International Journal of Internet, Broadcasting and Communication Vol.10 No.1 1-10 (2018)
http://dx.doi.org/10.7236/IJIBC.2018.10.1.1

IJIBC 18-1-1

Implementation of Spectrum Sensing with Video Transmission for Cognitive Radio using USRP with GNU Radio

Huynh Thanh Thien, Hiep Vu-Van, and Insoo Koo*

Department of Electrical and Electronic Engineering, University of Ulsan, Ulsan 680-749, Korea
thanhthien173@gmail.com, vvhiep@gmail.com, and iskoo@ulsan.ac.kr

Abstract

In cognitive radio (CR), secondary users (SUs) are able to sense the absence of primary users (PUs) in the spectrum. Then, SUs use this information to opportunistically access the licensed spectrum in the PUs' absence. In this paper, we present an implementation of real-time video transmission with spectrum-sensing between two points using GNU Radio and a National Instruments 2900 Universal Software Radio Peripheral (USRP). In our project, spectrum-sensing is implemented at both transmitter and receiver. The transmitter senses the channel, and if the channel is free, a video signal (which could be a real-time signal from a video file) will be modulated and processed by GNU Radio and transmitted using a USRP. A USRP receiver also senses the channel, but in contrast, if the channel is busy, the signal is demodulated to reproduce the transmitted video signal. This project brings in several challenges, like spectrum-sensing in the devices' environment, and packets getting lost or corrupted over the air.

Keywords: Cognitive Radio, Spectrum Sensing, Video Transmission, USRP, GNU Radio.

1. Introduction

Nowadays, wireless communications systems have grown significantly. However, using the radio frequency (RF) spectrum is increasing more dramatically in order to meet the growth in applications requiring broadband. Because usage of the spectrum is inefficient, new spectrum management models are being developed to opportunistically utilize wireless spectrum resources. Currently, technologists are interested in cognitive radio (CR) because attaining efficient use of the overall spectrum is increasingly significant [1, 7].

Cognitive radio was recognized as an enabling technology to mitigate the abuse of scarce RF spectrum; hence, dynamic spectrum access was proposed to share the available spectrum through opportunistic usage of the frequency bands by secondary operators without interfering with the primary networks. When a CR network (CRN) is installed, it enables secondary networks to perform the following tasks: spectrum sensing, spectrum management, and mobility management.

This technology offers the opportunity to optimize spectrum access, assists in preventing interference, and adapts to instant spectrum slot availability from the unused spectrum pool.

Various approached have been proposed for spectrum sensing, such as matched filters, energy detection (ED), feature detection, and more recently, wavelet detection [2]. In these methods, ED measures the energy of the received signal at the secondary user (SU) to determine the presence of signals from the primary user (PU)[8]. This simple method is able to gather spectrum occupancy information quickly. Other intelligent

Manuscript Received: Dec. 5, 2017 / Revised: Dec. 19, 2017 / Accepted: Dec. 30, 2017
Corresponding Author: iskoo@ulsan.ac.kr
Tel: +82-52-259-1249, Fax: +82-52-259-1686
School of Electrical Engineering, University of Ulsan, Korea.

schemes are used to detect the PU's presence: matched filters, feature detection, and cyclostationary detection, which require additional information about the primary signal, such as the modulation and/or coding type, signal processing capabilities, or other features to recognize the primary signal when it appears. Those schemes are more precise than ED, but they consume more computational resources in a CRN. Table 1 lists the schemes used to detect the transmitter signal, along with their pros and cons.

Table 1. Primary transmitter detection schemes

Sensing schemes	Pros	Cons
Energy detection	- Easy to implement - No information about the primary signal required	- False alarms - Uncertainty in noise power
Matched filter	- Fewer samples required - Less noise effects	- Primary signal information is required - Synchronization between SUs
Feature detection	- Robustness - Fewer false alarms	- Cyclostationary signal - Consumes computational power

There has been tremendous growth in the fields of multimedia and mobile communications. The convergence of these two fields has resulted in mobile multimedia communications, which has attracted the attention of the research community around the world [3]. A lot of research has been done in this area to find new methodologies to improvise or innovate ways to implement the technology with better bandwidth and energy efficiency, because these two resources are limited.

