2023

Dokumentation Trackingprototyp für eine automatische Laborstraße

Mirko Mettendorf

11.7.2023

Inhalt

1 Einleitung Prototyp

1.1 Kurzerklärung des Trackingprototyps

Bei diesem Prototypen handelt es sich um ein KI-gestütztes Kameratrackingsystem um eine Automatische Laborstraße zu überwachen. Dafür werden zwei Kameras verwendet, eine Deckenkamera und eine zum Scannen der Reaktionsgefäße. Nach dem Scannen der IDs werden die Reaktionsgefäße mithilfe von YOLOV8 detektiert und mit dem Trackingalgorithmus BoT-SORT getrackt. Die bei dem Tracking gesammelten Informationen über die Reaktionsgefäße werden in einer CSV-Logdatei abgespeichert.

1.2 Laborstraße

Die Laborstraße kann sich von Experiment zu Experiment unterscheiden. Der Tracker wird anhand einer bestehenden automatischen Laborstraße entwickelt, wird aber wegen der Möglichkeit eines wechselnden Aufbaus, darauf ausgelegt, für die verschiedensten Laborstraßen nutzbar zu sein. Die Laborstraße muss in den Bildausschnitt der Deckenkamera passen, der sich je nach Entfernung zum Arbeitstisch unterscheidet. Ansonsten gibt es keine Bedingungen an die Laborstraße. Es müssen nur alle Stationen und die verwendeten Tubes mit trainiert werden für den verwendeten Detektor. Eine Unterstützung von beweglichen Robotern ist ebenfalls vorhanden.

2 Aufbau Prototyp

2.1 Hardware

2.1.1 Komponenten

- Deckenkamera CTRONIC IP Kamera
  Hauptkamera, welche von dem Tracker benutzt wird
- Webcam XXX
  Zweitkamera, die über USB verbunden wird und für das Scannen der Micro-QR-Codes genutzt wird.
- Computer
  Zum Trainieren des Detektors ist ein Computer mit einer Nvidia Grafikkarte empfohlen, da das Training sonst sehr lange benötigt. Für den Betrieb kann jeder beliebige Computer verwendet werden.

2.1.2 Funktionsweise

Die Zweitkamera muss über USB mit dem verwendeten Computer verbunden sein und die Deckenkamera muss über WLAN in dem gleichen Netzwerk wie der Computer sein.

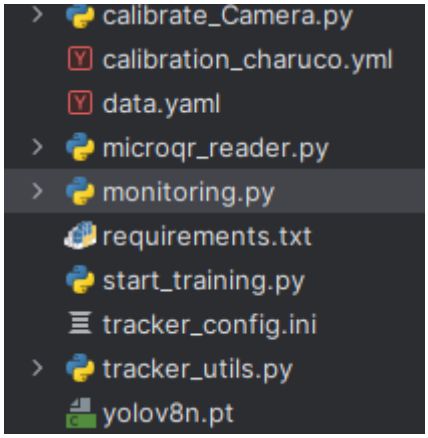
2.2 Software

2.2.1 Programmiersprache und Bibliotheken

- PYTHON 3.9
- JAVA
- OPENCV
- Ultralytics
- Supervision
- Pyboof 0.41
- CUDA Toolkit
- PyTorch
- Detektor YOLOv8
- Tracker BoT-SORT

2.2.2 Struktur

In dem Projektordner befinden sich die verschiedenen Programme, weitere benötigte Dateien und die Konfigurationsdatei, um den Prototypen vorzubereiten und zu betreiben.



Die genaue Funktion der einzelnen Dateien wird in den folgenden Kapiteln behandelt-

2.2.3 Funktionsweise

Zum Vorbereiten werden die Module calibrate_Camera.py und start_training.py verwendet. Diese Kalibrieren die Kamera und führen das Training des Detektors aus. Für den Betrieb wird die microqr_reader.py und die monitoring.py genutzt.

3 Vorbereitung

3.1 Montage und Software einrichten

Die Deckenkamera muss so mittig und tief wie möglich über der Laborstraße befestigt werden, sodass die gesamte Laborstraße noch im Bild der Kamera zu sehen ist. Sie muss an den Strom angeschlossen werden und mit dem WLAN oder LAN verbunden werden. Für WLAN muss man sich zuerst mit dem eigenen WLAN der Kamera verbinden und über die IP der Kamera im Browser kommt man auf die Konfigurationsseite der Kamera, in der man seine WLAN-Daten eintragen kann. Dort kann man auch noch andere Einstellungen der Kamera vornehmen.

Die Zweitkamera wird einfach über USB mit dem Computer verbunden und ist direkt einsatzbereit.

Für die Software wird Python 3.9 benötigt. Zusätzlich muss Java installiert sein und das Cuda Toolkit von Nvidia, wenn zum Trainieren die Grafikkarte verwendet werden möchte.

Für Ultralytics muss die richtige PYTorch Version installiert sein, dazu den Befehl „pip3 install --pre torch torchvision torchaudio --index-url https://download.pytorch.org/whl/nightly/cu121“ verwenden.

Es gibt eine requirements.txt mit der können alle weiteren Module automatisch installiert werden. Dazu in einem Terminal den Befehl „pip install -r requirements.txt“ ausführen.

Der Laborant kann über einen Telegram Bot Benachrichtigungen des Prototpys auf sein Smartphone bekommen. Dafür muss er sich den Bot einrichten. Eine Anleitung dazu findet man hier https://sean-bradley.medium.com/get-telegram-chat-id-80b575520659

Danach gibt man in der Konfiguratiosndatei den API Token und die ChatID ein.

