

# **Intuitiv bedienbare Programmiersysteme zur effizienten Programmierung von Handhabungsaufgaben**

## **Intuitive and Efficient Programming of Material Handling Tasks**

**C. Brecher, B. Schröter, C. Almeida**, RWTH Aachen,

E-Mail: B.Schroeter@wzl.rwth-aachen.de, Tel.: 0241/80-28227

**F. Dai, B. Matthias, S. Kock**, ABB AG, Ladenburg

### **Kurzfassung**

Um die Einsatzflexibilität von Industrierobotern zur Handhabung von Werkstücken zu steigern, müssen neue Programmierkonzepte entwickelt werden, die den aus dieser Flexibilität resultierenden Anforderungen gerecht werden. Es wird daher ein Programmiersystem vorgestellt, welches durch eine Kombination von Offline- und Online-Ansätzen eine schnelle Um- oder Neuprogrammierung von Handhabungsaufgaben ermöglicht. Der Schwerpunkt bei der Entwicklung des Systems liegt auf einer intuitiven Bedienbarkeit, die keine speziellen Qualifikationsanforderungen an den Anwender stellt.

### **Abstract**

To improve the flexibility of industrial robots for material handling, there is a need for new programming concepts. Hence, a programming system is introduced that combines the traditional approaches of online- and offline-programming in order to allow fast and easy reprogramming of material handling tasks. The main objective of the programming system is an intuitive user-interface that can be operated without special qualifications.

### **1. Motivation**

Die Bestückung von Werkzeugmaschinen in kleinen und mittleren Unternehmen (KMU) wird häufig noch manuell durchgeführt. Eine der Ursachen dafür sind die hohen Anschaffungskosten für Automatisierungslösungen, die aufgrund ihrer geringen Flexibilität nur mit hohen Anpassungsaufwänden bei wechselnden Produktspektren eingesetzt werden können. Um

die vorhandenen Potenziale zur Automatisierung der Fertigung in KMU besser nutzen zu können, wird im Rahmen des Forschungsprojekts Porthos [1] ein portables Handhabungssystem für den Einsatz in der Produktion entwickelt. Dieses portable Robotersystem soll auf einfache Weise transportiert und flexibel an verschiedenen Bearbeitungsstationen eingesetzt werden können. Es soll intuitiv programmierbar sein, sich automatisch an seine Umgebung anpassen und seinen Gefahrenbereich mittels Sensoren überwachen können. Das Projekt Porthos wird mit Mitteln des BMBF innerhalb des Rahmenkonzeptes „Forschung für die Produktion von morgen“ gefördert und vom Projektträger des BMBF für Produktion und Fertigungstechnologien, Forschungszentrum Karlsruhe, betreut.

Eine der Zielsetzungen des Porthos-Projekts ist die Entwicklung einer Programmierplattform, die den Randbedingungen von KMU gerecht wird. In der Regel steht dort kein oder nur wenig Personal mit Roboter-Fachwissen zur Verfügung. Durch die Entwicklung einer intuitiven, aufgabenorientierten Programmiermethodik können robuste und betriebssichere Roboterprogramme mit einem stark reduzierten Aufwand generiert werden.

Auf bestehende Programmiersysteme kann dafür nicht zurückgegriffen werden, da sich die Erstellung von Roboterprogrammen sowohl mit Online- als auch mit Offline-Verfahren als sehr komplex gestaltet. Die *Online-Programmierung* erfolgt direkt am Roboter. Dabei wird das Programm zuerst ohne Positionsdaten erstellt, wofür Kenntnisse in der entsprechenden Roboterprogrammiersprache vorliegen müssen. Im Anschluss daran können die fehlenden Positionsdaten durch Teach-In eingelesen und in das Programm übernommen werden. Die Überprüfung der so erstellten Programme kann sich allerdings sehr aufwändig werden [2]. Bei der *Offline-Programmierung* treten andere Nachteile in den Vordergrund. So muss vor Beginn der Programmierung ein vollständiges geometrisches Modell der Roboterzelle erstellt werden. Gerade bei einem geplanten Einsatz von Industrierobotern in bestehenden Fertigungsstrukturen kann nicht davon ausgegangen werden, dass die CAD-Daten aller Komponenten bereits vorliegen. Außerdem muss bei der Verwendung von Offline-Systemen aufgrund von Unterschieden zwischen dem geometrischen Modell und der Realität mit einer geringeren Genauigkeit der Roboterprogramme gerechnet werden [3].

