Proceedings of the 2002 IEEE
International Conference on Robotics & Automation
Washington, DC • May 2002

Experiments with Nonholonomic Manipulation

Siddhartha S. Srinivasa[1], Christopher R. Baker[1], Elisha Sacks[2],
Grigoriy B. Reshko[1], Matthew T. Mason[1], Michael A. Erdmann[1]
[1]The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA - 15217
[2]Computer Science Department, Purdue University, West Lafayette, IN - 47907

Abstract

This paper summarizes ongoing work with a mobile manipulator (*Mobipulator*). We describe the system architecture of the latest version of the robot, a hierarchy of robot motion commands (the Mobipulation library) that can be snapped together to generate complicated paths easily, a configuration space planner that plans wheel motions to manipulate paper, and a visual servoing system to monitor and correct errors in robot motion.

1 Introduction

The Mobipulator looks like a small car with four independently controlled wheels, none of them steered. It uses its wheels both for locomotion and for manipulating objects on a desktop. With two wheels on paper and the other two on the desktop, the robot behaves like two connected differential drives, one manipulating paper and the other locomoting the robot and the paper. This mode, which we termed the *dual-diff drive mode* (Fig.1), provides a simple and elegant decoupling of manipulation and locomotion. More importantly, in this mode, the paper's motion is unconstrained - a linear combination of the wheel vector fields spans the space of paper velocities[1]. In favoring manipulation over locomotion in the robot design we compromise on turning which the robot can achieve only by skid steering, a procedure that relies on wheel slip and is unpredictable and inaccurate.

Our most challenging goal is to build a robust system. This involved selecting the right hardware, and creating a language and a user interface that would make programming the robot easy. A higher goal is to try to understand the interconnectedness of locomotion and manipulation. An illustrative example is robot turning. By itself, the robot turns inaccurately. However, in the *dual-diff drive mode*, the two rear wheels behave like a single differential drive and can turn the robot accurately. Hence the robot needs the paper to turn accurately, while the paper needs the robot to move it. Another area of ongoing research is in the control of non-holonomic systems, a pertinent issue for our robot because of the slip between the wheels, the desktop and the paper resulting in deviations from the desired trajectory. Another goal was to test the limits of the system in terms of speed, dexterity, and reliability, and to devise challenging and interesting experiments.

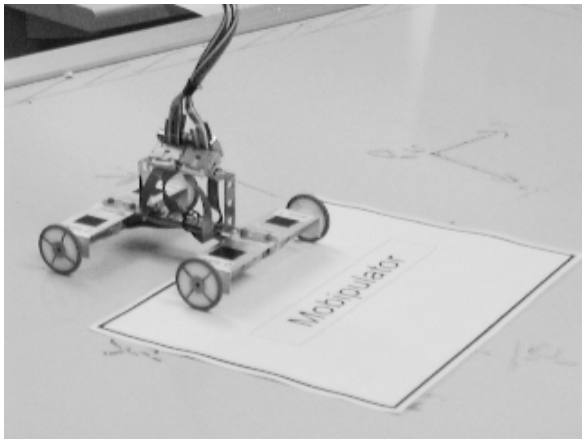
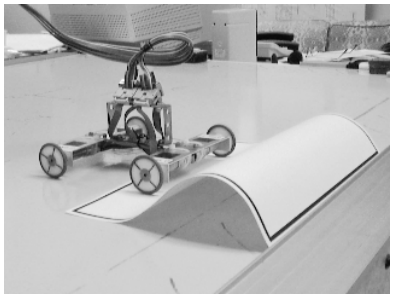
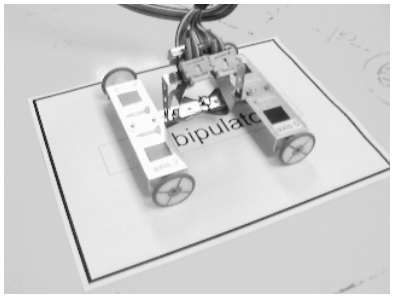