3.2 Kalibrierung der Kamera

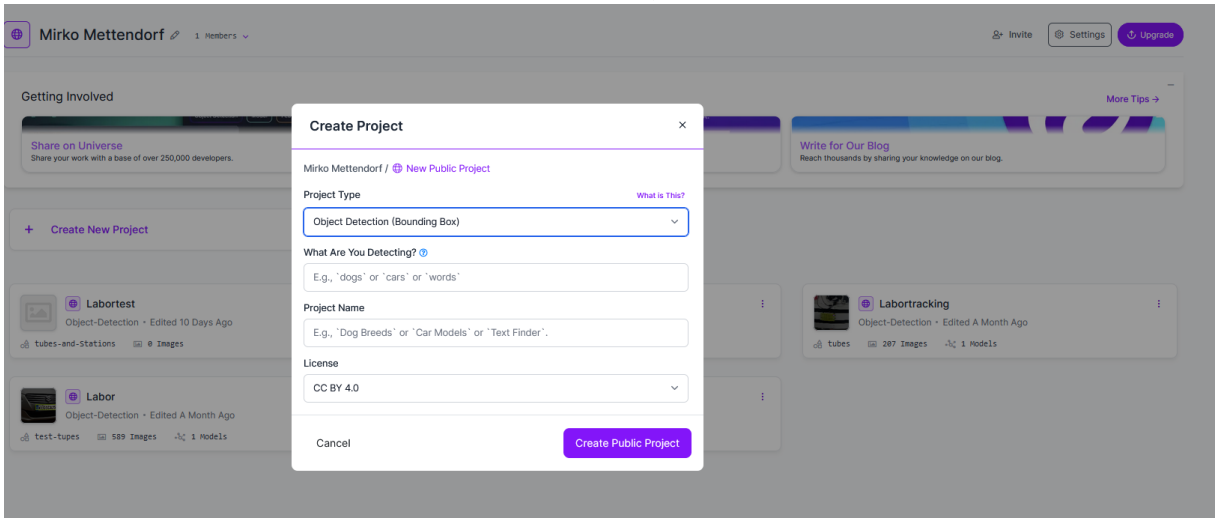
Um die Kamera zu Kalibrieren muss das ChAruCo Bord ausgedruckt werden. Das befindet sich ebenfalls in dem Projektordner. Danach wird es an verschiedenen Positionen der Kamera positioniert und 20-30 Bilder mit der Deckenkamera davon gemacht. Diese Bilder müssen von der Kamera runtergeladen werden und auf dem Computer abgespeichert werden. In der tracke_config.ini muss unter Calibration der Ablageort der Bilder in image-directory eingetragen werden. Zusätzlich muss noch die Länge der Marker auf dem ausgedruckten CharucoBord gemessen und eingetragen werden. Danach kann die Kalibrierung mit dem Befehl „python calibration_camera.py gestartet werden. Das Ergebnis wird in der calibration_charuco.yml abgelegt und wird beim Benutzen des Trackers automatisch verwendet. Es liegt auch bereits so eine Datei vor, für die genutzte Kamera. Sollte eine andere Kamera verwendet werden, muss die Kalibrierung erneut durchgeführt werden, ansonsten ist dies nicht notwendig.

3.3 Testdaten aufnehmen

Zum Trainieren benötigt man Testdaten. Dazu mithilfe der Kamera ein 2-3 minütiges Video aufnehmen, während die Laborstraße im Betrieb ist. Die Laborstraße muss alle Stationen beinhalten und auch die verwendeten Reaktionsgefäße. Dieses Video wird mithilfe der entzerren.py entzerrt. Dazu wird das Video in dem Projektordner unter dem Namen testvideo.mp4 abgelegt und danach mit dem Befehl „python entzerren.py“ ausgeführt. Das entzerrte Video befindet sich in dem Projektordner unter dem Namen testvideo_entzerrt.mp4

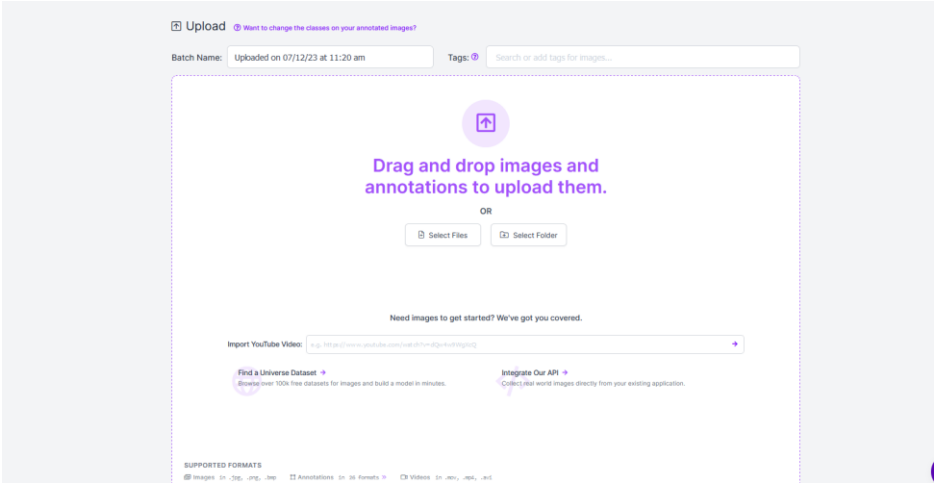
3.4 Annotieren der Testdaten

Zum Annotieren wird die Webseite RoboFlow verwendet. Auf der Seite https://app.roboflow.com kann man nach dem Erstellen eines kostenlosen Kontos, ein neues Projekt erstellen.

