Harnessing technology for livestock research: an online sheep behavior monitoring system

V. Cabrera, A. Delbuggio, H. Cardoso, D. Fraga, A. Gómez, M. Pedemonte, R. Ungerfeld and J. Oreggioni

***Abstract*—Sheep production in extensive conditions faces several challenges. These challenges could be addressed with behavior monitoring systems, contributing to animal well-being, enhancing animal research, and improving productivity.**

This work presents the design, manufacture, and test of an online sheep behavior monitoring system for extensive conditions. It comprises a wearable electronic collar device and a cloud server (deployed with Amazon Web Services) for storing data and providing a web user interface. The collar has an Icarus IoT Board, allowing motion data collection with a three-axis accelerometer, GNSS (Global Navigation Satellite System) location data acquisition, and Narrowband IoT communication. The device has solar panels and a battery. Our application acquires accelerometer data at 25 Hz, location data every 10-30 s, and battery level and cellular signal strength every 50 s. We encoded accelerometer samples to reduce the transmitted data.

We manufactured thirty collars that collect and transmit data to the cloud server. Our system facilitates data processing, both collar and server side. We introduce a preliminary Random Forest algorithm for behavior classification on the device that identifies 'still', 'walking', and 'running' with an 78 % general accuracy. The device's autonomy exceeds ten days in continuous operation (streaming raw and processed data) while if the device transmits only processed data and GNSS data every four hours, autonomy rises to 100 days. This allows us to glimpse the application of this system in long-term research experiments and farming production.

***Index Terms*—Wearable device, Internet of Things, IoT, embedded systems, animal behavior monitoring, accelerometer data processing, machine learning, random forest.**

I. Introduction

In Uruguay, sheep production is mainly developed in extensive systems based on grazing natural pastures in large paddocks, often grazing together with cattle [2]. A central production issue in these conditions is the high lamb mortality rate, which limits the efficiency. This situation is similar in other countries that use comparable production systems [3]. During wintertime, concurring with the gestation period, there is a severe reduction in pasture availability and quality, leading to undernutrition during the placenta and fetal growth. The weather, including temperature and rainfall, is highly variable throughout the year, and across different years, making it difficult to predict forage availability and also to make animal handling decisions according to the expected productive conditions. However, in extensive conditions and due to the farms extension and variability, it is difficult to have real-time data on productive situations. Therefore, having real-time information in key moments, such as breeding, shearing, or lambing, may be extremely helpful. This also includes health problems and diseases, or the presence of predators. In these situations, there are changes in ewes', rams', or lambs' activities and location in relation to the flock, so recording information with monitoring systems that can process the information rapidly can be useful to develop alarms or provide information for taking quick decisions. For example, farmers can apply different animal management strategies, such as using protected spaces or farrowing crates, offering specific diets, or aiding during lambing. Therefore, online monitoring systems to study sheep behavior that record, or ideally predict, and communicate main events, such as lambing, presence of predators, or estrus, would have a positive impact on productivity and welfare, providing the opportunity for the farmers to make decisions based on real-time flock information.

Behavior monitoring systems have to be low-cost, comfortable for the animal, and require low maintenance. The low-cost requirement is justified because a lamb costs between 60 and 100 United States dollars. Therefore, each collar should cost much less than this figure. On the other hand, low maintenance is necessary in extensive production where the sheep live for several weeks or even months freely, and there is no infrastructure or human resources to bring them in to change a battery, accommodate an antenna, download data, or clean or fix the device's case. Also, excessively moving animals for these purposes would affect animal welfare. Finally, it is essential that the device feels comfortable because if the animal is uncomfortable, it may attempt to remove the collar and could get hurt. Moreover, if the device affects the behavior, this would lead to the recording of biased behavior data. Some of these requirements do not apply to research because this context offers more human resources and infrastructure, many experiments do not last more than a week without controlling the animals, and the investment allowed is much higher than in farming production. We initially designed our monitoring system targeting research, but keeping present these requirements to be able to transfer the system to a production environment in the medium term.

Several research approaches have proposed systems for monitoring sheep behavior (for a comprehensive review, see [4]) and particularly for studying behavior with accelerometer

This work is a substantially extended and improved version of [1]. This work was partially funded by EI, CSIC and CAP, from Universidad de la República (Udelar), Uruguay. The procedures with animals were approved by the Ethics Committee, Fac. de Veterinaria, Udelar.

All authors are with Universidad de la República, Uruguay. Varinia Cabrera, Andrea Delbuggio, and Julián Oreggioni are with Departamento de Electrónica, IIE, Facultad de Ingeniería. Álvaro Gómez is with Departamento de Procesamiento de Señales, IIE, Facultad de Ingeniería. Hernán Cardoso, and Martín Pedemonte are with Laboratorio de Computación Heterogénea, INCO, Facultad de Ingeniería. Diego Fraga is with DVL Group (Industrial Design). Rodolfo Ungerfeld is with Departamento de Biociencias Veterinarias, Facultad de Veterinaria. E-mail: juliano@fing.edu.uy

data (see [5]). One approach uses commercial data loggers with accelerometers [6], [7]. These solutions are portable and comfortable for the animal. However, they are expensive, require maintenance, usually do not provide online information, and do not perform on-device classification. Another approach develops a custom device for behavior monitoring [8]–[12]. In this case, the solution can provide online information, perform on-device classification, transmit wireless raw data, and keep track of the extensive production requirements: low maintenance (high autonomy), low cost, and animal welfare.

