_node.dir/src/arena_camera/main.cpp.o
[100%] Linking CXX executable /home/ubuntu/catkin_ws/devel/lib/arena_camera/arena_camera_node
[100%] Built target arena_camera_node
ubuntu@ubuntupc:~/catkin_ws$
```

(https://thinklucid.com/wp-content/uploads/2019/11/build-arena-camera.jpg)

4. Run roscore to start the ROS server

```
$ roscore
```

```
ubuntu@ubuntupc:~/catkin_ws$ roscore
... logging to /home/ubuntu/.ros/log/203027b4-b2fe-11e9-b548-0800278aafa7/roslaunch-ubuntupc-18688.log
Checking log directory for disk usage. This may take awhile.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://ubuntupc:41957/
ros_comm version 1.12.14


SUMMARY
========

PARAMETERS
 * /rosdistro: kinetic
 * /rosversion: 1.12.14

NODES

auto-starting new master
process[master]: started with pid [18698]
ROS_MASTER_URI=http://ubuntupc:11311/

setting /run_id to 203027b4-b2fe-11e9-b548-0800278aafa7
process[rosout-1]: started with pid [18711]
started core service [/rosout]
```

(https://thinklucid.com/wp-content/uploads/2019/11/roscore.jpg)

> ⓘ   Note: Only one roscore can be run at a time.

5. Run **arena_camera** as a node in a new terminal. The first command listed below will overlay your
   workspace on top of your ROS environment. The second command will connect to the camera
   and the camera will start imaging. The received images will be published to a topic. See
   http://wiki.ros.org/ (http://wiki.ros.org/)Topics and http://wiki.ros.org/ROS/Tutorials
   /UnderstandingTopics (http://wiki.ros.org/ROS/Tutorials/UnderstandingTopics) for further
   information.

```
$ source ~/catkin_ws/devel/setup.bash
$ rosrun arena_camera arena_camera_node
```


(https://thinklucid.com/wp-content/uploads/2019/11/arena_camera-ROS.jpg)

6. View the raw images by opening a new terminal and subscribing the image_view node to topic
   name image_raw.

```
$ rosrun image_view image_view image:=/arena_camera_node/image_raw
```

(https://thinklucid.com/wp-content/uploads/2019/11/Image-RAW-ROS.jpg)

> ⓘ Note: Streaming with 12-bit pixel formats are not supported in this version of
> arena_camera. Depth visualization support is not included with this version of
> arena_camera. You will see a black image when viewing a Coord3D pixel format
> with the image_view node.

Using rosparam to Set Camera Parameters

1. ROS includes the rosparam tool which can be used to set camera parameters before starting arena_camera.

   To **set** a parameter:

   ```
   $ rosparam set /arena_camera_node/<parameter_name> value
   $ rosparam set /arena_camera_node/frame_rate 10
   $ rosparam set /arena_camera_node/image_encoding mono8
   ```

   To **list** of parameters that are already set:

```
$ rosparam list /arena_camera_node
```

To **view** a set parameter's value:

```
$ rosparam get /arena_camera_node/<parameter_name>
$ rosparam get /arena_camera_node/frame_rate
```

To **delete** a set parameter:

