RasPi

DESIGN
BUILD
CODE

1

Get hands-on with your Raspberry Pi

Program a DRONE

Powered by your Raspberry Pi

52 Pages

Awesome RasPi features and guides

Welcome

Where do Raspberry Pi makers and programmers go for new ideas? We all love tinkering with Scratch and Python, and there's no greater reward than building your first Raspberry Pi robot, so we launched **RasPi** just for you. Full of amazing projects for you to try and showcasing the best work in the Pi community, we're sharing all our knowledge, ideas and enthusiasm to help you build bigger and better robots. And if it's your first time with Pi? Check out the Special Editions section of this app for some fantastic guide books. Here at **RasPi** we believe we can drive innovation by working together – get in touch with us and we may show off your project! So swipe the page and let's get inspired.

Gavin Thomas

Deputy Editor

Get inspired

Discover the RasPi community's best projects

Expert advice

Got a question? Get in touch and we'll give you a hand

Easy-to-follow guides

Learn to make and code gadgets with Raspberry Pi

From the makers of

LinuxUser & Developer

Join the conversation at...

@linuxusermag | Linux User & Developer | RasPi@imagine-publishing.co.uk

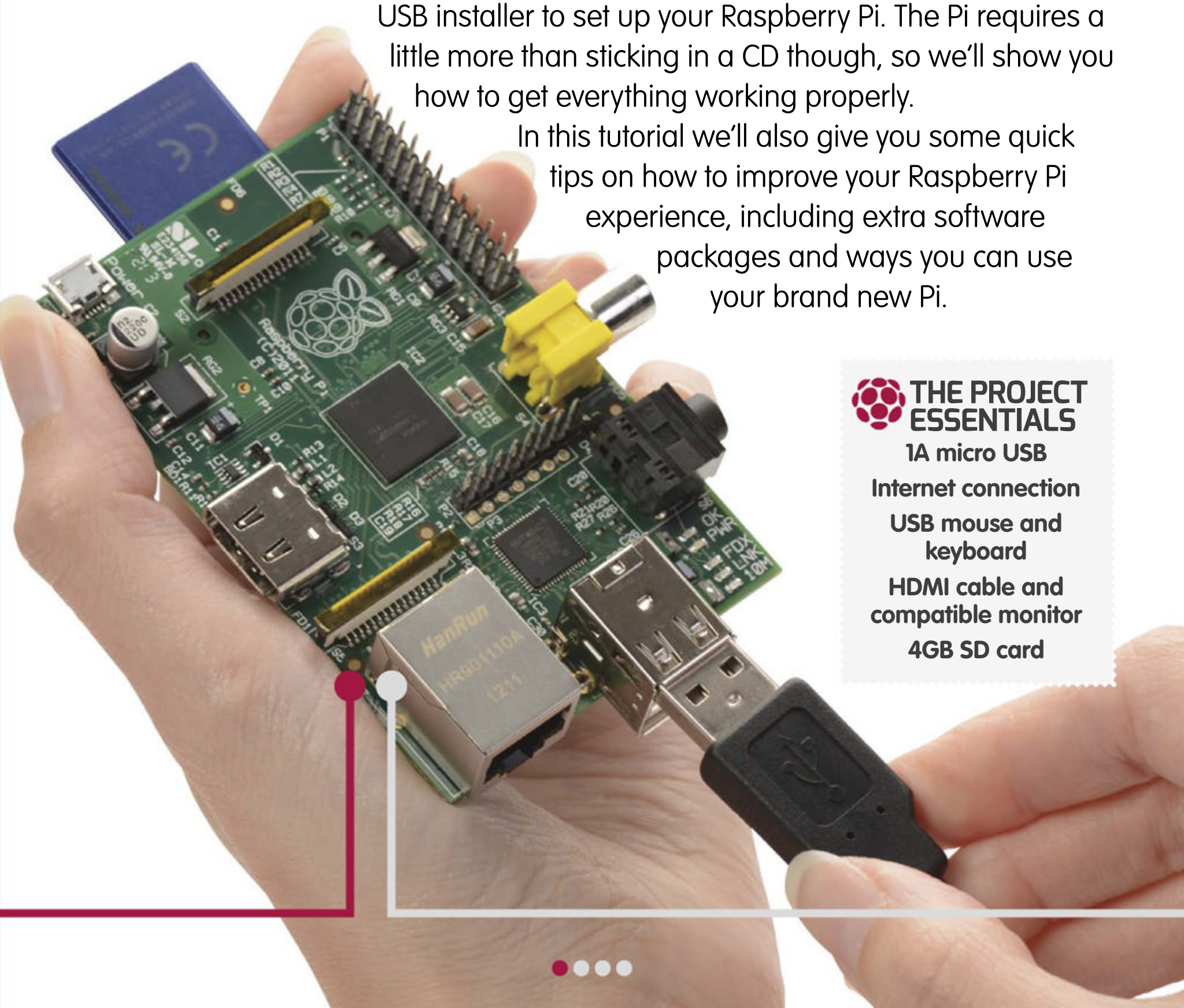
Contents

Set up your Raspberry Pi

Turn your Raspberry Pi into a fully functional PC and get to know the basics of setting it up

If you've just got your Raspberry Pi – you're probably wondering how exactly you're going to get started with it. If you're used to more traditional PCs, you might be expecting a CD drive or a USB installer to set up your Raspberry Pi. The Pi requires a little more than sticking in a CD though, so we'll show you how to get everything working properly.

In this tutorial we'll also give you some quick tips on how to improve your Raspberry Pi experience, including extra software packages and ways you can use your brand new Pi.



THE PROJECT ESSENTIALS

1A micro USB

Internet connection

USB mouse and keyboard

HDMI cable and compatible monitor

4GB SD card

01 Get your operating system

Head over to the Raspberry Pi website (**www.raspberrypi.org**) and download NOOBS, the New Out Of Box Software. This will come in a zip file; download it onto the SD card for your Raspberry Pi and extract the files from it here. Do not extract the files elsewhere and copy them over.

02 Connect your Pi

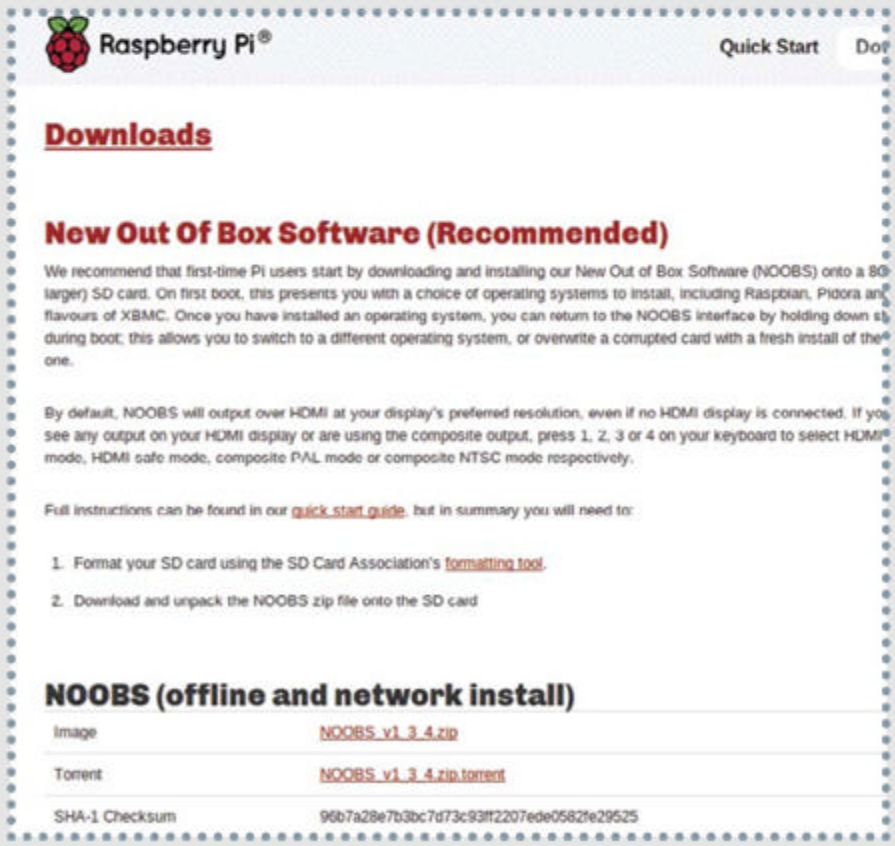
Without inserting the power cable, hook up everything to your Pi. You'll need a wired ethernet connection or a compatible wireless dongle, a USB mouse, a USB keyboard and a monitor or other form of display connected via HDMI.

03 Choose your distro

Plug in your SD card and finally connect the power. The Raspberry Pi will boot into NOOBS' distro selection menu. For all of our tutorials we will be using Raspbian, however the other distros each have their uses and you may want to consider using them at a later date.

Below Left The Official Raspberry Pi website hosts the New Out Of Box Software (NOOBS) that will get you off the starting blocks

Below right NOOBS will give you a choice of distros to use, including Debian-based Raspbian



04 Install Raspbian

Select Raspbian by clicking the check box to the left of it, and then click Install at the top of the Window. Confirm that you want to install, and it will go through the process of adding Raspbian to the SD card.

05 Set up Raspbian

There are a few things you need to do before Raspbian is ready. On the config screen, select Expand Filesystem to make sure the SD is being used properly. Then, go to Enable Boot to Desktop and select Desktop from the list. Go to Finish, and it will reboot Raspbian.

06 Update Raspbian

Make sure all the software on Raspbian is now up to date. To do this, open the LXTerminal and type:

```
$ sudo apt-get update
```

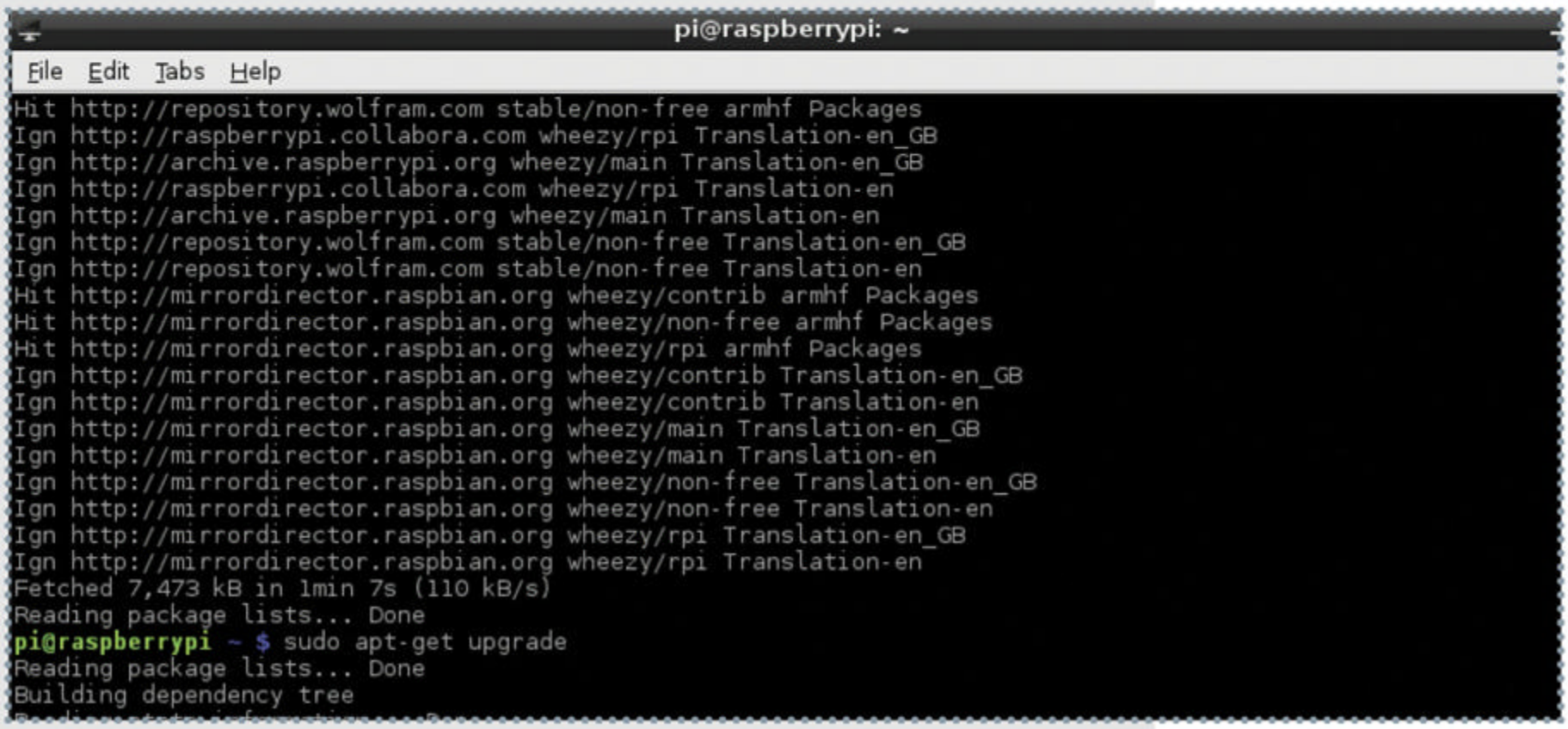
Once that's finished, follow it up with:

```
$ sudo apt-get upgrade
```

This may end up taking a few minutes, but it will update the software throughout Raspbian.

"The other distros each have their uses and you may want to use them at a later date"

Below The update part of the apt-get process checks to see what's available, then the upgrade part actually does the work



07 Get new software

There are two ways to get more packages for Raspbian – either through the Pi Store link, or via the package manager in the terminal. You'll get a different selection of apps on the two services, with a greater focus on general software tools in the package manager

08 Get an office suite

Raspbian does not have any form of office functionality by default, only a basic text editor. You can add LibreOffice though, which can be done via the Pi Store. Open up the Store and go to Apps; here you'll find LibreOffice as a free download. Once you've created an account, it will download and install it.

