Prototyping einer Fahrzeugsteuerung in virtueller 3D-Umgebung

Arnulf Braatz*, Stephan Flake**, Wolfgang Müller**, Engelbert Westkämper*
*Fraunhofer IPA, Stuttgart
**C-LAB, Paderborn

Zusammenfassung

Prototyping mit 3D-Modellen in virtueller Umgebung ist eine kostengünstige Alternative zur Entwicklung von physikalischen Modellen und Mock-ups. Für einige Anwendungen ergeben sich erhebliche Vorteile durch die Validierung in einer virtuellen Umgebung. In diesem Artikel stellen wir den Einsatz einer 3D-Animation zur Entwicklung und zum Test einer Fahrzeugsteuerung vor. Basierend auf dem Prinzip der schrittweisen Verfeinerung kann die Steuerung zunächst nur unter Berücksichtigung der Kommunikation zwischen den Basiskomponenten entworfen werden. Im zweiten Schritt wird die Ansteuerung der Aktoren unter Auswertung der Sensoren berücksichtigt. Letztendlich ist das System unter Einflussnahme der physikalischen Größen zu validieren. Wir stellen die Schnittstelle exemplarisch anhand einer Steuerung für fahrerlose Transportfahrzeuge im Szenario eines holonischen Transportsystems vor.

1 Einleitung

Prototyping mit virtuellen 3D-Modellen findet in vielen Anwendungsbereichen wie im Anlagenbau und der CAD-Konstruktion große Beachtung. 3D-Modelle sind zum einen meist erheblich kostengünstiger als der Bau eines ersten physischen Modells und zum anderen wesentlich schneller neuen Anforderungen anzupassen. Die Simulation mit virtuellen Objekten ermöglicht Untersuchungen über Modellverhalten, die an realen Prototypen oft nur mit großem technischen, zeitlichen und finanziellen Aufwand durchgeführt werden können. Reaktionen, die in Sekundenbruchteilen oder Tagen ablaufen, lassen sich im virtuellen Modell geeignet verlangsamen oder beschleunigen. An realen Objekten nicht sichtbare oder nur schwer nachvollziehbare Vorgänge können am Rechner anschaulich visualisiert werden. Kritische Situationen, wie z. B. Kollisionen, können so schon während der Konstruktionsphase am Rechner erkannt und behoben werden, ohne dass mit physischen Schäden am Objekt gerechnet werden muss.
Der nachfolgende Beitrag beschreibt eine Schnittstelle zu einem virtuellen Modell, das zur Entwicklung von Steuerungssoftware für autonome fahrerlose Transportfahrzeuge (FTF, engl. AGV: Automated Guided Vehicle) eingesetzt wird. Wir untersuchen ein Fertigungssystem zum Entgraten von Gussteilen. Werkstücke müssen mit FTFen von einem Eingangslager zu verschiedenen Bearbeitungsmaschinen (Bohren, Fräsen,

Waschen) und abschließend zu einem Ausgangslager transportiert werden. Im Zentrum der Betrachtung steht die flexible und robuste Durchführung von Werkstücktransporten.
Zur Entwicklung und schrittweisen Validierung der Steuerungssoftware stellen wir eine strukturierte Schnittstelle für die Ansteuerung der Fahrzeuge in einer virtuellen 3D-Umgebung vor. Die Schnittstelle unterstützt sowohl abstrakte Befehle, wie z. B. das Delegieren von Fahrzeugen zu Fertigungsmaschinen ohne Angabe ihrer Koordinaten, als auch sensororientierte Navigation und die Ansteuerung unter Berücksichtigung des physikalischen Verhaltens, wie z. B. Schlupf im Antrieb.
Im Folgenden gehen wir zunächst kurz auf existierende Arbeiten im Bereich des virtuellen Prototyping ein. Nach der Diskussion existierender Arbeiten führen wir in die Grundprinzipien der Fallstudie des Fertigungssystems und des fahrerlosen, holonischen Transportsystems ein. Kapitel 4 stellt den hierarchischen Entwurf einer Fahrzeugsteuerung und die damit verbundene Schnittstelle zusammen mit der Oberfläche der virtuellen Umgebung vor. In Kapitel 5 wird auf die Implementierung der Schnittstelle eingegangen und Kapitel 6 dient einigen Schlussbemerkungen.

