



MASTER UNIVERSITARIO EN INGENIERÍA DE TELECOMUNICACIONES

TRABAJO FIN DE MASTER

Utilización y comparación de tecnologías de
transmisión de Imágenes por satélite de
AMSAT-EA utilizando SSTV o SSDV

Autor: José María Moyano Suarez

Director: Pedro Olmos González

Madrid

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

Utilización y comparación de tecnologías de

transmisión de Imágenes por satélite de

AMSAT-EA utilizando SSTV o SSDV

en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el

curso académico **2020/21** es de mi autoría, original e inédito y

no ha sido presentado con anterioridad a otros efectos.

El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido

tomada de otros documentos está debidamente referenciada.



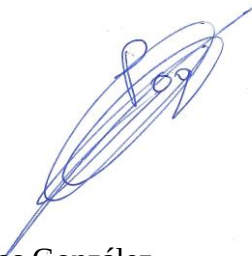
24-08-2021.
Fdo. José María Moyano.

Fdo.: José María Moyano Suarez

Fecha: 24/ 08/ 2021

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO



Fdo.: Pedro Olmos González

Fecha: 24/ 08/2021

AUTORIZACIÓN PARA LA DIGITALIZACIÓN, DEPÓSITO Y DIVULGACIÓN EN RED DE PROYECTOS FIN DE GRADO, FIN DE MÁSTER, TESINAS O MEMORIAS DE BACHILLERATO

1º. Declaración de la autoría y acreditación de la misma.

El autor D. José María Moyano Suarez

DECLARA ser el titular de los derechos de propiedad intelectual de la obra: Utilización y comparación de tecnologías de transmisión de Imágenes por satélite de AMSAT-EA utilizando SSTV o SSDV, que ésta es una obra original, y que ostenta la condición de autor en el sentido que otorga la Ley de Propiedad Intelectual.

2º. Objeto y fines de la cesión.

Con el fin de dar la máxima difusión a la obra citada a través del Repositorio institucional de la Universidad, el autor **CEDE** a la Universidad Pontificia Comillas, de forma gratuita y no exclusiva, por el máximo plazo legal y con ámbito universal, los derechos de digitalización, de archivo, de reproducción, de distribución y de comunicación pública, incluido el derecho de puesta a disposición electrónica, tal y como se describen en la Ley de Propiedad Intelectual. El derecho de transformación se cede a los únicos efectos de lo dispuesto en la letra a) del apartado siguiente.

3º. Condiciones de la cesión y acceso

Sin perjuicio de la titularidad de la obra, que sigue correspondiendo a su autor, la cesión de derechos contemplada en esta licencia habilita para:

- a) Transformarla con el fin de adaptarla a cualquier tecnología que permita incorporarla a internet y hacerla accesible; incorporar metadatos para realizar el registro de la obra e incorporar “marcas de agua” o cualquier otro sistema de seguridad o de protección.
- b) Reproducirla en un soporte digital para su incorporación a una base de datos electrónica, incluyendo el derecho de reproducir y almacenar la obra en servidores, a los efectos de garantizar su seguridad, conservación y preservar el formato.
- c) Comunicarla, por defecto, a través de un archivo institucional abierto, accesible de modo libre y gratuito a través de internet.
- d) Cualquier otra forma de acceso (restringido, embargado, cerrado) deberá solicitarse expresamente y obedecer a causas justificadas.
- e) Asignar por defecto a estos trabajos una licencia Creative Commons.
- f) Asignar por defecto a estos trabajos un HANDLE (URL *persistente*).

4º. Derechos del autor.

El autor, en tanto que titular de una obra tiene derecho a:

- a) Que la Universidad identifique claramente su nombre como autor de la misma
- b) Comunicar y dar publicidad a la obra en la versión que ceda y en otras posteriores a través de cualquier medio.
- c) Solicitar la retirada de la obra del repositorio por causa justificada.
- d) Recibir notificación fehaciente de cualquier reclamación que puedan formular terceras personas en relación con la obra y, en particular, de reclamaciones relativas a los derechos de propiedad intelectual sobre ella.

5º. Deberes del autor.

El autor se compromete a:

- a) Garantizar que el compromiso que adquiere mediante el presente escrito no infringe ningún derecho de terceros, ya sean de propiedad industrial, intelectual o cualquier otro.
- b) Garantizar que el contenido de las obras no atenta contra los derechos al honor, a la intimidad y a la imagen de terceros.
- c) Asumir toda reclamación o responsabilidad, incluyendo las indemnizaciones por daños, que

podieran ejercitarse contra la Universidad por terceros que vieran infringidos sus derechos e intereses a causa de la cesión.

- d) Asumir la responsabilidad en el caso de que las instituciones fueran condenadas por infracción de derechos derivada de las obras objeto de la cesión.

6º. Fines y funcionamiento del Repositorio Institucional.

La obra se pondrá a disposición de los usuarios para que hagan de ella un uso justo y respetuoso con los derechos del autor, según lo permitido por la legislación aplicable, y con fines de estudio, investigación, o cualquier otro fin lícito. Con dicha finalidad, la Universidad asume los siguientes deberes y se reserva las siguientes facultades:

- La Universidad informará a los usuarios del archivo sobre los usos permitidos, y no garantiza ni asume responsabilidad alguna por otras formas en que los usuarios hagan un uso posterior de las obras no conforme con la legislación vigente. El uso posterior, más allá de la copia privada, requerirá que se cite la fuente y se reconozca la autoría, que no se obtenga beneficio comercial, y que no se realicen obras derivadas.
- La Universidad no revisará el contenido de las obras, que en todo caso permanecerá bajo la responsabilidad exclusiva del autor y no estará obligada a ejercitar acciones legales en nombre del autor en el supuesto de infracciones a derechos de propiedad intelectual derivados del depósito y archivo de las obras. El autor renuncia a cualquier reclamación frente a la Universidad por las formas no ajustadas a la legislación vigente en que los usuarios hagan uso de las obras.
- La Universidad adoptará las medidas necesarias para la preservación de la obra en un futuro.
- La Universidad se reserva la facultad de retirar la obra, previa notificación al autor, en supuestos suficientemente justificados, o en caso de reclamaciones de terceros.

Madrid, a 24 de agosto de 2021

ACEPTA

Fdo José María Moyano Suarez



24-08-2021.
Fdo José María Moyano.

Motivos para solicitar el acceso restringido, cerrado o embargado del trabajo en el Repositorio Institucional:



MASTER UNIVERSITARIO EN INGENIERÍA DE TELECOMUNICACIONES

TRABAJO FIN DE MASTER

Utilización y comparación de tecnologías de transmisión de Imágenes por satélite de AMSAT-EA utilizando SSTV o SSDV

Autor: José María Moyano Suarez

Director: Pedro Olmos González

Madrid

Agradecimientos

Agradezco la labor y la dedicación de mi director y tutor del proyecto, D. Pedro Olmos González.

Agradecer a mi familia y amigos, debido a que sin ellos esto no me hubiera sido posible haber llegado donde me encuentro ahora mismo.

Gracias a todos.

UTILIZACIÓN Y COMPARACIÓN DE TECNOLOGÍAS DE TRANSMISIÓN DE IMÁGENES POR SATÉLITE DE AMSAT-EA UTILIZANDO SSTV O SSDV

Autor: Moyano Suarez, José María.

Director: Olmos González, Pedro.

Entidad Colaboradora: AMSAT

RESUMEN DEL PROYECTO

Palabras clave: SSDV, SSTV, AMSAT EA, satélites, radioaficionados, transmisión, imágenes, JPEG

1. Introducción

La Asociación AMSAT-EA[1], colaboradora de este proyecto, tiene interés en la transmisión de imágenes desde un satélite, ellos utilizan actualmente un método llamado SSTV (Slow Scan Televisión). Este consiste en transmitir imágenes líneas por la línea de forma analógica. Sus principales problemas son que no consigue una SNR (señal a ruido ratio) suficiente para que la calidad de la foto sea buena y se pueda obtener información relevante de las mismas. Además de utilizar una modulación muy ineficiente para el SNR disponible del enlace. En la siguiente ilustración podemos comprobar las pruebas realizadas sobre SSTV previamente al lanzamiento del satélite Hades de AMSAT EA.



Ilustración 1 -. Imagen transmitida SSTV sobre SNR - Amsat EA

SSDV (Slow Scan Digital Video)[2] es una forma digital paquetizada de SSTV (Slow Scan Televisión). Fue desarrollada por unos radioaficionados en Inglaterra y consiste en realizar lo mismo que hace SSTV, pero con la diferencia que, en vez de enviar líneas de la imagen, se envían frames codificados en formato JPEG. Este sistema consiste en la paquetiacion de la imagen en el siguiente formato, véase en la ilustración 2. Depende de la resolución y de

la calidad de la imagen y si se quiere implementar FEC o no, tendremos un diferente numero de paquetes que serán transmitidos.

Offset	Name	Size	Description
0	Sync Byte	1	0x55 - Puede ir precedido de uno o más bytes de sincronización
1	Packet Type	1	0x66 - Modo normal (paquete de 224 bytes + 32 bytes FEC) 0x67 - Modo sin FEC (paquete de 256 bytes)
2	Callsign	4	Indicativo codificado en base-40. Hasta 6 dígitos
6	Image ID	1	imagen
7	Packet ID	2	endian)
9	Width	1	Ancho de la imagen en bloques MCU (píxeles / 16) 0 = No válido
10	Height	1	Altura de la imagen en bloques MCU (píxeles / 16) 0 = No válido
11	Flags	1	Bandera EOI (1 = Último paquete), xx = Modo de submuestreo (0 = 2×2, 1 = 1×2, 2 = 2×1, 3 = 1×1)
12	MCU offset	1	carga útil, o 0xFF si no hay ninguno presente
13	MCU index	2	0xFFFF si no hay ninguno presente
15	Payload	205 / 237	Datos de la carga útil
220 / 252	Checksum	4	CRC de 32 bits
224	FEC	32	modo normal (0x66)

Ilustración 2 -. Formato paquete SSDV[2]

Además, los creadores argumentaban que puede utilizar cualquier modo digital que pueda transportar texto o datos, debido a que la implementación actual se limita a RTTY (Radio Teletipo) de 8 bits haciendo posible nuevas características de transmisión. Estos estaban en lo cierto, debido a que en 2018 se realizó la primera transmisión de SSDV sobre un satélite de radioaficionados. Por lo que SSDV es idóneo para este problema, ya que cuenta con una estandarización y una resolución de problemas avanzada, cubriendo en profundidad factores esenciales como son el formato del paquete, formato de la imagen, corrección de errores, recuperación de paquetes perdidos.

2. Definición del Proyecto

Los satélites de AMSAT siguen contando con tecnologías analógicas a la hora de realizar el procesamiento de señales y su respectiva transmisión de imágenes, como SSTV. Este proyecto busca adquirir información de transmisión digital por imágenes SSDV en satélites de radioaficionados, para simular en banda base cuales serian las mejores modulaciones que hicieran aumentar el bit rate considerablemente, además de la implementación de FEC en el sistema para la corrección de errores . Teniendo así mejor calidad en las imágenes sobre el ancho de banda y la SNR disponibles.

Primero, se ha estudiado como las características de transmisión del Modo SSTV actual implementado y se evaluado que este no es del todo óptimo, pero aun así se ha se decidió incluir la modulación en el sistema de simulación de transmisión de imágenes sobre SSDV, siendo FSK la utilizada a 50 bps.

Mas tarde se estudio en profundidad las primeras transmisiones SSDV por satélites de radioaficionados , JY1SAT[3] y la misión DSWLP[4] china, gracias a los módulos de recepción encontrados en la librería de gnu-radio, gr-satellites[5]. Pudiendo decodificar las transmisiones de estas misiones espaciales :

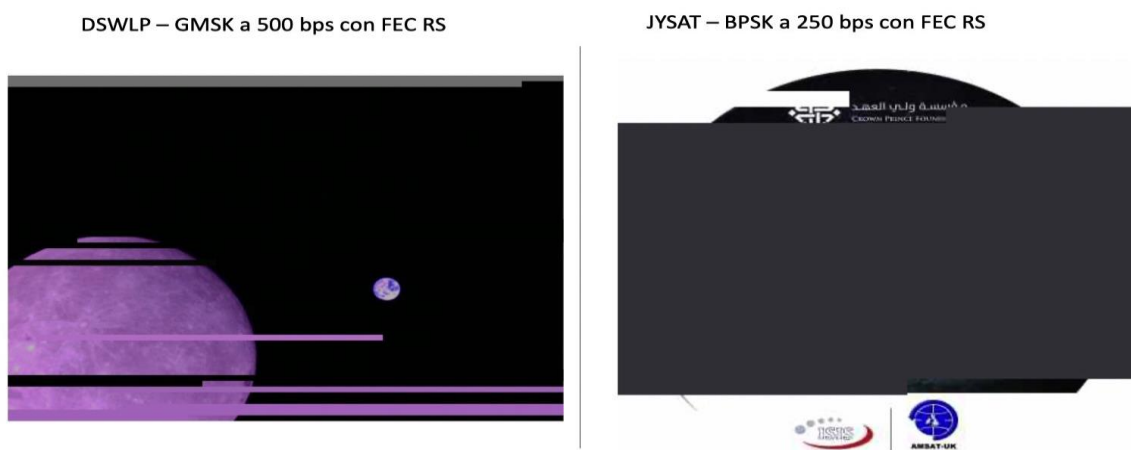


Ilustración 3 -. Imágenes SSDV recibidas procesada por gr-satellites

De este estudio concluimos que las primeras transmisiones SSDV fueron totalmente un éxito, comprobando que estos paquetes SSDV se recibían correctamente, gracias a estudio de las correlaciones y longitud de los paquetes de telemetría, solo realizado para DSWLP. Después, se realizo un análisis del sistema de recepción de telemetría e imágenes de los satélites concluyendo que sería mas optimo incluir una modulación digital BPSK para el sistema

desarrollado, además de ser recomendada por los creadores del modulo SSDV implementado en estas transmisiones.

3. Descripción del sistema

El sistema desarrollado consta de un transmisor y un receptor de imágenes SSDV. Para los módulos de codificación/decodificación, se ha utilizado el modulo SSDV mencionado previamente utilizado por misiones espaciales previas. Los bloques de modulación/demodulación han sido desarrollado en Python 3, con la ayuda de las librerías numpy (tratamiento de arrays), scipy(tratamiento de señales, filtros, FFT). Habiéndose desarrollado los siguientes módulos de las siguientes modulaciones :

- BPSK-QPSK, modulación en fase
- BFSK y QFSK., modulación en frecuencia.

Estos módulos necesitan los siguientes parámetros para la modulación/demodulación :

- frecuencia de muestreo, f_s
- Bit rate, bit_rate
- T_b , periodo de bit , $T_b = 1/bit_rate$
- Frecuencia de la portadora, f_c
- SNR con un sistema AGWN

Véase un esquema del orden de los bloques del sistema desarrollado :

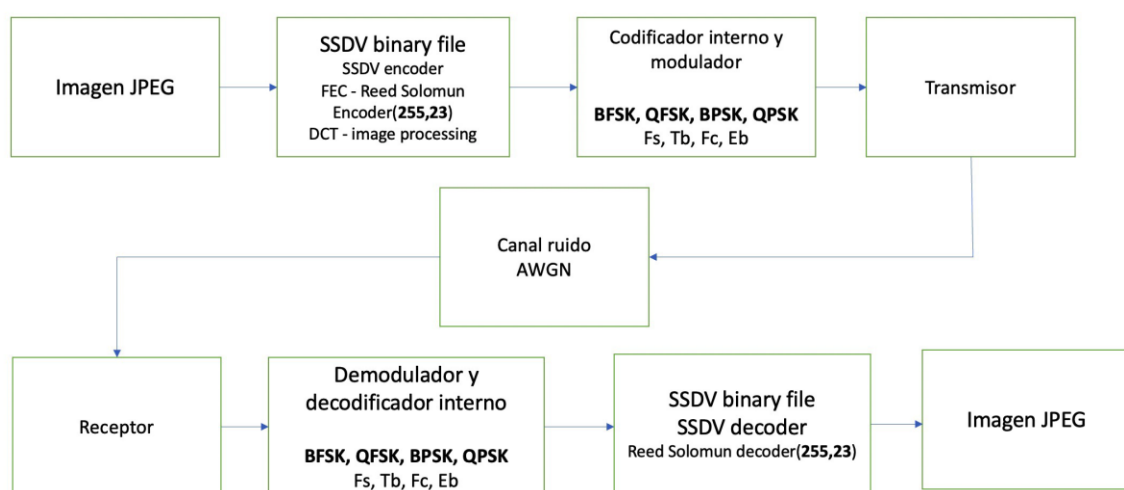


Ilustración 4 – Sistema desarrollado para la trasmisión/recepción de imágenes SSDV en banda base

Una vez hemos explicado como funciona el sistema en su totalidad, se han definido los parámetros de transmisión del trasmisor y recepción con los que ha sido simulado. Se van a

simular numerosos escenarios de Bit rate para poder debido a que como se ha comentado en apartados anteriores el sistema SSTV trasmitía en 50 bps, el satélite JY1SAT en 250 Bps y la misión DSWLP en 500 Bps. Por lo que simulará a los siguientes bits rates : 50, 100, 250, 500, 1000. Luego, para las modulaciones PSK y FSK, se utilizará una frecuencia de portadora de 1510 Hz, debido a que se encuentra justo en medio de la banda de transmisión SSTV además de ser optima para este tipo de modulación. Además, la banda en digital empieza en 300 Hz, por lo que la elección de este parámetro cumple con ese requisito.

Modulation	Bit rate [bps]	Band Width [Hz]	Sample frequency [Hz]	Carrier frequency [Hz]
BPSK	50 250, 500,1000	50 ,100, 250, 500,1000	48000	1510
QPSK	50,250, 500, 1000	50 ,100, 250, 500,1000	48000	1510
BFSK	50, 250, 500, 1000	100,500,1000,2000	48000	1510
QFSK	50, 250, 500, 1000	100,500,1000,2000	48000	1510

Ilustración 5 – Parametros Modulacion/Demodulacion del Sistema

4. Resultados

Lo primero se analiza los resultados de BER se obtuvieron realizando simulaciones de 3 frames SSDV transmitidos por un canal AWGN. La BER se calculó comparando el conjunto de muestras recibidas y enviadas.

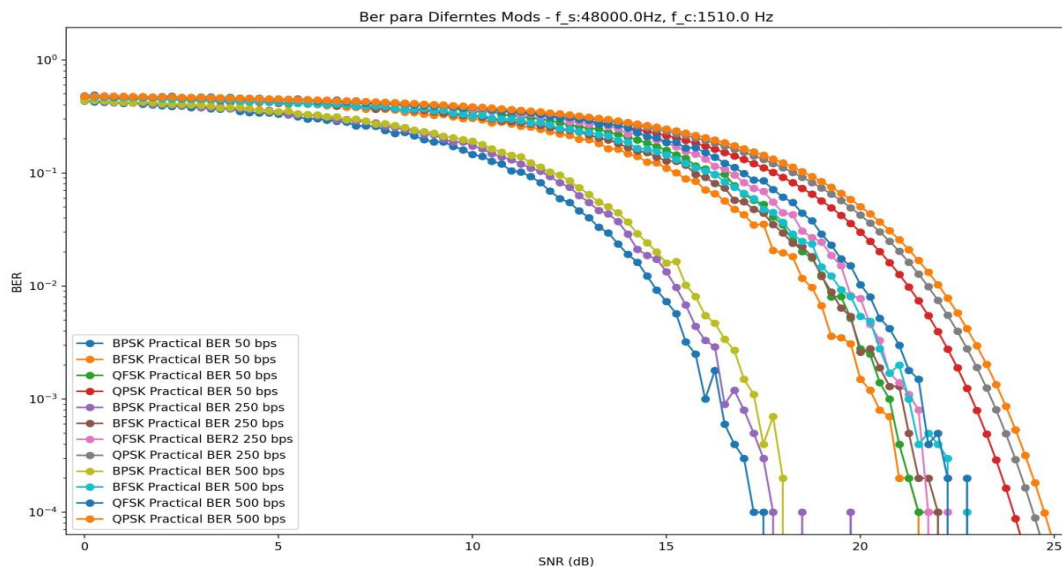


Ilustración 6 – BER Comparativa de las modulaciones BPSK, QPSK, BFSK y QFSK

Como se puede comprobar y se sabia de antemano, la modulación BPSK es la que mejores resultados frente a errores consigue, esto se debe a que la potencia de todos los símbolos es la misma y siendo la mas óptima desde el punto de vista de protección frente a errores. Comparándola con QPSK, esta se suele usar para duplicar la velocidad de datos, obteniendo la mitad de potencia por bit, por lo tanto, la mitad de SNR, por lo tanto, una mayor tasa de error de bit. Con respecto a FSK, se sabia que no es una modulación optima frente al ruido, por lo que se esperaba una peor BER comparada con BPSK. Pero si se hace un análisis con QPSK, se ve que incluso obtiene los mismos resultados. QFSK ha sido la peor de todas en términos de BER esto se debe a lo explicado anteriormente, si ya se esperaba que FSK no fuera tan optima debido a su sensibilidad frente al ruido si duplicamos la velocidad de transmisión obtendremos la mitad de potencia por bit.

En términos de decodificación de la imagen JPEG transmitida la mejor ha sido BPSK, para los diferentes SNR 5,10,15 y las mejores velocidades de transmisión han sido 50 y 250 bps. Las demás modulaciones han obtenido peores resultados según los paquetes decodificados. Si lo comparamos con la ilustración 1, de la transmisión de imágenes sobre SSTV, podemos decir que SSDV obtiene mejores resultados tanto en velocidades de transmisión a

implementar como a partir de los 10 dB de SNR. Véase en la siguiente ilustración la figura que muestra los resultados de la simulación del sistema de transmisión y recepción de imágenes en SSDV:

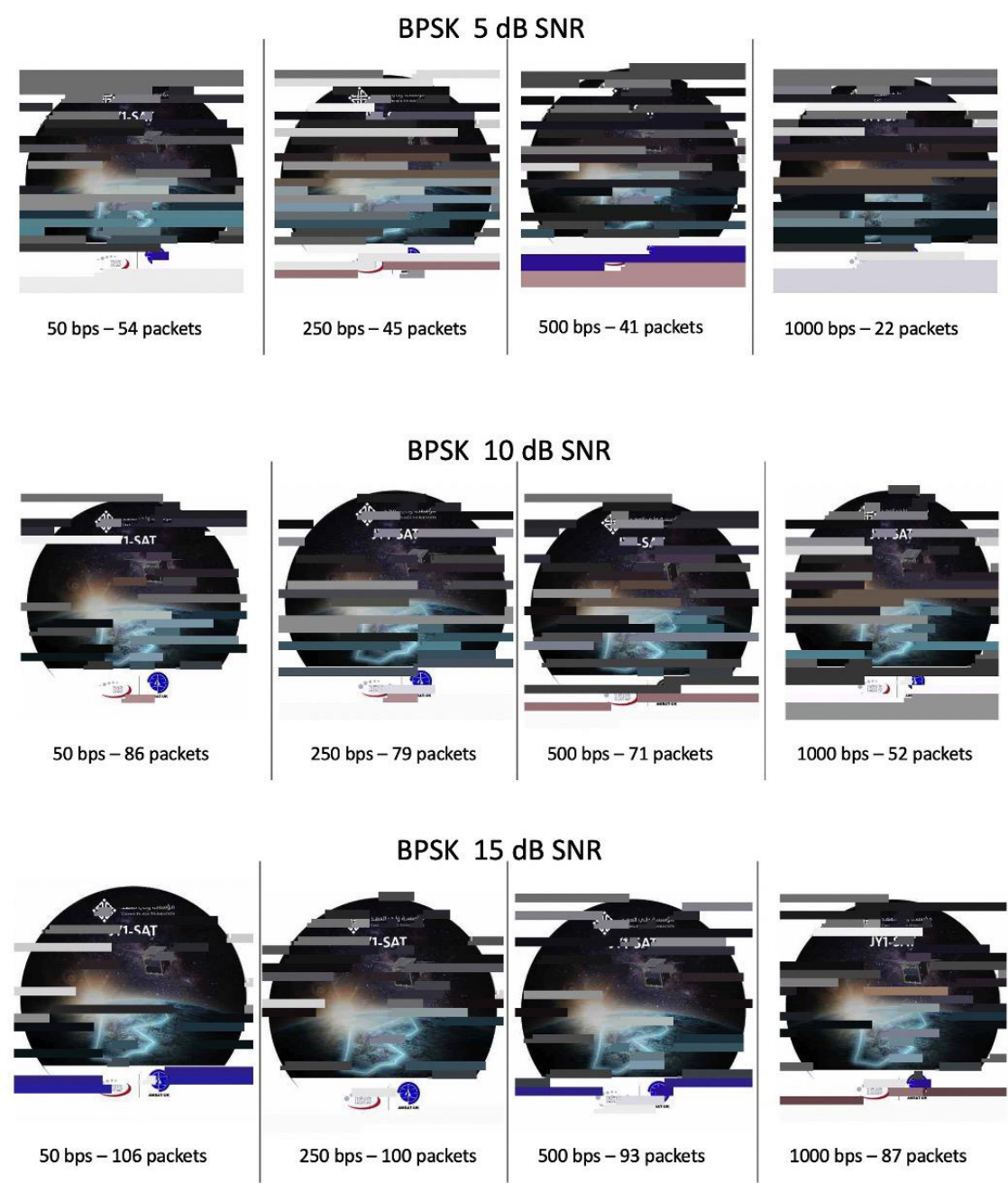


Ilustración 7– Imágenes recibidas del sistema para BPSK

5. Conclusiones

El objetivo de este proyecto era realizar un estudio de la tecnologías SSTV y SSDV en profundidad y desarrollar una forma de poder implementar SSDV para la transmisión de imágenes desde un satélite de radioaficionado. Se consiguió desarrollar un sistema de transmisión de imágenes con el Modulo SSDV que simularía esta transmisión en banda base.

Los resultados de la simulación de transmisión de imágenes en SSDV nos ayudan a poder saber si modulaciones BPSK, QPSK, BFSK y QFSK para diferentes bit rates funcionan correctamente. Concluyendo que BPSK a 50 y 250 bps es la modulación mas óptima para este tipo de transmisión. También se recomienda utilizar BFSK y QPSK a los mismos bit rates mencionados previamente, para escenarios con mas SNR, pero se conseguirán peores resultados que en BPSK.

Por lo que se recomienda utilizar BPSK a 250 bps, debido a que los resultados fueron muy favorables y además el sistema de transmisión de JY1SAT implementa la misma, pudiendo replicar el sistema que se encuentra en gr-satellites de forma sencilla, debido a que este modulo es de licencia MIT.

6. Referencias

- [1] *AMSAT EA - Satélites activos*. AMSAT EA - Radio Amateur Satellite Corporation Spain. (n.d.). <https://www.amsat-ea.org/sat%C3%A9lites-activos/>.
- [2] *UKHAS wiki. guides:ssdv* [UKHAS Wiki]. (n.d.). <https://ukhas.org.uk/guides:ssdv>.
- [3] *JY1SAT launch information and dashboard*. AMSAT. (2020, March 13). <https://amsat-uk.org/2018/11/16/jy1sat-cubesat-information/>.
- [4] *Longjiang-2 / lo-94: Journey to the moon*. AMSAT. (2020, June 6). <https://amsat-uk.org/2020/06/06/longjiang-2-lo-94-journey-to-the-moon/>.
- [5] Daniestevez. (n.d.). *Daniestevez/Gr-Satellites: GNU radio decoders for several amateur satellites*. GitHub. <https://github.com/daniestevez/gr-satellites>.

USE AND COMPARISON OF SATELLITE IMAGERY

TRANSMISSION OF SATELLITE IMAGES FROM AMSAT-EA

USING AMSAT-EA USING SSTV OR SSDV

Author: Moyano Suarez, José María .

Supervisor: Olmos Gonzalez, Pedro.

Collaborating Entity: ICAI

Keywords: SSDV, SSTV, AMSAT EA, satellites, radio ham, transmission, images, JPEG

1. Introduction

The AMSAT-EA Association [1], a partner in this project, is interested in the transmission of images from a satellite, currently using a method called SSTV (Slow Scan Television). This consists of transmitting images line by line in analog form. Its main problems are that it does not achieve a sufficient SNR (signal to noise ratio) for the quality of the picture to be good enough to obtain relevant information from them. In addition to using a very inefficient modulation for the available SNR of the link. In the following illustration we can check the tests performed on SSTV prior to the launch of AMSAT EA's Hades satellite.

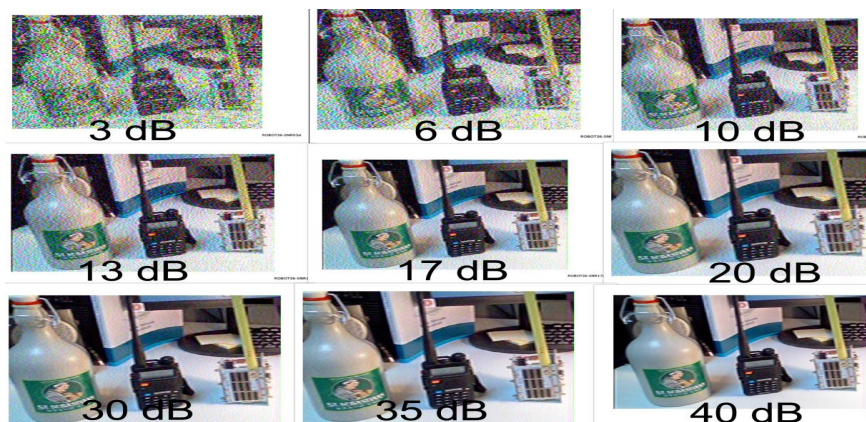


Illustration 5 -. Image transmitted SSTV over SNR - Amsat EA

SSDV (Slow Scan Digital Video)[2] is a packaged digital form of SSTV (Slow Scan Television). It was developed by radio amateurs in England and consists in doing the same as SSTV, but with the difference that, instead of sending lines of the image, it sends frames encoded in JPEG format. This system consists of packeting the image in the following format, see Figure 2. Depending on the resolution and quality of the image and whether FEC is implemented or not, we will have a different number of packets to be transmitted.

Offset	Name	Size	Description
0	Sync Byte	1	0x55 - May be preceded by one or more sync bytes
1	Packet Type	1	0x66 - Normal mode (224 byte packet + 32 byte FEC) 0x67 - No-FEC mode (256 byte packet)
2	Callsign	4	Base-40 encoded callsign. Up to 6 digits
6	Image ID	1	Normally beginning at 0 and incremented by 1 for each new image
7	Packet ID	2	The packet number, beginning at 0 for each new image (big endian)
9	Width	1	Width of the image in MCU blocks (pixels / 16) 0 = Invalid
10	Height	1	Height of the image in MCU blocks (pixels / 16) 0 = Invalid
11	Flags	1	00qqqexx: 00 = Reserved, qq = JPEG quality level (0-7 XOR 4), e = EOI flag (1 = Last Packet), xx = Subsampling Mode (0 = 2×2, 1 = 1×2, 2 = 2×1, 3 = 1×1)
12	MCU offset	1	Offset in bytes to the beginning of the first MCU block in the payload, or 0xFF if none present
13	MCU index	2	The number of the MCU pointed to by the offset above (big endian), or 0xFFFF if none present
15	Payload	205 / 237	Payload data
220 / 252	Checksum	4	32-bit CRC
224	FEC	32	Reed-Solomon forward error correction data. Normal mode only (0x66)

Illustration 6 -. SSDV packet format [2]

In addition, the creators argued that it can use any digital mode that can carry text or data, because the current implementation is limited to 8-bit RTTY (Radio Teletype) making new transmission features possible. They were right, because in 2018 the first transmission of SSDV over an amateur radio satellite took place. Therefore, SSDV is ideally suited for this problem, as it has advanced standardization and troubleshooting, covering in depth essential factors such as packet format, image format, error correction and lost packet recovery.

2. Project definition

AMSAT satellites still rely on analog technologies when processing signals and their respective transmission of images, such as SSTV. This project seeks to acquire information of digital transmission by SSDV images in amateur radio satellites, to simulate in base band which would be the best modulations that would increase the bit rate considerably, in addition to the implementation of FEC in the system for error correction. Thus, having better quality images over the available bandwidth and SNR.

First, we studied how the transmission characteristics of the current SSTV mode implemented and evaluated that it is not quite optimal, but still it was decided to include the modulation in the simulation system for image transmission over SSDV, being FSK the one used at 50 bps.

