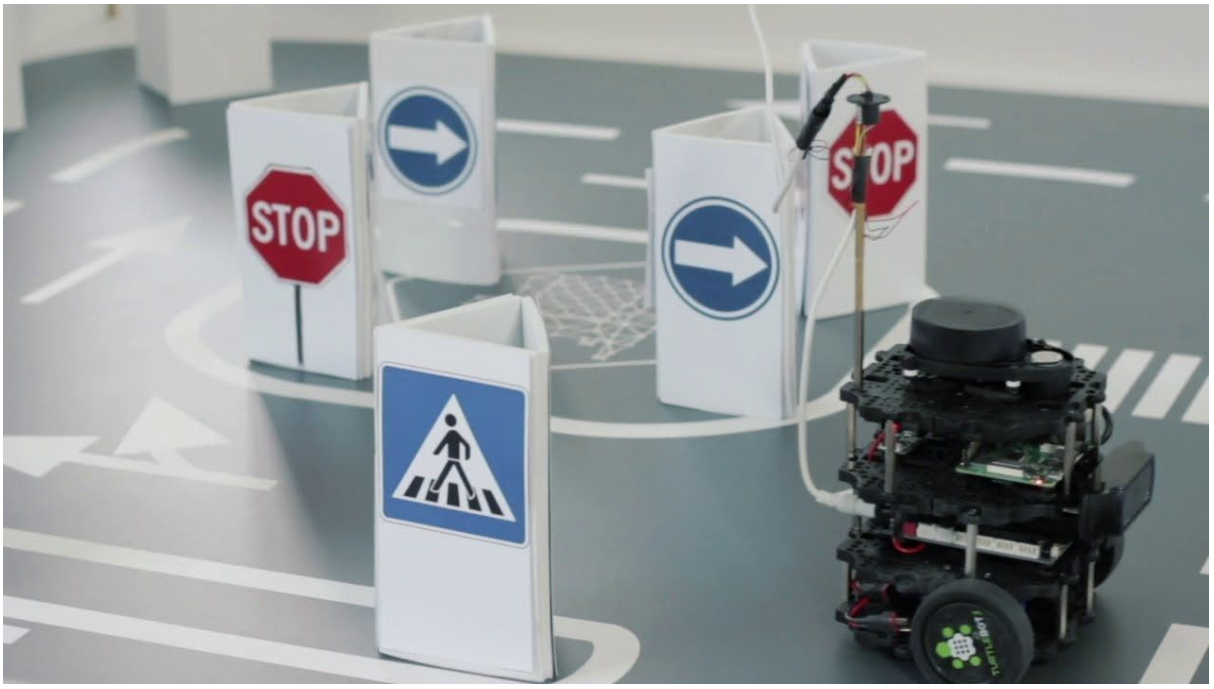
Working Procedure Draft 1.3	Written by Ricardo Téllez
Last updated	03/III/2021

How to provide to students remote access to your real robots using The Construct



This document explains how to configure a real robot setup so your students can access it from their remote location through a standard The Construct interface.

The configuration explained here is very convenient for Universities and Schools, since it prevents students from accessing the robot directly to its computer. Students will never have direct access to the robot files, just to the robot ROS topics, services and messages. This is very convenient since **you will never get a misconfigured robot due to mistakes done by the students**.

This configuration also allows the **quick switch between students**, allowing you to have a queue of students, ready to go one after another to test on the real robot.

We need to do three things:

1. PREPARE THE ROBOT AND ITS ENVIRONMENT
2. PREPARE THE ROUTER
3. PREPARE THE STUDENT

A. PREPARE THE ROBOT AND ITS ENVIRONMENT

1. Requirements of the robot

First, your robot must be running Ubuntu Linux. If you are running another Linux distribution contact The Construct team for further instructions.

Second, you must have your robot already running ROS. This means that your robot must be already running the drivers to access the sensors and actuators of the robot, and be publishing that information in some ROS topics. This is mandatory before anything else.

Any ROS distribution would do it. At present The Construct supports ROS1 distributions Kinetic, Melodic and Noetic. It also supports ROS2 distributions Dashing, Eloquent and Foxy. Installing any of those in your robot will do it.

