Stream Publisher Package

Stream Publisher Package

Description:

This page discusses the *stream_publisher_package* in detail. This package uses the Arena SDK and ROS2 to start an image stream and publish image messages over a topic.

This package is an adaptation of the ROS2 driver developed by LUCID Vision. Links to those resources are listed below:

- LUCID Vision ROS2 Documentation
- LUCID Vision Labs Github

Note that there is a "Trigger" mode in both the driver provided by LUCID Vision and the driver provided in this package. I've never used that functionality, and have solely used this package to publish image streams and record bag files.

Requirements:

- Ubuntu Focal Fossa
- ROS2 Foxy Fitzroy
- Arena SDK for Linux
- C++17 or higher

Configuration and Launch Files

Before using this package, make sure ALL paths in the configuration and launch files are set correctly. This will especially cause issues when cloning the repository to a new machine as the paths cloned from the remote repo are (most likely) not valid for the new local repo.

Note: All paths are set globally. Paths are also rarely set outside of configuration/launch files. This allows the user to point to different files in a more automated manner. Editing the configuration file DOES NOT require you to rebuild the package, but editing the launch file DOES!

***Configuration File*:**
`stream_config.yaml` is shown below:

src > stream_publisher_package > config > ! stream_config.yaml

```yaml
tri028s_cc:
  stream_publisher:
    ros__parameters:
      serial: "223600392"
      pixelformat: "rgb8"
      width: 1280 #1936 #1440 #1920
      height: 720 #1464 #1080  #1080
      gain: 40.0
      exposure_time: 4000.0 #-1.0
      stream_auto_negotiate_packet_size: true
      stream_packet_resend_enable: true
      tigger_mode: false
      topic: "stream"
      qos_history: ""
      qos_history_depth: 0
      qos_reliability: "reliable"

#tri028s_mc:
  #stream_publisher:
    #ros__parameters:
      #serial: "223301667"
      #pixelformat: "mono8"
      #width: 1936 #1280 #1440 #1920
      #height: 1464 #720 #1080  #1080
      #gain: 20.0
      #exposure_time: 4000.0 #-1.0
      #stream_auto_negotiate_packet_size: true
      #stream_packet_resend_enable: true
      #tigger_mode: false
      #topic: "stream"
      #qos_history: ""
      #qos_history_depth: 0
      #qos_reliability: "reliable"

#tri054s_mc:
  #stream_publisher:
    #ros__parameters:
      #serial: "223302308"
      #pixelformat: "mono8"
      #width: 1936 #1280 #1440 #1920
      #height: 1464 #720 #1080  #1080
      #gain: 20.0
      #exposure_time: 4000.0 #-1.0
      #stream_auto_negotiate_packet_size: true
      #stream_packet_resend_enable: true
      #tigger_mode: false
      #topic: "stream"
      #qos_history: ""
```

There are no paths set in this config file, only camera nodes. Note that only one camera can be up and running at one time. Switching between cameras will require reconfiguration (you can also create different config files for each camera and point to them accordingly - this may be an easier solution than commenting out).

ROS2 Parameters Configured:

- **serial:** Serial number of the camera
- **pixelformat:** Pixel format of the camera
- **width:** Desired image width resolution
- **height:** Desired image height resolution
- **gain:** Desired gain
- **exposure_time:** Desired exposure time (4000 recommended for faster fps)

- **stream_auto_negoiate_packet_size:** A node for setting dynamic packet sizes (true recommended for faster fps)
- **stream_packet_resend_enable:** Another node to handle packets - not actually sure what it does (true recommended for faster fps)
- **trigger_mode:** A functionality to publish frames manually instead of over stream (usually set to false)
- **topic:** Desired topic to publish image messages to
- **qos_history:** Quality of service parameter (usually set to "")
- **qos_history_depth:** Quality of service parameter (usually set to 0)
- **qos_relability:** Quality of service parameter (usually set to "reliable")

***Launch File*:**
`stream_launch.py` is shown below:

src > stream_publisher_package > launch > stream_launch.py > generate_launch_description

```python
from launch import LaunchDescription
from launch_ros.actions import Node

def generate_launch_description():

    config_path = "/home/tahnt/T3_Repos/camera_packages/ros2_ws/src/stream_publisher_package/config/stream_config.yaml"

    return LaunchDescription([
        Node(
            package="stream_publisher_package",
            executable="start_stream",
            namespace="tri028s_cc",
            name="stream_publisher",
            output="screen",
            emulate_tty=True,
            parameters=[config_path]
        ),
    ])
```

Typically, the launch file does not need to be edited often. Make sure **config_path** correctly points to `stream_config.yaml` . If streaming a different camera is desired, the configuration must be adjusted and the **namespace** indicated correctly (Or create different config files as mentioned earlier)!

