SHARED CONTROL POLICIES AND TASK LEARNING FOR HYDRAULIC EARTH-MOVING MACHINERY

A Thesis

by

MITCHELL ANTHONY ALLAIN

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Prabhakar Pagilla
Committee Members,	Bryan Rasmussen
	Xingyong Song
Head of Department,	Andreas Polycarpou

August 2017

Major Subject: Mechanical Engineering

Copyright 2017 Mitchell Anthony Allain

ABSTRACT

This thesis develops a shared control design framework for improving operator efficiency and performance on hydraulic excavation tasks. The framework is based on blended shared control (BSC), a technique whereby the operator's command input is continually augmented by an assistive controller. Designing a BSC control scheme is subdivided here into four key components. *Task learning* utilizes nonparametric inverse reinforcement learning to identify the underlying goal structure of a task as a sequence of subgoals directly from the demonstration data of an experienced operator. These subgoals may be distinct points in the actuator space or distributions over the space, from which the operator draws a subgoal location during the task. The remaining three steps are executed on-line during each update of the BSC controller. In real-time, the *subgoal prediction* step involves utilizing the subgoal decomposition from the learning process in order to predict the current subgoal of the operator. Novel deterministic and probabilistic prediction methods are developed and evaluated for their ease of implementation and performance against manually labeled trial data. The *control generation* component involves computing polynomial trajectories to the predicted subgoal location or mean of the subgoal distribution, and computing a control input which tracks those trajectories. Finally, the *blending law* synthesizes both inputs through a weighted averaging of the human and control input, using a blending parameter which can be static or dynamic. In the latter case, mapping probabilistic quantities such as the *maximum a posteriori* probability or statistical entropy to the value of the dynamic blending parameter may yield a more intelligent control assistance, scaling the intervention according to the confidence of the prediction.

A reduced-scale (1/12) fully hydraulic excavator model was instrumented for BSC experimentation, equipped with absolute position feedback of each hydraulic actuator. Experiments were conducted using a standard operator control interface and a common earthmoving task: loading a truck from a pile. Under BSC, operators experienced an 18% improvement in mean digging efficiency, defined as mass of material moved per cycle time. Effects of BSC vary with regard to pure cycle time, although most operators experienced a reduced mean cycle time.

ACKNOWLEDGMENTS

Before beginning this thesis, I want to first acknowledge and thank the many people that have made it possible by contributing their support over the years.

To my advisor Dr. Prabhakar Pagilla, thank you for lending your wisdom and experience to this thesis. Your calm and persistent questioning was a model for conducting sound scientific research.

To all the members of my research group: Shyam Konduri, Orlando Cobos, Angel Gomez, Yalun Wen, and Zongyao Jin, your friendship and spirit have made the long hours spent in the lab a true pleasure. I would like to also thank Harshal Maske of the DASLAB, University of Illinois at Urbana-Champaign, for offering his advice, fellowship, and knowledge of the hydraulic excavation domain in the early stages of this research.

To all the undergraduate mechanical engineering students of Texas A&M whom I have had the pleasure to work with as a teaching assistant, your energy and curiosity were a constant source of inspiration.

To my parents, Chris and Tammy, your support and encouragement over so many years have made the pursuit of this degree possible. You provide a constant example for living a happy, healthy, and fulfilling life. Thank you.

Finally, to Teal, your generous love, patience, and lightheartedness provide balance in everything I do. Despite the distance, you offered immediate and unhesitating support in my desire to attend graduate school, and I am immensely grateful.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This thesis was supported by a committee consisting of Professor Prabhakar Pagilla (advisor) and Professor Bryan Rasmussen of the Department of Mechanical Engineering and Professor Xingyong Song of the Department of Engineering Technology & Industrial Distribution.

This research was conducted as part of an ongoing collaborative effort, with complementary research being conducted by Dr. Girish Chowdhary and Harshal Maske of the Department of Agricultural and Biological Engineering, University of Illinois Urbana-Champaign, and Dr. Charles Abramson and Emily Kieson of the Department of Psychology, Oklahoma State University.

Discussions with H. Maske at UIUC helped to direct our investigations of subgoal prediction methods. All other work conducted for this thesis was completed by the student independently.

Funding Sources

Graduate study was generously supported by the Sally and Ray Bowen '58 Fellowship from Texas A&M University. This research was made possible in part by the National Science Foundation under Award No. 1527828.

NOMENCLATURE

SC	Shared Control
CSC	Continuously Shared Control
BSC	Blended Shared Control
HIL	Human-in-the-Loop
OOTL	Out-of-the-Loop
RL	Reinforcement Learning
IRL	Inverse Reinforcement learning
BNIRL	Bayesian Nonparametric IRL
DPMIRL	Dirichlet Process-Means IRL
GMM	Gaussian Mixture Model
CP-GMM	Changepoint GMM IRL
FSM-P	Finite State Machine Prediction
AC-P	Action Comparison Prediction
MVN-AC-P	Multivariate Normal Action Comparison Prediction

TABLE OF CONTENTS

Page

ABSTRACT . . . ii

ACKNOWLEDGMENTS . . . iv

CONTRIBUTORS AND FUNDING SOURCES . . . v

NOMENCLATURE . . . vi

TABLE OF CONTENTS . . . vii

LIST OF FIGURES . . . ix

LIST OF TABLES . . . xi

1. INTRODUCTION AND LITERATURE REVIEW . . . 1

1.1 Motivation . . . 1
1.1.1 Automation as a Continuum . . . 2
1.1.2 Application Domain . . . 2
1.1.3 Research Objective . . . 3
1.2 Thesis Overview . . . 3
1.3 Background and Relevant Literature . . . 3
1.3.1 Excavator Hydraulic Systems . . . 5
1.3.2 Kinematic and Dynamic Modeling . . . 7
1.3.3 Automatic Control of Hydraulic Manipulators . . . 8
1.3.4 Shared Control . . . 9
1.3.5 Types and Examples of Shared Control . . . 10
1.3.6 Learning from Demonstration . . . 14
1.3.7 Summary . . . 15
1.3.8 Contributions . . . 16

2. KINEMATIC AND DYNAMIC MODELING . . . 18

2.1 Nomenclature . . . 18
2.1.1 Hydraulic Characteristics . . . 18
2.1.2 Kinematics . . . 18
2.1.3 Actuator Mechanics . . . 18
2.1.4 Other Symbols . . . 18

2.2 Kinematics 20
2.3 Hydraulic Actuation 23
2.3.1 Pressure-Flow Relationship 25
2.3.2 Flow Continuity 26
2.3.3 Actuator Mechanics 28
2.4 Manipulator Dynamics 29

3. TASK LEARNING AND SUBGOAL PREDICTION 31

3.1 Task Learning 31
3.1.1 Markov Decision Processes 32
3.1.2 Reinforcement Learning and Inverse Reinforcement Learning . 33
3.1.3 Bayesian Nonparametric IRL 33
3.1.4 Dirichlet Process Means IRL 40
3.1.5 Changepoint Gaussian Mixture Model Clustering 44
3.1.6 Summary of Task Identification 47
3.1.7 Stochastic Transition Matrix 47
3.2 Prediction 49
3.2.1 Deterministic Prediction 49
3.2.2 Probabilistic Action Comparison Prediction 53
3.2.3 Multivariate Normal Action Comparison Prediction 54
3.2.4 Prediction Comparison and Summary 56
3.2.5 Wireframe Simulation 59

4. TRAJECTORY CONTROL AND BLENDING 60

4.1 Trajectory Generation 60
4.2 Low-Level Control 61
4.3 Blending of Control Inputs 64

5. EXPERIMENTAL DESIGN AND RESULTS 69

5.1 Experimental Setup 69
5.1.1 1/12th Scale Hydraulic Excavator 69
5.1.2 Instrumentation 70
5.1.3 Operator Interface 74
5.2 Deterministic BSC Trials 76

6. SUMMARY AND FUTURE WORK 82

REFERENCES 85

APPENDIX A. KINEMATIC AND ACTUATOR PARAMETERS OF THE REDUCED-SCALE MODEL 92

LIST OF FIGURES

FIGURE Page

1.1 Anatomy of an excavator 4

1.2 The SAE control interface 5

2.1 Denavit-Hartenberg frame selection and parameters, and points $A - H$ 20

2.2 A 4/3 spool valve 24

2.3 The closed-center (overlapped) valve configuration 25

2.4 Front and rear views of the rotary valve manifold on the experimental model; *two valves are hidden on the front view to illustrate the port arrangement on the manifold* 26

2.5 Orifice area versus valve displacement 27

3.1 Action likelihood over all angles θ and three β parameters 36

3.2 Observations in training data; states are circles, and actions are represented as vectors from that state 38

3.3 Resulting subgoals from BNIRL algorithm with varying parameters . 39

3.4 Posterior subgoal counts for a grid of parameter sets 39

3.5 The motion classes on each actuator for a manual dig cycle 43

3.6 Labeled observations and DPMIRL subgoal means 43

3.7 States and acceleration vector norms for a manual truck loading operation 45

3.8 Changepoint clustering results 46

3.9 The 6 termination sets illustrated in the workspace. Note that the sets are defined in the actuator space, from which the forward kinematics have been used to illustrate these regions in the end-effector space. . . 51

3.10 Prediction state diagram for subgoal k. Since the task is cyclic, the conventions $k-1$ and $k+1$ are used here merely for convenience, and actually refer to the previous and next subgoals, respectively. 52

3.11 Two test cases of the Gaussian action likelihood 56

3.12 Comparison of prediction methods 57

3.13 A wireframe kinematic simulation for developing prediction algorithms 59

4.1 Tracking performance for simultaneous tracking on all actuators; *the stick actuator receives low flow priority and cannot meet tracking demands, even with the valve saturated* 63

5.1 The 1/12th-scale instrumented hydraulic excavator 70

5.2 Steady-state actuator velocity as a function of PWM duty cycle . . . 71

5.3 String potentiometer assembly 73

5.4 Calibration curves for the string potentiometers 74

5.5 BeagleBone Cape PCB 74

5.6 Overview of experimental setup 75

5.7 Operator view of workspace 75

5.8 Logical overview of BSC software 77

5.9 Sampled data for the swing actuator in the three operating modes. Vertical dotted lines denote the end of a cycle, i.e., the end of subgoal 6. 78

5.10 The effect of blending on cycle time varies with the operator style and experience. (M: Manual, B: Blended, O: Operator, AU: Autonomous) 79

5.11 Comparison of cycle efficiency for different operators and operating modes. Cycle efficiency is measured as the mass of material moved in a cycle divided by the cycle time. Higher efficiency is desirable. (M: Manual, B: Blended, O: Operator) 79

5.12 Comparison of cycle time and cycle efficiency across all operators. (M: Manual, B: Blended) 80

LIST OF TABLES

TABLE Page

1.1 Traits of human and computer agents 10

1.2 Types and examples of shared control 12

2.1 Denavit-Hartenberg parameters for the 1/12 scale excavator model . . 22

3.1 Summary of task identification algorithms 48

3.2 Summary of prediction algorithm performance 58

3.3 Summary of subgoal prediction algorithms 58

5.1 Experimental parameters for learning and prediction. The trials were run prior to the development of the changepoint filtering process, so a set of points were selected from the data and clustered using a DP-GMM. The termination sets were defined by the mean of the subgoal distribution plus or minus 2 standard deviations on each actuator, i.e., $2\sqrt{\Sigma_{ii}}$. The initiation sets were identified by observing the common movements between subgoals and applying a threshold on those actuators in the subgoal direction. 81

A.1 Kinematic parameters of the reduced-scale excavator model; $^*r_{\text{cyl},1}$ denotes the length of the hydraulic cylinder (excluding the rod extension length), all other parameters are consistent with Sec. 2.2. 92

A.2 Measured actuator parameters; note that u_{db} here is the deadband as a percentage of PWM duty cycle, as in Fig. 5.2, and $\dot{\zeta}_{i,\max}$ and $\ddot{\zeta}_{i,\max}$ are the maximum velocity and acceleration, respectively, of the i^{th} actuator. 93

1. INTRODUCTION AND LITERATURE REVIEW

1.1 Motivation

The benefits of automatic control for industrial manipulation are well known and clearly documented in literature. By exploiting the repetitive nature of manufacturing and material handling processes, automation has reduced labor costs and improved repeatability and efficiency. However, these automation techniques have not translated swiftly to less structured application domains.

In particular, earthmoving tasks like digging, compacting, and grading are still executed manually by skilled operators, with few exceptions. Safe and efficient execution of these tasks with hydraulic earthmoving equipment plays a critical role in the construction, mining, and forestry industries. Recently, industry experts have anticipated a growing need for computer assisted control due to an expanding skill gap among operators and growing job and equipment complexity [1]. Also, market analysts suggest that the earthmoving equipment industry alone is expected to grow to \$180 billion USD by 2022, with strong emphasis on research and development as a major driving factor [2].

Fully autonomous earthmoving equipment were investigated as early as the 1990's for prospective benefits in task completion time, operational efficiency, performance, and repeatability, as in Bradley, *et al.* [3] and Rowe, *et al.* [4]. However, the lack of situational awareness and flexibility in autonomous architectures, combined with an industry that is notably risk averse, have driven research interest toward development of novel control schemes which are human-centric.

1.1.1 Automation as a Continuum

Key to these novel control schemes is the paradigm that automation exists on a *continuum* between two extrema, fully manual operation and full autonomy [5]. Where the current capabilities of full autonomy fail to meet certain application demands, inclusion of a human in the control loop, or *human-in-the-loop (HIL)*, enables a more robust set of system behaviors. These control strategies which leverage a *human agent (HA)* and *computer agent (CA)* in cooperation have been given numerous designations, but in this work are referred to collectively as *shared control (SC)* .

1.1.2 Application Domain

Among the range of earthmoving equipment in the market today, excavators are a popular tool due to their jobsite versatility. Analysts suggest that the excavator market will exhibit outsized growth due to broad demand in residential applications and infrastructure development [2]. Modern intelligent excavators already feature advanced sensor suites, which may enable implementation of shared control algorithms with limited cost or retrofitting. Even a small addition in productivity and efficiency in state-of-the-art excavator control systems could sum to major economic advantages. Therefore, the excavator will be of central study in this work, with the expectation that the core strategies may be extended to similar earthmoving machinery.

Although the methods of this work are developed primarily for the earthmoving industry, these techniques are not exclusively applicable to the earthmoving domain. The proposed control schemes 1) exploit the *cyclical* nature of earthmoving tasks to improve task efficiency in a dynamic environment, while 2) relinquishing control smoothly to the human operator during off-nominal situations.

1.1.3 Research Objective

The objectives of this research are to develop a shared control policy for excavators which improves digging task efficiency without sacrificing situational robustness, and to measure the effectiveness of the proposed control policy.

1.2 Thesis Overview

This thesis is organized as follows: a practical background of hydraulic excavators and review of relevant literature are presented in the remainder of Chapter 1. The kinematics and dynamics of an excavator are developed in Chapter 2, along with a discussion of low-level position control of an excavator. Chapters 3 and 4 discuss the theoretical frameworks for the individual components of the proposed shared control architecture, namely, task learning and subgoal prediction, and control generation and blending. Chapter 5 documents the development of a reduced-scale hydraulic excavator testbed, some practical considerations in software implementation, and a series of experiments conducted to validate the proposed control methodologies. Finally, Chapter 6 gives a brief summary of the key contributions of this work and identifies areas for further investigation.

1.3 Background and Relevant Literature

The excavator is a member of a family of hydraulic earthmoving equipment including excavators, front loaders or wheel loaders, backhoe loaders, and motor graders. This class of equipment leverages fluid power by means of hydraulic cylinders and hydraulic motors for various tasks including but not limited to: digging, drilling, compacting, and grading (leveling or producing a specified slope).

An excavator has the following three main sub-assemblies: a base or undercarriage, upper structure (cabin and engine bay), and manipulator. The typical excavator exhibits six degrees of freedom (DOF): two independently driven tracks, a swing motor which adjusts the azimuth orientation of the upper structure, and three DOF manipulator (or arm). The three revolute joints share parallel axes, and form an open kinematic chain usually terminating with a toothed bucket. The links and corresponding hydraulic actuators of the manipulator are referred to as the *boom*, *stick*, and *bucket*. These components are illustrated in Fig. 1.1.

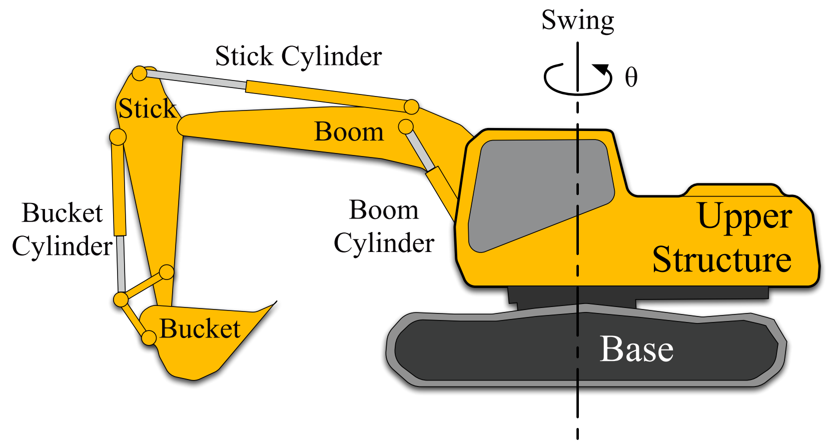


Figure 1.1: Anatomy of an excavator

Typically, a human operator controls the excavator from the cabin with a pair of levers and a pair of foot pedals, as illustrated in Fig. 1.2. The levers each have two axes, each axis controlling either the swing motor or one of the three hydraulic actuators. The foot pedals are used to drive the tracks on the undercarriage forward and backward. This configuration is standardized by the Society of Automative Engineers (SAE) and is used almost universally in the USA, irrespective of the manufacturer.

