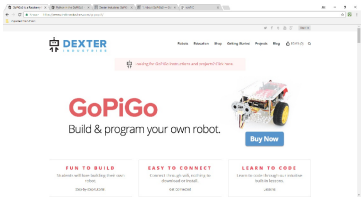
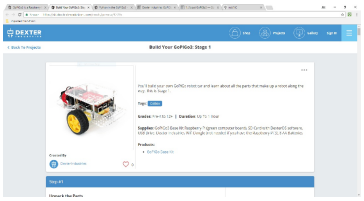
GoPiGo Experience
James H Phelan
2018.01

https://www.dexterindustries.com/gopigo3/
Select: Fun To Build, step-by-step tutorial

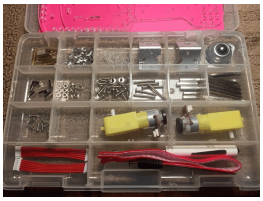


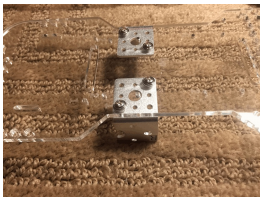
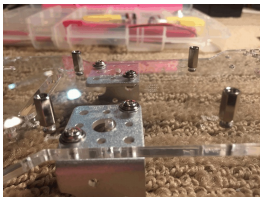
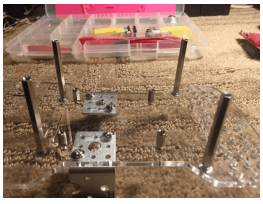
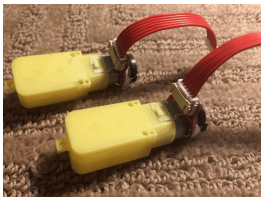
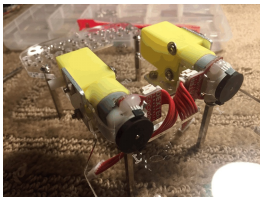
https://studio.dexterindustries.com/cwists/preview/1222x
Scroll down while you follow the instructions

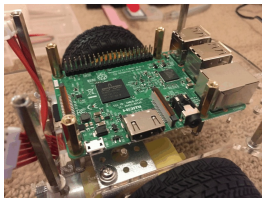
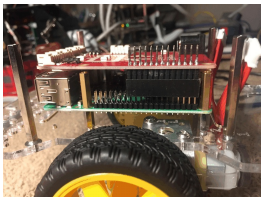
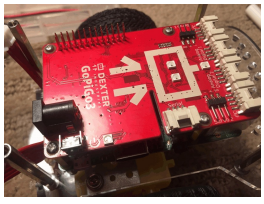
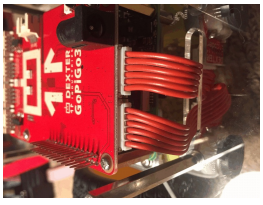


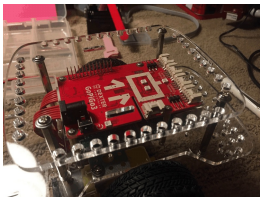


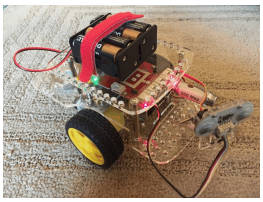
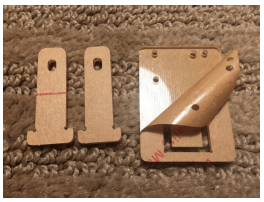
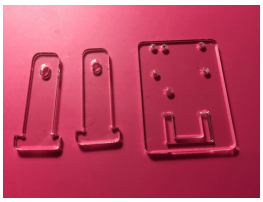
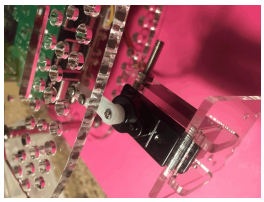
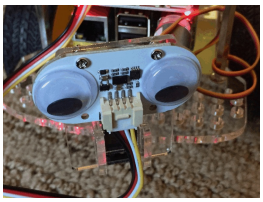


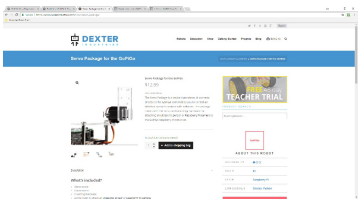






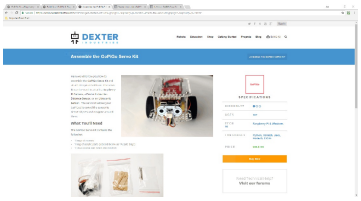


You need to look up Servo Kit for these instructions
https://www.dexterindustries.com/shop/servo-package/

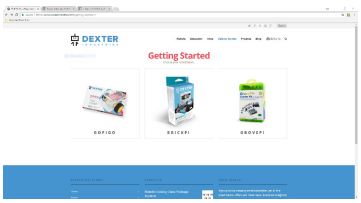


Assembly instructions can be found here:
https://www.dexterindustries.com/GoPiGo/get-started-with-the-gopigo3-raspberry-pi-robot/6-attach-the-servo-kit-gopigo3-raspberry-pi-robot/

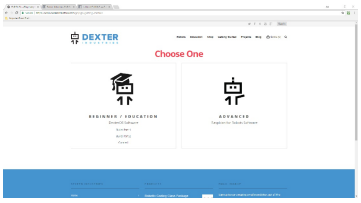


Connecting to the Robot
If you're using Raspbian for Robots instead of DexterOS DO NOT follow directly from assembly to Connecting! It won't work! Instead, go back to Getting Started:

https://www.dexterindustries.com/getting-started-2/
Choose GoPiGo.

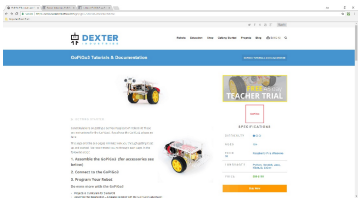


https://www.dexterindustries.com/gopigo-getting-started/
Choose ADVANCED Raspbian for Robots Software

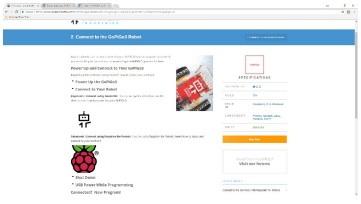
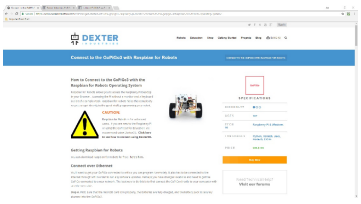


That will take you here:
https://www.dexterindustries.com/gopigo3-tutorials-documentation/

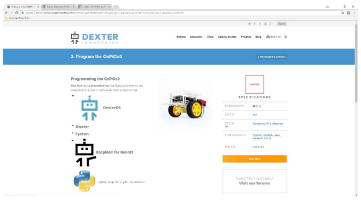
Scroll down to 2. Connect to the GoPiGo3
Which will take you here:



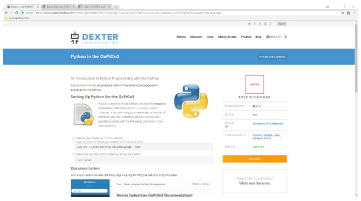
https://www.dexterindustries.com/GoPiGo/get-started-with-the-gopigo3-raspberry-pi-robot/2-connect-to-the-gopigo-3/
Expand Connect to Your Robot
Skip over Beginners: Connect using DexterOS and click on
Advanced: Connect using Raspbian for Robots
and finally you'll get here:
https://www.dexterindustries.com/GoPiGo/get-started-with-the-gopigo3-raspberry-pi-robot/2-connect-to-the-gopigo-3/Raspbian-For-Robots-Operating-System/

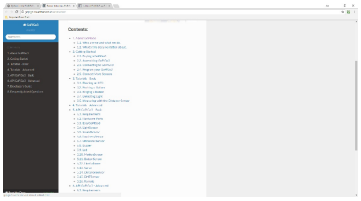
Connect over Ethernet
Unlike DexterOS, Raspbian for Robots doesn't set up its own GoPiGo network.
You have to connect using the ethernet cable, set up a network to join and connect to it.
Then you can disconnect the ethernet cable & log into the GoPiGo using WiFi.
Just follow the directions.

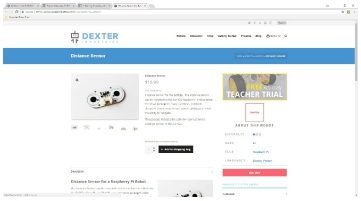


Programming the GoPiGo
At the end of Connect, select "Program the GoPiGo here"
and end up here:
https://www.dexterindustries.com/GoPiGo/get-started-with-the-gopigo3-raspberry-pi-robot/3-program-your-raspberry-pi-robot/
Expand Raspbian for Robots to reveal Python Programming for the GoPiGo3
select and end up here:
https://www.dexterindustries.com/GoPiGo/get-started-with-the-gopigo3-raspberry-pi-robot/3-program-your-raspberry-pi-robot/python-programming-language/
Skip over Setting Up Python for the GoPiGo3 *unless* you're installing on your own version of Raspbian instead of Dexter's Raspbian for Robots.

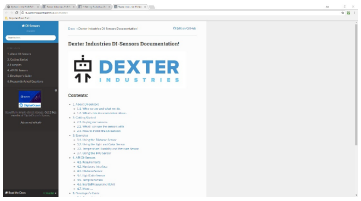


Click the line under Documentation and go here:
http://gopigo3.readthedocs.io/en/master/
Since you've already assembled & connected your GoPiGo3,
skip down to 2.4 Program your GoPiGo3 and proceed from there.
Since I have a distance sensor, I'm going to update the GoPiGo3 library:
sudo sh -c "curl -kL dexterindustries.com/update_gopigo3 | bash"
Update successful
and update the DI-Sensors package:
sudo sh -c "curl -kL dexterindustries.com/update_sensors | bash"
Update successful



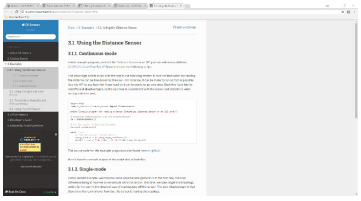


Then on to 2.5 Connect More Sensors, particularly the DI Distance Sensor:
https://www.dexterindustries.com/shop/distance-sensor/
This is just the page to *buy* it. It doesn't say how to *use* it.

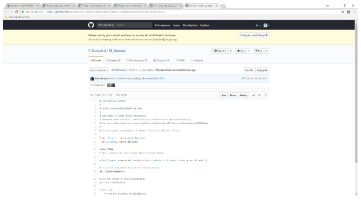


For that go to DI-Sensors documentation:
http://di-sensors.readthedocs.io/en/master/

and from there to Using the Distance Sensor:
http://di-sensors.readthedocs.io/en/master/examples/dist_sensor.html

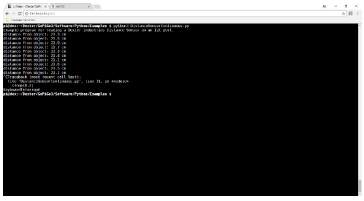


Get from github DistanceSensorContinuous.py using cURL:
https://github.com/DexterInd/DI_Sensors/blob/master/Python/Examples/DistanceSensorContinuous.py
Click the box "raw" and copy the path from the navigation bar.
Enter into terminal window: sudo curl -L -O <paste path here from browser>



pi@dex:~/Dexter/GoPiGo3/Software/Python/Examples **$ sudo curl -L -O https://raw.githubusercontent.com/DexterInd/DI_Sensors/master/Python/Examples/DistanceSensorContinuous.py**
You will then findDistanceSensorContinuous.py in the current directory

Testing DistanceSensorContinuous.py
If you open the program from the desktop by right-clicking & open-with Python3, you won't be able to modify it. But if you launch IDLE3 from the menu, then open the program, you can. You can run the program using F5 from IDLE or from the command window using python3 DistanceSensorContinuous.py.