Universal Software Radio Peripheral (USRP) is a hardware platform that allows general-purpose computers to function as high-bandwidth software radios [3]. The application layer communicates with the physical layer through some intermediate layers. For a stationary host, this activity seems to be a good option where the communications protocol is systematic and defined according to the environment where it is located. But for a mobile node, the environment conditions change over time, and hence, the transmit power, bandwidth, and quality of the channel have to be continuously monitored and passed on to the application layer in order to select a suitable algorithm. In turn, the physical layer has to change per the suggestions from the application layer from time to time. GNU Radio and USRP bring the application developer as close to the hardware as to the antenna itself, and provides the user with the flexibility to change the communications parameters on the fly [4].

In the research presented here, a spectrum sensing system based on ED combined with video transmission is developed using USRP and GNU Radio as hardware and software platforms, respectively. The video is transmitted and received after spectrum-sensing is implemented at both the transmitter and receiver.

This paper is organized as follows. Section II presents background on the spectrum sensing system model with ED, which introduces sensing performance metrics. Section III presents a spectrum-sensing experiment and video transmission. Preliminary results are presented in Section IV. Finally, Section V provides the conclusion and future works.

2. Theory

The absence or presence of the PU in the environment can be defined through two binary hypotheses: H_0 and H_1 [5]. The received signal at SU, $x(n)$, can be expressed as

$$x(n) = \begin{cases} w(n) & H_0 \\ h * s(n) + w(n) & H_1 \end{cases} \quad (1)$$

Where $n = 1, \dots, N$; N is the number of samples; h is the amplitude gain of the channel, which is assumed to be 0 and 1 under hypothesis H_0 and H_1, respectively; $w(n)$ is additive white Gaussian noise (AWGN) with zero mean and variance σ_w^2; and $s(n)$ is the transmitted signal. The principle of the spectrum sensing operation is to decide between H_0 and H_1 based on observation of received signal $x(n)$.

The presence of the PU can be detected simply by calculating the amount of received power in the

considered frequency band and comparing it with a set threshold. The algorithm for ED will make a decision on spectrum occupancy by the PU if the received signal power is greater than the threshold.

The mean power of *N* samples is collected and can be represented as random variable:

$$P_N = \frac{1}{N}\sum_{n=0}^{N-1}|x(n)|^2 \tag{2}$$

where P_N is the mean output of the energy detector. In traditional ED, the reliability of the spectrum sensing algorithm is crucially influenced by the threshold. Then, based on Equation (1), generic decision rule D_N can then be modified to the considered case:

$$D_N = \begin{cases} H_0, & if\ P_N \leq \gamma \\ H_1, & otherwise\ P_N > \gamma \end{cases} \tag{3}$$

3. Experiments

3.1 Hardware/Software Overview

GNU Radio and USRP provide a powerful radio communications platform with low-cost and high-quality realization of software defined radio (SDR). USRP delivers to the user various functionalities allowing for efficient, real-time realization of complicated wireless systems that operate in the radio frequency (RF) band. The URSP platform is used to convert the digital baseband signal delivered from the computer to an analog signal in the RF band. Complex processing like modulation and signal processing, which are conventionally implemented in hardware, can now be implemented in the software and are easily accessible to the user developing an application. The hardware has the antenna, RF front end, analog-to-digital and digital-to-analog converters, a universal serial bus (USB) interface, a programmable USB 2.0 controller for communications between USRP and GNU Radio, and a field-programmable gate array (FPGA) for implementing four digital down converters and high-rate signal processing. There are newer versions of USRPs that have Ethernet interfaces and powerful FPGAs for improved speed and processing. In our experiments, two USRP boards were used; the PU signal for spectrum sensing and the transmitted video signal on the first board, whereas the second one, acting as the SU, was also used for spectrum sensing and for video reception.

