Automatic Foley Machine

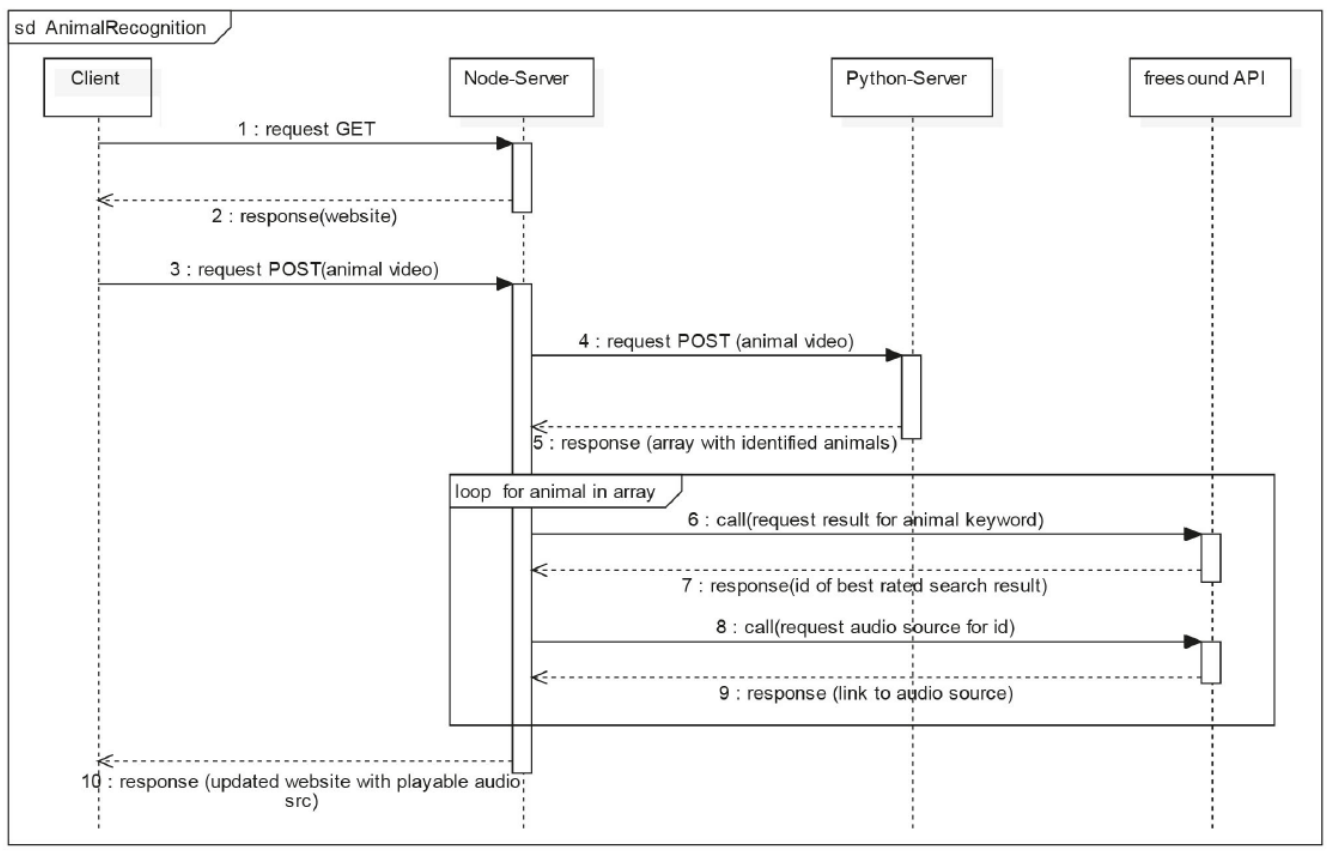
This project uses YOLOv5 to detect objects from a video without audio and then play appropriate sounds for the identified objects when they appear in the video using the freesound.org API. To create an interactive audio-visual experience, sounds can be adjusted with different knobs enabling effects like reverb, filtering and pitching.

The website is served by a node.js web server, which is also responsible for temporarily storing the uploaded video on its hard drive. The file can be uploaded via a dropzone. After receiving the file the web server sends a POST-request to the Python server which then runs the inference on a pre-trained machine learning model based on YOLOv5 for object detection.

After processing and analyzing the individual frames, the Python server sends back a JSON object containing all the detected animals and the respective framenumber to the node.js web server.

With this information, the web server makes multiple requests to the freesound.org API. The response consists of list related to the searched keywords and links to the specific audio files, which are then integrated into the website to create all required audio elements that stream the sound directly from the freesound servers.