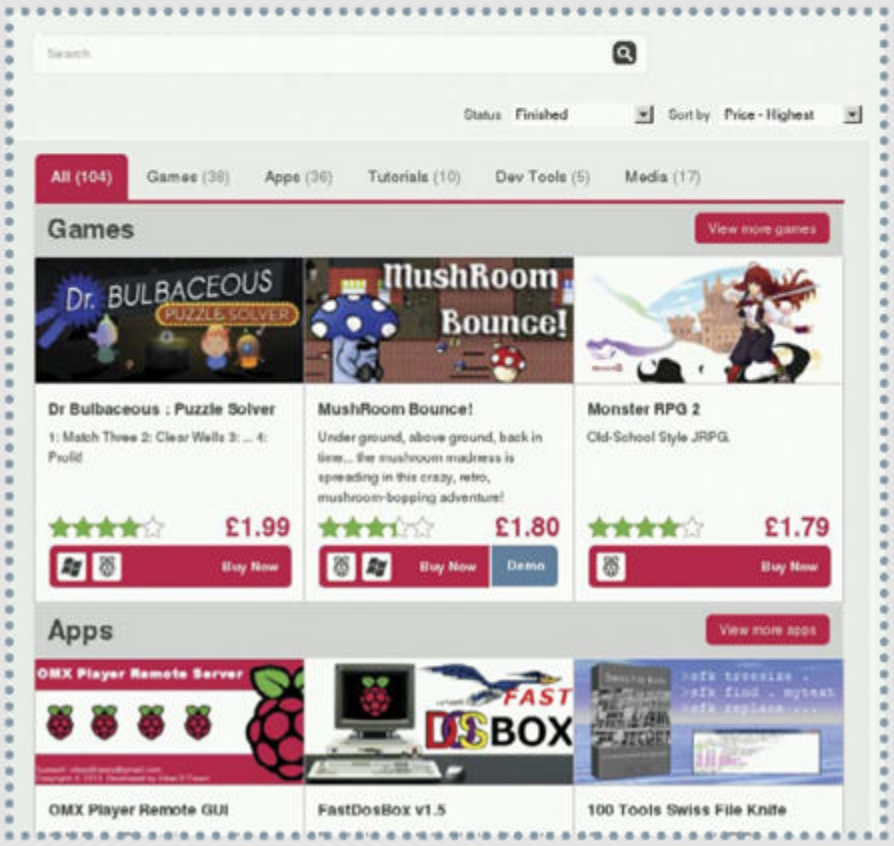
09 Get a better browser

Midori is an excellent browser, however you can also get Chromium to work on Raspbian. This is the open source version of Google's Chrome browser, which you can get by opening the terminal and typing:

```
$ sudo apt-get install chromium
```

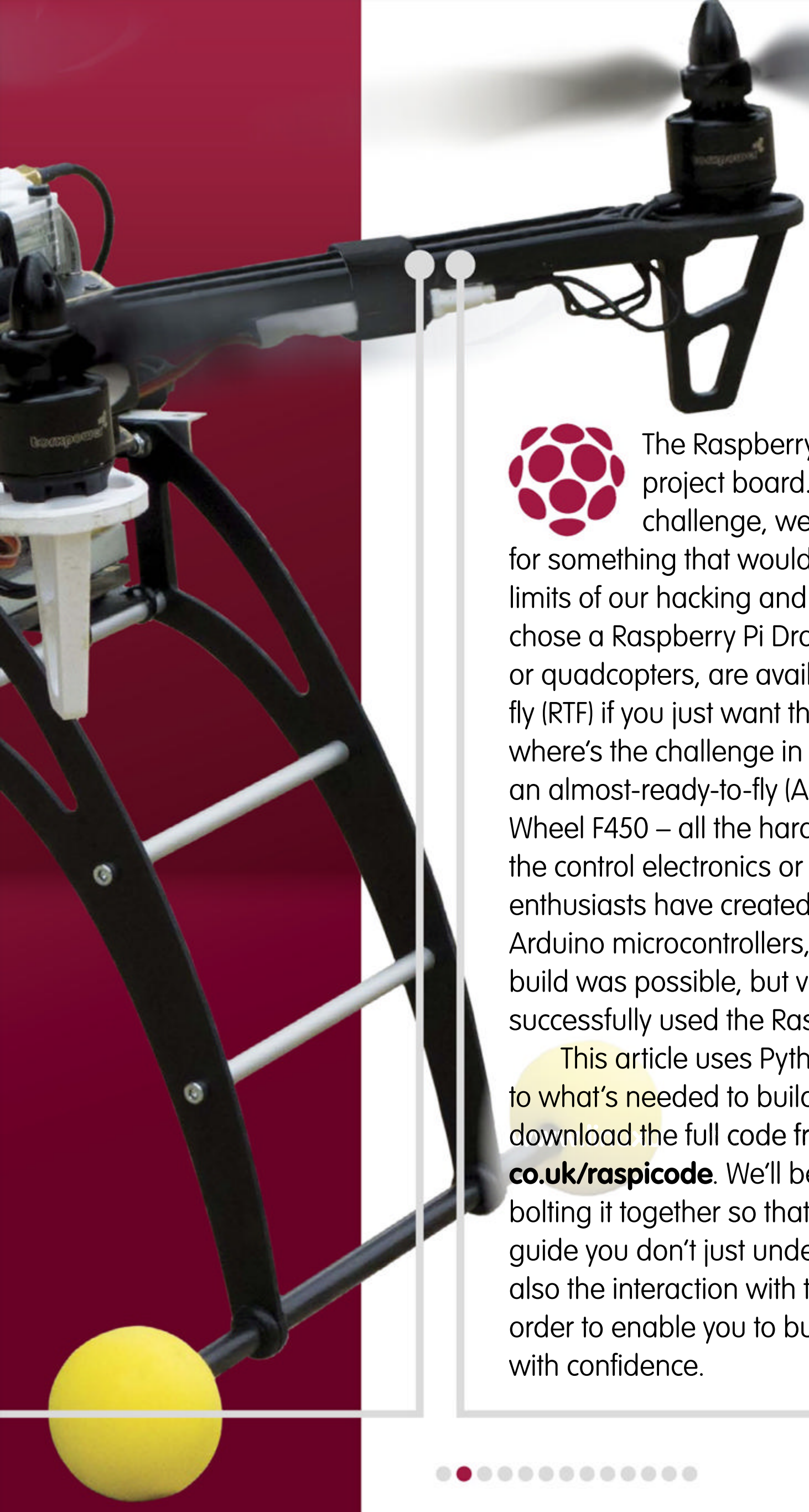
Below Left The Pi Store is packed full of games and utilities for your Raspberry Pi

Below right There are alternatives out there like OpenOffice, but LibreOffice is our favourite suite



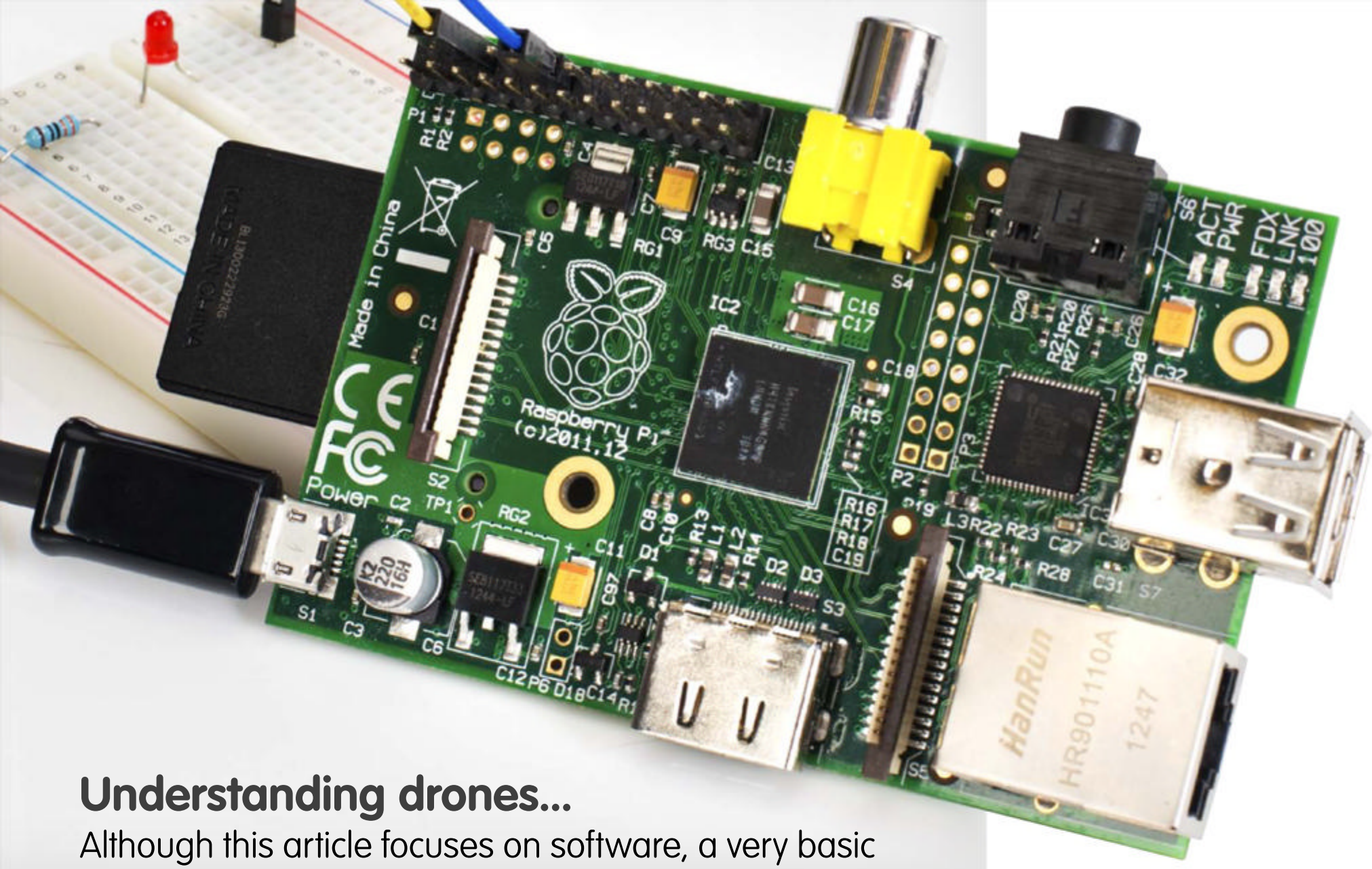
Program a Raspberry Pi Drone

How do you beat a Raspberry Pi robot? You give it wings. Andy Baker shows us how to code our way into the clouds



The Raspberry Pi is a fantastic project board. Since we love a challenge, we set out looking for something that would really push the limits of our hacking and coding skills. We chose a Raspberry Pi Drone. Kits for drones, or quadcopters, are available as ready-to-fly (RTF) if you just want the joy of flight, but where's the challenge in that? We started with an almost-ready-to-fly (ARF) kit – the DJI Flame Wheel F450 – all the hardware, but none of the control electronics or software. Many enthusiasts have created DIY drones using Arduino microcontrollers, so we knew a DIY build was possible, but very few, if any, have successfully used the Raspberry Pi.

This article uses Python code as a guide to what's needed to build a drone. You can download the full code from **www.linuxuser.co.uk/raspicode**. We'll be metaphorically bolting it together so that by the end of this guide you don't just understand the code but also the interaction with the real-world, all in order to enable you to build your own drone with confidence.



Understanding drones...

Although this article focuses on software, a very basic background in the hardware from the kit is necessary to provide context. A quadcopter-type drone, which we're building here, has four propellers pointing upwards to the sky (hence the name), each attached to its own brushless DC motor at one of the four corners of (usually) a square frame. Two motors spin clockwise, two anticlockwise, to minimise angular momentum of the drone in flight.

Each motor is driven independently by an electronic speed controller (ESC). The motors themselves have three sets of coils (phases), and the ESCs convert a pulse-width-modulation (PWM) control signal from software/hardware to the three phase high-current output to drive the motors at a speed determined by the control signal. The power for the ESCs and everything else on the system comes from a Lithium Polymer battery (LiPo) rated at 12V, 3300mA with peak surge current of 100A – herein lies the power!

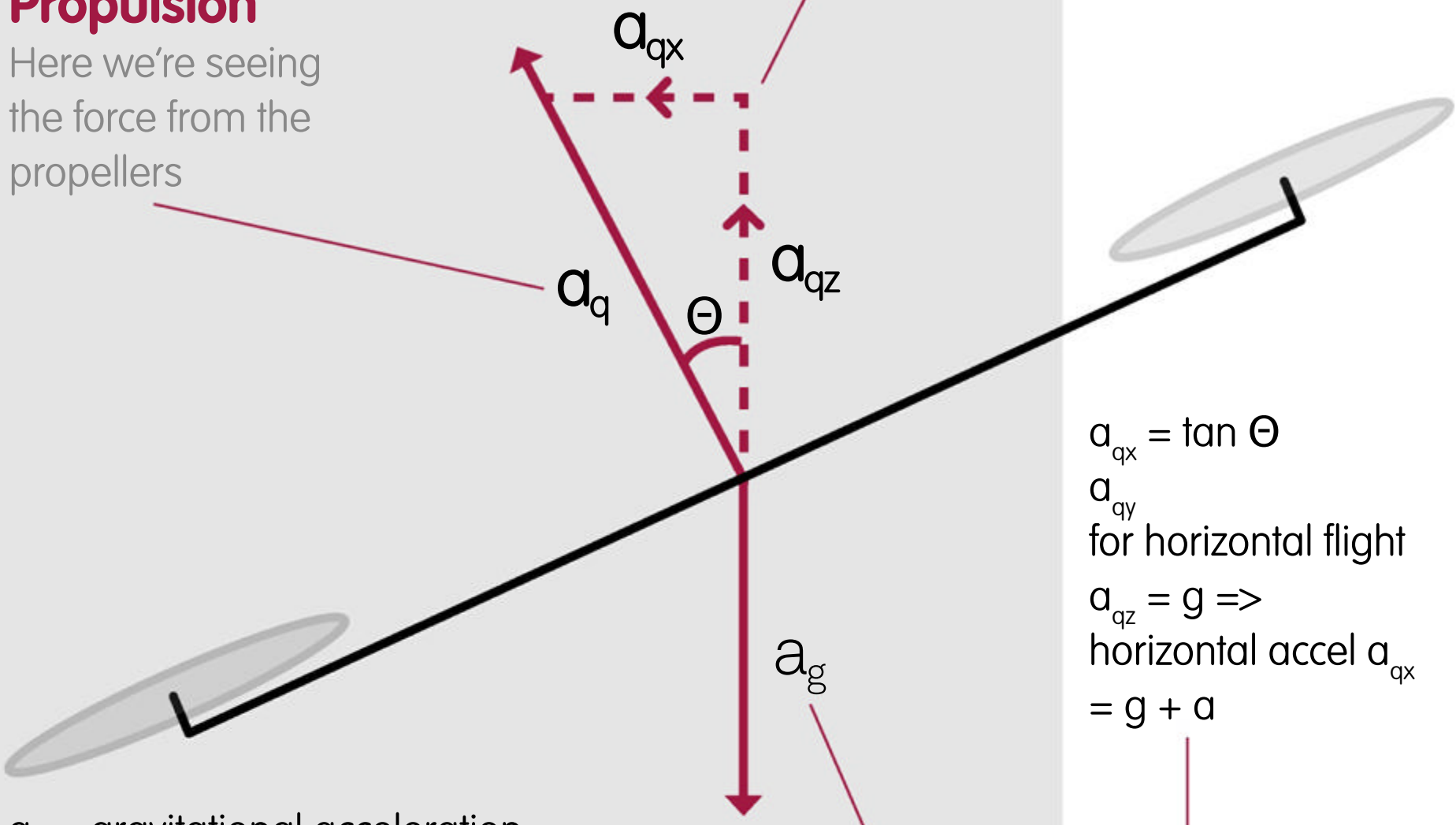
"A quadcopter-type drone has four propellers (hence the name)"

