

Praxisphasen Bericht

Automatisierte Erkennung von Klarschrift auf unterschiedlichen Containertypen

Jan Wille 1535115

26. April 2022

Professor(in)/Lehrbeauftragte(r): Prof. Dr.-Ing. Hanno Homann

Selbstständigkeitserklärung

Hiermit bestätige ich, dass die folgende Arbeit eigenständig von mir allein erstellt und unter Berücksichtigung der zur Verfügung gestellten Aufgabenstellung sowie dem Arbeitsmaterial unter Angabe aller verwendeten Quellen erarbeitet wurde. Die Regelungen und Konsequenzen eines Plagiats, inklusive disziplinarischer Maßnahmen, sind mir bewusst. Insbesondere wurden alle Zitate und gedanklichen Übernahmen als solche kenntlich gemacht.

Jan Wille

Inhaltsverzeichnis

1 Problemstellung	1
1.1 Container Typen	1
1.2 Hardware Voraussetzungen	1
1.3 Anforderungen an das System	2
2 Erstellen von simulierten Testbildern mittels Blender	3
3 Umsetzung in Python	4
4 Abschlusstest	5
Abbildungsverzeichnis	6
Tabellenverzeichnis	7

1 Problemstellung

Eine Reihe von Container (im folgenden auch als *Gebinde* bezeichnet) soll von einer Umladehalle an ihren Lagerort gebracht werden. Dabei soll vor dem Aufnehmen auf ein Stapelfahrzeug sichergestellt werden, dass es sich tatsächlich um das richtige Gebinde handelt.

Dazu sind alle Gebinde mit einem oder mehreren Codes in Klarschrift beschriftet, welche erkannt und verifiziert werden müssen. Wie genau diese Erkennung stattfinden ist im folgenden Dokumentiert.

1.1 Container Typen

In der Lagerhalle werden zwei unterschiedliche Arten von Containern eingelagert. Rechteckige Container werden direkt vom Stapelfahrzeug aufgenommen und haben nur einen einzigen zu lesenden Code. Die dem Erkennungssystem zugewandte Seite hat dabei eine Höhe von 1,7 m und eine maximale Breite von 3,2 m, es gibt aber auch schmalere Container. Die Skizze in Abbildung 1.1(a) zeigt dies ebenfalls.

Des Weiteren gibt es noch tonnenförmige Rundgebände. Diese werden auf Transportrahmen befestigt und dann als Gebinde bezeichnet. Ein Gebinde kann aus einem oder zwei Rundcontainer bestehen, welche mit dem Deckel zum Erkennungssystem liegend verladen sind. Jeder Rundcontainer ist einzeln auf dem Deckel mit einem Code versehen, diese können beliebig rotiert sein. Zusätzlich befindet sich noch ein weiterer Code auf dem Transportrahmen. Siehe auch Abbildung 1.1(b).

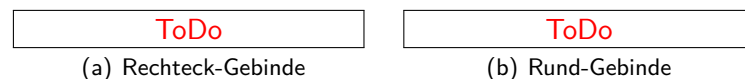


Abb. 1.1: Skizzen der unterschiedlichen Containertypen mit den möglichen Textpositionen (nach [pflichtenheft])

1.2 Hardware Voraussetzungen

Im Pflichtenheft ([pflichtenheft]) ist die bestellte Hardware aufgelistet. Für die hier behandelte Aufgabe sind aber nur einige wenige Komponenten relevant. Die genauen Bezeichnungen sind in Tabelle 1.1 aufgelistet, im Folgenden wird aber nur noch von *Kamera* und *Panel-PC* gesprochen.

Tab. 1.1: Für die Aufgabe relevante Hardware laut Pflichtenheft ([pflichtenheft])

01	SIMATIC MV440 UR optischer Leser; Auflösung: 1600×1200 Pixel; für 1D/2D-Codelesen, Texterkennung und Objekterkennung; Bildfeld und Abstand: variabel; PoE; IP67 (mit 6GF3440-8AC1X); Lieferung: Lesegerät, CD-ROM und Kunststoff-Schutz-Tubus; (ohne: Kabel, Leuchte, Objektiv, Lizenzen)
02	MV440 Montageplatte Lesegerät; Material: Edelstahl (4 mm), vielfältig anwendbar, Befestigung: metrische Gewinde, Fotogewinde (1/4-Zoll, 2x mittig) BxHxT (mm) 80×80×60
03	Mini-Objektiv 6 mm, 1: 1,4 PENTAX C60636KP mit fester Brennweite, Blende und Fokus einstellbar D = 32 mm, L = 37,5 mm
04	D65-Objektiv-Schutz Metall-Glas, Schutzart IP67 für MV440-Geräte; Frontscheibe: Glas, Gehäuse: Metall enthält: Tubus, O-Ring und Schutzkappen (M12, M12, M16), Innen-Durchmesser: 57 mm max. Objektivlänge: 57 mm geeignete Objektive (MLFB): z.B. 6GF9001-1BL01,...-1BF01, ...-1BG01,...-1BH01,...-1BJ01 geeignete Leuchten (MLFB): 6GF3440-8DA1,...-8DA2,...-8DA11 BxHxT (mm) 65×65×60
10	SIMATIC IPC277E (Nanopanel PC); 7" Touch TFT; 2x 10/100/1000 MBit/s Ethernet RJ45; 1x Display-Port Grafik; 1x USB 3.0; 2x USB 2.0; 1x seriell (COM 1); CFAST-Slot; DC 24V Stromversorgung Celeron N2807 (2C/2T) 4 GB RAM WIN Embedded Standard 7 P SP1, englisch; 64 Bit 80 GB SSD ohne SIMATIC Software

1.3 Anforderungen an das System

Das zu erstellende System soll auf Kommando des Stapelfahrzeug-Fahrers das vor dem Fahrzeug befindliche Gebinde scannen und die gefundenen Codes mit einer Liste von für diese Schicht gültigen Codes vergleichen. Dabei soll es keinen Unterschied machen, welcher Containertyp gerade vor dem Fahrzeug steht. Insbesondere die beliebige Rotation der Rundgebilde muss berücksichtigt werden.

Zusätzlich zu der hier behandelten Komponenten kann eine weitere Software vorausgesetzt werden. Diese läuft auf dem Panel-PC und präsentiert dem Fahrer eine Grafische Oberfläche (*GUI*). Am Anfang der Schicht erhält diese eine Liste mit Arbeitsaufträgen für die Schicht inklusive der Codes der zu transportierenden Container. Diese Software stößt den Scanprozess an und erhält das Ergebnis um es dem Fahrer grafisch darzustellen. Sollte der Lesevorgang nur Teilweise oder gar nicht funktionieren, soll der Fahrer so viele Teilinformationen wie möglichen erhalten und wird von der Zusatzsoftware zum manuellen Eingreifen aufgefordert.

Außerdem soll die Möglichkeit bestehen ein Foto des Abstellortes aufzunehmen und zur Archivierung an den erfolgreichen Auftrag anzuhängen.

2 Erstellen von simulierten Testbildern mittels Blender

3 Umsetzung in Python

4 Abschlusstest

Abbildungsverzeichnis

1.1	Skizzen der unterschiedlichen Containertypen mit den möglichen Textpositionen (nach [pflichtenheft])	1
-----	---	---

Tabellenverzeichnis

1.1	Für die Aufgabe relevante Hardware laut Pflichtenheft ([pflichtenheft])	2
-----	--	---