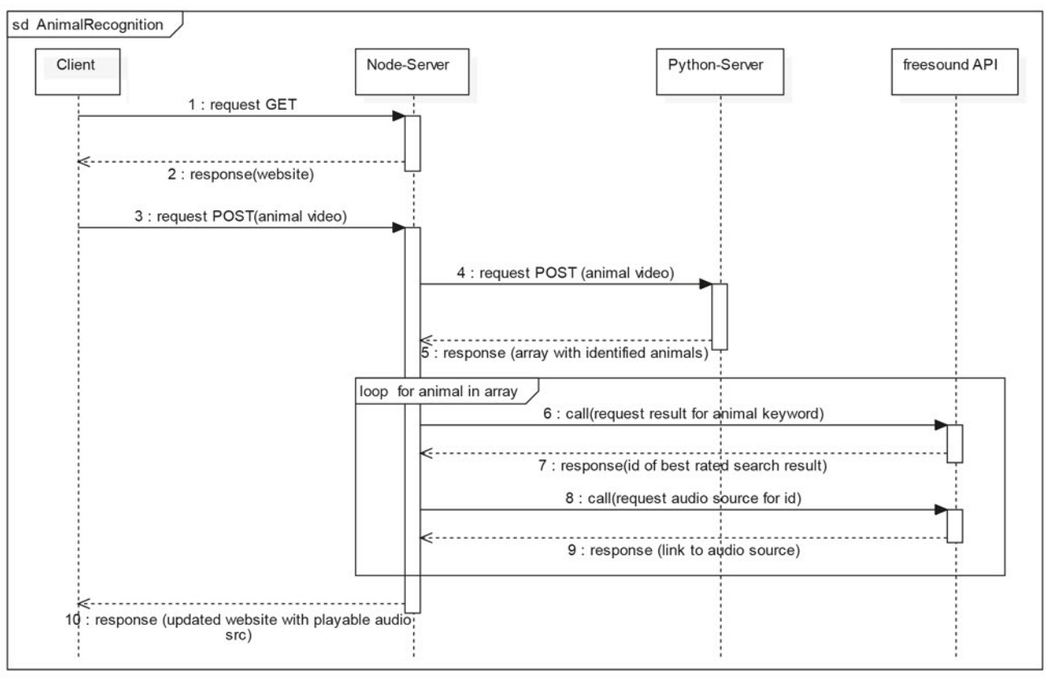
Automatic Foley Machine

Diese Website wird von einem node.js-Webserver bedient, der auch dafür zuständig ist, das hochgeladene Video vorübergehend auf seiner Festplatte zu speichern. Er teilt dann einem weiteren Server mit, an welchem Speicherort dieser das Video finden kann. Auf diesem zweiten Server läuft ein Python-Programm, das ein vortrainiertes Machine-Learning-Modell, welches auf dem s.g. YOLOv5-Algorithmus zur Objekterkennung basiert. Nach der Verarbeitung und Analyse der Einzelbilder sendet der Python-Server ein JSON-Objekt mit allen erkannten Tieren an den node.js-Webserver zurück.



Sequenzdiagramm Automatic Foley Machine [1]

Mit diesen Informationen fragt der Webserver nun mehrfach die API von freesound.org an. Die API Antwortet mit Ergebnissen zu den gesuchten Stichworten und letztlich mit konkreten Audiodateien, die dann in die Website eingebunden werden, um alle benötigten Audioelemente zu erstellen, die den Ton schließlich direkt von den freesound-Servern streamen.

Wir berechnen Zeitstempel, um sicherzustellen, dass die Töne zu dem Zeitpunkt abgespielt werden, wenn die Tiere auch im Video zu sehen sind. Die Audiodaten können dann während der Wiedergabe mit den Steuerelementen manipuliert werden, wofür wir die Möglichkeiten der Web Audio API verwenden.

FEATURES

* Objekterkennung mit Python (YOLOv5)
* AJAX Upload für dynamisches laden der Seiteninhalte
* File-Size Limit 40MB
* Auto-Löschen von Video-Files älter als eine Stunde
* Video-Metadaten auslesen, um Timestamps zu berechnen
* Integration YouTube-API zum Auslesen der Dauer
* Liste, die nach Tier-Objekten filtert
* Audio-Manipulation mit der Web Audio API
* Freesound.org API-Requests (Limit 60/Minute, 2000/Tag)
* API-Calls pro Video limitiert (max. 60 pro Upload)
* Python-Endpunkt, um Speicherort des Videos zu übermitteln
* Python-Response in Form einer JSON
* “Gleiche Tiere” in einer Node-Gruppe zusammengeführt
* Deployment auf AWS