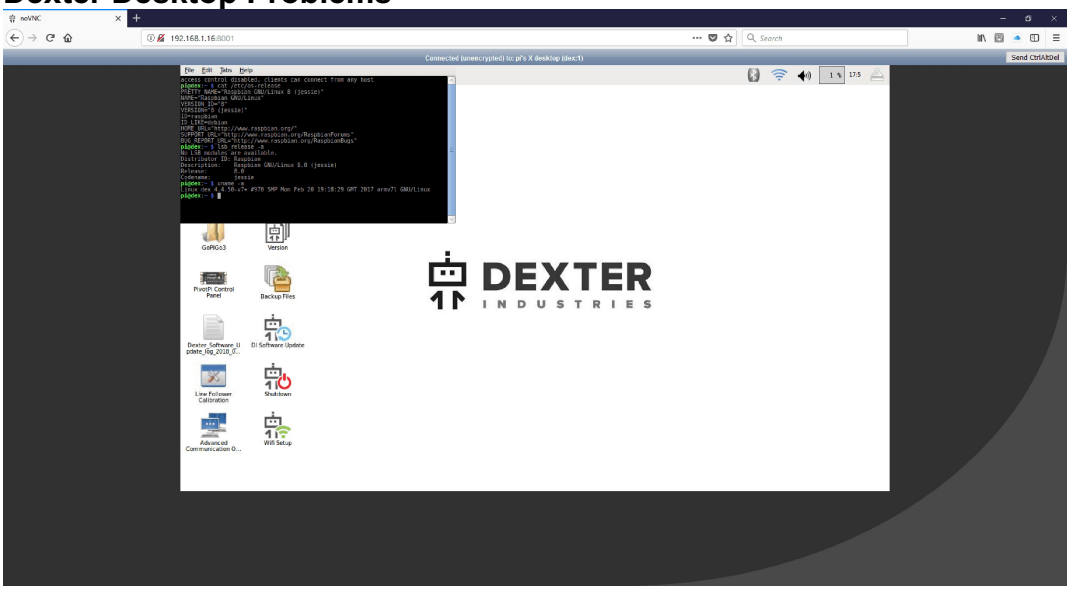
On the display will appear of the continuous distance in mm from the face of the sensor to the nearest object. This can be tested/calibrated placing a ruler with the zero mark below the face of the sensor, and objects at various distance from it and observing the readout.

Testing DistanceSensorSingleShot.py
pi@dex:~/Dexter/GoPiGo3/Software/Python/Examples **$ sudo curl -L -O https://github.com/DexterInd/DI_Sensors/blob/master/Python/Examples/DistanceSensorSingleShot.py**
Works ok.

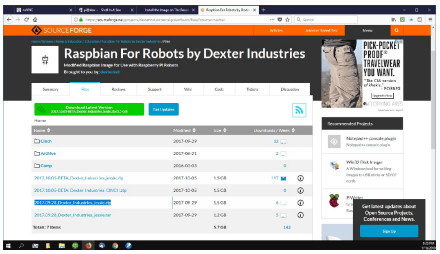
Install Remote Desktop *inbound* server RDP
The Dexter desktop accessed by browser http://dex.local/ is unsatisfactory. The window is less than full screen. Only one window or app can be seen at a time! Initially Remote Desktop, which is much better, didn't work. Then I recalled I had to install the Remote Desktop inbound server. The resulting desktop is at least full screen, but still only one window can open at a time! Comment sent to Dexter forum 2018.01,16.

http://blackeagle12.net/Comp/RPi/RDP.html
$ sudo apt-get install xrdp
Y
Use PC's Remote Desktop app to access 192.168.1.1xx
login as pi (default)
enter *password*
Mouse over center upper edge of window to drop down handle. Click x to close.

Dexter Desktop Problems



Something happened with the Dexter desktop. The menu bar clipped the top of the upper icons. Only one app/window could be opened at a time and couldn't be moved to reveal the menu bar. Correspondence with Cloe at the Dexter Forum agreed this was not normal. Perhaps an improper update using apt-get instead of the Dexter app? It started after downloading the latest Jessie and Beta zip files from:
https://www.dexterindustries.com/howto/install-raspbian-for-robots-image-on-an-sd-card/
Expand --Using a PC, then click on the link:
Raspbian for Robots: from Sourceforge here
Which takes you to:
https://sourceforge.net/projects/dexterindustriesraspbianflavor/
Under the Download box, click on the Files tab, where you see this:
Choose the 2017.09.28_Dexter_Industries_jessie.zip file to download (Unless you want to be "bleeding edge" and get the Beta version – 2017.10.05-BETA_Dexter_Industries_jessie.zip) [THERE IS A NEWER VERSION SINCE 6/2018]



Follow the original directions to unzip & Win32 Disk Imager burn the image to a fresh SD card.

Eject & remove the microSD card & put it in your GoPiGo3.

Follow the original directions to connect via ethernet cable, set up WiFi then go headless.
Open browser and set address as dex.local and choose VPN (or Terminal).
Now you're back where you started.
Configure your Pi for your local area via terminal using "sudo raspi-config"
It seems using the GUI - Menu/Preferences/Raspberry Pi Configuration screws things up:
Go to Park Bench, Do not pass GO, Do not collect $200.

DON'T USE the Windows format option, IT MESSES UP THE CARD making in inaccessible.
Use SDFormatter, NO size adjustment, Overwrite.
Win32DiskImager write the .iso to the card.
Put card in Pi.
sudo raspi-config, then try to figure out the international options! Reboot.
Do DI-Software Update desktop app. Software updates. Choose robot style. Update firmware.
Reboot.
Keyboard & mouse work on the Pi. Windows behave. Punctuation characters display ok.
Let's see what the weekend brings when we get down to programming!

...After long interlude of not documenting, I somehow crashed the system. I think by doing a terminal "sudo apt-get update" instead of doing the desktop Dexter software/firmware update. So I had to re-install from a NEW image from Dexter. (Actually from SourceForge where Dexter directed me.) It's HARD TO FIND the Raspbian for Robots download on Dexter. Google it. Turns out the lastest R4R is Raspbian stretch! Once installed do the Dexter update of firmware & software to be sure. ANY TIME YOU INSTALL SOMETHING NEW that suggests "sudo apt-get update", do the Dexter update instead! Otherwise you can overwrite the Dexter OS, loose things and kill the OS to where it won't even boot, will be missing desktop apps or won't recognize the GoPiGo or Grove Pi hardware.

xrdp
You'll want to re-install **xrdp** as way above to Remote Desktop in from your PC.

LibreOffice
You'll want to install LibreOffice to make use of documentation word processing & spreadsheet tools:
    $ sudo apt-get install libreoffice

Idle
    Options / Configure Idle / Fonts/Tabs / Size (at least 14 for old eyes!)

Dexter directory ownership
If you don't take ownership of the Dexter directory & its subdirectories, you won't be able to modify or save the files. Do this by:
    pi@dex:~ $ sudo chown -R pi:pi Dexter
User "pi" in group "pi" now owns all the Dexter files and can change them.

numpy and
matplotlib
You're going to need these for the fancier python programs. Will prob have to re-install them later. They can be a pain to install. They seem to install ok, but then cause program import errors.
    "ImportError: No module names 'matplotlib'
Yeah, they're gone. Gotta reinstall:
    $ pip3 install matplotlib (install w/o 'sudo' as this can cause ownenership issues)
    $ pip3 install numpy

uv4l
This is a streaming video package you'll want on the GoPiGo to get telepresence.
Watch this guy's videos of how to install the PiCamera & install & configure uv4l:
https://www.youtube.com/watch?v=GzTbm7yo4l4&index=3&list=PLWJfbQMqNj4mIoVblG8SVgCdbKZQ92ZiQ

uv4l

installation & configuration

Display

- on Raspberry Pi:
  - Open browser (ie Chromium, install if you haven't)
  - localhost:8080
- on PC
  - Open browser (ie Chrome)
  - 192.168.1.200:8080 (IP address of Rpi : port of uv4l)

Configure

- on Raspberry Pi:
  - /etc/uv4l/uv4l-raspicam
  - Right click, open with, Custom Command Line
    - sudo leafpad (or your editing tool, ie nano)
    - ok
  - open uv4l-raspicam
    - ### video display options
      - nopreview = no >> yes
    - ### streaming server options
      - # server option = –userpassword=myp4ssw0rd >>
        - server opton = –userpassword=userPass (or your choice)
      - # server option = config-password=myp4ssw0rd >>
        - server option = config-password=confPass (or your choice)
    - If PiCamera is inverted:
      - ### image settings options:
      - hflip = yes
      - vflip = yes
    - save, shutdown, power off & restart
- from browser on Rpi:
  - localhost:8080
    - MJPEG
      - user
      - userPass
    - Control Panel
      - width >> 640
      - height >> 480
      - jpeg quality >> 45
      - [apply]
    - To make PERMANENT -
    - Configuration
      - config
      - confPass
      - (reveals config file w/ changes commented # out)
      - remove #_ and SAVE changes

Dead man brake for GoPiGo

When driving the GoPiGo around the house using streaming video via uv4l and the GoPiGo Control Panel via Remote Desktop, it does fine until sometimes around the kitchen it loses its WiFi connection and no longer responds to commands. It blindly follows the last received command either into a wall, furniture, or around in circles. It needs some sore of "dead man brake" that says "WiFi connection ok, then follow commands. No WiFi, all stop & sound "come find me" alarm.

Followed this advice to maybe minimize problem, but don't think that's it since wifi running up until lost: https://gist.github.com/mkb/40bf48bc401ffa0cc4d3

This might be an idea: https://stackoverflow.com/questions/3764291/checking-network-connection

$ iwlist wlan0 scan | egrep "Quality|NUTHOUSE"
    Quality=52/70 Signal level=-58 dBm
    ESSID:"NUTHOUSE"
    Quality=70/70 Signal level=-40 dBm
    ESSID:"NUTHOUSE"
    Quality=34/70 Signal level=-76 dBm
    Quality=31/70 Signal level=-79 dBm

Let's look at
    ~/Dexter/GoPiGo3/Software/Python/Examples/GoPiGo3_Control_Panel.py

    Even though the desktop app clearly works, if I make a copy of is called ..._PHELAN.py and try to run it, I get a wxPython module doesn't exist. Same if I try to run the original program. Wx is there deep in the bowels of python if you search
    $ sudo find / -name wx
Maybe it's an ownership issue again. Let me try running it directly, and not via Remote Desktop.
    No that doesn't work. The original is recognized as a "python script" and will run if Right-Click Open, Execute. _PHELAN does not?!
I want to make an improved Desktop control panel. Is there some secret incantation?

[Leaving this for now. Next project is installing OpenCV. See below.]

OpenCV Installation

There are many ways to install OpenCV. I did a Google search “raspberry pi opencv install script” and limited to Tools/Past Year. I chose this one because 1) he used a virtual environment for safety and 2) I didn’t trust install scripts to work correctly. I wanted to see any install errors as they happened.

https://www.pyimagesearch.com/2017/09/04/raspbian-stretch-install-opencv-3-python-on-your-raspberry-pi/

I pulled up the web page on the GoPiGo3 so I could copy/paste while I watched the video on the laptop.