Later, the first SSDV transmissions by amateur radio satellites, JY1SAT[3] and the Chinese DSWLP[4] mission, were studied, thanks to the reception modules found in the gnu-radio library, gr-satellites [5]. Being able to decode the transmissions of these space missions:

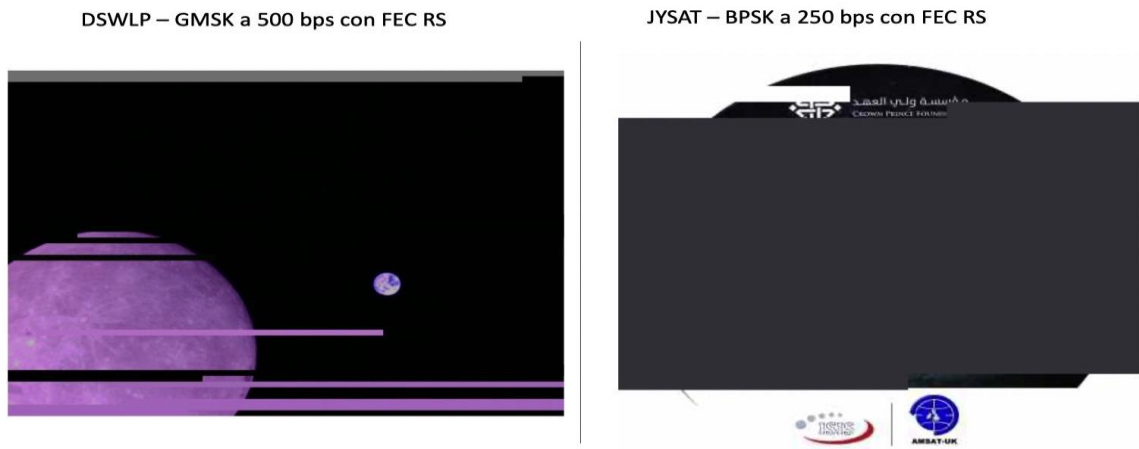


Illustration 3 -. SSDV images processed and decoded by gr-satellites

From this study we concluded that the first SSDV transmissions were totally successful, proving that these SSDV packets were received correctly, thanks to a study of the correlations and length of the telemetry packets, only carried out for DSWLP. Afterwards, an analysis of the telemetry and satellite image reception system was performed, concluding that it would be more optimal to include a BPSK digital modulation for the developed system, as well as being recommended by the creators of the SSDV module implemented in these transmissions.

3. System design

The developed system consists of a transmitter and a receiver for SSDV images. For the encoding/decoding modules, we have used the previously mentioned SSDV module used by previous space missions. The modulation/demodulation blocks have been developed in Python 3, with the help of the numpy (array processing), scipy (signal processing, filters, FFT) libraries. The following modules have been developed for the following modulations:

- BPSK-QPSK, in-phase modulation
- BFSK and QFSK, frequency modulation.

These modules need the following parameters for modulation/demodulation :

- sample rate, f_s
- bit rate, bit_rate
- T_b , bit period , $T_b = 1/\text{bit_rate}$
- Carrier frequency, f_c
- SNR with an AGWN system

See a schematic of the block order of the developed system :

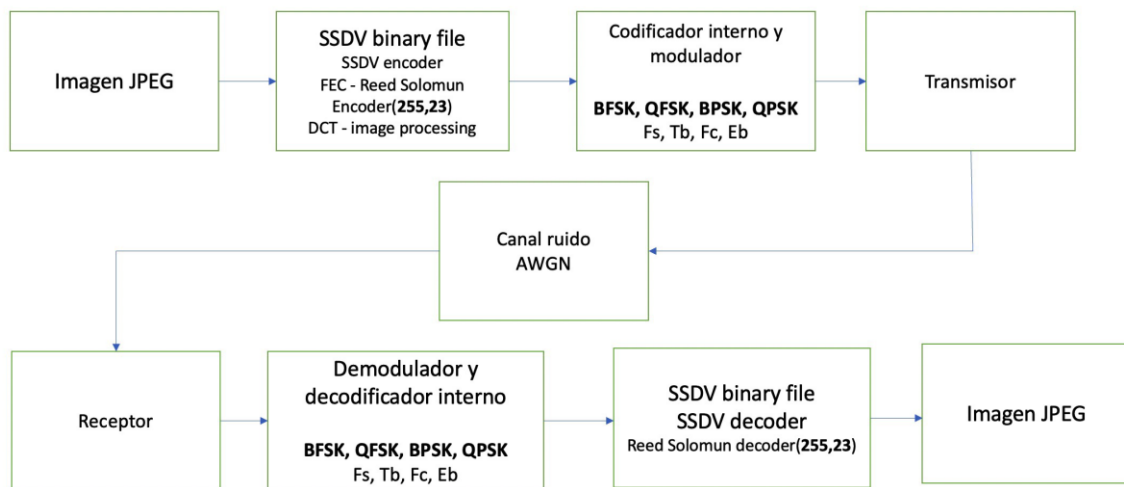


Illustration 4 – System developed for the transmission/reception of SSDV images in baseband

Once we have explained how the system works in its totality, we have defined the transmission parameters of the transmitter and the receiver with which was simulated. Numerous bit rate scenarios will be simulated to be able to simulate the system because, as mentioned in previous sections, the SSTV system transmits at 50 bps, the JY1SAT satellite at 250 Bps and the DSWLP mission at 500 Bps. Therefore, it has been simulated at the following bit rates: 50, 100, 250, 500, 1000. Then, for the PSK and FSK modulations, a carrier frequency of 1510 Hz was used, because it is right in the middle of the SSTV transmission band, and it is optimal for this type of modulation. In addition, the digital band starts at 300 Hz, so the choice of this parameter meets this requirement.

Modulation	Bit rate [bps]	Band Width [Hz]	Sample frequency [Hz]	Carrier frequency [Hz]
BPSK	50 , 250, 500,1000	50 ,100, 250, 500,1000	48000	1510
QPSK	50,250, 500, 1000	50 ,100, 250, 500,1000	48000	1510
BFSK	50, 250, 500, 1000	100,500,1000,2000	48000	1510
QFSK	50, 250, 500, 1000	100,500,1000,2000	48000	1510

Illustration 5 – . System Parameters and Modulation/Demodulation parameters

4. Results

BER results were obtained by performing simulations of 3 SSDV frames transmitted over an AWGN channel. The BER was calculated by comparing the set of received and sent samples.

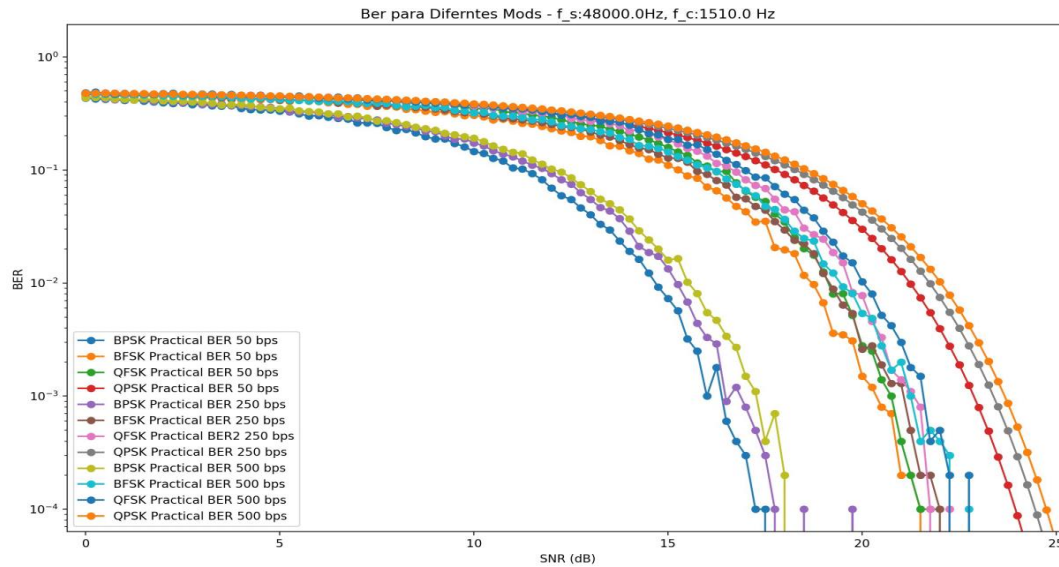


Illustration 6 – . BER Comparison of BPSK, QPSK, BFSK and QFSK modulations

As can be proven and was known in advance, BPSK modulation is the one that achieves the best results against errors, this is because the power of all symbols is the same and being the most optimal from the point of view of error protection. Compared to QPSK, this is usually used to double the data rate, obtaining half the power per bit, therefore, half the SNR, therefore, a higher bit error rate. Regarding FSK, it was known that it is not an optimal modulation against noise, so it was expected a worse BER compared to BPSK. But if you do an analysis with QPSK, you see that it even gets the same results. QFSK has been the worst of all in terms of SEE this is due to what was explained above, if it was expected that FSK was not so optimal due to its sensitivity to noise if we double the transmission rate we will get half the power per bit.

In terms of decoding of the transmitted JPEG image the best was BPSK, for the different SNR 5,10,15 and the best transmission rates were 50 and 250 bps. The other modulations obtained worse results according to the decoded packets. If we compare it with Figure 1 of the picture transmission over SSTV, we can say that SSDV obtains better results both in transmission rates to be implemented and from 10 dB of SNR. See the figure below showing the results of the simulation of the image transmission and reception system in SSDV:

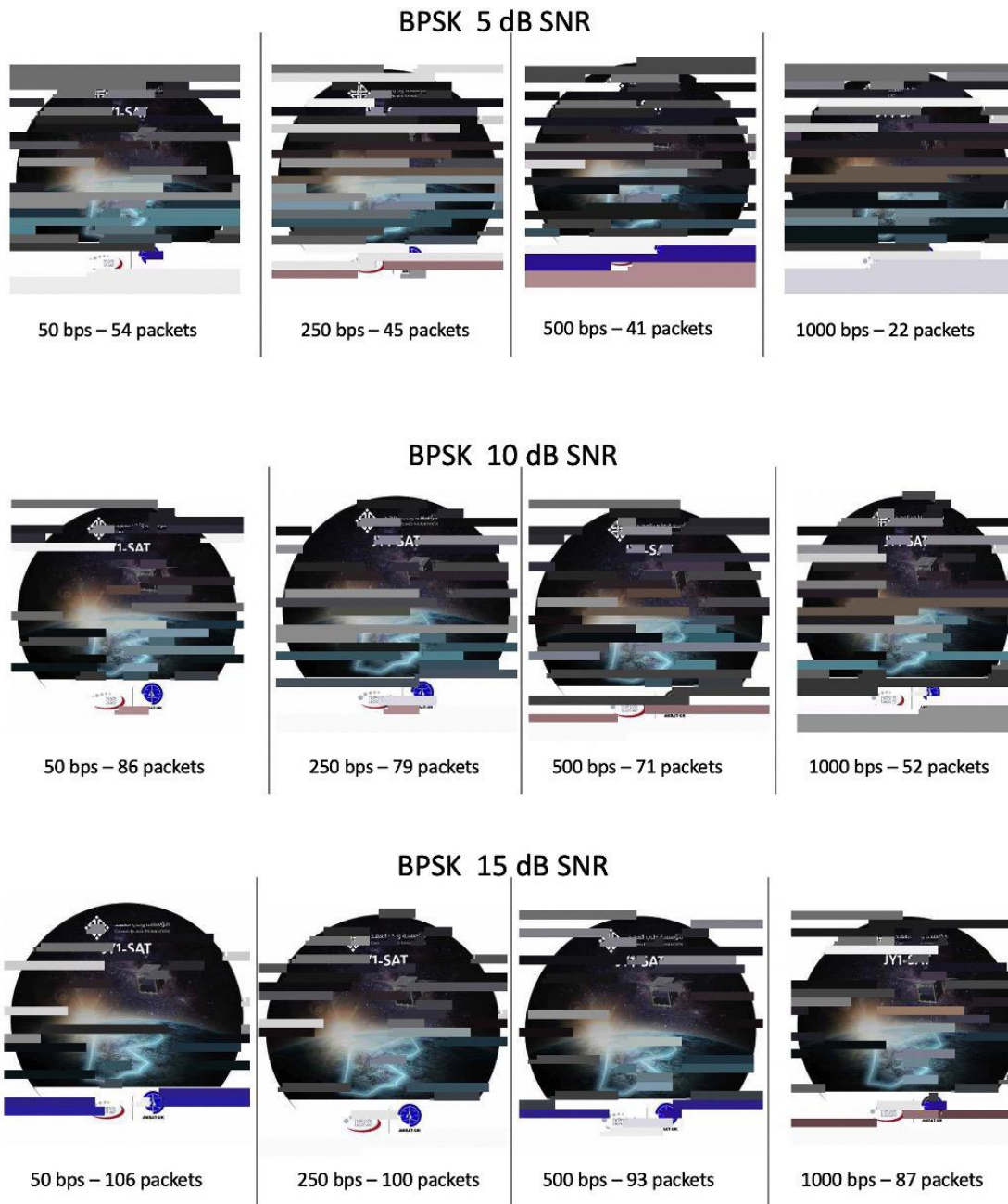


Illustration 7 – . Images received from the system for BPSK

5. Conclusions

The objective of this project was to carry out an in-depth study of SSTV and SSDV technologies and to develop a way to implement SSDV for the transmission of images from a ham satellite. We were able to develop an image transmission system with the SSDV module that would simulate this transmission in baseband.

The results of the simulation of the image transmission in SSDV help us to know if BPSK, QPSK, BFSK and QFSK modulations for different bit rates work correctly. We conclude that BPSK at 50 and 250 bps is the most optimal modulation for this type of transmission. It is also recommended to use BFSK and QPSK at the same bit rates previously mentioned, for scenarios with more SNR, but the results will be worse than in BPSK.

Therefore, it is recommended to use BPSK at 250 bps, because the results were very favorable and the JY1SAT transmission system implements the same, being able to replicate the system found in gr-satellites in a simple way, also this module is MIT licensed.

6. References

- [1] *AMSAT EA - Satélites activos*. AMSAT EA - Radio Amateur Satellite Corporation Spain. (n.d.). <https://www.amsat-ea.org/sat%C3%A9lites-activos/>.
- [2] *UKHAS wiki. guides:ssdv* [UKHAS Wiki]. (n.d.). <https://ukhas.org.uk/guides:ssdv>.
- [3] *JY1SAT launch information and dashboard*. AMSAT. (2020, March 13). <https://amsat-uk.org/2018/11/16/jy1sat-cubesat-information/>.
- [4] *Longjiang-2 / lo-94: Journey to the moon*. AMSAT. (2020, June 6). <https://amsat-uk.org/2020/06/06/longjiang-2-lo-94-journey-to-the-moon/>.
- [5] Daniestevez. (n.d.). *Daniestevez/Gr-Satellites: GNU radio decoders for several amateur satellites*. GitHub. <https://github.com/daniestevez/gr-satellites>.

Índice de la memoria

1. Introducción	8
1.1 Motivación	8
2. Descripción de las Tecnologías.....	11
2.1 SSTV en Satélites de Radioaficionados	11
2.1.1 Ancho de Banda.....	11
2.1.2 Modulaciones analógicas para SSTV.....	11
2.1.3 Frecuencias de transmisión.....	12
2.1.4 Modos de transmisión.....	14
2.2 SSTV Digital	15
2.2.1 Básicos de la comunicación digital.....	15
2.2.2 Detección y corrección de errores	16
2.2.3 Compresión de datos	18
2.3 Modulo SSDV utilizado	22
2.3.1 Definición	22
2.3.2 Formato del paquete.....	23
2.3.3 Formato de la Imagen	23
2.3.4 Corrección de Errores.....	24
2.3.5 Modulación Utilizada	24
2.3.6 Paquetes perdidos	24
2.3.7 Calidad y resolución de imagen	25
2.4 Software Defined Radio (SDR).....	25
2.4.1 GNU radio	25
2.4.2 gr-satellites	26
2.5 Python Modules.....	27
2.5.1 Versión de python	27
2.5.2 Numpy.....	27
2.5.3 Scipy	27
2.5.4 Matplotlib	27
3. Estado de la Cuestión	28
3.1 SSTV en transmisiones por Satélite	28

3.1.1	Inicios	28
3.1.2	Primera transmisión de imágenes por satélite	29
3.2	AMSAT	29
3.2.1	Satélites AMSAT EA	30
3.2.2	Sistema SSTV que utilizan los satélites de AMSAT EA.....	31
3.3	SSDV	32
3.3.1	Primeras transmisiones exitosas implementando SSDV	33
3.3.2	Estado actual de transmisión de imágenes SSDV en los satélites de Radioaficionados..	33
3.3.3	JY1SAT	33
3.3.4	Longjiang-2	34
4.	Definición del Trabajo	36
4.1	Justificación.....	36
4.1.1	Cambio a digital	36
4.1.2	SSDV en transmisiones de satélites de radioaficionados	37
4.2	Objetivos	38
4.3	Metodología.....	39
4.4	Planificación y Estimación Económica	40
5.	Estudio de Transmisión de Imágenes en SSDV	41
5.1	Análisis de una transmisión de un satélite en SSDV	41
5.1.1	Primera transmisión DSLWP	42
5.1.2	Transmisión estándar DSLWP	45
5.2	Diseño e implementación del receptor de Imágenes	47
5.2.1	Receptor de tramas DSWLP	47
5.2.2	Receptor de tramas JY1SAT	48
5.3	Resultados y Conclusiones	50
5.3.1	Resultados	50
5.3.2	Conclusiones.....	52
6.	Diseño del sistema de transmisión de imágenes en SSDV	53
6.1	Análisis del Sistema	53
6.2	Parámetros de transmisión y recepción	55
6.3	Diseño del transmisor	56
6.3.1	Codificador de imagen	56

6.3.2	Modulador y configuración de parámetros de transmisión	57
6.3.3	Modulación BPSK	57
6.3.4	Modulación QPSK.....	60
6.3.5	Modulación BFSK	63
6.3.6	Modulación QFSK.....	66
6.4	Diseño del receptor.....	69
6.4.1	Demodulador	69
6.4.2	Demodulación BPSK.....	69
6.4.3	Demodulación QPSK.....	70
6.4.4	Demodulación BFSK.....	71
6.4.5	Demodulación QFSK.....	72
6.4.6	Decodificador de imágenes	73
7.	Análisis de resultados de Simulación	75
7.1	BPSK.....	76
7.2	QPSK.....	79
7.3	BFSK.....	82
7.4	QFSK.....	85
7.5	Comparación	88
8.	Conclusiones y Trabajos Futuros	90
9.	Bibliografía	92
ANEXO A		94
ANEXO B		115

Índice de figuras

Figura 1. Imagen transmitida SSTV sobre SNR - Amsat EA	8
Figura 2. Imagen transmitida SSTV sobre SNR , FSK - Amsat EA	9
Figura 3. Link Budget SSTV Amsat EA	10
Figura 4. Espectro Frecuencia SSTV en Hz [2]	12
Figura 5. Waterfall SSTV end Hz [2].....	12
Figura 6. Principales modos de transmisión SSTV [3]	15
Figura 7. Ejemplo de Codificación Huffman [7].....	20
Figura 8. Compresión JPEG para un bloque de 8×8 de luminosidad. [4]	21
Figura 9. Formato Paquete SSDV. [1].....	23
Figura 10. Transmisión de imagen en SSDV recibida por JYSAT exitosamente	34
Figura 11. Transmisión de imagen en SSDV recibida por Longjiang-2 exitosamente	35
Figura 12. Tablón de tareas	39
Figura 14. Diagrama de gant del proyecto	40
Figura 14. Transmisor/receptor modos SSDV DSLWP	42
Figura 15. Muestra de Bits en GMSK	43
Figura 16. Diagrama de ojo primera transmisión SSDV DSLWP	44
Figura 18. Correlación primera transmisión SSDV DSLWP	44
Figura 18. Diagrama de ojo segunda transmisión SSDV DSLWP	46
Figura 19. Correlación ASM segunda transmisión SSDV DSLWP.....	47
Figura 20. Diagrama de bloques recepto Imágenes SSDV DSWLP.....	48
Figura 21. Transmisor/receptor modos SSDV JY1SAT	49
Figura 22. Diagrama de bloques receptor JY1SAT.....	50
Figura 23. Imagen SSDV recibida DSWLP procesada por gr-satellies	51
Figura 24. Imagen SSDV recibida DSWLP procesada en gr-satellies.....	51
Figura 25. Transmisor imágenes en SSDV para diferentes modulaciones.....	54
Figura 26. Receptor imágenes en SSDV para diferentes modulaciones	55

Figura 27. Sistema de transmisión/recepción de imágenes SSDV aplicando SNR en dB	55
Figura 28. Codificador de imágenes del sistema desarrollado	57
Figura 29. Diagrama de bloques modulación BPSK[23]	58
Figura 30. Señal modulada modulada en BPSK.....	58
Figura 31. Señal modulada+ruido en BPSK.....	59
Figura 32. Modulación BPSK en el dominio de la frecuencia	59
Figura 33. Diagramas de ojo de BPSK.....	60
Figura 34. Diagrama de bloques modulación QPSK[23]	61
Figura 35. Señal modulada en QPSK	61
Figura 36. Señal modulada+ruido en QPSK	62
Figura 37. Modulación QPSK en el dominio de la frecuencia	62
Figura 38. Diagramas de ojo de QPSK	63
Figura 39. Diagrama de bloques modulación BFSK[23]	64
Figura 40. Señal modulada en BFSK	64
Figura 41. Señal modulada+ruido en BFSK.....	65
Figura 42. Modulación BFSK en el dominio de la frecuencia	65
Figura 43. Diagramas de ojo de BFSK.....	66
Figura 45. Diagrama de bloques modulación QFSK[23]	66
Figura 45. Señal modulada en QFSK	67
Figura 46. Señal modulada+ruido en QFSK	67
Figura 47. Modulación BPSK en el dominio de la frecuencia	68
Figura 48. Diagramas de ojo de QFSK	68
Figura 49. Diagrama de bloques demodulación BPSK[23]	69
Figura 51. Demodulación de la señal en Bits, BPSK	69
Figura 51. Diagrama de bloques demodulación QPSK[23]	70
Figura 52. Demodulación de la señal en Bits, QPSK.....	70
Figura 53. Diagrama de bloques demodulación BFSK[23]	71
Figura 54. Demodulación de la señal en Bits, BFSK	71
Figura 55. Diagrama de bloques demodulación QFSK[23]	72
Figura 56. Demodulación de la señal en Bits, QFSK.....	72

Figura 57. Decodificador de imágenes del sistema desarrollado	73
Figura 58. Resultado final del sistema, imagen JPEG.....	74
Figura 59. Imagen elegida para la simulación de transmisión de imágenes sobre SSDV.....	75
Figura 60. BER BPSK para sistema desarrollado para diferentes bits rates	76
Figura 61. Resultado de imágenes transmitidas en BPSK.....	77
Figura 62. BER QPSK para sistema desarrollado para diferente bit rates	79
Figura 63. Resultado de imágenes transmitidas en QPSK	80
Figura 65. BER BPSK para sistema desarrollado para diferentes bit rates	82
Figura 65. Resultado de imágenes transmitidas en BFSK.....	83
Figura 66. BER QFSK para sistema desarrollado para diferentes bits rates	85
Figura 67. Resultado de imágenes transmitidas en QFSK	86
Figura 68. BER Comparativa de las modulaciones BPSK, QPSK, BFSK y QFSK	88

Índice de tablas

Tabla 1. Frecuencias SSTV Hz.....	12
Tabla 2. Tabla de frecuencias SSTV según Banda.....	13
Tabla 3. Paramtros Modulacion/Demodulacion del Sistema	56
Tabla 4. Resultados de la simulacion para BPSK	78
Tabla 5. Resultados de la simulacion para BPSK	81
Tabla 6. Resultados de la simulacion para BFSK	84
Tabla 7. Resultados de la simulacion para QFSK	87

1. INTRODUCCIÓN

1.1 MOTIVACIÓN

Actualmente la Asociación AMSAT-EA colaboradora de este proyecto, sigue mandando al espacio satélites de radioaficionados cada año para cubrir ciertas funciones y seguir mejorando año a año. Uno de sus campos de interés es la obtención de imágenes a través de sus satélites, por ejemplo, obtener imágenes de la tierra o de la luna y luego transmitir las para su posterior estudio. El método utilizado para esta transmisión de imágenes transmite y realiza las operaciones a de forma analógica, SSTV (Slow Scan Television). Sus principales problemas son que no consigue una SNR (señal a ruido) suficiente para que la calidad de la foto sea buena y se pueda obtener información relevante de las mismas, además de utilizar una modulación muy ineficiente para el SNR disponible del enlace. Sumando que este método de transmisión y recepción se encuentra bastante anticuado, siendo incapaz de obtener resultados eficientes con un bajo coste de los componentes utilizados. En las siguientes figuras podemos comprobar las pruebas realizadas sobre SSTV previamente al lanzamiento del satélite Hades. En la figura 1 vemos la señal original y una reconvertida y en la 2 como trasmite SSTV sobre una SNR incremental.



Figura 1. Imagen transmitida SSTV sobre SNR - Amsat EA



Figura 2. Imagen transmitida SSTV sobre SNR , FM - Amsat EA

Como se observa en la figura 2, para que se empiece a recibir una señal de la cual podamos distinguir bien los objetos con claridad, esta estaría por encima de los 10 dB, siendo este un SNR muy alto. Como se comentaba previamente este proyecto pretende buscar una solución a este problema, utilizando una transmisión con módulos digitales.

SSDV (Slow Scan Digital Video) es una forma digital paqueteada de SSTV (Slow Scan Televisión). Fue desarrollada por unos radioaficionados en Inglaterra y consiste en realizar lo mismo que hace SSTV, pero con la diferencia que, en vez de enviar líneas de la imagen, se envían frames de la misma codificada formato JPEG [3], además de poder utilizarse para transmitir pequeñas imágenes junto con la telemetría regular transmitida durante el vuelo. Los creadores argumentan que puede utilizar cualquier modo digital que pueda transportar texto o datos, aunque la implementación actual se limita a RTTY (Radio Teletipo) de 8 bits, haciendo posible nuevas características de transmisión. Estas nuevas casuísticas ya probadas por proyectos previos son el envío temporal de imágenes según su posición de transmisión

o la recuperación de frames perdidos, por lo que esto conlleva que se pueda saber con facilidad cuales son los frames que faltan, cosa que no pasa con SSTV, ya que para enviar una línea se debería volver a enviar toda la imagen. [1]

Por lo que la motivación del proyecto reside en poder simular los resultados que daría transmitir en SSDV en vez de en SSTV, ya que disponemos de los resultados de su transmisor de imágenes de SSTV de AMSAT EA, el cual obtuvo los siguientes parámetros gracias a la realización de un Link Budget, obteniendo resultados de SNR sobre el canal y calidad de imagen, Ancho de banda y demás parámetros que se verán en la siguiente figura:

LINK BUDGET 20200522										
TX OUTPUT POWER	Pt	mW	20		20		20		20	
TX ANTENNA			DIPOLE		DIPOLE		DIPOLE		DIPOLE	
RX ANTENNA			7EL		7EL		30EL		30EL	
PARAMETROS										
slant range	d	km	2000		2000		2000		2000	
frequency	f	MHz	436		436		436		436	
transmitter	Pt	dBm	13		13		13		13	
antenna TX	Gt	dBi	4.0		4.0		4.0		4.0	
antenna RX	Gr	dBi	7.0		7.0		20.0		20.0	
Tsystem YAGI 7EL+LNA NF=1dB	Tsys	K	376		376		376		376	
ENLACE										
effective radiated power	EIRP	dBm	17.0		17.0		17.0		17.0	
propagation losses	Lfs	dB	-151.3		-151.3		-151.3		-151.3	
beam steering losses	Lt	dB	-1.5		-1.5		-1.5		-1.5	
depolarization losses	Lp	dB	-1.5		-1.5		-1.5		-1.5	
receiver figure of merit	G/T	dB/K	-18.8		-18.8		-18.8		-18.8	
signal/noise ratio @1Hz	CNR0	dB/Hz	42.6		42.6		55.6		55.6	
SIGNAL NOISE RATIO										
bandwidth@3dB	BW	Hz	800		12500		800		12500	
signal/noise ratio	CNR	dB	13.6		1.6		26.6		14.6	
IMAGE QUALITY (0..5)			3		0		4		3	

Figura 3. Link Budget SSTV Amsat EA

Una de las motivaciones del Proyecto es poder simular a través de Software (SW) como con estos parámetros obtenidos en el link Budget (figura 1) como funcionara el modo SSDV y así poder compararlo con SSTV dentro del sistema y protocolos de transmisión de imágenes implementados por los satélites de AMSAT-EA. Siendo la adaptación del bit rate y SNR disponibles en el canal de transmisión, el uso de una modulación digital y la reducción del error de los paquetes enviados, las principales cosas a implementar/mejorar.

También debido a que SSDV no ha sido implementado por multitud de satélites, se hará un estudio de las transmisiones realizadas desde satélites de radioaficionados para así mas tarde aplicar este conocimiento aprendido a la simulación previamente comentada.

2. DESCRIPCIÓN DE LAS TECNOLOGÍAS

2.1 SSTV EN SATÉLITES DE RADIOAFICIONADOS

2.1.1 ANCHO DE BANDA

Cuando calculamos el ancho de banda, hay que tener en cuenta los efectos negativos que hacen que el canal:

- Atenuación
- Bias, distorsiones o imperfecciones del camino
- Intensidad de la señal
- Rango de frecuencias de modulación

Por lo que para calcular el ancho de banda deberemos tener muy en cuenta el rango de frecuencias que queremos utilizar, pero también el espectro de la señal. Además, debemos definir el ancho de banda considerándose una característica del trayecto de transmisión dada por el rango del espectro de la señal. Por lo que debemos instaurar una regla básica para el ancho de banda necesario, Nyquist. Su definición es que el ancho de banda óptimo es igual a la mitad de la velocidad de modulación.

2.1.2 MODULACIONES ANALÓGICAS PARA SSTV

Los Sistemas de transmisión broadcast SSTV utilizan SSB (Single Side Band) para modulaciones en amplitud con un transceptor de radioaficionado común. El rango de frecuencias va de 1500-2500 Hz, pero la máxima frecuencia utilizada es 2300Hz, debido a que es máximo nivel de SSTV el cual es la frecuencia del color blanco. Todas las frecuencias por encima de 2500 Hz son filtradas. [2]

Como sabemos este tipo de transmisiones son transmitidas a través de FM en una señal de Audio. Pero para evitar algún desfase o desviación, se utiliza una portadora auxiliar en 1900 Hz, llamada SCFM (Sub-carrier frequency modulation).

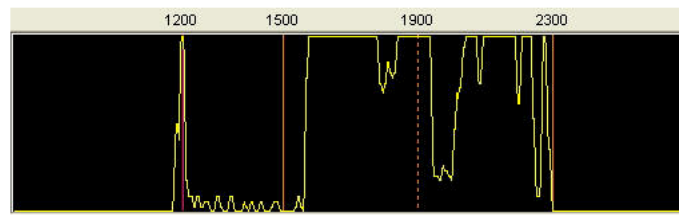


Figura 4. Espectro Frecuencia SSTV en Hz [2]

Parámetro	Valor
Tono de Sincronismo	1200 Hz
Tono Negro	1500 Hz
Tono	2300 Hz
Sincronismo de imágenes	30 ms
Sincronismo de línea	5 ms

Tabla 1. Frecuencias SSTV Hz

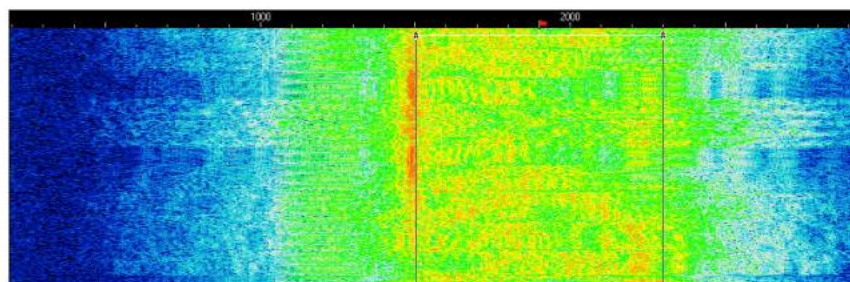


Figura 5. Waterfall SSTV end Hz [2]

2.1.3 FRECUENCIAS DE TRASMISIÓN

Las **frecuencias de transmisión** de SSTV se dividen en tres según su tipo de transmisión [3].

1. HF - High Frequency

La franja de frecuencias de este espacio es de 3 MHz a 30 MHz. Estas frecuencias se utilizan principalmente para el intercambio de fotografías a escala internacional. Otro nombre para la banda de HF es la banda decamétrica debido a que sus longitudes de onda son de 10 a 100 metros en general

2. VHF - Very High Frequency

La franja de frecuencias de radio VHF comienza justo donde termina la HF. Funciona con ondas electromagnéticas de entre 30 MHz y 300 MHz de frecuencia. Normalmente, esta banda se utiliza para comunicarse con personas cercanas a su localidad. Puedes recurrir a la ayuda de un sistema de repetidores enlazados para ampliar aún más su alcance.

3. Bandas laterales de Radio – UHF

Las dos bandas laterales de radio más comunes son LSB (banda lateral inferior) y USB (banda lateral superior). Ambas son populares para los usos en la banda de HF para la operación de radioaficionados. Esto significa que sólo se utilizará una de ellas a la vez. No existe ninguna limitación legal para el uso de cualquiera de ellas en las bandas de frecuencia.

Un problema es la variación de la calidad de la señal debido a la rotación del satélite, que provoca una señal de fuga. Para ello, se debe utilizar la antena con polarización circular, misma antena que aparece reflejada en el link Budget mostrado en la figura 3.

Banda de Frecuencias	Designador
21–30 MHz H	H
144–146 MHz	V
435–438 MHz	U

Tabla 2. Tabla de frecuencias SSTV según Banda

Las bandas de frecuencia de los dispositivos lineales se muestran en la tabla previa. Estas frecuencias describen modos de funcionamiento de los relés de los satélites. Se fija por el diseño del satélite en elegido por el centro de control. El designador como U/V describe el enlace ascendente 435-438 MHz (U) y el enlace descendente 144-146 MHz (V).