The software process realized in the open-source GNU Radio environment [6] has libraries for various modulation schemes, error-correcting codes, and scheduling. GNU Radio runs as an application that interacts with the USRP hardware. The whole signal process was realized in the GNU Radio environment, and in particular, in the graphical tool called GNU Radio Companion (GRC), where the whole system is built from blocks. Besides that, applications can be created using the Python script language behind the blocks. The performance-critical signal processing path is implemented in C++. A simplified wrapper and interface generator (SWIG) interface, which is an interface compiler, is used to link C++ with Python.

3.2 System Implementation

A schematic diagram and the whole system of spectrum sensing with video transmission and receiving are shown in Figure 1 and Figure 2, respectively. The experimental setup consisted of two USRPs, one laptop, and one PC. A dedicated photograph of the experimentation setup is shown in Figure 3.

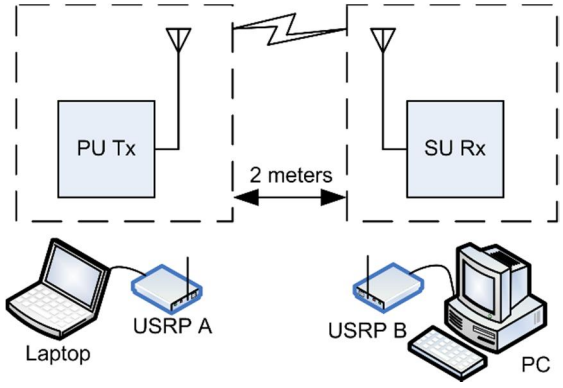

Figure 1. Schematic system diagram

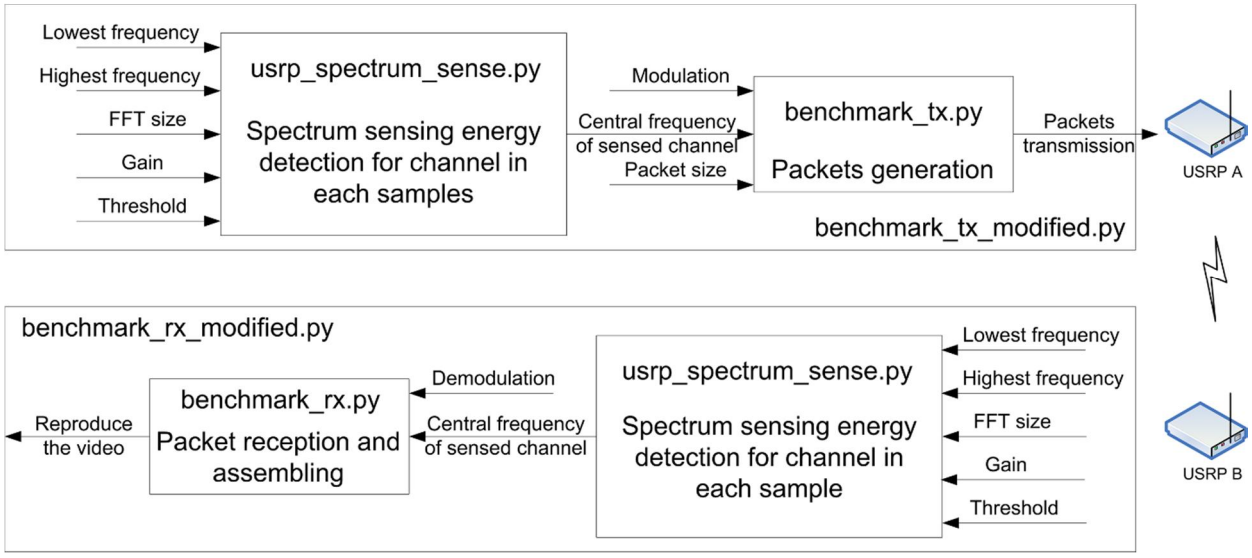

Figure 2. The whole system of spectrum sensing with video transmission and reception

Figure 3. PU transmitter and SU receiver with USRP hardware and GNU Radio software

