GNU Radio Energy Detector

Freek van Tienen
4094123

28-04-2016

Abstract

Due to regulations a lot of the usable radio spectrum is restricted. These restrictions are regulated per country and only small proportions of the total spectrum is available for use. But since these free to use bands are currently used a lot, due to the increase of wireless usage all around the world, it becomes more and more difficult to propagate a signal in these bands. Therefore new techniques have come around, like cognitive radio. Cognitive radio makes use of the restricted radio spectrum, by detecting if this spectrum is used at a certain moment. Making a perfect detector is difficult, since most of these detectors are prone to errors. In this document we describe an implementation of such a detector using GNU Radio, and analyse it's performance.

1 Problem description

When we want to use the restricted spectrum we have to make sure we don't interfere with the original signal purposed for this specific spectrum. This means we have to be able to detect the original signal in this spectrum and avoid these frequencies. For the purpose of this experiment we have chosen the DVB-T spectrum, which in most cities has vast amount of white spaces.

There are several ways of detecting if a signal is present and one of the most easy ways is the use of an energy detector. This energy detector calculates the energy that is being measured for a certain frequency. Based on this calculated energy it decides whether the frequency is free or used by a DVB-T channel. Hence the name energy detector.

To be able to measure the performance of such an detector a ROC can be used. This will show us the amount of True Positive errors versus the amount of False Positive errors. In the next chapter a design of such an energy detector is described and an automated way of generating these ROC curves is shown.

2 Design and implementation

The design of the energy detector is based on the sample design given in the Wireless Networking course. Some small adjustments were made in order to automate the experiment for reproducibility. The basic design of the energy detector is made in GNU Radio Companion, which generates python code. The result can be seen in Figure 1.

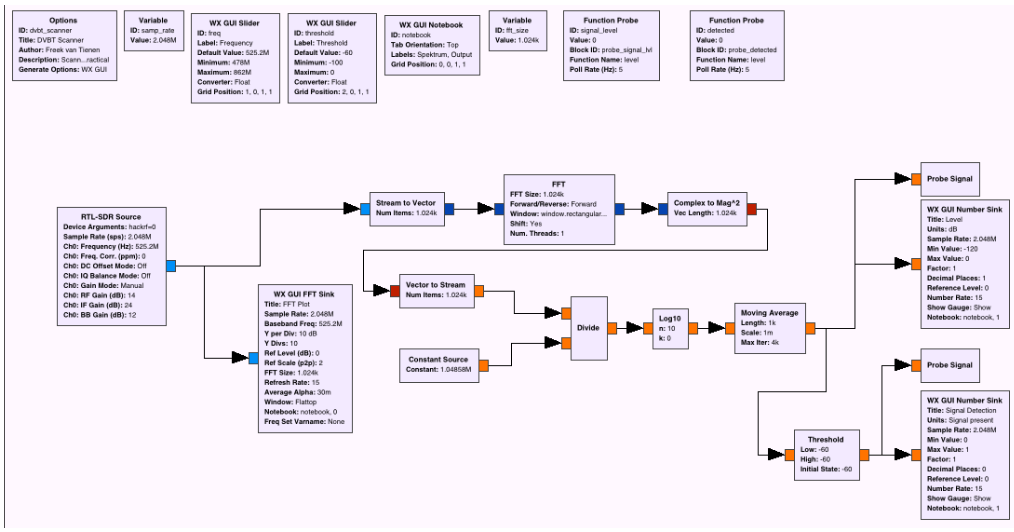

Figure 1: GNU Radio Companion implementation.

The main adjustment in the energy detector is the addition of an "Moving Average" block after the computation of the energy level. This is done to stabilise the measurement, since the initial calculation of the energy level was very noisy. The second adjustment that was made was the adding of "Probe" blocks which were added for the easy access to both the signal level and the detection state trough python code. This was needed in order to automate the fetching of the results.

