

Anomaliedetektion mit Computer Vision in der Gepäckförderanlage

ARIC Remote Brown Bag Session, 25.03.21



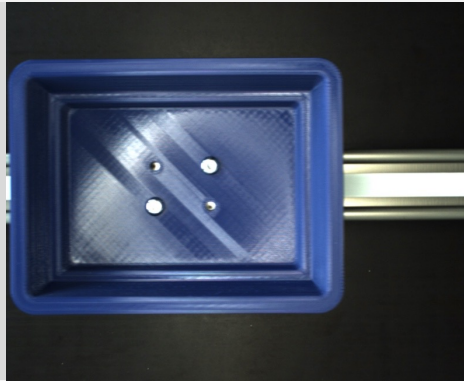
Ziel: Störungen vermeiden,
“anomalisch” beladene
Wannen identifizieren

Herausforderung

Störungen vermeiden

Wann befindet sich eine Wanne auf dem Förderband?

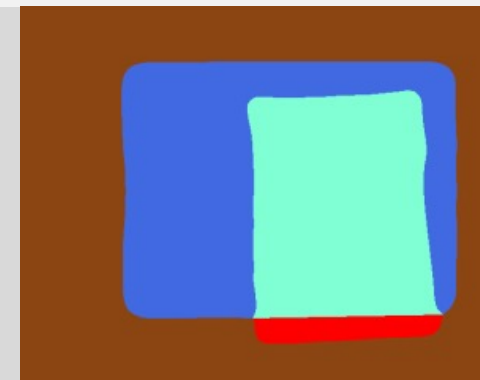
- Real-time
- Einmalig pro Wanne



Günstige Methoden

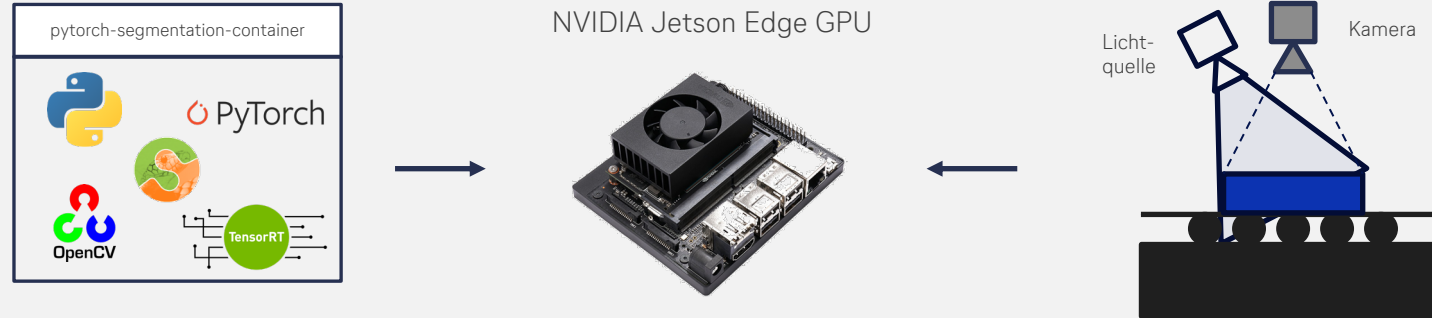
Ist die Wanne korrekt beladen?

