netRob

Endbericht | Call 14 | Projekt ID 4647

Inhalt

1 Einleitung

Das Practical Robotics Institute Austria widmet sich der Förderung der Jugend und der Forschung zu Robotik und verwandten Themen. Robotik ist dabei sowohl Thema als auch Werkzeug, insbesondere um Kindern und Jugendlichen Einblicke in Technik und Forschung bieten zu können. Das Projekt netRob war genau an der Schnittstelle zwischen Forschung und Bildung.

Im OpenSource Projekt netRob wurde ein webbasiertes Robotik-Framework entwickelt, mit dem man Industrieroboter einfach grafisch oder textuell programmieren und in 3D simulieren kann. Programme können dabei einerseits am virtuellen Roboter in der simulierten 3D-Umgebung ausgeführt werden, andererseits über ein Hardwareinterface auch auf großen Industrierobotern.

Aufgrund der COVID19-Krise und daraus resultierenden Schwierigkeiten in der Ressourcenplanung für diverse Projekte bei PRIA verzögerte sich das Projekt, weshalb eine Projektlaufzeitverlängerung bis Juni 2021 notwendig wurde. Die Reihenfolge zweier Arbeitspakete wurde auf Grund der inhaltlichen Zusammenhänge getauscht, und somit wurde das Arbeitspaket 5 (Hardwareintegration) vor den Arbeitspaketen 3 und 4 abgeschlossen. In der letzten Projektphase lag der Fokus auf dem Finalisieren von Arbeitspaket 4 sowie der Dissemination und Evaluierung aus Arbeitspaket 6.

2 Projektbeschreibung

Im Projekt netRob entwickelten wir ein webbasiertes Framework, mit dem man Industrieroboter einfach grafisch oder textuell programmieren und in 3D simulieren kann. Ziel war, dass Programme einerseits auf echten Industrierobotern ausführbar sein sollten und andererseits auch in der dazugehörigen Web-App einen virtuellen Roboter steuern und dessen virtuelles Umfeld, beispielsweise greifbare Objekte, beeinflussen könnten.

Zielgruppen:

Bildungseinrichtungen wie Schulen oder private Träger können netRob im Rahmen des Unterrichts oder angebotener Kurse nutzen. Hauptnutzer*innen sind Schüler*innen, die durch den Einsatz grafischer Programmiersprachen einen anfängerfreundlichen Zugang zur Programmierung und Industrierobotik bekommen werden. Das Framework kann aber auch von universitären Einrichtungen und privaten Personen für die Einführung in Robotik benutzt werden.

Realisiertes Framework:

Hauptkomponente ist die webbasierte Integrierte Entwicklungsumgebung (IDE), welche als Adaption der von PRIA entwickelten Hedgehog IDE implementiert wurde. Nach der Kombination mehrerer visueller Funktionsblöcke zu einem Programm wird der endgültige Ausführungscode unter Verwendung des für jeden Funktionsblock definierten Quellcodes generiert. Dieser

Ausführungscode wird dann entweder zur Ausführung in der Robotersimulation oder auf der physischen Ausrüstung verwendet.

Obwohl die IDE eine eigenständige Anwendung ist, kann ein Backend-Server verwendet werden, um Projekte zwischen IDEs zu übertragen. Beispielsweise wird in einem Workshop vor der Ausführung eines Programms auf realen Robotern das Programm von den Workshop-Teilnehmer*innen an den Workshop-Host, z.B. die Lehrperson, übermittelt. Nur diese Person kann dann nach einer Überprüfung das Programm auf der tatsächlichen Roboterhardware ausführen.

Die Robotersimulation ist eine eigenständige Komponente getrennt von der IDE und bietet Schnittstellen zu den Benutzerprogrammen über von der IDE bereitgestellte APIs. Die Simulation verwendet Vorwärts- und Inverse-Kinematik für die Berechnung des Werkzeugmittelpunkts bzw. der Achsenwinkel des simulierten Roboters. Darüber hinaus simuliert und zeigt sie den Roboter und seine Umgebung mit einer webbasierten Physik-Engine sowie einer Rendering-Engine.

Die physische Roboterhardware wird über einen Web-Socket über ein Gateway angebunden, das die Roboterbefehle an den eigentlichen Roboter weiterleitet. Dazu wird die Canonical Robot Control Language (CRCL) unter Verwendung einer JSON-basierten Syntax abgeleitet. Diese CRCL-Befehle werden von der SPS-Software des Roboters interpretiert und ausgeführt.

Implementierung:

Zur Lauffähigkeit im Browser war die Programmiersprache auf Javascript beschränkt, daher wurde der Großteil des Software-Stack mit Javascript implementiert, einschließlich des Backend-Servers, der serverseitiges Javascript (Node.JS) verwendet.

