Energy Detection Spectrum Sensing on RTL-SDR based IoT Platform

Conference Paper · October 2018

DOI: 10.1109/INFOCOMTECH.2018.8722360

1 author:

Saikat Majumder
National Institute of Technology Raipur

Energy Detection Spectrum Sensing on RTL-SDR based IoT Platform

Saikat Majumder
Department of Electronics and Telecommunication
National Institute of Technology
Raipur, India
e-mail: smajumder.etc@nitrr.ac.in

***Abstract*—Recent research trends are shifting toward Internet of Things (IoT) networks based on cognitive radio (CR) technology to provide dynamic spectrum access to numerous IoT nodes. Spectrum sensing is an essential task of CR-IoT node which allows cognitive uses to identify vacant frequency bands. In this article we propose a CR-IoT node using Raspberry Pi single board computer and an ultra low cost software defined radio called RTL-SDR. Real time energy detection spectrum sensing is performed on such a CR-IoT node and results are compared to the theory. Results show close correspondence between theoretical and experimental results.**

***Index Terms*—Spectrum sensing, RTL-SDR, Raspberry Pi, Cognitive radio, Software defined radio, GNU Radio**

I. Introduction

The primary goal of IoT is to extend internet connectivity to millions of “things” or devices through the wireless medium. But providing wireless connectivity to millions of devices is severely constrained by the shortage of frequency spectrum. The limited available spectrum calls for the requirement of a judicious spectrum allotment through dynamic spectrum access. Cognitive radio (CR) can help overcome the problem of spectrum scarcity and unpredictable nature of wireless networks [1]. Spectrum sensing constitutes an important task in cognitive radio.

In present scenario, the spectrum allocated to wireless systems are fixed. The radio frequency (RF) spectrum are controlled by the government and is licensed to firms and citizens for a large geographical area on a long term basis. With the advent of numerous wireless devices, specially in the form of IoT, the demand for frequency spectrum is expected to rise. Given the spectrum being a limited resource, static frequency allocation schemes cannot accommodate an increasing number of high data rate wireless devices. On the other hand, while most of the spectrum is allocated, it is severely underutilized. For example, a study in 2004 found that during the time of high bandwidth demand in New York city, only about 13 percent of spectrum were utilized [2].

In a CR scenario, RF spectrum is primarily available to licensed users, called *primary user* (PU) and PUs have higher priority over the spectrum allocated to them. CR users continuously ’sense’ the transmission of PUs over a specific band of frequencies on a spatio-temporal basis and utilize the spectrum when PU is idle or not transmitting. CR users are also known as *secondary users* (SU). Such a CR network is called overlay cognitive radio network. On the other hand, CR users may communicate in the network in such a way that both PUs and SUs use the same frequency band simultaneously. In such a network, total power transmitted by SUs should be sufficiently low so as to maintain total interference within a permissible limit [2].

Spectrum sensing is an integral part of CR system design. It is one of the important research issue in CR which has drawn attention of research community. Several other issues which have dominated research in CR are spectrum sharing, spectrum allocation management, secondary user’s transmission etc [3]. A spectrum sensing algorithm scans through a band of frequency to check their occupancy and presence of PUs.Its reliability determines the performance of rest of the system. Thus, in a overlay CR network, SUs must perform reliable spectrum sensing with the objective to maximize its throughput and at the same time avoiding unnecessary interference to the PUs. Two important measures of performance of spectrum sensing are probability of detection P_d and probability of false alarm P_f. P_d refers to probability of detecting PU signal when it is actually present; whereas, P_f is the probability of falsely identifying a PU when it is not actually present. Several spectrum sensing techniques has been reported in literature, namely energy detection, matched filtering, cyclostationary feature detection, generalized likelihood ratio test [4], [5]. Some recently reported techniques are spectrum sensing based on compressive sensing [6], wavelet based spectrum sensing [7] and based on bi-variate Markov chain model [8].

Recently researchers have turned their attentions to experimental implementation and verification of CR algorithms on real time hardware. Yusof et al. [9] performed real measurement of energy detection using USRP and GNU radio. In [10], using low cost SDR hardware, authors performed coordinated, distributed wideband spectrum sensing in a geographical area using energy detection spectrum sensing. Authors compared their results with results obtained with dedicated professional equipment- CRFS RFeye Node. Huang et al. [11] designed and implemented a cognitive radio platform based on USRP. Using this device they find the spectrum holes and access the most suitable one for their communication. Gurugopinath et al. [12] designed and implemented spectrum sensing for frequency hoping spread spectrum primary user using Lyrtech SDR.

