RoboLab

Institut für Computerwissenschaften Universität Salzburg

Helmut Mayer

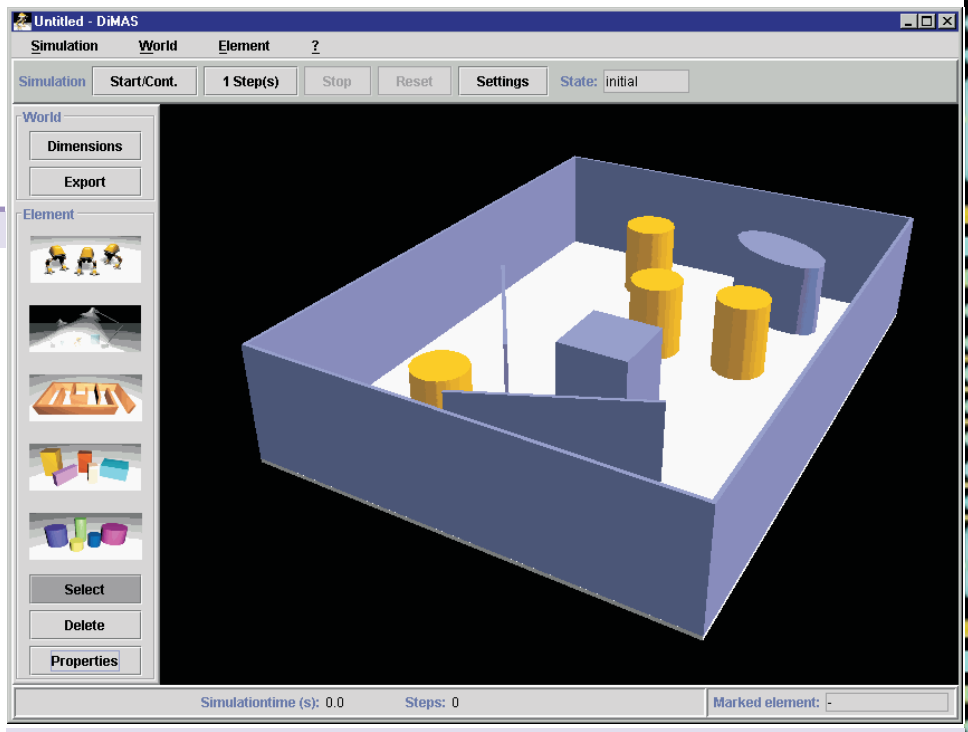


Bild 1: DiMas Screenshot, eine Welt aus Sicht des Experimentators (Zylinder sind Roboter = Agenten)

Im Experimentallabor RoboLab (http://www.cosy.sbg.ac.at/~robolab/) wurde unter Leitung von Prof. Pfalzgraf und Dr. Mayer ein mobiler, autonomer Roboter konstruiert (von der Mechanik bis zur Software), der als Ausgangspunkt für die Beschäftigung mit verschiedenen wissenschaftlichen Fragestellungen dienen soll. Der Roboter von der Größe einer Getränkedose hört auf den Namen EMMA (*Embedded Mobile Agent*), und wurde bisher als Fußballspieler(in) eingesetzt (***Bild 2***, ***Bild 4***). Roboterfußball erlangte in den letzten Jahren einen hohen Stellenwert an vielen Universitäten und industriellen Forschungseinrichtungen, da Fußball ein komplexes, dynamisches Problem darstellt, das die Fähigkeit der Roboter, sich auf ständig wechselnde Umgebungsbedingungen einzustellen, auf eine harte Probe stellt. Die Basis des "Gehirns" von EMMA ist der Infineon Mikrocontroller SAB C167CR auf dem eine *Fuzzy Rule Base* die Steuerung des Roboters übernimmt. Eine Kamera ist der derzeit einzige, aber sehr mächtige Sensor von EMMA. Das Kamerabild wird vom Mikrocontroller verarbeitet und an die *Rule Base* übergeben, die die entsprechenden Motorsignale generiert. Aufladbare Batterien gewährleisten einen völlig autonomen Betrieb beim Fußballspiel, bei dem sich im Spiel eins gegen eins auch menschliche Gegner (über eine ferngesteuerte EMMA) messen können. Die Softwareentwicklung für ein solches System gestaltet sich naturgemäß sehr aufwändig, da die winzigste Programmänderung ein Rebooten von EMMA (mit zeitaufwändigem Überspielen des Programms über eine serielle Schnittstelle) erfordert. Das Debugging besteht dann im Wesentlichen aus der Beobachtung des Verhaltens des Roboters. Wesentlich komfortabler wäre hingegen die Entwicklung der Programme über einen Simulator, der es ermöglichen würde, Programme vor dem Laden auf den Roboter effizient zu testen. Daher wurde mit der Entwicklung einer *Tool Chain* begonnen, die die Softwareentwicklung für EMMA wesentlich verbessert. Basis dieser Softwareumgebung ist DiMAS (*Distributed Multi Agent Simulator*), der in der Gruppe von DI Schwaiger entwickelt wird (***Bild1***, ***Bild 5***). Dieser Simulator erlaubt die Definition einer Welt (Positionierung von Objekten, Lichtquellen, etc.) und einer beliebigen Anzahl von Ro-