Für die grafische Programmieroberfläche wurde Blockly implementiert, das einen erweiterbaren visuellen Code-Editor bietet, um Programmierkonzepte von Schleifen, Variablen, logischen Ausdrücken usw. einfach per Drag-and-Drop mittels Funktionsblöcken zu implementieren. Für unsere Zwecke haben wir zusätzliche Industrierobotik-Bausteine implementiert, die beispielsweise das Handling von Variablen ermöglichen.

Im Wesentlichen haben wir 3 Hauptgruppen von Funktionsblöcken identifiziert: Bewegungsblöcke, Werkzeugblöcke sowie Vektor- und Rotationsfunktionsblöcke. Bewegungsblöcke sollen elementare Bewegungen auf kürzesten Wegen oder zusammen mit Linien/Kreisen für absolute Koordinaten inklusive Rotation und Werkzeugverschiebungen relativ zum aktuellen Tool Center Point (TCP) ermöglichen. Werkzeugblöcke ermöglichen das Zusammenspiel mit einem oder mehreren unterschiedlichen Werkzeugen wie Parallel- oder Vakuumgreifern. Da die unterschiedlichen Bewegungen stark auf Vektoren und Vektoroperationen basieren, wurde hierfür ein separater Datentypblock für Vektoren und Drehungen implementiert. Alle Blöcke sind im Hinblick auf eine intuitive Benutzerfreundlichkeit definiert, um beispielsweise komplexe homogene 4x4-Matrixmultiplikationen durch einfaches

Kombinieren von zwei Rotationen ohne fortgeschrittenes algebraisches Verständnis einfach zu ermöglichen.

Babylon.js wurde als In-Browser-Rendering-Engine basierend auf WebGL verwendet. Es wurde mit der Ammo.js-Engine für physikalisch korrekte Physiksimulation und Kollisionserkennung verbunden. Die Inverse Kinematik wurde mit IKFast implementiert, das den C++-Code für die Inverse Kinematik Berechnung für eine gegebene Roboterskelettgeometrie generiert. Durch die Verwendung von Emscripten, einem WebAssembly-Compiler, werden der generierte C++-Code und seine mathematischen Abhängigkeiten für die Browsernutzung in WebAssembly portiert.

Fazit:

Mit dem Projekt netRob konnten wir ein Framework realisieren mit dem Ziel, die Programmierung von Industrierobotern zugänglicher zu machen. Das Framework ist für Benutzer*innen mit geringen oder keinen Programmierkenntnissen zugänglich und verwendbar. Es ermöglicht die Programmierung und Simulation verschiedener Industrieroboter in einem Browser ohne jegliche Installation. Die grafische Programmierung ermöglicht die einfache visuelle Verbindung abstrakter Programmierkonzepte mit sofortiger Wirkung im dreidimensionalen Raum.

3 Verlauf der Arbeitspakete

3.1 Arbeitspaket 1 - Projektmanagement und Dokumentation

AP1 umfasste das gesamte Projektmanagement aber auch das Verfassen der Blogbeiträge.

Hinsichtlich des Projektmanagements wurden zumeist monatliche Meetings zum Status des Projekts abgehalten. Zusätzlich gab es seit September 2020 wöchentliche Kurzmeetings des nunmehrigen Developer-Teams. Des Weiteren wurden 7 Blogbeiträge veröffentlicht.

Wie bereits in der Einleitung beschrieben, hat sich das Projekt aufgrund der COVID19-Krise verzögert. Mit der angesuchten Projektverlängerung bis Ende Juni 2021 konnten die geplanten Entwicklungen dann jedoch umgesetzt werden, denn Timon Höbert wurde ab September 2020 von der neuen Mitarbeiterin Sarah Breit bei der Software-Entwicklung unterstützt.

3.2 Arbeitspaket 2 - Planung & Spezifikation

Das Ziel von AP2 war die Definition der zu entwickelnden Software. Dazu wurden aus Sicht der zukünftigen Benutzergruppen User-Stories beschrieben und darauf aufbauend Design-Mockups entworfen. Eine Evaluierung bestehender Frameworks mit Fokus auf webbasierten 3D-Rendering-Engines, Physik-Engines und Inverser Kinematik wurde erstellt. Außerdem wurden die Schnittstellen zwischen Server und Clientanwendung sowie die internen Schnittstellen innerhalb der Entwicklungsumgebung mithilfe eines Plugin-Ansatzes spezifiziert. Bei der Spezifikation des

Hardwareprotokolls, wurde außerdem ersichtlich, dass die 3D-Simulation der Hardware und die konkrete Hardwareanbindung inhaltlich sehr überlappend sind, weshalb eine Verschiebung des Arbeitspaketes bezüglich Hardwareanbindung vor der Implementierung der Entwicklungsumgebung naheliegend war.

