Ein modulares Framework für adaptive Bewegungssteuerungen für mobile Roboter zur Ausführung ortsbezogener Anwendungen

Pavel Rabov, Daniel Graff

Fachgebiet Kommunikations- und Betriebssysteme
Technische Universität Berlin
Einsteinufer 17 / EN 6
10587 Berlin
rabocinj@mailbox.tu-berlin.de, daniel.graff@tu-berlin.de

Einhergehend mit Moore's Law werden Fertigungsstrukturen kontinuierlich kleiner, woraus ein klarer Trend hin zu mobilen elektronischen Geräten resultiert, die zusätzlich hochgradig miteinander vernetzt sind. Für die steigende Anzahl heterogener mobiler Geräte wurde in [GRW13] ein Ansatz vorgestellt, ein verteiltes Laufzeitsystem zu konzipieren, welches als Ausführungsplattform für verteilte, kontextbewußte Anwendungen dient. Erfordert das Ausführen einer Anwendung Kontextparameter wie Ort (geographische Koordinate) und Zeit, so wird die Anwendung vom Scheduler des Laufzeitsystems in Raum und Zeit eingeplant und es wird ein kollisionsfreier Pfad für einen Roboter zum Zielort berechnet. In dieser Arbeit werden ein modulares Framework für Bewegungssteuerungen, welches das einfache Umschalten zwischen verschiedenen Bewegungssteuerungen kontextabhängig ermöglicht und ein adaptiver Regelalgorithmus vorgestellt, der eine Raum-Zeit-Trajektorie präzise abfährt. Der Algorithmus wurde für einen mobilen Roboter mit 2 separat ansteuerbaren Elektromotoren entwickelt. Da es sich um keine Schrittmotoren handelt und die Motoren in Abhängigkeit von der Wärmeentwicklung unterschiedliche Verhalten aufweisen, ist ein exaktes Abfahren einer Trajektorie nur schwer möglich.

Abbildung 1 zeigt die Architektur des Frameworks für die Bewegungssteuerungen. Falls zur Ausführung einer Aufgabe das Bewegen eines Roboters notwendig ist, so berechnet der Scheduler die Raum-Zeit-Trajektorie und gibt diese an den *NodeMgr* weiter, der Steuereinheit des lokalen Roboters, der diese an die *MotionCtrl*-Komponente weiterreicht, welche aus dem *Dispatcher* und den *GoTo*-Modulen besteht, die die einzelnen Bewegungssteuerungen implementieren. In Abhängigkeit von topologischen Eigenschaften kann der Scheduler bei der Berechnung der Trajektorie an einzelne Streckensegmente konkrete Bewegungssteuerungen vorgeben, z.B. geradliniges Abfahren einer Strecke oder Abfahren von Kurven, die als Spline vorliegen. Die Aufgabe des Dispatchers ist es, die Raum-Zeit-Trajektorie zu traversieren und zum richtigen Zeitpunkt das entsprechende GoTo-Modul aufzurufen und die abzufahrenden Streckeninformationen zu übermitteln. Status-Informationen werden direkt vom GoTo-Modul an den Dispatcher zurückgeschickt. Falls eine Formation mit mehreren Robotern gefahren werden soll, kann ein lo-

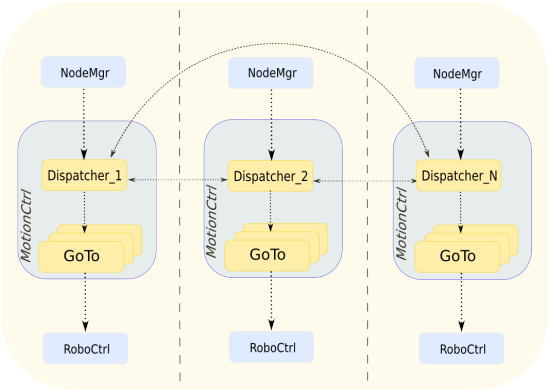


Abbildung 1: Architektur des Frameworks

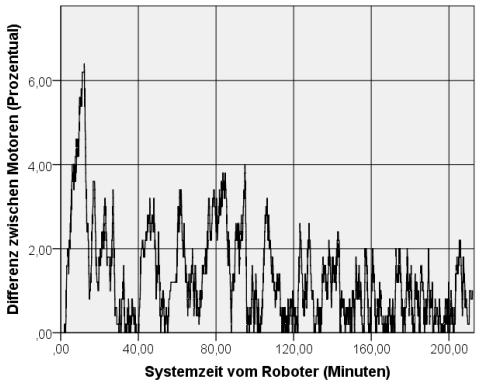
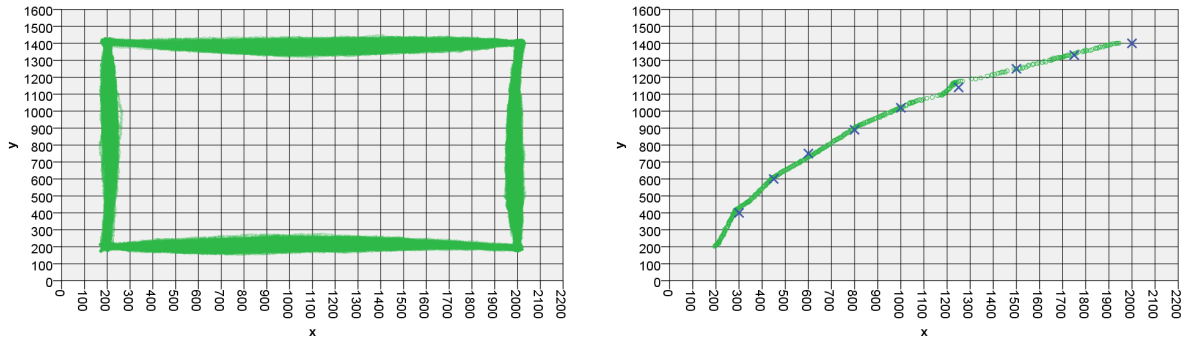


