Bei diesem Beitrag handelt es sich um einen wissenschaftlich begutachteten und freigegebenen Fachaufsatz („reviewed paper“).

doi.org/10.37544/1436-4980-2025-09-34

Kamerabasierte Planung von kollisionsfreien Pfaden für CNC-gesteuerte Industrieroboter

Kopplung von TwinCAT-CNC und ROS für CNC-Pfadplanung

M. Marquart, B. Kaiser, S. Ajdinović, A. Verl

ZUSAMMENFASSUNG In der Produktionstechnik werden vermehrt CNC-gesteuerte Fertigungszellen mit Robotern erweitert, die auch von der vorhandenen CNC gesteuert werden können. CNC-Steuerungen verfügen allerdings nicht über Funktionen zur Planung kollisionsfreier Bewegungspfade. Das Robot Operating System (ROS) bietet Algorithmen zur kollisionsfreien Pfadplanung, ist jedoch nur begrenzt industrietauglich. In diesem Beitrag wird eine Schnittstelle entwickelt, mit der TwinCAT-CNC mit den ROS-Algorithmen zur Pfadplanung erweitert werden kann.

STICHWÖRTER

Industrieroboter, Automatisierung, Handhabungstechnik

TwinCAT-ROS integration for CNC path planning – Camera-based collision-free path planning for CNC robot systems in industrial automation

ABSTRACT In production engineering, CNC-controlled manufacturing cells are increasingly extended to include CNC-controlled robots. However, CNC controllers do not provide functions for planning collision-free motion paths. The Robot Operating System (ROS) offers algorithms for collision-free path planning, but is only partially suitable for industrial use. In this paper, an interface is developed that extends the TwinCAT-CNC with ROS algorithms for path planning.

1 Einleitung

Der Einsatz von CNC-gesteuerten Werkzeugmaschinen ist in der modernen Fertigungstechnik etabliert und unverzichtbar. Diese Maschinen ermöglichen eine hochpräzise und automatisierte Bearbeitung komplexer Werkstücke. Im Kontext von Industrie 4.0 ist zudem ein deutlicher Trend hin zu einer steigenden Variantenvielfalt von Produkten zu beobachten [1]. Gleichzeitig führen Fachkräftemangel und hohe Lohnkosten dazu, dass der Automatisierungsgrad vor allem in Industrieländern kontinuierlich steigt [2]. Mit der zunehmenden Variantenvielfalt und der wachsenden Komplexität industrieller Automatisierung werden vermehrt Roboter zur Bestückung von CNC-Maschinen, zur Handhabung von Bauteilen oder zur Bearbeitung eingesetzt [3].

Bei diesen Anwendungen entstehen vermehrt Fertigungszellen, in denen Industrieroboter und CNC-Maschinen miteinander kollaborieren, etwa in [4]. In diesem Szenario von kollaborierendem Roboter und CNC-Maschine bietet die Steuerung eines Roboters mit einer CNC-Steuerung erhebliche Vorteile gegenüber einer klassischen Robotersteuerung. So lassen sich beispielsweise synchronisierte Bewegungen zwischen Roboter und CNC-Maschine deutlich einfacher umsetzen [5]. Auch wird die Programmierung erleichtert, wenn sowohl CNC-Maschine als auch Roboter in der gleichen, standardisierten und weit verbreiteten Programmiersprache (G-Code [6]) sowie im selben Koordinatensystem programmiert werden [7]. Auf diese Weise können CNC-gesteuerte Roboter nahtlos in bestehende CNC-gesteuerte Fertigungsumgebungen integriert werden. Ein weiterer Vorteil von CNC-Steuerungen gegenüber klassischen Robotersteuerungen ist der vergrößerte Look-Ahead, der besonders bei Bewegungen entlang komplexer Raumtrajektorien von Vorteil ist und eine besonders ruhige Bahnführung ermöglicht.

Ein wesentlicher Nachteil CNC-gesteuerter Roboter und industrieller Robotersteuerungen liegt in der fehlenden Fähigkeit zur automatischen Planung kollisionsfreier Bewegungen. Solche Pfade müssen bislang manuell erstellt und verwaltet werden: ein aufwendiger, unflexibler und fehleranfälliger Prozess, der besonders in komplexen und dynamischen Fertigungsabläufen eine große Einschränkung darstellt [7]. Ziel dieser Veröffentlichung ist daher, eine bestehende CNC-Steuerung um die Funktionalität zur automatischen Planung kollisionsfreier Bewegungspfade zu erweitern.

In dieser Arbeit wird die Pfadplanung mit dem Robot Operating System (ROS) umgesetzt. ROS ist ein Open-Source-Framework zur Entwicklung von Software für komplexe Robotersysteme [8]. Es ist frei verfügbar und erlaubt somit die Realisierung kostengünstiger Automatisierungslösungen [9]. Durch die große und aktive Community bietet ROS Zugang zu zahlreichen vorgefertigten Bibliotheken (etwa “Nav 2” [10], “MoveIt” [11], “ros2_control” [12] oder ROS Industrials “Tesseract” [13]) und Schnittstellen unter anderem zu modernen Pfadplanungsalgorithmen aus der Open Motion Planning Library OMPL [14] sowie zu “Chomp” [15] oder “Stomp” [16]. Zudem erleichtern offene Kommunikationsschnittstellen die Integration von Kamerasystemen und den Zugriff auf bestehende Bildverarbeitungs- und KI-Algorithmen.

Trotz dieser Vorteile ist ROS in der industriellen Automatisierung bislang kaum verbreitet [17]. Ein Grund ist die fehlende

herstellerseitige ROS-Unterstützung. Der Vergleich in [18] zeigt, dass zwar viele kollaborative Roboter bereits herstellerseitige ROS-Unterstützung bieten, Industrieroboter jedoch oft nur minimale Treiber oder nur den Zugriff auf eine Schnittstelle bieten und der eigentliche ROS-Treiber nicht vom Hersteller stammt (Fanuc [19], KUKA: Robot Sensor Interface [20], ABB: Externally Guided Motion [20]).

