2017

How to use the ISAS-Crawlers

Contents

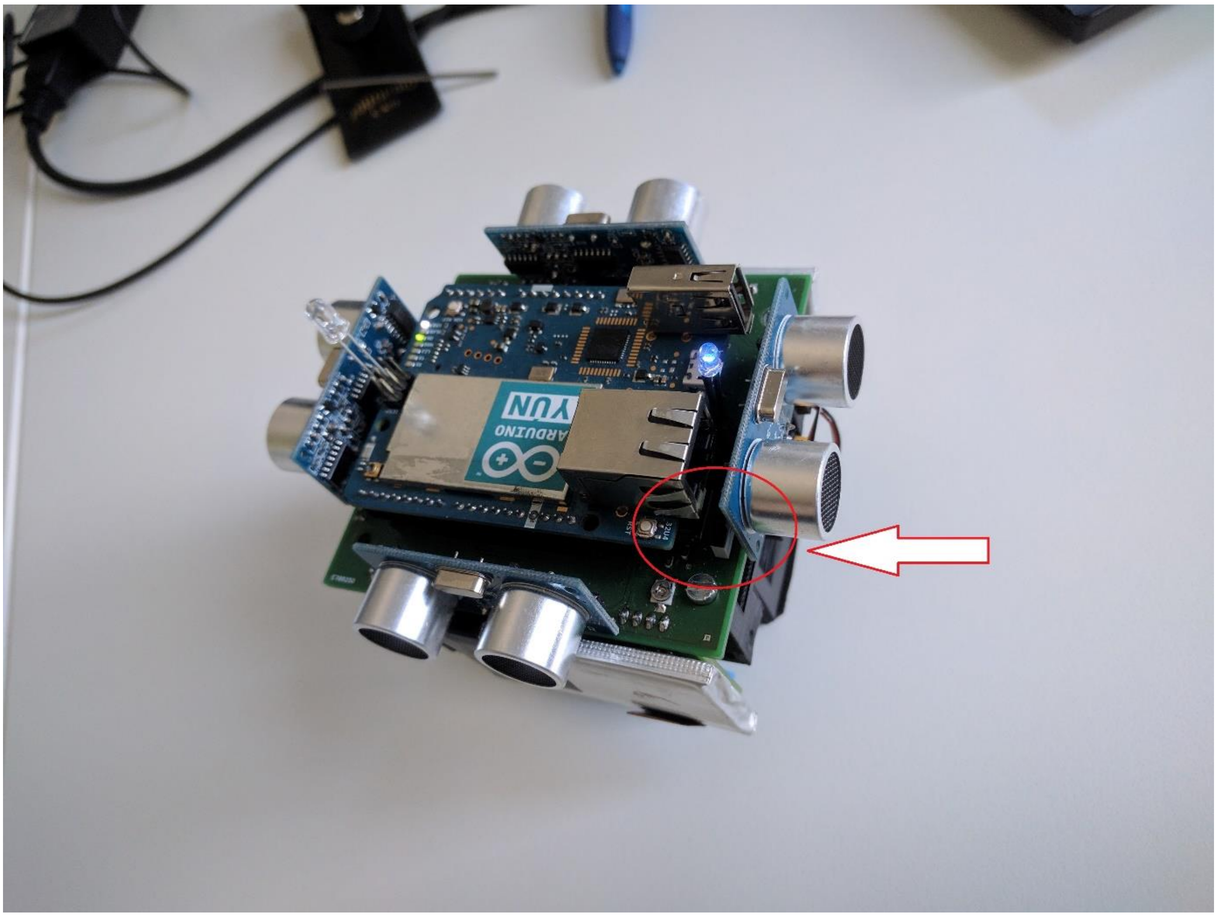
1 General

1.1 Basic Information

Once the crawler is booted it will create an ad-hoc Wi-Fi called Arduino Yun-{some MAC address} for the blue one. The white one's WI-FI is called Linino-{some MAC}. The **static IP** of the crawler is: *192.168.240.1.* DHCP is enabled on the robot so it will automatically assign an IP to hosts connecting to the network.