I also started a script recording of the session so I could review:

$ script OpenCV_Install.txt

to exit the script ie before rebooting, just

$ exit

if you need to add to the script, ie /p reboot, append the script

$ script -a OpenCV_Install.txt

Per instruction above...
Expand file system
Exit script
Reboot

Do DI Software Update using desktop app instead of
$ ~~sudo apt-get update && sudo apt-get upgrade~~
as they can screw up the Dexter OS

Update Dexter Software. Firmware Update Robot GoPiGo3

Reboot

Resume script -a
Bring browser & above web site back up.

$ sudo apt-get purge wolfram-engine
Not installed, not removed
$ sudo apt-get purge libreoffice*
No, I want to keep this
$ sudo apt-get clean
Done
$ sudo apt-get autoremove
Done

$ sudo apt-get install build-essential cmake pkg-config
    Done
$ sudo apt-get install libjpeg-dev libtiff5-dev libjasper-dev libpng12-dev
    Done
$ sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libv4l-dev
    Done
$ sudo apt-get install libgtk2.0-dev libgtk-3-dev
    Done
$ sudo apt-get install libatlas-base-dev gfortran
    Done
$ sudo apt-get install python2.7-dev python3-dev
    These are probably already installed, but... Yes.
    Done
$ cd ~
    already there
$ wget -O opencv.zip https://github.com/Itseez/opencv/archive/3.3.0.zip
    Latest release is 3.4.2. Changed numbers to match:
    $ wget -O opencv.zip https://github.com/Itseez/opencv/archive/3.4.2.zip
    Done
$ unzip opencv.zip
    Done
$ wget -O opencv_contrib.zip https://github.com/Itseez/opencv_contrib/archive/3.3.0.zip
    Done.
$ unzip opencv_contrib.zip
    Done
    Oops! Should have been 3.4.2. Try again.
$ wget -O opencv_contrib.zip https://github.com/Itseez/opencv_contrib/archive/3.4.2.zip
    Done
$ unzip opencv_contrib.zip
    Done
$ wget https://bootstrap.pypa.io/get-pip.py
    Done
$ sudo python get-pip.py
    Uninstalled 9.0.1
    Installed 10.0.1
    Done
$ sudo python3 get-pip.py
    Cache entry deserializaton failed, entry ignored
    Cache entry deserializaton failed, entry ignored
    Uninstalled 9.0.1
    Installed 10.0.1
    Done

"If you're a longtime PyImageSearch reader, then you'll know that I'm a huge fan of both virtualenv and virtualenvwrapper. Installing these packages is not a requirement and you can absolutely get OpenCV installed without them, but that said, I highly recommend you install them as other existing PyImageSearch tutorials (as well as future tutorials) also leverage Python virtual environments. I'll also be assuming that you have both virtualenv and virtualenvwrapper installed throughout the remainder of this guide.

So, given that, what's the point of using virtualenv and virtualenvwrapper ?

First, it's important to understand that a virtual environment is a special tool used to keep the dependencies required by different projects in separate places by creating isolated, independent Python environments for each of them.

In short, it solves the "Project X depends on version 1.x, but Project Y needs 4.x" dilemma. It also keeps your global site-packages neat, tidy, and free from clutter.

If you would like a full explanation on why Python virtual environments are good practice, absolutely give this excellent blog post on RealPython a read.

It's standard practice in the Python community to be using virtual environments of some sort, so I highly recommend that you do the same:"

$ sudo pip install virtualenv virtualenvwrapper
    Done
$ sudo rm -rf ~/.cache/pip
    For cleanup
    Done

Now that both virtualenv and virtualenvwrapper have been installed,
we need to update our **~/.profile** file to include the following lines at the bottom of the file:

$ sudo nano /.profile [already at ~]

    # virtualenv and virtualenvwrapper
    2 export WORKON_HOME=$HOME/.virtualenvs
    3 export VIRTUALENVWRAPPER_PYTHON=/usr/bin/python3
    4 source /usr/local/bin/virtualenvwrapper.sh
^X, Y

Now that we have our ~/.profile updated, we need to reload it to make sure the changes take affect. You can force a reload of your ~/.profile file by:

Logging out and then logging back in.
Closing a terminal instance and opening up a new one
Or my personal favorite, just use the source command:

$ source ~/.profile

Note: I recommend running the source ~/.profile file each time you open up a new terminal to ensure your system variables have been setup correctly.

$ source ~/.profile
Done

Creating your Python virtual environment
Next, let's create the Python virtual environment that we'll use for computer vision development:

$ mkvirtualenv cv -p python2
bash: mkvirtualenv: command not found

$ sudo find / -name mkvirtualenv
nothing
$ sudo find / -name virtualenv
/usr/local/bin/virtualenv
... so it was installed

Googled:"bash: mkvirtualenv: command not found raspbian" and chose this site:

http://vicpimakers.ca/links/python-links/setting-up-python-projects-with-virtual-environments/

Gave helpful hint, see below

Reboot

append script

In the comments on the install web page there was this:
Danny March 8, 2018 at 1:17 pm #
I am using Python 3 and was able to fix the problem by running this command:

$ sudo pip3 install virtualenv virtualenvwrapper

using pip3 instead of pip to make the virtual env.

$ sudo pip**3** install virtualenv virtualenvwrapper

...Requirement already satisfied
..."...

$ mkvirtualenv cv -p python**2**
bash: mkvirtualenv: command not found
$ mkvirtualenv cv -p python**3**
bash: mkvirtualenv: command not found

$ source ~/.profile
$ workon cv
bash: workon: command not found

Adding sudo... didn't change anything

From --
http://vicpimakers.ca/links/python-links/setting-up-python-projects-with-virtual-environments/

"Create a Virtual Environment in $WORKON_HOME
$ virtualenv newenv -p python3

forces the installation of the latest Python 3. You can specify any version of Python currently installed on your computer with:

$ virtualenv -p python3.2 newenv

You must first activate the Virtual Environment before you start installing software packages into it. This keeps your system Python entirely separate from your project. It makes your project self-contained within its directory and thus transportable. "

Substitute cv for newenv
So instead of
$ mkvirtualenv cv -p python**3**
it should be

$ virtualenv cv -p python**3**
Already using interpreter /usr/bin/python3
Using base prefix '/usr'
New python executable in /home/pi/cv/bin/python3
Also creating executable in /home/pi/cv/bin/python
Installing setuptools, pip, wheel...done.

My comment to the install web site comments section:
mkvirtualenv error
bash: mkvirtualenv: command not found
I had the same problem as in the Troubleshooting and FAQ section.
After Googling the problem I came upon:
http://vicpimakers.ca/links/python-links/setting-up-python-projects-with-virtual-environments/
which said to use — (substitute "cv" for "newenv"
virtualenv newenv -p python3
instead of
mkvirtualenv newevn -p python3
I tried it and it worked, creating the cv directory and contents.
Apparently recently mkvirtualenv got changed to just virtualenv???

Then:
…but
$ workon cv
bash: workon: command not found

so we're one step closer, but not there yet.

And further:
….but then they say:
Activate a Virtual Environment
You must do this from within the Virtual Environment directory:

bash$ cd $WORKON_HOME/newenv
bash$ source bin/activate
instead of
$ workon cv
it would be
$ cd cv
$ source bin/activate
(cv) pi@dex:~/cv $

bingo!

The directions from Victoria to deactivate using
(cv) pi@dex:~/cv *$ source bin/deactivate*
is wrong
it's just
(cv) pi@dex:~/cv *$ deactivate*
pi@dex:~/cv *$*

After this, then I tried
~/cv $ source ~/.profile
/usr/bin/python: No module names virtualenvwrapper
virtualenvwrapper.sh: There was a problem running the initialization hooks:
If Python could not import the module virtualenvwrapper.hook_loader,
check that virtualenvwrapper has been installed for
VIRTUALENVWRAPPER_PYTHON=/usr/bin/python and that PATH is
set properly.

When I look back at ~/.profile, my my added lines are GONE and new lines are added by the systemt to the previously BLANK file! So I added them back at the bottom as before.

NOW mkvirtualenv

FINALLY my virtual environment is installed! (I hope..!) To carry on w/ OpenCV:

(cv) pi@dex: ~/cv $ pip install numpy
Done
Should this be pip2 install numpy?
...Requirement already satisfied...
~~(cv) pi@dex: ~/cv $ cd ~/opencv-3.3.0/~~
(cv) pi@dex: ~/cv $ cd ~/opencv-3.4.2/
Done
(cv) pi@dex: ~/opencv-3.4.2 $ mkdir build
Done
(cv) pi@dex: ~/opencv-3.4.2 $ cd build
Done
(cv) pi@dex: ~/opencv-3.4.2/build $ cmake -D CMAKE_BUILD_TYPE=RELEASE \
-D CMAKE_INSTALL_PREFIX=/usr/local \
-D INSTALL_PYTHON_EXAMPLES=ON \
-D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib-3.3.0/modules \
-D BUILD_EXAMPLES=ON ..

Done with errors (unlike example)
see /home/pi/opencv-3.4.2/build/CMakeFiles/CMakeError.log & .../CmakeOutput.log
Probably the result of Raspbian for Robots instead of regular Raspbian stretch.

Open your /etc/dphys-swapfile and then edit the CONF_SWAPSIZE variable:
$ sudo nano /etc/dphys-swapfile
# CONF_SWAPSIZE=100
CONF_SWAPSIZE=1024
$ sudo /etc/init.d/dphys-swapfile stop
$ sudo /etc/init.d/dphys-swapfile start
Done

Note: It is possible to burn out the Raspberry Pi microSD card because flash memory has a limited number of writes until the card won't work. It is highly recommended that you change this setting back to the default when you are done compiling and testing the install (see below). To read more about swap sizes corrupting memory, see this page:
https://www.bitpi.co/2015/02/11/how-to-change-raspberry-pis-swapfile-size-on-rasbian/

(cv) pi@dex: ~/opencv-3.4.2/build $ make -j4
make: * No targets specified and no makefile found. Stop.**
Prob @ errors on –
cmake -D CMAKE_BUILD_TYPE=RELEASE \
-D CMAKE_INSTALL_PREFIX=/usr/local \
-D INSTALL_PYTHON_EXAMPLES=ON \
-D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib-**3.3.0**/modules \
-D BUILD_EXAMPLES=ON ..
The 3.3.0 should be **3.4.2**. Changed & re-run w/ some awkwardness & errors.

– Configuring done
– Generating done
– Build files have been written to: /home/pi/opencv-3.4.2/build

(cv) pi@dex: ~/opencv-3.4.2/build $ make -j4
This time it's rocking!

Good night!...

...next am:
...
[100%] Built target opencv-python3
[100%] Built target opencv-python2
(cv) pi@dex: ~/opencv-3.4.2/build $
Done!
I didn't have any overheating problem (that I know of, I was asleep!) using the -j4, unlike the video.