Vectors

Propeller force relative to Earth's axis (horizontal / vertical)

Propulsion

Here we're seeing the force from the propellers



a_g = gravitational acceleration
a_q = drone acceleration
a_{qx}, a_{qy} } = a_q (reorientated to Earth's axes)
Θ = angle of tilt derived from accel + gyro

Gravity

This term denotes the force of gravity

$a_{qx} = \tan \Theta$
a_{qy}
for horizontal flight
$a_{qz} = g \Rightarrow$
horizontal accel a_{qx}
$= g + a$

Angle

This is the angle of tilt as defined by the drone's sensors

Interpreter

The command interpreter converts a series of commands either from a radio control or programmed into the code itself. The commands combine the direction and speed compared to the horizon that the user want the drone to follow. The code converts these commands into a series of targets for vertical speed, horizontal speed and yaw speed – any command from a pair of joysticks can be broken down into a set of these targets.

Above How sensors in the drone point of view are converted to the Earth (horizontal/vertical) viewpoint to provide horizontal motion

Inputs

The inputs to the drone come from a series of electronic sensors providing information about its movement in the air. The main two are an accelerometer which measures acceleration force (including gravity) in the three axes of the drone, and a gyroscope which measures the angular speed with which the drone is pitching (nose/tail up and down), rolling (left/right side up and down), and yawing (spinning clockwise and anticlockwise around the central axis of the drone itself).

Axes

The accelerometer is relative to the orientation of drone axes, but the command targets are relative to the Earth's axes – the horizon and gravity. To convert the sensor output between the drone axes and the Earth axes needs a little trigonometry and knowledge of the tilt angles in pitch and roll axes of the drone with respect to the Earth.

Angles

Both the accelerometer and gyro can provide this angle information, but both have flaws.

The accelerometer output can be used to calculate the angle by using the Euler algorithm. However, the accelerometer output is plagued by noise from the motors/propellers, meaning a single reading can be hugely inaccurate; on the plus side, the average reading remains accurate over time.

In contrast, the gyro output does not suffer from the noise, but since it is the angular speed being measured, it needs to be integrated over time to find the absolute angle of the drone in comparison to the horizon. Rounding errors in the integration lead to ever increasing errors over time, ultimately curtailing the maximum length of a flight.

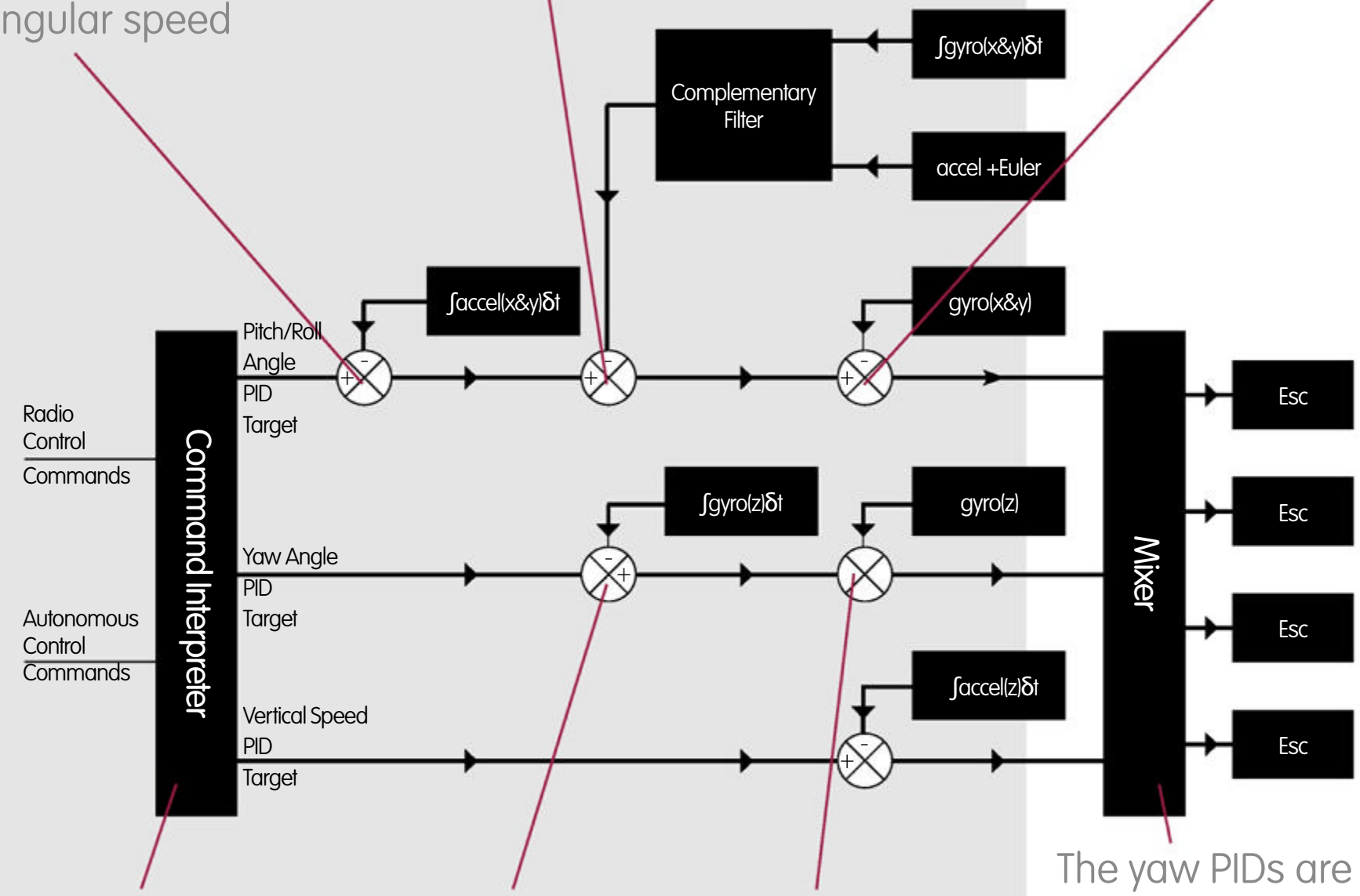
"Both the accelerometer and gyro can provide the angle information, but both have flaws"

As you read this article, you can follow the corresponding code by searching for an equivalent tag comment; for example, to find the code related to the '# Angles' section of the article, simply search the code for '# Angles'.

The angle PID takes the desired tilt angle from the horizontal speed PID and feedback from the gyro and the accelerometer combined to produce the tilt angle feedback. This results in the required angular speed

The angular speed PID takes the required angle PID out, with feedback from the gyro to apply power to the motors. This produces the required tilt and horizontal movement

The PID outputs are applied appropriately to each propeller ESC to effect the change defined by the proceeding commands



The horizontal speed PID takes movement commands and feedback from integrated gyro sensors to define the desired tilt and angle to achieve the required speed

The vertical speed PID directly controls the desired rate of climb or descent with feedback from integrating Z-axis accelerometer data

Commands from the radio-control or programmed route are processed and converted into a set of desired movements in the X, Y and Z axes

The yaw PIDs are used to ensure the drone doesn't rotate its Z-axis in flight

Above PIDs connecting commands and sensor feedback to produce an output driving the propeller blade (E)lectronic (S)peed (C)ontrollers

Filter

Although independently they are both flawed, they can be merged mathematically such that each compensates for the flaws in the other, resulting in a noise-free, long-term accurate reading. There are many versions of these mathematical noise/drift filters. The best common one is by Kalman; the one we've chosen is slightly less accurate, but easier to understand and therefore to code: the complementary filter.

Now with an accurate angle in hand, it's possible to convert accelerometer sensor data to inputs relative to the Earth's axes and work out how fast the drone is moving up, down, left, right and forwards and backwards compared to the targets that have been set.

PIDs

So we now have a target for what we want the drone to do, and an input for what it's doing, and some motors to close the gap between the two; all we need now is a way to join these together. A direct mathematical algorithm is nigh on impossible – accurate weight of the drone, power per rotation of each blade, weight imbalance etc would need to be incorporated into the equation. And yet none of these factors is stable: during flights (and crashes!), blades get damaged, batteries move in the frame, grass/mud/moisture changes the weight of the drone, humidity and altitude would need to accounted for. Hopefully it's clear this approach simply won't fly.

Instead, an estimation method is used with feedback from the sensors to fine-tune that estimate. Because the estimation/feedback code loop spins at over 100 times a second, this approach can react to 'errors' very quickly indeed, and yet it knows nothing about all the factors which it is compensating for – that's all handled blindly by the feedback; this is the PID algorithm.

> "A drone in a headwind will drift backwards due to the force applied by the wind"

It takes the target, subtracts the feedback input, resulting in the error. The error is then processed via a Proportional, Integral and a Differential algorithm to produce the output.

Blender

The outputs are applied to each ESC in turn: the vertical speed output is applied equally to all blades; the pitch rate output is split 50/50 subtracting from the front blades and adding to the back, producing the pitch. Roll is handled similarly. Yaw too is handled in a similar way, but applied to diagonal blades which spin in the same direction.

These ESC-specific outputs are then converted to a PWM signal to feed to the hardware ESCs with the updated propeller/motor speeds.

"The vertical speed output is applied equally to all blades"

Code and reality

In this code, there are nine PIDs in total. In the horizontal plane, for both the X and Y axes, the horizontal speed PID converts the user-defined desired speed to required horizontal acceleration/ angle of tilt; the angles PID then converts this desired tilt angle to desired tilt rate which the rotation speed PID converts to changes in motors speeds fed to the front/back or left/right motors for pitch/ roll respectively.

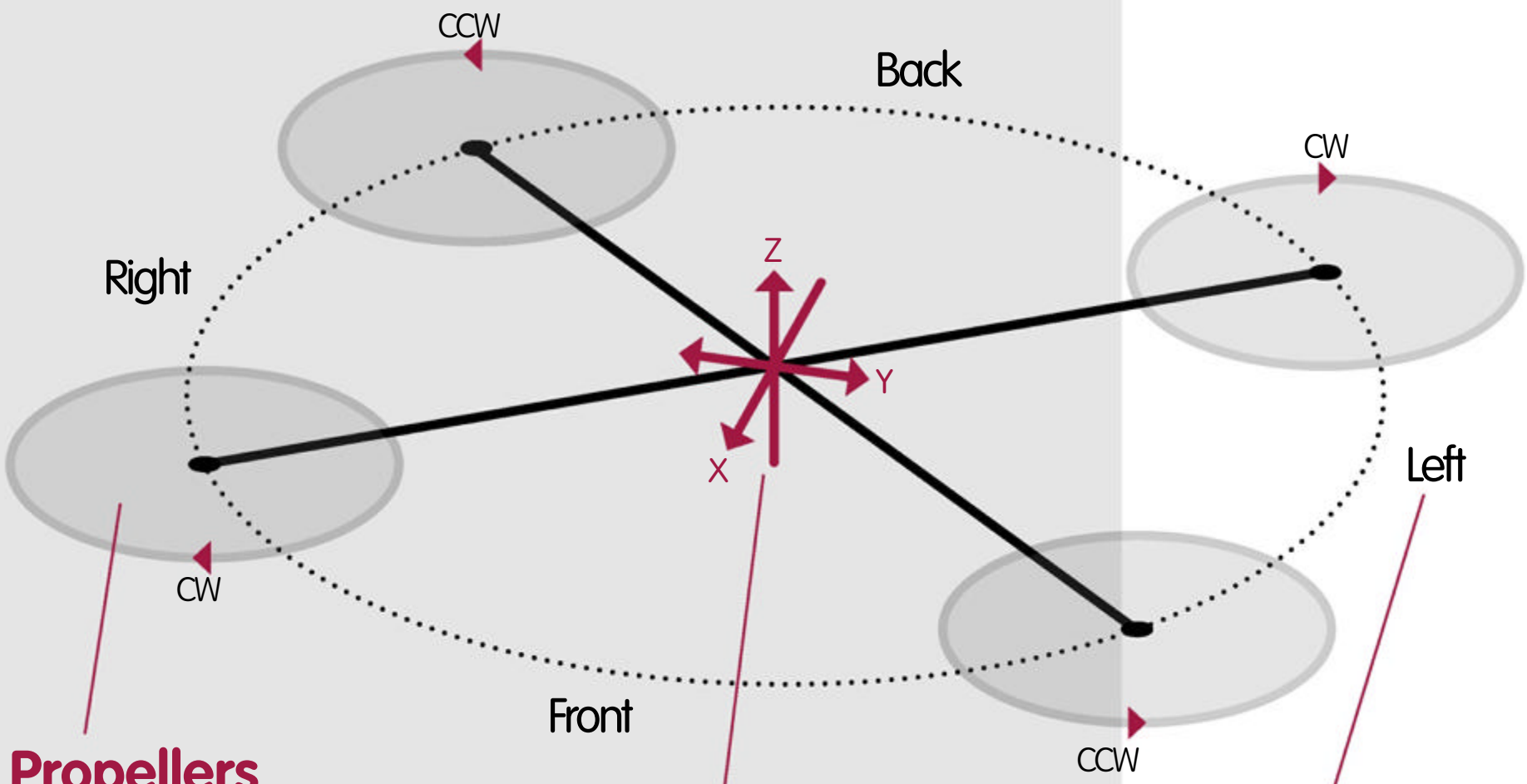
In the vertical direction, a single PID converts the desired rate of ascent/descent to the acceleration output applied to each plate equally.

