Distributed Manipulation of Multiple Objects using Ropes

Bruce Donald*, Larry Gariepy, and Daniela Rus†
Dartmouth Computer Science Department‡

Abstract

This paper describes a system in which multiple robots cooperate to move multiple objects such as groups of boxes using a constrained prehensile manipulation mode, by wrapping ropes around them. The system consists of three manipulation skills: tieing ropes around objects, affecting rotations using a flossing manipulation gait, and affecting translations using a ratcheting manipulation gait. We present algorithms for these operations, a numerical analysis for the motion of groups of boxes, and experimental results.

1 Introduction

We are working on distributed manipulation problems, in which two or more robots apply spatially-distributed forces to rearrange the objects in their work space; the computation is distributed among the robots so as to qualify as a distributed computation. In our previous work [12, 2, 3] we presented several distributed manipulation protocols for non-prehensile pushing and reorienting of large objects, where the robots have no models of these objects, there is no explicit synchronization between their actions, and there is no explicit communication between the robots. In our paper [4] we presented the concept of distributed manipulation with ropes and demonstrated it by describing a system in which a single object can be repositioned with this technique. In this paper we describe a new class of distributed manipulation algorithms in which multiple mobile robots manipulate multiple objects at the same time, using ropes. Distributed manipulation with ropes allows us to explore the passive kinematics of ropes, when a rope is attached to the waistline of the robots. There are two interesting properties worth noting: (1) ropes cannot be actuated independently; and (2) when a rope is attached to a robot, it can *affect* the motion of the robot and also it can *effect* manipulation when the robot moves. Thus, our approach is an exploration of underactuated distributed manipulation systems based on a simple biomimetic design inspired by flagella and cilia.

Manipulating objects with ropes has several advantages over the direct manipulation of an object using a team of robots [12, 2, 3]. First, the method allows for the parallel and synchronized manipulation of multiple objects. (That is, several objects can be repositioned together). Second, wrapping a rope around an object permits *constrained prehensile manipulation* with non-prehensile robots. The rope can conform to any object geometry on-line, in that the same wrapping protocol works for all geometries and there is no need to specify a model in advance. Finally, the rope can be viewed as a tool that allows the robot system to exert torques that are larger than the torque limit of each individual robot. Our notion of constrained prehensile manipulation is different than the classical definition of prehensile manipulation, which usually denotes a force closure grasp. In *constrained prehensile manipulation,* perturbations along any differential direction can be resisted (in our case, due to the taut rope) but the object can be moved only along certain lower dimensional degrees of freedom. Thus, constrained prehensile manipulation systems are *non-holonomic.*

Our results on distributed manipulation have been inspired by previous work in motion planning [7], manipulation [5, 6, 14], cooperative robotics[9, 10, 15, 1], and distributed manipulation [12, 2, 4]. Our previous work on cooperative manipulation treats pushing and reorienting a large object from a minimalist viewpoint [2, 3]. We present and analyze four pushing and four reorientation protocols that require different amount of communication and sensing. Our previous mobile sensing work [13] describes our box manipulation system using "mover" and "scout" robots. Finally, our work on moving a single object using ropes is described in [4].

*Bruce Donald is grateful to the NSF for support provided through the following grants: NSF IIS-9906790, NSF EIA-9901407, NSF 9802068, NSF CDA-9726389, NSF EIA-9818299, NSF CISE/CDA-9805548, NSF IRI-9896020, and NSF IRI-9530785.

†Daniela Rus is grateful to the NSF for support provided through the NSF CAREER award IRI-9624286, NSF award IRI-9714332, NSF award EIA-9901589, and NSF award IIS-98-18299.

‡{brd,gariepy,rus}@cs.dartmouth.edu

2 A Distributed Manipulation System

Consider the task whose goal is to change the pose of a a group of boxes. A team of robots can use a sequential non-prehensile strategy such as those described in [12, 2, 3] to accomplish this task: each object will be located individually. A different approach is to use a parallel constrained prehensile strategy: the objects will be repositioned simultaneously. Because the objects we wish to move are large, it is impractical to envision robots equipped with big enough grippers to do such manipulation tasks. Instead, we focus on strategies that augment the basic architecture of mobile robots with simple tools. In this section we describe an approach where the robots can achieve a grasp on collections of objects by wrapping the objects with a rope (independent of the overall geometry). We then explore cooperative strategies for manipulating a collection of objects wrapped with a rope.

