User Facing Documentation

Edit New page

Jump to bottom

Emil Badura edited this page now · 1 revision

Backend Services

Webcam Service

The webcam service is our source for video frames using an available local webcam. It offers a WebRTC signaling endpoint `/offer/`, which is used by the analyzer service and the frontend as well to get a stream of video frames with a H264 codec. For this we use the `aiortc` python library. Internally, the `recv()` function in the VideoStreamTrack object is called regularly to get a frame.

Analyzer Service

The analyzer service connects with the webcam service to process video frames using YOLO models for object detection and MiDaS models afterwards for depth estimation of each detected object in the frame. The frontend can receive the resulting metadata like bounding boxes, confidence, class and depth by connecting via a websocket to it.

Frontend Components/Hooks

Hooks

WebRTC Player

The frontend hook `useWebRTCPlayer.ts` offers an interface consisting of multiple functions to control the video player integrated in the UI. One can connect/disconnect to the webcam service, toggle play/pause or enter full screen mode. Latency polling is also included in the WebRTC receiver part. It is being used by the `WebRTCStreamPlayer` component.

Analyzer WebSocket

Analogous to the WebRTC player we also have a `userAnalyzerWebsocket.ts` hook, which is used to connect to the analyzer serivce using websocket and to update the `latestMetadata` variable, important for updating the video overlay. There is also a disconnect function as well for completeness.

Components

Icons

A simple component which handles the rendering of icons for the player controls like play, pause and maximize (fullscreen mode).

Player Controls

A react component which contains the button for play/pause as well as the fullscreen functionality. It is used in the WebRTC Stream Player component together with the video component.

Video Overlay

This component is placed together with the actual video component in a div called `video-container` and draws the latest metadata over the current video stream. The rendering is done in a central `useEffect()` component where bounding boxes together with confidence, class label and depth estimation in meter is being drawn on a `HTMLCanvasElement` which is being placed right on top of where the current video frames are shown. Some statistical information like FPS, displayed above the video container, is also being updated.

WebRTC Stream Player

This component is responsible for actually using the WebRTC player hook to establish a connection and to show the video frames along with the overlay in the UI. It also includes buttons to connect/disconnect the WebRTC stream, showing the current connection status at the same time, and the player control component. Effectively this component puts everything together and structures it appropiately.

+ Add a custom footer

Pages 4

Find a page...

- Home
- Project Architecture & Development Guide
- Technical Documentation
- User Facing Documentation

+ Add a custom sidebar

Clone this wiki locally

```
https://github.com/amosproj/amos2025ws04-robot-visual-perception.wiki.git
```