Figure 1: The Mobile Manipulator in *dual diff-drive mode*

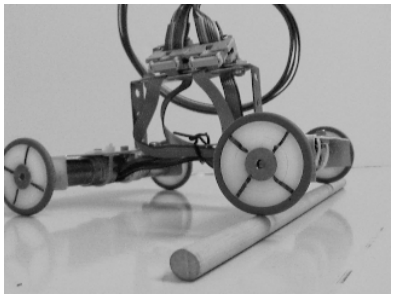
The Mobipulator is a concept car. It is designed to explore different concepts of how a mobile robot can perform manipulation tasks, and in particular, the interconnectedness of locomotion and manipulation. We have succeeded in building a system that is easy to program and accurate in action. Our experiments have not only demonstrated the robot's ability to effectively manipulate desktop objects, but have also provided insight on friction reduction and on using natural compliance for alignment. The uncertainty in skid steering has been reduced by using visual servoing. We have also effectively demonstrated our configuration space planner on the real robot.



(i) : Paper Manipulation



(ii) : Paper Scooting



(iii) : Pencil Rolling

Figure 2: Experiments with the Mobipulator

1.1 Mobipulator Experiments

The Mobipulator's task domain is the desktop. One of the most common objects to clutter a desktop is paper. Hence we decided to first experiment with manipulating paper. One can envision the Mobipulator behaving as a desktop assistant and collecting paper strewn on a desktop into a neat stack. In moving paper we noticed several interesting phenomena. Here are a few :

1. **Paper Aligning** : We use the raised edges of the desktop and the natural compliance of paper to align the paper flush with the table edge. The robot pushes the paper against the table edge causing the paper to bend up (Fig.2(i)). As the robot slowly backs up, the paper unfolds and aligns.

2. **Paper Scooting** : In this mode, all four of the robot's wheels are on the paper. The robot uses an asymmetric vibration to move both itself and the paper (Fig.2(ii)).

3. **Friction Reduction** : The friction between the paper and the desktop can reduced by vibrating the manipulating wheels in the dual-diff drive mode. This results in a smoother, more accurate motion.

4. **Parallel Parking** : While mobile robots use parallel parking to move themselves sideways, the Mobipulator uses parallel parking to move the paper sideways. Since paper is much lighter than a robot, the Mobipulator is able to move the paper sideways rapidly.

5. **Pencil Rolling** : The robot can also move cylinders, if it can get on top of them. The robot uses the edge of the table to mount the pencil in Fig.2(iii). Once on top, the robot is surprisingly stable.

2 Related Work

This section is a brief review of mobile manipulation and its relation to the present work. A more thorough survey can be found in[1].

Several preceding systems have explored the connection between manipulation and locomotion. One of the earliest influential robots, Shakey[2], pushed boxes and other objects. More recently, the task domain of Sojourner in Mars included elements of manipulation.

A more direct approach is to attach a manipulator to a mobile platform. The JPL Cart[3] provides an early example, while Romeo and Juliet[4] provide a current example. These robots have demonstrated effective coordination of wheels and arms in manipulation tasks.

The distributed manipulation work of Donald *et al.*[5] included a set of mobile robots pushing objects, as if each robot were a finger in a multi-fingered grasp. The Platonic Beast[6] had several limbs, each of which could be used for locomotion or manipulation.

Rus *et al.*'s Fiat[7], the paper-lifting robot, is also a desktop mobile manipulator. But unlike the Mobipulator, Fiat, uses an attachment with sticky adhesive tape to immobilize the paper to the robot. Once this is achieved, the robot moves to the goal location and uses another attachment to release the paper.

All of these works illustrate that there is an underlying connection between locomotion and manipulation. In our case, the robot is just one of several movable objects in the task. The function of each actuator is resolved according to the task, be it manipulation, locomotion, or something not clearly classifiable as either.