## **2. Anforderungen an das Programmiersystem**

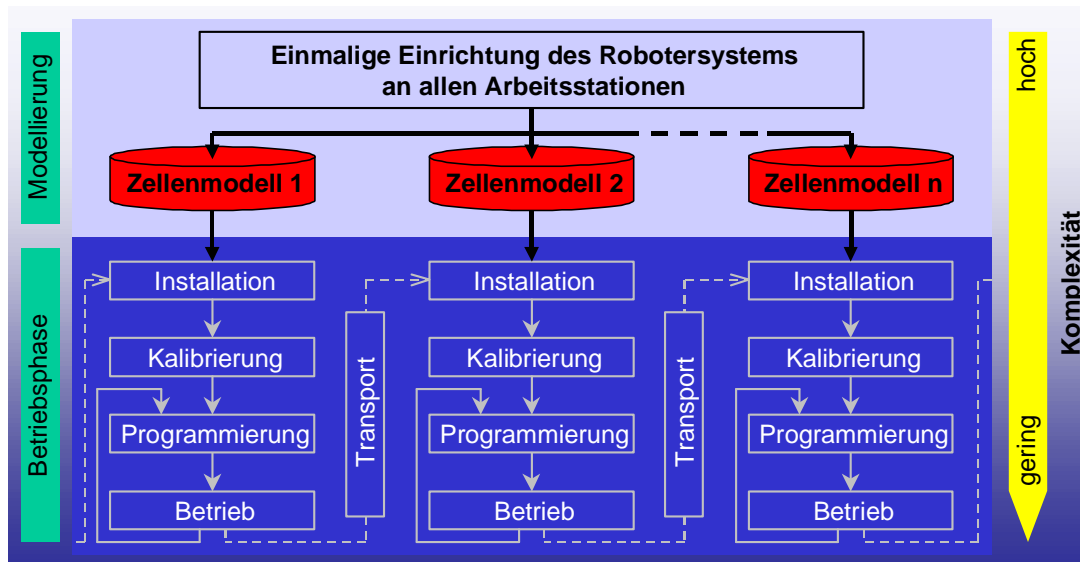
Aus den genannten Defiziten der herkömmlichen Programmierung lassen sich folgende Anforderungen an ein Programmiersystem ableiten, welches optimal für den Einsatz in KMU mit ständig wechselnden Handhabungsaufgaben ausgelegt ist.

