Gesture2Robot

Projekt 3D-Erfassung SoSe 2023, Betreuer: Prof. Dr. Varanasi, Projektteam: Benedikt Beigang, Elmar Kresse, Jan Philipp Seeland

Benutzerhandbuch und Dokumentation

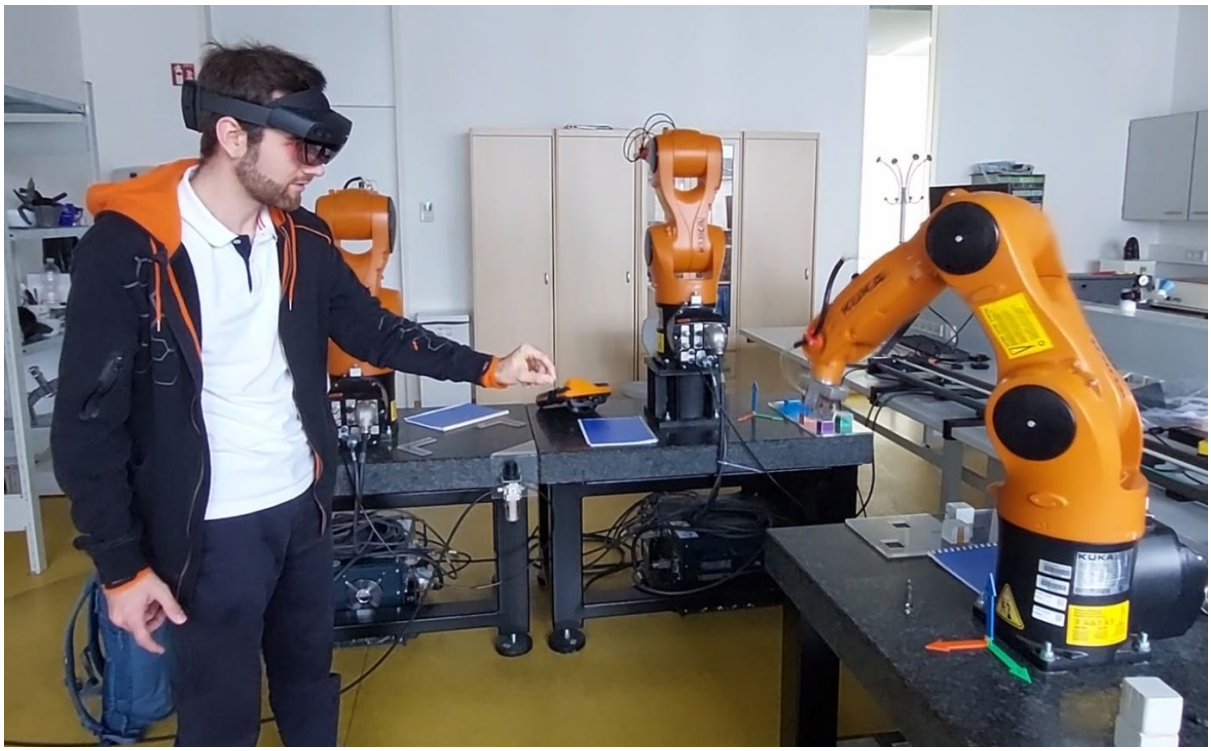
Rev. A-160823/DE

Inhaltsverzeichnis

1 Projektbeschreibung
2 Benutzerhandbuch
    2.1 Hardwarevoraussetzungen
    2.2 Softwarevoraussetzungen
        2.2.1 Hololens Applikation
        2.2.2 Netzwerk
        2.2.2 Controller Software
        2.2.3 Roboterprogramm
    2.3 Bedienung
        2.3.1 Positionierung des digital Twins
        2.3.2 Freie Bewegung des Roboters
        2.3.3 Ansteuern des Tools
        2.3.4 Ausblenden des digital Twins
        2.3.5 Bewegung mit UI-Slidern
        2.3.6 Teach-in Programmierung
        2.3.7 Speichern und Laden der Teach-in Programmierung
3 Dokumentation
    3.1 KRL Programm zur Bewegungsausführung
    3.2 Programm zur Bewegungsberechnung
    3.3 Raspberry Pi
    3.4 TCP Kommunikation
    3.5 User Interface mit MRTK-2
        3.5.1 Unity-Projekt
        3.5.2 Benutzeroberfläche und Komfortfunktionen