2.1.4 MODOS DE TRANSMISIÓN

Se enumeran los modos de transmisión mas relevantes de SSTV [3]:

1. Robot

La Robot Research Corporation desarrolló este modo de transmisión para su propia unidad SSTV que funciona con frecuencias HF y VHF. Entre sus variantes, las llamadas Robot 72 y Robot 36. Es compatible con los esquemas de color BW, RGB y YUV. Utiliza líneas de barrido de 120, 128, 240 o 256 para transmitir o recrear la imagen.

2. Martin:

Martin Emmerson elaboró originalmente este modo sólo para garantizar la mejora de las unidades SSTV de Robot hasta su máximo potencial. Sin embargo, el propio SSTV Martin se ha convertido en una opción bastante popular para la transmisión de fotos mediante frecuencias de radio en Europa. Escanea 240 líneas de 256, pero se reserva las 16 primeras para que el emisor tenga un espacio para el rótulo en la parte superior o inferior de la foto.

3. Scottie:

Surgió como parte de la mejora de la capacidad de ROM de Robot. Escanea las 256 líneas para el espacio extra del rotulo. Como ventajas, es ligeramente más rápido que Martin y usa RGB como esquema de color.

Pero cada modo tiene también una serie de sub-modos, que difieren en el número de líneas de barrido (por ejemplo, 120 o 240) y en el tiempo de transmisión (más largo se traduce en

un potencial de mayor calidad de imagen). En la imagen siguiente se enumeran algunos ejemplos (B&W = blanco y negro; RGB = rojo-verde-azul, YC = luminancia/crominancia).

Mode	Color	Time (sec/image)	Usable scan lines	Mode	Color	Time (sec/image)	Usable scan lines
Martin				Robot Color			
M1	RGB	114	240	12	YC	12	120
M2	RGB	58	240	24	YC	24	120
M3	RGB	57	120	36	YC	36	240
M4	RGB	29	120	72	YC	72	240
Scottie				Robot B&W			
S1	RGB	110	240	8	B&W	8	120
S2	RGB	71	240	12	B&W	12	120
S3	RGB	55	120	24	B&W	24	240
S4	RGB	36	120	36	B&W	36	240
DX	RGB	269	240	72	B&W	72	240

Figura 6. Principales modos de transmisión SSTV [3]

2.2 SSTV DIGITAL

2.2.1 BÁSICOS DE LA COMUNICACIÓN DIGITAL

Lo que más nos interesa es la velocidad a la que se puede transmitir la información: distinguimos entre velocidad de transmisión y modulación [4]:

Symbol rate – tasa de símbolo

Es el número de cambios a de la señal portadora por segundo. Se mide en unidad de baudios (Bd) o símbolos por segundo (S/s). La tasa de símbolos no dice nada sobre la cantidad de información transmitida en la señal portadora.

$$v_m = \frac{1}{a} [Bd]$$

Bit Rate – Velocidad de bits

Cantidad de información transferida por segundo. Se expresa en bits por segundo (bps). La tasa de bits no dice nada sobre la rapidez con la que cambia la portadora de la señal.

$$V_p = v_m \times \log_2 m \text{ [bps]}$$

Como sabemos un básico de las comunicaciones digitales es el ancho de banda, B , el cual viene definido por el teorema de Nyquist:

$$B = \frac{v_m}{2}$$

$$v_p = 2 \times v_m \times \log_2 m$$

Se podría pensar que es posible alcanzar cualquier velocidad gracias a la mejora de la modulación y a un mayor número de estados, esto no es así debido a que hay límites físicos muy marcados. La capacidad máxima del canal C (tasa de bits) en bps es Ng Soven por la ley de Shannon, que depende del ancho de banda B (Hz) y de los parámetros del canal relación señal/ruido S/N (dB):

$$C = B \times \log_2 \left(1 + \frac{S}{N}\right)$$

Como vemos, la velocidad máxima de bits no se ve afectada por la tecnología utilizada, sino el ancho de banda B y la relación señal/ruido (SNR), que no se pueden modificar. La SNR se expresa en decibelios (dB) y describe la relación entre la potencia de la señal y la potencia del ruido de un ancho de banda procesado.

2.2.2 DETECCIÓN Y CORRECCIÓN DE ERRORES

Los códigos de detección de errores se utilizan para comprobar la ausencia de errores en la transmisión. La idea se basa en añadir datos adicionales (redundancia) a un mensaje. La redundancia se genera a partir de algunos datos de entrada en un lado de la transmisión (FEC - Forward Error Correction) y en un lado de la recepción es posible comprobar si los datos se han transferido.

El código utilizado también puede tener capacidad de corrección de errores, de modo que los datos afectados durante la transmisión pueden repararse en el lado de la recepción sin necesidad de retransmitirlos.

2.2.2.1 *Cyclic redundancy check*

La comprobación de redundancia cíclica (CRC) es un código de detección de errores que se utiliza habitualmente en redes digitales y dispositivos de almacenamiento para detectar cambios accidentales en los datos sin procesar. A los bloques de datos que entran en estos sistemas se les añade un breve valor de comprobación, basado en el resto de una división polinómica de su contenido. Al recuperarlos, se repite el cálculo y, en caso de que los valores de comprobación no coincidan, se pueden tomar medidas correctivas contra la corrupción de los datos. Los CRC pueden utilizarse para la corrección de errores (ver filtros de bits) [5].

Los CRC se llaman así porque el valor de comprobación (comprobación de datos) es una redundancia (amplía el mensaje sin añadir información) y el algoritmo se basa en códigos cíclicos. Los CRC son populares porque son sencillos de implementar en hardware binario, fáciles de analizar matemática y especialmente buenos para detectar errores comunes causados por el ruido en los canales de transmisión. Como el valor de comprobación tiene una longitud fija, la función que lo genera se utiliza ocasionalmente como función hash.

2.2.2.2 *Hamming code*

En el ámbito de las comunicaciones de datos (por ejemplo, el teletexto de la televisión) se utiliza a veces el código Hamming, que puede detectar hasta dos errores y, en el caso de un error, es capaz de determinar en qué punto del código se produjo el error y puede fijar los bits recibidos.[4]

Básicamente, utiliza para sus propósitos la paridad. Mientras que los bits de paridad están en el codeword final posicionado en el número de serie es igual al cuadrado de 2 (1., 2., 4., 8,...). Bajo la posición del bit de control se selecciona entonces cierta secuencia de palabras de información, que se utiliza para determinar el valor del bit de control.

2.2.2.3 Reed-Solomon code

Los códigos Reed-Solomon funcionan sobre un bloque de datos tratado como un conjunto de elementos de campo finito llamados símbolos. Los códigos Reed-Solomon son capaces de detectar y corregir errores de múltiples símbolos. Añadiendo $t = n - k$ símbolos de comprobación a los datos, un código Reed-Solomon puede detectar (pero no corregir) cualquier combinación de hasta t símbolos erróneos, o localizar y corregir hasta $t/2$ símbolos erróneos en lugares desconocidos. Como código de borrado, puede corregir hasta t borrados en ubicaciones conocidas y proporcionadas al algoritmo, o puede detectar y corregir combinaciones de errores y borrados. Los códigos Reed-Solomon también son adecuados como códigos de corrección de errores de bits de ráfagas múltiples, ya que una secuencia de $b + 1$, errores de bits consecutivos puede afectar como máximo a dos símbolos de tamaño b . [6]

Los parámetros del RS (n, k) se definen como sigue:

Al igual que el CRC, el código RS es sistemático. Para su generación se utilizan los cálculos algebraicos del campo de Galois.

- s - número de bits en un símbolo informativo,
- k - número de símbolos de s bits en el bloque de datos,
- n - número de bits en la palabra de código.

RS (n, k) es capaz de corregir $n-k$ errores en k símbolos. Normalmente es utilizado RS (255,223), utilizando 223 símbolos de 8-bits para la creación de 255 símbolos.

El código RS ha encontrado aplicación en diversos ámbitos: es utilizado por la NASA para las comunicaciones espaciales.

2.2.3 COMPRESIÓN DE DATOS

Como es de esperar no podremos transmitir una imagen totalmente en bruto debido a que la velocidad de transmisión se elevaría demasiado, teniendo así una comunicación no exitosa. Debido a esto, los módulos digitales de transmisión de imágenes utilizan los siguientes

algoritmos y estándares para la compresión de imágenes en digital, para su posterior transmisión.

2.2.3.1 Codificación Huffman

El código Huffman es una forma de codificar la información utilizando cadenas de longitud variable para representar símbolos en función de su frecuencia de aparición. La idea es que los símbolos que se utilizan con más frecuencia sean más cortos, mientras que los que aparecen más raramente pueden ser más largos. De este modo, el número de bits que se necesita para codificar un determinado mensaje será más corto, por término medio, que si se utiliza un código de longitud fija. En los mensajes que incluyen muchos símbolos poco frecuentes, la cadena producida por la codificación de longitud variable puede ser más larga que la producida por una codificación de longitud fija. Dado que las codificaciones de longitud variable son susceptibles de ser ambiguas, hay que tener cuidado de generar un esquema en el que se evite la ambigüedad. [7]

La codificación Huffman utiliza un algoritmo de codificación para construir un árbol binario de codificación que optimiza el esquema de codificación para que los símbolos más utilizados tengan la codificación más corta. El árbol describe la codificación la cual garantiza que el código de un símbolo concreto nunca es un prefijo de la cadena de bits y que representa a cualquier otro símbolo. Para determinar la asignación binaria de un símbolo, hay que hacer que las hojas del árbol correspondan a los símbolos, y la asignación será el camino que se recorre para llegar desde la raíz del árbol hasta esa hoja.

Árbol de Huffman generado para las frecuencias de apariciones exactas del texto "ESTO ES UN EJEMPLO DE UN ARBOL DE HUFFMAN", véase en la siguiente figura :

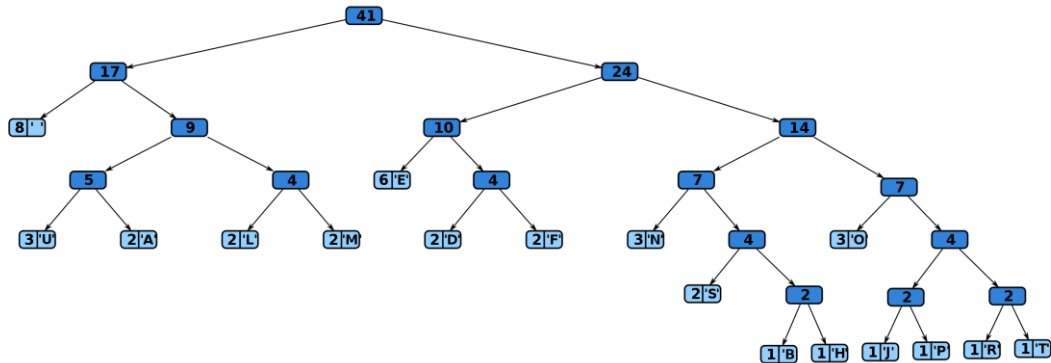


Figura 7. Ejemplo de Codificación Huffman [7]

2.2.3.2 JPEG

JPEG no utiliza la codificación de color RGB, sino que utiliza la codificación YCrCb. Esto se debe a que el ojo humano percibe el brillo y los colores con diferente sensibilidad. El almacenamiento de los colores YCrCb, sobre todo en la proporción 4: 2 : 0, reduce el tamaño del archivo, pero no es suficiente por lo que se tienen que seguir mas pasos, véase la imagen de como se transforma posteriormente, véase el esquema de la figura.[4]

En el primer paso, la imagen se divide en un bloque cuadrado de 8×8 píxeles y estos 64 puntos se transforman desde el dominio espacial (x, y) a la frecuencia (i, j) mediante la transformada discreta del coseno:

$$DCT(i, j) = \frac{1}{4} C(i) \cdot C(j) \sum_{x=0}^7 \sum_{y=0}^7 \text{pixel}(x, y) \cdot \cos \left[\frac{(2x+1)i\pi}{16} \right] \cdot \cos \left[\frac{(2y+1)j\pi}{16} \right],$$

$$\text{where } C(a) = \begin{cases} \frac{1}{\sqrt{2}} & \text{for } a = 0; \\ 1 & \text{in other cases.} \end{cases}$$

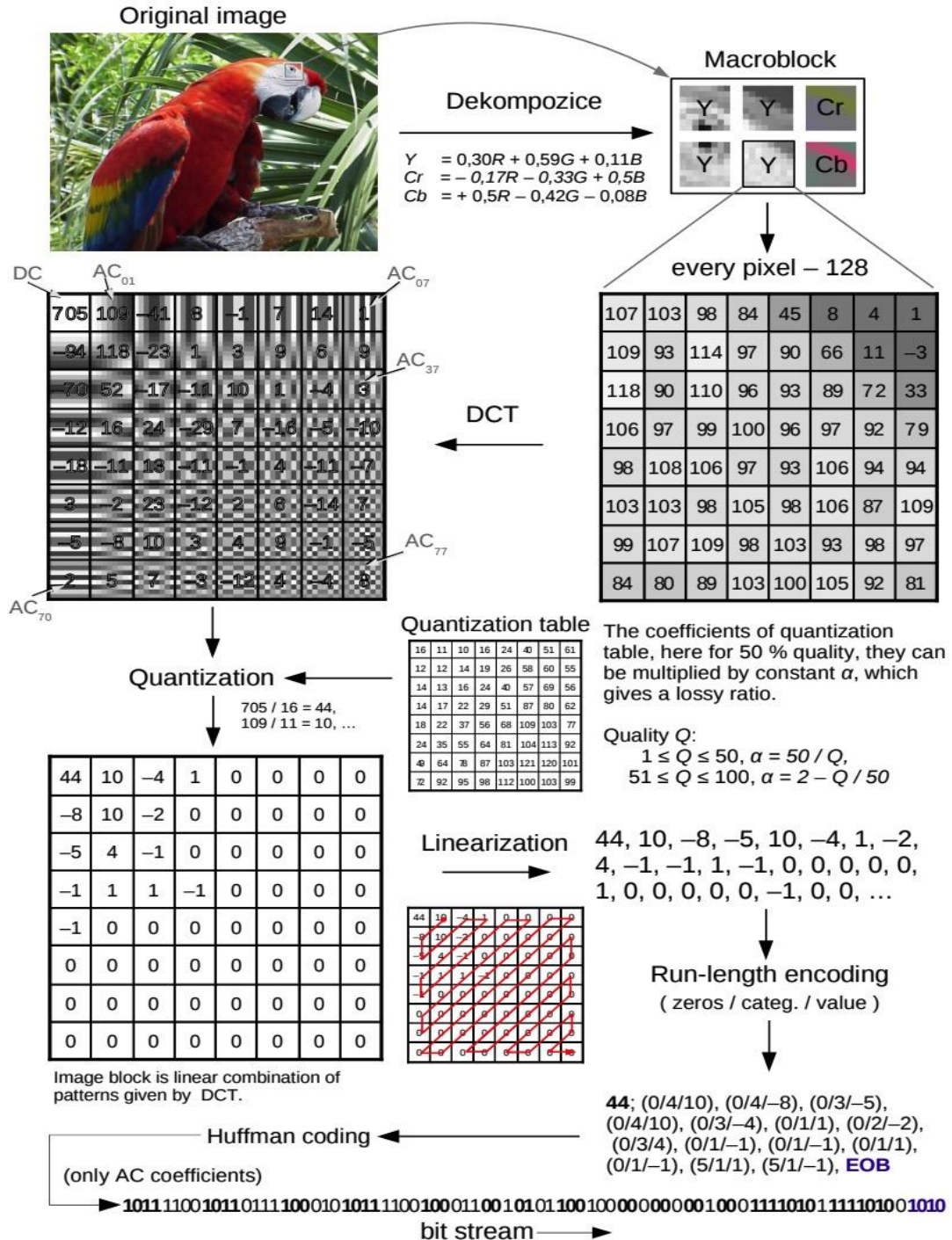


Figura 8. Compresión JPEG para un bloque de 8×8 de luminosidad. [4]

A continuación, se realiza el paso que más influye en la imagen resultante y en la percepción del nivel de compresión con pérdidas, la cuantificación. Un miembro del bloque en la posición 00 se divide por un miembro 00 de la tabla de cuantificación y la posición de la parte entera del número se almacena en la posición 00, continúa 01/01, 02/02, hasta que cada valor se divide por su coeficiente correspondiente. El resultado de este proceso es una matriz cuadrada, donde la mayor parte de la información se almacena en la esquina superior izquierda y alrededor de la esquina inferior derecha sólo hay ceros. Esta matriz se linealiza en una secuencia gracias a la lectura en "zig-zag", los valores no nulos aparecen delante de la secuencia y la parte restante se rellena con ceros.

A continuación, la secuencia se divide en categorías, la primera es el coeficiente DC y luego continúan los demás valores y para cada uno se determinan los siguientes valores (número de ceros previos / categoría / valor intrínseco). Los ceros redundantes se reducen mediante la codificación RLE y a partir de algún lugar se muestran ceros. Todos los ceros se omiten y se sustituyen por la marca EOB (fin de bloque). Así, los valores de los coeficientes de corriente continua, luminosidad y crominancia tienen sus codificaciones.

Más tarde se realiza una codificación Huffman y se guarda en un archivo. Para volver a mostrar la imagen se realizaría el proceso, pero al revés.

2.3 MODULO SSDV UTILIZADO

En este apartado se explica la tecnología utilizada en el módulo de transmisión de imágenes SSDV utilizado en este proyecto. [1]

2.3.1 DEFINICIÓN

SSDV (Slow Scan Digital Video) es una forma digital empaquetada de SSTV (Slow Scan TeleVision). Puede utilizarse para transmitir pequeñas imágenes junto con la telemetría regular transmitida por una carga útil durante el vuelo. Se puede utilizar cualquier modo digital que pueda transportar texto o datos. SSDV es idóneo para este problema, ya que cuenta con una estandarización y una resolución de problemas avanzada, cubriendo en

profundidad factores esenciales como son el formato del paquete, formato de la imagen, corrección de errores, recuperación de paquetes perdidos, modulación y futuras implementaciones, explicándose en la siguiente pagina mucho mas en detalle [2].

2.3.2 FORMATO DEL PAQUETE

Offset	Name	Size	Description
0	Sync Byte	1	0x55 - Puede ir precedido de uno o más bytes de sincronización
1	Packet Type	1	0x66 - Modo normal (paquete de 224 bytes + 32 bytes FEC) 0x67 - Modo sin FEC (paquete de 256 bytes)
2	Callsign	4	Indicativo codificado en base-40. Hasta 6 dígitos
6	Image ID	1	imagen
7	Packet ID	2	endian)
9	Width	1	Ancho de la imagen en bloques MCU (píxeles / 16) 0 = No válido
10	Height	1	Altura de la imagen en bloques MCU (píxeles / 16) 0 = No válido
11	Flags	1	Bandera EOI (1 = Último paquete), xx = Modo de submuestreo (0 = 2×2, 1 = 1×2, 2 = 2×1, 3 = 1×1)
12	MCU offset	1	carga útil, o 0xFF si no hay ninguno presente
13	MCU index	2	0xFFFF si no hay ninguno presente
15	Payload	205 / 237	Datos de la carga útil
220 / 252	Checksum	4	CRC de 32 bits
224	FEC	32	modo normal (0x66)

Figura 9. Formato Paquete SSDV. [1]

2.3.3 FORMATO DE LA IMAGEN

El **Formato de la imagen** de la imagen debe contener sólo los datos escaneados de una imagen JPEG, sin ninguno de los marcadores o cabeceras que normalmente componen una imagen JPEG. Al no haber cabeceras, la imagen debe ajustarse a las siguientes reglas: Formato de color YCbCr, una resolución de 4080 x 4080, las tablas de cuantificación y huffman son fijas, la longitud total de los datos escaneados no debe superar aprox los 12,8 MB, 205 bytes por paquete, 65536 paquetes como máximo, el número total de bloques MCU no debe superar los 65535 bytes.

2.3.4 CORRECCIÓN DE ERRORES

Para la **corrección de errores** cada paquete termina con 32 bytes, calculados desde los 223 bytes que comienzan con el tipo de paquete (offset 1) hasta el final de los datos de la carga. Para manejar mejor esto, el receptor debería ser capaz de detectar e insertar un número apropiado de bytes de relleno, permitiendo al decodificador verlos como errores y corregirlos.

2.3.5 MODULACIÓN UTILIZADA

La **modulación** implantada actualmente utiliza RTTY de 8 bits a varias velocidades de transmisión. Sus creadores ven las siguientes líneas de investigación en un **futuro**. Utilizar **tablas de huffman** y cuantificación dinámicas o adaptativas, realizar pruebas con diferentes Bps o utilizar una Modulación BPSK.

2.3.6 PAQUETES PERDIDOS

Cuando se pierde un paquete hay una serie de escenarios que pueden necesitar ser tratados:

- Un bloque MCU del paquete anterior estaba incompleto. Si la interrupción se produce durante los coeficientes de CA de un componente, se emite el símbolo EOB para finalizarlos antes. Para cada componente restante de la MCU emita un símbolo EOB para los coeficientes DC y AC. Esto debería terminar la MCU incompleta limpiamente.
- Reste el valor del índice de la MCU del índice del último bloque decodificado de la MCU para calcular cuántos bloques se han perdido. Por cada bloque perdido emite un bloque MCU vacío.
- Es probable que el paquete actual comience con el final de uno de los bloques MCU perdidos. Estos datos no pueden ser utilizados y deben ser descartados. El campo MCU offset de la cabecera indica el número de bytes que hay que saltar antes de reanudar la decodificación.

Una vez resueltas estas condiciones, se puede reanudar la decodificación normal.

2.3.7 CALIDAD Y RESOLUCIÓN DE IMAGEN

En este modulo se puede elegir la calidad de la imagen a transmitir y también la resolución. La anchura y la altura deben ser un múltiplo de 16 (hasta una resolución de 4080 x 4080)

2.4 SOFTWARE DEFINED RADIO (SDR)

La radio definida por software (SDR) es un sistema de radiocomunicación en el que los componentes que tradicionalmente se han implementado en hardware (por ejemplo, mezcladores, filtros, amplificadores, moduladores/demoduladores, detectores, etc.) se implementan mediante software en un ordenador personal o un sistema integrado. Aunque el concepto de SDR no es nuevo, la rápida evolución de las capacidades de la electrónica digital hace que sean prácticos muchos procesos que antes sólo eran posibles en teoría. [8]

Un sistema SDR básico puede consistir en un ordenador personal equipado con una tarjeta de sonido, u otro convertidor analógico-digital, precedido por algún tipo de extremo frontal de RF. Una cantidad importante de procesamiento de la señal se transfiere al procesador de propósito general, en lugar de realizarse en hardware de propósito especial (circuitos electrónicos). Este tipo de diseño produce una radio que puede recibir y transmitir protocolos de radio muy diferentes (a veces denominados formas de onda) basados únicamente en el software utilizado.

Las radios por software tienen una gran utilidad para los servicios militares y de telefonía móvil, que deben dar servicio a una gran variedad de protocolos de radio cambiantes en tiempo real. A largo plazo, defensores como el Foro de Innovación Inalámbrica esperan que las radios definidas por software se conviertan en la tecnología dominante en las comunicaciones por radio.

2.4.1 GNU RADIO

GNU Radio es un kit de herramientas gratuito y de código abierto para el desarrollo de software que proporciona bloques de procesamiento de señales que permiten implementar radios por software. Puede utilizarse con hardware de RF externo de bajo coste y fácilmente

disponible para crear radios definidas por software, o sin necesidad de hardware en un entorno de simulación. Se utiliza ampliamente en entornos de aficionados, académicos y comerciales para apoyar tanto la investigación de las comunicaciones inalámbricas como los sistemas de radio del mundo real.[9]

2.4.2 GR-SATELLITES

gr-satellites es un módulo de GNU Radio que abarca una colección de decodificadores de telemetría que soporta muchos satélites de aficionados diferentes. Este proyecto de código abierto comenzó en 2015 con el objetivo de proporcionar decodificadores de telemetría para todos los satélites que transmiten en las bandas de radioaficionados. [10]

Soporta los protocolos más populares, como AX.25, los módems GOMspace NanoCom U482C y AX100, una parte importante de la pila CCSDS, el protocolo AO-40 utilizado en los satélites FUNCube, y varios protocolos ad-hoc utilizados en otros satélites.

Este módulo puede utilizarse para decodificar tramas transmitidas desde la mayoría de los satélites de aficionados en órbita, realizando demodulación, corrección de errores hacia adelante, etc. Las tramas decodificadas pueden guardarse en un archivo o mostrarse en formato hexadecimal. Para algunos satélites, la definición del formato de telemetría está incluida en gr-satellites, por lo que las tramas de telemetría decodificadas pueden imprimirse como valores legibles para el ser humano, como tensiones y corrientes del bus. Además, algunos satélites transmiten archivos como imágenes JPEG. gr-satellites puede utilizarse para desensamblar estos archivos e incluso mostrar las imágenes en tiempo real mientras se reciben.

gr-satellites puede utilizarse como un conjunto de bloques de construcción para implementar decodificadores para otros satélites u otras soluciones de estaciones terrestres. Algunos de los bloques de bajo nivel de gr-satellites también son útiles para otros tipos de protocolos de comunicaciones de RF.

2.5 *PYTHON MODULES*

2.5.1 VERSIÓN DE PYTHON

Se ha utilizado la versión 3.8.1.

2.5.2 NUMPY

NumPy es una librería de Python especializada en el cálculo numérico y el análisis de datos, especialmente para un gran volumen de datos. Incorpora una nueva clase de objetos llamados **arrays** que permite representar colecciones de datos de un mismo tipo en varias dimensiones, y funciones muy eficientes para su manipulación. En estos arrays guardaremos las señales en el tiempo. Véase la bibliografía [11] para una mejor explicación.

2.5.3 SCIPY

SciPy es una biblioteca libre y de código abierto para Python. Contiene módulos para optimización, algebra-lineal, integración, interpolación, funciones especiales, FFT, procesamiento de señales y de imagen, resolución de ODEs y otras tareas para la ciencia e ingeniería.

2.5.4 MATPLOTLIB

Matplotlib es una biblioteca para la generación de gráficos a partir de datos contenidos en listas o arrays en el lenguaje de programación Python y su extensión matemática NumPy. Proporciona una API, pylab, diseñada para recordar a la de MATLAB. Véase la bibliografía [13] para una mejor explicación de uso.

3. ESTADO DE LA CUESTIÓN

3.1 *SSTV EN TRANSMISIONES POR SATÉLITE*

La televisión de barrido lento (SSTV) es un modo de comunicación diseñado para la transferencia de imágenes. Dado que la SSTV es un modo de banda estrecha, puede transmitirse a través de canales de voz con un transceptor de BLU estándar en todas las bandas de frecuencia de radioaficionados. También es posible la comunicación a nivel mundial en buenas condiciones en las bandas de alta frecuencia.

3.1.1 INICIOS

Remontándonos unos años hacia atrás, en 1960, el Sputnik 5, envió unas de primeras fotos realizadas desde el espacio gracias a la ayuda del método SSTV (Slow Scan TV) [13]. Este fue creado por Cophorne Mcdonald en 1957-58, el desarrollo consistió en un monitor electrostático y un tubo vidicón, que su misión era para la transmisión de imágenes, utilizado principalmente por los radioaficionados, para transmitir y recibir imágenes estáticas por radio en monocromo o en color. Pero, también podemos considerar a SSTV un término literal para la televisión de banda estrecha, requiriendo al menos 6 MHz de ancho de banda, ya que transmite 25 o 30 cuadros de imagen por segundo (en los sistemas de color NTSC, PAL o SECAM), pero la SSTV sólo suele ocupar un máximo de 3 kHz de ancho de banda. Por lo que es un método que utiliza en la transmisión lenta de imágenes fijas, estimando que suele tardar entre ocho segundos y un par de minutos, según el modo utilizado, en transmitir un cuadro de imagen. Este método de transmisión de Televisión de barrido lento ha sido muy utilizado en la transmisión de imágenes durante la primera época de la exploración espacial. Por ejemplo, se utilizó un concepto similar, también llamado SSTVD, se utilizó en el Faith 7 así como en los primeros años del programa Apolo de la NASA. La cámara del Faith 7 transmitía un fotograma cada dos segundos, con una resolución de 320 líneas para utilizar menos ancho de banda que una transmisión de televisión normal. Las cámaras de televisión del Apolo utilizaron la SSTV para transmitir imágenes del interior del Apolo 7, Apolo 8 y

Apolo 9, así como la televisión del módulo lunar del Apolo 11 desde la Luna. Cabe destacar que estos primeros desarrollos de SSTV utilizados por la NASA difieren bastante de los sistemas SSTV utilizados actualmente por los radioaficionados. Pero este método se ha ido quedando anticuado con los años debido a que es totalmente analógico, por lo que se creo una implementación digital del mismo.

3.1.2 PRIMERA TRANSMISIÓN DE IMÁGENES POR SATÉLITE

La primea transmisión de televisión de barrido lento (SSTV) es transmitida por el equipo ARISS de Rusia desde la estación de radioaficionados situada en el módulo de servicio ruso de la Estación Espacial Internacional con el indicativo RS0ISS.[1]

En el pasado se enviaron doce imágenes diferentes en 145.800 MHz FM utilizando el modo SSTV PD180, con un tiempo de espera de 3 minutos entre cada imagen.

3.2 AMSAT

Cuando en 1958 EE. UU. lanzó su satélite EXPLORER-1, radioaficionados de todo el mundo soñaron con construir y lanzar su propio satélite. Así, en 1960 un grupo de radioaficionados de California (EE. UU.) crean el proyecto OSCAR (Orbiting Satellite Carrying Amateur Radio). Mas tarde --en 1969-- se crea la organización AMSAT, la cual tiene delegaciones en muchos países, en ellos España. [14]

El objetivo de AMSAT es lanzar satélites no comerciales para uso y disfrute de los radioaficionados. Bien podríamos decir que los objetivos de este programa son una utopía hecha realidad. Han conseguido hacer frente a muchos problemas, entre los cuales destaca especialmente el enorme coste del lanzamiento. La reducción de costes se debe a las siguientes causas:

- Lanzamientos compartidos con otros satélites comerciales. Las empresas y agencias gubernamentales les permiten poner a bordo sus satélites, a cambio ellos ofrecen sus conocimientos y asistencia técnica en el ámbito de la radio

- Reciben ayudas de los gobiernos en concepto de I+D. Es muy frecuente que utilicen materiales "surplus" o donados por la industria. Desde sus comienzos, se han lanzado más de 70 satélites de radioaficionado en los últimos 40 años. De todos ellos, sólo 28 más o menos dan servicio hoy en día.

3.2.1 SATÉLITES AMSAT EA

En este enlace se encuentran los satélites de AMSATEA desplegados o por lanzar [15]. Actualmente AMSAT EA tiene en espera para lanzar dos satélites.

3.2.1.1 EASAT-2

Estado de operatividad: En espera lanzamiento

Órbita: 500-600 km

Lanzamiento: *Esperado diciembre 2021

Frecuencias del repetidor: Subida en FM VHF (145.875 MHz) y bajada en UHF (436.666 MHz)

Polarización: Lineal.

Subtonos: Sin subtono.

Tabla recomendada de frecuencias:

- Subida 145.875 Mhz Bajada 436.676 MHz (AOS - Adquisición de señal en el horizonte)
- Subida 145.875 Mhz Bajada 436.671 MHz (acercándose)
- Subida 145.875 Mhz Bajada 436.666 MHz (sobre nosotros)
- Subida 145.875 Mhz Bajada 436.661 MHz (alejándose)
- Subida 145.875 Mhz Bajada 436.656 MHz (LOS - Pérdida de señal en el horizonte)

Telemetría / baliza: 436.666 MHz CW FSK 50 bps, baliza voz FM con indicativo AM5SAT

3.2.1.2 HADES

Estado de operatividad: En espera lanzamiento

Órbita: 500-600 km

Lanzamiento: *Esperado diciembre de 2021

Frecuencias del repetidor: Subida en FM VHF (145.925 MHz) y bajada en UHF (436.888 MHz)

Polarización: Lineal.

Subtonos: Sin subtono.

Tabla recomendada de frecuencias:

- Subida 145.925 Mhz Bajada 436.898 MHz (AOS - Adquisición de señal en el horizonte)
- Subida 145.925 Mhz Bajada 436.893 MHz (acercándose)
- Subida 145.925 Mhz Bajada 436.888 MHz (sobre nosotros)
- Subida 145.925 Mhz Bajada 436.883 MHz (alejándose)
- Subida 145.925 Mhz Bajada 436.878 MHz (LOS - Pérdida de señal en el horizonte)

Telemetría / baliza: 436.888 MHz CW FSK 50 bps, SSTV Robot 36, baliza voz FM con indicativo AM6SAT

3.2.2 SISTEMA SSTV QUE UTILIZAN LOS SATÉLITES DE AMSAT EA

AMSAT EA actualmente utiliza este sistema para la transmisión de imágenes en sus últimos lanzamientos. Consiste en una placa transmisora SSTV que proporciona un generador de

señales de televisión de barrido lento (SSTV) con telemetría PSK/CW, enlace ascendente APRS y un módulo para la cámara. Está basada en el microcontrolador STM32F446RET6[16], transmitiendo la telemetría en PSK y las imágenes SSTV en los modos Robot36 y MP73, para el Satélite Hades. Pero, actualmente quieren mejorar su transmisión de imágenes, ya que su calidad y su corrección de errores, se ven muy limitadas por el uso de esta tecnología.