3 System Architecture

This section describes the hardware and the software architecture of our system.

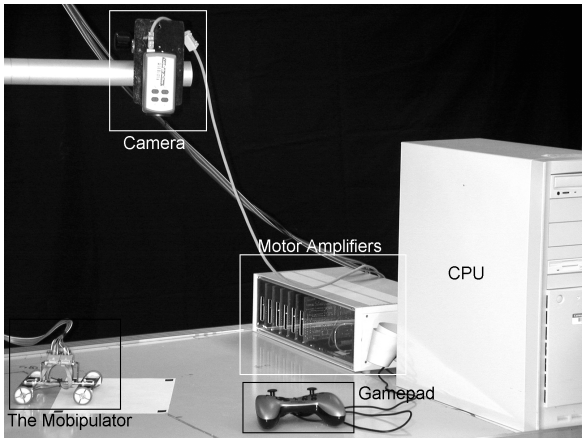


Figure 3: The Mobipulator system overview

3.1 Hardware

The robot chassis is made of aluminium. The wheels are outfitted with rubber O-rings. The motors are controlled using an ISA servo control card from ServoToGo, Inc. The card provides eight independent DAC channels, each of which is connected to a LC-3002A Linear Servo Amplifier from Micro Mo Electronics, Inc. The amplifiers are housed in a custom-made control box with a PCB backplane, which makes it easy to plug and unplug all the connectors. We have also integrated a Gravis Eliminator game pad into our system to provide manual control. A Sony DFW-VL500 digital Firewire camera is mounted vertically above the desktop at a height of about 1.4 meters. The camera can grab color images at a rate of up to 30Hz at a resolution of 640x480 pixels.

3.2 Software

The software system consists of the following four components, running under Windows 2000 on a Pentium II.

3.2.1 The Device Driver: The Kernel-mode device driver has a top half and a bottom half. The bottom half is driven by the interrupt controller on the card (currently set at 1 kHz) to implement a PID motor controller at regular, predictable intervals. Since the driver runs in kernel mode, extended computations in an ISR can cause sluggish system behavior. Thus, all calculations are done with fixed-point values instead of floats and complex functions (such as sine) are done with lookup tables. As a result, any given ISR takes no more than 60 μs to complete on a 400 MHz Pentium II.

3.2.2 The Interface Library: The top half of the device driver interacts with the interface library to provide user programs a means of communicating with the driver. These are largely based on source code supplied by ServoToGo under the GNU Public License. Using the interface library, user programs may enable or disable closed-loop control and set the gains for the PID controller.

We have implemented a trajectory generator that generates velocity profiles given a demand position, velocity, and acceleration for each motor. The generator outputs various functions in real-time such as trapeziods, quadratic curves, sinusoids, and sawtooth waves that may be concatenated to produce more interesting trajectories. The various parameters of the generator can be accessed via the interface library.

The library can also be used to retrieve data from the driver's short-term (1000 interrupt cycles) log and write directly to the DAC to affect some form of continuous control from a user program.

3.2.3 The Mobipulation Library: The third component, dubbed *libmobip*, combines calls to the interface library (for simplicity) with calls directly to the driver (for efficiency) to provide motion commands in physical units (meters, radians, seconds). There are four primary groups of commands.

- *Program control* : These commands to start up/shut down the library, start/stop robot motion, set up robot parameters, flush motion queues, and print error strings.
- *Four-wheel drive commands* : These commands for translating and rotating the entire robot treat the robot as a single differential drive at the center of the robot. These commands, in particular benefit from visual feedback, as described in §4.
- *Dual diff drive commands* : This command set is similar to the four-wheel drive set, except that each command also takes a parameter to indicate which half of the robot will perform the action. The two halves may be controlled either one at a time or simultaneously.
- *Logging commands* : These log the desired and actual trajectories of the robot into a simple file format for display and analysis.

