Intelligence at the Speed of Light™

Cepton ROS Driver and Matlab API

Cepton Webinar Series #3

2022-03-16

Agenda

ROS2 Driver

- Installation, compilation
- Cepton Publisher and Subscriber Nodes
- Message formats and available ROS topics
- Driver internals and command-line options

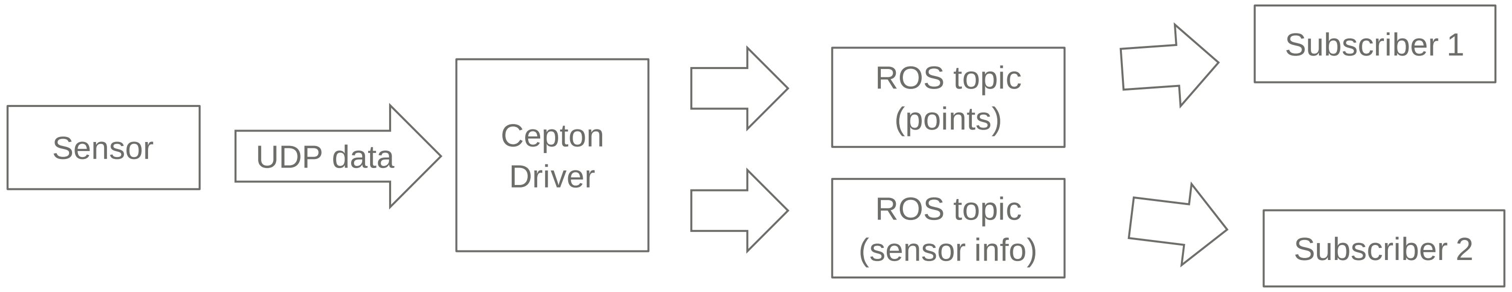
Matlab API

- Frame FIFO facility
- Data structures
- Code samples

Cepton is hiring interns and new college grads. Check out our LinkedIn or Handshake job page

Why ROS

- A distributed framework of processes (Nodes)
  - Nodes can be designed independently and then coupled at runtime
  - http://wiki.ros.org/ROS/Introduction
- Accumulate data from different devices
- Cepton driver handles the conversion from SDK sensor data stream into ROS topics



Cepton is hiring interns and new college grads. Check out our LinkedIn or Handshake job page

Publisher and Subscriber

Publisher:

- Receives the point cloud from the SDK, either via receiving UDP packets or by reading from a pcap
- Converts to ROS message formats
- Publishes over different topic names
- Continuously broadcasts data

Subscriber

- Sets up callbacks to the different ROS topics, and listens for the data

Cepton is hiring interns and new college grads. Check out our LinkedIn or Handshake job page

