**Abstract**—The task of obtaining the performance of a communications receiver is difficult, since it involves the implementation of the physical layer transmission and reception chains and also requires great computational time, since Monte Carlo methods make up the base of link-level simulations. The DVB-RCS2 standard provides some receiver performance data which is rather restrict. This limits the capacity analysis made through a system level simulator. The objective of this work is to assess the DVB-RCS2 performance using a Python simulator developed by the authors for a much broader range of SNR values and for a comprehensive set of transmitter-receiver configurations.

**Keywords**—DVB-RCS2, turbo coding, satellite communications, simulation.

**I. INTRODUCTION**

Satellite systems have been broadly used by the communications industry. One example is the DVB-S system, that was developed to deliver digital television over satellite and serves over 100 million receivers [1]. The DVB-S defines only a broadcast channel, and was expanded (DVB-S2 and DVB-S2X) in order to be more robust and to adapt to the oncoming technologies. In response to requests from the industry, a return channel (DVB-RCS) was incorporated into the DVB standard to provide interactivity with the remote user. Further, the RCS standard evolved to DVB-RCS2, encompassing a mobility element, and more advanced network and physical layer techniques [1].

While deploying a new communications system technology, it is crucial to evaluate the performance of the proposed link interface. This information can be used not only to aid the development of the receiver terminal and characterize its capabilities, but also to feed higher level simulators, such as system simulators, with the physical link statistics. For the DVB-RCS2 system, the standard provides the user with the SNR required for only a chosen set of packet error rate (PER) performance, namely $10^{-3}$ and $10^{-5}$, for the additive white gaussian noise (AWGN) channel [2].

This work presents the RCS2 receiver performance for a much broader range of SNR’s, that can be used to allow a broader set of design rules through system level simulations. It also reviews key aspects of Turbo channel coding and decoding schemes and its implementation using the Python programming language.

Rodrigo A. R. Fischer, Electrical Engineering Department, University of Brasília, Brasília, Brazil, e-mail: rodrigoafischer@gmail.com; João Paulo Leite, Electrical Engineering Department, University of Brasília, Brasília, Brazil, e-mail: jpauleite@unb.br.
The extrinsic information obtained from the second decoder is summed with the first decoder input LLR’s. The Max-Log-MAP algorithm also introduces an overestimation of the extrinsic LLR’s, which requires the extrinsic information to be multiplied by a scaling factor on every but the last one iteration [4]. The best observed scaling factor on the reference [4] was 0.7, and shall be adopted on this work.

IV. RESULTS

The performance curves, shown in Figures 3 and 4, containing the packet error rate (PER) of a system composed of the whole DVB-RCS2 transmission-reception (TX-RX) chain with AWGN channel were obtained for the 28 waveform ID’s (WF-ID) whose performances are in [2]. The packet sizes are the same as provided by [6], varying from 38 to 599 bytes, and coincide with the burst sizes. The performance points provided by [2] are shown as dots at $\text{PER} = 10^{-3}$.

V. CONCLUSION

A Python simulator was developed to assess the performance of the DVB-RCS2 satellite communications standard. The obtained performance has matching points in accordance with the standard, thus validating the implemented TX-RX processing chain. For a further work, channel models such as Rayleigh fading or Rice fading could be implemented to assess the impact of the propagation channel on the performance of the system.

REFERENCES