Finally, the robot must be connected to the wifi that gives the robot access to the internet. This means, if you log into the robot computer, you must be able to do a ping to Google as well as update the Ubuntu packages (*sudo apt update* command must work properly).

2. Additional install in the robot

Now you have to install the OpenVPN server in the robot. This is the one that will establish the connection between the student account in The Construct and the real robot computer as if both computers were side by side.

(the following instructions for OpenVPN have been copied from this post, included here just for convenience in the document)

In this tutorial, we will install the OpenVPN server on Ubuntu 20.04. To do this, you need to log in as the root user. You also must know the public IP of the server with which clients will establish a secure VPN channel.

OpenVPN installation and configuration

We will use the script to install and configure all the necessary packages to start the OpenVPN server. All you have to do is provide it with the correct public IP address of your server. Let’s download it.

```
wget https://raw.githubusercontent.com/angristan/openvpn-install/master/openvpn-install.sh
```

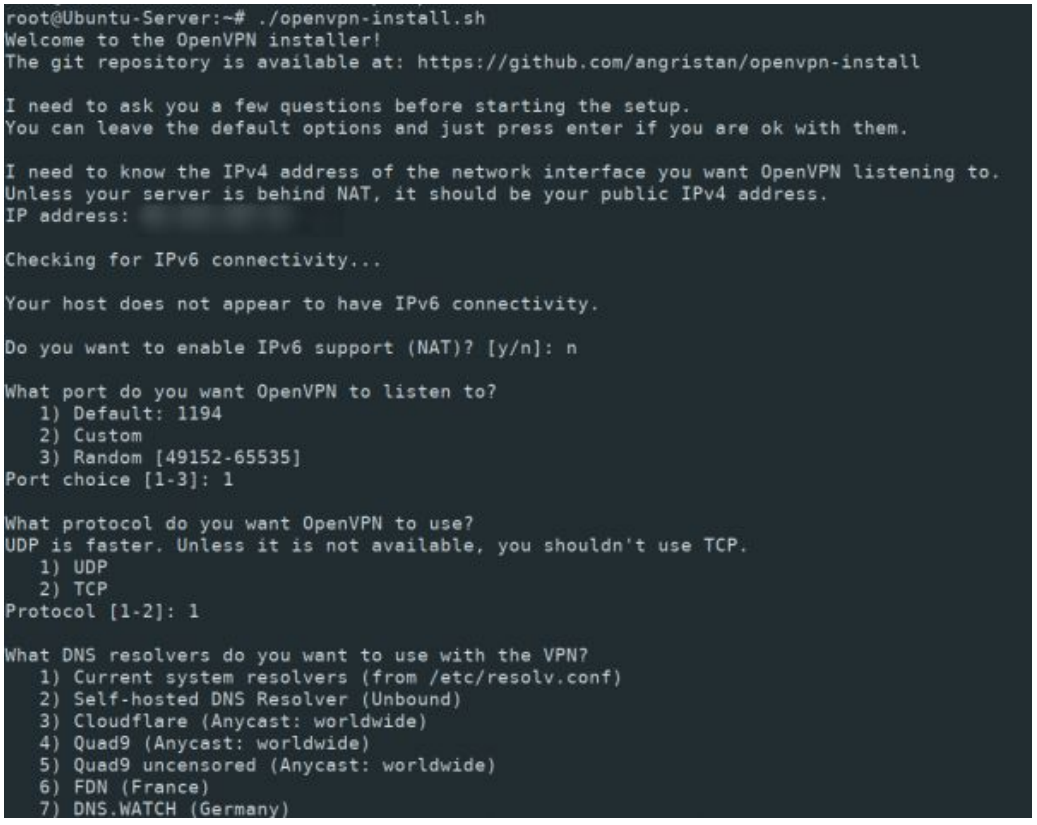
Make it executable.

```
chmod +x openvpn-install.sh
```

Now run the script.

```
./openvpn-install.sh
```

You will be asked to confirm some parameters that have optimal values by default. The only thing that is really worth checking is the public IP of the server. Other parameters should only be changed if you understand what you are doing and why.