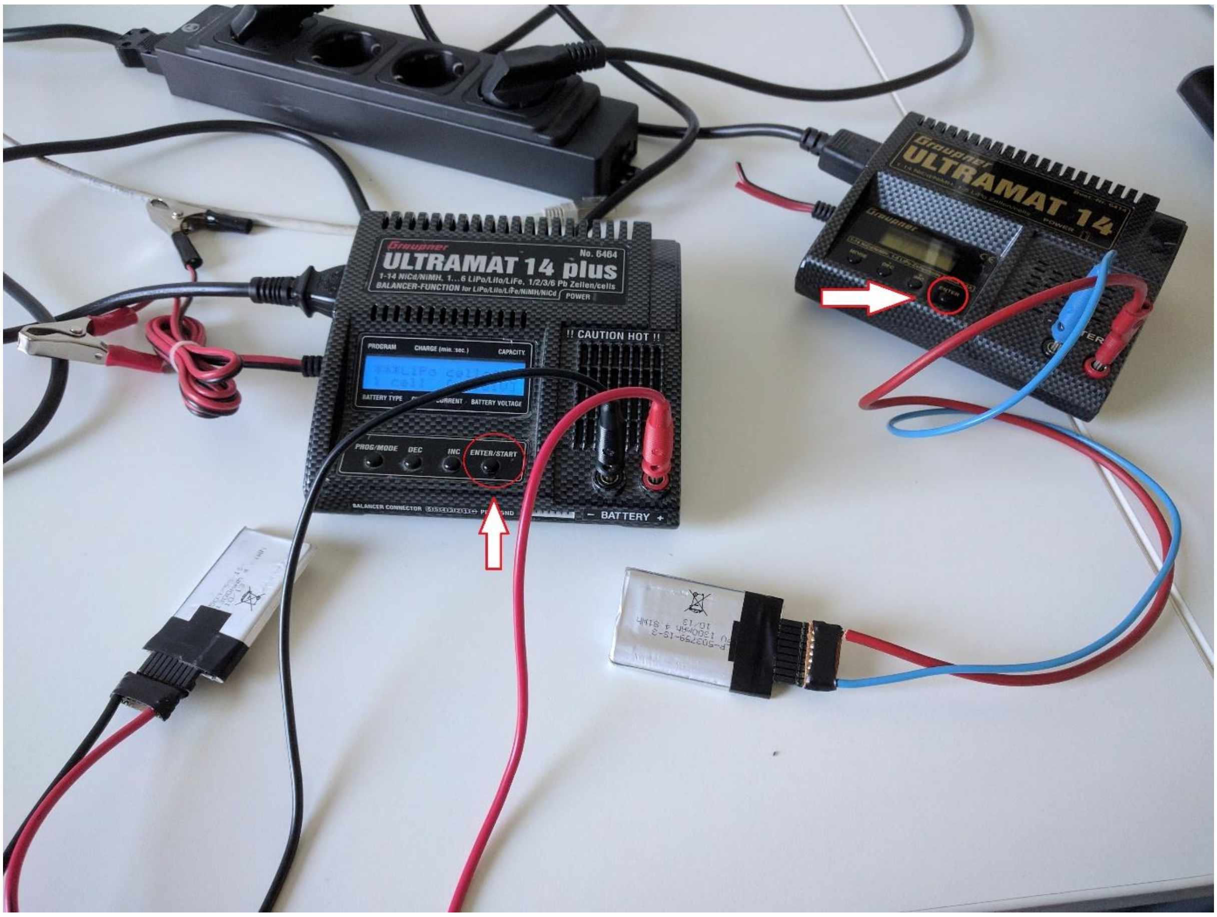
1.2 Powering on the Crawler

On the crawler, there is small switch just in between the ultrasound and Arduino. This switch can be used to turn on the robot. It takes around 1 to 2 minutes till the robot has finished booting. During this time, the robot will make some motor tests. Once the LEDs are turned on, the robot is fully booted.



1.3 Charging the LiPo batteries

The two LiPo batteries can be charged using the adapters in the lab. This process only involves the enter buttons of the chargers. Once all options are confirmed on the charger the charging process can be started by pressing and holding the start button for a few seconds.