Finally, prevention of yaw (like a spinning top) uses two PIDs – one to set the desired angle of yaw, set to 0, and one to set the yaw rotation speed. The output of these is fed to the diagonally opposing motors which spin in the same direction.

The most critical of the nine are pitch/roll/yaw stability. These ensure that whatever other requirements enforced by other PIDs and external factors, the drone is stable in achieving those other targets; without this stability, the rest of the PIDs cannot work. Pitch is controlled by relative speed differences between the front and back propellers; roll by left and right differences, and yaw by clockwise/ anticlockwise differences from the corresponding PIDs' outputs. The net outputs of all three PIDs are then applied to the appropriate combination of motors' PWM channels to set the individual pulse widths.

With stability assured, some level of take-off, hover and landing can be achieved using the vertical speed PID. Placing the drone on a horizontal surface, set the target to 0.5 m/s and off she zooms into the air, while the stability PID ensures that the horizontal attitude on take-off is maintained throughout the short flight, hover and landing.

"With stability assured, some level of take-off, hover and landing can be achieved using the vertical speed PID"



Propellers

The propellers are set diagonally to the x, y axes, and rotate as shown to reduce yaw (rotation about the z-axis)

Sensors

The drone's sensors report data according to these x, y and z axes

Orientation

The overall orientation of the drone depicting front, back, left and right in relation to the sensor and propeller layouts

Up to this stage, the PIDs are independent. But what about for horizontal movement target, and suppression of drifting in the wind?

Taking the drift suppression first, a drone in a headwind will drift backwards due to the force applied by the wind. To compensate, it must tilt nose down at some angle so that some of the propellers' thrust is applied horizontally to counteract the wind. In doing so, some of the power keeping the drone hovering at a fixed height is now battling the wind; unless the overall power is increased, the drone will start descending.

Horizontal movement is more complex still. The target is to move forwards at say one metre per second. Initially the requirement is similar to the headwind compensation

Above The orientation of the drone compared to the direction of travel, the rotation of the propellers and the axes used in the code

“The X and Y axes horizontal speed PIDs’ outputs are used as the pitch and roll angle PIDs targets”

– nose down plus increased power will apply a forward force leading to forward acceleration. But once that horizontal speed is attained, the drone needs to level off to stop the acceleration, but at the same time, friction in the air will slow the movement. So there’s a dynamic tilting fore/aft to maintain this stable forward velocity.

Both wind-drift suppression and controlled horizontal movement use nested PIDs; the X and Y axes horizontal speed PIDs’ outputs are used as the pitch and roll angle PIDs targets; their output feeds the pitch and roll rate

PIDs to ensure stability while meeting those angular targets. The sensor feedback ensures that as the desired horizontal speed is approached, the horizontal speed PID errors shrink, reducing the targets for the angular pitch PID, thus bringing the drone's nose back up to horizontal again.

Hopefully it now becomes clearer why accurate angle tracking is critical: in the nose-down, headwind example, the input to the vertical speed PID from the sensors is reduced by the cosine of the measured angle of 'copter tilt with respect to the horizon.

Similarly, X and Y axis speed PID sensor inputs need compensating by pitch and roll angles when comparing target speeds against accelerometer readings.

Experimentation and tuning

While the code accurately reflects everything we've described here, there's one critical set of steps which can only be found through live testing; these are the PID gains. For each PID running, there is an independent Proportional, Integral and Differential gain that can only be found with estimation/ experimentation. The results for every drone will be different. Luckily there is a relatively safe way to proceed.

First, find the PWM take-off speed: this is done by sitting your drone on the ground and slowly increasing the PWM value until she starts looking light-footed – for your expert, this was about the 1590us pulse width (or 1000us + 590us, as shown in the code).

Next, sorting out the stability PIDs – assuming your drone is square and its balance is roughly central, then the result of pitch tuning also applies to yaw tuning. For pitch tuning, disable two diagonally opposed motors and rest these on a surface – the drone sits horizontal in between. Power up the dangling motors' PWM to

"For pitch tuning, disable two diagonally opposed motors and rest these on a surface – the drone sits horizontal in between"

just under take-off speed (1550us pulse width in our expert's case). Does the drone rock manically, wobble in some pretence of control, self-right when nudged, or do nothing? Tweak the P gain accordingly. Once P gain is good, add a touch of I gain – this will ensure return to 0 as well as stability. D gain is optional, but adds firmness and crisp response. Tapping a D-gain stable drone is like knocking on a table – it doesn't move.

Vertical speed PID can be guesstimated. 1590us is taking off; desired take-off speed is 0.5m/s so a P gain of 100 is okay. No I or D gain needed.

With that a real take-off, hover and landing are safe, which is good as these are the only way to tune the directional PIDs. Just be cautious here – excessive gains lead to drones slamming into walls or performing somersaults in mid-air before powering themselves into the ground. Best executed outside in a large open field/ garden/park where the ground is soft after overnight rain!

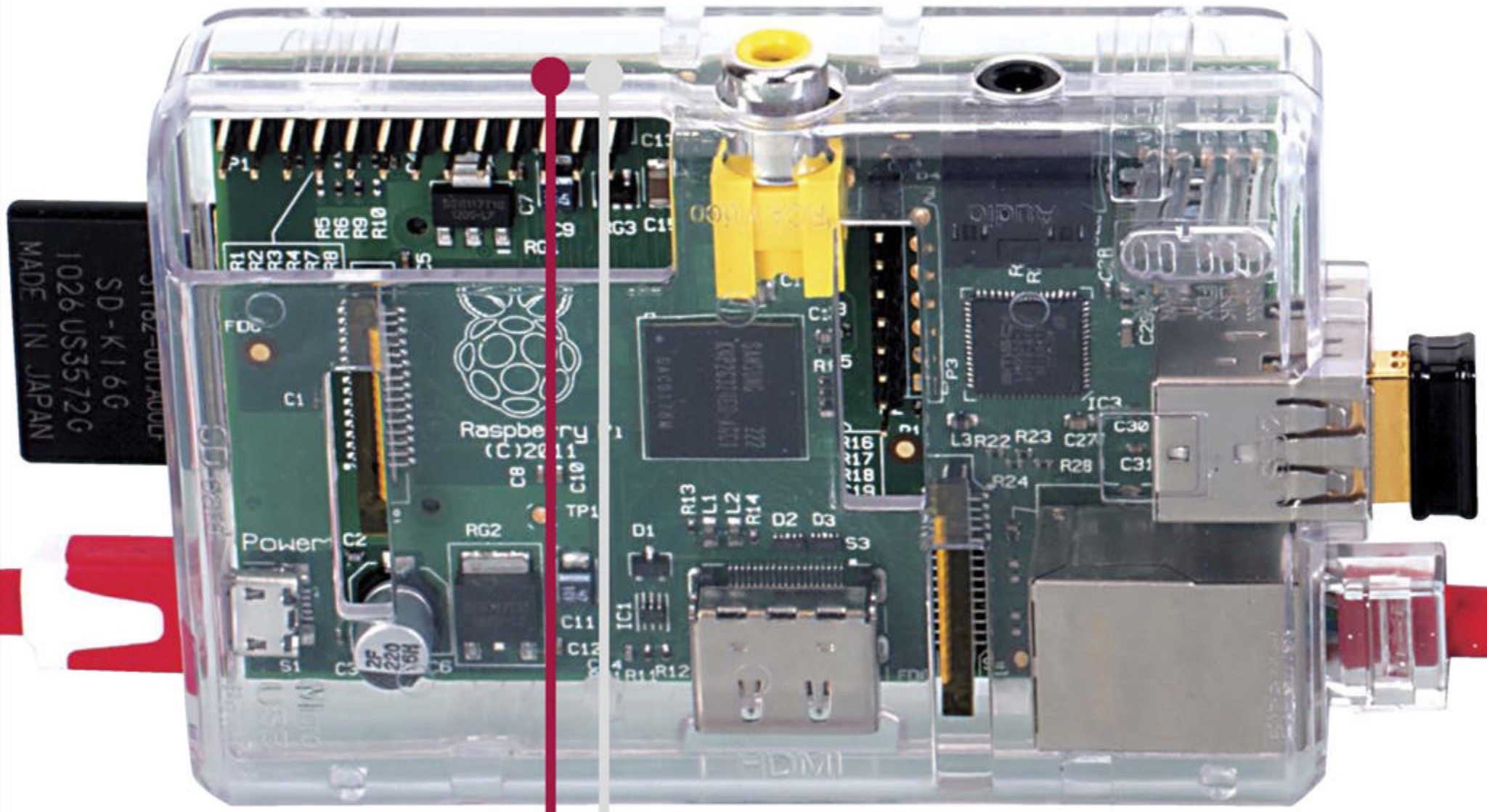
There isn't a shortcut to this, so just accept there will be crashes and damage and enjoy the carnage as best you can!

Assuming all the above has gone to plan, then you have a drone that takes off, hovers and lands even in breezy conditions. Next step is to add a remote control, but that's for another issue of **RasPi**…

"Excessive gains lead to drones slamming into walls or performing somersaults in mid-air before powering themselves into the ground"

What is Onion Pi?

Not a poorly named competitor to the Raspberry Pi, but an interesting way to use your Pi for added security



"Basically, all the information is encrypted every time it's sent to another node in the network, and it creates several layers of encryption"

Q So what is an Onion Pi then? You say it's not a competitor?
A That's correct. Onion Pi is the name for a Raspberry Pi kit that Adafruit Industries has put together which enables you to create a secure wireless router using the Raspberry Pi.

Q Okay, so it's a product that uses the Raspberry Pi. What is Adafruit Industries, though?
A Adafruit Industries is an online resource for learning about electronics and helping makers of every age and skill level build projects. Adafruit also provides and sells tools and equipment that its experts have tested themselves to make sure these meet its quality standards.

Q You said it's for makers – who and what is a maker?
A The term 'maker' has been hitting the mainstream media a lot recently, but it's still not widely known. Makers are people that do simply that – they make. It's mostly used for people who create practical projects in their spare time for fun, although some of the more famous maker projects have gone on to be sold as actual products. There are several Maker Faires held around the world where people can showcase their creations just for the fun of it. You can learn more about these Maker Faires at the official website: makerfaire.com. If you can attend a Faire, we recommend it: they're fun and inspirational.

Q Okay, so Adafruit Industries is for makers. How is this related to Linux, FOSS and even the Raspberry Pi?
A As showcased often in **Linux User & Developer**, the Raspberry Pi has become extremely popular as a cheap and effective way to power the computerised and/or automated parts of a practical project. Adafruit itself is a champion of open hardware and has its own distro image

What you'll need

SD card with Raspbian
The easiest distro to get, frankly, but the one you'll need in order to turn your Raspberry Pi into a fully functional privacy box.

Compatible Wi-Fi adaptor
Adafruit sells Wi-Fi adaptors that it knows work, and provides some general drivers that some adaptors can use to work as a wireless hotspot.

"There are several Maker Faires held around the world"

for the Pi, based on Raspbian. This image comes preloaded with drivers for some of the products Adafruit sells.

Q All right, that's Adafruit explained. Tell me more about the Onion Pi. First of all, you're making it a wireless router?

A That's correct – the simple concept is that we're using the hardwired Ethernet port to deliver internet directly to the Raspberry Pi. This can be done by just plugging in the cable, nothing too difficult. We then also set up a wireless USB stick to work as a wireless receiver or hotspot, effectively allowing other devices to connect to the Raspberry Pi. We then pass-through the internet via this wireless connection, allowing the connected devices to connect wirelessly to the internet through the Raspberry Pi itself.

Q Right. What would you really use that for, though?

A Many things, really. Some hotels offer a wired connection but no Wi-Fi, so this way you'd be able to connect multiple devices to the internet, such as phones or laptops, that you may not necessarily want to tether to one location. It may also be cheaper than the Wi-Fi service they offer, and only requires you to register one device on their network. It could also work as a sort of cheap wireless repeater in a house with bad signal strength, or replace a temporarily broken router.

Q Fair enough I suppose. What's the secure part all about then?

A Well, the Onion Pi routes all the internet traffic through Tor, an 'anonymity network'. Traffic is redirected through a worldwide network of thousands of volunteer relays, effectively hiding a user's location and internet usage. Like when someone's doing a trace in a bad Nineties hacking

What you'll need

SD card with Raspbian
The easiest distro to get, frankly, but the one you'll need in order to turn your Raspberry Pi into a fully functional privacy box.

Stable power supply
While it's always recommended to power a Raspberry Pi with a dedicated power supply, the Onion Pi will happily run off a laptop's USB port.

"The Onion Pi routes all the internet traffic through Tor, an 'anonymity network'"

film and they've 'bounced the signal' off various satellites or something. Only a little more complex, and with less Robert Redford.

Q Hey hold on, what's wrong with Sneakers?
A Actually, we really like Sneakers. We retract our previous statement. No more secrets.

Q Good. Anyway, what's all this signal bouncing in aid of?
A Security and anonymity, as we've already mentioned. It's for your own privacy, makes you difficult to trace. While it could technically be used for secret spy work, most people use Tor so they don't get hammered with weird targeted ads and increased prices from particularly shady online companies. This was before the whole NSA and PRISM thing.

Q So Tor let's me hide what I'm doing from the NSA?
A According to recent news reports, maybe not, but you probably have bigger things to worry about if there's a possibility the NSA is keeping an eye on you.

Q Just checking. Wait, hold on, why is it called the Onion Pi?
A Tor stands for the The Onion Router, and performs onion routing. Basically, all the information is encrypted every time it's sent to another node in the network, and it creates several layers of encryption. Onions have several layers. Computer scientists like to give things simple or descriptive names, so it stuck. It's actually patented by the US Navy apparently.

