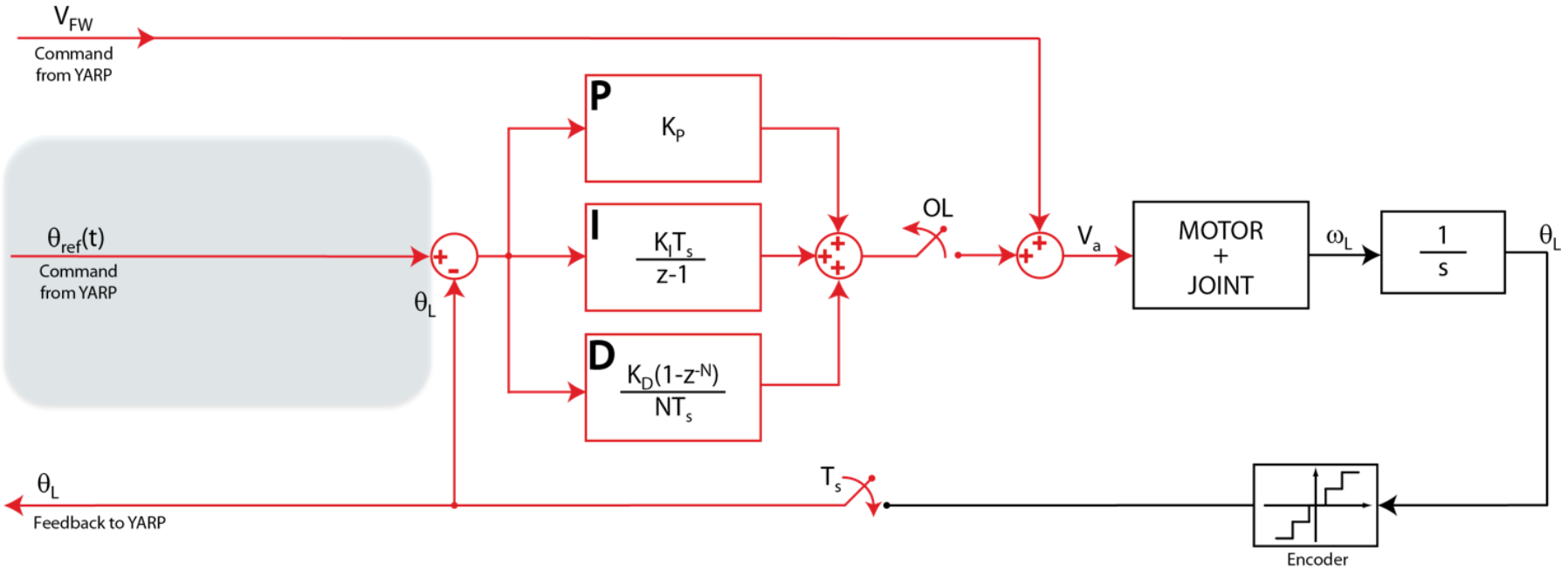
ISTITUTO ITALIANO
DI TECNOLOGIA

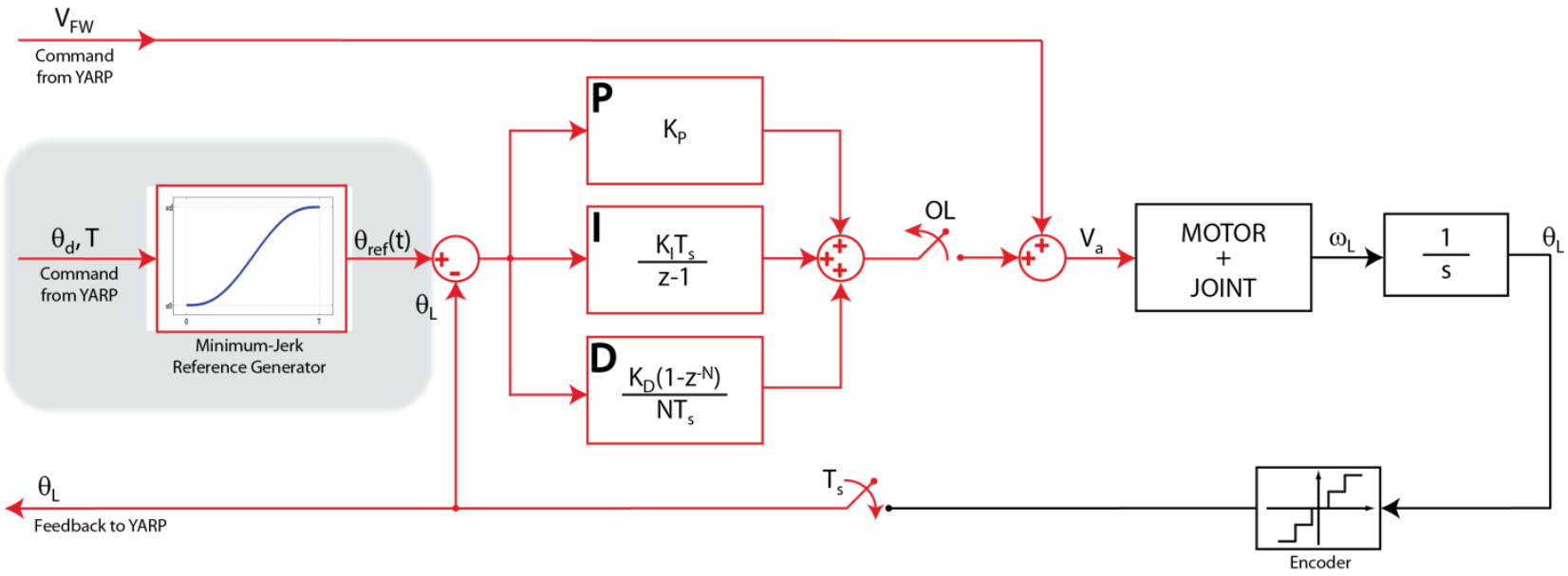
Joint Level Motor Control

Lorenzo Natale
iCub Facility
Istituto Italiano di Tecnologia, Genova, Italy

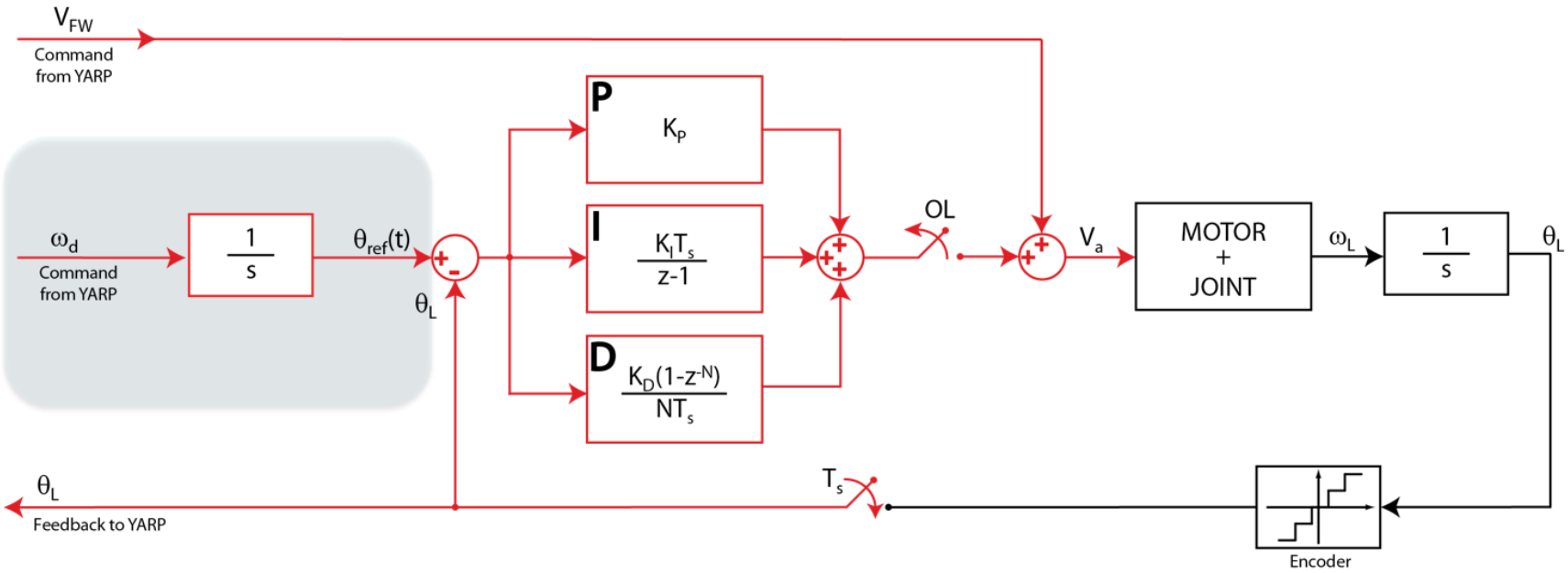
Position Direct



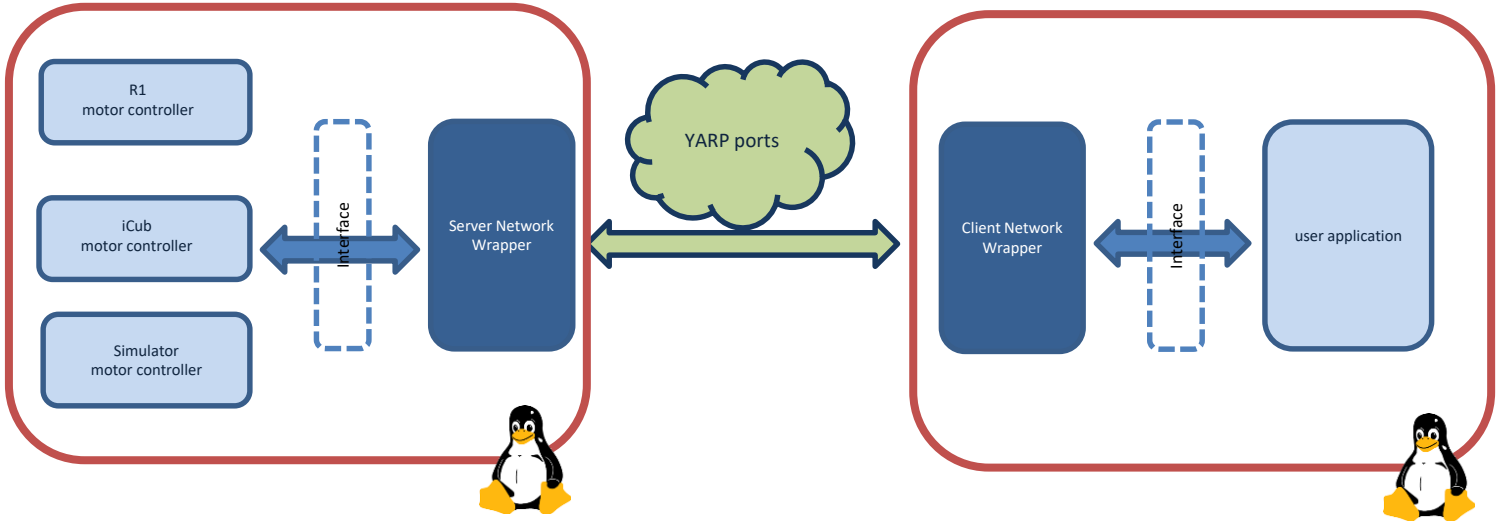
Position Control



Velocity Control



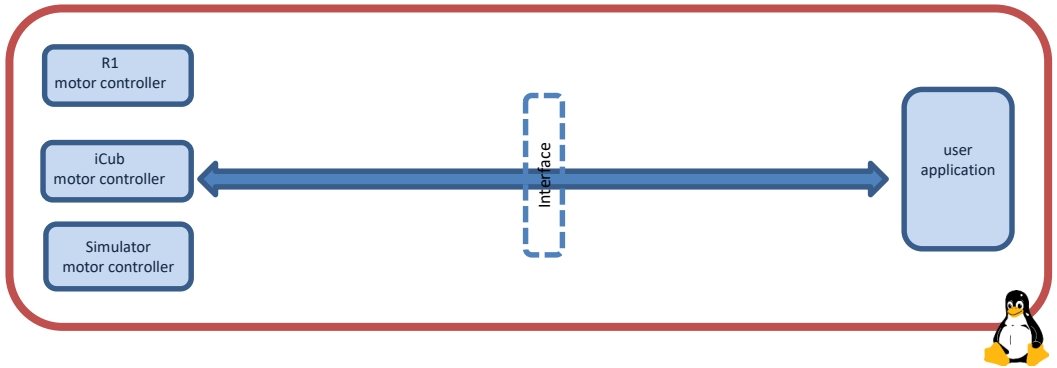
Hardware abstraction



/iCubSim/left_arm/rpc:i
/iCubSim/left_arm/state:i
/iCubSim/left_arm/command:o

/client/left_arm/rpc:i
/client/left_arm/state:i
/client/left_arm/command:o

Client & Server on the same machine



Interfaces: IPositionControl

A class with pure virtual methods.

Servers provide functionalities by implementing required methods.

Clients use the functionalities by calling provided methods.

```
IPositionControl::getAxes() = 0;
