RECONFIGURATION FOR MODULAR ROBOTS USING KINODYNAMIC MOTION PLANNING

Ioan A. Şucan
Department of Computer Science
Rice University
Houston, Texas
Email: isucan@rice.edu

Jonathan F. Kruse
Mark Yim
Dept. of Mech. Eng. & Applied Mechanics
University of Pennsylvania
Philadelphia, PA
Email: jonathaf@seas.upenn.edu
yim@grasp.upenn.edu

Lydia E. Kavraki
Department of Computer Science
Rice University
Houston, Texas
Email: kavraki@rice.edu

ABSTRACT

This paper presents computational and experimental evidence that it is possible to plan and execute dynamic motions that involve chain reconfiguration for modular reconfigurable robots in the presence of obstacles. At the heart of the approach is the use of a sampling-based motion planner that is tightly integrated with a physics-based dynamic simulator. To evaluate the method, the planner is used to compute motions for a chain robot constructed from CKbot modules to perform a reconfiguration, attaching more modules and continuing a dynamic motion while avoiding obstacles. These motions are then executed on hardware and compared with the ones predicted by the planner.

INTRODUCTION

When modular systems reconfigure themselves, their motions have been largely assumed to be quasi-static. Past work often focuses on the problem of finding attach/detach sequences that transform a robot from a given configuration to a goal configuration.

Chain reconfiguration involves chains of modules which form articulated arms and have been demonstrated in [1] and references therein. The reconfiguration is typically not very robust.

When a chain style modular robot performs a chain reconfiguration, it typically must avoid self-collision while using arms with potentially many modules (high DOF). Casal has shown self-reconfiguration of chain systems using a probabilistic roadmap planner [2]. In that work, the purely kinematic collision-free motions of these arms were planned and verified in a gravity-free, friction-free simulation.

Traditional control-theoretic approaches such as model predictive control (MPC) [3] have examined high dimensional spaces and even obstacle avoidance. However, they cannot be used here since the basic structure of the system (the kinematic relationship of masses) changes each time a reconfiguration occurs. The traditional motion planning search methods however can easily accommodate these changes.

Reif and Slee have examined the kinodynamic motion for a lattice style system [4]. The kinodynamic motions were not restricted to having one module carrying only itself (as with previous lattice system work). This leads to a theoretical $O(\sqrt{n})$ execution time of a reconfiguration, where n is the number of modules. However, there was no implementation of this work, the lattice motions are assumed to have only Cartesian motions (no rotational motions or torques), and the assumed environment had no gravity and no friction.

In this paper, we propose kinodynamic motion planning methods to perform chain style reconfiguration in the presence of gravity, friction and obstacles. The planning method is verified with a simple physical demonstration requiring dynamic motions and one reconfiguration, an attach sequence.

EXPERIMENTAL SETUP

An experiment that validates the method needs to show that pure kinematic planners are not sufficient. A simple demonstration of this would be lifting a weight to a position that is beyond the static torque abilities of the hardware, but within reach, if dynamic motions (such as swinging for momentum) are included. Even for modern kinodynamic planners, the task is extremely

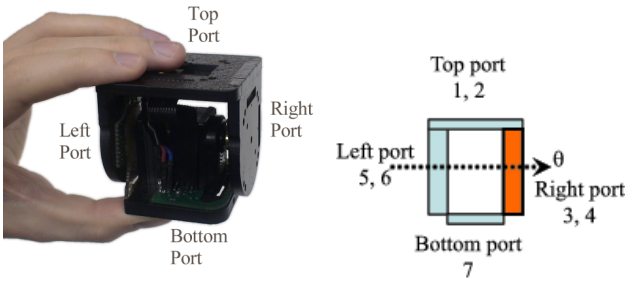

Figure 1. One CKbot module with a schematic representation. The arrow indicates the rotational axis.

difficult and very few planners can tackle this problem. Incorporating the full state dynamics is essential to allow a motion planner to find dynamic motions, which exploit momentum and lead to solutions.

A physical model for the hardware is constructed for the Open Dynamics Engine (ODE) physics simulation library [5]. A motion planner using this library (Path-Directed Subdivision Tree Planner – PDST [6]) is used to compute the motions for performing the task. The motions computed by the planner are then executed on the hardware.

Hardware

CKbot (Connector Kinetic roBot) is a modular reconfigurable robot that is used in this work (Figure 1). The kinematics and connecting topology is typical for many chain style reconfigurable modular robots [1] and is described in detail in [7].

