CSci-5552: Sensing and Estimation in Robotics

Introduction to Pioneer Robots

Organizational Matters

- Undergraduate Robotics Lab: KHKH 1-202
- Need to get access to the lab
- Each team will be assigned a robot/sensor
- Robots are in cages with padlocks
- Re-charge the robots after use

Hardware

- Pioneer robot (Pioneer 3 or Pioneer1)
- Laptop w/ Ubuntu (Login: csci5551 / Password: csci5551)
- Sensor (Laser scanner or Kinnect)

Pioneer 1

Pioneer 3-DX

The Pioneer Robot

- Serial Connection for communication
- Differential Drive
  - Control wheel velocities independently
  - Max linear velocity 0.7 m/s , Max rotational velocity 140 deg/s
- Sensors
  - Wheel Encoders, Resolution: 100 ticks/rev
  - Sonar: 5 in front, 1 on each side
  - SICK Laser Scanner
  - Kinect

SICK Laser Scanner

- Measures the distance to objects
- Connected via Serial (USB to Serial)
- Angular Resolution: 0.5 - 1 deg
- Distance Accuracy : +/-15 mm
- Range: 1 m - 8 m

- Notes:
  - Unreliable for distances below 20 cm and near scanner edges (+/- 90 deg)
  - Connect using “Blocking Connect”

Battery Power

- Power-on Cycle
  - Main power switch controls sensors and robot
  - Red/White Buttons for Motor Control
  - Powering off the hardware at any point is OK
- Battery Usage
  - Monitor battery charge from Green/Yellow LED or LCD panel
  - Do not let charge drop below 11 V (can monitor w/ software)
  - Do not forget to turn off robot when finished
- Charging the Battery
  - 4 charging stations in the lab
  - Full charge requires 2-3 hours

How to Break Your Hardware

- Pioneers with SICK are Very Top-heavy
  - If the robot is unstable, it can turn upside down
  - Do not stop robot suddenly
  - Do not operate the robot on an incline
  - Monitor robot at all times during operation
- Be careful when removing the laptop from the robot

Robot Programming

- Aria (ActiveMedia Robotics Interface for Application)
  - Open source C++ development library enables control and communicate with the robot.
  - http://www.mobilerobots.com/Software.aspx
- ROS (Robot Operating System)
  - A set of software libraries and tools that help you build robot applications.
  - http://www.ros.org/

Installing Aria & MobileSim

- Download ARIA and MobileSim from
  - http://robots.mobilerobots.com/wiki/ARIA
  - Follow installation instructions
- After installation, the default directory
  - /usr/local/Aria
  - /usr/local/MobileSim

Setting The ARIA Environment

- ARIA Directories
  - Main: /usr/local/Aria
  - Example: /usr/local/Aria/examples
  - Documentation: /usr/local/Aria/docs/index.html
- Set Environment Variables
  - export LD_LIBRARY_PATH = $LD_LIBRARY_PATH: /usr/local/Aria/lib
  - ARIA should be set to /usr/local/Aria
    export ARIA=/usr/local/Aria

Some Aria Methods

- void Aria::init()
  - Performs OS-specific initializations.
  - MUST be called before any other Aria functions.
- void Aria::shutdown()
  - Shutdown all Aria/Process threads
- void Aria::setKeyHandler(ArKeyHandler *)
  - Sets a key handler function

Some Aria Methods

- void ArRobot::addRangeDevice(ArRangeDevice *)
  - Add a range device object to the current robot
  - Sonars and Lasers must be added in this fashion
- void ArRobot::setDeviceConnection(ArDeviceConnection*)
  - Sets the robot connection (sim or hardware)
- bool ArRobot::blockingConnect()
  - Block until successful robot connection
- void ArRobot::addAction(ArAction *,int)
  - Add an ArAction and set its priority
- void ArRobot::run()
  - Start the robot running in this thread

Some Aria Methods

- void ArRobot::runAsync(bool)
  - Start the robot running in its own thread
- void ArRobot::waitForRunExit()
  - Blocks until the robot finishes running
- int ArRobot::lock()
  - Lock the robot object (for thread-safe operation)
- int ArRobot::unlock()
  - Unlock the robot object
- bool ArRobot::comInt(char, int)
  - Poke the hardware (activate/deactivate sound/sonars, etc...)