Ein weiterer Grund für die geringe Verbreitung von ROS im industriellen Umfeld sind die hohen Echtzeitanforderungen, die für den sicheren und wiederholgenauen Betrieb notwendig sind. ROS 2 wurde als Nachfolger von ROS 1 zwar mit dem Data Distribution Service (DDS) ausgestattet, um bessere Echtzeitfähigkeit und deterministische Kommunikation zu bieten [21]. Allerdings müssen dafür die Einstellungen des nativen ROS 2 gezielt auf Echtzeit hin optimiert werden [22]. In Performance Tests von ROS 2 wurden harte Echtzeitanforderungen bei Lastspitzen nicht eingehalten [23, 24]. Zudem mangelt es ROS an einer intuitiven grafischen Nutzeroberfläche, wodurch die Einarbeitung und Verwendung von ROS mit Einarbeitungsaufwand verbunden ist [25]. Weiterhin äußert [17] Bedenken zur IT-Sicherheit bei der Verwendung von ROS im industriellen Kontext.

Gegenüber ROS bietet TwinCAT den entscheidenden Vorteil der garantierten Echtzeitfähigkeit. Als SPS (Speicherprogrammierbare Steuerung)- und CNC-System arbeitet TwinCAT mit Zykluszeiten im Kilohertzbereich [26]. Dabei werden Reaktionszeiten im Millisekundenbereich deterministisch eingehalten – ohne besondere Parametrierung oder spezielle Betriebssystemkonfiguration. Durch seine stabile Laufzeitumgebung und konsequente Trennung von Engineering- und Laufzeitebene erfüllt TwinCAT die Anforderungen an industrielle Robustheit und Prozesssicherheit.

Um die Vorteile CNC-gesteuerter Industrieroboter mit dem Funktionsumfang von ROS zu kombinieren, wird in dieser Arbeit eine Schnittstelle zwischen ROS und der TwinCAT-3-CNC-Steuerung von Beckhoff entwickelt. Ziel ist es, eine bestehende industrietaugliche CNC-Steuerung mit automatischer der automatischen Planung kollisionsfreier Pfade zu ergänzen, ohne grundlegende Anpassungen an der bestehenden Steuerungsarchitektur vorzunehmen. Die entwickelte TwinCAT-ROS-Schnittstelle wird zunächst in einer Software-in-the-Loop-Simulation an einem digitalen Zwilling getestet. Anschließend wird das Setup an einem Hardwareaufbau umgesetzt. Um die Kollisionsfreiheit auch in dynamischen Szenen robust zu gewährleisten, wird zusätzlich eine 3D-Kamera zur Szenenerfassung integriert. Die Bildverarbeitung und die kamerabasierte Pfadplanung erfolgen in ROS, während die Ansteuerung der Achsen weiterhin mit der TwinCAT-Steuerung erfolgt.

2 Stand der Technik

Die Kopplung von TwinCAT und ROS wurde bereits in verschiedenen wissenschaftlichen Arbeiten untersucht, jeweils mit unterschiedlichen Anwendungsschwerpunkten. Dieses Kapitel vergleicht Arbeiten, in denen TwinCAT in Kombination mit ROS zur Steuerung eines Roboters eingesetzt wurde.

Malecha et al. [27] verwenden ROS für die intelligente Mensch-Roboter-Kollaboration in der Fertigung von Flugzeugbauteilen mit einem KUKA-Industrieroboter. Dabei wird TwinCAT verwendet, um dem Roboter über die KUKA-RSI-Schnittstelle Gelenkwinkel vorzugeben und die aktuellen Gelenkwinkel auszulesen. CNC-Funktionalitäten werden in diesem Setup nicht verwendet. TwinCAT ist in diesem Setup nicht die zentrale Steuerung, sondern wird lediglich zur Weitergabe der Positionsdaten verwendet.

Eine ähnliche Architektur wie Malecha et al. verwenden Liang et al. [28] zur Echtzeitanbindung eines KUKA-Industrieroboters an einen digitalen Zwilling in der Baurobotik. Die TwinCAT-CNC wird hier zur Feininterpolation der Achswerte verwendet. Eine CNC-Programmierung des Roboters ist aber nicht vorgesehen. Beim Vergleich verschiedener Schnittstellen zur Kommunikation zwischen der TwinCAT-Steuerung und dem digitalen Zwilling in ROS ziehen Liang et al. das Fazit, dass Automation Device Specification (ADS) die performanteste Schnittstelle ist.

In Sterk-Hansen et al. [29] wird eine ähnliche Architektur wie in der vorliegenden Arbeit vorgeschlagen. Eine TwinCAT-Steuerung steuert die Servo-Motoren eines Knickarm-Kranes. Die TwinCAT-Steuerung kommuniziert über ADS mit der Middleware ROS. In ROS wird die „NVIDIA-Isaac-Sim" zur Simulation und Visualisierung verwendet. Eine Kamera wird direkt an den ROS-PC angeschlossen und misst die aktuelle Position des Kran-Endeffektors. Die Architektur, die Sterk-Hansen et al. vorschlagen, ist in großen Teilen ähnlich zur vorliegenden Arbeit, jedoch ist sie nicht direkt auf das hier beschriebene Problem anwendbar, da die Ansteuerung der Motoren nicht in einer CNC umgesetzt wird, sondern der Positionsregler der Motoren direkt aus der SPS mit Sollwerten versorgt wird.

Terei et al. [30] verwenden eine ROS- und TwinCAT-basierte Steuerungsarchitektur zur Steuerung eines Montageroboters für die Mikroelektronikfertigung. Der Fokus liegt auf der automatische Erstellung von kollisionsfreien Bahnen zur Programmierung eines Montageroboters, um im Industrie-4.0-Kontext mit geringen Losgrößen umgehen zu können. Für die automatische Bahnplanung wird MoveIt verwendet. Für die Simulation des Montageprozesses wird die Unity-Engine an ROS angebunden. Die zentrale Kommunikation findet über OPC UA statt. Die Peripherie und die Achsen werden direkt von TwinCAT angesteuert. Die Bildverarbeitung findet in ROS statt. Die Steuerungsarchitektur von Terei et al. ist in vielen Punkten auf die hier vorgestellte Steuerungsarchitektur übertragbar, allerdings fehlt der Fokus auf die CNC-Steuerung.

Es zeigt sich, dass TwinCAT bereits in verschiedenen Anwendungsfällen mit ROS kombiniert wurde. Jedoch beschränken sich die bisherigen Arbeiten auf die Ansteuerung einzelner Roboter in spezifischen Szenarien. Eine allgemeine Lösung, mit der eine bestehende TwinCAT-CNC-gesteuerte Fertigungszelle um die automatische Pfadplanung von ROS erweitert werden kann, fehlt bislang und wird daher in dieser Arbeit untersucht.