Además, se dispone de los resultados de su transmisor de imágenes en SSTV obtuvo los siguientes parámetros gracias a la realización de un Link Budget, obteniendo resultados muy significativos de SNR y calidad de imagen, Ancho de banda y demás parámetros que se verán en la figura 3 del proyecto.

3.3 SSDV

Desde hace años, los globos de gran altura pueden enviar imágenes en directo durante sus vuelos y ofrecer una experiencia más inmediata de su seguimiento. Esto ha sido posible gracias a la tecnología Slow Scan Digital Video (SSDV), creada por Philip Heron (fsphil). Es una forma de empaquetar una imagen codificada en JPEG y enviarla entre cadenas de telemetría ordinarias. En la forma original usando RTTY generalmente a 300-600 baudios, o hoy en día a velocidades mucho más altas. [1]

Cubre los siguientes puntos de una transmisión de imágenes, los cuales son explicados con mas profundidad en el capitulo de descripción de tecnologías:

- Formato de paquete
- Formato de imagen
- Corrección de errores
- Paquetes perdidos
- Modulación / Modo

3.3.1 PRIMERAS TRANSMISIONES EXITOSAS IMPLEMENTANDO SSDV

Una de las primeras transmisiones exitosas fue impulsada por un grupo de aficionados de la Universidad de Bristol, el 1 de Julio de 2017. Utilizaron para la transmisión de SSDV, 300 baudios sobre RTTY, con modulación GMSK.[17]

La primera transmisión real de SSDV de un satélite de radioaficionados se realizó en el satélite JY1SAT. Este activó el modo SSDV que envía frames de imágenes digitales junto a la telemetría en febrero de 2019, siendo la primera transmisión desde un satélite radioaficionados.

3.3.2 ESTADO ACTUAL DE TRANSMISIÓN DE IMÁGENES SSDV EN LOS SATÉLITES DE RADIOAFICIONADOS

Actualmente hay más de 100 satélites de radioaficionados transmiten en las bandas de radioaficionados. Los más populares incorporan un transpondedor lineal (FUNcube) o repetidor FM (LilacSat-2). La mayoría sólo transmiten datos de telemetría, realizando misiones educativas, científicas o experimentales y pertenecen a grupos de Radioaficionados, centros educativos y entidades privadas. Pero su principal problema reside en que todavía se transmite mucha telemetría en CW, ya que es muy fácil de implementar y recibir, pero obtiene poco volumen de datos.

Por lo que desde hace unos años se busca cambiar el paradigma y transmitir esta telemetría en diferentes modos digitales, incluyendo también voz digital e imágenes. Por ejemplo, mucha telemetría digital se transmite usando packet radio: 1k2 AFSK AX.25, 9k6 FSK AX.25, incluso 19k2 FSK AX.25. En los siguientes sub-apartados se explica más en profundidad los satélites de radioaficionados que han transmitido alguna vez imágenes en SSDV

3.3.3 JY1SAT

JY1Sat es un FUNcube 1U mejorado. Es capaz de descargar imágenes en formato SSDV. Este formato de imagen, desarrollado por Phil Heron. [18]

La frecuencia del enlace descendente de telemetría es de 145,840 MHz, que utiliza es el formato estándar FUNcube 1k2 BPSK. El transpondedor lineal, para los modos de Banda Lateral Única (SSB) y CW, realizará el enlace descendente en 145,855-145,875 MHz y el ascendente en 435,100-435,120 MHz.

Este satélite además se incorpora en la librería de GNU radio gr-satellites, en el cual podemos ver sus bloques SDR definidos.

Imagen transmitida por este satélite :



Figura 10. Transmisión de imagen en SSDV recibida por JYSAT exitosamente

3.3.4 LONGJIANG-2

Longjiang-2 / LO-94, desarrollada por estudiantes e investigadores del Instituto de Tecnología de Harbin, es la nave espacial más pequeña del mundo en entrar en la órbita lunar de forma independiente. Fue lanzado el 20 de mayo de 2018 y los radioaficionados siguieron su progreso mientras viajaba hacia la Luna y entró con éxito en la órbita lunar.[19]

La nave espacial transmitió señales de vuelta a la Tierra en 435,400 y 436,400 MHz. Se utilizó SSDV para enviar imágenes de la Luna y WSJT JT4G para los mensajes. Se utilizaron dos transceptores SDR VHF/UHF a bordo para proporcionar la baliza, la telemetría, la tele

comando, el enlace descendente de imágenes digitales y un repetidor GMSK-JT4, cuya potencia de transmisión era de unos 2 vatios.

Este satélite además se incorpora en la librería de GNU radio gr-satellites, en el cual podemos ver sus bloques SDR definidos.



Figura 11. Transmisión de imagen en SSDV recibida por Longjiang-2 exitosamente

4. DEFINICIÓN DEL TRABAJO

4.1 JUSTIFICACIÓN

En esta sección se detallan los motivos principales del desarrollo de este proyecto.

4.1.1 CAMBIO A DIGITAL

Como se ha mencionado anteriormente, AMSAT EA es una organización que lleva poniendo satélites en órbita desde hace mucho tiempo, pero todos ellos cuentan con tecnologías analógicas a la hora de realizar el procesamiento de señales y su respectiva transmisión de imágenes. Puede ser de buena práctica el cambio de algunos módulos de analógico a digital, sustituyendo en un futuro el SSTV por la transmisión digital por imágenes SSDV.

Los siguientes puntos explican con mas profundidad por que el uso de SSDV mejorará las trasmisiones de imágenes:

4.1.1.1 Imágenes en Digital

Aprovecharse de un sistema de imágenes como JPEG. Además, si se pierda una línea o un paquete falla se podrán reconstruir la imagen con los paquetes recibidos correctamente.

4.1.1.2 Modulo SSDV en funcionamiento

Esta razón sirve para verificar que esto no es una prueba de concepto, sino que satélites de radioaficionados implementan esta tecnología. Se verificará en el apartado 5, haciendo un estudio de este. Además de que los satélites de AMSAT comparten características esenciales con los mencionados en el apartado 3.3.

4.1.1.3 Optimización del Canal frente a la Señal a Ruido

Como se explica en la motivación del proyecto y se puede ver en las figuras 1,2,3. El sistema actual que se implementa no es capaz de sacar partido a la Señal a ruido del canal. Al transmitir en digital esto se asegura que casi siempre estas tecnologías lo aprovechan casi al máximo.

4.1.1.4 Uso de redundancia

Como se ha cementado previamente sobre la implementación de SSTV el uso de redundancias avanzadas como FEC con Reed-Solomon no han sido implementadas. Por lo que la investigación y simulación de un sistema que lo implemente ayudará a esta organización a una mejor recepción.

4.1.1.5 Recuperación de paquetes

Gracias al uso de tecnología SSDV actualmente algunos satélites de radioaficionados son capaces de poder recuperar frames perdidos en transmisiones de imágenes relevantes. Debido a que el sistema podrá enviar solo algún frames en cuestión si este no se ha recibido. Esto en SSTV es imposible debido a que, si quiero obtener una parte de la imagen transmitida, debería volverla a recibir desde el principio.

4.1.1.6 Implementación de diferentes modulaciones

Como comenta la motivación del proyecto, las modulaciones actuales utilizadas son analógicas, esto hace que la SNR no sea aprovechada al máximo. Pero en cambio con modulaciones digitales se es capaz de obtener muchos mejores resultados. Por lo que se quiere analizar la modulación FSK que implantan los satélites de AMSAT EA y una PSK que utiliza JY1SAT, con diferente bit rates.

4.1.1.7 Optimización del Ancho de Banda

Debido a que la transmisión en FM es aceptada por módulos digitales, la transmisión de imágenes no tiene que ser solo en las bandas de frecuencia marcadas por SSTV. Por lo que se podría estudiar si sobre esa modulación conseguimos utilizar un ancho de banda diferente.

4.1.2 SSDV EN TRANSMISIONES DE SATÉLITES DE RADIOAFICIONADOS

En el estado de la cuestión se ha explicado que ya ha habido mas de dos satélites de radioaficionados que han utilizado SSDV como forma de transmisión de sus imágenes, dejando código abierto y módulos SDR los cuales nos servirán para poder diseñar y simular un sistema de SSDV funcional.

Debido a que los satélites de AMSAT EA, como hemos compartido las mismas características de transmisión en frecuencias y telemetría, se estudiará estas transmisiones SSDV lo cual nos permitirá adquirir el conocimiento necesario para poder desarrollar un sistema de transmisión de imágenes SSDV.

4.2 OBJETIVOS

El objetivo principal de este proyecto está basado en la investigación y un posible desarrollo de una solución que implemente nuevas técnicas de electrónica de Satélites de radioaficionados para mejorar la transmisión de imágenes vía Satélite en formato analógico en la transmisión como digital en la codificación, buscando mejorar y profundizar en aspectos como la adaptación del Bit-rate y SNR disponibles en el canal, disminuir el Packet Error Rate y decisión del tipo de redundancia a utilizar. Este proyecto se sustenta en una colaboración AMSAT-EA y el radioaficionado Pedro Olmos, el director del proyecto. Con su ayuda de ellos se han definido los siguientes objetivos para mejorar la transmisión de imágenes desde sus satélites y probar nuevas técnicas y desarrollos no utilizados por ellos previamente. Para la consecución de este objetivo principal, el proyecto se ha dividido en los siguientes sub-objetivos:

- Estudio de las tecnologías SSTV y SSDV utilizadas en sistemas de radioaficionados
- Investigación de módulos SSDV
- Estudio de transmisiones de imágenes en SSDV desde satélites de radioaficionados
- Diseño de un sistema para la simulación de transmisión imágenes SSDV sobre diferentes modulaciones en banda base
- Análisis de resultados y elección de los modos mas óptimos de transmisión de imágenes en SSDV.

4.3 METODOLOGÍA

Para la realización de este proyecto, se está utilizando las metodologías de trabajos ágiles, de tal forma que se ha dividido el trabajo en sub-tareas que se van cumplimentando cada día, semana o mes. Se está utilizando la plataforma Notion, herramienta para la organización de tareas y también orientada a la documentación. Es ideal para la coordinación de equipos de trabajo y se basa en la metodología Kanban, la cual propone un sistema de uso colaborativo, donde se pueden implementar metodologías Sprint, delegación de tareas.

Para tener una gestión del proyecto eficiente, se ha realizado un tablón de gestión de la realización de las tareas, donde cada tarea puede estar en los siguientes estados de ejecución:

- To Do: tareas a realizar.
- Doing: tareas que están en pleno desarrollo.
- Done: tareas realizadas las cuales han sido testeadas.

Véase visualmente:

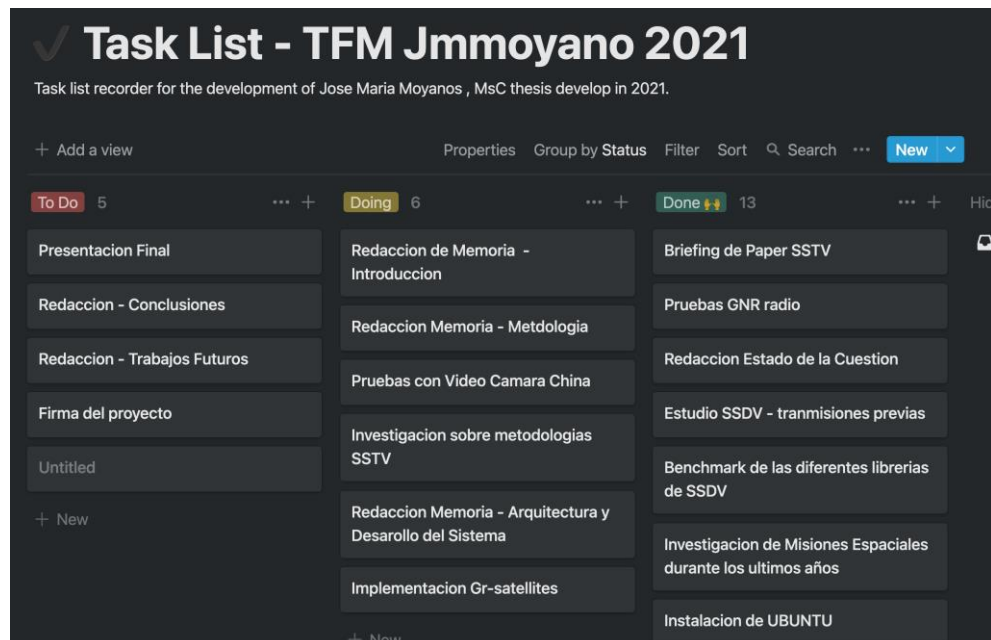


Figura 12. Tablón de tareas

Este sistema ha ayudado a ir cumpliendo los objetivos del proyecto mediante la consecución de las tareas del tablón presentado en la figura anterior.

4.4 PLANIFICACIÓN Y ESTIMACIÓN ECONÓMICA

Para realizar una planificación de las principales tareas e hitos principales que se han ido cumpliendo durante el desarrollo del proyecto, se optó por una presentación temporal que permitiera tener un control de los tiempos de este. Se ha utilizado la plataforma notion, previamente comentada, la cual incluye la posibilidad de implementar diferentes vistas de estos hitos, como son los calendarios, diagramas de Gant o la galería de Hitos dentro del proyecto. [Timeline TFM JMMOYANO - 2021](#)



Figura 13. Diagrama de gant del proyecto

Para la planificación económica del proyecto no se ha desarrollado. Esto se debe a que todas estas tecnologías y software utilizado es open source, siendo gratuitos y su futura implementación dependerá propiamente de AMSAT EA.

5. ESTUDIO DE TRANSMISIÓN DE IMÁGENES EN SSDV

En este capítulo se va a explicar el sistema que actualmente se está implementando en los satélites de radioaficionados que transmiten en SSDV sus transmisiones de imágenes. Como se ha explicado en el estado de la cuestión ya son varios los que lo han realizado, por lo que el objetivo de este apartado es ver las posibles modulaciones, redundancias, formatos y modos de transmisión.

5.1 ANÁLISIS DE UNA TRANSMISIÓN DE UN SATÉLITE EN SSDV

Buscando transmisiones recibidas por algún satélite que hubiera transmitiera SSDV, se ha encontrado un repositorio de estas del satélite chino LONGJIANG-2, llamada la misión DSLWP. Este apartado consistirá en un análisis de la primera transmisión y de una de las muchas transmitidas, para así verificar que la librería de SSDV a utilizar es robusta y además que estos frames/paquetes son recibidos exitosamente la mayoría de las veces.

Como introducción a los sistemas de transmisión del satélite, este utiliza modulación GMSK y bps de 500, la FEC se implementa en la transmisión de telemetría, no dentro de las tramas SSDV, lo cual es una opción en el módulo. Para la recepción de la señal se ha utilizado una frecuencia de muestreo de 40 KHz.

Sistema de trasmision y recepcion de DSLWP

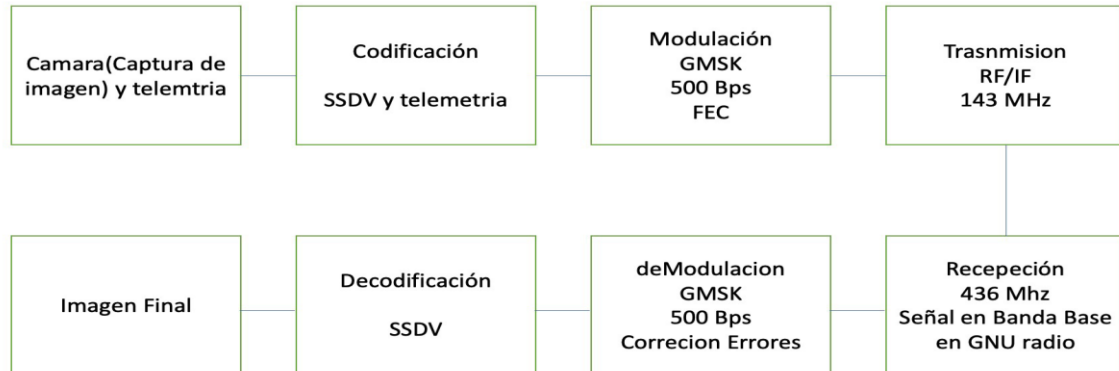


Figura 14. Trasmisor/receptor modos SSDV DSLWP

5.1.1 PRIMERA TRASMISIÓN DSLWP

En este apartado se va a analizar si la primera transmisión de imágenes sobre SSDV del satélite chino, la cual fue la correcta, es decir, el sistema de transmisión funcionó adecuadamente. El análisis se ha realizado en Python 3 sobre un ejemplo descargado del repositorio de imágenes transmitidas por este satélite [20], obtenidas por el telescopio Dwingeloo [21], esperando una buena SNR. Estas grabaciones se han estudiado por Daniel Estévez [22]. Código e instalación en el anexo A del proyecto.

Empezamos con el análisis de la primera transmisión SSDV por el satélite. Este código que se explica a continuación sirve para comprobar que la transmisión de los frames SSDV es la correcta en términos de bits, correlación y longitud del paquete donde se encuentra el frame.

Cargamos las librerías

```
%matplotlib inline
import numpy as np
import scipy.signal
import matplotlib.pyplot as plt
import datetime
```

El archivo que contiene la transmisión GMSK pertinente, contiene 4 tramas SSDV. La señal GMSK de 250bps se convierte a banda base y se filtra a 800Hz.

```
x = np.fromfile('./dslwp_photo_short.c64', dtype='complex64')
fs = 40e3
f = 2410
x = x * np.exp(-1j*2*np.pi*np.arange(x.size)*f/fs)
h = scipy.signal.firwin(1000, 0.01)
x = scipy.signal.lfilter(h, 1, x)
```

Como la SNR de la grabación es muy buena, podemos realizar una demodulación no coherente de la señal GMSK. La figura siguiente muestra algunos de los bits GMSK.

```
plt.plot(np.diff(np.unwrap(np.angle(x[500000:510000]))))
plt.ylim([-0.02,0.02]);
```

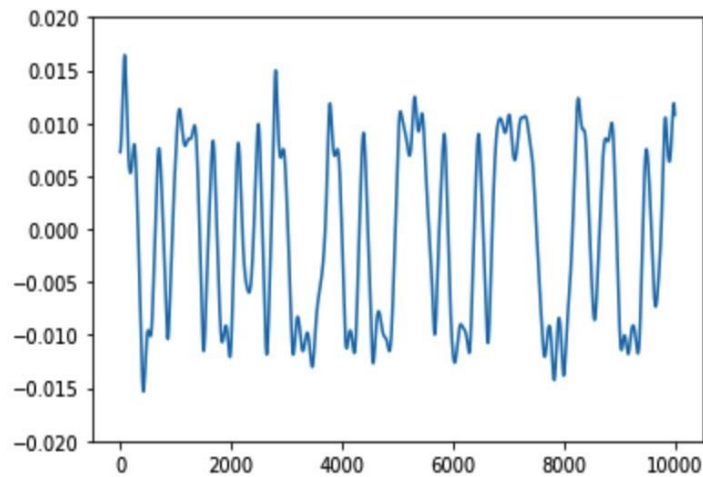


Figura 15. Muestra de Bits en GMSK

Extraemos los bits blandos, con la fase correcta del reloj. No se ha realizado ningún filtrado de forma de pulso, y el seguimiento de la frecuencia del reloj tampoco parece necesario. La separación entre los bits 1 y 0 es suficientemente buena para la demodulación sin errores de bit, como se muestra en la siguiente figura.

```
phase = 50
softbits = np.diff(np.unwrap(np.angle(x)))[phase::160]
plt.plot(softbits, '.')
plt.ylim([-0.02,0.02])
plt.title('Diagrama de Ojo - Primera trasmision SSDV');
```

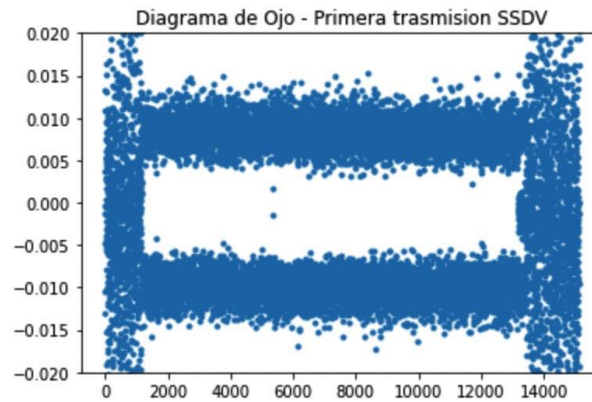


Figura 16. Diagrama de ojo primera transmisión SSDV DSLWP

Correlacionamos los bits duros recibidos con el ASM precodificado. La longitud del ASM es de 63 bits por definición. Como el ASM tiene 64 bits de longitud y los códec se transmiten uno detrás de otro, sin ningún espacio entre ellos, esperamos ver una serie de picos en la correlación de los ASM, que corresponderán con los frames SSDV. Al parecer uno de los frames no esta correcto esto se debe a la demodulación gmsk y el baudrate.

```
asm_corr = scipy.signal.correlate(2*bits-1, 2*asm_diff.astype('float')-1)
plt.plot(asm_corr)
```

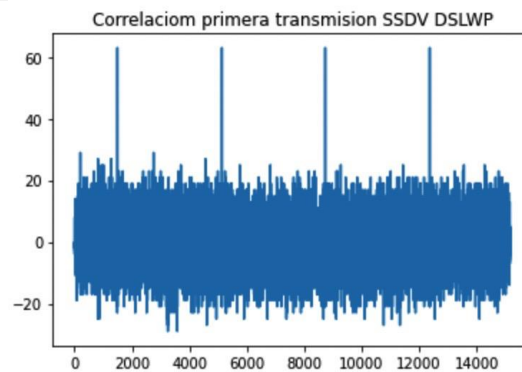


Figura 17. Correlación primera transmisión SSDV DSLWP

Se han obtenido 3 frames SSDV correctamente y uno con problemas, pero la explicación va mas allá de lo que concierne a este proyecto.

5.1.2 TRASMISIÓN ESTÁNDAR DSLWP

Como se comento previamente, contamos con un repositorio de transmisiones del satélite, por lo que ha elegido una transmisión que cumpliera con los requisitos su posterior recepción. Se ha elegido la siguiente transmisión DSLWP-B 201-08-12 SSDV[20].

Se va a realizar el mismo estudio que en apartado anterior, pero con una transmisión que contiene 117 paquetes SSDV.

Cargamos un archivo que contiene la transmisión GMSK correspondiente.

```
x = np.fromfile('/Users/jmmoyano/Downloads/DSLWP-B_PI9CAM_2018-08-12T06-58-02_436.4MHz_40ksps_complex.raw', dtype='complex64')
```

La señal GMSK de 250bps se convierte en banda base y se filtra en paso bajo a 800Hz.

```
fs = 40e3  
f = 1500  
x = x * np.exp(-1j*2*np.pi*np.arange(x.size)*f/fs).astype('complex64')  
h = scipy.signal.firwin(1000, 0.01).astype('float32')  
x = scipy.signal.lfilter(h, 1, x).astype('complex64')
```

Se realiza la demodulación GMSK y se corrige la fase.

```
s = np.diff(np.angle(x).astype('float32'))  
s[s > np.pi] -= 2*np.pi  
s[s < -np.pi] += 2*np.pi
```

Extraemos los bits soft adivinando la fase correcta del reloj y decimando. No se ha realizado ningún filtrado de forma de pulso, y el seguimiento de la frecuencia del reloj tampoco parece necesario. La separación entre los bits 1 y 0 es suficientemente buena para la demodulación sin errores de bit. Se puede observar en la siguiente figura que esta vez hay bastante menos distorsión en el diagrama de ojo.

```
phase = 50  
softbits = s[np.int32(np.arange(phase, s.size, 160 - 0.0005))]
```

```
softbits = softbits + 1e-8 * np.arange(softbits.size) - 0.005 # correction for
frequency offset and drift
plt.plot(softbits, '.')
plt.axhline(y = 0, color='green')
plt.ylim([-0.03,0.03])
plt.title('Diagrama de ojo Tramision DSLWP 2');
```

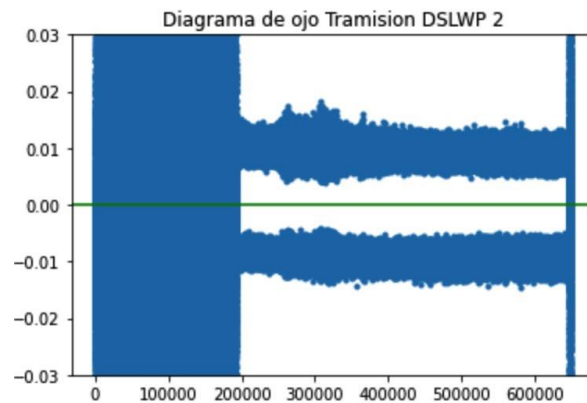


Figura 18. Diagrama de ojo segunda transmisión SSDV DSLWP

Los bits blandos se convierten ahora en bits duros. Construimos el ASM CCSDS que se utiliza para marcar el inicio de cada palabra codificada con Turbo. El ASM está precodificado como indica la norma CCSDS. Correlacionamos los bits hard recibidos con el ASM precodificado. La longitud del ASM es de 63 bits, por lo que una correlación de 63 indica que el ASM se encuentra sin errores de bit. Vemos que el ASM se encuentra 4 veces sin errores de bit. Cada una de estas ocurrencias del ASM marca el inicio de una codificación Turbo que contiene una sola trama SSDV.

```
bits = (softbits > 0) * 1
asm = np.unpackbits(np.array([0x03,0x47,0x76,0xC7,0x27,0x28,0x95,0xB0],
dtype='uint8'))
asm_diff = asm[:-1] ^ asm[1:]
asm_diff[1::2] ^= 1
asm_corr = scipy.signal.correlate(2*bits-1, 2*asm_diff.astype('float')-1)
plt.plot(asm_corr)
```

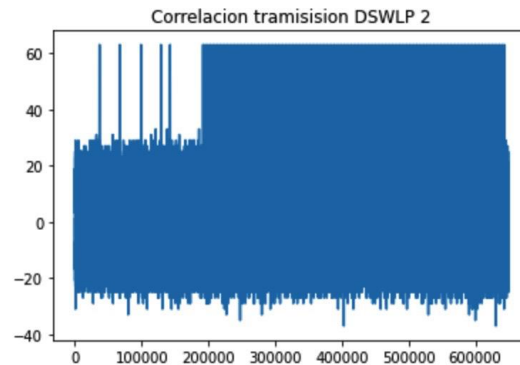


Figura 19. Correlación ASM segunda transmisión SSDV DSLWP

```
asm_corr[asm_corr > 40].size  
resultado -> 130
```

Hemos encontrado un total de 130 ASM. De ellos, 117 marcan paquetes SSDV, de los cuales unos cuantos no fueron exitosos

5.2 DISEÑO E IMPLEMENTACIÓN DEL RECEPTOR DE IMÁGENES

Para la recepción de imágenes por ordenador, los radioaficionados utilizan la Software Defined Radio (SDR), GNU-RADIO. En este caso en concreto se utilizará el módulo gr-satellites, especialmente diseñado para transmisiones de telemetría por satélite. Una breve descripción de este es la instalación de las librerías necesarias y componentes básicos para una recepción exitosa de telemetría de un satélite de radioaficionados. Este módulo implementa los receptores complejos de la misión DSLWP y del satélite JY1SAT. Se han estudiado y probado los módulos de recepción de imágenes de estos satélites y se han recibido correctamente las imágenes.

5.2.1 RECEPTOR DE TRAMAS DSLWP

Como se muestra en el diagrama de transmisión/recepción de DSLWP, se ha utilizado su sistema receptor en gr-satellites para la segunda transmisión analizada en el apartado anterior de 2018. La figura a continuación muestra el sistema definido en GNU radio que encontramos en gr-satellites para la recepción. Para este caso en concreto se utilizará la

transmisión mencionada previamente la cual es filtrada, de modulada, se le aplica corrección de errores. Mas tarde se taguea y se escriben las tramas SSDV en un escritorio temporal, para su posterior conversión a JPEG gracias al modulo SSDV.

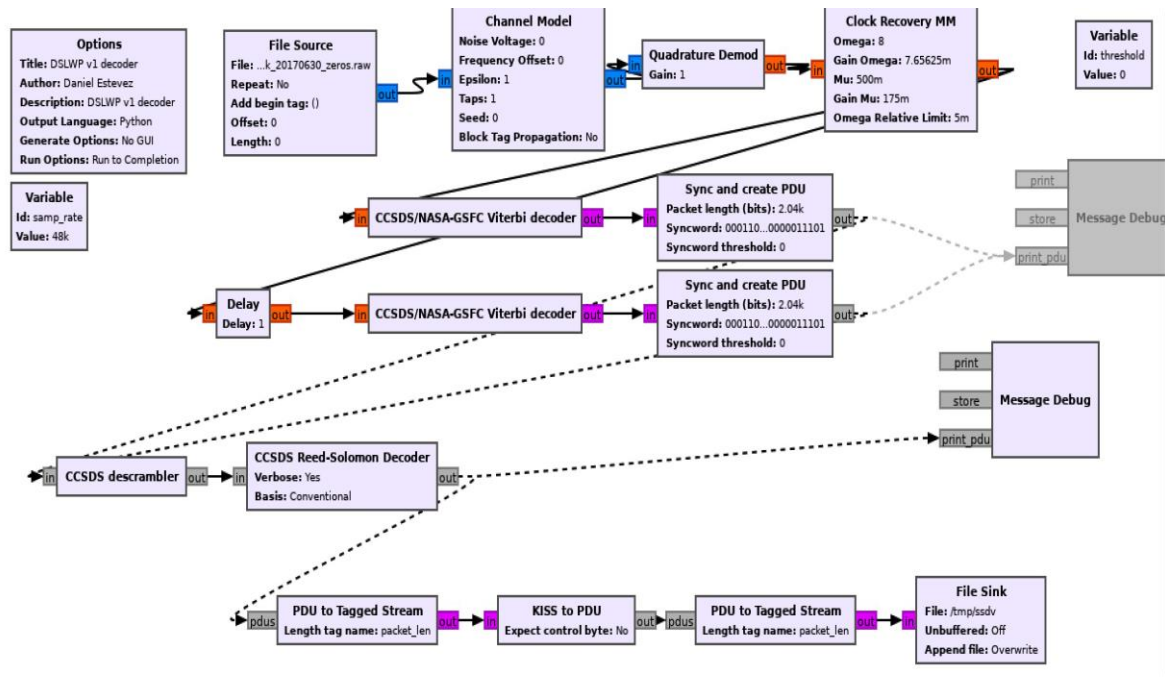


Figura 20. Diagrama de bloques recepto Imágenes SSDV DSWLP

Ahora con los frames ya recibidos toca convertirlos en una imagen JPEG, por lo que utilizaremos el modulo SSDV instalado.

```
SSDV -e -D ssdv_frames.bin ./output_dswlp.jpeg
```

Obteniendo la imagen en JPEG.

5.2.2 RECEPTOR DE TRAMAS JY1SAT

Como se comento previamente en el estado de la cuestión el satélite JY1SAT también incluía un modulo de transmisión de imágenes en SSDV. Se ha conseguido obtener una transmisión obtenida por un radioaficionado en un .wav. Esta transmisión intercala voz CODEC2 y 4 paquetes SSDV.

El transmisor y receptor son descritos en la siguiente imagen, utilizando la modulación BPSK y con un bit rate de 250 Bps.

Sistema de trasmision y recepcion de **JY1SAT**

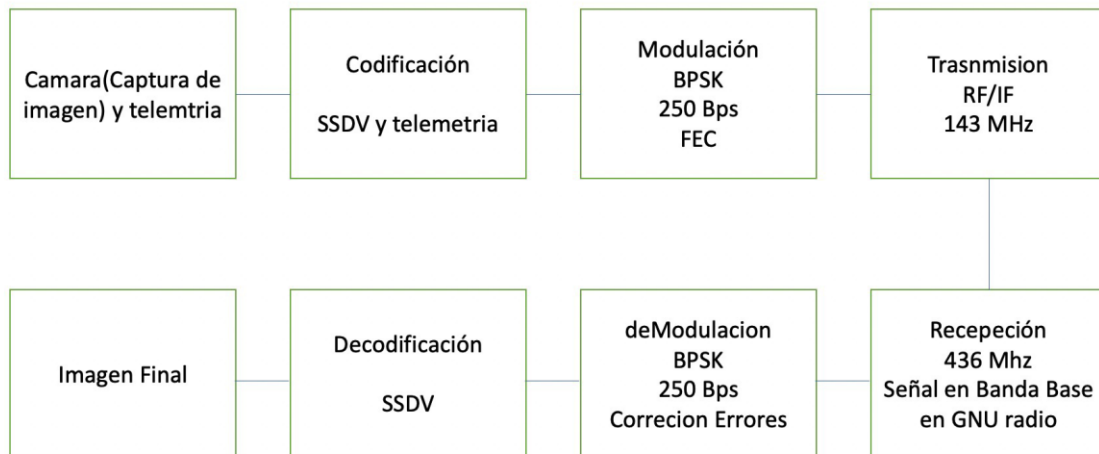


Figura 21. Trasmisor/receptor modos SSDV JY1SAT

La figura a continuación muestra el sistema definido en GNU radio que encontramos en gr-satélites para la recepción. Para este caso en concreto se utilizará la transmisión mencionada previamente la cual, será filtrada, demodulada, se seleccionarán los paquetes SSDV, se le aplicará la corrección de errores y más tarde los paquetes serán etiquetados y escritos en el escritorio temporal.