3.2.4 Gamepad Library and GUI: We have also written a small library that allows a user to control the Mobipulator using a gamepad. The gamepad has two joysticks and eight buttons. Each joystick controls one differential drive. The buttons can be mapped to various controls like locking of individual wheels (for manipulation) and the path logger. The gamepad provides is an essential tool for testing new ideas rapidly without any programming.

4 Perception

The motivation for using the camera is the inaccuracy in locomotion and manipulation caused by slip between the robot, the paper and the desktop. We use the camera both for measuring these errors and for correcting them by visual servoing.

4.1 The Tracker

The robot uses color thresholding to to locate the fiducials in each image frame. For ease of tracking, we have added four orange fiducials on the robot chasis and four green fiducials on the corners of the paper. We implemented a simple thresholding scheme in (R,G,B) space. At times when the robot is unable to track a feature, it uses its prior knowledge of the fiducial's geometry to reacquire them. Here is an outline of our algorithm :

1. *Initialization* : Initialize the thresholds manually.

2. *Tracking* : Use the thresholds to find the four largest connected orange and green blobs in the image. Find the centroid of each of the blobs and store them as the points tracked.

3. *Check Rectangle* : Check if the points tracked form two rectangles, one each for robot and paper, within a tolerance. Check for both orthogonality and Euclidian distance.

4. *Fit Rectangle* : If Check Rectangle fails, use the tracked points and the lengths and widths of the rectangles to fit the remaining points. Note that this can be done only if one point in a rectangle is lost.

5. *Store Old Points* : If Fit Rectangle fails, save the last set of tracked points as currently tracked.

6. *Return* : Return the position and orientation of the tracked rectangles.

The algorithm runs at 15Hz at a resolution of 640x480 pixels. The tracker runs in a separate thread from the motion commands and can thus be queried when desired. Note that the camera is first calibrated and turned until its axis is orthogonal to the desktop, prior to tracking.

The tracker has a few drawbacks. At the height at which the camera is mounted(1.4m), each pixel in the image corresponds to 3mm on the desktop. The tracker is also sometimes unable to locate the fiducials - robot fiducials are occluded by the tether and paper fiducials are occluded by the robot. We are still working on techniques (predicting fiducial location, larger fiducials, smaller tether) to reduce this loss. As a reference, 10 out 24 of runs of the path shown in Fig.4 were completely tracked.

4.2 Visual Servoing

We have implemented a simple visual servoing algorithm to ensure that the robot turns accurately. The robot turns by 5 degrees (which is approximately the angular resolution of the tracker) and checks its orientation with the tracker upon completion. The robot continues to execute the turns until it is within 5 degrees of its desired orientation. Although the entire turn is composed of many small turns, the motion is very smooth because the delay between turns is in the order of microseconds. We have also implemented the same algorithm for the dual differential-drive mode for turning the paper or the robot.

Visual servoing is work in progress. While visual skid steering ensures that the final orientation is as desired, it cannot ensure that the robot has turned in place. In reality, the center of the robot usually drifts by as much as 3cm while performing a 90^o turn. Another drawback is that control is discrete, the robot has to stop and query the tracker after every turn. One reason for this is the inability of the tracker to track very fast motions - while the robot is capable of velocities of up to $50cm/sec$, the comparatively low camera frame rate and the ensuing motion blur make tracking very hard.

5 Planning

This section describes a path planning algorithm that uses dual diff-drive mode to navigate a sheet of paper from a start configuration to a goal configuration while avoiding known, static obstacles. We also describe the implementation of the generated plan on the real robot.

Fig.4 shows six snapshots from an example in which the robot drags the paper from configuration 1 (C1) to C6. The plan is generated in a fraction of a second by a lisp program on a Pentium IV running Linux.