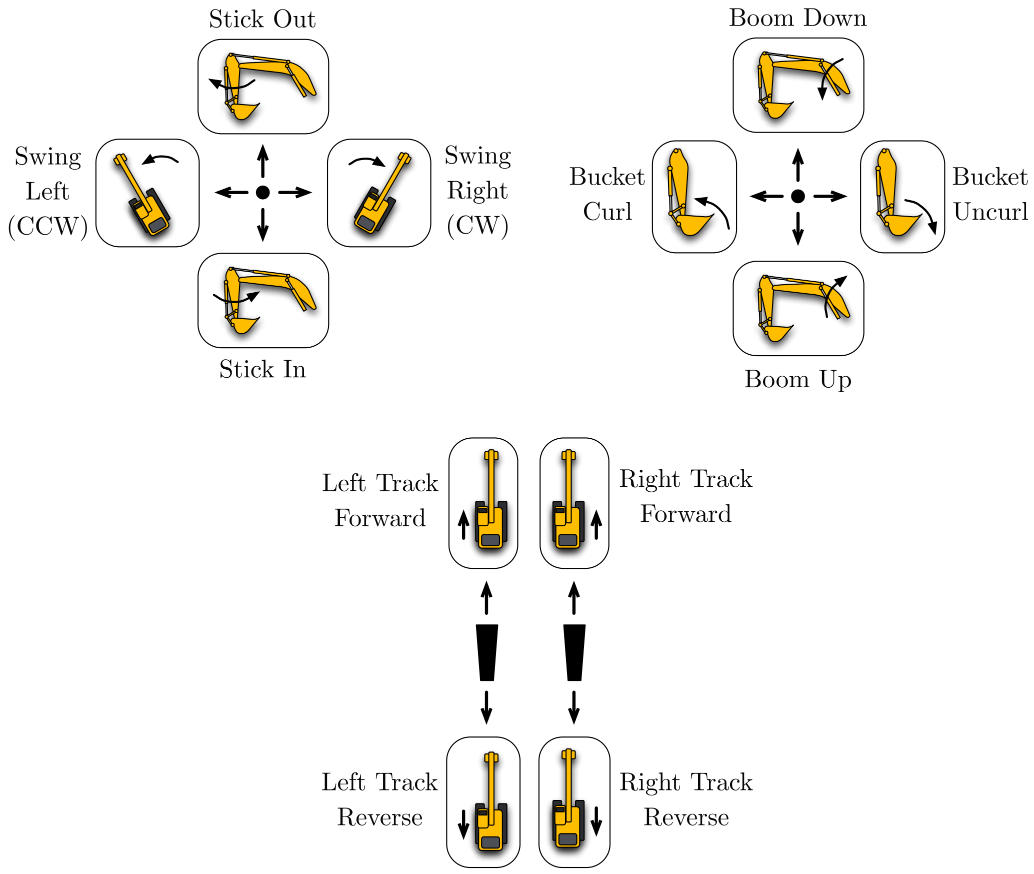


Figure 1.2: The SAE control interface

1.3.1 Excavator Hydraulic Systems

Modern electro-hydraulic actuator systems used in earthmoving machinery vary greatly, with the latest control and hardware adaptations targeting fuel efficiency and responsiveness. However, some components are reasonably consistent between new model excavators.

Typically, hydraulic fluid is circulated by one or more variable displacement axial-piston pumps. The fluid is directed to the actuators and throttled by means of directional control valves, usually of the spool type, which consist of a linear spool that obstructs the ports of the valve until actuated. The main hydraulic spool valves are typically actuated by solenoids (i.e., servovalves) or hydraulic pilot lines, which are often driven by a separate gear pump. Usage of a low pressure pilot line stiffens the joystick spool command against the flow forces on the main spool.

One classification of valve spool arrangements involves the operation of the valve in the neutral position, i.e., with no spool displacement. Open center valves, also called underlapped, have a geometry which allows fluid passage with no valve displacement. A single open center actuator circuit will maintain a low pressure flow with no actuator load and can employ a standard fixed displacement pump. Closed center, or overlapped, spools will build a high pressure at the neutral spool position, due to the completely impeded flow. Closed center valves usually enable better fuel performance, but require the use of more costly variable displacement pumps.

Among variable displacement pump applications, some discussion of the pump speed regulation is warranted as it impacts the responsiveness of the actuators. Simple constant pressure schemes set a reference pressure and modulate pump displacement to maintain that pressure. This reference pressure also dictates the maximum load capability of the actuator.

Alternatively, load sensing (LS) involves feeding back the pressure drop across the valve orifice to increase pump displacement and maintain constant flow in the presence of a large load. LS hydraulic circuits raise system pressure to match the largest load requirement, while maximizing efficiency during light operation. This feedback is currently achieved hydraulically, although electric systems are in development and are expected to provide a faster response, better stability, and less hydraulic power loss.

1.3.2 Kinematic and Dynamic Modeling

Hydraulic system dynamics and hydraulic control system design have been presented in detail by many authors [6, 7]. In these works, dynamic models of varying complexity are established for most hydraulic system topologies, with various pump, valve, and actuation types. Despite the different hydraulic topologies, some dynamic characteristics are common to hydraulic systems. One notable nonlinearity in hydraulic actuation is valve flow, governed by the classical orifice equation, which is nonlinear in terms of the pressure differential. Other nonlinearities include flow coupling and valve deadband, the latter of which is only present in closed-center valve spool arrangements [8]. The nonlinear coupling effects of shared flow sources in excavators were explored by Sepehri [9].

Prior to the 1990's, more comprehensive models of hydraulic excavators received little scientific attention. However, the pursuit of automatic control systems to execute digging tasks prompted several authors to develop more extensive dynamic models. In most of these developments, the models were then subsequently used in model-based design and simulation of automatic control systems. Vaha and Skibniewski developed a dynamic model for the 3-DOF excavator manipulator, utilizing Newton-Euler equations for each local joint frame [10]. Soon after, Koivo, *et al.* presented a systematic development of the kinematics and dynamics of the 4-DOF manipulator, with the inclusion of the swing axis, using similar Newton-Euler modeling techniques [11]. In Koivo's frequently cited paper, a soil-bucket interaction model is also presented, which attempts to quantify the reaction force on the bucket, and is based on work by Alekseeva *et al.* [12]. Although useful, these soil models are highly empirical and require a priori knowledge of soil parameters.

1.3.3 Automatic Control of Hydraulic Manipulators

Linear and nonlinear control techniques for hydraulic systems have a rich and lengthy history, which is described with appropriate detail in [6] and [7]. These control design methods are hereafter referred to as *low-level* for the sake of clarity. These so-called *low-level* techniques seek to solve classical control problems of stability, reference tracking, and disturbance robustness. Typically, in autonomous systems, a *high-level* controller is designed to address the problem of generating the behaviors or trajectories needed to execute a task and passing them to the low-level controller.

One of the earliest fully integrated autonomous excavation systems was the Lancaster University Computerised Intelligent Excavator (LUCIE). LUCIE was designed to efficiently dig trenches with a heavily hierarchical and rule-based high-level controller, which sends velocity commands to a low-level controller [3]. Position control was initially avoided due to tracking difficulty during soil contact and reverse kinematic complexities. LUCIE was later adapted by Gu and Seward to utilize a proportional-integral-plus gain scheduling (PIP) methodology for position trajectory following in free air only.

Sirouspour and Salcudean developed an adaptive nonlinear controller for hydraulic manipulators with the backstepping design method, and demonstrated its effectiveness on a hydraulic Stewart platform [13, 14].

Hydraulic earthmoving machines exhibit both kinematic coupling via the manipulator arm and hydraulic coupling due to shared flow and pressure sources among actuators. Although current control design methods are mostly based on single-input single-output (SISO) techniques, some researchers have proposed robust controllers which take into account the multi-input multi-output (MIMO) nature of earthmoving systems. In control of a wheel loader, which is kinematically similar to an excavator, Fales and Kelkar presented an H_∞ controller with feedback linearization for automatic bucket leveling, based on a high fidelity dynamic model [15, 16].

One control issue that is characteristic of earthmoving automation is the force discontinuity that occurs when the bucket contacts the soil and transitions from free motion to motion that is force-constrained. With soil-tool interaction forces being non-negligible and difficult to model, several researchers have turned to active compliance control, which is comprised of two main categories: *hybrid position/force control* and *impedance control*. In hybrid position/force control, as in [17], the task space is divided into non-conflicting position and force controlled subspaces, with the undesirable consequence of having to switch control laws at the soil surface. In contrast, impedance control, as presented in Hogan's influential series of papers [18], focuses on shaping the dynamics between the environmental contact forces and endpoint position of the manipulator. Impedance control is generally favored for excavation systems over hybrid position/force approaches, as it provides a unified control law. Impedance control has been applied both in fully autonomous systems [19], and as a means of giving transparency to teleoperated systems, by matching impedances at the machine and at the remote control interface [20, 21].

1.3.4 Shared Control

The study of man-machine interaction saw much attention in the mid-to-late 20th century, particularly with regard to aviation and space exploration systems, driven by the need for high performance in hazardous, remote, and/or complex environments. Study of these man-machine systems has attracted a unique array of disciplines, synthesizing engineering and machine design with human physiology and psychology. The books by Sheridan, *et al.*, illustrate this multi-faceted approach and provide elaborate taxonomies on human behavior with respect to man-machine systems [22, 23].

Over time, distinct Shared Control (SC) architectures emerged in the man-machine system category, including tele-robotics and human supervisory control. These subtopics exhibit many of the same challenges, but suffer from a very broad and disconnected research base, as well as a lack of formal design or analysis techniques, as acknowledged by the IEEE Technical Comittee on Shared Control [24, 25]. As a result, design and evaluation is still largely in the hands of the individual designer, and the metrics used are typically very specific to the application domain.

Despite a deficiency of formal design methods, proper selection of a shared control architecture stems from an explicit understanding of the traits of both the HA and CA, as listed in Table 1.1. Although these assumed traits are broad and may have exceptions, they are generally self-evident, and provide a framework for discussion. Most SC scenarios seek to synthesize the advantages of both agents in an architecture that is germane to the task at hand.

Human	Computer
robust	fast
adaptive	reliable
safety conscious	precise
able to reason	inexhaustible

Table 1.1: Traits of human and computer agents

1.3.5 Types and Examples of Shared Control

Perhaps the simplest SC architecture is known as *traded control*. Here, control of the system is literally traded between the human and computer, as in aircraft autopilot systems. In critical scenarios such as manned flight, traded automation strategies suffer from the human *out-of-the-loop (OOTL)* problem. Since humans are typically only required to resume manual control under abnormal circumstances where the automation has failed, numerous studies have shown that operators resuming active control are much less aware of system state and less capable of fault-management [26, 27, 28, 29, 30]. Mitigating the OOTL performance problem is a central objective of human-centric automation, and provides strong motivation for a more smooth transition of control authority, when such a transition is necessary.

Another common form of SC is *collaborative control*, wherein certain functions of the machine are human operated while the remaining functions are automatically controlled. For instance, automotive cruise control regulates the speed of the vehicle, while the driver maintains control of the steering angle.

Many other forms of SC have been assessed in excavation systems, both theoretically and experimentally. *Coordinated control*, which consists of mapping operator inputs in a general three-dimensional workspace to the end-effector position in the manipulator workspace, has been explored by many authors [9, 31, 32, 33]. Coordinated control offers a more natural command interface for the operator, at the expense of non-standard and typically more costly control interfaces and mapping complexities.

Virtual constraint SC policies have been most successful in industry adoption for hydraulic excavation systems, as evidenced by Komatsu's recently launched Intelligent Machine Control (IMC) system, which can help to fix digging grades (angles), square the bucket position with the digging surface, and even stop the bucket from overdigging. The IMC outfitted Komatsu PC210LCi-10 (Komatsu America Corp.) utilizes stroke-sensing cylinders, an IMU, and GNSS data to sense the machine pose and environment [34]. These SC methods are summarized in Table 1.2.

Recently, in robotics and automation, promising developments have been made

Type	Application	Description
Traded control	Aircraft autopilot	Pilot may turn autopilot on/off
Collaborative control	Automotive cruise control	Human steers, computer maintains vehicle speed
Coordinated control	DaVinci robot	Maps operator joystick input to scaled manipulator workspace
Virtual constraint	DaVinci robot	Surgeon cannot leave bounded operating environment

Table 1.2: Types and examples of shared control

in *continuously shared control (CSC)*, where the operator input is continuously combined with automated assistance. One attractive example of CSC is a potential field approach (PFA) which incorporates the operator input as an additional vector in a virtual potential field, which together guide the motion of the robot. This PFA-based approach has been implemented in several domains, but defining such a field could be very complex in certain scenarios [35, 36].

In the earthmoving domain, Enes presented a *Blended Shared Control (BSC)* strategy, which combined the operator command with a time-optimal control perturbation by means of a variable blending parameter [37]. The BSC architecture is particularly well suited to the highly dynamic earth-moving environment. In Enes' presentation and other robotics implementations similar to BSC such as [38, 39], the control scheme can be broadly divided into three distinct steps: *task identification*, *task optimization*, and *control blending*.

Traditionally, the task identification step utilizes so-called motion primitives, mapping the input space to motion classes with corresponding actuator endpoints. The predicted motion classes and endpoints are then used in a real-time optimization to derive an optimal control input, denoted u^*. The final blending step is achieved by means of the following blended control law:

$$u = \bar{u} + \alpha(u^* - \bar{u}) \tag{1.1}$$

where $\bar{u}$ is the human operator command and α is a variable blending parameter such that

$$\alpha \in [0, 1] \tag{1.2}$$

The effect of the blending parameter, α, is a continuous control over the level of authority granted to the human agent and the optimal controller. Enes acknowledged that varying this blending parameter could offer better situational performance in specific tasks or environments. However, determination of this blending parameter is the subject of ongoing investigations.

More recently, Dragan and Srinivasa presented a holistic policy-blending formalism for shared control, characterizing the fundamental trade-offs present in BSC [40]. The authors coined the terms *prediction* and *arbitration* for the task identification and blending steps, respectively. Further, the authors suggested that the blending parameter increase with the prediction *confidence*. They proposed many mathematical definitions of confidence, including that confidence be a measure of the probability of the operator's goal being a certain state, or the entropy of the goal probability distribution. In the latter case, a higher entropy would result in a lower confidence, and minimal or no assistive control action. In either case, this confidence value can be directly mapped to the alpha parameter, and the properties of this mapping define the arbitration as either *aggressive* or *timid*. A more aggressive mapping risks incorrect control assistance, whereas a more timid mapping may provide too little assistance.

1.3.6 Learning from Demonstration

Designing high-level logic and trajectories for autonomous execution of multi-modal tasks would typically be accomplished by a robotics engineer, perhaps in coordination with domain experts. However, this design model is poorly scalable and suffers from the lack of a standardized description of the task.

Crucial to providing automated task assistance is the capability to store a symbolic representation of the task. Historically, this is the job of the robot programmer. However, modern robots like Baxter indicate the desire for a shift from *experts programming behaviors* to *non-experts demonstrating behaviors*. The capability to learn from demonstration makes an autonomous system more agile and versatile.

The problem of Learning from Demonstration (LfD) is typically posed in one of two ways: given some demonstration set, 1) learn the underlying *policy*, which is the relationship between the observed states and demonstrated actions, or 2) learn the underlying latent *reward function*, which can then be used to derive policies for executing the task.

Solving the first problem is more direct, but seeks only to reproduce operator behaviors and therefore suffers with imperfect or suboptimal demonstration data. Methods in the second category seek a high-level task description, with one notable class of algorithms denoted inverse reinforcement learning (IRL). IRL was first posed by Ng and Russell, and has since been adapted to suit many real-world learning problems [41, 42].

Implementing solutions to the IRL problem has many practical difficulties, which include the following:

- Original IRL methods seek to fit a *single* reward function to the entire state space. The designer must select some class of candidate reward functions.

- Closed-form solutions suffer the curse of dimensionality, and can become intractable with large state spaces.

- Other existing techniques for discovering reward functions in continuous state spaces begin with discretization and do not scale well.

1.3.7 Summary

Modern hydraulic excavator systems have complex nonlinear dynamics. Although the primary goal of this work is to develop high-level control algorithms, these algorithms must be implemented with a sufficient understanding of the underlying low-level control issues. These include nonlinearity, flow and kinematic coupling, valve deadbands, and force discontinuity at the soil interface. These dynamic characteristics have been addressed in the past with an array of different control methods, including gain-scheduling, feedback linearization, robust control, and force control techniques.

With regard to high-level control and autonomy, some trajectory generation methods have seen success in experimentation, but full autonomy remains too complex and fragile. For these reasons, the excavation industry has been reluctant to adopt autonomous excavation.

This provides some evidence that, at least for the foreseeable future, removal of the human operator may not be feasible. Instead, shared control research suggests that performance benefits are attainable by utilizing automatic control in conjunction with a human agent in the control loop. In particular, continuously shared control could optimize excavation task performance while avoiding the out-of-the-loop performance problem. However, there is still a need to derive meaningful representations of earth-moving tasks and to develop rules for smoothly transferring authority between the human and the computer.

1.3.8 Contributions

This thesis presents a design framework for shared control architectures for hydraulic excavation tasks. To begin with, a compact representation of the task must be formed and stored in a data structure. To that end, this work expands on the notion of a task being decomposed into task subgoals. Rather than subgoals being distinct points in the workspace of the excavator, this work proposes each subgoal as being drawn from a mixture of subgoal distributions, such that the uncertainty of points of interest can be modeled explicitly.

This research emphasizes learning from demonstration techniques which require little specification of subgoal structure, and are therefore more scalable to different manipulation environments and task complexities. Two existing inverse reinforcement learning frameworks are adapted and implemented on real world excavator operation data. A new learning technique is introduced which identifies *change-points*, or locations where the operator redirects motion, as a basis for modeling subgoal distributions.

To utilize these task representations, the intention of the human operator must be identified within this subgoal decomposition. This work presents three novel real-time subgoal prediction schemes. The first uses strict deterministic logic to toggle control assistance, and requires the control system designer to encode some aspects of how the task should be performed. The second and third methods are probabilistic, using Bayes theorem to compute the posterior probability of the current subgoal.

A reduced-scale (1/12th) hydraulic excavator model has been instrumented and evaluated as an experimental platform for shared control research. The challenges of conducting excavation experiments with the reduced-scale model as well in scaling up the results are discussed. Experiments have been conducted with operators of varying skill, showing an increase in digging efficiency (mass of material moved per cycle time) and pure cycle time.