1 Projektbeschreibung

Das Anlernen von modernen Industrie-Robotern wie von KUKA oder Fanuc stellt häufig eine Herausforderung dar, denn das Anlernen einer Bewegungsabfolge geschieht über ein Teach-pendant (eine Art klobiges Tablet für Industrieanwendungen). Dieses ist oft mit veralteter Software ausgestattet, sodass wir uns zum Ziel gesetzt haben, diesen Prozess in das 21. Jahrhundert zu hieven.
Unsere Idee ist ein KUKA-Roboter, sowie parallel seinen "Digital Twin" mithilfe von Gestensteuerung zu bewegen. Dies ermöglicht es, auf einfache Art und Weise ohne Programmierkenntnisse den Roboter anzulernen.



Der derzeitige Softwarestand demonstriert die erfolgreiche Ansteuerung eines KUKA Industrieroboters mithilfe der Gestenerkennung der HoloLens 2. Es wurden die wichtigsten Komfortfunktionen realisiert, sodass der Roboter angelernt werden kann. Im Kapitel *Bedienung* gehen wir näher auf alle Funktionen ein. Im Kapitel Dokumentation gehen wir außerdem auf die verwendete Hardware, die Protokolle und internen Berechnungen ein, die nötig waren, um den KUKA zum Leben zu erwecken.

2 Benutzerhandbuch

2.1 Hardwarevoraussetzungen

Zur Inbetriebnahme der Gesture2Robot Software wird folgende Hardware benötigt:

- KUKA Sunrise Cabinet
- Microsoft Hololens 2
- Netzwerkschnittstelle (Router, AP, RasPI)

2.2 Softwarevoraussetzungen

2.2.1 Hololens Applikation

Die Executable (.appx) zur Installation auf der Hololens ist im GitLab-Repository unter den Releases zu finden. Dort kann die aktuelle Version der Software heruntergeladen und auf der Hololens 2 installiert werden.

GitLab Link: https://gitlab.imn.htwk-leipzig.de/22-INM/3d-erfassung-projekt

2.2.2 Netzwerk

Um die Hololens mit der Robotersteuerung verbinden zu können, muss eine Netzwerkverbindung über einen Router oder Access-Point zwischen beiden Geräten hergestellt werden.

	Die KUKA Steuerung verfügt über **KEINE Firewall**. Es wird davon abgeraten, den Roboter in einem öffentlichen Netzwerk zu betreiben. Die Absicherung des Netzwerks liegt in der Verantwortung des Nutzers.

Es wird empfohlen, ein über USB betriebenes Gerät zu verwenden, da die bereits vorhandenen Ports an der Steuerung verwendet werden können. Hierzu eignet sich beispielsweise ein Raspberry PI, dessen LAN- und WLAN-Schnittstelle als AP umkonfiguriert werden.

2.2.2 Controller Software

Zum Empfangen der Positionsdaten über die Hololens wird der

C3 Bridge Interface Server

verwendet.

Diese Software läuft im Hintergrund auf der Robotersteuerung und sollte mit dem Betriebssystem automatisch gestartet werden. Die Software ist frei erhältlich und wurde mit dem Release 1.4.0 getestet.

GitHub Link: https://github.com/ulsu-tech/c3bridge-server

2.2.3 Roboterprogramm

Auf der Steuerung muss ein Roboterprogramm in Schleife laufen, welches immer wieder die Roboter Variablen abfragt und auf Änderungen reagiert.
Dabei müssen mehrere Schritte erfüllt sein:

1. Anlegen der lokalen Variable (siehe Dokumentation 2.2.1)
2. Roboter befindet sich im AUTO Modus
3. KRL Programm auf dem Teach Pendant läuft

Das KRL Programm befindet sich am gleichen Ort wie die Hololens Applikation im Repository: https://gitlab.imn.htwk-leipzig.de/22-INM/3d-erfassung-projekt

Beachten Sie, dass das Roboterprogramm aus 2 Dateien besteht, welche sich im gleichen Ort auf dem Roboter befinden müssen.

2.3 Bedienung

Im Folgenden wird erklärt, wie mit dem digitalen Zwilling gearbeitet werden kann, um diesen und den echten KUKA-Roboter zu steuern.

