

# Building Radar Speed Camera and Traffic Logger

with a Raspberry Pi

Published on March 31, 2015  internet of things

(<http://blog.durablescope.com/tags/internet-of-things/>) · polestar

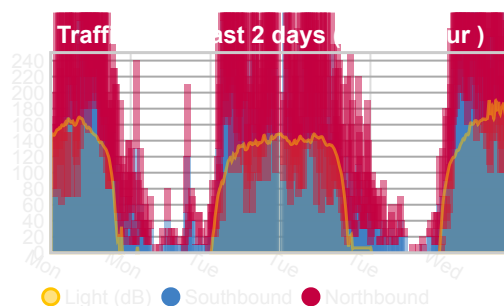
(<http://blog.durablescope.com/tags/polestar/>) · sensors

(<http://blog.durablescope.com/tags/sensors/>)

There is a **major update to this article**, please see this later post Radar Speed Camera Rebooted (</post/TrafficRadarRevisited/>).

Live Traffic (mph)

22.1 ↓



I've always been interested in connecting things up to computers that they were never designed for. I could reminisce about my first drawing capture arm that used captured the two joint angles with potentiometers and used them to control the pulse width oscillators that fed into the cassette input on a ZX Spectrum. (I've just done a google search for this idea and came up blank - I'm

sure there used to be commercial products – it looks like this technology has been lost in the digital dark ages.) Over the last couple of decades it's been hard going as most business focused and mass produced hardware has little hackability. With the advent of the Raspberry Pi and the looming Internet of Things it's now really easy to have a lot of fun. Partly it's about control but it's also about curiosity and being able to see and understand things that are not normally visible either because we don't have the physical senses or the timescales are too short or long.

In this post I want to describe my latest “experiment” with a doppler radar. I've played around with PIR sensors and light sensors in the past and both of these have their limitations as motion sensors. Radars are becoming quite popular for automotive applications from parking sensors all the way to autonomous vehicles so I decided to see what I could get working.

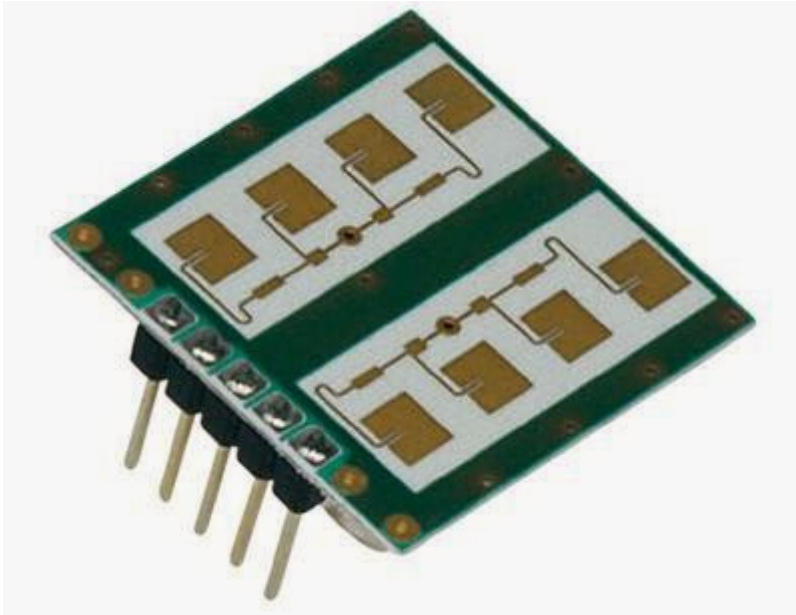


*High level system architecture*

## Hardware - Radar Module

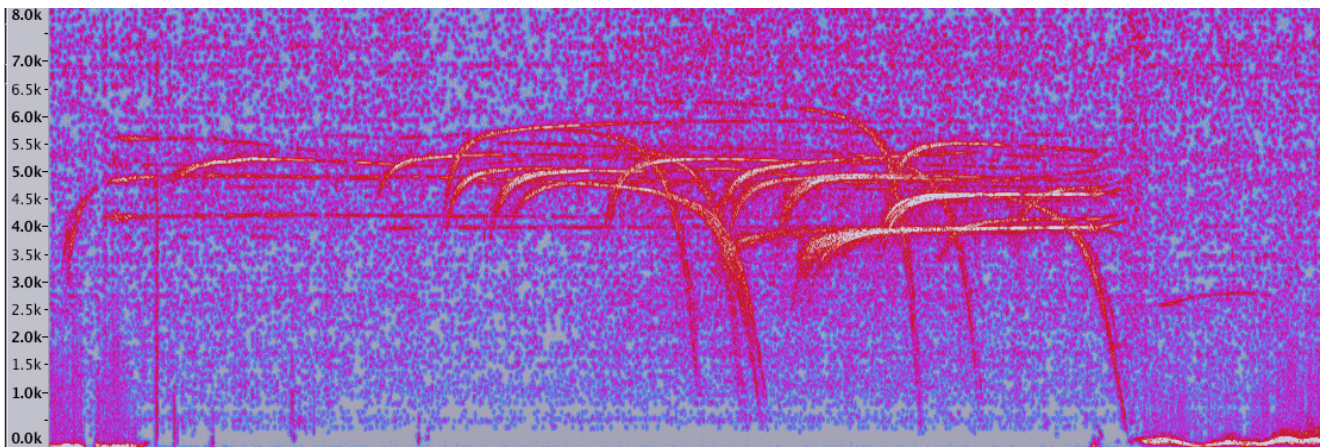
The radar technology I found was a 24Ghz stereo doppler radar module (<http://www.rapidonline.com/electronic-components/24-24-250ghz-stereo-radar-sensor-module-rsm-2650-50-7299>). Cheaper 10Ghz modules are available on ebay and it appears that most automotive radars use 77Ghz. It appears to be the case that higher frequencies give better range. The module I have used has the limitation that it doesn't measure distance to a target – all it can detect is movements towards or away from the sensor. This is achieved due to the doppler effect which causes the echo from a moving object to be of higher or lower frequency than the original frequency depending upon if it's moving towards or away respectively. Because the 24Ghz signal is way too high to sample and process directly the sensor module works by subtracting the transmitted signal from the received echo. This causes beating ([http://en.wikipedia.org/wiki/Beat\\_%28acoustics%29](http://en.wikipedia.org/wiki/Beat_%28acoustics%29)) and the result is a much lower frequency which is proportional to the speed of a target. The reason the sensor is called stereo is because it has two outputs. These two