- *Geringe Qualifikationsanforderung zur Erstellung von Roboterprogrammen*  
Im Idealfall sollte die Erstellung von Roboterprogrammen zur Werkstückhandhabung durch einen Facharbeiter mit eingeschränkten Kenntnissen auf dem Gebiet der Robotik durchgeführt werden können. Die Phasen, in denen Fachwissen benötigt wird, sind zeitlich von der Erstellung des endgültigen Programms entkoppelt.
- *Einfache Erweiterbarkeit des Programmiersystems*  
Der Einsatz des Programmiersystems in neuen Fertigungsumgebungen oder die Erweiterung um neue Roboterprogrammiersprachen muss mit minimalem Aufwand realisierbar sein. Außerdem sollte die Möglichkeit bestehen, einmal modellierte Komponenten wiederverwenden zu können.
- *Austausch des herkömmlichen Teach-Ins durch innovative Verfahren*  
Damit die Erstellung von Roboterprogrammen fehlerfrei durchgeführt werden kann, muss gewährleistet werden, dass alle dafür notwendigen Positionsdaten bekannt sind. Eine strukturierte Anleitung des Bedieners durch die Robotersteuerung beim Einlesen der Punkte kann dabei zu einer Reduzierung des Zeitaufwands führen.
- *Möglichkeit zur Integration von Kommunikationszyklen*  
Bei fast allen Handhabungsaufgaben müssen Steuer- und Synchronisationsanweisungen zwischen der Robotersteuerung und der Peripherie ausgetauscht werden. Es muss z.B. möglich sein, zu definierten Zeitpunkten Ausgänge zu setzen oder auf Eingänge zu warten. Dadurch können Bearbeitungsprozesse gestartet oder auf Freigaben gewartet werden.
- *Einfache Kalibrierung der Werkstückspeicher*  
Für den Fall, dass der Roboter oder einzelne Werkstückspeicher sich nicht an ihrer exakten Sollposition befinden, muss eine schnelle Kalibrierung dieser Komponenten möglich sein.
- *Aufgabenorientierte Programmerstellung*  
Eine Zusammenfassung der Verfahrbefehle zu aufgabenorientierten Operationen, wie z.B. „Entnehme Rohteil aus Rohteilpalette“, kann zu einer weiteren Vereinfachung der Programmierung beitragen. Aus solchen Operationen kann eine allgemeinverständliche Beschreibung des Programmablaufs zusammengesetzt und grafisch dargestellt werden.
- *Unterstützung des Bedieners*  
Die Aktionen des Bedieners sollten, wenn möglich, direkt auf Plausibilität geprüft werden. Dies kann z.B. während der Programmerstellung (auf jede Entnahme-Operation folgt eine Einlege-Operation) oder auch in vorgelagerten Phasen realisiert werden (keine Programmierung möglich, wenn Positionsdaten fehlen).

### 3. Ableitung eines geeigneten Konzepts

Die im letzten Kapitel aufgeführten Anforderungen lassen sich in Form eines zweistufigen Benutzungskonzepts realisieren (Bild 1). In der ersten Phase wird ein Datenmodell aller Arbeitsplätze erstellt, an denen das Robotersystem eingesetzt werden soll. Dieser Schritt muss im Idealfall nur einmal durchgeführt werden. Das Ergebnis sind Datenmodelle, die alle zur Programmerstellung notwendigen Informationen enthalten.

An die Modellierung schließt sich die sogenannte Betriebsphase an. Dabei wird die Handhabungsaufgabe durch die grafisch-interaktive Zusammensetzung einzelner Operationsblöcke definiert. Außerdem besteht die Möglichkeit zur nachträglichen Kalibrierung der Roboterzelle. Die Erstellung des endgültigen Roboterprogramms geschieht automatisch auf Basis der während der Modellierung eingegebenen Daten.



**Bild 1:** Zweistufiges Benutzungskonzept

Zur Erstellung des Modells können sowohl vordefinierte Komponenten als auch Standardkomponenten, die durch entsprechende Parameter an die reale Zelle anpassbar sind, verwendet werden. Um die Modellierungsaufwände so gering wie möglich zu halten, wird auf eine vollständige geometrische Beschreibung der Zelle verzichtet. Da über das Programmiersystem nur Handhabungsaufgaben programmiert werden sollen, ist es ausreichend, eine Roboterzelle auf die Komponenten „Roboter“, „Greifer“, „Werkstück“ und „Werkstückspeicher“ zu beschränken. Unter dem Begriff „Werkstückspeicher“ werden sowohl Massenspeicher, wie z.B. Euro-Paletten, als auch Einzelspeicher, z.B. in Form von Ablagen, Rüstplätzen oder Spannpositionen in Maschinen, verstanden.

Für alle Komponenten müssen im Rahmen der Modellierung nur die jeweils erforderlichen Daten eingegeben werden. Diese Daten (Bild 2) können je nach Komponente sehr unterschiedlich ausfallen.