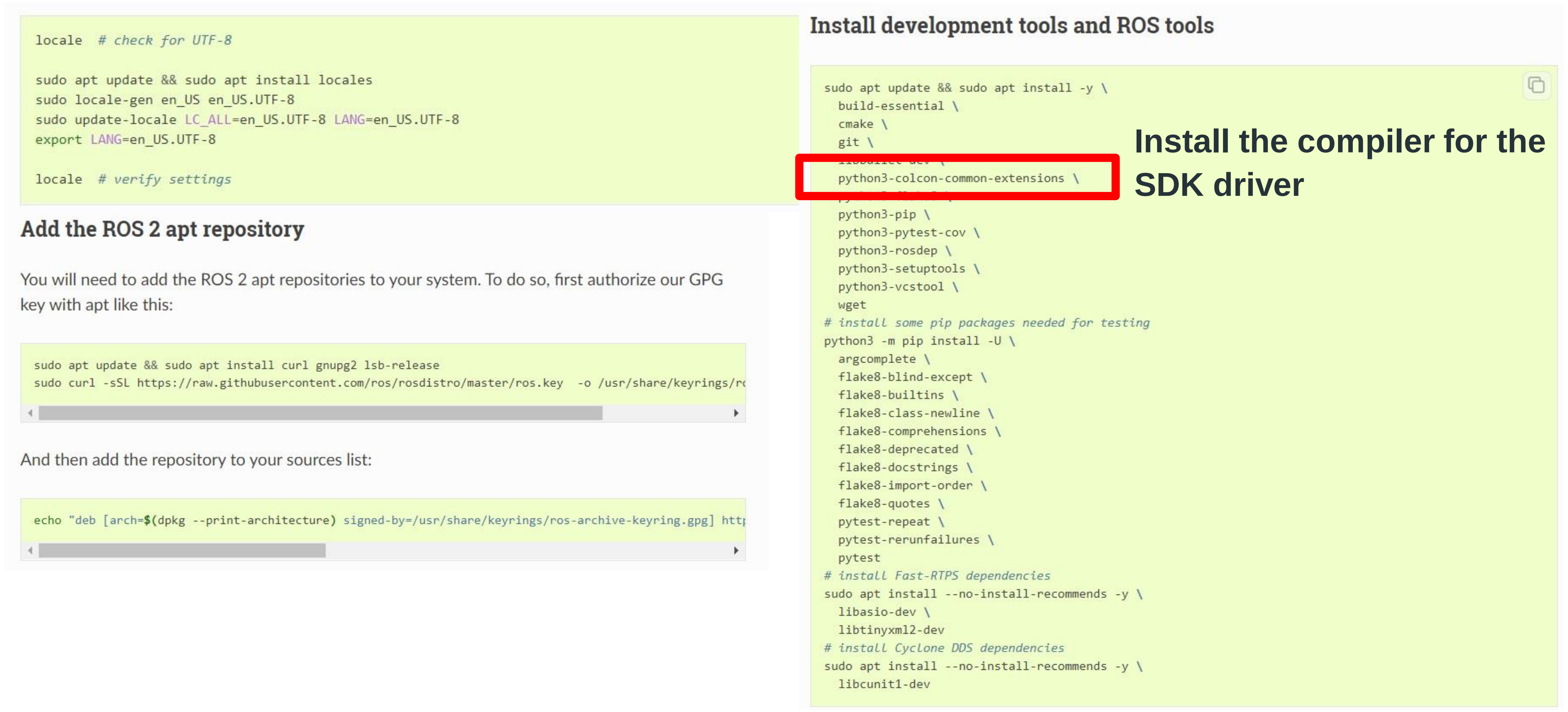
Overview

- Support Ubuntu 18 and 20 (Dashing / Foxy / Galactic)
- Contains three packages
  - **CeptonPublisher**
    - Publishes topics for different lidar messages. Topics are:
      - **cepton_pcl2** (`sensor_msgs::msg::PointCloud2`)
        - http://docs.ros.org/en/melodic/api/sensor_msgs/html/msg/PointCloud2.html
      - **cepton_info** (`cepton_messages::msg::CeptonSensorInfo`)
        - Ros2/cepton_messages/msg/CeptonSensorInfo.msg
      - **cepton_points** (`cepton_messages::msg::CeptonPointData`)
        - Ros2/cepton_messages/msg/CeptonPointData.msg
  - **CeptonSubscriber**
    - Reference code showing use of topics for different lidar messages.
  - **CeptonMessages**
    - Contains ROS2 topic definitions for Cepton lidar messages. Contains CeptonSensorInfo and CeptonPointData message formats

Cepton is hiring interns and new college grads. Check out our LinkedIn or Handshake job page

ROS Installation

Building ROS 2 on Ubuntu Linux — ROS 2 Documentation: Foxy documentation



```
locale  # check for UTF-8

sudo apt update && sudo apt install locales
sudo locale-gen en_US en_US.UTF-8
sudo update-locale LC_ALL=en_US.UTF-8 LANG=en_US.UTF-8
export LANG=en_US.UTF-8

locale  # verify settings
```

Add the ROS 2 apt repository

You will need to add the ROS 2 apt repositories to your system. To do so, first authorize our GPG key with apt like this:

```
sudo apt update && sudo apt install curl gnupg2 lsb-release
sudo curl -sSL https://raw.githubusercontent.com/ros/rosdistro/master/ros.key  -o /usr/share/keyrings/ro
```

And then add the repository to your sources list:

```
echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/ros-archive-keyring.gpg] http
```

Install development tools and ROS tools

```
sudo apt update && sudo apt install -y \
  build-essential \
  cmake \
  git \
  libbullet-dev \
  python3-colcon-common-extensions \
  python3-flake8 \
  python3-pip \
  python3-pytest-cov \
  python3-rosdep \
  python3-setuptools \
  python3-vcstool \
  wget
# install some pip packages needed for testing
python3 -m pip install -U \
  argcomplete \
  flake8-blind-except \
  flake8-builtins \
  flake8-class-newline \
  flake8-comprehensions \
  flake8-deprecated \
  flake8-docstrings \
  flake8-import-order \
  flake8-quotes \
  pytest-repeat \
  pytest-rerunfailures \
  pytest
# install Fast-RTPS dependencies
sudo apt install --no-install-recommends -y \
  libasio-dev \
  libtinyxml2-dev
# install Cyclone DDS dependencies
sudo apt install --no-install-recommends -y \
  libcunit1-dev
```

Install the compiler for the SDK driver

Cepton is hiring interns and new college grads. Check out our LinkedIn or Handshake job page

Building the Cepton ROS Driver

Build system

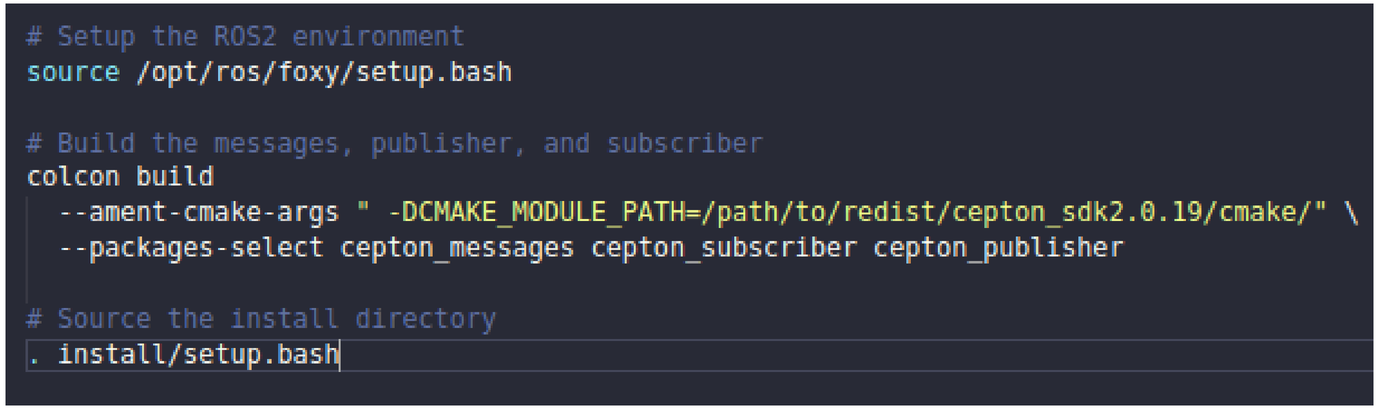
- Uses **ament**
  - https://design.ros2.org/articles/ament.html