In this paper we experimentally evaluate the feasibility of using low cost SDR for energy detection spectrum sensing in CR enabled IoT terminal. We prototype a CR IoT terminal using Raspberry Pi (RPi) device [13] with ultra low cost software defined radio (SDR) receivers based on RTL2832U integrated circuit (IC). Specifically, a energy detection spectrum sensing model is developed on RPi interfaced with RTL SDR using GNU Radio software [14]. The performance of this experimental system is compared to theoretical performance of energy detection spectrum sensing.

Rest of the article is as follows. Section II introduces energy detection spectrum sensing and relevant theoretical background. In Section III, setup consisting of RTL-SDR and Raspberry Pi is explained. This section also discusses the GNU Radio programming for performing the experiment. Results obtained in the experiment are presented and discussed in Section IV. Finally, Section V is conclusion.

II. Energy Detector Based Spectrum Sensing

Energy detection based spectrum sensing is the most widely used method of spectrum sensing. It is characterized by its low computational and implementation complexity. Also, this method does not require any knowledge of primary users signal. Presence of the primary user is detected by comparing the output of energy detector with a threshold which depends on the noise floor. Some of the challenges associated with this technique include selection of the threshold, inability to differentiate interference from primary users and noise, poor performance under low signal to noise ratio (SNR) conditions [15].

Let the received signal be given as

$$y(n) = s(n) + w(n), n = 1, ..., N \quad (1)$$

where $s(n)$ is the signal to be detected, $w(n)$ is noise, and n is sample index. The decision metric for energy detection is

$$M = \sum_{n=0}^{N} |y(n)|^2 \quad (2)$$

where N is the length of observation vector. Decision metric in terms of signal in frequency domain is given as $M = 1/N \sum_{n=0}^{N} |Y(n)|^2$, where $Y(n)$ is the N-point DFT of y(n). Relation in frequency domain will be utilized the measurement setup given in next section.

Spectrum sensing finds whether the measured spectrum is occupied by some transmitted signal or not. Since the part of the spectrum under measurement is either occupied or is not occupied, it can be formulated as a binary hypothesis testing problem.

$$\mathcal{H}_0 : \quad y(n) = w(n) \quad (3)$$

$$\mathcal{H}_1 : \quad y(n) = s(n) + w(n) \quad (4)$$

Quality of detection algorithm can be quantified using two performance measures; namely, the probability of detection P_d and the probability of false alarm P_f. As the name suggests, P_d is probability of detecting a signal that truly is present. On the other hand, P_f is the probability of reporting the present of primary user signal which is actually not there. Both these probabilities are defined as [15]

$$P_d = Pr(M > \lambda | \mathcal{H}_1) \quad (5)$$

$$P_f = Pr(M > \lambda | \mathcal{H}_0) \quad (6)$$

It can be observed that selection of threshold λ determines the value of two parameters P_d and P_f. Lower value of P_d results in underutilization of spectrum and should be as high as possible. One way to achieve high P_d is by decreasing threshold λ, but this also results in an increase of false alarm. Based on these definitions, probability of missing a signal is given as

$$P_m = 1 - P_d \quad (7)$$

theoretical determination of P_d and P_f required the knowledge of noise and detected signal powers. For energy detection spectrum sensing, P_d and P_f are calculated as [16]

$$P_f = Q((\lambda/(\sigma_w^2) - 1)\sqrt{N} \quad (8)$$

$$P_d = Q((\lambda/(\sigma_w^2) - \rho - 1)\sqrt{N/(2\rho + 1)}) \quad (9)$$

where, σ_w^2 is noise variance and is signal to noise ratio given by $\rho = \sigma_s^2/\sigma_w^2$. Primary user signal power is given as σ_s^2. In practice, P_f should be less than a given standard value and λ is found for the selected value of P_f. For example, according to IEEE 802.22 standard, desired P_f should be less than 0.1 and P_d should be greater than 0.9 [15].