(cv) pi@dex: ~/opencv-3.4.2/build $ sudo make install
Done
(cv) pi@dex: ~/opencv-3.4.2/build $ sudo ldconfig
Done

For Python 2.7:
(cv) pi@dex: ~/opencv-3.4.2/build $ ls -l /usr/local/lib/python2.7/site-packages/
total 0 (not 1852 as expected)
Note: In some cases, OpenCV can be installed in /usr/local/lib/python2.7/dist-packages (note the dist-packages rather than site-packages). If you do not find the cv2.so bindings in site-packages , we be sure to check dist-packages .
(cv) pi@dex: ~/opencv-3.4.2/build $ ls -l /usr/local/lib/python2.7/**dist**-packages/
total 5212
(cv) pi@dex: ~/opencv-3.4.2/build $ cd ~/.virtualenvs/cv/lib/python2.7/site-packages/
bash: ~/.virtualenvs/cv/lib/python2.7/site-packages/: **No such file or directory**
I'm thinking this should be **dist**-packages/
(cv) pi@dex: ~/opencv-3.4.2/build $ cd ~/.virtualenvs/cv/lib/python2.7/site-packages/
bash: ~/.virtualenvs/cv/lib/python2.7/dist-packages/: **No such file or directory**
Looking in File Manager, in ~/.virtualenvs/cv/lib/ there's a directory for python3.5 but not for python2.7
Well, never mind python2, I really just want python3

For Python 3:
(cv) pi@dex: ~/opencv-3.4.2/build $ ls -l /usr/local/lib/python3.5/site-packages/
total 4576
-rw-r--r– 1 root staff 4683596 Jul 16 00:36 **cv2.cpython-35m-arm-linux-gnueabihf.so**

I honestly don't know why, perhaps it's a bug in the CMake script, but when compiling OpenCV 3 bindings for Python 3+, the output .so file is named cv2.cpython-35m-arm-linux-gnueabihf.so (or some variant of) rather than simply cv2.so (like in the Python 2.7 bindings).
Again, I'm not sure exactly why this happens, but it's an easy fix. All we need to do is rename the file:

(cv) pi@dex: ~/opencv-3.4.2/build $ cd /usr/local/lib/python3.5/site-packages/
(cv) pi@dex: /usr/local/lib/python3.5/site-packages $
sudo mv cv2.cpython-35m-arm-linux-gnueabihf.so cv2.so
(cv) pi@dex: /usr/local/lib/python3.5/site-packages $ cd ~/.virtualenvs/cv/lib/python3.5/site-packages/

(cv) pi@dex:~/.virtualenvs/cv/lib/python3.5/site-packages $
ln -s /usr/local/lib/python3.5/site-packages/cv2.so cv2.so

Step #7: Testing your OpenCV 3 install
Congratulations, you now have OpenCV 3 installed on your Raspberry Pi 3 running Raspbian Stretch!

But before we pop the champagne and get drunk on our victory, let's first verify that your OpenCV installation is working properly.

*Open up a **new** terminal, execute the source and workon commands, and then finally attempt to import the Python + OpenCV bindings:*

[You have to do this *within* the cv directory]:
pi@dex:~ $ cd cv
pi@dex:~/cv $ source ~/.profile
pi@dex:~/cv $ workon cv
(cv)pi@dex:~/cv $ python
Python 3.5.3 (default, Jan 19, 2017 14:11:04)
...
>>> import cv2
ImportError: numpy.core.multiarray failed to import
...
Crap! I've run into this before with numpy!
Google search suggest it's a numpy version conflict.
Try updating or uninstalling numpy: $ pip -U install numpy
But with python3 it should be $ pip3 install -U numpy
(cv)pi@dex:~/cv $ pip3 install -U numpy
...
Successfully installed numpy-1.14.5

Try again
(cv)pi@dex:~/cv $ python
>>> import cv2
>>> cv2.__version__
'3.4.2'
^D to exit python

Once OpenCV has been installed, you can remove both the opencv-3.3.0 and opencv_contrib-3.3.0 directories to free up a bunch of space on your disk:

$ rm -rf opencv-3.3.0 opencv_contrib-3.3.0
Done

Don't forget to change your swap size back!
Open your /etc/dphys-swapfile and then edit the CONF_SWAPSIZE variable:

(cv)pi@dex:~/cv $ sudo nano /etc/dphys-swapfile
CONF_SWAPSIZE=100
# CONF_SWAPSIZE=1024
^x y

To revert to the smaller swap space, restart the swap service:
$ sudo /etc/init.d/dphys-swapfile stop
$ sudo /etc/init.d/dphys-swapfile start

PiCamera install & test

Follow the link provided in the OpenCV tutorial -
https://www.pyimagesearch.com/2015/03/30/accessing-the-raspberry-pi-camera-with-opencv-and-python/

Test the camera:
(cv)pi@dex:~/cv $ raspistill -o output.jpg
    see the picture
    see the file in File Manager, open to see picture

While the standard picamera module provides methods to interface with the camera, we need the (optional) array sub-module so that we can utilize OpenCV. Remember, when using Python bindings, OpenCV represents images as NumPy arrays — and the array sub-module allows us to obtain NumPy arrays from the Raspberry Pi camera module.

Change suggested pip to pip3

(cv)pi@dex:~/cv $ pip3 install "picamera[array]"
    Done

Note: you can open Idle3 from command line (or menu?) but the code will give an error on importing cv2. If you save the file & run from command line:
(cv)pi@dex:~/cv $ python3 test_image.py
it runs ok but with the warning below.

Open up a new file, name it test_image.py , and insert the following code:

```
# import the necessary packages
from picamera.array import PiRGBArray
from picamera import PiCamera
import time
import cv2

# initialize the camera and grab a reference to the raw camera capture
camera = PiCamera()
rawCapture = PiRGBArray(camera)

# allow the camera to warmup
time.sleep(0.1)

# grab an image from the camera
camera.capture(rawCapture, format="bgr")
image = rawCapture.array

# display the image on screen and wait for a keypress
cv2.imshow("Image", image)
cv2.waitKey(0)
```

(cv)pi@dex:~/cv $ python3 test_image.py
/home/pi/.virtualenvs/cv/lib/python3.5/site-packages/picamera/encoders.py:544:
PiCameraResolutionRounded: frame size rounded up from1920 x 1080 to 1920 x 1088
width, height, fwidth, fheight)))
** (Image:3321): **warning** **: Error retrieving accessibility bus address:
org.freedesktop.DBus.Error.ServiceUnknown:
The name org.a11y.Bus was not provided by any .service files
[...whatever that means...]

We'll start by importing our necessary packages on Lines 2-5.

From there, we initialize our PiCamera object on Line 8 and grab a reference to the raw capture component on Line 9. This rawCapture object is especially useful since it (1) gives us direct access to the camera stream and (2) avoids the expensive compression to JPEG format, which we would then have to take and decode to OpenCV format anyway. I highly recommend that you use ***PiRGBArray*** *whenever you need to access the Raspberry Pi camera — the performance gains are well worth it.*

From there, we sleep for a tenth of a second on Line 12 — this allows the camera sensor to warm up.

Finally, we grab the actual photo from the rawCapture object on Line 15 where we take special care to ensure our image is in ***BGR format*** *rather than RGB. OpenCV represents images as NumPy arrays in BGR order rather than RGB — this little nuisance is subtle, but very important to remember as it can lead to some confusing bugs in your code down the line.*

Step 6: Accessing the video stream of your Raspberry Pi using Python and OpenCV.
You might guess that we are going to use the cv2.VideoCapture function here — but I actually recommend against this. Getting cv2.VideoCapture to play nice with your Raspberry Pi is not a nice experience (you'll need to install extra drivers) and something you should generally avoid.

And besides, why would we use the cv2.VideoCapture function when we can easily access the raw video stream using the picamera module?

test_video.py

```
# import the necessary packages
from picamera.array import PiRGBArray
from picamera import PiCamera
import time
import cv2

# initialize the camera and grab a reference to the raw camera capture
camera = PiCamera()
camera.resolution = (640, 480)
camera.framerate = 32
rawCapture = PiRGBArray(camera, size=(640, 480))

# allow the camera to warmup
time.sleep(0.1)

# capture frames from the camera
```

```
for frame in camera.capture_continuous(rawCapture, format="bgr", use_video_port=True):
	# grab the raw NumPy array representing the image, then initialize the timestamp
	# and occupied/unoccupied text
	image = frame.array

	# show the frame
	cv2.imshow("Frame", image)
	key = cv2.waitKey(1) & 0xFF

	# clear the stream in preparation for the next frame
	rawCapture.truncate(0)

	# if the `q` key was pressed, break from the loop
	if key == ord("q"):
		break
```

This example starts off similarly to the previous one. We start off by importing our necessary packages on Lines 2-5.

And from there we construct our camera object on Line 8 which allows us to interface with the Raspberry Pi camera. However, we also take the time to set the resolution of our camera (640 x 480 pixels) on Line 9 and the frame rate (i.e. frames per second, or simply FPS) on Line 10. We also initialize our PiRGBArray object on Line 11, but we also take care to specify the same resolution as on Line 9.

Accessing the actual video stream is handled on Line 17 by making a call to the capture_continuous method of our camera object.

This method returns a frame from the video stream. The frame then has an array property, which corresponds to the frame in NumPy array format — all the hard work is done for us on Lines 17 and 20!

We then take the frame of the video and display on screen on Lines 23 and 24.

An important line to pay attention to is Line 27: You must clear the current frame before you move on to the next one!

If you fail to clear the frame, your Python script will throw an error — so be sure to pay close attention to this when implementing your own applications!

Finally, if the user presses the q key, we break form the loop and exit the program.

To execute our script, just open a terminal (making sure you are in the cv virtual environment, of course) and issue the following command:

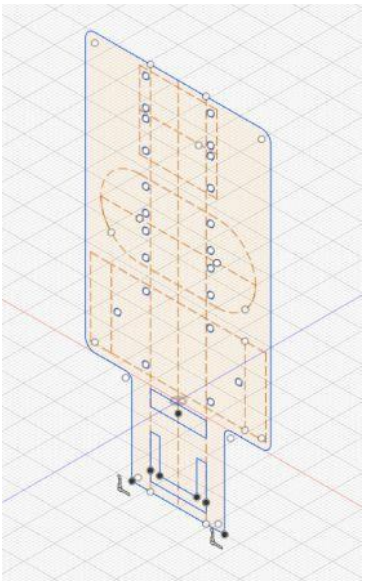
```
(cv)pi@dex:~/cv $ python test_video.py
q to quit
```

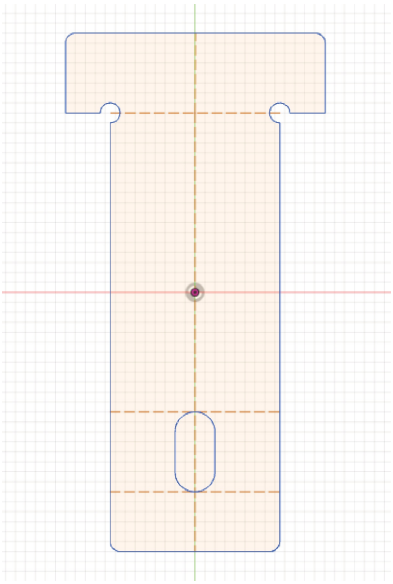
It works.

His free “Image Search Engine Resource Guide” is just an ad for his books plus some links. Skip.

Custom Laser Cut Sensor Mount

Inspired by original Dexter GoPiGo servo kit. Created to mount Pi Camera, IR sensor, and US sensor on one servo swivel mount. Sensor mounting holes duplicated and mirrored to allow mounting upside down so wires could go through slot on bottom or over the top. Side clamp bars widened to fit extended servo slot and made longer at the top for better strength. Created in Fusion 360 then transferred at the HCC Stafford fab lab to CorelDraw, formatted to raster etch the sensor outlines and cut the holes, slots and outline. Then sent to laser cutter.