2.1 Experimental Setup

Our distributed manipulation team consists of three mobile robots: two RWI B14 robots equipped with sonar, discrete contact sensors, IR, and radio modems; and one RWI Pioneer robot equipped with sonar, a radio modem, and a parallel-jaw gripper. The B14 robots are named Bonnie and Clyde and the RWI Pioneer robot is named Ben. We augmented the basic hardware of these robots by using a long rope (over 5 meters) which is tied at one end to Bonnie and at the other end to Clyde. We incorporated a tool in Ben's gripper that extends the effective height of the gripper from 2 cm. to 15 cm.

Each experiment begins with the two B14 robots, Bonnie and Clyde situated on opposite sides of a group of boxes, with a rope strung between them. The helper robot, Ben, is situated next to the side of the box closest to the rope.

2.2 The Binding Algorithm

The binding algorithm begins with Clyde performing a calibration routine. Let F_p be the force on the passive robot (Bonnie) and F_a the force on the active robot (Clyde). In order for the initial rope tension not to affect the calibration step, Clyde first turns 90 degrees to face toward Bonnie, and moves forward to release all tension in the rope ($F_a = 0$). He then measures the amount of rotational power necessary to achieve rotational compliance.[1] Ready to begin, he now returns to his original position and orientation.

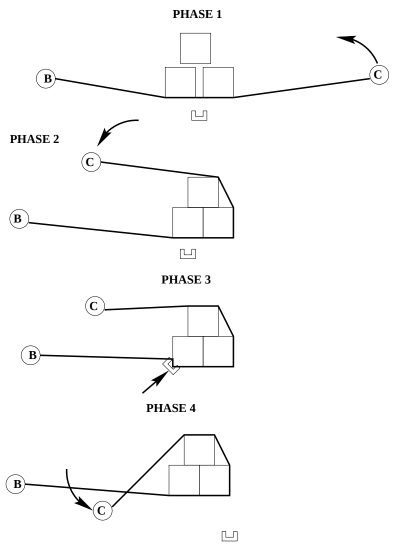

Figure 1: The 4 phases of binding (wrapping a rope around a group of objects). The RWI robots are denoted by disks. Bonnie is denoted by B; Clyde is denoted by C. The Pioneer helper is denoted by the gripper icon. The objects are denoted by rectangles.

Clyde applies the rotational power he just computed, and begins to move forward. As he moves forward, the tension in the rope combined with Clyde's state of rotational compliance causes him to turn and follow a (piecewise) circular path around the group of boxes. The tension in the rope stays roughly constant ($F_a = 4lbs.$) and Clyde monitors his odometry so that once his torso has rotated through 180 degrees, he will stop.

At the point he stops, he should be less than two feet from the length of rope running from Bonnie to the box nearest to her. Now, he waits (for a predetermined amount of time, about 30 seconds[2]) and the action shifts to Bonnie and Ben.

Through this first phase of the experiment, Bonnie and Ben have been dormant,[3] but now they come to life (using timing delays: no communication occurs between Clyde and the other two robots at this point).

[1] Clyde measures the rotational power for compliance by searching among power levels: He tries values of rotational power, until he finds the largest value that doesn't cause the robot to freely rotate.

[2] All the timeouts can be replaced by sensing operation.

[3] Our robots can be in one of three states: *active*, *passive*, and *dormant*. A passive robot may be in motion (for example, it may be towed compliantly by an active robot). A dormant robot is stationary, and it exerts no forces directly on the manipulandum. A dormant robot may act as a fixture with respect to the rope.

Bonnie turns toward the box nearest to her, and enters a state of translational compliance. Ben seeks out the corner of the box nearest to both the rope and to Bonnie. Once he locates the corner, he moves out and aligns himself with the box diagonal passing through that corner. He is now ready to grasp the rope against the box, and he slowly begins moving toward the corner.