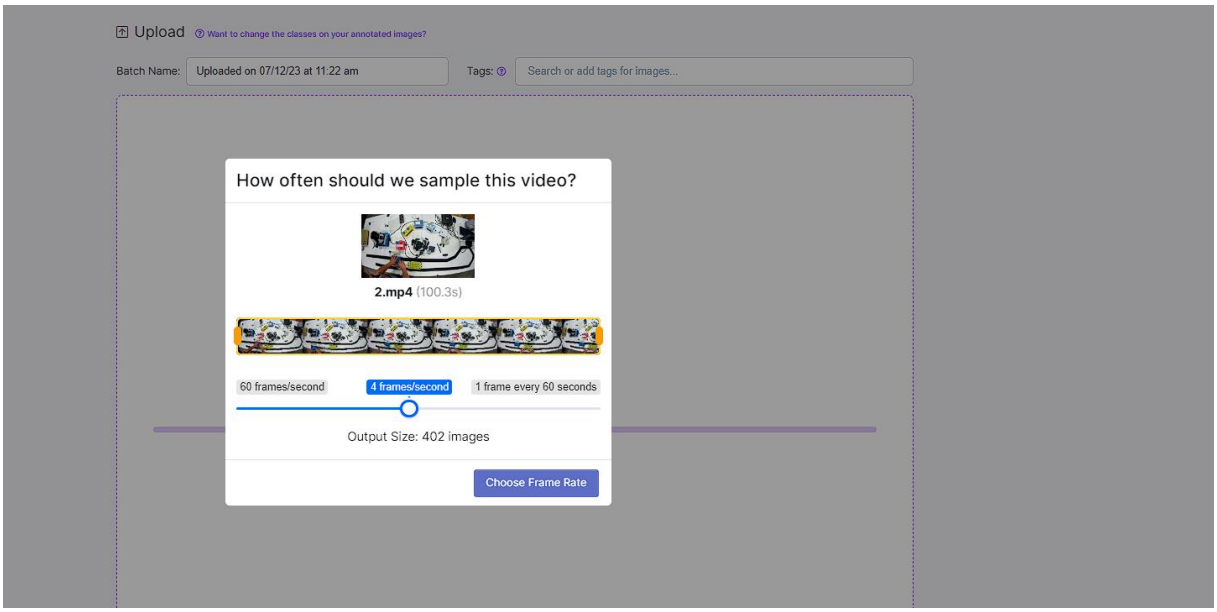


Dort wählt man den Projekt Type Object Detection aus, einen beliebigen Projektnamen und Objektname und drückt auf Create Public Projekt.

Auf der nächsten Seite kann man sein Video einfach reinziehen oder auswählen.



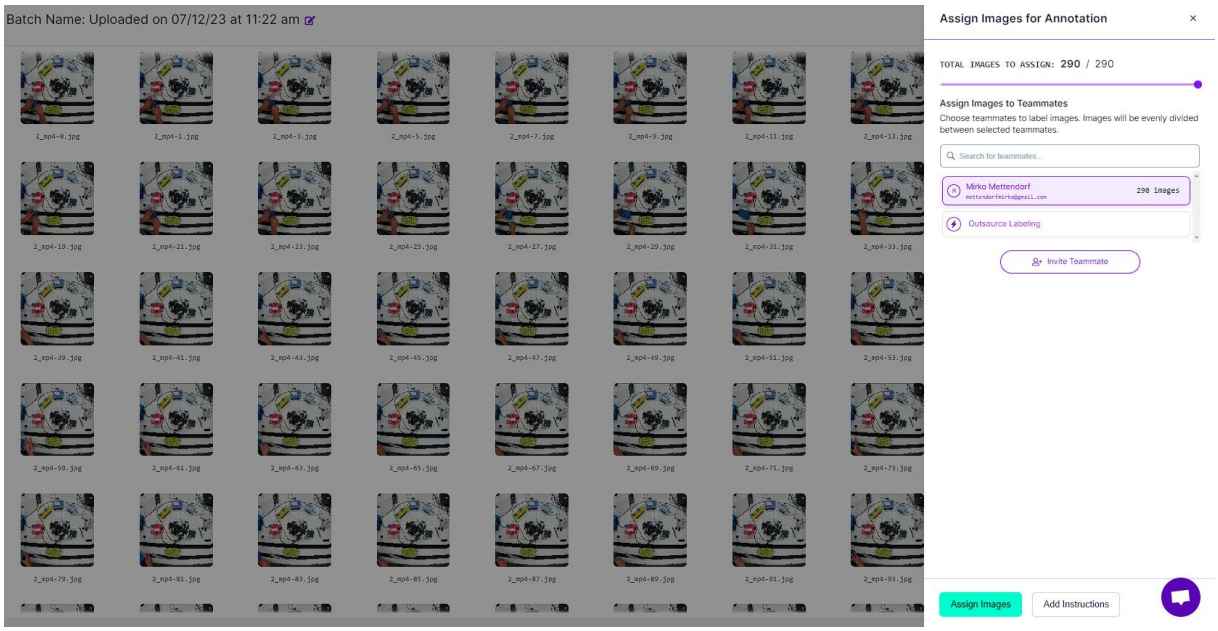
Danach wählt man die Framerate von seinem Video aus und drückt auf Choose Frame Rate.



Jetzt werden die Frames extrahiert und als Einzelbilder in dem Projekt abgespeichert. Dies dauert 1-2 Minuten.