In dieser Demo erkannte Objekte: bird cat dog horse sheep cow elephant bear zebra giraffe

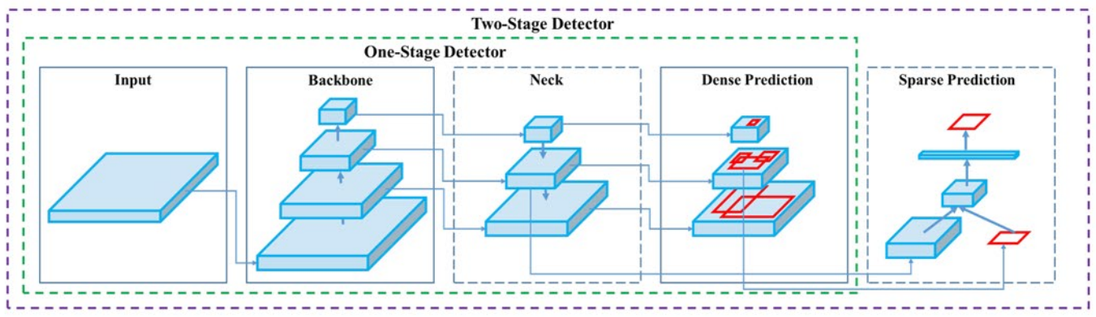
OBJEKTERKENNUNG

Für die Objekterkennung gibt es zwei unterschiedliche Arten von Modellen: Ein-Stufen-Detektoren und Zwei-Stufen-Detektoren. Das von uns verwendete YOLOv5 ist ein Ein-Stufen-Detektor (grüner Kasten). Es verwendet ein einzelnes neuronales Netzwerk, um die Objekt-Rahmen und Klassenwahrscheinlichkeiten direkt aus dem vollständigen Bild in einem Durchlauf vorherzusagen.

Backbone: Der Backbone ist das "Basis"-Klassifikationsmodell, auf dem das Objekterkennungsmodell basiert. Bei YOLOv5 ist das ein Convolutional Neural Network (CNN), das verschiedene Merkmale (features) extrahiert.

Neck: Eine Reihe verschiedener Schichten, in denen Bildmerkmale miteinander kombiniert werden und anschließend zur Vorhersage an den Head geschickt werden.

Head (Dense Prediction): Der Teil eines Objekt-Detektors, an dem die Vorhersage gemacht wird. Dazu werden die im Neck erzeugten Merkmale genutzt.



Single-Stage Detector Architecture [2]

Anpassung des Modells

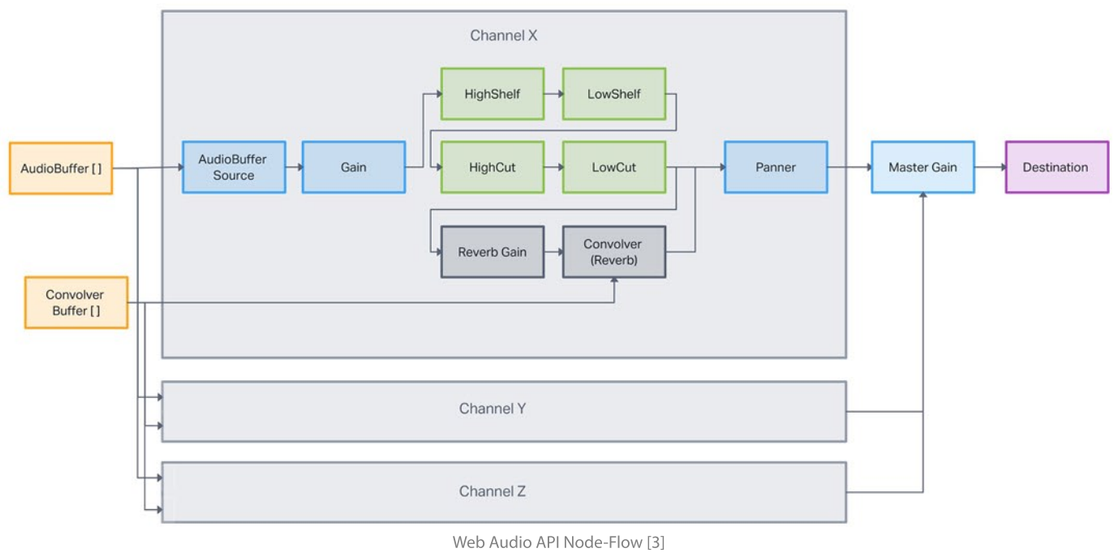
Für unser Projekt haben wir den Output von YOLOv5 auf unsere Bedürfnisse angepasst. Konkret haben wir den Video-Output inkl. Objekt-Rahmen deaktiviert und entsprechende Anpassungen im Code vorgenommen, damit die erkannten Objekte in Form von Key-Value-Paaren zurückgegeben werden.