Control the Robot

- void ArRobot::setVel(double)
  - Sets the linear velocity of the robot
- void ArRobot::setRotVel(double)
  - Sets the rotational velocity of the robot
- void ArRobot::move(double)
  - Moves the robot straight
- void ArRobot::setHeading(double)
  - Sets “absolute” heading of the robot
- void ArRobot::setDeltaHeading(double)
  - Sets “relative” heading of the robot
- bool ArRobot::isMoveDone(double)
  - Is the last specified move done?
- void ArRobot::stop()
  - Stops the robot

- All of these must be wrapped in lock() and unlock()

Setting up SICK

```
ArSick sick;
ArSerialConnection laserConn;
sick.configureShort(usingSim, ArSick::BAUD38400,
ArSick::DEGREES180, ArSick::INCREMENT_HALF);
sick.setDeviceConnection(&laserConn);
laserConn.open("/dev/ttyUSB1");
sick.runAsync();
sick.blockingConnect();
```

- Resolution
  - ArSick::INCREMENT_HALF is 0.5 deg – 361 readings
  - ArSick::INCREMENT_ONE is 1 deg – 181 readings

Reading data from SICK

```
std::list<ArSensorReading *> *readings;
std::list<ArSensorReading *>::iterator it;
mySick->lockDevice();
readings = mySick->getRawReadings();
if (NULL!= readings) {
if ((readings->end() != readings->begin())) {
for (it = readings->begin(); it != readings->end(); it++) {
std::cout << (*it)->getRange() << " ";
}
std::cout << std::endl;
} else {
std::cout << "(readings->end() == readings->begin())" << std::endl;
}
} else {
std::cout << "NULL == readings" << std::endl;
}
mySick->unlockDevice();
```

Example: SickWanderUSB

- Uses actions to cause the robot to wander around and not hit obstacles
- Steps:
    1. Switch on the robot
    2. Connect the robot to the computer (Serial-USB)
    3. Connect the laser scanner to the computer (Serial-USB)
    4. Run command: ls /dev/ttyUSB*
    5. Check that robot port = /dev/ttyUSB0, sick port = /dev/ttyUSB1,
    6. Run the command: ./sickwanderusb –rp /dev/ttyUSB0 –lp /dev/ttyUSB1

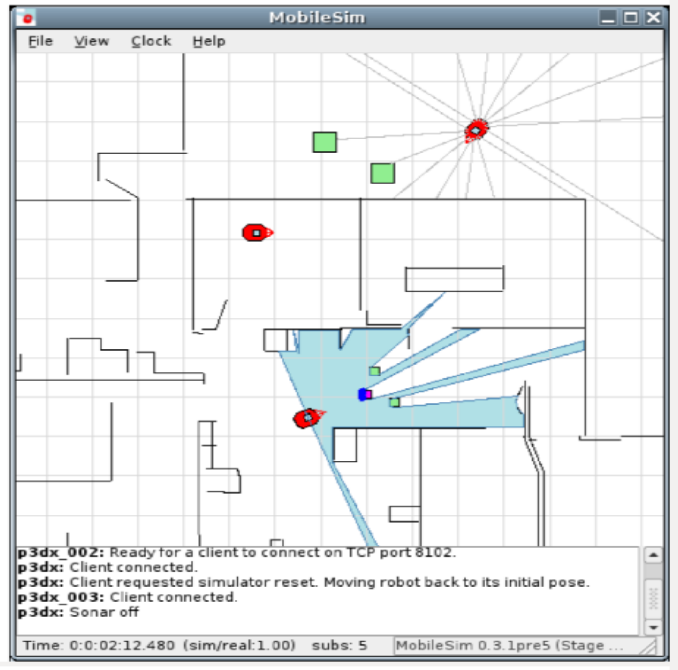
Running MobileSim

- MobileSim –m <mapfile> -r <robot>

  %MobileSim –m AMRoffice.map

- Run your own program

  %./sickWander



Some Suggestions

- If an Aria program freezes or refuses to exit properly:
  CTRL-Z, then ‘killall –9 <progname>’
- Pioneer 1 motors must be enabled manually
  - robot::comInt(ArCommands::ENABLE,1) does nothing
- Start project early
- Debug software issues with simulator
- Fine tune performance with hardware

References

- Aria documentation

http://robots.mobilerobots.com/docs/api/ARIA/2.9.0/docs/index.html

- Pioneer 3-DX datasheet

http://www.mobilerobots.com/Libraries/Downloads/Pioneer3DX-P3DX-RevA.sflb.ashx

- Sick laser scanner datasheet

https://www.mysick.com/PDF/Create.aspx?ProductID=45446&Culture=en-US