3.3 Spectrum Sensing

The script *usrp_spectrum_sense.py* was considered for the design of the spectrum-sensing functionality implemented in the testbed; it can be found in the toolkit provided by the GNU Radio software. This script has been used as basic code for implementing a wideband spectrum analyzer to properly sense the desired spectrum bands. However, it was extended to properly sense the spectrum bands considered in the different spectrum management strategies implemented in the testbed. The script receives different input parameters from the user, such as the lowest and highest frequencies of the band to be sensed; the fast Fourier transform (FFT) size parameter for the number of samples; the bandwidths considered in performing the magnitude analysis of the sensed signal; gain; and threshold. Then, the script computes the signal energy detected in each sample during execution of the spectrum sensing functionality. Finally, the output of the script provides

the center frequency at the sensed channel.

Both transmitter and receiver, respectively, also implemented spectrum sensing for the sensed channel that is free or detected to transmit or receive video. The transmitter will sense the channel, and if the channel is free (its mean average signal is less than a given threshold based on measurements), a video signal is transmitted on the sensed frequency. Conversely, if the channel is busy, a video signal will be transmitted on the next frequency[9]. The USRP receiver also senses the channel, but in contrast, if the channel is detected, the signal is demodulated to reproduce the transmitted video signal. Figure 4 shows the flow chart for sensing the channel in the system.

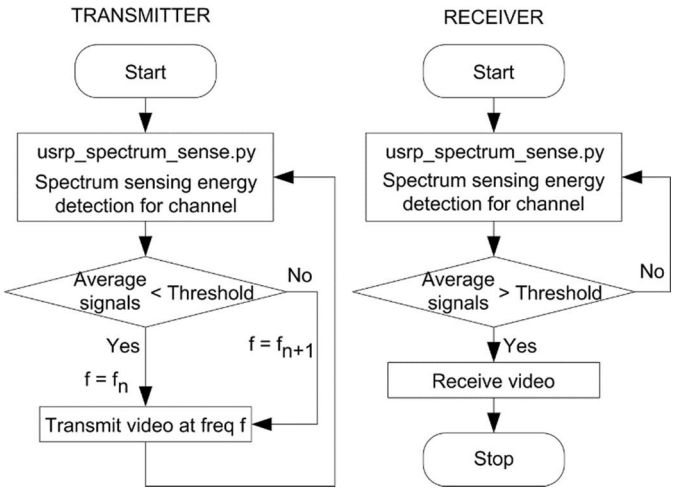

Figure 4. The flow charts of the transmitter and receiver that are integrated for sensing the channel in the system

3.4 Video Transmission

The data transmission and receiving functionalities were implemented through GNU Radio's *benchmark_tx.py* and *benchmark_rx.py* scripts, respectively. They are also found in the toolkit provided by the GNU Radio software. In this paper, we integrated *usrp_spectrum_sense.py*, *benchmark_tx.py*, and *benchmark_rx.py* scripts into *benchmark_tx_modified.py* and *benchmark_rx_modified.py* scripts, *so* implementation of spectrum sensing with video transmission and receiving is more convenient. These scripts take the following input parameters from users: lowest and highest frequency, FFT size, transmitter and receiver gain, and threshold for the spectrum-sensing functionality; a modulation scheme for Gaussian minimum shift keying (GMSK); the sensed frequency from the spectrum sensing; and packet size (only for the script *benchmark_tx.py*).

On the transmitter side, after implementation of spectrum sensing, tasks of the *benchmark_tx* Python file are as follows.

- Data files are read frame by frame from hard disk, and then each frame is packed with packet sequence numbers as the application layer header.
- The application layer packets are sent to the data link layer and the physical layer, where information on the preamble, access codes, cyclical redundancy checking, etc. is assembled.
- In the last step, the assembled packets are modulated with GMSK and sent to the USRP through a USB interface. USRP then transmits each packet on the sensed channel frequency.

On the receiver side, after the signal is processed with spectrum sensing, the *benchmark_rx* Python file will be implemented. Tasks of this script are as follows.