Using the Package

Once the camera nodes are set accordingly, follow the steps below to start streaming/publishing image and recording a bag file (also include in `README.md`):

Before Use:

- **Make sure your ethernet configurations are set to maximize fps! For more information, check the Jopling documentation under the *Getting Started* page.**
- **Make sure ALL PATHS ARE SET CORRECTLY in the launch and config files before use!**
- **These steps assume you have already created a workspace folder and a `/src` directory within it!**

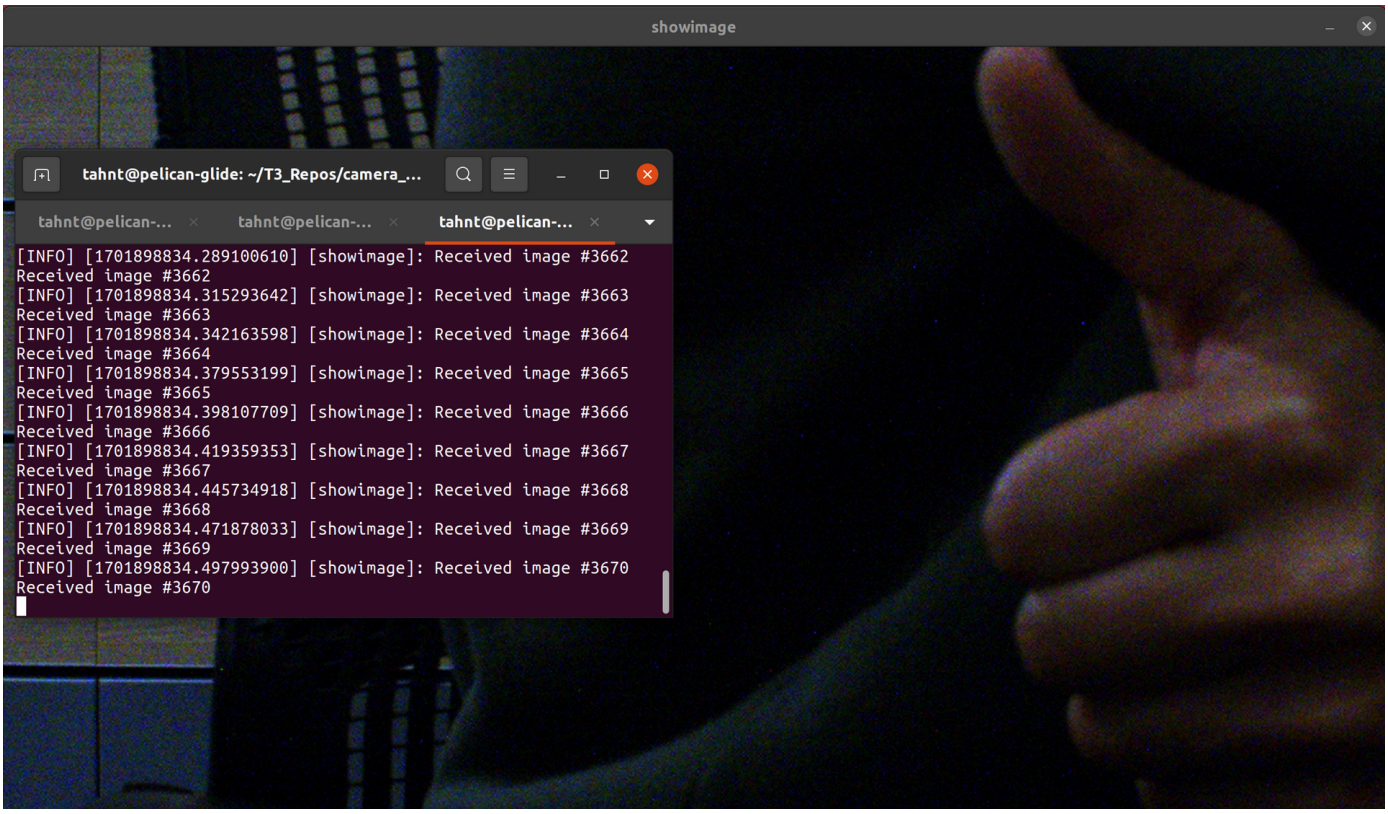
***Stream/Publish Steps*:**

1. Navigate into the `\src` directory of your workspace and clone the repo using `git clone`
2. Navigate back into the workspace directory and source `$ source /opt/ros/foxy/setup.bash`
3. Build package `$ colcon build` or `$ colcon build --packages-select <package_name>`
4. Open a new terminal and source it `$ . install/setup.bash`
5. Run launch file `$ ros2 launch <package_name> <launch_file_name>` in this case it is `$ ros2 launch stream_publisher_package stream_launch.py`. The terminal should output the following (or similar):