Abbildung 2: Anpassung der Motorkoeffizienten



(a) 200-faches Abfahren eines Rechtecks

(b) Abfahren einer Trajektorie

Abbildung 3: Evaluation der Bewegungssteuerung (visualisiert sind abgefahrene Pfade)

kaler Dispatcher mit entfernten Dispatchern kommunizieren. Die GoTo-Module leiten Fahrbefehle bestehend aus der Motor-Id und der Drehzahl weiter an *RoboCtrl*, welches auf die Hardware zugreift.

Aktuell sind zwei GoTo-Module implementiert: Abfahren eines linearen Streckenabschnitts sowie einer Kurve. Im Folgenden wird die erste Variante, die auf dem Ansatz der Linienverfolgung basiert [AA02], beschrieben. Der Algorithmus ermöglicht das präzise Abfahren einer Trajektorie, die durch lineare Streckensegmente zwischen Punktpaaren gegeben ist und eignet sich für enge Gassen oder Labyrinthe. Drehoperationen finden daher auf der Stelle statt. Der Algorithmus unterscheidet zwischen zwei Zuständen: Ausrichtung sowie Fahren zum nächsten Zielpunkt. Im ersten Zustand dreht der Roboter auf der Stelle durch gegenläufiges Ansteuern der Motoren bis die Ausrichtung zum nächsten Zielpunkt erreicht ist. Der Algorithmus wird mit einem Akzeptanz-Schwellwert für die Ausrichtung zum Ziel konfiguriert. Je kleiner dieser Wert ist, desto öfter wechselt der Roboter in den Korrekturmodus und richtet sich neu aus, um ein präziseres Abfahren der Trajektorie zu ermöglichen, welches mit höheren Zeit- und Energiekosten verbunden ist. Sobald die Ausrichtung abzüglich der Toleranzgrenze in Richtung des nächsten Zielpunkts zeigt, wechselt der Algorithmus in den Zustand „Fahren“. In Abhängigkeit von der zu fahrenden Geschwindigkeit gibt der Algorithmus für beide Motoren die Drehzahl vor. Durch die oben beschriebenen Störeinflüsse reicht es nicht aus, die Motoren einmalig zu kalibrieren. Vielmehr muss dieses kontinuierlich während der Fahrt erfolgen. Daher gleicht der Algorithmus kontinuierlich die vom Ortungssystem ermittelten Koordinaten mit den Soll-Werten ab und berechnet bei einer bestimmten vorliegenden Differenz einen neuen Korrekturwert für die Motoren und speichert diese in den Motorkoeffizienten ab. Abbildung 2 zeigt, wie oft auf einer Test-Fahrt die Motorkoeffizienten aktualisiert werden mussten, damit bei angeforderter synchroner Drehgeschwindigkeit der beiden Motoren tatsächlich eine geradlinige Fahrt vorlag.

Zur Evaluation des Ansatzes präsentieren wir 2 Ergebnisse, die auf einem Spielfeld der Größe 2200 mm $\times$ 1800 mm mit einem Akzeptanz-Schwellwert von 9 Grad durchgeführt wurden. Das verwendete Ortungssystem weist eine mittlere Ungenauigkeit der Ausrichtung von 2 Grad und von 9 mm bzgl. der Position auf. Abbildung 3(a) visualisiert den entstandenen Pfad, der durch 200-faches Abfahren eines Rechtecks entstanden ist. Die maximale Abweichung von der vorgegebenen Spur beträgt in diesem Test 78 mm bei einem Durchmesser des Roboters von 160 mm. Abbildung 3(b) zeigt das einmalige Abfahren einer Trajektorie, die durch mehrere Punkte gegeben ist. Hier ergab sich eine maximale Abweichung von der Spur von 33 mm. Präziseres Abfahren der Trajektorie kann durch die Verkleinerung des Akzeptanz-Schwellwerts oder durch eine höhere Punktdichte erreicht werden.

Literatur

[AA02] Jens Altenburg und Uwe Altenburg. *Mobile Roboter : vom einfachen Experiment zur Künstlichen Intelligenz*. München u.a. : Hanser, 3., überarb. aufl.. Auflage, 2002. 2

[GRW13] Daniel Graff, Jan Richling und Matthias Werner. Programming and Managing the Swarm – An Operating System for an Emerging System of Mobile Devices. In Kai Lin, Heng Qi, Keqiu Li, Ivan Stojmenovic, Albert Zomaya, Hongyi Wu, Song Guo und Symeon Papavassiliou, Hrsg., *9th IEEE International Conference on Mobile Ad-hoc and Sensor Networks (MSN 2013)*, Seiten 9–16. IEEE Computer Society, December 2013. 1