AI-based workflow for recording animal-plant interactions data with camera traps

Pablo Villalva & Pedro Jordano

with the collaboration of: **Francisco Rodríguez-Sánchez**, **Eva Moracho**, **Jorge Isla**, **Elena Quintero** and additional help with field work from **Gemma Calvo** and **Pablo Homet**.

Contact address:

Integrative Ecology Group

Estación Biológica de Doñana, EBD-CSIC

Avda. Americo Vespucio 26

E-41092 Sevilla, Spain

Voice: +34 95 4466700

fax: +34 95 4621125

E-mail: jordano@ebd.csic.es, villalva@ebd.csic.es

http://pjordanolab.ebd.csic.es

A separate GitHub repository includes detailed scripts for this protocols.

https://github.com/Cyanopica/Ecological-interactions-camtrap-protocol

Version 1.1. May 2023

This study was funded by MICINN through European Regional Development Fund [SUMHAL, LIFEWATCH-2019-09-CSIC-4, POPE 2014-2020]], CSIC Interdisciplinary Thematic Platform (PTI) Síntesis de Datos de Ecosistemas y Biodiversidad (PTI-ECOBIODIV).

Index

A. Introduction

This is a summary of our protocol for processing large numbers of video-recorded file sets generated during camera trap surveys for monitoring animal-plant interactions (i.e visits of frugivores to fruiting plants). We combine a camera trap field protocol (preprocessing) with several tools for time-saving processing (and post processing). This workflow enables us to manage camera traps in the field, transfer and storing data to the lab and post-process large video files batches, reducing effort and time in the database compilation. The method combines a field protocol based on camera trap operation and data standards together with Artificial Intelligence for image recognition and a viewer program to view and tag images. The main objective is to build an accurate and fully annotated dataset for animal-plant interactions records while time effort is minimised.

A.1 ***Study area***

Doñana National Park is a unique protected area located in Huelva, SW Spain. It is characterized by a large variety of terrestrial and aquatic ecosystems ranging from pine and cork oak forests to scrublands, grassland, sand dunes, and marshlands. The rich diversity of ecosystems is the main reason for harbouring a great biodiversity, evidenced by more than 1300 plant species (170 of which are endemic), over 300 bird species and 50 mammal species, including emblematic species such as the imperial eagle and Iberian lynx (Green et al. 2016).

Plant species have a crucial role in maintaining the mentioned animal diversity through bottom up processes, while herbivores control vegetation through top down regulation. However animals not only maintain plant communities by freely eating upon them, but offer an important ecosystem service as dispersal vectors for a variety of plant species. The offer of fleshy fruit for seed dispersion is a common evolutionary strategy in Doñana in the so called endozoochorous dispersal syndrome. In fact 56 % of woody species in Spanish Mediterranean scrublands are adapted to endozoochorous seed dispersal by vertebrates (CITAS: Herrera 84, Jordano 84), becoming a central process in plant populations where natural regeneration strongly depends upon seed dissemination by animals (Jordano, 2000).

Frugivorous birds and mammals visiting fleshy fruit trees and scrubs in Doñana may behave as seed dispersers, pulp consumers, or seed predators. Even though the mutualistic-antagonistic continuum is the rule, some species such as *Sus scrofa* or *Chloris chloris* are prominently seed predators, while others such as *Vulpes vulpes* and *Erithacus rubecula* can be considered fully legitimate seed dispersers.

The local plant populations studied here are located in the Doñana Biological Reserve, a core area within Doñana National Park. In this area fleshy fruit species are spread throughout the landscape generally occurring in isolated patches, some species may be more continuous across the landscape such as *Olea europaea* and some are associated with ecotone areas, such as *Rubus ulmifolius* while others are more associated to a specific type of soil such as *Corema album* that occupies coastal dunes.

A.2 *Study species*

The study species correspond to the most relevant plant species producing fleshy fruits in Doñana National Park (SW Spain). The selected twelve species are listed and briefly described below:

- ***Corema album*** is a shrub endemic from the west coast of the Iberian Peninsula, and the Azores Islands growing mainly on sand dunes. It is considered an endangered species (Ortiz *et al*., 1998) that has experienced a notable decline in size and number of populations. It is a dioecious shrub that rarely exceeds 1 m height. It is wind pollinated and its fruits are quasi-spherical, white drupes. The pulp has a high water and sugar content (Calviño-Cancela, 2003; Herrera, 1987). Ripe fruits are available in summer and early autumn, with the highest availability occurring in July - August.

- ***Juniperus phoenicea*** is a gymnosperm shrub inhabiting coastal dunes and rocky habitats in the western Mediterranean and Macronesian archipelagos. It is an anemophilous species characterized by masting cycles of fleshy cone (galbule) production (Jordano, 1993; Roques et al., 1984). Brown-red galbules are consumed and dispersed by several thrush species and medium-sized generalist mammals. The fruiting period is in autumn, spanning from October - December.

- ***Juniperus oxycedrus subsp. macrocarpa*** is a gymnosperm growing in the northern mediterranean basin and northern Africa. It is a dioecious and anemophilous species that produces berry-like spherical, fleshy cones. Unripe cones are green, ripening in 18 months when they turn to orange-red with a variable pink waxy coating that are available in winter and early spring, between December - March.

- ***Rubus ulmifolius*** is a rosaceous vine-shrub native across Western Europe and naturalized in N. America, NW and S. Africa and Australasia. It is unique among subgenus *Rubus* in displaying sexual entomophilous reproduction while all others are facultative apomicts. The fruit is a polidrupe, dark purple, almost black with a summer fruiting season that expands from July - August.