The thesis also delineates how probabilistic prediction methods could be incorporated into a dynamic blending framework, where the amount of control intervention is directly determined by the confidence of the predicted subgoal.

2. KINEMATIC AND DYNAMIC MODELING

This section presents a more detailed discussion of the system architecture of a hydraulic excavator, along with the development of kinematic and dynamic models for the 1/12th-scale hydraulic experimental platform. Critical differences between full-scale excavator hydraulic systems and the reduced-scale excavator which was used for this thesis will be indicated and discussed in context.

2.1 Nomenclature

2.1.1 Hydraulic Characteristics

β	bulk modulus of hydraulic fluid
ΔP_{ij}	pressure differential across orifice, $P_i - P_j$
ρ	fluid density
$A_{p,A}$	actuator cap-end (A) pressurized area
C_d	orifice discharge coefficient
P_R	return (tank) pressure
P_S	supply (pump) pressure
P_A	actuator cap-end (A) pressure
P_B	actuator rod-end (B) pressure
Q_{ij}	flow across orifice, from fluid volume i to j
s	spool position

V_{A0}	actuator cap-end (A) dead volume

2.1.2 Kinematics

$\boldsymbol{o}_i^{i-1}$	coordinates of origin of frame $\{o_i\}$ with respect to frame $\{o_{i-1}\}$
$\boldsymbol{p}_{m,n}^i$	vector from point m to point n in frame $\{o_i\}$
ζ_i	total hydraulic cylinder length
$\{o_i\}$	the i^{th} coordinate frame
R_i^j	rotation matrix from frame $\{o_j\}$ to frame $\{o_i\}$

2.1.3 Actuator Mechanics

b	viscous friction component in actuator
F_d	unknown disturbance forces
x	hydraulic rod displacement

2.1.4 Other Symbols

$\_{A}$	cap-end variable
$\_{B}$	rod-end variable
$\_{R}$	return (tank) variable
$\_{S}$	supply pump variable

2.2 Kinematics

The base of the excavator is considered fixed in this model, which is an appropriate assumption for many earth-moving tasks. The forward kinematics relating the remaining four joint angles to the end-effector position were developed using the Denavit-Hartenberg (DH) convention [43], with the selected joint frames shown in Fig. 2.1, resulting in the DH parameters shown in Table 2.1. The notation $\{o_0\}$ represents the global frame, which is aligned with the swing motor, positioned at ground height for convenience. Frames $\{o_1\}$, $\{o_2\}$, and $\{o_3\}$ are attached to the boom, stick, and bucket joints, respectively, and frame $\{o_4\}$ denotes the end-effector frame at the tip of the bucket.

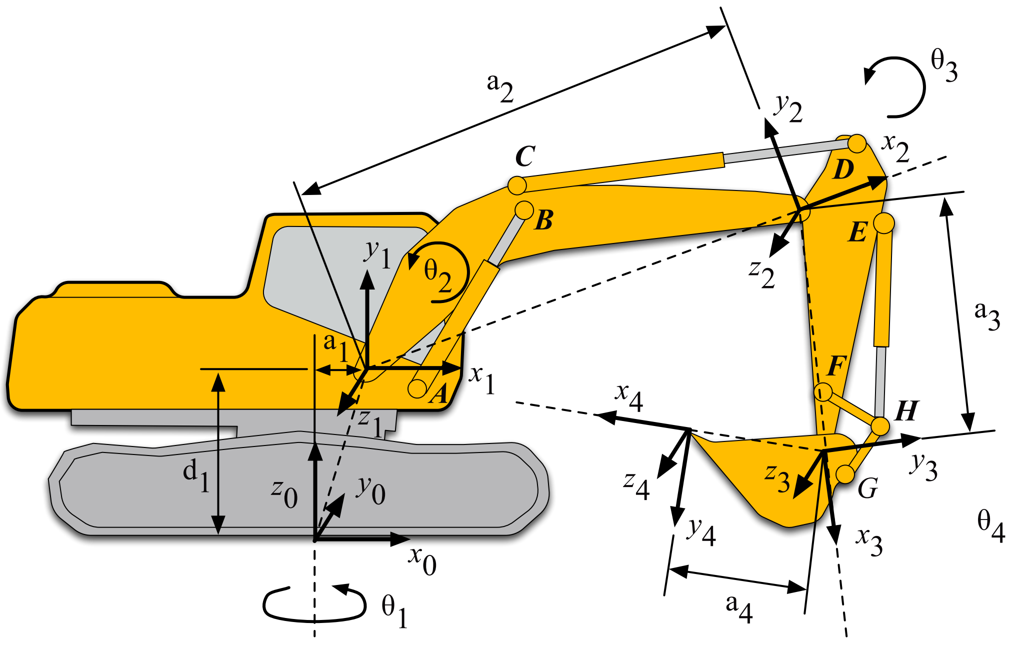


Figure 2.1: Denavit-Hartenberg frame selection and parameters, and points $A - H$

The vector $\boldsymbol{o}_i^{i-1}$ denotes the coordinates of the origin of the frame $\{o_i\}$ with respect to the frame $\{o_{i-1}\}$. A generic position vector is defined using the convention $\boldsymbol{p}^i_{m,n}$, indicating the vector from point m to point n expressed in frame $\{o_i\}$. The angle θ_i represents the i^{th} joint angle, or more precisely, the angle between x_{i-1} and x_i. The normal distance between joint axes z_i and z_{i-1} is a_i. The homogeneous transformation from frame $i-1$ to i can be represented by

$$T^i_{i-1} = \begin{bmatrix} R^i_{i-1} & \boldsymbol{o}^{i-1}_i \\ \mathbf{0}^{(1\times 3)} & 1 \end{bmatrix} \tag{2.1}$$

Where $R^i_{i-1} \in SO(3)$ is the rotation matrix from frame $\{o_{i-1}\}$ to the frame $\{o_i\}$.

The homogeneous transformation relating a coordinate vector to a point in the end-effector frame to the coordinate vector of the same point in the base frame is found by composing successive transformations T^4_3, T^3_2, T^2_1, and T^1_0. The resulting transformation T^0_4 is a function of the joint variables, $\boldsymbol{q} = [\theta_1, \theta_2, \theta_3, \theta_4]^T$, and is given by,

$$T^0_4 = T^0_1 T^1_2 T^2_3 T^3_4 = \begin{bmatrix} c_1 c_{234} & -c_1 s_{234} & s_1 & c_1(a_4 c_{234} + a_3 c_{23} + a_2 c_2 + a_1) \\ s_1 c_{234} & -s_1 s_{234} & -c_1 & s_1(a_4 c_{234} + a_3 c_{23} + a_2 c_2 + a_1) \\ s_{234} & c_{234} & 0 & (a_4 s_{234} + a_3 s_{23} + a_2 s_2 + a_1) + d_1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{2.2}$$

where c_i and s_i are $\cos(\theta_i)$ and $\sin(\theta_i)$, respectively, and c_{ijk} and s_{ijk} are $\cos(\theta_i + \theta_j + \theta_k)$ and $\sin(\theta_i + \theta_j + \theta_k)$. Here the DH parameter $\alpha_1 = \frac{\pi}{2}$ has been applied, but the remaining parameters are left in symbolic form. The DH parameters for an experimental 1/12th scale platform are given in Table 2.1.

The following equations relate the hydraulic cylinder lengths and the manipulator joint angles. The Euclidean distance between any two points m and n is represented as $r_{m,n}$ and the angle $\theta_{k,m,n}$ represents the angle between lines $\overline{km}$ and $\overline{mn}$. The

Link i	a_i	α_i	d_i	θ_i
1	3.7 cm	$\frac{\pi}{2}$	17.1 cm	θ_1
2	48.2 cm	0	0	θ_2
3	25.3 cm	0	0	θ_3
4	11.5 cm	0	0	θ_4

Table 2.1: Denavit-Hartenberg parameters for the 1/12 scale excavator model

length of the i^{th} hydraulic cylinder is represented by ζ_i.

$$\theta_2 = \cos^{-1}\left(\frac{r_{1,B}^2 + r_{1,A}^2 - \zeta_1^2}{2r_{1,B}r_{1,A}}\right) - \theta_{B,1,2} - \theta_{A,1,x_1}$$

$$\theta_3 = 3\pi - \cos^{-1}\left(\frac{r_{2,C}^2 + r_{2,D}^2 - \zeta_2^2}{2r_{2,C}r_{2,D}}\right) - \theta_{1,2,C} - \theta_{D,2,3} \tag{2.3}$$

$$\theta_4 = 3\pi - \theta_{F,3,H} - \theta_{H,3,G} - \theta_{G,3,4} - \theta_{2,3,D}$$

where

$$\theta_{F,3,H} = \cos^{-1}\left(\frac{r_{3,H}^2 + r_{3,F}^2 - r_{F,H}^2}{2r_{3,F}r_{3,H}}\right)$$

$$\theta_{H,3,G} = \cos^{-1}\left(\frac{r_{3,H}^2 + r_{3,G}^2 - r_{G,H}^2}{2r_{3,H}r_{3,G}}\right)$$

$$r_{3,H} = \sqrt{r_{3,F}^2 + r_{F,H}^2 - 2r_{3,F}r_{F,H}\cos(\theta_{H,F,3})} \tag{2.4}$$

$$\theta_{H,F,3} = \pi - \theta_{D,F,E} - \theta_{E,F,H}$$

$$\theta_{E,F,H} = \cos^{-1}\left(\frac{r_{E,F}^2 + r_{F,H}^2 - \zeta_3^2}{2r_{E,F}r_{F,H}}\right)$$

The transformations from joint angles $\boldsymbol{q}$ to the cylinder lengths ζ_i can be found by inverting the above relations.

The linear and angular velocities of the i^{th} link are computed using the following Jacobian,

$$J_i = \begin{bmatrix} J_{vi} \\ J_{\omega i} \end{bmatrix} = \begin{bmatrix} R_{i-1}^0 \hat{k} \times (\boldsymbol{p}_{0,4}^0 - \boldsymbol{p}_{0,i-1}^0) \\ R_{i-1}^0 \hat{k} \end{bmatrix} \tag{2.5}$$

where J_{v1} ,$J_{\omega i}$ are 3×1 vectors describing the linear and angular velocity.

2.3 Hydraulic Actuation

Although multiple stages of valves may be used in large hydraulic systems, this model considers direct control of the main valves for generality. Two main types of proportional hydraulic valves are considered in this thesis: the spool-type valve which is ubiquitous in full-scale hydraulic excavators and a rotary-type valve which is employed by the scale experimental model.

A spool valve is fundamentally classified by the number of ways, or fluid paths leading in and out of the valve. Four ways are required to actuate a double-acting cylinder, and these are denoted cap-end (A), rod-end (B), supply (S), and return (R), illustrated in Fig. 2.2.

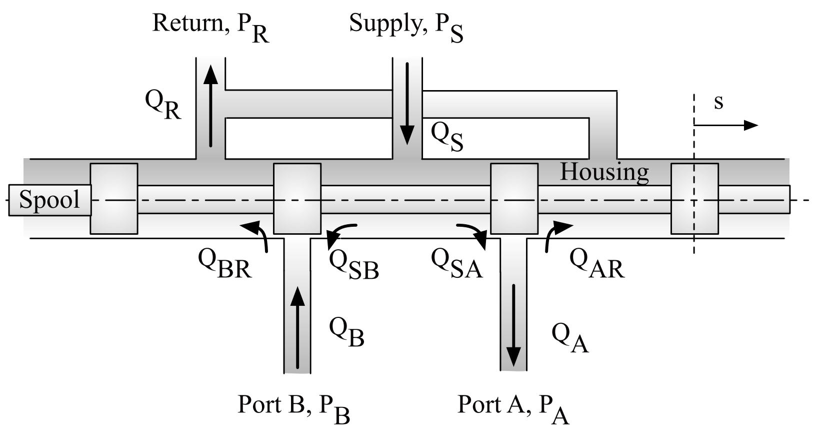


Figure 2.2: A 4/3 spool valve

In this depiction a positive displacement s will cause flow from the supply line (pump) to the cap-end (A) of the actuator, and flow from the rod-end (B) of the actuator to the return line (tank). However, due to the closed-center configuration, the displacement must exceed the valve deadband s_o for fluid to flow, shown in Fig. 2.3.

Rather than a spool valve, the scale experimental platform employs a rotary valve arrangement. The rotary valve exchanges the linear actuation of a spool with a rotating servomotor. The principle of operation of the rotary valve block is illustrated in Fig. 2.4. When the servomotors rotate the valve past the deadband, half-moon shaped ways simultaneously connect the pump to the actuated end of the cylinder

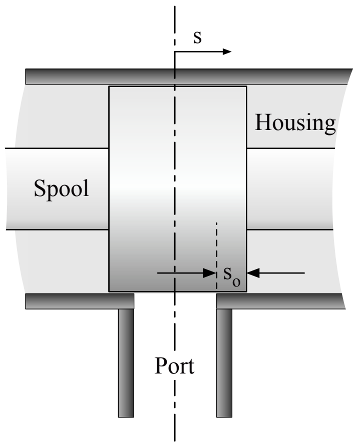


Figure 2.3: The closed-center (overlapped) valve configuration

and the tank to the opposing end of the cylinder. Similar to the discussed spool valve configuration, the rotary valve is also closed-center and experiences deadband around the neutral position.

2.3.1 Pressure-Flow Relationship

Flow through an orifice can be modeled by the classical orifice flow relation, given by

$$Q = AC_d\sqrt{\frac{2}{\rho}|\Delta P_{ij}|}\ sgn(\Delta P_{ij}) \tag{2.6}$$

where viscous effects are captured by the empirical discharge coefficient, C_d, and fluid momentum and pressure are modeled explicitly. The term ΔP_{ij} is the pressure differential across the orifice, or $P_i - P_j$. The valve discharge area (a function of spool displacement) is given by A, and fluid density by ρ. The signum function and absolute value are present to correct for positive flow direction. This equation is nonlinear due to the square-root pressure term and a nonlinear discharge area function.

The orifice area is a nonlinear function of the valve displacement and depends

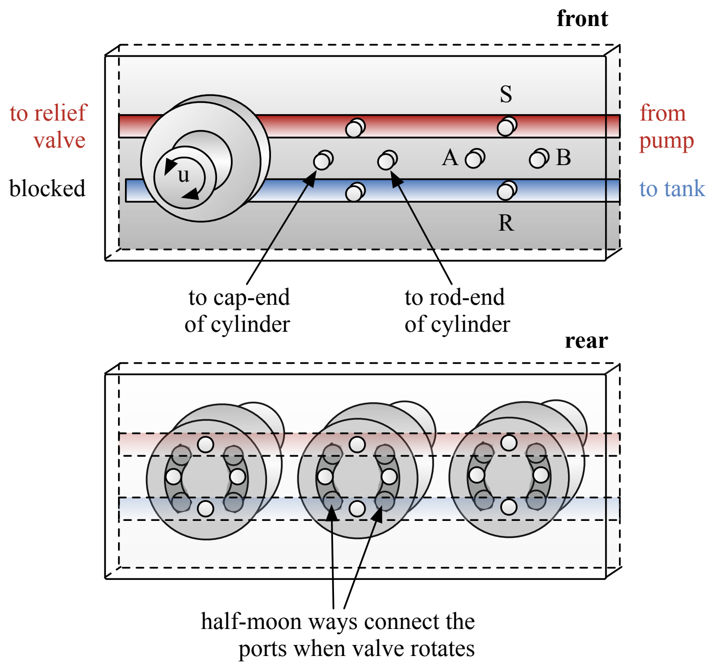


Figure 2.4: Front and rear views of the rotary valve manifold on the experimental model; *two valves are hidden on the front view to illustrate the port arrangement on the manifold*

heavily on valve port geometry. The area is frequently approximated by a constant gain on the valve position, which is often adequate. For spool valves the orifice area is a partially obstructed circle, and for a rotary valve, the area is formed by intersecting circular areas. The difference in valve gains are shown in Fig. 2.5.

2.3.2 Flow Continuity

The mass of the fluid in the cap-end (A) of the cylinder is described by

$$m_A = \rho V_A \tag{2.7}$$

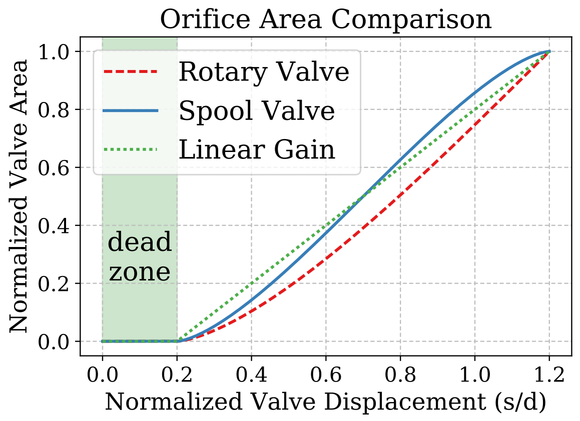


Figure 2.5: Orifice area versus valve displacement

Taking the derivative with respect to time, we arrive at

$$\dot{m}_A = \rho \dot{V}_A + \dot{\rho} V_A \tag{2.8}$$

Fluid bulk modulus, β, describes the compressibility of fluid under constant temperature by the following relation:

$$\beta = \rho \left(\frac{\partial P}{\partial \rho} \right)_T \tag{2.9}$$

$$\approx \rho \frac{\dot{P}}{\dot{\rho}} \tag{2.10}$$

The mass balance of flow across valve orifices gives the following equality:

$$\dot{m}_A = \rho Q_A \tag{2.11}$$

$$Q_A = Q_{SA} - Q_{AR} \tag{2.12}$$

where leakage across the hydraulic piston (i.e., Q_{AB}) is assumed to be negligible. The rod displacement, x, and velocity, $\dot{x}$, are related to the cylinder volume by

$$V_A = A_{p,A}x + V_{A0} \tag{2.13}$$

$$\dot{V}_A = A_{p,A}\dot{x} \tag{2.14}$$

Where $A_{P,A}$ is the cap-end (A) pressurized area on the piston, and $A_{P,A}$ is the rod-end (B) pressurized area. The parameter V_{A0} is the dead (or inactive) volume in the cap-end of the cylinder. Synthesizing Eqs. (2.8)-(2.14), we have the following relation:

$$\dot{P}_A = \frac{\beta}{A_{p,A}x + V_{A0}}(Q_{SA} - Q_{AR} - A_{p,A}\dot{x}) \tag{2.15}$$

This relation can be extended to the rod-end (B), yielding

$$\dot{P}_B = \frac{\beta}{V_{B0} + A_{p,B}(x_{max} - x)}(Q_{SB} - Q_{BR} + A_{p,B}\dot{x}) \tag{2.16}$$

where x_{max} is the full stroke of the piston.