Screen 1. OpenVPN installation settings.

How to get the public IP for the configuration of the VPN

Get the public IP of your router. To get that data, just log into the robot computer and type the following command:

```
host myip.opendns.com resolver1.opendns.com
```

The public IP will appear on the screen. Use that number during the config phase above.

Also, you need to decide the port at which to attach the server. We will use the default one because it is running in the robot.

In the last step, you need to set the client name and choose whether to protect the configuration with a password or not. For security reasons, it's better to set a password.

Now reboot the computer.

When the process is over, you can check whether the OpenVPN server is listening for incoming connections.

```
ss -tupln | grep openvpn
```

```
root@Ubuntu-Server:~# sudo ss -tupln | grep openvpn
udp     UNCONN   0        0               0.0.0.0:1194          0.0.0.0:*       users:(("openvpn",pid=1895,fd=7))
```

Screen 2. OpenVPN server is listening for incoming connections.

After this configuration, the OpenVPN server will start automatically everytime that you switch on the robot.

Also, very important, by doing that installation, you have created a VPN config file.
You will need this file later in order to connect to the robot.

WARNING: sharing the VPN config file is a dangerous thing because the config file can be running around the world and anybody can have access to it and try to connect to your robot. We are at present implementing an option at The Construct by which you will not have to share the config file with any student. To be released in March 2021. Please stay tuned.

WARNING-2: Never provide to your students the password for login into to the robot. If you provide that, they will be able to access it remotely using the VPN and then login into it.
STUDENTS DO NOT NEED THE LOGIN PASSWORD TO USE REMOTELY THE ROBOT!
Please, never, ever provide to them the password or your robot can end in a very bad status (because students log in, and crash the system due to their lack of knowledge).

EXTRA NOTE: HOW TO UNINSTALL OVPN

In case you need to remove the OVPN server from the robot, use the following commands:

```
$ service openvpn stop
$ sudo rm -rf /etc/openvpn/*

$ sudo rm -rf /var/log/openvpn
$ sudo apt purge -y openvpn
```

3. Add an external camera

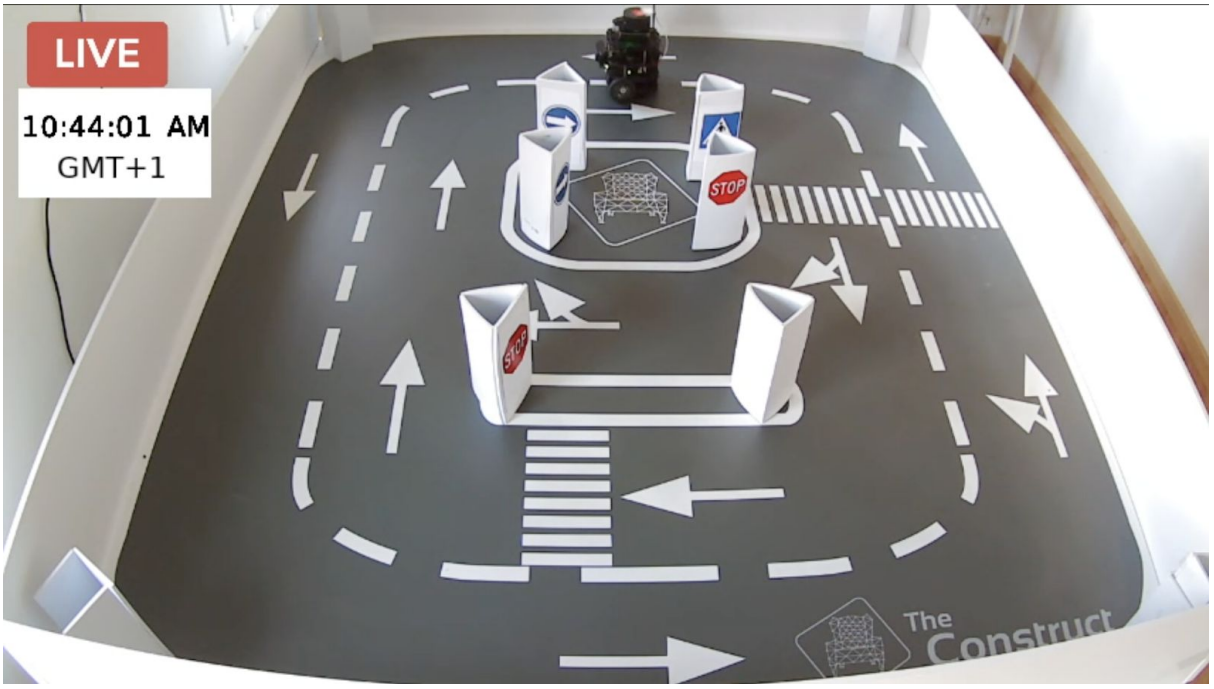
You will need to have an external camera that broadcasts the robot situation to your students so they can actually see what the robot is doing when they are running their program.

