Maker Session: Working with Raspberry Pi

In this session we will learn to work with Raspberry Pi computers and the Raspberry Pi camera module. Raspberry Pi computers are small, single-board computers that cost between $20-35. The Raspberry Pi computers you will work with today are Pi 2 Model B ($35, 4-core ARM CPU, 1GB RAM) Link. Our Raspberry Pis are running a Linux operating system "Raspbian" which is a fork of Debian 7 "Wheezy."

Outline

1. Basic Linux operations
2. Using the Raspberry Pi camera

Basic Linux operations

What is Linux?: Linux is an operating system, which is software that supports basic computer functions. Other examples of operating systems are Unix, Mac OS, Windows XP or Windows 7. You can do things like, make files, move files, copy files, make directories (folders) etc. etc. but instead of seeing icons like a Windows or Mac interface you would use text (the command line).

Your computer should boot to a desktop with Raspberry on it, if it just looks like a black screen with text you might be in the command line interface: To go to the desktop environment from the command line interface, type after logging in:

```
startx
```

From the desktop you can run a terminal program to continue to use the command-line interface (called LXTerminal).

Basic Linux shell commands

If you need help on how/why to use a command the best thing to do is to google it.

pwd - print the current working directory path. The "working directory" means the folder you are currently in.

ls - list the files and directories in the current working directory

cd - change directories directly to *home*

cd *dir* - change directories from the current working directory to *dir*

touch *file* - create the empty file *file*

rm *file* - remove the file *file*

mkdir *dir* - make the directory *dir*

rmdir *dir* - remove the directory *dir* (has to be empty)

cp *file1 file2* - create a copy of *file1* called *file2*

cp -r *dir1 dir2* - create a copy of *dir1* and its contents called *dir2*

mv *file1 file2* - move/rename *file1* to *file2*

head *file* - print the first 10 lines of *file* to *stdout*

tail *file* - print the last 10 lines of *file* to *stdout*

less *file* - opens *file* using a paging viewer

htop - display the current running processes (task manager/activity monitor). It is a modern version of **top** which should be available if **htop** is not.

df -h - display disk usage with human-readable units

gzip *file* - compress *file*

gunzip *file.gz* - decompress *file.gz*

tar zcf *archive.tar.gz dir* - create a compressed archive of *dir* (or a set of files)

tar zxf *archive.tar.gz* - decompress and extract the contents of *archive.tar.gz*

Ctrl+C - stop the current command

exit - log out of the current session

Using the Raspberry Pi camera

The Raspberry Pi camera module is a 5 megapixel, fixed-focus camera add-on for the Raspberry Pi. Each of your Raspberry Pi computers has a camera module installed. Your camera module is the Pi NoIR model (the infrared filter has been removed).

Take a picture

To take a picture, see it, and save it with the name *image.jpg* use the following command:

```
# this command will take a picture but it will put it in whatever working directory
(folder) you are in
# the -o tells the program what to name the image and also where to put it, if it
```

```
is just the name it puts the image in whatever folder you are currently in.

raspistill -o image1.jpg

# To see where you've save the image (what folder you are currently in) type

pwd

# To see the names of the files in your current folder type

ls -l
```

Pro-tip, if you want to rerun a command that you recently used, you can hit the uparrow to get to previous commands

To actually view the image, you can go to the file folder icon on the desktop and open the file or on the command line...

```
display /home/pi/images/image1.jpg
```

Flip a picture if it is upside down

Is your picture upsidedown when you look at it? If it is you will need to add the -vf (vertical flip) flag to your command.

```
raspistill -vf -o image1.jpg
```

Save images to a directory

The image *image.jpg* is saved to your current working directory. Instead you might want to save images to a specific directory.

```
mkdir images

#now you've made a folder named images

raspistill -o /home/pi/images/image1.jpg

#The -o flag tells the raspistill program that you want to save the file to a
specific place.
# remember if your camera is upsidedown to use the -vf flag.
```

Save images with unique names and some metadata

To take a picture, see it, and save it with the time stamp in front of the name image.jpg in the directory images in your home folder use the following command:

```
# if you add the timestamp to the name you can use the same command over and over
again (gives the imaage a new name and doesn't overwrite)
# the name of the image below would include the current year, month, day, hour, min,
and second

raspistill -o /home/pi/images/$(date +"%Y-%m-%d_%H:\%M:%S")_images.jpg

# now press the up arrow, the command you just ran should show up, take another
picture.
```