As Ben's left claw encounters the rope, it exerts a force upon the rope that drags Bonnie a little way forward. However, at this point he must stop to avoid a motor stall (he is alerted to his contact with the rope using his touch sensors). When Bonnie notices that she has moved, she takes this as a signal from Ben that he is almost in position. At first, she gives just a little bit of slack ($F_p = 1lb.$). This allows Ben to get in position against the box once he re-engages his motors. After waiting a couple of seconds, Bonnie assumes that Ben is in position, and she moves forward some more. This time, the segment of rope between Ben and Bonnie falls to the ground. Note that for the crossover phase, F_a and F_p are not equal! F_a is still about 1 lb., whereas F_p is 0 lbs.

Now the stage is set for Clyde to actually cross over the rope. When his timeout expires, he once again goes to a state of rotational compliance, and moves forward for another 45 degrees worth of rotation. He then goes to a dormant state again, waiting for Bonnie and Ben to tidy up. When Bonnie wakes up, she backs up to retension the rope (so that once again, $F_p = F_a = 4lbs.$). This has a side effect of signaling to Ben that his job is done. The tension in the rope moves him backward a small distance. When he detects this movement, Ben backs up and moves out of the way so that Bonnie and Clyde can manipulate the group of boxes.

In order to perform flossing and ratcheting tasks, Bonnie needs to move around to the opposite side of the objects from Clyde. She now goes to a state of rotational compliance, and moves forward for about 135 degrees worth of rotation. When she reaches the final position, she rotates 90 degrees to face the objects. Clyde does the same, and the two robots are now ready to perform manipulation tasks with the objects.

2.3 Ratcheting

The ratcheting algorithm translates a group of objects. However, one side effect of the ratcheting algorithm is that the group also rotates as it translates.

The setup for ratcheting assumes that a wrapping operation has been performed, and that the robots are positioned on opposite sides of the bound group, facing each other. For a ratcheting operation, we designate one robot to be active, and the other to be passive. The active robot will do the work of moving the group, while the passive robot's main job is to help maintain the "grip" on the group with the rope. Initially, the rope is at moderately high tension ($F_a = F_p = 8lbs.$).

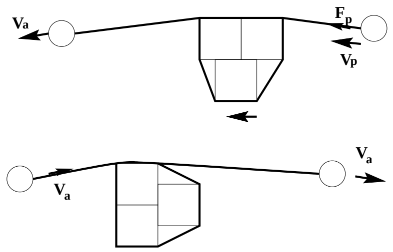

Figure 2: The 2 phases of translating a group of objects with a ratchet motion. The robots are denoted by disks. The group of objects is denoted rectangles. V_a denotes the velocity of the active robot. V_p denotes the velocity of the passive robot. F_p is the force felt through the rope by the passive robot.

In order for the robots to be in synch, the passive robot begins the experiment by signaling the active robot that it is ready to begin. The signal that is used is a small tug on the rope – just enough for the active robot to notice itself moving. The active robot begins the experiment by sitting in a loop that monitors the odometer, while maintaining a state of translational compliance. When it notices movement, it switches into its active role.

After giving the signal, the passive robot goes into a state of translational compliance. The active robot begins towing both the group of objects and the passive robot (at this point, we have $F_a = F_p =$ [*friction resistance of the box*] + [*resistance of the passive robot*] which in practice is about 15 lbs). The active robot will continue pulling until it senses the stop signal from the passive robot. During the first iteration, this occurs when the passive robot senses with its bump sensors that it is touching one of the boxes. To signal the active robot, the passive robot "digs its heels in" by applying significant power to its translational motors in the direction opposite to the direction of motion ($V_p \to 0$). The effect of this is to slow down the active robot (F_a reaches its maximum value at this point). While the active robot is pulling the group of objects, it is also monitoring its velocity (V_a). When V_a drops below a specified threshold (we use approx. $.65V_{max}$, where V_{max} is the velocity with which the robot was commanded to move) the active robot assumes it is receiving the stop signal, and it stops.