2.3.1 Positionierung des digital Twins

Da der Nullpunkt der Szene bei jedem Start der Software an einer anderen Stelle sein kann, abhängig von der Position des Anwenders im Raum, kann der digitale Zwilling in der Szene bewegt werden. Dazu kann die *weiße Platte* unter dem Roboter gegriffen und verschoben werden. So kann der Anwender nach Wunsch den digitalen Zwilling über den echten Roboter legen oder daneben.

2.3.2 Freie Bewegung des Roboters

Die Spitze des digitalen Zwillings kann gegriffen werden. So kann diese grob an eine beliebige Position verschoben und rotiert werden. Der Roboter passt entsprechend seine Achswinkel an und bewegt sich zu dieser Stelle.

2.3.3 Ansteuern des Tools

Um das Tool an der Spitze der Roboterarme anzusteuern, kann der Knopf namens *Tool* verwendet werden. Mit diesem lässt sich der Greifer öffnen und schließen.

2.3.4 Ausblenden des digital Twins

Mit dem Knopf *Hide Digital Twin* ist es möglich den digitalen Zwilling auszublenden. Dies ermöglicht je nach Lichtverhältnissen eine leichtere Positionierung des Kuka-Roboters.

2.3.5 Bewegung mit UI-Slidern

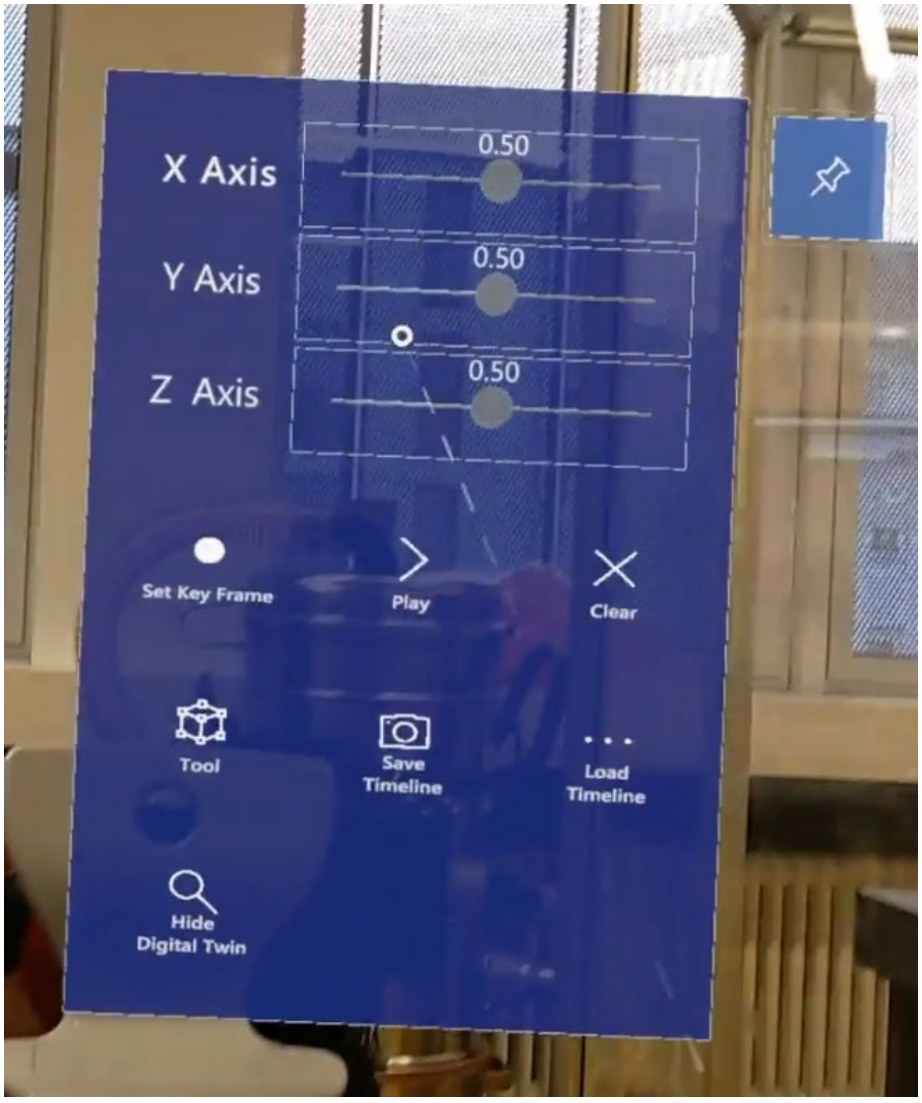
Für eine feinere Positionierung sind auf der blauen Benutzeroberfläche drei Slider, mit denen in X-, Y- und Z-Richtung der Roboter bewegt werden kann.