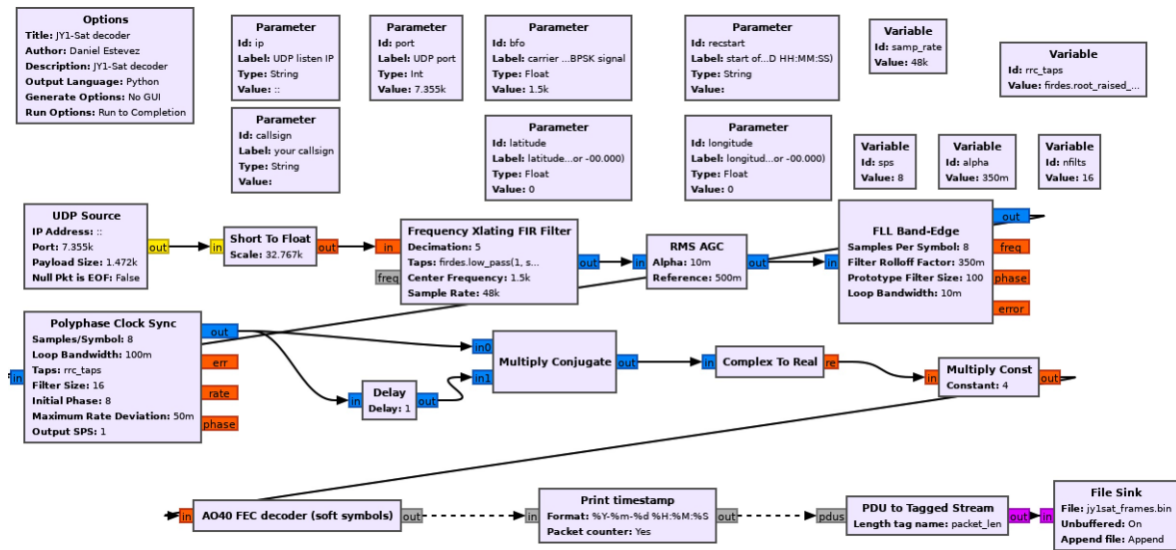


Figura 22. Diagrama de bloques receptor JY1SAT

Una vez hemos recibido los frames correctamente, se hace un filtrado de los paquetes SSDV debido a que se conoce la cabera explicada en el formato explicado en el capítulo 3. Más tarde se realiza la decodificación de estos frames SSDV.

```
jy1sat_ssdrv.py jy1sat_frames.bin jy1sat_output.jpg
```

Obteniendo la imagen en JPEG.

5.3 RESULTADOS Y CONCLUSIONES

5.3.1 RESULTADOS

En este apartado analizaremos los resultados de las transmisiones SSDV de los diferentes satélites.

Para DSWLP hemos podido comprobar los diagramas de ojo de las transmisiones son muy buenos, esto se puede deber a que la señal fue captada por Dwingeloo, muy buena SNR, pero aun así haciendo un análisis de la correlación de los bits y contabilizando la longitud se puede catalogar como éxito. Aun, así como se muestra continuación se pierden algunos paquetes SSDV, pero es algo de esperar en todo tipo de comunicaciones.

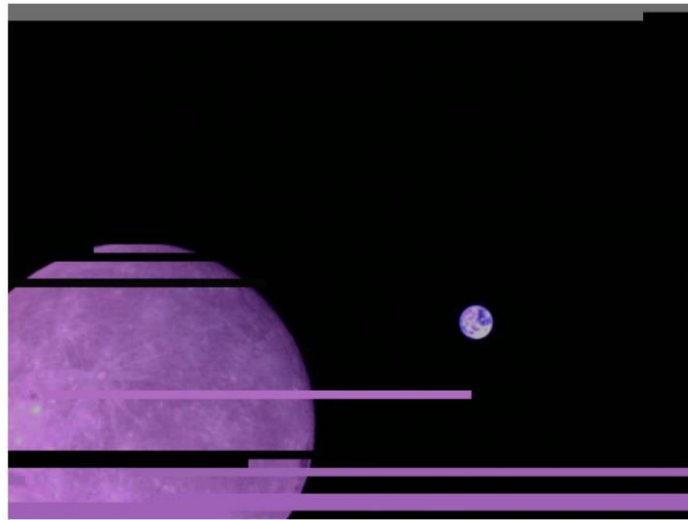


Figura 23. Imagen SSDV recibida DSWLP procesada por gr-satellites

Para el satélite JY1SAT hemos obtenido la siguiente imagen, véase en la siguiente figura. Esta es una transmisión que capto un radioaficionado y no posee la calidad SNR adecuada, pero nos sirve para poder comprobar que los sistemas funcionan y que la recepción de frames y su posterior conversión a imagen es exitosa.



Figura 24. Imagen SSDV recibida DSWLP procesada en gr-satellites

5.3.2 CONCLUSIONES

Como se puede comprobar las primeras transmisiones SSDV fueron totalmente un éxito. Comprobando que estos paquetes SSDV llegan correctamente, gracias al estudio de las correlaciones y longitud de los paquetes desmodulados, para mas tarde introducirlos en un receptor. Este estudio nos ayuda a ver que transmisiones son optimas para su decodificación.

Para DSWLP, sus resultados no pueden ser mejores y solo se puede catalogar que su sistema de transmisiones es muy robusto. La modulación utilizada es GMSK, con un bit rate de 500 bps, utilizando FEC con reed Solomun, en este caso el mismo que utiliza el formato SSDV, pero en este caso se aplica previamente debido a que en las tramas se encuentran datos de telemetría. Sus resultados no pueden ser mejores y su sistema implementado robusto. Pero el sistema es demasiado complejo, por lo que utilizar GMSK queda descartado del proyecto.

Para JY1SAT hay que destacar su éxito debido a que se han recibido numerosos frames correctamente. Por lo que utilizaremos sus parámetros de transmisión y recepción para realizar una transmisión con a modulación utilizada es BPSK con 250 Bps y implementa FEC Reed Solomun.

6. DISEÑO DEL SISTEMA DE TRANSMISIÓN DE IMÁGENES EN SSDV

En este capítulo se explica el diseño un sistema que simule la transmisión de imágenes sobre el modo SSDV. Este sistema esta pensado para ser simulado en un ordenador, gracias al uso del lenguaje de programación Python, para ver como seria una transmisión de una imagen en el modo SSDV y su posterior recepción. Debido a que no se ha podido disponer de HW de radiofrecuencia para simular la transmisión con los parámetros reales, el sistema se ha diseñado para una transmisión en Banda base.

En este caso el sistema se ha diseñado para poder simular la transmisión y recepción de una imagen JPEG en el modo SSDV, implementando las siguientes modulaciones:

- BPSK-QPSK, modulación en fase
- BFSK y QFSK., modulación en frecuencia.

Se han elegido estas modulaciones debido a que la modulación FSK es implementada actualmente por los satélites de AMSAT EA, como se comentaba en la justificación del proyecto y el uso de modulación PSK es debido a que es utilizada por JY1SAT, argumentado en la conclusión del capítulo 5.

6.1 ANÁLISIS DEL SISTEMA

En este apartado se va a explicar el sistema desarrollado en una maquina local, sin ningún tipo de componente analógico. El sistema ha sido desarrollado en Python 3, con la ayuda de las librerías numpy (tratamiento de arrays), scipy(tratamiento de señales, filtros, FFT) y matplotlib (visualización).

En primer lugar, se ha diseñado un transmisor de imágenes SSDV. Este consta de un codificador de imágenes utilizando el modulo SSDV y mas tarde un bloque de modulación. Este bloque necesitará una serie de parámetros de entrada para poder modular la imagen codificada, las cuales son las siguientes:

- frecuencia de muestreo, f_s
- Bit rate, bit_rate
- T_b , periodo de bit , $T_b = 1/bit_rate$
- Frecuencia de la portadora, f_c
- Energía de Bit, E_b

Una vez son configurados estos parámetros de transmisión se modula la señal con el modo elegido :

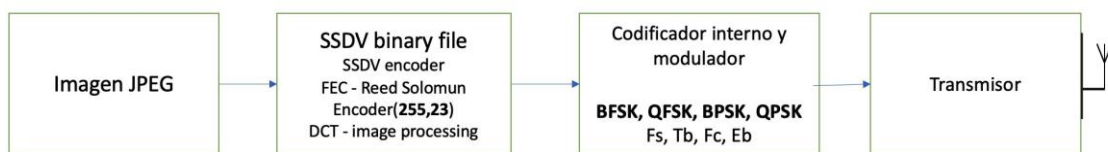


Figura 25. Transmisor imágenes en SSDV para diferentes modulaciones

En segundo lugar, se ha diseñado un receptor de imágenes SSDV sobre las diferentes modulaciones mencionadas previamente. Este bloque funciona igual que el modulador, necesita una serie de parámetros de recepción para poder demodular la señal de forma correcta. Estos serán los mismos que los del modulador. Una vez demodulamos la señal, obtenemos los paquetes SSDV, los cuales son tageados y para que estos ser procesados por el decodificador de SSDV, el cual aplicará la corrección de errores Reed Solomon. Una vez procesados todos los paquetes el sistema convertirá los mismos en una imagen JPEG.

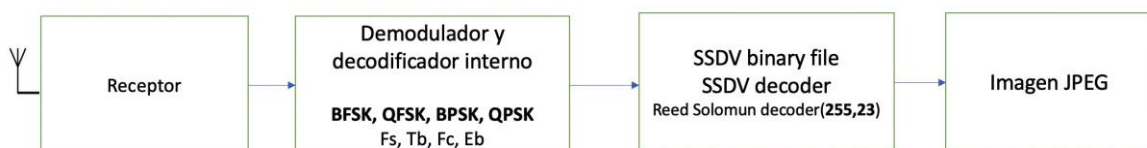


Figura 26. Receptor imágenes en SSDV para diferentes modulaciones

Habiendo explicado previamente como funcionan los transmisores y receptores, se implementan en serie en un mismo sistema, debido a que es una simulación y no un escenario real de transmisión. Por lo que se transmite sobre un canal con Ruido blanco Gaussiano, el cual es ajustado por la SNR. Véase en la siguiente figura:

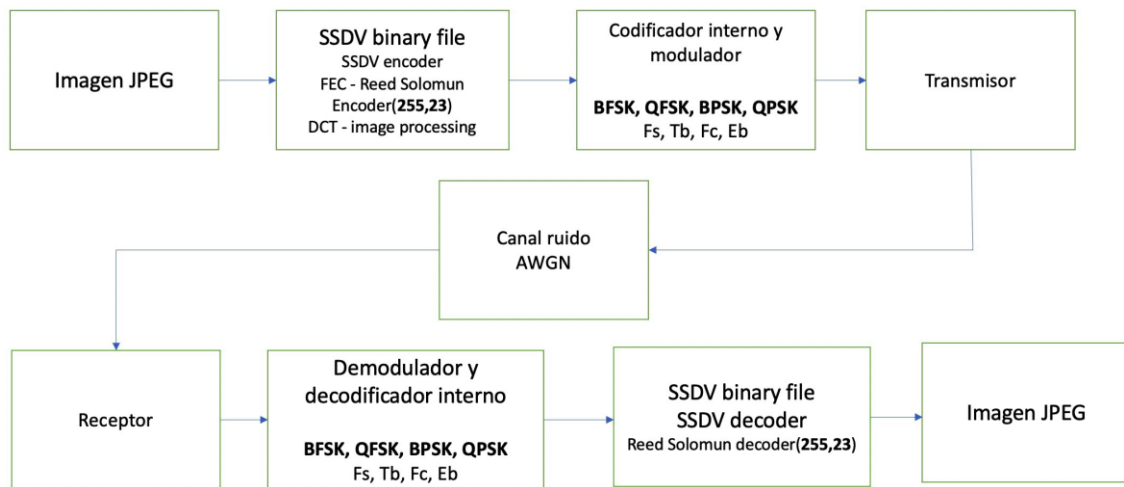


Figura 27. Sistema de transmisión/recepción de imágenes SSDV aplicando SNR en dB

6.2 PARÁMETROS DE TRANSMISIÓN Y RECEPCIÓN

Una vez hemos explicado como funciona el sistema en su totalidad, definimos los parámetros de transmisión del transmisor y recepción con los que será probado el sistema. Se van a simular numerosos escenarios de Bit rate para poder sacar conclusiones de si en estas velocidades es óptimo transmitir o si directamente el ancho de banda no es compatible. Como se ha comentado en apartados anteriores el sistema SSTV transmitía en 50 bps, el satélite JY1SAT en 250 Bps y la misión DSWLP en 500 Bps. Por lo que simulará a los siguientes bits rates : 50, 250, 500, 1000. Se utilizará la misma E_b para todos los sistemas, calculada según la Potencia de la portadora.

Para las modulaciones PSK y FSK, se utilizará una frecuencia de portadora de 1510 Hz, debido a que se encuentra justo en medio de la banda de transmisión SSTV además de ser

optima para este tipo de modulación. Además, la banda en digital empieza en 300 Hz, por lo que la elección de este parámetro cumple con ese requisito.

Modulation	Bit rate [bps]	Band Width [Hz]	Sample frequency [Hz]	Carrier frequency [Hz]
BPSK	50 , 250, 500,1000	50 ,100, 250, 500,1000	48000	1510
QPSK	50,250, 500, 1000	50 ,100, 250, 500,1000	48000	1510
BFSK	50, 250, 500, 1000	100,500,1000,2000	48000	1510
QFSK	50,, 250, 500, 1000	100,500,1000,2000	48000	1510

Tabla 3. Paramtros Modulacion/Demodulacion del Sistema

6.3 DISEÑO DEL TRANSMISOR

Como se ha explicado previamente el trasmisor incluye dos grandes módulos, el de la codificación y compresión de una imagen JPEG a formato SSDV y el de la modulación de estos datos para su posterior transmisión.

6.3.1 CODIFICADOR DE IMAGEN

El codificador de la imagen sigue los siguientes pasos. Lo primero realiza la compresión de la imagen JPEG según el estándar, para mas tarde paquetizarlo como se refleja en la figura 9. Este modulo mientras cada paquete/frame le añadirá FEC reed Solomon. Mas tarde se comprobará que la estructura del paquete es la optima y se traducirá de bytes a bits para su modulación. Véase en la siguiente figura una explicación detallada:

(PM) es que mientras en esta la variación de fase es continua, en función de la señal moduladora, en la PSK la señal moduladora es una señal digital y, por tanto, con un número de estados limitado.[24]

Esta modulación se ha diseñado y programado en python según el siguiente diagrama de bloques :

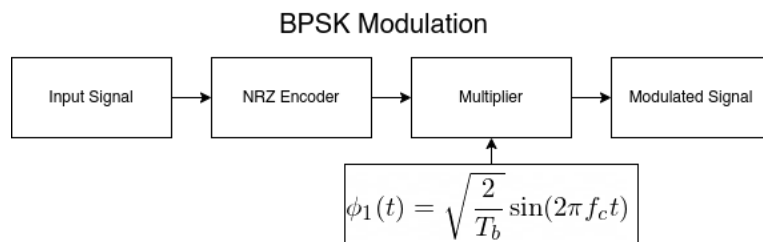


Figura 29. Diagrama de bloques modulación BPSK[23]

Una vez implementada, modulamos los bits de salida con los parámetros de modulación indicados en la tabla, teniendo una modulación BPSK exitosa :

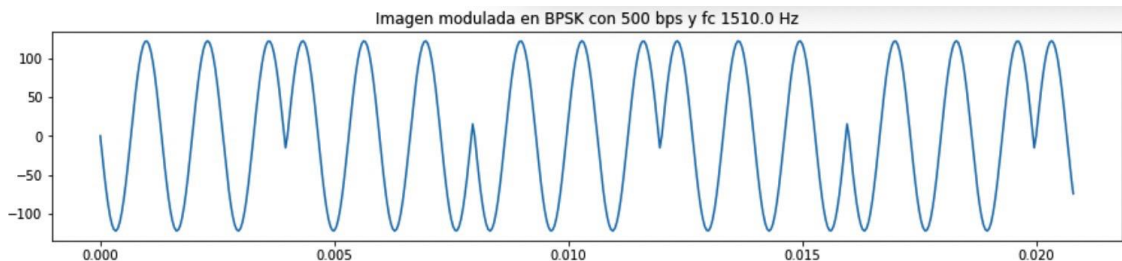


Figura 30. Señal modulada modulada en BPSK

Vemos que la modulación es la correcta estudiamos su modulación incluyendo a la señal modulada ruido blanco gaussiano :

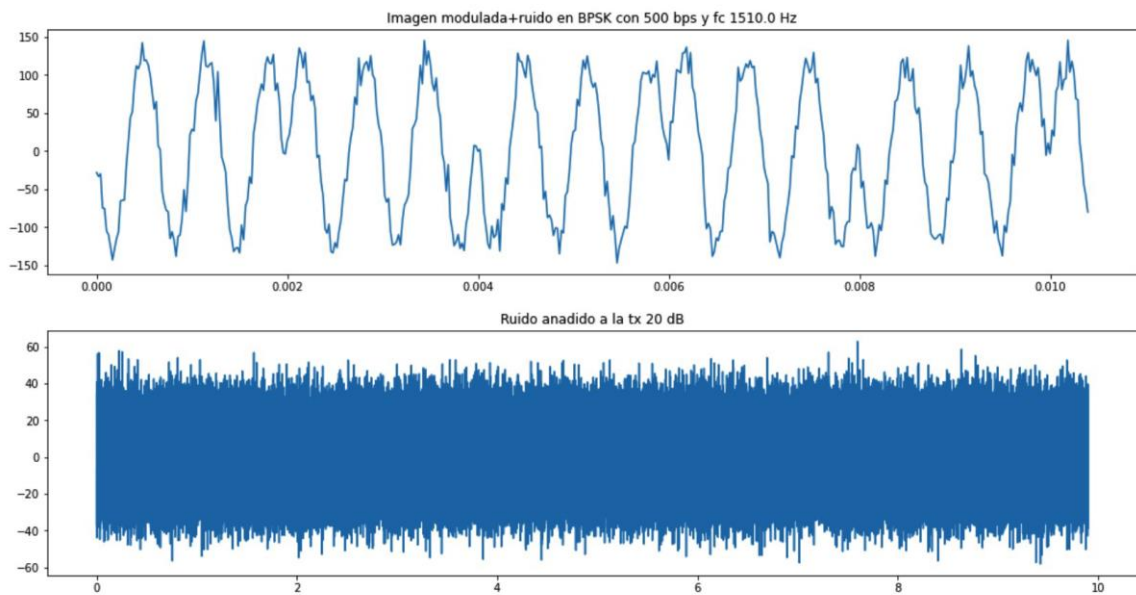


Figura 31. Señal modulada+ruido en BPSK

Ahora vemos como se comporta en el dominio de la frecuencia, tanto en su transformada discreta de Fourier como en un espectrograma :

BPSK en el dominio de la frecuencia

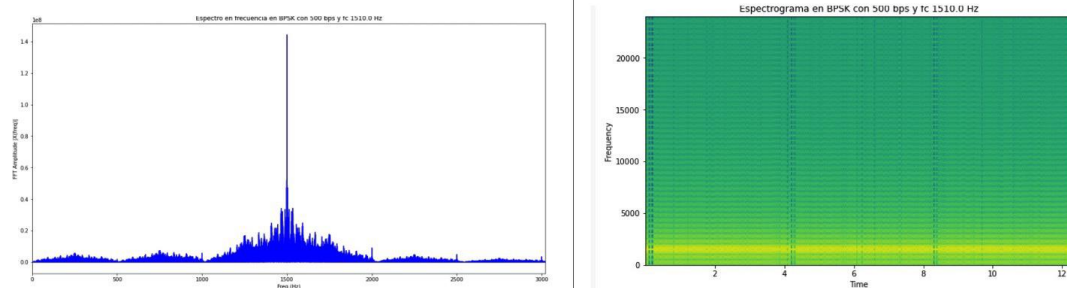


Figura 32. Modulación BPSK en el dominio de la frecuencia

Vemos que el análisis en frecuencia es el esperado.

Vemos como es el diagrama de ojo de la modulación en cuestión :

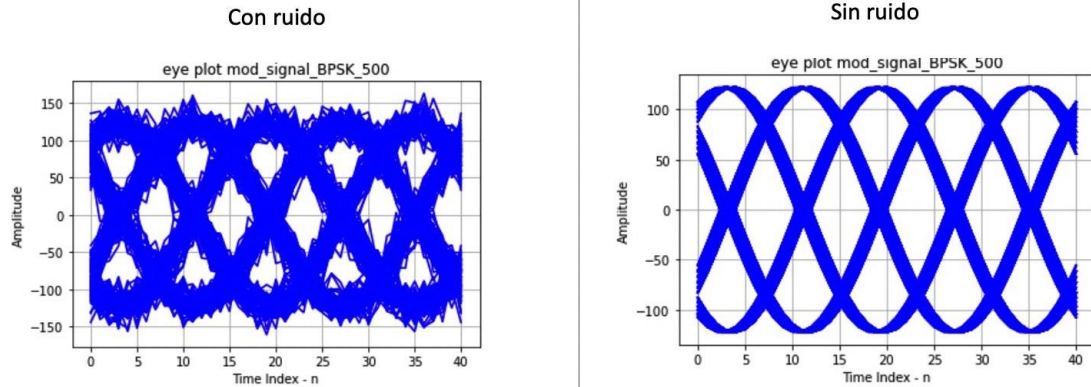


Figura 33. Diagramas de ojo de BPSK

Se concluye que la modulación funciona correctamente.

6.3.4 MODULACIÓN QPSK

Este esquema de modulación es conocido también como Quaternary PSK (PSK Cuaternaria). Esta modulación digital es representada en el diagrama de constelación por cuatro puntos equidistantes del origen de coordenadas. Con cuatro fases, QPSK puede codificar dos bits por cada símbolo. La asignación de bits a cada símbolo suele hacerse mediante el código Gray, que consiste en que, entre dos símbolos adyacentes, los símbolos solo se diferencian en 1 bit, con lo que se logra minimizar la tasa de bits erróneos.[24]

Esta modulación se ha diseñado y programado en python según el siguiente diagrama de bloques :

QPSK Modulation

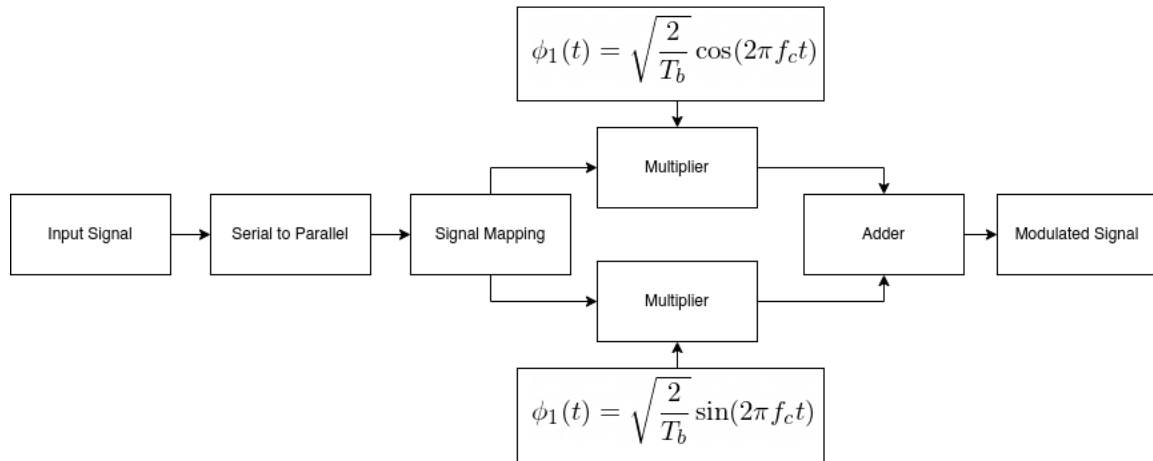


Figura 34. Diagrama de bloques modulación QPSK[23]

Una vez implementada, modulamos los bits de salida con los parámetros de modulación indicados en la tabla, teniendo una modulación QPSK exitosa :

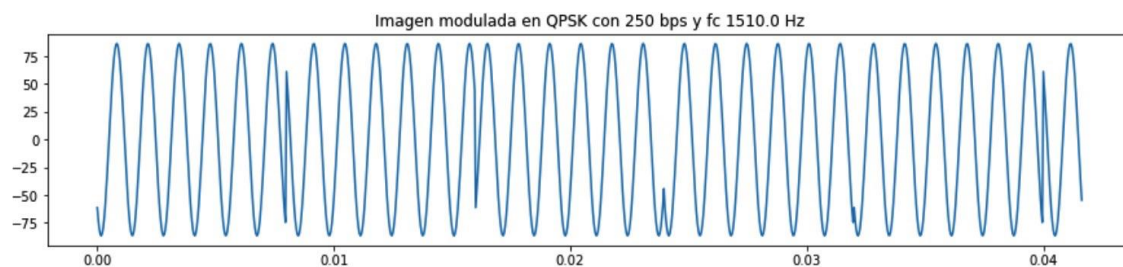


Figura 35. Señal modulada en QPSK

Vemos que la modulación es la correcta estudiamos su modulación incluyendo a la señal modulada ruido blanco gaussiano :

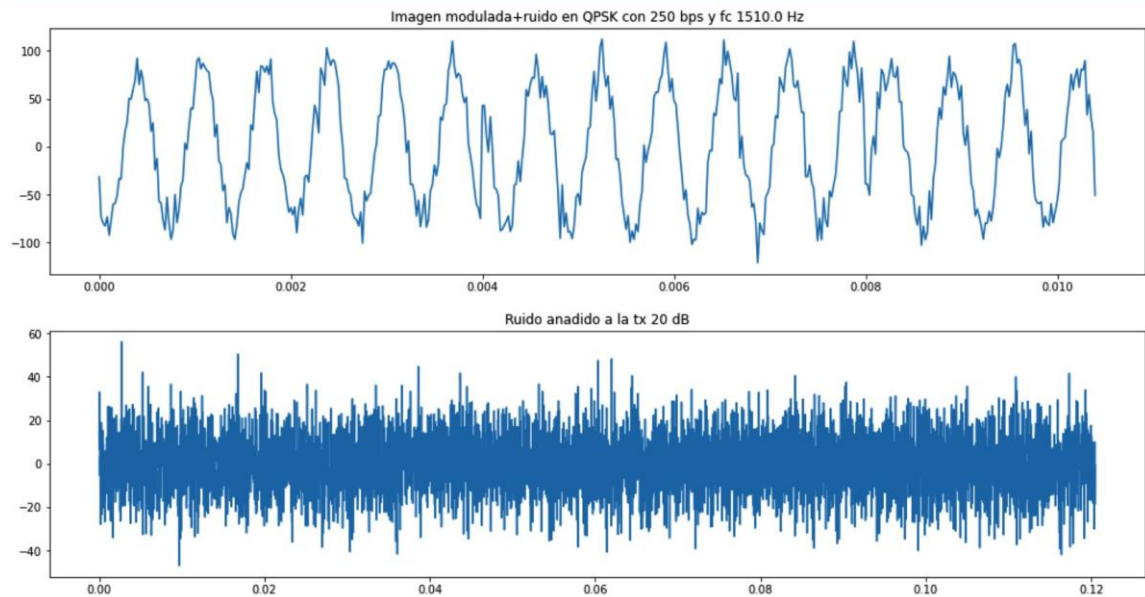


Figura 36. Señal modulada+ruido en QPSK

Ahora vemos como se comporta en el dominio de la frecuencia, tanto en su transformada discreta de Fourier como en un espectrograma :

QPSK en el dominio de la frecuencia

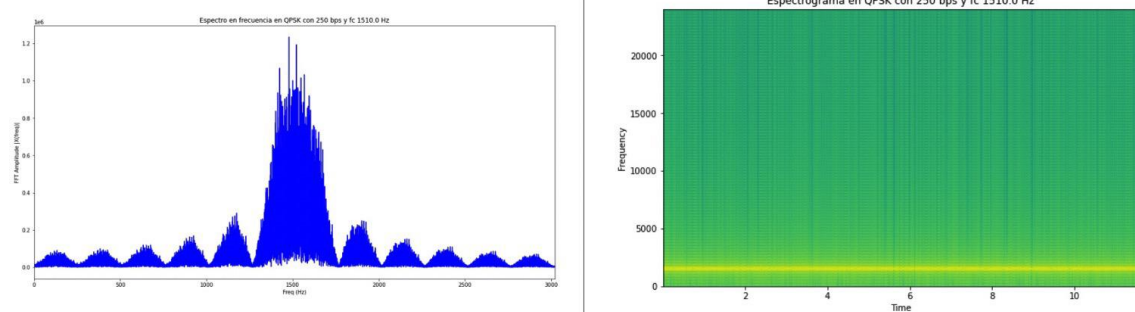


Figura 37. Modulación QPSK en el dominio de la frecuencia

Vemos como es el diagrama de ojo de la modulación en cuestión :

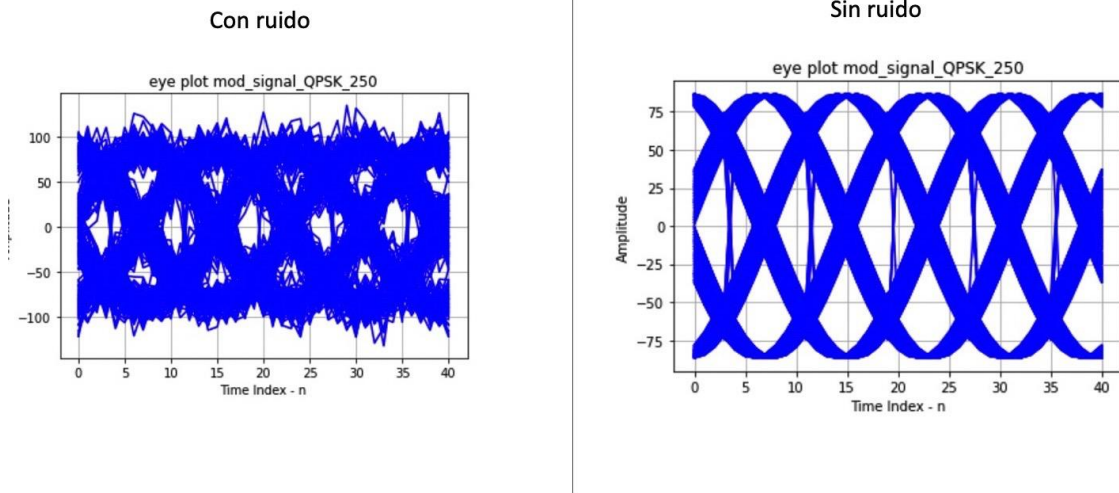


Figura 38. Diagramas de ojo de QPSK

Se concluye que la modulación funciona correctamente.