Now in order to automate the generation of a ROC curve a simple python script was made to automatically perform measurements of the energy level. This script can be seen in Figure 2.

```
# Set the detection level
scanner.set_threshold(threshold)

# Go trough the frequencies
for freq in range(int(freq_min*1e6), int(freq_max*1e6), int(freq_step*1e6)):
  # Set the frequency
  scanner.set_freq(freq)

  # Wait for some time
  time.sleep(wait_time)

  # Do the measurement
  freq_mhz = freq / 1e6
  signal_level = scanner.get_signal_level()
  detected = scanner.get_detected()
  f.write("%.2f,%.2f,%.2f,%d\n" % (freq_mhz, threshold, signal_level,
      detected))
```

Figure 2: Energy detection scanner.

As can be seen, this script scans trough the frequencies with *freq_step*. And for each frequency it will set the frequency, wait for some time to make sure the moving average is stable. Then it will get the signal level and the detected state and log this to a csv file.

After the measurements are taken an ROC curve must be generated. In order to do this we need to have a way to verify if there really is a DVB-T channel on a certain frequency. This is done by looking up online at which frequencies the stations are for a specific city, in this case Delft. In Figure 3 can be seen how this check is implemented in the ROC graph script.

```
dvbt_freq = [498, 522, 698, 722, 762]    # The DVB-T frequencies in Delft
dvbt_width = 7.61                         # Width of a DVB-T channel in MHz

# Check if this is an actual DVB-T station
is_dvbt = False
for freq in dvbt_freq:
  if (freq - dvbt_width/2) < float(row[0]) < (freq + dvbt_width/2):
    is_dvbt = True
    break
```

Figure 3: DVB-T check.

To generate the ROC curve a python package called *sklearn* can be used. This needs both the actual values(if a certain frequency is an actual DVB-T channel) and the measurements(of the signal level) as input. It will then calculate both the false positive rate and true positive rate, which are then plotted on the x and y axis. But to simulate the results as a normal distribution some statistics are calculated, like the mean and standard deviation. Then random numbers will be generated from these distributions in order to generate a PDF graph. Also a new ROC will be generated based on the normally distributed signals. This can be seen in Figure 4.

```
# Plot the actual ROC curve
false_positive_rate, true_positive_rate, thresholds = roc_curve(actual,
    measurement)
plot_roc_curve(false_positive_rate, true_positive_rate, "roc_real")

# Calculate statistics
(pos_mean, pos_std, pos_rvs, pos_pdf, pos_cdf) = calc_statistics(
    positive_meas)
(neg_mean, neg_std, neg_rvs, neg_pdf, neg_cdf) = calc_statistics(
    negative_meas)

# Plot the PDF graph
plot_pdf_curve(pos_rvs, pos_pdf, neg_rvs, neg_pdf, "pdf")

# Calculate the ROC based on statistics
curve_x = []
curve_y = []
for x in np.arange(-85, -65, 0.1):
  curve_x.append(neg_cdf[find_nearest(pos_rvs, x)])
  curve_y.append(pos_cdf[find_nearest(neg_rvs, x)])

# Plot the ROC curve base on statistics
plot_roc_curve(curve_x, curve_y, "roc_norm")
```

Figure 4: ROC and PDF curve generation.

The full source code is available on GitHub (`https://github.com/fvantienen/wireless-practical`).

3 Results

At two different locations measurements were taken in order to generate ROC curves. The first location was at Tanthof inside my student room close to the window at the 7th floor next to other high flats. The second location was inside the Aerospace building in a room without any outside walls or windows. For the tests a frequency range from 400MHz to 800MHz with a step of 0.5MHz was scanned and checked for the energy level for detecting DVB-T channels.