- Near-real time



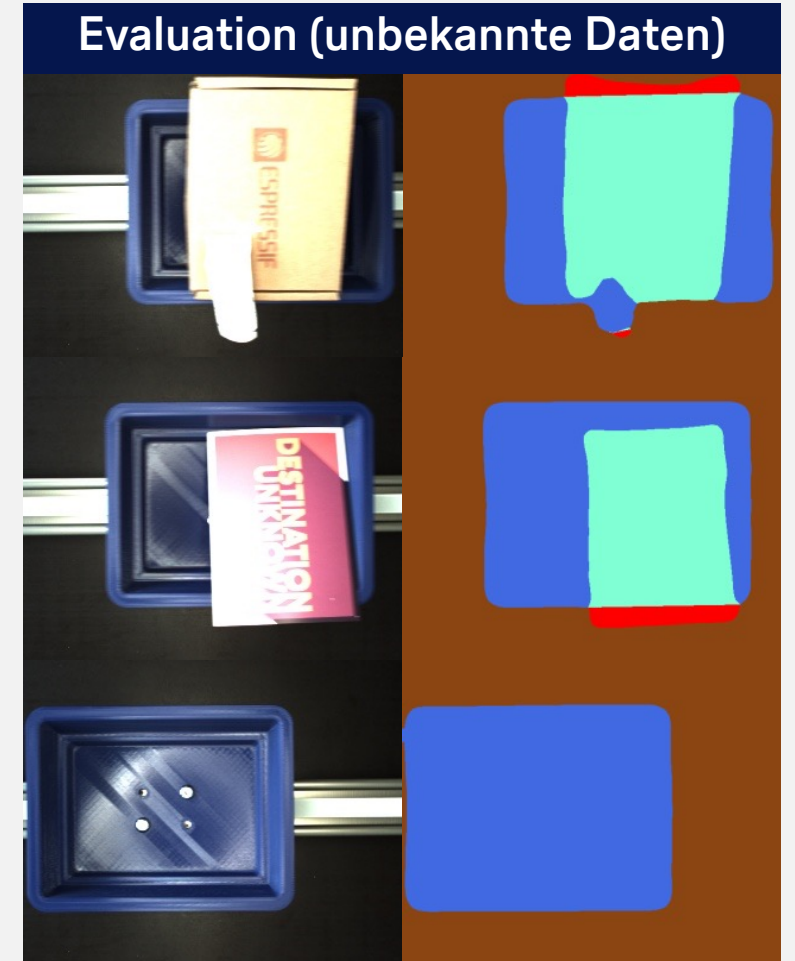
Detailgenaue
Segmentierung

Vorgehensweise

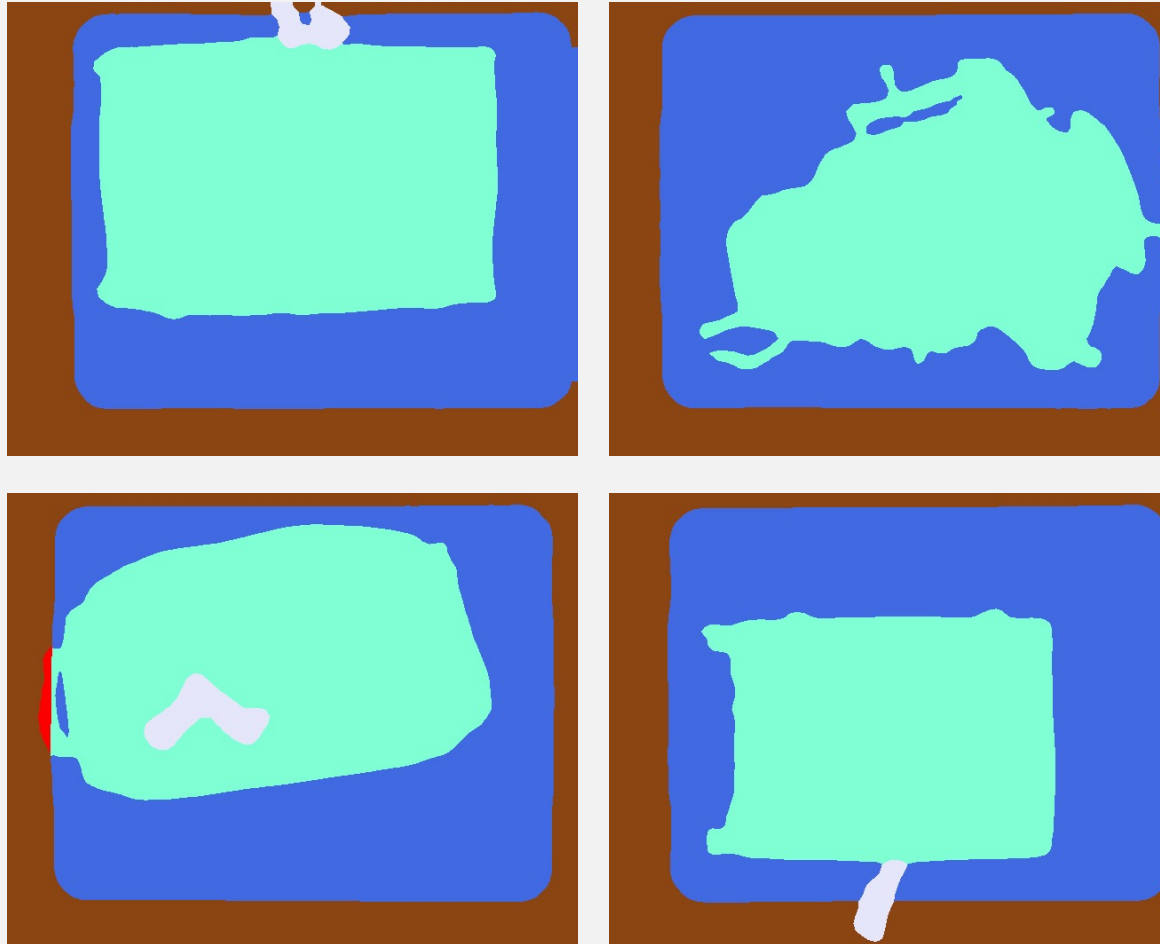


- Jetpack + Containerisierung
- Semantische Segmentierung mit PyTorch
- Unterschiedliche Architektur/Backbone Ansätze
- Transfer Learning
- GPU optimierte Konversion der Modelle nach TensorRT

