RP7 Application Documentation

**Safety!!!**

A few things to keep in mind when working with the RP7i:

* It weighs a lot and moves with a good amount of force
* There is about a one second delay between what happens in real time versus what you see on the video
* ALWAYS have someone near the robot ready to hit the emergency button (unless you are in a controlled environment, but still be cautious!)
* The sensors will (in most cases) stop the robot’s motion. **BUT** it takes a moment or two for the robot to come to a complete stop
* I’ve tried to write code that would prevent accidents should anything go wrong (erring on the side of make the robot stop) **BUT** in the case that the network connection is unexpectedly lost the robot does not detect it for about 15 seconds (!!!!) The sensors should stop the robot (in most cases as mentioned above) I wasn’t able to find a solution to this in the time that I had.
* The bumpers are not functional and will not detect that something has been hit
* Be careful on inclines. If the robot’s application is shut down or crashes, the brakes disengage, and the robot will roll away. Another good reason to have someone nearby.

**Network:**

**Option 1:**

Connect the client computer and the robot to a router.

**Option 2:**

Set up a hosted network on the client computer following these:

[Open Command Prompt](https://www.lifewire.com/how-to-open-command-prompt-2618089) and enter this command, replacing the italicized works with your own network name and password for the wireless network:

**netsh wlan set hostednetwork mode=allow ssid=***network name* **key=***password*

Start the hosted network directions (This needs to be done each time you boot the computer):

**netsh wlan start hostednetwork**

Stop the hosted network:

**Netsh wlan stop hostednetwork**

(<https://www.lifewire.com/set-up-an-ad-hoc-peer-wifi-network-818272>)

**Software:**

Running software on the Robot:

At this time there are two applications that run on the robot.

1. The Robot Application (to control the robot)
   1. Located at C:\Program Files\IntouchHealth\Installation Utilities\RP Test\NSCC\_main.tcl
   2. Double click to run

**Note:** This application depends on many of the files located in the same directory and subdirectories. If you move it, the other files need to be moved as well or the source paths referenced at the start of this file can be changed.

1. **VLC media player** (only needed if you require video)
   1. Launch **VLC media player** from the desktop shortcut
   2. From the **Media** menu select **Stream…**
   3. Click the **Capture Device** tab
   4. Select **Euresys PICOLO PRO2 sn/77037 – VID1** for the **Video device name**
   5. Click the **Advanced options** button
   6. Enter **25** for the **Video input frame rate** and click **Ok**
   7. Click **Stream**
   8. Click **Next**
   9. Check **Display locally** (if you want to see the video streaming on the robot)
   10. Select **RTSP** from the drop-down menu and click **Add**
   11. Following the “/” in the **Path** field enter “rp7.stream” and click **Next**
   12. Click **Next** again and then click **Stream**

Running software on a remote client computer:

1. Open the **RpApplication** solution in visual studio
2. Click **Start** in the tool bar
3. Select **Connect** from the **Network Menu**
4. Enter the robots IP address and click **Connect**
5. You should now be connected. The video stream takes several seconds to start.

**Controls:**

**Motion**

W = Forward

S = Backward

A = Rotate Left

D = Rotate Right

Q = Strafe Left

E = Strafe Right

**Head**

I = Tilt Up

K = Tilt Down

J = Pan Left

L = Pan Right

Enable mouse control in the **Head** menu.

Click and drag on the video panel to control head movements.

Double click on the video panel to center head.

**Sounds**

Sound bank can be changed in the **Sounds** menu.