In der jetzigen Ausbaustufe verarbeitet die Foley Machine lediglich *.mp4-Dateien. Die Architektur des Projekts wurde aber von vornherein so aufgebaut, dass Erweiterungen wie z.B. die Erkennung per Webcam, YouTube, andere Videoformate oder Bilder nach und nach möglich sind.

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

AUDIO SIGNAL FLOW

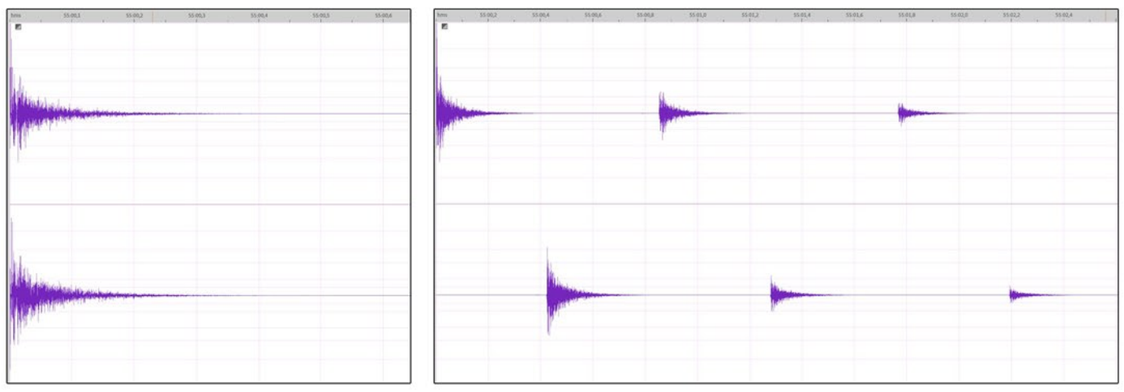
Bei der Audiosignal-Verarbeitung haben wir uns von dem Signalweg eines klassischen Mischpultes inspirieren lassen. So gibt es für jeden Kanal (Sound) einen eigenen Gain-Regler, eine Equalizer-Sektion sowie einen Effektweg, über den ein Reverb hinzugemischt werden kann. Anschließend kann über einen Panorama-Regler noch die Position im Stereo-Signal gewählt werden. Alle Kanäle laufen abschließend über einen Master-Gain-Regler mit dem die Gesamtlautstärke des Mixes angepasst wird.

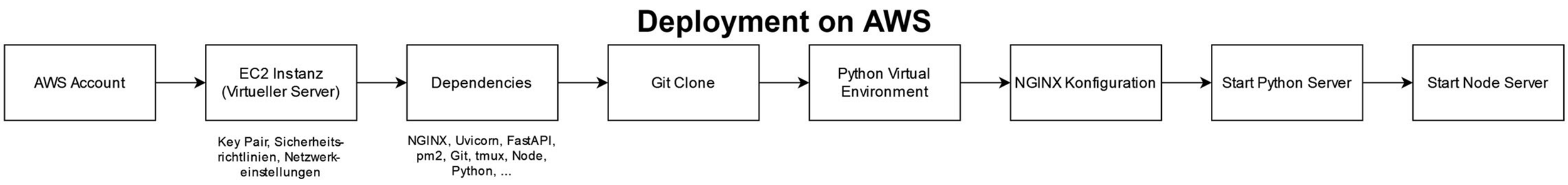


Web Audio API Node-Flow [3]

Bei den Reverb-Arten haben wir uns mit den „Room“-, „Garage“- und „Church“-Settings für relativ gängige Reverb-Arten entschieden. Um das Ganze etwas aufzulockern haben wir für das „Ping Pong“-Setting eine eigene Input-Response-File gebaut, die den Widerhall abwechselnd von links und rechts kommen lässt.

In der Abbildung sehen wir die Wellenformdarstellung der Input-Response-File für den „Room“ Reverb im Vergleich zu der für den “Ping Pong”-Reverb.


Wellenformdarstellung Input-Response-Files (Room vs. Ping Pong) [4]



Kevin Kröll | Moritz Uhlig | Leander Wernst