DHBW
Duale Hochschule
Baden-Württemberg
Ravensburg
Campus Friedrichshafen

Automatisierung eines Roboterarms mit kamerabasierter Steuerung durch SPS

Studienarbeit T3_3200

Studiengang Elektrotechnik

Studienrichtung Automation

Duale Hochschule Baden-Württemberg Ravensburg, Campus Friedrichshafen

von

Norman Kübler, Erik Kähm

Abgabedatum:	14. Juli 2025
Bearbeitungszeitraum:	07.04.2025 - 14.07.2025
Matrikelnummer:	5163670, 2607414
Kurs:	TEA22
Betreuerin / Betreuer:	Edvin Hamzic, Thomas Dietmüller

Erklärung

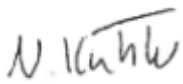
gemäß Ziffer 1.1.14 der Anlage 1 zu §§ 3, 4 und 5 der Studien- und Prüfungsordnung für die Bachelorstudiengänge im Studienbereich Technik der Dualen Hochschule Baden-Württemberg vom 29.09.2017 in der Fassung vom 24.07.2023.

Ich versichere hiermit, dass ich meine Studienarbeit T3 3200 mit dem Thema:

Automatisierung eines Roboterarms mit kamerabasierter Steuerung durch SPS

selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Ich versichere zudem, dass die eingereichte elektronische Fassung mit der gedruckten Fassung übereinstimmt.

Friedrichshafen, den 14. Juli 2025





Norman Kübler Erik Kähm

Inhaltsverzeichnis:

1. Einleitung und Projektüberblick....1
2. Ablauf und Funktionsweise von Station 1 1
2.1. Bauteilzuführung (Startpunkt des Systems)....1
2.2 Abnahme fertiger Bauteile (Endpunkt des Systems)....1
2.3. Demontageprozess (Rückführung) 1
3. Technische Herausforderungen und Randbedingungen 2
Optimale Greifweise 3
4. Der Transportroboter „Husky“....3
5. Ausblick 4
6. Roboterarmprogrammierung Grundlagen: 4
7. Bewegungsgrundlagen des Roboters 8
8. Schreiben eines einfachen Programmes 10
9. Palettierung in Polyscope 11
10. Grundlagen ifm Kamera....14
11. Anbindung an SPS 16
12. Anpassungen und Zwischenfazit....39
12. Fazit und Ausblick für folgende Gruppen und Verbesserungsbedarf an dem Projekt 42
Abbildungsverzeichnis:....43
Quellenangabe....44

1. Einleitung und Projektüberblick

Im Rahmen dieser Studienarbeit wurde der Bereich „Station 1 – Belt Feeding“ innerhalb der Wunschfabrik im PTZ-Labor untersucht und technisch umgesetzt. Die Station stellt einen wesentlichen Teil des Gesamtsystems dar und ist sowohl als Start- als auch als Endpunkt des Materialflusses konzipiert. Ziel der Station ist es, den Vorgang des Auflegens und Abnehmens von Bauteilen auf ein Fließbandsystem zu automatisieren und somit den menschlichen Eingriff auf ein Minimum zu reduzieren.

Für die Umsetzung stehen mehrere technische Komponenten zur Verfügung, welche eine Lösung für das Projekt durch eine Zusammenarbeit untereinander bieten:
Ein Roboterarm der Firma Universal Robots, eine 3D-Kamera von ifm zur Positionsbestimmung sowie eine speicherprogrammierbare Steuerung (SPS) von Siemens zur Koordination und Ablaufsteuerung der Prozesse.

---

2. Ablauf und Funktionsweise von Station 1

Station 1 erfüllt im Gesamtsystem drei zentrale Aufgaben:

2.1. Bauteilzuführung (Startpunkt des Systems)

Zu Beginn befinden sich acht Bauteile auf einer Lagerfläche. Der Roboterarm erkennt die Position eines Bauteils, greift es präzise mithilfe des Greifers und legt es auf eine Bauteilträgerplattform des Fließbandsystems. Diese Plattform transportiert das Bauteil durch verschiedene Stationen, in denen es schrittweise zusammengebaut wird. Der Roboter greift dabei nacheinander die Bauteile in einer programmierten Reihenfolge auf – beginnend mit dem Bauteil an Position 1, anschließend Position 2, usw.

2.2 Abnahme fertiger Bauteile (Endpunkt des Systems)

Am Ende der Förderstrecke befindet sich das nun zusammengesetzte, fertige Bauteil auf einer definierten Übergabeposition. Dort wird es vom Roboterarm wieder aufgenommen und auf die Ladefläche eines mobilen Transportroboters (Husky) abgelegt. Dieser Roboter übernimmt den Weitertransport in ein externes Gebäude (Ritz-Gebäude), wo weitere Bearbeitungsschritte erfolgen.

2.3. Demontageprozess (Rückführung)

Neben der Montage wird auch ein Rückführungsprozess abgebildet. Dabei werden fertige Bauteile, die vom Husky-Roboter zurückgeführt wurden, erneut auf das Fließband aufgelegt und Schritt für Schritt demontiert. Diese Funktion dient z. B. zur Simulation von Rückläufen oder Wiederverwendung von Komponenten. In Abbildung 1 wird ein schematischer Ablaufplan abgebildet, von welcher Position ein Bauteil im Prozess überführt wird. Das Bauteil ist in der Farbe Grün veranschaulicht.

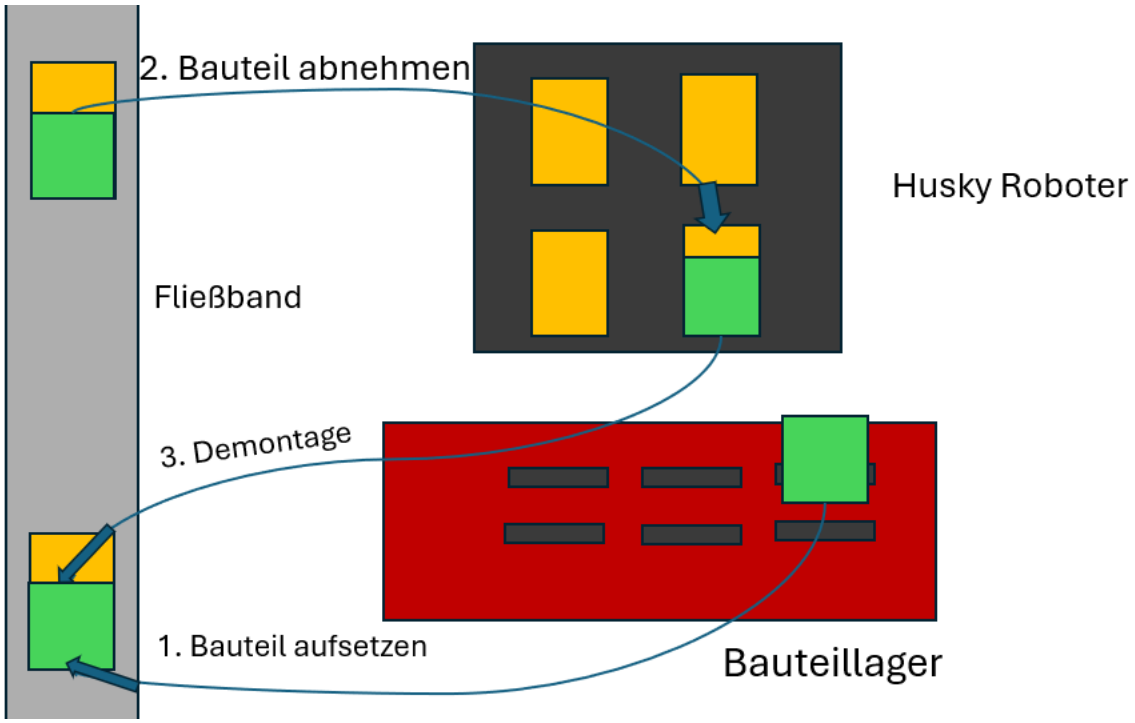


Abbildung 1: Schematischer Ablaufplan

---

3. Technische Herausforderungen und Randbedingungen

Die Automatisierung der Bewegungsabläufe stellt hohe Anforderungen an Präzision, Wiederholgenauigkeit und Kollisionserkennung:

- Der Greifvorgang muss millimetergenau erfolgen, insbesondere beim Entnehmen aus dem Lager, um ein Verkanten oder Beschädigen der oft aus Kunststoff bestehenden Bauteile zu verhindern.
- Die Greifstrategie ist entscheidend: Die Bauteile dürfen nur auf bestimmte Weise aufgenommen werden, da sonst die spätere Positionierung auf dem Fließband nicht exakt erfolgen kann.
- Die Bewegungsfreiheit des Roboterarms ist physikalisch begrenzt. Manche Positionen sind aufgrund von Reichweitenbeschränkungen oder Singularitäten nicht anfahrbar. Von zehn möglichen Ablageplätzen können daher nur acht effektiv genutzt werden.
- Kollisionen mit der Umgebung (z. B. PC, Gerüst oder Kabel) müssen durch exakte Bahnplanung und Hinderniserkennung vermieden werden. Ein Zusammenstoß würde den Prozess abbrechen und in manchen Fällen die Stromzufuhr trennen, etwa durch gelöste Kabelverbindungen am Greifer. Die Ermittlung der Fehlerquelle kann dabei sei zeitintensiv sein.

Optimale Greifweise

Der Greifer muss das Bauteil an den Seiten des abstehenden Blocks greifen. Die Hauptfläche, also die größte Fläche des Bauteils, muss frei bleiben und darf von dem Greifer nicht bedeckt werden, da diese Fläche zum Aufsetzen auf die Bauteilträger des Fließbandes und des Transportroboters dienen. Die Hand dient in Abbildung sinnbildlich als Greifer. Dadurch muss das Bauteil nur um 90° gedreht werden, um mit der Hauptfläche auf ein Bauteilträger abgelegt zu werden. Der Greifer fährt die erste Reihe nach rechts ab und anschließend die zweite Reihe (In der Abbildung oben).



Abbildung 2: richtige Greifweise

4. Der Transportroboter „Husky“

Der Husky ist ein autonomer mobiler Roboter, der den Materialfluss zwischen der Wunschfabrik (PTZ-Gebäude) und dem weiterführenden Bearbeitungsbereich im Ritz-Gebäude übernimmt. Die Strecke beträgt etwa 100 Meter.
Der Roboter fährt auf vier Rädern und besitzt eine flache Auflagefläche für derzeit bis zu sechs Bauteilträger, Anzahl ausbaubar. Er befindet sich innerhalb eines mit gelbem Klebeband markierten Bereichs unter der befindlichen 3D-Kamera.

Da der Husky-Roboter nicht exakt positioniert ist, bzw. Ziel ist es das keine exakt festgelegte Position erforderlich ist. Dafür wurde eine Lösung zur toleranzbasierten Positionsbestimmung entwickelt. Eine an einem Item-Profil fest montierte 3D-Kamera von ifm erkennt die Lage und Ausrichtung des Roboters relativ zur Ablageposition. Auf eine starre mechanische Positionierung (z. B. Einrasten in eine Schiene) wurde bewusst verzichtet, um die Flexibilität zu erhöhen. Die 3D-Kamera erlaubt es dem System, eine gewisse

Ungenauigkeit in der Position des Huskys zu kompensieren und dennoch zuverlässig Bauteile auf- und abzuladen.

5. Ausblick

Station 1 bildet als automatisierte Schnittstelle zwischen Lager, Fließband und Transportroboter eine zentrale Funktionseinheit innerhalb des Gesamtsystems. Durch den gezielten Einsatz von Robotertechnik, 3D-Bilderkennung und SPS-Steuerung wurde ein beachtlicher Schritt für einen hohen Automatisierungsgrad erreicht. Lediglich die Bauteile für den Ablagebereich beim ersten Prozess müssen manuell per Hand zugeführt werden. Zukünftige Arbeiten können auf dieser Grundlage aufbauen, um z. B. den Bauteilfluss zu optimieren, den Greifer zu verbessern oder den Transportprozess weiter zu flexibilisieren.

6. Roboterarmprogrammierung Grundlagen:

Die Programmierung des Roboterarmes bedarf einer eigenen Programmierweise des Hersteller Universal Robots. Die Programmierung erfolgt über Polyscope. Anstelle eines PCs kann dieser Roboterarm über ein sogenanntes Teach Pendant erfolgen und wurde für die Programmierung des Roboterarmes genutzt. Das Teach Pendant ist ein handliches Gerät mit einer GUI (Graphical User Interface) mit dessen Mitteln der Programmierprozess anderen Programmiersprachen gegenüber einfach erfolgt und der Roboterarm angesteuert werden kann. Dies wird ermöglicht durch die Abstraktion des Codes in Form von Programmierbausteinen, welche ähnlich wie beim TIA Portal mit dem Funktionsplan (FUP), das Schreiben von Code weitestgehend vermeiden. Zunächst soll ein kleines Tutorial gegeben werden, wie man ein einfaches Roboterprogramm ansteuert und was zu Beginn zu beachten ist, sodass man mit den mitgegebenen Kenntnissen weiß, wie am Anfang vorgegangen werden soll.



Abbildung 3: Teach Pendant

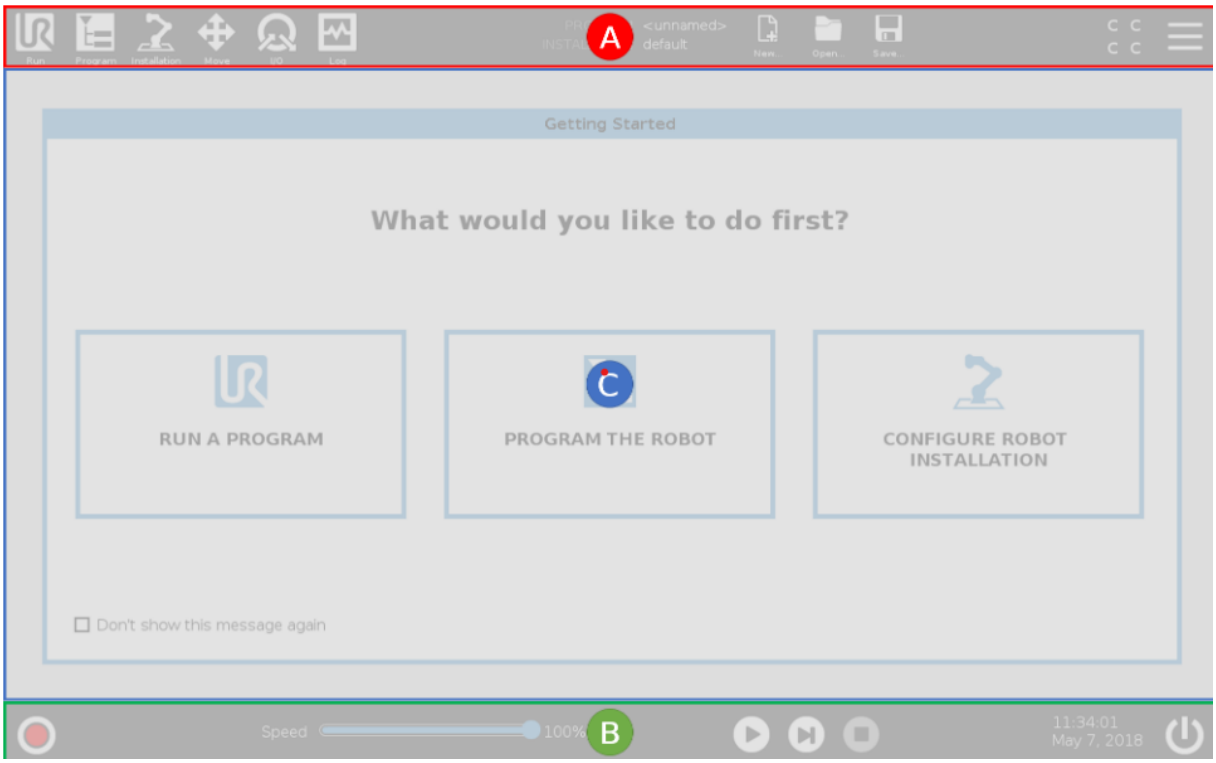


Abbildung 4: Startoberfläche

Wenn die Stromversorgung richtig gelegt ist, kann man über den Anschaltknopf das Gerät einschalten und das Teach Pendant wird hochgeladen. Anschließend werden drei Aktionsoptionen angezeigt, die man durchführen kann. Z.B. ein bestehendes Programm hochladen, den Roboter neu programmieren oder den Roboter konfigurieren. Dieses Textfeld dient zur Hilfe und kann ignoriert werden. Über die rot markierte Leiste kann genauso direkt

auf Programme zugegriffen oder ein neues Programm erstellt werden. Über das obere linke Feld 

werden alle Programme und Ordner aufgelistet, die gespeichert wurden und auf die man zugreifen kann.