The passive robot has now gone as far forward as it can with the length of rope it was given. It is time to "ratchet." In order for the rope to slide around the group of boxes (as opposed to moving them), the tension in the rope must first be minimized. This is easily accomplished by having the active robot begin to move back toward the group immediately after receiving the stop signal, but having the passive robot wait for a short time (about 1.5 sec.) before moving away from the group. During this phase, $F_p = F_a < 1lb$. Now their roles are reversed. The active robot monitors its bump sensors to watch for contact with an object, and the passive robot moves, towing the rope (which slips along the box), while waiting for the stop signal from its partner. Again, the stop signal is transmitted through the rope. However, one important difference is that the active robot is not in a state of translational compliance. It is moving of its own volition, as is the passive robot.

Once the active robot reaches group of objects, the first iteration of the ratcheting task is complete. Further iterations can now be executed, each one moving the group further along its path. We decided to implement one minor change for the subsequent iterations, however. Rather than having contact with a box determine when the stop signal was issued, we preferred instead to have the robots measure the length of rope they have to work with during the first iteration, and use that measurement during later iterations, so that it was not necessary to bump into the boxes every time. This gives the experiments a smoother feel.

2.4 Flossing

The flossing algorithm reorients a group of objects. The task itself does not actually effect a pure rotation, but also translates the group slightly as a side effect. To demonstrate reorientation in either direction, we have the robots perform a kind of tug-o-war over the group of boxes. The algorithm for this task is symmetric with respect to the two robots. They simply take turns towing the group (and each other), and rotating the group back-and-forth. The robots alternate between the active and passive roles in the algorithm. We use a stop signal just like the one in the ratcheting experiment to allow the passive robot to signal the active robot that it is time to switch roles.

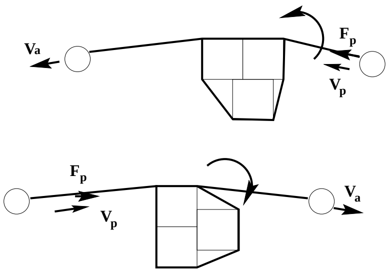

Figure 3: The 2 phases of rotating a group of objects by "flossing". The robots are denoted by disks. The group of objects is denoted by rectangles. V_a denotes the velocity of the active robot. V_p denoted the velocity of the passive robot. F_p is the force felt through the rope by the passive robot.

3 Analysis

In our lab, we implemented and experimented with the three manipulation primitives described in the previous section. We observed experimentally that while wrapping and manipulating groups of boxes, the boxes almost always settled into an arrangement where each box was flush (i.e. touching face to face) against its neighbors. This property leads to simplifications in the manipulation and wrapping operations. In this section we analytically explore this observation. We ask: for what class of box geometries does wrapping lead to a final configuration where each box is flush against its neighbors?

Our approach is to calculate the perimeter of the convex hull of the box arrangement and view it as a potential function. By pulling on the rope during the wrapping operation, we reduce this potential energy and force the boxes into a "lower energy state". We describe the geometry of these states. First, we consider the case of wrapping two boxes and present an analytical approach to defining the perimeter function and computing its local minima. We then consider the more complicated case of three boxes and describe a numerical simulation algorithm that can be generalized to an arbitrary number of boxes. Finally, we will examine the correspondence between simulation results and physical experiments.

3.1 Two Box Analysis

Consider a simple system with two boxes: box A has dimensions $\ell_1 \times w_1$, and box B has dimensions $\ell_2 \times w_2$ (see Figure 4), where the lengths and widths of the boxes are free variables. Our objective is to demonstrate analytically that by pulling on the rope and reducing the perimeter of the convex hull, C, of the boxes (we will call $|C|$ the *perimeter* of the boxes), we can achieve a stable state where the two boxes are flush against each other.

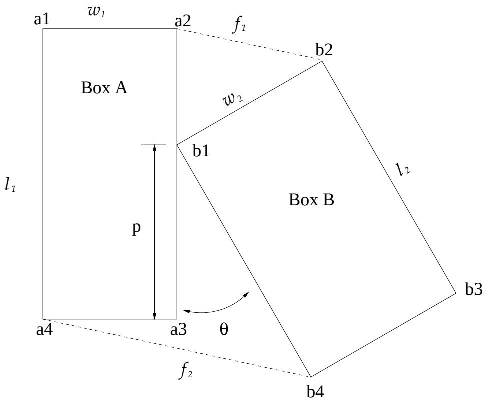

Figure 4: Basic two box setup

We view the perimeter of the boxes as a potential function. This function is a positive, real-valued function on a 6-dimensional manifold, parameterized by the lengths and widths of the two boxes, the angle formed between them, and the distance from the point of contact to an endpoint of the edge on which it lies.

