C++ for ROS

Summer Course on Developing on ROS Framework
Day 2

July 24, 2013

Introduction

Language Evolution

- C++98
- C++03
  - Technical Report 1 (TR1)
- C++11

- Boost Libraries

Objectives for Today

- Create ROS nodes in C++
  - Logging
  - Publishers and Subscribers
  - Services
  - Parameters
  - Time and Timers
- Images

C++ and ROS

Start up information

- Includes
- Namespaces
- Classes

Node Structure

- Node class
- `main` function
- Compiling

Logging

```
ROS_DEBUG_STREAM("Hello " << "World");
ROS_DEBUG_STREAM_NAMED("test_only", "Hello " << "World");
ROS_DEBUG_STREAM_COND(x < 0, "Uh oh, x = " << x << ", this is bad");
ROS_DEBUG_STREAM_COND_NAMED(cond, name, args);
ROS_DEBUG_ONCE("This message will only print once");
ROS_DEBUG_STREAM_ONCE_NAMED(name, args);
ROS_DEBUG_THROTTLE(60, "This message will print every 60 seconds");
ROS_DEBUG_STREAM_THROTTLE_NAMED(period, name, args);

ROS_ASSERT(cond);
ROS_ASSERT_MSG(cond, ...);

ROS_BREAK();
```

Publishers and Subscribers

- Include the header of the message type
- Publisher variable
- Initialize Publisher
- Create the message
- Publish
- Create the callback
- Subscriber variable
- Initialize Subscriber

Services

- Include the header of the service type
- Create the callback
- ServiceServer variable
- Initialize ServiceServer
- Wait for Service
- Call

Parameters

- has
- get
- getCached
- set

Time and Timers

- Time
- now
- Duration
- sleep

- Create the callback
- Timer variable
- Initialize Timer

About C++

Standard C++ overview

- `cmath`
- `string`
- `vector`, `list`, `map`, `set`
- `algorithm`
- `stdint.h`

Best Practices

- About comments, variable scope, functions
- About optimization
- About adding features and optimization
- About idioms (RAII)
- About object orientation and design patterns
- About functional programming
- About exceptions
- About `const` correctness

Language Tools

- Default arguments
- Function overloading
- Function (and operator) overriding
- Casts
- Default constructors and operators
- `this`

Boost Overview

- Circular Buffer
- Foreach
- Lexical Cast
- Smart Ptr
- Random
- Python
- Multithreading: Thread
- Netowrk programming: ASIO
- Functional programming: Function, Bind, Lambda
- Parsing files: Spirit

Images

OpenCV Mat Class

- Stores an image (or other 2D data)
- Works like a `shared_ptr`: Copies data only if you call `clone()`
- Easy to draw shapes or text

CV Bridge

Careful!

For old OpenCV 1 there was a `CVBridge.h`.
The new version for OpenCV 2 is `cv_bridge.h`.

- Converts `sensor_msgs::Image` to `cv::Mat`
- Convert before publishing
- Convert after receiving

Image Publisher

- Special publisher and subscriber for images
- Publisher actually creates several ROS Publishers: one for each image compression format
- Use as a normal Publisher or Subscriber, after creating a ImageTransport variable

Image View

```
rosrun image_view image_view image:=/camera/image _autosize:=true
```