In this work, we present the development of a research platform that represents the new generation of our prior work [11], [12]. The proposed platform allows experiments with a remarkable capacity to handle large amounts of data, including data collection, on-device data processing, and real-time raw data transmission. The platform enables the researchers to perform two kinds of experiments. On the one hand, *short-term experiments* (up to 11 days), where the collar streams raw acceleration data and on-device processed behavioral data (the state of the sheep). On the other, *medium-term experiments* (up to 100 days) where the collar sends on-device processed behavioral data. In these two types of experiments, the device can send other information, like GNSS location or status data. We send raw data for three main reasons. Firstly, to gather new data to improve our classification algorithms by increasing the number of states and enhancing its accuracy. Secondly, to store and publish the raw data along with the labeled one. Finally, to process the data at the server level, which allows the integration of other sources of information (data to improve GNSS accuracy, satellite images, weather, data from other platforms, from other collars, etc.).

This work significantly extends our conference paper [1]. The main advancements include developing a second version of the device's embedded software with an integrated sheep behavior classifier, fine-tuning the accelerometer frequency and power settings to improve autonomy. Additionally, we have performed live sheep behavior observations to obtain real raw and labeled data. We also expanded the experimental analysis and provided more details on communication security, and data acquisition and transmission limitations.

The rest of the article is organized as follows: Section II describes the proposed system, including the design of the electronic collar and cloud server solution. Section III covers accelerometer data processing for sheep behavior classification on a personal computer (PC) and in the collar. Section IV details the empirical study, evaluating behavior classification, device autonomy, geopositioning accuracy, and communication quality. Section V is dedicated to discuss future work and compare our solution with the state of the art. Finally, Section VI outlines the conclusions.

II. Proposed Solution

The proposed system (see Fig. 1) consists of a wearable collar electronic device that collects movement and location data, and a cloud server that stores data and provides the web user interface on a Personal Computer (PC), cellphone or similar. It uses the Message Queuing Telemetry Transport (MQTT) protocol and Narrowband IoT (NB-IoT) communication.

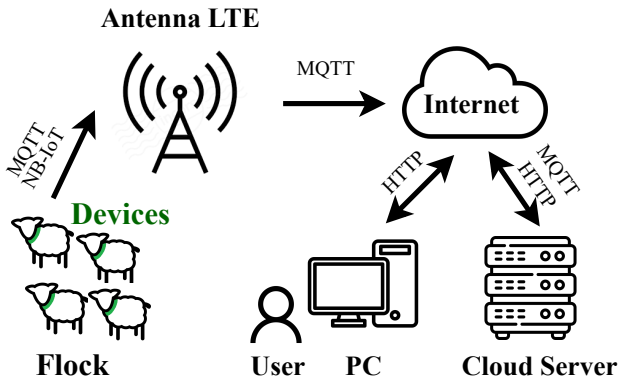

Fig. 1. Functional diagram of the system.

We opted for NB-IoT because the cellular operator manages all the communication infrastructure. In our solution, the device does not require additional components, such as antennas, servers, or hubs. Besides, all the devices have identical designs and are easy to use and install. Extensive livestock farming solutions require simple solutions, as lacking simplicity would result in a non-operative system.

The device sends encoded accelerometer data collected at 25 Hz, behavioral data computed every 5 s, location data obtained every 10-30 s, battery level, and Long-Term Evolution (LTE) signal strength every 50 s, to the cloud server. All these frequencies are configurable, and we can use any lower value that suits the particular application. Our previous version [1] acquired accelerometer data at 100 Hz. In this work, we reduced this value to 25 Hz because the original value pushes the system to its limit (mainly in terms of transferred data in short-term experiments), and the state of the art does not typically use frequencies higher than 25 Hz.

A. Hardware

A hardware diagram of the device can be seen in Fig. 2. The device features Actinius' Icarus IoT Board, which includes Nordic Semiconductor's nRF9160 chip. This chip is a low-power system in a package that includes a NB-IoT modem and a Global Navigation Satellite System (GNSS) chip. The nRF9160 chip is equipped with an ARM Cortex-M33 processor, which includes 1 MB of flash and 256 kB of RAM. The board incorporates the LIS2DH 3-axis accelerometer from STMicroelectronics. The power management circuit, based on the BQ24074 chip from Texas Instruments, allows the device to be powered from various sources: three solar panels (totaling 2.8 W), a 2500 mA h Lithium Polymer battery, and a USB cable. Other energy harvesting and storage solutions were considered and discarded (see [13] for futher details).

B. Embedded software

The embedded software, developed in *Visual Studio Code* using *nRF Connect*, utilizes *Zephyr RTOS* and C programming language. The software is based on the *Asset Tracker v2* application from Nordic Semiconductor [14]. The *Asset Tracker v2*

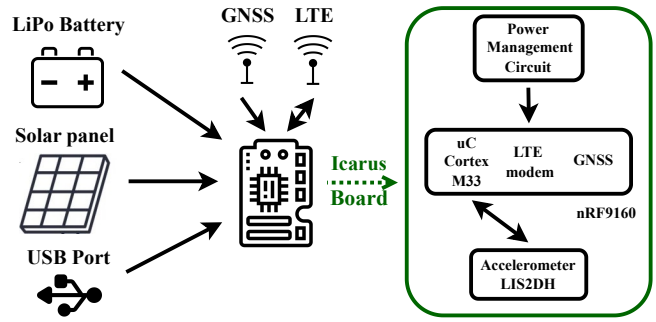

Fig. 2. Hardware diagram of the device. Figure taken from [1]

.

software architecture is modular, featuring an event manager for software modules communication. Modules can submit and subscribe to events. Modules *with threads* convert events into messages and enqueue them on the system work queue. Modules *without threads* directly process events by converting them into messages. Certain modules have dedicated threads for handling time-consuming messages.