outputs don't detect in different directions but instead are used to detect if a target is moving towards or away from the radar. This is done by detecting the phase of the received echo with one output receiving the real and the other the complex part. A target moving towards the sensor leads to a positive phase shift and away a negative phase shift. The output frequencies are in the range of human hearing with each 1mph roughly 72Hz (or 44Hz per 1kmh)



*B+B Sensors RSM2650 24GHz Stereo Radar Sensor Module*

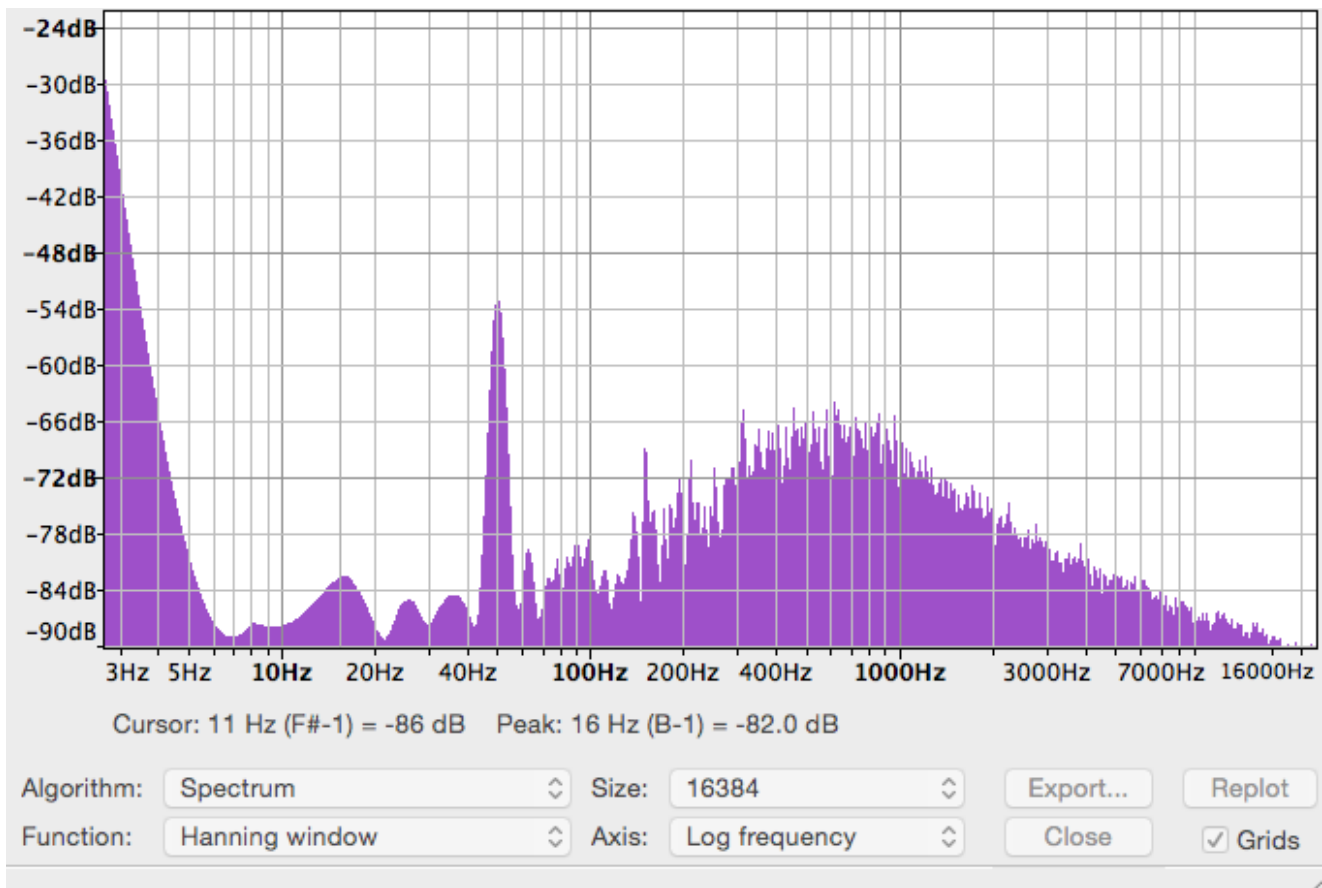
Multiple objects moving at different speeds are independently detectable because of their different frequencies. The following spectrogram (Audacity is great) shows a recording of the response pointing at a fairly busy motorway. You can also see how the relative speed and hence frequency varies as the vehicles move away from the sensor - this is known as the cosine effect (<http://copradar.com/preview/chapt2/ch2d1.html>). You can hear what this sounds like from this audio sample on soundcloud (<https://soundcloud.com/concession/doppler-radar-from-traffic-on-motorway>).



