Cameras

Overview

- The cameras that we use for the sub are Point Grey Flea3 GigE Vision ethernet cameras. Specs and downloads are available here. Flea3
- The cameras come with an SDK which contains configuration software and programmable interfaces for the camera. You can access it on their website.

Datasheets

basic_specs-flea3-gige.pdf

fl3-ge-imaging-performance.pdf

fl3-ge_gettingstarted.pdf

flea3-ge-technical-reference.pdf

First Time Setup

- Since the cameras are ethernet cameras the only thing you need for setup is an ethernet connection and a 12V power supply. The cameras take a few seconds to boot up and usually give themselves a default IP that is not on the network. You can use the flycap software to detect the cameras and add them to the network.

Configuration

- Before you edit any camera settings, be sure to read the ENTIRE guide on the flycap configuration software as well as the IEEE-1394 camera standard. These documents give great insight into the design, terminology, and typical settings of the cameras.
- Note, that the Flea3 cameras are somewhat tricky to configure correctly. Using factory settings, the cameras will most likely cause image inconsistency errors due to packet loss. Since most ethernet packet sizes are not suitable for streaming, it is almost necessary to increase packet size to jumbo (9000 bytes) in order to get the smoothest streaming possible.
- On Windows this can be done in Network Devices by configuring the driver to have jumbo packets. In Linux this can be configured with: sudo ifconfig eth0 mtu 9000.
- The packet delay should also be changed to account for the packet size increase. It should be the lowest delay that does not cause consistency errors depending on the setup. This can be tested by configuring the delay and running the cameras for about 5 minutes. If the cameras report no loss then those settings are optimal.
- Be aware that the jumbo packet size cannot usually be maxed out. (On my system I could only get to ~7000 byte packets before the stream would freeze) In order to get optimal settings you

have to play with the packet size and delay until you get the highest estimated bandwidth without errors.

Serial Numbers

- Left:
- Right: 14406634

Camera Settings

- The cameras can use multiple encodings depending on bandwidth and desired end types. For low bandwidth setups, Raw8 is the best encoding. You can configure post processing to convert the Raw8 into RGB8. Rigorous will return the best results. For medium bandwidth YUV422 returns great results, and for high bandwidth RGB8 is best as it requires no additional conversion once it reaches the computer.
- Since the cameras use orthographical fisheye lenses, the best image would be a square centered on the image circle. Since the Flea3 only supports resolutions up to 1032p, the best image is 1032×1032 centered. (I guessed around 162px right)
- Since conditions can change at the competition, it seems it is best to allow the cameras to auto adjust exposure, gain, and whitebalance. Although this makes the vision input somewhat unpredictable, you can stamp each frame with this information in the top left hand corner. The flycap SDK can save it once it reaches the computer if you want to analyze the footage later and figure out what settings the camera has at that point in time.

Problems / Fixes

- PGR cameras have issues on linux with large packet types. In order to fix this the buffer needs to be increased.

Linux Streaming Fix:

Derived from here: Linux Streaming Fix

CAUSE: When streaming images from a GigE Vision camera on Linux Ubuntu 8.04 systems, a high number of lost data packets may be observed. In FlyCapture SDK applications, dropped packets result in IMAGE_CONSISTENCY_ERRORS returned.

ANSWER:

To fix, try one or both of the following:

Increase packet delay time using the FlyCapture2 API or the FlyCap2 program. Increase the amount of memory Linux uses for receive buffers using the sysctl interface. Whereas the system standard (default) and maximum values for this buffer default to 128 KB and 120 KB respectively, increasing both of these parameters to 1 MB significantly improves image streaming results. Note: On some ARM boards, you may need to increase the receive buffer size to greater than 1 MB before noticing improved streaming results. Increasing the buffer size can enhance receive performance, but it also

uses more memory.

The following sysctl command updates the receive buffer memory settings:

```
sudo sysctl -w net.core.rmem_max=1048576 net.core.rmem_default=1048576
```

Note: In order for these changes to persist after system reboots, the following lines must be manually added to the bottom of the /etc/sysctl.conf file:

```
net.core.rmem_max=1048576
net.core.rmem_default=1048576
```

Once changes are persisted, they can be reloaded at any time by running the following command in sysctl:

```
sudo sysctl -p
```

Running With ROS

Point Grey supplies drivers for our Gig E cameras. You will need to have installed them:

```
sudo aptitude install ros-indigo-pointgrey-camera-driver
```

If you would like to simply enumerate the cameras connected to the network, run:

```
rosrun pointgrey_camera_driver list_cameras
```

Sometimes the cameras are a little fiddly and you may need to cycle power to them before they show up on this list. After you have confirmed that any and all cameras you would like to use are connected, you can manually begin publishing of the images from each camera:

```
roslaunch pointgrey_camera_driver camera.launch
```

This can become a little tedious after a while, so the current Robosub repository contains a launch file of its own. This can be used as follows:

```
roslaunch robosub cameras.launch
```

The cameras should begin publishing on the **/camera/[left|right|bottom]/image** topic. If you wish to change the serial number used for the left/right camera simply remap it when launching.

```
roslaunch robosub cameras.launch left_serial:=12345678
right_serial:=12345679
```

In the future, a downward facing camera is also planned to be added, though many of the steps will be similar.

One more note, when using the Point Grey drivers, the cameras will publish a WFOVImage message so many standard ros systems may not be able to use the data without it being republished. A

republisher has been implemented in the Robosub repository and can be run with the following command:

```
rosrun robosub camera_repub
```

This will republish the WFOVImage message messages as sensor_msgs/Image messages on the **/camera/[left|right|bottom]/undistorted** topic which aligns with the undistortion nodes so nodes are agnostic as to whether or not undistortion is being performed.