- ***Pyrus bourgaeana*** is a rosaceous tree, widely distributed across the southern Iberian Peninsula and northern Morrocco but with fragmented populations that occur at low densities and small patches. It is pollinated by insects and its fruits are non-dehiscent globose pomes with green or brown skin inconspicuous to birds and a styptic pulp. The autumn fruiting season expands from September - November.

- ***Smilax aspera*** is a perennial, evergreen climber with a flexible and delicate stem, with sharp thorns from the family Smilacaceae, widespread in Africa, Europe and temperate and tropical Asia. The entomophilous flowers are very fragrant, and the fruits are globose berries, gathered in clusters that are initially red, later turn black and have an extended phenology from October - March.

- ***Myrtus communis*** is an evergreen shrub or small tree from the family of Myrtaceae native to southern Europe, North Africa, Asia and Macaronesia. Reproduction is entomophilous and the fruit is a blue-colored fleshy berry when ripe. Fruiting occurs in late winter from December - February.
- ***Arbutus unedo*** is an evergreen shrub or small tree in the family Ericaceae, native to the Mediterranean region W Europe. The entomophilous hermaphodite flowers are white and hang from a reddish panicle. The fruit is a red, spherical berry with a rough surface. Fruit ripening occurs in about 12 months after anthesis, at the same time as the next flowering but with a fast ripening between December - January.
- ***Olea europaea, var. sylvestris*** is a species of evergreen tree or shrub native to Mediterranean Europe, Asia, and Africa in the family Oleaceae. Pollination is anemophilous and the fruits are small drupes black when ripe, thinner-fleshed and smaller in plants of this wild subspecies than in orchard cultivars of olive trees. Fruiting (fruit ripening) occurs from November - December.
- ***Asparagus aphyllus*** is a dioecious species, climbing plant in the family Asparagaceae native from the Mediterranean basin. Flowers are pollinated by insects and fruits are globose dark-green berries that torn blackish when ripe. Fruiting occurs in autumn or early winter, between October - December.
- ***Rubia peregrina*** is a herbaceous perennial plant species belonging to the family Rubiaceae mainly present in the Mediterranean basin, Great Britain and North Africa. The hermaphroditic flowers are pollinated by insects. The fruits are fleshy green berries, black when ripe which occurs in late summer and early autumn between September - November.
- ***Osyris lanceolata*** is a a hemiparasitic evergreen shrub in the family of Santalaceae found in low densities across Africa and the southern half of the Iberian Peninsula and Macaronesia. They are self-fertile, so the species produces fertile seeds prolifically. The fruit is a single spheroid colorful drupe, progressing from greenish tones to bright orange as they ripen. The species produces ripe fruits almost continuously, and most individuals have fruiting periods virtually encompassing the entire year. The peak of the fruiting period of individual plants may occur in almost any month of the year.

A.3 *General approach*

Our approach for monitoring animal-plant interactions in natural habitats involves the strategic placement of camera traps, aimed at specific plant species, referred to as focal plants. These cameras are set in video mode, providing us with valuable insights for species identification and behavior, as well as an accurate quantification of fruit consumption.

Our video-based monitoring method offers several advantages over traditional camera-trap techniques based in still pictures. By capturing movement, we are able to identify animal species and their behavior, allowing us to gain a deeper understanding of the complex interactions between animal and plant species. Additionally, by accurately measuring fruit consumption at least in some video recordings, we can determine consumption rates and fruit feeding behavior,

and gain insights into the importance of different plant species for wildlife in the ecosystem as well as for the ecosystem service provided by animals for plant dispersal.

However, camera-trap monitoring can also present challenges, especially in environments with high wind levels. In these conditions, incorrect triggering can easily occur through the movement of grasses and tree branches, leading to a large number of empty images. To mitigate this issue, we have developed a protocol that streamlines the process of generating large databases from video recordings and reduces the time and effort required to do so.

This camera-trap monitoring approach provides a powerful tool for studying plant-animal interactions in natural habitats. By accurately identifying species and capturing animal behavior, as well as fruit consumption rates, we can gain valuable insights into the functioning of complex ecosystems.