The *Asset Tracker v2* operates in both *active* and *passive* modes, supporting the *AWS (Amazon Web Services) IoT Core* cloud service. In *active* mode, it transmits location, environmental data, battery status, and LTE information at regular intervals. In *passive* mode, it sends this data when movement is detected by the accelerometer, if no movement is detected, then data is transmitted at extremely large intervals. We used the general structure of *Asset Tracker v2*'s *active* mode, incorporating the accelerometer data collection present in the *passive* mode (rather than implementing a motion detector, since we are interested in raw accelerometer data).

We made several adaptations and improvements to the *Asset Tracker v2* application to tailor it for our specific requirements. The main modifications included: (a) integrating a driver for the LIS2DH accelerometer, and (b) adapting the overall software's operation to acquire acceleration data up to 25 times per second. We also, (c) disabled the retrieval of environment data since it is not required for our application, and (d) modified the collection of battery level to work with our power management circuit. Additionally, due to the high volume of accelerometer data, (e) we encoded this information before sending it to the cloud server. Finally, (f) we integrated an animal behavior classifier based on machine learning techniques (Section III).

Due to memory limitations, approximately 5000 acceleration samples (50 s) can be stored on the nRF9160 chip. Each acceleration sample consists of 3 values (one value for each accelerometer axis). As the sensor is configured with a $\pm$ 4000 mg scale, the acceleration take values from -4000 mg to 4000 mg. For this reason, we need five characters (four digits and one character for the sign) to represent the acceleration samples in a human-readable format. If we represent each acceleration value with five characters, it gives a total of 5 x 3 = 15 characters per sample. To reduce this data, we encoded the acceleration values in base64 according to the RFC4648 standard. In this way, we represent each sample with three characters (no padding characters are sent since the samples have a fixed width), reducing the number of bytes needed from 15 to 3 x 3 = 9 bytes, which is 60 % of the original payload.

Regarding transmission and acquisition times, the nRF9160 has one limitation: the GNSS can not obtain a location while the modem is transmitting. Therefore, the time between transmissions to the cloud server depends on the GNSS configuration. Location data is acquired once at startup, switched off after obtaining the first satellite fix, and periodically activated. The GNSS needs at least 30 consecutive seconds when locating the satellite for the first time or when satellite position data expires (every 2 or 3 hours). Hence, in these cases, LTE must remain idle for at least 30 s. Furthermore, the acceleration samples occupy most of the packet that is transmitted to the server. The transmission of this packet takes the modem about 20 s. As a result, while the modem is not transmitting, up to three locations per packet are acquired (at 10, 20, or 30 s) and sent to the cloud server every 50 s.

C. Cloud server and web user interface

The software solution architecture shown in Fig. 3 includes the back-end (MQTT Broker and Computer resource), which processes the received device messages; the front-end (Website), which provides a user interface to track information; and the Database, which stores the information. We deployed the solution using the AWS ecosystem as a cloud server.

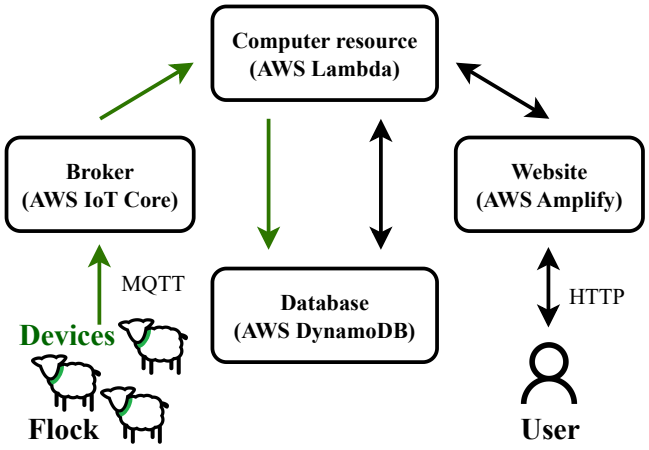

Fig. 3. Architecture of the software solution.

AWS IoT Core enables us to connect multiple devices and efficiently direct received messages to other services. Messages received by the Broker in a topic are immediately sent to AWS Lambda functions via an AWS feature called *Rules*. For each message, a new instance of a *Node.js* environment (that runs JavaScript code) is created, achieving scalability without bottlenecks. We also opted for AWS DynamoDB to ensure scalability. This service offers a key-value non-tabular database (NoSQL), suiting our needs as it prioritizes writing data over reading and handles messages that may not always have the same fields.

The cloud server (AWS Amplify) also hosts a website for interacting with the stored data (see Fig. 4). We developed the website using NextJS 13, a React framework that provides

server-side rendering. This decision was made with the end users in mind, considering they may access the application from various devices (particularly older devices).