In this formulation, we do not compute how the boxes move but focus on their relative equilibrium configurations. Clearly, if we wrap two boxes with a rope and pull on the ends we can move them together until they are in contact, assuming there are no obstacles in between them and large enough forces can be exerted to move the boxes. The contact between the boxes can be vertex-edge or edge-edge.

Consider the case of vertex-edge contact between the two boxes. Suppose box B has a vertex contact with an edge of box A. Let θ be the angle between the boxes and p the distance from the point of contact to the side of the box being touched. Box B has two edges touching box A (see Figure 4): b_4b_1 forms angle θ with A, and b_1b_2 forms angle $\pi/2-\theta$. We choose the edge with $\theta < \pi/4$ (the case where $\theta = \pi/4$ will need special consideration).

Note that without loss of generality, we can rescale the distances so that $\ell_1 = 1$.

Before defining a perimeter function for a general arrangement of two boxes, we observe that this function depends upon the arrangement of the boxes. For a fixed set of box geometries (i.e. fixing ℓ_1, w_1, ℓ_2, and w_2), we have a two dimensional slice, $\mathcal{C}$ of our manifold of configurations. This space is partitioned into several pieces, each with a different form for its perimeter function. For example, the perimeter function for the boxes in Fig. 4 can be expressed as

$$s(p,\theta) = \ell_1 + w_1 + f_1(p,\theta) + \ell_2 + w_1 + f_2(p,\theta).$$

This general form for the perimeter depends only upon the subset of vertices of the two boxes that lie on the convex hull. For the example in Figure 4 we have the sequence $\mathcal{A} = a_4, a_1, a_2, b_2, b_3, b_4$.

We define an *arrangement class* to be a set of pairs (p,θ), such that the corresponding arrangements all have the same sequence of vertices on their convex hulls. Hence, we are using vertex sequences to define arrangement classes. For a given arrangement class, $\mathcal{A}$, we can give a formula for the perimeter, and that formula will be valid as long as $(p,\theta) \in \mathcal{A}$.

There are nine distinct arrangement classes[4]:

$\mathcal{A}_1 = a_4, a_1, a_2, b_2, b_3, b_4$
$\mathcal{A}_2 = a_4, a_1, a_2, b_3, b_4$
$\mathcal{A}_3 = a_3, a_4, a_1, a_2, b_2, b_3, b_4$
$\mathcal{A}_4 = a_4, a_1, a_2, b_2, b_3$
$\mathcal{A}_5 = a_3, a_4, a_1, a_2, b_3$
$\mathcal{A}_6 = a_3, a_4, a_1, a_2, b_3, b_4$
$\mathcal{A}_7 = a_3, a_4, a_1, b_2, b_3, b_4$
$\mathcal{A}_8 = a_3, a_4, a_1, a_2, b_2, b_3$
$\mathcal{A}_9 = a_4, a_1, b_2, b_3, b_4$

For each one of these arrangements, there is a unique formula expressing the perimeter as a function of p and θ that can be expressed in closed form. The local minima of the perimeter functions can be ob-

[4]Note that for fixed dimensions of the two boxes, some of these arrangement classes will be empty.

Notation: θ_2, θ_3–orientations of Box 2 and Box 3.
p_2, p_3–positions of Box 2 and Box 3.
$\mathcal{C}^{\text{Box } i}_{\text{Box } j,\theta_j}$–configuration space obstacle of Box i w.r.t. Box j oriented at angle θ_j.

Input: Geometries of Box 1, Box 2, and Box 3.