3.3 Arbeitspaket 3 - 3D Simulation

Das Arbeitspaket betreffend der 3D-Simulation zielte auf die Implementierung einer dynamischen 3D-Umgebung im Webbrowser mittels WebGL ab, inklusive korrekter Kinematik der simulierten Roboter und physikalisch korrekte Kollisionserkennung. Ein 6-Achsen-Gelenkarmroboter sowie ein kartesischer Roboter mit 3-Achsen wurde für die WebGL-Rendering-Engine portiert. Diese werden automatisiert eingelesen und können bei ihren Gelenken kinematisch korrekt bewegt werden. Dazu wurde, wie im zugehörigen Blogpost beschrieben, erfolgreich ein State-of-the-Art Inverse-Kinematik Framework von C++ in WebAssembly zur Verwendung im Browser transpiliert.

Nachdem sich die mathematischen Konventionen der integrierten Frameworks bezüglich Koordinatensystem und mathematischer Primitive (Matrizen, Quaternions, Euler-Winkel) deutlich unterscheiden, musste in diesem Schritt mehr Aufwand für Konvertierung zwischen den verschiedenen Konventionen getätigt werden als ursprünglich angenommen.

3.4 Arbeitspaket 4 - Programmierumgebung

Das Ziel von Arbeitspaket 4 war die Umsetzung einer einfachen Umgebung, um die dargestellten Industrieroboter grafisch und textuell programmieren zu können. Wie schon erwähnt, wurde dieses Arbeitspaket nachgereiht, um die Hardware-Integration vorzuziehen. Entwickelt wurde zuerst eine Spezifikation der notwendigen textuellen APIs bzw. der zugehörigen grafischen Funktionsblöcken und deren Parameter, welche anschließend umgesetzt wurde.

Für den Einsatz im Rahmen eines Robotik-Workshops wurde eine Sharing-Lösung vorgesehen, wodurch Kursteilnehmer ihre Programme an den Workshopleiter zur Validierung und Ausführung am physischen Roboter übersenden können.

Abschließend wurden Schnittstellen zur Interaktion mit der 3D-Simulation integriert um Interaktionen mit der 3D-Simulation zu ermöglichen mittels Upload von eigenen 3D-Objekten sowie deren räumlichen Bewegung.

3.5 Arbeitspaket 5 - Hardware-Integration

Die Hardware-Integration umfasste die Spezifikation eines eigenen Roboterprotokolls sowie eines zugehörigen Hardwareadapters, welcher die Protokollbefehle zur Roboterhardware überträgt. Ein generischer Protokoll-Interpreter wurde implementiert, welcher die empfangen

Roboterbefehle parst, in eine Warteschlange übermittelt, ausführt und über den aktuellen Ausführungszustand Rückmeldungen zurückliefert. Die Implementierung der Hardware-Integration war zeitintensiver als geplant, nachdem die Dokumentation der Hardwarehersteller sehr dürftig ist, sowie Testen und Debuggen mit physischer Hardware jedenfalls zeitintensiver ist im Vergleich zur Software.

3.6 Arbeitspaket 6 - Dissemination & Evaluierung

In diesem Arbeitspaket konnten einige Erfolge der nicht-wissenschaftlichen und wissenschaftlichen Dissemination erzielt werden, wie in Abschnitt 7 dieses Berichts ersichtlich ist.

Geplant war für die letzte Projektphase außerdem eine stärkere Einbindung der Zielgruppen SchülerInnen, LehrerInnen und Privatpersonen um Feedback für eine Evaluierung einzuholen. Leider wurde dies durch die anhaltende Pandemie verunmöglicht und es konnten somit keine öffentlichen Testing-Tage oder Workshops durchgeführt werden.

Eine HTL-Diplomarbeit als Subprojekt wurde an der HTL Wien 20 (TGM) in der Abteilung Informationstechnologie/Systemtechnik ausgeschrieben. Dazu wurde ein Lastenheft spezifiziert, aber leider hatte sich im Herbst 2020 keine Schüler*innengruppe für das Projekt entschieden. Mit den definierten Requirements und kombiniert mit einem neuen Forschungsprojekt (Thema: Zerlegung von Produkten für Recycling) wurde die HTL-Diplomarbeit für den Herbst 2021 nochmal adaptiert ausgeschrieben, jedoch hat sich leider auch diesmal keine Schüler*innengruppe für das Projekt entschieden.