**Bild 2:** Erforderliche Daten für die Modellierung von Roboterzellen

Sowohl die vordefinierten Komponenten als auch die Standardkomponenten werden in einer Bibliothek gespeichert. Alle in dieser Bibliothek vorhandenen Komponenten können vom Bediener zur Modellierung von Roboterzellen verwendet werden. Darüber hinaus besteht die Möglichkeit, der Bibliothek eigene, vollständig modellierte Komponenten hinzuzufügen.

Eine weitere Anforderung an das Programmiersystem war eine möglichst problemlose Erweiterung um neue Roboterprogrammiersprachen. Dazu werden alle während der Modellierung eingegebenen Daten nach roboterspezifischen und roboterunabhängigen Daten getrennt gespeichert. Im Fall der Neuaufnahme einer RC-Programmiersprache müssen daher nur die Daten neu eingegeben werden, die sich aufgrund der neuen Programmiersprache auch tatsächlich ändern. Allgemeingültige Informationen, wie z.B. der Aufbau der Werkstückspeicher oder Interpolationsarten, können übernommen werden.

Ein sehr zeitaufwändiger Arbeitsschritt während der Modellierung ist das Einlesen der notwendigen Positionsdaten. Das für dieses Programmiersystem entwickelte Konzept sieht vor, dass zuerst alle Komponenten ohne Positionsdaten vormodelliert werden. Erst zum Schluss erfolgt das Einlesen aller benötigten Daten in einem Schritt. Dazu wird der Bediener nach Möglichkeiten zur Reduzierung des Teach-Aufwands gefragt. Durch geschickte Mehrfachverwendung bestimmter Positionsdaten kann sich die Anzahl der erforderlichen Daten um mehr als die Hälfte verringern.

Nach Berücksichtigung dieser Symmetrien wird automatisch ein Roboterprogramm erstellt, welches den Bediener schrittweise zum Anfahren der notwendigen Positionen auffordert. Dabei können auch komplexe Bewegungen abgebildet werden, da der Bediener bei jeder Bewegung die Möglichkeit hat, beliebig viele Zwischenpunkte einzufügen.

Für die Realisierung der erforderlichen Kommunikation zwischen der Robotersteuerung und den beteiligten Maschinen werden nur digitale Ein- und Ausgänge verwendet, da diese Elemente an jeder Robotersteuerung vorhanden sind. Während der Abarbeitung von Handhabungsaufgaben existieren zwei verschiedene Arten von Kommunikationszyklen. Zum einen gibt es Steueranweisungen, die direkt mit einzelnen Komponenten verknüpft sind. Beispiele dafür sind das Öffnen der Maschinentür vor dem Einlegen eines Werkstücks oder die Synchronisation zwischen dem Greifer und dem Spannfutter einer Drehmaschine. Diese Kommunikationszyklen werden im Verlauf der Modellierung einzelnen Komponenten und Operationen zugeordnet und später bei der Programmerstellung automatisch berücksichtigt. Neben diesen komponentengebundenen Kommunikationszyklen gibt es auch solche, die während der Programmerstellung frei anwählbar sein sollten, z.B. eine Anweisung zum Starten eines Bearbeitungsprozesses durch die Robotersteuerung. Diese Kommunikationszyklen werden im Verlauf der Modellierung zwar vordefiniert, die endgültige Terminierung geschieht jedoch erst während der Programmerstellung.

Da alle zur Programmierung relevanten Daten nach Abschluss der Modellierungsphase eingegeben worden sind, kann die Programmierung auf eine allgemeinverständliche Beschreibung der Handhabungsaufgabe reduziert werden. Für jeden Werkstückspeicher ist eine Auswahl an Operationen definiert, die während der Programmierung grafisch-interaktiv zu einem Programmablauf zusammengestellt werden können. Zur Strukturierung des Programmablaufs stehen dem Bediener Verzweigungen, Schleifen oder Warte-Elemente zur Verfügung. Als Bedingungen können sowohl die Eingänge der Robotersteuerung als auch die Bestückungszustände der einzelnen Werkstückspeicher verwendet werden. Auch das Befüllen oder Leeren eines Werkstückspeichers kann im Programmablauf vorgesehen werden. In diesem Fall muss der Bediener während der Abarbeitung des Programms den neuen Füllstand quittieren oder ggf. korrigieren.