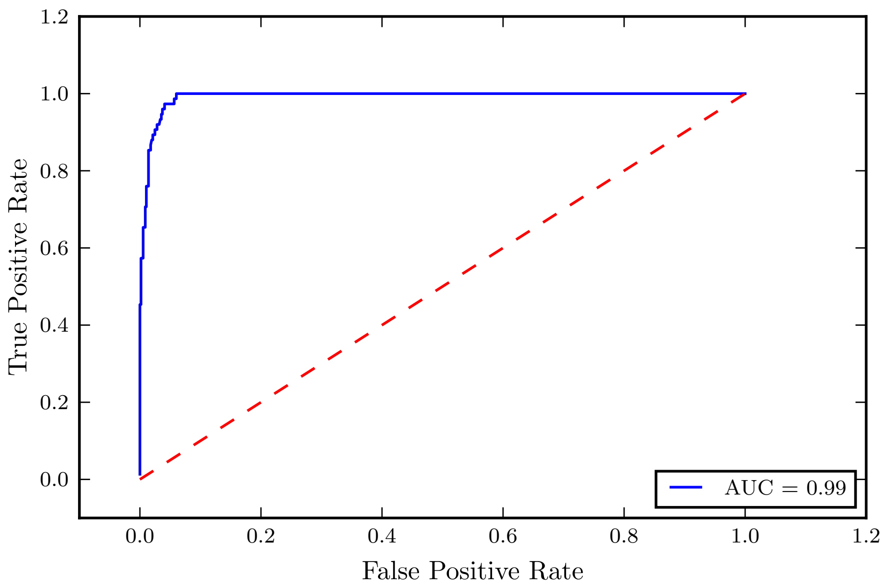

Figure 5: ROC curve from Tanthof based on real measured data

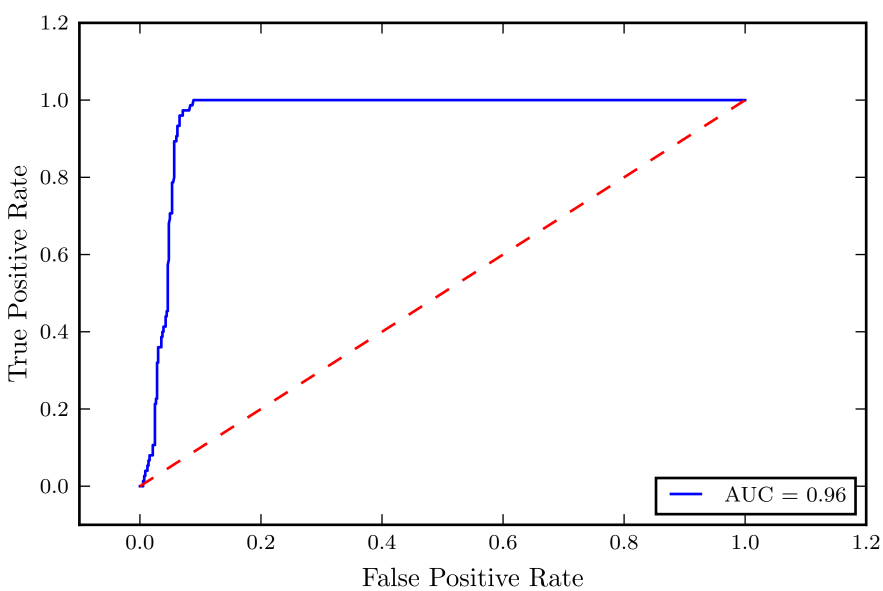

Figure 6: ROC curve from LR based on real measured data

Base on the real data the ROC were generated and can be seen in Figure 5 and Figure 6. Here we can see that at Aerospace)the amount of false positives is slightly bigger than the ones from Tanthof. Which can be described by the effect of propagation through concrete, which is more difficult for higher frequencies. This makes these channel's energy more difficult to detect between the noise. Both of these graphs have a very high AUC number from which you could conclude that this is an effective way of detecting DVB-T channels.