2 Existierende Arbeiten

Virtuelle Prototypen finden mittlerweile in vielen Bereichen, wie z. B. in der Automobilkonstruktion, Fertigungsproduktion, Medizin, Gebäudearchitektur sowie in der Landschafts- und Städteplanung, ihre Anwendung. Meist steht das Aussehen einzelner Objekte (Design) oder die korrekte gegenseitige Ausrichtung oder Einpassung von Objekten (Layout) im Vordergrund. Neben dem reinen Navigieren durch die Szene und dem Betrachten von Objekten aus verschiedenen Perspektiven können oft auch Benutzerinteraktionen wie das Verschieben von Objekten, z. B. zum Zweck des Einpassens von Bauteilen in den Motorraum eines Automobils, durchgeführt werden [12].
In jüngster Zeit lässt sich ein zunehmendes Interesse an einer Validierung von Systemverhalten (Funktionalität) am virtuellen Modell feststellen. Am Heinz Nixdorf Institut (HNI) wurde z. B. die interaktive virtuelle Fahrradfabrik „Cyberbikes“ entwickelt [5], welche auch im Heinz Nixdorf MuseumsForum in einem VR-Theater vorgeführt wird [6]. Das Cyberbikes-Projekt konzentrierte sich sowohl auf die Architektur als auch auf die funktionale Darstellung eines Unternehmens, in dem die Abhängigkeiten und der Informationsfluss zwischen verschiedenen Abteilungen visualisiert und vom Benutzer erkundet werden.
Mit dem ebenfalls am HNI entwickelten Design Rewiew Werkzeug „DesiRe“ wurde im Rahmen des internationalen Projekts „European Pressurized Water Reactor“ ein virtuelles Modell eines Kraftwerks erstellt [4]. In DesiRe wird neben der Navigation durch das Modell u. a. auch Einblenden textueller Information beim Anklicken von Objekten, Messen von Abständen mit einem virtuellen Messstab und visuelle Kollisionserkennung unterstützt. Mit dem virtuellen Prototyp als Ersatz für mehrere maßstabsgetreue Modelle konnten hohe Kosten im Entwicklungsprozess eingespart werden.
Im Bereich Prototyping von Prozessen in der Fertigungstechnik wird zurzeit am Fraunhofer IPA im Rahmen des internationalen EU-Projektes „Holonic Manufacturing Systems“ (HMS) [9] eine agentenbasierte Steuerung in Verbindung mit dem

Materialflusstool QUEST® erstellt. Hierbei wird eine beispielhafte Linienfertigung aus der Automobilindustrie mit holonischen Transportfahrzeugen ergänzt, um z. B. bei Störfällen Arbeitsstationen zu überbrücken und so den Hauptmaterialfluss aufrecht zu erhalten.
Ein anderes Beispiel aus dem Bereich rechnerintegrierter Produktion liefert die Animation einer Autolackierstraße [7]. Diese Animation visualisiert die Einlagerungsstrategien eines vorgelagerten Farbsortierspeichers in einer Fahrzeuglackieranlage, um zur Minimierung der Rüstzeiten die Anzahl der Farbwechsel in den Lackierlinien zu verringern.
An der Universität Paderborn wurde in Kooperation mit dem HNI und dem C-LAB ein virtueller 3D-Prototyp bei der Entwicklung einer Steuerungssoftware für einen zweibeinigen Schreitroboter im Rahmen einer interdisziplinären Projektgruppe eingesetzt [8]. Dieses Vorgehen hatte u. a. den Vorteil, dass bei Fehlfunktionen der Steuerungssoftware während der ersten Testphasen keine Beschädigung des eigenentwickelten Roboters befürchtet werden musste.

3 Holonisches Transportsystem

Das im Rahmen dieser Arbeit verwendete Szenario basiert auf einem sog. holonischen Fertigungssystem, welches als Fallstudienszenario im Zuge der IMS Initiative[1] im TC 5 eingeführt [13] und später im Rahmen eines DFG-Schwerpunktprogramms weiter entwickelt wurde [2]. In diesem Zusammenhang wurde die Idee eines holonischen Fertigungssystems basierend auf der Lehre der Holarchien entwickelt, welche sich mit Organisation von Chaos und Komplexität befasst. Ein Holon wird als autonomer, kooperativer Teil eines komplexen Ganzen verstanden, welches nur in der Summe der Einzelteile bestehen kann [11]. Holonische Systeme zeichnen sich durch die Selbstähnlichkeit und Selbstkonfigurationsfähigkeit ihrer Komponenten aus, was zu einer hohen Fehlertoleranz und Stabilität des Gesamtsystems führt.
Als Grundkonfiguration in diesem Szenario sei eine werkstattorientierte Fertigungsumgebung zum maschinellen Entgraten von Gussteilen gegeben. Zum Entgraten dieser Werkstücke, z. B. Motorblöcke, Kurbelwellen, Krümmer u. ä. m., werden Werkzeugmaschinen eingesetzt, z. B. eine Drei-Achsen- und eine Fünf-Achsen-Fräsmaschine und zum nachträglichen Reinigen eine Waschmaschine. Ferner soll die Fertigungsanlage aus einem automatischen Hochregallager im Eingangsbereich, in welchem sich die zu entgratenden Werkstücke befinden, sowie einem Ausgangszwischenlager bestehen, an welches die fertiggestellten Werkstücke zum Zwecke der weiteren werksinternen Kommissionierung übergeben werden. Die Bearbeitungsmaschinen besitzen neben dem eigentlichen Bearbeitungsplatz für jeweils ein Werkstück einen Eingangspuffer, in welchen die zu bearbeitenden Werkstücke zwischengelagert werden können.

[1] IMS (Intelligent Manufacturing Systems) ist ein industrieorientiertes F&E-Programm, das von mehr als 250 Firmen und 200 Forschungsinstituten getragen wird, ausgerichtet auf die Entwicklung von Fertigungs- und Prozesstechnologien der nächsten Generation.