2.3.3 Actuator Mechanics

By balancing the fluid pressure forces and load forces, the net force on the rod is given as:

$$m_{rod}\ddot{x} = P_A A_{p,A} - P_B A_{p,B} - f(x, \dot{x}) - b\dot{x} \tag{2.17}$$

where f is the external load, and $b\dot{x}$ is viscous friction in the actuator.

2.4 Manipulator Dynamics

Since we are modeling the excavator as a four revolute joint mechanism, the dynamics can be expressed in the standard robotics form as

$$D(\boldsymbol{q})\ddot{\boldsymbol{q}} + C(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + G(\boldsymbol{q}) + B(\dot{\boldsymbol{q}}) = T - F_e \tag{2.18}$$

where $\boldsymbol{q} = [\theta_1, \theta_2, \theta_3, \theta_4]^T$, D is the mass and inertia matrix, C is the Coriolis matrix, G and B are gravity and friction vector, respectively, and T and F_e are the input joint torque and external force vectors, respectively. The external forces are a combination of the forces due to the mass of the soil in the bucket while traveling, digging forces during digging operation and any other external effects. The mass and inertia matrix is computed as

$$D = \sum_i \{m_i J_{vi}(\boldsymbol{q})^T J_{vi}(\boldsymbol{q}) + J_{\omega i}(\boldsymbol{q})^T R_i(\boldsymbol{q}) I_i R_i(\boldsymbol{q})^T J_{\omega i}(\boldsymbol{q})^T\} \tag{2.19}$$

where I_i is the inertia tensor of the i-th link. If the elements of the inertia matrix are denoted by d then, the terms of Coriolis matrix are computed using

$$c_{jk} = \sum_{i=1}^{n} \frac{1}{2}\left(\frac{\partial d_{kj}}{\partial q_i} + \frac{\partial d_{ki}}{\partial q_j} - \frac{\partial d_{ij}}{\partial q_k}\right)\dot{q}_i. \tag{2.20}$$

The gravity vector for the current configuration can be computed using

$$G = \begin{bmatrix} 0 \\ [g(m_2 l_{c2} + m_3 l_3 + m_4 l_4)c_2 + g(m_3 l_{c3} + m_4 l_4)c_{23} + g m_4 l_{c4} c_{234}] \\ g(m_3 l_{c3} + m_4 l_4)c_{23} + g m_4 l_{c4} c_{234} \\ g m_4 l_{c4} c_{234} \end{bmatrix} \tag{2.21}$$

where the distance between the link joint to its center of mass is denoted l_{ci} and the height of the link center of gravity above the ground is denoted by l_i.

For simplicity, friction in the link joints is assumed to be negligible. In excavators, the joint torques are a result of hydraulic cylinder forces, which are described by the dynamics of the hydraulic actuator. The cylinder forces f_i, each described by Eq. (2.17), are converted to joint torques τ_i using the geometry of the excavator as follows,

$$T = \begin{bmatrix} \tau_1 \\ \tau_2 \\ \tau_3 \\ \tau_4 \end{bmatrix} = \begin{bmatrix} \tau_1 \\ (\boldsymbol{p}^2_{1B} \times \boldsymbol{f}^2_{1B})\zeta_1 \\ (\boldsymbol{p}^3_{2D} \times \boldsymbol{f}^3_{2D})\zeta_2 \\ (\boldsymbol{p}^4_{3G} \times \boldsymbol{f}^4_{3G})\zeta_3 \end{bmatrix} \tag{2.22}$$

where τ_1 is the swing motor torque and f_{1B}, f_{2D} and f_{3G} represent the hydraulic cylinder forces acting on cylinder and link contact points B, D and G in the link coordinate frames, i.e., $\{o_1\}$, $\{o_2\}$ and $\{o_3\}$.

The forward and inverse kinematics are used frequently throughout this work for visualizing and identifying locations in the 3-dimensional (Cartesian) workspace in analysis. The low-level actuator control is conducted using reference trajectories defined in the actuator space, eliminating the need for costly inverse kinematics at run-time. Some dynamic characteristics such as deadband and flow coupling are identified on the scale model and addressed in Chapter 4.

3. TASK LEARNING AND SUBGOAL PREDICTION

This chapter addresses the identification and prediction of operator subgoals and is divided into two components: Sec. 3.1 *Task Learning*, corresponds to offline inference of latent subgoal locations in complete demonstration datasets, and Sec. 3.2 *Prediction*, which utilizes the learned subgoal structure from the task learning process to predict the operator's current subgoal on-line.

These problems are addressed separately for two main reasons: 1) most learning from demonstration (LfD) routines rely on computationally expensive iterative Bayesian algorithms, and 2) we desire the learning of a task to be non-parametric, i.e., the number of task segments or waypoints is not specified directly in the learning process, but rather is inherent in the dataset. However, both sections share a common need: a model for the likelihood of operator action conditioned on a specific subgoal, which is called the *action likelihood*.

Note that all of the methods of this chapter will be evaluated against a simple benchmark task: loading a truck from a pile of dirt. Briefly, this task consists of scooping material from a pile and dumping the material into the truck bed by uncurling the bucket. All operator data visualizations will use the same 3-dimensional viewpoint, where the pile is on the left, the excavator swing axis is at the origin of the frame, and the truck is on the right, as in Fig. 3.2.

3.1 Task Learning

As outlined in Chapter 1, inverse reinforcement learning (IRL) is the process of seeking a latent reward function which explains operator behavior, and was first posed by Ng and Russell [41]. The current reinforcement learning and inverse reinforcement learning solution frameworks both assume that the underlying process dynamics are governed by a Markov Decision Process (MDP). Some introduction of MDP preliminaries is presented next, followed by some recent IRL algorithms applicable to real-world control tasks.

3.1.1 Markov Decision Processes

A (finite) MDP is a tuple $(\mathcal{S}, \mathcal{A}, T, \gamma, R)$, where

- $\mathcal{S}$ is a set of N states
- $\mathcal{A} = \{a_1, ..., a_k\}$ is a set of k actions
- $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S}$ is the **transition probability**, where $T(s_1, a_1, s_2)$ is the probability of being in state s_2 after taking action a_1 in state s_1
- $\gamma \in [0, 1)$ is a **discount factor**
- $R : \mathcal{S} \to \mathbb{R}$ is the reward or reinforcement function

A *policy*, π is defined formally as a mapping

$$\pi : \mathcal{S} \to \mathcal{A}$$

and the cumulative value function for a policy π, evaluated at state s_1 is given by

$$V^\pi(s_1) = E[R(s_1) + \gamma R(s_2) + \gamma^2 R(s_3) + ...|\pi]$$

In addition the Q-function or action-value function is defined as

$$Q^\pi(s, a) = R(s) + \gamma E_{s'\ T}[V^\pi(s')] \tag{3.1}$$

3.1.2 Reinforcement Learning and Inverse Reinforcement Learning

Reinforcement learning is the process of seeking an optimal policy, π^* such that the cumulative value function $V^{\pi^*}(s)$ is maximized for all $s \in S$. This corresponds to seeking the policy for which the action-value function is maximized, given by

$$\pi^*(s) = \arg\max_a Q^\pi(s, a, R) \tag{3.2}$$

The optimal policy is most often computed in discrete state spaces using the *value iteration* algorithm, which is based on dynamic programming, or the *Q-learning* algorithm in cases where the model for rewards or state transitions is not known [44]. In contrast, IRL seeks to discover a latent reward function $R(s)$ given MDP$/R$, which is an MDP with all parameters specified except the reward function R, and some set of observations

$$\mathcal{O} = \{(s_1, a_1), (s_2, a_2), ..., (s_N, a_N)\}$$

where (s_i, a_i) is the i^{th} observed state-action pair from a length N demonstration.

3.1.3 Bayesian Nonparametric IRL

In 2015, Michini presented a Bayesian nonparametric approach to IRL (BNIRL) which partitioned the candidate reward function into several reward functions [45, 42]. The author suggested each partition could utilize a simple reward function, $R_g(s)$, which consists of a constant reward, c, at a single coordinate g, which could be considered a *subgoal* of the operator

$$R_g(s) = \begin{cases} c & s = g \\ 0 & s \neq g \end{cases} \tag{3.3}$$

The candidate subgoal set $\mathcal{G}$ is simply all of the states in the demonstration set; *i.e.*, it is assumed that the demonstrator reaches all of his/her subgoals at some point in the demonstration, although not necessarily in an optimal fashion. The BNIRL algorithm utilizes a Bayesian nonparametric generative model for the subgoal partition assignment of the form

$$\underbrace{P(z_i|z_{-i}, \mathcal{O})}_{\text{assignment posterior}} \propto \underbrace{P(z_i|z_{-i})}_{CRP} \underbrace{P(O_i|R_i)}_{\text{action likelihood}} \tag{3.4}$$

where z_i is the partition label (subgoal label) for state s_i, and the assignment prior is a *Chinese restaurant process (CRP)*, according to

$$P(z_i|z_{-i}) = \begin{cases} \frac{\sum(z_{-i}=j)}{N-1+\eta} & \text{If partition } j \text{ already exists} \\ \frac{\eta}{N-1+\eta} & \text{If partition } j \text{ is new} \end{cases} \tag{3.5}$$

with hyperparameter η controlling the tendency to form new partitions. The CRP construction is used to generate samples from a Dirichlet process, for which a more thorough introduction is provided in [46].

The *action likelihood* is formed by an exponential rationality model

$$P(O_i|R_{z_i}) = P(a_i|s_i, z_i) \propto e^{\beta Q^*(s_i, a_i, R_{z_i})} \tag{3.6}$$

where the optimal value function must be solved to compute Q^*, and β is a scaling parameter[*]. One major contribution of the BNIRL approach was to approximate Q^* with a *closed-loop action comparison*, such that

$$P(O_i|R_{z_i}) = P(a_i|s_i, z_i) \propto e^{-\beta\|a_i - a_{CL}\|_2} \tag{3.7}$$

where a_{CL} is the action of a proportional feedback controller whose setpoint is the location of the z_i^{th} subgoal g_{zi}. In words, given some state s_i and subgoal partition z_i, the most likely actions are those that approach the action of a closed-loop controller a_{CL}. In BNIRL, Gibbs sampling over (3.4) is used to form the joint posterior, which involves iteratively sampling the conditional distribution over partition assignments to form a posterior joint distribution. In a video demonstration, Michini moves a quadcopter manually through a series of waypoints in a plane, after which the algorithm effectively converges on the subgoals of the demonstration.

In seeking a compact representation of earthmoving tasks from demonstration data, the BNIRL algorithm was implemented and executed on manual operation data. However, the magnitude of the closed-loop action was difficult to calibrate to various operator styles and actuator saturation ranges. Therefore, rather than using the closed-loop action comparison, a less strict action comparison is formulated as the following

$$P(a_i|s_i, z_i) \propto e^{\beta[\cos(\theta)-1]} \tag{3.8}$$

where θ is the angle subtended by the action vector, a_i and the vector from the current state to the subgoal, p_{s_i,g_zi}, the cosine of which is equivalent to

$$\cos(\theta) = \frac{p_{s_i,z_i} \cdot a_i}{\|p_{s_i,z_i}\| \; \|a_i\|} \tag{3.9}$$

[*]the original publication used α, but this conflicts with our blending parameter

The resulting action likelihood over all angles, θ, and multiple scaling parameters, β, is shown in Fig. 3.1.

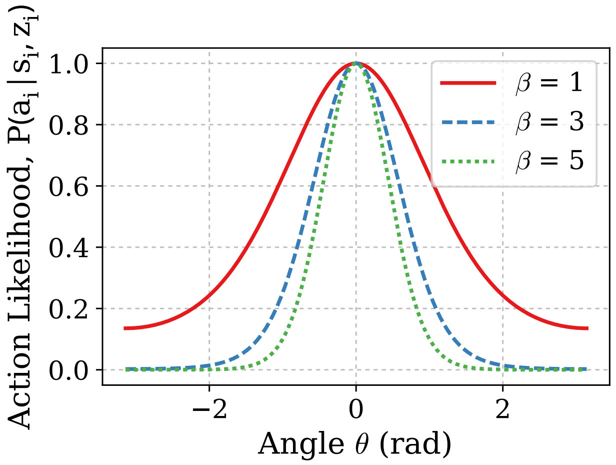


Figure 3.1: Action likelihood over all angles θ and three β parameters

Using this modified action likelihood, the BNIRL sampling process is parameterized by only the CRP concentration parameter, η, and the action comparison scaling parameter, β. The BNIRL Gibbs sampling process is described in Algorithm 1.

After some preliminary testing with the BNIRL algorithm, a parameter grid was constructed with several values of η and β to assess the parametric sensitivity of the BNIRL approach on excavator operation data. The training dataset is shown as a 3-dimensional quiver plot in Fig. 3.2. The BNIRL algorithm was executed on state observations in the 4-dimensional actuator space, however the forward kinematics are used henceforth for the sake of visualizing the results in 3-dimensional space as the position of the end-effector. The action input sequence was the computed velocity of each actuator. Although in this scenario we also have direct access to the joystick inputs of the operator, these inputs would need to be scaled appropriately before

Algorithm 1 Gibbs sampling for BNIRL [42]

```
1: function GIBBS-SAMPLER($\mathcal{O}, \eta, \alpha, K$)
2:     for each Gibbs sampling sweep $k < K$ do
3:         for each observation $O_i = (s_i, a_i) \in \mathcal{O}$ do
4:             for each current subgoal partition $j^k$ do
5:                 $p(z_i = j|z_{-i}, O_i) \leftarrow p(z_i = j|z_{-i})p(a_i|s_i, z_i)$    ▷ Probability of assignment to subgoal $j$ from Eq.
6:             end for
7:             $p(z_i = k|z_{-i}, O_i) \leftarrow p(z_i = j|z_{-i})p(a_i|s_i, z_i)$    ▷ Probability of a new randomly drawn subgoal
8:             $z_i^k \sim P(z_i|z_{-i}, O_i)$    ▷ Sample partition assignment from normalized probabilities in lines 5 and 7
9:         end for
10:    end for
11:    return assignment vectors $\boldsymbol{z}^{1:K}$ for each iteration
12: end function
```

using the action comparison likelihood. For instance, since we are concerned with the direction of the action in the actuator space, each joystick input would need to be scaled to the magnitude of velocity that it produces. Each Gibbs sampling process was run for 400 iterations, and the first 40 samples were discarded for burn-in. The posterior partition assignment modes (subgoals) for selected pairs of parameters are illustrated in Fig. 3.3, where all observed states are illustrated as colored circles and the resulting subgoal configurations are shown as wireframe excavator arms. A summary of the number of unique posterior modes (i.e., number of subgoals) for each parameter set is shown in Fig 3.4.

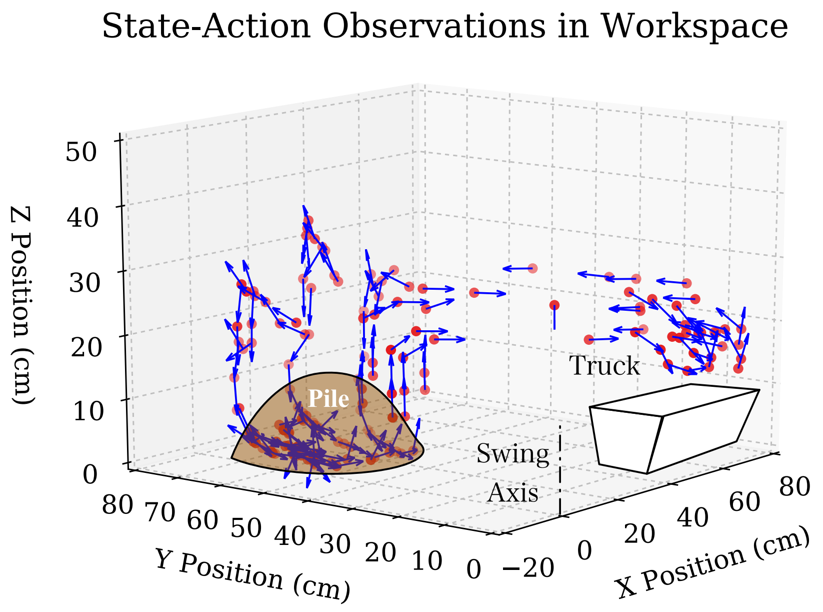


Figure 3.2: Observations in training data; states are circles, and actions are represented as vectors from that state