Compiler

- Uses **colcon**
  - https://docs.ros.org/en/foxy/Tutorials/Colcon-Tutorial.html

Building the packages:

Nav to `sdk/ros2`



```
# Setup the ROS2 environment
source /opt/ros/foxy/setup.bash

# Build the messages, publisher, and subscriber
colcon build
  --ament-cmake-args " -DCMAKE_MODULE_PATH=/path/to/redist/cepton_sdk2.0.19/cmake/" \
  --packages-select cepton_messages cepton_subscriber cepton_publisher

# Source the install directory
. install/setup.bash
```

Required to build `cepton_messages`, since it is needed by both subscriber and publisher

- The publisher compiled as a shared library *

Cepton is hiring interns and new college grads. Check out our LinkedIn or Handshake job page

Running the Cepton Publisher

Command-line options:

- `Publish_pcl2`: (bool) If true, then publish the point cloud over cepton_pcl2 topic with PointCloud2 format
- `Publish_cepton_points`: (bool) If true, then publish the point cloud over cepton_points topic with CeptonPointData format
- `Publish_cepton_info`: (bool) If true, then publish info messages over cepton_info topic with CeptonSensorInfo format
- `Keep_invalid`: (bool) If set to false, then invalid points (no-return points) are discarded
  - No returns are distance 100m, have a pera set – need this to distinguis
- `Sensor_port:` (int) If running with a live sensor, listen on this port, 8808
- `Capture_file`: (string) If specified, then replay data from a pcap with this filename
- `Capture_play_loop`: (bool) If true, continuously loop the pcap

Run with default args:

```
ros2 run cepton_publisher cepton_publisher_node
```

Can specify other arguments using –ros-args –p <key/value>

```
ros2 run cepton_publisher cepton_publisher_node --ros-args -p capture_file:="/path/to/pcap" -p capture_play_loop:=true
```

*

Cepton is hiring interns and new college grads. Check out our LinkedIn or Handshake job page

Cepton Subscriber

Sample application of using each provided ROS topic

- Dependent only on `cepton_messages` package

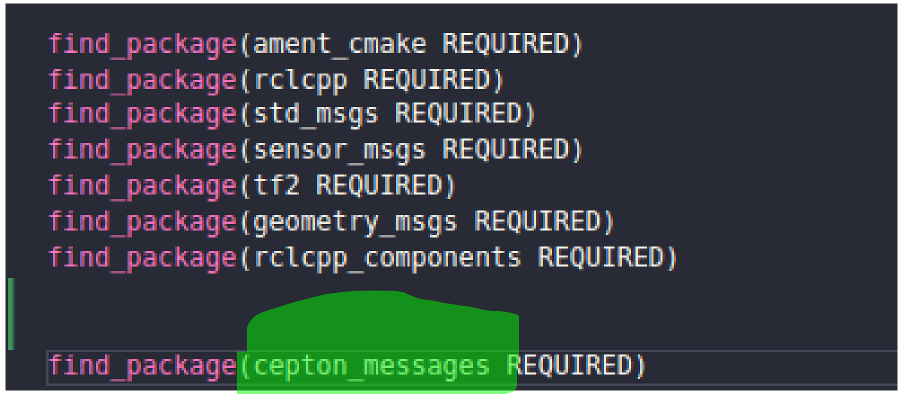
Using **CeptonPointsData**

- Subscription callback is `CeptonSubscriber::recvCepPoints`

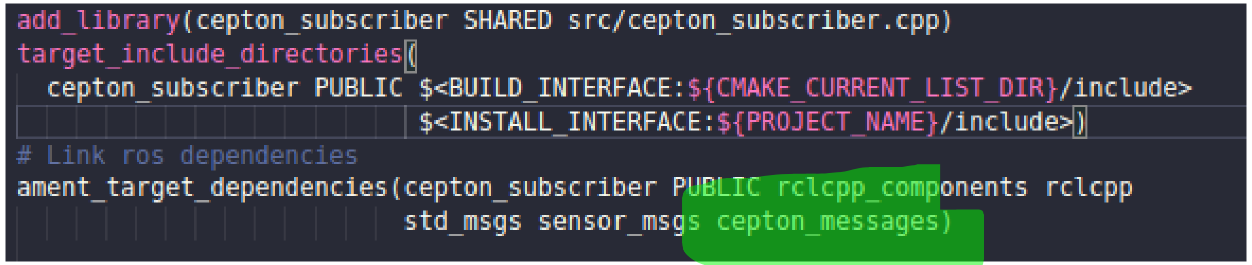
Using **PointCloud2**

- Subscription callback is `CeptonSubscriber::recvPoints`

Cepton_subscriber/CMakeLists.txt



```
find_package(ament_cmake REQUIRED)
find_package(rclcpp REQUIRED)
find_package(std_msgs REQUIRED)
find_package(sensor_msgs REQUIRED)
find_package(tf2 REQUIRED)
find_package(geometry_msgs REQUIRED)
find_package(rclcpp_components REQUIRED)


find_package(cepton_messages REQUIRED)
```