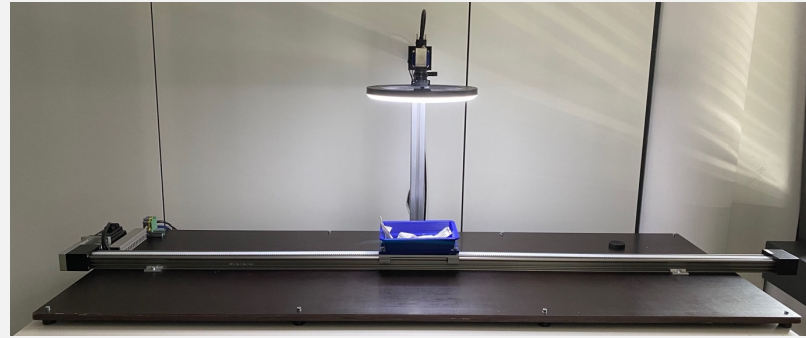
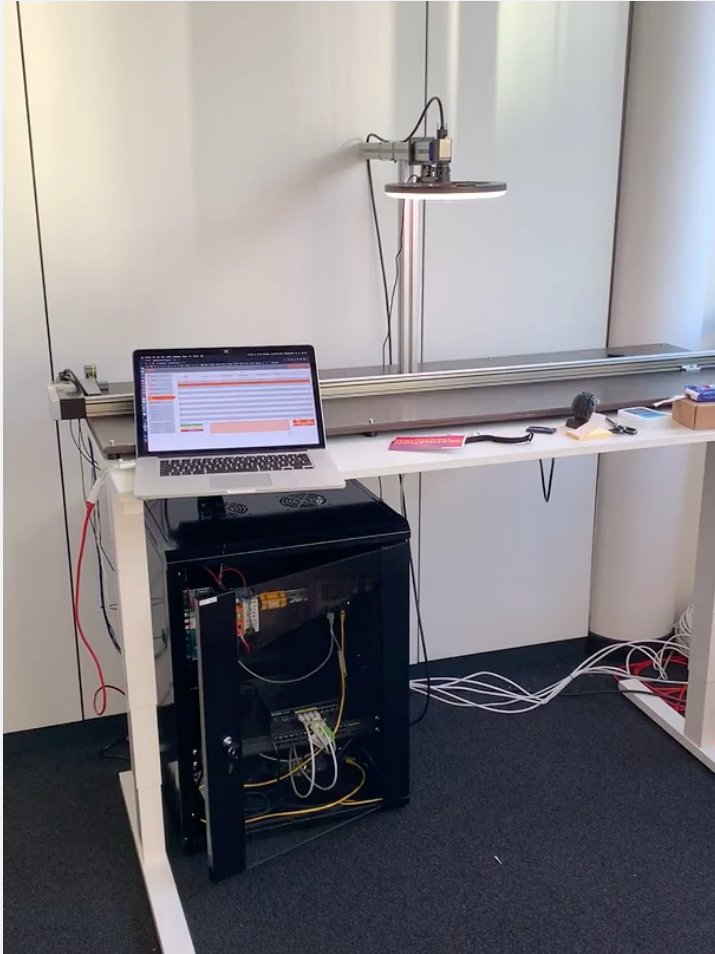
Korrekte Beladung?