2.3.6 Teach-in Programmierung

Mit einer Teach-in Programmierung ist es möglich eine Folge von Positionen abzufahren und damit den Roboter anzulernen. Auf der blauen Benutzeroberfläche sind drei Knöpfe zu sehen: *Set Keyframe*, *Play* und *Clear*. Ist der Roboter auf einer Position, die abgefahren werden soll, so kann mit *Set Keyframe* die Pose gespeichert werden. Nun kann der Roboter auf die nächste Position gezogen werden, um neue Posen zu speichern, solange bis alle nötigen Punkte abgefahren wurden. Mit dem Knopf *Play* kann anschließend das Nachfahren aller Positionen gestartet werden. Mit dem Knopf *Clear* können alle bisher gespeicherten Posen gelöscht werden.

2.3.7 Speichern und Laden der Teach-in Programmierung

Mit dem *Save Teach-in* und *Load Teach-in* Knopf ist es möglich, eine gelernte Abfolge von Positionen abzuspeichern, sodass diese auch beim Neustarten der Software noch zur Verfügung stehen.



3 Dokumentation

Technische Präsentation:
https://docs.google.com/presentation/d/1YEPjAY1UzZVU2uuMi_DlWRbjr9ujvvkMIf6JgkPkOl8/edit#slide=id.g25bb57828a4_4_7

Hardware	
Microsoft HoloLens 2	AR-Headset
Kuka Roboter	Industrie-Roboter
Windows Computer	Entwicklungsumgebung
Raspberry PI	Schnittstelle zwischen HoloLens und Kuka

Software	
Unity	Entwicklungsumgebung für die AR-Anwendung
Visual Studio 2022	Bereitstellen und Kompilieren der Unity Anwendung für die HoloLens
Mixed Reality-Toolkit 2	Diverse Komponenten und Funktionen, die zum Beschleunigen der Entwicklung von plattformübergreifenden MR-Apps in Unity dienen

Das 3D Modell des KUKA Roboters wurde von der Herstellerseite bezogen:

- https://www.kuka.com/de-de/services/downloads?terms=Language:de:1;Language:en:1;product_name:KR%206%20R700%20sixx;&q=KR%20Agilus%20Series%20step

Der Kuka Roboter wurde durch die folgende Software erweitert

- C3 Bridge Interface Server
- KRL Programm zum Lesen der Variablen und Bewegungsausführung

Zur Kommunikation zwischen der Hololense und dem Roboter wurde der Raspberry PI genutzt.

- Raspberry Pi OS Lite
- DHCP und Routing Software (**dnsmasq, hostapd, iptables**)

Für die Entwicklung der Software mit der Hololense wurde das MRTK-Toolkit 2 verwendet. Für die Anwendungsfelder der GestureToRobot Anwendung sind Funktionen wie das Eingabesystem und Bausteine für räumliche Interaktionen und die Benutzeroberfläche sehr hilfreich.
So konnte die Hand- und Gestenerkennung mithilfe des Frameworks realisiert werden. Das testen der Anwendung ist durch das Ausführen in Unity und

- Ermöglicht **schnelle Prototyperstellungen** mithilfe von Simulationen im Editor, die Ihnen die Möglichkeit geben, Änderungen sofort zu sehen.
- Fungiert als **erweiterbares Framework**, das Entwicklern die Möglichkeit zum Austausch von Kernkomponenten bietet.