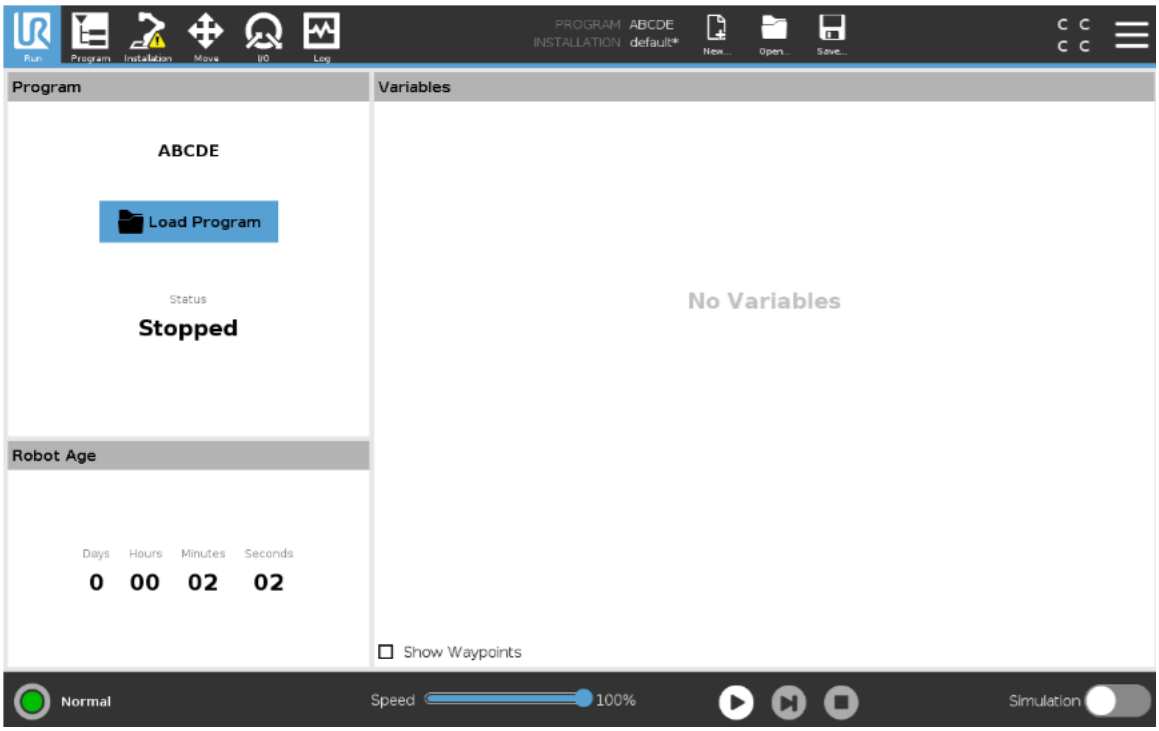


Abbildung 5: Programm laden

Über das Feld Programm 

kann man ein neues Programm erstellen. Hierbei wird man von einer Leiste an Programmierfunktionen unterstützt, die das Programmieren erleichtern.

Über den Button Installation 

werden die Roboterarmeinstellungen konfiguriert, sowie die externen Geräte, die genutzt werden. Ein Beispiel für ein externes Gerät wäre in unserem Fall der Greif, welcher an der letzten Achse des Roboters montiert ist.

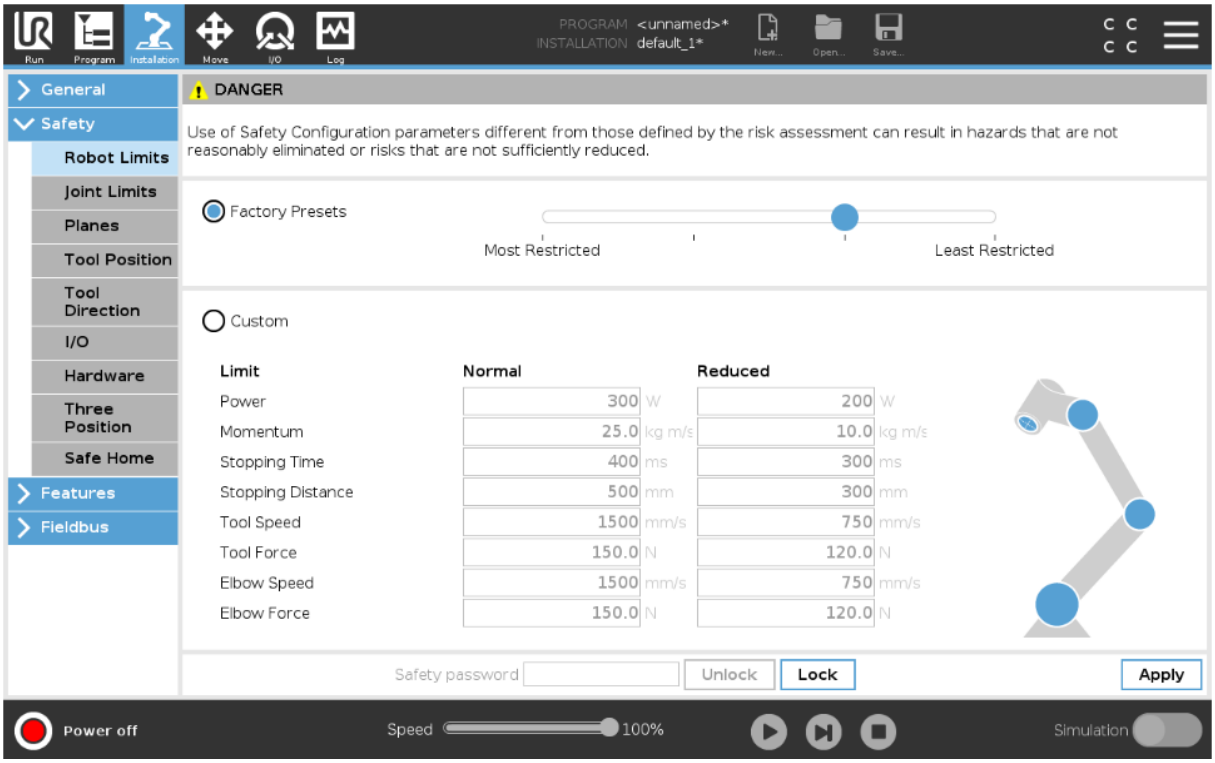


Abbildung 6: Robotereinstellungen

Es werden unter anderem roboterspezifische Daten angezeigt, wie z.B. Leistung, Moment, Geschwindigkeit oder Kraft. Eine Veränderung der Werte bedarf eines Passworts. Ein Eingriff in die Werte war für unsere Anwendung jedoch nicht notwendig.



In der I/O Leiste sind alle Inputs und Outputs aufgelistet, mit denen man bestimmte Logikoperationen umsetzen kann. Enthalten sind z.B. die Variablen für die Spannungszufuhr des Greifers, und die booleschen Operanden für das Ein und Ausschalten des Greifers, welche für die Programmumsetzung wichtig sind.

Mit dem Button „Neu“ 

öffnet man ein neues leeres Programm.

Mit „Speichern“ 

wird das Programm gespeichert. Man kann bei der Speicherung unterscheiden, ob man nur das Programm oder auch die Konfigurationen mitspeichern möchte. Regelmäßiges speichern, ist wichtig den Fortschritt zu behalten, im Falle eines Systemabsturzes oder das Zwischenspeichern, zur Vermeidung von Fehlern.

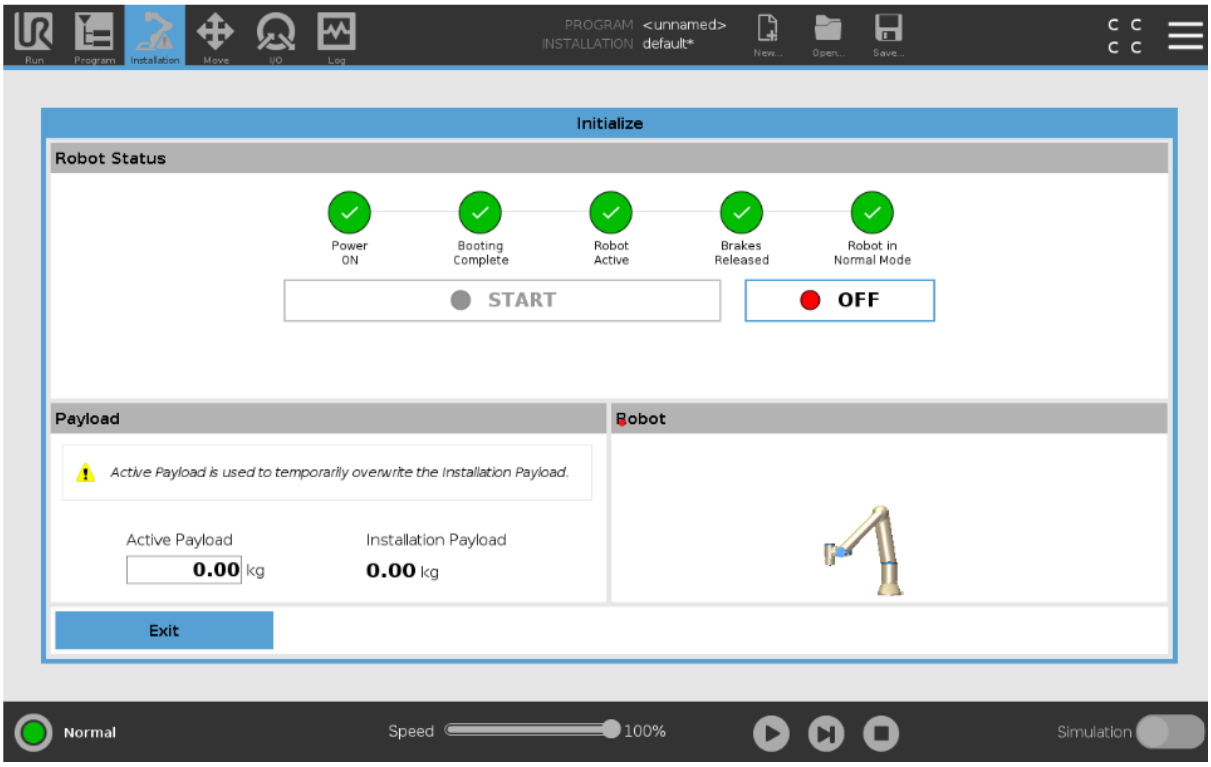


Abbildung 7: Roboterarm aktivieren

Um den Roboter bewegungsfähig zu machen, müssen zuerst ein paar wenige Klicks zur Sicherheit unternommen werden. Man klicke dafür auf den unteren Button und öffnet mit Anschalten die Stromversorgung für den Roboter. Mit Start wird die Bremse des Roboters deaktiviert. Sobald der Button grün leuchtet, kann man beginnen.

7. Bewegungsgrundlagen des Roboters

Bevor man sich an das Schreiben eines Programmes setze, soll zunächst erläutert werden, wie man manuell den Roboterarm bewegen kann. Dieses Verständnis wird das Schreiben von Programmen erleichtern sowie das Erreichen bestimmter Posen.

Dafür klickt man den Button „Bewegen“ 

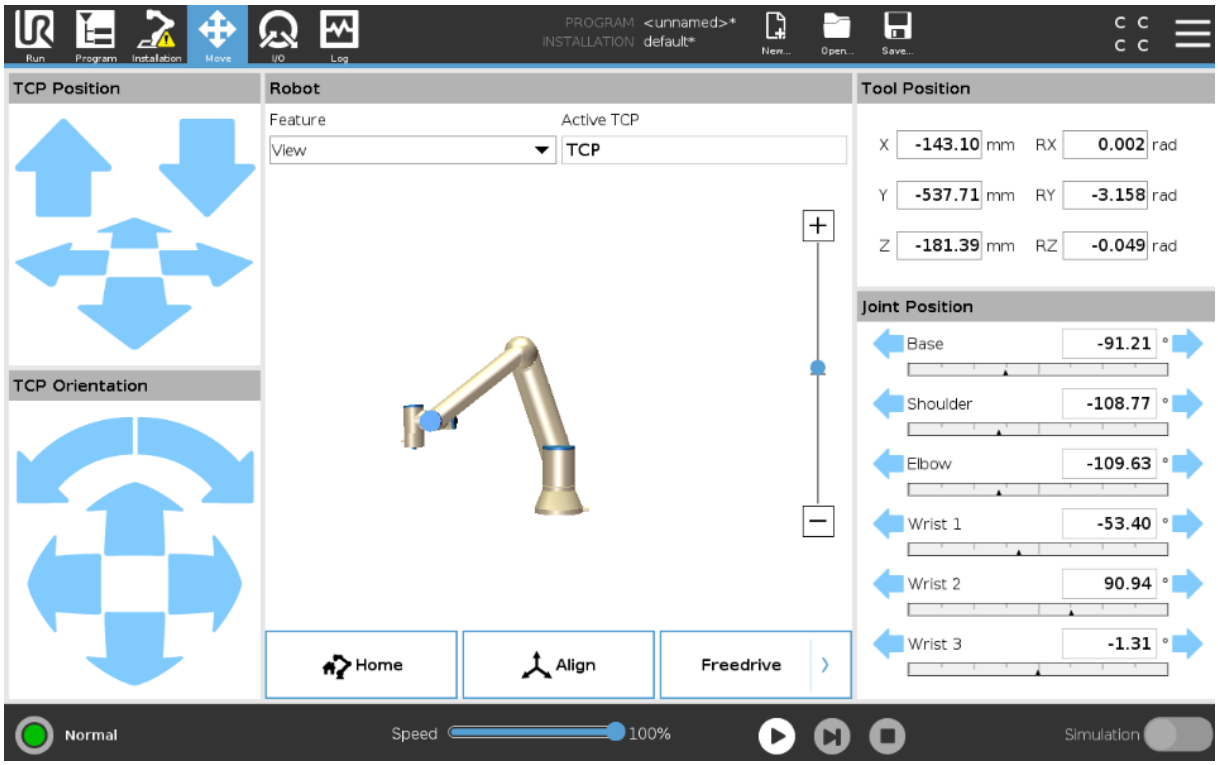


Abbildung 8: Manuelle Bewegung

Auf der linken Seite kann man manuell die Koordinaten des TCP verändern. Mit TCP ist der Tool Center Point gemeint. Der TCP beschreibt einen festgelegten Nullpunkt am Ende des Roboters. In unserem Fall ist der TCP der Greifer. Mit TCP Position fährt der TCP linear an, also gerade in die entsprechende X,Y oder Z Richtung (Translation). Mit TCP Orientation verschiebt man den TCP um die X,Y oder Z Achse (Rotation). Rechts stehen die Werte des TCP mit Abweichung zu dem festgelegten Koordinatensystem.

Mit „Freedrive“ kann man den Roboterarm manuell per Hand bewegen. Dafür muss man am Teach Pendant den Knopf auf der Rückseite immer gedrückt halten und kann den Roboterarm nun beliebig bewegen.



Abbildung 9: Relative Geschwindigkeit einstellen

Mit dem Geschwindigkeits-Slider kann man einstellen, wie schnell sich der Roboterarm fortbewegen soll. 100% entspricht demnach der Maximalgeschwindigkeit. Beim Testen eines Programmes macht es Sinn mit einer geringeren Geschwindigkeit den Roboter anfahren zu lassen, dadurch wirken sich Kollisionen weniger schädigend aus. Im endgültigen Betrieb kann der Roboterarm für maximalen Durchsatz auf 100% eingestellt werden, wenn sichergestellt ist, dass sich der Roboter nach Wunschvorstellung bewegt.