For that, you will need two things: the external camera (any webcam would do it) and an external computer that connects the USB camera and broadcasts to the world.

For the external camera, select one that suits you and connect it to the external computer. You may need an USB cable extension. Put the camera in a location that allows a full view of the environment and the robot.

For the broadcast program, since the number of students will be small, you do not need a very good broadcaster. One option is to use VLC. Check the instructions here, to setup your live stream of the camera. **Just pay attention that this can be a security risk** because anybody on the Internet can connect to it. Use it at your own risk.

In The Construct Remote Real Robot Lab, we use a more complex setup to provide more users access as well as security.



B. PREPARE THE ROUTER

It is very likely that your robot is connected to the internet through a router somewhere in the network of your University or Lab. **You must be able to have admin access to the router** (or at least, know the name of the person who can access it). We at The Construct, we are preparing a new connection method that doesn't require this step (to be included in the March release), but at present, this step is mandatory.

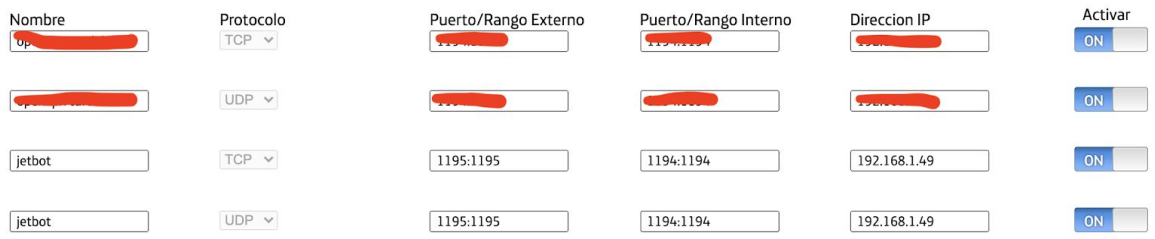
What we are going to do now is the following:

1. When a student tries to connect to the robot using the VPN config file (see more instructions below), he will be trying to connect to the VPN server. That server, if

installed properly in the robot will be running on the default port 1194 of the robot computer.

2. However, when the student tries to establish connection, his request will be stopped by the router.
3. Hence, you must log in into the router and change its configuration to indicate that:
    a. When there is a call coming from the internet to the 1194 port, this call has to be redirected to the IP of the robot.
4. First get the IP of your robot inside the wifi network. To do that, log in into the robot and type the command:
    a. Ifconfig
    b. You will see the IP of your robot in the *wlan* device
    c. example:

5. Then log into the router and go to the tables configuration. This step is different for every router. Let me put some screen shots of our router hre to serve as a reference for you
6. Once in the table configuration, you must indicate the following rule:
    a. When getting calls for the 1194 port, redirect to the IP of your robot



How to test

- Now in order to test this working, you can go to your account at The Construct
- Launch a rosject (any would do. This is only for testing purposes).
- Once the rosject is open, launch the IDE.
- Use the IDE to upload the VPN config file to your home directory
- Launch a terminal
- On the terminal, type the following command (from your home directory)

```
Sudo openvpn --config /path/to/configuration.ovpn
```

At that point, you should be connected to your robot through VPN. Make sure you have the robot switched on, running a roscore, and connected to the wifi.Test now that you can see the robot topics. For that, first you need to get the IP the robot has on the VPN. For that, go to the robot computer and type th command:

```
ifconfig
```

Then on the output, look for the `tun` interface, and take note of the IP. We well call it the ROBOT_TUN_IP.