III. Measurement Setup

A. RTL SDR and RPi based IoT Terminal

An IoT node consisting of small RPi computer and SDR dongle is shown in Fig. 1. Additional peripheral like HDMI display and keyboard is also connected for programming purpose. The SDR platform chosen for this setup is R820T based RTL SDR. It primarily consists of two chips, the Raphael Micro R820T tuner and the Realtek RTL2832U consisting of 8-bit analog to digital converter (ADC) and USB data pump. The functionality of RTL SDR is explained by the behavioural level model [17] in Fig. 2. Input signal $y(t)$ consists of the desired signal s(t) and background noise w(t) due to receiver front end in combination with antenna. RF signal is amplified by a wideband low noise amplifier (LNA) and downconverted to a low intermediate frequency (IF) f_{if}. This is done by multiplying incoming signal with locally generated carrier of frequency f_{lo} and filtering the product with a filter of center frequency f_{if}. IF signal is them amplified (automatic gain control or AGC) by variable gain amplifier (VGA) and sampled at a rate of f_s samples per second. Sampled and quantized signal are then accessed by a computer using USB.

In practice, R820T tuner uses a low IF of 3.57 MHz and down-converts $y(t)$ to signal of bandwidth about 6 MHz centered around this IF. Next stage consists of RTL2832U IC and is common to all the RTL SDR devices. RTL2832U samples the signal with its analog to digital converter (ADC) at a rate of 28.8 MHz and performs quadrature demodulation

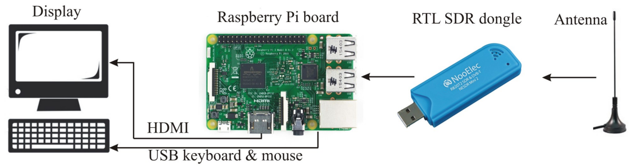

Fig. 1. An IoT node consisting of RPi and RTL SDR.

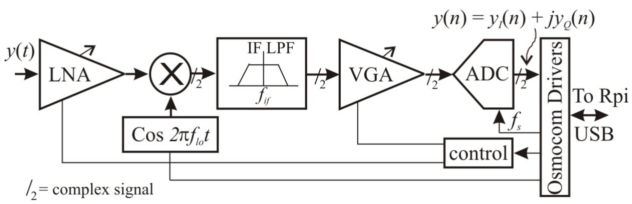

Fig. 2. Functional block diagram of RTL SDR.

to produce separate I and Q streams. Decimation is performed to reduce the sampling rate to a lower frequency and resulting signal is output using USB. Reducing the sampling rate may improve the performance of baseband processing on less powerful computers. A reduction in sampling rate also has the effect of limiting the bandwidth of the receiver. With the reduced sampling rate f_s for both I and Q channels, the signal received at the RPi has bandwidth of f_s and spans from $-f_s/2$ to $f_s/2$. Thus, RTL SDR receives and translates any signal of center frequency f_c, within the frequency range of 24 MHz to 1766 MHz, to the frequency band $(-f_s/2, f_s/2)$.

Raspberry Pi 3 Model B (RPi 3B) is the third generation RPi. RPi 3B supersedes the original RPi Model B+ and RPi Model B. At its heart, RPi 3B boards use Broadcom BCM2837 quad core system on a chip (SoC). BCM2837 is a 64-bit ARMv8 quad-code ARM Cortex-A53 processor that has NEON data engine and full entitlement to an ARMv7 software architecture. RPi 3B board with 1 GB of memory and 1.2 GHz quad-core processor provide the best performance out of all the RPi boards.

B. Spectrum Sensing using GNU Radio Companion (GRC)

The signal acquired by the RTL SDR is processed in RPi using GNU radio software environment. GNU radio is a open-source software toolkit for development using SDRs, in which all baseband signal processing is done by software. GNU radio enables generation of any modulation waveforms and their demodulation, filtering, mixing and other signal processing tasks. GNU radio toolkit provides a library of signal processing blocks and the glue to connect these blocks into a functional SDR system. The signal processing blocks are implemented in C++ and the flow graphs are constructed using Python [18]. GNU Radio Companion (GRC) is the graphical user interface (GUI) for GNU radio. This allows users to develop GNU radio program in a way similar to Matlab Simulink or National Instruments LabView. The blocks and connections work on drag and drop model. A block in GRC contains various inputs and outputs, as well as data type for the inputs and outputs. Output of one block can be connected to input of another block only if their data type are same.