The audio data can then be manipulated during playback using the controls, for which we utilize the capabilities of the Web Audio API.


sd AnimalRecognition
Client
Node-Server
Python-Server
freesound API
1 : request GET
2 : response(website)
3 : request POST(animal video)
4 : request POST (animal video)
5 : response (array with identified animals)
loop for animal in array
6 : call(request result for animal keyword)
7 : response(id of best rated search result)
8 : call(request audio source for id)
9 : response (link to audio source)
10 : response (updated website with playable audio src)

Sequencediagram Automatic Foley Machine [1]

FEATURES

- ∗ Object detection with YOLOv5
- ∗ AJAX Upload for dynamic loading of page content
- ∗ File size limit 40 MB
- ∗ Automatic deletion of video files older than one hour
- ∗ Reading video metadata to calculate timestamps
- ∗ Audio manipulation with Web Audio API
- ∗ Freesound.org API requests (max. 60/min, 2000/day)
- ∗ API calls per video limited (max. 60/upload)
- ∗ Deployment on AWS

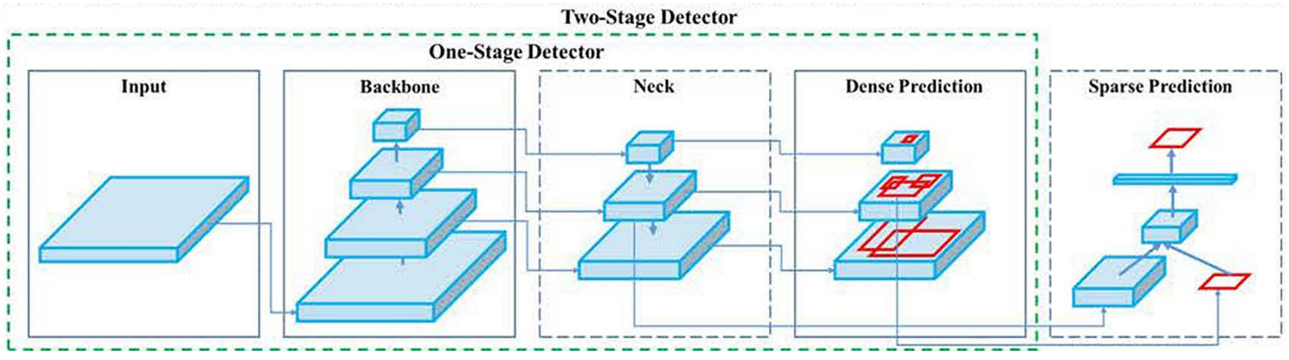
Object Detection

There are two different types of models for object detection: one-stage detectors and two-stage detectors. YOLOv5 is a one-stage detector (green box). It uses a single neural network to predict the object frames and class probabilities directly from the full image in one pass.

Backbone: The backbone is the base classification model upon which the object detection model is built. In YOLOv5, it is a Convolutional Neural Network (CNN) that extracts various features.

Neck: A series of different layers where image features are combined and then sent to the head for prediction.

Head (Dense Prediction): The part of an object detector where the prediction is made. It utilizes the features generated in the neck.



Single-Stage Detector Architecture [2]

Model Adjustment

For our project, we adapted the output of YOLOv5 to our needs. Specifically, we disabled the video output including object frames and made corresponding adjustments in the code so that the detected objects are returned in the form of key-value pairs.

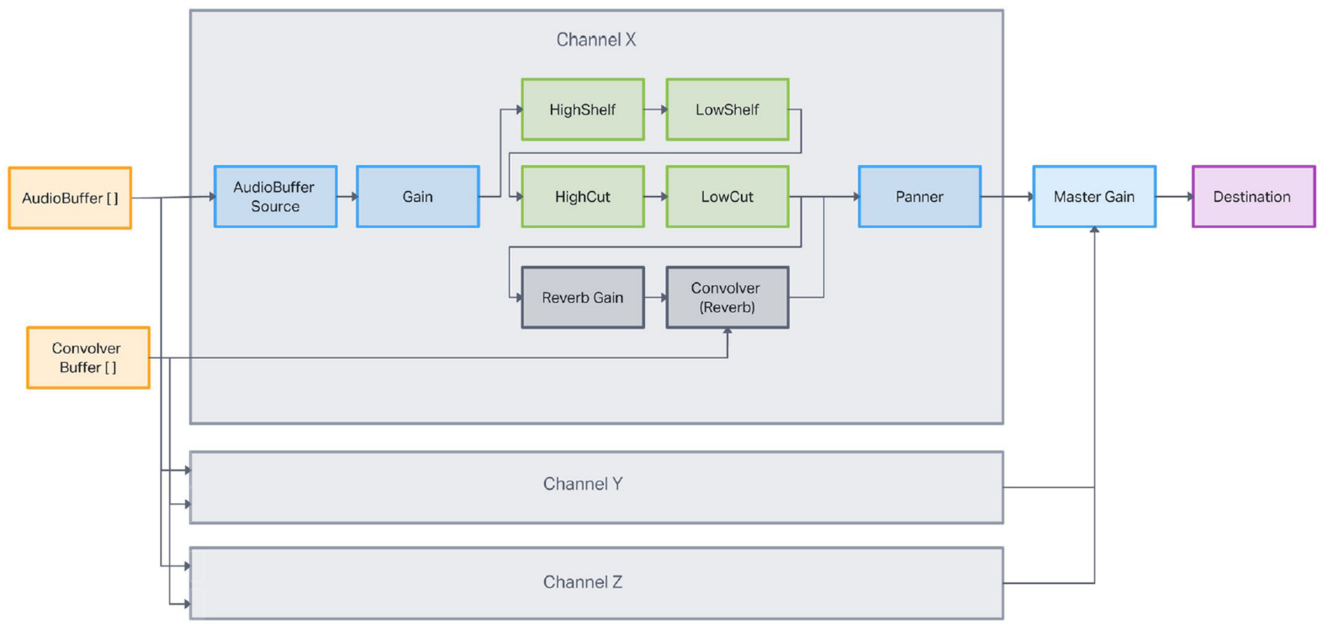
```
{
 "detections": [
  {
   "60": [
    {
     "object": "bear",
     "count": "1"
    }
   ]
  },
  {
   "120": [
    {
     "object": "bird",
     "count": "3"
    }
   ]

}
```