A.4 ***Video data workflow outline***

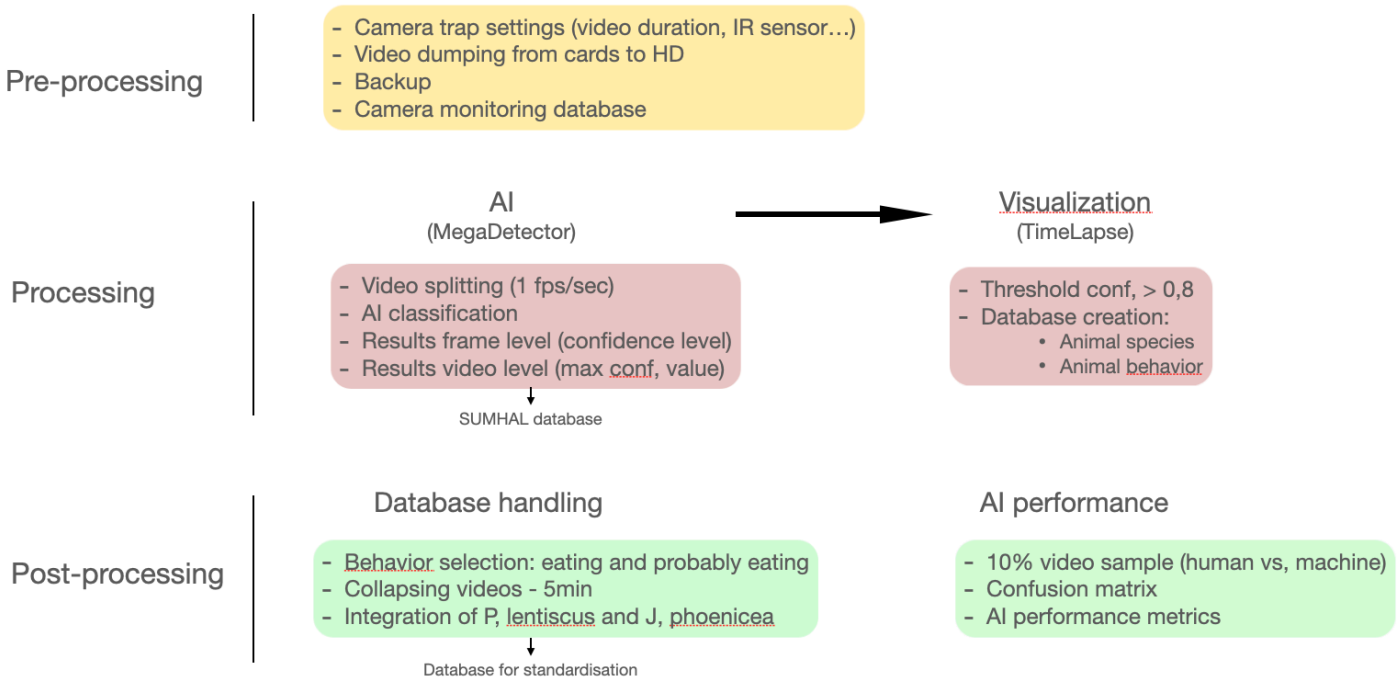

Figure 1. Summary of the protocol streamline.

B. *Pre-processing*

Most likely a camera trap field sampling for ecological interactions involves the simultaneous deployment of multiple cameras, in replicated positions to target different individual plants, throughout the fruiting season of different focal plant species. The cameras are checked at regular intervals, typically weekly, biweekly, or monthly, which can result in a large number of videos with the same name and date.

Effective management of such a large and complex data set is crucial for a successful database creation. To achieve this, a structured field database is required to keep track of the data at every stage of the process. For this purpose, we recommend the use of the Camera Trap Data Package (Camtrap DP), a community-developed data exchange format that is under development as a Biodiversity Information Standard (TDWG). This package provides a useful structure for controlling camera-trap data at three levels, from which we will adopt the structure: *deployments*, *revisions*, and *observations*.

Camtrap DP offers a standardized format for organizing camera-trap data, ensuring consistency and reducing the risk of errors or inconsistencies. This structure includes all necessary information, such as camera settings, deployment locations, and video file names, allowing for easy management and analysis of the data. A template for the Camtrap DP is available for use, and you can find a template for our ad-hoc structure in the GitHub repository https://github.com/Cyanopica/Ecological-interactions-camtrap-protocol/tree/main/Preprocess or see the following descriptions.

The main data is structured in three related plain text files (.csv) as follows:

File	Description
deployments.csv	Table with camera trap deployments.
video.csv	Table with media files captured by the camera traps.
observations.csv	Table with observations based on media files (after viewing in Timelapse)

B.1 ***Database structure***

B.1.1 ***Deployments***

Table with camera trap deployments. Includes `deploymentID` (focal species acronym + `cameraID`), `Location` and camera `Setup` information for each camera.

Name	Definition	Type
`Deployment_ID`	Unique identifier of the deployment. Name of the focal species followed by the individual number low dash camera number. Example: `Aune003_58`	string
`Location`	Name given to the deployment location. Survey area.	string
`Longitude`	Longitude of the deployment location in decimal degrees, using the WGS84 datum.	number
`Latitude`	Latitude of the deployment location in decimal degrees, using the WGS84 datum.	number
`Start`	Date and time at which the deployment was started. Formatted as an ISO 8601 string with timezone designator (YYYY-MM-DDThh:mm:ss±hh:mm).	datetime

End	Date and time at which the deployment was ended. Formatted as an ISO 8601 string with timezone designator (YYYY-MM-DDThh:mm:ss±hh:mm).	datetime
Days	Number of days the deployment was set in the field. End_date - Start_date	number
Setup_by	Name(s) or unique identifier of the person that deployed the camera.	string
Camera_ID	Unique identifier of the camera used for the deployment (could be the serial number but also a simple number)	string
Camera_model	Manufacturer and model of the camera.	string
Comments	Comments or notes about the deployment.	string

B.1.2 *Videos*

Table with video files captured by camera traps. Associated with `deployments` (by `deploymentID`) and organised in revisions (`revision_ID`). Includes `Timestamp_Issues` and `File_path`.

Name	Definition	Type
Deployment_ID	Unique identifier of the deployment the media file belongs to. Foreign key to Deployments.Deployment_ID.	string
Revision_ID	Unique identifier of the revision the media file belongs to. Revisions contain one or more media files (e.g. a single image or video or a sequence of successive images or videos). Example: Rev_01	string
Videos	Number of media files.	number
First_video	Datetime for the first video in the revision sequence.	datetime
Last_video	Datetime for the last video in the revision sequence.	datetime
Days	Number of days that the deployment was set in the field during the current revision.	number
Setup_date	Date at which the camera was set on in the current revision.	datetime
Revision_date	Date at which the camera was set off in the current revision.	datetime
Functioning_days	Number of days where the camera was functioning. Revision_date - Setup_date	number
Battery	Percentage of battery in the revision datetime.	string
Timestamp_Issues	True if timestamp in the media have been detected.	boolean
File_path	URL or relative path to the media files, respectively for externally hosted files.	string
Favourite	True if it contains videos tagged as favorite.	boolean

Comments	Comments or notes about the revision.	string

B.1.3 *Observations*

Table with video files results from visualization. Associated with deployments (deploymentID) and with revisions (revision_ID) through Videos.file_path. Note that this data will be generated directly with Timelapse software as explained below.