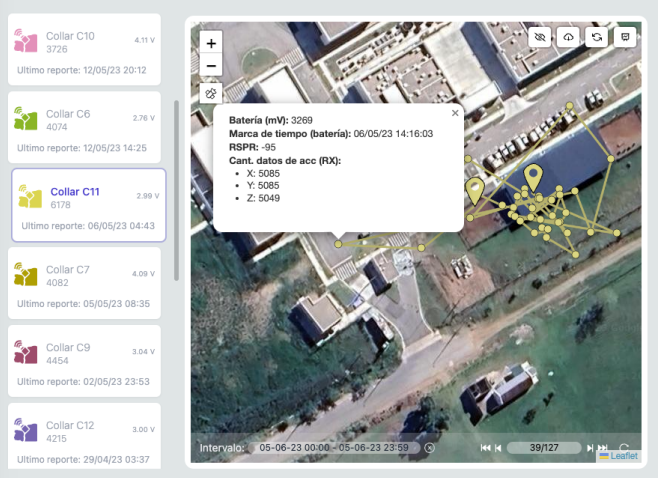

Fig. 4. User interface showing sheep location data. Figure taken from [1].

D. Secure communication

Our devices connect to AWS IoT Core over a secure Transport Layer Security (TLS) using MQTT transmission protocol. The secure communication to AWS IoT Core involves authentication and authorization mechanisms. Authentication verifies a device's identity; here, MQTT authentication uses mutual TLS with X.509 certificates.

The CA (Certificate Authority) allows the device to verify the identity of the server-side. The device certificate is a key pair, public-private, allowing AWS to verify the identity of the device. The public key is used to encrypt data that only the corresponding private key can decrypt. The public key is then uploaded to AWS IoT, while the private key remains securely stored on the device. To generate the certificates (CA certificate, public-private key pair) for one device, an *Thing* (endpoint) can be created in AWS IoT Core. Certificates can be generated for the *Thing* and then configured in the device. The *Cellular Monitor App* within *nRF Connect SDK* from Nordic Semiconductor is used to provision the certificates onto the device. Additionally, the device's embedded software configuration must include the *Thing* name and internet address.

Authorization defines what the device is allowed to do after it has been authenticated. AWS IoT Core policies are created to determine permissions for a *Things Group* (a group of devices that can have the same permissions).

E. Mechanical design of the case

The device's case was designed to ensure proper functioning while also considering animal welfare (Figs. 5 and 6). The device must withstand adverse weather and endure impacts resulting from interactions between the animals. Regarding animal welfare, weight (the goal was 500 grams or less), volume, and color were important. In the initial stages, it was decided to use fluorescent/bright colors for visibility purposes. However, using these colors would not be advisable as they could attract predators. Therefore, neutral colors such as white or light gray were chosen. Black was discarded due to its tendency to accumulate heat.

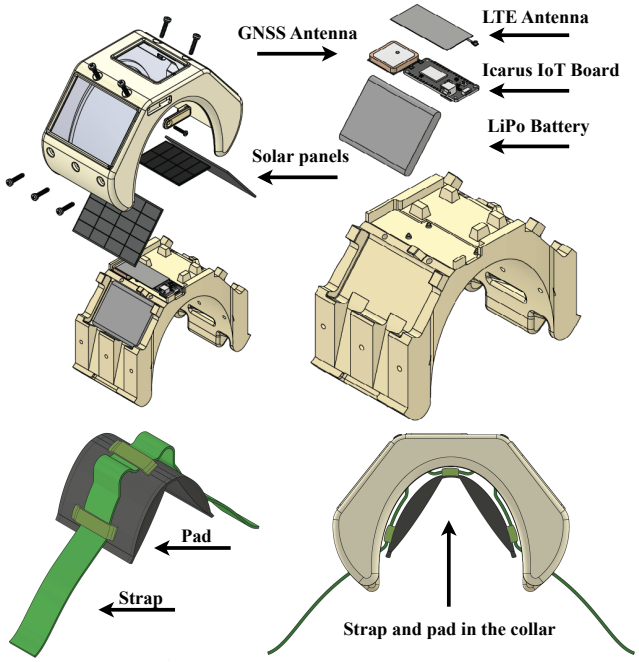

Fig. 5. Device's mechanical design: complete collar (left, up), bottom case with internal components (right, up), strap and pad (left, down), strap and pad in the collar (right, down). Figure taken from [1].

Furthermore, a variation in neck size due to wool growth/shearing was a challenge, as well as the lanolin fat. Regardless of the collar adjustment system, different-sized enclosures were necessary to accommodate morphological differences in sheep, considering ages, sex and breeds: a small enclosure with a filled pad for small animals, a small enclosure with an empty pad for medium-sized animals, and a big enclosure with an empty pad for larger animals. This helps with collar fixation, preventing it from rotating.

Because of the quantity needed (thirty collars), the Fused Deposition Modeling (FDM) 3D printing technology was chosen, with the addition of transparent polycarbonate windows in the solar panel areas. Production was done using polyethylene terephthalate glycol-modified (PETG) filament considering the high mechanical resistance requirements. This was complemented by an internal cellular structure, which allowed weight reduction without compromising strength, achieving a weight of 480 grams (no ill-being caused to the sheep).