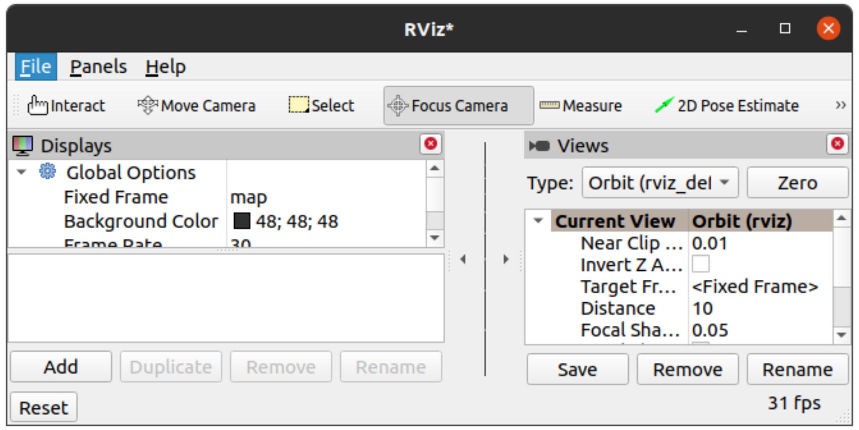
```
add_library(cepton_subscriber SHARED src/cepton_subscriber.cpp)
target_include_directories(
  cepton_subscriber PUBLIC $<BUILD_INTERFACE:${CMAKE_CURRENT_LIST_DIR}/include>
                           $<INSTALL_INTERFACE:${PROJECT_NAME}/include>)
# Link ros dependencies
ament_target_dependencies(cepton_subscriber PUBLIC rclcpp_components rclcpp
                          std_msgs sensor_msgs cepton_messages)
```

Cepton is hiring interns and new college grads. Check out our LinkedIn or Handshake job page

Viewing the data in Rviz

- Point cloud can be viewed in `Rviz2` using the `cepton_pcl2` topic
- Run `CeptonPublisher` node, make sure that it is set to publish `PointCloud2` data format
  - Alternately, can replay data from a `rosbag`
- In Rviz2 UI
  - Add `cepton_pcl2` topic
  - Change the frame to `lidar_frame`
  - To see proper intensity, disable `autocompute intensity` and set `max intensity` to be some sane value

*



Global Options

Fixed Frame	map
Background Color	■ 48; 48; 48
Frame Rate	30

Type: Orbit (rviz_del

Current View	Orbit (rviz)
Near Clip ...	0.01
Invert Z A...	☐
Target Fr...	<Fixed Frame>
Distance	10
Focal Sha...	0.05

31 fps

Cepton is hiring interns and new college grads. Check out our LinkedIn or Handshake job page

Cepton Sensor Info message format

- Published at ~1hz, over `cepton_info` topic
- `Ros2/cepton_messages/msg/CeptonSensorInfo.msg`

```
uint32 serial_number
uint32 handle
string model_name
uint16 model
uint32 part_number
uint32 firmware_version
int32 power_up_timestamp
int32 time_sync_offset
int32 time_sync_drift
uint8 return_count
uint8 channel_count
uint32 status_flags
uint16 temperature
```

Cepton is hiring interns and new college grads. Check out our LinkedIn or Handshake job page