Each module in the system consists of a laser cut plastic (ABS) body with a hobby servo actuator to control one rotational DOF. The hobby servos have enough torque to lift five modules out in a cantilever fashion. Modules attach face to face either manually, with screws or passively, with "magnet faces". Eight neodymium (NIB) permanent magnets (aligned four north and four south) are arranged such that two opposing faces will attract each other at 90° rotations. When the faces of two modules get close enough, the magnetic attraction pulls the modules together. The magnet faces can support the weight of seven modules before falling apart. Reconfiguration and self-assembly using CKbot with magnet faces was demonstrated in [7].

Modeling this module in a physics simulator consists of 1) generating the appropriate geometry, 2) measuring the appropriate properties and 3) creating a model for the torque generated by the module from given commands.

To create a model in ODE that better approximates the performance of the servo, a model of the reflected inertia of the servo is empirically determined. We model the motor inertia and transmission by adding an offset rotational inertia (ideally without adding mass to the system, e.g. two small point masses symmetrically and rigidly attached to the motor axis at a large radial distance away). The reflected inertia is accounted for by increasing the rotational inertia of one module by approximately 22 times.

Motion Planner

Because of the complexity of motion planning, sampling-based planners [8] have become very popular in the last decade. These planners are not complete, but instead provide *probabilistic completeness* [8], which means they will eventually find a solution if one exists. General kinodynamic sampling-based planners are usually tree planners. These planners compute a tree of collision-free motions that obey the robot's dynamic constraints.

With the development of kinodynamic motion planning, it is possible to tackle more complex problems that take into account high order dynamic constraints of the robot, friction, and gravity [6]. Earlier work [9] has shown that for problems that involve chain robots such as the ones used in this paper, advanced kinodynamic planners such as PDST [6] should be used.

For this work, PDST has been augmented to address the issue of reconfiguration. After a reconfiguration step, the topology of the robot changes. This in turn causes the dimension of the state space to change. In the general case, this means that the planner needs to view the robot system as a *hybrid system* [10].

In this work, a motion planner for a hybrid system was implemented by adding support for multiple goals. This allows for the definition of reconfiguration steps and other motions by running the planner multiple times in different state spaces. The only caveat is that proper bookkeeping is required to ensure that the shared states between state spaces are consistent. This choice does not influence the capabilities of the motion planner, but allows for easier testing of the feasibility of reconfiguration through motion planning. For the experiments in this work, the state space corresponding to a chain robot consisting of five modules is used to plan the necessary motions to achieve connection to the stationary modules. The state space is then augmented to include the new modules and the motions necessary for lifting the system vertically up are computed.

EXPERIMENTS

Several tests were run on the planner and then verified with the physical CKbot modules. These tests serve as a proof of concept for this approach.

To argue the use of the PDST planner, other planners [9] were used as a first stage for comparison purposes. These planners attempted to plan the lifting motion for a chain robot with increasing number of modules. For 7 modules, the other planners did not find a solution within one hour, while PDST found a solution within minutes [9]. This path was then verified on the physical hardware.

The specific demonstration we show as an example of kinodynamic motion planning with chain style reconfiguration has a modular chain robot attach its endpoint to another chain of modules and lift the entire system as shown in Figure 2. All joint axes are parallel – the arm manipulates in a plane. One end of the arm is attached to a base and the other end is free. The initial chain – the one that attaches, consists of five modules. The additional chain of modules lies at rest nearby with four modules: two in

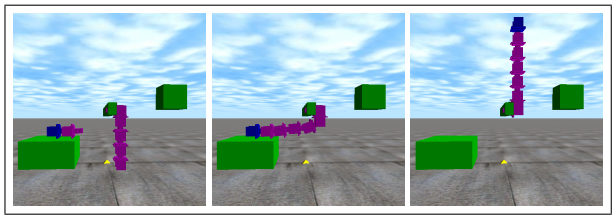

Figure 2. The two phases of the experiment: a chain first attaches additional modules and then lifts up. The last module in the connected chain has three module masses.

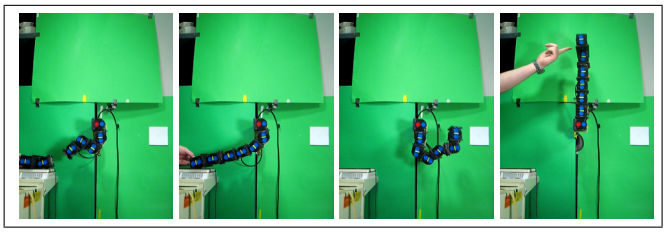

Figure 3. Screen-shots from the hardware execution.

line with the initial arm and one module on each side forming a "T". In effect, the total mass of the system is nine module masses.