3 Systementwurf

Die in diesem System verwendeten Komponenten und deren Schnittstellen sind in **Bild 1** zu sehen.

In Rot dargestellt sind die Hardwarekomponenten der Roboterzelle, in der Bearbeitungs- und Handhabungsaufgaben durchgeführt werden. Die Komponenten der Fertigungszelle sind der Industrieroboter und die Kamera, die den Zustand der Umgebung erfasst. Die Kamera erkennt dabei Ablage- und Greifpositionen und erzeugt eine 3D-Karte der Zelle zur Hinderniserkennung.

In Gelb dargestellt ist die TwinCAT-Steuerung, in welcher die zentrale Ablaufsteuerung realisiert ist. Im SPS-Teil wird die Pro-

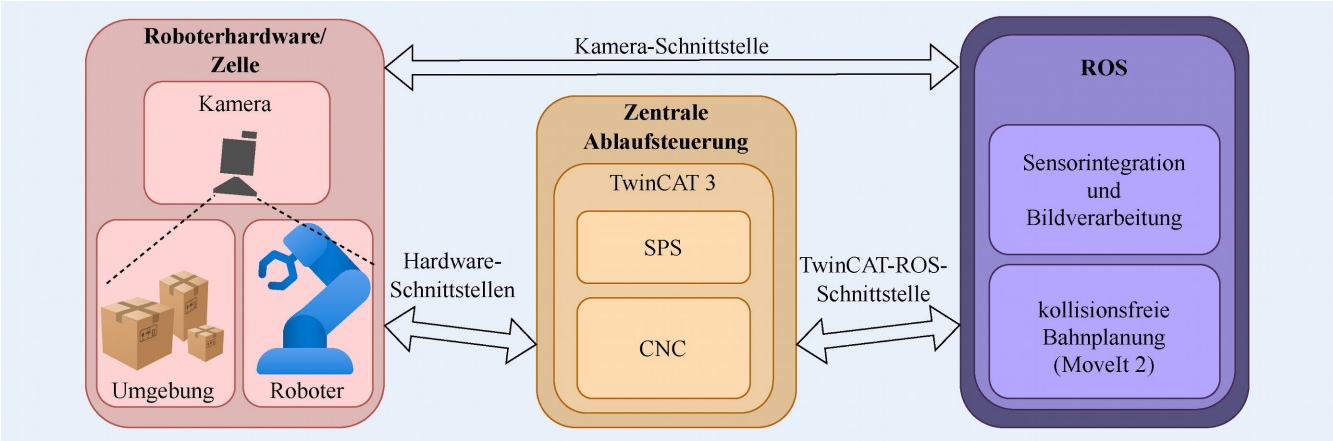

Bild 1 Komponenten und Schnittstellen des Systementwurfes. *Grafik: ISW Uni Stuttgart*

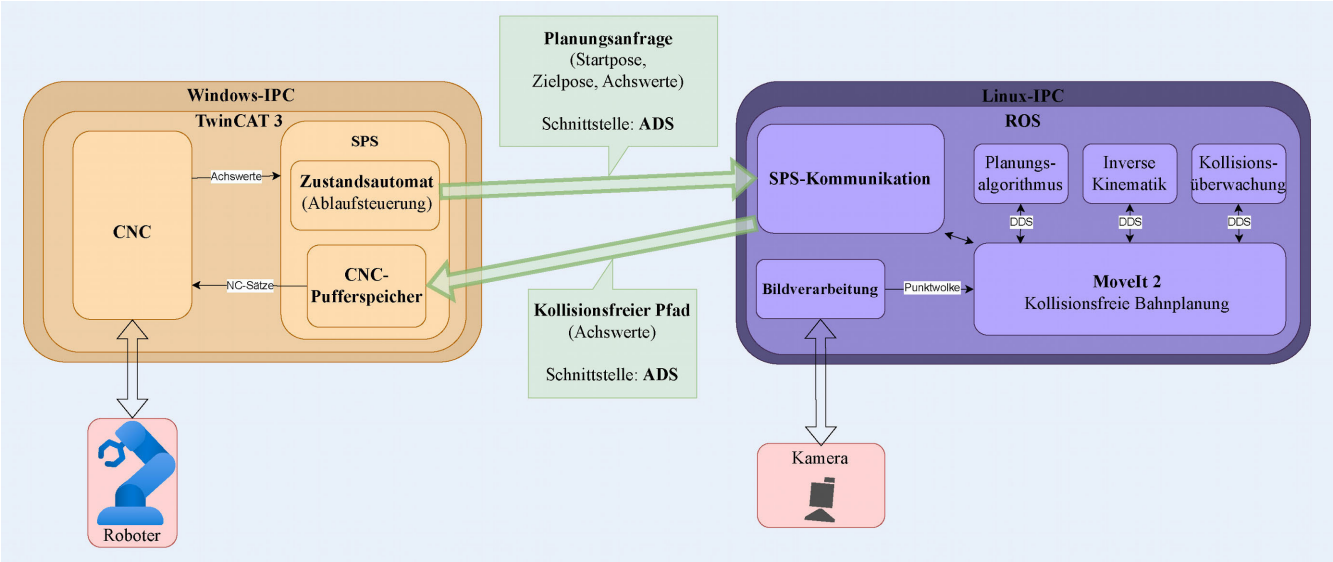

Bild 2 Softwarearchitektur der TwinCAT-ROS-Schnittstelle. *Grafik: ISW Uni Stuttgart*

zesslogik abgearbeitet, die Peripherie angesteuert und die Sicherheitsüberwachung durchgeführt. Außerdem steuert die SPS den Programmablauf in der Fertigungszelle. Im CNC-Teil findet die Bewegungssteuerung statt. Mit Hardware-Schnittstellen kann direkt auf die Positionsregelung der Antriebe zugegriffen werden.

In Blau dargestellt ist ROS als dritte zentrale Komponente des Systementwurfs. Mithilfe des Planungsframeworks „MoveIt 2" [11] erfolgt hier die kollisionsfreie Bahnplanung. Über die TwinCAT-ROS-Schnittstelle überträgt ROS den geplanten Pfad an die zentrale Ablaufsteuerung. Zur Szenenerkennung ist die Kamera direkt mit ROS verbunden, und die Bildverarbeitung erfolgt ebenso innerhalb von ROS.