Problems with sensor code

Since upgrading Raspbian-For-Robots-Operating-System to stretch, I'm getting python errors in programs that previously worked:

Example #1:

home/pi/Dexter/PHELAN/Dual_Sensor_IR-US_PHELAN.py

```
# import the GoPiGo3 drivers
import time
import easygopigo3 as easy
# Create an instance of the GoPiGo3 class.
# GPG will be the GoPiGo3 object.
gpg = easy.EasyGoPiGo3()

# Create an instance of the Distance Sensor class.
# I2C1 and I2C2 are just labels used for identifyng the port on the GoPiGo3 board.
# But technically, I2C1 and I2C2 are the same thing, so we don't have to pass any port to
the constructor.
IR_distance_sensor = gpg.init_distance_sensor()

# import the GrovePi drivers
import grovepi

# Connect the Grove Ultrasonic Ranger to digital port D4
ultrasonic_ranger = 4
# US sensor output is is cm, use this to convert to inches
inch = 2.54 #cm

while True:
      try:
            # Read distance value from Ultrasonic
            print("Reading US sensor")
            US_dist_cm = grovepi.ultrasonicRead(ultrasonic_ranger)     ## Fails here
            #print(grovepi.ultrasonicRead(ultrasonic_ranger))
            ##US_dist_cm = 666
            US_dist_in = US_dist_cm / inch
            print("US %8.2f cm  %8.2f in" % (US_dist_cm, US_dist_in))
      except TypeError as t: print (t)
      except Exception as e: print(e)
      except IOError as i: print (i)

      try:
            IR_dist_cm = IR_distance_sensor.read_mm()
            #convert mm to cm
            IR_dist_cm = IR_dist_cm / 10.0
            IR_dist_in = IR_dist_cm / inch
            # Directly print the values of the sensor.
            ## print("Distance Sensor Reading (mm): " + str(my_distance_sensor.read_mm()))
            print("IR %6.2f cm %6.2f in" % (IR_dist_cm, IR_dist_in))
      except TypeError:
            print ("IR Error")
      except IOError:
            print ("IR Error")

      print("IRcm %6.2f UScm %6.2f IRin %6.2f USin %6.2f" % (IR_dist_cm, US_dist_cm,
IR_dist_in, US_dist_in))


======== RESTART: /home/pi/Desktop/PHELAN/Dual_Sensor_IR-US_PHELAN.py ========
```

```
Reading US sensor
'int' object is not subscriptable
IR  69.80 cm  27.48 in
Traceback (most recent call last):
  File "/home/pi/Desktop/PHELAN/Dual_Sensor_IR-US_PHELAN.py", line 79, in <module>
    print("IRcm %6.2f UScm %6.2f IRin %6.2f USin %6.2f" % (IR_dist_cm, US_dist_cm,
IR_dist_in, US_dist_in))
NameError: name 'US_dist_cm' is not defined
```

Example #2 This one works using AD1 on GoPiGo so demonstrates the sensor is working:

```
#!/usr/bin/env python
# /home/pi/Desktop/GoPiGo3/Software/Python/Examples/Grove_US.py
# https://www.dexterindustries.com/GoPiGo/
# https://github.com/DexterInd/GoPiGo3
#
# Copyright (c) 2017 Dexter Industries
# Released under the MIT license (http://choosealicense.com/licenses/mit/).
# For more information see https://github.com/DexterInd/GoPiGo3/blob/master/LICENSE.md
#
# This code is an example for using the grove ultrasonic sensor with the GoPiGo3.
#
# Hardware: Connect a grove ultrasonic sensor to port AD1 of the GoPiGo3.
#
# Results: When you run this program, the distance measured with the ultrasonic sensor
should be printed.

from __future__ import print_function # use python 3 syntax but make it compatible with
python 2
from __future__ import division       #                           ''

import time     # import the time library for the sleep function
import gopigo3 # import the GoPiGo3 drivers

GPG = gopigo3.GoPiGo3() # Create an instance of the GoPiGo3 class. GPG will be the GoPiGo3
object.

try:
    GPG.set_grove_type(GPG.GROVE_1, GPG.GROVE_TYPE.US)

    while(True):
        try:
            print("%4dmm" % GPG.get_grove_value(GPG.GROVE_1))
        except gopigo3.SensorError as error:
            print(error)
        except gopigo3.ValueError as error:
            print(error)
        time.sleep(0.05)

except KeyboardInterrupt: # except the program gets interrupted by Ctrl+C on the keyboard.
    GPG.reset_all()        # Unconfigure the sensors, disable the motors, and restore the
LED to the control of the GoPiGo3 firmware.
```

```
Python 3.5.3 (default, Jan 19 2017, 14:11:04)
[GCC 6.3.0 20170124] on linux
Type "copyright", "credits" or "license()" for more information.
>>>
=== RESTART: /home/pi/Desktop/GoPiGo3/Software/Python/Examples/Grove_US.py ===
get_grove_value error: Invalid value
 855mm
```

```
 860mm
 860mm
 860mm
get_grove_value error: Object not detected within range
get_grove_value error: Object not detected within range
get_grove_value error: Object not detected within range
 860mm
 864mm
 855mm
 860mm
 850mm
 579mm
^C interrupt
```

Example 3: Grove Pi example, gives TypeError:

```
#!/usr/bin/env python
# /home/pi/Desktop/GrovePi/Software/Python/grove_ultrasonic.py
# GrovePi Example for using the Grove Ultrasonic Ranger
(http://www.seeedstudio.com/wiki/Grove_-_Ultrasonic_Ranger)
#
# The GrovePi connects the Raspberry Pi and Grove sensors.  You can learn more about
GrovePi here:  http://www.dexterindustries.com/GrovePi
#
# Have a question about this example?  Ask on the forums here:
http://forum.dexterindustries.com/c/grovepi
#

import grovepi

# Connect the Grove Ultrasonic Ranger to digital port D4
# SIG,NC,VCC,GND
ultrasonic_ranger = 4

while True:
    try:
        # Read distance value from Ultrasonic
        US_dist = grovepi.ultrasonicRead(ultrasonic_ranger)
        # print(grovepi.ultrasonicRead(ultrasonic_ranger))
        print(US_dist)
    except TypeError:
        print ("TypeError")
    except IOError:
        print ("IOError")
```

```
Python 3.5.3 (default, Jan 19 2017, 14:11:04)
[GCC 6.3.0 20170124] on linux
Type "copyright", "credits" or "license()" for more information.
>>>
=== RESTART: /home/pi/Desktop/GrovePi/Software/Python/grove_ultrasonic.py ===
TypeError
...
TypeError
Traceback (most recent call last):
  File "/home/pi/Desktop/GrovePi/Software/Python/grove_ultrasonic.py", line 45, in
<module>
    US_dist = grovepi.ultrasonicRead(ultrasonic_ranger)
  File "/home/pi/Desktop/GrovePi/Software/Python/grovepi.py", line 256, in ultrasonicRead
    number = read_i2c_block(address)
  File "/home/pi/Desktop/GrovePi/Software/Python/grovepi.py", line 194, in read_i2c_block
    return bus.read_i2c_block_data(address, 1)
```

```
  File "/usr/local/lib/python3.5/dist-packages/smbus_cffi-0.5.1-py3.5-linux-
armv7l.egg/smbus/util.py", line 59, in validator
    return fn(*args, **kwdefaults)
  File "/usr/local/lib/python3.5/dist-packages/smbus_cffi-0.5.1-py3.5-linux-
armv7l.egg/smbus/smbus.py", line 257, in read_i2c_block_data
    arg, data):
KeyboardInterrupt
>>>
```

Searching “'int' object is not subscriptable” on Dexter forum suggested firmware update fixed it.
But that was last year before stretch. Just updated Dexter software & firmware for GoPiGo3 and GrovePi. We shall see….

Reboot

GoPiGo3 test log:

```
GoPiGo3 Troubleshooting Script log

Checking for hardware, and checking hardware and firmware version.
=================================================================
Manufacturer    :  Dexter Industries
Board           :  GoPiGo3
Serial Number   :  8B254010514E343732202020FF110B27
Hardware version:  3.x.x
Firmware version:  1.0.0
Battery voltage :  11.71
5v voltage      :  4.997
```

GrovePi test originally run w/o motors connected w/ errors. Didn’t think motors would affect GrovePi, but reconneded anyway & re-ran. Still errors:

GrovePi test log:

```
Check space left
================
Filesystem      Size  Used Avail Use% Mounted on
/dev/root        30G  9.7G   19G  35% /
devtmpfs        370M     0  370M   0% /dev
tmpfs           375M     0  375M   0% /dev/shm
tmpfs           375M   11M  365M   3% /run
tmpfs           5.0M  4.0K  5.0M   1% /run/lock
tmpfs           375M     0  375M   0% /sys/fs/cgroup
/dev/mmcblk0p1   42M   22M   20M  53% /boot
tmpfs            75M     0   75M   0% /run/user/1000

Check for dependencies
======================
python 2.7.13-2 install ok installed
python-pip 9.0.1-2+rpt2 install ok installed
git 1:2.11.0-3+deb9u3 install ok installed
libi2c-dev 3.1.2-3 install ok installed
python-serial 3.2.1-1 install ok installed
python-rpi.gpio 0.6.3~stretch-1 install ok installed
i2c-tools 3.1.2-3 install ok installed
python-smbus 3.1.2-3 install ok installed
arduino 2:1.0.5+dfsg2-4.1 install ok installed
minicom 2.7-1.1 install ok installed
scratch 1.4.0.6~dfsg1-5 install ok installed
```

```
find: ‘/run/user/1000/gvfs’: Permission denied
wiringPi Found
find: ‘/run/user/1000/gvfs’: Permission denied
wiringPi Found
I2C already removed from blacklist
SPI already removed from blacklist


Check for addition in /modules
==============================
I2C-dev already there
i2c-bcm2708 already there
spi-dev already there


Hardware revision
=================
gpio version: 2.36
Copyright (c) 2012-2015 Gordon Henderson
This is free software with ABSOLUTELY NO WARRANTY.
For details type: gpio -warranty

Raspberry Pi Details:
  Type: Pi 3, Revision: 02, Memory: 1024MB, Maker: Sony
  * Device tree is enabled.
  *--> Raspberry Pi 3 Model B Rev 1.2
  * This Raspberry Pi supports user-level GPIO access.


Check the /dev folder
=====================
i2c-1
spidev0.0
spidev0.1
ttyAMA0

USB device status
=================
Bus 001 Device 005: ID 413c:3012 Dell Computer Corp. Optical Wheel Mouse
Bus 001 Device 004: ID 413c:2105 Dell Computer Corp. Model L100 Keyboard
Bus 001 Device 003: ID 0424:ec00 Standard Microsystems Corp. SMSC9512/9514 Fast Ethernet
Adapter
Bus 001 Device 002: ID 0424:9514 Standard Microsystems Corp. SMC9514 Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub

/:  Bus 01.Port 1: Dev 1, Class=root_hub, Driver=dwc_otg/1p, 480M
    |__ Port 1: Dev 2, If 0, Class=Hub, Driver=hub/5p, 480M
        |__ Port 1: Dev 3, If 0, Class=Vendor Specific Class, Driver=smsc95xx, 480M
        |__ Port 3: Dev 4, If 0, Class=Human Interface Device, Driver=usbhid, 1.5M
        |__ Port 5: Dev 5, If 0, Class=Human Interface Device, Driver=usbhid, 1.5M
Raspbian for Robots Version
===========================
V 9

This version of Raspbian was modified by Dexter Industries on the Jessie Raspbian Build.
This version was updated on 26th of June 2018.


Hostname
========
dex
```

```
Checking for Atmega chip
========================


avrdude: Version 5.10, compiled on Jun 18 2012 at 12:38:29
         Copyright (c) 2000-2005 Brian Dean, http://www.bdmicro.com/
         Copyright (c) 2007-2009 Joerg Wunsch

         System wide configuration file is "/etc/avrdude.conf"
         User configuration file is "/root/.avrduderc"
         User configuration file does not exist or is not a regular file, skipping

         Using Port                    : unknown
         Using Programmer              : gpio
         AVR Part                      : ATMEGA328P
         Chip Erase delay              : 9000 us
         PAGEL                         : PD7
         BS2                           : PC2
         RESET disposition             : dedicated
         RETRY pulse                   : SCK
         serial program mode           : yes
         parallel program mode         : yes
         Timeout                       : 200
         StabDelay                     : 100
         CmdexeDelay                   : 25
         SyncLoops                     : 32
         ByteDelay                     : 0
         PollIndex                     : 3
         PollValue                     : 0x53
         Memory Detail                 :

                                  Block Poll               Page
Polled
           Memory Type Mode Delay Size  Indx Paged  Size   Size #Pages MinW  MaxW
ReadBack
           ----------- ---- ----- ----- ---- ------ ------ ---- ------ ----- ----- -------
--
           eeprom        65     5     4    0 no       1024    4      0  3600  3600 0xff
0xff
           flash         65     6   128    0 yes     32768  128    256  4500  4500 0xff
0xff
           lfuse          0     0     0    0 no          1    0      0  4500  4500 0x00
0x00
           hfuse          0     0     0    0 no          1    0      0  4500  4500 0x00
0x00
           efuse          0     0     0    0 no          1    0      0  4500  4500 0x00
0x00
           lock           0     0     0    0 no          1    0      0  4500  4500 0x00
0x00
           calibration    0     0     0    0 no          1    0      0     0     0 0x00
0x00
           signature      0     0     0    0 no          3    0      0     0     0 0x00
0x00

         Programmer Type : GPIO
         Description     : Use sysfs interface to bitbang GPIO lines

avrdude: AVR device not responding
avrdude: initialization failed, rc=-1
```

```
         Double check connections and try again, or use -F to override
         this check.


avrdude done.  Thank you.


Checking I2C bus for devices
============================

Checking I2C bus 0
==================
Error: Could not open file `/dev/i2c-0' or `/dev/i2c/0': No such file or directory

Checking I2C bus 1
==================
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          03 04 -- -- -- 08 -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- 29 -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- 3e --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- 62 -- -- -- -- -- -- -- -- -- -- -- -- --
70: 70 -- -- -- -- -- -- --

Checking for firmware version
=============================
Traceback (most recent call last):
  File "/home/pi/Dexter/GrovePi/Software/Python/grove_firmware_version_check.py", line 40,
in <module>
    print("GrovePi has firmware version: %s" %grovepi.version())
  File "/home/pi/Dexter/GrovePi/Software/Python/grovepi.py", line 266, in version
    return "%s.%s.%s" % (number[1], number[2], number[3])
TypeError: 'int' object has no attribute '__getitem__'
```

Test report still states based on Raspbian **jessie**, even though OS states it’s **stretch**:

```
pi@dex:~ $ cat /etc/os-release
PRETTY_NAME="Raspbian GNU/Linux 9 (stretch)"
NAME="Raspbian GNU/Linux"
VERSION_ID="9"
VERSION="9 (stretch)"
ID=raspbian
ID_LIKE=debian
HOME_URL="http://www.raspbian.org/"
SUPPORT_URL="http://www.raspbian.org/RaspbianForums"
BUG_REPORT_URL="http://www.raspbian.org/RaspbianBugs"
```

So,again searching “'int' object is not subscriptable” on Dexter forum suggested firmware update fixed it with more detail re how to do it, not just the desktop app:
https://forum.dexterindustries.com/t/this-makes-no-sense-ultrasonic-ranger/4662/2

Don’t know if that is any different from the desktop update app.
Doesn't seem to have fixed the final error on the update log.

Since I've updated, I'll have to reclaim ownership of Dexter files. Done.

Still getting same errors. The command line firmware update no different than desktop app so far as I can tell.

NOTES FOR NEXT TIME:
Maybe it's the GrovePi or that port.
Try a different port & change the code accordingly.

Changing to D3 made no difference.

Searching Dexter Forum again for: 'int' object has no attribute '__getitem__'

Third choice led me to:
https://forum.dexterindustries.com/t/solved-grovepi-firmware-update-fails-avr-device-not-responding/3144

Soldered a 6 pin header to the GrovePi.
Jumpered according to the directions.

```
$ cd ~/Desktop/GrovePi/Firmware
~/Desktop/GrovePi/Firmware $ sudo bash firmware_update.sh
      no such file
The actual file is now –
~/Desktop/GrovePi/Firmware $ sudo bash grovepi_firmware_update.sh
~/Desktop/GrovePi/Firmware $
      no output
```