A GRC implementation of energy detection spectrum sensing is shown in Fig. 3 and Fig. 4. Though, in practice, the configuration (modulation, bandwidth, data rate etc.) of PU is not required to be known in case of energy detection spectrum sensing, an example implementation of a PU is shown in Fig. 3. The primary source transmits Gaussian minimum shift keying (GMSK) modulated data packets using Universal software radio peripheral (USRP) SDR.

The sensing setup residing in SU is shown in Fig. 4. It works by applying Parseval theorem on (2), resulting in

$$M = \frac{1}{N}\sum_{n=0}^{N} |Y(n)|^2 \tag{10}$$

where, $Y(n)$ is N point discrete Fourier transform (DFT) or fast Fourier transform (FFT) of $y(n)$. Power spectrum density $S(n)$can be calculated from DFT as

$$S(n) = \frac{1}{N f_s}|Y(n)|^2 \tag{11}$$

Since, in case of RTL SDR, incoming signal is filtered to a bandwidth of f_s, the signal power available at the output of RTL-SDR source block is approximated as $P = f_s \sum_{n=1}^{N} S(n)$. The moving average (calculated over L number of samples) value of the measured power is finally displayed in 'WX GUI Number Sink' block. This method can be used to estimate the SNR of the received signal. Let, P_w be the signal power measured in the absence of PU. When measured in the presence of PU, the signal power P_t is sum of both signal power P_s and noise power P_w. SNR is then calculated as

$$\rho = \frac{P_s}{P_w} = \frac{P_t}{P_w} - 1 \tag{12}$$

The data flow processing through the CR IoT node (or SU) shown in Fig. 4 and is described as follows. RTL SDR source intercepts a signal of bandwidth f_{BW}, centered around f_c. Since the output of RTL SDR source is a continuous stream of eight bit symbols, it is converted into vectors of length $1024N$, where $N = 1, ..., 30$. 1024 N-point FFT (for calculation of DFT) is applied on each of these vectors and squared magnitude $|Y(n)|^2$ is calculated for each of them. PSD $S(n)$ is then calculated by multiplying with $1/(N f_s)$. Calculated values of PSD are saved in for later processing and calculation of decision metric M using Matlab.

IV. Results

The PU node runs GNU radio environment on Ubuntu Linux 14.04 LTS operating system. The transmitter implemented in GRC is shown in Fig. 3. The symbols are Gaussian minimum shift keying (GMSK) modulated and transmitted at center frequency $f_c = 866.1$ MHz. This frequency is relevant considering the advent of LoRa devices for IoT applications in 868 MHz band. There are eight channels in 868 MHz LoRa band, starting from 865.2 MHz to 868 MHz [19].

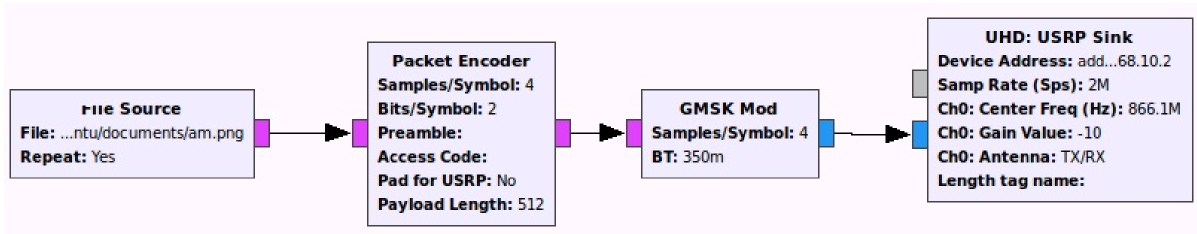

Fig. 3. GNU Radio block diagram of transmitter implemented on USRP.

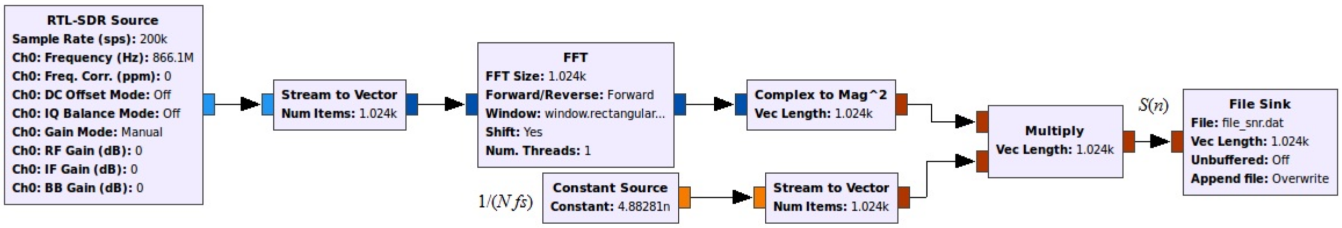

Fig. 4. GNU Radio block diagram of receiver implemented on RTL SDR.