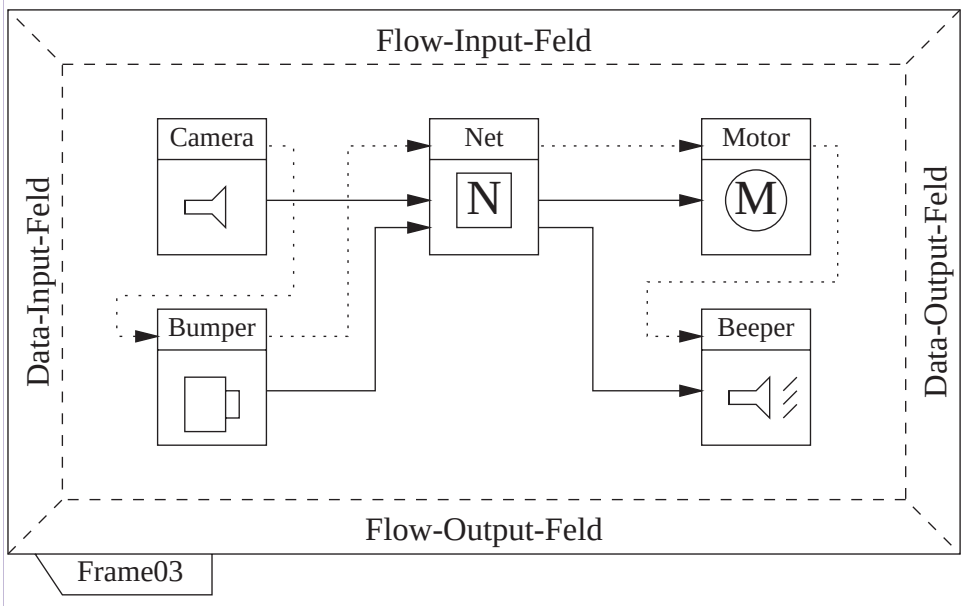


Bild 3: Graphische Programmierung eines Roboters

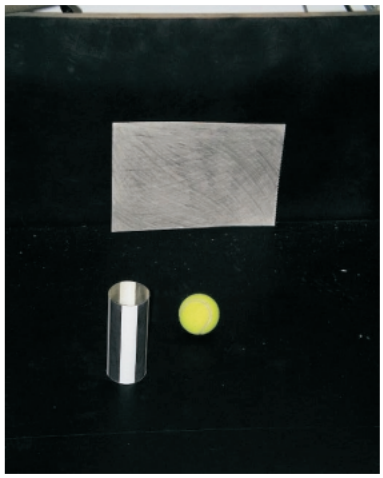


Bild 2: Aufstellung für Fussball, EMMA (Sträflingsgewand), Ball, Tor

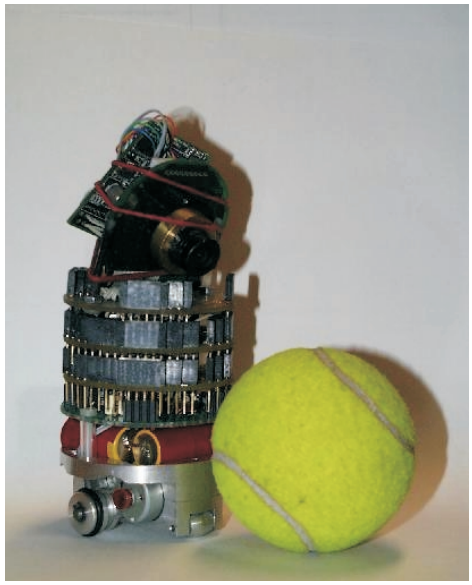


Bild 4: EMMA mit Fussball = Tennisball

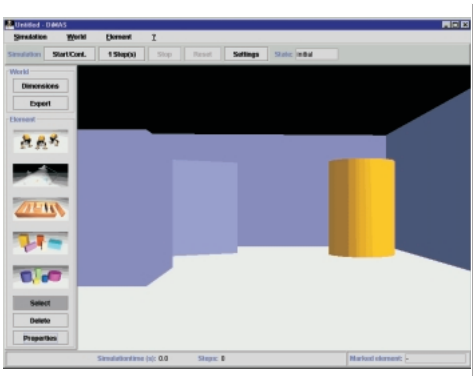


Bild 5: DiMas Screenshot, Welt aus Sicht eines Roboters

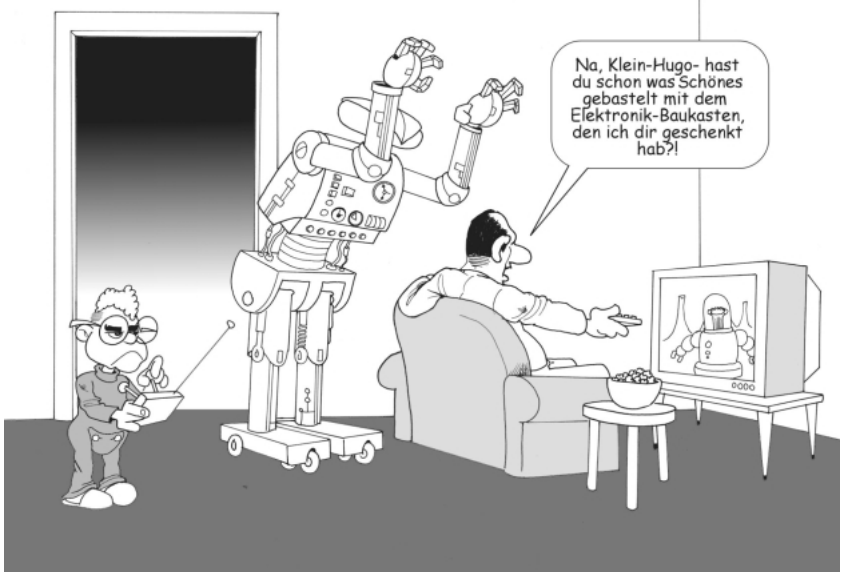
botern, die sich in dieser Welt bewegen. Die Steuerprogramme für die einzelnen Roboter können mit der graphischen Programmierumgebung SOfA (*Salzburger Obsession for Agents*) konstruiert werden (***Bild 3***). Damit kann z. B. ein Neuronales Netz als "Gehirn" des Roboters definiert werden, ohne dass der Konstrukteur eine einzige Programmzeile zu schreiben hat! Das Zusammenspiel dieser Tools erlaubt eine Simulation des Gesamtsystems Welt-Roboter, wobei sogar die Hardware des Roboters konfiguriert werden kann. Dies ermöglicht z. B. die Ermittlung der optimalen Positionierung von Sensoren am Roboter.

Alle Programmpakete werden in Java entwickelt, daher müssen die am Simulator entwickelten Steuerprogramme entweder auf den Befehlssatz des Mikrocontrollers abgebildet werden, oder aber eine *Java Virtual Machine* für den Controller verwendet werden. Letzteres erscheint zur Zeit aus Performancegründen unrentabel, daher wird auch ein Codeumsetzer entwickelt, der den Java Byte Code in den C167-Befehlssatz übersetzt. Für den Ablauf der Programme auf dem Controller wird ein Management-System entwickelt, das einfaches Multi-Tasking, Speicherverwaltung und ein I/O-System über Gerätetreiber unterstützt. Damit ist auch eine wesentlich effizientere Übertragung der Programme vom Entwicklungsrechner auf einen oder mehrere Roboter und beliebige Vernetzung von Systemen möglich. Zur Übertragung ist der Einsatz von Bluetooth-Komponenten geplant.