0 – 9 = Play sounds

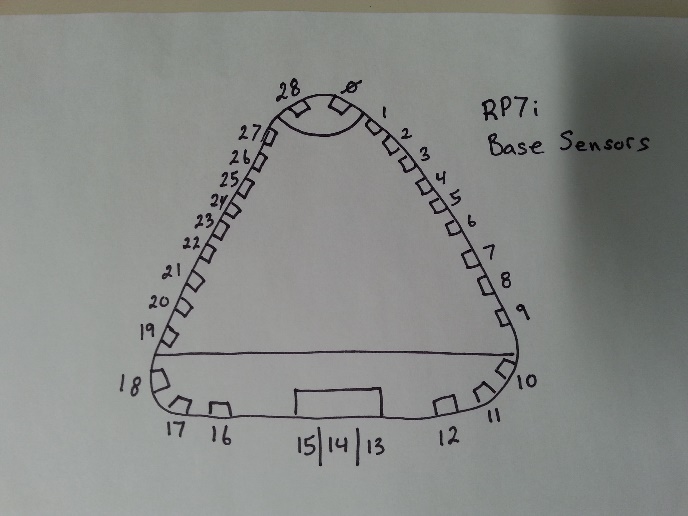
**Text-To-Speech Messages**

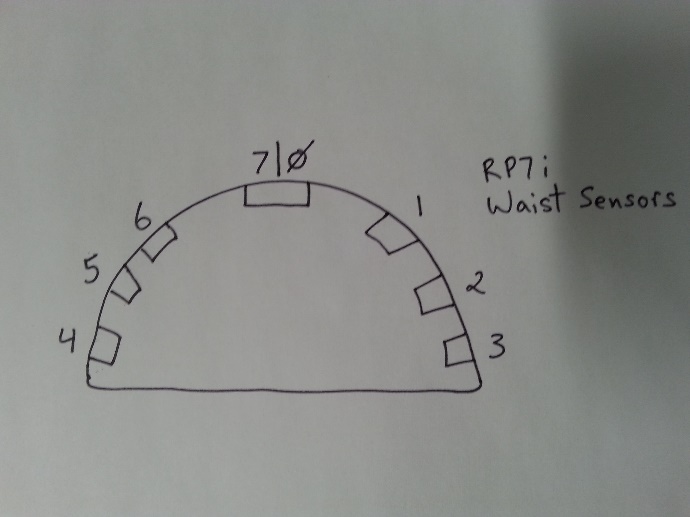
M = Puts focus on the TTS message box

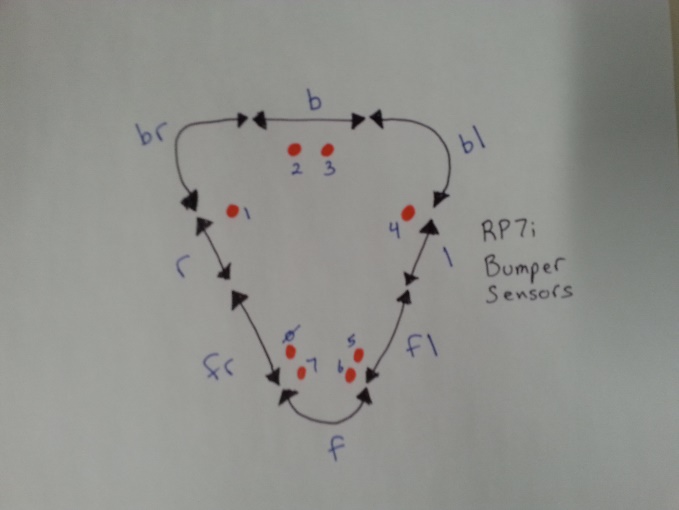
Enter = Sends the message

**RP7i Sensors**

The following diagrams show how the sensors are indexed in the arrays in the TCL code. The bumpers are reference both a **logical** and **physical. Logical** refers to the bumper area (b = back, f = front, r = right etc.). **Physical** must be the actual sensors that are triggered (shown in red). I didn’t get around to writing much code for the bumpers. The diagram shows my best interpretation from reading the code. More testing is needed to see if the diagram is correct.







**Things to work on**

* Get the bumper sensors working
* Sometimes communication between the client and robot happens to quickly and the stream ends up with more than one command (mostly when controlling the head with the mouse)
* Video stream from the client computer to the robot
* Stream sound
* Allow the user to select which camera the video comes from
* Add another element to the application that works in between the client and the TCL code to take advantage of the strengths of another language. Todd suggested maybe adding a second computer to the robot so that it can run a more recent OS.
* Figure out a better way to stream the video with better performance and quality
* Allow the user to adjust the speed of the robot
* Allow the user to disengage/override the sensors if it gets stuck in a tight space
* Make the GUI prettier
* Get the volume knob working
* Add a better wireless card to improve the video streaming
* Make the GUI fit all screen sizes

**Backups**

* The original HD from the RP7 has a few bad sectors but was still working. Labelled “RP7i original”.
* A clone of the original hard drive with bad sectors skipped. Labelled “RP7i Backup”.
* Another hard drive that contains two images of the original HD. One is 100GB in size the other 160GB. Labelled “RP7i Images”.
* The images are also on Todd’s Samsung SSD.
* All code is pushed to the NSCC’s GitHub and Michael Robinson’s GitHub. And copied to Todd’s Samsung SDD.