4 Softwarearchitektur der TwinCAT-ROS-Schnittstelle

Dieses Kapitel beschreibt anhand von **Bild 2** die Softwarearchitektur der entwickelten Schnittstelle zwischen TwinCAT und dem Robotik Framework ROS. Dabei werden sowohl Aufbau der Schnittstelle als auch Ablauf der Kommunikation zwischen den beteiligten Systemkomponenten erläutert.

Die zentrale Ablaufsteuerung erfolgt in einem Zustandsautomaten. Die SPS empfängt kontinuierlich die aktuellen Achspositionen des Roboters von der integrierten CNC-Steuerung. Sobald ein neuer kollisionsfreier Pfad benötigt wird, sendet die SPS eine Planungsanfrage an das ROS-System. Während der Pfadplanung wird der Ablauf in der Roboterzelle unterbrochen, sodass sich der Zustand der Zelle nicht verändert und die geplante Bahn gültig bleibt. Die Planungsanfrage umfasst die Start- und Zielpose in kartesischen Koordinaten sowie die aktuelle Gelenkkonfiguration des Roboters.

Die Kommunikation zwischen TwinCAT und ROS erfolgt über ADS [31]. Dabei handelt es sich um eine TCP/IP-basierte Schnittstelle mit geringer Latenz, die den zuverlässigen Datenaustausch zwischen dem Windows- und dem Linux-basierten System ermöglicht. Die Performance von ADS wurde bereits in [32] und [26] untersucht. Die Arbeit von [32] misst für die Kommunikation von ROS über TwinCAT zu einem Servo-Motor eine maximale Latenz von 30 Millisekunden. Die Arbeit von [26] erreicht für Schreibzugriffe auf eine TwinCAT-Variable über ADS eine Frequenz von etwa 10 kHz. Da für die Schnittstelle keine harten Echtzeitanforderungen gelten und der Pfadplanungsalgorithmus

unter Umständen mehrere Sekunden Berechnungszeit benötigt [33], ist die Performance der Schnittstelle ausreichend.

Das ROS-System ist modular aufgebaut und besteht aus mehreren Knoten, die über den DDS kommunizieren. Ein dedizierter Kommunikationsknoten empfängt die Planungsanfrage von der SPS, wandelt sie in ein internes ROS-Format um und übergibt sie an das Pfadplanungsframework MoveIt 2 [11]. MoveIt ist eine weit verbreitete Open-Source-Plattform für Bewegungsplanung in der Robotik und stellt somit eine der Kernfunktionalitäten von ROS bereit [34]. Die eigentliche Planung erfolgt dort durch die Kombination verschiedener Subsysteme: ein Planungsalgorithmus, ein Lösungsalgorithmus für die inverse Kinematik und eine Kollisionsüberwachung. In MoveIt ist ein CAD-Modell der Roboterzelle hinterlegt, das für die Kollisionsprüfung verwendet wird. Zusätzlich erhält MoveIt eine aktuelle Punktwolke der realen Umgebung, welche zuvor durch eine Bildverarbeitungseinheit berechnet und über das Netzwerk bereitgestellt wurde. Diese Punktwolke wird dynamisch in die Planung einbezogen, sodass Hindernisse, die nicht Teil des ursprünglichen CAD-Modells sind, erkannt und umfahren werden können.

Die Berechnung des kollisionsfreien Pfades dauert in einem Vergleich von [33] zwischen 0,2 Sekunden und 10 Sekunden, abhängig vom verwendeten Planungsalgorithmus und der Komplexität des Szenarios. Das Ergebnis der Planung ist eine kollisionsfreie Trajektorie in Form einer Liste von Gelenkwinkeln. Diese Gelenkpositionen werden zunächst in NC (Numerical Control)-Sätze nach DIN 66025 [6] übersetzt, also in ein CNC-kompatibles Format, das von der numerischen Steuerung interpretiert werden kann. Da die Länge der übertragenen Datensätze begrenzt ist, erfolgt die Übertragung blockweise über die Streamingschnittstelle von TwinCAT-CNC [35].

Die Abarbeitung des geplanten Pfades findet schließlich in der TwinCAT-CNC statt. Dort erfolgt die Dynamikplanung unter Berücksichtigung von Geschwindigkeits- und Beschleunigungsgrenzen sowie die Interpolation der Bahnbewegung durch zyklisches Senden von Sollwerten an die Roboterachsen. Durch diese Aufteilung in eine Linux-basierte Planungsebene und eine Windows-basierte Ausführungsebene ergibt sich eine leistungsfähige und nachrüstbare Architektur, die sich gut in bestehende Industrieumgebungen integrieren lässt.

5 Validierung

Ziel dieses Kapitels ist es zu überprüfen, ob sich mit der entwickelten TwinCAT-ROS-Schnittstelle eine bestehende, CNC-gesteuerte Fertigungszelle um eine automatisierte, kollisionsfreie Pfadplanung erweitern lässt. Dazu wird zunächst eine Software-in-the-Loop-Simulation durchgeführt. In dieser Simulationsumgebung sollen verschiedene Pfadplanungsalgorithmen getestet und der Aufwand für die manuelle Programmierung mit der automatisierten Planung verglichen werden. Anschließend erfolgt die Integration der Schnittstelle in eine reale Roboterzelle, um die Praxistauglichkeit sowie den Integrationsaufwand zu evaluieren.

5.1 Software-in-the-Loop-Simulation

Eine Software-in-the-Loop-Simulation erlaubt das realitätsnahe Testen eines Steuerungssystems, indem die reale Steuerungssoftware mit einem digitalen Zwilling der Produktionsanlage gekoppelt wird [36]. Der digitale Zwilling bildet das kinematische und dynamische Verhalten der realen Fertigungszelle in Echtzeit ab. Dadurch kann das Steuerungsprogramm unter realistischen Bedingungen auf Funktionalität und Kollisionsfreiheit geprüft werden – ohne das Risiko einer Beschädigung realer Hardware. In dieser Arbeit wird die Simulationssoftware ISG-Virtuos [37] eingesetzt. Diese lässt sich nahtlos in die TwinCAT-Steuerung integrieren und ermöglicht die simulationsbasierte Abbildung des gesamten Maschinenverhaltens. Die entwickelte TwinCAT-ROS-Schnittstelle kann so effizient unter verschiedenen Bedingungen getestet werden, ohne aufwändige Umbauten an der Hardware vorzunehmen.

Zur Validierung wurde eine robotische Handhabungsaufgabe in einer dynamischen Umgebung in ISG-Virtuos modelliert (**Bild 3**).