Der Materialfluss wird von freifahrenden, fahrerlosen Transportfahrzeugen realisiert, welche nicht als Befehlsempfänger einer übergeordneten PPS- (Produktions-Planungssystem) oder Dispositionssteuerung arbeiten, sondern als eigenständige, autonome und untereinander kooperierende Holonische Transportfahrzeuge (HTF, engl. H-AGV: Holonic Automated Guided Vehicle), die mit voller Entscheidungs- und Verantwortungsbefugnis für sich selbst und damit auch für den kompletten Fertigungsablauf agieren. Zusammen bilden die Fahrzeuge das Holonische Transportsystem (HTS). Die Reihenfolge der Arbeitsstationen aus der Sicht des Materialflusses ist durch die Arbeitsschritte fest vorgegeben. Es wird weiterhin angenommen, dass pro Transportfahrt aufgrund des vereinheitlichten Werkstückträgersystems vom HTF immer nur ein einzelnes Werkstück übernommen und transportiert werden kann.
Im laufenden Betrieb wird das System informationstechnisch im Wesentlichen durch die dynamische Interaktion mittels drahtloser Vernetzung (Broadcast) zwischen den HTFen und den Werkzeugmaschinen realisiert. Das Verhalten der Transportfahrzeuge und Werkzeugmaschinen, die im Folgenden auch als Stationen bezeichnet werden, ist hierbei wie folgt skizziert.

Eine Station
1. sendet eine Anforderung für eine Auslieferung an alle HTFe
2. geht zu Schritt 1, falls kein HTF nach einer bestimmten Zeitspanne geantwortet hat
3. beendet die Verhandlungsrunde nach einer definierten Zeitspanne und HTF h_i mit dem besten Angebot wird somit ausgewählt
4. geht zurück zu Schritt 1

Ein HTF
1. wartet, bis eine Auslieferungsanforderung von einer Station s_j empfangen wird
2. sendet das Angebot zur Auslieferung an s_j
3. fährt zu s_j, falls das Angebot das Beste in der Verhandlungsrunde war
4. nimmt das erste Werkstück von s_j auf, transportiert es zur nächsten Station und geht zurück zu Schritt 1

Physisch lässt sich eine Station als eine Bearbeitungseinheit mit einem Ein- und Ausgangsbuffer beschränkter Größe betrachten. Die Puffer besitzen eine Be- und eine Entladeeinheit, die z. B. als einfache Förderbänder realisiert sind. Beide Einheiten müssen mit einem Sensor (z. B. einem Tastsensor) bestückt sein, der erkennt, wenn sich ein HTF an der Einheit befindet und zum Be- oder Entladen bereitsteht. In einer Ausbaustufe wäre hier auch ein realitätsnahes Handshake-Protokoll über Punkt-zu-Punkt Funkverbindung denkbar. Die zeitliche Funktion der Bearbeitungseinheit wird nur abstrakt mit einer Bearbeitungsdauer betrachtet und umfasst keine anderen Detailgrößen wie z. B. die Rüstzeit.
Das HTF soll physisch als batteriebetriebener Serviceroboter realisiert werden. Um die Manövrierbarkeit in engen Umgebungen, wie z. B. Drehen auf der Stelle, Rückwärtsfahren oder reaktives Navigieren, zu gewährleisten, wird der Roboter durch zwei differential-getriebene Räder mit Drehmomentbegrenzung im Hauptantrieb in Kombination mit vier unabhängigen Stützrädern bewegt. Das Be- und Entladen soll durch einen 6-achsigen Industriemanipulator mit elektrischem Greifer realisiert werden. Durch eine

Erweiterung der Freiheitsgrade des Manipulatorarms mittels zwei Rotationsachsen im Fahrzeuggehäuse können so im Prinzip beliebige Positionen auf engstem Raum angefahren werden. Sicherheitstechnisch sei der Serviceroboter mit mechanischen Not-Aus-Tasten und einen orts- und kraftsensitiven Sicherheitsbumper ausgerüstet. Zur Umgebungserfassung sei an der Frontplatte ein Ultraschallsensor mit einem Abtastbereich von 60^0 befestigt. Alternativ könnte die Objekterkennung auch mit einer CCD-Kamera oder einem hochauflösenden 2D-Laserscanner realisiert sein. Sensoren und Aktoren sind modular durch einen Feldbus (CAN-Bus) verbunden und werden durch einen Master-PC gesteuert. Dies soll in der 3D-Animation durch eine Recheneinheit abstrahiert werden.

4 Virtuelles Prototyping mit 3D-Animation

Beim virtuellem Prototyping werden die Bewegungsabläufe mechanischer Systeme in 3D-Umgebungen unter Echtzeitbedingungen simuliert. Dieses Simulationsverfahren bietet enorme Vorteile sowohl hinsichtlich der relativ geringen Kosten und der zusätzlichen Visualisierungsmöglichkeiten von in der Realität nicht sichtbaren Größen als auch gegenüber der Unflexibilität und Unübersichtlichkeit mancher physikalischer Realitäten.
Im Bereich des Tests von Steuerungen am virtuellen Modell lässt sich insbesondere die ressourcenschonende Überprüfung als Vorteil festhalten. Extremsituationen und Unfälle, wie z. B. Kollisionen, können bewusst in Kauf genommen und provoziert werden, ohne dass die Gefahr der Beschädigung für Gehäuse, Antrieb, Gelenke u. ä. besteht, was am physischen Modell meist aufwendige Reparaturarbeiten zur Folge hätte.
In virtueller Umgebung lassen sich System- und Objektparameter innerhalb kürzester Zeit nahezu beliebig verändern. Im Bereich der Fertigungssteuerung können Werkzeug-maschinen individueller Größe einfach hinzugefügt oder an einen anderen Ort versetzt werden. Zur Validierung der Interaktion zwischen Servicerobotern können - sogar während der Laufzeit - beliebig viele weitere in die virtuelle Umgebung hinzugefügt werden. Virtuelle Modelle von Servicerobotern können einfach um zusätzliche Sensoren und Aktoren erweitert bzw. ersetzt werden. In der Animation können bestimmte physikalische Größen, wie z. B. Schlupf des Antriebs und die von der Masse des Fahrzeugs und Motor abhängige Beschleunigung, einfach eingestellt und verändert werden. Die interaktive Navigation bietet dem Beobachter eine übersichtliche Darstellung mit der Reduktion auf das Wesentliche. Die Möglichkeiten der Simulation realer Größen hängen jedoch sehr von den in die 3D-Animation integrierten Simulationssystemen ab. So lassen sich z. B. Verformungen und Kippverhalten von Fahrzeugen meist nicht ohne Weiteres realitätsnah in einer VR-Umgebung nachbilden, da es sich hier um eine Echtzeitdarstellung handelt.
Ein weiterer großer Vorteil beim virtuellen Prototyping liegt darin, dass das virtuelle Modell die Realität hinsichtlich Raum und Zeit fast beliebig verzerren kann. So können Abläufe im Zeitraffer oder in Zeitlupe abgebildet werden. Nichtsichtbare Größen wie Ultraschall oder Laserlicht können durch zusätzliche Effekte, Objekte oder über-