*Spectrogram of radar output pointing at motorway*

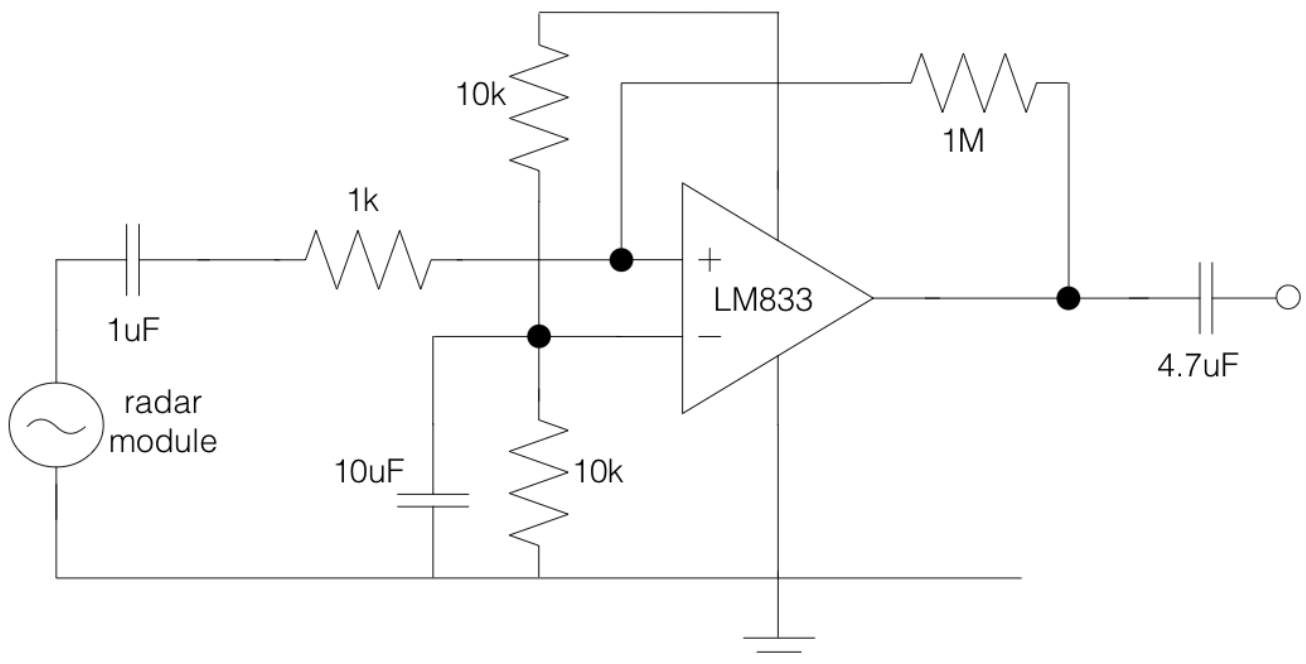
## Hardware - Signal Amplifier

Unfortunately the output from the radar module is really feeble. After experimenting with a number of approaches I settled on using an operational amplifier circuit fed into a the line-in on a sound card. It seems that the choice of operational amplifier is not that critical. I tried three (LM833n, NE5532a and LM358) and few different circuit designs but the signal to noise ratio was pretty much constant. High frequency noise from the radar module and it's sensitivity to picking up electromagnetic hum 50Hz/60Hz seem to be the main issues. Here is a plot of the noise spectrum when no movement is present:



*Noise Spectrum from output of amplifier*

This is the circuit I ended up using, obviously times two for the two outputs from the radar:



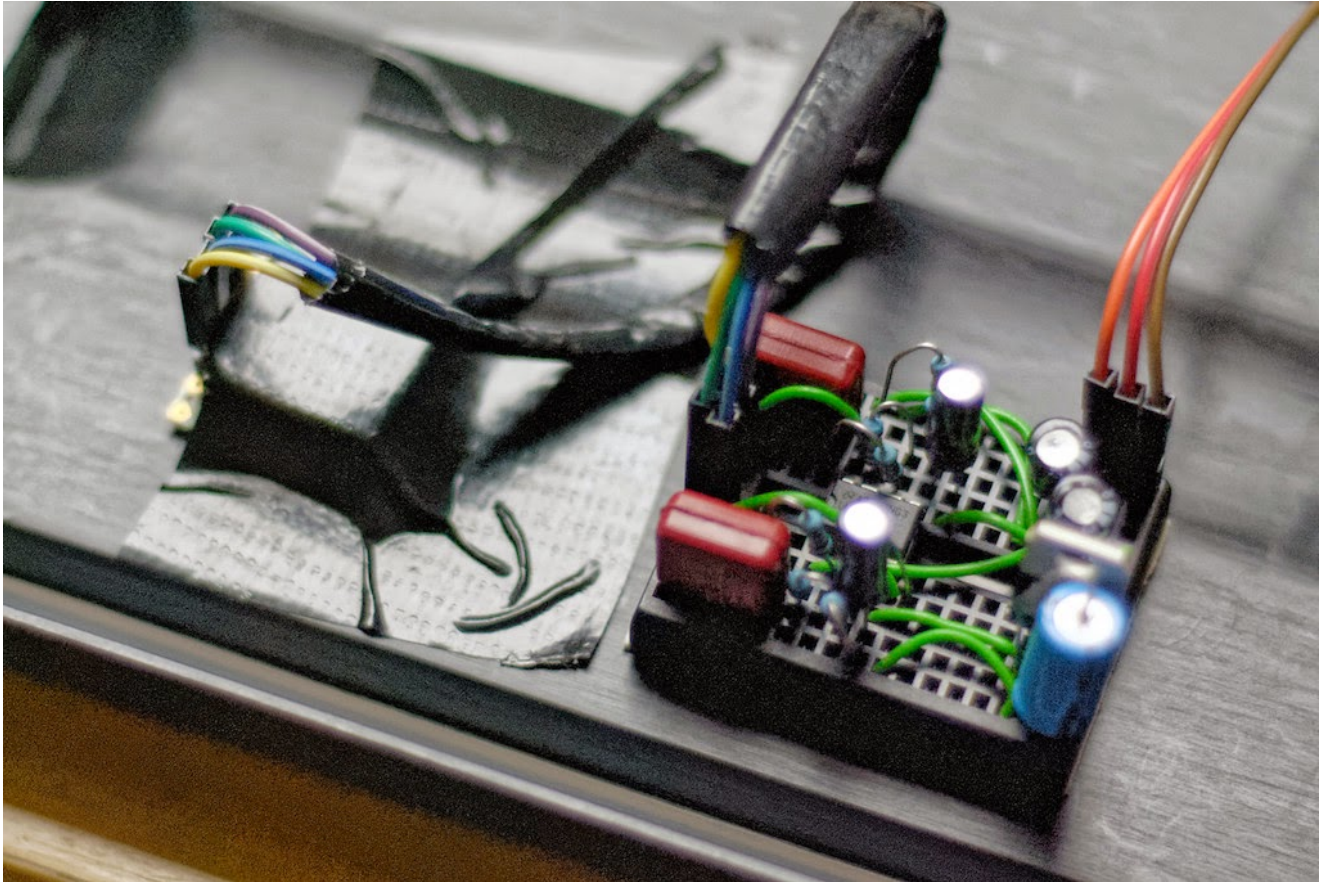
*Amplifier circuit diagram*

I powered this from a separate regulated 5v supply because the noise on the USB 5v power is horrendous.



Update: Peter has been experimenting with a pre build audio amplifier module based on the LM386 (<http://www.ebay.co.uk/itm/141507760671>) and has had success eliminating the 50Hz hum)

This is what it looks like assembled with radar module connected:



*Amplifier circuit diagram*

## Hardware - Soundcard

Because the signals are in the audible range use of the sound card solves a lot of problems such as sampling at high rate, avoiding sampling jitter and buffering the samples. Sound cards are also pretty cheap. I got this one (<http://www.ebay.co.uk/itm/USB-External-6-Channel-5-1-S-PDIF-Optical-Sound-Card-Audio-For-Netbook-PC-Laptop-/321511484257>) from eBay for £10 It looks like overkill but it's hard to get one with a proper line-in rather than just a microphone.



*USB Soundcard with LineIn*

## Hardware - Raspberry Pi

I used a standard Raspberry Pi module B. The two USB ports are just enough to attach the sound card and a WiFi dongle. I installed it with the latest Raspbian image and set up the WiFi and SSH access.

## Hardware - Enclosure

After experimenting with circuit I wanted to mount it outside near the road - the closer the better in terms of signal to noise. For the strength of reflection from vehicles it looks like a range of about 50m is the absolute maximum. Luckily I have power from a nearby circuit for a pond pump. After problems with hum pickup and also noise from the WiFi signal I placed the sensitive radar module and amplifier separate from the Raspberry Pi and power supplies. I found these great DriBox (<http://dri-box.com/>)es work really well. Here is what it looks like installed near the road.





*Radar installed in garden*

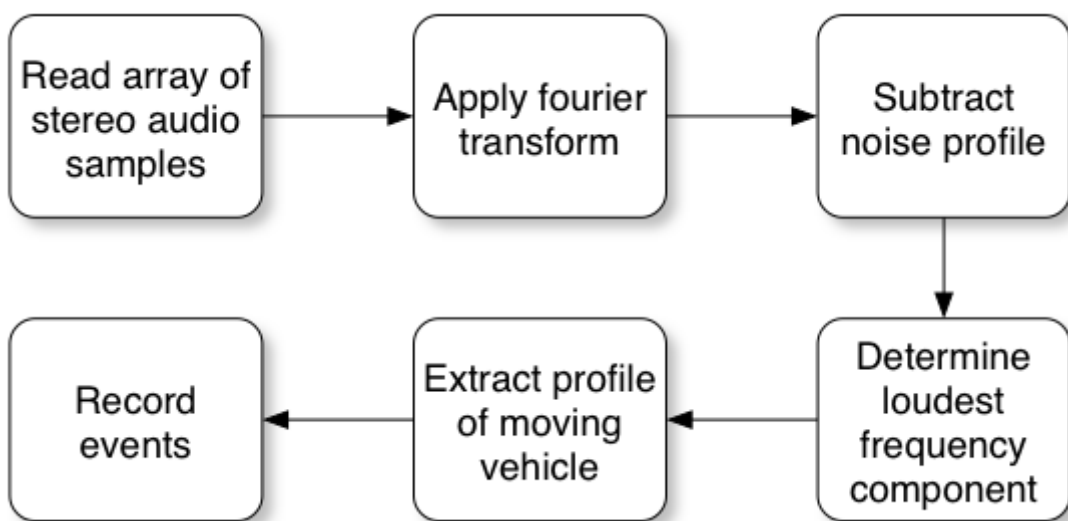
## Software



On the Raspberry Pi I have a single Python program which performs all the processing to capture the signal from the sound card through to emitting events when vehicles are detected.

The code has dependencies on python-alsaaudio, python-numpy and optionally python-httpplib2 for posting events to a HTTP rest web service.

Here is a block diagram of the processing performed. The code is available on this GitHub repository (<https://github.com/tonbut/rpi-traffic-radar>).