Now go back to your local computer. Open another terminal and type the following:

```
export ROS_MASTER_URI=http://ROBOT_TUN_IP:11311
rostopic list
```

At this point, you should see the list of topics on your robot.

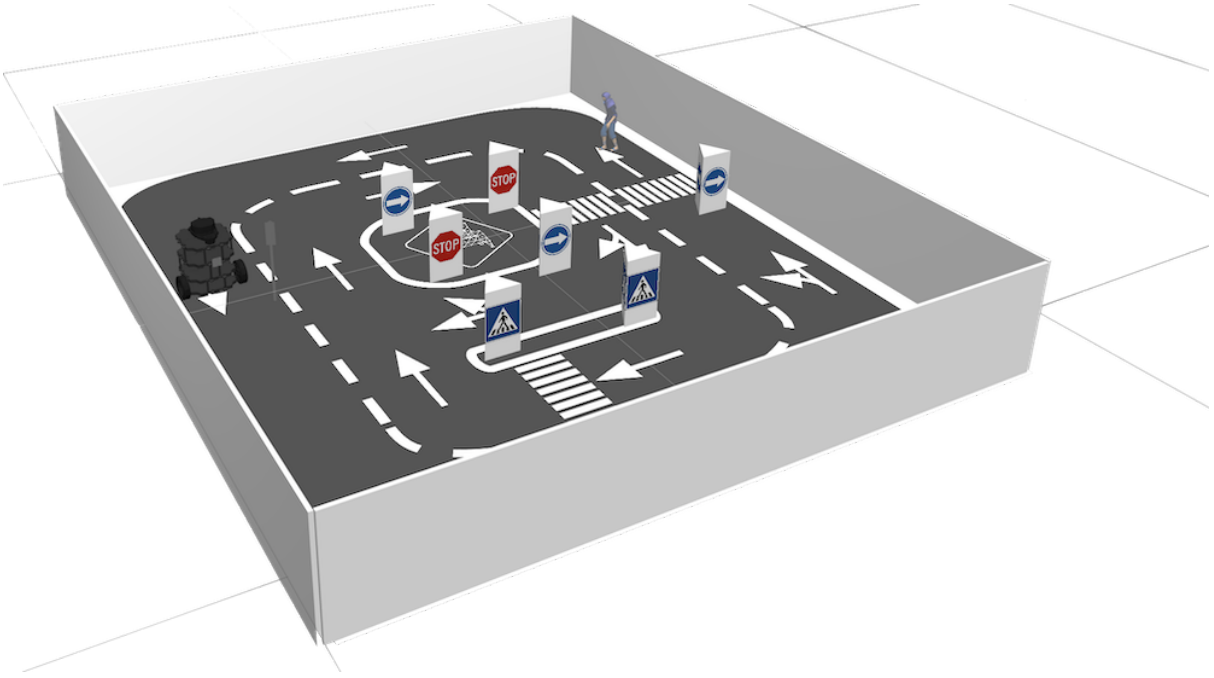
C. PREPARE THE STUDENTS

Provide a proper rosject to students

You, as the teacher, need to create a rosject containing the simulation of the real environment at which your students will be practicing. It is mandatory that they test their ROS programs in the simulation and make them work there before trying on the real robot. If you do not enforce this policy, they will be running programs in the robot that make it do strange things like crashing against the wall at full speed.

Steps:

1. Create a new empty rosject. We recommend you to make it private because it is going to contain information about how to connect to your robot. Remember that private rosjects remain private even if you share with other people. And people who receive a private rosject cannot share it with other people
2. with the simulation as close as possible to the real environment. One example is our rosject for our real lab, whose rosject can be found here. Also make sure the names of the topics of the simulated robot and the real one match each other. Another source of mismatching is the TF names and values. Make ssure the TF names and values are the same for both real and simulation



3. Include in the rosject the VPN config file. Use the IDE of the rosject to upload the config file to it. Save it on the home folder (*/home/user*). Then make it hidden by modifying its name to *.config.ovpn*
4. Include instructions in the notebook about how to launch the simulation and how to connect to the real robot.