The task is for the five module arm to join with the second set and lift them to a vertical position above the base without hitting any obstacles. Thus the arm must first move to a position close enough to dock with the separated modules. The added weight of the arm is significantly beyond the static torque capabilities of the modules to lift in a cantilever fashion.

The planner generates runs as sequence of commands for the hardware. The runs successfully use dynamic motion to reconfigure and invert to a vertical position (Figure 3). However, the success of the runs tested require minor outside influence in two sections. Once the modules are in position for reconfiguration, a slight push is given to the tabled modules to aid in the magnetic face connections. The second outside influence occurs near the very end of the gait, once the dynamic swinging has been completed and the modules enter a near-static regime for the final extension towards the vertical. Additional torque is supplied to the fifth module in the chain to aid in lifting the weighted modules beneath it. The necessity to supply this additional torque most likely stems from slight discrepancies between the `ODE` model of the servo, and the real-world abilities and behavior it possesses.

In order to quantitatively measure the differences between the instruction data generated by `PDST` and the real path execution, a vision system finds the angles between adjacent modules. These results are then compared to the instruction input from `PDST` to find the average error between hardware output and instruction input per module for each instruction given. The average error ranges from 6.1° to 12.8°.

CONCLUSIONS

In this work, we have examined the feasibility of reconfiguration with dynamic motions through motion planning. The basic principle of the method is to use a motion planner coupled with a physics simulator to compute the necessary motions. When reconfiguration occurs, the system changes some of its parameters. Since the planner sees the physics simulator as a black box, as long as the simulator can handle discrete events (reconfigurations), the planner only needs to update the state space it uses.

A first step in experimental validation of this approach was shown by performing one reconfiguration while achieving a dynamic motion avoiding obstacles in the presence of friction and gravity. The model of the physical system was difficult to achieve due to the use of internal position control code on the servos.

In addition, a new set of hardware with a direct drive motor and no proprietary control is being built with the explicit purpose of being more easily modeled. It is expected that with well modeled hardware combined with the further development of hybrid systems planner (integrating the discrete reconfigurations with the continuous motions), the system should yield much more interesting and complex reconfiguration and dynamic motions in cluttered environments.

REFERENCES

[1] Yim, M., Shen, W.-M., Salemi, B., Rus, D., Moll, M., Lipson, H., and Klavins, E., 2007. "Modular self-reconfigurable robot systems: Challenges and opportunities for the future". *IEEE Robotics & Automation Magazine,* **14**(1), Mar., pp. 43–52.

[2] Yim, M., Goldberg, D., and Casal, A., 2000. "Connectivity Planning for Closed-Chain Reconfiguration". *Proc. SPIE, Sensor Fusion and Decentralized Control in Robotic Systems III,* **4196**, pp. 402–412.

[3] Mayne, D., Rawlings, J., Rao, C., and Scokaert, P., 2000. "Constrained model predictive control: Stability and optimality". *Automatica,* **36**(6), June, pp. 789–814.

[4] Reif, J., and Slee, S., 2007. "Optimal kinodynamic motion planning for self-recongurable robots between arbitrary 2d configurations". In Robotics: Science and Systems.

[5] Smith, R., 2008. Open dynamics engine. http://www.ode.org. seen March 12, 2008.

[6] Ladd, A. M., 2006. "Direct motion planning over simulation of rigid body dynamics with contact". PhD thesis, Rice University, Houston, Texas, December.

[7] Yim, M., Shirmohammadi, B., Sastra, J., Park, M., Dugan, M., and Taylor, C., 2007. "Towards robotic self-reassembly after explosion". In International Conference on Intelligent Robots and Systems, pp. 2767–2772.

[8] Choset, H., Lynch, K. M., Hutchinson, S., Kantor, G. A., Burgard, W., Kavraki, L. E., and Thrun, S., 2005. *Principles of Robot Motion: Theory, Algorithms, and Implementations*. MIT Press, June. ISBN 0-262-03327-5.

[9] Şucan, I. A., Kruse, J. F., Yim, M., and Kavraki, L. E., 2008. "Kinodynamic motion planning with hardware demonstrations". In International Conference on Intelligent Robots and Systems.

[10] Chutinan, A., and Krogh, B., 2003. "Computational techniques for hybrid system verification". *IEEE Transactions on Automatic Control,* **48**(1), January, pp. 64–75.