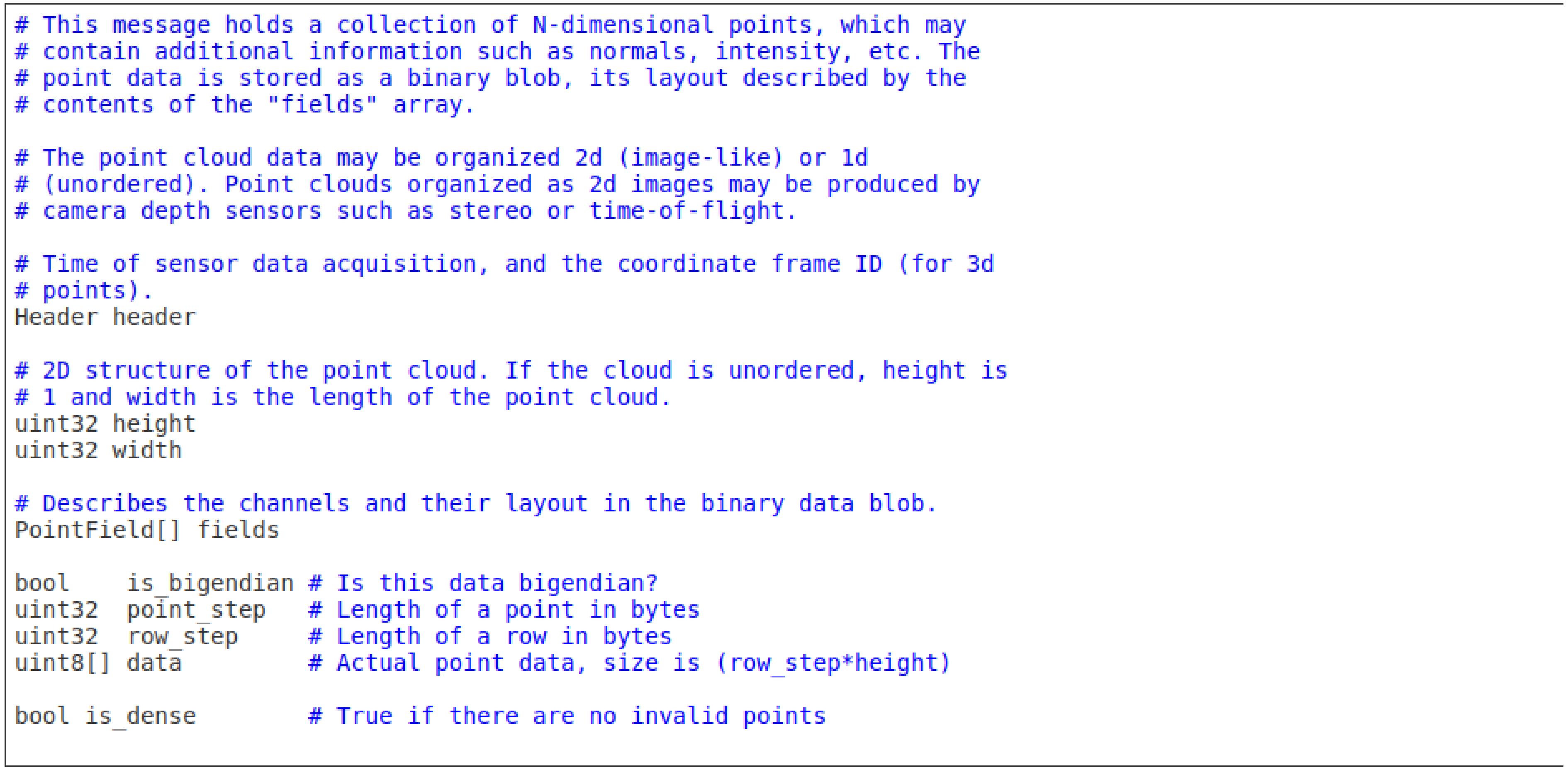
Cepton Point Data message format

- Published over `cepton_points` topic
  - ~10hz for P61
- Handle = unique identifier for each sensor, == 4-bytes sensor IP address.
  - Default sensor is 192.168.X.X, last 2 bytes set by serial number.
- Points data is the same as from SDK callback
  - Stride buffer,
  - Timestamp is relative,
  - Reflectivity as input to LUT
- `start_timestamp` is PTP or GPS timestamp (if synced)
  - Otherwise, is the host network receive time

CeptonPointData.msg

```
uint64 handle
int64 start_timestamp
uint32 n_points
uint32 stride
uint8[] points
```

ROS2 sensor_msgs::msg::PointCloud2



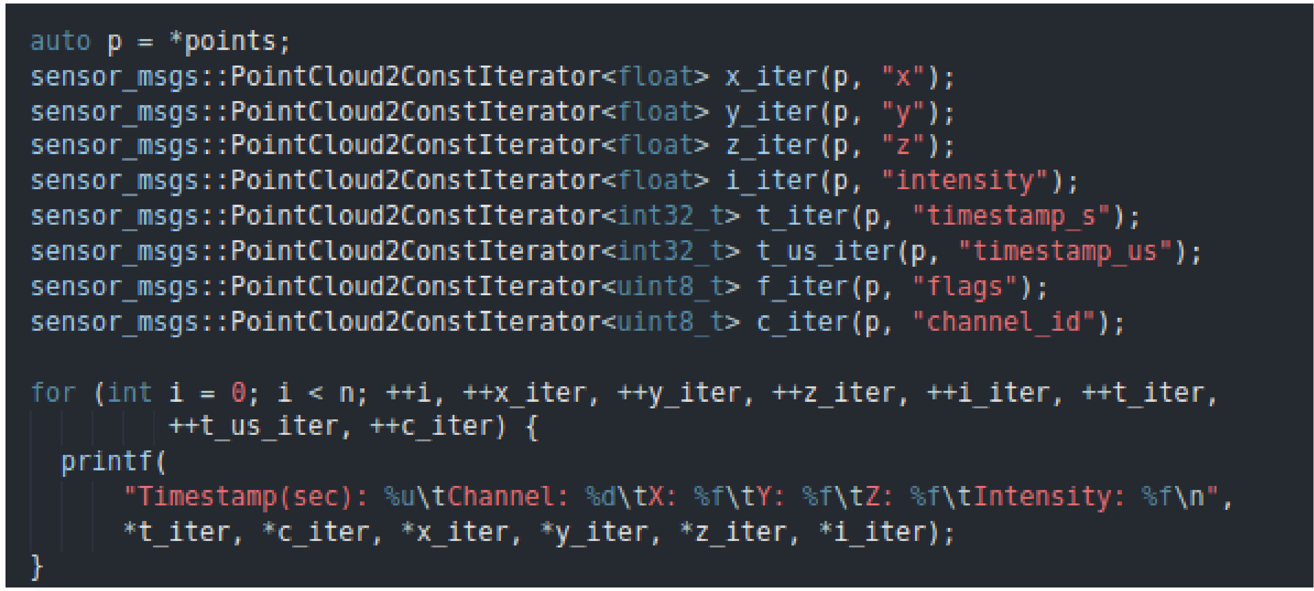
```
# This message holds a collection of N-dimensional points, which may
# contain additional information such as normals, intensity, etc. The
# point data is stored as a binary blob, its layout described by the
# contents of the "fields" array.

# The point cloud data may be organized 2d (image-like) or 1d
# (unordered). Point clouds organized as 2d images may be produced by
# camera depth sensors such as stereo or time-of-flight.

# Time of sensor data acquisition, and the coordinate frame ID (for 3d
# points).
Header header

# 2D structure of the point cloud. If the cloud is unordered, height is
# 1 and width is the length of the point cloud.
uint32 height
uint32 width

# Describes the channels and their layout in the binary data blob.
PointField[] fields

bool    is_bigendian # Is this data bigendian?
uint32  point_step   # Length of a point in bytes
uint32  row_step     # Length of a row in bytes
uint8[] data         # Actual point data, size is (row_step*height)

bool is_dense        # True if there are no invalid points
```