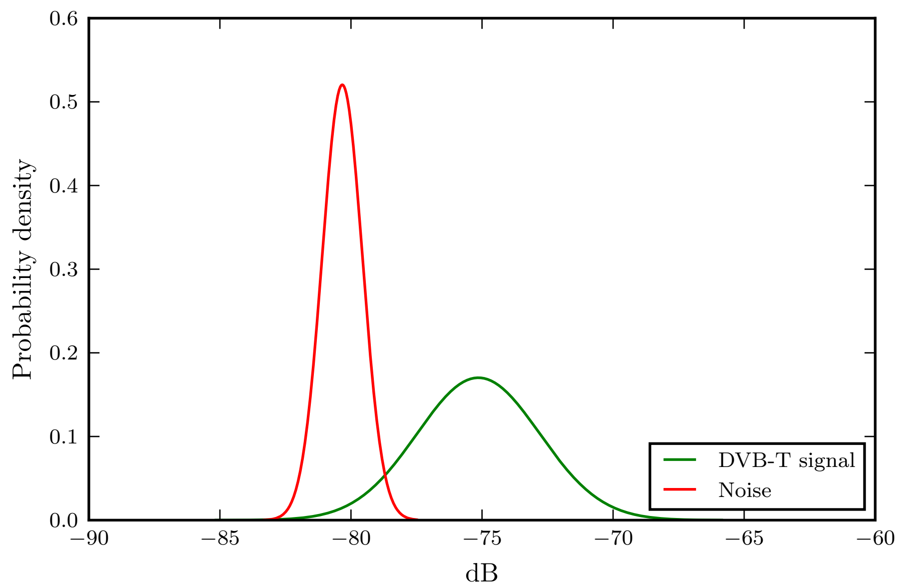

Figure 7: PDF curve from Tanthof based on Normal distribution.

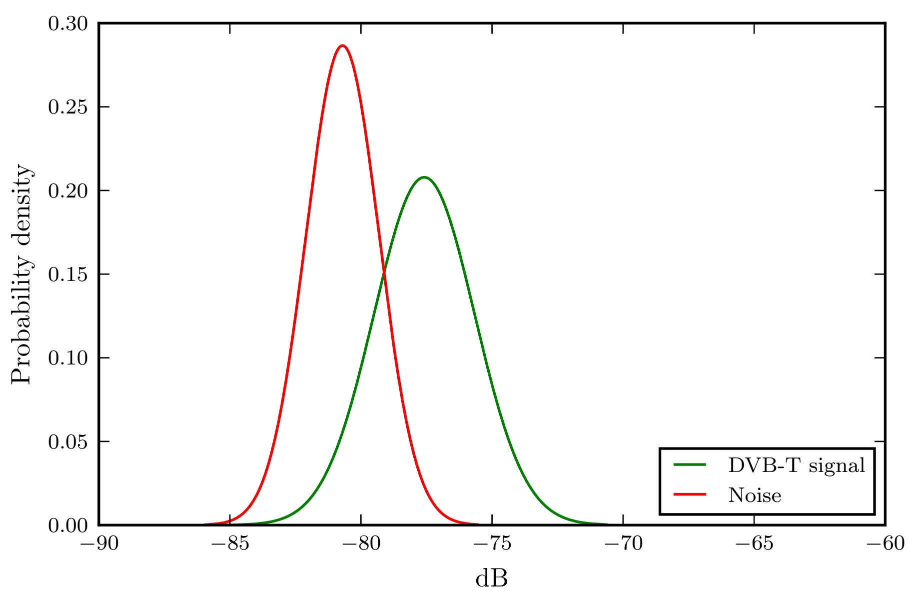

Figure 8: PDF curve from Aerospace based on Normal distribution.

But if we look mare closely at the data we know that in the range from 400MHz to 800MHz only around 38MHz of these frequencies contains DVB-T channels. This makes the detected signal versus the noise biased, since these measurements contained a lot more noise compared to the amount DVB-T channels. In order to solve this I calculated the mean and standard deviation to generate a new dataset with even number of samples to generate a nice ROC curve. The PDF of these new dataset can be seen in Figure 7 and Figure 8. These show us that the detection inside the Aerospace building was a lot worse than expected.