8. Schreiben eines einfachen Programmes

In der Abbildung sieht man folgende Programmumgebung:

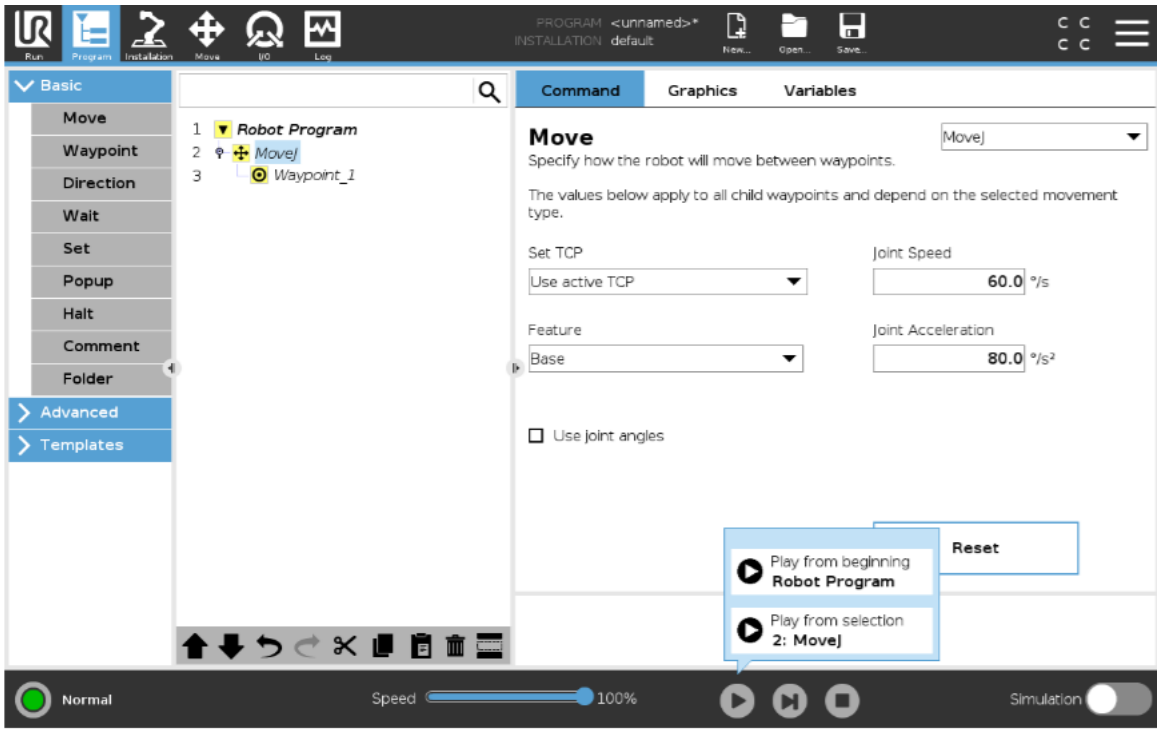


Abbildung 10: Programmumgebung

Zunächst erstellt man ein neues Roboterprogramm. Dazu klickt man oben im Menü auf „Programm“ und wählt anschließend „Neu“, um ein neues leeres Projekt zu starten. In der Programmübersicht auf der linken Seite erscheint nun standardmäßig der Haupteintrag „Roboterprogramm“. Dies ist der übergeordnete Programmknoten, unter dem alle Befehle eingefügt werden.

Im nächsten Schritt fügt man einen Bewegungsbefehl hinzu. Dazu klickt man im linken Menübereich auf „Bewegung“ (unter dem Reiter Basis) und wählt beispielsweise den Befehl BewegungJ aus. Dieser entspricht einer gelenkwinkelbasierten Bewegung (Joint-Bewegung), bei der der Roboterarm über die kürzeste Strecke durch die einzelnen Gelenkstellungen zu einer Zielposition fährt. Der hinzugefügte Befehl erscheint nun unter dem Programmknoten.

Innerhalb des Bewegungsbefehls kann man nun einen oder mehrere sogenannte Wegpunkte definieren. Diese Wegpunkte beschreiben die Zielpositionen, die der Roboter anfahren soll. Um einen Wegpunkt zu setzen, bringt man den Roboterarm manuell oder über die Teach-Funktion in die gewünschte Position. Anschließend klickt man im Programmbaum auf „Wegpunkt hinzufügen“ und danach auf „Diese Position setzen“. Damit speichert man die aktuelle Pose des Roboters als Wegpunkt ab.

Auf der rechten Seite erscheinen nun die Parameter des Bewegungsbefehls. Dort kann man unter anderem den aktiven Werkzeugmittelpunkt (TCP) einstellen. Standardmäßig ist „Aktiven TCP verwenden“ ausgewählt, was in den meisten Fällen korrekt ist. Unter dem

Punkt Bezugssystem wählt man typischerweise „Basis“, um Bewegungen relativ zum festen Roboterfußpunkt durchzuführen. Außerdem lassen sich die Gelenkgeschwindigkeit (z. B. 60 °/s) und die Gelenkbeschleunigung (z. B. 80 °/s²) individuell anpassen, je nachdem wie schnell oder sanft der Roboter sich bewegen soll.

Es besteht zudem die Möglichkeit, die Option „Gelenkwinkel verwenden“ zu aktivieren. Dies führt dazu, dass die Position ausschließlich über exakte Gelenkwinkel definiert wird – praktisch bei präzisen Wiederholungen oder beim Einsatz mehrerer Roboter mit identischen Achsstellungen.

Ist das Programm aufgebaut, kann es getestet werden. Dazu verwendet man unten in der Bedienoberfläche entweder „Ab Start abspielen“, um das gesamte Programm vom Beginn an auszuführen, oder „Ab Auswahl abspielen“, um ab einem bestimmten Programmschritt zu beginnen – beispielsweise direkt beim Bewegungsbefehl.

Mit diesen wenigen Schritten ist ein funktionaler Bewegungsablauf erstellt: Der Roboter fährt aus seiner Startposition zu einem zuvor gespeicherten Punkt im Raum. Dieses Grundgerüst bildet die Grundlage für komplexere Anwendungen wie Pick-and-Place, Palettierung oder Montageprozesse. Durch das Hinzufügen weiterer Wegpunkte, IO-Signale, Schleifen oder Bedingungen lässt sich das Programm jederzeit erweitern und an neue Anforderungen anpassen.

9. Palettierung in Polyscope

Bei der Umsetzung der drei Roboterarmbewegungen kam der Einsatz einer Palettierungsfunktion sehr nützlich. Natürlich könnte man dies auch mit einfachen Programmzeilen selbst konstruieren, dies wäre jedoch ein um einiges größerer Aufwand und um leichter mit der in Polyscope enthaltenen Palettierungsfunktion umzusetzen, die im Folgenden vorgestellt werden soll.

Im Kontext von Palettierung in PolyScope (also dem Programmiersystem von Universal Robots) bezeichnet eine Palette eine strukturierte Anordnung von Ablage- oder Aufnahmepositionen für Bauteile. Sie stellt ein Raster oder Gitter dar, das angibt, wo genau der Roboter ein Objekt ablegen oder aufnehmen soll, z. B. auf einer Palette im echten physischen Sinne (z. B. Europalette), einem Gitterfach, einem Sortierfeld oder ähnlichem.

„Palettieren“ ist eine Vorlage (bzw. ein Programmmuster), mit der sich Palettier- und Depalettieraufgaben einfach programmieren lassen – also das Aufnehmen und Ablegen von Teilen, z. B. aus Trays, Vorrichtungen usw. Dabei führt der Roboter wiederholbare Bewegungsabläufe für verschiedene Objekte in mehreren Schichten mit unterschiedlichen Anordnungen aus.

In unserem Projektbereich gibt es demnach insgesamt zwei Paletten. Die Palette aus dem ersten Programm sind die festmontierten Löcher, auf denen zu Beginn die Bauteile aufgesetzt werden.

Die zweite Palette sind die sechs montierten Trägerflächen auf dem Husky. Beide Paletten sind als Gitter zu klassifizieren mit einer gewissen Anzahl an Spalten und Reihen. So ist erste Palette ein 2x4 Gitter, und die zweite Palette am Husky ein 2x3 Gitter.

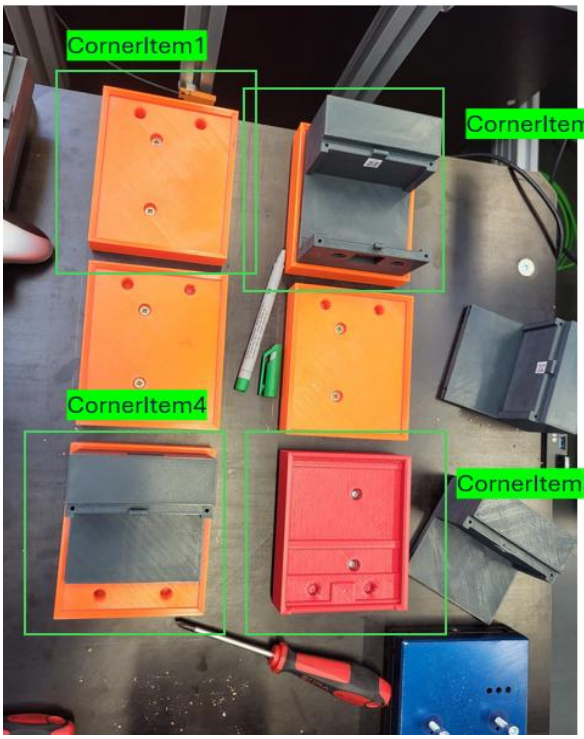


Abbildung 11: Palette 2

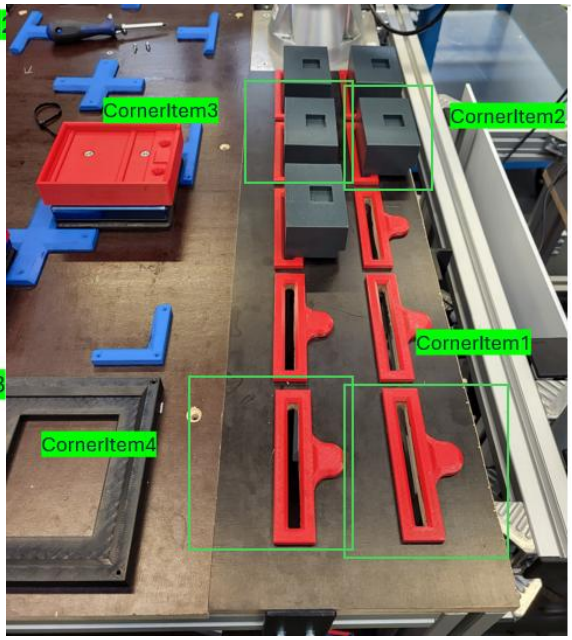


Abbildung 12: Palette 1

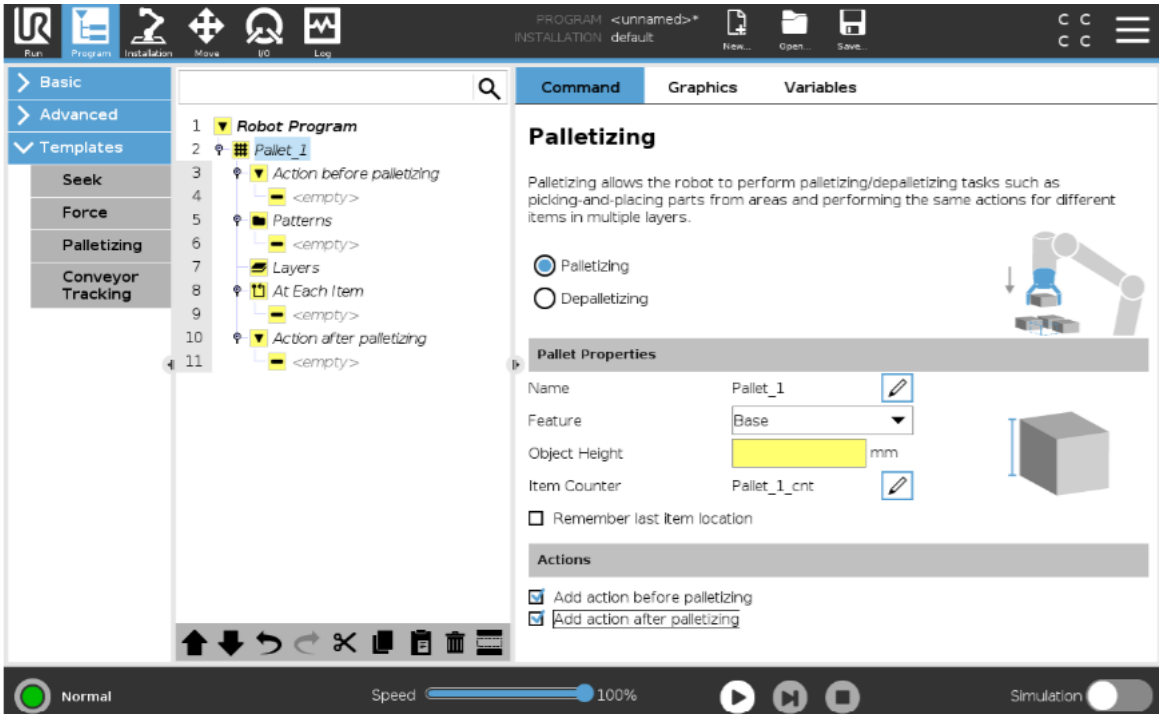
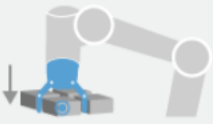
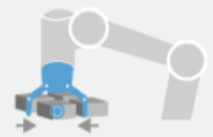


Abbildung 13: Palettierungsfunktion

	**Grid** To teach the positions, select each item in the Program Tree: • CornerItem1 • CornerItem2 • CornerItem3 • CornerItem4 Insert the number of rows and columns in the appropriate text boxes to set the dimensions of the pattern.

Abbildung 14: Gitterstruktur auswählen

Man habe anschließend die Auswahl zwischen einer Reihe, einem Gitter und einem irregulären Pattern zu wählen. In unserem Beispiel handelt es sich um eine Gitterstruktur. Es wurde hierbei darauf geachtet, dass der Abstand zwischen den Trägern so gleich wie möglich ist, da man sonst ein ungleichmäßiges Pattern hat. Weil die Abstände gleich sind, muss man nicht für jeden Träger die Position abfahren und einstellen, dass übernimmt das Programm. Man muss jedoch die Eckpunkte händisch anfahren und festlegen, das sind die sogenannten Corner-Items. Bei vier Eckpunkte, müssen dementsprechend alle vier Eckpunkte angefahren werden.

 ToolActionPoint	Der **ToolActionPoint**-Wegpunkt ist die Position, an der der Roboter eine Aktion an einem Objekt in einer Lage ausführt, z. B. das Greifen oder Ablegen. Standardmäßig entspricht er dem Referenzpunkt, kann aber im Programmbaum angepasst werden. Der Referenzpunkt wird beim Einsatz des Assistenten automatisch auf die erste Position der ersten Lage gesetzt und dient zur Definition der Anfahr-, Aktions- und Abfahrpunkte für jedes Objekt in einer Lage.
 Approach	Der **Approach**-Wegpunkt ist die kollisionsfreie Position und Ausrichtung, aus der sich der Roboter einem Objekt in einer Lage nähert. Also eine lineare Annäherungsbewegung
 Tool Action	**Tool Action** bezeichnet die Aktion, die man ausführen möchte, z.B. den Greifer öffnen oder schließen

 Exit	**Exit** – Endposition, die man erreichen möchte, um das Item der Palette zu verlassen. In der Regel wie eine Umkehrung des Approach-Wegpunktes

10. Grundlagen ifm Kamera

Die ifm Kamera 03D304 ist eine 3D-Sensorik-Kamera, die mithilfe der sogenannten Time-of-Flight-Technologie dreidimensionale Umgebungen erfassen kann. Sie sendet infrarotes Licht aus und misst die Zeit, die das Licht braucht, um von Objekten reflektiert wieder zurück zur Kamera zu gelangen. Aus diesen Informationen wird für jeden Punkt im Bild ein präziser Abstandswert berechnet. So entsteht ein detailliertes Tiefenbild, mit dem sich nicht nur die Position, sondern auch die Form und Ausrichtung von Objekten erkennen lassen.



Abbildung 15: ifm 3D Sensor Kamera (O3D304)

Im Rahmen unseres Projekts kommt diese Kamera zum Einsatz, um die Trageflächen des Transportroboters Husky zu erkennen. Da der Husky-Roboter die fertig bearbeiteten Bauteile zwischen zwei Anlagen transportiert, ist es wichtig, dass der Roboterarm die genaue Position der Ablageflächen auf dem Husky erfassen kann – unabhängig davon, wie genau der Husky steht. Die ifm Kamera ermöglicht es uns, diese Positionen automatisch zu erkennen, ohne dass eine millimetergenaue Ausrichtung des Transportroboters notwendig ist.

Die Kamera erfasst also die räumliche Lage der Ablageflächen und berechnet aus dem 3D-Bild die X-, Y- und Z-Koordinaten der relevanten Punkte sowie deren Orientierung im Raum (also Neigung und Verdrehung; RX, RY, RZ). Der Abstand ist visuell auf der Software durch eine Veränderung des Farbspektrums zu erkennen. So erscheinen weitergelegene Objekte blauverschoben und nähere sind rotverschoben. Diese Werte werden anschließend an die SPS (speicherprogrammierbare Steuerung) übermittelt, die sie verarbeitet und an den Roboterarm weiterleitet. So weiß der Roboterarm genau, wo er das Bauteil ablegen oder aufnehmen muss – selbst wenn sich die Position des Transportroboters leicht verschoben hat. Näheres zu der Anbindung an die SPS wird in Kapitel 11 erläutert.