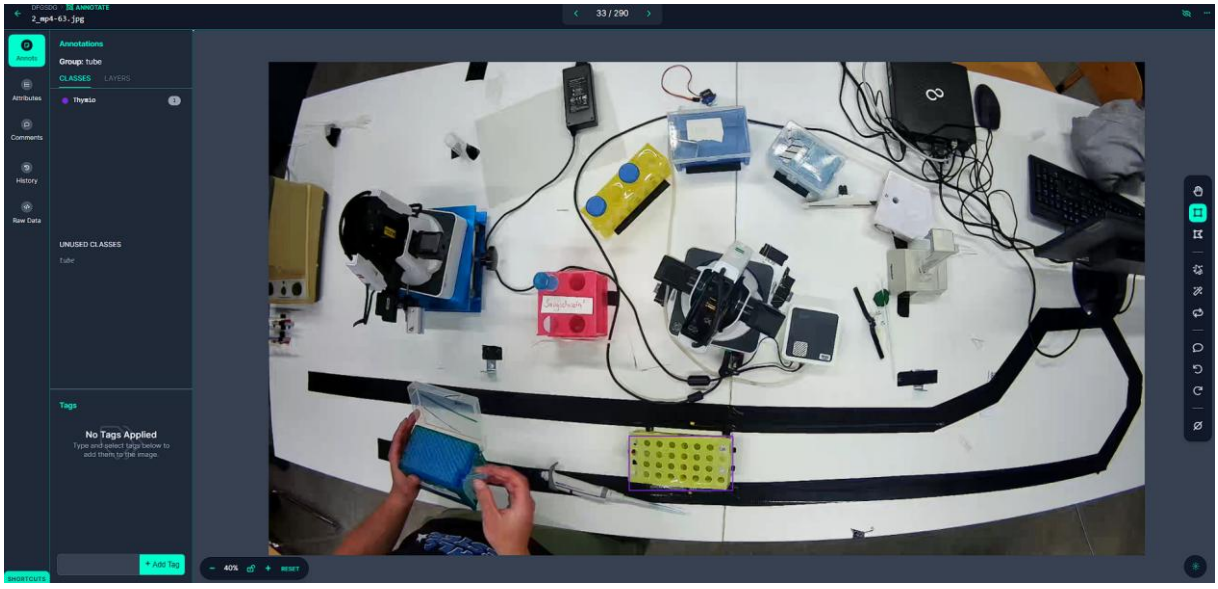
Danach kann man sich die Einzelbilder anschauen und falls gewünscht noch welche entfernen. Danach drückt man auf Save and Continue.

Jetzt können die Bilder annotiert werden. Dazu drückt man auf Assign Images. Falls man andere Nutzer auch zum Annotieren einladen möchte, wählt man Invite Teammate aus und teilt die Bilder auf den verschiedenen Nutzern auf.



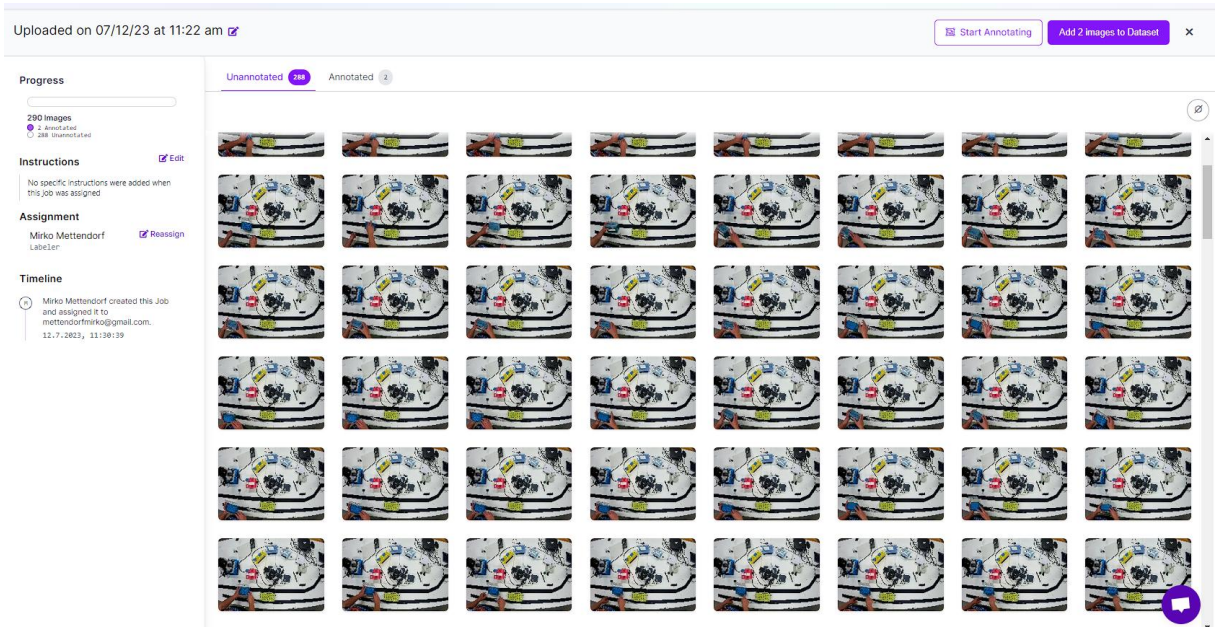
Jetzt wurde ein Job erzeugt dieser kann mit Start Annotating ausgeführt werden.

In jedem Bild müssen alle Stationen markiert werden und die sichtbaren Reaktionsgefäße.