Analyze-Config: Choose θ_2.
Compute $\mathcal{C}^{\text{Box } 1}_{\text{Box } 2,\theta_2}$.
Choose p_2.
Choose θ_3.
Compute $\mathcal{C}^{\text{Box } 1}_{\text{Box } 3,\theta_3}$ and $\mathcal{C}^{\text{Box } 2}_{\text{Box } 3,\theta_3}$.
Merge $\mathcal{C}^{\text{Box } 1}_{\text{Box } 3,\theta_3}$ and $\mathcal{C}^{\text{Box } 2}_{\text{Box } 3,\theta_3}$.
Choose p_3.

Output: Perimeter of convex hull of Box 1, Box 2, and Box 3.

Figure 5: Constructing Box Arrangements.

tained by examining their derivatives[5]. For example, if $s_i(p, \theta)$ denotes the perimeter for the arrangement class $\mathcal{A}_i$, we have: $s_1(p, \theta) = 1+w_1+\ell_2+w_2+g_1(p, \theta)+h_1(p, \theta)$ and $s_5(p, \theta) = 1 + 2w_1 + g_2(p, \theta) + h_3(p, \theta)$, where

$$g_1(p, \theta) = \sqrt{p^2 - 2p + 1 + w_2^2 - 2w_2 \sin\theta + 2pw_2 \sin\theta}$$
$$g_2(p, \theta) = \sqrt{p^2 - 2p + 1 + w_2^2 + \ell_2^2 + t_1(p, \theta)}$$
$$t_1(p, \theta) = -2w_2 \sin\theta + 2\ell_2 \cos\theta + 2pw_2 \sin\theta - 2p\ell_2 \cos\theta$$
$$h_1(p, \theta) = \sqrt{w_1^2 + \ell_2^2 + p^2 + 2w_1\ell_2 \sin\theta - 2p\ell_2 \cos\theta}$$
$$h_3(p, \theta) = \sqrt{w_2^2 + \ell_2^2 + p^2 + 2pw_2 \sin\theta - 2p\ell_2 \cos\theta}$$

3.2 A Numerical Model for Three Boxes

Ideally, we would like to obtain closed form solutions for arrangements of any number of boxes that minimize their perimeter. In the case of two boxes, these local minima correspond to box arrangements where the boxes have edge-edge contacts. However, it is not clear whether or not local minima of multiple box systems contain only edge-edge contacts. For example, with three boxes we can imagine a "jammed" configuration where one box is stuck in the corner defined by the other two boxes. Because the closed form solutions are rather involved even for the case of two boxes, we present a numerical approach to modeling three boxes that can be generalized to an arbitrary number of boxes.

Figure 5 shows the numerical algorithm that analyzes arrangements of three boxes whose geometries are known. With fixed box geometries, there are 4 variables to consider. We assume, without loss of generality, that the first box is centered at the origin, with sides parallel to the x and y axes and that all the boxes are touching. The positions and orientations of the other two boxes are (one-dimensional) free variables. The set of all possible configurations of three boxes forms a four dimensional manifold. We can define a positive real-valued function on this manifold that is the perimeter of the convex hull of the three boxes.

The algorithm works by assigning values to the free variables one at a time, using discretization and enumeration. The first loop chooses an orientation θ_2 for box 2. We now view box 1 as a configuration space obstacle for box 2 in order to position box 2 with respect to box 1. The second loop runs over the perimeter of this configuration space obstacle (using increments determined by a position resolution parameter), and generates values for p_2.

The same method can be used to add box 3 to the arrangement. Thus, the third loop chooses an orientation θ_3 for box 3 and computes its configuration space. The configuration space obstacle for the union of box 1 and box 2 is computed as the union of the configuration space obstacle for box 1 and the configuration space obstacle for box 2. This is a non-convex object. The algorithm computes the perimeter of the convex hull of this configuration space obstacle (a function of θ_2, p_2, and θ_3), and the fourth loop chooses a position p_3 along the perimeter of this configuration space obstacle for box 3. Having completely defined an arrangement of the three boxes, we can compute its perimeter and identify the smallest perimeter value.

[5] Due to space limitations we omit the complete description of these functions, which are quite long.