Zwei Förderbänder liefern nacheinander unterschiedliche Werkstücke, die von einem 6-Achs-Knickarmroboter auf Paletten abgelegt werden. Dabei variieren Start- und Zielpose bei jedem Zyklus. Zudem ändern sich sowohl die Geometrie der gegriffenen Objekte als auch die Kollisionsumgebung. Dieses Szenario erlaubt eine Bewertung der Robustheit der Pfadplanung unter realistischen, dynamischen Bedingungen.

Für die Greif- und Ablagebewegung (1) wurde mit dem Pilz Industrial Motion Planner (PIMP) [38] eine lineare kartesische Bahn geplant, mit der das Bauteil von oben herab abgelegt und gegriffen werden kann. Der PIMP funktioniert ähnlich wie der Befehl zum linearen Verfahren in einer industriellen Robotersteuerung. Mögliche Kollisionen werden erkannt, aber nicht umfahren. Für die Bewegungen quer durch den Arbeitsraum (2), also für die eigentliche kollisionsfreie Pfadplanung, können die verschiedenen Planungsalgorithmen aus MoveIt verwendet werden. Der Planungsalgorithmus kann einfach ausgetauscht und so an das jeweilige Szenario angepasst werden.

Die Vorteile der automatisierten Pfadplanung werden besonders in dynamischen, komplexen Fertigungszellen deutlich. Gerade in Szenarien mit variabler Bauteillage und sich verändernden Umgebungen ist die manuelle Programmerstellung nicht praktikabel. Die automatisierte Pfadplanung stellt hier eine zentrale Voraussetzung für eine wirtschaftlich sinnvolle Automatisierung dar. Larsen schätzt den Zeitaufwand für die manuelle Programmierung eines kollisionsfreien Roboterpfades in solchen Umgebungen auf bis zu einen halben Arbeitstag. Mit der hier entwickelten TwinCAT-ROS-Schnittstelle konnte ein vergleichbarer Pfad in 5 bis 90 Sekunden (abhängig von der Komplexität der Anwendung) automatisch berechnet und somit die Effizienz deutlich gesteigert werden.

5.2 Integrationsaufwand an einer bestehenden Roboterzelle

Dieses Kapitel untersucht, welcher technische und organisatorische Aufwand nötig ist, um eine bestehende, CNC-gesteuerte Roboterzelle um die entwickelte Pfadplanung zu erweitern.

5.2.1 Bestehende Hardware

Die in **Bild 4** dargestellte Roboterzelle soll durch die entwickelte TwinCAT-ROS-Schnittstelle erweitert werden. In der Zelle ist der zweiarmige Roboter “ABB IRB 14000” (YuMi) montiert. Jeder Arm verfügt über sieben Gelenke, sodass insgesamt 14 Freiheitsgrade bei der Pfadplanung berücksichtigt

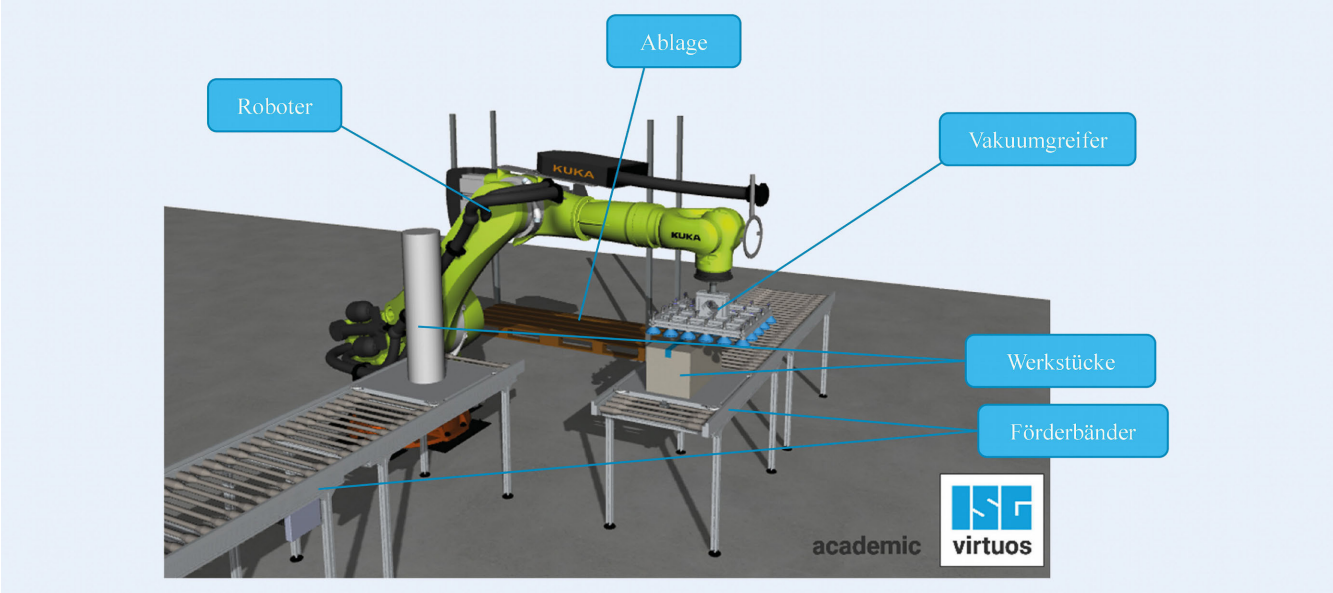

Bild 3 Software-in-the-Loop-Simulation in der Simulationsumgebung "ISG-Virtuos". *Grafik: ISW Uni Stuttgart*