Q I think I'm starting to understand now. Why would you want to do this on a Raspberry Pi, though?
A Well, some of the same reasons as the normal

"Tor stands for the The Onion Router, and performs onion routing"

Raspberry Pi wireless router apply; however, you can also use this as your main, encrypted router. Most commercial wireless routers won't allow you to do this unless you start flashing the firmware, whereas you can just hook a Raspberry Pi up and it handles all the Tor information. Anything connected to the network won't need to be set up specifically to use Tor itself, as the Pi will already be routing it through the network.

> "You can just hook a Raspberry Pi up and it handles all the Tor information"

Q Okay, I think you've sold me on the concept. Convenient privacy. Do I need to buy the specific kit to make an Onion Pi, though?

A No, you don't! It doesn't even specifically require the Adafruit OS either. What you need is a Raspberry Pi running Raspbian, a wired connection, and a USB wireless adaptor that can be used in this specific hotspot setup. You can find out more on what you need, and how to do it, on the Adafruit tutorial: **bit.ly/1jctkPS**.

Q You seem to know an awful lot about this. Have you covered this before in the magazine?

A I'm glad you asked, yes we have. We first did the wireless router tutorial in issue 129 of **Linux User & Developer** magazine, and did a full tutorial on it in issue 131. You can buy both digitally on **GreatDigitalMags.com**, or buy the physical copy in the Imagine Shop (**imagineshop.co.uk**) – the printed back issues disappear fast, so you'll need to be quick.

Use an accelerometer with Raspberry Pi

Adding tilt technology to your next Raspberry Pi project is easier than you might think

We use accelerometers every day. They're in our phones, game controllers and help keep our cars on the ground and planes in the air. In this project we'll be using an ADXL345 accelerometer kit to create a prototype controller for a *Space Invaders*-style game we're working on – we'll return to this code in later issues. The kit is only £20, but does require about 15 minutes of basic soldering. Beyond that, the physical setup is simple. We'll be placing the ADXL345 on our breadboard controller and using just four wires to communicate with the Raspberry Pi's GPIO pins using I2C. Let's start by making sure your Pi is up to date. Open a terminal window and type:

```
sudo apt-get update && sudo apt-get upgrade
```

01 Install i2c-tools

First, ensure your Pi can use I2C to talk to the ADXL345 module. In a terminal, type:

```
sudo apt-get install i2c-tools
```

followed by

```
sudo nano /etc/modprobe.d/raspi-blacklist.conf
```

This will open a file in nano – locate the line 'blacklist i2c-bcm2708' and comment it out so it looks like this:

```
  #blacklist i2c-bcm2708
```

Save with Ctrl+X, then press Y and Enter to exit. Finally, edit the /etc/modules file with nano, adding the line

```
i2c-bcm2708 i2c-dev
```

to ensure it loads every time you boot. Reboot to finish the operation.

> "We'll be using an ADXL345 accelerometer kit to create a prototype controller for a Space Invaders-style game"

THE PROJECT ESSENTIALS

ADXL accelerometer
shop.pimoroni.com

Latest Raspbian Image
www.raspberrypi.org/downloads

Breadboard
shop.pimoroni.com

Male-to-female prototyping cables
shop.pimoroni.com

30W soldering iron
shop.pimoroni.com

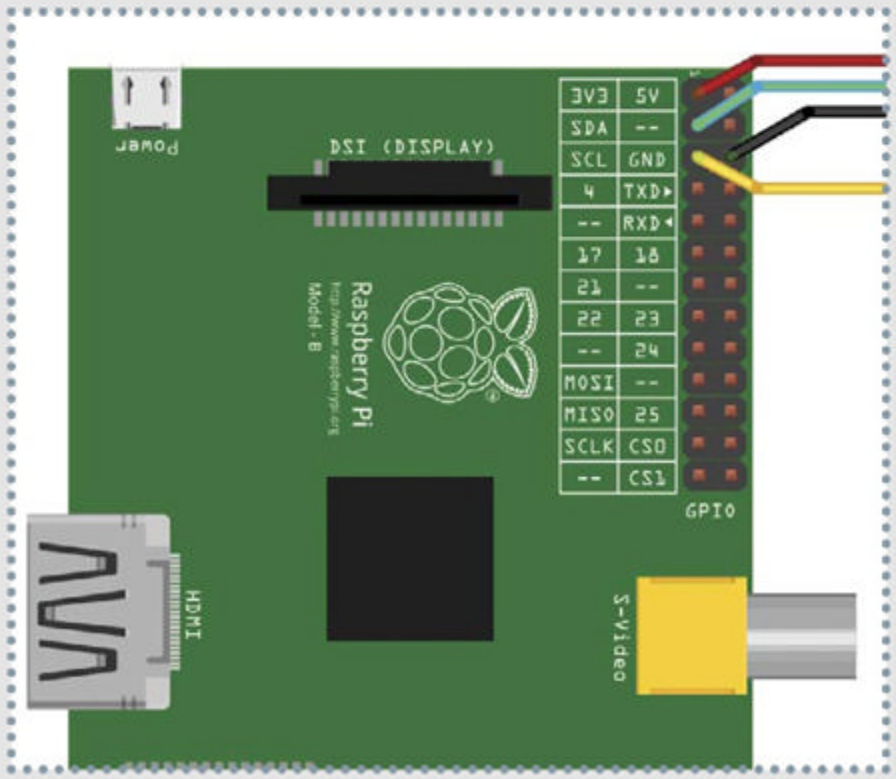
02 Connect the hardware

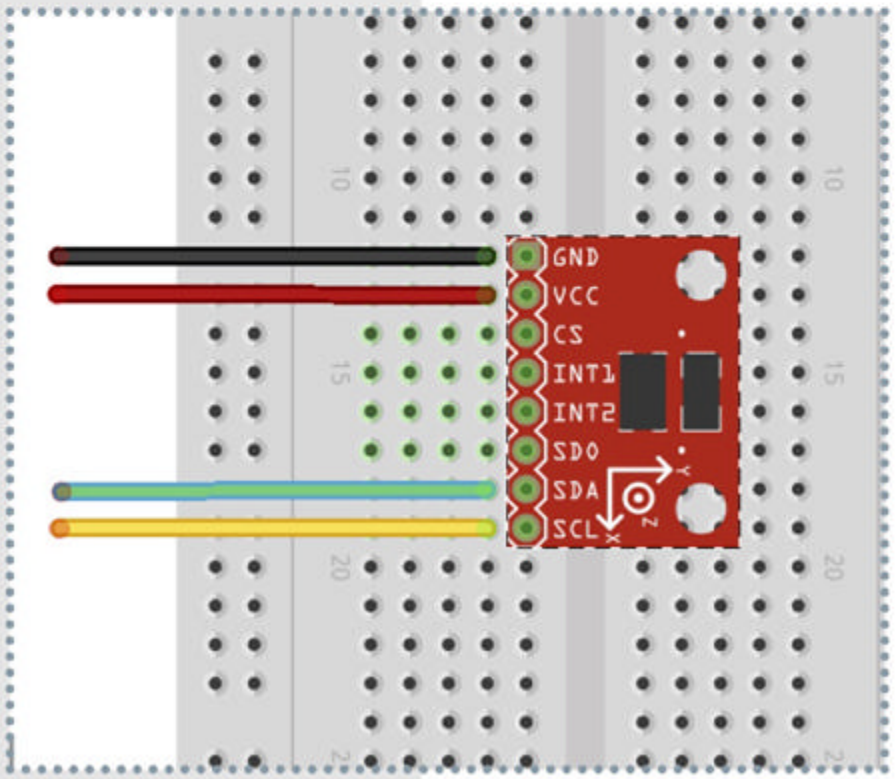
Since we're using I2C with Python, we'll need to install the smbus Python library. Type

```
sudo apt-get install python-smbus
```

in the terminal. Then wire up the accelerometer to the Pi with male-to-female prototyping cables. The correct pins are marked on the ADXL345, but check our pictures for where to plug things into the Pi.

Below The ADXL345 accelerometer only requires four cables – two for power and ground and two for I2C communication with the RasPi





03 Final checks

With accelerometer attached to Pi, power it up, open a teminal window and type:

```
sudo i2cdetect -y 1
```

The '1' at the end assumes you're using a Rev 2 Pi – replace it with '0' if yours is older. You should see some numbers denoting that the ADXL345 has been recognised. Finally, we'll use Git to grab the project – type the following into the terminal:

```
sudo apt-get install git
```

04 Cloning and testing

From the terminal, navigate to your home folder (with cd ~) and create a project folder for the project to live in using:

```
mkdir adxl345_project
```

Cd into the folder and from the terminal, type the following to clone the finished project from our Github.com account

```
git clone https://github.com/russb78/adxl345_project.git
```

05 How it works

To run the project for the first time, from the adxl345_project folder type:

```
sudo python adxl_345_project.py
```

A Pygame window will open and a spaceship will appear on screen on a space backdrop. Pick up your breadboard 'controller' and tilt it around. The ship should move around the screen in reaction to your movements. If the movements are backwards in either direction, you can adjust the orientation of the breadboard – it might be 'upside down'!

"If the movements are backwards …adjust the orientation of the breadboard – it might be 'upside down'!"

06 Why it works

The code for this project is an excellent backbone for a *Space Invaders*-style game written in Python and Pygame. The code collects data from the ADXL345 accelerometer in the update_pos() function using the move_data variable. This data is piped to the player_position array – which is in the form of x and y co-ordinates. The move_data is added to the player_position co-ordinates to push the image of the spaceship around the screen. You can adjust the multiple (for us, 20 was a good number) to change the speed of the ship. We've also used a check_pos() function, which makes the ship 'wrap around' the screen. If it exits the screen on the left, it will reappear on the right.

"The code for this project is an excellent backbone for a Space Invaders-style game written in Python and Pygame"

Above The finished project uses free images from opengameart.org

The Code

WRITE THE PROTOTYPE CONTROLLER FOR YOUR SPACE GAME

```
import pygame
from adxl345 import ADXL345

adxl345 = ADXL345() # Initialise the accelerometer
pygame.init() # Initialise Pygame

# Create a screen of 800x600 resolution
screen = pygame.display.set_mode([800, 600])

# Name the game window
# Set the mouse visibility and start an FPS clock
pygame.display.set_caption('ADXL345 Space Test - Press ESC to quit')
pygame.mouse.set_visible(False)
clock = pygame.time.Clock()

# Load the images we're using from:
# http://opengameart.org/users/rawdanitsu
background_image = pygame.image.load("Space-Background-4_0.jpg").convert()
player_image = pygame.image.load("ship0.png").convert()

# We can use colour key method to remove the background from the ship
player_image.set_colorkey([0, 0, 0])

player_position = [450, 350] # Initial starting point of the ship
game_over = False # Global variable to decide if the game should end

def update_pos():
    """" Poll the adxl345 and update player pos based on readings"""
    move_data = adxl345.getAxes(True) # Returns a dict of axes results
    if move_data['x'] < -0.1 or move_data['x'] > 0.1:
        player_position[0] += move_data['x'] * 20
    if move_data['y'] < -0.1 or move_data['y'] > 0.1:
        player_position[1] += move_data['y'] * 20
```

The Code

WRITE THE PROTOTYPE CONTROLLER FOR YOUR SPACE GAME

```
def check_pos():
    """ Check player pos to make it wrap-around the game window"""
    if player_position[0] > 850:
        player_position[0] = -75
    elif player_position[0] < -75:
        player_position[0] = 850

    if player_position[1] > 650:
        player_position[1] = -75
    elif player_position[1] < -75:
        player_position[1] = 650

###### MAIN PROGRAM LOOP ######
while not game_over: # Handle control events while the game is in play
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            game_over = True # Quit if close button is pressed
            if event.type == pygame.KEYDOWN:
                if event.key == pygame.K_ESCAPE:
                game_over = True
# Quit if escape key is pressed

    update_pos() # Update the player's position
    check_pos() # Check the player's position

# Update the background then the player's position on the screen
    screen.blit(background_image, [0, 0])
    screen.blit(player_image, [player_position[0],player_position[1]])

    pygame.display.flip() # Refresh the screen
    clock.tick(20) # Force frame-rate to desired number

pygame.quit() # Game quits gracefully when 'game_over' turns True
```

3D full-body scanner

We talk to Richard Garsthagen about why he needs 40 Raspberry Pis to take a picture of you

How did you get started creating these practical projects and working with the Raspberry Pi?

"In the last five to six years I began working with 3D printers and CNC machines. I started to build stuff, such as furniture and gadgets, and my first Raspberry Pi project was the Pi Snap Box. It's the size of a mini-PC and is a box you put on the wall with one button on it. If you press the button, it takes three photos. It posts the first photo to a Facebook account for whoever the box belongs to. So for example, if you hang it up in a hairdresser's salon and get your hair done all nicely, people could then see the good results on the hairdresser's Facebook page.

This is good commercially to attract visitors, but we've had one of those boxes in my home for a year now, and

"I started to build stuff, such as furniture and gadgets, and my first Raspberry Pi project was the Pi Snap Box"

Richard Garsthagen has worked in IT his entire career and is self-taught in coding. He's been making practical projects for almost six years now.

If you like

Richard's Pi Snap Box project is a much simpler implementation of the Pi and Camera, and can be found over at **www.pisnapbox.com**

Further reading

To learn more about the 3D Scanner and where you might see it next, visit **www.pi3dscan.com**

my son has been posting pictures of himself to Facebook on his own for the past year. That was my first Pi camera project, and I really fell in love with the camera as it's easy to use and very low cost. That gave me the idea that if one was so cheap, I could scale it up to 40 at least."

What gave you the idea to create the Pi 3D scanner in the first place?

"I have to say that as I have kids in my life I have a lot of challenges I'd like to do – kids give you a lot of reasons to build projects. As for my 3D scanner, my youngest kid is two, and of course there are lots of methods out there to do 3D scanning. But pretty much all of them [require you] to sit still, and my two-year-old does not sit still, I can tell you that! So I really wanted them to be able to be actual 3D scans and then print my youngest son.

I was aware that Hollywood did this with multiple cameras, but it is very, very expensive to do – typically they use 80-100 cameras, costing about $1,500 plus all the other gear. I could buy a house for that!"

"I have to say that as I have kids in my life I have a lot of challenges I'd like to do – kids give you a lot of reasons to build projects"

Below left Sit back and relax, the scanner will take less than a second to get your best side

Below The image scan is rendered before being sent to be printed as a 3D model

What made you decide to use the Pi Camera over standard digital cameras? Surely they wouldn't be much more than a Pi and Camera board together?