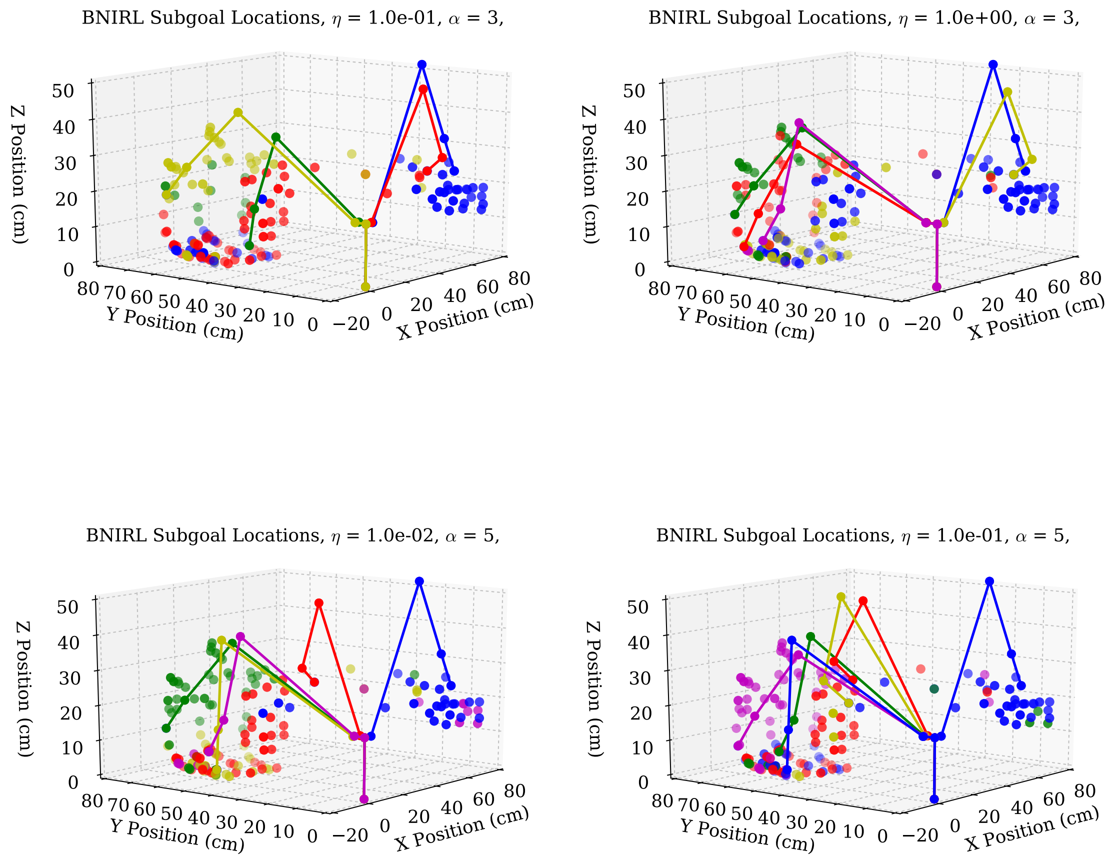


Figure 3.3: Resulting subgoals from BNIRL algorithm with varying parameters

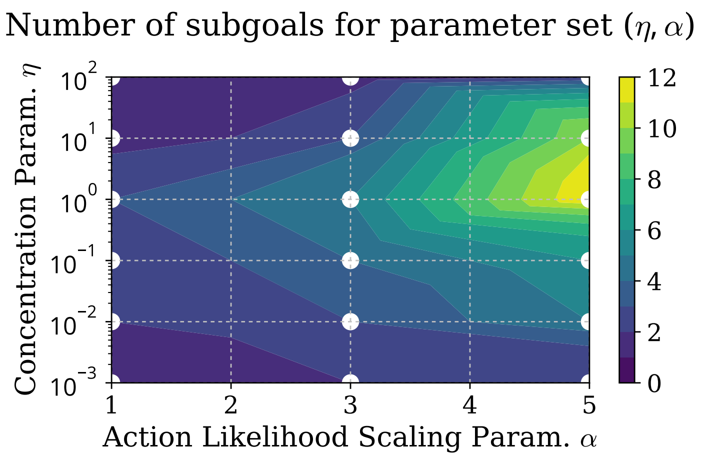


Figure 3.4: Posterior subgoal counts for a grid of parameter sets

The BNIRL generative model presupposes that the discovered subgoals are distinct states which must be selected from observation set. However, in practice operator subgoals corresponding to earthmoving manipulation tasks may actually be distributed over, for example, the volume of a pile of dirt. Therefore, reforming the representation of subgoals from a set of waypoints to a set of distributions of waypoints may offer a more general representation of the task.

3.1.4 Dirichlet Process Means IRL

Using the reward partitioning framework of BNIRL, Maske *et al.* presented DP-MIRL, which assumes a Dirichlet process Gaussian mixture model over subgoal locations [47]. By assuming that the operator also acts as closed-loop controller, the action likelihood reduces to a measure of proximity to the subgoal. The DP-means clustering algorithm was presented by Kulis and Jordan and extends the popular k-means clustering technique by exchanging the specification of the number of clusters k with a cluster penalty parameter, λ [48].

Maske also introduced the notion of *action primitives*, which merge the demonstration states into a more compact sequence of discrete motion classes. To form action primitives, the computed velocities of each actuator are clustered via k-means with $k = 3$, where the three clusters naturally correspond to positive, negative, and near zero velocity. A parameter η is introduced to limit the variance of the zero velocity cluster. For a system of m actuators, the action primitive at each sample is a m-tuple composed of the cluster labels for each actuator. For instance, by assigning arbitrary numerical labels for each actuator direction cluster

$$\mathcal{C} = \{1, 2, 3\}$$

then the set of actions may be written as

$$\mathcal{A} = \mathcal{C}^m$$

where the i^{th} action would be

$$a_i = (c_1, c_2, ..., c_m), c_j \in \mathcal{C}$$

and there are $|\mathcal{A}| = |\mathcal{C}|^m = 3^m$ possible action primitives.

The observations at the beginning of each new action primitive are merged to form a new observation set, $\mathcal{O}$. Next, the DPMIRL algorithm uses *a priori* knowledge of objects of interest in the workspace to pre-partition the data, by minimizing the Euclidean distance to an object of interest. Therefore, with k objects of interest, the algorithm returns a minimum of k subgoals. The DP-means algorithm is shown in Algorithm 2.

In implementing the DPMIRL algorithm on the excavator, we utilize a similar segmentation of observation data, defining 3 classes of motion for each actuator. Rather than clustering the actuator velocities, a simple piecewise labeling function is used: velocities within a certain threshold of zero are null actions, and the remaining positive and negative velocity time indexes are grouped into positive actions and negative actions, respectively. Each actuator motion class is arbitrarily assigned an integer label: 1, 2, and 3 for negative, neutral, and positive velocity, respectively. Labeled velocity data for each actuator of a manually operated trial on the excavator is shown in Fig. 3.5. Considering the four main actuators, we have $|\mathcal{A}| = |\mathcal{C}|^m = 3^4 = 81$ possible classes, although less than half of these classes occur in a typical trial.

Algorithm 2 DP-Means IRL [47]

```
1: function DP-MEANS-CLUSTERING($\mathcal{O}, \lambda, M, K$)
2:     for each object of interest $m_j \in M$ do
3:         for each state $s_i$ do                                ▷ pre-partition data
4:             if $j = \arg\min_k \|s_i - m_k\|$ then      ▷ if object $m_j$ is closest to state $s_i$
5:                 $z_i \leftarrow j$                              ▷ assign state $s_i$ to partition $j$
6:             end if
7:         end for
8:         for each iteration of DP-Means $k < K$ do        ▷ find subclusters at each object $m_j$
9:             run DP-Means for states in partition $j$ to obtain subclusters $C_j = \{c_{j1}, ..., c_{jk}\}$    ▷ See Ref. [48]
10:        end for
11:        for each state observation $s_i$ do
12:            assign $z_i$ a unique label for all subclusters $c_{jk}$
13:        end for
14:    end for
15:    return subcluster assignment vector $\boldsymbol{z}$
16: end function
```

Since it utilizes the Euclidean norm and locations of points of interest, DPMIRL is much better suited for clustering observations in the end-effector domain. The DPMIRL algorithm was run on the same training dataset as before, except transformed from the actuator space to the Cartesian end-effector space (XYZ). The DPMIRL algorithm implementation is shown in Algorithm 2. The results are illustrated in Fig. 3.6.

Although DPMIRL does produce a nice distance based clustering of states in the demonstration set, the algorithm is only effective in the end-effector space, and does not directly incorporate information in the operator's action.

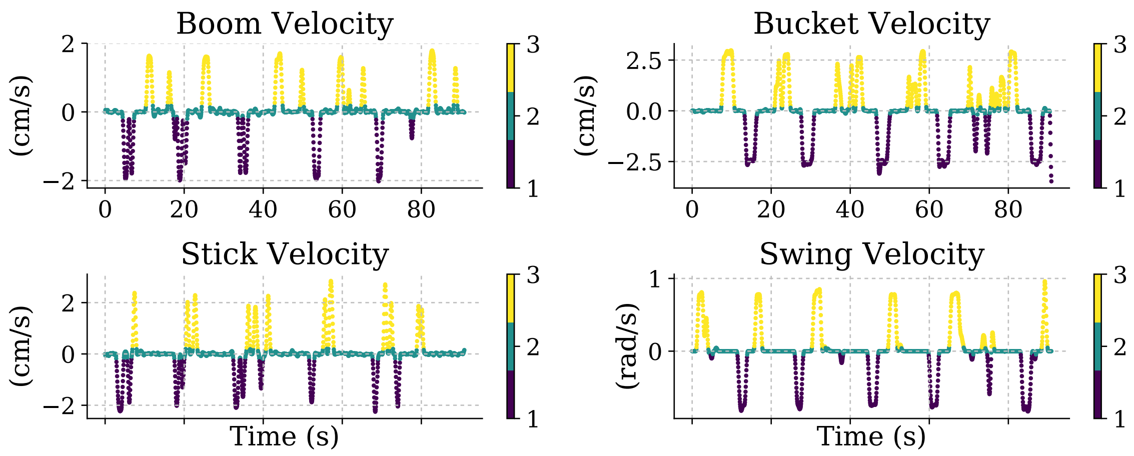


Figure 3.5: The motion classes on each actuator for a manual dig cycle

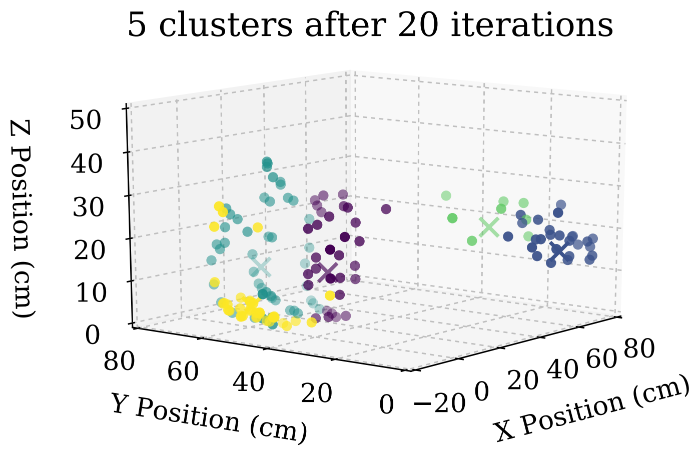


Figure 3.6: Labeled observations and DPMIRL subgoal means

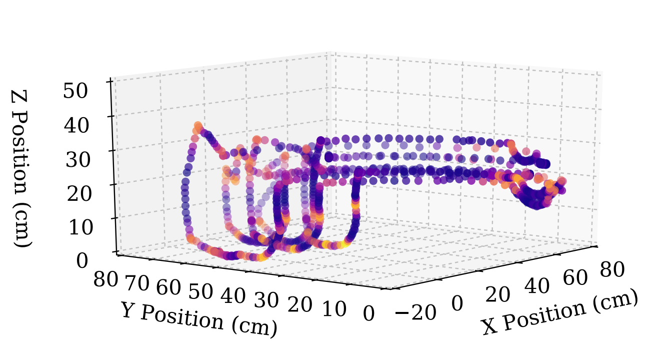
3.1.5 Changepoint Gaussian Mixture Model Clustering

A different perspective on subgoal identification involves considering states where the machine velocity changes abruptly, which we will call *changepoints*. The impetus for a changepoint may be internal to the operator or environmental, and could include the following:

1. the operator's subgoal has changed
2. the operator is pausing
3. the operator has given an erroneous input
4. the operator has decomposed a subgoal into multiple consecutive movements, between which there is an abrupt change of command
5. the location of an object in the workspace has changed

We wish to capture the distribution of states in category 1, while disregarding items 2-5. However, it is reasonable to suggest that items 2-5 may be somewhat randomly distributed through a demonstration dataset, especially with a more experienced operator. Under this assumption, over many cycles of a task the changepoints in categories 2-5 may only appear as random outliers. For example, consider the single dig cycle illustrated in Fig. 3.7. The color of the markers reflect the Euclidean norm of the acceleration vector. The acceleration norm appears to effectively identify the vertices of the operators path. However, the interaction of the bucket tool with the soil does cause some spikes in acceleration at the bottom of the digging process (lower left).

Intuitively, the change points are states where the velocity of the end-effector undergoes an abrupt change. More precisely, change points are states where the



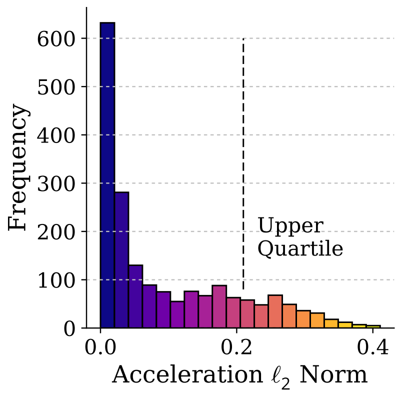


Figure 3.7: States and acceleration vector norms for a manual truck loading operation

ℓ_2-norm of the acceleration exceeds a threshold parameter $\bar{a}$

$$\mathcal{C} = \{a_i : \ \|a_i\|_2 > \bar{a}, \ \forall a_i \in \mathcal{A}\} \tag{3.10}$$

The threshold parameter can be determined by finding the value at the third quartile of the data (or any quantile), such that Eq. (3.10) simply selects the accelerations from the upper quartile.

Once the change points are separated from the trial observations, they can be clustered using a Gaussian mixture model with a Dirichlet process prior and variational inference algorithms such as scikit-learn's `BayesianGaussianMixture` [49]. The variational inference algorithm requires an upper bound on number of clusters and a weight concentration prior, γ. The changepoint clustering process is shown in Algorithm 3.

Results from the changepoint clustering algorithm are illustrated in Fig. 3.8.

Algorithm 3 Changepoint Gaussian Mixture Model

```
1: function CP-GMM-CLUSTERING($\mathcal{O}, q, \gamma$)
2:     $a_i \leftarrow \|\frac{s_{i-1} - 2s_i + s_{i-1}}{T_s^2}\|_2$    ▷ compute norm of acceleration vector for each state using central finite difference
3:     $\bar{a} \leftarrow$ compute acceleration at $q^{th}$ quantile    ▷ e.g. $q = 75\%$
4:     for each index i do
5:         if $a_i > \bar{a}$ then push state $s_i$ to list of changepoints $C$
6:         end if
7:     end for
8:     $\boldsymbol{z}, \boldsymbol{\mu}_{1:k}, \boldsymbol{\Sigma}_{1:k} \leftarrow$ BayesianGaussianMixture($C, \gamma$)    ▷ run variational inference on changepoint states and store cluster labels, means, and covariances
9:     return cluster labels $\boldsymbol{z}$, cluster means $\boldsymbol{\mu}_{1:k}$ and covariances $\boldsymbol{\Sigma}_{1:k}$
10: end function
```

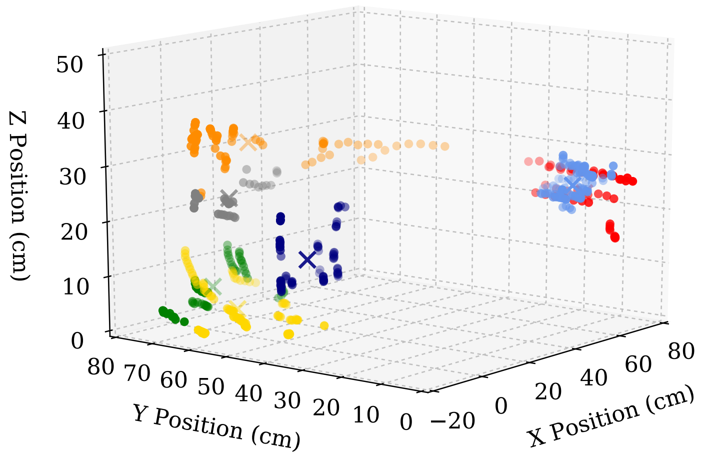


Figure 3.8: Changepoint clustering results

3.1.6 Summary of Task Identification

In summary, both the BNIRL and DPMIRL introduce useful extensions of the IRL framework, for capturing underlying task objectives from complex demonstration data. The assumption of multiple subgoals simplifies the search for candidate reward functions and creates a useful abstract representation of the task. Bayesian Nonparametric IRL (BNIRL) utilizes an action comparison likelihood which creates meaningful subgoal assignments based on the action direction, but does not incorporate a distribution over subgoal locations.

DP-Means IRL (DPMIRL) simplifies the action likelihood by assuming the operator already behaves as a closed-loop controller, and the remaining likelihood depends only on proximity to the subgoal. DPMIRL also pre-partitions the data with objects of interest in the workspace, and outputs subgoal clusters rather than distinct subgoal locations. However, this approach makes no use of the information contained in the operator's action when seeking subgoals.

Finally, changepoint Gaussian Mixture Model clustering (CP-GMM) examines only states at time instances when the acceleration magnitudes exceed a certain threshold. These states, deemed *changepoints*, are then clustered with a nonparametric Gaussian mixture model, yielding cluster means and covariances in the actuator space.