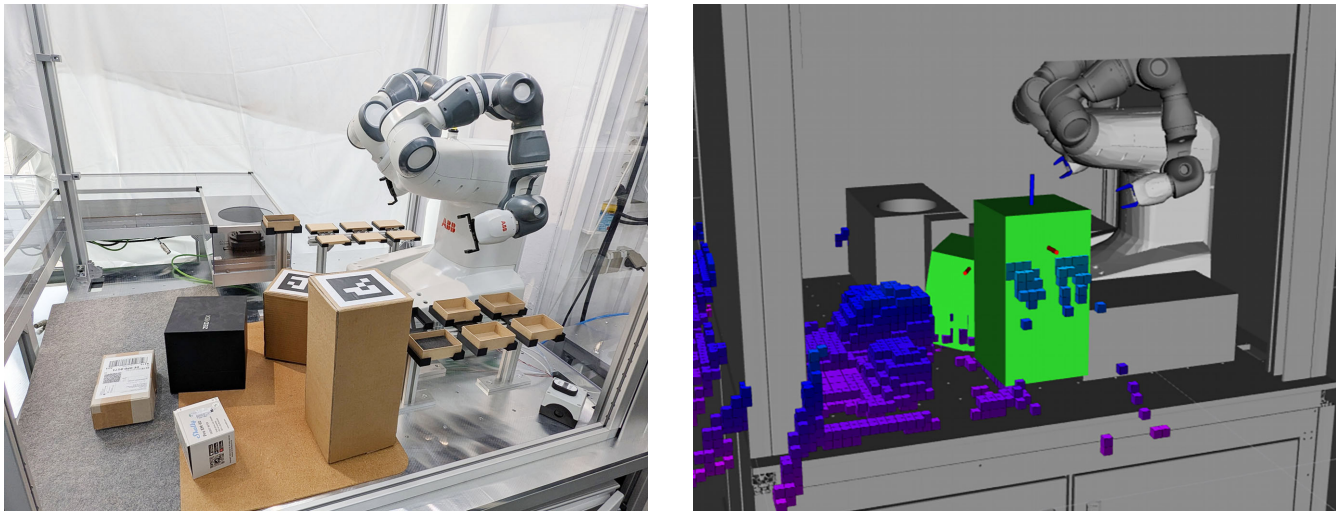

Bild 4 Links: CNC-gesteuerte Roboterzelle bei der Ausführung einer Handhabungsaufgabe. Rechts: ROS-Modell der Roboterzelle mit dem Transportsystem "XPlanar". *Foto bzw. Grafik: ISW Uni Stuttgart*

werden müssen. Dies erhöht die Komplexität der Bewegungsplanung erheblich.

Die Ansteuerung des Roboters erfolgt über die Externally Guided Motion (EGM)-Schnittstelle, bei der kontinuierlich Achspositionen von einer externen Ablaufsteuerung an den Roboter gestreamt werden. Diese zentrale Ablaufsteuerung wird mit der Software-SPS "TwinCAT 3" der Firma Beckhoff Automation umgesetzt. TwinCAT ist auf einem Windows-basierten Industrie-PC (IPC) installiert und übernimmt sowohl die echtzeitfähige Ablaufsteuerung als auch die Bewegungssteuerung der Roboterachsen. Zusätzlich werden über TwinCAT-Feldbus-Systeme, sicherheitsrelevante Komponenten und weitere Peripherie integriert. Zum Safety-System der Zelle gehören auch Lichtschranken sowie eine automatisch betriebene Hubtür, die den Zugang zur Zelle absichert. Beides ist direkt in TwinCAT integriert und angesteuert.

In der betrachteten Anwendung übernimmt der YuMi eine Montage- und Handhabungsaufgabe, bei der er mit dem CNC-gesteuerten Transportsystem "XPlanar" von Beckhoff kollaboriert. Nach der Montage durch den YuMi wird das Bauteil auf einem XPlanar-Element abgelegt. Sowohl der Roboter als auch das Transportsystem werden über TwinCAT-CNC angesteuert, wobei verschiedene Systeme miteinander kollaborieren.

5.2.2 Erweiterung der Hardware

Für die Integration der ROS-Umgebung in die bestehende Steuerungsarchitektur bestehen zwei Optionen: Die Installation von ROS auf einem separaten IPC oder auf dem bestehenden TwinCAT-Steuerungsrechner. Aufgrund der begrenzten Rechenleistung des vorhandenen TwinCAT-Steuerungsrechners und zur klaren funktionalen Trennung zwischen der Echtzeitsteuerung („TwinCAT Runtime") und der ROS-Perzeptions- und Planungsumgebung wurde ROS in dieser Arbeit auf einem separaten Linux-basierten IPC installiert. Die Kommunikation zwischen den beiden IPCs erfolgt über das maschineninterne Netzwerk.

Eine parallele Installation der TwinCAT Runtime und einer ROS-Umgebung auf demselben IPC wäre potenziell möglich. Entweder als Windows-Installation von ROS [40] oder unter Verwendung des „Windows-Subsystem for Linux" (WSL). Dies hätte den Vorteil, dass kein zusätzlicher IPC benötigt wird. Für das hier beschriebene Setup wird „ROS 2 Humble Hawksbill" [41] mit offizieller Unterstützung bis Mai 2027 auf dem Betriebssystem „Ubuntu 22.04" verwendet.

Während in der Simulationsumgebung alle Kollisionsobjekte vorher bekannt sind, muss im Realbetrieb der aktuelle Zustand der Umgebung dynamisch erfasst werden. Dafür kommt eine Stereo-Kamera („ZED X Mini") der Firma Stereolabs zum Einsatz. Die Kamera ist über GMSL2 mit einer Auswerteeinheit verbunden, die auf einem „NVIDIA Jetson Orin NX" (16 GB) basiert. Diese speziell für KI-gestützte Robotik-Anwendungen entwickelte Auswerteeinheit generiert eine Punktwolke der erkannten Objekte. Diese Punktwolke wird an den ROS-Rechner übermittelt und in MoveIt als dynamisches Kollisionsobjekt zur Pfadplanung integriert.

Fazit der durchgeführten ROS Anbindung an die bestehende Roboterzelle ist, dass sich eine bestehende, CNC-gesteuerte Roboterzelle erfolgreich mit einer ROS-basierten Pfadplanung erweitern lässt. Dazu sind folgende Erweiterungen erforderlich: ein zusätzlicher Linux-IPC mit ROS 2, Eine Kamera zur Szenenerkennung sowie die zugehörige Auswerteeinheit, Konfiguration der Kommunikationsschnittstellen (ADS, Netzwerk) und die Erweiterung der TwinCAT-Steuerung um eine Zustandsmaschine zur Prozesskoordination

Ein wesentlicher Vorteil dieses Konzepts ist, dass weder die bestehende Robotersteuerung noch das Sicherheitskonzept der Zelle angepasst werden müssen. Die erweiterte Pfadplanung arbeitet vollständig in Ergänzung zur bestehenden Ablaufsteuerung und kann somit auch in bereits validierten Industrieumgebungen nachgerüstet werden.