IPositionControl::positionMove(…) = 0;
IPositionControl::relativeMove(…) = 0;
IPositionControl::checkMotionDone(…) = 0;
IPositionControl::setRefSpeed(…) = 0;
IPositionControl::setRefAcceleration(…) = 0;
IPositionControl::getRefSpeed(…) = 0;
IPositionControl::getRefAcceleration(…) = 0;
IPositionControl::getTargetPosition(…) = 0;
IPositionControl::stop(…) = 0;
```

Getting Interfaces

Devices are opened by mean of a special class called “PolyDriver”.

PolyDriver is a polymorphic class which can turn into any device.

Keyword “device” tell YARP which device we really want to open.

All other parameters will be propagated to the specified device.

Device devoted to provide remote access to the robot motor control is the “remote_controlboard”

Required parameter to configure it are:

- Remote port prefix: remote
- Local port name: local

```
PolyDriver poly;

Property config;

config.put("device","remote_controlboard");
config.put("remote", "/icub/head");
config.put("local", "/<myApplication>");
...

poly.open(config);
```

```
IPositionControl2 *posControl = NULL;
poly.view(posControl);                    // Get the interface


int joints;
posControl->getAxes(&joints);             // Get number of joints

posControl->setRefSpeed(0, 5);            // set a speed of 5 degrees/s for joint 0
posControl->positionMove(0, 30);          // move the joint 0 to +30 degrees

bool done = false;
do
{
    checkMotionDone(&done);               // this function checks the movement completion
}
while(!done);

posControl->positionMove(0,0);            // move joint back to position 0


IVelocityControl2 *velControl = NULL;   // Velocity control
poly.view(velControl);

velControl->velocityMove(…);
```