Steuerungs Variablen des Kuka Roboter Systems KRL (Kuka Robot Language)
https://openkuka.github.io/krl/reference/system.variables/

3.1 KRL Programm zur Bewegungsausführung

Um die über das C# Script des Digitalen Zwillings errechneten Achswinkel zu übertragen, wird eine Benutzerdefinierte Variable über die TCP Verbindung der HoloLens geschrieben. Diese Variable wurde der Datei **“$config”** auf dem KUKA Roboter hinzugefügt.
Diese Datei wurde über das Teach Pendant modifiziert und befindet sich unter dem Pfad:
R1-> SYSTEM -> $config -> [öffnen]
An der Stelle **“USER GLOBALS”** unter **“18 ; Userdefined Variables”** wurde folgende Zeile eingetragen:

```
DECL E6AXIS MRK2_AXIS_JOG = {A1 0.0, A2 90.0, A3 90.0, A4 0.0, A5
0.0, A6 0.0, E1 0.0, E2 0.0, E3 0.0, E4 0.0, E5 0.0, E6 0.0}
```

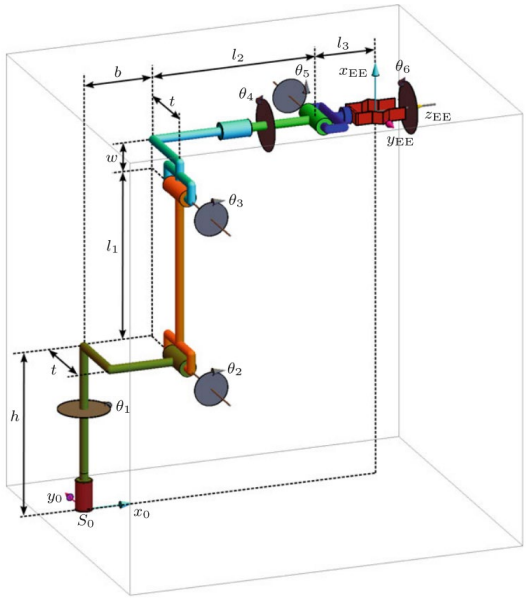
Die Steuerung des Greifers wird über das **“$FLAG[1]”** gesteuert. Diese Schnittstelle darf nicht belegt sein.
Ebenfalls muss in der Datei **R1 -> SYSTEM -> $SPS.sub** folgendes Mapping unter **USER SPS** eingetragen werden:

```
IF $FLAG[1] THEN
$OUT[4]=TRUE
$OUT[1]=FALSE
ELSE
$OUT[4]=FALSE
$OUT[1]=TRUE
ENDIF
```

3.2 Programm zur Bewegungsberechnung

Das Skript **“CalcIKsldr.cs”** enthält die DH-Parameter des Roboters und die Berechnung der Inversen Kinematik. Zusätzlich stellt es sicher, dass die Achswinkel nicht überschritten werden. Die Korrektheit der Achswinkel werden ein zweites mal beim Zusammenbauen der TCP Pakete überprüft, um eine zweite Kontrollinstanz zu haben.
Die Kugel am TCP (Tool Center Point, dt.: Werkzeugmittelpunkt) bestimmt die Zielposition im Kartesischen Koordinatensystem. Die vom Skript berechneten Winkel werden in Grad ausgegeben und manipulieren das digitale Robotermodell.

Inverse Kinematik



Für die Inverse Kinematik haben wir uns an Beispielen aus dem Internet orientiert und an unsere Bedürfnisse angepasst. Es gab ein paar Beispiele zu Implementierungen in Unity mit Robotern anderer Hersteller von welchen wir Teile verwenden konnten. Trotzdem hat es sich als sehr schwer herausgestellt, das kinematische Modell des KUKA Roboters nachzubauen. Viele Daten und Parameter waren nur spärlich zu finden.
Einige gute Quellen zur Erklärung der Inversen Kinematik ließen sich in wissenschaftlicher Literatur finden. (https://link.springer.com/chapter/10.1007/978-3-662-52759-7_3)

Die berechneten Werte werden über die TCP Verbindung an den Roboter gesendet.
Für die Kommunikation ist das Skript **“KUKA_Socket.cs”** zuständig. Hier werden die Achswinkel in eine für den Roboter interpretierbare Positionsvariable geschrieben und das TCP Paket erstellt (vgl. Kapitel TCP Kommunikation).

3.3 Raspberry Pi

Der Raspberry Pi wird im Projekt Aufbau als Vermittlungsstelle zwischen der Hololens verwendet. Da der Kuka Roboter nur über eine LAN Schnittstelle verfügt und die Hololens als Kommunikationsschnittstelle WLAN nutzt, fungiert der Raspberry Pi als Router mit Access Point.