Durch den Einsatz der Kamera wird der gesamte Prozess robuster, flexibler und unabhängiger von exakten mechanischen Positionierungen. Sie übernimmt damit die Funktion eines intelligenten Sensors, der die Schnittstelle zwischen der realen Umgebung und der digitalen Steuerung des Roboters bildet.

Die Sensor Kamera befindet sich unmittelbar über der Oberfläche des Husky Roboters mit einem Abstand von etwa 80 cm. Eine Verringerung des Abstandes zur Oberfläche wäre für die Messung von Vorteil und würde mit kürzerem Abstand bessere Messwerte liefern, jedoch muss ein gewisser Abstand gewährleistet sein, sodass der Roboterarm kollisionsfrei Bauteile von A nach B transportieren kann. Im Rahmen der Projektarbeit wurde die Kamera verschoben, sodass sie sich genauer im Mittelpunkt des Huskys befindet. Des Weiteren wurde die Auslenkung der Kamera von etwa 20° auf maximal 1° reduziert, sodass die Ebene der Kamera nahezu planar ist mit der Oberfläche und genauere Messwerte liefern kann.

In der vorherigen Projektarbeit T3100 sollte die Kamera in der Lage sein, Veränderungen in der Anzahl der Bauteile wahrzunehmen. Die Kamera sollte demnach erkennen können, ob der Husky leer oder voll besetzt ist und wie viele Bauteile anliegen und diese Daten an die SPS senden. In diesem Projekt, werden von der Kamera Koordinatenwerte angefordert, weshalb mehr Präzision und Genauigkeit gefordert ist. Die Kamera erfasst die Oberfläche des Husky Roboters und berechnet den Mittelpunkt der Oberfläche, auch Centerpoint genannt. Dieser Mittelpunktwert trägt die Information die räumlichen Koordinaten und die Verdrehung zum Ursprungskoordinatensystem (X, Y, Z, RX, RY, RZ) und wird an die SPS gesendet. Wenn der Husky verschoben wird, ändern sich die Werte entsprechend der Verschiebung, sodass diese Werte Variablen entsprechen und sich der Messung anpassen. An der Oberfläche sind die Abstände der Trägerflächen zum Mittelpunkt immer relativ gleich, da eine Verschiebung des Huskys auch einer Verschiebung der Trägerteile um denselben Betrag entspricht. In dem Roboterprogramm wird diese Differenz jeweils händisch für jeden Bauteilträger ermittelt. In dieser Version mit der variablen Position des Huskys konnte eine Palettierungsfunktion mit aktuellem Wissen nicht umgesetzt werden.

11. Anbindung an SPS

Nach ein wenig Recherche wurde ein Programmentwurf innerhalb des Panels entdeckt, welches von Universal Robotics zur Verfügung gestellt wurde. Es handelt sich hierbei um ein Palettierungsprogramm. Um das Palettierungsprogramm mit der SPS zu verbinden, erhält das Panel Daten mittels eines Registers. Hier wird der Datentyp „Boolean“ verwendet. Je nachdem, welches Bit vom Register „True“ ist, soll der gewünschte Prozess ablaufen. Bit [0] befiehlt dem Greifarm, Werkstücke aus dem Magazin auf das Fließband zu legen, für den Montage-Prozess. Bit [1] setzt das fertig montierte Werkstück auf den Husky auf. Bit [2] sorgt für den Demontage Vorgang, indem das Werkstück von dem Husky auf das Förderband gelegt wird.

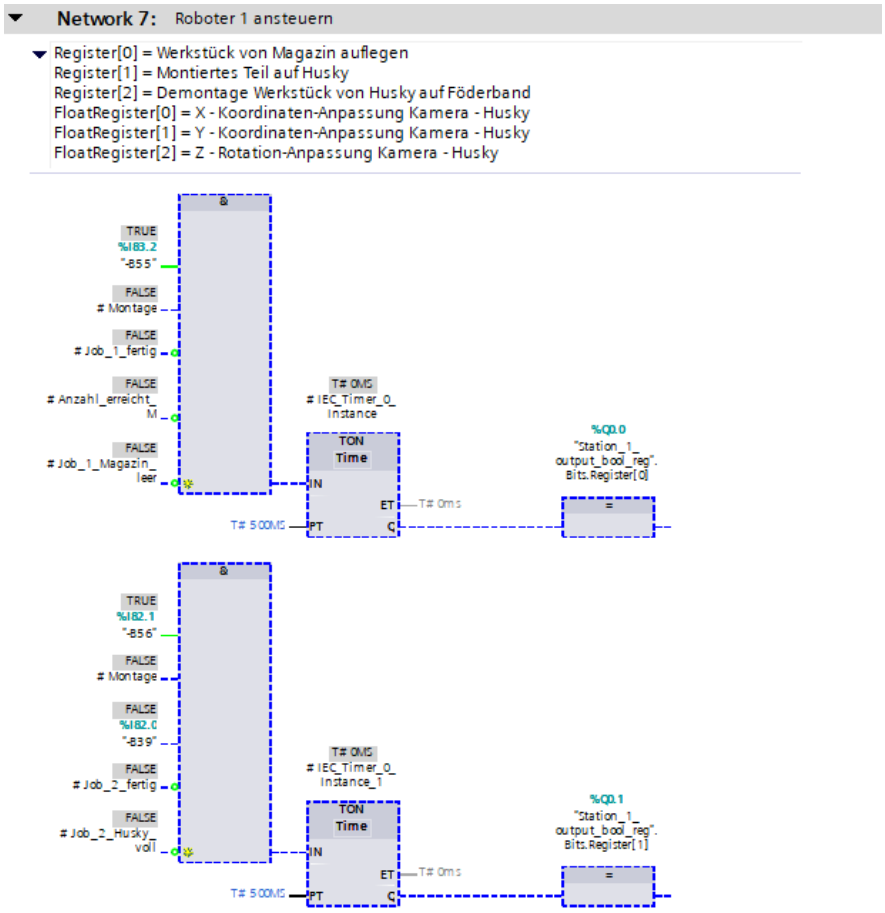


Abbildung 16: SPS Aktionsaufruf

Nachdem die Ansteuerung des jeweiligen Prozesses über die SPS programmiert wurde, folgte die Programmerstellung innerhalb des Panels. Zu Beginn der Programmausführung sollen erstmals gewisse Bedingungen erfüllt werden. Der Greifer muss mit Spannung versorgt und der Greifarm geöffnet werden. Die Variable „i“, welche für die Zuordnung der Werkstücke benötigt wird, wird auf den Wert Null gesetzt. Nun beginnt das Roboterprogramm mit einer

ständigen Abfrage, ob die oben genannten Bits, 0, 1, oder 2 „True“ sind. Ist eines dieser von der SPS kommenden Bits wahr, so wird ein dementsprechendes Unterprogramm aufgerufen. In diesem Unterprogramm befindet sich die Programmierung des jeweiligen Prozesses. BF1 beschreibt den Prozess der Montage, wo die Werkstücke aus dem Magazin genommen werden. BF2 Palettierung beschreibt den Prozess, bei dem die fertig montierten Werkstücke auf den Husky aufgeladen werden und BF3 Palettierung beschreibt den Prozess, in dem die Werkstücke für die Demontage von dem Husky heruntergenommen und auf das Fließband gelegt werden.

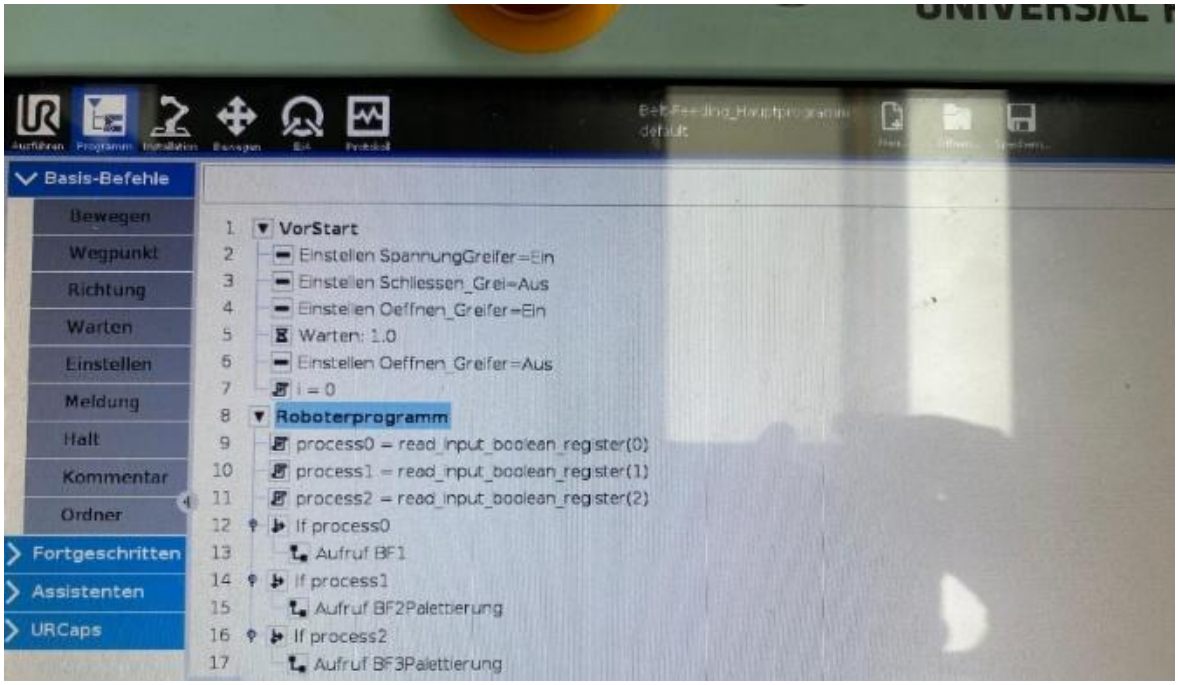


Abbildung 17: Register für Bewegungsabläufe

Für die Veranschaulichung wird BF1 betrachtet.

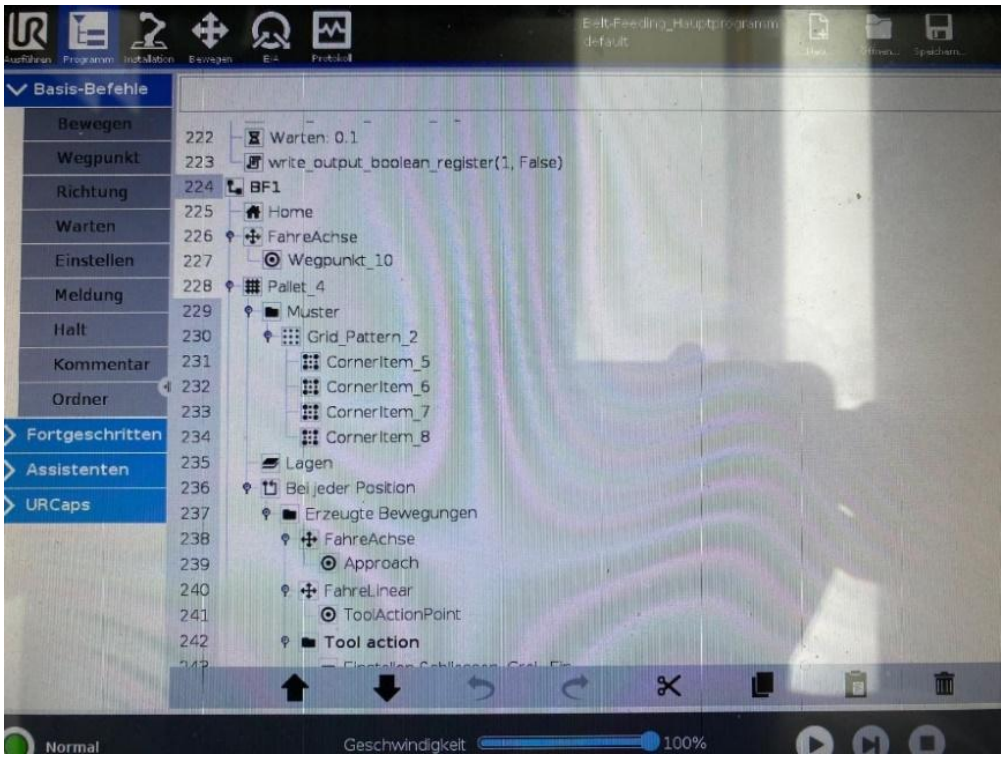


Abbildung 18:BF1

Im ersten Schritt wird ein Wegpunkt über dem Husky angefahren. Nun folgt das von Universal Robots zur Verfügung gestellte Programm, welches den Palettierungsvorgang für den Anwender vereinfacht, sofern die zu bearbeitende Fläche konstant die gleichen Abstände zwischen den Ablageflächen hat. Hier werden 4 Corner-Items, welche die äußeren 4 Punkte, der zu bearbeitenden Fläche beschreiben, festgelegt. Mit einem Muster wird angegeben, dass es sich hierbei um ein 2x3 Muster handelt. Der Approach-Punkt ist ein Zielpunkt, der vor jedem Greifen eines Werkstücks in eine sichere Position über dem Husky anfährt, bevor der Greifer Richtung Werkstück fährt. Im Tool Action wird programmiert, was im Falle eines Anfahrens eines Corner-Items passieren soll. In diesem Fall soll der Greifer schließen und das Werkstück aus dem Magazin greifen. Für die anderen Prozesse muss dies dementsprechend angepasst werden, da in einem Prozess das Werkstück auf dem Husky abgelegt und in dem anderen das Werkstück von dem Husky entnommen wird. Der Exitpoint ist eine Position nach dem Greifen, an welchem aus dem Palettierungsprogramm ausgetreten wird, um restliche Programmcodes durchzuführen.

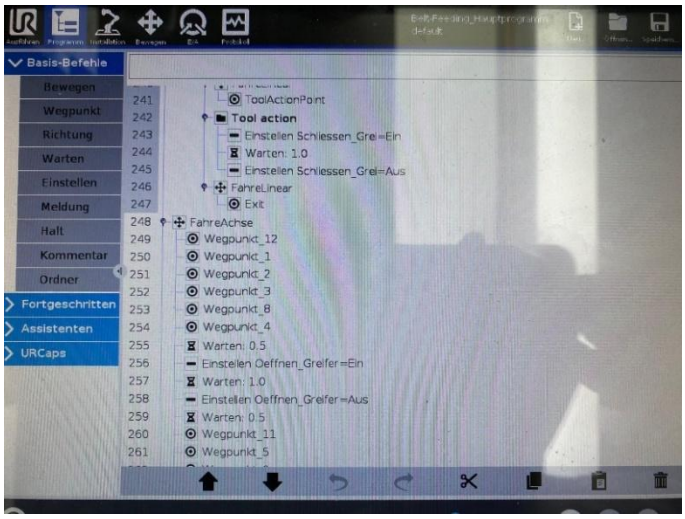


Abbildung 19: Palette Implementierung (1)

Nachdem das Palettierungsprogramm verlassen wurde, können die verschiedenen Wegpunkte mit verschleifen für ein anstoßloses und reibungsloses Verhalten bis zum Fließband abgefahren werden. An der gewünschten Position angekommen, öffnet der Greifer und legt das Werkstück ab.

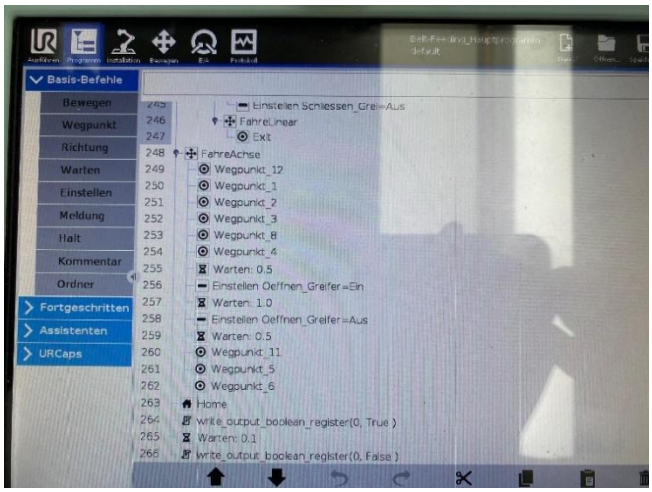


Abbildung 20: Palette Implementierung (2)

Der Greifer fährt in die Home Position und sendet über ein Register ein kurzes Signal zurück an die SPS, eine positive Flanke für eine Zeit von 0.1s, dass der Auftrag abgearbeitet wurde. Somit können weitere Schritte innerhalb der gesamten Anlage abgearbeitet werden.