III. Accelerometer Data Processing

A signal processing technique running on the device must have reliable performance and be simple enough to be implemented in a constrained resource environment (like a microcontroller) to achieve on-animal classification. For instance [15] compare different algorithms' performances (Support Vector Machine, Linear Discriminant Analysis, Deep Neural Network, etc.), whereas in [16] excellent results are obtained

using Principal Component Analysis to select significant features and Random Forest (RF) to classify activities.

At this stage, we aim to successfully implement an algorithm trained on the PC with the inference performed on the microcontroller. This algorithm should be considered a proof of concept; we are not focusing on the performance classification results. Instead, we are centered on effectively running a classification algorithm in our collar. Our first approach is based on RF, which utilizes multiple decision trees for both classification and regression tasks. In RF each decision tree independently predicts a class (sheep behavior), and the final prediction is determined by selecting the most frequent class prediction among all the trees. The simplicity of RF makes it suitable for implementation with limited resources.

A. Classifying sheep's behavior on the PC

We developed a PC version of the sheep's behavior classifier using RF and the dataset from [15] in Python language. Our goal was to classify three behaviors ('still', 'walking', 'running'). The dataset provides raw accelerometer data from two sheep recorded at 200 Hz with a data logger. We performed decimation by selecting every 8th sample to obtain a 25 Hz time series. This process reduces the sampling rate by a factor of 8. Then, we calculated nine different features in a 5-second sliding window (see Table I). Calculations were performed both for each acceleration axis and for the acceleration magnitude vector (36 features total = 9 features x 4 variables). Also, we selected the five most relevant features with a K Best technique. The selected features were minimum value and standard deviation in the magnitude vector, standard deviation and zero crossings in the z-axis and standard deviation in the y-axis. A specific RF with 10 trees was selected with 10-fold cross-validation and achieved a general 98 % accuracy.

TABLE I
CLASSIFIER FEATURES DESCRIPTION

Feature	**Definition**
Mean	$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i$
Standard deviation	$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i-\mu)^2}$
Minimum value	Minimum value in N samples
Zero crossing	Number of zero crossings in N samples
Crest factor	Maximum value in N samples divided by RMS[I]
25% percentile	Value below which 25% of samples are found
Entropy	$-\sum_{i=1}^{N} p(x_i)log(p(x_i))$ [II]
Kurtosis	$(\frac{1}{N}\sum_{i=1}^{N}(x_i-\mu)^4)/\sigma^4$
Skewness	$(\frac{1}{N}\sum_{i=1}^{N}(x_i-\mu)^3)/\sigma^3$

[I] Root Mean Square.
[II] $p(x_i) = \frac{|X_k|}{\sum_{k=1}^{N}|X_k|}$, $X_k = \sum_{n=0}^{N-1} x_n e^{-i2\pi kn/N}$, $k = 0, .., N-1$

B. Classifying sheep's behavior on the collar.

To integrate the PC-trained RF, presented in Section III-A, into our microcontroller, we employed the *Embedded Learn* (*Emlearn*) tool [17]. This tool is designed to export a Python-trained machine-learning algorithm, enabling inference execution on a device with a C99 compiler.

Emlearn generates C code housing an RF model, capable of inference that closely approximates the one trained in Python. The term "approximates" is employed because, while in this case the overall accuracy matches that of Python, the confusion matrix shows slight discrepancies. This difference may lead to a marginal reduction in accuracy for the RF model implemented in C. The tool also provides a C library, *eml_trees.h*, with the capability to execute inference on the microcontroller, using the specific RF model previously exported.

To incorporate the RF classifier into our embedded software, it was necessary to compute the RF's input features directly on the microcontroller. The features are calculated using a sliding window of 125 accelerometer samples, equivalent to a 5-second window with samples acquired at 25 Hz, and a 52 % overlap. The features values enable the inference of a sheep's state every 2.4 s, creating a state vector that is transmitted to the cloud server at intervals of 50 s along with the corresponding raw acceleration data.

IV. EXPERIMENTAL RESULTS

Before manufacturing the devices, several testing stages were conducted. A resistance study was executed using Autodesk's Inventor software. Also, tests with volumetric models without internal components were performed. This validated sealing, attachment, and resistance. Then, manufactured collars were tested, including electronics (see Fig. 6). The case has adequate robustness, is watertight, suitable for outdoor operation, and does not cause any discomfort to the animals. We verified the system's functionality in its entirety, including data collection, processing (behavior classification), real-time transmission to the cloud server, and data display on the user interface. All the experimental testing was conducted in Corriedale and East Friesian ewes.

Fig. 6. Designed devices being tested in several sheep.

A. Behavior characterization through live observations

Our collar device enables the user to monitor the sheep's behavior in real-time via the classification algorithm. Therefore, evaluating the performance of the classification algorithm requires access to the actual sheep's behavior information. Thus, we need to compare the predicted states by the device to the labeled states collected through live observations. Additionally, we need to synchronize collar data with observation data.

Characterizing actual sheep behavior presents its challenges, particularly due to natural overlaps in common activities, such as 'walking' and 'grazing'. While a sheep might be 'grazing' and 'walking' simultaneously, it could also be grazing and standing concurrently. To ensure qualitative and clear information, we opted to construct an ethogram (see table II) with three distinct categories – one for locomotion, one for posture, and one for activity –. Each category has mutually exclusive states. This ethogram is similar as the one presented in [18]. Although we labeled data with three categories, in this stage we only used the locomotion category since our algorithm is trained only for this category.