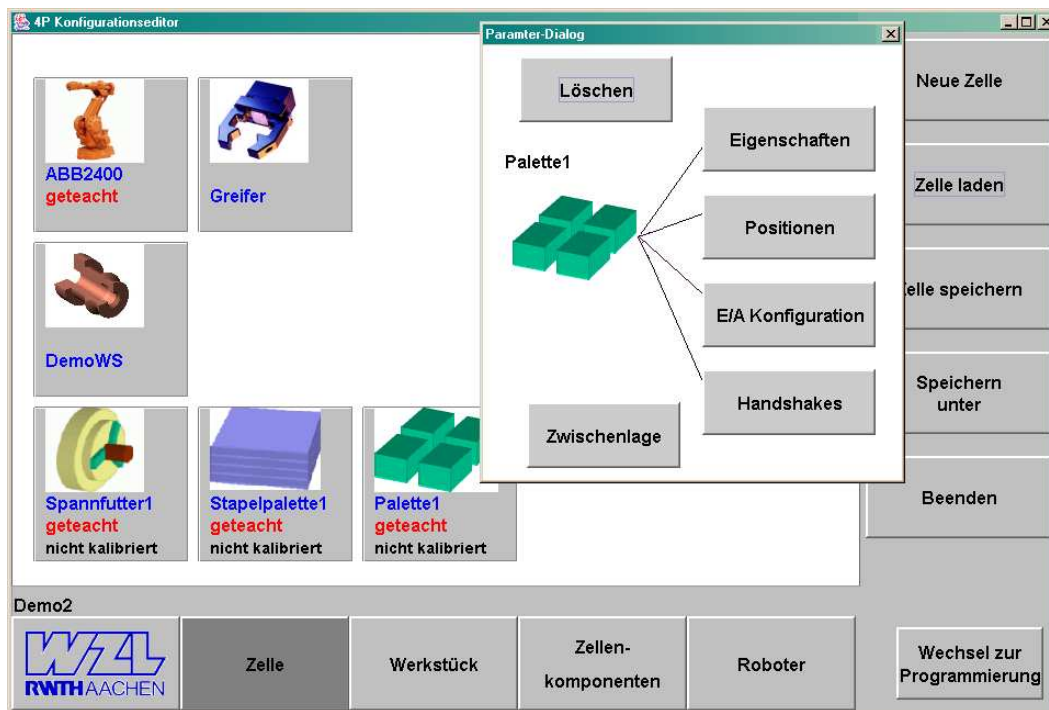
Der letzte Schritt ist die Umwandlung des Programmablaufs in das endgültige Roboterprogramm und die Übertragung auf die Robotersteuerung. Dies wird auf Basis des Zellenmodells automatisch durchgeführt.

#### **4. Prototypische Realisierung**

Zum Nachweis der Funktionsfähigkeit der vorgestellten Konzepte ist ein Prototyp implementiert worden. Bild 3 zeigt die grafische Umsetzung der Modellieroberfläche.

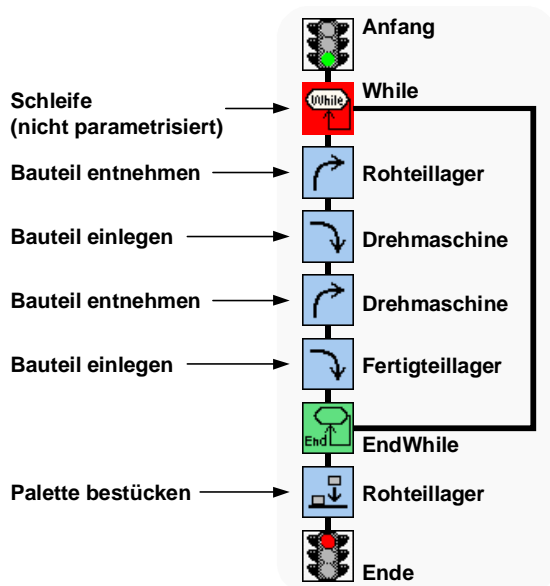
Durch Anklicken der einzelnen Komponenten werden jeweils kontextabhängige Menüs geöffnet (vgl. Bild 3), über die alle gespeicherten Daten angesehen und editiert werden können. Das Zellenmodell, d.h. alle zu einer Zelle gehörigen Daten, werden in Form eines