6.3.5 MODULACIÓN BFSK

La modulación por desplazamiento de frecuencia o FSK —del inglés Frequency Shift Keying— es una técnica de modulación para la transmisión digital de información utilizando dos o más frecuencias diferentes para cada símbolo.¹ La señal moduladora solo varía entre dos valores de tensión discretos formando un tren de pulsos donde uno representa un "1" o "marca" y el otro representa el "0" o "espacio". En la modulación digital, a la relación de cambio a la entrada del modulador se le llama bit-rate y tiene como unidad el bit por segundo (bps). A la relación de cambio a la salida del modulador se le llama baud-rate. En esencia el baud-rate es la velocidad o cantidad de símbolos por segundo. En FSK, el bit rate = baud rate. Así, por ejemplo, un 0 binario se puede representar con una frecuencia f_1 , y el 1 binario se representa con una frecuencia distinta f_2 . [25]

Esta modulación se ha diseñado y programado en python según el siguiente diagrama de bloques :

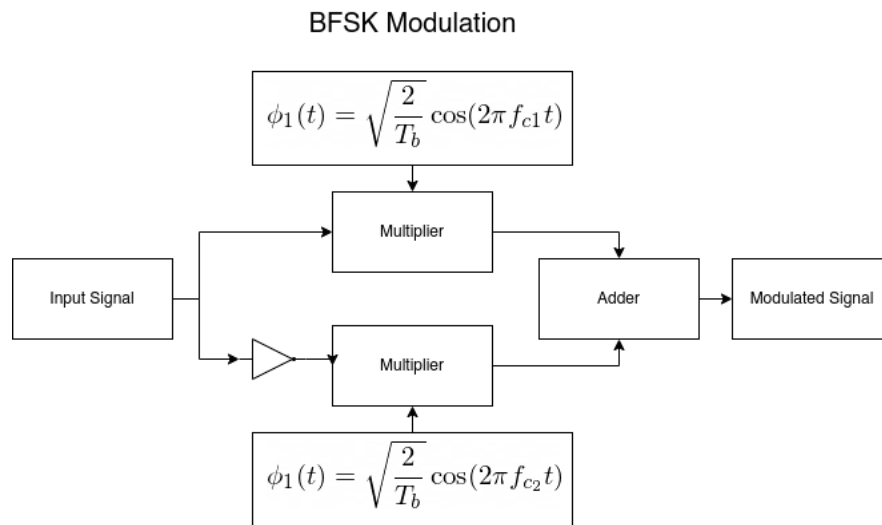


Figura 39. Diagrama de bloques modulación BFSK[23]

Una vez implementada, modulamos los bits de salida con los parámetros de modulación indicados en la tabla, teniendo una modulación BFSK exitosa :

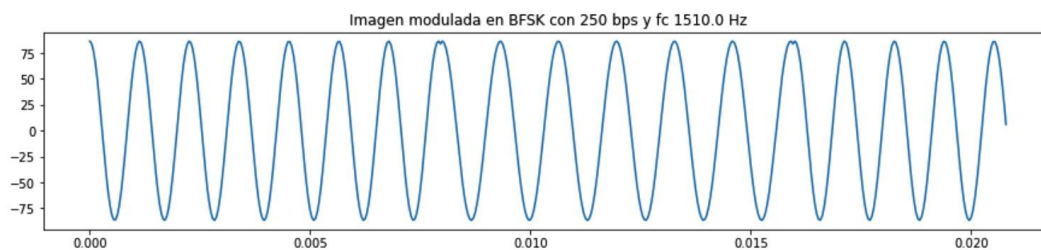


Figura 40. Señal modulada en BFSK

Vemos que la modulación es la correcta estudiamos su modulación incluyendo a la señal modulada ruido blanco gaussiano :

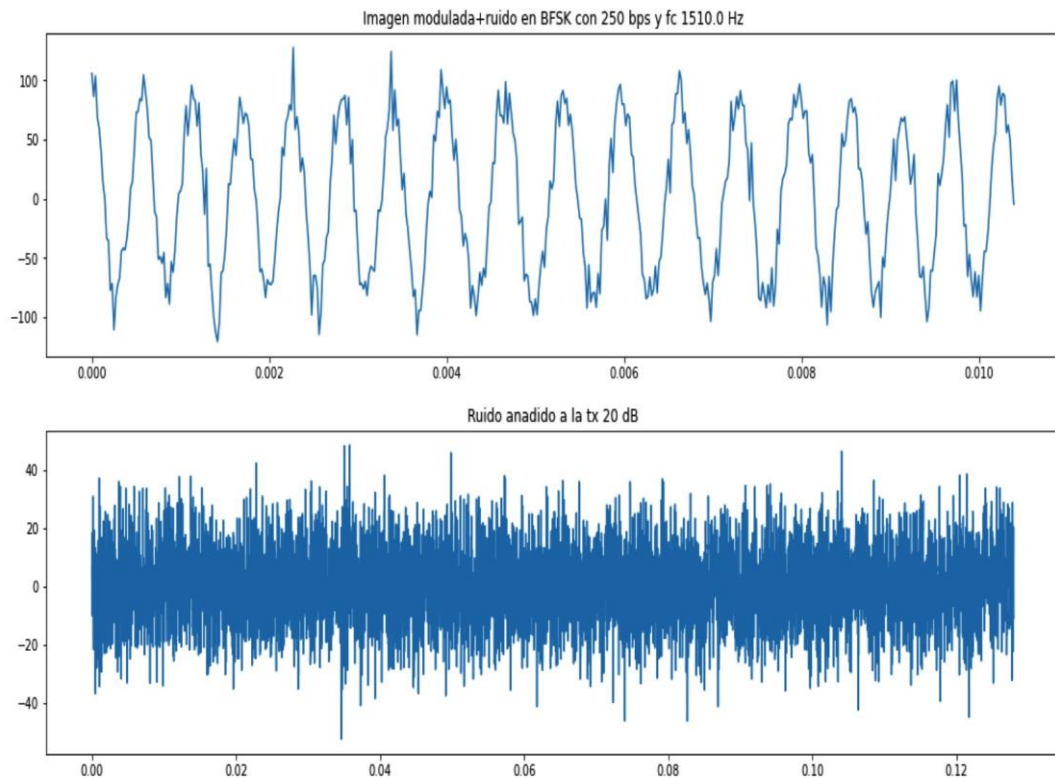


Figura 41. Señal modulada+ruido en BFSK

Ahora vemos como se comporta en el dominio de la frecuencia, tanto en su transformada discreta de Fourier como en un espectrograma :

BFSK en el dominio de la frecuencia

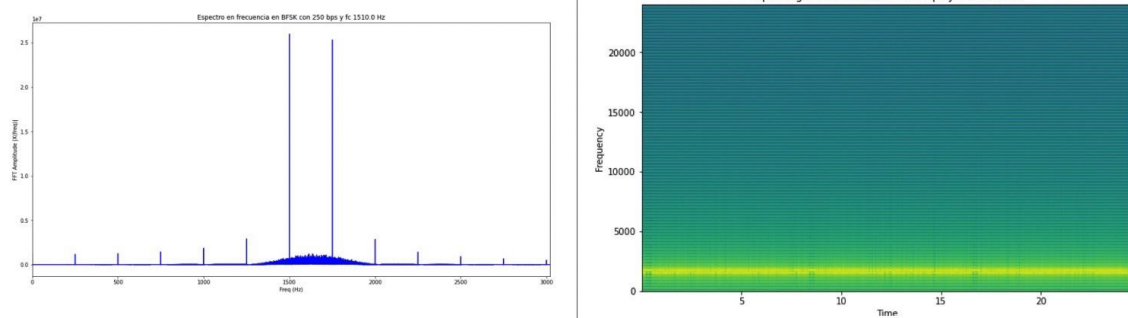


Figura 42. Modulación BFSK en el dominio de la frecuencia

Vemos como es el diagrama de ojo de la modulación en cuestión :

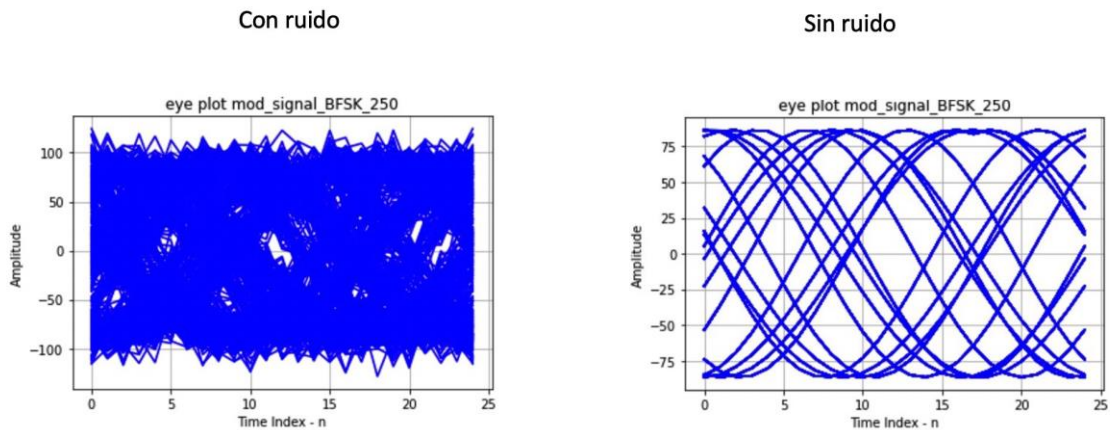


Figura 43. Diagramas de ojo de BFSK

Se concluye que la modulación funciona correctamente.

6.3.6 MODULACIÓN QFSK

Esta modulación se ha diseñado y programado en python según el siguiente diagrama de bloques :

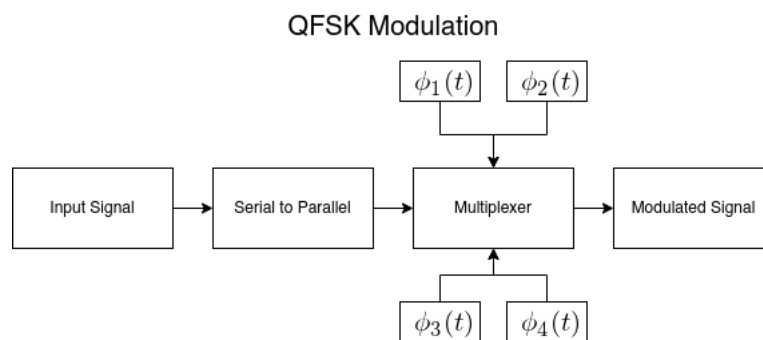


Figura 44. Diagrama de bloques modulación QFSK[23]

Una vez implementada, modulamos los bits de salida con los parámetros de modulación indicados en la tabla, teniendo una modulación QFSK exitosa :

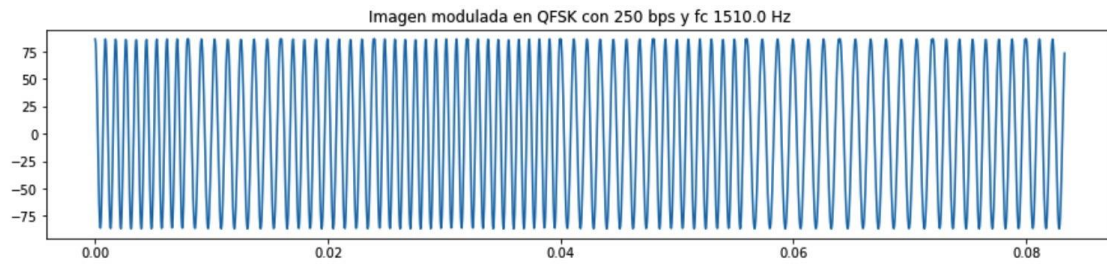


Figura 45. Señal modulada en QFSK

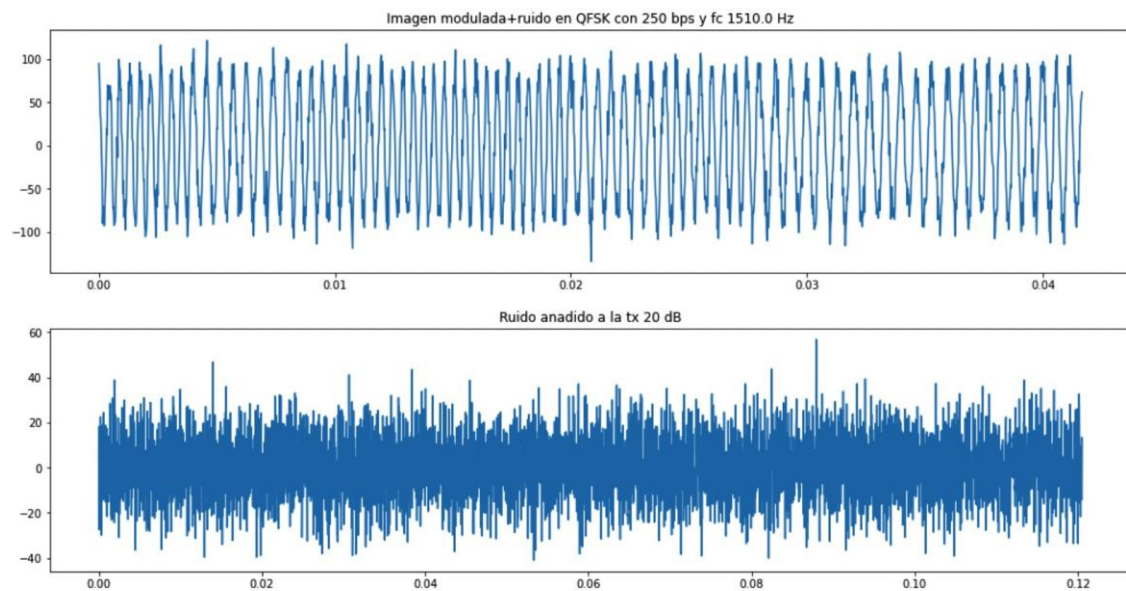
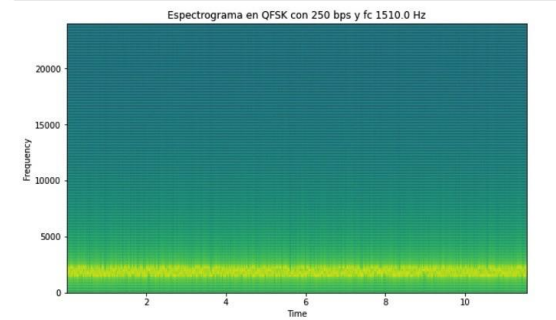
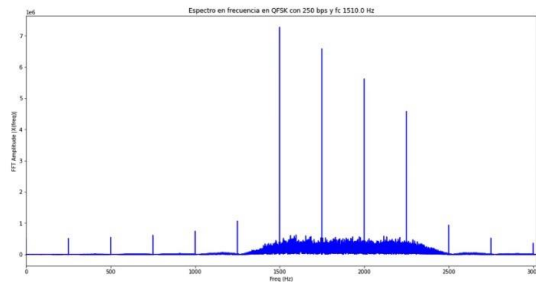


Figura 46. Señal modulada+ruído en QFSK

Ahora vemos como se comporta en el dominio de la frecuencia, tanto en su transformada discreta de Fourier como en un espectrograma :

QFSK en el dominio de la frecuencia



Analizamos como es el diagrama de ojo de la modulación en cuestión :

Figura 47. Modulación BPSK en el dominio de la frecuencia

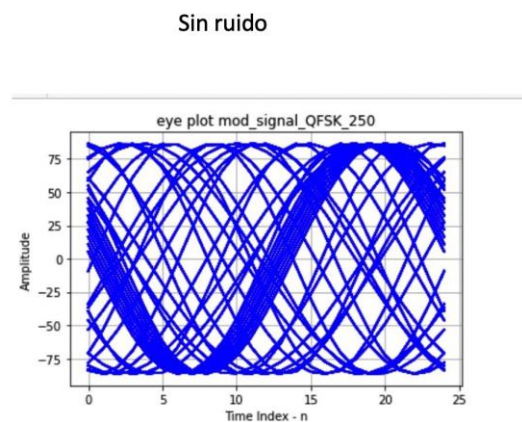
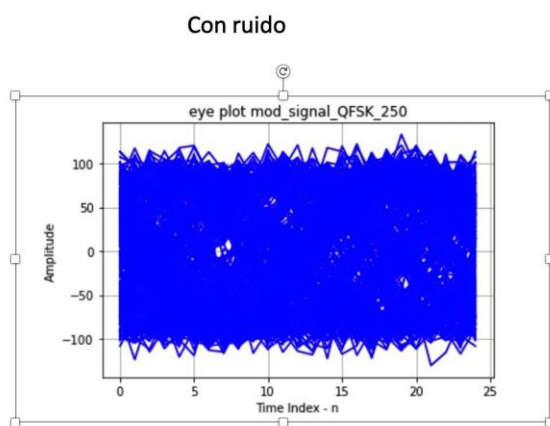


Figura 48. Diagramas de ojo de QFSK

Se concluye que la modulación funciona correctamente.

6.4 DISEÑO DEL RECEPTOR

6.4.1 DEMODULADOR

Como se ha explicado en el capítulo anterior vamos a utilizar diferentes demodulaciones. Por lo que se va a hacer una explicación del diagrama de bloques de cada una de ellas. En los siguientes módulos se han implementado todos los parámetros de demodulación mencionados previamente.

6.4.2 DEMODULACIÓN BPSK

En la siguiente imagen se puede ver el sistema demodulador de BPSK :

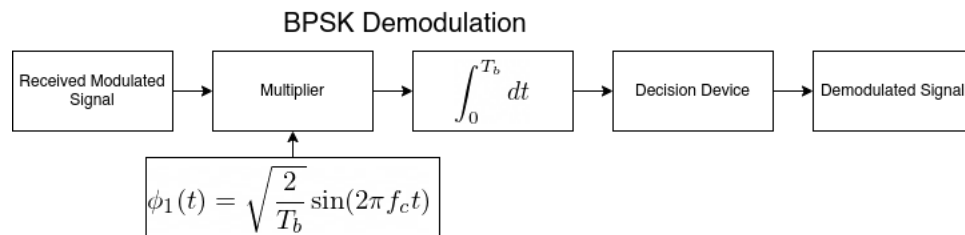


Figura 49. Diagrama de bloques demodulación BPSK[23]

Véase en la siguiente figura los resultados del demodulador :

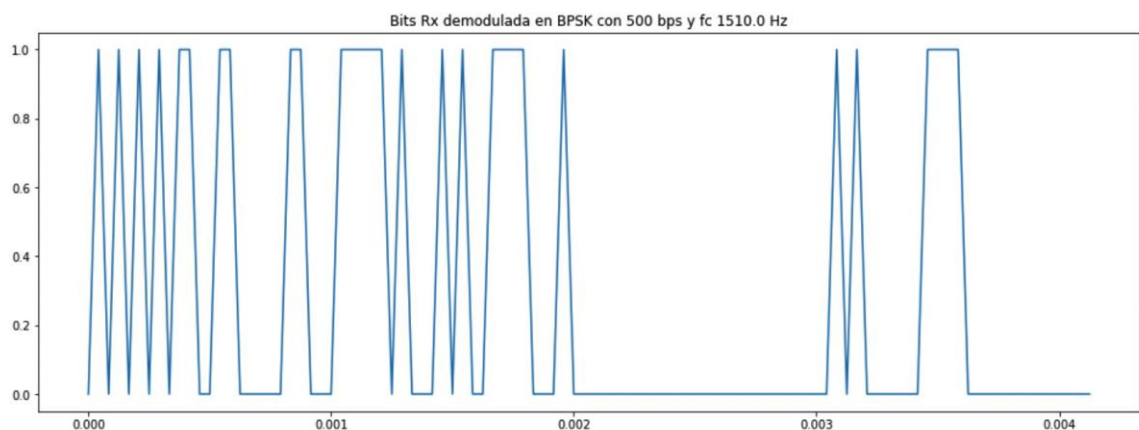


Figura 50. Demodulación de la señal en Bits, BPSK

6.4.3 DEMODULACIÓN QPSK

En la siguiente imagen se puede ver el sistema demodulador de QPSK :

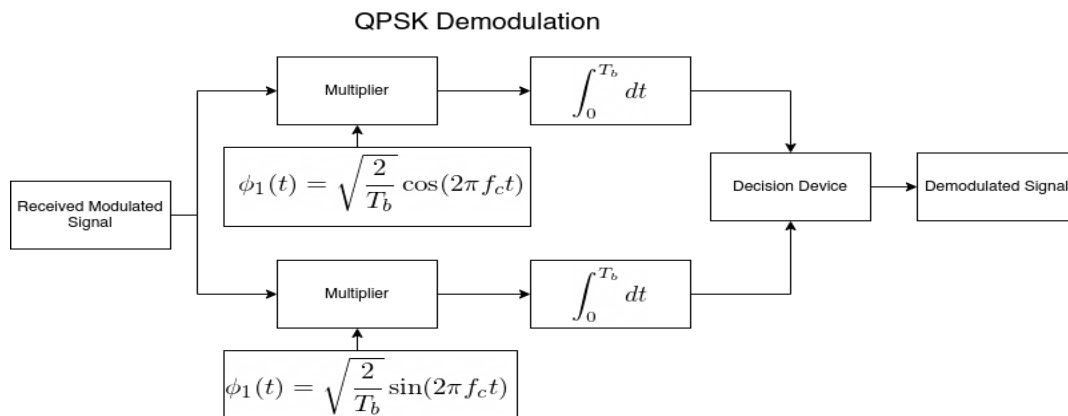


Figura 51. Diagrama de bloques demodulación QPSK[23]

Véase en la siguiente figura los resultados del demodulador :

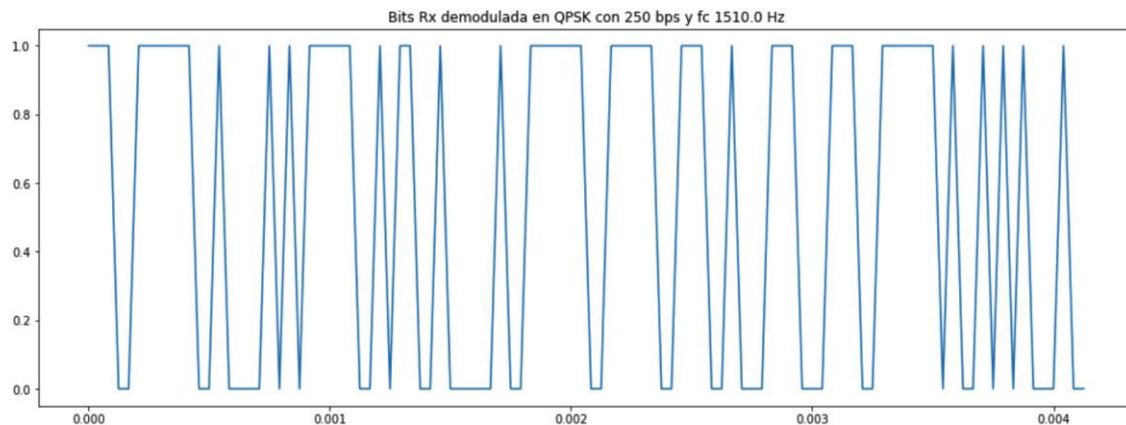


Figura 52. Demodulación de la señal en Bits, QPSK

6.4.4 DEMODULACIÓN BFSK

En la siguiente imagen se puede ver el sistema demodulador de BFSK :

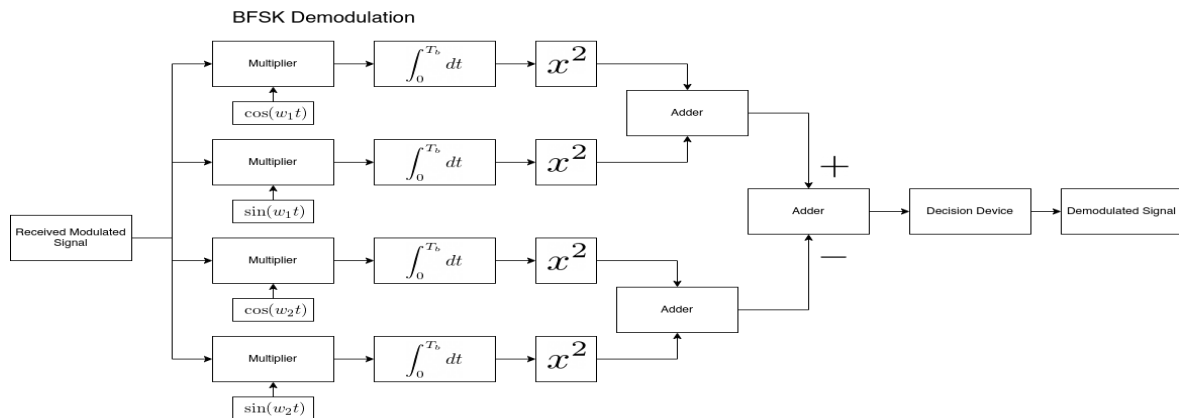


Figura 53. Diagrama de bloques demodulación BFSK[23]

Véase en la siguiente figura los resultados del demodulador :

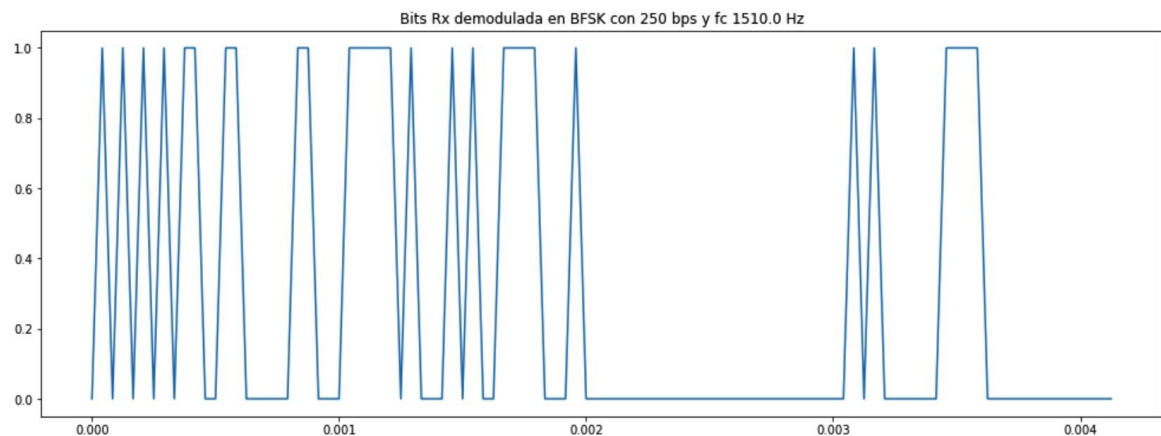


Figura 54. Demodulación de la señal en Bits, BFSK

6.4.5 DEMODULACIÓN QFSK

En la siguiente imagen se puede ver el sistema demodulador de QFSK :

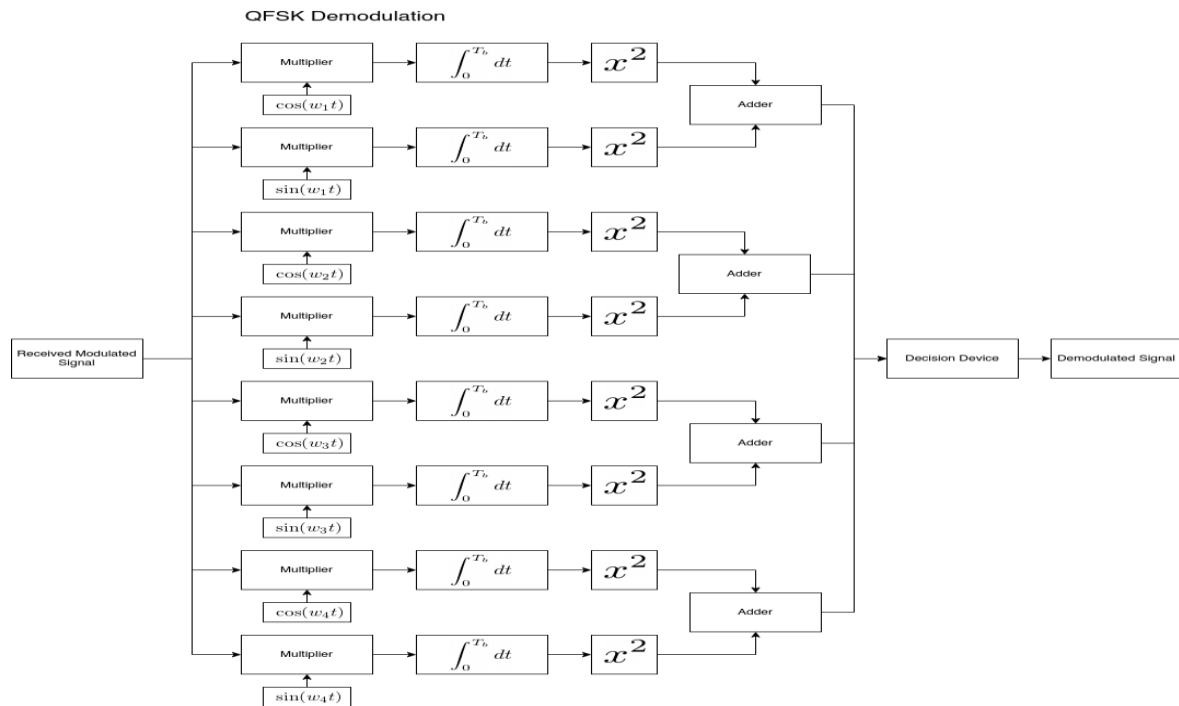


Figura 55. Diagrama de bloques demodulación QFSK[23]

Véase en la siguiente figura los resultados del demodulador :

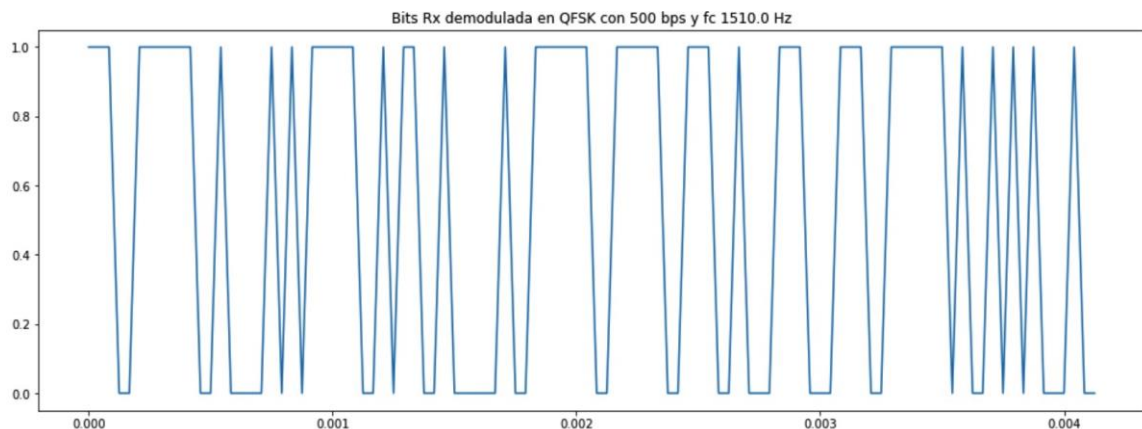


Figura 56. Demodulación de la señal en Bits, QFSK



Figura 58. Resultado final del sistema, imagen JPEG

7. ANÁLISIS DE RESULTADOS DE SIMULACIÓN

En este apartado se analizarán los resultados de transmisión de una imagen sobre las diferentes modulaciones y parámetros explicados en la tabla 3.

Modulation	Bit rate [bps]	Band Width [Hz]	Sample frequency [Hz]	Carrier frequency [Hz]
BPSK	50 , 250, 500,1000	50 ,100, 250, 500,1000	48000	1510
QPSK	50,250, 500, 1000	50 ,100, 250, 500,1000	48000	1510
BFSK	50, 250, 500, 1000	100,500,1000,2000	48000	1510
QFSK	50, 250, 500, 1000	100,500,1000,2000	48000	1510

La imagen para transmitir será la siguiente, la cual cuenta con 133 paquetes SSDV con FEC reed Solomun, debido a que las pruebas sin FEC no fueron óptimas.



Figura 59. Imagen elegida para la simulación de transmisión de imágenes sobre SSDV

Los resultados de BER se obtuvieron realizando simulaciones de 3 frames SSDV trasmitidos por un canal AWGN. La BER se calculó comparando el conjunto de muestras recibidas y enviadas.

7.1 BPSK

Se ha realizado un estudio de la probabilidad de error de la modulación para un rango de SNR diferentes, que se encuentra de 0 a 30dBs. También incluye las diferentes curvas de BER para cada bit rate indicado :

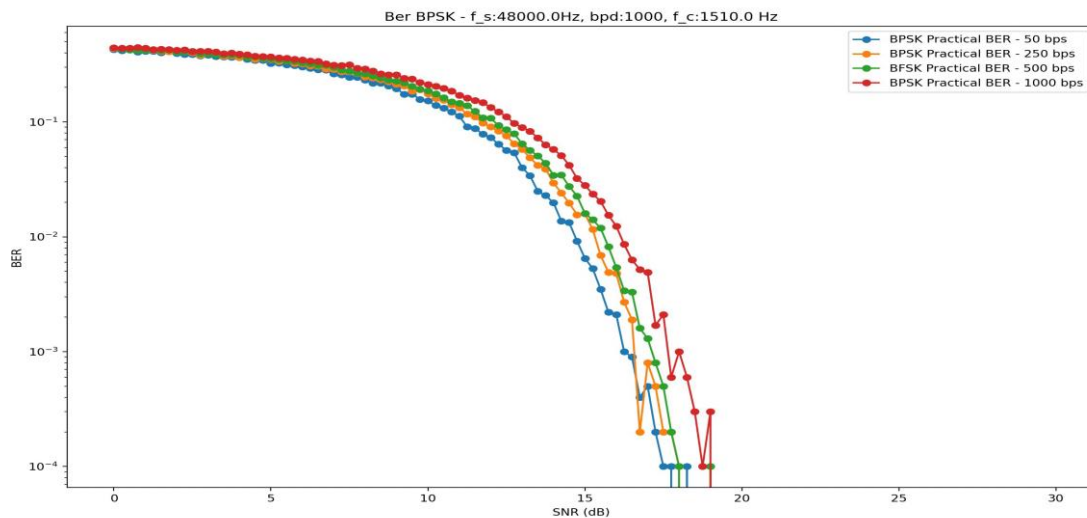


Figura 60. BER BPSK para sistema desarrollado para diferentes bits rates

En la figura previa se puede apreciar como cuando se aumenta la velocidad de transmisión de bit, la BER aumenta. Esto es debido a que se utiliza menor ancho de banda y la potencia de transmisión es mayor en ese rango. Pero si observamos mas en profundidad vemos que 50 bps posee una BER ligeramente mejor que las demás. Luego 250 bps y 500 bps se comportan de manera similar, por lo que se estudiara en la siguiente transmisión de imágenes, cual de los dos funciona mejor en términos de recepción de frames exitosos. Los resultados para 1000 bps no son malos, pero se espera un comportamiento en la transmisión de imágenes mucho peor que el del resto.

Se ha realizado una simulación de la transmisión para esta modulación, para las diferentes velocidades de transmisión y canales AWGN (5,10,15,20) utilizando los parámetros de transmisión explicados en el apartado 6. 2.

