THE BIOROBOTICS INSTITUTE

Sant'Anna
School of Advanced Studies – Pisa

Introductio to YARP

Egidio Falotico

Outline

- Introduction to YARP
- Introduction to Communication mechanisms in YARP
- Programming with YARP

What is YARP

YARP is a set of libraries, protocols, and tools to keep modules and devices cleanly decoupled. It is a middleware, YARP as ROS is a meta operating system.

YARP is free and open software that, as ROS, is developed to speed software development

YARP Components

The components of YARP can be broken down into:

- **YARP_os** - interfacing with the operating system(s) to support easy streaming of data across many threads across many machines. YARP is written to be **OS neutral**, and has been used on Linux, Microsoft Windows, Apple macOS and iOS, Solaris, and Android. YARP uses the open-source ACE (ADAPTIVE Communication Environment) library, which is portable across a very broad range of environments, and YARP inherits that portability. YARP is written almost entirely in C++.
- **YARP_sig** - performing common signal processing tasks (visual, auditory) in an open manner easily interfaced with other commonly used libraries, for example OpenCV.
- **YARP_dev** - interfacing with common devices used in robotics: framegrabbers, digital cameras, motor control boards, etc.

YARP Communication system: nameserver

The name server is a YARP program that maintains a list of all YARP Ports and how to connect to them (as ROS master).

The name server itself has a YARP Port, usually named **"/root".** All other YARP programs communicate with the name server through this port. This communication is usually hidden within YARP library calls, but we document it here in order to allow communication with the name server by clients not using the YARP libraries.

YARP name server

- Connecting to the name server is just like connecting to any other YARP Port (see **Port Protocol**). The one problem is that you have to find out where the name server is (what machine, what socket port number) somehow. For other YARP Ports, you can solve that problem by asking the name server, but that option isn't available for the name server itself.
- One option is simply to make sure the name server is started on a particular known machine (e.g. **192.168.0.1**) on a known socket port number (say **10000**, the default).
- The name server itself, once started, records its contact information in a configuration file (you can type "yarp conf" to find out where that file is). Other YARP programs will check this file to see how to reach the name server. If that doesn't work, there is a multicast protocol for discovering the server.

YARP Port

- A **Port** is an object that can read and write values to peer objects spread throughout a network of computers. It is possible tocreate, add and remove connections either from that program, from the command line, or from another program.
- **Ports are specialized for streaming communication**, such as camera images or motor commands. You can switch network protocols for any or all your connections without changing a line of code.
- The YARP library supports transmission of a stream of user data across various protocols – **TCP, UDP, MCAST** (multi-cast), shared memory – insulating a user of the library from the idiosyncratic details of the network technology used.

YARP PORT PROPERTIES

- For the purposes of YARP, communication takes place **through Connections**'' between named entities called Ports''. These form a directed graph, the ``YARP Network'', where Ports are the nodes, and Connections are the edges.

- The purpose of Ports is to **move ``Content**'' (sequences of bytes representing user data) from one thread to another (or several others) across process and machine boundaries. The flow of data can be manipulated and monitored externally (e.g. from the command-line) at run-time.

- **A Port can send Content to any number of other Ports**. A Port can receive Content from any number of other Ports. If one Port is configured to send Content to another Port, they are said to have a **Connection**. Connections can be freely added or removed.

- The **YARP name server tracks information about ports**. It indexes this information by name, playing a role analogous to DNS on the internet. To communicate with a port, the properties of that port need to be known (the machine it is running on, the socket it is listening on, the carriers it supports). The YARP name server offers a convenient place to store these properties, so that only the name of the port is needed to recover them.