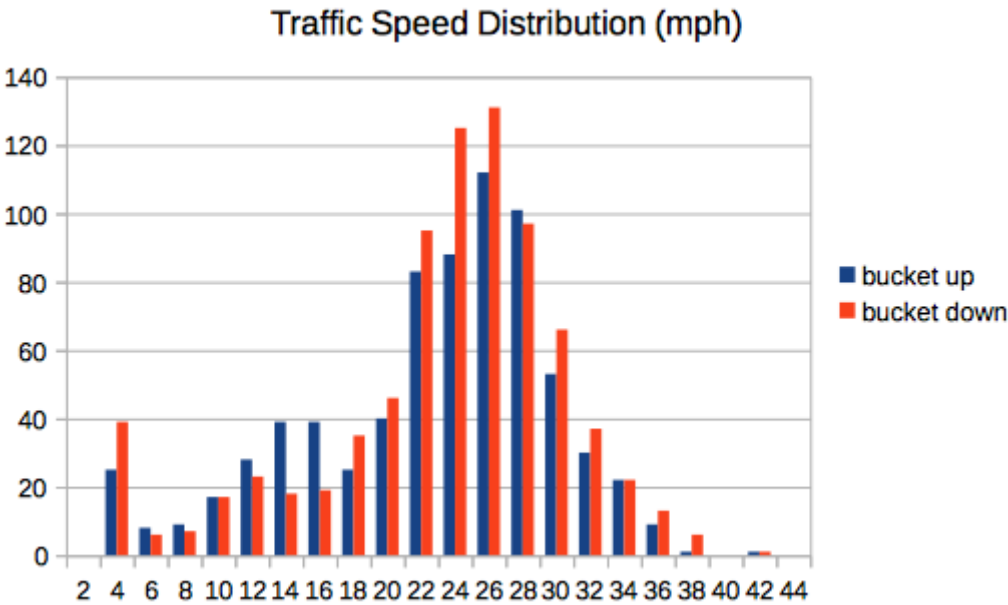


*Software block diagram*

## Results

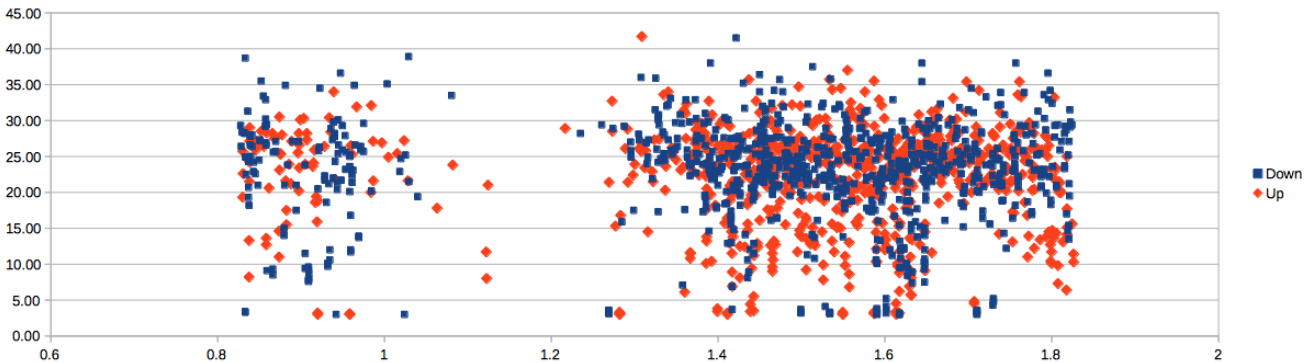
It's early days in terms of analysing the data. My initial data has been simply captured in a csv file and then analysed with a spreadsheet. I've also started send the cumulative flow numbers into polestar (<https://polestar.io/>). On there you can see a live feed.

Here is the speed distribution of the first 24 hours of captured data:



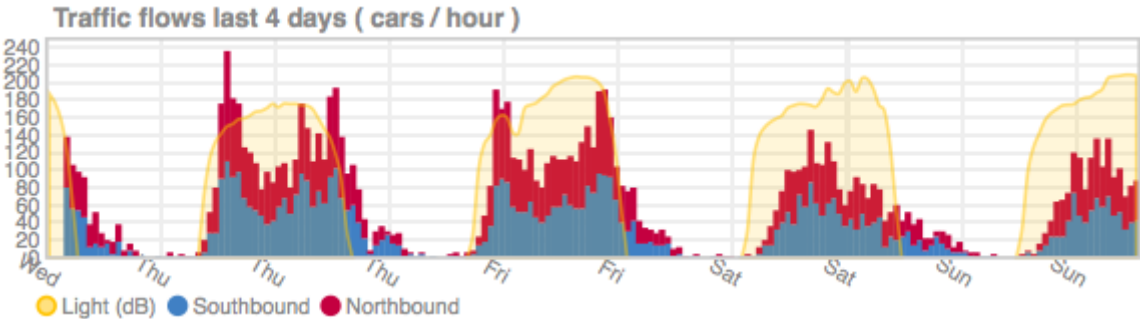
Distribution of speeds over 24 hours

Here is a scatter graph of traffic speed vs time for the first 24 hours of data:



Distribution of speeds over 24 hours

Here is a time series bar chart for the first 4 days of traffic flow:



Time series bar chart with 4 days of traffic flow




Read next

(<http://blog.durablescope.com/post/HomeMonitorUpdate/>)

## Home Monitor Update

It has been about three months since I first got the NetKernel based home monitor system live. This post is an update about how it has performed and how it has evolved. Issues The major issues over this period have been hardware related. Loose connections where plentiful until I decided to commit the circuitry to a PCB, place it in a box and attach proper connectors. The rain sensor has been a real problem. (<http://blog.durablescope.com/post/HomeMonitorUpdate/>)

---

 Published Feb 11, 2005  3 min read  internet of things  
(<http://blog.durablescope.com/tags/internet-of-things/>) · sensors  
(<http://blog.durablescope.com/tags/sensors/>)

(<http://blog.durablescope.com/post/NetKernelBasedHomeMonitoringSystem/>)

## NetKernel based Home Monitoring System

For a bit of fun I put together a low budget home monitoring system using an old PC, a few wires, sensors and 1060 NetKernel. The aim was to create a system to remotely monitor my home and its ambient weather conditions. I also wanted the data to be stored so that historical graphs could be generated. The Hardware The host PC was an old Gateway Solo Laptop with a Pentium 166MHz MMX, 64MB RAM running Win98SE with a USB network adaptor and with an in-built joystick port. (<http://blog.durablescope.com/post/NetKernelBasedHomeMonitoringSystem/>)

---

 Published Nov 25, 2004  4 min read  internet of things  
(<http://blog.durablescope.com/tags/internet-of-things/>) · sensors  
(<http://blog.durablescope.com/tags/sensors/>)