Name	Definition	Type
File	Name of the video file. If more than one video files use concatenate separated by ",".	string
Path	URL or Relative path to the first Obs.File, respectively for externally hosted files.	string
Plant_sp	Name of the focal plant species.	string
Plant_ID	Unique identifier of the plant individual. Example: Sasp003	string
DateTime	Date and time at which the video started. Formatted as an ISO 8601 string with timezone designator (YYYY-MM-DDThh:mm:ss±hh:mm).	datetime
Sp1	Latin binomial for the principal animal species recorded in the video. Example: Athene noctua.	string
Behaviour		string
Sp2	Latin binomial for a secondary animal species recorded in the video.	string
Behaviour_Sp2		string
Sp3	Latin binomial for a third animal species recorded in the video.	string
Behaviour_Sp3		string
n_cam	Number of cameras set in the same focus individual.	number
Videos	Number of videos recorded in the current revision. Vid.Videos	number
Days	Number of days for the camera recording in the current revision. Sampling effort for a video in a given revision.	number
Video_events	When videos are collapsed in time (i.e 5 minutes) number of videos that where collapsed. It will be coincident with the number of different Obs.File names.	number
Duration	Number of second for the duration of the event. Could be the sum of durations from different Obs.video_events.	number
TimeStamp_Issue	True if timestamp in the video have been detected.	Boolean
Long	Longitude of the deployment location in decimal degrees, using the WGS84 datum.	number
Lat	Latitude of the deployment location in decimal degrees, using the WGS84 datum.	number

`Favourite`	True if it contains videos tagged as favorite.	boolean
`Observations`	Comments or notes about the revision.	string

B. 2 ***Camera trap settings***

In this section, we provide recommendations for conducting effective camera-trap field surveys and optimizing camera settings. As previously mentioned, behavioral observations require the use of video mode. To optimize memory cards/batteries and post-processing workflow, it is recommended to set the video length consistently to 10 s.

The motion trigger should be set with a delay of 1 s or even less, if possible, to maximize the recording of animal behavior. However, it is important to note that the motion sensor of some cameras may have a wider angle of sensitivity than the camera lens, which may result in the beginning of the video being empty if the animal is not yet within view. The sensitivity of the camera should be managed with caution, as it varies depending on the camera model/make. To ensure optimal detection, preliminary trials to set sensitivity correctly are strongly recommended.

The placement of the camera is critical to achieving effective detection, and it is recommended to place the camera at shoulder height of the target species (Palencia et al. 2021). In practice, for our scheme this may not always be possible due to the diverse range of species, including birds and mammals of different sizes and behaviors, that may be present. In such cases, multiple cameras may be necessary to capture the full range of visitors. To avoid duplication of data, it is extremely important to ensure that the field-of-view (FOV) of each camera does not overlap.

To optimize field workload and avoid bias in the final dataset, it is recommended to use a single camera model/make when possible. If a set of different makes is used, a systematic rotation of the deployment/camera may be a solution, but this will make the rest of the workflow more challenging. All cameras should be clearly identified with a unique identifier, such as the camera serial number or a 2-digit code for simplicity, and should be set with the same parameters.

For Browning Dark OPS cameras, the following setup parameters are recommended:

Table 1. Recommendations for camera trap settings. Note that this settings correspond to Browning Dark Ops Trail cams.

Parameter	**Recomendation**
Date/Time	Set correct Date and time and check for Timestamp Issues in each camera/revision. Time in Universal Time (UTC) is suggested.
Capture Mode	Video
Capture Delay	1 second
Cap start/end	24 hours (if less correct sampling effort in database)
Video quality	Ultra (1600 x 900 @ 24 fps)
Video lenght	10 seconds

Smart IR	OFF - video clip is not allowed to keep recording if still detects movement
Info strip	ON - mandatory for revising Timestamp Issues
SD management	OFF - erase of older pictures is not allowed
Motion detect	Pwr safe

Key factors - recommendations for data field surveys.

By following these key factors and recommendations, field surveys can be conducted with increased accuracy and efficiency, resulting in high-quality camera trap data for capturing animal-plant interactions.

1. Model/makes:

It is recommended to use the same camera trap model throughout the study to avoid the potential bias in the final dataset. Using a single model ensures that the probability of detection is consistent, as detection can vary between different models and makes (Palencia et al. 2022)

2. Sensitivity:

The sensitivity of the camera traps should be set cautiously, taking into account the surrounding environment and the plant species being studied. In open areas with minimal movement interference, a higher sensitivity can be set, whereas in more cluttered areas with higher levels of movement, a lower sensitivity setting is recommended.

3. Height:

There is a higher probability of detection when camera traps are set at shoulder height of the target species. It is recommended to place more than one camera at different heights, avoiding overlap of the field of view (FOV) to prevent data duplication.

4. Trigger speed:

Minimizing the activation time before the trigger will maximize the recording of animal behavior, so it is recommended to set the trigger speed as low as possible, ideally at 1 second or less.

5. Distance from focus plant:

The cameras should be placed between the minimum focal distance (as set for the specific camera model) and no more than a few meters beyond, in order to minimize wind-triggered files and ensure small animal identification.

6. Sampling effort control:

It is important to systematically record a file at the beginning and end of each revision for each camera to accurately control the sampling effort. This helps to ensure that the dataset is comprehensive and free of any sampling bias. You can read video files from the parent directory and collect sampling effort information automatically with an ad-hoc code contained in the repository: https://github.com/Cyanopica/Ecological-interactions-camtrap-protocol/blob/main/Preprocess/sampling_effort_extraction.R

B.3 *Video dumping and data storage:*

For the efficient temporary storage of camera trap data, we recommend the use of SD cards with a capacity of 32 GB. It is important to choose high-quality memory cards, and check the maximum capacity for the camera make, as some cameras have been known to experience storage issues with larger cards. Most cameras are compatible with standard class 10 or higher SD cards, up to 32 GB. For higher storage capacity, SDXC cards should be used. The graph below illustrates the number of videos that can be stored on SD cards with capacities ranging from 8-32 GB.