- Read data from the USRP through the USB interface, which are demodulated with GMSK
- Then, the received link layer packets are disassembled into application layer packets with packet sequence numbers.

- In the last step on the receiver side, all correctly received packets are saved to a file, displaying the decoded video sequences in real time on a display terminal.

In this research, we set input parameters for transmitter (*./benchmark_tx_modified.py 893M 911M --fft 512 -g 60 -q 10)* and receiver (*./benchmark_rx_modified.py 893M 911M --fft 512 -g 30 -q 4*), where *893M* and *911M* are the lowest and highest frequency in megahertz (MHz), *fft* is FFT size, *g* is gain in decibels (dB), and *q* is the threshold in dB. All parameters can change for each requirement from users.

4. Results and Discussion

4.1 Implementation of Video Transmission with H.264 and MP4 on a Specific Channel

In this testbed, we used two computers, one PC (the receiver) had the Linux operating system (OS) installed, and one laptop (the transmitter), also had the Linux OS. For more conveniently displayed video, we used a *ffplay* platform to show the video. In this scheme, we transmitted under two scenarios: first, a video in H.264 format was 225,235 bytes in size, and second was a video in MP4 format in a higher resolution at 9.3 MB in size, setting input parameters at both transmitter and receiver for the specific channel (*./benchmark_tx.py –f 900M* for the transmitter and *./benchmark_rx.py –f 900M* for the receiver). The received videos in H.264 and MP4 formats with information about received packets, file length, and status of reception are shown in Figure 5 and Figure 6, respectively.

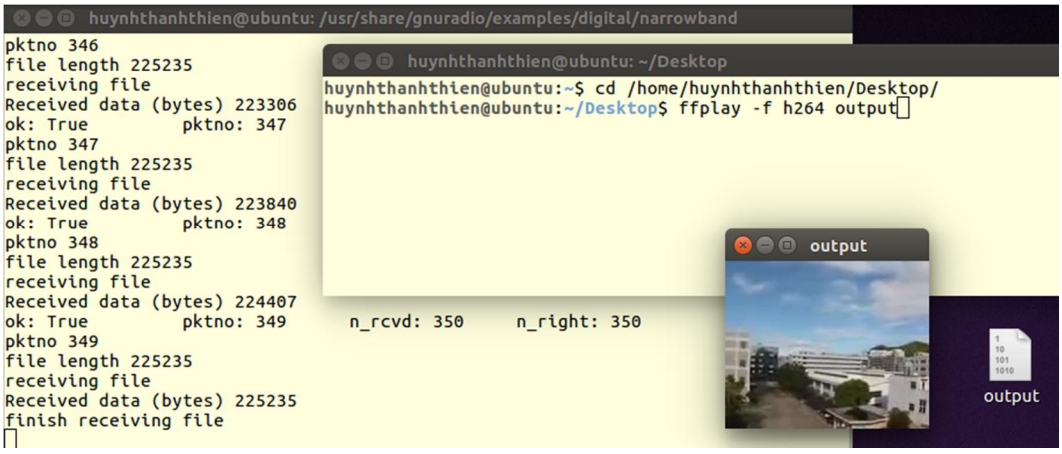

Figure 5. Video received in real time in H.264 format on the specific channel (frequency = 900 MHz)