SSID	“G2R”
Passwort	“kuka2023”
Kuka IP Adresse	141.57.54.129
WLAN Netzwerk	192.168.1.1 - 192.168.1.255

Zur Konfiguration und Installation wurden **dnsmasq, hostapd, iptables** als Software Pakete genutzt. Der Raspberry Pi kann im Projekt auch durch jede beliebige andere Netzwerkschnittstelle ausgetauscht werden, wie beispielsweise ein WLAN-Router oder ein Firmen-Netzwerk. So könnte auch Remote eine Steuerung oder Fernwartung am Roboter vorgenommen werden.

IP RasberryPI	192.168.1.1
Benutzername	“pi”
Passwort	“kuka”

3.4 TCP Kommunikation

Zur Kommunikation mit dem C3 Bridge Interface auf dem Kuka Roboter wird eine Kommunikation mittels TCP Protokoll benötigt.
Aufgebaut wird eine Socket Verbindung in C# unter Verwendung der IP Adresse und dem entsprechenden Port der Bridge.

```
IPEndPoint(IPAddress.Parse("141.57.54.129"), 7000)
```

Gesendet werden die Informationen dann als System Variables Writes des Kuka Roboters. Das TCP Protokoll sieht dafür folgenden Header und Payload vor.

	Offset (bytes)	Size (bytes)	Type	Meaning
HEADER	0	2	UINT16	Tag ID
	2	2	UINT16	Message Length
	4	1	UINT8	Message Type (Value: 1)
PAYLOAD	5	2	UINT16	**LVN** (Length of Variable)
	7	**LVN**	STRING	Variable Name
	variable	2	UINT16	**LVV** (Length of Variable Value)
	variable	**LVV**	STRING	Variable Value

Bei der Kommunikation über TCP ist es wichtig, zu den gesendeten Anfragen/Requests auch die Antworten/Responses zu lesen bzw. zu empfangen.
In den Antworten sind wichtige Informationen wie Fehlercodes, aber auch eine Bestätigung, dass die Anfrage erfolgreich angenommen und verarbeitet wurde. In der Entwicklung am Projekt ist uns zudem auch der Fall aufgetreten, dass wenn zu viele Anfragen gesendet werden und die Antworten nicht entgegengenommen werden, der Stack für die Antworten wächst. Diese Antworten haben ein Timeout, welches dazu führt, dass diese verfallen. Wenn sich zu viele Antworten auf der Serverseite sammeln, führt dies jedoch zu einem TCP Verbindungsabbruch, was die Bedienung des Roboters dann unmöglich macht.
Gelöst wurde das Problem mit der Reduzierung der zu sendenden Nachrichten und dem Entgegennehmen und Auswerten der Antworten vom TCP Server. Dennoch hinkt der echte Roboter etwas dem digitalen nach, sodass hier weiterhin Optimierungsbedarf besteht.

3.5 User Interface mit MRTK-2

3.5.1 Unity-Projekt

Zur Einarbeitung mit der Hololense und dem MTRK-2 wurde ein Guide von Microsoft genutzt, welcher in detaillierten Schritten die Grundlagen und Verwendung der Toolkits und der Hololense mit Unity erklärt.
Aufbauend zum Guide wurden dann erste Versuche mit eigenen Objekten und Szenen erstellt, in denen Gesten und Manipulation getestet wurden.
Parallel dazu wurden in Blender und Unity erste Versionen des Kuka Modells importiert und Achswinkel gemappt.
Die “Six Degrees Freedom Inverse Kinematic” stellt den Arbeitsteil mit den meisten aufgebrachten Arbeitsstunden dar. Die Anpassung und Optimierung der Bewegungen sowie das Definieren von Achswinkel-Limits sind essenzielle Bestandteile der Software. So kann eine sichere Ausführung und Bewegung des Roboters ohne Kollision und Schäden gewährleistet werden.

3.5.2 Benutzeroberfläche und Komfortfunktionen

Im Projekt selbst wurden weitere Funktionen wie eine feingranulare Achssteuerung und ein Aufnahmemodus für die Playback-Funktion implementiert.
Für die Interaktion und Ausführung wurden UI-Elemente wie Schalter und Fenster eingefügt. Zudem kann der Roboter selbst im Raum adjustiert werden, durch Bewegung der Bodenplatte. So kann das Hologramm direkt auf den Roboter gelegt werden oder als “digital twin” (digitaler Zwilling) neben dem originalen Roboter positioniert werden.