TABLE II
ETHOGRAM USED FOR LABELING DATA

Category 1 Locomotion	Category 2 Posture	Category 3 Activity
Still	Standing	Grazing
Walking	Lying	Ruminating
Running	Moving	Drinking
		Other

The live observation was conducted in several trials, each lasting two to three hours within a controlled environment. A group of five adult ewes was selected, with devices installed on two or three of them simultaneously. This aimed to validate that the integration of collared sheep with those without collars had no discernible impact on the collective behavior of the entire group due to the presence of the devices.

Live observation comprised both human annotations and video recordings of the sheep. Video recordings were captured using a GoPro Hero 3+ camera during the time when the devices were affixed to the sheep. Synchronization was performed in front of the camera with a significant movement on the collar after a period of no movement at all. This synchronization could be observed in accelerometer data and video. Video annotations were created on the recorded footage. The video also recorded the initiation of human annotations.

Human annotations were made as a support for the video annotations by one observer per sheep at one-minute intervals over a period of two hours. To minimize distractions, each observer was relieved every 20 minutes. Combining both annotations, the sheep's activity was accurately labeled.

B. Classification algorithm performance

As mentioned in Section IV-A, evaluating the performance of the classification algorithm requires access to the actual animal behavior. The sheep's behavior predicted by the RF algorithm implemented in the microcontroller with 10 trees and three classes ('still', 'walking', and 'running') was compared to the true labeled behavior. This resulted in a general accuracy of 78 %. Results for the 'still' class are excellent, with 90 % precision, 85 % recall, and 87 % F1-score. But neither the 'running' nor 'walking' classes present good performance (47 % and 44 % F1-score, respectively). Hence, we decided to combine the 'running' and 'walking' classes into a new class, 'movement'. With only two classes ('still' and 'movement'), there is a slight improvement in the 'movement' class (52 % F1-score). This improvement is because 'running' is mostly confused with 'walking', while 'walking' is mostly confused with 'still'.

C. Autonomy

Fig. 7 shows a battery discharge cycle reported by three different versions of our device during the experiments on animals. Table III summarizes main settings of these versions. In version 1 (v1, red line), we disconnected the solar panels, decreased the accelerometer data rate from 100 Hz to 25 Hz (which relates to the amount of transferred data), and switched off the GNSS. These changes almost double the autonomy compared to our previous published version (blue line) [1]. Preliminary results indicate that GNSS is almost exclusively responsible for the increase. The version 2 (v2, yellow line) shows that reducing the PSM-AT (Power Save Mode Active Time) to zero increases the autonomy by 20 %, and running the RF behavior classifier has negligible impact.

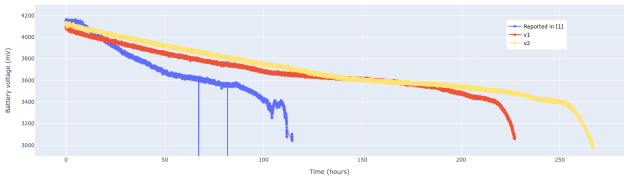

Fig. 7. Discharge of the device's battery for different settings (see Table III)

TABLE III
AUTONOMY OF THE DEVICE FOR DIFFERENT SETTINGS

Ver	Accel Rate (Hz)	Payload (kB)	GNSS	PSM AT (s)	RF	Solar panel	Autonomy (hs)
[1]	100	45.2	yes	20	no	yes	114
v1	25	11.3	no	20	no	no	227
v2	25	11.3	no	0	yes	no	267

The PSM-AT regulates the time the device waits to receive messages from the cloud server [19]. Setting it to zero reduces power consumption, not allowing the reception of messages. In our application, where the cloud server only sends messages at the beginning of the connection (i.e., to change configuration parameters), setting the PSM-AT to zero is not a disadvantage.

Then, the autonomy exceeds 267 hours (11 days) in continuous operation (characterized by streaming raw and processed accelerometer data). We estimate that if the device transmits only processed sheep behavioral data to the cloud server, the autonomy rise to more than 100 days. These results are promising as we still have improvements to implement on the microcontroller side and fully incorporate the solar panels.

D. Accuracy of the GNSS

Although Fig. 4 shows some points outside the paddock, it was not possible that sheep had been there. Based on these observations, we did tests on still collars. The mean of the reported error is $14.4\,\mathrm{m}$ with a standard deviation of $13.9\,\mathrm{m}$. These experiments show that GNSS accuracy needs to be improved. Preliminary analysis shows that filtering outliers significantly increases accuracy. Moreover, differential correction will be evaluated in the next stages.

E. Assessment of the Device - cloud server communication

Table IV presents the results of tests performed using the national carrier's NB-IoT service. Some tests were performed using only one device from our previous work (marked as "[1]" in Table IV), and others were performed using four devices v2 running simultaneously. RSRP (Reference Signal Received Power) indicates the cellular signal strength. The higher the values of RSRP, the better. For instance, an RSRP value higher than -80 dBm means an excellent cellular signal strength, between -80 dBm to -90 dbm is good, between -90 dBm to -100 dBm is fair, and below -100 is an indication of a poor signal strength. These tests are conducted as part of an ongoing collaboration and provide valuable feedback for the carrier to enhance its infrastructure. However, it is important to note that the NB-IoT infrastructure in the Uruguayan countryside still requires improvements.