Properties of a YARP network

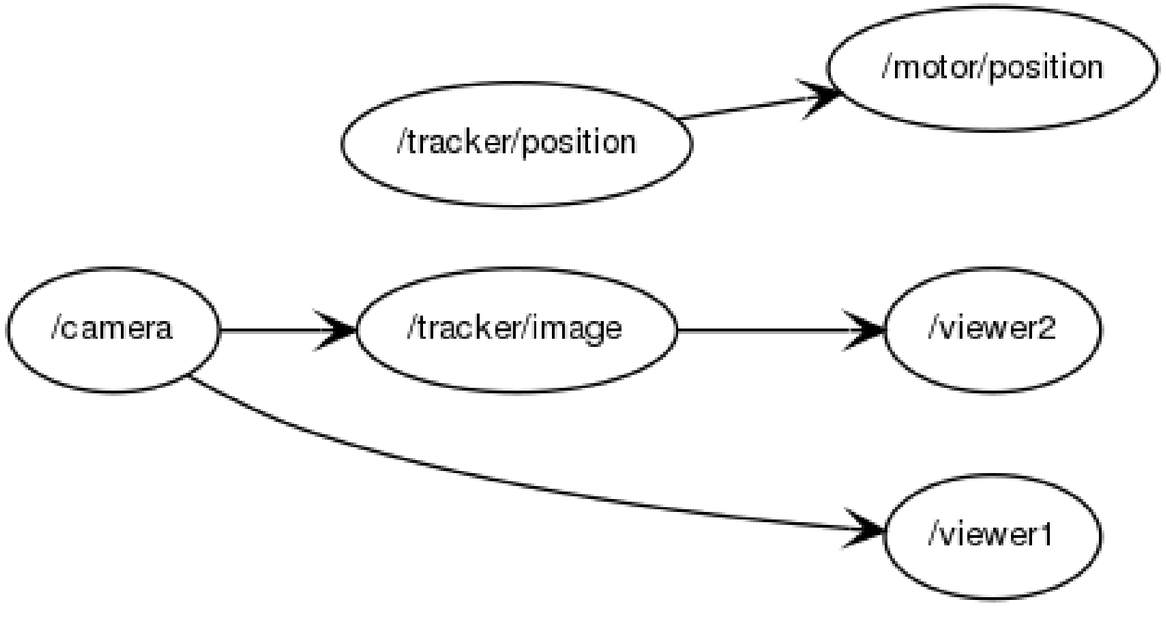
A YARP network consists of the following entities:
a set of ports, a set of connections, a set of names, a name server, and a set of registrations.

- Every port has a unique name.
- Every connection has a source port and a target port.
- Each port maintains a list of all connections for which it is the target port.
- Each port maintains a list of all connections for which it is the source port.
- There is a single name server in a YARP network.
- The name server maintains a list of registrations. Each registration contains information about a single port, identified by name.

Communication in a YARP Network

- Communication within a YARP network can occur between two ports, between a port and the name server, between a port and an external entity, and between the name server and an external entity.
  - Communication between two ports occurs if and only if there is a connection between them.
  - Connections involving a port can be created, destroyed, or queried by communication between an external entity and that port. This is done by sending ``port commands'' using the YARP connection protocol.
  - Ports communicate with the name server using the ``YARP name server protocol''. Such communication is needed to create, remove, and query registrations.

An example of a YARP network



Crating ports with the console

Run yarpserver

```
roscourse@cloud-pc:~$ yarpserver

     ==========================
    |  __ __ ___  ____   ____  |
    |  \ \/ //   ||  _ \ |  _ \ |
    |   \  // /| || |/ / | |/ / |
    |   / // ___ ||  _ \ |  _/  |
    |  /_//_/  |_||_| \_\|_|    |
    |                           |
     ==========================

Call with --help for information on available options
Using port database: :memory:
Using subscription database: :memory:
IP address: default
Port number: 10000
[INFO] |yarp.os.Port|/root| Port /root active at tcp://10.0.2.15:10000/
Registering name server with itself
 * register "/root" tcp "10.0.2.15" 10000
```

Make a reading port */read*

```
roscourse@cloud-pc:~$ yarp read /read
[INFO] |yarp.os.Port|/read| Port /read active at tcp://10.0.2.15:10002/
```

Make a writing port sending messages to */read*

```
roscourse@cloud-pc:~$ yarp write /write /read
[INFO] |yarp.os.Port|/write| Port /write active at tcp://10.0.2.15:10003/
[INFO] |yarp.os.impl.PortCoreOutputUnit|/write| Sending output from /write to /r
ead using tcp
```