Beispiele



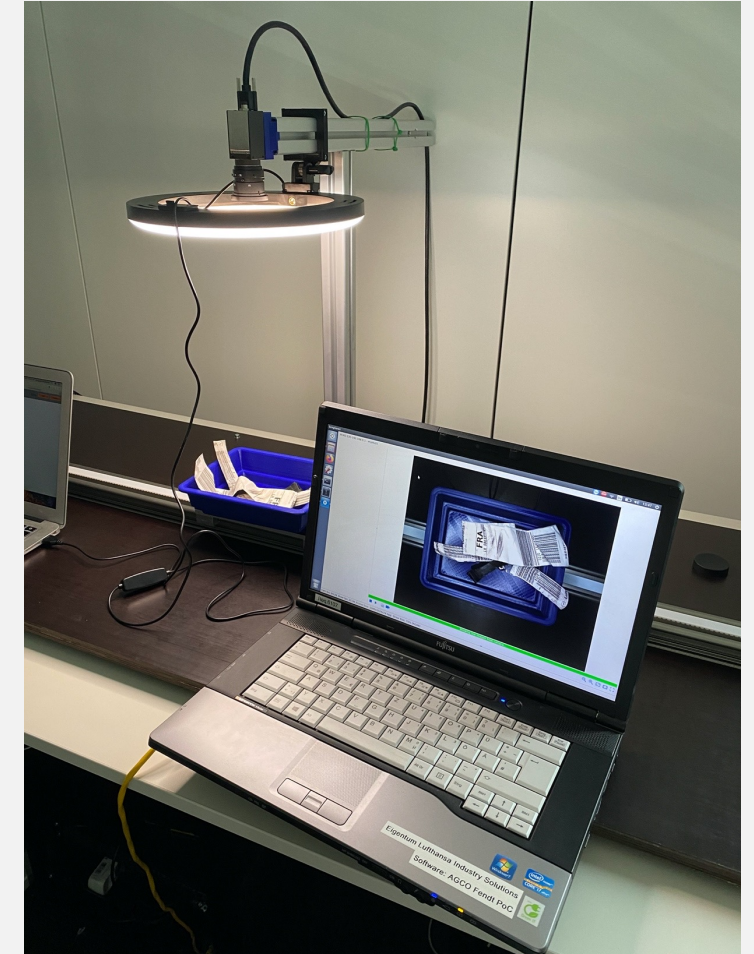
Lessons Learned



Laboraufbau

Stay on edge

Own model,
own
problems



Lessons Learned

HELLO AI WORLD
NVIDIA JETSON

Back | Next | Contents

System Setup

Setting up Jetson with JetPack

note: if your Jetson Nano or Xavier NX has already been setup with the SD card image (which includes JetPack), or your Jetson has already been setup with JetPack, you can skip this step and continue to [Running the Docker Container](#) or [Building the Project](#).

NVIDIA JetPack is a comprehensive SDK for Jetson for both developing and deploying AI and computer vision applications. JetPack simplifies installation of the OS and drivers and contains the following components:

- L4T Kernel / BSP
- CUDA Toolkit
- cuDNN
- TensorRT
- OpenCV
- VisionWorks
- Multimedia API's

Before attempting to use the Docker container or build the repo, make sure that your Jetson has been setup with the latest version of JetPack.

Jetson Nano and Jetson Xavier NX

The recommended install method for the Jetson Nano Developer Kit and Jetson Xavier NX Developer Kit is to use the SD card images.

It comes pre-populated with the JetPack components already installed and can be flashed from a Windows, Mac, or Linux PC. If you haven't already, follow the Getting Started guide for your respective Jetson to flash the SD card image and setup your device:

- [Getting Started with Jetson Nano Developer Kit](#)
- [Getting Started with Jetson Nano 2GB Developer Kit](#)
- [Jetson Xavier NX User Guide](#)

Jetson TX1/TX2 and AGX Xavier

Other Jetson's should be flashed by downloading the NVIDIA SDK Manager to a host PC running Ubuntu 16.04 x86_64 or Ubuntu 18.04 x86_64. Connect the Micro-USB or USB-C port to your host PC and enter the device into Recovery Mode.

SD Manager

STEP 01
PREPARE
ENVIRONMENT

PRODUCT CATEGORY
Jetson

STEP 02
SELECT
CONFIGURATION

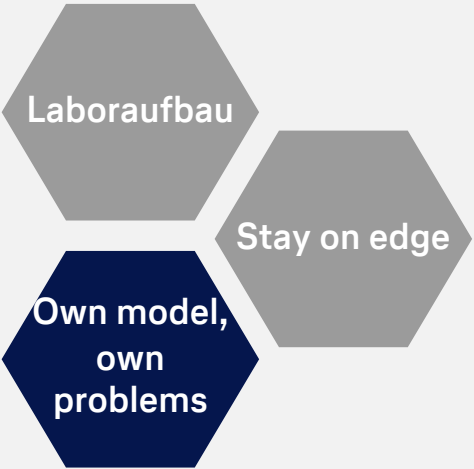
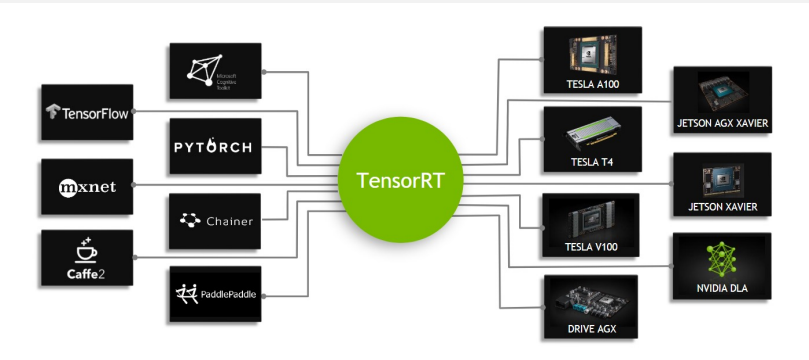
HOST MACHINE
Host Machine