TABLE IV
DEVICE - CLOUD SERVER COMMUNICATION TESTS

Location of the facility	RSRP (dbm)	Data rate (packets/min)	Payload (bytes)	Received packets	Device
rural	-135	12.0	300	54.7 %	[1]
rural	-105	12.0	300	83.3 %	[1]
urban	-95	1.2	45.2k	95.8 %	[1]
urban	-75	1.2	45.2k	96.0 %	[1]
suburban	N/A	3.0	300	95.9 %	[1]
suburban	N/A	0.5	300	99.6 %	[1]
suburban	$>$ -95	1.2	11.3k	95.0 %	v2

V. DISCUSSION

The implemented platform enables research on animal behavior for extensive livestock production and allows us to envision the generation of concrete online and offline services for farmers. With the data that will be collected using the developed platform, it will be possible to design algorithms that can run in the cloud or the collars, enabling the creation of behavioral analytics, alarms, and other relevant services.

Regarding classification we successfully implemented the proof of concept of an algorithm with limited computational resources in the microcontroller. The RF algorithm with 10 trees and three classes ('still', 'walking', and 'running') has shown a general accuracy of 78 % predicting sheep's behavior in live experiments. Our algorithm has great results for the 'still' class. However, the classification falls short in the other classes. Current work includes training an new RF algorithm only with our own dataset. This algorithm is showing promising results but they are still not adequate. This is mainly because the data points for 'running' and 'walking' are insufficient to properly train our classifier. 'Running' and 'walking' have approximately 60 % fewer data points than 'still'. Imbalanced data can negatively impact an RF classifier. Hence, we need to generate more data, which translates into more experiments with live observations. Expanding our dataset could fix problems like training with imbalanced data.

Table V compares our platform with other works [7]–[10]. The overall performance of the system aligns with the state-of-the-art. Our work stands out due to its ease of use and installation, long-range communication without adding infrastructure, adequate autonomy, and on-device processing. These characteristics make it suitable for extensive livestock.

Future work includes training other RF algorithms to have behavior prediction for sheep's locomotion, activity and posture. The comprehensive predictions could give the animal researcher and the farmer a complete picture of the sheep's behavior in real time. Moreover, technological solutions for extensive livestock farming should be affordable. Not only the cost of the devices must be considered, but also the cost of their installation, operation, and maintenance, including the communications infrastructure. Currently, we are analyzing placing collars only on certain representative animals instead of the entire flock to reduce costs. Identifying the "representative" animals is an interesting challenge. Also, we plan to integrate more sensors into the platform. For example, a subcutaneous temperature sensor, which communicates and powers itself wirelessly. Also, cardiac and respiratory frequencies are variables that we are considering incorporating.

VI. CONCLUSIONS

The end-to-end platform for data acquisition, processing, storage and display, is completed. Tests have been conducted on research facilities using our collars, the national carrier NB-IoT service, and AWS cloud server. This achievement has been possible thanks to the close collaboration between researchers from Veterinary, Engineering, and Industrial Design.

Our tests indicate that the device's autonomy exceeds 267 hours (11 days) in continuous operation (characterized by streaming raw and on-device processed accelerometer data). These results are promising because the performance is enough to execute common research experiments, and we still have improvements to implement on the device side, including fully incorporating the solar panels. We estimate that if the device transmits processed sheep behavioral data and GNSS data to the cloud server every four hours, the device's autonomy rises to 100 days. Regarding behavior classification, we implemented a proof of concept algorithm in the microcontroller that shows an accuracy of 78 %. We believe that a more substantial dataset will improve classification. The whole system performance allows us to glimpse the application of this system in long-term research experiments not carried out until now, and the use of this technology in farming.

ACKNOWLEDGMENT

The authors are grateful to Victoria Fernández, Jimena Fernández, Lucía Sirio, Ana María García, Alicia Fernández, Aline Freitas de Melo, Livia Pinto, and Noelia Zambra.

TABLE V
STATE-OF-THE-ART COMPARISON

	[7]	[8]	[9]	[10]	This work
Application	Parturition	GBH	GBH	GBH	GBH
Communication	No	propietary	propietary	propietary	NB IoT
Online information	No	Yes	Yes	Yes	Yes
On-device classification	No	Yes	Yes	Yes	Yes
Raw data transmission	No	No	No	Yes	Yes
Device					
Custom-made device	No	Yes	Yes	Yes	Yes
Device location	Neck and Ear	Neck	Ear	Neck	Nape
Autonomy (days)	>15	>1	2.4	N/A	11 - 100
Accelerometer sample rate (Hz)	12.5	100	16	50	25
GNSS sample period (min)	2 - 3	0 - timer to be set	No	No	0.2 - 240
Memory	64 MB	2 GB	384 kB	32 kB	1 MB
Weight (g)	N/A	281	N/A	N/A	480
Size (cm)	N/A	14.6 × 8 × 6.5	N/A	N/A	19.2 x 10.2 x 12.0
Microcontroller / SoC	N/A	MSP430FR5739	Quark SE C1000	CC1110	nRF9160
Classification					
Number of behaviors	1 (Partuition)	5	3	5	3
Accuracy (%)	91	82.4	85.2	91.8	78
Classification algorithm	SVM	LDA	KMeans & KNN	DT	RF
Number of features	4	12	20	11	5

GBH = General Behaviour Classifier, N/A = Not Available, SVM = Support Vector Machine, LDA = Linear Discrimant Analysis KNN = k-nearest neighbors, DT = Decision Tree, RF = Random Forest, SFS = Sequential forward selection.