dimensionale Vergrößerung sichtbar gemacht werden, was die Beobachtbarkeit der Einzelobjekte und deren Zustände im Gesamtsystem signifikant erhöhen kann.
Letztendlich - und das wollen wir mit dem vorliegenden Artikel auch demonstrieren - kann die Animation den Prozess zur Entwicklung von Steuerungssoftware mittels schrittweiser Verfeinerung geeignet unterstützen. Die Funktionalität der Animation kann so gestaltet werden, dass sich der Entwickler zunächst auf das lokale Verhalten der einzelnen Objekte und dann auf deren Kommunikation beschränkt. In der nächsten Stufe kann dann die Umgebungserfassung und Wegeplanung betrachtet werden, während erst zum Schluss die Einflussnahme einiger physikalischer Parameter berücksichtigt werden muss. In unserem Szenario stellen wir die Schnittstellen zur schrittweisen Entwicklung der Steuerung eines holonischen Transportsystems vor. Zunächst beschränkt man sich auf das abstrakte Verhalten eines HTFs. In erster Näherung interessiert nur die Teilfunktionalität der Angebotsabgabe und Auftragsannahme. Die Animation übernimmt zunächst vollständig die Wegeplanung und Hinderniserkennung. Im nächsten Schritt werden die abstrakten Transportbefehle durch Befehle zur Ansteuerung des Antriebs mit detaillierter Wegeplanung ersetzt. Unter Auswertung der Sensoren müssen jetzt durch die Steuerungssoftware Hindernisse eigenständig erkannt und umfahren werden. In letzter Stufe kann eine Feindimensionierung der Software durch Aktivierung von physikalischen Parametern erfolgen. Abbildung 1 skizziert diesen Prozess, bei dem das HTF zunächst nur in der Grundfunktionalität implementiert wird. Einen großen Teil des Verhaltens übernimmt die Animationsumgebung. Am Ende des Entwicklungsprozesses steht die fertig erstellte Steuerung, die nur noch die Basisfunktionalität der Animation zur Visualisierung in Anspruch nimmt.

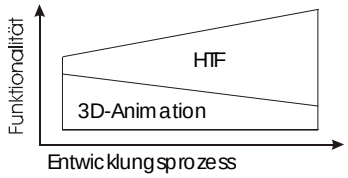

Abbildung 1. Prozess zur Entwicklung der Fahrzeugsteuerung

Entsprechend dieser Vorgehensweise stellen wir im Folgenden die Funktionalität der Schnittstelle zur Entwicklung einer HTF-Steuerung vor, unterteilt in die drei Ebenen

(a) abstraktes Verhalten,
(b) sensor-/aktororientiertes Verhalten und
(c) physikalische Parameter.

Tabelle 1 gibt einen Überblick über die wichtigsten Funktionen inklusive der Grundfunktionalität zum Erzeugen, Initialisieren und Entfernen von HTFen. Zum Beeinflussen

von Stationen steht ein ähnlicher Befehlssatz zur Verfügung, auf den aber hier nicht näher eingegangen wird.

