Analyzing Teams of Cooperating Mobile Robots[1]

Bruce Randall Donald **James Jennings** **Daniela Rus**
Robotics & Vision Laboratory
Computer Science Department
Cornell University
Ithaca, New York

Abstract

In [Don4], we described a manipulation task for cooperating mobile robots that can push large, heavy objects. There, we asked whether explicit local and global communication between the agents can be removed from a family of pushing protocols. In this paper, we answer in the affirmative. We do so by using the general methods of [Don4] analyzing *information invariants*. We discuss several measures for the information complexity of the task of pushing with cooperating mobile robots, and we present a methodology for creating new manipulation strategies out of existing ones. We develop and analyze synchronous and asynchronous manipulation protocols for a small team of cooperating mobile robots than can push large boxes. The protocols we describe have been implemented in several forms on the Cornell mobile robots in our laboratory.

1 Introduction

In this paper, we develop and analyze synchronous and asynchronous manipulation protocols for a small team of cooperating mobile robots than can push large boxes. The boxes are typically several robot diameters wide, and 1-2 times the mass of a single robot, although the robots have also pushed couches that are heavier (perhaps 2-4 times the mass, and 8×3 robot diameters in size). We build on the ground-breaking work of [Mason, EM] and others on planar sensorless manipulation. Our work differs from previous work on pushing in several ways. First, the robots and boxes are on a similar dynamic and spatial scale. Second, a single robot is not always strong enough to move the box by itself (specifically, its "strength" depends on the effective lever arm). Third, we do not assume the robots are globally coordinated and controlled. Fourth, our protocols assume neither that the robot has a geometric model of the box, nor that the first moment of the center of friction is known. Instead, the robot combines sensorimotor experiments and manipulation strategies to infer the necessary information (the experiments have the flavor of [JR]). Finally, the pushing literature generally regards the "pushers" as moving kinematic constraints. In our case, because (i) there are at least two robot pushers and (ii) the robots are less massive than the box, the robots are really "force-appliers" in a system with significant friction. In this sense, our task is in some ways closer in flavor to *dynamic manipulation* [ML], even though the box dynamics are essentially quasi-static.

We develop a framework for analysis and synthesis, based on *information invariants* [Don4], to reveal these assumptions and expose the information structure of the task. We believe our theory has implications for the parallelization of manipulation tasks on spatially distributed teams of cooperating robots. To develop a parallel manipulation strategy, we start with a perfectly synchronous protocol. Next, in distributing it among cooperating, spatially separated agents, we relax it to a MIMD[2] protocol with local communication and partial synchrony. Finally, we remove all explicit communication. The final protocols are essentially "uniform" in that the same program runs on each robot. However, the result is asynchronous, and hence it cannot be characterized as exclusively SIMD[3] nor MIMD. Ultimately, the robots must be viewed as communicating implicitly through the task dynamics, and this implicit communication confers a certain degree of synchrony on our protocols.

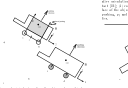
1.1 The Big Picture

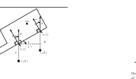
Our goal here is to investigate the information requirements for the pushing tasks. This paper uses the theoretical framework and methodology introduced in [Don,Don4]. A central theme to previous work [Don] has been to determine what information is required to solve a task, and to direct a robot's actions to acquire that information to solve it. Key questions concern: (a) How much internal state should the robot retain? (b) How many cooperating agents are required, and how much communication between them is necessary? (c) How can the robot change (side-effect) the environment in order to record state or sensory information to perform a task? (d) How much information is provided by sensors? (e)

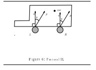
[1] Support for this work is provided in part by the NSF under grants No. IRI-8802390, IRI-9000532, IRI-9006137, by ONR under contracts No. N00014-92-J-1989 and N00014-92-J-39, by a Presidential Young Investigator award, and by the Air Force Office of Sponsored Research, the Mathematical Sciences Institute, Intel Corporation, and AT&T Bell laboratories.

[2] Multiple Instructions, Multiple Data

[3] Single Instruction, Multiple Data







Left Robot	Communication	Right Robot
measure($x_1(0)$)		measure($x_2(0)$)
	$\boxed{L \to R}$ ($x_1(0)$)	
		$\Sigma_0 = x_2(0) - x_1(0)$
Repeat :		Repeat :
measure($x_1(t)$)		measure($x_2(t)$)
	$\boxed{L \to R}$ ($x_1(t)$)	
		$\Sigma = x_2(t) - x_1(t)$
		$s = \mathrm{sign}(\Sigma - \Sigma_0)$
	$\boxed{L \leftarrow R}$ (s)	
case ($s = 0$) ⇒ push(p)		case ($s = 0$) ⇒ push(p)
($s = -1$) ⇒ move(R)		($s = -1$) ⇒ move(R)
($s = 1$) ⇒ move(L)		($s = 1$) ⇒ move(L)

Figure 5: Protocol I (Quasi-Static). The case for breaking contact is not shown; it can be handled as in Figure 3.

Left Robot	Right Robot
$\theta_0 \leftarrow \theta(0)$	$\theta_0 \leftarrow \theta(0)$
Repeat :	Repeat :
case (break?) ⇒ guarded-move(p)	case (break?) ⇒ guarded-move(p)
($\theta(t) \approx \theta_0$) ⇒ push($p$)	($\theta(t) \approx \theta_0$) ⇒ push($p$)
($\theta(t) \gg \theta_0$) ⇒ move(L)	($\theta(t) \gg \theta_0$) ⇒ $\emptyset$ (∗)
($\theta(t) \ll \theta_0$) ⇒ $\emptyset$ (∗∗)	($\theta(t) \ll \theta_0$) ⇒ move(R)

Figure 7: Protocol II. This protocol is "almost uniform," and can be made uniform by changing the $\emptyset$ lines (∗) to MOVE(L) and (∗∗) to MOVE(R). Note that "uniform" does not quite imply SIMD, since the protocols run asynchronously.

cuits." (See [Don4] for a full treatment of these concepts.) We model the circuits as graphs. Vertices correspond to different sensori-computational components of the system (what we will call "resources" below). Edges correspond to "data paths" through which information passes. Different immersions of these graphs correspond to different spatial allocation of the "resources." We also define an operator $+$ as a way to "combine" sensori-computational systems. Below we use the term "*sensor system*" to mean "*sensori-computational system*" where it is mellifluous.

3.1 Situated Sensor Systems

Definition 3.1 *A labelled graph $\mathcal{G}$ is a directed graph (V, E) with vertices V and edges E, together with a labelling function that assigns a label to each vertex and edge. Where there is no ambiguity, we denote the labelling function by ℓ.*

Definition 3.2 *A sensor system S is represented by a labelled graph (V, E). Each vertex is labelled with a component. Each edge is labelled with a connection.*

See Figures 8-9 for illustrations for the circuits—that is, the sensor systems for protocols I (QS) and II. We will use the terms *protocol* to refer to the computer programs in Figures 5-7, and *circuit* for the sensor systems in Figures 8-9. In the figures, the components correspond to boxes: $\boxed{\mathrm{ODOM}}$ is an odometer, the "signal"[4] coming out of this box is the odometry reading. The box $\boxed{-}$ performs subtraction, the boxes $\boxed{x_1(0)}$ and $\boxed{\Sigma_0}$ are registers, and they implement internal state. The part of the circuit labelled **case** interfaces to the control part of the circuits,

[4] We use "signal" as a metaphor; our circuits are strictly digital. Either *message* or *stream* would be better, but both have distracting religious connotations.