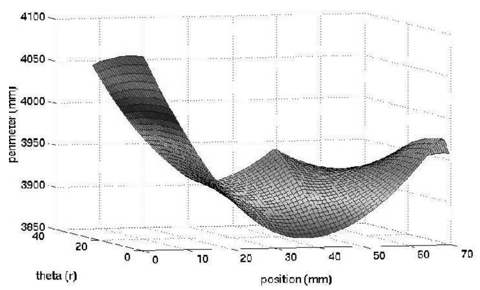

Figure 6: A two-dimensional slice of the potential function for an arrangement of three boxes.

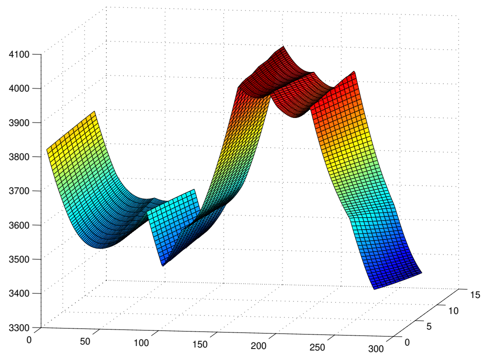

Figure 7: Wide range plot

4 Experiments

4.1 Simulation Experiments

We have implemented the algorithm outlined in Figure 5. Since the perimeter of a three-box arrangement is a function $s : R^4 \longrightarrow R$, we cannot graphically

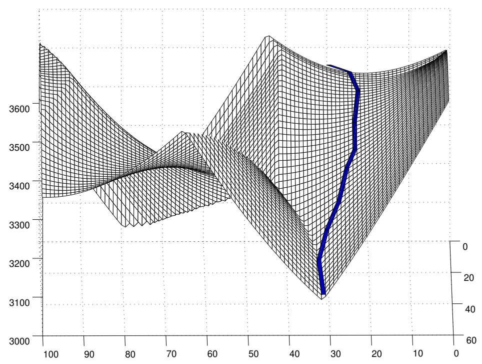

Figure 8: This figure shows a potential plot with a highlighted simulation path.

Task	Successes	Tries	Reliability
Flossing	17	18	94.4 %
Ratcheting	13	20	65 %

Figure 9: This table contains reliability data for our experiments for manipulation of three boxes by flossing and ratcheting.

depict where the local minima of the function occur. However, we can examine s along 2-dimensional slices, generally by fixing the position and orientation of Box 2 (p_2 and θ_2), and letting Box 3 move freely. Figures 6, 7, and 8 show sample potential surfaces computed in this manner.

Figure 6 shows details for how the perimeter function changes with respect to p_3 and θ_3, and Figure 7 shows a more global view of how perimeter changes with respect to p_3.

In Figure 8, we demonstrate how the box arrangements depicted in Figures 10 and 11 correspond to the potential surface drawn with the simulation program. The initial snapshot of Figure 10 corresponds to the upper end of the path in Figure 8. As the box rotates and falls into place in the corner, the configuration moves down the path and into a valley, eventually stopping at a local minimum.

4.2 Physical Experiments

In our lab, we built a distributed manipulation system using ropes by implementing the manipulation primitives described in Section 2 using the robots described in Section 2.1. The RWI robots are programmed in MOBOT-SCHEME [11] and the helper Pioneer robot is programmed in C++.

Figure 9 shows reliability data for some recent experiments run with this system. The failures in flossing and ratcheting were due to signaling errors. Occasionally, the passive robot stops prematurely. We believe that these errors are due to low battery levels during those specific experiments, but more investigation is necessary.

We also have run many binding experiments and compared the arrangements obtained in the physical system to the simulation predictions. Figures 10 and 11 shows some snapshots from a typical run, for corresponding box geometries and initial arrangements. Since the physical experiments are subject to frictional effects that are not modeled by our simulator, we note that we sometimes observe jammed arrangements (in physical equilibrium) that do not correspond to local minima. However, in most experiments, boxes that are set up in general position very rarely achieve this state.

5 Summary

In this paper we described the constrained prehensile manipulation of multiple objects bound by a rope. We presented three algorithms for the distributed manipulation of a bound group of objects: binding (tieing a rope around a group of objects), flossing (effecting rotations of a group of bound objects), and ratcheting (effecting translations of a group of bound objects). Our algorithms use a team of three robots that do not communicate directly but use the tension in the rope as a signaling mechanism. We presented an analytical and a numerical analysis for the final configuration of a group of bound objects after wrapping. We implemented the three manipulation skills and conducted physical experiments. We also implemented the numerical solution and examined the accuracy of the numerical predictions against physical experiments.