3.1.7 Stochastic Transition Matrix

Several learning methods have just been discussed for discovering either the location or distribution of locations of a subgoal from demonstration data, however, the order in which subgoals or subgoal distributions are drawn during a task has not been explicitly considered. In our forthcoming discussion of real-time prediction,

Algorithm	Inputs	Outputs
BNIRL [42]	states and actions in *actuator space*, hyperparameters η and β	partition labels for all states corresponding to the index of the subgoal state
DPMIRL	segmented states and actions in *end-effector space*, distance hyperparameter λ, points of interest in *end-effector space*	subgoal labels and subgoal distributions in the end-effector space
CP-GMM	segmented states and actions in the *actuator space*, acceleration quantile q	subgoal labels and distributions in the actuator space

Table 3.1: Summary of task identification algorithms

this ordering can be exploited as a prior probability of subgoal transition, written as

$$P(z_i = k | z_{prev} = j), \ \forall j, k \in \mathcal{K} \tag{3.11}$$

where z_i is the subgoal label of the current observation and z_{prev} is the last confirmed subgoal partition label, which may have occurred at any time index previous to i. The script $\mathcal{K}$ denotes the set of all subgoal partition labels. The transition probability can be stored in a stochastic matrix, T, where element T_{jk} describes the probability of transitioning from subgoal j to subgoal k. This matrix can be built empirically from any of the previous subgoal discovery techniques by counting the transitions between subgoal labels, setting element T_{jk} equal to total number of observed transitions from subgoal j to subgoal k, and normalizing the rows of the matrix, such that

$$\sum_{k=1}^{|\mathcal{K}|} T_{jk} = 1 \tag{3.12}$$

3.2 Prediction

Once a suitable representation of the task has been formed using the aforementioned techniques, the task model may be utilized to predict the operator's current subgoals in real-time. Three approaches are presented and compared here: 1) a *deterministic* finite-state machine (FSM) which monitors the machine state and operator input to trigger new subgoals, 2) a *probabilistic* model which compares the current action with the direction to the subgoal location, and 3) a second *probabilistic* model which also considers the uncertainty in subgoal locations to predict the likelihood of each subgoal. In all cases the set of possible inputs for real-time prediction are the following:

- the current state-action pair, $O_i = (s_i, a_i)$
- previous state-action pairs, $O_{0:i-1} = \{(s_0, a_0), ..., (s_{i-1}, a_{i-1})\}$
- a task model determined using one of the techniques in the previous section, either represented as a collection of distinct points, or a collection of Gaussian distributions.

3.2.1 Deterministic Prediction

Consider the truck loading task represented by six subgoals: above the pile, at the pile, after the dig, after material is lifted, the position over the truck, and after dumping material in the truck. The order in which the operator seeks these subgoals during nominal operation is fairly consistent, and could inform the mechanism which predicts the current subgoal of the operator. For a simple deterministic prediction scheme which exploits this ordering, we can define a Finite-State Machine (FSM), which uses state and action triggers to iterate through the subgoals, and boolean states of the blending parameter. Thus, if there are K subgoals, the prediction FSM has $2K$ states, defined by

$$S : s_{k,\alpha}$$
$$k \in \{1, ..., K\}$$
$$\alpha \in \{\alpha_o, 0\}$$

where α_o is the designed blending parameter.

Since the actual operator subgoals may vary during operation with changes in the workspace, a region is defined around each subgoal location, which we refer to as *termination sets.* When the machine pose lies within a termination set, the state machine transitions to the next subgoal in the task. Termination sets are complemented by *initiation sets*, which are subsets of the input space, and toggle the blending parameter state. If the joystick command from the operator lies within the initiation set of the current subgoal, then blending is *active.* Otherwise, blending does not occur.

The termination set for subgoal k can be formally defined as

$$T_k = \{\boldsymbol{\zeta} : |\zeta_i^k - \zeta_i| \leq \sigma_i\} \tag{3.13}$$

where $\boldsymbol{\zeta}$ is a vector of the current actuator displacements, ζ_i^k is the displacement of the i^{th} actuator at the k^{th} subgoal, and σ_i is a design parameter for the size of the termination set. The parameter, σ_i, can be selected from the variance of subgoal clusters. The termination sets employed for the truck loading task are illustrated in Fig. 3.9 in the end-effector workspace.

Overlapping termination sets cause the prediction state to not be uniquely defined, and must be avoided. In contrast, termination sets too narrow can result

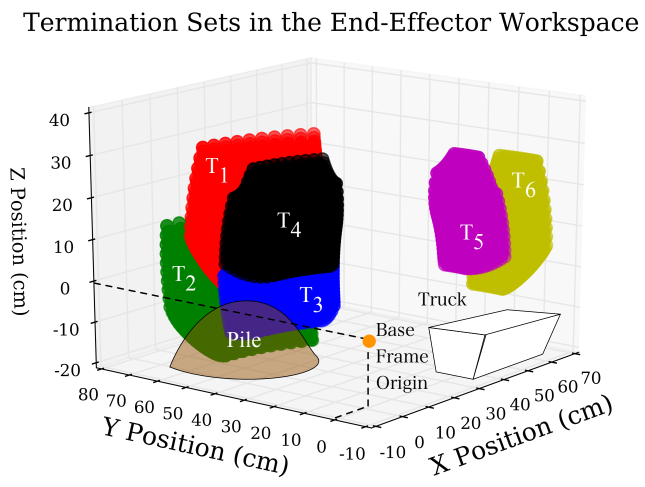


Figure 3.9: The 6 termination sets illustrated in the workspace. Note that the sets are defined in the actuator space, from which the forward kinematics have been used to illustrate these regions in the end-effector space.

in missed subgoal cues. Further, large termination sets reduce the blended shared control assistance length.

The initiation sets can be defined as

$$I_k = \{\bar{\mathbf{u}} : |\bar{u}_j - \bar{u}_{j,k}| > 0\} \tag{3.14}$$

where $\bar{u}_j$ is the operator input on the j^{th} input axis, and $\bar{u}_{j,k}$ is the initiation threshold. The prediction FSM states for subgoal k are illustrated in Fig. 3.10, and a general implementation of the prediction update algorithm is shown in Algorithm 1.

This deterministic prediction strategy was implemented within a BSC architecture and tested with human operators, for which the results are shown in Section 5.1. This strategy also simply degrades to manual control if a termination set is missed by the predictor. However, this approach does not produce any probability

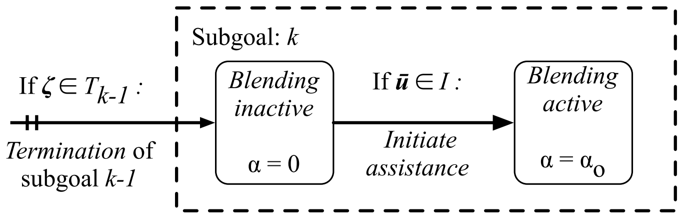


Figure 3.10: Prediction state diagram for subgoal k. Since the task is cyclic, the conventions $k-1$ and $k+1$ are used here merely for convenience, and actually refer to the previous and next subgoals, respectively.

Algorithm 4 FSM Prediction Update

```
1:  function FSM-UPDATE-PREDICTION($\boldsymbol{\zeta}, \bar{\mathbf{u}}$)
2:      for each subgoal termination set $T_k$ do
3:          if $\boldsymbol{\zeta} \in T_k$ then          ▷ if in termination set of any subgoal $k$
4:              $subgoal \leftarrow (k+1)\ \%\ K$          ▷ iterate to next subgoal
5:          end if
6:      end for
7:      if $\bar{\mathbf{u}} \in I_k$ then          ▷ if in initiation set of current subgoal $k$
8:          $\alpha \leftarrow \alpha_o$          ▷ activate blending
9:      else
10:         $\alpha \leftarrow 0$          ▷ otherwise deactivate blending
11:     end if
12:     return $subgoal$, $\alpha$
13: end function
```

or confidence in the identified subgoal, a measure which may be useful in the blending step of our control scheme. Further, the designer is required to identify and encode some information about the task in the form of initiation sets, which has not been discovered algorithmically through the IRL process.

3.2.2 Probabilistic Action Comparison Prediction

The Bayesian model from the previous task learning sections can be revisited to produce a statistical distribution over subgoal assignment labels, given by

$$P(z_i = j | z_{-i}, O_i) \text{ for } j \in \mathcal{K}$$

where O_i is the i^{th} observation, z_i is the subgoal assignment at observation i, z_{-i} are all other subgoal assignments in the demonstration, and $\mathcal{K}$ is the complete set of subgoal partition labels. However, we need to be able to compute the posterior probability in real-time. One natural solution is to transform the statistical model from our Bayesian nonparametric IRL implementation with

$$\underbrace{p(z_i | z_{i-1}, O_i)}_{\text{assignment probability}} \propto \underbrace{P(z_i | z_{i-1})}_{\text{transition model}} \underbrace{P(a_i | s_i, z_i)}_{\text{action likelihood}} \tag{3.15}$$

where we have exchanged the nonparametric CRP prior with a transition model derived from operator data, and z_{i-1} is the simply last confirmed subgoal[†]. The action likelihood remains as stated in Eq. (3.8), comparing the vector from the current state, s_i, to the partition subgoal, g_{zi}, with the current action vector, a_i.

To implement this prediction scheme, the action likelihood is computed over only the subgoal locations, $\boldsymbol{g_j}$, or means of the subgoal distributions, $\boldsymbol{\mu_j}$, for $j \in \mathcal{K}$. However, if from our task learning process, we have a representation of subgoals which is a mixture of probability distributions over the state space, then we are unable to incorporate the variance of these subgoal distributions with this approach.

[†] *confirming* that the operator has visited a subgoal is a significant challenge, as discussed in Ch. 6

3.2.3 Multivariate Normal Action Comparison Prediction

In order to utilize the full subgoal information generated in the DPMIRL or changepoint clustering algorithm, two modifications to the Bayesian model above can be proposed:

1. Subgoals are drawn each cycle from a multivariate normal (MVN) distribution, such that the location of the subgoal z_i in the actuator space is

$$\mathbf{g}_{z_i} \sim \mathcal{N}(\boldsymbol{\mu}_{z_i}, \boldsymbol{\Sigma}_{z_i})$$

   where $\boldsymbol{\mu}_{z_i}$ and $\boldsymbol{\Sigma}_{z_i}$ are the multivariate normal mean and covariance which are discovered during the GMM clustering step of the changepoint clustering algorithm.

2. The action likelihood should prioritize any actions which move towards the largest *mass* of this subgoal distribution, rather than just towards the mean of the subgoal distribution.

Our goal is to define the likelihood that a particular action will be taken given the subgoal z_i and its MVN distribution, $\mathcal{N}(\boldsymbol{\mu}_{z_i}, \boldsymbol{\Sigma}_{z_i})$. This can be achieved by integrating the multivariate probability density function over a custom domain, denoted $\mathcal{D}$. The domain is defined in the actuator state-space coordinates $\mathcal{D} \subset \mathbb{R}^m$ according to the following:

On axes in which the action input is null, the domain is infinite (corresponding to marginalizing out the null input axes). On the active input axes the domain is semi-finite, bounded by the current state and infinity in the action direction. Mathematically, the domain $\mathcal{D}$ is defined by:

$$\mathcal{D} = \mathcal{D}_1 \times \mathcal{D}_2 \times \mathcal{D}_3 \times \mathcal{D}_4 \tag{3.16}$$

where

$$\mathcal{D}_j = \begin{cases} [s_{ij}, +\infty) & a_{ij} > 0 \\ (-\infty, s_{ij}] & a_{ij} < 0 \\ (-\infty, +\infty) & a_{ij} = 0 \end{cases} \tag{3.17}$$

Then the action likelihood becomes

$$p(\boldsymbol{a}_i | \boldsymbol{s}_i, z_i) = \int ... \int_{\mathcal{D}} f_{z_i}(x_1, x_2, ..., x_m)\ dx_1 ... dx_m \tag{3.18}$$

where $f_{z_i}(x_1, x_2, ..., x_m)$ is the probability density function (PDF) of subgoal z_i

$$f_{z_i}(\mathbf{x}) = \frac{\exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_{z_i})^{\mathrm{T}} {\boldsymbol{\Sigma}_{z_i}}^{-1} (\mathbf{x} - \boldsymbol{\mu}_{z_i})\right)}{\sqrt{|2\pi\boldsymbol{\Sigma}_{z_i}|}} \tag{3.19}$$

This integration can be computed by Alan Genz's multivariate normal FORTRAN routines in the SciPy package [50]. However, in most cases the action will only be along one or two unit vector directions, such that for the other directions $a_{ij} = 0$. In this case, the domain in the j^{th} direction is infinite. Computing this integral simply corresponds to marginalizing out the j^{th} variable. Therefore, with multivariate normal distributions we can simply omit the covariances of marginalized variables from the covariance matrix and compute the integral over the remaining variables.

Two example state-action pairs are given in Fig. 3.11 in a 2-dimensional domain with three multivariate normal subgoal distributions. The resulting subgoal likelihoods are shown below each plot.

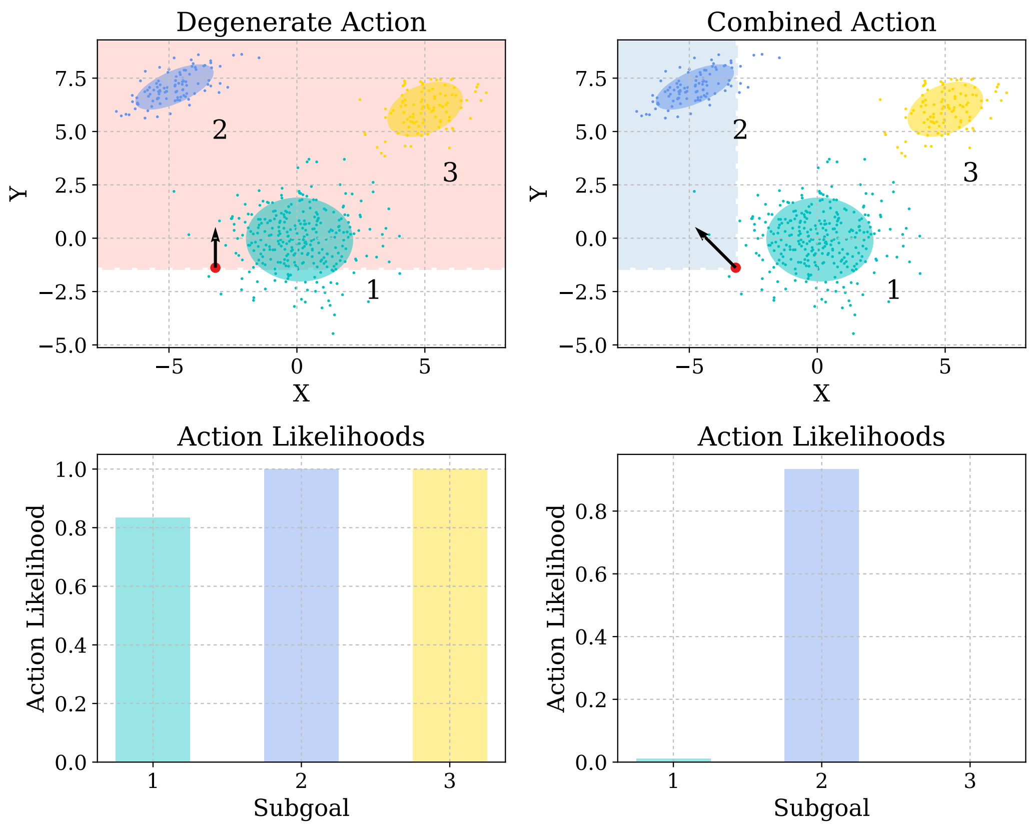


Figure 3.11: Two test cases of the Gaussian action likelihood

3.2.4 Prediction Comparison and Summary

A single dataset consisting of nine truck loading task cycles was used to evaluate each prediction method with the same predefined subgoal decomposition, which consisted of six subgoals clustered with the GMM algorithm in actuator space. In the case of the finite state machine (FSM-P) and action comparison (AC-P) methods, the subgoal means were used as subgoal locations, whereas the multivariate normal (MVN-AC-P) technique utilized both the means and covariances. First the data were hand labeled according to their position in the task, and then the observations were iterated over, storing the predicted subgoal and subgoal probabilities where applicable. The results are shown in Table and the labels are illustrated in Fig. 3.12. Table 3.2 illustrates the total accuracy of each method compared to the hand labeled data. If a threshold of 70% confidence is used in the probabilistic methods before blending occurs, we can also discuss the number of blended samples and *blending active accuracy* of the each mode, which is also shown in the table.

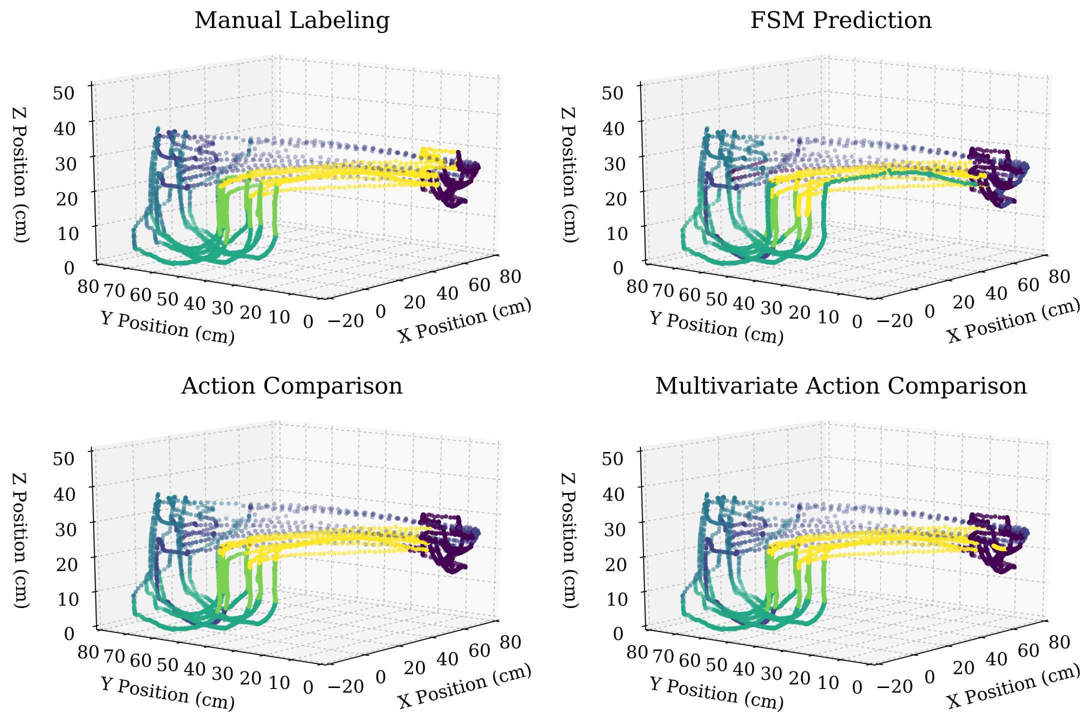


Figure 3.12: Comparison of prediction methods

The FSM-based prediction (FSM-P) appears less accurate but is highly accurate when the blending is active (i.e., when the input is within the initiation set of the current subgoal), which is only about half of the task. The action comparison (AC-P) and multivariate normal action comparison (MVN-AC-P) methods both perform