Note: The terminal displays the camera nodes set, and starts logging each image published

```
tahnt@pelican-glide:~/T3_Repos/camera_packages/ros2_ws$ ros2 launch stream_publisher_package stream_launch.py
[INFO] [launch]: All log files can be found below /home/tahnt/.ros/log/2023-12-06-13-36-09-812421-pelican-glide-39605
[INFO] [launch]: Default logging verbosity is set to INFO
[INFO] [start_stream-1]: process started with pid [39608]
[start_stream-1] [INFO] [1701898569.987389771] [tri028s_cc.stream_publisher]: Creating "stream_publisher" node
[start_stream-1] [INFO] [1701898569.987800923] [tri028s_cc.stream_publisher]:
[start_stream-1] serial: 223600392
[start_stream-1] [INFO] [1701898569.987828070] [tri028s_cc.stream_publisher]: pixelformat: rgb8
[start_stream-1] [INFO] [1701898569.987840820] [tri028s_cc.stream_publisher]: width: 1280
[start_stream-1] [INFO] [1701898569.987851617] [tri028s_cc.stream_publisher]: height: 720
[start_stream-1] [INFO] [1701898569.987862556] [tri028s_cc.stream_publisher]: gain: 40.000000
[start_stream-1] [INFO] [1701898569.987879259] [tri028s_cc.stream_publisher]: exposure_time: 4000.000000
[start_stream-1] [INFO] [1701898569.987892804] [tri028s_cc.stream_publisher]: stream_auto_negotiate_packet_size: 1
[start_stream-1] [INFO] [1701898569.987904778] [tri028s_cc.stream_publisher]: stream_packet_resend_enable: 1
[start_stream-1] [INFO] [1701898569.987914695] [tri028s_cc.stream_publisher]: trigger_mode: 0
[start_stream-1] [INFO] [1701898569.987924574] [tri028s_cc.stream_publisher]: topic: stream
[start_stream-1] [INFO] [1701898569.987934106] [tri028s_cc.stream_publisher]: qos_history:
[start_stream-1] [INFO] [1701898569.987944783] [tri028s_cc.stream_publisher]: qos_history_depth: 0
[start_stream-1] [INFO] [1701898569.987955385] [tri028s_cc.stream_publisher]: qos_reliability: reliable
[start_stream-1]
[start_stream-1] [INFO] [1701898570.060279401] [tri028s_cc.stream_publisher]:   QoS history     = keep_last
[start_stream-1]                                 QoS depth       = 5
[start_stream-1]                                 QoS reliability = reliable
[start_stream-1]
[start_stream-1] [INFO] [1701898570.060431115] [tri028s_cc.stream_publisher]: Created "stream_publisher" node
[start_stream-1] [INFO] [1701898571.376191762] [tri028s_cc.stream_publisher]: 1 arena device(s) has been discoved.
[start_stream-1] [INFO] [1701898572.341465244] [tri028s_cc.stream_publisher]: device created 1c:0f:af:04:33:5b | 223600392 |
TRI028S-C | 169.254.92.51 |  |
[start_stream-1] [INFO] [1701898572.426319118] [tri028s_cc.stream_publisher]:   default profile is loaded
[start_stream-1] [INFO] [1701898572.455476047] [tri028s_cc.stream_publisher]:   ROI set to 1280X720
[start_stream-1] [INFO] [1701898572.483363425] [tri028s_cc.stream_publisher]: OffsetX: 328
[start_stream-1] [INFO] [1701898572.483499201] [tri028s_cc.stream_publisher]: OffsetY: 372
[start_stream-1] [INFO] [1701898572.493833896] [tri028s_cc.stream_publisher]:   Gain set to 40.000000
[start_stream-1] [INFO] [1701898572.504429612] [tri028s_cc.stream_publisher]:   PixelFormat set to RGB8
[start_stream-1] [INFO] [1701898573.288122032] [tri028s_cc.stream_publisher]: image 1 published to stream
[start_stream-1] [INFO] [1701898573.306733101] [tri028s_cc.stream_publisher]: image 2 published to stream
[start_stream-1] [INFO] [1701898573.332827133] [tri028s_cc.stream_publisher]: image 3 published to stream
[start_stream-1] [INFO] [1701898573.359224200] [tri028s_cc.stream_publisher]: image 4 published to stream
[start_stream-1] [INFO] [1701898573.385419705] [tri028s_cc.stream_publisher]: image 5 published to stream
[start_stream-1] [INFO] [1701898573.411563337] [tri028s_cc.stream_publisher]: image 6 published to stream
[start_stream-1] [INFO] [1701898573.437765459] [tri028s_cc.stream_publisher]: image 7 published to stream
[start_stream-1] [INFO] [1701898573.463934225] [tri028s_cc.stream_publisher]: image 8 published to stream
[start_stream-1] [INFO] [1701898573.490237037] [tri028s_cc.stream_publisher]: image 9 published to stream
[start_stream-1] [INFO] [1701898573.516313377] [tri028s_cc.stream_publisher]: image 10 published to stream
[start_stream-1] [INFO] [1701898573.542627232] [tri028s_cc.stream_publisher]: image 11 published to stream
```