As described earlier, the sensing setup or CR IoT node consists of RTL SDR and RPi 3B. The system model shown in Fig. 4 is implemented using GNU radio. RPi 3B is running Ubuntu MATE 15.10 operating system. Advantage of using Ubuntu MATE operating system is that most of the GNU radio libraries developed for desktop version of Ubuntu can be easily ported for RPi. In this setup, center frequency is set to $f_c = 866.1$ MHz, sampling rate to $f_s = 2 \times 10^5$ samples per second for both I and Q channels, resulting in RTL SDR bandwidth of $f_{BW} = 2 \times 10^5$. As reference, all the gains (RF, IF and BB) in RTL- SDR Source block are set to 0 dB.

A real time measurement setup consists of a desktop PC connected to USRP acts as PU transmitter and spectrum sensing system (SU) consists of RTL SDR and RPi. For the purpose of programming and observing the results, a keyboard and HDMI monitor is also connected to RPi. Screenshots of the spectrum of the received signal, in the absence and presence of signal from PU is shown in Fig. 5 and Fig. 6, respectively.

Fig. 7 shows the measurement results for probability of false alarm (P_f) plotted as function of number of samples, $N \times 1024$, in a FFT block. Three curves are plotted for different values of threshold λ, -45.39 dB, -45.40 dB, -45.41 dB. Measured noise power is around -45.1 dB. It can be seen from the figure that with increase in threshold, probability of P_f curve approaches the origin and false alarm rate decreases. Though this is desirable, it results in decrease in probability of detection and underutilization of spectrum resources. P_f also decreases with increase in N without adversely affecting spectrum utilization, but causes increase in measurement duration and computational requirement. As a reasonable compromise, the point in the curves is selected which achieves $P_f < 0.1$. This is attained for $\lambda = -45.40$ dB curve and $N = 5$. For comparison, a theoretical curve obtained by (8) is also shown for threshold value of $\lambda = -45.40$ dB. It can be seen that

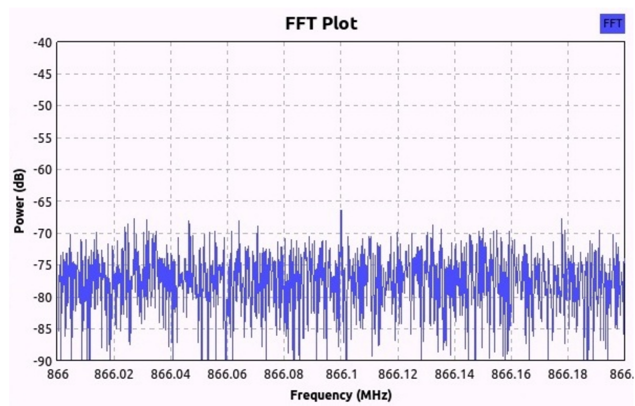

Fig. 5. Spectrum of the received signal in the absence of signal from PU.

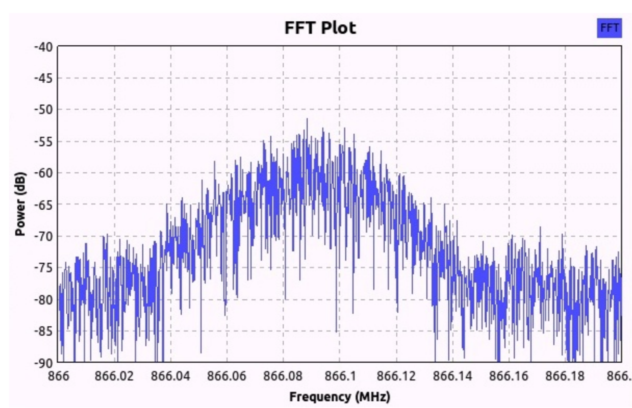

Fig. 6. Spectrum of the received signal in the presence of signal from PU.