(<http://blog.durablescope.com/post/HomeMonitorReloaded/>)

## Home Monitor Reloaded

I can't believe that the last time I mentioned my home monitor project was exactly six years ago! For those who haven't heard of it before the home monitor project is a home automation system running



in my house which attempts to capture as much useful information and control as much as is possible with the absolute minimum of expensive equipment. It is also acting as a bit of a playground for doing fun things with NetKernel. (<http://blog.durablescope.com/post/HomeMonitorReloaded/>)

📅 Published Feb 10, 2011 🕒 4 min read 🔖 internet of things  
(<http://blog.durablescope.com/tags/internet-of-things/>)

[Comments](#)[Community](#)[1 Login](#) ▾[❤ Recommend](#) 5[🐦 Tweet](#)[f Share](#)[Sort by Best](#) ▾

LOG IN WITH

OR SIGN UP WITH DISQUS [?](#)**Emlyn** • 2 years ago

Hi Tony,

Thanks for sharing this great project, I have tried to give this a go unfortunately as a complete noob I'm going around in circles and am not sure how to get it running.

I've gathered the components and built the circuit as described, I've installed a clean install of raspbian Stretch and I think I've configured the usb sound card as the default input device.

I've download the [Radar.py](#) code and tried running it under python 2 and python 3 thats included in raspbian however I'm not sure its working. Does the code generate a .csv file as vehicles pass or how is the information recorded?

Do i need to install polestar and/or netKernal on the Pi?

Regards,

Emlyn

1 ^ | ▾ • Reply • Share ›

**Tony Butterfield** Mod ➔ Emlyn • 2 years ago

Hi Emlyn,

there are two aspects to why it may not be working. But firstly no you don't need polestar or any additional dependencies. I'm assuming the code is running without errors as you haven't mentioned any so that excludes the software side not having the right stuff. So things to check on hardware are:

- 1) make sure you have the line-in correctly selected and gain turned up in alsamixer. Test this by using a

program that can record sounds to see this working.  
2) test the amplifier and radar module by plugging in headphones into the amplifier output rather than into the soundcard. If you do this you should hear low frequency sweeping sounds if you move your hand around in front of the sensor.

Give these a try and let me know how you get on.

^ | v • Reply • Share ›



**Emlyn** → Tony Butterfield • 2 years ago

Hi Tony,

Thanks for the tips, I managed to confirm the hardware side of things by connecting the outputs of the pre-build audio amplifiers direct to a set of earphones and heard the tone change as I moved my hand in front of the detector.

Next I tried echoing the input direct to the output as described in an earlier post using:

```
arecord -c 2 -f S16_LE -r 44100 | aplay
```

I quickly realised I hadn't my pi (running Stretch) configured correctly.

After a fair bit of searching I found a helpful guide on the Google Developers site:

<https://developers.google.c...>

I then was able to input the audio and echo it to the output proving everything was setup correctly at last.

Next like Jon in an earlier post I ran into trouble with Line 132 of [radar.py](#). I also had to set the period parameter to 2000 as I kept getting:

"alsaaudio.ALSAudioError: Capture data too large. Try decreasing period size" error

Next I tried to configure Polestar and NetKernal but I am struggling badly any help would be gratefully appreciated.

Regards, Emlyn

^ | v • Reply • Share ›



**Luca Toldo** • 2 months ago

Hi Tony,

I'm trying to reproduce your project, and so far the

hardware works however I get problems in getting the sound into the raspberry.

I've posted that in

<https://www.raspberrypi.org/forums/viewtopic.php?t=233817> since on this board was always removed.

Basically, arecord does not give error but the output is completely silent, inspite of amixer being properly setup...

^ | v • Reply • Share ›



**Rende** • 2 years ago

Hi Tony,

Thanks for your explanation.

Regards, Rende

^ | v • Reply • Share ›



**Rende** • 2 years ago

Hi Tony,

I saw your project a week ago. It is exactly what I was looking for. So, I ordered all components and have built the pre-amplifier but the radar sensor I ordered was broken. Next week I will receive a new one. In the meantime I have all software parts running including the audio card. But I have a small question regarding the output of the file [example.py](#). Can you help me and explain what the column "count" is showing?

It is a great project you created Tony.

Thanks

Regards, Rende

^ | v • Reply • Share ›



**Joakim A** → Rende • 2 years ago

Hi Rende! How did you know your module was broken? I am trying to set this up now and I have connected the line in to my computer (Since I did not get anything useful from the usb sound card I got). One channel is almost totally quiet and the other seems to flip flop between 1, 0 and -1. Does this make sense? Tony do you have any suggestions? Sadly I do not own a oscilloscope.

I have had all parts home for months just waiting for me to get the time. I am really eager to get it running :)

Thanks!

^ | v • Reply • Share ›



   **Rende**  Joakim A • 2 years ago

Hi Jocke and Tony,

After ordering a new sensor I discovered that my sensor wasn't broken.

My pre-amp wasn't working correctly on 5V (strange).

I lowered the voltage to 3.8 Volt and the pre-amps were working but the gain was much too low.

So, I bought four pre-fabricated pre-amps and mounted two pre-amps behind each other. Two for each channel of course.

Finally the gain is OK and my speed detector works great but still I get too much noise from my raspberry pi zero w.

Now I have ordered a ground loop isolator as Tony, to get rid off it.

I have powered both (analog/digital) parts with two separate 5V power pack's.

The software works fine for me and I have expanded it with a MySQL module to store every 100 tracked speeds on one of my web servers.

It is a great project with a lot of fun and a very handy tool to track the traffic in front of your home :).

Regards, Rende

   [Show more replies](#)**Tony Butterfield** Mod  Rende • 2 years ago

Hi Rende, the count column is the number of samples captured that were matched to that vehicle. So it's proportional to the length of time that the vehicle was tracked for. A smaller or faster moving vehicle would expect to give a smaller count. This value probably isn't that useful except for debugging. I thought maybe I could use it to determine kind of vehicle - person, bike, car, truck but it didn't appear to work that well.