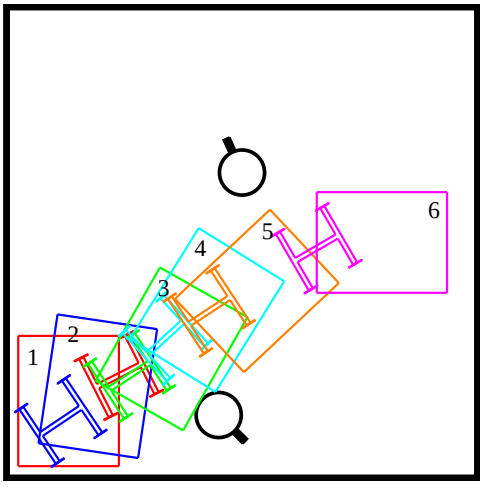


Figure 4: Planner for paper manipulation

5.1 The Algorithm

The Mobipulator path planner is an extension of Sacks's planner for a planar linkage with static and movable obstacles [8]. The inputs to that planner are the link shapes, joint constraints, and initial configurations, the obstacle shapes, and the goal configuration for a designated critical link. The output is a linkage motion path that brings the critical link to the goal configuration while enforcing the constraints.

The planner generates a path in two phases. The first phase searches the critical link/obstacle configuration space for a free-space path to the link goal configuration. The configuration space is represented by an exact contact space partition that is computed by a sweep plane algorithm [9]. The planner performs an A* search based on a heuristic distance function. The search nodes are straight-line paths from the current configuration to the goal and contact patches (subsets of contact space where a specific link feature touches a specific obstacle feature). In our example, the paper is the critical link and the path consists of a line from C1 to C3, a curved contact path to C4, and a free path to C6.

The second phase extends the critical link path to the entire linkage. It constructs a velocity field that drives the critical link along its path and that enforces the joint constraints. It integrates the velocity field until the critical link reaches its goal. If another link hits an obstacle, the planner modifies the velocity field by subtracting the component that is normal to the obstacle. If the path ends without reaching the goal, the planner resumes the A* search and tries the second phase on the next best path.

The Mobipulator is subject to non-holonomic motion constraints, which are not handled by Sacks's planner. Also, the wheels have bounds on angular velocity and acceleration. We are working on a general planner for these types of constraints. In this paper, we extend the planner to handle the special case of dual diff-drive mode.

The first phase plans a path for the paper to the goal configuration as before. The second phase constructs a velocity field for the four wheels that drives the paper along this path. The planner integrates the wheel velocity field until the paper reaches its goal. If dual diff-drive mode is violated, the simulation is halted.The planner then uses a series of heuristics to restore dual diff-drive mode by moving the robot without moving the paper. The simplest heuristic is to move forward or backward with four equal wheel velocities. In our example, the robot drags the paper with its back wheels from C1 until just before C3 when wheel 4 comes off the paper. The robot moves backward to C3 and simulation proceeds.

5.2 The Implementation

The planner output is four wheel angle functions. These functions can contain velocity discontinuities at contact changes, such as C3 and C4 in the example, and can violate the Mobipulator velocity and acceleration bounds. We obtain a valid plan by fitting piecewise Hermite splines to the angle functions, setting the tangents to zero at velocity discontinuities, and time re-scaling.

When implemented on the robot without visual servoing, the sections involving the robot driving straight, or manipulating the paper worked very well. Sections of the plan involving moving the robot in the dual differential-drive mode were, however, inaccurate. This is because, in this mode, the manipulating wheels exert a torque on the locomoting wheels, resulting in the latter not tracking their reference points accurately. The system is non-holonomic and hence much more sensitive to errors than a holonomic system. As a result, the plan is not executed perfectly. On average, there was a 3cm error in position and 10^o error in orientation of the paper when the plan in Fig.4 was executed. We are currently working on continuous control using the camera to correct these errors.

6 Discussion

The duality of manipulation and locomotion permeates robotics. Some examples border on the silly yet can be illuminating. For instance, one can view legged locomotion as dual to fingered manipulation: the legs manipulate the ground, the fingers walk over the object. Other examples are more intriguing. For instance, whole arm manipulation[10] evokes the dual of snaked gaits in locomotion space.