Für die Einbindung der Kamera muss die Hardware im TIA Portal konfiguriert werden.

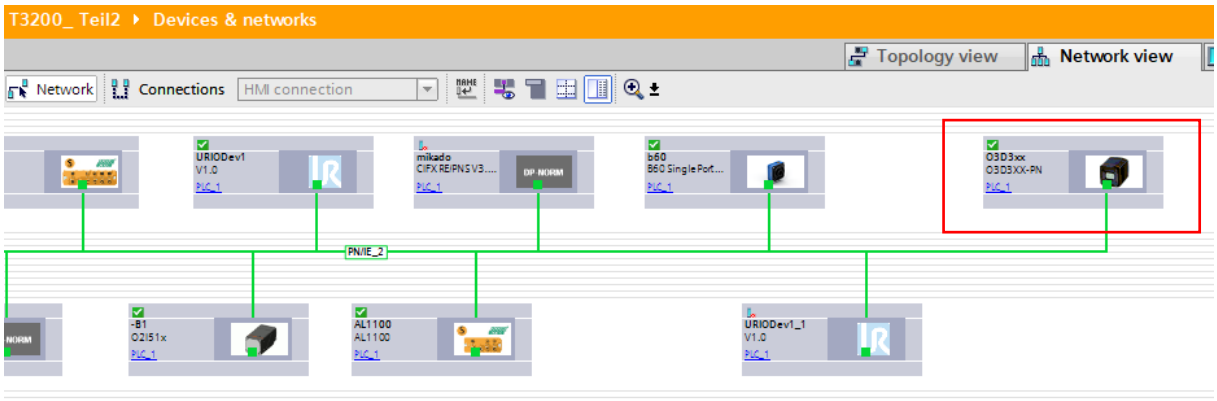


Abbildung 21: Teilnehmer Anbindung PROFINET

Der Anschluss der Kamera, sowohl PROFINET, als auch Spannungsversorgung, erfolgte bereits in der vorangegangenen Studienarbeit. Der erste Schritt für die Konfiguration beinhaltet das Einfügen einer GSD Datei. Diese Datei wurde von dem Gerätehersteller ifm zur Verfügung gestellt. Nachdem Speichern der Datei in einem Verzeichnis auf der Festplatte kann im TIA Portal unter dem Menüpunkt „Extras“ zu „GSD Dateien verwalten“ navigiert werden. Nachdem Auswählen der hinterlegten GSD Datei kann diese im TIA Portal installiert werden. Ist die Installation erfolgreich, wird dies über eine Meldung bestätigt. Nun ist die GSD Datei im Hardwarekatalog zu finden.

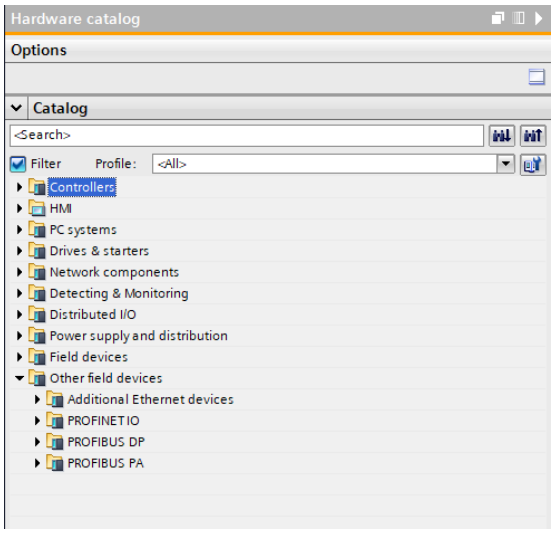


Abbildung 22: Hardware-Katalog

Im Hardwarekatalog unter „other field devices“ werden Installierte GSD Dateien hinterlegt. Unter „ifm- electronic“ kann der O3D3xx – Sensor als Gerät per drag and drop hinzugefügt werden.

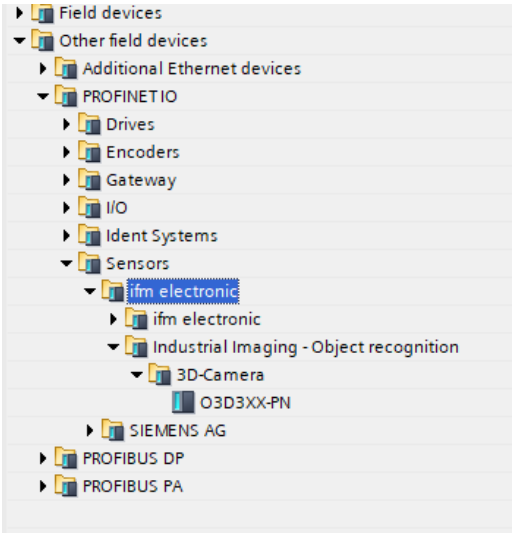


Abbildung 23: Hardwarekatalog-Sensoren

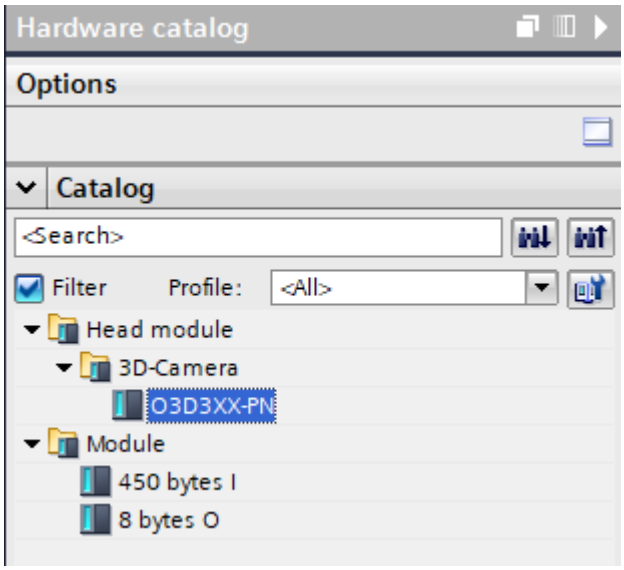


Abbildung 24: 3D-Kamera auswählen

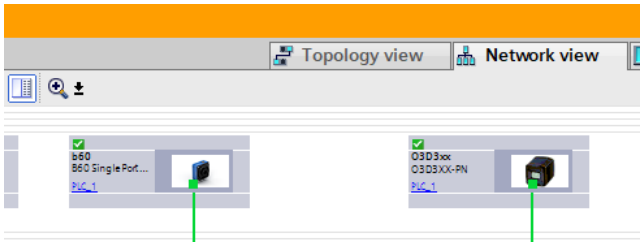


Abbildung 25: Eingefügte Kamera

Befindet sich der Sensor nun als Gerät in der Topologie beziehungsweise in der Netzwerkansicht, wird per Drag and Drop das Werkstück an das vorhandene PROFINET-Netzwerk angebunden. Wichtig sind die Hardwareeinstellungen der Kamera im Tia Portal. Subnetz- und IP Adresse müssen für eine funktionsfähige Verbindung mit der Subnetz- und der IP Adresse im Vision Assistant übereinstimmen. Nun kann die Hardware übersetzt werden und eine Onlineverbindung kann aufgebaut werden.

Gerätehersteller bieten neben den Einleitungen und Dokumentationen für die Inbetriebnahmen auch Bausteine ihrer GSD-Dateien zu Verfügung, um dem Programmierer einfach und sauber die volle Funktionsfähigkeit zu liefern., so auch ifm. Die zur Verfügung stehende Bibliothek wird auf der ifm Seite heruntergeladen und im TIA Portal als globale Bibliothek geöffnet.
Nachdem sich die globale Bibliothek im Tia Portal befindet, können je nach gewünschter Arbeitsweise der Kamera, verschiedene Werkstücke in das Tia Projekt übertragen werden.
Dazu werden die Bausteine per Drag and Drop eingefügt.

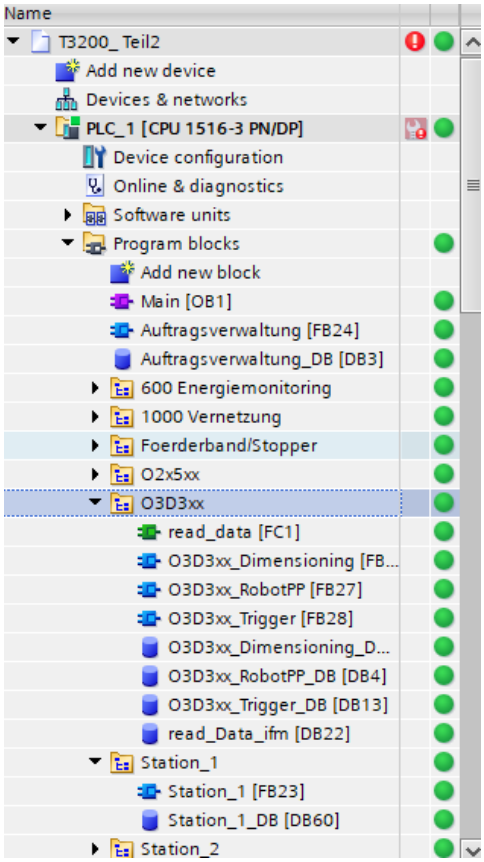


Abbildung 26: Funktionsbibliothek O3D3xx

Für die bessere Übersicht wurde zuerst ein Ordner mit dem Namen „O3D3xx“ erstellt. Hier befinden sich die aus der globalen Bibliothek hinzugefügten Bausteine „Dimensioning“ für die Oberflächenvermessung, „RobotPP“ für die Roboter-Greifernavigation und „Trigger“ um den Triggereingang zu konfigurieren. Da es sich hier um die Programmiersprache SCL handelt, wurden die Bausteine per Drag and Drop in einen neuerstellen „FC“ gezogen, „read_data“.

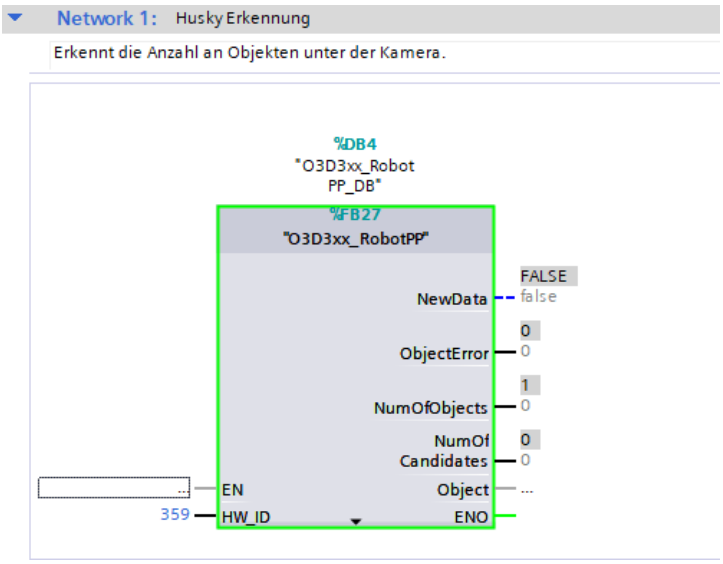


Abbildung 27: Roboter-Greifernavigations-Baustein

Mit dem Roboter-Greifernavigations-Baustein können verschiedene Werte ausgelesen werden.
sobald ein Objekt erkannt wird, wird NewData true. Object Error gibt aus, ob sich ein Fehler in einem erkannten Objekt erkennen lässt. NumOfObjects gibt die Anzahl der erkannten Objekte durch. „Object“ selbst ist ein Array, in welchem für jedes erkannte Werkstück sowohl x, y, z Koordinaten, als auch die Rotation um x, y, und z-Achse ausgegeben werden. Damit der Roboter-Greifernavigation Baustein funktioniert, muss die Hardware erkannt werden. PROFINET-Geräte im TIA Portal haben eine einzigartige Hardwareerkennung, eine ID. Diese ID befindet sich im Gerätekatalog unter den Einstellungen.

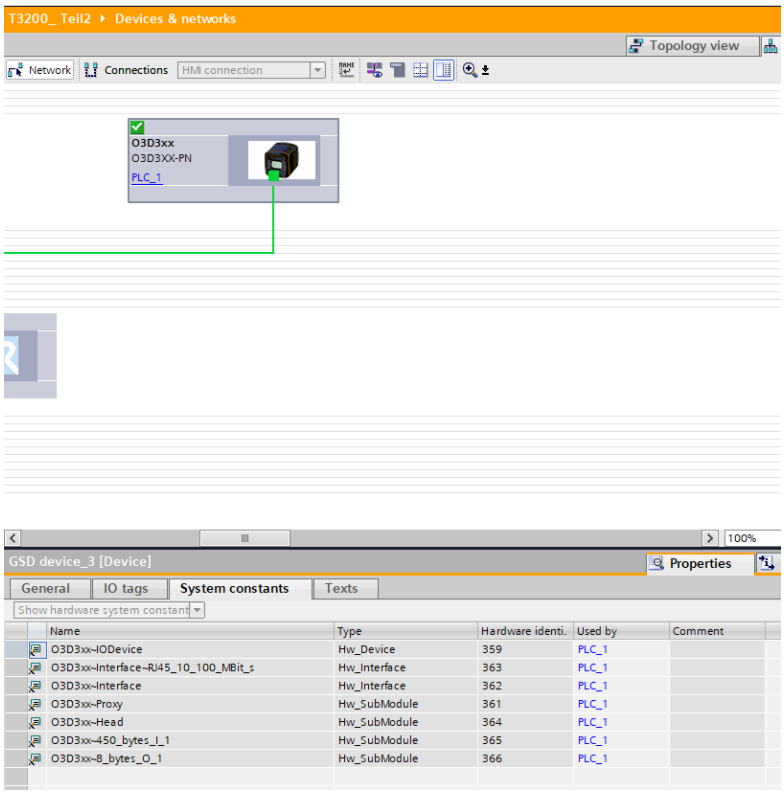


Abbildung 28: Hardware ID

In den Einstellungen angekommen gibt es den Reiter System „constants“. Hier kann die Hardware ID eines Gerätes gefunden werden. Mit der HW ID Eingabe von „359“ im obigen „RobotPP“ Baustein stelle ich eine direkte Verbindung mit der Hardware her und erste Daten über eine Werkstücks Erkennung unter der Kamera konnten getestet werden. Damit war die Kamera eingebunden.

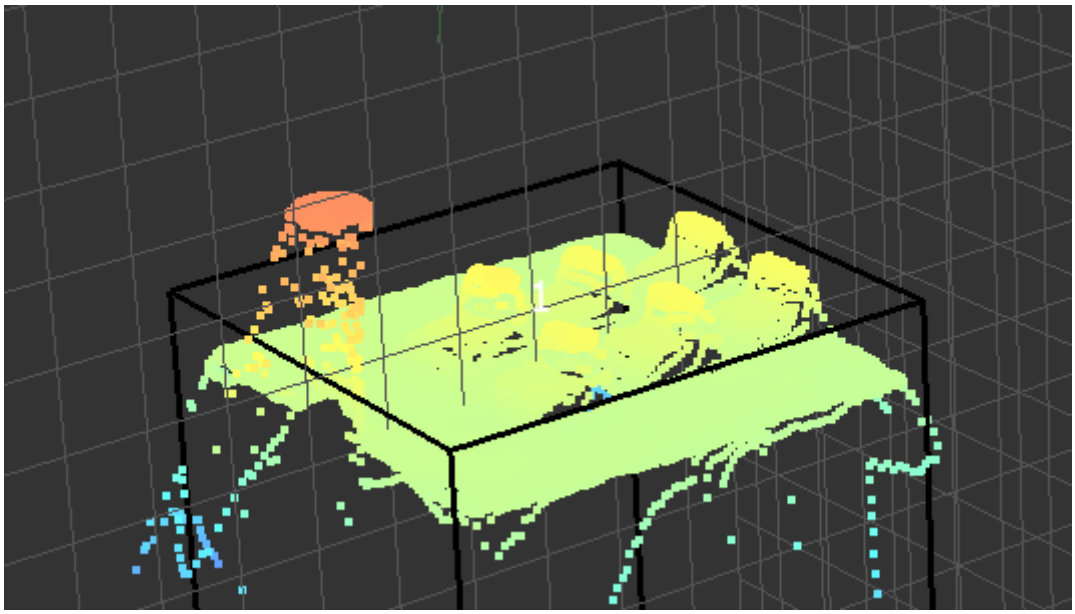


Abbildung 29: Husky unter 3D Kamerabild

Diese Abbildung zeigt eine Aufnahme des Huskys unterhalb der Kamera. Zu Beginn stellte sich die Frage, ob jedes Werkstück einzeln, oder der gesamte Husky als ein einziges großes Objekt erkannt werden soll. Da die Kamera mehrere Werkstücke erkennen kann und von jedem erkannten Werkstück x, y und z-Koordinaten liefern kann wurden erste Versuche mit dem Erkennen der 6 Werkstücke durchgeführt. Nach der Parametrierung mittels „Vision Assistant“ wurde schnell festgestellt, dass die Umsetzung so einfach nicht möglich war.