Mit dieser neuen Kommunikationstechnologie erweitert sich das wissenschaftliche Betätigungsfeld enorm, da dann Phänomene wie *Emergent Behavior* in Gruppen von Robotern studiert werden können. Weltweit sind Forschergruppen zur Zeit aktiv, um solche Robotergruppen so weit zu bringen, dass sie sich an geänderte Umweltbedingungen adaptieren können und möglicherweise selbst neue Strategien zur Bewältigung von bisher unbekannten Problemen entwickeln. Dies mag etwas phantastisch klingen, doch sind heutige Testumgebungen sehr einfach. So versucht man zum Beispiel eine Gruppe von Robotern durch künstliche Evolutionsprozesse so zu "konditionieren", dass sie sich gegenseitig die "lebenswichtige" Information über die Position der Aufladestation mitzuteilen lernen.

In den nächsten Monaten wird in Salzburg also fleißig an der Weiterentwicklung der angesprochenen Softwarekomponenten gearbeitet. Diese sind Teil und Grundlage von Projektanträgen an wissenschaftliche Fonds zur finanziellen Abdeckung weiterer Aktivitäten. Ein zentrales Thema wird dabei die Beschäftigung mit Neurocontrollern erhöhter Plastizität sein, wodurch einzelne Roboter durch direkte Kommunikation mit einem menschlichen Trainer, aber auch von anderen Robotern, lernen können.

Diese Ideen finden auch großes Interesse in einem Arbeitskreis (Prof. Bernroider) von Neurobiologen und Computerwissenschaftlern an der Naturwissenschaftlichen Fakultät der Universität Salzburg. Dort denkt man nämlich schon daran, Roboter mit Wachtelkücken aufzuziehen, und die gegenseitige Beeinflussung des Verhaltens und damit Phänomene wie Intelligenz und Bewusstsein zu studieren...



ROBOSIM V1.0

Peter Ullrich

Dieser einfache Robotersimulator ist das Ergebnis der ersten Anstrengungen, einen einfachen Simulator für einen später anzusteuernden Roboterarm zu erhalten, dessen Bewegungen am Bildschirm in Echtzeit vom Roboterarm ausgeführt werden sollen. Um die Position des Armes räumlich zu sehen, wurde der Simulator in der vom Fernsehen bekannten Rot/Grün-3D-Technik programmiert.

Zum Betrachten benötigt man eine Rot/Grün-Brille, die leicht selbst hergestellt werden kann: Eine einfache rote und grüne Folie (zum Bücher einpacken) auf einen selbstgebastelten Kartonrahmen oder auf eine eventuell vorhandene optische Brille kleben. Das linke Auge muss rot, das rechte Auge grün gefiltert werden. Der 3D-Effect zeigt sich am besten in einem Abstand von 75cm vor einem 14-Zoll-Monitor.

Es ist durchaus normal, dass der 3D-Effekt erst nach einigen Sekunden auftritt, da sich erst die Augen und das Gehirn an das "andere" Sehen gewöhnen müssen.

Steuerung

Der Roboterarm kann am besten mit einer 3-Knopf-Maus gesteuert werden:

- Mausbewegung links/rechts bewegt den ersten Teil des Armes
- Mausbewegung hinauf/hinunter bewegt den zweiten Teil des Armes
- linke/rechte Maustaste drehen den Arm
- Klicken der rechten Maustaste synchronisiert den Winkel der Hand mit dem
- Winkel des ersten Armabschnittes.
- Ein beliebiger Tastendruck auf der Tastatur beendet die Simulation.

Falls Sie zum Simulator Fragen haben, so schreiben sie mir doch einfach eine

E-Mail: *ullrich@kapsch.net*

Betreff: ROBOSIM

Download der Simulation

http://www.ullrich.at.tt/

->Roboter-Seite