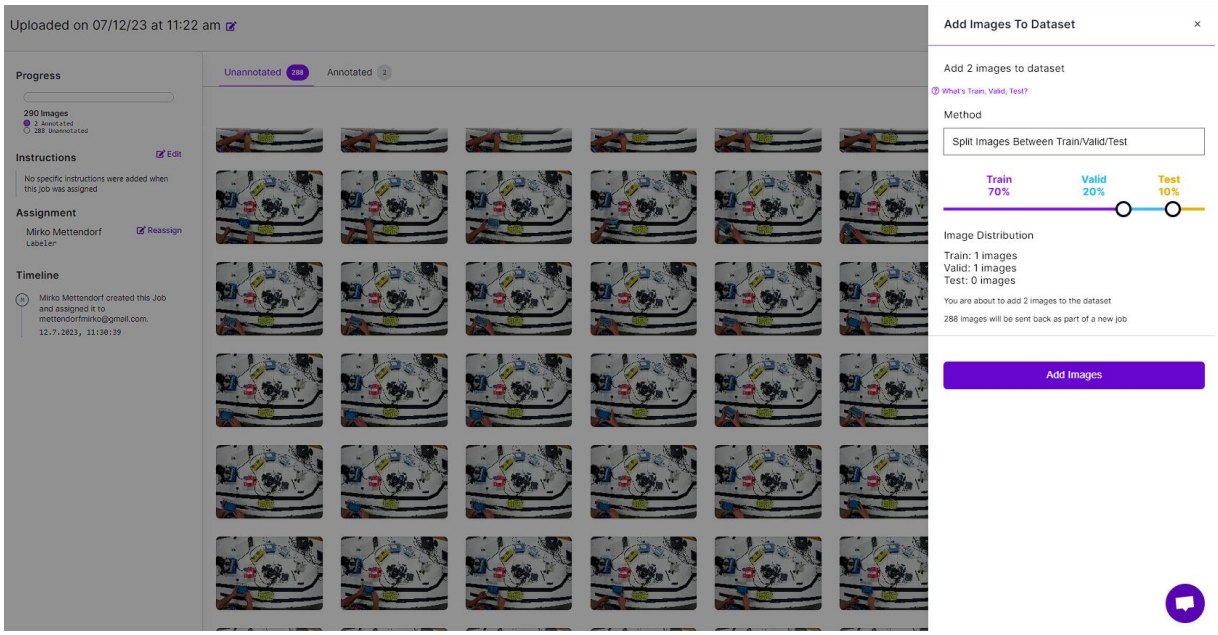


Dazu können rechts verschiedene Tools verwendet werden. Es gibt die klassische Bounding Box, zum Auswählen der Objekte, aber auch noch das Smart Polygon, da kann durch ein Klick das Objekt anhand seiner Umrisse markiert werden, und nur noch eventuell durch kleine Anpassungen korrigiert werden. Hat man bereits ein trainiertes Modell kann man dies auch in der Label Assist Funktion genutzt werden, um automatisch die Objekte zu erkennen. Hat ein Objekt sich nicht bewegt, kann man es mit der Repeat Previous Funktion direkt an der gleichen Stelle markieren.

Ist man mit dem Annotieren aller Bilder fertig, klickt man oben links auf den Pfeil und kann dann die Annotierten Bilder zu einem Dataset hinzufügen mit dem Lila Button oben rechts.

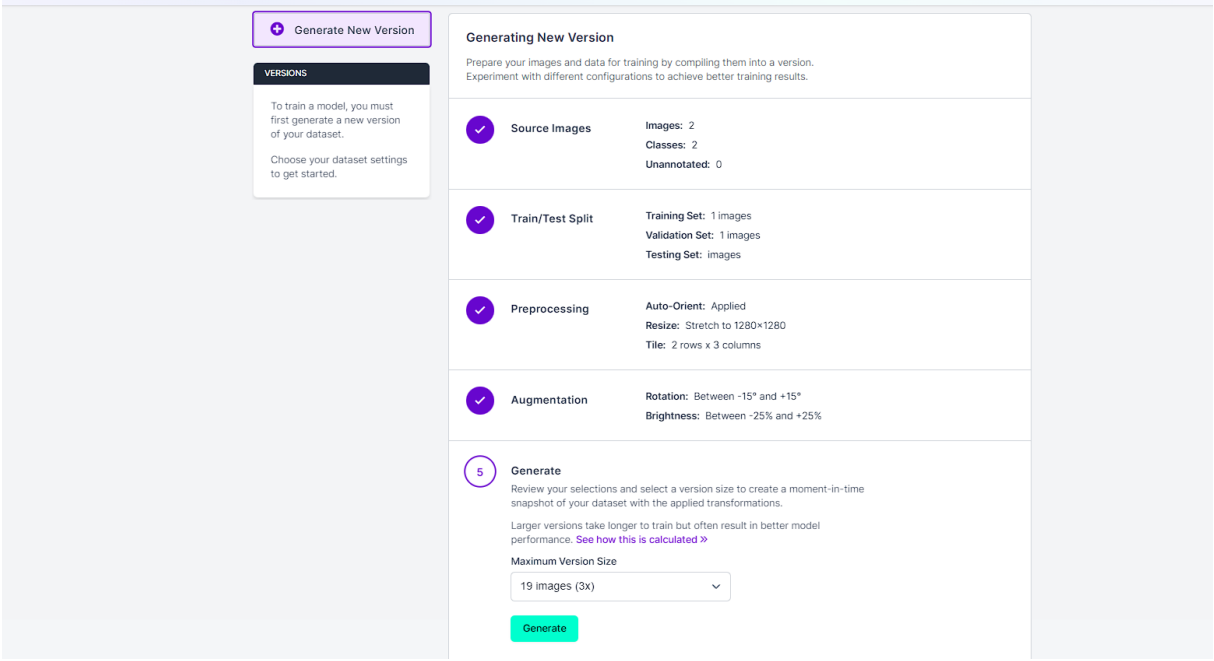


Dann öffnet sich ein Reiter, indem die Bilder in Training,-Validierungs- und Testbilder aufgeteilt werden.



Die Standard Aufteilung ist 70%,20% und 10%. Diese behalten wir bei und drücken auf Add Images.