Take a time-lapse

After you take a few pictures and are satisfied with the setup you can take a time-lapse

```
# to take a timelapse you need to use the -tl option, which is the time between
images, but the time is in milliseconds
# you also need the -t option, which tells the program how long to run timelapse
# the command below takes an image named using the timestamp every 3 seconds for 1
min
# We cannot use the timestamp to name the images because how the program is written,
but we can use another way of naming the images
# by nameing the images image%05d.jpg we tell the program to give the image a 5
digit number at the end of the name
# So the first image would be named image00001.jpg

raspistill -t 60000 -tl 3000 -o /home/pi/images/image%05d.jpg
```

What happens if the computer shuts off during the time-lapse?

In order to make sure that the computer will keep running the time-lapse if the computer shuts off you need to modify crontab.

Use cron to take images on a schedule (time-lapse)

Crontab is an easy way to schedule repeating tasks so it can be used as another way to do time-lapse imaging, like time-lapse imaging.

```
#First let's make another folder so that we don't confuse it with the first
timelapse

mkdir /home/pi/timelapse2
```

```
#This opens up the crontab file in using the leafpad text editor program

sudo leafpad /etc/crontab
```

Crontab file format:

```
# Example of job definition:
# .---------------- minute (0 - 59)
# |  .------------- hour (0 - 23)
# |  |  .---------- day of month (1 - 31)
# |  |  |  .------- month (1 - 12) OR jan,feb,mar,apr ...
# |  |  |  |  .---- day of week (0 - 6) (Sunday=0 or 7) OR
sun,mon,tue,wed,thu,fri,sat
# |  |  |  |  |
# *  *  *  *  * command to be executed
```

add the following line to the cron file (you should see a list of commands that also start with an astericks).

```
#this tells the command to run every min of every hour every day of the month every
day of the week.

* * * * * pi /usr/bin/raspistill -o /home/pi/timelapse2/$(date
+"\%Y-\%m-\%d_\%H:\%M:\%S")_timelapse.jpg

# That might be too many pictures, so let's take a picture every 10 minutes

*/10 * * * * pi /usr/bin/raspistill -o /home/pi/timelapse2/$(date
+"\%Y-\%m-\%d_\%H:\%M:\%S")_timelapse.jpg

#save the file and close it
```

now we need to restart the cron to make sure the changes take effect. On the command line type

```
sudo service cron restart
```

Can I test my commands before putting them into crontab? - Yes! Just run them on the command line first in the terminal!!!! - Why use crontab? - It will automatically run task as scheduled, even after reboot! - Time is set after booting and internet access, but if no internet, it will set time to last available time, which can cause problems. Hope you have wifi!!!! - Alt: can hardwire a clock with battery.

Assembling your time-lapse into a short movie

first we need to write all of image names into a text file, but we can do that using the command line

```
# This takes all of jpg files in your current folder and writes their names to a
file called stills.txt

cd /home/pi/timelapse2/

ls *.jpg > stills.txt
```

now we can assemble the files into a time-lapse movie

```
# -ocv tells the mencoder program what video format to use
# -lavcopts tells the opctions you want for the video format (size etc.)
# -o tell the program what to name your file
# -mf tells the program what file format and the number of frames per second that
you want, as well as the file with all the image names

mencoder -nosound -ovc lavc -lavcopts vcodec=mpeg4:aspect=16/9:vbitrate=8000000 -vf
scale=1920:1080 -o timelapse.avi -mf type=jpeg:fps=24 mf://@stills.txt

#if you need to flip your images
mencoder -nosound -ovc lavc -lavcopts vcodec=mpeg4:aspect=16/9:vbitrate=8000000 -vf
scale=1920:1080 -flip -o timelapse.avi -mf type=jpeg:fps=24 mf://@stills.txt
```

To see your videos we will add them to a playlist here

playlist link

Now let's move your movie to a folder with your name

```
#let's make a directory (folder) with your name (no spaces)

mkdir /home/pi/yourname

#for example: mkdir /home/pi/malia

# then let's list the files to see if your folder is there

ls -l /home/pi/

# you should see a folder with your name

#now let's move your timelapse.mp4 movie there, make sure you replace the "yourname"
with your actual name

mv /home/pi/images/timelapse2/timelapse.mp4 /home/pi/yourname/timelapse.mp4

#check that the file is in there
```

```
ls -l /home/pi/yourname/
```

Browse all the images you have taken and pick your favorite one. Move that image to the folder with your name and we'll get it printed.