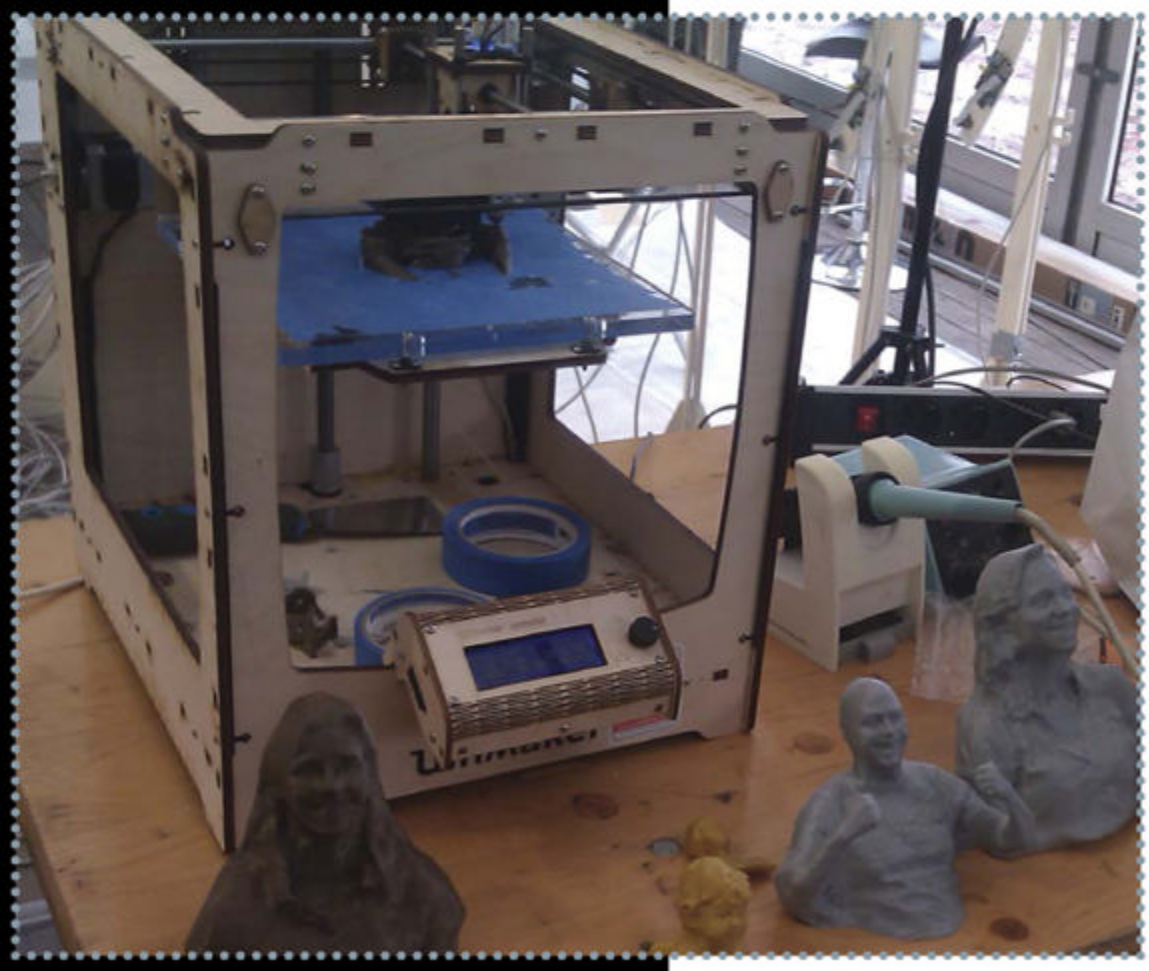
"I started researching that, thinking maybe I could buy a cheap digital camera, very cheap, with an eye at $40 to $50 for each camera. I probably could wire them up so that I could automatically synchronise them, but then I would still have the problem of all the images sitting on all the SD cards in each camera. With the Raspberry Pi, at the time I had only one, it had a camera and it was network connected so I could see its potential.

One thing I liked a lot about the Raspberry Pi camera was that I could set full shutter speed, exposure, ISO settings like a manual camera. Plus I could upload, fully automatically, all the images, so the whole process was completely automated. You hit the button, all the cameras take an image, they send a copy to a central file server where you can either render the 3D image yourself locally or send it to Autodesk's free cloud software to turn it into a 3D model."

How have you been able to use the scanner so far? How much time does it need to develop an image?

"We went to the Maker Faire in Gronigen where we allowed everyone to get a scan of themselves for free – and the system worked extremely well. Taking the images to create the 3D object takes a second – less than a second actually."

"I was aware Hollywood did 3D scans with multiple cameras, but it is very expensive – typically $1,500 each plus other gear. I could buy a house for that!"



Below 3D scans can then be sent to a 3D printer, to give you a physical record of your appearance

Raspberry Pi music streamer

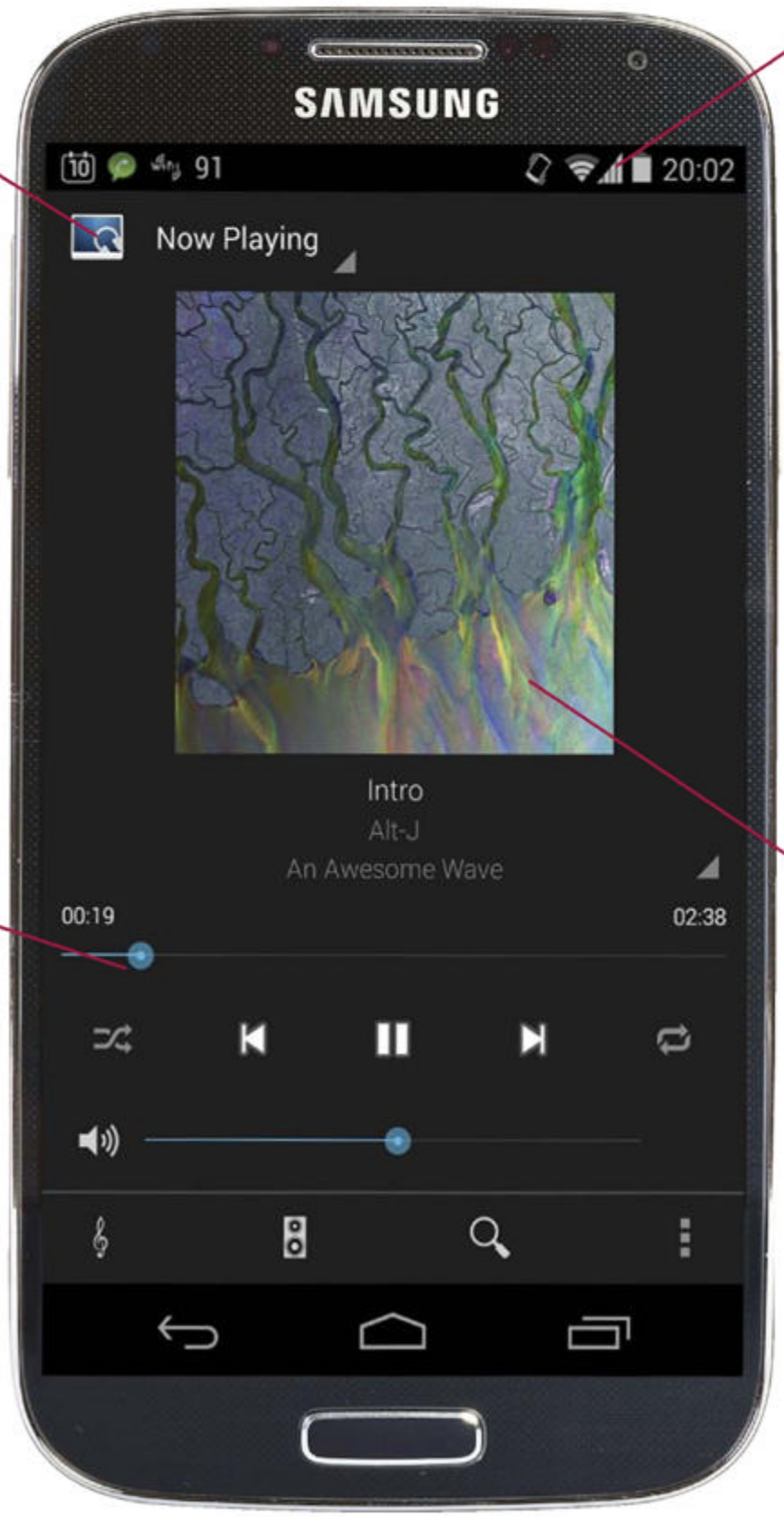
Remotely control a Raspberry Pi that plays your music collection and stream your music to your phone

Music collection

Your music collection must be stored in the directory /var/lib/mpd/ music on your Raspberry Pi. Once the daemon is set up, you can access it from any mobile device or computer

Streaming daemon

We can configure the Raspberry Pi Music Player Daemon to listen on all interfaces so we can access the music from all kinds of devices



Android & iOS

While there are lots of applications suitable for this project, we recommend MPDroid for Android and MPoD for Apple's iOS devices

Server connection

Once your Android or iOS app is set up, we can connect to our music server using the Raspberry Pi's IP address

Music Player Daemon (MPD) is a piece of software that acts as both a music player and a music server. It can output audio to any sound card connected to the system, and be controlled by an MPD client. Clients are available for almost any platform, including iOS and Android. MPD can also output audio to a stream, which can be used by most clients. This is great for people with large music libraries who can't fit it all on their device.

"Music Player Daemon can output audio to any sound card connected to the system, and be controlled by an MPD client – on almost any platform"

THE PROJECT ESSENTIALS

A router or switch on your network to plug your Raspberry Pi into
Latest Raspbian image www.raspberrypi.org/downloads
An iOS or Android device to control your music from
A Linux computer that can scp (secure copy protocol) music to the Pi and optionally act as a client

01 Install the required software

Log into the Raspbian system with the username `pi` and the password `raspberry`. First, find the IP address of the Pi using `ip addr | grep inet` and note it down for use later. Get the latest package lists using the command `sudo apt-get update`. Then install MPD using `sudo apt-get install mpd`. There may be some errors, but you should be able to ignore those.

02 Add music

The default music directory for mpd is `/var/lib/mpd/music`. We will first make this folder world-readable, writable and executable so that the Pi user can write to it. Do this with `sudo chmod 777 /var/lib/mpd/music`. Then find some music you'd like to copy on your Linux computer and use scp to copy it. For example: `scp -r Alt-J/ pi@192.168.157.28:/var/lib/mpd/music/`

03 Fix permissions

The files that we just copied will be owned by the Pi user, which isn't what we want. We're going to change the ownership of the music directory, and all subfiles/subdirectories, to the mpd user and the audio group: `sudo chown -R mpd:audo /var/lib/mpd/music`

04 Configure the daemon

We want to edit `/etc/mpd.conf` (using sudo). The first change is to make the daemon listen on all interfaces, so we can use MPD clients from other devices. Do this by changing the line:

```
bind_to_address		"localhost"
```

to...

```
bind_to_address		"any"
```

05 Configure a stream

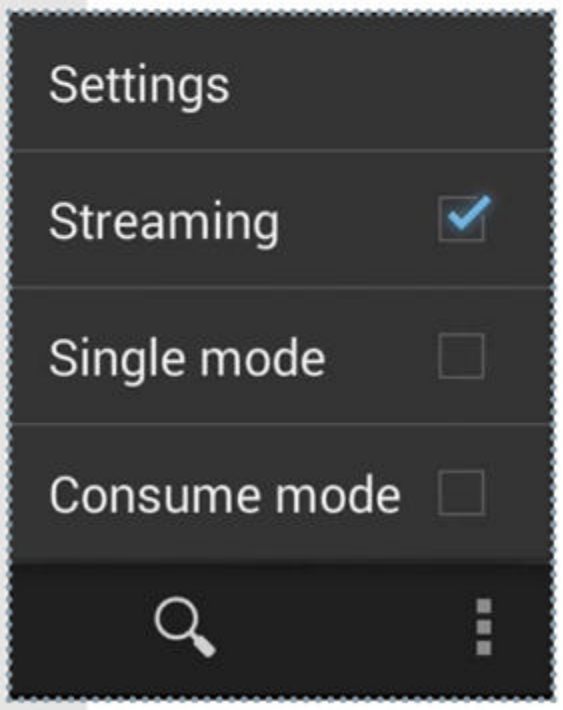
At the moment, the only audio output is the 3.5mm one on the Pi. To set up a stream, scroll down the config file until you find the httpd stream output that is commented out. Uncomment the entry, and change the format line to produce stereo output instead of mono. Our entry was as follows:

```
audio_output {
        type            "httpd"
        name            "My HTTP Stream"
        encoder         "vorbis"
        port            "8000"
        quality         "5.0"
#bitrate                "128"
        format          "44100:16:2"
}
```

Save the changes and restart the daemon with

```
sudo /etc/init.d/mpd restart
```

Below Stream your music library to a mobile device using a compatible client



06 Set up a client

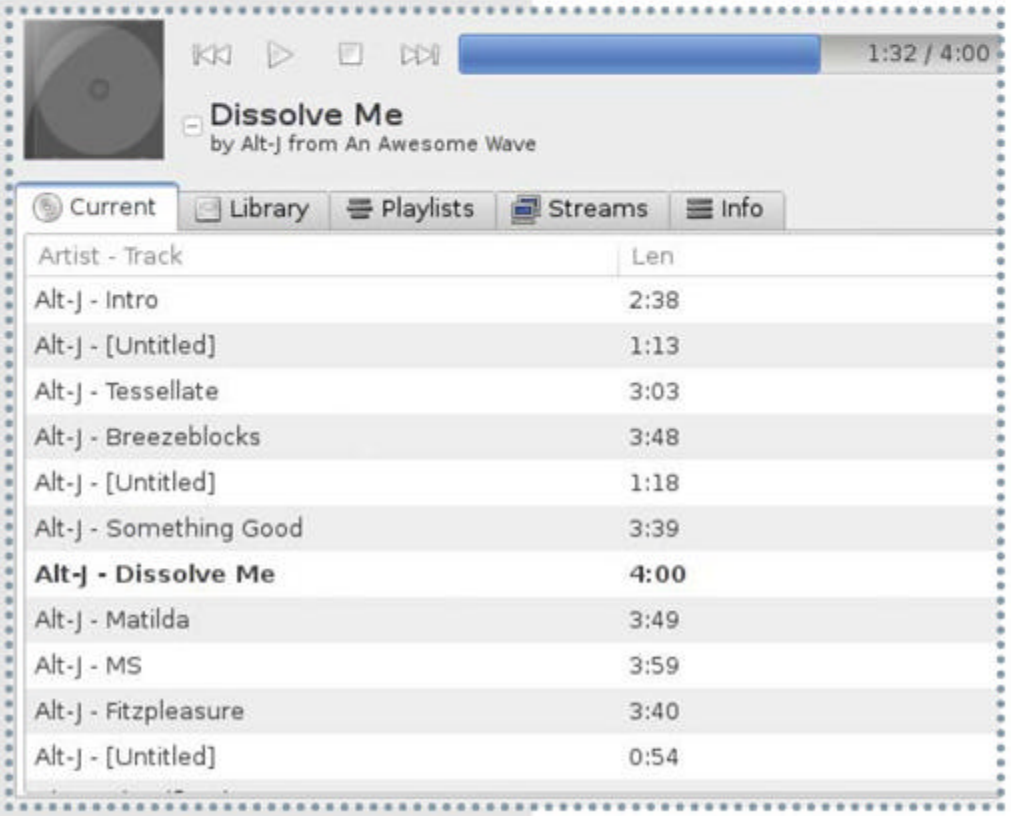
It's difficult to walk through setting up a client on each different platform, but the steps translate fairly easily. For Linux, we suggest Sonata, for Android we suggest MPDroid, and for iOS we suggest MPoD. We're going to set up MPDroid on Android, so go ahead and download that from the Play Store.