6 Fazit und Ausblick

Mit der entwickelten TwinCAT-ROS-Schnittstelle konnte gezeigt werden, dass sich bestehende CNC-gesteuerte Fertigungszellen ohne grundlegende Anpassungen um eine leistungsfähige, kollisionsfreie Bahnplanung erweitern lassen. Durch die Kombination der industrietauglichen, echtzeitfähigen TwinCAT-CNC-Steuerung mit den flexiblen Schnittstellen zu Planungsalgorithmen von ROS entstehen neue Möglichkeiten für die Automatisierung variantenreicher Fertigungsprozesse.

Die kollisionsfreie Pfadplanung in ROS ermöglicht eine signifikante Reduktion der Rüstzeiten und damit eine Erhöhung der Prozesseffizienz. Gleichzeitig bleiben etablierte Vorteile der CNC-Steuerung, wie Echtzeitfähigkeit, Stabilität, Sicherheitszertifizierungen und Kompatibilität mit industriellen Schnittstellen vollständig erhalten. Die vorgestellte Lösung bietet somit einen vielversprechenden Ansatz, um bestehende Automatisierungssysteme zukunftsfähig und flexibler zu gestalten – insbesondere im Kontext von Industrie 4.0 und hochvarianten Produktionsumgebungen.

In der aktuellen Implementierung generiert ROS einen kollisionsfreien Bewegungspfad zwischen einem definierten Start- und Zielpunkt. Dieser Pfad wird an die CNC-Steuerung übergeben und dort deterministisch ausgeführt. Dieses Vorgehen eignet sich insbesondere für statische Umgebungen, in denen während der Ausführung keine relevanten Änderungen zu erwarten sind.

In vielen industriellen Anwendungen verändert sich jedoch die Umgebung während des Prozesses dynamisch, etwa durch den Eingriff von Bedienpersonal, wechselnden Werkstückpositionen oder sich bewegende Betriebsmittel. In solchen Fällen besteht trotz initial kollisionsfreier Planung das Risiko einer Kollision während der Ausführung. Zukünftig soll daher eine dynamische Kollisionsüberwachung implementiert werden, die kontinuierlich die Umgebungssituation analysiert und mögliche Gefahren frühzeitig erkennt. Darauf aufbauend wird eine reaktive Kollisionsvermeidung angestrebt, die es dem System ermöglicht, während der Ausführung auf Änderungen im Umfeld zu reagieren und den Bewegungspfad adaptiv anzupassen. Dies eröffnet Perspektiven für eine noch flexiblere, robuste und sichere Interaktion von Robotersystemen mit dynamischen Fertigungsumgebungen.

FÖRDERHINWEIS

Gefördert durch die Deutsche Forschungsgemeinschaft (DFG) im Rahmen der Exzellenzstrategie des Bundes und der Länder – EXC 2120/1 – 390831618.

DANKSAGUNG

Die Umsetzung wurde unterstützt durch die ISG Industrielle Steuerungstechnik GmbH.

LITERATUR

[1] Lasi, H.; Fettke, P.; Kemper, H.-G. et al.: Industrie 4.0. Wirtschaftsinformatik 56 (2014) 4, S. 261–264
[2] UN Trade and Development: Digital Economy Report 2019. Value Creation and Capture: Implications for Developing Countries. New York: United Nations Publications 2019
[3] Soori, M.; Jough, F. K. G.; Dastres, R. et al.: Robotical Automation in CNC Machine Tools: A Review. Acta Mechanica et Automatica 18 (2024) 3, pp. 434–450
[4] Brecher, C.; Breitbach, T.; Hoffmann, F. et al.: Hybrides Bearbeitungszentrum für den Werkzeug- und Formenbau. wt Werkstattstechnik online 98 (2008) 11–12, S. 893–898. Internet: www.werkstattstechnik.de. Düsseldorf: Springer-VDI-Verlag
[5] Pfeifer, D.; Fauser, S.; Scheifele, C. et al.: Reusage of Extended Digital Twins from Virtual Commissioning for Dynamics-Based Trajectory Planning in CNC Controlled Robotics. International Conference on Flexible Automation and Intelligent Manufacturing, 2024, pp. 151–159
[6] DIN-Normenausschuss Werkzeugmaschinen: DIN 66025–1: Programmaufbau für numerisch gesteuerte Arbeitsmaschinen; Allgemeines. Deutsche Fassung, Januar 1983
[7] Zhou, Z.; Xiong, R.; Wang, Y. et al.: Advanced Robot Programming: a Review. Current Robotics Reports 1 (2020) 4, pp. 251–258
[8] Open Robotics: Open platforms for robotics. ROS. Stand: 2025. Internet: www.openrobotics.org/. Zugriff am 27.08.2025
[9] ROS: ROS – Robot Operating System. Stand: 2024. Internet: ros.org/. Zugriff am 27.08.2025
[10] Macenski, S.; Martin, F.; White, R. et al.: The Marathon 2: A Navigation System. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2020, pp. 2718–2725, doi.org/10.1109/IROS45743.2020.9341207
[11] Coleman, D. T.; Sucan, I. A.; Chitta, S. et al.: Reducing the Barrier to Entry of Complex Robotic Software: a MoveIt! Case Study. AsXiv 2014, doi.org/10.48550/arXiv.1404.3785

[12] ROS: Acknowledgements — ROS2_Control: Rolling Jul 2025 documentation. Stand: 22.07.2025. Internet: control.ros.org/rolling/doc/acknowledgements/acknowledgements.html. Zugriff am 27.08.2025

[13] Github: ROS-Industrial: Tesseract. Motion Planning Environment. Stand: 22.07.2025. Internet: https://github.com/tesseract-robotics/tesseract. Zugriff am 27.08.2025

[14] Sucan, I. A.; Moll, M.; Kavraki, L. E.: The Open Motion Planning Library. IEEE Robotics & Automation Magazine 19 (2012) 4, pp. 72–82

[15] Ratliff, N.; Zucker, M.; Bagnell, J. A. et al.: CHOMP: Gradient optimization techniques for efficient motion planning. 2009 IEEE International Conference on Robotics and Automation (ICRA), Kobe, 2009, pp. 489–494

[16] Kalakrishnan, M.; Chitta, S.; Theodorou, E. et al.: STOMP: Stochastic trajectory optimization for motion planning. 2011 IEEE International Conference on Robotics and Automation (ICRA), Shanghai, China, 2011, pp. 4569–4574