Method	Total Accuracy	Blending Active Accuracy (Pct. of Trial Active)
Deterministic FSM (FSM-P)	60.9%	94.9% (55.5%)
Action Comparison Prediction (AC-P)	78.9%	79.7% (97.1%)
Multivariate Normal Action Comparison Prediction (MVN-AC-P)	78.9%	80.59% (94.8%)

Table 3.2: Summary of prediction algorithm performance

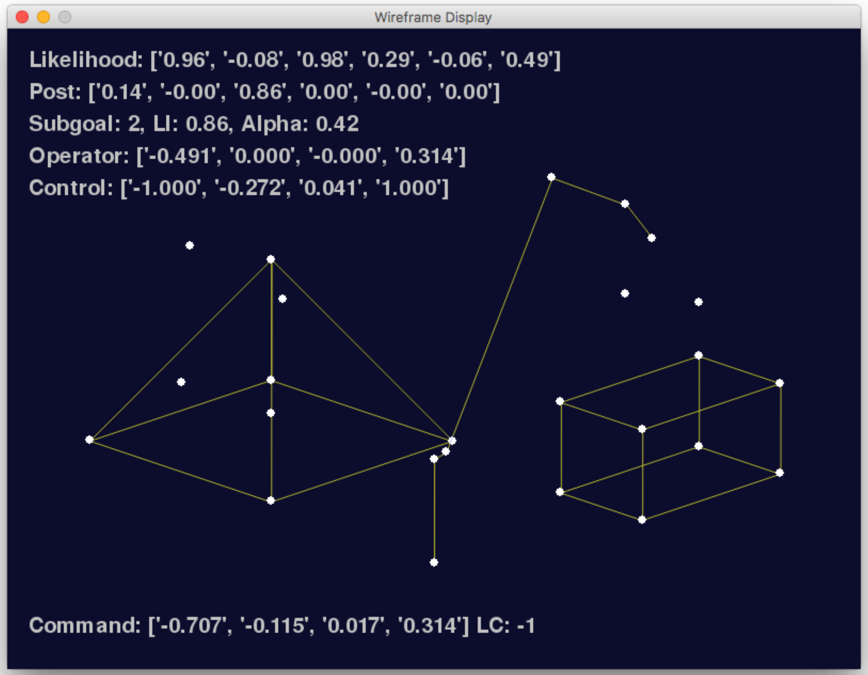
very similarly due to sharing the same stochastic transition matrix, with about 80% accuracy and blending nearly all of the time. Increasing the active blending threshold would produce higher accuracy at the expense of less time spent assisting, and these results are only meant to illustrate these trade-offs. Also, these trials do not expose how each method responds to off-nominal situations, which will remain the subject of further investigation and is beginning to be addressed in published research [51]. A qualitative summary of prediction methods is presented in Table 3.3.

Method	Outputs	Limitations
Deterministic FSM (FSM-P)	subgoal and boolean blending parameter	does not quantify confidence in prediction
Subgoal Action Comparison (AC-P)	probability vector for all subgoals	does not consider the distribution of subgoals
Multivariate Normal Action Comparison (MVN-AC-P)	probability vector for all subgoals	computationally expensive, requires covariances of each subgoal

Table 3.3: Summary of subgoal prediction algorithms

3.2.5 Wireframe Simulation

Testing many different prediction algorithms requires a method for rapidly prototyping and evaluating different strategies. Rather than deploying each algorithm to the experimental model which will be presented in Chapter 5, a simple wireframe simulation was developed as a drop-in replacement for the scale model. The real-time simulator is purely kinematic, mapping joystick inputs to a velocity on each actuator, but allows the operator to explore resulting prediction for any combination of states and actions. The interface is shown in Fig. 3.13, with printed values for subgoal likelihoods, posterior probability, maximum a posteriori subgoal, operator input, control input, and blended command.



```
Likelihood: ['0.96', '-0.08', '0.98', '0.29', '-0.06', '0.49']
Post: ['0.14', '-0.00', '0.86', '0.00', '-0.00', '0.00']
Subgoal: 2, Ll: 0.86, Alpha: 0.42
Operator: ['-0.491', '0.000', '-0.000', '0.314']
Control: ['-1.000', '-0.272', '0.041', '1.000']
Command: ['-0.707', '-0.115', '0.017', '0.314'] LC: -1
```

Figure 3.13: A wireframe kinematic simulation for developing prediction algorithms

4. TRAJECTORY CONTROL AND BLENDING

This chapter addresses the generation of minimum-time smooth trajectories to seek the subgoals of the operator, and the low-level controller designed to track the trajectories. Then, the determination of the blending parameter is discussed along with a method of incorporating the prediction confidence in the blending law.

4.1 Trajectory Generation

In the deterministic BSC approach, once a new subgoal is identified, a smooth quintic polynomial trajectory is generated to that subgoal, using the velocity and acceleration-limited minimum time trajectory. Quintic polynomial templates are standard for smooth point to point motion within robotics contexts, and a good overview of various point to point trajectory generation techniques is given by Spong [43]. First, the maximum velocity and acceleration are used to determine a minimum trajectory duration on each actuator. The longest duration (i.e., slowest actuator movement) is used to determine quintic trajectories for all actuators.

The initial and final velocities and acceleration are set to zero, while the initial and final machine poses, $\boldsymbol{\zeta}_i$ and $\boldsymbol{\zeta}_f$, are set to the current state and the subgoal, respectively. The initial time is set to 0, and the final time t_f is the duration determined previously. The coefficients of the quintic polynomial trajectory for each actuator are given by the columns of the coefficient matrix, C, computed as

$$C = Q^{-1}L = \begin{bmatrix} 1 & t_i & t_i^2 & t_i^3 & t_i^4 & t_i^5 \\ 0 & 1 & 2t_i & 3t_i^2 & 4t_i^3 & 5t_i^4 \\ 0 & 0 & 2 & 6t_i & 12t_i^2 & 20t_i^3 \\ 1 & t_f & t_f^2 & t_f^3 & t_f^4 & t_f^5 \\ 0 & 1 & 2t_f & 3t_f^2 & 4t_f^3 & 5t_f^4 \\ 0 & 0 & 2 & 6t_f & 12t_f^2 & 20t_f^3 \end{bmatrix}^{-1} \begin{bmatrix} \text{---}\ \zeta_i\ \text{---} \\ \text{---}\ 0\ \text{---} \\ \text{---}\ 0\ \text{---} \\ \text{---}\ \zeta_f\ \text{---} \\ \text{---}\ 0\ \text{---} \\ \text{---}\ 0\ \text{---} \end{bmatrix} \tag{4.1}$$

An example trajectory is given in Fig. 4.1 in the following section.

4.2 Low-Level Control

A discrete-time PI controller was developed and tuned as a low-level controller for trials with BSC. The continuous-time parallel PID transfer function is given is the Laplace domain by

$$C(s) = K_P + \frac{K_I}{s} + K_D s \tag{4.2}$$

Applying Tustin's (bilinear) transformation the discrete time control law in the Z-domain is

$$C(z) = \frac{\alpha z^2 + \beta z + \gamma}{z^2 - 1} \tag{4.3}$$

where the coefficients are precomputed according to

$$\alpha = \frac{4K_D + K_I T^2 + 2K_P T}{2T} \tag{4.4}$$

$$\beta = \frac{2K_I T^2 - 8K_D}{2T} \tag{4.5}$$

$$\gamma = \frac{4K_D + K_I T^2 - 2K_P T}{2T} \tag{4.6}$$

After applying the shift operator and taking the inverse Z-transform, the discrete control law is given by

$$u[k] = \alpha e[k] + \beta e[k-1] + \gamma e[k-2] + u[k-2] \tag{4.7}$$

Deadband compensation is achieved by mapping the normalized control input, u, according to

$$f : u \mapsto (u_{db}\,\mathrm{sgn}(u) + (1 - u_{db})u) \tag{4.8}$$

where u_{db} is the positive normalized deadband width as discussed in Sec. 5.1.2. This function maps the interval, $u \in [-1, 1]$, to two disjoint intervals and the zero element, i.e., $f(u) \in [-1, -u_{db}) \cup \{0\} \cup (u_{db}, 1]$.

The Ziegler-Nichols tuning method was used initially to determine the gain values, after which the gains were adjusted by hand to reduce oscillations about the setpoint. The resulting tracking performance of Ziegler-Nichols tuning with deadband compensation for point to point quintic trajectories is illustrated in Fig. 4.1. Performance degrades significantly when all actuators are moved simultaneously, in which case, flow limiting conditions cause certain actuators to receive a large portion of the fluid. This issue is likely to be less limiting in full-scale excavator equipment, due to the typical use of multiple variable-displacement (axial-piston) pumps and load sensing circuits to meet flow demand.

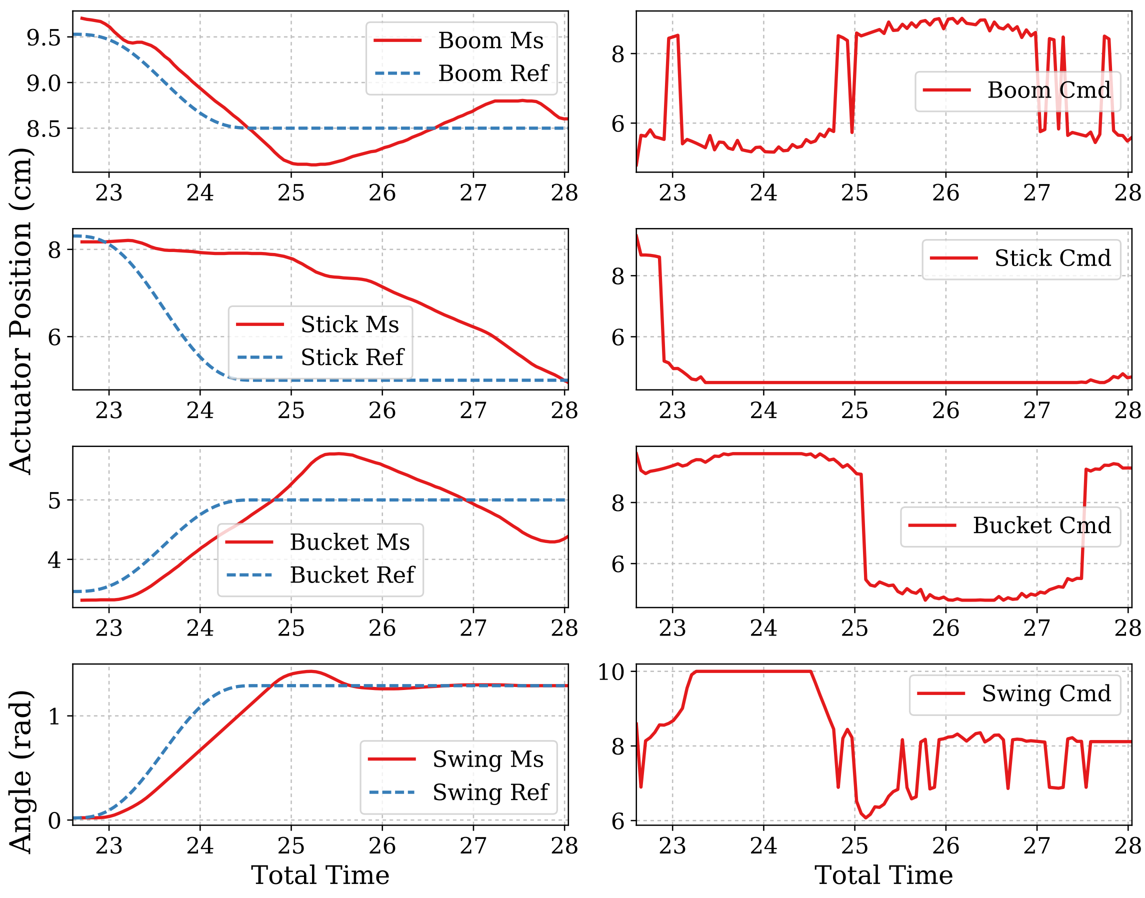


Figure 4.1: Tracking performance for simultaneous tracking on all actuators; *the stick actuator receives low flow priority and cannot meet tracking demands, even with the valve saturated*

4.3 Blending of Control Inputs

Control blending is the process by which the operator input (human agent, or HA) is combined with the output of the previously defined automatic controller (computer agent, or CA). As a review, in BSC the blending step is achieved through the following law

$$u = \bar{u} + \alpha(u^* - \bar{u}) \tag{4.9}$$

or alternatively

$$u = \alpha u^* + (1 - \alpha)\bar{u} \tag{4.10}$$

where $\bar{u}$ is the human operator command, u^* is the automatic control input, α is a variable blending parameter such that

$$\alpha \in [0, 1] \tag{4.11}$$

Many researchers have sought to develop an expression for the blending parameter which continuously and smoothly trades authority between the two agents. However, at this point, it is not clear which information to use as a basis for this relationship. On Zermelo's ship navigation problem, Enes experimented with using a combination of proximity to the goal state and action comparison with the optimal control input, given by[*] [52]

$$\alpha = \max\left(0, 1 - \frac{d}{d_0}\right) \max\left(0, 1 - \left(\frac{\Delta}{\Delta_0}\right)^2\right) \tag{4.12}$$

where d_0 is a distance threshold and Δ_0 is an input deviation threshold. Under this relation the control becomes fully manual if the distance from the operator to the goal exceeds d_0 or the command input deviates too far from the optimal control input

[*]original publication used e in place of α, they are exchanged here for the sake of clarity

(measured by Δ).

Some authors have used proximity to subgoals alone as a basis for the blending parameter, however proximity alone cannot discriminate between seeking a subgoal and departing from the same subgoal. Therefore, proximity as a sole means of determining the blending parameter is not practical.

As discussed in Sec. 1.3, Dragan and Srinivasa contributed the heuristic that the blending parameter should reflect the degree of confidence with which the operator's goal has been predicted. This confidence may be expressed either as the probability of a certain subgoal or as the entropy of the subgoal probability distribution.

Using either of the probabilistic prediction methods presented in Secs. 3.2.2 and 3.2.3, the prediction confidence can be directly related to the blending parameter α. After normalizing the assignment posterior of Eq. (3.15) for all subgoals, we are left with a discrete distribution over subgoal assignments

$$P(z_i = j | z_{i-1}, O_i) \text{ for } j \in \mathcal{K} \tag{4.13}$$

where j is a subgoal label from the set of all subgoal labels $\mathcal{K}$.

MAP Probability as Confidence. Using the following conditions, the *maximum a posteriori* (MAP) subgoal probability can be mapped to the value of alpha parameter. First, a minimum threshold probability is defined p_0, and a target alpha interval, $A = [\alpha_l, \alpha_h]$, where $A \subset [0, 1]$. If the MAP subgoal assignment probability exceeds the *blending threshold* p_0, that subgoal is designated the current subgoal, and the α parameter is defined by

$$\alpha = f(p_{MAP}) \tag{4.14}$$

where the MAP subgoal probability p_{MAP} is defined as

$$p_{MAP} = \max_{j} P(z_i = j | z_{i-1}, O_i) \tag{4.15}$$

and f is the affine mapping from the interval $[p_0, 1] \to [\alpha_l, \alpha_h]$, given by

$$f(x) = (x - p_0) \left(\frac{\alpha_h - \alpha_l}{1 - p_0} \right) + \alpha_l \tag{4.16}$$

As one function, the blending parameter α can be defined piecewise by

$$\alpha = \begin{cases} f(p_{MAP}) & p_{MAP} \geq p_0 \\ 0 & p_{MAP} < p_0 \end{cases} \tag{4.17}$$

Inverse Entropy as Confidence. Alternatively, using the statistical entropy definition of confidence, the blending parameter can be defined as

$$\alpha = g \left(\frac{H(z_i)}{\log |\mathcal{K}|} \right) \tag{4.18}$$

where $H(z_i)$ is the statistical entropy of the subgoal assignment distribution

$$H(z_i) = - \sum_{j=1}^{|\mathcal{K}|} P(z_i = j | z_{i-1}, O_i) \log P(z_i = j | z_{i-1}, O_i) \tag{4.19}$$

The logarithm in the denominator of Eq. (4.18) serves to normalize the range of statistical entropy using the number of subgoals in the model, $|\mathcal{K}|$, and must share the same base of the logarithm used in Eq. (4.19). This way the argument of the mapping function will always follow

$$\frac{H(z_i)}{\log|\mathcal{K}|} \in [0,1] \tag{4.20}$$

However, a *high entropy* should map to a *low confidence*, so the mapping function must invert the interval, given by

$$g(x) = (x - H_0)\left(\frac{\alpha_l - \alpha_h}{1 - H_0}\right) + \alpha_h \tag{4.21}$$

where H_0 is the blending threshold for normalized entropy *below which* blending is active, and the high and low bounds of the blending parameter have been exchanged from Eq. (4.16). A piecewise definition of the inverse entropy approach is given by

$$\alpha = \begin{cases} g\left(\frac{H(z_i)}{\log|\mathcal{K}|}\right) & \frac{H(z_i)}{\log|\mathcal{K}|} \leq H_0 \\ 0 & \frac{H(z_i)}{\log|\mathcal{K}|} > H_0 \end{cases} \tag{4.22}$$

Composing either determination of the blending parameter with the prediction approaches developed in Sec. 3.2 produces an arbitration step which is very capable at either determining the current subgoal from a fixed set, or disabling any control intervention. However, there are still potential sources of poor performance in real-world excavation tasks, which must be addressed if a shared control architecture is to be implemented on a physical excavator and tested with human operators. For instance, the subgoal set may become inaccurate, subject to a changing work environment. Further, the task model may be incomplete.

For these reasons, it is useful to impose some practical constraints on the implementation of a blending law and the determination of the blending parameter:

1. *Human-in-the-loop (HIL) constraint*: At *all* times, the operator is present in

the control loop.

2. *Input responsive constraint*: The input to the plant (hydraulic valves and swing motor) is *null* if the operator's joystick input is null.

3. *Non-conflicting constraint*: The CA may not oppose the HA, *i.e.*, if $\mathrm{sgn}(\bar{u}) \neq \mathrm{sgn}(u^*)$, then $\alpha = 0$, and control is fully manual.

