User Guide

SDP Group 15-H

March 5, 2016

1 Installation

1.1 Cloning the repository and setting up

To clone the repository, execute in terminal:

```
$ git clone https://github.com/julijonas/venus.git
```

Then go to the root directory of the project and create a Python virtual environment:

```
$ cd venus
$ virtualenv --system-site-packages env
$ source env/bin/activate
```

Install the libraries (currently there is only pyserial 3.0.1 that needs to be installed):

```
$ pip install -r requirements.txt
```

To open the Arduino IDE, execute:

```
$ arduino
```

You'll need to add three libraries which can be found under `arduino/` in the project directory: *ArduinoSerialCommand*, *SDPArduino*, *SimpleTimer*. In order to add a library, go to:

```
Sketch -> Import Library... -> Add Library... and choose the library folder
    you want to import.
```

2 Overview of the robot

At this stage of the course, we can say that the hardware part of our robot is done. Venus is a three wheels robot. There are two main driving wheels and one holonomic wheel placed sideways at the back which allows the robot to turn as well as does not

cause problems moving backwards and forwards due to its holonomic property. All wheels are powered by NXT motors.
There is also a kicker that is powered by a NXT motor with gears. The kicker operates by going backwards inside the robot from the starting low position and then kicks the ball forwards.

The grabber consists of two parts resembling arms and is placed on the sides of the front part of the robot. It is powered by Electric Technic Mini-Motor 9v with gears.

2.1 Use

First thing you would need to do to be able to communicate to Venus, is to connect the RF stick to the machine you are running the commands from. Then after performing steps described in "Running instructions" section, you will be able to operate the robot.
If you make any changes to the Arduino code, you need to upload it before running the commands. Uploading can be done either via RF stick or cable connected to the Arduino board.
In order to turn the robot on, connect the battery pack to the power board. In order to swap the batteries, take them out, put them to charge and insert the new batteries into the battery pack. Easy!

3 Running instructions

Run the command in the terminal from the project root directory if you haven't done so already:

```
source env/bin/activate
```

Then, in order to be able to send commands the robot:

```
python main.py
```

Executing *main.py* makes a connection to the RF stick and to the vision feed automatically. After that you should be able to send commands to the robot (make sure the RF stick is plugged in). In order to change the room of the pitch (0 - 3.D03, 1 - 3.D04) and colors of your robot, change hard-coded values in *vision* method in */control/commands.py*. For example if you are in room 3.D04 and your robot has yellow team-color and one green dot, change as follows:

```
def vision(self, room_num=1, team_color='yellow', our_color='green')
```

NB In case you make changes to Arduino code, you should upload your changes to the board!

The list of possible commands is below (this is for milestone 3 and some basic commands, will be changed later):

- To move forward and turn (negative values will change the direction):

```
f <distance>
c <degrees>
```

- To engage the grabber:

```
g
```

- To kick the ball:

```
kick <distance>
```

- To release the grabber (two modes):

```
open_narrow
open_wide
```

- To stop all the motors:

```
s
```

- For catching the ball:
  The robot detects where the ball is, turns in that direction and opens the grabber. When the ball is kicked towards the robot, it grabs it. In case the ball bounces off, the robot starts moving in order to grab the ball.

```
catch_ball
```

- To catch the ball and then pass it to the teammate:
  Venus turns towards the ball and opens the grabber, then when it grabs the ball, it turns to the teammate robot and kicks the ball (the distance by which the ball should be kicked is calculated such that the ball can reach the teammate robot).

```
catch_pass
```

- To intercept the ball that is passed between two opposite team robots:
  The robot goes to the closest point from itself on the line between two enemy robots.

```
intercept
```

- To block the goal:

```
block_goal
```

  Later, this part should look more like what should be done in order to fully prepare robot for the game, and things like commands/strategy will be described in the technical specifications.

4 Troubleshooting guide

Problem 1 When trying to upload the code to the Arduino board, you get the error message below.

```
processing.app.SerialNotFoundException: Serial port '<port>' not found. Did
    you select the right one from the Tools > Serial Port menu?
```

Solution It can't detect the Arduino board. Make sure it's connected to the computer through the RF stick or the cable. If it is, disconnect and connect again, or power the board off and on again. The Arduino IDE is only able to program the Arduino when the serial interface of the RF stick or Arduino itself is registered as `/dev/ttyACM0` on the PC, that is, zeroth device. Also, make sure you are not running 'python control/control.py' or have not opened the serial interface any other way when you're uploading the changes.

Problem 2 When sending commands the robot doesn't do anything and you're sure it should move.
Solution Power the Arduino board off and on again.

Problem 3 We cannot distinguish robots on the video frames, because they are flipping.

Solution We have implemented a probabilistic way of detecting robots, so that we can see them even when not all colors are detected (more details on that in technical specifications).

Problem 4 The distortion varies on different machines/pitches.
Solution It is still ongoing problem, we are working on to make it more accurate.

Problem 5 The kicker is not reliable. Sometimes the kick is not straight.
Solution It is still ongoing problem, we are working to make it more accurate.