1.4 Connecting to the Crawler

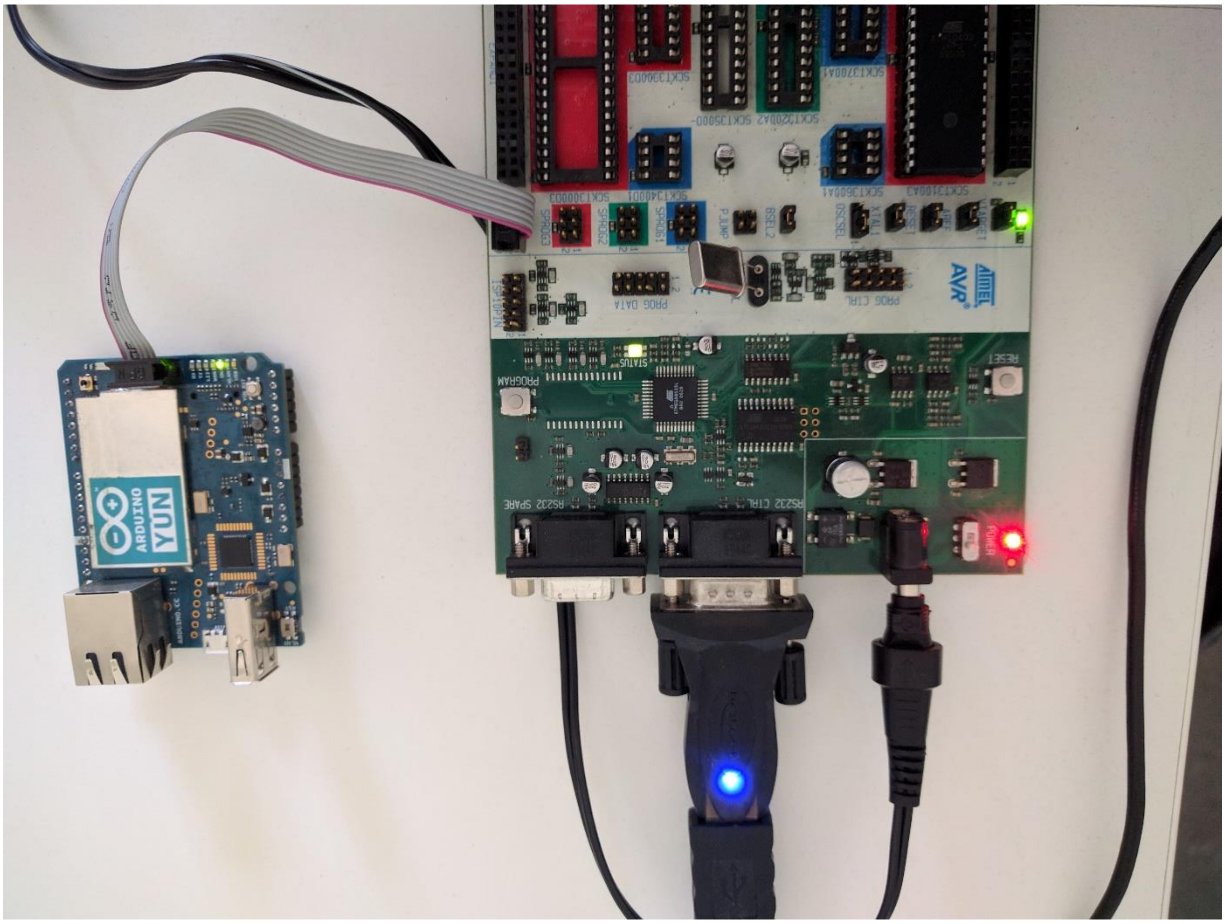
The crawler automatically starts an HTTP server as well as SSH server.

The **username** and **password** for logging in is: *root* / *arduino*.

The white crawler has a different password: *doghunter*

1.5 Uploading a sketch using a programmer

The Arduino is connected to the development as shown in the following figure



The Arduino IDE is then configured using the following parameters:

Board: Arduino Yun

Port: e.g. COM3

Programmer: Atmel STK500 development board

The development board works with 7.5V input.

2 Commands

2.1 Move commands

The crawler features six movement primitives which are coded as follows

- move forward (0)

- move backward (1)
- move right (2)
- move left (3)
- rotate right (4)
- rotate left (5)

These primitives can be sent to the crawler via a TCP/IP socket. The general format of the movement command is

```
MOVE PRIMITVE, PERCENT, STEPS
```

where *PRIMITIVE* is the code of the movement, *PERCENT* a value between 0 and 100 and *STEPS* the number of repetitions.

2.2 IR MEASUREMENT

In order to take a measurement using the IR camera on the crawler the following command needs to be sent

```
IR
```

The crawler will perform a measurement and return an array of detected *x* and *y* coordinates of the landmarks. In case that there is no landmark in the field of view of the camera then *-1* will be returned by the call.

2.3 US MEASUREMENT

The command to perform an ultrasound measurement is defined by the format

```
US ID NUMBEROFSCANS
```

where *ID* is the identifier of the ultrasound sensor and *NUMBEROFSCANS* defines the repetitions. The crawler will return an averaged result of the readings.

The identifiers are numbered as follows

- Backwards (0)
- Left (1)
- Forward (2)
- Right (3)

2.4 IMU MEASUREMENT

The IMU returns Euler-angles in the format ZYX and in radians. The command is just as simple as

```
IMU
```

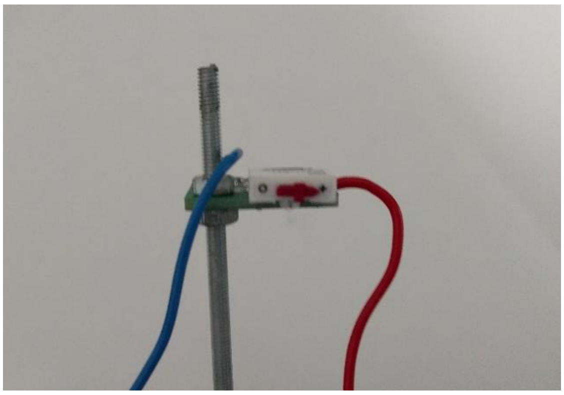
There are two ways how to get the data from the IMU.