Link: http://docs.ros.org/en/melodic/api/sensor_msgs/html/msg/PointCloud2.html

Cepton is hiring interns and new college grads. Check out our LinkedIn or Handshake job page

Cepton Data (PointCloud2)

- Native ROS2 format, published over `cepton_pcl2` topic
- Contains fields
  - `X, Y, Z` (units=meters)
  - `Reflectivity`
    - range from 0-50; corresponds to 0-5000% scale
  - `Timestamp` (split into seconds, microseconds)
  - `Flags` (same as `CeptonPoint`), refer to `cepton_sdk2.h`
  - `Channel_id` (same as in `CeptonPoint`)
- This topic is used by `Rviz2` for visualization



```
auto p = *points;
sensor_msgs::PointCloud2ConstIterator<float> x_iter(p, "x");
sensor_msgs::PointCloud2ConstIterator<float> y_iter(p, "y");
sensor_msgs::PointCloud2ConstIterator<float> z_iter(p, "z");
sensor_msgs::PointCloud2ConstIterator<float> i_iter(p, "intensity");
sensor_msgs::PointCloud2ConstIterator<int32_t> t_iter(p, "timestamp_s");
sensor_msgs::PointCloud2ConstIterator<int32_t> t_us_iter(p, "timestamp_us");
sensor_msgs::PointCloud2ConstIterator<uint8_t> f_iter(p, "flags");
sensor_msgs::PointCloud2ConstIterator<uint8_t> c_iter(p, "channel_id");

for (int i = 0; i < n; ++i, ++x_iter, ++y_iter, ++z_iter, ++i_iter, ++t_iter,
         ++t_us_iter, ++c_iter) {
  printf(
      "Timestamp(sec): %u\tChannel: %d\tX: %f\tY: %f\tZ: %f\tIntensity: %f\n",
      *t_iter, *c_iter, *x_iter, *y_iter, *z_iter, *i_iter);
}
```

Upcoming Features

Multiple sensor support

- Distinguished by the message-header Frame field

In-situ transforms

- SDK data is published in cartesian coordinates, with Y-axis along sensor boresight. May be different than the convention used on client side. For now, just apply transform when receiving the data

Cepton is hiring interns and new college grads. Check out our LinkedIn or Handshake job page

Potential pitfalls

Mixing with ROS1

- Be careful of potential bottleneck tools
- e.g. It is possible to transfer data over a rosbridge to yield ROS1 bagfiles, but this may have potential to drop data (seen previously by timestamp analysis)

Rviz2 Display Lag

- If seen, most likely is owing to the GPU buffering, and not from packet loss

~~ASIO conflicts~~

- ~~apt install libasio-dev installs 1.10.8, used by Fast-DDS. Worked around in our ROS driver by using proxy loader with alternate dlopen flag. Alternate middleware also resolves issue.~~
  - ~~https://github.com/eProsima/Fast-DDS/issues/1484~~
- ~~Should now be resolved now with ROS2 driver, but good to keep in mind.~~

Cepton is hiring interns and new college grads. Check out our LinkedIn or Handshake job page

Questions for ROS section?

cepton.com | cepton ceptontech

Matlab API

- New addition to the SDK
- Uses direct model for receiving data, same as the current python SDK
  - Fifo is maintained by the C code
- Data processed as structure-of-arrays rather than array-of-structures
- Toolbox is at `sdk/matlab/cepton_sdk2.mltbx`

Cepton is hiring interns and new college grads. Check out our LinkedIn or Handshake job page

Matlab SDK

Direct Model (Matlab) vs Subscriber Model (ROS, Cepton C SDK)

- In the subscriber model, data is always being sent out, and client code can choose to subscribe to the data stream. Less convenient when doing offline analysis, and harder to work with in some languages
- In the direct model (Matlab), want to be able to examine each frame from a REPL environment or terminal, so calls for data are explicit.