Funktion	Parameter	Rückgabe	Beschreibung
Grundfuntionen			
CreateAGV	X, Y, Direction	HId, Status	Erzeugt neues HTF
StartAGV	HId	Status	Initialisiert erzeugtes HTF
RemoveAGV	HId	Status	Entfernt HTF
InitBattery	HId	Status	Aktiviert Batterie
CreateSTAT	X, Y, Direction, Type	SId, Status	Erzeugt Station
...			
Abstraktes Verhalten			
GetStationInCoord	SId	X, Y, Status	Liefert Koordinaten der Eingabe
GetStationOutCoord	SId	X, Y, Status	Liefert Koordinaten der Ausgabe
MoveAGVtoStationIn	HId, SId	Status	Bewegt HTF zum Entladen
MoveAGVtoStationOut	HId, SId	Status	Bewegt HTF zum Beladen
MoveAGVtoParking	HId	Status	Bewegt HTF zur Parkposition
StartAGVHandOver	HId	Status	Startet Be- oder Entladen
GetAGVBatteryState	HId	Val, Status	Liefert Batterieladezustand in %
SetAGVState	HId, Color	Status	Wechselt Farbe des HTF
IndicateComm	-	Status	Zeigt Kommunikationsaktiv. An
Sensor/Aktor Verhalten			
RelMoveAGV	HId, Direction, Distance, Speed	Status	Fährt HTF mit Gschw. u. relativer Richtung; Rückgabe bei Ankunft
AbsMoveAGV	HId, Direction, Distance, Speed	Status	Wie vorher, jedoch mit absoluter Richtungsangabe
GetAGVPosition	HId	X, Y, Status	Liefert HTF Position
GetAGVDirection	HId	Direct., Status	Liefert HTF Ausrichtung (Grad)
StopAGV	HId	Status	Stoppt HTF mit Not-Aus
SetAGVEventHdlBumper	HId	Status	Liefert Bumper-Signal
SetAGVEventHdlTouch	HId	Status	Liefert Touchsensor-Signal
GetAGVNextObstacle	HId	X, Y, Status	Liefert Pos. von nächstem Objekt
Physikalisches Verhalten			
SetAGVAccelerate	HId, Type, Acc	Status	Setzt Beschl.wert zum Anfahren
SetAGVDecelerate	HId, Type, Dec	Status	Setzt Wert zum Abbremsen
SetAGVSpeed	HId, Speed	Status	Setzt neue Geschwindigkeit
SetAGVSlip	HId, Probability	Status	Setzt Schlupf
GetAGVSlip	HId	Probabilty, Status	Liefert Schlupf

Tabelle 1. Funktionen zur Ansteuerung der 3D-Animation

4.1 Abstraktes Verhalten

Funktionen dieser Ebene dienen zur Entwicklung des abstrakten lokalen Verhaltens eines HTFs sowie der Kommunikation mit Stationen (wie in Kapitel 3 skizziert), insbesondere zur Entwicklung von Protokollen zur Auftragsvergabe der Stationen an HTFe und der damit verbundenen Optimierungsstrategien. So sind hinsichtlich der Kommunikations-

protokolle genau die Rollen des Initiators und der Responder sowie deren detaillierter Nachrichtenaustausch festzulegen und zu analysieren, um in erster Näherung u. a. Deadlocks und Lifelocks zu vermeiden. Wegesteuerung, Hinderniserkennung und -vermeidung werden hier zunächst nicht betrachtet. Das Gesamtsystem wird erstmalig hinsichtlich verschiedener Parameter wie Art und Anzahl der Stationen und deren räumlicher Partitionierung grob dimensioniert und in ersten Analysen nach Durchsatz optimiert.
Lokales Verhalten und Kommunikation der HTFe wird z. B. in Form von erweiterten endlichen Automaten beschrieben, die zur Visualisierung Animationsbefehle aufrufen. Aus Sicht eines HTFs umfasst dies nach der Aufforderung durch eine Station A die Abgabe eines (Transport-)Angebots und bei dessen Annahme durch die Station die Durchführung des Transportauftrags von Station A nach Station B. Da die Wegeplanung und Hinderniserkennung nicht berücksichtigt werden sollen, müssen diese Aufgaben zunächst von der Animationsumgebung übernommen werden.

Aufruf	Rückmeldung
CreateAGV 50.0 50.0 270.0	holon1 ok
StartAGV holon1	ok
MoveAGVtoStationOut holon1 in	ok
StartAGVHandOver holon1	ok
MoveAGVtoStationIn holon1 mill	ok
StartAGVHandOver holon1	ok
MoveAGVtoParking holon1	ok

Tabelle 2. Animationsansteuerung zum Be- und Entladen von holon1

So werden hier Befehle zur Verfügung gestellt, die ein HTF an die Ein- oder Ausgabe einer Station bewegen (MoveAGVtoStationIn/Out). Geometrische Daten werden alleine von der Animation verwaltet. Man beachte hier, dass keine metrischen Werte, sondern nur abstrakte Rastergrößen verwendet werden, die dann in der Realität in metrische Daten (z. B. 1m pro Einheit) umgesetzt werden müssen. Das Be- bzw. Entladen wird durch den Befehl StartAGVHandOver initiiert, dessen Abschluss von der Animation quittiert wird.
Tabelle 2 zeigt eine kurze Animationsansteuerung, die ein HTF (holon1) erzeugt, einschaltet, zur Station "in" bewegt, dort ein Werkstück aufnimmt und dieses bei Station "mill" ablegt.[2] Damit das HTF die Position dort nicht blockiert, besteht die Möglichkeit, es zu einer Parkposition zu bewegen, deren Verwaltung auch von der Animation übernommen wird.

[2] Zur Identifikation der Rückmeldung wird jedem Befehl beim Aufruf eine eindeutige Identifikationsnummer mitgegeben (z. B. ProzessID + Befehlszähler), die zur Identifikation der Antwort auch wieder über die Schnittstelle zurückgeliefert wird. Aus Platzgründen ist diese Identifikation in den Tabellen jedoch nicht enthalten.