6. To view the stream open a new terminal and source it (same as step 4), and run `image_tools` via `$ ros2 run image_tools showimage --ros-args --remap image:=<topic_name>`



Note: `<topic_name>` also includes the namespace and can be found using `$ ros2 topic list` in a sourced terminal (as shown below):

```
tahnt@pelican-glide:~/T3_Repos/camera_packages/ros2_ws$ ros2 topic list
/parameter_events
/rosout
/tri028s_cc/stream
tahnt@pelican-glide:~/T3_Repos/camera_packages/ros2_ws$
```

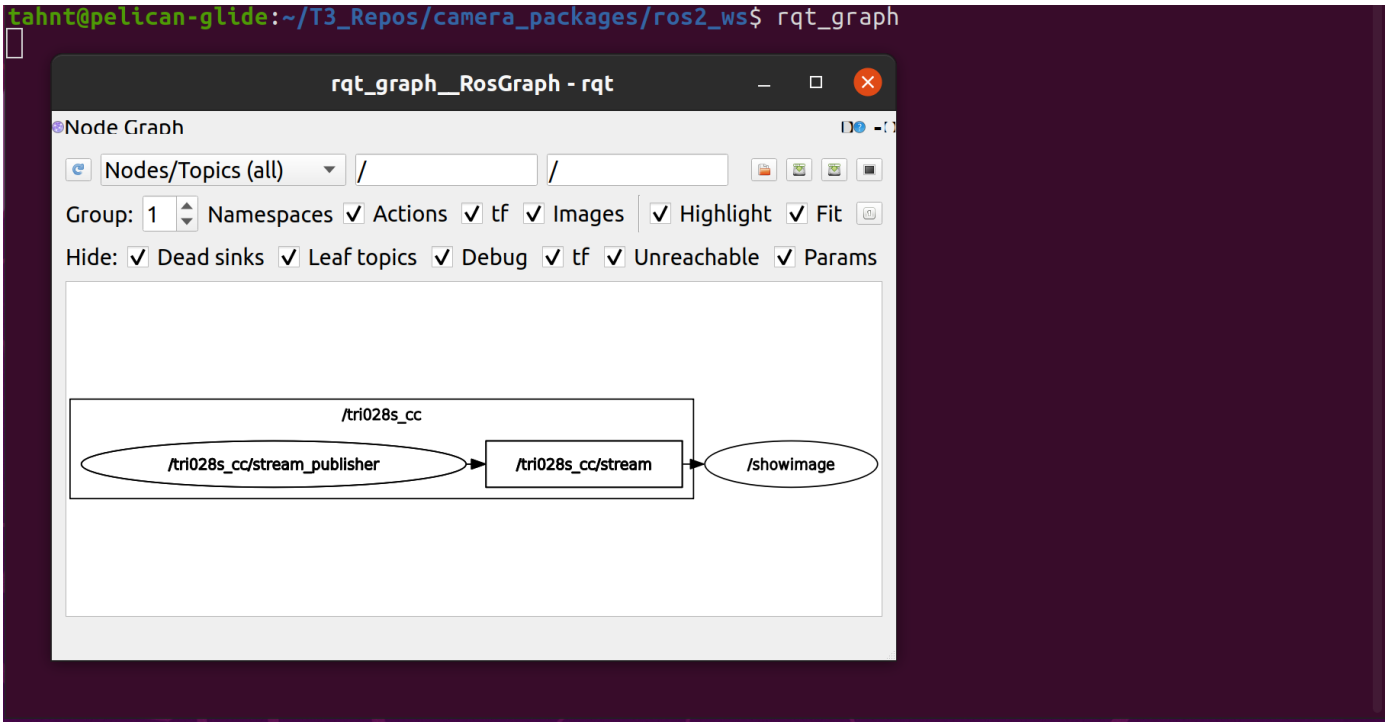
where **tri028s_cc** is the namespace and **stream** is the specified topic name.

***Recording a ROS bag Steps*:**

1. Make sure you are publishing images using the previous steps
2. Create the desired directory to record bags, navigate to it, and source it using `$source /opt/ros/foxy/setup.bash`
3. Record bag using `$ros2 bag record <topic_name> -o <bag_name>`. Where `<bag_name>` can be specified by user choice.

Additional Validation

Checking rqt_graph is a good way to check that the node is working properly:



From this, we can see the the **stream_publisher** node is publishing image messages to the **stream** topic under the **tri028s_cc** namespace, and the **showimage** is subscribed to the topic!