Make use of the Cepton Frame FIFO

- Idea is that for Python and for Matlab, we want to receive frames via direct model rather than subscribing to the callbacks.
- Fifo is a ring buffer which is populated by an SDK frame callback internally, but cannot grow past a maximum size (user-defined number of frames)
- `CeptonFrameFifoGet` returns a new handle to a point data buffer
- `CeptonFrameFifoRelease` releases the handle so that a new frame can be received from the sensor

From the user side:

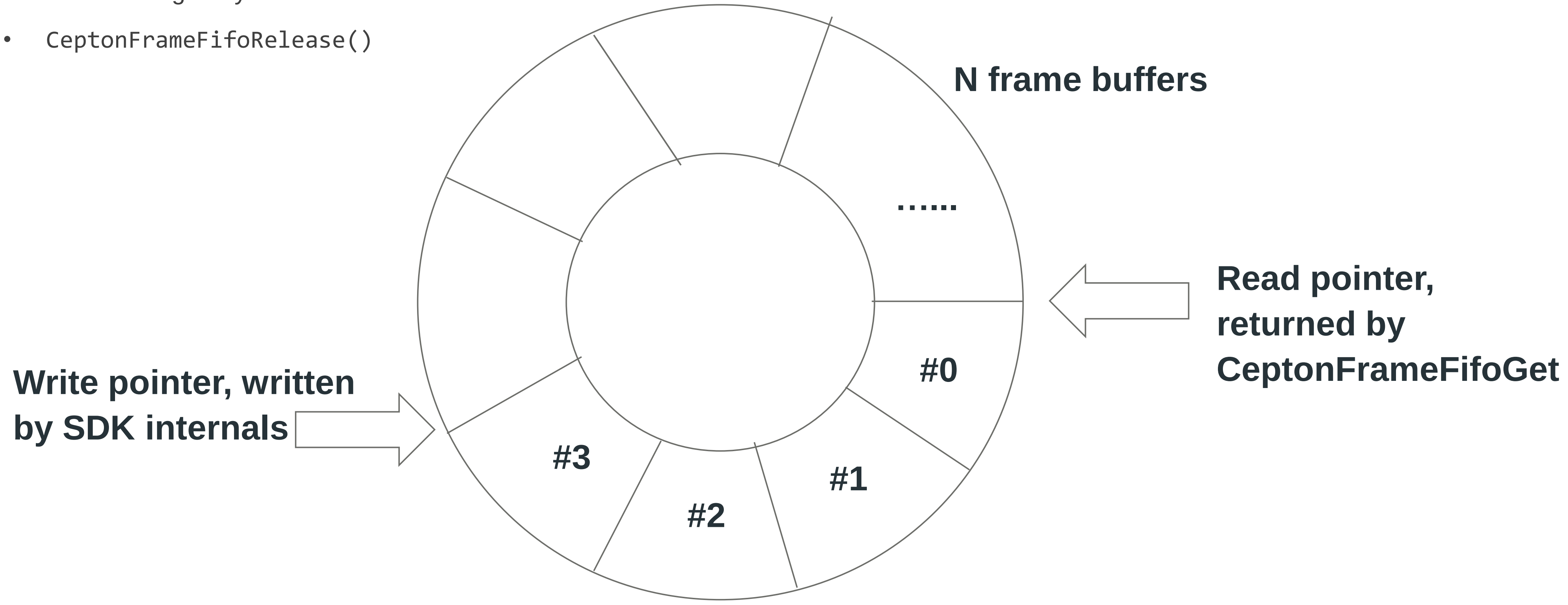
- Enable the frame FIFO with a maximum size (number of frames)
- Get and release frames
- Disable the frame fifo

Cepton is hiring interns and new college grads. Check out our LinkedIn or Handshake job page

Cepton Frame FIFO Facility

Fifo Control Functions

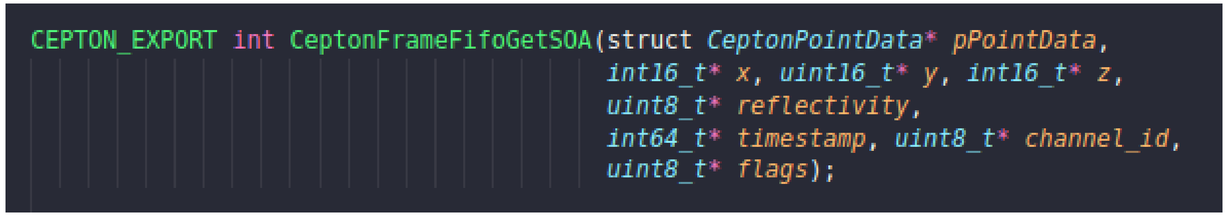
- `CeptonEnableFrameFifo(int aggMode, int numBufferedFrames)`
- `CeptonFrameFifoGet(struct CeptonPointData*)`
  - Data managed by sdk
- `CeptonFrameFifoRelease()`


N frame buffers
……
Read pointer, returned by CeptonFrameFifoGet
#0
#1
#2
#3
Write pointer, written by SDK internals

Cepton is hiring interns and new college grads. Check out our LinkedIn or Handshake job page