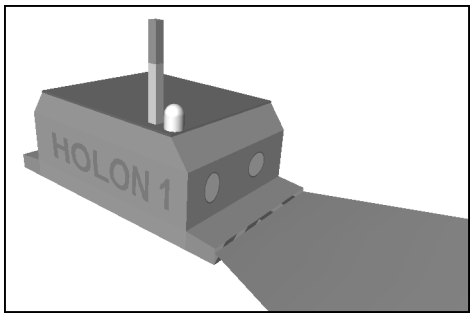

Abbildung 2. Virtuelles HTF Objekt

Die Schnittstelle stellt auf dieser Stufe neben der Darstellung und Bewegung der virtuellen Objekte einige andere wichtige Visualisierungen zur Unterstützung der Analyse zur Verfügung. So besitzt jedes HTF an der Oberseite eine Blinkleuchte, die anzeigt, wann ein HTF in Bewegung ist. Ferner befindet sich am HTF ein Stab, der den Batteriepegel darstellt (siehe Abbildung 2). Die Batterieentladung wird durch den Befehl InitBattery initiiert. Es besteht ferner die Möglichkeit, den Zustand eines HTFs individuell durch Setzen einer Farbe (SetAGVState) sichtbar zu machen. Die Verwaltung der Farben zur Kenntlichmachung interner Zustände liegt in der Verantwortung der Steuer-programme.
Der Animation kann außerdem vorgegeben werden, das Kommunikationsaufkommen über ein Broadcast-Medium zu visualisieren. Zu diesem Zweck stellt die Schnittstelle den Befehl IndicateComm bereit, der bei jeder Nachricht ein kurzes Aufblinken des Fußbodens bewirkt. Hierdurch kann ein Gefühl für den Kommunikationsverkehr und ein Hinweis auf etwaige Engpässe gewonnen werden. Diese Funktionalitäten sind optional und nur im Bedarfsfall einzusetzen, um die Visualisierung nicht mit sekundärer Information zu überladen.

4.2 Sensor-/Aktororientiertes Verhalten

Um im nächsten Schritt eine Wegesteuerung und Hinderniserkennung zu entwickeln bzw. eine existierende zu integrieren, werden auf einer weiteren Stufe Funktionen zur Verfügung gestellt, die einer Ansteuerung bzw. Rückmeldung von reellen Aktoren und Sensoren sehr nahe kommen. Dies umfasst im Wesentlichen die Fortbewegung eines HTFs. Hier werden Befehle zum Bewegen eines HTFs mit Angabe der Richtung, Geschwindigkeit und Zeit zur Verfügung gestellt. Die Richtung wird in Grad bzgl. der momentanen Position oder absolut bzgl. des globalen Koordinatensystems angegeben, was durch zwei verschiedene Befehle (RelMoveAGV und AbsMoveAGV) unterstützt wird. Dies entspricht dem ungefähren Verhalten von Antrieben, die eine bestimmte Zeit angeschaltet werden.
Wir abstrahieren hier bewusst von der Ansteuerung von zwei Antriebsmotoren, um die Schnittstelle flexibel genug bzgl. anderer Antriebsformen zu halten. Der hier zur Verfügung gestellte Befehl dürfte einfach auf die Ansteuerung von zwei oder mehr Motoren durch Implementierung von je einem Ansteuerbefehl pro Motor ersetzt werden

können. Bei dieser Ansteuerung werden idealisierte Werte zur Beschleunigung auf die Endgeschwindigkeit und zum Abbremsen angenommen (+/- Unendlich).
In der Sensorik unterscheiden wir passive und aktive Sensoren. Erstere liefern nur auf Anfrage einen Wert, wie z. B. die Objekterkennung mittels Ultraschall. Letztere liefern nach Initialisierung bei jeder Aktivierung einen Wert, wie z. B. der Bumper. In der Realisierung der Schnittstelle bedarf es bei passiven Sensoren jeweils einer expliziten (synchronen) Abfrage. Für aktive Sensoren wird zur Initialisierung ein sog. Event-Handler gesetzt. Bei Aktivierung des Sensors wird dann jeweils immer (asynchron) eine Rückmeldung ausgelöst.

Aufruf	Rückmeldung
AbsMoveAGV holon1 270 37 10	
...	ok
GetAGVPosition holon1	50 87 ok
GetAGVNextObstacle holon1	14.48003125 in
SetAGVEventHdlBumper	
...	ok
...	
...	ok

Tabelle 3. Ansteuerung der Animation auf sensor-/aktororientierter Ebene

Tabelle 3 stellt eine Beispielsequenz der Ansteuerung der Schnittstelle vor. Der erste Befehl bewegt holon1 Richtung 270^0 10 Zeiteinheiten lang mit Geschwindigkeit 37 (man beachte die Angabe von abstrakten Zeit- und Geschwindigkeitseinheiten). Nach Erreichen der Endposition wird asynchron eine Rückmeldung ausgelöst. Die weiteren Befehle stellen die absolute Position von holon1 fest, fragen die Objekterkennung ab und setzen einen Event-Handler für den Bumper. Die Objekterkennung liefert Abstand und Art des nächsten Objekts.
Visualisiert wird auf dieser Ebene im Wesentlichen die Aktivität der Sensoren. Dies umfasst bei Aktivierung der Abstandsmessung die Darstellung eines Ultraschallkegels und ein kurzes Aufblinken des nächsten erkannten Objekts. Die Aktivierung anderer Sensoren wird teilweise durch temporäre Vergrößerung oder Aufleuchten des Sensors mit einer Signalfarbe wie Rot oder Orange kenntlich gemacht. Die Form der Visualisierung hängt hier sehr stark von der Art, Größe und Lage des individuellen Sensors ab.