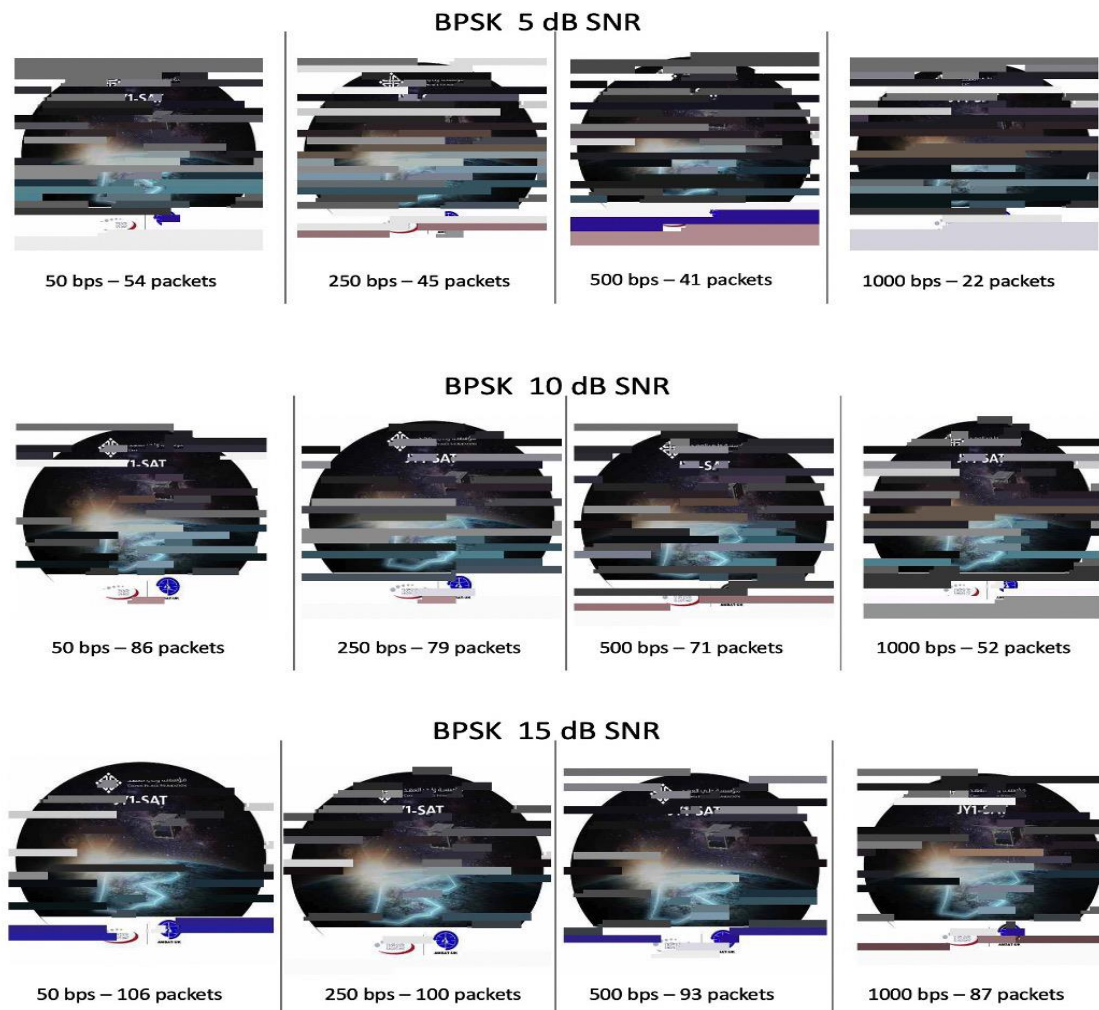


Figura 61. Resultado de imágenes transmitidas en BPSK

Como se ha explicado anteriormente en la grafica de la BER se esperan los mejores resultados para una velocidad de transmisión mas bajas, como se puede apreciar en la figura. Pero, 250 bps transmite casi con la misma calidad y sin perder paquetes tanto como 50 bps, pero para 500 bps se comportan bastante peor. Si tenemos una SNR alta de 15 dB podríamos escoger esta antes que 250, pero hablando en términos de robustez es mucho mejor 250 bps. Luego 1000 bps funciona mejor de lo esperado, pero se podría empezar a utilizar cuando tengamos una SNR un poco mas alta de 15 dB. Todos los sistemas han decodificado la imagen en su totalidad cuando se ha simulado la transmisión con una SNR de 20 dB.

Tabla de resultados de la transmisión de imágenes SSDV sobre BPSK:

Modulación	Bit rate[bps]	Fs[Hz]	F_c[Hz]	BW [Hz]	SNR	Packets	% Packets
BPSK	50	48000	1510	50	5	54	40,60
BPSK	50	48000	1510	50	10	86	64,66
BPSK	50	48000	1510	250	15	106	79,70
BPSK	50	48000	1510	500	20	133	100,00
BPSK	250	48000	1510	1000	5	45	33,83
BPSK	250	48000	1510	50	10	79	59,40
BPSK	250	48000	1510	250	15	100	75,19
BPSK	250	48000	1510	500	20	133	100,00
BPSK	500	48000	1510	1000	5	41	30,83
BPSK	500	48000	1510	50	10	71	53,38
BPSK	500	48000	1510	250	15	93	69,92
BPSK	500	48000	1510	500	20	133	100,00
BPSK	1000	48000	1510	1000	5	32	24,06
BPSK	1000	48000	1510	50	10	58	43,61
BPSK	1000	48000	1510	250	15	87	65,41
BPSK	1000	48000	1510	500	20	133	100,00

Tabla 4. Resultados de la simulacion para BPSK

7.2 QPSK

Se ha realizado un estudio de la probabilidad de error de la modulación para un rango de SNR diferentes, que se encuentra de 0 a 30dBs. También incluye las diferentes curvas de BER para cada bit rate definido en la tabla de parámetros:

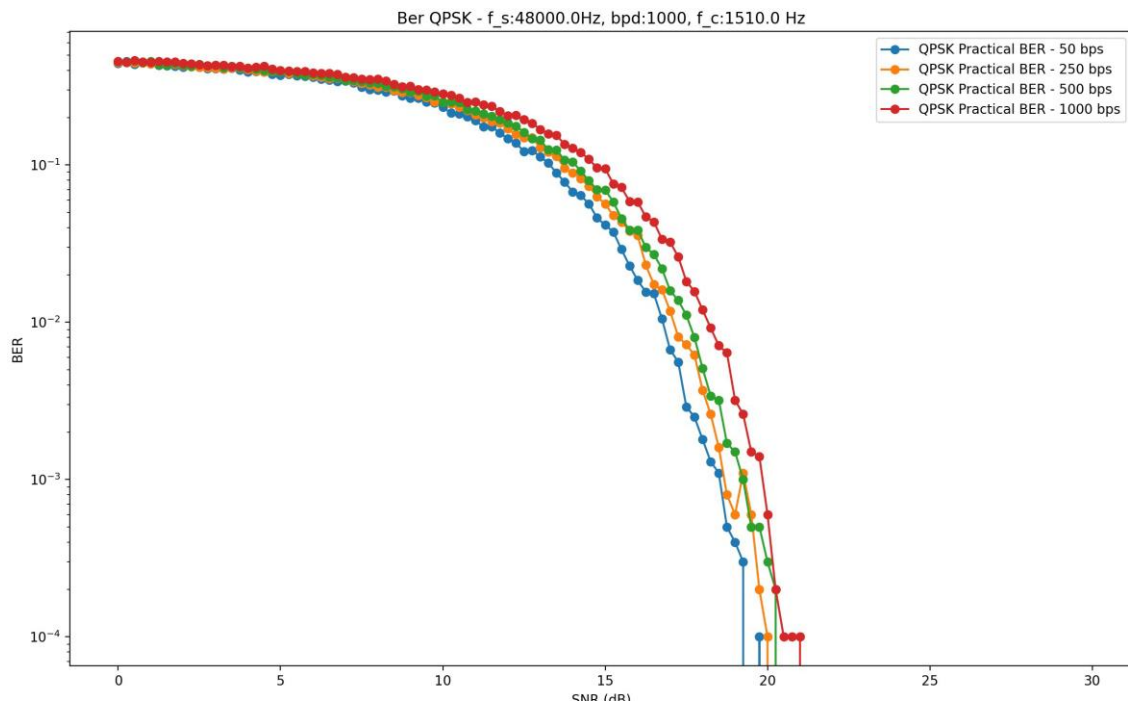


Figura 62. BER QPSK para sistema desarrollado para diferente bit rates

En la figura previa se puede apreciar como cuando se aumenta la velocidad de transmisión de bit, la BER aumenta. Esto es debido a que se utiliza menor ancho de banda y la potencia de transmisión es mayor en ese rango. Pero si observamos mas en profundidad vemos que 50 bps posee una BER ligeramente mejor que las demás. Luego 250 bps y 500 bps se comportan de manera similar. Los resultados para 1000 bps son los peores de la grafica, pero si se tiene un canal con una buena SNR se podría plantear su uso.

Mas tarde se realizo la simulación de la transmisión para esta modulación, para las diferentes velocidades de transmisión y canales AWGN (5,10,15,20) :

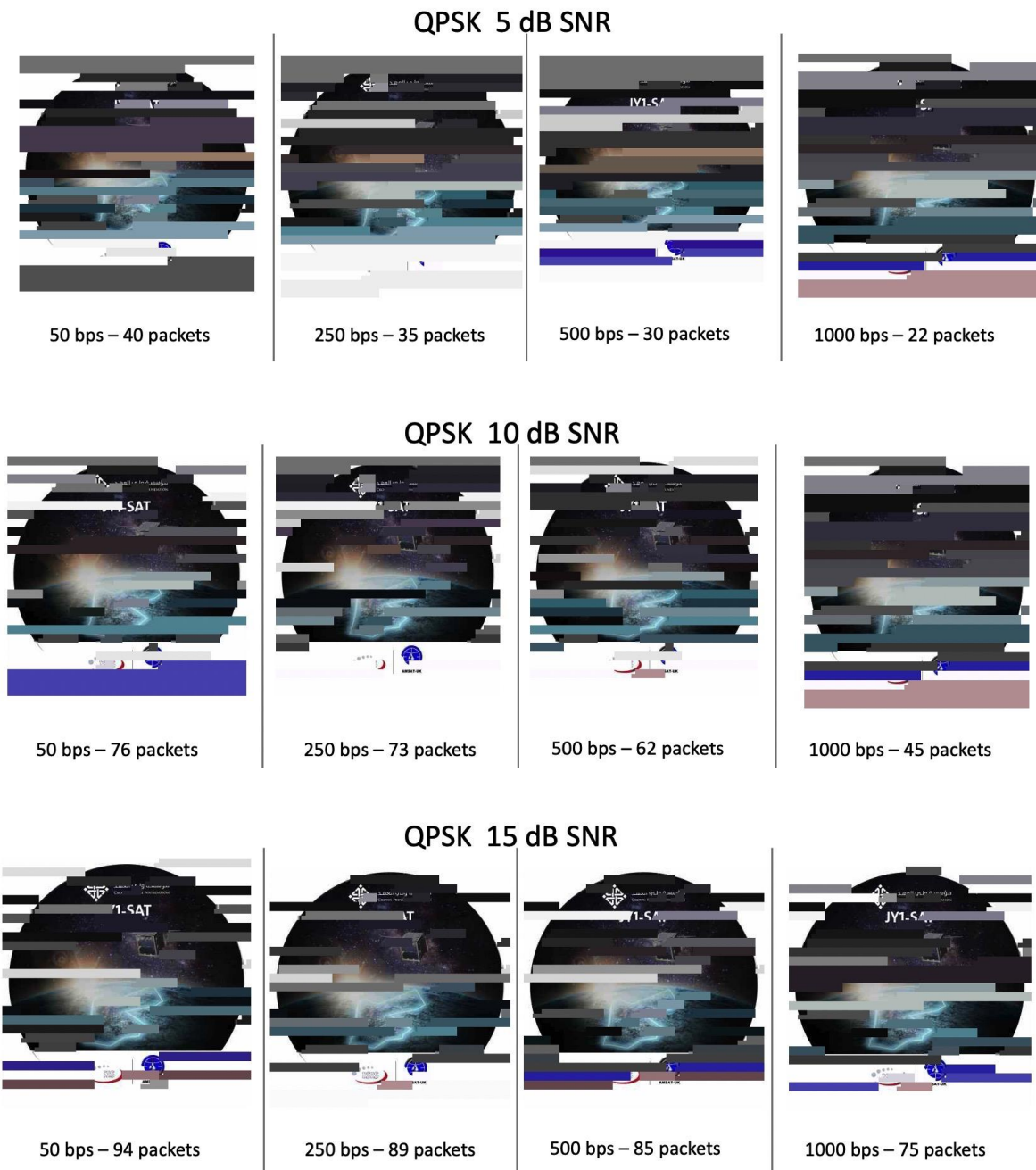


Figura 63. Resultado de imágenes transmitidas en QPSK

Como se ha explicado anteriormente en la grafica de la BER se esperan los mejores resultados para una velocidad de transmisión mas bajas, como se puede apreciar en la figura

previa. Quiero destacar que para esta modulación los resultados de 50 bps y 250 bps son bastante semejantes, pero se puede ver que siempre son mejores en 50. Luego las velocidades de 500 y 1000, no tan buenos obtienen resultados. Todos los sistemas han decodificado la imagen en su totalidad cuando se ha simulado la transmisión con una SNR de 20 dB.

Todos los diferentes sistemas han decodificado la imagen en su totalidad cuando se ha simulado la transmisión con una SNR de 20 dB.

Tabla de resultados de la transmisión de imágenes SSDV sobre QPSK:

Modulación	Bit rate	Fs	F_c	BW	SNR	Packets	%Packets
QPSK	50	48000	1510	50	5	40	30,08
QPSK	50	48000	1510	50	10	76	57,14
QPSK	50	48000	1510	250	15	94	70,68
QPSK	50	48000	1510	500	20	133	100,00
QPSK	250	48000	1510	1000	5	35	26,32
QPSK	250	48000	1510	50	10	73	54,89
QPSK	250	48000	1510	250	15	89	66,92
QPSK	250	48000	1510	500	20	133	100,00
QPSK	500	48000	1510	1000	5	30	22,56
QPSK	500	48000	1510	50	10	62	46,62
QPSK	500	48000	1510	250	15	85	63,91
QPSK	500	48000	1510	500	20	133	100,00
QPSK	1000	48000	1510	1000	5	22	16,54
QPSK	1000	48000	1510	50	10	45	33,83
QPSK	1000	48000	1510	250	15	75	56,39
QPSK	1000	48000	1510	500	20	133	100,00

Tabla 5. Resultados de la simulacion para BPSK

7.3 BFSK

Se ha realizado un estudio de la probabilidad de error de la modulación para un rango de SNR diferentes, que se encuentra de 0 a 30dBs. También incluye las diferentes curvas de BER para cada bit rate, 50,250,500,1000 :

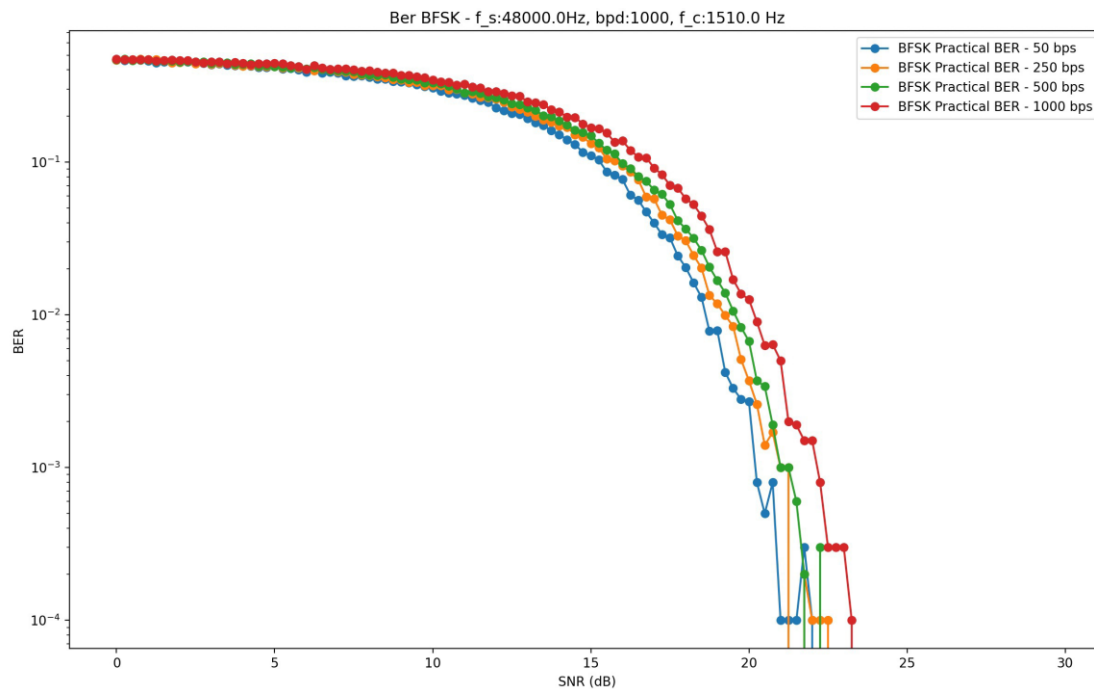
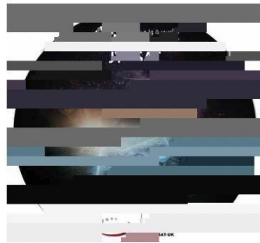


Figura 64. BER BPSK para sistema desarrollado para diferentes bit rates

Como se ha mencionado en los apartados anteriores, según va subiendo la velocidad de bit, los resultados de BER van empeorando. Compárese que su SNR es peor que BPSK debido a que al ser en FM, no aprovecha tan bien la potencia de Bit y el canal. Además, se sabe que las modulaciones FSK son muy sensibles al ruido.

Mas tarde se realizo la simulación de la transmisión para esta modulación, para las diferentes velocidades de transmisión y canales AWGN (5,10,15,20) :

BFSK 5 dB SNR



50 bps – 35 packets



250 bps – 29 packets

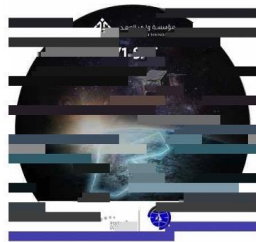


500 bps – 25 packets

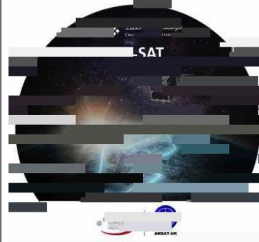


1000 bps – 18 packets

BFSK 10 dB SNR



50 bps – 77 packets



250 bps – 71 packets



500 bps – 63 packets



1000 bps – 49 packets

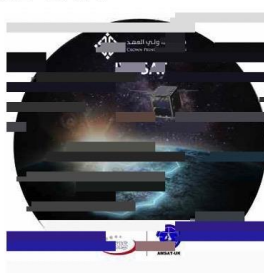
BFSK 15 dB SNR



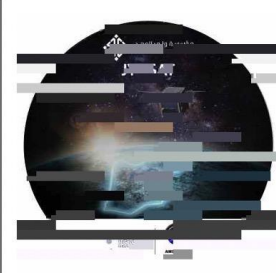
50 bps – 93 packets



250 bps – 90 packets



500 bps – 83 packets



1000 bps – 81 packets

Figura 65. Resultado de imágenes transmitidas en BFSK

Quiero destacar que para esta modulación los resultados de 50 bps y 250 bps son bastante semejantes, pero se puede ver que siempre son mejores en 50. Luego las velocidades de 500 y 1000, no tan buenos obtienen resultados. Todos los sistemas han decodificado la imagen

en su totalidad cuando se ha simulado la transmisión con una SNR de 20 dB. Los resultados de esta modulación son bastantes favorables según lo esperado.

Tabla de resultados de la transmisión de imágenes SSDV sobre BFSK:

Modulación	Bit rate[bps]	Fs[Hz]	F_c[Hz]	BW [Hz]	SNR	Packets	% Packets
BFSK	50	48000	1510	50	5	35	26,32
BFSK	50	48000	1510	50	10	77	57,89
BFSK	50	48000	1510	250	15	93	69,92
BFSK	50	48000	1510	500	20	133	100,00
BFSK	250	48000	1510	1000	5	29	21,80
BFSK	250	48000	1510	50	10	71	53,38
BFSK	250	48000	1510	250	15	90	67,67
BFSK	250	48000	1510	500	20	133	100,00
BFSK	500	48000	1510	1000	5	25	18,80
BFSK	500	48000	1510	50	10	63	47,37
BFSK	500	48000	1510	250	15	87	65,41
BFSK	500	48000	1510	500	20	133	100,00
BFSK	1000	48000	1510	1000	5	18	13,53
BFSK	1000	48000	1510	50	10	49	36,84
BFSK	1000	48000	1510	250	15	81	60,90
BFSK	1000	48000	1510	500	20	133	100,00

Tabla 6. Resultados de la simulacion para BFSK

7.4 QFSK

Se ha realizado un estudio de la probabilidad de error de la modulación para un rango de SNR diferentes, que se encuentra de 0 a 30dBs. También incluye las diferentes curvas de BER para cada bit rate, 50,250,500,1000 :

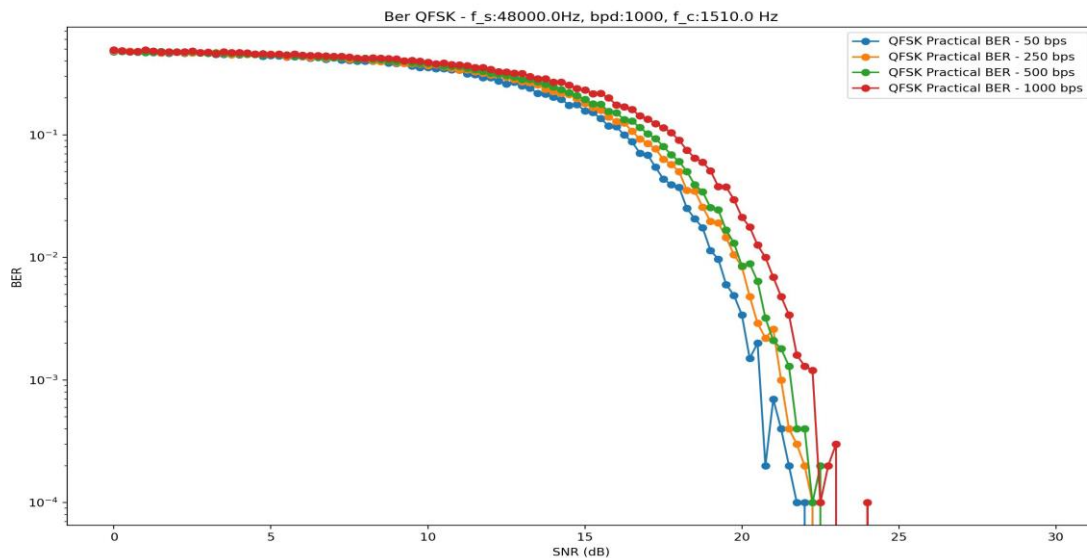


Figura 66. BER QFSK para sistema desarrollado para diferentes bits rates

Como se ha mencionado en los apartados anteriores, según va subiendo la velocidad de bit, los resultados de BER van empeorando. Compárese que su SNR es peor que QPSK debido a que al ser en FM, no aprovecha tan bien la potencia de Bit y el canal, siendo muy sensible al ruido. Hay que destacar que es la peor modulación en términos de resultados, esto es debido a que esta transmite al doble de velocidad, debido a que su Baud rate es el doble, pero esto afecta considerablemente a la BER. .

Mas tarde se realizo la simulación de la transmisión para esta modulación, para las diferentes velocidades de transmisión y canales AWGN (5,10,15,20) :

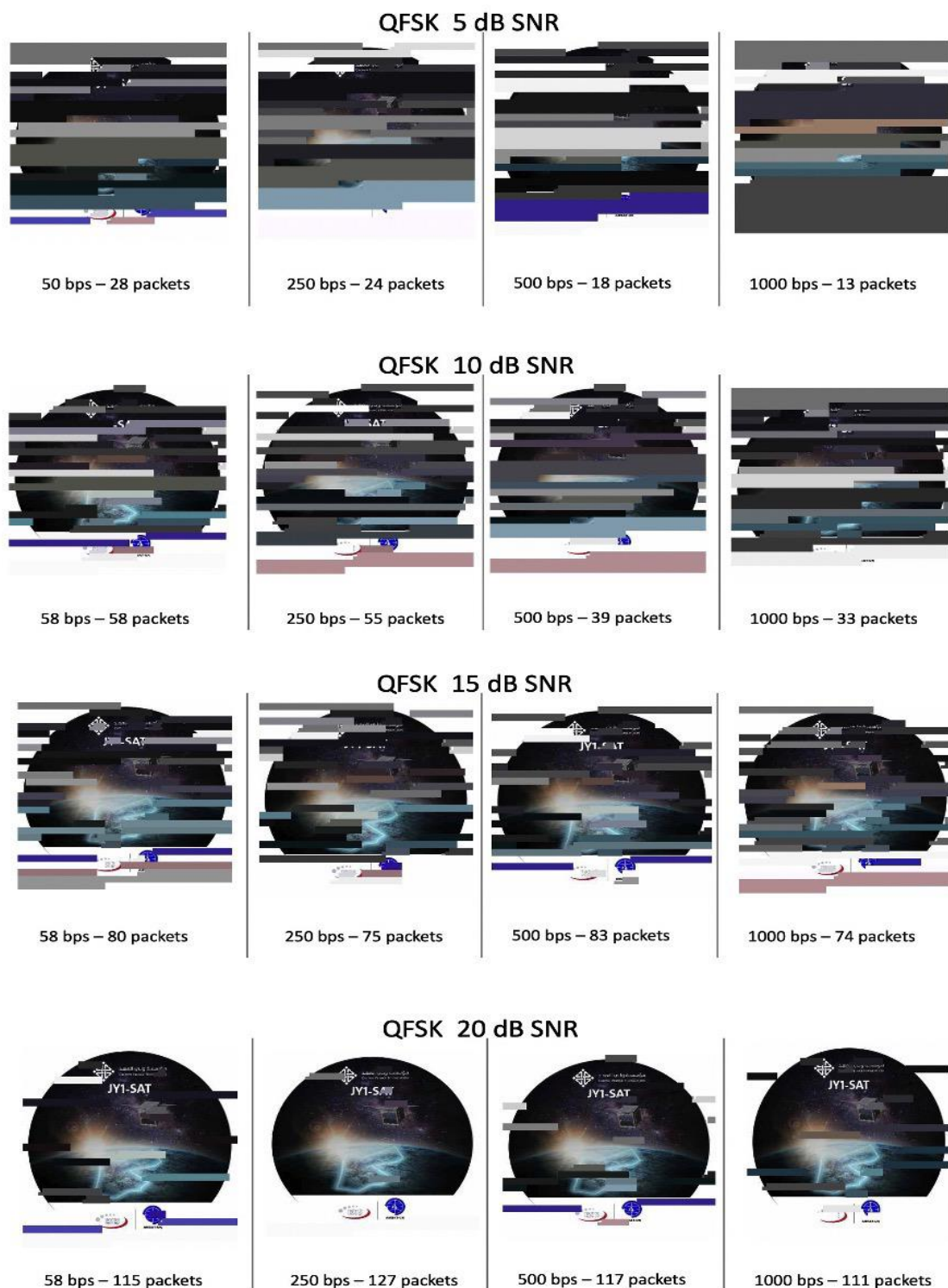


Figura 67. Resultado de imágenes transmitidas en QFSK

Hay que destacar que para esta modulación los resultados han sido los peores según las otras modulaciones. Luego las velocidades de para 50 y 250 con SNR 5, 10 y 15 los resultados son bastante óptimos, pero no tan buenos obtienen resultados como las otras modulaciones. Para SNR de 20 dB podemos ver que se comporta casi de la misma manera en todos los escenarios de velocidad de transmisión.

Tabla de resultados de la transmisión de imágenes SSDV sobre QFSK:

Modulación	Bit rate[bps]	Fs[Hz]	F_c[Hz]	BW [Hz]	SNR	Packets	% Packets
QFSK	50	48000	1510	50	5	28	21,05
QFSK	50	48000	1510	50	10	58	43,61
QFSK	50	48000	1510	250	15	80	60,15
QFSK	50	48000	1510	500	20	115	86,47
QFSK	250	48000	1510	1000	5	24	18,05
QFSK	250	48000	1510	50	10	55	41,35
QFSK	250	48000	1510	250	15	73	54,89
QFSK	250	48000	1510	500	20	127	95,49
QFSK	500	48000	1510	1000	5	20	15,04
QFSK	500	48000	1510	50	10	39	29,32
QFSK	500	48000	1510	250	15	83	62,41
QFSK	500	48000	1510	500	20	117	87,97
QFSK	1000	48000	1510	1000	5	13	9,77
QFSK	1000	48000	1510	50	10	33	24,81
QFSK	1000	48000	1510	250	15	74	55,64
QFSK	1000	48000	1510	500	20	111	83,46

Tabla 7. Resultados de la simulacion para QFSK

7.5 COMPARACIÓN

En este apartado se ha realizado un análisis comparativo entre todas las modulaciones implementadas. Véase en la siguiente figura la BER calculada para las diferentes modulaciones y sus respectivos bits rates.

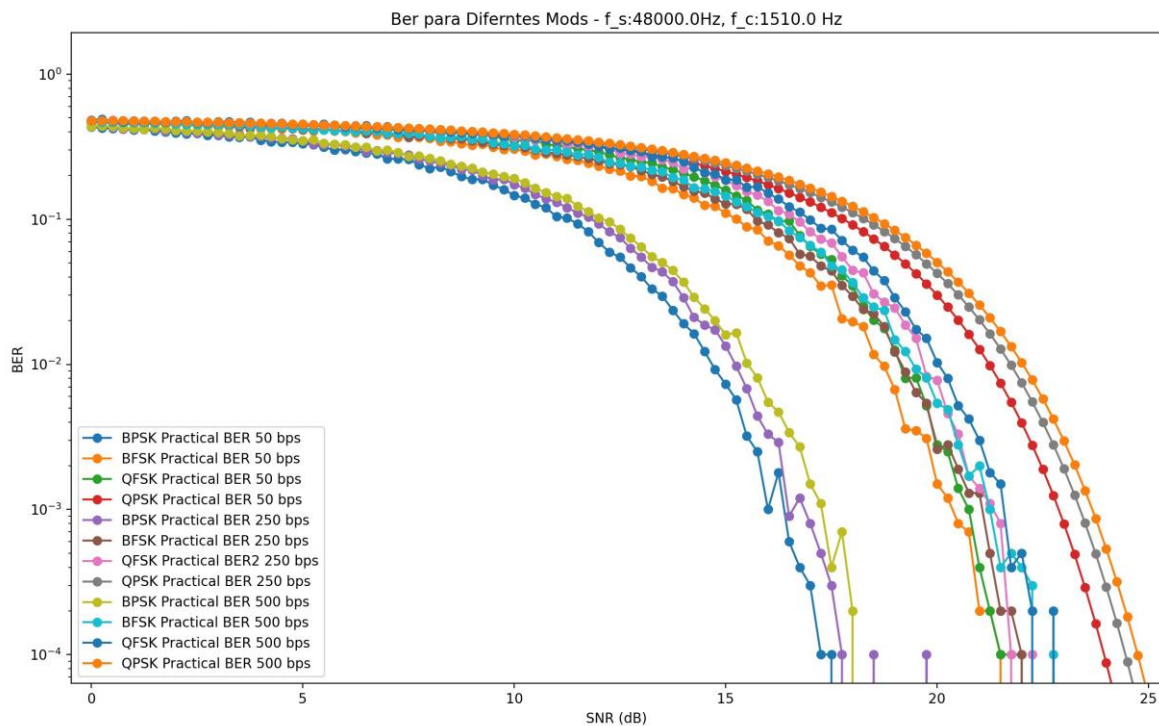


Figura 68. BER Comparativa de las modulaciones BPSK, QPSK, BFSK y QFSK

Como se puede comprobar y se sabia de antemano, la modulación BPSK es la que mejores resultados frente a errores consigue, esto se debe a que la potencia de todos los símbolos es la misma y siendo la mas óptima desde el punto de vista de protección frente a errores. Comparándola con QPSK esta se usa para duplicar la velocidad de datos, obteniendo la mitad de potencia por bit, por lo tanto, la mitad de SNR, por lo tanto, una mayor tasa de error de bit. Con respecto a FSK, se sabia que no es una modulación optima frente al ruido, por lo que se esperaba una peor BER comparada con BPSK. Pero si se hace un análisis con QPSK, se ve que incluso obtiene los mismos resultados. QFSK ha sido la peor de todas en términos de BER esto se debe a lo explicado anteriormente, si ya se esperaba que FSK no fuera tan

optima debido a su sensibilidad frente al ruido si duplicamos la velocidad de transmisión obtendremos la mitad de potencia por bit.

En términos de decodificación de la imagen JPEG transmitida, como se puede comprobar la mejor ha sido BPSK, para los diferentes SNR 5,10,15 y las mejores velocidades de transmisión han sido 50 y 250 bps. Las demás modulaciones han obtenido peores resultados según los paquetes decodificados, declarando que la mas optima ha sido BPSK. Si lo compramos con la figura 2, del proyecto sobre SSDV, podemos decir que SSDV obtiene mejores resultados tanto en velocidades de transmisión a implementar como a partir de los 10 dB de SNR.

Si lo compramos con la figura 1 , de la transmisión de imágenes sobre SSTV, podemos decir que SSDV obtiene mejores resultados tanto en velocidades de transmisión a implementar como a partir de los 10 dB de SNR.

8. CONCLUSIONES Y TRABAJOS FUTUROS

El objetivo de este proyecto era realizar un estudio de la tecnologías SSTV y SSDV en profundidad y desarrollar una forma de poder implementar SSDV para la transmisión de imágenes desde un satélite de radioaficionado. Esto es debido a que SSTV tiene como principales problemas, que no consigue una SNR (señal a ruido) suficiente para que la calidad de las fotos transmitidas sean buenas y se pueda obtener información relevante de las mismas. Por lo que el estudio y simulación de un sistema SSDV nos daría solución a este problema, ya que cuenta con una estandarización y una resolución de problemas avanzada, además de funcionar sobre radio digital y cubrir en profundidad factores esenciales como son el formato del paquete, formato de la imagen, corrección de errores, recuperación de paquetes perdidos la capacidad de poder transmitir a una velocidad de datos bastante más elevada que en SSTV.