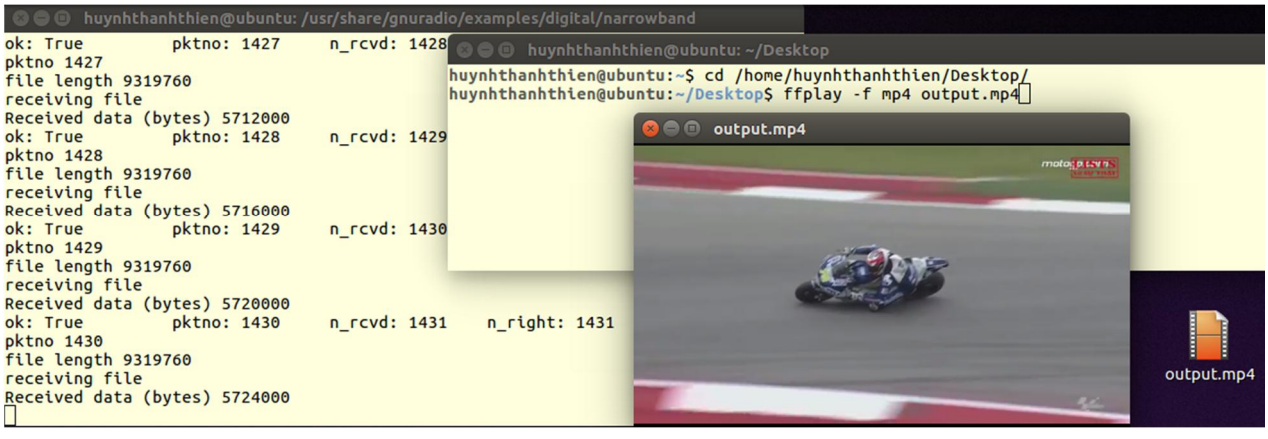

Figure 6. Video received in real time at a higher resolution and in MP4 format on the specific channel (frequency = 900 MHz)

For evaluation of performance in the testbed, we considered received packet error rate (R-PER) and right received packet error rate (RR-PER) parameters. At the transmitter, we assembled the 225,235 bytes into 350 packets, and we then estimated performance of the testbed for PER based on the number of received packets and right received packets. We estimated performance of the testbed under two scenarios: first, the distance between transmitter and receiver was fixed, with increasing transmitter gain; second, transmitter gain was fixed, but with changes in transmit distance. In both cases, receiver gain was fixed at 30 dB. The results of the two scenarios are shown in Figure 7 and Figure 8, respectively. Figure 7 and Figure 8 show number of transmitted packets (left side column), received packets (middle column), and right received packets (right side column) in the same considered value. Due to influence of some factors such as transmission distance, reception and transmission gains, some packets will be lost in transmission process. In case receiver can not decode packets, number of right received packets will be less than number of received packets.

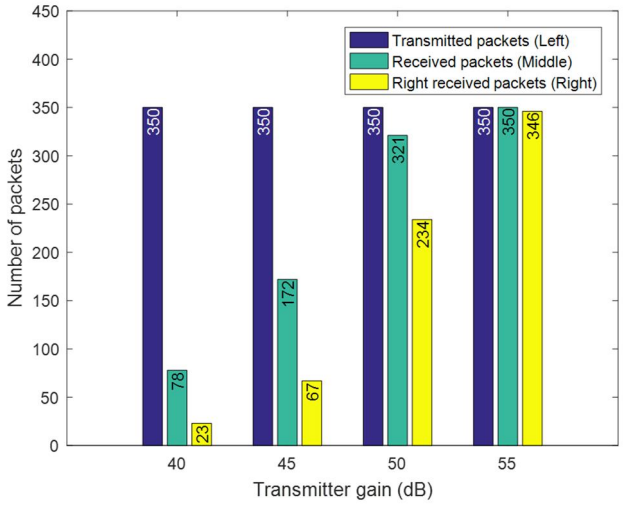

Figure 7. Number of received packets, right received packets, and transmitted packets with changes in transmitter gain but a fixed receiver gain of 30 dB