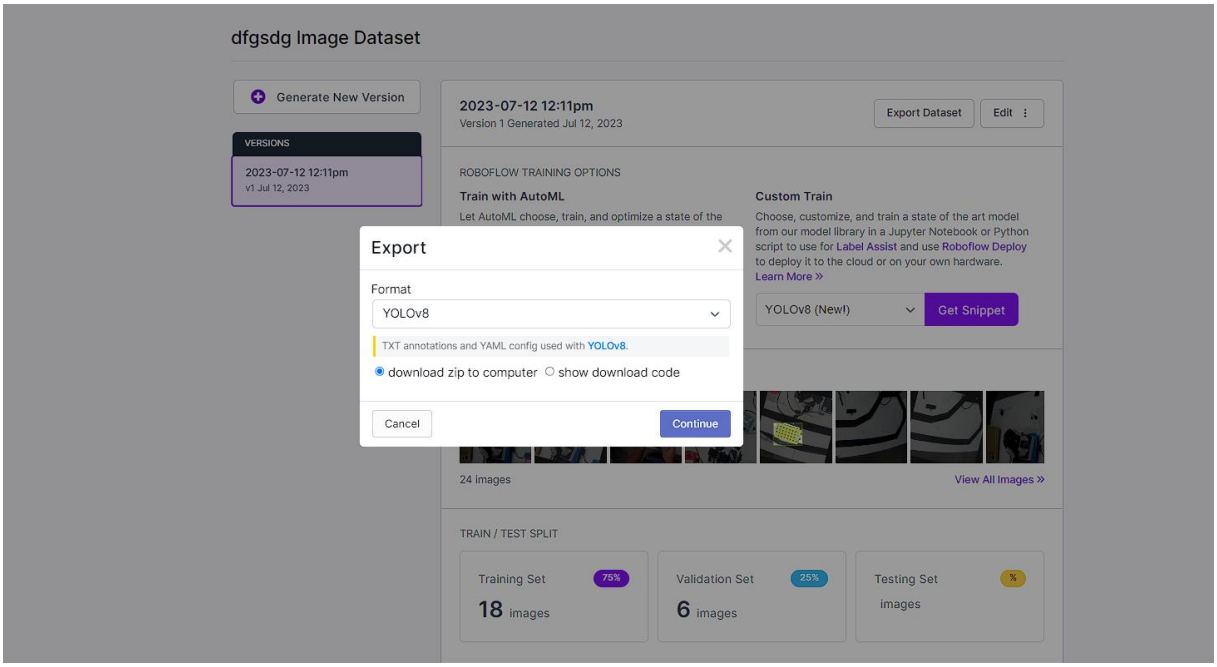
Jetzt wählen wir links den Reiter Generate aus und fügen unter Preprocessing den Schritt Tile mit den Werten 2 x 3 hinzu und ändern Resize auf 1280 x 1280. Damit wurde ein 4k Bild in sechs Teilbilder der Größe 1280 mal 1280 aufgeteilt, mit denen das Training optimal durchgeführt werden kann.



Unter Augmentation können noch Rotation und Brightness hinzugefügt werden.

Danach wird unter Generate die Versionsgröße auf 2x gestellt und Generate gedrückt. Es sollten mindestens 500 Bilder sein.

Jetzt befindet man sich in dem Versions Reiter und dort wählt man bei Custom Train YOLOv8 aus und drück Get Snippet.



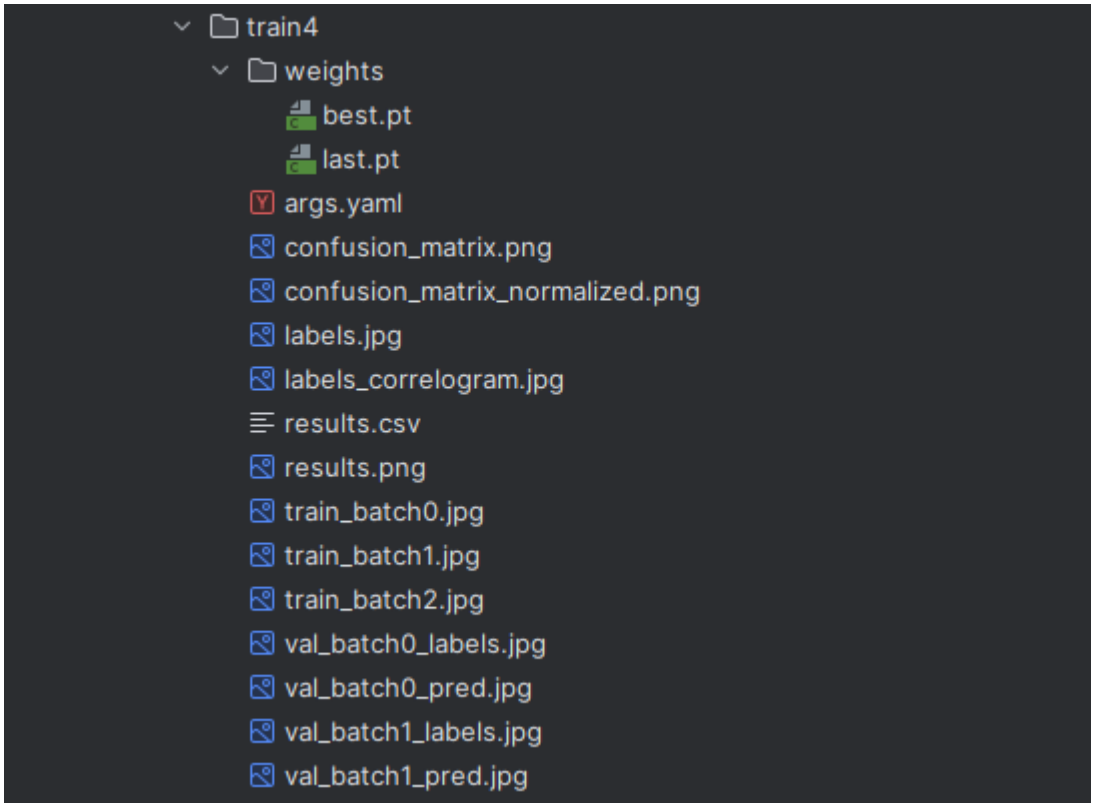
Jetzt wählt man “download zip to computer“ aus und drückt Continue. Die Datei mit den Trainingsdaten wird nun heruntergeladen.

3.5 Training des Detektors

Zum Trainieren benötigt man nun die heruntergeladenen Trainingsdaten. Diese werden entpackt und dann in einen neu erstellen Ordner neben dem Projektordner namens Datasets abgelegt. Die data.yaml legt man in den

Projektordner. In der Konfigurationsdatei kann wenn gewünscht unter Detektor die Anzahl an Epochen noch angepasst werden. Um das Training zu Starten wird der Befehl „python start_training.py“ ausgeführt. Dieser Vorgang kann mehrere Stunden dauern. Zu Beginn sieht man, ob die Grafikkarte verwendet wird. Sollte das nicht der Fall sein, muss erneut nach der korrekten CUDA Version und dem korrekten Pytorch geschaut werden.