Este problema nos llevó a hacer un análisis de los satélites de radioaficionados que habían utilizado SSDV previamente en sus transmisiones de imágenes, descubriendo que dos de ellos lo habían realizado con éxito, el JYSAT (modulación BPSK a 250 bps) y la misión DSWLP(modulación GMSK a 500 bps). Se pudo decodificar las imágenes de transmisiones de estos satélites con éxito debido a que sus módulos de recepción y decodificación estaban implementados en una librería de SDR llamada gr-satellites, que se implementa por encima de GNU Radio, sistema utilizado por la mayoría de radioaficionados.

Una vez se supo cómo diseñar el sistema que simularía esta transmisión en banda base, obtuvimos los resultados de este para las modulaciones BPSK, QPSK, BFSK y QFSK para diferentes bit rates. Concluyendo en el apartado anterior que BPSK a 50 y 250 bps era la modulación más óptima para este tipo de transmisión, pero las demás menos QPSK se podrían utilizar en dicho caso a los mismos bit rates, siempre que antes se realicen pruebas con las frecuencias de radio específicas, que en este proyecto no se pudieron probar y debido a la falta de HW y tiempo. Por lo que se recomienda utilizar BPSK a 250 bps, debido a que los resultados fueron muy favorables y además el sistema de transmisión de JY1SAT

implementa la misma, pudiendo replicar el sistema que se encuentra en gr-satellites de forma sencilla, debido a que este modulo es de licencia MIT.

También se recomienda a AMSAT EA el cambio a tecnologías digitales como se mencionaba en la motivación y mas tarde verificaba en el análisis de resultados del sistema desarrollado. Estos resultados nos dan claridad en que el modulo SSDV es robusto frente a la corrección de errores y, además, para una SNR a partir de 10 dB el performance de SSDV frente a SSTV es muy grande, como se ha podido comprobar.

Respecto a los trabajos futuros relacionados con el proyecto se pueden dividir en dos partes. La primera sería implementar el sistema de transmisión de JY1SAT sobre HW que tenga una antena de transmisión conectada, para posteriormente recibirla y con el receptor implementado sobre gr-satellites, y luego ver la forma de poder integrarlo con sus sistemas de transmisión. El segundo sería seguir desarrollando el sistema desarrollado en este proyecto y probarlo también en una raspberry pi (debido a que permite el uso de python) que pueda transmitirlo en HF, para ver si los resultados siguen siendo los mismos o cambian debido a esta implementación.

9. BIBLIOGRAFÍA

- [1] UKHAS wiki. guides:ssdv [UKHAS Wiki]. (n.d.). <https://ukhas.org.uk/guides:ssdv>.
- [2] SSTT concepts. N4spp - sstv. (n.d.).
https://www.nonstopsystems.com/radio/frank_radio_sstv.htm.
- [3] SSTT concepts. N4spp - sstv. (n.d.).
https://www.nonstopsystems.com/radio/frank_radio_sstv.htm.
- [4] Bruchanov, M. (2019, July). *sstv-handbook*. <https://www.sstv-handbook.com/download/sstv-handbook.pdf>.
- [5] Wikimedia Foundation. (2021, May 28). *Cyclic redundancy check*. Wikipedia.
[https://en.wikipedia.org/wiki/Cyclic_redundancy_check#:~:text=A%20cyclic%20redundancy%20check%20\(CRC,polynomial%20division%20of%20their%20contents](https://en.wikipedia.org/wiki/Cyclic_redundancy_check#:~:text=A%20cyclic%20redundancy%20check%20(CRC,polynomial%20division%20of%20their%20contents).
- [6] Wikimedia Foundation. (2021, July 23). *Reed–Solomon error correction*. Wikipedia.
https://en.wikipedia.org/wiki/Reed%E2%80%93Solomon_error_correction.
- [7] Wikimedia Foundation. (2021, February 14). *Codificación Huffman*. Wikipedia.
https://es.wikipedia.org/wiki/Codificaci%C3%B3n_Huffman.
- [8] Wikimedia Foundation. (2021, August 14). *Software-defined radio*. Wikipedia.
https://en.wikipedia.org/wiki/Software-defined_radio.
- [9] Gnuradio. (n.d.). Gnuradio/Gnuradio: GNU radio – the free and open software Radio Ecosystem. GitHub. <https://github.com/gnuradio/gnuradio>.
- [10] Daniestevez. (n.d.). *Daniestevez/Gr-Satellites: GNU radio decoders for several amateur satellites*. GitHub. <https://github.com/daniestevez/gr-satellites>.
- [11] Alberca, A. S. (2020, October 4). *La librería Numpy*. Aprende con Alf.
<https://aprendeconalf.es/docencia/python/manual/numpy/>.
- [12] Alberca, A. S. (2020, October 4). *La librería matplotlib*. Aprende con Alf.
<https://aprendeconalf.es/docencia/python/manual/matplotlib/>.
- [13] Wikimedia Foundation. (2021, July 20). *Slow-scan television*. Wikipedia.
https://en.wikipedia.org/wiki/Slow-scan_television.
- [14] AMSAT (the radio amateur Satellite Corporation). (n.d.).
<http://javiervalcarce.eu/amsat/index.html>.

- [15] AMSAT EA - *Satélites activos*. AMSAT EA - Radio Amateur Satellite Corporation Spain. (n.d.). <https://www.amsat-ea.org/sat%C3%A9lites-activos/>.
- [16] Alpov. (n.d.). *alpov/SatCam: Psat-2 PSK/SSTV transponder*. GitHub. <https://github.com/alpov/SatCam>.
- [17] *UBSEDS25 balloon has slow scan digital video*. AMSAT. (2017, June 28). <https://amsat-uk.org/2017/06/28/ubseds25-balloon-ssdv/>.
- [18] *JY1SAT launch information and dashboard*. AMSAT. (2020, March 13). <https://amsat-uk.org/2018/11/16/jy1sat-cubesat-information/>.
- [19] *Longjiang-2 / lo-94: Journey to the moon*. AMSAT. (2020, June 6). <https://amsat-uk.org/2020/06/06/longjiang-2-lo-94-journey-to-the-moon/>.
- [20] Camras. (n.d.). *CAMRAS DSLWP data*. DSLWP-B data. <https://charon.camras.nl/public/dslwp-b/>.
- [21] Wikimedia Foundation. (2020, December 10). *Dwingeloo radio Observatory*. Wikipedia. https://en.wikipedia.org/wiki/Dwingeloo_Radio_Observatory.
- [22] Destevez. (2018, August 10). *First ssdv transmission from DSLWP-B*. Daniel Estvez. <https://destvez.net/2018/08/first-ssdv-transmission-from-dslwp-b/>.
- [23] PrayagS. (n.d.). *PrayagS/Dayumcommunication: Making digital communication LOOK DAYUMM!* GitHub. <https://github.com/PrayagS/Dayumcommunication>.
- [24] Wikimedia Foundation. (2021, July 3). *Modulación por desplazamiento DE fase*. Wikipedia. https://es.wikipedia.org/wiki/Modulaci%C3%B3n_por_desplazamiento_de_fase.
- [25] Wikimedia Foundation. (2020, September 6). *Modulación por desplazamiento DE FRECUENCIA*. Wikipedia. https://es.wikipedia.org/wiki/Modulaci%C3%B3n_por_desplazamiento_de_frecuencia.

ANEXO A

Modulo SSDV

En este anexo se explica como instalar el modulo SSDV. Los requisitos son tener instalado gcc, el compilador de lenguaje c. Una vez comprobamos eso, nos bajamos el repositorio de código de github:

```
git clone https://github.com/fsphil/ssdv
```

Una vez hemos clonado en el directorio deseado, realizamos los siguientes pasos para su instalación :

```
cd ssdv  
make  
sudo make install
```

Con estos comandos habremos instalado el modulo. Ahora queda saber como utilizarlo, estos son los paramretos que acepta :

- -e Codificar paquetes JPEG a SSDV
- -d Decodificar paquetes SSDV a JPEG
- -n Codificar paquetes sin FEC
- -t Para las pruebas, deja caer el porcentaje especificado de paquetes durante la decodificación
- -c Establece el indicativo. Acepta A-Z 0-9 y espacio, hasta 6 caracteres
- -i Establece el ID de la imagen (0-255)
- -q Establecer el nivel de calidad JPEG (0 a 7, por defecto 4)
- -v Imprimir los datos de cada paquete decodificado.

Para codificar una imagen jpeg a un paquete SSDV utiliza el siguiente comando :

```
ssdv -e -c {callsign} -i {id_} -q {quality} {infile} {outfile}
```

Para decodificar paquetes SSDV en un fichero binario a una imagen jpeg el siguiente comando :

```
ssdv -d {input_binaro} {output_jpeg}
```

Modulo SSDV adaptado para transmisiones por satélite

Es el mismo modulo que explicado anteriormente, pero con unos ligeros cambios para los satélites DSWLP y JYSAT1.

Clonar el repositorio :

```
git clone https://github.com/daniestevez/ssdv.git
```

Para utilizar DSWLP:

```
ssdv -d -D -v /tmp/dslwp_ssdv.bin /tmp/dslwp_ssdv.jpg
```

Para JY1SAT :

```
ssdv -d -J /tmp/jylsat_ssdv.bin /tmp/jylsat_ssdv.jpg
```

GNU RADIO

En este caso se utilizo la versión 3.8 de GNU radio sobre un sistema Linux Ubuntu 20.

clonamos el repositorio :

```
Git clone -branch maint-3.8 https://github.com/gnuradio/gnuradio.git
```

Una vez clonado, realizamos las siguientes acciones en la terminal en el directorio donde se clono:

```
$ mkdir build
```

```
$ cd build
$ cmake ..
$ make
$ sudo make install
$ sudo ldconfig
```

Gr-satellites

Es preciso tener instalado gnu radio, previamente.

En este caso se utilizó la versión 3.8_v2 de gr-satellites sobre un sistema Linux Ubuntu 20.

Clonamos el repositorio :

```
git clone -branch maint-3.8_v2 https://github.com/daniestevez/gr-satellites.git
```

Una vez clonado, realizamos las siguientes acciones en la terminal en el directorio donde se clono:

```
$ mkdir build
$ cd build
$ cmake ..
$ make
$ sudo make install
$ sudo ldconfig
```

Sistema de transmisión/recepción

BPSK

```
#!/usr/bin/env python3
import matplotlib.pyplot as plt
import numpy as np
from scipy.stats import norm

def modulate(msg, Eb, Tb, f_c, f_s):
    modulated_signal = []
    t = np.linspace(0, Tb, int(Tb * f_s))
    for i in msg:
        s = np.sqrt(2 * Eb / Tb) * np.sin(2 * np.pi * f_c * t)
        if i == 0:
            s = -s
        modulated_signal.extend(s)
    t = np.linspace(0, len(msg) * Tb, int(len(msg) * Tb * f_s))
    modulated_signal = np.array(modulated_signal)
    return modulated_signal

def demodulate(signal, Tb, f_c, f_s):
    t = np.linspace(0, Tb, int(Tb * f_s))
    phi = np.sqrt(2 / Tb) * np.sin(2 * np.pi * f_c * t)
    N = len(signal) // len(t)
    signal = np.array_split(signal, N)
    received_msg = []
    for i in signal:
        x = i * phi
        sm = x.sum() / f_s
        if sm > 0:
            received_msg.append(1)
        else:
            received_msg.append(0)
    return received_msg

def error_probabilities(msg, decoded_msg, Eb, N0):
    Pb = norm.sf(np.sqrt(2 * Eb / N0))
    # print('Theoretical Bit Error Probability:', Pb)
    Pb_pr = np.count_nonzero(np.array(msg) != np.array(decoded_msg)) / len(msg)
    # print('Practical Bit Error Probability:', Pb_pr)
    return Pb, Pb_pr
```

BFSK

```
#!/usr/bin/env python3
from math import ceil
import numpy as np

def modulate(msg, Eb, Tb, f_c1, f_s):
    f_c2 = ceil(f_c1 + 1 / Tb)
    signal = []
    t = np.linspace(0, Tb, int(Tb * f_s))
    for i in msg:
        s = np.sqrt(2 * Eb / Tb) * np.cos(2 * np.pi * f_c1 * t)
        s1 = np.sqrt(2 * Eb / Tb) * np.cos(2 * np.pi * f_c2 * t)
        if i == 0:
            s = s1
        signal.extend(s)
    t = np.linspace(0, len(msg) * Tb, int(len(msg) * Tb * f_s))
    return np.array(signal)

def demodulate(signal, Tb, f_c1, f_s):
    f_c2 = ceil(f_c1 + 1 / Tb)
    t = np.linspace(0, Tb, int(Tb * f_s))
    Ts = int(Tb * f_s) # no of samples of carrier for 1 bit
    e1 = np.cos(2 * np.pi * f_c1 * t) # cosomega1t
    e2 = np.sin(2 * np.pi * f_c1 * t) # sinomega1t
    e3 = np.cos(2 * np.pi * f_c2 * t) # cosomega2t
    e4 = np.sin(2 * np.pi * f_c2 * t) # sinomega2t
    received_msg = []
    for x in range(int(len(signal) / Ts)):
        samplearr = signal[x * Ts: (x + 1) * Ts]
        e5 = (samplearr * e1).sum() / len(samplearr)
        e6 = (samplearr * e2).sum() / len(samplearr)
        e7 = (samplearr * e3).sum() / len(samplearr)
        e8 = (samplearr * e4).sum() / len(samplearr)

        e9 = e5 + e6
        e10 = e7 + e8
        if e9 < e10:
            received_msg.append(0)
        else:
            received_msg.append(1)
    return received_msg

def error_probabilities(msg, decoded_msg, Eb, N0):
    Pb = (1 / 2) * np.exp(-Eb / N0)
    Pb_pr = np.count_nonzero(np.array(msg) != np.array(decoded_msg)) / len(msg)
```

```
return Pb, Pb_pr
```

QPSK

```
#!/usr/bin/env python3
import matplotlib.pyplot as plt

import numpy as np

from scipy.signal import periodogram

from scipy.stats import norm

from MPSK import error_probabilities

def modulate(msg, Eb, Tb, f_c, f_s):

    t = np.linspace(0.0, Tb, int(Tb * f_s))

    # Serial to parallel with k=2 (QPSK)

    symbols = np.array([msg[0::2], msg[1::2]])

    theta = np.empty(np.size(symbols, axis=1), dtype="float")

    for k in range(np.size(symbols, axis=1)):

        b_0 = symbols[0, k]

        b_1 = symbols[1, k]

        # Page 8, Lecture 16

        if b_0 == 0 and b_1 == 0:

            theta[k] = 7.0 * np.math.pi / 4.0

        # theta[k] = 3.0 * np.math.pi / 2.0
```

```
elif b_0 == 0 and b_1 == 1:

    theta[k] = 5.0 * np.math.pi / 4.0

    # theta[k] = np.math.pi

elif b_0 == 1 and b_1 == 1:

    theta[k] = 3.0 * np.math.pi / 4.0

    # theta[k] = np.math.pi / 2.0

elif b_0 == 1 and b_1 == 0:

    theta[k] = np.math.pi / 4.0


# A = 1.0

A = np.sqrt(Eb)

I = A * np.cos(theta) # in-phase component

Q = A * np.sin(theta) # quadrature component


modulated_signal = np.empty(np.size(symbols, axis=1) * len(t), dtype="float")

phi_1 = np.sqrt(2 / Tb) * np.cos(2.0 * np.math.pi * f_c * t)

phi_2 = np.sqrt(2 / Tb) * np.sin(2.0 * np.math.pi * f_c * t)

for k in range(np.size(symbols, axis=1)):

    #

    modulated_signal[k * len(t) : (k + 1) * len(t)] = I[k] * phi_1 - Q[k] * phi_2


return modulated_signal
```

```
def demodulate(signal, Tb, f_c, f_s):  
  
    t = np.linspace(0, Tb, int(Tb * f_s))  
  
    phi_1 = np.sqrt(2 / Tb) * np.cos(2.0 * np.math.pi * f_c * t)  
  
    phi_2 = np.sqrt(2 / Tb) * np.sin(2.0 * np.math.pi * f_c * t)  
  
    N = len(signal) // len(t)  
  
    split_modulated_signal = np.array_split(signal, N)  
  
    received_symbols = [[] for i in range(2)]  
  
    for i in split_modulated_signal:  
  
        s_1 = i * phi_1  
  
        s_2 = i * phi_2  
  
        x = s_1.sum() / f_s  
  
        y = s_2.sum() / f_s  
  
        if x > 0 and y > 0:  
  
            received_symbols[0].append(0)  
  
            received_symbols[1].append(0)  
  
        elif x < 0 and y > 0:  
  
            received_symbols[0].append(0)  
  
            received_symbols[1].append(1)  
  
        elif x < 0 and y < 0:  
  
            received_symbols[0].append(1)  
  
            received_symbols[1].append(1)  
  
        elif x > 0 and y < 0:  
  
            received_symbols[0].append(1)  
  
            received_symbols[1].append(0)
```

```
received_msg = [  
  
    received_symbols[i][j]  
  
    for i, _ in enumerate(received_symbols[0])  
  
    for j, _ in enumerate(received_symbols)  
  
]  
  
return received_msg
```

QFSK

```
#!/usr/bin/env python3  
  
from math import ceil  
  
import numpy as np  
  
def modulate(msg, Eb, Tb, f_c1, f_s):  
    f_c2 = ceil(f_c1 + 1 / Tb)  
  
    f_c3 = ceil(f_c1 + 2 / Tb)  
  
    f_c4 = ceil(f_c1 + 3 / Tb)  
  
    symbols = np.array([msg[0::2], msg[1::2]])  
  
    signal = []  
  
    t = np.linspace(0.0, Tb, int(Tb * f_s))  
  
    for k in range(np.size(symbols, axis=1)):  
  
        b_0 = symbols[0, k]  
  
        b_1 = symbols[1, k]
```

```
if b_0 == 0 and b_1 == 0:

    s = np.sqrt(2 * Eb / Tb) * np.cos(2 * np.pi * f_c1 * t)

elif b_0 == 1 and b_1 == 0:

    s = np.sqrt(2 * Eb / Tb) * np.cos(2 * np.pi * f_c2 * t)

elif b_0 == 0 and b_1 == 1:

    s = np.sqrt(2 * Eb / Tb) * np.cos(2 * np.pi * f_c3 * t)

elif b_0 == 1 and b_1 == 1:

    s = np.sqrt(2 * Eb / Tb) * np.cos(2 * np.pi * f_c4 * t)

signal.extend(s)

return np.array(signal)

def demodulate(signal, Tb, f_c1, f_s):
    t = np.linspace(0.0, Tb, int(Tb * f_s))

    f_c2 = ceil(f_c1 + 1 / Tb)

    f_c3 = ceil(f_c1 + 2 / Tb)

    f_c4 = ceil(f_c1 + 3 / Tb)

    Ts = int(Tb * f_s) # no of samples of carrier for 1 bit

    e1 = np.cos(2 * np.pi * f_c1 * t) # cosomega1t

    e2 = np.sin(2 * np.pi * f_c1 * t) # sinomega1t

    e3 = np.cos(2 * np.pi * f_c2 * t) # cosomega2t

    e4 = np.sin(2 * np.pi * f_c2 * t) # sinomega2t

    e5 = np.cos(2 * np.pi * f_c3 * t) # cosomega1t

    e6 = np.sin(2 * np.pi * f_c3 * t) # sinomega1t

    e7 = np.cos(2 * np.pi * f_c4 * t) # cosomega2t
```

```
e8 = np.sin(2 * np.pi * f_c4 * t) # sinomega2t

decmsg = []

for x in range(int(len(signal) / Ts)):

    samplearr = signal[x * Ts:(x + 1) * Ts]

    e9 = (samplearr * e1).sum() / len(samplearr)

    e10 = (samplearr * e2).sum() / len(samplearr)

    e11 = (samplearr * e3).sum() / len(samplearr)

    e12 = (samplearr * e4).sum() / len(samplearr)

    e13 = (samplearr * e5).sum() / len(samplearr)

    e14 = (samplearr * e6).sum() / len(samplearr)

    e15 = (samplearr * e7).sum() / len(samplearr)

    e16 = (samplearr * e8).sum() / len(samplearr)

    e17 = e9 + e10

    e18 = e11 + e12

    e19 = e13 + e14

    e20 = e15 + e16

    if e17 > e18 and e17 > e19 and e17 > e20:

        decmsg.append(0)

        decmsg.append(0)

    elif e18 > e17 and e18 > e19 and e18 > e20:

        decmsg.append(1)

        decmsg.append(0)
```

```
elif e19 > e18 and e19 > e17 and e19 > e20:

    decmsg.append(0)

    decmsg.append(1)

elif e20 > e17 and e20 > e18 and e20 > e19:

    decmsg.append(1)

    decmsg.append(1)

return decmsg

def error_probabilities(msg, decoded_msg, Eb, N0):
    Pe = (3 / 2) * np.exp(-Eb / (N0))

    Pb = 2 * Pe / 3

    Pb_pr = np.count_nonzero(np.array(msg) != np.array(decoded_msg)) / len(msg)

    return Pe, Pb, Pb_pr
```

Channel

```
import numpy as np

def generate_noise(signal, N0, f_s):
    N0_unit_power = 0.0004

    ns = len(signal)

    noise = np.random.normal(size=ns)

    noise *= np.sqrt(N0 / N0_unit_power)

    return noise
```

Funciones SSDV

```
import struct
import os
import numpy as np

_ssdv_callsign_alphabet = '-0123456789---ABCDEFGHIJKLMNOPQRSTUVWXYZ'

def ssdv_decode_callsign(code):
    """ Decode a SSDV callsign from a supplied array of ints,
    extract from a SSDV packet.
    Args:
        list: List of integers, corresponding to bytes 2-6 of a SSDV packet.
    Returns:
        str: Decoded callsign.
    """

    code = bytes(bytearray(code))
    code = struct.unpack('>I',code)[0]
    callsign = ""

    while code:
        callsign += _ssdv_callsign_alphabet[code % 40]
        code = code // 40

    return callsign

def ssdv_packet_info(packet):
    """ Extract various information out of a SSDV packet, and present as a dict. """
    packet = list(bytearray(packet))
    # Check packet is actually a SSDV packet.
    if len(packet) != 256:
        return {'error': "ERROR: Invalid Packet Length"}

    if packet[0] != 0x55: # A first byte of 0x55 indicates a SSDV packet.
        return {'error': "ERROR: Not a SSDV Packet."}

    # We got this far, may as well try and extract the packet info.
    try:
        packet_info = {
            'callsign': ssdv_decode_callsign(packet[2:6]), # TODO: Callsign decoding.
            'packet_type': "FEC" if (packet[1]==0x66) else "No-FEC",
            'image_id': packet[6],
            'packet_id': (packet[7]<<8) + packet[8],
        }
```

```
'width' : packet[9]*16,
'height' : packet[10]*16,
'error' : "None"
}

return packet_info
except Exception as e:
    return {'error': "ERROR: %s" % str(e)}

def ssdv_packet_string(packet):
    """ Produce a textual representation of a SSDV packet. """
    packet_info = ssdv_packet_info(packet)
    if packet_info['error'] != 'None':
        return "SSDV: Unable to decode."
    else:
        return "SSDV: %s, Callsign: %s, Img:%d, Pkt:%d, %dx%d" %
(packet_info['packet_type'], packet_info['callsign'], packet_info['image_id'], packet_info['packet_id'], packet_info['width'], packet_info['height'])

def open_ssdv_file():
    print('Open SSDV file')
    f = open('./output.bin', 'rb')
    packets = os.path.getsize('./output.bin') >> 8
    return f, packets

def open_ssdv_input_file(path):
    print('Open SSDV file')
    f = open(path, 'rb')
    packets = os.path.getsize(path) >> 8
    return f, packets

def encode(data):
    # TODO: Add in RS232 framing
    tx = np.frombuffer(data, dtype=np.uint8)
    raw_data = np.unpackbits(tx)
    return raw_data

def decode(data):
    rx = np.packbits(data)
    return rx.tobytes('F')
```

```
def compress_image(infile="/input/txtemp.jpg", id_=0, outfile="/input/txtemp.bin", callsign="N0CALL",
quality=4):
    """ Attempt to compress a JPEG using SSDV. """

    _command = f"./ssdv_2 -e -c {callsign} -i {id_} -q {quality} {infile} {outfile}"

    retcode = os.system(_command)

    if retcode == 0:
        return True
    else:
        return False

def compress_image_no_fec(infile="/input/txtemp.jpg", id_=0, outfile="/input/txtemp.bin", callsign="N0CALL",
quality=4):
    """ Attempt to compress a JPEG using SSDV. """

    _command = f"./ssdv_2 -e -n -c {callsign} -i {id_} -q {quality} {infile} {outfile}"

    retcode = os.system(_command)

    if retcode == 0:
        return True
    else:
        return False
```

Transmisor/receptor SSDV

```
#!/usr/bin/env python
# coding: utf-8

## Trasmision de Imagenes por SSDV
```

```
# In[1]:

import numpy as np
from reedsolo import RSCodec, ReedSolomonError
from numpy import sqrt
from numpy.random import rand, randn
from scipy.signal import periodogram
import matplotlib.pyplot as plt
import os, BPSK, QPSK, BFSK, QFSK, channel
#import Coding
import channel
from math import ceil
from scipy.stats import norm
from ssdv_func import open_ssdv_input_file, compress_image, open_ssdv_file, encode, decode,
ssdv_decode_callsign, ssdv_packet_info, ssdv_packet_string

# ## Eleccion de la imagen SSDV

# In[2]:

path_input = './input/jy1sat.jpg'
path_output = './input/output_jy1sat.bin'
compress_image(path_input, 2, path_output, "EARTH_N0", 7)
file, packets = open_ssdv_input_file(path_output)

# In[3]:

# ## Ajustamos los valores de Tramision

# Elegir el tipo de Modulacion

mod = 'QPSK'

# tipo SSDV

tipo_ssdv = 'payload'

# Senal portadora
f_c = 1510.0
t_c = 1.0 / f_c
```

```
# Frecuencia de muestro y Bitrate
f_s = 48000.0
sr = 48000
t_s = 1.0 / f_s

# Mod Parameters
bit_rate = 50
Tb = 1/bit_rate

# In[5]:

Eb = 0.013095
# Senal a ruido
EbNodB = 5

# ## Cargamos la imagen SSDV

# In[12]:

tipo_ssdv = 'payload'

#file, packets = open_ssdv_file()

if tipo_ssdv == 'all':
    txBits = encode(file.read(256*packets))

elif tipo_ssdv == 'payload':
    list_header = []
    list_pay= []
    for p in range(packets):
        data = file.read(256)
        data0 = data[0]
        list_pay.append(encode(data[15:]))
        list_header.append(data[:15])

    txBits = np.concatenate(list_pay)

elif tipo_ssdv == 'header':
    list_pay= []
    for p in range(packets):
        data = file.read(256)
```

```
data0 = data[0]
list_pay.append(encode(data[1:]))
txBits = np.concatenate(list_pay)

# ## Funcion de Trasmision

# In[15]:

def transmit_recieve_with_noise(mod, txBits ,EbNodB,Eb, Tb, f_c, f_s):

    EbNo=10.0*(EbNodB/10.0)
    noise_std = 1/sqrt(2*EbNo)
    #msg = Coding.encodebits(txBits)
    msg = txBits
    if mod == 'BPSK':

        mmsg_norm = BPSK.modulate(msg, EbNo*Eb, Tb, f_c, f_s)
        mmsg_noise = mmsg_norm+channel.generate_noise(mmsg_norm, noise_std, f_s)
        dmsg = BPSK.demodulate(mmsg_noise, Tb, f_c, f_s)
        #dsmsg_decoded = Coding.decodebits(dmsg)
        Pb, Pb_pr = BPSK.error_probabilities(msg, dmsg, EbNo*Eb, noise_std)

    elif mod == 'BFSK':

        mmsg_norm = BFSK.modulate(msg, EbNo*Eb, Tb, f_c, f_s)
        mmsg_noise = mmsg_norm+channel.generate_noise(mmsg_norm, noise_std, f_s)
        dmsg = BFSK.demodulate(mmsg_noise, Tb, f_c, f_s)
        #dsmsg_decoded = Coding.decodebits(dmsg)
        Pb, Pb_pr = BFSK.error_probabilities(msg, dmsg, EbNo*Eb, noise_std)

    elif mod == 'QFSK':

        mmsg_norm = QFSK.modulate(msg, EbNo*Eb, Tb, f_c, f_s)
        mmsg_noise = mmsg_norm+channel.generate_noise(mmsg_norm, noise_std, f_s)
        dmsg = QFSK.demodulate(mmsg_noise, Tb, f_c, f_s)
        #dsmsg_decoded = Coding.decodebits(dmsg)
        ser, Pb, Pb_pr = QFSK.error_probabilities(msg, dmsg, EbNo*Eb, noise_std)

    elif mod == 'QPSK':
        M = 4
        k = int(np.log2(M))
        mmsg_norm = QPSK.modulate(msg, EbNo*Eb, Tb, f_c, f_s)
        mmsg_noise = mmsg_norm+channel.generate_noise(mmsg_norm, noise_std, f_s)
```

```
dmsg = QPSK.demodulate(mmsg_noise, Tb, f_c, f_s)
#dmsg_decoded = Coding.decodebits(dmsg)
ser, Pb, Pb_pr = QPSK.error_probabilities(msg, dmsg, EbNo*Eb, noise_std,k, M)

return mmsg_norm, mmsg_noise, dmsg, Pb, Pb_pr

# ## Tx/Rx con SNR

# In[16]:

mmsg_norm, mmsg_noise, dmsg, Pb, Pb_pr = transmit_recieve_with_noise(mod,txBits,EbNodB,Eb,Tb, f_c, f_s)

# ## Transformacion señal a imagen SSDV
#

# In[18]:

# Open temporary file for storing data.
temp_f = open("./temp/rxtemp_decod_1.bin",'wb')

if tipo_ssdv=='all':
    for p in range(0,packets):
        # Extract packet information.
        print(p)
        data = decode(dmsg[8*256*p:(p+1)*8*256])
        packet_info = ssdv_packet_info(data)
        packet_as_string = ssdv_packet_string(data)

        if packet_info['error'] != 'None':
            print('Error')
            continue

        print(packet_as_string)
        temp_f.write(data)

    temp_f.close()

if tipo_ssdv=='payload':
    p=0
    for packet_info in list_header:
        data = decode(dmsg[8*(256-15)*p:(p+1)*8*(256-15)])
```

```
packet = packet_info + data
packet_info = ssdv_packet_info(packet)
packet_as_string = ssdv_packet_string(packet)

if packet_info['error'] != 'None':
    print('Error')
    continue

print(packet_as_string)
temp_f.write(packet)
p+=1

temp_f.close()

if tipo_ssdv=='header':
    for p in range(0,packets):
        # Extract packet information.
        data = decode(dmsg[8*255*p:(p+1)*8*255])
        packet = b'\x55' + data
        packet_info = ssdv_packet_info(packet)
        packet_as_string = ssdv_packet_string(packet)

        if packet_info['error'] != 'None':
            print('Error')
            continue

        print(packet_as_string)
        temp_f.write(packet)

    temp_f.close()

# In[19]:

file,packets = open_ssdv_file()
callsign = ssdv_packet_info(file.read(256))['callsign']
os.system(f'./ssdv -d ./temp/rxtemp_decod_1.bin ./output/{mod}_{bit_rate}_{callsign}_{EbNodB}dB.jpg')
print(f'{mod}_{bit_rate}_{Eb}_{callsign}_{EbNodB}dB.jpg')

# In[20]:

from PIL import Image
im = Image.open(f'./output/{mod}_{bit_rate}_{callsign}_{EbNodB}dB.jpg')
```

```
plt.figure()  
plt.imshow(im)  
plt.show()
```

ANEXO B

Este anexo hace referencia a la relación con los Objetivos de Desarrollo Sostenible que involucran a este trabajo.

Este trabajo esta asociado con el ODS número 7 que se corresponde con el de “energía asequible y no contaminante”. No utiliza energías no contaminantes como tal, pero propone un sistema de transmisión de imágenes que saque mas provecho a la SNR dada por el canal para una mejor obtención de resultados en la visualización de imágenes obtenidas por los satélites mientras se encuentran en orbita. Además, muchos de los satélites implantados se cargan con luz solar, por lo que hay que optimizar la potencia de trasmisión de imagen para

Este trabajo se encuentra relacionado con el “Objetivo 4: educación de calidad” La lectura de este documento puede servir a futuros radioaficionados o estudiantes en como entender un sistema de transmisión de imágenes por satélite y su posible trasmisión en digital, mas las mejoras o ventajas que esto supone.

Este trabajo se encuentra relacionado con el “Objetivo 9: Construir infraestructuras resilientes, promover la industrialización sostenible y fomentar la innovación”, debido a la propuesta y investigación de la mejora de los satélites de radioaficionados de AMSAT para una mejor obtención de imágenes de los satélites.

Este trabajo se encuentra relacionado con el “Objetivo 17: Revitalizar la Alianza Mundial para el Desarrollo Sostenible”, todos los países lanzan satélites anualmente, SSDV y gr-satellites buscan una manera de unificar una este de trasmisión de imágenes por satélite de manera digital.