   **Tim Earl** • 2 years ago

Hi Tony,

Great project! I have ordered all the parts and am eager to

test it out. We live in a small street, one car length wide and cars going like bats out of hell. I have been studying the options for years. I can't use camera as here (in France) it is against the law to point a camera at the road. I considered buying a commercial panel with led's to tell people they are speeding but the price is awefull (3000€+), then I came across your project. There is no law preventing me from pointing a radar at the road. With this I can gather data and graph charts to present to the local road committee hard evidence of an accident (as in pedestrian getting hit) waiting to happen. I hope to couple it to a webapp using ajax to see real time data from my smart phone. Will share the results. Anyways thanks for sharing! Really happy to find this project :)

^ | v • Reply • Share ›



**Leonhard** • 2 years ago

Hi Tony,  
first of all, nice project! You got me hooked.  
I'm currently trying to understand the alsa part of your code (version 2, [radar.py](#)), but it seems strange to me (maybe because I'm not familiar with alsa).

The part that I don't get is, how long the time samples are, that you will feed to your FFT. You setup your alsa device with the following parameters:

```
inp =
alsaaudio.PCM(alsaaudio.PCM_CAPTURE,alsaaudio.PCM_1
inp.setchannels(2)
inp.setrate(44100)
inp.setformat(alsaaudio.PCM_FORMAT_S16_LE)
inp.setperiodsize(1024)
```

According to the pyalsaudio documentation ( <http://larsimisch.github.i...> ) that will result in a read call to read 1024 frames. In line 180 you basically fill a buffer with 2048 of those read calls meaning, that you will need  $2048 * 1024 = 2097152$  samples to complete that loop. At 44100 Hz sample rate you get of course 44100 frames per second which results in this loop recording for ~48 seconds. I extracted this loop from your code and let it run to time it:

```
root@FriendlyARM:~# time python2.7 testAudio.py
real 0m48.161s
user 0m1.330s
sys 0m0.370s
```

After recording for 48 seconds, you put your time series into a FFT effectively losing all time information in the process. Then you do your detection of moving vehicles in the frequency domain. But does that work with such big

timeframes?

Thanks,  
Best regards,  
Leonhard

^ | v • Reply • Share ›



**Tony Butterfield** Mod → Leonhard • 2 years ago

Hi Leonhard, the code I have in the example takes a recording of only 2048 samples I believe. It does this in two lots of 1024 samples. The reason they are all not captured in one call is that some soundcard drivers throw errors if you set period size to large. Each frame needs to be small enough such that you can be processing several a second to get good traffic tracking in realtime. It definitely doesn't only process one frame every 48 seconds! I may have got something wrong as i'm no expert in the ALSA api however the code supplied does work on the RPI with my soundcard.

Cheers, Tony

^ | v • Reply • Share ›



**Leonhard** → Tony Butterfield • 2 years ago

Ok, thanks. Now I see my error, silly. I did not copy&paste your code and, due to the funny font my firefox (or github) uses, mistook the l (small letter L) for a 1 (in line 187: length+=l).

^ | v • Reply • Share ›



**Alex** • 2 years ago

Hi Tony (and others),

I came across your speed camera and found it very interesting for a project I'm currently planning. I want to build a "golf launch monitor" which should detect the ball speed. Therefore I want to use 3 radar modules and triangulate the ball movement with the different speeds and angles. Therefore I thought about a setup with 3 doppler modules (the "cheaper" ones as I know the direction of the ball). Those signals should be processed on a Raspberry Pi... I like the solution you found with the sound card as this solves a lot of the sampling problems. Also I like the pre-built amplifiers. So big thanks for the inspirations... great project ;)

Regarding my project I have a few questions:

- 1) Do you think a golf ball could be "tracked"/recognized within about 5 Meters.
- 2) Could you just explain how you handle the audio signals?



I think your using the stereo in of the sound card. I assume that this limits the input sources to 2. So in my case I would need 2 sound cards?

3) Do you think the noise will make problems. According to your charts the big hump is around 500 to 1000 Hz. The speeds will be around 80 to 160 mph which will transfer to about 6000 to 12000 Hz.

Thanks,  
Regards,  
Alex

^ | v • Reply • Share ›



**Tony Butterfield** Mod → Alex • 2 years ago

Hi Alex, sounds like a fun project. I see no reason why it wouldn't work though the triangulation maths might be simpler with just two sensors but that's just a hunch. The sensors are certainly sensitive enough to detect a movement like that within 5 meters. I've not had the ability to test the sensors with anything moving quite that fast, but at motorway speeds they appeared to work fine.

Yes I handle the audio by sending one of the signals into the left audio channel and one into the right (both after amplification). I think the raspberry pi and raspbian would have no problem with multiple sound cards attached.

Noise is always a problem with this sensor. I am still planning a follow up post after creating a battery powered system in a box with an LCD display. I spent a lot of time trying to create a clean power supply for the sensors and amplifier decoupled from the the raspberry pi which generates a lot of noise. Unfortunately the sensor is very prone to RF noise and power supply noise. I ended up using two separate regulated 5v supplies driven from a battery. The other noise concern might be the audio amplifiers. If you only have weak signals due to distance and size and then these are high frequencies it might be a small problem. But I think there are some very good audio amplifiers out there. I experimented with a few low noise OpAmp chips but most noise came from other sources and the cheap amps worked just as well.

Hope this helps,  
Tony

^ | v • Reply • Share ›

**skot9000** • 2 years ago

That "24.250GHz Stereo Radar Sensor Module RSM-2650" is a tough one to track down in the US. Have you found any other similar modules recently? I'd love to play around with this.

^ | v • Reply • Share ›

**Jon** • 3 years ago

Oh yes, and I forgot to mention that I live right in the sticks and at one point had to wait for a quarter of an hour for a vehicle to pass last night. And even then it was someone I knew who, waved... I need to find a busier road for testing I think...