Table 2. Storage capacity for SD cards

Videos (10 second)	**8GB**	**16GB**	**32GB**
Ultra resolution	120	240	480
High resolution	110	220	330

Upon collection, the data will be transferred to a hard drive (HD) in the lab. To minimize the time required for data dumping, we recommend using high-transfer speed cards (i.e., 130 MB/s) and a high-performance storage system with fast hard drives (at least 7200 RPM) and high bandwidth connectors (such as Thunderbolt). Determining the necessary storage capacity can be a challenge, as it depends on the amount of data generated during processing and post-processing. In our own study, which included 12 focal species across two fruiting seasons, we used 4 G-Raids with a total capacity of 60 TB. It is also essential to create a backup of the data stored in the HD to protect against loss or corruption. We suggest avoiding HD mirroring in order to prevent systematic duplication of corrupted or modified files, and to create a backup after data dumping for each revision.

Key factors - recommendations for data storage

1. Hardware Requirements:

Ensure the use of high-quality SD cards with appropriate storage capacity (32 GB or higher). Use the right type of memory card (i.e. SD, SDXC) that is compatible with your camera trap. Consider the battery life and capacity of your camera traps and ensure that they are fully charged and replaced as needed.

2. Backup Strategy:

Regular manual backups should be performed to ensure data safety and prevent loss. Avoid using mirroring as a backup method, as it can result in duplication of corrupted or modified files.

3. Data Dumping:

Use an integrated card reader on the motherboard for quick data transfer from the SD card to the Hard Drive (HD). Consider the speed of the SD card (i.e. 130 MB/s or higher) and the performance of the storage system to minimize data dumping time.

C. Processing

The processing of large volumes of video data collected through camera trap surveys can be challenging and time-consuming, making manual review an expensive and impractical task. Artificial intelligence (AI) technology can be an effective solution to this problem by automating data collection. The use of AI for species recognition and identification has shown promising results, with studies reporting high classification accuracy measures. For example, Norouzzadeh et al. (2018) reported accuracy of over 99%, while Schneider et al. (2019) reported 93% accuracy for one recognition algorithm. Tabak et al. (2019) achieved 97.6% accuracy in species identification.

However, it is important to be cautious when interpreting these results, especially if the training dataset is not diverse enough. The training process for AI models requires a large amount of data that is specific to the ecosystem being studied. This is particularly challenging for projects that aim to identify a wide variety of species in regions without pre-existing trained models, as is the case with most plant-animal interaction surveys. In such projects, obtaining sufficient funding for appropriate model training may be beyond the possibilities of conservation funding.

C.1 ***Eliminating empty images***

However, a simpler recognition result can still provide a big win in efficiently analyzing images.

Differentiating between empty and non-empty images in camera trap data is critical for efficient analysis. The majority of video files obtained through camera trapping often contain no significant information, typically due to accidental motion triggers caused by wind effects. The elimination of these empty videos can significantly reduce the time required for review.

To achieve this objective, we use MegaDetector (MD), a state-of-the-art object detection model that is capable of identifying animals, people, and vehicles in camera trap images. The model has demonstrated high accuracy in various species, ecosystems, and scenarios (Norouzzadeh et al., 2018; Tabak et al., 2019). While MD has been designed to analyze still images, this protocol represents, to our knowledge, the first attempt to apply object recognition to large camera trap video sets.

C.2 ***AI implementation***

Our objective is to apply a trained AI model to classify camera trap videos and differentiate between those that contain animals and those that do not. To accomplish this, we must first convert the video clips into individual still frames and run the object detection model (MegaDetector or hereafter MD) on each frame to estimate the probability (confidence score) that it contains an animal. The outputs from the model for each frame will constitute the frame-level results. Next, we will extract the maximum probability value from the set of frames associated with each video to determine the probability that the video contains an animal, which will result in the video-level results. Finally, we will set a confidence threshold to select only those videos with high probabilities of containing animals of interest, thereby streamlining the visualization process.