REFERENCES

[1] V. Cabrera, A. Delbuggio, H. Cardoso, D. Fraga, A. Gómez, M. Pedemonte, R. Ungerfeld, and J. Oreggioni, "Research platform to study sheep behavior," in *2023 IEEE Conference on AgriFood Electronics (CAFE)*, 2023, pp. 60–64.

[2] SUL, "Manual práctico de producción ovina (in spanish)," https://www.sul.org.uy, Montevideo, Uruguay, 2018, last accessed 1 June 2023.

[3] F. Montossi, M. Font-i Furnols, M. Del Campo, R. San Julián, G. Brito, and C. Sañudo, "Sustainable sheep production and consumer preference trends: Compatibilities, contradictions, and unresolved dilemmas," *Meat science*, vol. 95, no. 4, pp. 772–789, 2013.

[4] E. S. Fogarty, D. L. Swain, G. Cronin, and M. Trotter, "Autonomous on-animal sensors in sheep research: A systematic review," *Computers and Electronics in Agriculture*, vol. 150, pp. 245–256, 2018.

[5] L. Riaboff, L. Shalloo, A. F. Smeaton, S. Couvreur, A. Madouasse, and M. T. Keane, "Predicting livestock behaviour using accelerometers: A systematic review of processing techniques for ruminant behaviour prediction from raw accelerometer data," *Computers and Electronics in Agriculture*, vol. 192, p. 106610, 2022.

[6] J. Barwick, D. W. Lamb, R. Dobos, M. Welch, and M. Trotter, "Categorising sheep activity using a tri-axial accelerometer," *Computers and Electronics in Agriculture*, vol. 145, pp. 289–297, 2018.

[7] E. S. Fogarty, D. L. Swain, G. M. Cronin, L. E. Moraes, D. W. Bailey, and M. Trotter, "Developing a simulated online model that integrates gnss, accelerometer and weather data to detect parturition events in grazing sheep: a machine learning approach," *Animals*, vol. 11, no. 2, p. 303, 2021.

[8] S. P. Le Roux, J. Marias, R. Wolhuter, and T. Niesler, "Animal-borne behaviour classification for sheep (dohne merino) and rhinoceros (ceratotherium simum and diceros bicornis)," *Animal Biotelemetry*, vol. 5, pp. 1–13, 2017.

[9] J. A. Vázquez-Diosdado, V. Paul, K. A. Ellis, D. Coates, R. Loomba, and J. Kaler, "A combined offline and online algorithm for real-time and long-term classification of sheep behaviour: Novel approach for precision livestock farming," *Sensors*, vol. 19, no. 14, p. 3201, 2019.

[10] L. Nóbrega, P. Gonçalves, M. Antunes, and D. Corujo, "Assessing sheep behavior through low-power microcontrollers in smart agriculture scenarios," *Computers and Electronics in Agriculture*, vol. 173, p. 105444, 2020.

[11] V. Campiotti, N. Finozzi, J. Irazoqui, V. Cabrera, R. Ungerfeld, and J. Oreggioni, "Wearable device to monitor sheep behavior," *IEEE Embedded Systems Letters*, vol. 15, no. 2, pp. 89–92, 2023.

[12] N. Acosta, N. Barreto, P. Caitano, R. Marichal, M. Pedemonte, and J. Oreggioni, "Research platform for cattle virtual fences," in *2020 IEEE International Conference on Industrial Technology (ICIT)*. IEEE, 2020, pp. 797–802.

[13] N. Barreto, J. Oreggioni, and L. Steinfeld, "Energy harvesting and storage solutions for low-power iot devices in livestock industry," in *2024 IEEE 15th Latin America Symposium on Circuits and Systems (LASCAS)*, 2024, pp. 1–5.

[14] Nordic Semiconductor, "Asset tracker v2 application," https://developer.nordicsemi.com/nRF_Connect_SDK/doc/latest/nrf/applications/asset_tracker_v2/README.html, 2019, last accessed 25 May 2023.

[15] J. W. Kamminga, H. C. Bisby, D. V. Le, N. Meratnia, and P. J. Havinga, "Generic online animal activity recognition on collar tags," in *Proceedings of the 2017 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2017 ACM International Symposium on Wearable Computers*, 2017, pp. 597–606.

[16] N. Kleanthous, A. Hussain, W. Khan, J. Sneddon, and A. Mason, "Feature extraction and random forest to identify sheep behavior from accelerometer data," in *Intelligent Computing Methodologies: 16th International Conference, ICIC 2020, Bari, Italy, October 2–5, 2020, Proceedings, Part III 16*. Springer, 2020, pp. 408–419.

[17] J. Nordby, M. Cooke, and A. Horvath, "Emlearn: Machine Learning inference engine for Microcontrollers and Embedded Devices," Mar. 2019. [Online]. Available: https://doi.org/10.5281/zenodo.2589394

[18] E. Price, J. Langford, T. W. Fawcett, A. J. Wilson, and D. P. Croft, "Classifying the posture and activity of ewes and lambs using accelerometers and machine learning on a commercial flock," *Applied Animal Behaviour Science*, vol. 251, p. 105630, 2022.

[19] O. Khalil, "Maximizing battery lifetime in cellular IoT: An analysis of eDRX, PSM, and AS-RAI," *Nordic Semiconductor Blog*, Aug 2023.