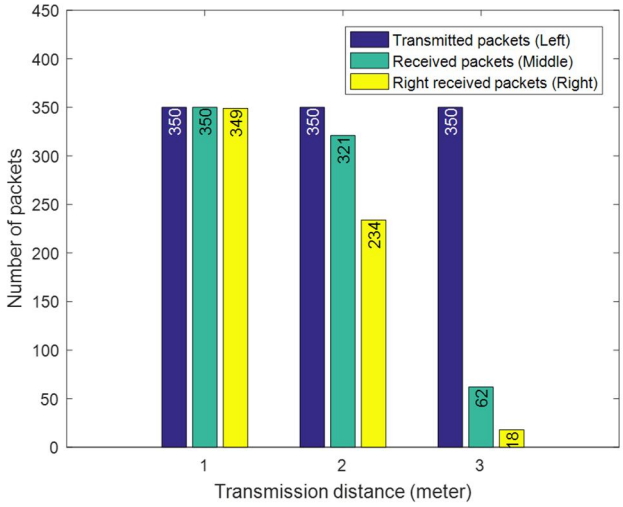

Figure 8. Number of received packets, right received packets, and transmitted packets with changes in transmission distance, but with fixed receiver and transmitter gain (30 dB and 50 dB, respectively)

Figure 7 shows that the more transmitter gain we use, the lower the PER. Besides that, PER can also be estimated based on the transmission distance between transmitter and receiver: the farther the distance, the higher the PER as shown in Figure 8.

4.2 Implementation of Spectrum Sensing with Video Transmission at Transmitter and Receiver

We set up the system with input parameters for transmitter and receiver as explained in the video transmission part of Section III. The input transmitter Python script and the results of spectrum sensing on the transmitter side at thresholds equal to 8 dB and 4 dB are shown in Figure 9 and Figure 10, respectively.

```
htt@ubuntu:~/Desktop/test$ ./benchmark_tx_modified.py 893M 911M --fft 512 -g 60 -q 8
number of sensing: 1
Odata ............center_freq 894000000.0 average_power_db 5.61509255189 decision 0 interval 512
center_freq for transmit 894000000.0
Warning: failed to enable realtime scheduling
number of transmitting: 1
................................................................................
.....sent one file
begin retransmission
total data 65597
number of transmitting: 2
```

Figure 9. Spectrum sensing and video transmission at the transmitter with a threshold of 8 dB

```
htt@ubuntu:~/Desktop/test$ ./benchmark_tx_modified.py 893M 911M --fft 512 -g 60 -q 4
number of sensing: 1
Odata ............center_freq 894000000.0 average_power_db 6.13188746146 decision 1 interval 512
center_freq for transmit 896000000.0
Warning: failed to enable realtime scheduling
number of transmitting: 1
................................................................................
....sent one file
begin retransmission
total data 65597
number of transmitting: 2
```

Figure 10. Spectrum sensing and video transmission at the transmitter with a threshold of 4 dB

In Figure 9, we can see that at a channel frequency of 894 MHz, the mean power (equal to 5.615 dB) is less than the threshold (8 dB), so the transmitter will transmit the video on the same channel. Conversely, in Fig. 10, if mean power (equal to 6.131 dB) is greater than the threshold (4 dB), the transmitter transmits the video on the next channel (a frequency of 896 MHz).

At the receiver, the input receiver Python script and the results of spectrum sensing on the receiver side with a threshold of 10 dB and 7 dB, are shown in Figure 11 and Figure 12, respectively.

```
huynhthanhthien@ubuntu:~/Desktop/test$ ./benchmark_rx_modified.py 893M 911M --fft 512 -g 60 -q 10
number of sensing: 1
Odata......center_freq 894000000.0 average_power_db 8.2325469596 decision 0 interval 512
data......center_freq 896000000.0 average_power_db 4.89588930676 decision 0 interval 512
data......center_freq 898000000.0 average_power_db 5.46638444503 decision 0 interval 512
data......center_freq 900000000.0 average_power_db 4.84901052474 decision 0 interval 512
data......center_freq 902000000.0 average_power_db 5.3561857417 decision 0 interval 512
data......center_freq 904000000.0 average_power_db 5.22716985635 decision 0 interval 512
data......center_freq 906000000.0 average_power_db 4.74496769023 decision 0 interval 512
data......center_freq 908000000.0 average_power_db 7.14135384557 decision 0 interval 512
data......center_freq 910000000.0 average_power_db 5.48912710605 decision 0 interval 512
number of sensing: 2
data......center_freq 894000000.0 average_power_db 9.09895512577 decision 0 interval 512
data......center_freq 896000000.0 average_power_db 6.12622834463 decision 0 interval 512
data......center_freq 898000000.0 average_power_db 4.41426363849 decision 0 interval 512
```