^ | v • Reply • Share ›

**Jon** • 3 years ago

Hey Tony,

Thanks for the new software, it runs straight off of the bat with no additional fiddling - which is great!

OK, so I've had a good couple of evening's worth of trialling the new software. Indoors, it records people going back and forth, and also the speed of my son's toy tractor when whizzed along the carpet. Outside (on a wheelie bin by the kerbside) it doesn't seem to notice cars or even sizable vans.

Originally, I thought that it was something to do with the angle from our gate (somewhere between 30 and 40 degrees to the road) so I then moved the whole lot to the wheelie bin - which was as close to parallel to the road as I could manage.

My thoughts:

---

see more

^ | v • Reply • Share ›

**Tony Butterfield** Mod → Jon • 3 years ago

Hi Jon,

you'll get there. Just need a bit of diagnostics!

First capture a sample of the audio using command line tool arecord. Listen to it and check there are not any stray noise sources. Also use audacity to do a frequency plot and check that you are picking up all that is expected.

I found plastic boxes to be perfect but I put the sensor right up against the surface to avoid

sensor right up against the surface to avoid reflections that might drown the real signal.

You can also put some debug into the python script to output stuff as it runs. I never saw any problem with not getting high frequencies and thus fast movements.

As far as the angle goes it's important but not critical. I have quite a long range of view so at the furthest point I am almost parallel to the road. The software looks for the highest frequency in a movement and hence gets the most correct reading it can. But I think if you do the maths it's only marginal (see the curves in the spectrogram in my original post - the flat frequency for a period of time is when the vehicle is furthest from the sensor)

Good luck! Tony

^ | v • Reply • Share ›



**Jon** • 3 years ago

Excellent. I'll give that a whirl this evening, Tony. I guess [example.py](#) pulls in (or calls out to) the [radar.py](#) and that in turn does the same to the [tracker.py](#)? Cheers!

^ | v • Reply • Share ›



**Jon** • 3 years ago

Hmmm. So I'm still having a few problems here: I can't for the life of me set the period parameter in [radar.py](#) to anything above 2000, and maybe that's why everyone appears to be doing just under 11mph (which they clearly aren't). I've tried all manner of devices offered by the soundcard - plughw, hw, default, etc. I'll keep plugging away at it...we'll get there in the end...

Cheers!

^ | v • Reply • Share ›



**Tony Butterfield** Mod ➔ Jon • 3 years ago

Hey Jon, as I mentioned earlier to you I set up another instance of the radar earlier in the year as a standalone box. I vaguely remember some issues at that time (and with a more modern version of Raspbian) regarding the sampling buffer size.

I've uploaded my latest code base now at

<https://github.com/tonbut/r...>

You will see amongst quite a few changes that this does concatenate the buffer from potentially several smaller parts. Can you give this a try and see if it



helps? I'm not sure this is so well documented at ATM so give me a ping if it doesn't make sense.

Cheers, Tony



^ | v • Reply • Share ›



**Jon** • 3 years ago

Hah! Success. I finally have it all working, albeit with a couple of mods. I had to fiddle a lot with alsamixer in order to make sure that the line-in was not muted and also was set to be a capture device. Then I had to mod your code slightly to fit my weird set-up:

Line 134 of [radar.py](#) now reads:

```
inp =
alsaaudio.PCM(alsaaudio.PCM_CAPTURE,alsaaudio.PCM_1
```

Line 132 of [radar.py](#) has the period parameter set to 2000. I'd have loved to get to 4096 - but I keep getting the "alsaaudio.ALSAudioError: Capture data too large. Try decreasing period size" error and 2000 is the maximum I can get away with.

I seem to be able to happily track the speed of my wife coming and going out of the kitchen (max 4mph) and can track the speed of a sweeping hand movement.

Glad the neighbours can't see in as they will think we've gone stark-staring mad.

I will continue to get to the bottom of the period parameter business and also probably re-solder, re-box and pretty up the whole thing...

Nice one, Tony. It's a work of genius...

Cheers!

^ | v • Reply • Share ›



**Tony Butterfield** Mod → Jon • 3 years ago



Great news. I'm glad it's working now. The results should be worth the pain. Mine has been happily without tweaks for 18 months now. (Hence forgetting the details of how to get Rasbian to play ball)

I'm interested to see how you use the data. I did consider playing with a sound laser <http://www.soundlazer.com/> to let drivers who were speeding know. ;-) (Another cool technology)

^ | v • Reply • Share ›



**Jon** • 3 years ago

Definitely no .asoundrc file in your home directory? That's odd... I've so far (in this lunchtime) managed to record audio through the line-in (using a cheap pair of headphones as a mic) and played the sound back. That was only after tweaking the .soundrc to make PCM device the default. Trouble is I think I've got the wrong one as your code causes a "Capture data too large, try decreasing period size" when I run it...and that normally means it's looking at the wrong device - or at least the wrong bit of it... Hmm. If you can excuse the pun, I'll keep plugging away. Thanks for the notes, BTW, I think I've got most of that done. I'm now so close to resolving this...

^ | v • Reply • Share ›



**Jon** • 3 years ago

I'm still having a few issues actually capturing the sound from the electronics through the line in. I don't suppose you could paste up the contents of your ~/.asoundrc file could you please, Tony? Thanks!



(<https://github.com/tonbut>)



(<https://twitter.com/tonbut>)



(<https://linkedin.com/in/tonbut>)



(<https://stackoverflow.com/users/3669135/tonbut>)



(<https://soundcloud.com/concession>)



(<https://www.youtube.com/c/TonyButterfield1060>)



(<http://blog.durablescope.com/index.xml>)

©Tony Butterfield • 2018 • Durable Scope (<http://blog.durablescope.com/>)

Theme based on beautiful-jekyll (<http://deanattali.com/beautiful-jekyll/>)