```
$ rosparam delete /arena_camera_node/<parameter_name>
$ rosparam delete /arena_camera_node/frame_rate
```

Other rosparam commands

Command	Description
rosparam dump	Dump current parameters to a file
rosparam load	Load parameters from a file

> Note: If arena_camera is already running, you must stop and start arena_camera for new or changed parameters to take effect.

Supported arena_camera_node Parameters

Parameter	Description
device_user_id	Device User ID (user defined name)
frame_rate	Frame Rate (Hz)
image_encoding	Pixel Format
binning_x	Binning parameter for X axis (must match binning_y)
binning_y	Binning parameter for Y axis (must match binning_x)

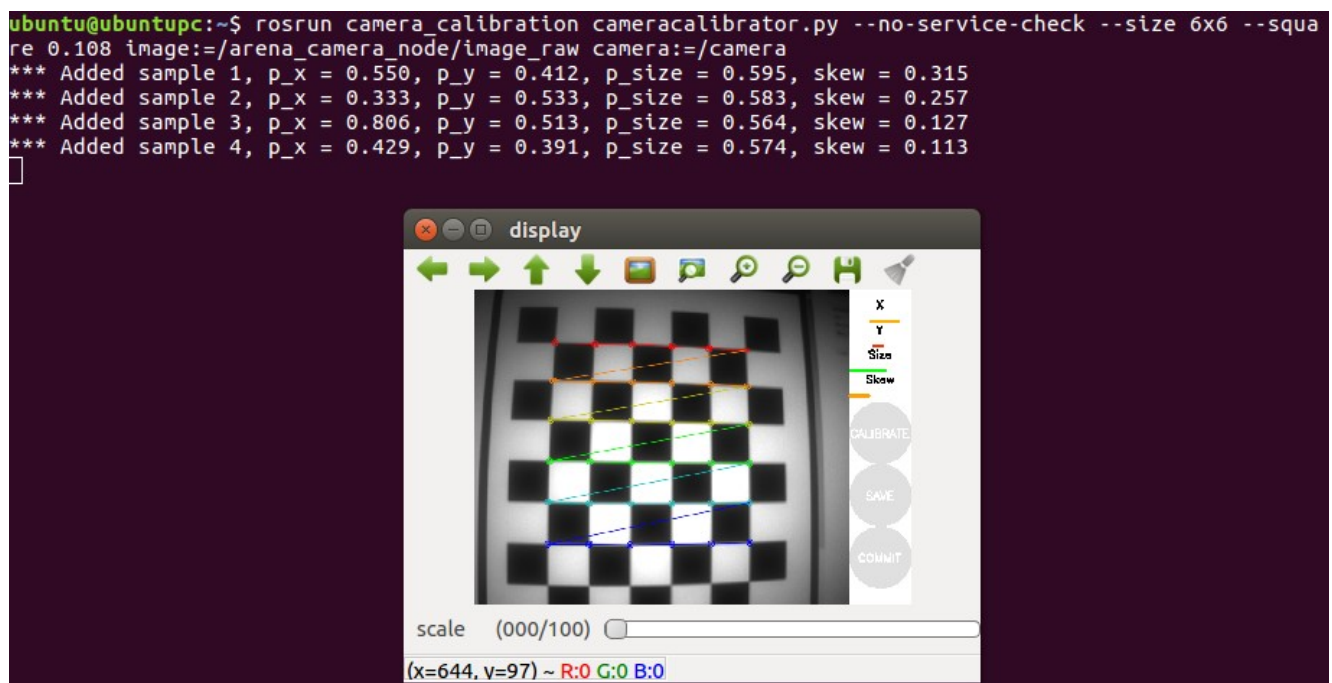| exposure	Exposure Time (microseconds)
gain	Gain (dB)
gamma	Gamma correction of pixel intensity/td>
exposure_auto	Auto Exposure control (true/false)
gain_auto	Auto Gain control (true/false)

ROS camera_calibration Node

An optional step is to calibrate a single camera using the ROS camera_calibration node. This will assist in calibrating a monocular camera from the image_raw topic. See http://wiki.ros.org/camera_calibration/Tutorials/MonocularCalibration (http://wiki.ros.org /camera_calibration/Tutorials/MonocularCalibration) for more detailed information.

Once you have installed camera_calibration with rosdep, you can use it with arena_camera:

```
$ rosrun camera_calibration cameracalibrator.py   --no-service-check --size 8x6
--square 0.108 image:=/arena_camera_node/image_raw camera:=/camera
```

where **8×6** is the number of vertices in your checkerboard pattern.

(https://thinklucid.com/wp-content/uploads/2019/11/8x6.jpg)

Once there is enough data for calibration, click Calibrate to start the calibration process and click Save to save the calibration data to a file.

> ℹ️ Note: You will not be able to Commit the calibration data onto the camera.

Once the calibration data is saved to file, you can extract the calibration tarball contents and apply the calibration to the camera using **rosparam**. The following steps assume the calibration data is extracted to **~/calibration**.

```
$ mkdir ~/calibration

$ tar -xvf /tmp/calibrationdata.tar.gz -C ~/calibration
$ rosparam set /arena_camera_node/camera_info_url "file://home/ubuntu/calibrati
on/ost.yaml"
$ rosrun arena_camera arena_camera_node
```

```
ubuntu@ubuntupc:~/catkin_ws$ rosrun arena_camera arena_camera_node
[ WARN] [1564522965.991886744]: Autoflash: 0, line2: 1 , line3: 1
[ INFO] [1564522965.992009953]: No Device User ID set -> Will open the camera device found first
[ INFO] [1564522966.643079874]: camera calibration URL: file:///home/ubuntu/calibration/ost.yaml
```

(https://thinklucid.com/wp-content/uploads/2019/11/rosparam.jpg)

© 2020 LUCID Vision Labs Inc.

Looking to purchase our cameras?
Visit the LUCID Webstore at thinklucid.com (https://thinklucid.com)