TARGET HARDWARE
Jetson AGX Xavier

STEP 03
FLASH
SYSTEM

TARGET OPERATING SYSTEM
Linux JetPack 4.1

Jetson AGX Xavier (TX2)



dusty-nv / jetson-containers

Watch 10 Star 231 Fork 75

Code Issues Pull requests Actions Projects Wiki Security Insights

master 6 branches 0 tags

Go to file Add file + Code

packages added container for building ROS2 Foxy 8 months ago

scripts updates for L4T R32.5.1 20 days ago

test update PyTorch test scripts 3 months ago

gitignore updated tests 3 months ago

Dockerfile.mtl updated JupyterLab instal 20 days ago

Dockerfile.pytorch updates for torchaudio v0.8.0 17 days ago

Dockerfile.ros.eloquent Use latest l4t-base version: r32.4.4 2 months ago

Dockerfile.ros.foxy Use latest l4t-base version: r32.4.4 2 months ago

Dockerfile.ros.melodic Use latest l4t-base version: r32.4.4 2 months ago

Dockerfile.ros.noetic Use latest l4t-base version: r32.4.4 2 months ago

Dockerfile.tensorflow updates for JetPack 4.4.1 5 months ago

LICENSE.md updated repo 11 months ago

README.md Corrected typo in comments for test ROS commands 2 months ago

About

Machine Learning Containers for NVIDIA Jetson and JetPack L4T

[docker](#) [dockerfiles](#)

[tensorflow](#) [ros](#) [catkin](#) [jetson](#) [pytorch](#) [mxnet](#) [caffe2](#) [chainer](#) [paddle](#) [paddle](#) [paddle](#)

Readme MIT License

Releases

No releases published

Packages

No packages published

Contributors

[dusty-nv](#) [Dustin Franklin](#) [f-flo](#) [dingo02](#)

Languages

Shell 91.0% Python 32.0% CMake 15.0%

Machine Learning Containers for Jetson and JetPack

Hosted on NVIDIA GPU Cloud (NGC) are the following Docker container images for machine learning on Jetson:

- [l4t-mel](#)
- [l4t-pytorch](#)
- [l4t-tensorflow](#)

Dockerfiles are also provided for the following containers, which can be built for JetPack 4.4 or newer:

- ROS Melodic ([ros:melodic-ras-base-l4t-r32.4.4](#))
- ROS Noetic ([ros:noetic-ras-base-l4t-r32.4.4](#))
- ROS2 Eloquent ([ros:eloquent-ras-base-l4t-r32.4.4](#))
- ROS2 Foxy ([ros:foxy-ras-base-l4t-r32.4.4](#))

Below are the instructions to build and test the containers using the included Dockerfiles.

Docker Default Runtime

To enable access to the CUDA compiler (nvcc) during `docker build` operations, add `"default-runtime": "nvidia"` to your `/etc/docker/daemon.json` configuration file before attempting to build the containers:

```
{  "runtimes": {    "nvidia": {      "path": "nvidia-container-runtime",      "runtimeArgs": []    },  },  "default-runtime": "nvidia"}
```

You will then want to restart the Docker service or reboot your system before proceeding.

Building the Containers

To rebuild the containers from a Jetson device running JetPack 4.4 or newer, first clone this repo:

```
$ git clone https://github.com/dusty-nv/jetson-containers
$ cd jetson-containers
```


ML Containers

To build the ML containers (`l4t-pytorch`, `l4t-tensorflow`, `l4t-mel`), use `scripts/docker_build_all.sh` - along

8

25.03.21

Confidentiality: Public | © by Lufthansa Industry Solutions

 **Lufthansa
Industry Solutions**

Vielen Dank für die Aufmerksamkeit



Sebastian Seck

Data Scientist

+49 (0)151 586 22633

sebastian.seck@lhind.dlh.de

www.Lufthansa-Industry-Solutions.com