As often is the case with duality, there is as much to be learned from the differences between dual points as from their commonalities. The commonality between walking and grasping lies in the static force diagrams. To grasp an object, the fingers must achieve force-closure. To stand stably, the legs must enclose the robot's center of gravity. The difference between walking and grasping lies in the dynamics, and thus in the power diagrams. The legged robot moves itself relative to the world, the fingers move an object relative to the hand. This dynamic difference served the Mobipulator well in the parallel parking example. The low mass of the paper allowed the Mobipulator to convert a slow locomotion operation into a fast manipulation action.

One of the lessons of robotics over the past 25 years has been the tradeoff between special purpose and general purpose robots. For any given task, it is likely that someone will invent a special purpose robot better at performing the task than any general purpose robot. However, in a pinch, a general purpose robot can emulate a special purpose robot, perhaps with a slight cost in speed and accuracy. That emulation ability is the foundation of most initial research inquiries.

By combining locomotion and manipulation into a single system, the line of research proposed in this paper takes general purpose to an extreme. It is therefore natural to ask whether anything really is to be gained by doing so. For instance, rather than build a robot that uses its wheels both to locomote itself over a table and to manipulate paper on the table, perhaps it would be better to use specialized subsystems, e.g., a differential drive to locomote and a sticky tape actuator to impale the paper. There is no clear answer to that question; there will always a tradeoff. However, it is clear that there are some advantages to using a general purpose Mobipulator. The biggest advantage is the use of multi-functional actuators - the robot does not have to reorient itself to get the right actuator in the right place.

7 Future Work

The most serious impediments to the Mobipulator are friction between the paper and the desktop, and wheel slip. Thus closed loop control of some sort is essential. In the future,we hope to weave continuous visual servoing into entire plans, not just discrete motions. We are also exploring alternate mechanical designs for a rough-terrain Mobipulator, as many desktops are stacked with books and other items that make them non-planar.

References

[1] Matthew Mason, Dinesh Pai, Daniela Rus, Lee Taylor, and Michael Erdmann, "A mobile manipulator," in *IEEE Int. Conf. on Robotics and Automation*, 1999.

[2] N.J. Nilsson, "Shakey the robot," Technical Report 223, SRI International, 1984.

[3] A.M. Thompson, "The navigation system of the jpl robot," in *Proc. 5th Int. Joint Conf. on Artificial Intelligence*, 1977.

[4] O. Khatib, K. Yokoi, K. Chang, D. Ruspini, R. Holmberg, A. Casal, and A. Baader, "Force strategies for cooperative tasks in multiple mobile manipulation systems," in *Robotics Research : The Fifth Int. Symposium*, 1996.

[5] B. Donald, J. Jennings, and D. Rus, "Analyzing teans of cooperating mobile robots," in *IEEE Int. Conf. on Robotics and Automation*, 1994.

[6] D.K. Pai, R.A. Barman, and S.K. Ralph, "Platonic beasts : a new family of multilimbed robots," in *IEEE Int. Conf. on Robotics and Automation*, 1994.

[7] M.T. Mason, D.K. Pai, D. Rus, J. Howell, L.R. Taylor, and M.A. Erdmann, "Experiments with desktop mobile manipulators," in *Int. Symp. on Experimental Robotics*, 1999.

[8] Elisha Sacks, "Configuration space path planning for planar mechanical systems," in *IEEE Int. Conf. on Robotics and Automation*, 2001.

[9] Elisha Sacks, "Practical sliced configuration spaces for curved planar pairs," *Int. Journal of Robotics Research*, vol. 18, pp. 59–63, 1998.

[10] J. K. Salisbury, "Whole arm manipulation," in *Proc. of the 4th Int. Symp. on Robotics Research*, 1987.