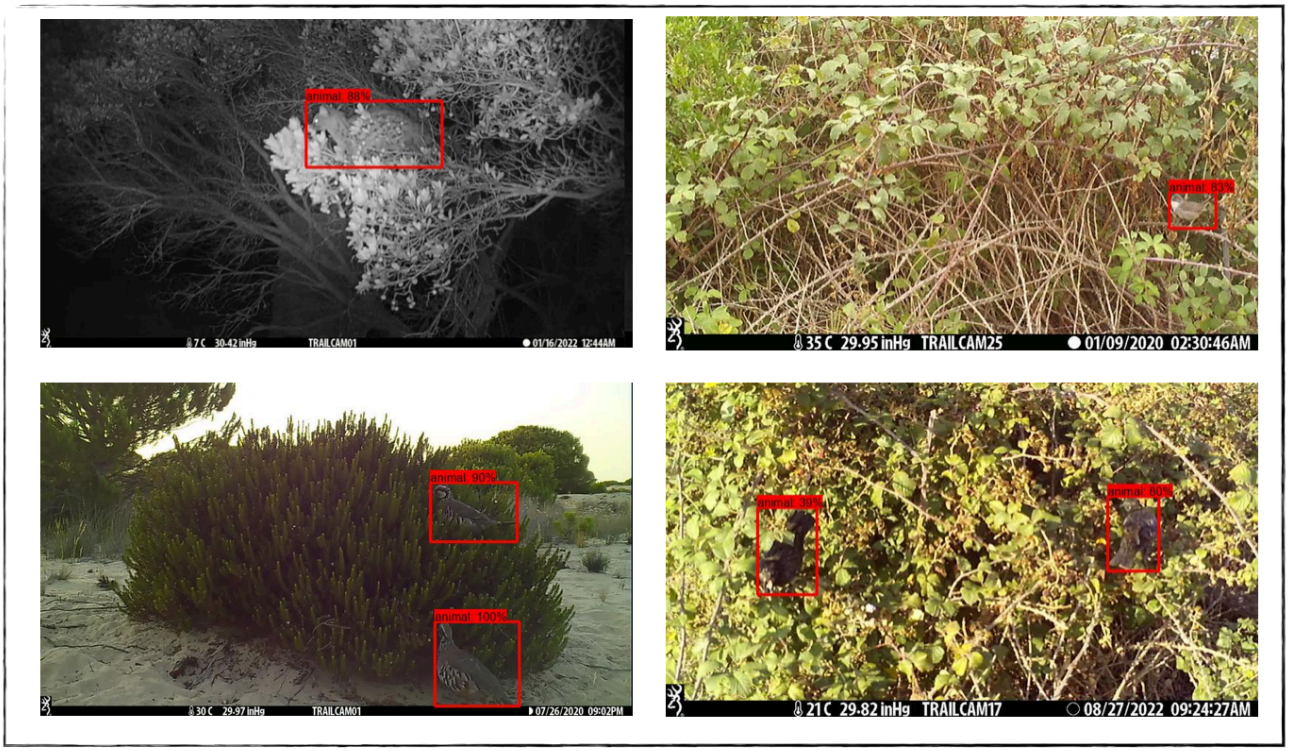

Figure 2. Preview of detection bounding boxes with confidence level for four still images

C.3 ***Video splitting***

The implementation of our AI model requires the pre-processing step of splitting video files into individual frames. To streamline this process, we created an R script that automates the video splitting process. The script uses the av library for R and the construction of a command line script for video listing and a custom function for splitting video batches. The code for the R script is available in publicly in https://github.com/Cyanopica/Ecological-interactions-camtrap-protocol.

It is important to maintain the structural consistency between the split images and the original video files, including the preservation of the relative path, which is crucial for the subsequent steps of the process. Although video splitting can be time-consuming, it significantly reduces the file size, particularly if the data is intended to be processed using enhanced GPU capabilities.

However, it is worth noting that the script has limitations and may crash if the moov.atom metadata is missing, which typically occurs when videos end abruptly while recording.

C.4 ***Running the model***

There are two primary methods for executing the MD object detection model:

1. Running the model on a local computer: The model is freely accessible as a downloadable option on https://github.com/microsoft/CameraTraps/blob/main/megadetector.md#downloading-the-model. The ease of executing the model locally depends on the quantity of images to process, the specifications of the computer hardware, and the user's proficiency with Python programming language. It is recommended to run the model on a GPU-enabled computer for improved performance. For instance, processing a few thousand images per week can be accomplished with a typical laptop, but processing 20 million images as efficiently as possible

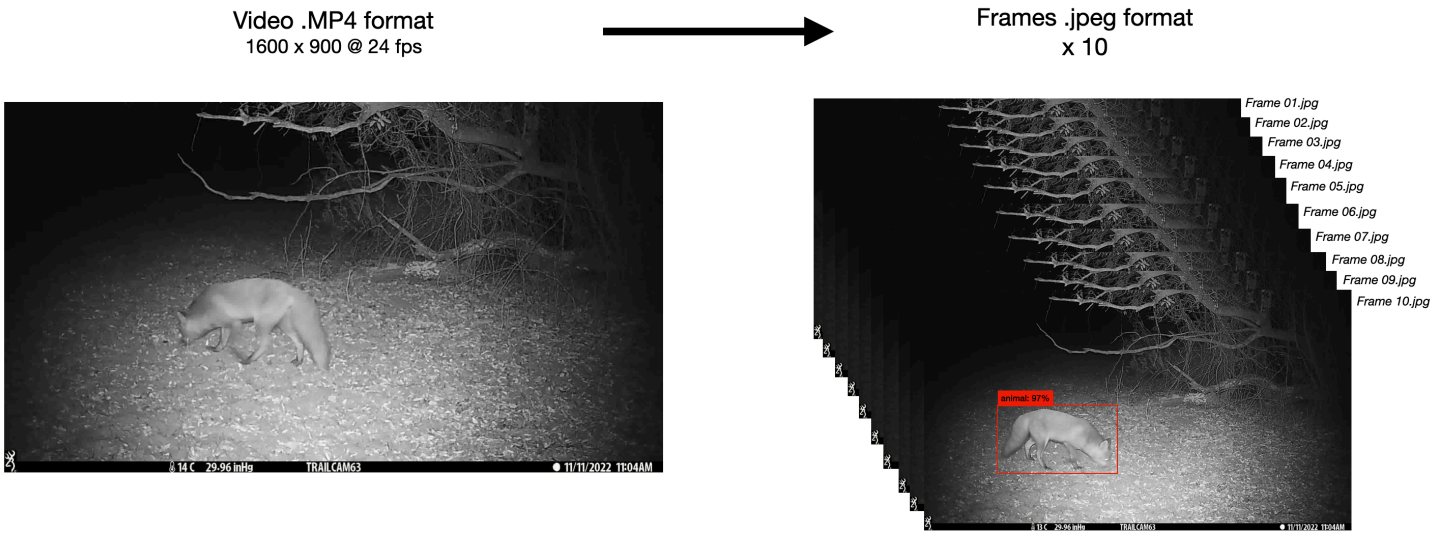

Figure 3. Video splitting scheme. Ten second duration videos in MP4 or AVI format (left) is converted in 10 still frames in jpeg format (right).

would require at least one GPU. Information regarding the execution of MD on a local computer can be found in the MD GitHub repository.