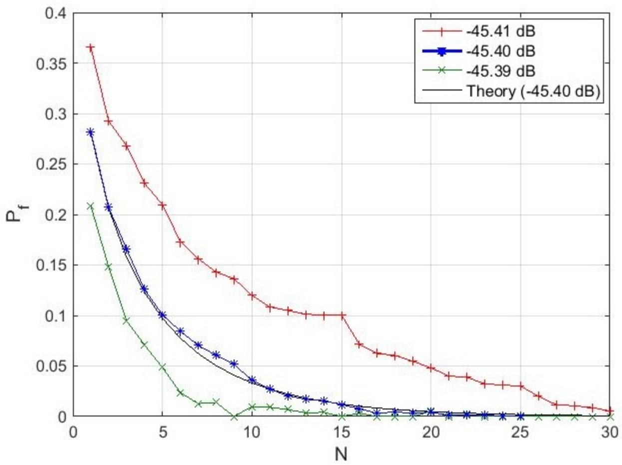

Fig. 7. Results for probability of false alarm (P_f) plotted as function of number of samples.

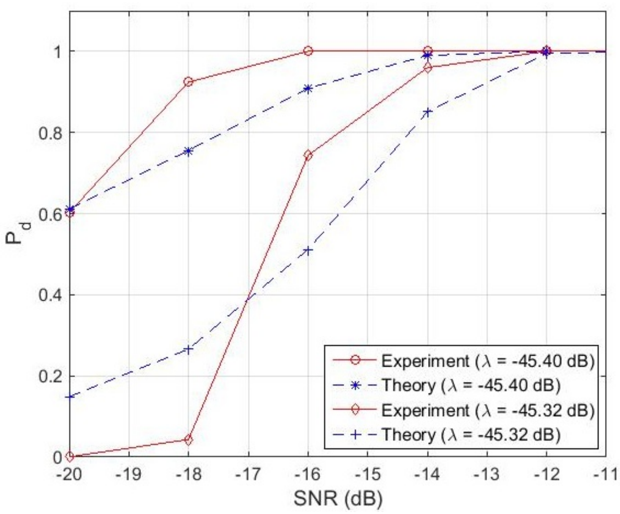

Fig. 8. Plots for probability of detection P_d as function of SNR ρ, where $N = 5$.

the measurement plot at $\lambda = -45.40$ dB closely follows theoretical plot.

In Fig. 8 and Fig. 9, experimental plots for probability of detection P_d as function of SNR ρ and is compared to theoretical plot obtained by (9). Fig. 8 shows the plot for P_d when N = 5. SNR is varied from -20 dB to -12 dB, in steps of 2 dB. Experimental and theoretical curves are plotted for threshold values of -45.40 dB and -45.32 dB. It can be seen from the figure that the experimental plot largely follows the theoretical plot. It should be noted that the curves are very sensitive to estimated value of noise power and SNR, and deviation of experimental result from theory may be attributed to it. Similar results are obtained for $N = 20$ and is shown in Fig. 9.

V. Conclusion

In this article an experimental setup for energy detection spectrum sensing was proposed. The setup was implemented

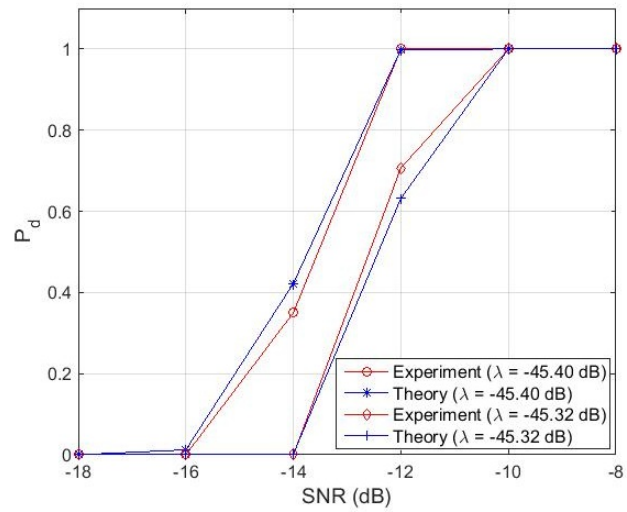

Fig. 9. Plots for probability of detection P_d as function of SNR ρ, where $N = 20$.

on Raspberry pi and RTL SDR and is suitable for IoT cognitive radio application. Software implementation was done using GNU Radio running on Ubuntu operating system. Using the experimental results obtained, empirical value of probability of detection and probability of false alarm were found to closely follow the theoretical values.