07 Connect to the server

Once in the MPDroid app, select WLAN-based connection and choose your access point. Then fill in the Host field with the IP address of your Pi and fill in the 'Streaming host' field with the same details. Everything else should be the default. Once you've done this, go to the Now Playing screen. We need to update the music library, as it has never been scanned before. To do this, press the Menu button,and go to Settings. Then select the Update option, with the caption 'Refresh MPD's Database'.

08 Playing music

Press the 'treble clef' button in the bottom-left corner to go to the music library. This will take you to the Artists section of the library. To play music from an artist, long-press on the artist and select 'Add, replace and play'. If you have speakers or headphones connected to the Pi, you should hear music coming out of them. Use the volume slider on the Now Playing screen to adjust the volume.



Below This is the Sonata client for Linux

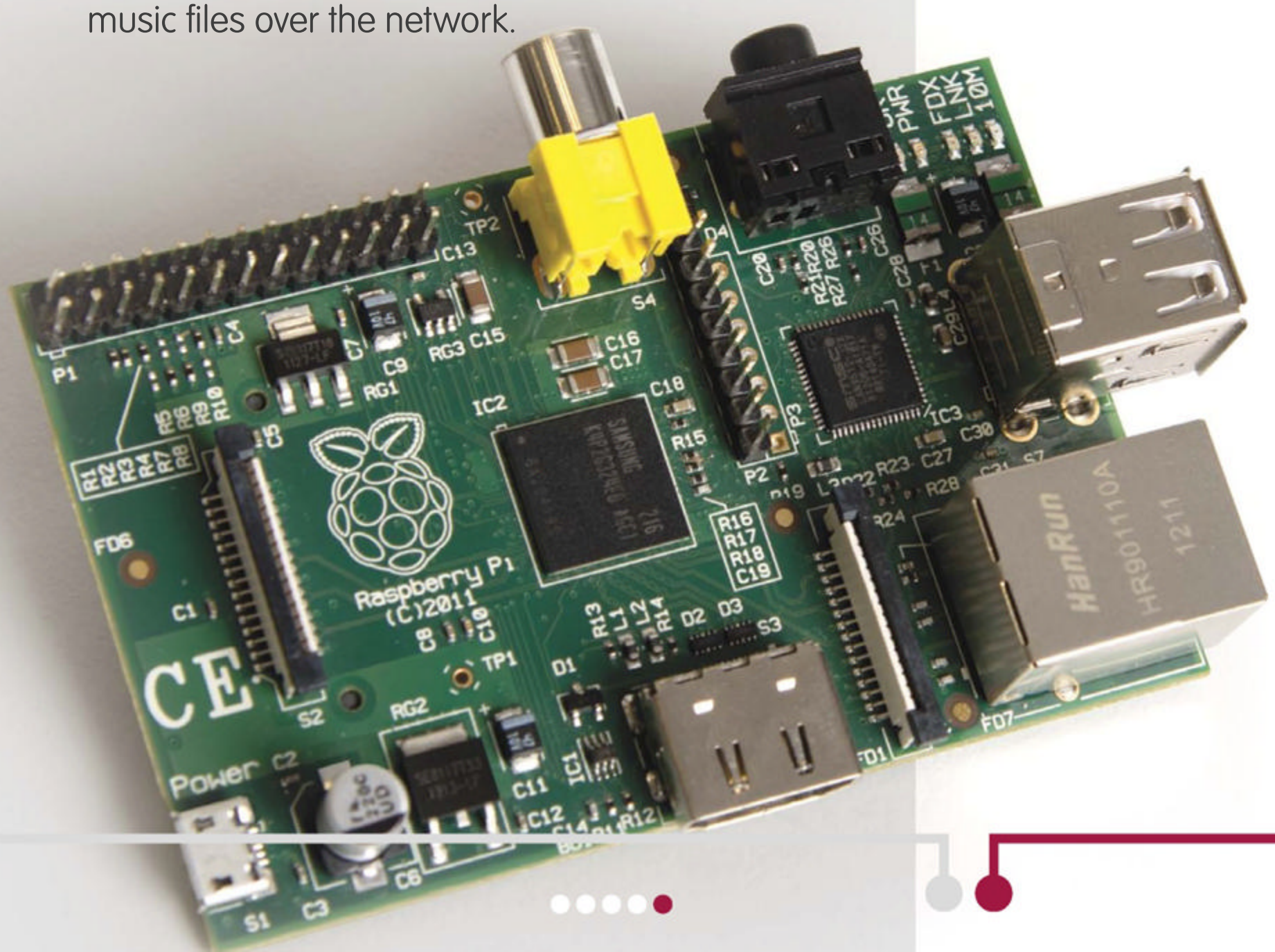
To enable the stream, press the Menu button and tick the Stream option. After about ten seconds of buffering, the sound will be coming out of your Android device. Although this is a long time to buffer, once you have a playlist the device will play it seamlessly. You may be able to reduce this buffer time by looking at the improvements section…

09 Further improvements

This article has illustrated a very simple MPD setup. Further possible improvements include:

- Putting music on an external hard drive so that you have more storage space;
- Tweaking the streaming settings to tax the Pi's CPU less (look at the 'encoder plugins' section of the user manual at **www.musicpd.org/doc/user**);
- Setting up a Samba share, to give access to the music files over the network.

> "After about ten seconds of buffering, the sound will be coming out of your Android device"



The object of Python

Python is built as an object-oriented language. But how do you use Python objects? Joey Bernard explains...

"All data in a Python program is represented by objects or by relationships between objects"

Welcome to the first of what will be a series of articles talking about what is great about Python. Even though all of us at **RasPi** are fans of the language, we're not so enamoured as to be blind to its weaknesses.

So over the coming months we will not only learn about how to do certain tasks in Python, but also what pitfalls you may run into during your travels. Once we have some Python basics under our collective belts, we will start looking at ways to apply this new-found knowledge to create some small programs for the Raspberry Pi.

The first subject we will look at is using objects in Python. According to the official documentation, objects are Python's abstraction for data. All data in a Python program is represented by objects or by relationships between objects. Actually, almost everything in Python is an object. So learning a bit more about how they behave is important if you intend to develop code of any real value.

More specifically, we will look at classes and how to work with objects of your own design.

Classes in Python support all of the most common concepts in object-oriented programming. The mechanisms of classes in Python were inspired by C++ and Modula-3. A class can inherit from multiple base classes and can override any methods in the base classes. All of the methods in the base classes are also available to be called from the inheriting class. As for data, objects can contain arbitrary amounts and types. And because Python is a dynamic language, all of this is modifiable at runtime. As an example, let's say that you want to develop a program that will do some kind of geometric processing. One of the core objects that you will want to use is something that represents a point in space. Looking at two-dimensional geometry first, you will need to store two values, an x and a y. In code, this would look like:

```
class Point:
    pass
```

This gives us a blank class called Point. The first line is how you define a new class. It is very similar to how you define a new function, except that you exchange the keyword 'def' with the keyword 'class'. A class always needs to have at least one statement. If you don't want your new class to do anything yet, you can use the keyword 'pass'. This essentially tells Python, 'there's nothing to see here just yet'. You can now use this class to create new objects

"All of the methods in the base classes are also available to be called from the inheriting class"

of type 'Point'. You can do this with:

```
my_point = Point()
```

This is not really all that useful yet. There is nowhere to store our point values. Or is there? One of the really cool things about Python objects is that they are fully dynamic. You can add x and y values with the statements:

```
my_point.x = 10.0
my_point.y = 20.0
```

This particular instance of the Point class now has an x and y value. That was easy! Unfortunately, we can't really use them effectively just yet. It would be better to have them as part of the definition of the class so that we can write methods that know how to use this data. It many cases, it also makes sense to initialise these variables. One way to do this is to simply write statements within the class definition directly:

```
class Point:
    x = 0.0
    y = 0.0
```

The other way to initialise values is to use the __init__ wwmethod in classes. This method gets automatically called when a new object gets instantiated, so every object you create will have those values by default:

```
class Point:
    def __init__(self):
        x = 0.0
        y = 0.0
```

You can then reset your values for x and y after creating a new instance with:

```
my_point = Point()
my_point.x = 10.0
my_point.y = 20.0
```

While this works well, one thing to remember is that programmers are inherently lazy and do not want to type more than absolutely necessary. Following this idea, it

"While this works well, one thing to remember is that programmers are inherently lazy and do not want to type more than absolutely necessary"

would be great if you could assign your x and y values at the same time you are creating a new object. In Python, you can. The __init__ function can be written to take parameters.

```
  class Point:
      def __init__(self, xvalue, yvalue):
          self.x = xvalue
          self.y = yvalue
```

Now when you want to create a 'Point' object with values you want to assign at its creation, all you need to do is type:

```
  my_point = Point(10.0, 20.0)
```

Isn't that much easier? Now that you have a point defined, you can add methods to work with this data. One of the first methods for a point is its absolute value. To add this to your class, you can just add the function definition to the class definition like this:

```
  class Point:
      def __init__(self, xvalue, yvalue):
          self.x = xvalue
          self.y = yvalue
      def abs_val(self):
          sqr_val = self.x**2 + self.y**2
          return sqr_val**0.5
```

Now that you have a basic two-dimensional point defined, you may want to create objects that can handle three-dimensional points. Ultimately, code reuse is something that should be aimed for – especially within the realms of object-orientated programming – and it's easy to do in Python. You can create a new class that builds on the existing code for two-dimensional points. This would look like:

```
  class ThreeDPoint(Point):
      def __init__(self, xvalue, yvalue,
zvalue):
```

"Ultimately, code reuse is something that should be aimed for"

```
        self.x = xvalue
        self.y = yvalue
        self.z = zvalue
```

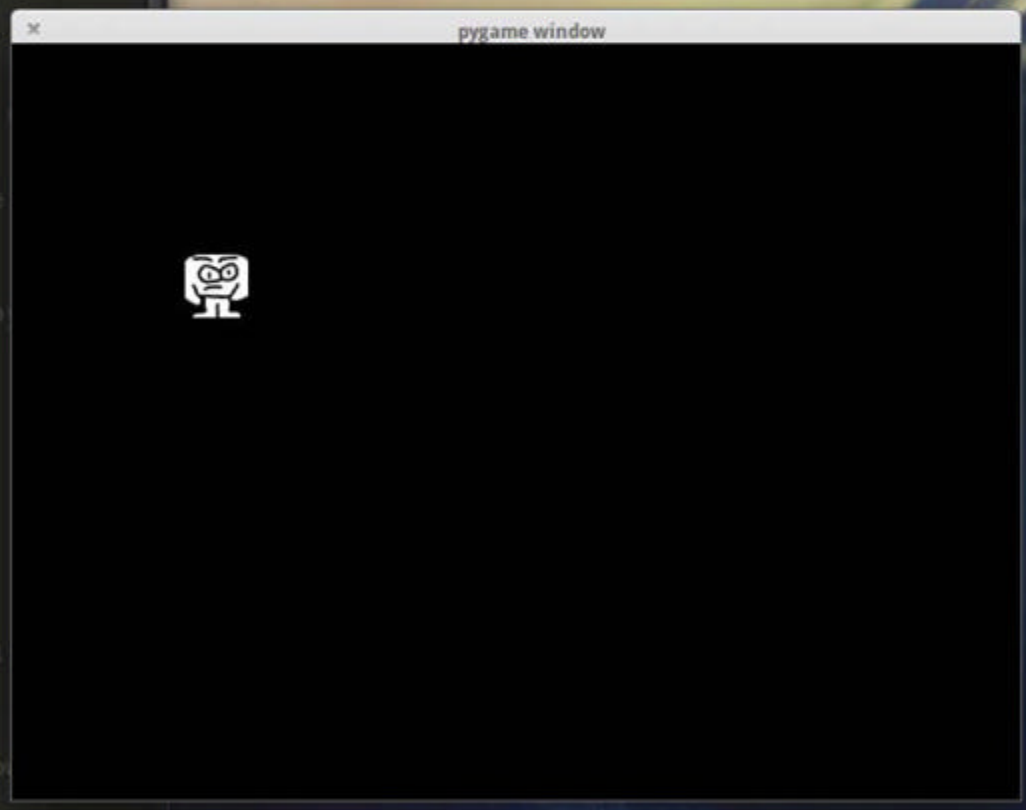
Written this way, the initialisation method has been overridden to take three parameters rather than two. But, nothing has been done to the method abs_val, inherited from the Point subclass. So when you call this from an instantiated ThreeDPoint object, you will get the length of the x and y parts of this point. If you want a proper three-dimensional absolute value, you will need to override the `abs_val` function.