2. Submitting the images to the MD staff for model execution: This option is ideal for high-volume users who require access to high-performance processors not readily available on personal computers. The images can be submitted either via a physical hard drive or uploaded to the cloud with an sftp protocol, depending on the location of the user and its internet connection speed.

It is important to note that high-performance computers are typically required for efficient recognition, as the process can be computationally demanding. Additionally, regardless of whether the model is executed locally or through the MD staff, it is advisable to run the model on a few thousand images as a preliminary check to ensure its appropriate functionality in the target dataset.

C.5 ***Model output***

After executing the model, a JSON file will be produced as the output. This standard data interchange format represents information in a text-based format and is capable of preserving the inherent structure of the input data. As a result, it is able to maintain a record of each video file processed through the frame and video-level analysis. The output will consist of a frame-level analysis, which includes the probability, or confidence level, at which the model detects the presence of an animal, and a video-level analysis that provides the probability of each video containing an animal. The frame-level data is used to construct the video-level data through aggregation, as described in the section on AI implementation.

Table 3. JSON structure for a MD output at video level. Categories, version information and results from image detection are shown. Note that file shows the relative path to the video file, $detections shows confidence level for each frame as a list and $max_detection_conf shows the maximum value for $detections.

Categories	Example
$detection_categories	
[1] animal	"animal"
[2] person	“person"
[3] vehicle	“vehicle"
$info	
$detection_completion_time	2023-01-10 02:26:23
$format_version info.detector	1.2 md_v5a.0.0.pt
$detector_metadata	v5a.0.0
$images	
$file	Rubus/Rev11_20220831/Rulm015_15/IMG_0035.AVI
$detections	1, 0.888, 0.256, 0.18, 0.155, 0.155
$max_detection_conf	0.888

C.6 ***Confidence selection***

After the video-level output is generated, it is imperative to incorporate it into the workflow in an efficient manner. One option is to load the AI results into a Timelapse software, but this approach may pose challenges when dealing with large video-level results. As an alternative, a confidence threshold can be employed to select only those videos that meet the required criteria, while ignoring the others. This approach, while not very flexible, enables stepwise selection of confidence intervals in successive rounds. For instance, it is recommended to start with a higher confidence threshold (e.g. above 0.8) and then move on to lower confidence ranges (e.g. from 0.7 to 0.8) in later reviews. However, it is important to note that different versions of MD can have varying confidence profiles, so the optimal threshold values may vary greatly. For example, in our dataset, the confidence threshold was set to 0.8 when using MD version 4, while it was set to 0.15 for MD version 5.

In order to select and manipulate files, the *file.copy* function from base R was used as a low-level interface to the computer's file system to copy and paste the selected videos. A script, which includes the creation of destination folders, selection of file lists, and copy and paste function, can be accessed via https://github.com/Cyanopica/Ecological-interactions-camtrap-protocol/tree/main/Process.

C.7 ***Visualisation and database creation***

We used Timelapse an open-source tool for reviewing camera trap images and videos. It boasts good support for videos, and a multitude of interface tools for accelerating the visual analysis and encoding. The program automatically extracts file information and metadata, presenting a custom interface for data entry and supporting visual searches. All the data will be saved to a CSV file. If you need help using Timelapse, you can find the reference guide here.

To get started, you'll need to create a template specifically for your project using the template manager. This will allow you to gather data from each image. For gathering animal-plant interactions, the template should match the structure of the "Observations.csv" file explained in the preprocessing section, as it will be automatically generated once you've finished viewing and annotating the video set.

Once your template is set up, load the video files into Timelapse, making sure to preserve the relative path structure. Then, visualize the video set and make note of at least the animal species and its behavior.

Here's a list of example behaviors that we recorded for our dataset:

1. Eating.
2. Probably eating.
3. Searching for food.
4. Visiting (using plant).
5. Walking (or flying by the image).
6. Others (note in observations).