Then I tried the desktop DI Software Update app to update the GrovePi firmware
and got a through & apparently successful log w/o errors:

```
Script started on Thu 26 Jul 2018 06:44:04 PM CDT
access control disabled, clients can connect from any host
None
write_state: dex
read_state: dex
default drop down
read_state: dex
dex.png
GrovePi
write_state: GrovePi
read_state: GrovePi
read_state: GrovePi
Start Firmware test!GrovePi

avrdude: AVR device initialized and ready to accept instructions


Reading |                                                    | 0% 0.00s
Reading | ##################                                 | 33% 0.00s
Reading | ##################################                 | 66% 0.00s
Reading | ################################################## | 100% 0.00s

avrdude: Device signature = 0x1e950f
avrdude: reading input file "0xFF"
avrdude: writing lfuse (1 bytes):


Writing |                                                    | 0% 0.00s
Writing | ################################################## | 100% 0.00s

avrdude: 1 bytes of lfuse written
avrdude: verifying lfuse memory against 0xFF:
avrdude: load data lfuse data from input file 0xFF:
```

```
avrdude: input file 0xFF contains 1 bytes
avrdude: reading on-chip lfuse data:


Reading |                                                    | 0% 0.00s
Reading | ################################################## | 100% 0.00s

avrdude: verifying ...
avrdude: 1 bytes of lfuse verified

avrdude: safemode: Fuses OK

avrdude done.  Thank you.


avrdude: AVR device initialized and ready to accept instructions


Reading |                                                    | 0% 0.00s
Reading | ##################                                 | 33% 0.00s
Reading | ##################################                 | 66% 0.00s
Reading | ################################################## | 100% 0.00s

avrdude: Device signature = 0x1e950f
avrdude: reading input file "0xDA"
avrdude: writing hfuse (1 bytes):


Writing |                                                    | 0% 0.00s
Writing | ################################################## | 100% 0.00s

avrdude: 1 bytes of hfuse written
avrdude: verifying hfuse memory against 0xDA:
avrdude: load data hfuse data from input file 0xDA:
avrdude: input file 0xDA contains 1 bytes
avrdude: reading on-chip hfuse data:


Reading |                                                    | 0% 0.00s
Reading | ################################################## | 100% 0.00s

avrdude: verifying ...
avrdude: 1 bytes of hfuse verified

avrdude: safemode: Fuses OK

avrdude done.  Thank you.


avrdude: AVR device initialized and ready to accept instructions


Reading |                                                    | 0% 0.00s
Reading | ##################                                 | 33% 0.00s
Reading | ##################################                 | 66% 0.00s
Reading | ################################################## | 100% 0.00s

avrdude: Device signature = 0x1e950f
avrdude: reading input file "0x05"
avrdude: writing efuse (1 bytes):
```

```
Writing |                                                            | 0% 0.00s
Writing | ########################################################## | 100% 0.00s

avrdude: 1 bytes of efuse written
avrdude: verifying efuse memory against 0x05:
avrdude: load data efuse data from input file 0x05:
avrdude: input file 0x05 contains 1 bytes
avrdude: reading on-chip efuse data:


Reading |                                                            | 0% 0.00s
Reading | ########################################################## | 100% 0.00s

avrdude: verifying ...
avrdude: 1 bytes of efuse verified

avrdude: safemode: Fuses OK

avrdude done.  Thank you.


avrdude: AVR device initialized and ready to accept instructions


Reading |                                                            | 0% 0.00s
Reading | ##################                                         | 33% 0.00s
Reading | ##################################                         | 66% 0.00s
Reading | ########################################################## | 100% 0.00s

avrdude: Device signature = 0x1e950f
avrdude: NOTE: FLASH memory has been specified, an erase cycle will be performed
         To disable this feature, specify the -D option.
avrdude: erasing chip
avrdude: reading input file "grove_pi_firmware.hex"
avrdude: input file grove_pi_firmware.hex auto detected as Intel Hex
avrdude: writing flash (13816 bytes):


Writing |                                                            | 0% 0.00s
Writing | #                                                          | 1% 0.04s
Writing | #                                                          | 2% 0.08s
Writing | ##                                                         | 3% 0.12s
Writing | ##                                                         | 4% 0.16s
Writing | ###                                                        | 5% 0.21s
Writing | ###                                                        | 6% 0.25s
Writing | ####                                                       | 7% 0.29s
Writing | ####                                                       | 8% 0.33s
Writing | #####                                                      | 9% 0.37s
Writing | #####                                                      | 10% 0.41s
Writing | ######                                                     | 11% 0.45s
Writing | ######                                                     | 12% 0.49s
Writing | #######                                                    | 13% 0.53s
Writing | #######                                                    | 14% 0.57s
Writing | ########                                                   | 15% 0.61s
Writing | ########                                                   | 16% 0.65s
Writing | #########                                                  | 17% 0.69s
Writing | #########                                                  | 18% 0.73s
Writing | ##########                                                 | 19% 0.77s
Writing | ##########                                                 | 20% 0.82s
```

```
Writing | ###########                                                | 21% 0.86s
Writing | ###########                                                | 22% 0.90s
Writing | ############                                               | 23% 0.94s
Writing | ############                                               | 24% 0.98s
Writing | #############                                              | 25% 1.02s
Writing | #############                                              | 26% 1.06s
Writing | ##############                                             | 27% 1.10s
Writing | ##############                                             | 28% 1.14s
Writing | ###############                                            | 29% 1.18s
Writing | ###############                                            | 30% 1.22s
Writing | ################                                           | 31% 1.26s
Writing | ################                                           | 32% 1.30s
Writing | #################                                          | 33% 1.34s
Writing | #################                                          | 34% 1.38s
Writing | ##################                                         | 35% 1.42s
Writing | ##################                                         | 36% 1.46s
Writing | ###################                                        | 37% 1.50s
Writing | ###################                                        | 38% 1.55s
Writing | ####################                                       | 39% 1.59s
Writing | ####################                                       | 40% 1.63s
Writing | #####################                                      | 41% 1.67s
Writing | #####################                                      | 42% 1.71s
Writing | ######################                                     | 43% 1.75s
Writing | ######################                                     | 44% 1.79s
Writing | #######################                                    | 45% 1.83s
Writing | #######################                                    | 46% 1.87s
Writing | ########################                                   | 47% 1.91s
Writing | ########################                                   | 48% 1.95s
Writing | #########################                                  | 49% 1.99s
Writing | #########################                                  | 50% 2.03s
Writing | ##########################                                 | 51% 2.07s
Writing | ##########################                                 | 52% 2.11s
Writing | ###########################                                | 53% 2.15s
Writing | ###########################                                | 54% 2.19s
Writing | ############################                               | 55% 2.23s
Writing | ############################                               | 56% 2.27s
Writing | #############################                              | 57% 2.31s
Writing | #############################                              | 58% 2.35s
Writing | ##############################                             | 59% 2.39s
Writing | ##############################                             | 60% 2.43s
Writing | ###############################                            | 61% 2.47s
Writing | ###############################                            | 62% 2.51s
Writing | ################################                           | 63% 2.56s
Writing | ################################                           | 64% 2.60s
Writing | #################################                          | 65% 2.63s
Writing | #################################                          | 66% 2.67s
Writing | ##################################                         | 67% 2.71s
Writing | ##################################                         | 68% 2.76s
Writing | ###################################                        | 69% 2.80s
Writing | ###################################                        | 70% 2.83s
Writing | ####################################                       | 71% 2.87s
Writing | ####################################                       | 72% 2.91s
Writing | #####################################                      | 73% 2.95s
Writing | #####################################                      | 74% 2.99s
Writing | ######################################                     | 75% 3.03s
Writing | ######################################                     | 76% 3.08s
Writing | #######################################                    | 77% 3.12s
Writing | #######################################                    | 78% 3.16s
Writing | ########################################                   | 79% 3.20s
Writing | ########################################                   | 80% 3.24s
```

```
Writing | ##########################################          | 81% 3.28s
Writing | ##########################################          | 82% 3.32s
Writing | ###########################################         | 83% 3.36s
Writing | ###########################################         | 84% 3.40s
Writing | ############################################        | 85% 3.44s
Writing | ############################################        | 86% 3.48s
Writing | #############################################       | 87% 3.52s
Writing | #############################################       | 88% 3.56s
Writing | ##############################################      | 89% 3.60s
Writing | ##############################################      | 90% 3.64s
Writing | ###############################################     | 91% 3.68s
Writing | ###############################################     | 92% 3.72s
Writing | ################################################    | 93% 3.76s
Writing | ################################################    | 94% 3.80s
Writing | #################################################   | 95% 3.84s
Writing | #################################################   | 96% 3.88s
Writing | ##################################################  | 97% 3.92s
Writing | ##################################################  | 98% 3.96s
Writing | ################################################## | 99% 4.00s
Writing | ################################################## | 100% 4.04s

avrdude: 13816 bytes of flash written
avrdude: verifying flash memory against grove_pi_firmware.hex:
avrdude: load data flash data from input file grove_pi_firmware.hex:
avrdude: input file grove_pi_firmware.hex auto detected as Intel Hex
avrdude: input file grove_pi_firmware.hex contains 13816 bytes
avrdude: reading on-chip flash data:


Reading |                                                    | 0% 0.00s
Reading | #                                                  | 1% 0.04s
Reading | #                                                  | 2% 0.08s
Reading | ##                                                 | 3% 0.11s
Reading | ##                                                 | 4% 0.15s
Reading | ###                                                | 5% 0.18s
Reading | ###                                                | 6% 0.22s
Reading | ####                                               | 7% 0.25s
Reading | ####                                               | 8% 0.29s
Reading | #####                                              | 9% 0.32s
Reading | #####                                              | 10% 0.36s
Reading | ######                                             | 11% 0.39s
Reading | ######                                             | 12% 0.43s
Reading | #######                                            | 13% 0.46s
Reading | #######                                            | 14% 0.50s
Reading | ########                                           | 15% 0.53s
Reading | ########                                           | 16% 0.57s
Reading | #########                                          | 17% 0.60s
Reading | #########                                          | 18% 0.64s
Reading | ##########                                         | 19% 0.67s
Reading | ##########                                         | 20% 0.71s
Reading | ###########                                        | 21% 0.74s
Reading | ###########                                        | 22% 0.78s
Reading | ############                                       | 23% 0.81s
Reading | ############                                       | 24% 0.85s
Reading | #############                                      | 25% 0.88s
Reading | #############                                      | 26% 0.92s
Reading | ##############                                     | 27% 0.95s
Reading | ##############                                     | 28% 0.99s
Reading | ###############                                    | 29% 1.02s
Reading | ###############                                    | 30% 1.06s
```

```
Reading | ################                                    | 31% 1.09s
Reading | ################                                    | 32% 1.13s
Reading | #################                                   | 33% 1.16s
Reading | #################                                   | 34% 1.20s
Reading | ##################                                  | 35% 1.23s
Reading | ##################                                  | 36% 1.27s
Reading | ###################                                 | 37% 1.30s
Reading | ###################                                 | 38% 1.34s
Reading | ####################                                | 39% 1.37s
Reading | ####################                                | 40% 1.41s
Reading | #####################                               | 41% 1.44s
Reading | #####################                               | 42% 1.48s
Reading | ######################                              | 43% 1.51s
Reading | ######################                              | 44% 1.55s
Reading | #######################                             | 45% 1.58s
Reading | #######################                             | 46% 1.61s
Reading | ########################                            | 47% 1.65s
Reading | ########################                            | 48% 1.69s
Reading | #########################                           | 49% 1.72s
Reading | #########################                           | 50% 1.76s
Reading | ##########################                          | 51% 1.79s
Reading | ##########################                          | 52% 1.83s
Reading | ###########################                         | 53% 1.86s
Reading | ###########################                         | 54% 1.90s
Reading | ############################                        | 55% 1.93s
Reading | ############################                        | 56% 1.97s
Reading | #############################                       | 57% 2.00s
Reading | #############################                       | 58% 2.04s
Reading | ##############################                      | 59% 2.07s
Reading | ##############################                      | 60% 2.11s
Reading | ###############################                     | 61% 2.14s
Reading | ###############################                     | 62% 2.18s
Reading | ################################                    | 63% 2.21s
Reading | ################################                    | 64% 2.25s
Reading | #################################                   | 65% 2.28s
Reading | #################################                   | 66% 2.32s
Reading | ##################################                  | 67% 2.35s
Reading | ##################################                  | 68% 2.39s
Reading | ###################################                 | 69% 2.43s
Reading | ###################################                 | 70% 2.46s
Reading | ####################################                | 71% 2.50s
Reading | ####################################                | 72% 2.53s
Reading | #####################################               | 73% 2.57s
Reading | #####################################               | 74% 2.61s
Reading | ######################################              | 75% 2.64s
Reading | ######################################              | 76% 2.68s
Reading | #######################################             | 77% 2.71s
Reading | #######################################             | 78% 2.75s
Reading | ########################################            | 79% 2.78s
Reading | ########################################            | 80% 2.82s
Reading | #########################################           | 81% 2.85s
Reading | #########################################           | 82% 2.89s
Reading | ##########################################          | 83% 2.93s
Reading | ##########################################          | 84% 2.96s
Reading | ###########################################         | 85% 3.00s
Reading | ###########################################         | 86% 3.03s
Reading | ############################################        | 87% 3.07s
Reading | ############################################        | 88% 3.10s
Reading | #############################################       | 89% 3.14s
Reading | #############################################       | 90% 3.18s
```

```
Reading | ##############################################     | 91% 3.21s
Reading | ##############################################     | 92% 3.25s
Reading | ##############################################     | 93% 3.28s
Reading | ##############################################     | 94% 3.32s
Reading | ##############################################     | 95% 3.35s
Reading | ##############################################     | 96% 3.39s
Reading | ##############################################     | 97% 3.42s
Reading | ##############################################     | 98% 3.46s
Reading | ##############################################     | 99% 3.49s
Reading | ##############################################     | 100% 3.53s

avrdude: verifying ...
avrdude: 13816 bytes of flash verified

avrdude: safemode: Fuses OK

avrdude done.  Thank you.

Beginning to Update the GrovePi Firmware!
============================
/home/pi/Dexter/GrovePi/Firmware /home/pi/di_update/Raspbian_For_Robots
/home/pi/di_update/Raspbian_For_Robots
Finished updating the GrovePi Firmware!
============================
```

OK, so stacked the GrovePi on the Raspberry Pi *without* the GoPiGo3.
~/Desktop/GrovePi/Software/Python/grove_ultrasonic.py on D4 worked ok!

Tried .../PHELAN/DualSensor_IR-US_PHELAN.py
but the GoPiGo3 wasn't attached.
So, shut-down & put the GoPiGo3 board back in the stack,
& power up this time through the GoPiGo3 board.

$.../PHELAN/DualSensor_IR-US_PHELAN.py
YES!!! Output!

Now to put the stack back on the GoPiGo chassis. Done.

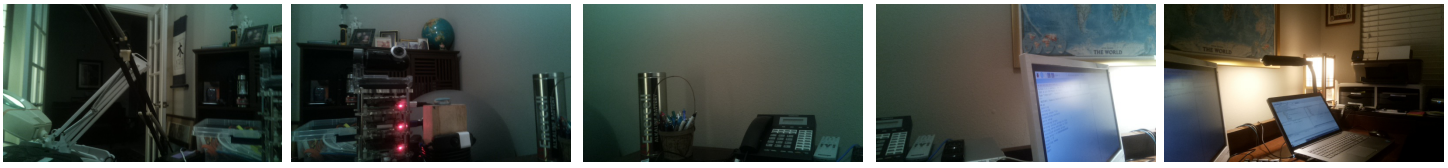
With the stack reassembled, was able to run
...PHELAN/Dual_Sensor_Calibration_PHELAN.py properly!

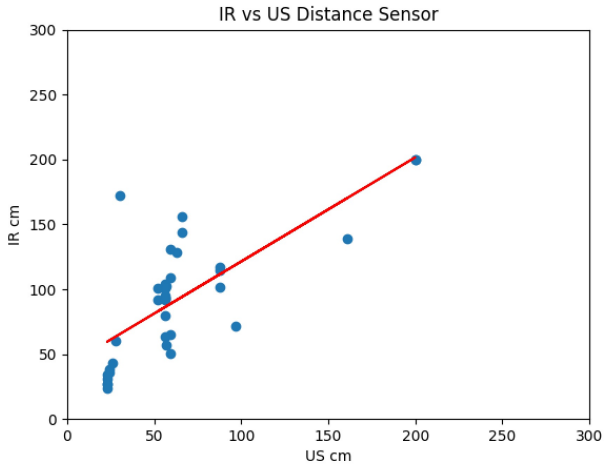
GoPiGo3 additions & improvements - see code to follow:

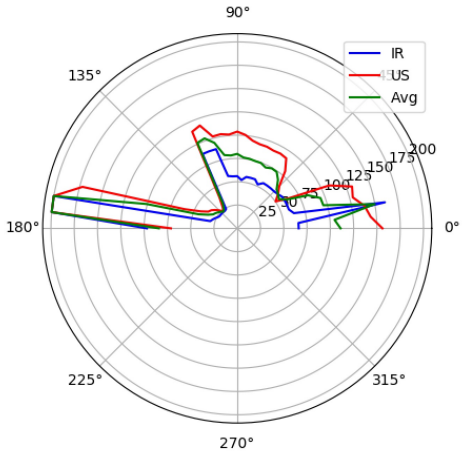
Safety module: Added beeper warning & flashing lights - red "blinkers" & yellow flashing "eyes" on the GoPiGo3 board.

Photo of scan points: Added snap photo at 0, 45, 90, 135 & 180 degrees around the sweep and save to file to show terrain to compare to polar plot. Shown below in reverse order as they're taken from right to left.