Figure 11. Spectrum sensing at the receiver with a threshold of 10 dB

```
huynhthanhthien@ubuntu:~/Desktop/test$ ./benchmark_rx_modified.py 893M 911M --fft 512 -g 60 -q 7
number of sensing: 1
Odata......center_freq 894000000.0 average_power_db 9.92778212075 decision 1 interval 512
Enter to receive video:
Warning: Failed to enable realtime scheduling.
DDOok: True      pktno: 47       n_rcvd: 1       n_right: 1
pktno 47
transmitted 699 bytes
file length 65597
receiving file
Received data (bytes) 699
```

Figure 12. Spectrum sensing at the receiver with a threshold of 7 dB

In Figure 11, we can see that the mean power on the channels is less than the threshold (10 dB), so the receiver will continue to sense the channel. Conversely, Figure 12 shows that the mean power (9.927 dB) is greater than the threshold (7 dB), so the receiver downloads the video on the same channel (a frequency of 894 MHz).

5. Conclusion

The paper presents the results of research into the implementation of USRP and GNU Radio for a spectrum-sensing system with video transmission. In this research, video in H.264 and MP4 formats can be transmitted on a specific channel or on a sensed channel. Performance of the video transmission is evaluated through PER, which is based on the number of received packets compared to the number of transmitted packets. Performance on the testbed was evaluated based on exploring parameters, such as gain and distance between USRPs. We anticipate that gain and distance between two USRPs plays an important role. Results show that the transmission and reception of the video files should be smooth and successful when the testbed is implemented at high gain and over a short distance. There could be distortion in the video as the distance increases due to a loss of packets or from packet errors.

Acknowledgment

This work was supported by the National Research Foundation of Korea funded by the MEST under Grant NRF 2015R1D1A1A09057077 and NRF 2016K2A9A1A01950711.

References

[1] Mahmood A. Abdulsattar, and Zahir A. Hussein, “Energy Detection Technique for Spectrum Sensing in Cognitive Radio: A Survey,” *International Journal of Computer Networks & Communications (IJCNC)*, Vol. 4, No. 5, pp.

223-242, Sep 2012.

[2] D. Cabric, S. M. Mishra, and R. W. Brodersen, “Implementation Issues in Spectrum Sensing for Cognitive Radios," *Conference Record of the thirty-eighth Asilomar Conference on Signals, Systems and Computers*, Vol. 1, pp. 772-776, IEEE, 2004.

[3] S. Nimmi, V. Saranya, Theerthadas, and R. Gandhiraj, “Real-Time Video Streaming using GStreamer in GNU Radio Platform,” *The International Conference on Green Computing Communication and Electrical Engineering (ICGCCEE)*, pp. 1-6, Oct.16, 2014.

[4] A. Khattab, and D. Perkins, and B. Magdy, *Cognitive Radio Networks: From Theory to Practice, Analog Circuits and Signal Processing*, Springer, pp. 71-72, 2013.

[5] M. Sowmiya, M. Sangeetha, “Energy Detection Using NI USRP 2920,” *International Journal of Science and Research (IJSR)*, Vol. 5, pp. 597-603, May 2016.

[6] Gnu radio - the free and open software radio ecosystem. http://gnuradio.org.

[7] J. Ghosh, I. Koo, "An Approach to maximize throughput for Energy Efficient Cognitive Radio Networks," The International Journal of Advanced Culture Technology, Vol.1, No.2, pp.18-23, 2013

[8] L.Tan, J. Lee, H. Kong, Primary User Detection using a Generalized Selection Combining over RayleighFading Channel,” The International Journal of Internet, Broadcasting and Communication, Vol.2 No.1, pp.1-3, 2010.

[9] T. Tuan, I. Koo, “An Efficient Channel Selection Algorithm for Cognitive Radio Sensor Networks,” The International Journal of Internet, Broadcasting and Communication, Vol.3 No.2, pp.26-30, 2010.