4.3 Physikalische Parameter

Mobile Fahrzeuge unterliegen in der realen Welt physikalischen Bedingungen, z. B. treten durch Schlupf Ungenauigkeiten bei der Durchführung von Bewegungen auf. Auch das Losfahren bzw. Anhalten erfolgt aufgrund der Massenträgheit immer mit einer positiven bzw. negativen Beschleunigung, die z. B. bei der Annäherung an Stationen berücksichtigt werden muss. Um diesem Aspekt in der virtuellen 3D-Animation gerecht zu werden, stehen hier zusätzliche Funktionen zum Setzen physikalischer Parameter zur Verfügung. Es erfolgt jedoch nur eine grobe Dimensionierung der Steuerung an die physikalischen Daten.

Im Wesentlichen umfasst die Funktionalität der Schnittstelle eine weitere Verfeinerung der Bewegung. Zum einen kann der Bewegung (Abs/RelMoveAGV) ein Beschleunigungs- und Abbremsverhalten hinzugefügt werden. Dies ersetzt die initialen idealisierten Werte mit unendlicher Beschleunigung. Des Weiteren wird ein Befehl zur Modifikation der momentanen Geschwindigkeit angeboten. Letztendlich besteht die Möglichkeit, geeignete Werte für den Schlupf im Antrieb eines individuellen HTFs unter Angabe einer Wahrscheinlichkeit zu setzen. Die Bewegung wird danach zufallsgesteuert um diesen Wert verfälscht.

Aufruf	Rückmeldung
SetAGVAccelerate holon1 log 30	ok
SetAGVDecelerate holon1 log 10	ok
SetAGVSpeed holon1 50	ok
SetAGVSlip holon1 0.003	ok

Tabelle 4. Befehlssequenz zur Ansteuerung physikalischer Parameter

Tabelle 4 stellt eine kurze Befehlssequenz zur Benutzung der Schnittstelle vor. Die ersten beiden Befehle setzen eine logarithmische Beschleunigung bzw. Verlangsamung von 30 bzw. 10 Einheiten pro Sek^2. Der dritte Befehl reduziert bzw. erhöht die Geschwindigkeit eines sich gerade in Bewegung befindlichen HTFs. Schließlich wird ein Schlupf für holon1 mit einer Wahrscheinlichkeit von 0.003 gesetzt. Ein Wert von 0 würde die Initialeinstellung wieder herstellen.
Auf der Ebene dieser Eigenschaften werden zurzeit keine Visualisierungen angeboten, da es sich um Eigenschaften mit extrem kurzfristiger Auswirkung handelt.

5 Implementierung

Bis jetzt wurden die wesentlichen Teile der im vorherigen Kapitel beschriebenen Schnittstelle implementiert. Die realitätsgetreue Nachbildung muss nur für einige Details der HTFe und Stationen noch vervollständigt werden.[3]
Die 3D-Animation wurde auf Basis der unter Windows NT und Irix verfügbaren und auf OpenInventor basierenden 3D-Animationsbibliothek AAL (Animated Agent Layer) realisiert (siehe Abbildung 3) [3]. Aus Gründen der Plattformunabhängigkeit wurde eine Kommunikationsschnittstelle auf Basis von TCP/IP zur Verfügung gestellt. Wir verwenden hier das Tcl/Tk-Binding von AAL [10]. Auf der Seite der 3D-Animation empfängt ein einfacher Kommandointerpreter Nachrichten im ASCII-Format und wandelt diese in AAL-Aufrufe um. Eine Anbindung der Schnittstelle über CORBA wird zurzeit evaluiert.

[3] In der Animation schwebt ein Werkstück zurzeit beim Entladen noch zur Station, da in der jetzigen Implementierung ein HTF noch keinen Roboterarm besitzt.

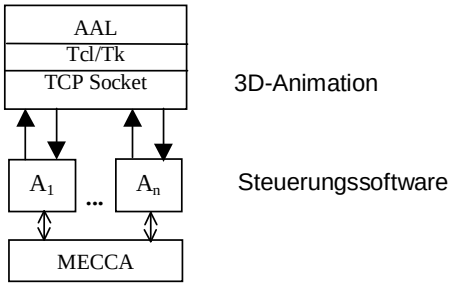

Abbildung 3. Architektur der Entwicklungsumgebung

Unsere Testumgebung zur Ansteuerung der Schnittstelle ist in Java unter Verwendung des Agentenverwaltungssystems MECCA [1] realisiert. Stationen und Holone können so als autonome Softwareagenten $A_1,...,A_n$ mit je einem Kommunikationskanal zur Ansteuerung des korrespondierenden 3D-Objektes implementiert werden. Die zugrunde liegende Architektur erlaubt aber auch die Ansteuerung der 3D-Objekte durch ein sequentielles Programm einer beliebigen Programmiersprache über Sockets.