Matlab SDK Continued

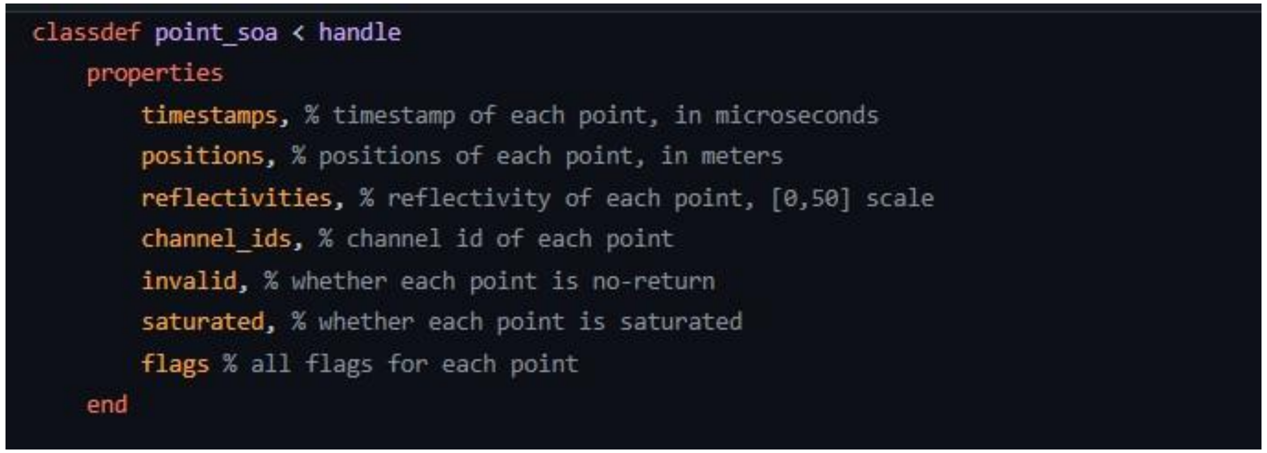
- Cannot directly use the C SDK buffer b/c Matlab doesn't permit "pointer-inside-struct"
- So, the Matlab bindings uses an extra API function in `cepton_sdk2_matlab.h` and copies data from frames FIFO into individual 1D arrays



```
CEPTON_EXPORT int CeptonFrameFifoGetSOA(struct CeptonPointData* pPointData,
                                        int16_t* x, uint16_t* y, int16_t* z,
                                        uint8_t* reflectivity,
                                        int64_t* timestamp, uint8_t* channel_id,
                                        uint8_t* flags);
```

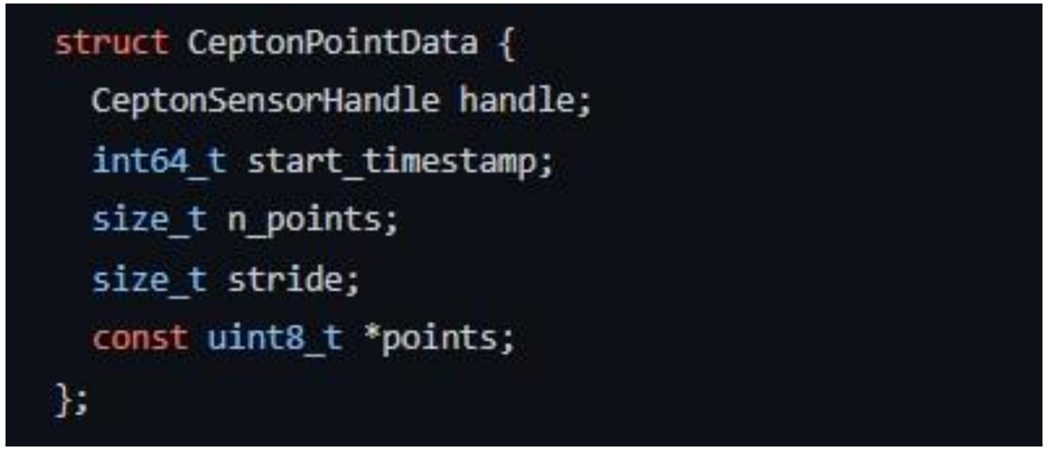
- `CeptonFrameFifoGet` returns **points_soa**
  - Structure of arrays better to make use of SIMD
  - Faster processing of large, linearly indexed data
- test_replay_get_set.m

From Matlab CeptonFrameFifoGet()



```
classdef point_soa < handle
    properties
        timestamps, % timestamp of each point, in microseconds
        positions, % positions of each point, in meters
        reflectivities, % reflectivity of each point, [0,50] scale
        channel_ids, % channel id of each point
        invalid, % whether each point is no-return
        saturated, % whether each point is saturated
        flags % all flags for each point
    end
```

From C CeptonFrameFifoGet()



```
struct CeptonPointData {
  CeptonSensorHandle handle;
  int64_t start_timestamp;
  size_t n_points;
  size_t stride;
  const uint8_t *points;
};
```

Cepton is hiring interns and new college grads. Check out our LinkedIn or Handshake job page

Questions for Matlab section?

cepton.com | cepton ceptontech

In summary

- Covered the ROS driver
  - Compilation
  - Options and usage
  - Relationship to the SDK
- Covered the Matlab API
  - Data structures
  - The frame FIFO and its purpose
  - Code sample

Cepton is hiring interns and new college grads. Check out our LinkedIn or Handshake job page