References

[1] A. Ghasemi, and E. S. Sousa, "Spectrum sensing in cognitive radio networks: requirements, challenges and design trade-offs." IEEE Communications magazine, vol. 46, no. 4, 2008. G. Eason, B. Noble, and I. N. Sneddon, "On certain integrals of Lipschitz-Hankel type involving products of Bessel functions," Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955.

[2] E. Biglieri, A. J. Goldsmith, L. J. Greenstein, N. B. Mandayam, and H. V. Poor, "Principles of cognitive radio," Cambridge University Press, 2013.

[3] S. P. Maity, S. Chatterjee, and T. Acharya, "On optimal fuzzy c-means clustering for energy efficient cooperative spectrum sensing in cognitive radio networks," Digital Signal Processing, vol. 49, pp. 104-115, 2016.

[4] M. Lpez-Bentez, and F. Casadevall, "Improved energy detection spectrum sensing for cognitive radio," IET communications, vol. 6, no. 8, 2012.

[5] M. Subhedar, and G. Birajdar, "Spectrum sensing techniques in cognitive radio networks: A survey," International Journal of Next-Generation Networks, vol. 3, no. 2, 2011.

[6] F. Zeng, C. Li, and Z. Tian, "Distributed compressive spectrum sensing in cooperative multihop cognitive networks," IEEE Journal of Selected Topics in Signal Processing, vol. 5, no. 1, 2011.

[7] Z. Tian, and G. B. Giannakis, "A wavelet approach to wideband spectrum sensing for cognitive radios," in 1st International Conference on Cognitive Radio Oriented Wireless Networks and Communications, pp. 1-5. IEEE, 2006.

[8] T. Nguyen, B. L. Mark, and Y. Ephraim, "Spectrum sensing using a hidden bivariate Markov model," IEEE Transactions on Wireless Communications, vol. 12, no. 9, 2013.

[9] K. H. Yusof, M. K. A. Rahim, K. M. Yusof, and M. E. Jalil, "Energy detection spectrum sensing measurement using GNU Radio and USRP B200 at Wi-Fi Frequency," In Theory and Applications of Applied Electromagnetics, pp. 223-230. Springer, Cham, 2015.

[10] S. Grnroos, K. Nybom, J. Bjrkqvist, J. Hallio, J. Auranen, and R. Ekman, "Distributed spectrum sensing using low cost hardware," Journal of Signal Processing Systems, vol. 83, no. 1, 2016.

[11] Z. Huang, W. Wang, and Y Zhang, ”Design and implementation of cognitive radio hardware platform based on USRP,” in IET International Conference on Communication Technology and Application (ICCTA 2011), pp. 160-164, 2011.

[12] S. Gurugopinath, R. Akula, C. R. Murthy, R. Prasanna, and B. Amrutur, ”Design and implementation of spectrum sensing for cognitive radios with a frequency-hopping primary system,” Physical Communication, vol. 17, pp. 172-184, 2015.

[13] Raspberry Pi, ”model B.” URl: https://www. raspberrypi. org/products/raspberrypi-2-model-b (2).

[14] G. N. U. Radio, ”The gnu software radio.” Available from World Wide Web: https://gnuradio. org, 2018.

[15] T. Yucek and H. Arslan, ”A survey of spectrum sensing algorithms for cognitive radio applications,” IEEE communications surveys and tutorials, vol. 11, no. 1, 2009.

[16] G. V. Chaitanya, P. Rajalakshmi, and U. B. Desai, ”Real time hardware implementable spectrum sensor for cognitive radio applications,” In International Conference on Signal Processing and Communications (SPCOM), pp. 1-5, IEEE, 2012.

[17] http://www.eas.uccs.edu/ mwickert

[18] R. Gandhiraj, and K. P. Soman, ”Modern analog and digital communication systems development using GNU Radio with USRP,” Telecommunication Systems, vol. 56, no. 3, 2014.

[19] A. S. Bharadwaj, R. Rego, and A. Chowdhury. ”IoT based solid waste management system: A conceptual approach with an architectural solution as a smart city application,” In India Conference (INDICON), 2016 IEEE Annual, pp. 1-6. IEEE, 2016.