Now that you've seen how to create your own objects, next issue we'll look at some details around how Python manages these objects in memory. On most desktop systems, memory is not really a concern. But the Raspberry Pi is relatively constrained, so knowing how memory is handled will become more important for your code.

> "But, nothing has been done to the method abs_val, inherited from the Point subclass"

Below This ball class could be the perfect backbone to a game. It's easy to replace your ball picture with anything you like, too

```
# You can find pygame in the package manager for a
# Debian-based users can install it with:
# sudo apt-get install python-pygame

import pygame
import sys

# initialise pygame with the following:
pygame.init()

# we're creating a clock to control the frame
clock = pygame.time.Clock()

# We're assigning some global variables, hense
BLACK = [0,0,0]
SIZE = WIDTH, HEIGHT = 640,480

# we need to create a pygame surface to displa
screen = pygame.display.set_mode(SIZE)

# Let's create our small class:
class My_ball:
    speed = [2,2]
    ball = pygame.image.load("ball.png")
    ballrect = ball.get_rect()

#Next we create an object from it:
ball = My_ball()

# This is our pygame 'loop'. While true, we'll
while 1:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            pygame.quit() # If we close the windo
            sys.exit()

    ball.ballrect = ball.ballrect.move(ball.speed)

    # This is our main 'game' logic.
```



The Code THE OBJECT OF PYTHON

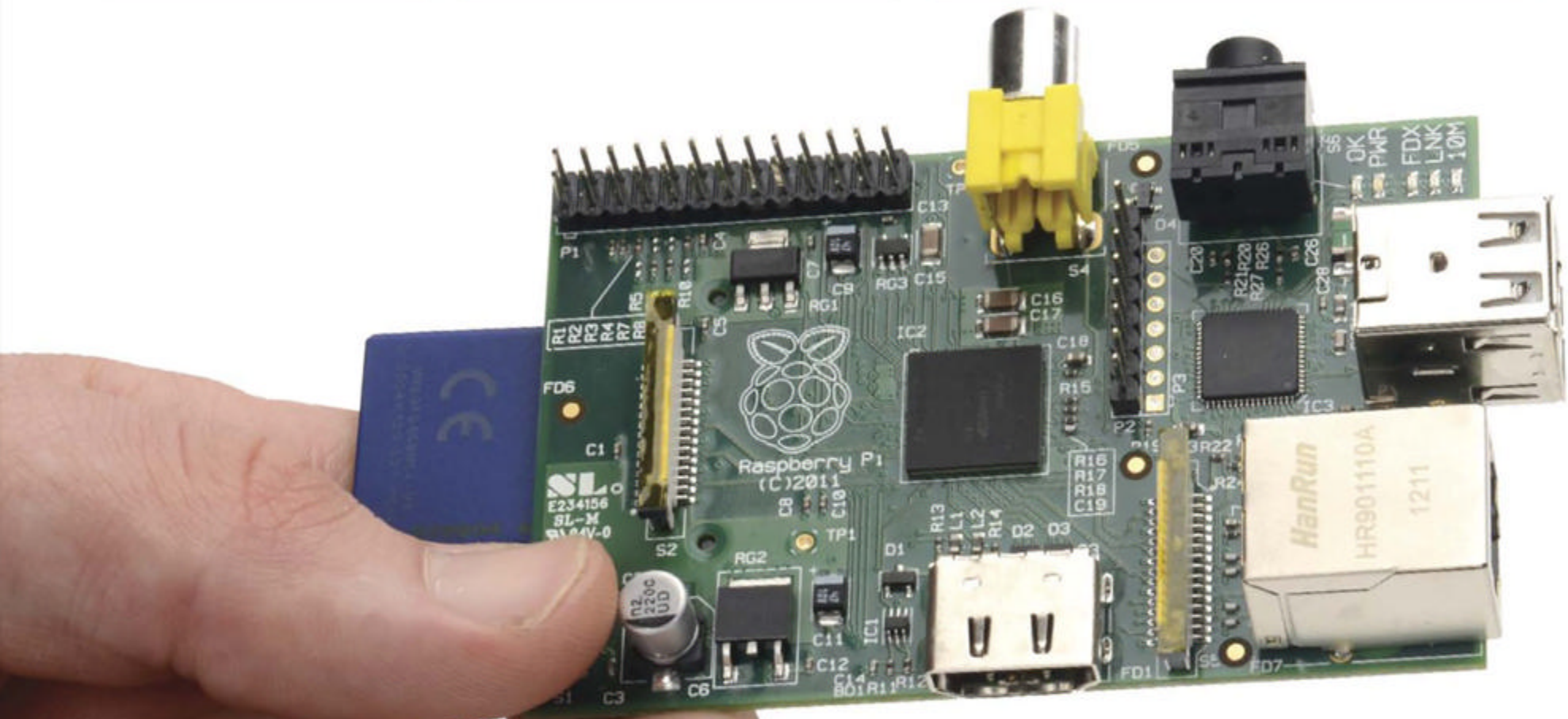
```
# If Pygame isn't installed, this code won't work for you.
# You can find pygame in the package manager for all major distros.
# Debian-based users can install it with:
# sudo apt-get install python-pygame

import pygame
import sys

# initialise pygame with the following:
pygame.init()

# we're creating a clock to control the frame rate:
clock = pygame.time.Clock()

# We're assigning some global variables, hence they are all caps.
BLACK = [0,0,0]
SIZE = WIDTH, HEIGHT = 640,480

# we need to create a pygame surface to display everything to:
screen = pygame.display.set_mode(SIZE)

# Let's create our small class:
class My_ball:
    speed = [2,2]
    ball = pygame.image.load("ball.png")
    ballrect = ball.get_rect()

#Next we create an object from it:
ball = My_ball()

# This is our pygame 'loop'. While true, we'll do the following:
while 1:
    for event in pygame.event.get():
```

```
        if event.type == pygame.QUIT:
            pygame.quit() # If we close the window, it quits gracefully
            sys.exit()

    ball.ballrect = ball.ballrect.move(ball.speed)

    # This is our main 'game' logic.

    # We're telling the ball to flip direction horizontally if it hits the
sides:
    if ball.ballrect.left < 0 or ball.ballrect.right > WIDTH:
        ball.speed[0] = -ball.speed[0]

    # and to flip vertical direction if it hits the top or bottom:
    if ball.ballrect.top < 0 or ball.ballrect.bottom > HEIGHT:
        ball.speed[1] = -ball.speed[1]

    # wiping the screen stops the old ball location from still showing
    screen.fill(BLACK)
    screen.blit(ball.ball, ball.ballrect) # draw our ball to the screen
    pygame.display.flip() # 'Flipping' the screen shows us the new ball location
    clock.tick(60) # We'll limit it to 60 frames a second
```

Talking Pi

Join the conversation at...

@linuxusermag | Linux User & Developer | @ RasPi@imagine-publishing.co.uk



The Raspberry Pi encourages a lot of hands-on work and this means it doesn't quite work like – or as easy as – your laser focus-tested smartphone interface, or even a normal computer. So we're answering your burning Raspberry Pi questions each issue – get in touch with us on Twitter, Facebook or by email.

I have an 8 Gb SD card that I use for Raspbian. Can I make it use up all the available space?
Bob via email

There are a couple of ways to add this extra space to Raspbian but the main way is actually built into Raspbian itself. Open up the LXTerminal and type into the interface:

```
$ sudo raspi-config
```

This is the original config screen you see when you start up Raspbian for the first time. Press Enter on the first option, which is entitled 'Expand Filesystem', and it will begin to extend Raspbian over the rest of the SD card. You can also do this when first installing Raspbian on the Pi but it will take a while to do either way.

What's the best way to get around the Raspberry Pi's lack of USB ports? Can you really not use a hub?
Jill Burton via Facebook

Running out of USB ports is something we've managed to do on occasion, especially when you want to add Wi-Fi to the Pi. You can add a USB hub to the Raspberry Pi, although it needs to be powered externally – ie from a plug socket or another computer. Alternatively, look for keyboards that include USB ports to stick mice into, so that your inputs only take up so much space.

Keep up with the latest Raspberry Pi news by following @LinuxUserMag on Twitter. Search for the hashtag #RasPiMag

LinuxUserMag: Finally back at base after a great day with the Cambridge @Raspberry_Pi community. A rather inspiring bunch! #camjam

RasPiTV: @LinuxUserMag #CamJam is a superb Jam. The hard part is deciding what to miss.

LinuxUserMag: "In schools the computers are locked down. It's not healthy. @Raspberry_Pi gives the ability to mess about, experiment and hack." #camjam

oneOfManyHelens: @LinuxUserMag Yes, but someone has to fix the computers afterwards if the kids can do what they like. They need a sandbox like @Raspberry_Pi

I like using Raspbian but is there much reason to use one of the other operating systems like Arch?
Jon Call via email

Usually, if you're asking that question then the answer is no. Arch is a more advanced form of Linux that doesn't come with conveniently installed software like Raspbian does. If you're making advanced projects it does mean it will run faster with only the software you need, however it will require a little more understanding of Linux before you dive in and start playing around.

Can I use my Raspberry Pi as a home theatre PC or media centre? Is there an easy way to do it?
Sam B. via Facebook

You certainly can and a lot of people do – there are even mounts for certain Raspberry Pi cases that attach to the back of TVs. The easiest and best way to do this is by using RaspBMC or OpenELEC; operating systems for the Raspberry Pi that use the XBMC software to create a fully-featured media interface that can easily stream videos and music over your network or from the internet with very little configuration or setup.
We'd personally recommend OpenELEC as it's designed by folks that actually develop XBMC, although RaspBMC is still excellent.

ferg_mc: @LinuxUserMag @Raspberry_Pi I learned more from trying to bypass the security in ICT lessons than I ever did from the actual class.

LinuxUserMag: We're currently mining BTC at 0.3 khash/s on a Raspberry Pi. This is why you don't cpu mine, kids

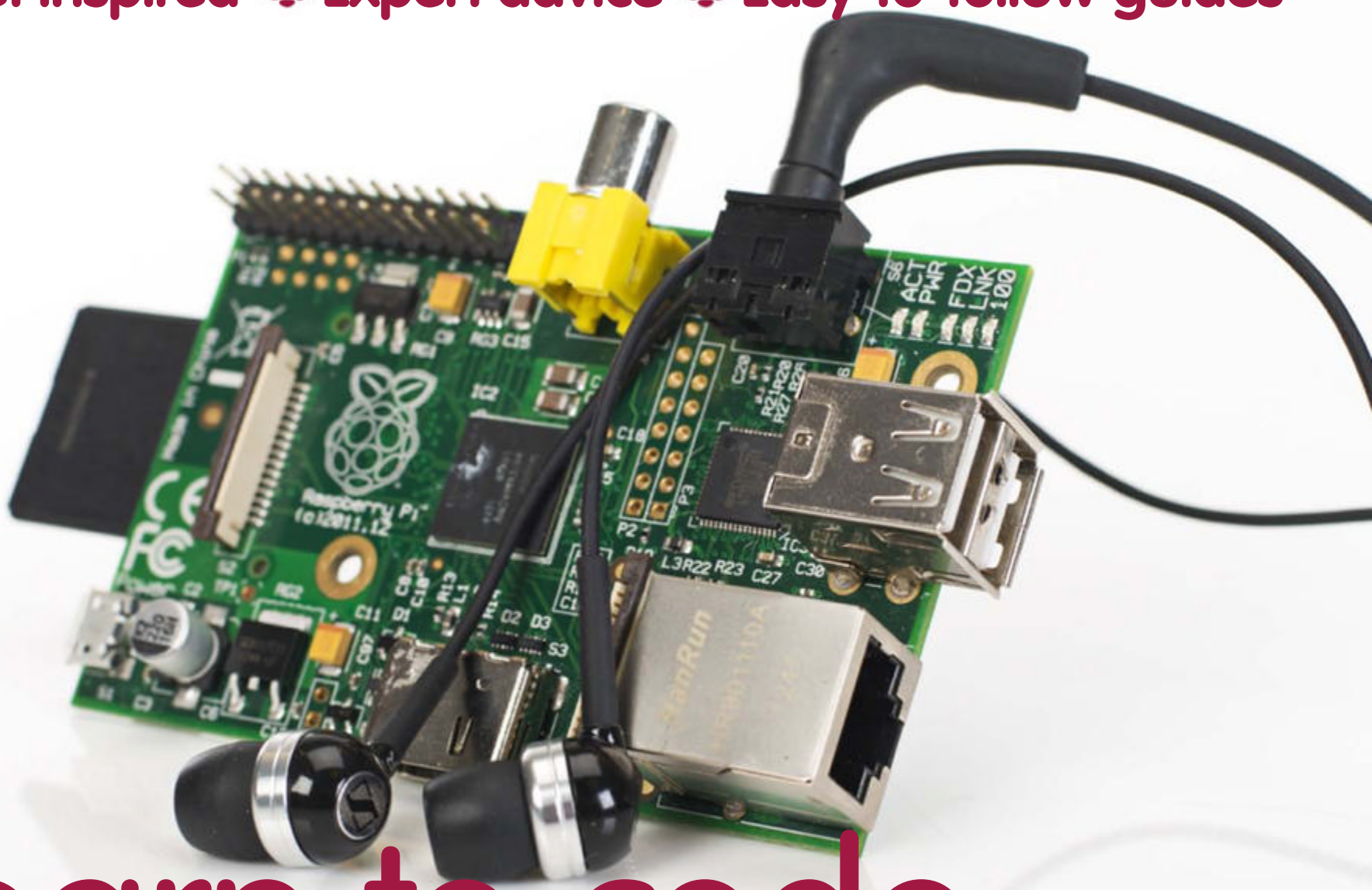
ghalfacree: @LinuxUserMag I've had great success with the AntMiner: 2GH/s for under £30. Works great on the Pi.

LinuxUserMag: @ghalfacree yeah this is mostly for fun. We've not overclocked the Pi and the memory split is all out of whack as well

i_renton: Amazing day at @LinuxUserMag #RaspberryJam! Loads of kids writing their first code, Raspberry Tank was declared "infinite fun"!

Next issue

Get inspired · Expert advice · Easy-to-follow guides



Learn to code with Sonic Pi

Plus Take photos with your Pi, code Scratch games and more

Get this issue's source code at:
www.linuxuser.co.uk/raspicode