In its current stage of development, the Foley Machine only processes *.mp4 files. However, the project has been constructed in such a way that extensions, for example, detection via webcam, YouTube, other video formats, or images can be added gradually.

Audio Signal Flow

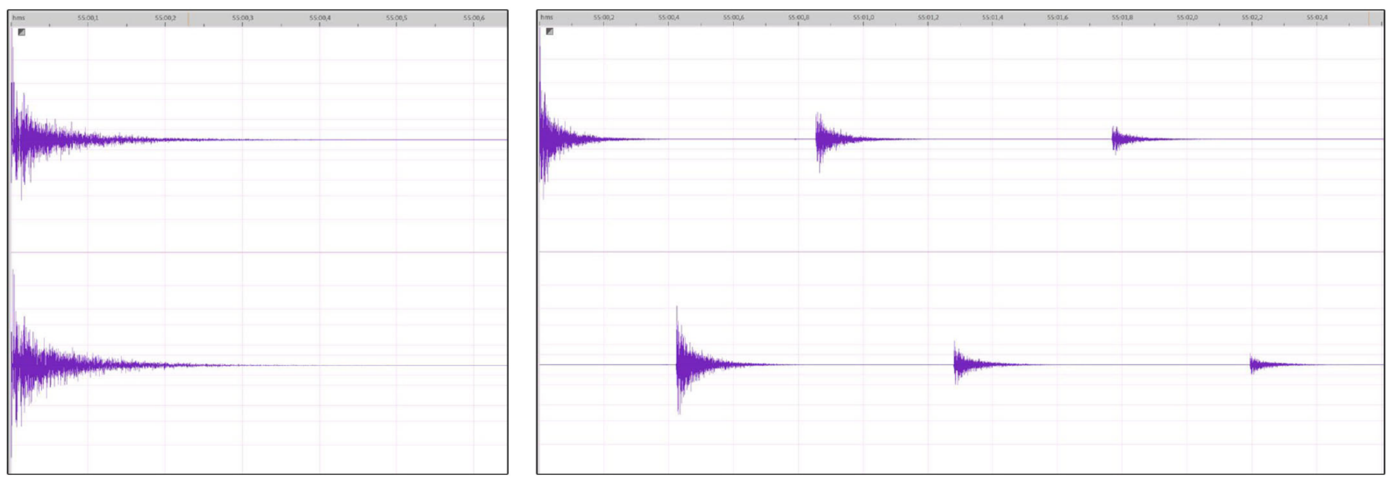
For the audio signal processing, we were inspired by the signal path of a classical mixing console. Thus, for each channel (sound), there is a separate gain control, an equalizer section, and an effects path through which a reverb can be mixed in. Subsequently, the position in the stereo signal can be selected via a panorama control. All channels ultimately run through a master gain control, with which the overall volume of the mix is adjusted.



Web Audio API Node-Flow [3]

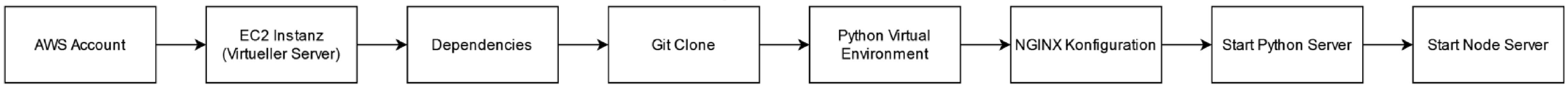
For the types of reverb, we opted for relatively common types with the "Room", "Garage", and "Church" settings. To spice things up a bit, we built our own input-response file for the "Ping Pong" setting, which allows the echo to alternate between coming from the left and right.

In the illustration, we see the waveform representation of the input-response file for the "Room" reverb compared to that for the "Ping Pong" reverb.



Waveform Input-Response-Files (Room vs. Ping Pong)

Deployment on AWS



Kevin Kröll | Moritz Uhlig | Leander Wernst