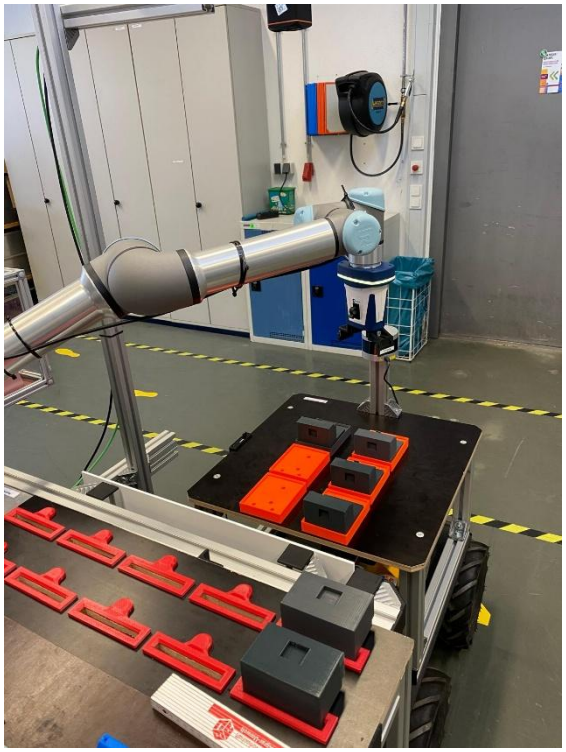


Abbildung 30: Husky verschoben von Referenzpunkt

Die Kamera von ifm erkennt unterschiedlichen Höhen. Je weiter sich die Kamera von der

Referenzebene weg befindet, umso größer muss ein Höhenunterschied sein, damit die Kamera einen Unterschied zwischen Referenzebene und dem Werkstück erkennen kann. Der Husky wurde dementsprechend als Referenzebene genutzt. Die Problematik hier war, dass sich auf dem Husky im Montage-Prozess kein Werkstück befindet. Die 6 Ablageflächen für die Werkstück haben eine Höhe von 2cm, was bei der installierten Kamera nicht erkannt werden kann.
Leider war das Verstellen der Kamera auf eine niedrigere Position erfolglos, da bei geringerem Abstand zum Husky die Kamera nicht mehr alle 6 Ablageflächen erkennen kann und ein Licht der Kamera so auf der Oberfläche des Huskys reflektiert, sodass die Ergebnisse verfälscht werden.
Aus diesem Grund wurde die ifm Kamera auf einer Höhe angebracht, in der diese den Husky relativ mittig und ohne viele Störfaktoren erkennen kann.
Da die einzelnen Ablageflächen nun nicht erkannt werden konnten, musste der gesamte Husky als ein Werkstück erkannt werden, wie in der obigen Abbildung. Nun spielt es keine Rolle ob Werkstücke aufliegen oder nicht.

Damit die Kamera nicht kontinuierlich Werte für jegliche Koordinaten des Huskys ausgibt, wird ein Eingang für eine Trigger-Quelle genutzt.

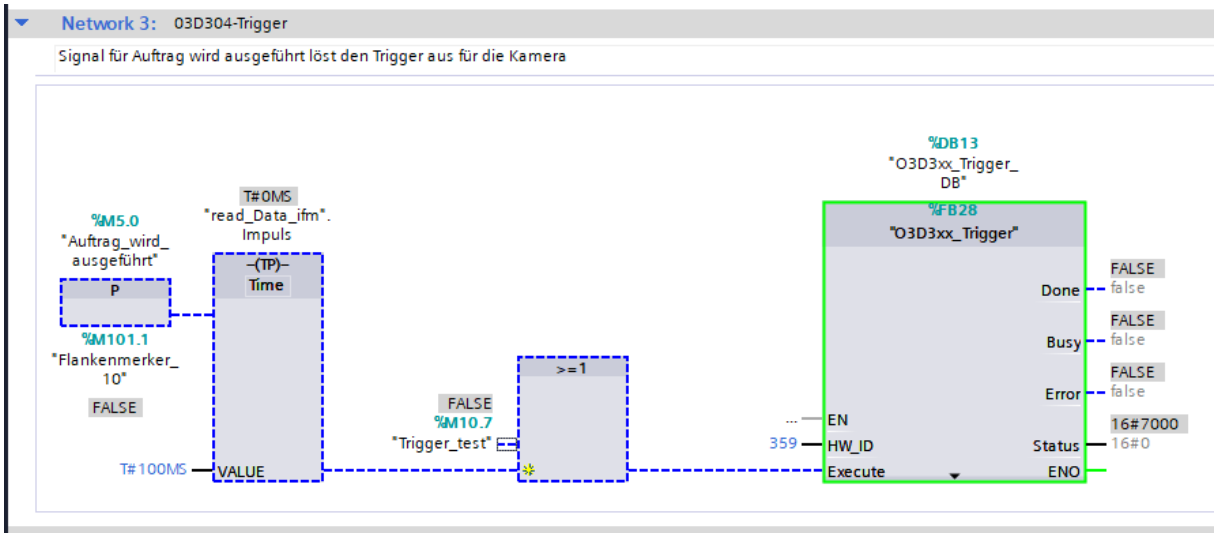


Abbildung 31: Kamerasensorwerte Trigger

Die obige Abbildung zeigt die Schaltung in dem „read_Data“ Baustein. Vorrauschschauend wurde der Trigger-Eingang so verschaltet, dass sobald ein Auftrag (Demontage oder Montage) gestartet wird, eine positive Flanke auf den Trigger-Eingang der Kamera gegeben wird. Nun wird eine Momentaufnahme des Huskys unter der Kamera erstellt, welche konstante Koordinaten an die SPS liefert. Die Variable „Trigger-test“ kann angesteuert werden, um diesen Vorgang zu simulieren. Der Trigger Baustein kann ebenso ein Fehlerfall melden, ob er beschäftigt ist und sobald er fertig ist.

Für die Anpassung der Koordinatensysteme zwischen Kamera und dem Greifarm war die Überlegung folgende:

Die Kamera erkennt ein Objekt. Die Koordinaten der x y und z Koordinaten beschreiben den Mittelpunkt des erkannten Objekts. Darum soll der Greifarm zu exakt dieser Position fahren, an der die Kamera den Mittelpunkt des Huskys ausliest. Von hier aus kann der Greifarm über variable Positionen ein Palettierungsprogramm abarbeiten.

Dafür kann in die Station 1 navigiert werden, in der sich auch die Register-Bits befinden.

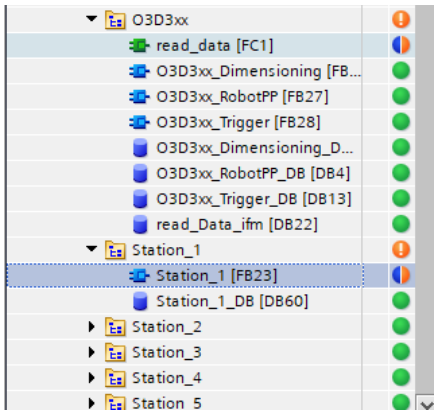


Abbildung 32: Funktionsbaustein Station 1

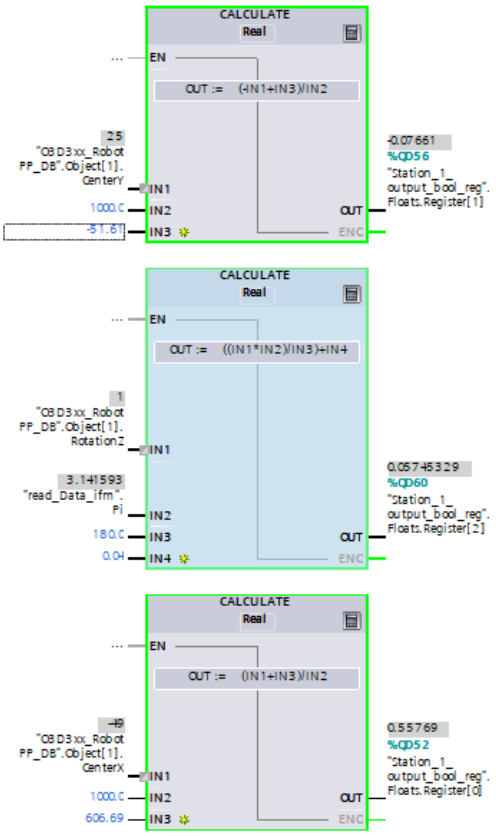


Abbildung 33:Funktionsbaustein Station 1 (2)

Drei Calculate-Bausteine sorgen für die Anpassung der Koordinaten zwischen Kamera und Roboter.

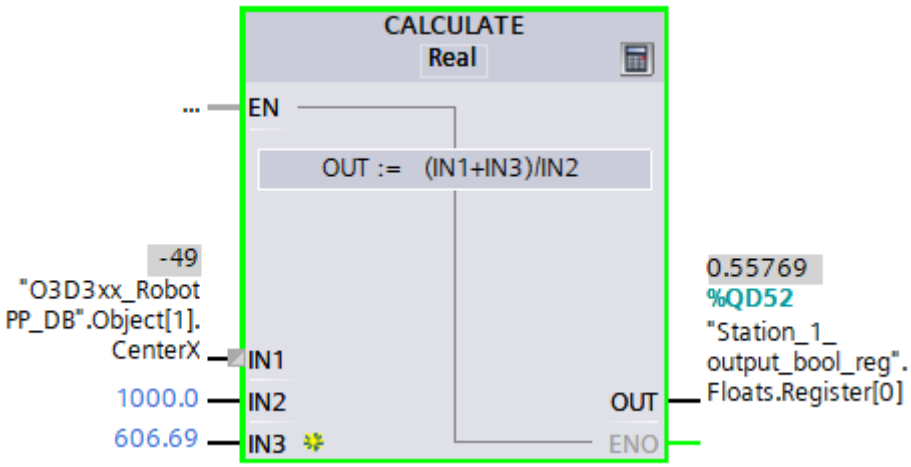


Abbildung 34: Calculate-Baustein Station 1 (1)

Der erste Calculate Baustein mit dem Datentyp „Real“ sorgt für die Anpassung der x-Koordinate. IN1 beinhaltet die x-Koordinate für den Mittelpunkt des Huskys. Mit IN3 wird die x-Koordinate des Greifarms so angepasst, dass er sich auf den Mittelpunkt des Huskys bewegt. IN2 rechnet den gemessen Wert von der Einheit mm (Millimeter) in m (Meter) um. Die Meter-Einheit wird für die Programmierung im Panel des Greifarms benötigt.

Der Ausgang des Bausteins wird über ein weiteres Register mit dem Datentyp „Float“ an das Panel übergeben.

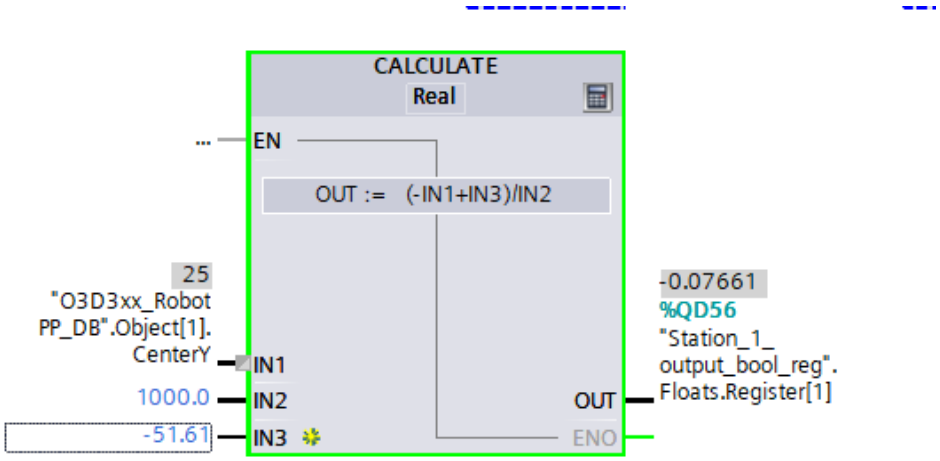


Abbildung 35: Calculate-Baustein Station 1 (2)

Der 2. Calculate Baustein hat dieselbe Funktion wie der Erste. Der einzige Unterschied hierbei liegt darin, dass hier die y-Koordinate angepasst wird.

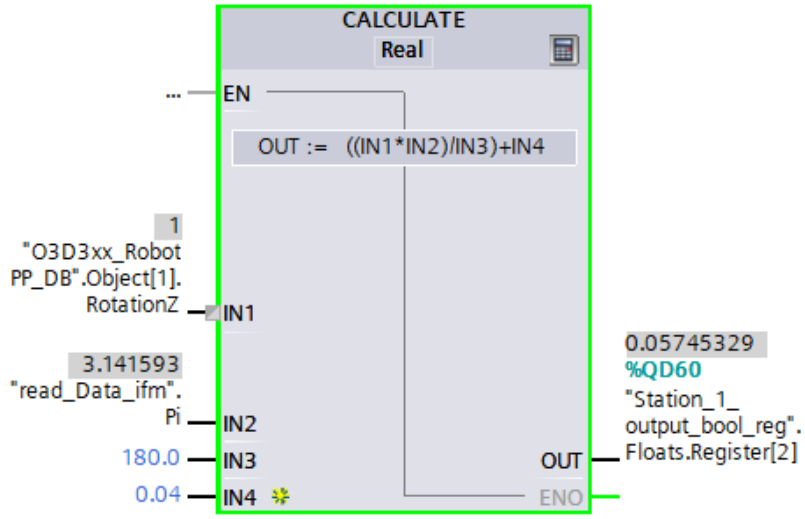


Abbildung 36: Calculate-Baustein Station 1 (3)

Der dritte und letzte Calculate-Baustein soll die Rotation um z-Achse berechnen. Die Überlegung war, dass eine Verschiebung in der Z-Achse nicht nötig ist, da sich die Höhe des Huskys und die Rotation um x und y-Achse nie verändert. Darum nur die Rotation um die z-Achse. IN1 beschreibt die Rotation um die z-Achse, welche von der Kamera gesendet wird. IN4 ist der Wert, um die Rotation des Greifarms anzupassen. IN3, der Wert „180“, und IN2, die Pi-Variable, sind für die Umrechnung von Grad in Rad. Die Kamera liefert eine Rotation in Grad, wobei das Panels des Greifarms eine Rotation in Rad misst. Die Variable PI wurde mit der SCL Sprache erstellt, da Tia Portal diese Konstante nicht zur Verfügung stellt.

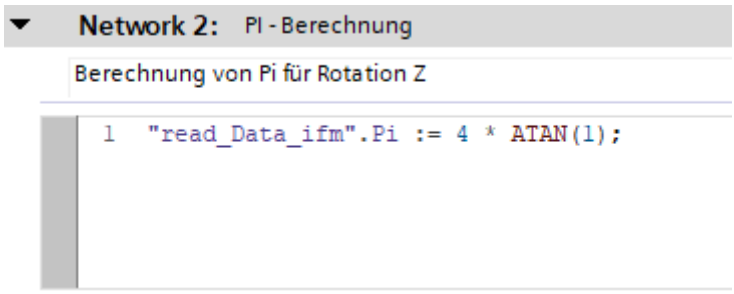


Abbildung 37: Pi für Rotation berechnen

Wie bereits in der vorigen Studienarbeit, beinhaltet der Vision Assistant eine Programmausführung, die jedem Anwender eine einfache und übersichtliche Konfigurierung der Kamera bietet. Da bereits eine Schritt-für-Schritt-Anleitung durch den Vision Assistant durchgeführt wurde, wurden einige Anpassungen nochmal zusammengefasst.