180o 135o 90o 45o 0o







Plot to file: added saving plots to file to compare to pix. Added code to allow plots to appear together without having to close first one to see the second.

```
## /home/pi/Dexter/PHELAN/Dual_Sensor_Servo_Polar_Plot_Safety_Photo_PHELAN.py
# Calibrate servo for /home/pi/Dexter/GoPiGo3/Projects/IntelligentObjectAvoider/robot.py
# James H Phelan 2018.05.20 - 2018.06.14+
# James H Phelan 2018.07.28 average US + IR into 3d polar plot
# Takes both IR & US readings at a given inverval across up to 180o sweep
# in front of the robot
# then it plots them on a polar plot as in a radar/sonar display
# eventually will select the "best" path among the readings to move fwd
# part of the measured "best" distance, then scan again
# eventually maybe scan while it's moving,
# suspect will require threads, that may be my next diversion

# IMPORT
print("Importing")
import sys  # so you can sys.exit(0) on KeyboardInterrupt
from easygopigo3 import EasyGoPiGo3     # get GPG utilities
import grovepi  # for US sensor Can do on GoPiGo3 w/ other code
from time import sleep  # to wait servo
from matplotlib import pyplot as plt    # for plots
import numpy as np      # for plots
import picamera     # for camera
from datetime import datetime   # for time stamp

# INITIATE
print("Initiating")
# try to connect to the GoPiGo3
try:
    GPG3 = EasyGoPiGo3()
    #print("GoPiGo3 robot initiated")    #DEBUG#
except IOError:
    print("GoPiGo3 robot not detected")

# initialize a Servo object
servo = GPG3.init_servo("SERVO1")
#print("SERVO1 initiated")    #DEBUG#
```

```
# instantiate beeper
try:
    beeper = GPG3.init_buzzer("AD2")
except:
    print("No beeper")

# Create an instance of the IR Distance Sensor class.
# I2C1 and I2C2 are just labels used for identifyng the port on the GoPiGo3 board.
# But technically, I2C1 and I2C2 are the same thing, so we don't have to pass any port to
the constructor.
IR_distance_sensor = GPG3.init_distance_sensor()

# Create an instance of the US Distance Sensor class.
# Connect the Grove Ultrasonic Ranger to digital port D4
ultrasonic_ranger = 4
# US sensor output is is cm, use this to convert to inches
# inch = 2.54 #cm ## not used
# cmFromWall = 0  # starting distance from wall

print("Opening output file")
# Open/create an output file for the distance readings
# "w" = write
# "+" = create if doesn't exist
try:
        f=open("sensor_output.csv","w+")
except:
        print("Can't open sensor_output.csv")
f.write("IR vs US Distance Sensors in Servo Sweep\n")

# ESTABLISH CONSTANTS
print("Establishing constants")
min_degrees = 0
center_degrees = 90
max_degrees = 180
#try a narrower sample angle than 10o
step_degrees = 5
# change steps (number of divisions between MIN & MAX
# to steps + 1 (total number of reading angles to include end points)
# to avoid having to say "steps+1" in future loops
steps = int(((max_degrees - min_degrees) / step_degrees ) + 1)
# give servo mech & sensor support time to settle
servo_delay = 0.01
wait_before_it_starts = 1
# for GoPiGo3 binkers
left = 0
right = 1
# for GoPiGo3 "eyes"
yellow = (255,255,1)

# Calculate & Initiate Servo Angle values
print("Servo setup")
min_servo = 0       # established by trial & error calibration
max_servo = 172     # established by trial & error calibration
center_servo = 83   # established by trial & error calibration
step_servo = (max_servo - min_servo) / steps   # 10.62 to achieve 10o sweep.
```

```
# INSTANTNIATE ARRAYS
print("Initiating arrays")
# Initiate angle nparray (instead of just python list)
# because polar plot needs array, not list
np.set_printoptions(precision=3)    #limits np.arange to 3 dec places (supposedly!)
degrees = np.arange(min_degrees, max_degrees + 1, step_degrees)
#print ("degrees", degrees)    #DEBUG
radians=degrees*(np.pi/180)
#print ("radians", radians)       #DEBUG

# Write degrees list to file
print("Writing initial arrays to file")
f.write("degrees: \t")
for i in range(steps):
    f.write(str(degrees[i]))
    f.write(",\t")
f.write("\n")

# Write radians list to file
f.write("radians:\t")
for i in range(steps):
    f.write(str(radians[i])[:5])
    f.write(",\t")
f.write("\n")

# Configure servo_angle array
np.set_printoptions(precision=3)
servo_angle = np.arange(min_servo, max_servo + 1, step_servo)
for i in range(steps):
    servo_angle[i] = int(servo_angle[i])
#print ("servo_stops", servo_angle) #DEBUG

# Write Servo settings to file
print("Writing servo settings to file")
f.write("servo stops:\t")
for i in range(steps):
    f.write(str(servo_angle[i]))
    f.write(",\t")
f.write("\n")

# init reading array/lists
print("Initialize distance arrays")
distance_IR = np.zeros(steps, dtype=float)
#print("distance_IR", distance_IR)     #DEBUG

distance_US = np.zeros(steps, dtype=float)
#print("distance_US", distance_US)     #DEBUG

distance_Avg = np.zeros(steps, dtype=float)
#print("distance_Avg=", distance_Avg)     #DEBUG

print("step_servo =", step_servo)
print("steps =", steps)
#DEBUG input("ENTER to proceed")

photoStops = [0, 45, 90, 135, 180]
```

```
# CAMERA ROUTINE
def TakePicture(degrees):
    print("SNAP!")  #DEBUG
    picPath = "/home/pi/Dexter/PHELAN/Odyssey/"
    currentTime = datetime.now()
    picName = currentTime.strftime("%Y.%m.%d-%H%M%S-") + degrees + '.jpg'
    with picamera.PiCamera() as camera:
        #camera.resolution = (1280, 720)
        camera.resolution = (640, 360)
        camera.capture(picPath + picName)

# BEEP & FLASH LIGHTS AS WARNING
def Safety():
        GPG3.set_eye_color(yellow)
        GPG3.open_eyes()
        beeper.sound(523.25)    #C5
        GPG3.led_on("left")
        sleep(0.25)

        beeper.sound_off()
        GPG3.close_eyes()
        GPG3.led_off("left")
        GPG3.led_on("right")
        beeper.sound(1046.50)   #C6
        sleep(0.25)

        beeper.sound_off()
        GPG3.open_eyes()
        GPG3.led_off("right")
        GPG3.led_on("left")
        beeper.sound(2093.00)   #C7
        sleep(0.25)

        beeper.sound_off()
        GPG3.close_eyes()
        GPG3.led_off("left")
        GPG3.led_on("right")
        sleep(.5)
        GPG3.led_off("right")

Safety()
sleep(wait_before_it_starts)
# Demonstrate full 180o servo range
# rotate the servo to the rightmost position
#DEBUG input('ENTER to test SERVO1 rightmost {}' .format(angle[0]))    #DEBUG#
servo.rotate_servo(servo_angle[-1])
sleep(wait_before_it_starts)
# rotate the servo to the leftmost position
#DEBUG input('ENTER to test SERVO1 leftrost {}' .format(angle[-1]))    #DEBUG#
servo.rotate_servo(center_servo)
sleep(wait_before_it_starts)
# rotate the servo to the center position
#DEBUG input("ENTER to center SERVO1 90o")
servo.rotate_servo(servo_angle[0])
sleep(wait_before_it_starts)

### continued...>
```

```
#DEBUG input("ENTER to begin scan")
# Sweep through angles & measure distances
print("Starting sweep")
for i in range(steps):
    try:
        servo.rotate_servo(servo_angle[i])
        #DEBUG print("degrees = ", degrees[i], "radians = ", radians[i] " servo = ",
servo_angle[i])
        # Take picture
        if degrees[i] in photoStops:
            TakePicture(str(degrees[i]))

        # Read US
        distance_US[i] = grovepi.ultrasonicRead(ultrasonic_ranger)
        # Read IR
        # IR distance in mm converted to cm
        distance_IR[i] = IR_distance_sensor.read_mm() / 10
        # Assign out of range readings to 200cm
        if (distance_US[i] > 495 or distance_IR[i] > 295):
            distance_US[i] = 200
            distance_IR[i] = 200
        distance_Avg[i] = (distance_US[i] + distance_IR[i])/2
        string_angle = str(int(servo_angle[i]))
        #print("degrees ", degrees[i], "\tservo ", string_angle[:4],
        #    "\tIR ", distance_IR[i], "\tUS ", distance_US[i], "\tAvg ", distance_Avg[i])
 #DEBUG
        #sleep(servo_delay)
        #input("ENTER for next step")
    except KeyboardInterrupt:
        print("\nResetting.  Good-Bye!\n")
        GPG3.reset_all()
        sys.exit(0)
        #break

# rotate the servo to the center position
#DEBUG input("ENTER to center SERVO1 90o")
servo.rotate_servo(center_servo)

#ALL-CLEAR SIGNAL
### NEXT: flashing lights OFF
beeper.sound(2093.00)   #C7
sleep(0.5)
beeper.sound_off()
#sleep(0.25)
beeper.sound(1046.50)   #C6
sleep(0.50)
beeper.sound_off()

### continued...>
```

```
print("Writing arrays to file") #DEBUG
f.write("distance_IR,\t")
for i in range (steps):
    f.write(str(distance_IR[i]))
    f.write(",\t")
f.write("\n")
f.write("distance_US,\t")
for i in range (steps):
    f.write(str(distance_US[i]))
    f.write(",\t")
f.write("\n")
f.write("distance_Avg,\t")
for i in range (steps):
    f.write(str(distance_Avg[i]))
    f.write(",\t")
f.write("\n")
f.close()

#for i in range(steps):
#   print("IR: ", distance_IR[i], "\tUS: ", distance_US[i], "\tAvg: ", distance_Avg[i])
#DEBUG

# SCATTER PLOT OF IR/US
print("Plotting")   #DEBUG
splat = plt.figure(1)
plt.scatter(distance_US, distance_IR)
plt.xlabel('US cm')
plt.ylabel('IR cm')
plt.title('IR vs US Distance Sensor')
plt.xlim([0, 300])
plt.ylim([0, 300])
# calc trend line
z = np.polyfit(distance_US, distance_IR, 1)
p = np.poly1d(z)
plt.plot(distance_US, p(distance_US), "r-")
#print("y=%.6fx+(%.6f)"%(z[0],z[1]))    #DEBUG
# save plot
print("saving scatter plot")    #DEBUG
plotPath = "/home/pi/Dexter/PHELAN/Odyssey/"
currentTime = datetime.now()
plotName = 'scatter_' + currentTime.strftime("%Y.%m.%d-%H%M%S-") + '.png'
plt.savefig(plotPath+plotName)
#plt.close()

# POLAR PLOTS OF IR, US
#print(degrees, "\t", radians, "\t", distance_IR, "\t", distance_US, "\t", distance_Avg)
#DEBUG
radar = plt.figure(2)
plt.polar(radians, distance_US, color="blue", label="IR")
plt.polar(radians, distance_IR, color="red", label="US")
plt.polar(radians, distance_Avg, color="green", label="Avg")
plt.legend()
print("saving polar plot")    #DEBUG
plotName = 'polar_' + currentTime.strftime("%Y.%m.%d-%H%M%S-") + '.png'
plt.savefig(plotPath+plotName)

### continued...>
```

```
plt.show()
plt.close()
#input()

# RESET particularly servo
print("Resetting")
GPG3.reset_all()
### end of program
```