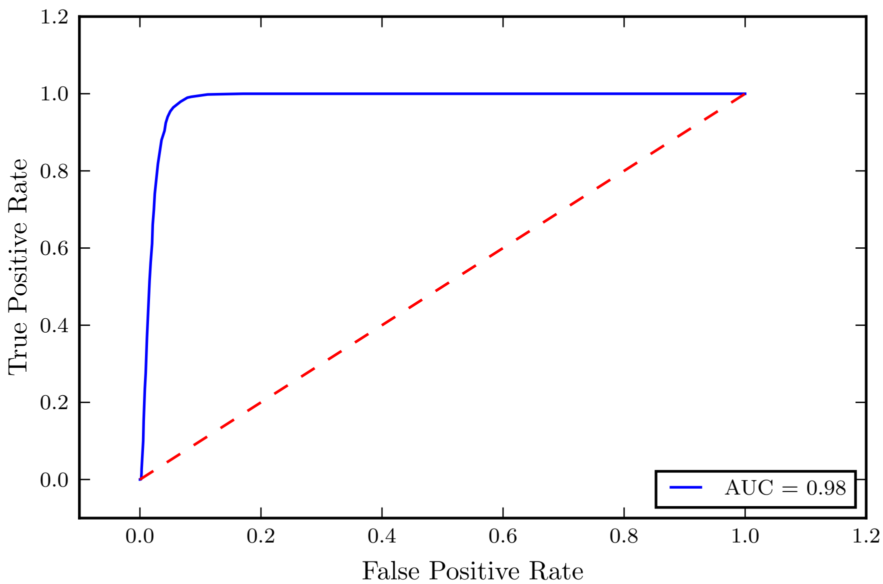

Figure 9: ROC curve from Tanthof based on Normal distribution.

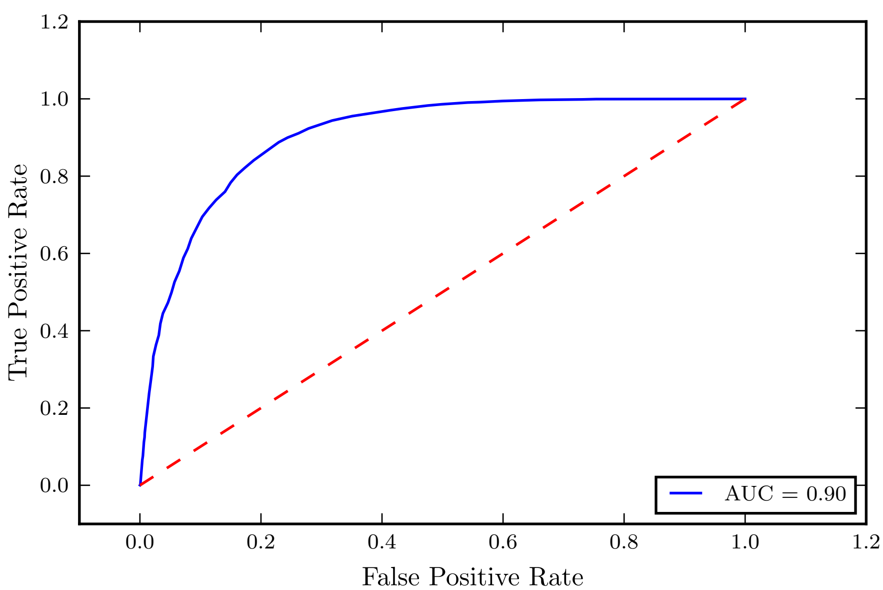

Figure 10: ROC curve from Aerospace based on Normal distribution.

If we then take these PDF and generate a new ROC curve from it you get Figure 9 and Figure 10. Which immediately shows us that our detector functioned not as good as expected from the first graphs in Figure 6. Also the AUC is much less s before, it dropped with 0.06 for Aerospace and 0.01 for the Tanhof based measurements.

4 Conclusion

As can be seen in the ROC curves the detector functions almost perfectly in reasonable conditions. When the conditions are tested to it's limits (in a room with no windows or outside walls) it becomes more difficult to really detect the DVB-T signals.

To conclude I would like to say that this is an effective way to measure the performance of an DVB-T detector and still great improvements can be made in the way a DVB-T channel can be detected. The automation of the whole process makes the generation and calculation of ROC curves extremely easy.