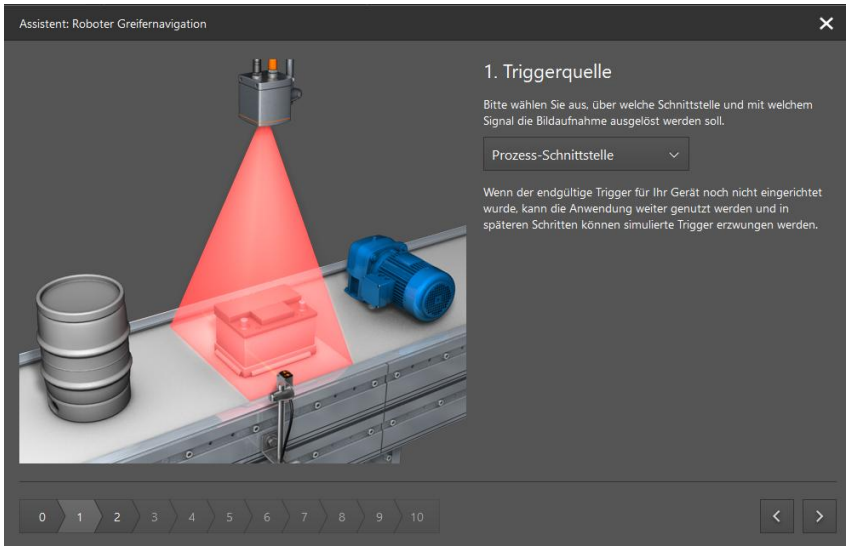


Abbildung 38: Triggerquelle

Dank des Trigger-Bausteins konnte hier die „Prozess-Schnittstelle“ ausgewählt werden, statt eines Konstanten Wertes.

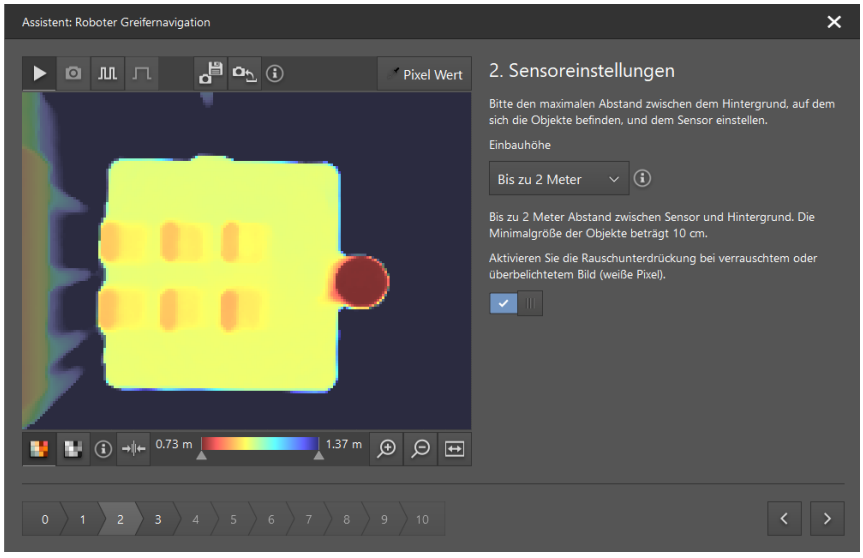


Abbildung 39: Sensoreinstellungen

Für die Sensoreinstellung wurde eine Höhe bis zu 2-Meter festgelegt, da sich die Kamera ungefähr 1m über dem Husky und unter 2m über dem Boden befindet.

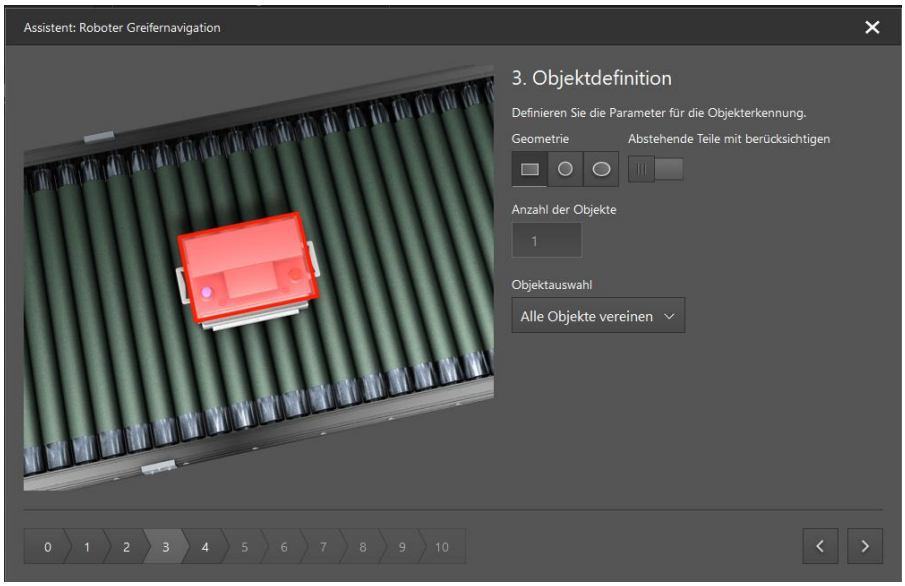


Abbildung 40: Objektdefinition

Für die Anzahl der Objekte wurde „1“ gewählt, da wir keine 6 Werkstücke erkennen, sondern den Husky als 1 ganzes Objekt mit der quadratischen Form.

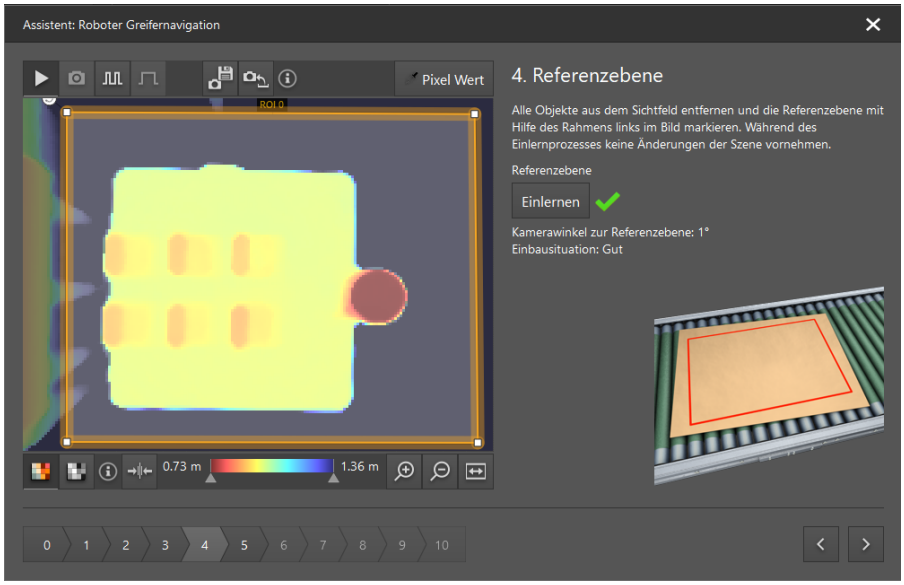


Abbildung 41: Referenzebene

Die Referenzebene musste neu eingelernt werden. In der vorigen Arbeit diente der Husky als Referenzebene und die einzelnen Werkstücke sollten per Höhenunterschied erkannt werden. Nun dient der Boden als Referenzpunkt, damit der gesamte Husky erkannt werden kann. Der Husky wurde auf die Seite geschoben, die Referenzebene eingelernt und somit konnte der Husky seine Position wieder einnehmen.

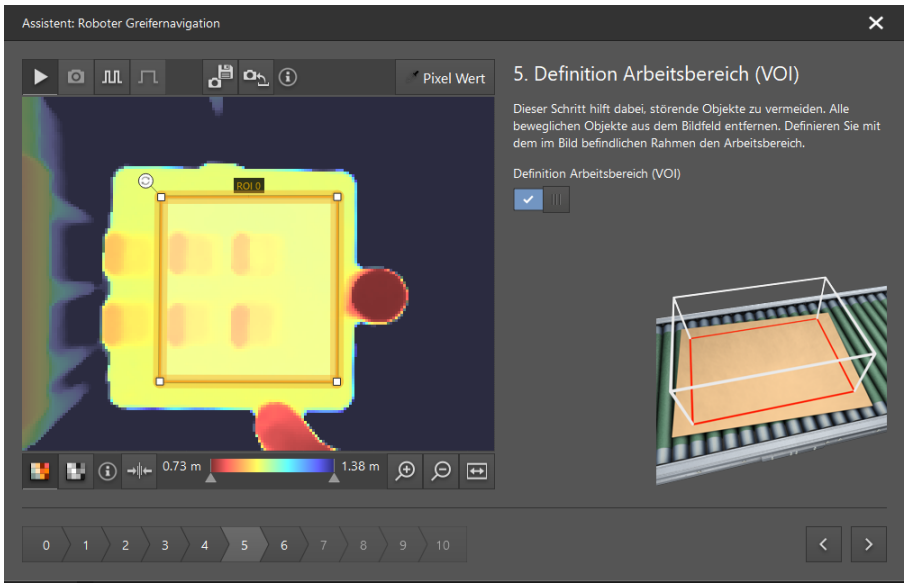


Abbildung 42: Arbeitsbereich

Mit dem Definitionsbereich wurde der Rahmen festgelegt, in welchem ein Werkstück ordentlich erkannt werden kann. Wird der Definitionsbereich zu groß ausgelegt, werden nebenstehende Teile erkannt, die das Ergebnis verfälschen können. Wird der Definitionsbereich zu klein ausgewählt, können Teile des Huskys eventuell nicht mehr erkannt werden.

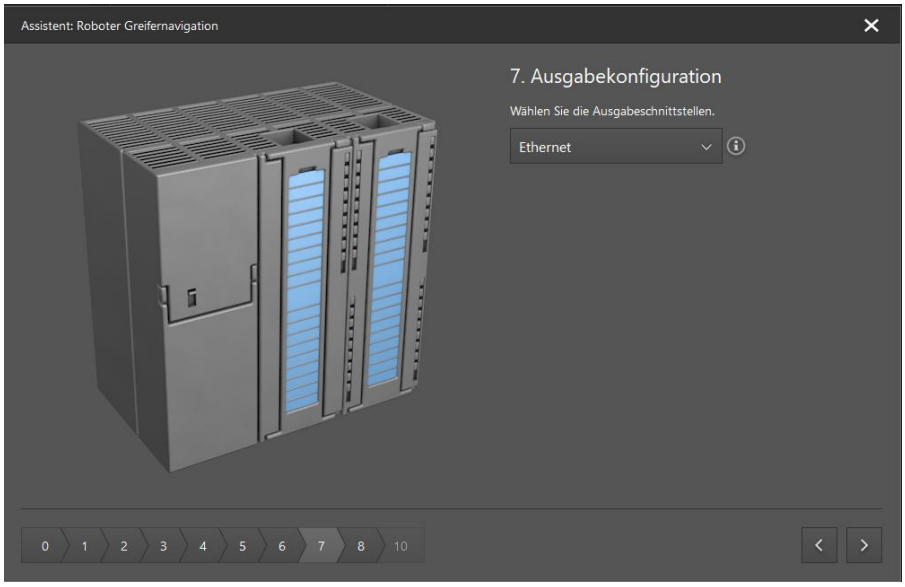


Abbildung 43: Ausgabekonfiguration

Da die Daten über PROFINET übermittelt werden, wird für die Ausgabekonfiguration nur Ethernet ausgewählt.

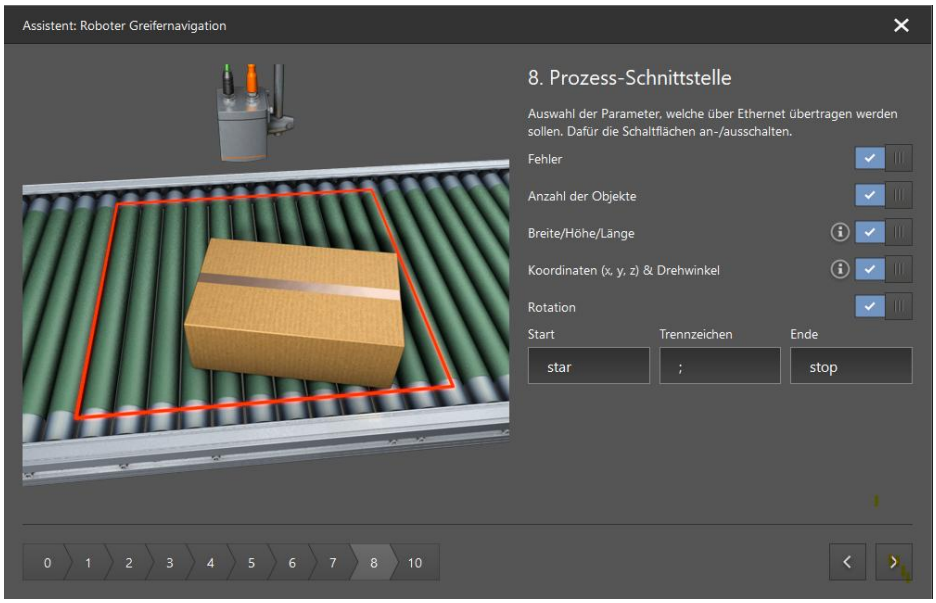


Abbildung 44: Prozessschnittstelle

Da jegliche Daten übermittelt werden sollen, werden alle Daten zur Ermittlung aktiviert.

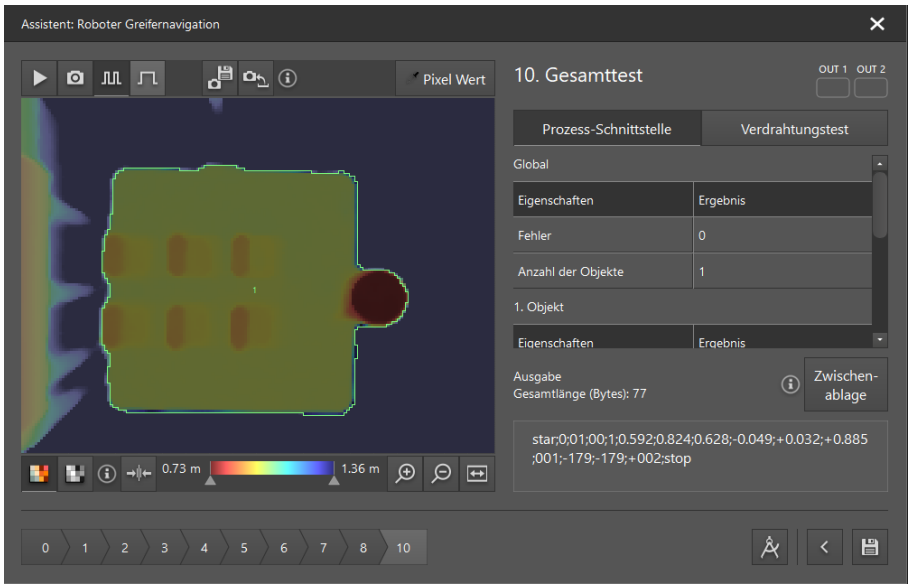


Abbildung 45: Gesamttest

Kurz vor Abschluss der Konfigurierung bietet Vision Assistant noch einmal eine genaue Datenerfassung, welche die Werte wiedergibt und das Kamerabild visuell darstellt. Nach dem Abschluss können auch die Daten separat im Vision Assistant vom Anwender ausgelesen werden.

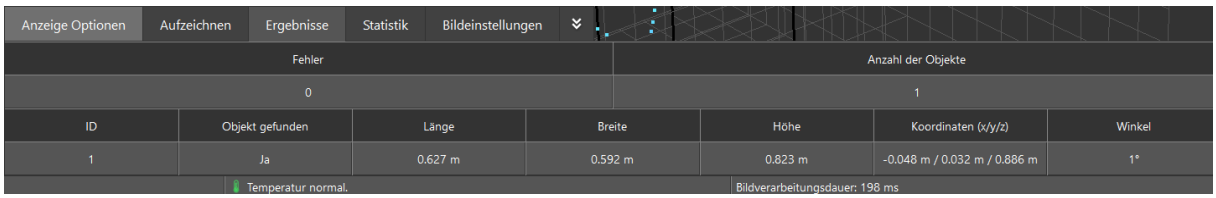


Abbildung 46: Berechnete Werte

Für die Panelprogrammierung wurde das Palettierungsprogramm als Vorlage genutzt. Da im Palettierungsprogramm von Universal Robotics jedoch keine variablen Positionen programmiert werden konnten, wurde die Palettierung händisch programmiert.