Ist das Training beendet, befindet sich in dem Projektordner unter runs/detect/train/weights die Datei best.pt. Das ist das trainierte Yolov8 Model. Dieser Pfad muss in der Konfigurationsdatei unter tracker_weights_path angegeben werden



Zusätzlich befinden sich in dem train Ordner noch Bilder der Trainingsergebnisse.

Alternativ kann man auch mithilfe des train.ipynb Notebooks aus dem Tracker_Config Ordner mithilfe von Google Colab das Training mit einer besseren Grafikkarte durchgeführt werden.(Alternativ sogar mit einer noch stärkeren, die ist aber kostenpflichtig) Dann alle Code Zeilen nacheinander ausführen. Am Schluss die Zip Datei runterladen und die best.pt wieder am richtigen Ort ablegen.

4 Bedienung

4.1 Start des Prototyps

Einschaltereihenfolge:

- Deckenkamera mit Strom verbinden
- Warten bis sie sich einmal gedreht hat
- Mobiler Hotspot am PC aktivieren
- Warten bis Kamera verbunden ist
- Überprüfen, ob IP der Kamera mit IP, die in der Konfig Datei steht übereinstimmt
- System ist betriebsbereit

Die Deckenkamera muss eingeschaltet werden und der mobile Hotspot muss aktiviert werden und die Kamera muss eine Verbindung aufgebaut haben.

Zum Starten werden die Module microqr_reader.py und monitoring.py verwendet. Diese können in das Programm zum Steuern der automatischen Laborstraße importiert und verwendet werden. Das Lesen der Micro- QR-Codes kann mit der microqr_reader(count) Methode gestartet werden. Mit Count wird die gewünschte Anzahl an Reaktionsgefäßen angegeben. Das Programm läuft so lange, bis alle Gefäße erkannt wurden. Sollte die in der Konfigurationsdatei angegebene Wartezeit (wait_qr_time) überschritten werden, dann wird eine Nachricht an den Laboranten geschickt. War die Erkennung erfolgreich wird ein Tupel bestehend aus der Anzahl und einer Liste mit den gelesenen IDs und ihren Mittelpunktkoordinaten zurückgegeben. Mithilfe der Liste kann nun der Tracker gestartet werden, mit der start_tracking(). Diese führt den Tracker parallel in einem eigenen Thread auf, sodass die Steuerung der automatischen Laborstraße weiter ausgeführt werden kann.

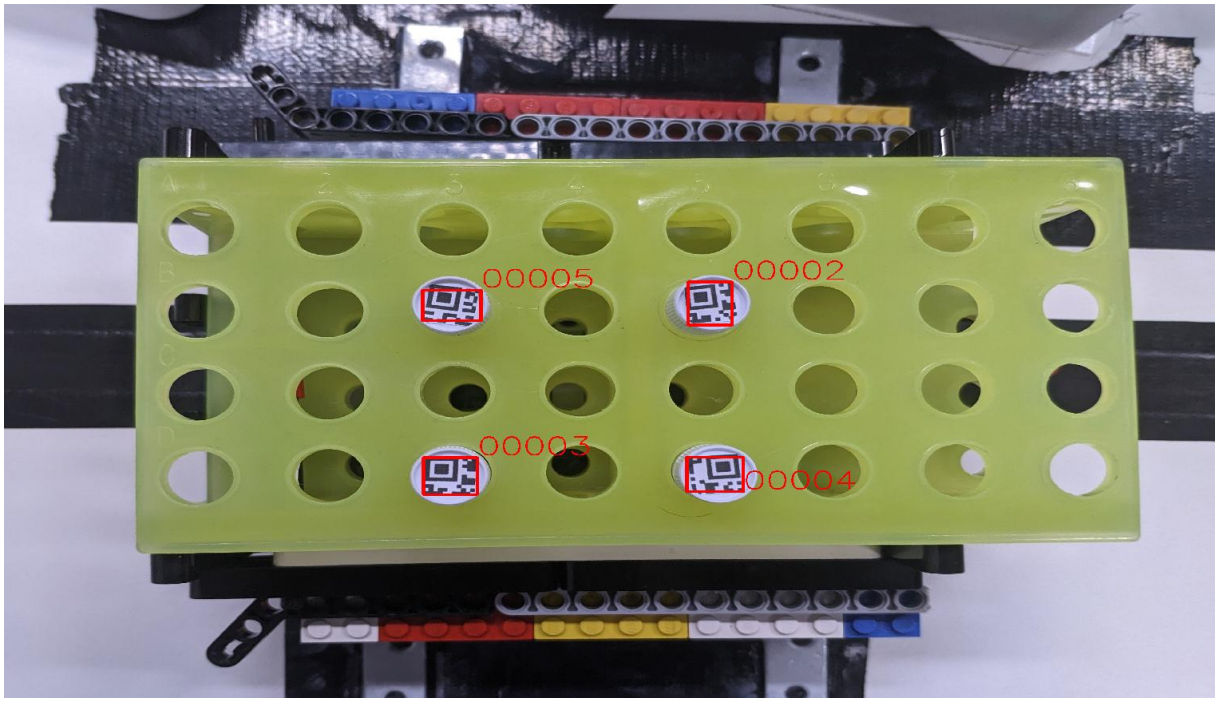
Vor dem Starten müssen noch einige Einstellungen in der Konfigurationsdatei tracker_config.ini vorgenommen werden.

```
[Camera]
cameraIp = 192.168.137.245

[Calibration]
image_directory=C:\\Users\\Mirko\\Desktop\\Bachelorarbeit\\Kalibrierbilder
image_format=jpg
marker_length=2.35
square_length=3.9

[Detektor]
epochs=200

[Tracker]
tracker_weights_path =C:\\Users\\Mirko\\repositories\\Laborautomatisierung\\untitled\\Labortracking\\best.pt
wait_qr_time = 30
station_names = ['Thymio','Deckelentnahme']
moving_station_names = [['Thymio_Anfangsstation','Thymio_Zwischenstation','Thymio_Endstation'],None]
moving_stations_distance_limit = 15
tube_rows=3
tube_column=4
length_station1=25
zielpfad_log =C:\\Users\\Mirko\\Desktop\\Bachelorarbeit\\Testdaten\\
error_wait_time=30

[Telegram]
api_token=5907423764:AAHPvern0hwCGbJ__rANnxZX5jJVxl-hbxU
chat_id='954707756'
```