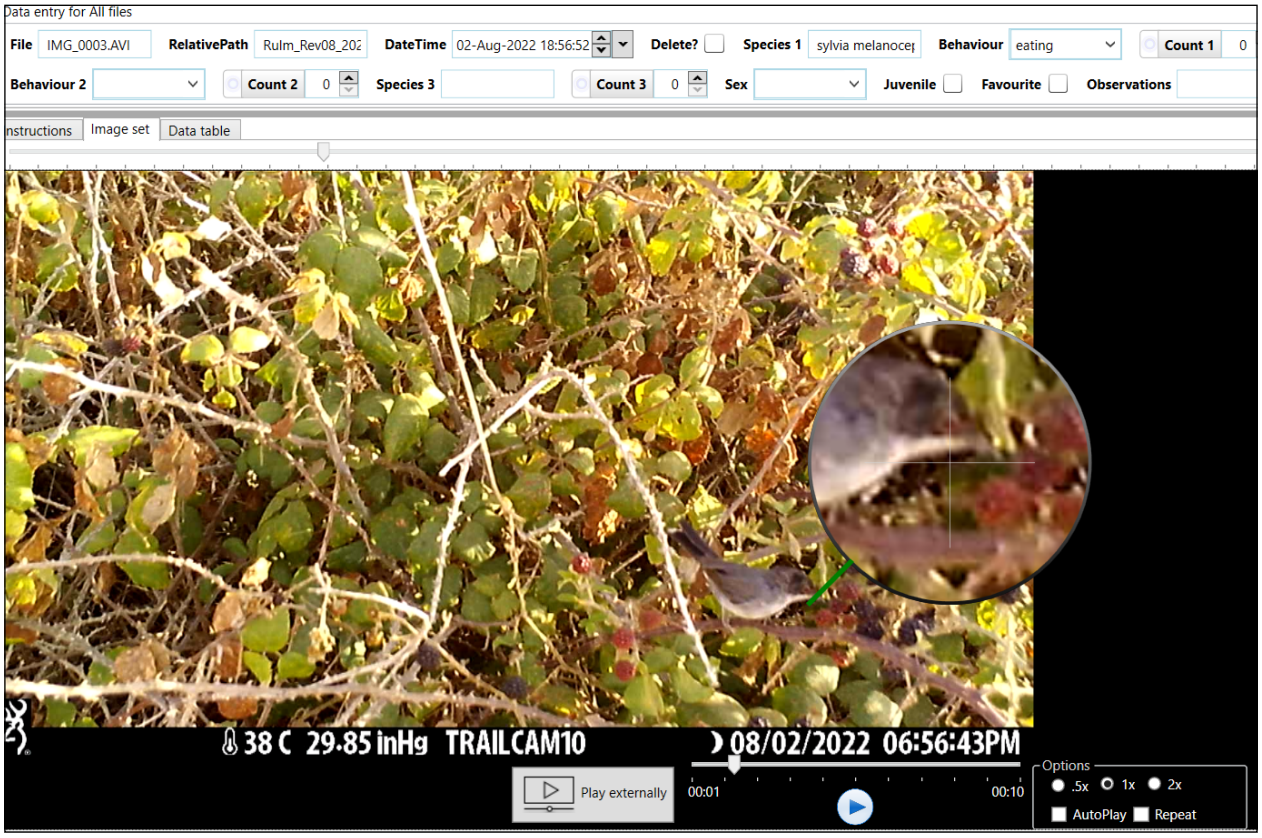

Figure 4. Time-lapse user interface.

D. Post-processing

D.1 ***Database handling***

Once the video-level database is generated, it is necessary to have a clear understanding of the analysis that will be carried out to handle the information. It is important to note that the most refined data for the dataset is at the video level. This means that each entry represents a 10 second duration video (although duration may be longer depending on the camera setup) with different animal species exhibiting different behaviors. First steps for managing this data will require to select the desired behaviors which in our frugivore context should be related to fruit consumption. If a conservative approach is desired, it is possible to select only those videos where the animal was found ingesting the fruit. However, this type of selection is likely to be unrealistic and too strict, actually underestimating the frequency of fruit feeding within a specific visit to the plants. Adding those videos where the animals are probably eating (videos where the animal was not recorded ingesting due to its position or maybe has allegedly been eating out of shot) would be more realistic. However this approach may also under-estimate the real amount of frugivore events as fruit-eating events may not be captured within the camera shot. Including videos where the animals are in a food-seeking attitude under or on top the focal plant is the most realistic approach to record animal-plant interactions for this protocol.

Once behaviors are identified and selected, it is recommended to establish a baseline for defining the events, as if we were to keep the data at the video level, temporal autocorrelation of the data would inevitably add a significant bias to the analysis. One objective criterion for defining an event can be related to time. Assigning a certain duration can help create independent events to minimize this noise. For this protocol and the data generated with this approach, we propose creating independent 5 minute events. You can find the code for summarizing and aggregating data from a larger video-level data frame in https://github.com/Cyanopica/Ecological-interactions-camtrap-protocol/tree/main/Postprocess. The purpose of this code is to group the video data (10 s duration entries) by 5 minute intervals and then calculate summary statistics for each group calculating the sampling effort and maintaining associated data for each collapsed entry. The rartionale is that a visit sequence starts with the arrival of the animal to the plant and ends with its departure. During this visit, which typically extends beyond a 10 s duration, the animal may feed on fruits or not. Only in some instances the camera will record a fruit handling and/or ingestion, and probably even in more rare instances, will record the whole visit duration. Hence, pooling successive videos within 5 min intervals is an adequate way to record a single interaction bout.

D. 2 ***AI performance metrics*** (image ?? - panel with example metrics for all species)

AI for image recognition is not infallible and has many limitations of how your data respond to the trained model. Thus measuring the performance of AI on your data is vital to be aware if you may be missing something important. Recognition performance can be measured in various ways and the best measure depends on the goal. Some measures used to evaluate recognition performance include precision, recall, accuracy, F-score and MCC. Precision measures the

Table 4. Some AI performance metrics used for measuring recognition performance. For a given concussion matrix TP true positive; TN true negative: FP false positive; FN false negative

Measure		Equation
TPR	True positive rate = Sensitivity	TP/(TP+FN)
TNR	True negative rate = Specificity	TN/(FP+TN)
FPR	False positive rate = Fall out	FP/(FP+TN)
FNR	False negative rate = Miss rate	FN/(FN+TP)
PPV	Positive predictive value = Precision	TP/(TP+FP)
NPV	Negative predictive value	TN/(TN+FN)
FDR	False discovery rate	FP/(FP+TP)
FOR	False omisión rate	FN/(FN+TN)
ACC	Accuracy	(TP+TN)/(TP+TN+FP+FN)
ERR	Error rate	(FP+FN)/(TP+TN+FP+FN)
F1score	Harmonic mean between TPR and PPV	(2*TP)/((2*TP)+FP+FN)
MCC	Mathews correlation coefficient	((TP*TN)-(FP*FN))/sqrt((TP+FP)*(TP+FN)*(TN+FP)*TN+FN))

proportion of correct results in the classifications, recall measures the proportion of returned positive results compared to the total true positives, accuracy measures the proportion of correct results (positive or negative), F-score combines precision and recall into an average. We provide access to the R script to help analyzing and plotting this and other performance metrics related to Image recognition.