5. Once you have that rosject done, distribute it among your students. To share, go to the list of your rosjects and click on the rosject you have prepared. Then click on *Share* and copy the link that appears. Send that link to your students so they can have a copy of it.
6. On the day of practice, make the student open the rosject. The student must prepare his ROS program and test with the simulation.
7. Once his program is working on the simulation, you can make him connect to the real robot environment. For that, first make him close the simulation (to avoid conflicts with the real robot).
8. Then make him open a terminal and type:

```
sudo openvpn --config /home/user/.config.ovpn
```

At this point, the ROS cloud environment is connected to the VPN of your robot. This means that it is possible to see the topics and services of the robot and control it from within the rosject.

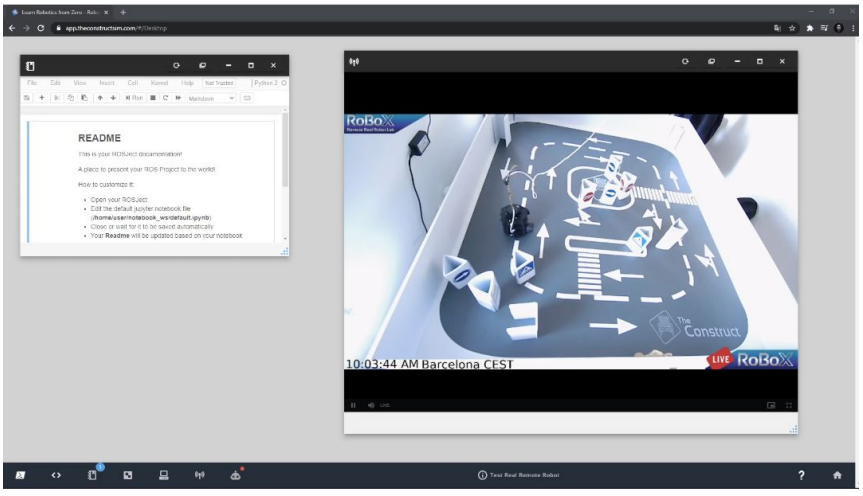
Let's see how:

1. Now, on every terminal that he opens he must change the ROS_MASTER_URI to point to the robot. To do that, the student must type on every terminal that he wants to use:

```
export ROS_MASTER_URI=http://ROBOT_TUN_IP:11311
export ROS_IP=COMPUTER_TUN_IP
export ROS_HOSTNAME=COMPUTER_TUN_IP
```

Now the student's cloud computer is receiving the topics of the real robot

1. Now the student can launch his program and see the results on the real robot



Final considerations

We assume you are not doing a 24/7 lab. For that, you would need a more complex scenario

You will have to manage the access time. This means that you will need somebody to be taking care of the lab, even once everything is properly setup. This includes:

1. Opening the lab at selected times according to your criterias. Opening the lab basically entails:
    a. switching on the robot at your lab times, and off after those times
    b. Keeping the robot batteries charged
    c. Managing robot repairs
2. While the lab is open, control students access through a chat. Current configuration of VPN only allows a single person to be connected. However, you will need to policy that and make sure that the person connected is the one that should. Use a live chat with your students for that (I personally did that with my students and explained it all here).
3. Students may do strange things in the robot. You will need to:
    a. Make sure the student that is about to test has tested his program first on the simulation and that it works as expected. If not the case, do not allow him to test on the real
    b. Still, some programs will do stupid things (that's kind of inevitable). You will need somebody there to be recovering the robot and environment when something goes wrong.
    c. Also take into account that students can make the robot do things that can break it. Depending on the complexity of your robot, you may need to include in the robot protection programs (for instance, in our robot we have a speed limiter, that prevents people sending large speeds).

Problems while setting the environment up

1. You don't have a static IP in your router. (this will be also solved on The Construct March update).
2. You have a more complex network than just a router. (this will be also solved on The Construct March update).