4 Umsetzung Förderauflagen

Keine.

5 Liste Projektendergebnisse

1	*Zwischenbericht*	*CC-BY-SA*	*https://www.netidee.at/netrob*
2	*Endbericht*	*CC-BY-SA*	*https://www.netidee.at/netrob*
3	*EntwicklerInnen-Dokumentation*	*CC-BY-SA*	*https://www.netidee.at/netrob*
4	*AnwenderInnen-Dokumentation*	*CC-BY-SA*	*https://www.netidee.at/netrob*

5	*Zusammenfassung*	*CC-BY-SA*	*https://www.netidee.at/netrob*
6	*Dokumentation Externkommunikation*	*CC-BY-SA*	*https://www.netidee.at/netrob*
7	*netRob Source Code*	*AGPL v3.0*	*https://github.com/PRIArobotics/netRob*

6 Verwertung der Projektergebnisse in der Praxis

PRIA führt regelmäßig Workshops mit Schüler*innen in Schulen bzw. im eigenen Labor in den Ferien durch. Hierfür wird in Zukunft auch die im Rahmen von netRob entwickelte Software eingesetzt, da es diese ermöglicht, ohne große Vorkenntnisse einen Industrieroboter zu programmieren und das erstellte Programm in der Simulation zu testen.

7 Öffentlichkeitsarbeit/ Vernetzung

Wir haben eine Projektseite auf der netidee-Website (https://www.netidee.at/netrob) angelegt, regelmäßig in Form von Blog-Posts über die Projektfortschritte berichtet und das Gesamtprojekt auf GitHub (https://github.com/PRIArobotics/netRob) zur Verfügung gestellt.

Des Weiteren wurden folgende nicht-wissenschaftlichen Publikationen veröffentlicht:

- Teilnahme bei der #Youtube Challenge der European Researchers Night 2020: https://www.youtube.com/watch?v=HdsrzKYj5oc
- Beitrag im DigitalMondayBlog der DigitalCity.Wien: https://www.digitalcity.wien/netrob-industrieroboter-programmieren-ganz-einfach/
- Beitrag im OCG-Journal, Ausgabe 01-02, 2021, Jahrgang 46, Seiten 34-35: https://www.ocg.at/sites/ocg.at/files/medien/pdfs/OCGJournal2021-01-02.pdf
- Beitrag im Newsletter #2/2021 von Österreich forscht, 22. Juni 2021: https://mailchi.mp/5ab984f37f18/neues-von-sterreich-forscht-8077853

Außerdem wurden folgende wissenschaftlichen Publikationen veröffentlicht:

- T. Hoebert, C. Koza, S. Breit, W. Lepuschitz, M. Merdan: “A Web-based Simulation and Programming Environment of Industrial Robots for Education”; Proceedings of the Austrian Robotics Workshop, Seiten 83-84, 2021: https://www.roboticsworkshop.at/pluginAppObj/pluginAppObj_32/download.php?action=download

- T. Hoebert, W. Lepuschitz, M. Vincze, M. Merdan: “Knowledge-driven Framework for Industrial Robotic Systems”; Journal of Intelligent Manufacturing, akzeptiert – noch kein Link verfügbar

8 Eigene Projektwebsite

Keine.

9 Geplante Aktivitäten nach netidee-Projektende

Derzeit verwenden wir das in netRob entwickelte Framework und erweitern es um eine Virtual Reality Komponente im Rahmen unseres EU-Projekts VR4HRC (Virtual Reality for Human-Robot-Collaboration). Damit ist es möglich, mittels VR-Brille direkt in die Simulation einzutauchen und beispielsweise die für den Roboter erstellte Pfadplanung genauer zu prüfen und mögliche Kollisionen zu identifizieren.

Des Weiteren sollen die Projektergebnisse auch in das erst kürzlich gestartete FFG-Projekt SmartDis (Smart Disassembly with a Knowledge-based Automation System) einfließen, das sich mit dem Einsatz von Robotern für die Demontage von Produkten für Recycling-Zwecke beschäftigt.

10 Anregungen für Weiterentwicklungen durch Dritte

Die derzeitige Implementierung der Roboterbewegungen umfasst derzeit die häufigsten beiden Roboterbewegungsformen mittels Punkt-Zu-Punkt Bewegungen sowie entlang von Linien. Diese könnten noch um Bewegungen entlang von Kreisen (Zirkularbewegungen) sowie entlang von Splines, erweitert werden.

Generell kann die Effizienz der Animationen noch verbessert werden. Hierbei könnte durch effizienteres Zwischenspeichern der Datenstrukturen der Animationen performantere Animationen mit höheren Frameraten erzielt werden.