XML-Dokuments abgelegt. Ein entscheidender Vorteil von XML ist, dass damit auch größere Datenmengen strukturiert und plattformunabhängig gespeichert werden können.



**Bild 3:** Oberfläche zur Modellierung einer Roboterzelle

Für die Programmerstellung ist ebenfalls eine entsprechende Oberfläche implementiert worden. Die einzelnen Funktionsblöcke können dem Programmablaufplan (Bild 4) durch einfaches Anklicken hinzugefügt werden. Damit der Bediener schnell erfassen kann, ob ein Programm vollständig ist, werden noch nicht parametrierte Funktionsblöcke rot hinterlegt.



**Bild 4:** Programmablaufstruktur

Bevor das Roboterprogramm erstellt werden kann, muss noch der Anfangszustand der Zelle beschrieben werden. Dafür muss dem Programmiersystem bekannt sein, wie viele Werkstücke sich in welchen Werkstückspeichern befinden. Diese auftragsbezogenen Informationen ändern sich regelmäßig und müssen daher vor jedem Programmstart überprüft und gegebenenfalls aktualisiert werden.

## **5. Zusammenfassung und Ausblick**

Das vorgestellte Programmiersystem wurde speziell für die Programmierung von Handhabungsaufgaben entwickelt, die durch ein portables Robotersystem ausgeführt werden. Dies führt zu der Randbedingung, dass der Roboter häufig umgesetzt und damit auch neu kalibriert und programmiert werden muss.

Um den daraus resultierenden Anforderungen gerecht zu werden, ist ein zweistufiges Benutzungskonzept entwickelt worden. In der ersten Phase, der Modellierung, werden für alle geplanten Einsatzorte die für die Programmierung notwendigen Daten eingelesen und gespeichert. Die dadurch entstehenden Zellenmodelle bilden die Grundlage für die automatische Programmerstellung. Während der Programmierung ist es ausreichend, die Handhabungsaufgabe in aufgabenorientierter Form zu beschreiben (z.B. „Entnehme Rohteil aus Rohteilpalette“). Die so entstehende Programmablaufstruktur kann schließlich automatisch in ein lauffähiges Roboterprogramm übersetzt werden.

Durch die prototypische Realisierung konnte gezeigt werden, dass das Konzept prinzipiell zu einer Reduzierung des Programmieraufwandes führen kann. Es stehen jedoch noch intensive Evaluierungsphasen in konkreten, industriellen Piloteinsätzen aus.

Auch das Konzept des Programmiersystems kann noch erweitert werden. So ist z.B. die Anbindung an ein Roboter-Simulationssystem geplant. Auch ein Feedback von der Robotersteuerung soll integriert werden. Dies kann z.B. durch Darstellung eines Programmzeigers innerhalb der Ablaufstruktur geschehen, was bei der Behebung von eventuell auftretenden Fehlern während der Programmabarbeitung sehr hilfreich sein kann.

## **6. Literatur**

- [1] Porthos-Projektseite,  
URL: <http://www.porthos-roboter.de> [Stand: 14.04.2004]
- [2] Weck, M.: Werkzeugmaschinen – Automatisierung von Maschinen und Anlagen, Springer, Berlin, 2001
- [3] Weber, W.: Industrieroboter – Methoden der Steuerung und Regelung, Carl Hanser Verlag, Leipzig, 2002