[17] Mayoral-Vilches, V.; Pinzger, M.; Rass, S. et al.: Can ROS be used securely in industry? Red teaming ROS-Industrial. ArXiv 2020, doi.org/10.48550/arXiv.2009.08211

[18] Picknick: ROS 2 Compatible Hardware. Stand: 21.07.2025. Internet: picknik.ai/hardware-ecosystem/. Zugriff am 27.08.2025

[19] Franceschi, P.; Faroni, M.; Baraldo, S. et al.: ros2 fanuc interface: Design and Evaluation of a Fanuc CRX Hardware Interface in ROS2. ArXiv 2025, doi.org/10.48550/arXiv.2506.14487

[20] Arbo, M. H.; Eriksen, I.; Sanfilippo, F. et al.: Comparison of KVP and RSI for Controlling KUKA Robots Over ROS. IFAC-PapersOnLine 53 (2020) 2, pp. 9841–9846

[21] ROS 2 Documentation: Humble documentation: DDS implementations. Stand: 23.07.2025. Internet: docs.ros.org/en/humble/Installation/RMW-Implementations/DDS-Implementations.html. Zugriff am 27.08.2025

[22] Ye, Y.; Nie, Z.; Liu, X. et al.: ROS2 Real-time Performance Optimization and Evaluation. Chinese Journal of Mechanical Engineering 36 (2023) 1, #144

[23] Gutiérrez, C. S. V.; Juan, L. U. S.; Ugarte, I. Z. et al.: Towards a distributed and real-time framework for robots: Evaluation of ROS 2.0 communications for real-time robotic applications. ArXiv 2018, doi.org/10.48550/arXiv.1809.02595

[24] Barut, S.; Boneberger, M.; Mohammadi, P. et al.: Benchmarking Real-Time Capabilities of ROS 2 and OROCOS for Robotics Applications. 2021 IEEE International Conference on Robotics and Automation (ICRA), Xi'an, China, 2021, pp. 708–714

[25] Fresnillo, P. M.; Vasudevan, S.; Perez Garcia, J. A. et al.: An Open and Reconfigurable User Interface to Manage Complex ROS-Based Robotic Systems. IEEE Access 12 (2024), pp. 114601–114617

[26] Weber, A.; Eichelberger, H.; Hildebrand, J.: ADS Performance Revisited. ArXiv 2024, doi.org/10.48550/arXiv.2410.15853

[27] Malecha, M.; Gottardi, A.; Terreran, M. et al.: Human-robot collaboration for complex draping processes of Carbon-Fibre-Reinforced Polymers for aerospace part. International Conference on Mechanical and Aerospace Engineering, ICMAE 2024. 15th International Conference on Mechanical and Aerospace Engineering (ICMAE), 2024

[28] Liang, C.-J.; McGee, W.; Menassa, C. C. et al.: Real-time state synchronization between physical construction robots and process-level digital twins. Construction Robotics 6 (2022) 1, pp. 57–73

[29] Sterk-Hansen, A.; Saghaug, B. H.; Hagen, D. et al.: A ROS 2 and TwinCAT Based Digital Twin Framework for Mechatronics Systems. 2023 11th International Conference on Control, Mechatronics and Automation (ICCMA), 2023, pp. 485–490

[30] Terei, N.; Wiemann, R.; Raatz, A.: ROS-Based Control of an Industrial Micro-Assembly Robot. Procedia CIRP 130 (2024), pp. 909–914

[31] Beckhoff Automation GmbH & Co. KG: Beckhoff Information System. ADS. Stand: 2025. Internet: infosys.beckhoff.com/index.php?content=./content/1031/tcinfosys3/11291871243.html&id=. Zugriff am 27.08.2025

[32] Domingues, B.; Costa, T. L.; Meireles, M. et al.: Protocol Performance in Robotics: Analyzing ADS vs. UDP Protocols for ROS2 and TwinCAT Integration. 2025 Brazilian Conference on Robotics (CROS), Belo Horizonte, Brazil, 2025, pp. 1–6

[33] Liu, S.; Liu, P.: Benchmarking and optimization of robot motion planning with motion planning pipeline. The International Journal of Advanced Manufacturing Technology 118 (2022) 3–4, pp. 949–961

[34] Koubaa, A. (Hrsg.): Robot Operating System (ROS). The Complete Reference, Volume 1. Cham: Springer International Publishing 2016

[35] Beckhoff Automation: Funktionsbeschreibung. TF5200 | TwinCAT 3 CNC . Data Streaming. Internet: download.beckhoff.com/download/document/automation/twincat3/TF5200_data_streaming_de.pdf. Data Streaming. Zugriff am 01.09.2025

[36] Verl, A.; Röck, S.; Scheifele, C. (Hrsg.): Echtzeitsimulation in der Produktionsautomatisierung. Beiträge zu Virtueller Inbetriebnahme, Digitalem Engineering und Digitalen Zwillingen. Heidelberg: Springer Vieweg 2024

[37] ISG Industrielle Steuerungstechnik GmbH: ISG-VIRTUOS. Die Simulationsplattform für digitale Zwillinge zur virtuellen Inbetriebnahme. Stand: 2023. Internet: www.isg-stuttgart.de/produkte/softwareprodukte/isg-virtuos. Zugriff am 27.08.2025

[38] Pilz Industrial Motion Planner: MoveIt Documentation: Rolling documentation. Stand: 30.06.2025. Internet: moveit.picknik.ai/main/doc/how_to_guides/pilz_industrial_motion_planner/pilz_industrial_-motion_planner.html. Zugriff am 27.08.2025

[39] Larsen, L.: Auf dem Weg zu einer flexiblen Produktion: automatische und kollisionsfreie Bahnplanung für kooperierende Industrieroboter. Doktorarbeit, Universität Augsburg, 2019

[40] ROS 2 Documentation: Humble documentation. Installation. Stand: 22.07.2025. Internet: docs.ros.org/en/humble/Installation.html. Zugriff am 27.08.2025

[41] Macenski, S.; Foote, T.; Gerkey, B. et al.: Robot Operating System 2: Design, architecture, and uses in the wild. Science Robotics 7 (2022) 66, eabm6074

Matthias Marquart, M.Sc.
matthias.marquart@isw.uni-stuttgart.de

Benjamin Kaiser, M.Sc.

Samed Ajdinović, M.Sc.

Prof. Dr.-Ing. Alexander Verl

Universität Stuttgart
Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen (ISW)
Seidenstr. 36, 70174 Stuttgart
www.isw.uni-stuttgart.de