- station_names: hier müssen in eine Liste die Namen aller Stationen in chronologischer Reihenfolge
- moving_station_names: Sollte eine Station beweglich sein, dann müssen alle Unterstationsnamen der Station in diese zweidimensionale Liste, an die Stelle wo sie in station_names ist
- moving_stations_distance_limit: Anzahl an Zentimeter, ab der der Unterstationsname zum nächsten springen soll
- tube_rows: Die Anzahl an Reaktionsgefäßen in einer Zeile an der Startposition
- tube_columns: Die Anzahl an Reaktionsgefäße in einer Spalte an der Startposition
- length_station1: Länge der ersten Station, um den Faktor zwischen Pixeln und Zentimetern zu bestimmen.
- zielpfad_log: Ordner, in dem die Logdateien abgelegt werden sollen
- error_wait_time: Zeit in Sekunden, nach der eine Warnung an den Laboranten geschickt wird, wenn ein Reaktionsgefäß, solange sich nicht mehr von einer Station wegbewegt hat.

4.2 Im Betrieb

Während der Erkennung der Micro-QR-Codes wird ein Fenster mit dem Kamerabild angezeigt und den darin erkannten Codes.



Dieses dient zur Kontrolle und bleibt so lange offen, bis die Erkennung der gewünschten Codes abgeschlossen wurde.

Während der Tracker läuft, wird ebenfalls ein Fenster mit dem Video der Deckenkamera angezeigt, welches live die getrackten Objekte markiert

Sobald der Tracker gestartet wurde, trackt er die Reaktionsgefäße und die Stationen im Bild. Der genutzte Trackingalgorithmus ist BoT-SORT. Dieser wurde ausgewählt, da er zur Zeit einer der Multi-Objekt-Tracker ist, mit den besten MOTA und IDF1 Ergebnissen und auch in dem gleichen Modul verfügbar ist, wie der YOLOv8 Detektor.

Es wird erfasst, wenn ein Tube in einer Station ist. Sobald das Tube diese verlässt, wird das ebenfalls erfasst und gespeichert. In jedem Frame wird die Entfernung des Tubes zur nächsten Station gemessen. Sobald das Tube die nächste Station erreicht, wird ein Logeintrag in die CSV-Datei geschrieben, mit der ID des Tubes, mit der Start und Zielstation, den jeweiligen Uhrzeiten, der Dauer zwischen den Stationen und dem Zeitstempel, an welcher Stelle im Video es sich befindet. Dieses Log dient zur übersichtlichen Dokumentation der Fahrtwege der Tubes von Station zu Station. Das Programm wird mit der Taste s auf der Tastatur beendet

ID	Startstation	Startzeit	Zielstation	Zielzeit	Dauer	Zeitstempel
1	Thymio	20.06.23:12:14:10	Deckelstation	20.06.23:12:14:20	10	0:14:10
2	Thymio	20.06.23:12:14:30	Deckelstation	20.06.23:12:14:30	10	0:14:20

Zusätzlich gibt es noch eine detaillierte Logdatei, die für alle Tubes pro Frame ihre ID, ihre letzte Station, ob sie außerhalb einer Station sind und den Namen und die Entfernung der nächsten Station enthält. Dies dient hauptsächlich zur Fehlerüberprüfung und zur Datensammlung.

ID	Letzte Station	In Station	Nächste Station	Entfernung
1	Thymio	True	Thymio	0
2	Thymio	False	Deckelstation	10

Sollte eine Tube zu lange nicht mehr eine Station verlassen oder betreten, dann wird eine Warnung an den Laboranten geschickt, um eine manuelle Überprüfung vorzunehmen.

5 Ausblick

Während der Entwicklung sind noch einige Aspekte aufgefallen, die bei dem Prototypen verbessert werden könnten oder den Prototypen um weitere Funktionalitäten erweitern könnten. Jene hätten aber den Umfang dieser Arbeit überschritten und werden deshalb im Folgenden aufgeführt, um den Prototypen vielleicht in Zukunft noch zu optimieren und erweitern, und zu Nachfolgeprojekten zu führen.

5.1 Optimierungsmöglichkeiten

- Weitere Warnmöglichkeiten des Trackers, zurzeit nur, falls eine Tube zu lange in einer Station bleibt. Zum Beispiel könnte eine Warnung gemeldet werden, wenn ein Tube unterwegs verloren geht.
- Andere Deckenkamera testen, mit besserer Qualität und direkter Verbindung, um die Verzögerung der Funkübertragung, sowie die Komprimierung der Daten zu minimieren und damit vielleicht ein besseres Trackingergebnis zu erreichen.
- Laborcomputer mit Grafikkarte verwendet für bessere Performance

5.2 Ideen für Nachfolgeprojekte

- Micro-QR-Code mit Handscanner lesen können, und direkt die Logeinträge anzuzeigen
- Verknüpfung des digitalen Laborbuchs mit der Logdatei
- Das Tracken des Doobot Greifers, um mithilfe der Tube- und Stationkoordinaten, die Positionierung gegebenenfalls zu korrigieren, um das Tube aufzunehmen, oder in eine Station abzustellen.
- Eine grafische Oberfläche zum Bedienen des Trackers und einer Liveansicht der Trackingdaten.