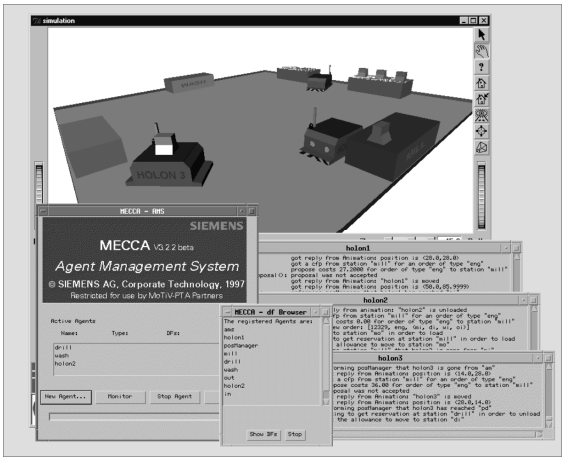

Abbildung 4. Bildschirmfoto der Testumgebung

Abbildung 4 zeigt die Testoberfläche inklusive 3D-Animation unter Windows NT mit einem zusätzlichen Fenster pro steuerndem Agent. Im Vordergrund befindet sich das Hauptfenster des MECCA Systems und rechts daneben das Fenster mit den registrierten Agenten.

6 Schlussbemerkungen

Im Rahmen der Arbeiten zur Entwicklung und Verifikation einer HTF-Steuerung leistete uns die Schnittstelle aufgrund der zur Verfügung stehenden zusätzlichen Visualisierungen bereits wertvolle Dienste. Erste Fehlfunktionalität durch Systemverklemmungen konnten sofort entdeckt und behoben werden. Im Zuge des nachfolgend genannten DFG-Schwerpunktprogramms soll die Animation anderen Projektteilnehmern zur Bearbeitung der Fallstudie zur Verfügung gestellt werden.
Im Teilprojekt GRASP, das sich mit der Modellprüfung der hier beschriebenen Fallstudie befasst, soll die Animation zur Visualisierung von Gegenbeispielen an den RAVEN Modelchecker angebunden werden.
Unsere Erfahrung in diesen Bereichen zeigte uns, dass die Entwicklung der Steuerung in den frühen Entwicklungsphasen zur Betrachtung des abstrakten und des sensor-orientierten Verhaltens von der Animation geeignet unterstützt wird. Inwieweit dies auch für das physikalische Verhalten zutrifft, können wir zurzeit noch nicht beurteilen. Hier sind wegen des Echtzeitverhaltens der virtuellen Umgebung Grenzen gesetzt. So ist auf Basis der momentanen Software z. B. die Simulation des Kippverhaltens der Fahrzeuge nur extrem aufwendig umzusetzen.

Danksagungen

Die hier beschriebenen Arbeiten wurden durch die DFG im Rahmen des Schwerpunktprogramms 1064 „Integration von Techniken der Softwarespezifikation für ingenieurwissenschaftliche Anwendungen“ als Kooperation der Projekte IOSIP und GRASP gefördert. Wir bedanken uns ferner bei Waldemar Rosenbach, der die wesentlichen Teile der 3D-Animation realisierte und uns jederzeit hilfsbereit zur Seite stand.

Literatur

[1] Bauer, B.; Steiner, D.: MECCA - System Reference Manual. Interne Dokumentation, Siemens AG, München, 1998.

[2] Braatz, A.; et al.: Referenzfallstudie Produktionstechnik v1.3: Holonischer Materialfluß, 1999. http://www.tfs.cs.tu-berlin.de/projekte/indspec/SPP/

[3] Dücker, M.; Geiger, Ch.; Hunstock, R.; Lehrenfeld, G.; Müller, W.: Visual Textual Prototyping of 4D Scenes. In: IEEE Symposium on Visual Languages, Capri, Italien, September 1997.

[4] Ebbesmeyer, P.; Gehrmann, P.; Grafe, M.; Krumm, H.: Virtual Reality for Power Plant Design. 19th ASME: Computers and Information in Engineering Conference, ASME '99, Las Vegas, 1999.

[5] Gausemeier, J.; von Bohuszewicz, O.; Ebbesmeyer, P.; Grafe, M.: Cyberbikes - Interactive Visualization of Manufacturing Processes in a Virtual Environment. Proceedings of PROLAMAT 98, Trento, Italien, 9.-11. September 1998.

[6] Gausemeier, J.; von Bohuszewicz, O.; Ebbesmeyer, P.; Grafe, M.: Gestaltung industrieller Leistungserstellungsprozesse mit Virtual Reality. Industrie Management 13, 1997. http://www.cyberbikes.hnf.de

[7] Geiger, Ch.; Hunstock, R.; Lehrenfeld, G.; Müller, W.; Quintanilla, J.; Weber, A.: Visual Modeling and 3D-Representation with a Complete Visual Programming Language - A Case Study in Manufacturing. In: IEEE Symposium on Visual Languages, Boulder, Colorado, USA, Oktober 1996.

[8] Geiger, Ch.; Lehrenfeld, G.; Müller, W.: Virtuelles Prototyping einer Robotersteuerung durch interaktive 3D-Simulation. Simulation und Visualisierung 99, Magdeburg, 4.-5. März 1999.

[9] HMS Homepage. http://hms.ifw.uni-hannover.de, 1999.

[10] Ishell Project Homepage. http://www.c-lab.de/vis/software/ishells, 1999.

[11] Koestler, A.: The Ghost in the Machine, Arkana Series, Penguin, 1967.

[12] Symietz, M.: Grundliegende Interaktionen mit virtuellen Prototypen. Simulation und Animation 97, Magdeburg, 6.-7. März 1997.

[13] Westkämper, E.; Höpf, M.; Schaeffer, C.: Holonic Manufacturing Systems (HMS) - Test Case 5. Proceedings of Holonic Manufacturing Systems, Lake Tahoe, CA, 9.-16. Februar 1994.