1. Every move command returns the incremental angles between two movements by default.
2. By sending the IMU command the current absolute angles are returned.

3 LANDMARKS

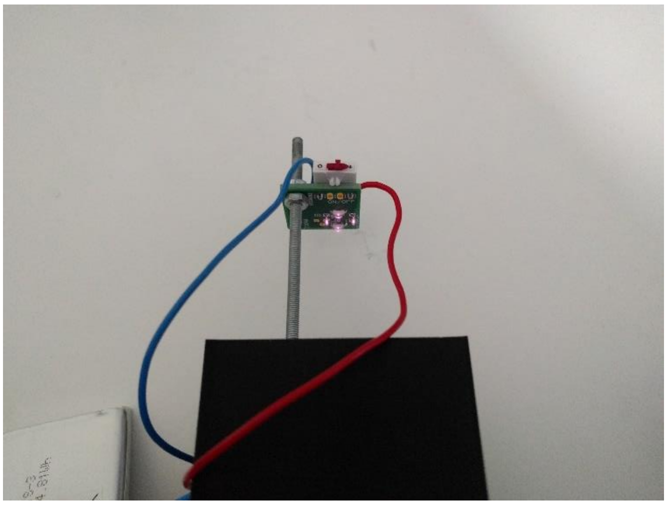
3.1 TURNING ON THE LANDMARKS

The landmarks are powered by LiPo-batteries. Therefore, the board needs to be connected with the battery. In order to turn on the landmarks the switch needs to be on +. Don't get confused with the *On/Off* label underneath the board.



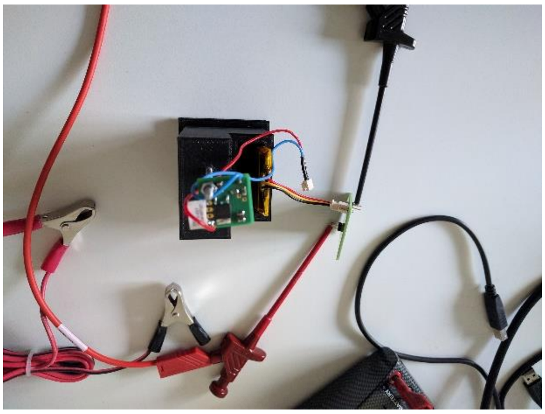
3.2 CHECK IF THE LANDMARKS ARE WORKING

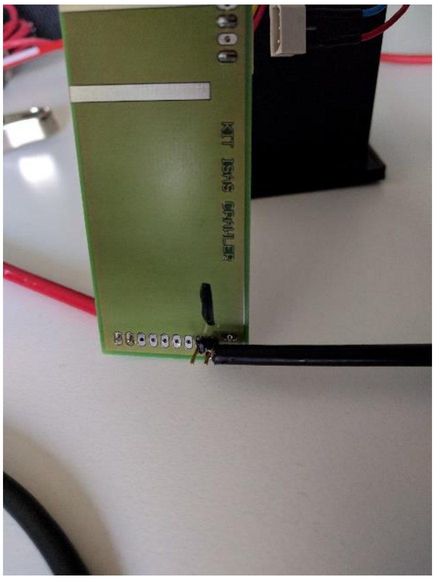
A straightforward way to check if the landmarks are working correctly is by using a camera from a smartphone.



3.3 Charging the landmarks

The batteries of the landmarks can be charged by using the same charger as for the crawler batteries.





4 Bridge

4.1 Python bridge

The crawler does not use the default bridge for the MIPS processor anymore. The bridge is the connection between the two processors. This was done for performance reasons.

As a result, new crawlers need to be adapted before they can communicate with the Matlab code. Obviously, if one wishes to keep the default bridge this modification can be omitted.

In order to install the new bridge, the old one needs to be disabled first. This is done by uncommenting the following line to */etc/inittab*

```
# ttyATH0::askfirst:/bin/ash –login
```

The new one can then be used by adding a start command to */etc/rc.local*, such as

```
python /usr/lib/python2.7/bridge/ownbridge/tcpipBridge.py
```

where tcpipBridge.py is the new bridge.

4.2 TCP/IP SOCKET AND MATLAB COMMUNICATION

In order to connect to a socket from Matlab, the *Instrument Control Toolbox* needs to be installed.

https://mathworks.com/products/instrument.html

This toolbox provides a class called *tcpip* which can be used to connect to the socket.

5 CAMERA

In room R137 there is a camera hanging from the ceiling which can be used to obtain a ground truth of the robot’s trajectory. The current implementation relies on the *Image Acquisition Toolbox*.

Furthermore, to calibrate the camera the *Camera Calibration Toolbox for Matlab* can be used:

http://www.vision.caltech.edu/bouguetj/calib_doc/