Acknowledgements

We are grateful to Keith Kotay and Neville Bonwitt for for extensive discussions and for constructing Ben's gripper.

References

[1] R. Beckers, O. Holland, and J. Deneubourg. From local actions to global tasks: stigmergy and collective robotics. in *Artificial Life IV*, R. Brooks and P. Maes eds., MIT Press, 1994.

Figure 10: Four snapshots from a typical wrapping run.

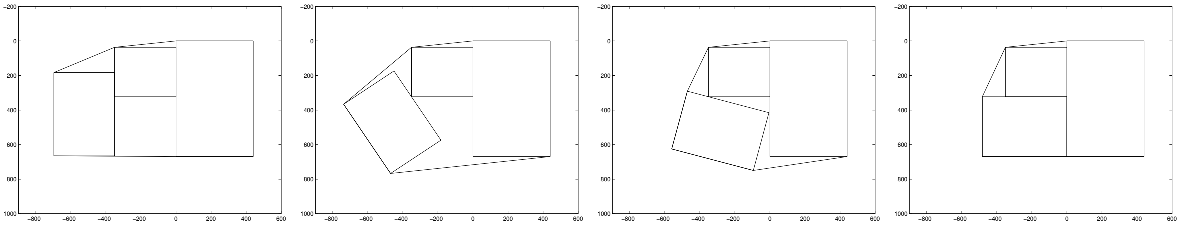

Figure 11: Box outlines for simulation

[2] B. Donald, J, Jennings, and D. Rus. Minimalism + Distribution = Supermodularity. *Journal of Experimental and Theoretical Artificial Intelligence*, 9(1997) 293-321, 1997.

[3] B. Donald, J. Jennings, and D. Rus. Information invariants for cooperating autonomous mobile robots. *International Journal of Robotics Research*, 16(5) 673-702, 1997.

[4] B. Donald, L. Gariepy, and D. Rus. Experiments in constrained prehensile manipulation: distributed manipulaiton with ropes. *The 1999 International Symposium on Experimental Robotics, eds. P. Corke*, LNCS, to appear (1999).

[5] M. Erdmann and M. Mason. An Exploration of Sensorless Manipulation. In *Proceedings of the 1986 IEEE International Conference on Robotics and Automation*, 1986.

[6] K. Y. Goldberg. Orienting polygonal parts without sensing. *Algorithmica*, 10(2/3/4):201–225, August/September/October 1993.

[7] J.-C. Latombe. *Robot Motion Planning*. Kluwer Academic Press, 1991.

[8] M. Mason. Manipulator grasping and pushing operations. *International Journal of Robotics Research*, 5(3):53–71, 1986.

[9] F. Noreils Toward a Robot Architecture Integrating Cooperation Between Mobile Robots: Application to Indoor Environment. *International Journal of Robotics Research* Vol. 12, 1993.

[10] L. Parker. *Heterogeneous Multi-Robot Cooperation*. PhD thesis, MIT EECS department, 1994.

[11] J. Rees and B. Donald. Program Mobile Robots in SCHEME. *Proceedings of the 1992 International Conference on Robotics and Automatio.* Nice, France 1992.

[12] D. Rus, B. Donald, and J. Jennings. Moving furniture with teams of autonomous robots. *Proceedings of the 1995 Conference on Intelligent Robot Systems*. 1995.

[13] D. Rus, A. Kabir, K. Kotay, and M. Soutter. Guiding Distributed Manipulation with Mobile Sensors. In *Proceedings of the 1996 International Conference on Multisensor Fusion and Integration for Intelligent Systems*, Washington, 1996.

[14] D. Rus. In-hand dexterous manipulation of piecewise-smooth objects. *International Journal of Robotics Research*, 18(4), 355-381, 1999.

[15] Y. Yamamoto. Coordinated Control of A Mobile Manipulator. Techical Report MS-CIS-94-12 / GRASP LAB 372, Department of Computer Science, The University of Pennsylvania, 1994.