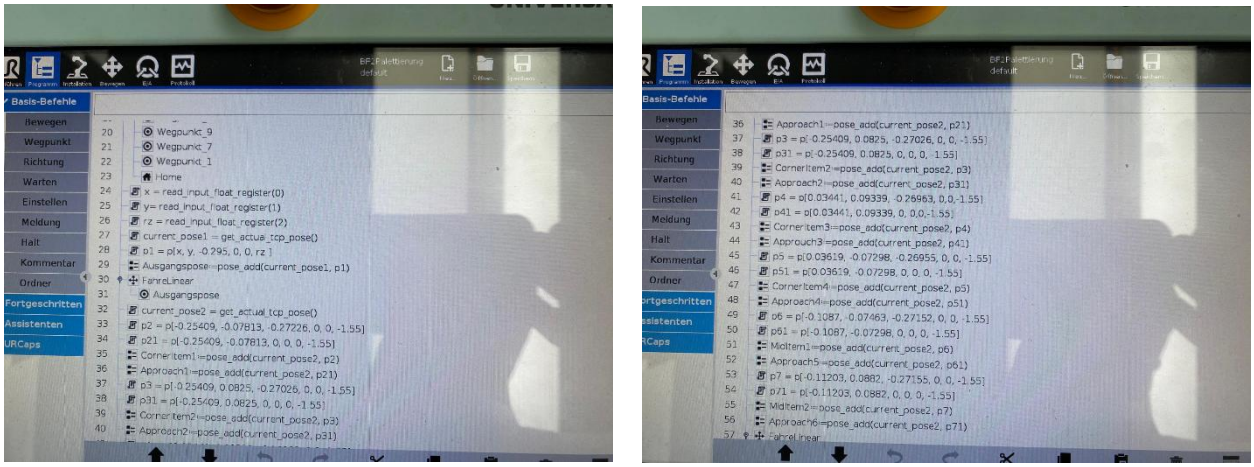


Abbildung 47: Variable Position (1) und (2)

Schritt für Schritt wurde das Palettierungsprogramm programmiert. Die Kamera liefert über die SPS „Float“ Datentyp Werte an das Universal Robotics Panel. Hier werden die Werte den Variablen x (Anpassung der x-Koordinate, y (Anpassung der y-Koordinate) und rz (Anpassung der Rotation um die Z-Achse) zugewiesen. Um den Greifarm an der gewünschten Position mittig über den Husky zu platzieren, werden die Daten von der SPS und die aktuelle Position des Huskys (ausgehend von der Home Position) zusammengefasst zu einem Punkt. Dies entspricht dem Ausgangspunkt über dem Husky, von dem das weiterführende Programm abhängt. Von diesem Punkt aus wurden alle Positionen mit dem Greifarm angefahren. Um den Greifarm auf die gewünschten Ablageflächen zu navigieren, wurde eine Variable „i“ erstellt, welche nach jedem Durchlauf inkrementiert wird. Beginnend von „0“ bis „5“ arbeitet der Greifarm die 6 Ablageflächen ab. Wurde das letzte Teil von dem Husky entnommen bzw. auf den Husky aufgeladen, wird die Variable „i“ auf „0“ zurückgesetzt. Somit kann nach Beenden von 6 Werkstücken oder bei erneutem Starten des Programms der Montage oder Demontageprozess vom ersten Werkstück aus angefangen werden. Dies wurde mittels einer If-Anweisung programmiert.

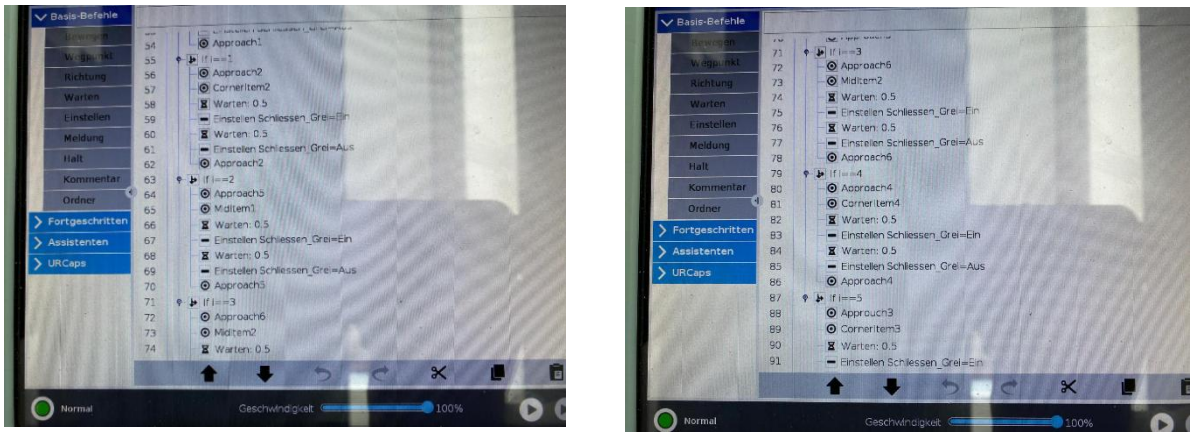


Abbildung 48: Variable Position (3) und (4)

Da die Programmierung einen Demontage- und einen Montageprozess beinhaltet mussten bei zwei der drei Programmierungen dieser Code verwendet werden. Kleinere Anpassungen innerhalb des Codes wurden gemacht, um das Ablegen der Werkstücke auf den Husky im Montageprozess zu besänftigen. Damit legt der Greifarm die Werkstücke sorgfältig auf die Ablagefläche auf dem Husky, ohne anzustoßen. Im Demontageprozess, indem die Werkstücke vom Husky aus auf das Fließband befördert werden, konnte der Greifarm präziser an das Werkstück herangeführt werden, um für ein reibungsloses Aufnehmen des Werkstücks zu garantieren. Für den Montage-Prozess, an dem die Werkstücke von dem Magazin auf das Fließband gelegt werden, konnte der bisherige Code für feste Positionen verwendet werden. Hier spielt eine Positionsveränderung des Huskys keine Rolle.

12. Anpassungen und Zwischenfazit

Die Anpassung der Koordinaten war so weit fertiggestellt. Die Daten wurden über das Register an das Panel des Greifarms geschickt. Das Panel greift auf die Werte von der SPS zu und bindet diese in die Greifarmsteuerung ein.
Der Umgang mit dem Einstellen des Vision Assistant bedarf viel Fingerspitzengefühl. Da sich die Position des Huskys von jedem Arbeitsvorgang unterscheiden kann, muss ein gewisser Spielraum gewährleistet werden. Als Orientierungspunkt wurde die gelbe abgeklebte Fläche um den Husky verwendet.

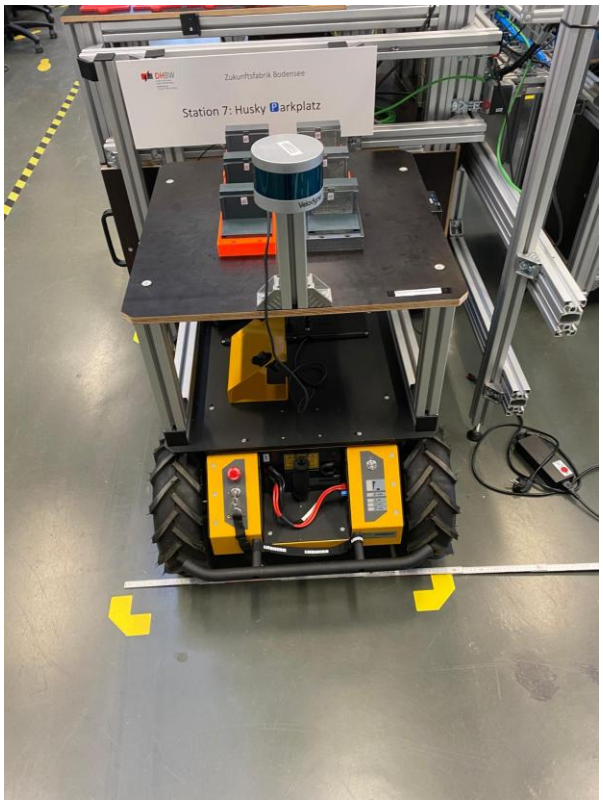


Abbildung 49: Positionsabmessung Husky

Dieser Spielraum kann mit dem Definitionsbereich und dem Einlernen der Referenzebene gewährleistet werden. Die Problematik hierbei ist, dass sowohl Referenzebene als auch Definitionsbereich in gewissen Situation zu hoch, bzw. zu niedrig definiert werden können. Wird beispielsweise der Definitionsbereich erhöht, kann die Kamera theoretisch den Husky in verschiedenen Positionen besser erkennen. Sprich, es gibt einen größeren Spielraum des Huskys. Nun wird aber auch die Fläche größer, in denen Höhenunterschiede erkannt werden können. Da der Husky keine Ebene Fläche beinhaltet, erkennt die Kamera jegliche Höhenunterschiede. Das Gerüst um den Husky wird erkannt und liefert falsche Ergebnisse. Somit sollte der Definitionsbereich recht klein gehalten werden. Wird dieser aber zu klein gehalten, so werden Teile des Huskys abgeschnitten und verfälschen erneut die Ergebnisse.

Bei den Testversuchen wurde der Husky mittig der gelben Fläche platziert. Von hieraus wurden die Daten innerhalb der SPS so angepasst, dass der Greifarm exakt mit der Position der Werkstücke interagiert. Wird der Husky in der x-Ebene (Abb. nach hinten und vorne)

verschoben, so kann der Greifarm innerhalb der gelben Fläche relativ genau die Positionen anfahren.

Innerhalb der y-Achse (obige Abbildung links und rechts) war der Spielraum etwas kleiner.



Abbildung 50: Ermittlung des Spielraumes

Mit einem Spielraum von ca. 12cm kam der Greifarm nah an seine Grenzen. Während die x-Koordinaten sehr genau angepasst werden, fehlt es an der y-Anpassung an Präzision. Grund hierfür war die Problematik mit dem festgelegten Definitionsbereich der Kamera im Vision Assistant. Befindet sich der Husky innerhalb der gelben Flächen, so wird die Position der Werkstücke zwar funktionsfähig abgefahren, allerdings mit einem leichten Versatz vom Greifarm zum Werkstücke. Befindet sich der Husky zu weit außerhalb der tolerierten Fläche, so wird der Versatz größer und der Greifarm stößt bei dem Versuch ein Werkstück aufzunehmen auf die obere Kante des Werkstücks und blockiert.

Bei dem Testen der Grenzen innerhalb des Bereichs, an dem der Greifarm gut genug arbeiten kann, wurde ebenso festgestellt, dass in dem Eckpunkt, welcher am weitesten von dem Greifarm entfernt ist ein Fehler auftritt. (Abbildung unten links)

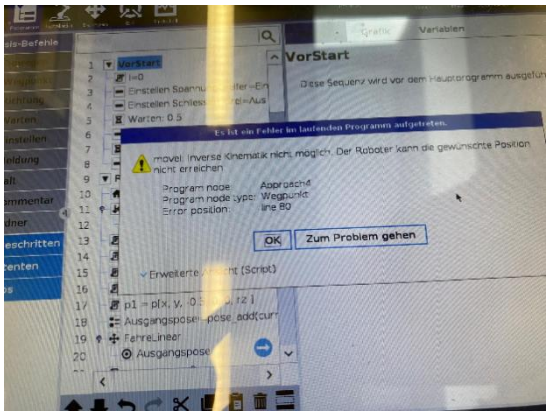


Abbildung 51: Fehlerquelle Inverse Kinematik

Der Greifarm kann die Position aufgrund der Mechanik nicht erreichen.

12. Fazit und Ausblick für folgende Gruppen und Verbesserungsbedarf an dem Projekt

Im Rahmen des Projekts wurde die 3D-Kamera ifm O3D304 erfolgreich an die SPS angebunden und über die ifm-Software in Betrieb genommen. Dabei wurde der relevante Überwachungsbereich definiert, in dem sich der Transportroboter (Husky) bewegt. Zur weiteren Automatisierung kam ein Greifarmroboter zum Einsatz, der über ein Polyscope Panel programmiert wird. Für den Materialtransport wurden insgesamt drei Roboterprogramme entwickelt, die das Aufsetzen, Abnehmen und die Demontage von Bauteilen ermöglichen. Zwei dieser Programme sind flexibel einsetzbar: eines für eine feste Position des Transportroboters und eines für eine verschobene, variable Position. Innerhalb des definierten Arbeitsbereichs – den gelben Linien – kann der Transportroboter somit beliebig platziert werden, solange er geradlinig einfährt.

Für das Wissensmanagement wurde entsprechende Verbesserungspotential aufgeführt. Diese sollen nochmal in dieser Arbeit aufgelistet werden:

Erweiterung der Variablen um die Rotation RX, RY, RZ:
Es wurden bisher zur variablen Steuerung des Husky Roboters, Raumkoordinaten X, Y, Z in das Roboterprogramm mitaufgenommen, die Variablen für die Verdrehung sind noch nicht berücksichtigt worden.

Zusätzliche Funktion schreiben zur Verbesserung der Auffassung des Center-Points:
Der Roboterarm bewegt sich präzise für die Ausgangsposition des Husky Roboters. Sobald der Abstand in X oder Y Richtung zur Ausgangsursprung zu hoch ist, wird das Aufnehmen von Bauteilen zunehmend unpräziser.

Folgende Gruppe sollte verstehen wie Center Point berechnet wird und entsprechende Korrektur berechnen können für eine positionelle Abweichung in X und in Y Richtung.

Vermutung besteht das Center Point bei präzise berechnet wird, wenn der Mittelpunkt des Huskys senkrecht zum Husky steht. Bei einer Verschiebung, um mehr Spielraum kommt es zu Abweichungen

Abbildungsverzeichnis:

Abbildung 1: Schematischer Ablaufplan 2
Abbildung 2: richtige Greifweise....3
Abbildung 3: Teach Pendant....5
Abbildung 4: Startoberfläche 5
Abbildung 5: Programm laden....6
Abbildung 6: Robotereinstellungen 7
Abbildung 7: Roboterarm aktivieren....8
Abbildung 8: Manuelle Bewegung....9
Abbildung 9: Relative Geschwindigkeit einstellen 9
Abbildung 10: Programmumgebung....10
Abbildung 11: Palette 2 Abbildung 12: Palette 1....12
Abbildung 13: Palettierungsfunktion 12
Abbildung 14: Gitterstruktur auswählen....13
Abbildung 15: ifm 3D Sensor Kamera (O3D304) 14
Abbildung 16: SPS Aktionsaufruf 16
Abbildung 17: Register für Bewegungsabläufe....17
Abbildung 18:BF1....18
Abbildung 19: Palette Implementierung (1)....19
Abbildung 20: Palette Implementierung (2)....20
Abbildung 21: Teilnehmer Anbindung PROFINET 21
Abbildung 22: Hardware-Katalog....21
Abbildung 23: Hardwarekatalog-Sensoren....22
Abbildung 24: 3D-Kamera auswählen....22
Abbildung 25: Eingefügte Kamera 22
Abbildung 26: Funktionsbibliothek O3D3xx....23
Abbildung 27: Roboter-Greifernavigations-Baustein 24
Abbildung 28: Hardware ID 25
Abbildung 29: Husky unter 3D Kamerabild 26
Abbildung 30: Husky verschoben von Referenzpunkt 26
Abbildung 31: Kamerasensorwerte Trigger....28
Abbildung 32: Funktionsbaustein Station 1....29
Abbildung 33:Funktionsbaustein Station 1 (2) 30
Abbildung 34: Calculate-Baustein Station 1 (1) 30
Abbildung 35: Calculate-Baustein Station 1 (2) 31
Abbildung 36: Calculate-Baustein Station 1 (3) 31
Abbildung 37: Pi für Rotation berechnen....32
Abbildung 38: Triggerquelle 33
Abbildung 39: Sensoreinstellungen 33
Abbildung 40: Objektdefinition 34
Abbildung 41: Referenzebene 34
Abbildung 42: Arbeitsbereich....35
Abbildung 43: Ausgabekonfiguration....35
Abbildung 44: Prozessschnittstelle 36
Abbildung 45: Gesamttest....36
Abbildung 46: Berechnete Werte 36
Abbildung 47: Variable Position (1) und (2)....37
Abbildung 48: Variable Position (3) und (4)....37

Abbildung 49: Positionsabmessung Husky 39
Abbildung 50: Ermittlung des Spielraumes 40
Abbildung 51: Fehlerquelle Inverse Kinematik 41

Quellenangabe

-Universal Robots: s3-eu-west-1.amazonaws.com/ur-support-site/77201/99453_UR10e_User_Manual_de_Global.pdf

-ifm 3D Kamera: https://www.ifm.com/de/de/product/O3D304

-Softwarehandbuch ifm: media.ifm.com/CIP/mediadelivery/asset/f23ddeb9155da54b95c2a22815f92fbd/7391232_DE.pdf?contentdisposition=inline

-SPS Screenshots aus Siemens TIA Portal V.18