The first constraint is realized by imposing a conservative upper bound on the alpha parameter, *e.g.*, $\alpha_h \approx 0.7$. The second is already achieved in the FSM-P prediction scheme. Further, with either probabilistic prediction method, if the joysticks are released, the action likelihood reduces to zero and blending cannot be active. The third and final constraint is the most severe, and sacrifices many potential performance improvements, such as reducing operator overshoot or overdigging. Handling conflicting intent is a key problem in artificial intelligence domains, and one that merits its own discussion. However, this third constraint allows a failure of the prediction mechanism to not become a system failure, by never counteracting the operator's command.

5. EXPERIMENTAL DESIGN AND RESULTS*

This chapter introduces a series of experiments conducted in order to evaluate the presented task learning, prediction, and BSC control methods. A 1/12th scale fully hydraulic excavator was instrumented for on-board embedded control and integrated with a remote operator interface to send joystick commands. Trials were conducted with three operators and the deterministic prediction method presented previously in Sec. 3.2.1. The results of each trial are analyzed for improvements in fundamental performance metrics such as cycle time and digging efficiency, as well as for practical limitations such as scalability and operator comfort.

5.1 Experimental Setup

All digging experiments were conducted on an instrumented 1/12th scale electro-hydraulic remote-controlled excavator, illustrated in Fig. 5.1 and presented by subsystem in the following section.

5.1.1 1/12th Scale Hydraulic Excavator

The 1/12th scale electro-hydraulic remote-controlled excavator (RC4WD 4200XL) is designed with all functions controlled by an 8-channel 2.4 GHz receiver. The two independent tracks and swing function are coupled to three 12 V brushed DC motors, which are controlled by three brushed electronic speed controllers (BESCs). The hydraulic pump is driven by a brushless DC motor, controlled by an electronic speed controller (ESC). Three DC servomotors (Tower Pro MG945) are mounted to an aluminum valve block and coupled to three closed-center rotary valves.

* Part of this section is reprinted from "Blended Shared Control of a Hydraulic Excavator" by M. Allain, S. Konduri, H. Maske, P. Pagilla, and G. Chowdhary, "Blended Shared Control of a Hydraulic Excavator," in International Federation of Automatic Control, 2017.

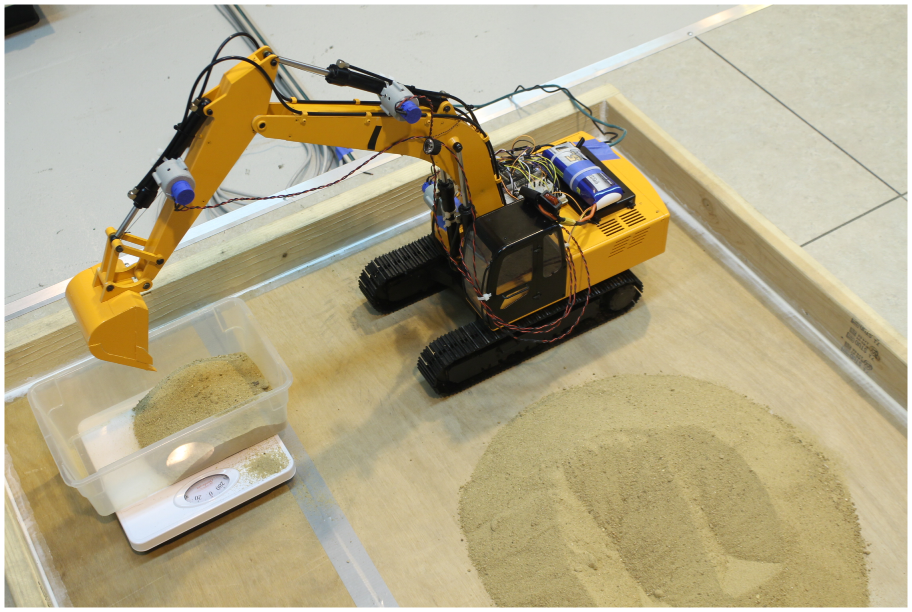


Figure 5.1: The 1/12th-scale instrumented hydraulic excavator

5.1.2 Instrumentation

The task of instrumenting the scale excavator consisted of three main steps: 1) *identification*, 2) *control*, and 3) *measurement*.

Identification. Before a controller can be implemented, the valve control signals must be balanced so that the valves operates identically in both directions (extension and retraction of the cylinder). Also, the deadband of each valve must be identified. Actuating each valve with a specific pulse width modulation (PWM) duty cycle and measuring the steady-state velocity response of the actuator yields the curves in Fig. 5.2. The steady-state response of the swing motor in *rad/s* is also illustrated. The valve deadbands and the duty cycles corresponding to the midpoint of each valve vary across all valves, and these valve parameters are stored in a data structure and utilized by the control software. Observing the steady-state response curves, the width of the deadband (illustrated as a rectangular patch) is significant compared to the control input range where the actuator experiences motion. Further, the steady-state velocity is maximized at a specific valve position, and in the case of the boom actuators, can even decrease with larger control input.

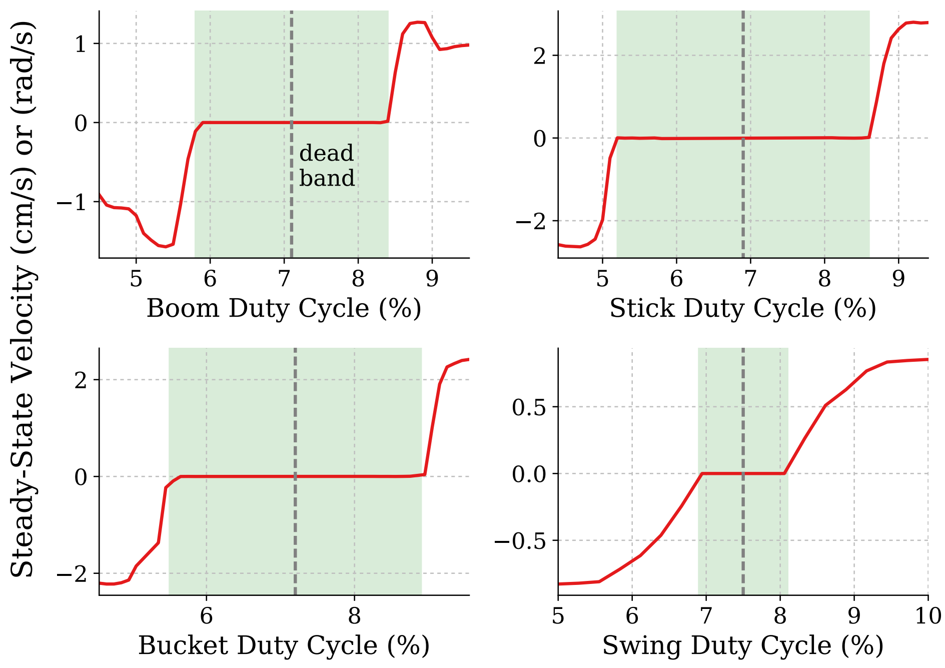


Figure 5.2: Steady-state actuator velocity as a function of PWM duty cycle

Control. The BeagleBone Black (BBB) RevC board, based on the TI Sitara system on a chip (SoC) was selected as an embedded controller for the excavator. The BBB is a capable development platform with an AM335x 1GHz ARM Cortex-A8 processor (TI), which excels in memory access speed and has 69 configurable general-purpose inputs and outputs (GPIOs). Control development and experimentation were conducted on top of a standard Linux-kernel based operating system (Debian 7.5).

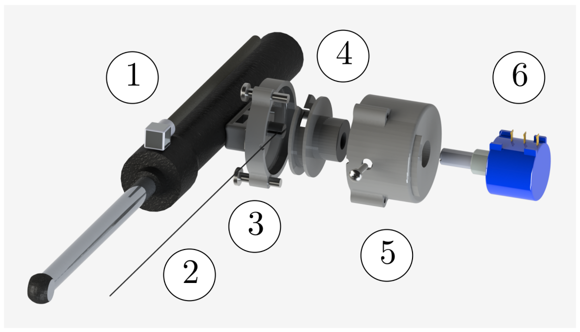
The main hydraulic servo valves and swing function are controlled via an on-board

BeagleBone Black running real-time control algorithms in Python. The Python scripts are heavily factored and compiled ahead of time to support the computationally demanding prediction and control algorithms. Further, most computation heavily utilizes the efficient data types and routines in the numerical Python package (NumPy) to improve execution speed.

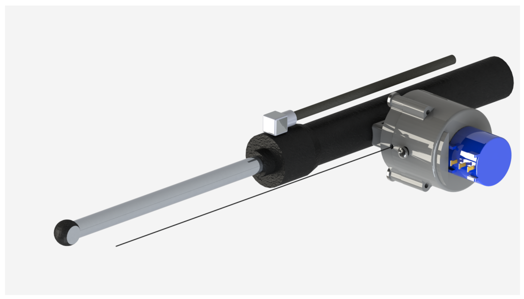
Measurement. A quadrature encoder was equipped on the swing motor shaft to retrieve incremental measurements of swing rotation. The quadrature encoder counting is managed by the eQEP module available on the BBB.

Several different linear position sensors were considered for providing feedback of actuator displacements. Preference was given to absolute position sensors to avoid the homing routines and drift associated with incremental encoders. In the earth-moving equipment industry, Komatsu has been issued a patent for a roller-based displacement sensor, which is mounted on top of the cylinder and rolls along the hydraulic piston rod [53]. The roller revolutions can then be counted by an incremental encoder, or a multi-turn potentiometer. After preliminary tests, this technique is difficult to reproduce at reduced scale due to the lower friction between the rod and roller relative to the internal friction of available encoders and potentiometers.

Instead, custom string potentiometers were adapted from an open-source design [54] which offered cost-effective and sufficiently precise absolute position measurement. The design was modified to support mounting on the hydraulic cylinders. Three identical string potentiometers assemblies were manufactured from ABS plastic using a commercial 3D printer and mounted to each cylinder. The components are illustrated in an expanded view in Fig. 5.3a, showing the 1) hydraulic cylinder, 2) braided nylon string, 3) spring housing, 4) spool, 5) case, and 6) 5k Ω multi-turn precision potentiometer (Bourns 3590S-2-502L) and the collapsed view in Fig. 5.3b. The string potentiometers house a spool which is tensioned by a radial spring and mounted to the shaft of a precision multi-turn potentiometer.



(a) Expanded view


(b) Assembled view

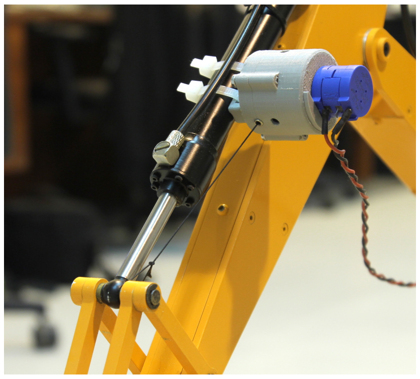

(c) String potentiometer mounted to the excavator

Figure 5.3: String potentiometer assembly

Example calibration curves can be seen in Fig. 5.4 for the three string potentiometers. In initialization of the control script, a look-up table (LUT) is formed from the calibrated value pairs, so that in real-time the table can be linearly interpolated to find the current displacement of the actuators.

A simple PCB was designed in EAGLE (Autodesk Inc.) to provide a robust point of connection for I/O (PWM, quadrature encoder, and potentiometer analog inputs), which can be seen in Fig. 5.5.

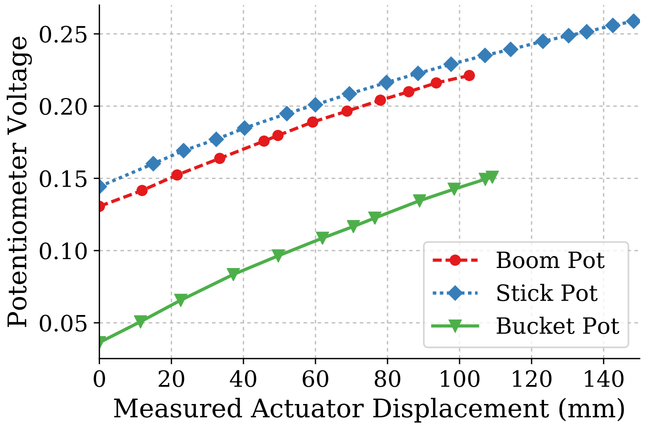
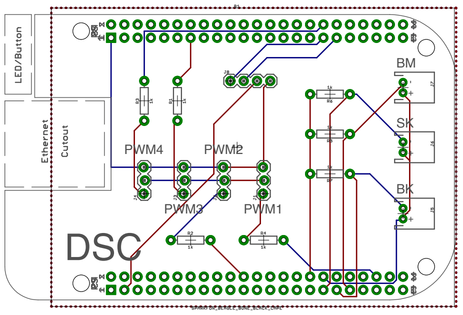
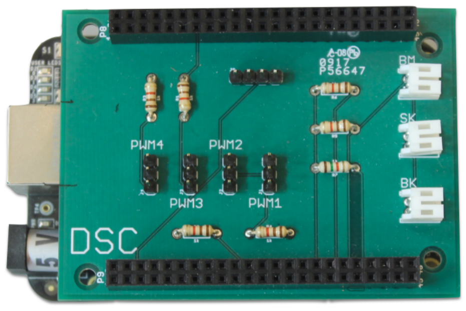


Figure 5.4: Calibration curves for the string potentiometers



(a) PCB design



(b) Manufactured PCB

Figure 5.5: BeagleBone Cape PCB

5.1.3 Operator Interface

Human subjects operated the excavator using two joysticks (Thrustmaster TM-1600[1]) with the standard 4-axis input mapping found in full-scale hydraulic excavators. Joystick inputs were then transmitted via UDP from a remote computer terminal to the on-board processor at a rate of 20 Hz.

[1]the TM-1600 joysticks are ambidextrous, so that one can be reconfigured to a left-handed grip

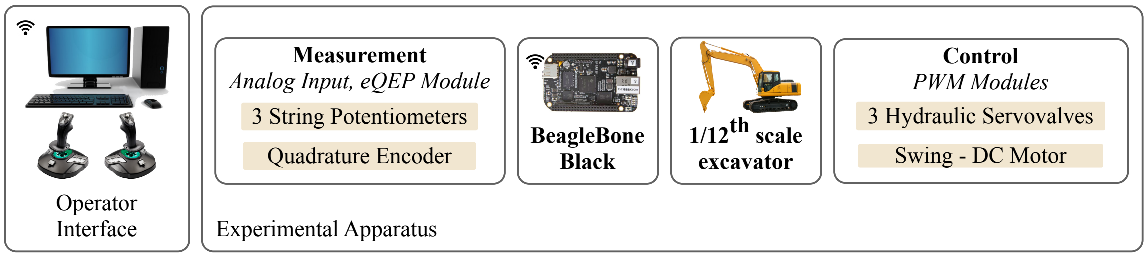


Figure 5.6: Overview of experimental setup

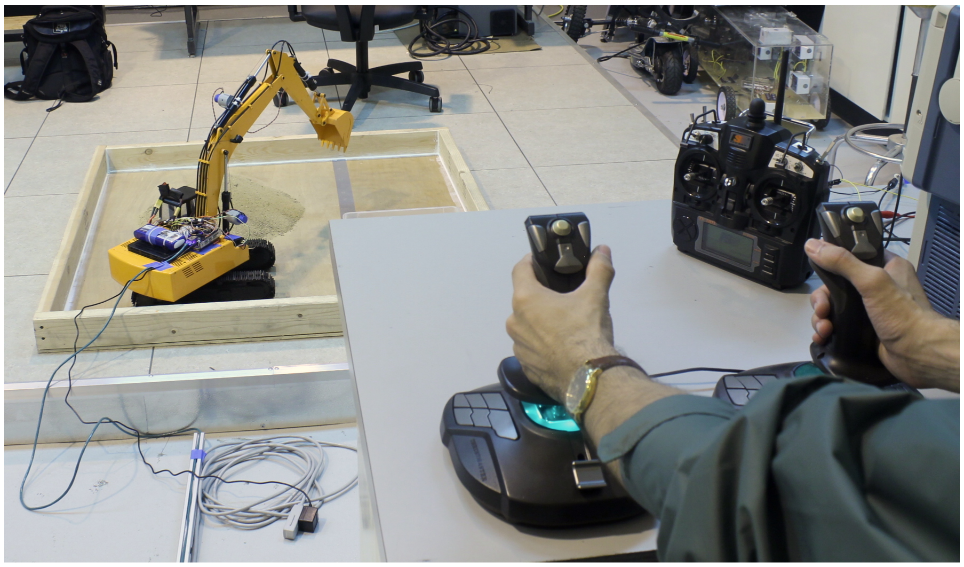


Figure 5.7: Operator view of workspace

5.2 Deterministic BSC Trials

Three sets of experiments were conducted for the truck loading task. For the first set of experiments, manual trials provided a benchmark for operator skill level. Next, autonomous trials established a baseline for machine capability with respect to cycle time. Finally, the blended trials utilize blended shared control with the deterministic prediction method summarized in Sec. 3.2.1. The actuator positions over time and the mass of dirt moved were recorded in all trials. In addition, the controller, prediction, and joystick states were recorded over time, where applicable.

In the manual control operation, subjects were given the task objective and the opportunity to practice operating the excavator for a short duration. After the practice period, the subjects were asked to perform 5 task cycles, which constitute one trial. Each of the 3 subjects performed 3 trials under manual control, and 3 trials under blended shared control. Under blended shared control, participants were informed that their command inputs would be augmented continuously, and were again given time to become familiar with the system. The logical software overview of the BSC implementation is shown in Fig. 5.8.

In order to establish a baseline for cycle time, autonomous trials were run with point-to-point trajectories. The end-points of the autonomous trajectories were taken from real operation data, so that they mimic the variation of the pile depth and location. The trajectories were optimized offline to minimize the time taken and the control effort required to complete the task.

The angle of the swing function for three different sample trials is shown in Fig. 5.9, with vertical markers at the end of each cycle, i.e., when the machine reaches subgoal 6. For this operator, the BSC controller clearly reduces cycle time, but this effect was not universal across all operators. In the bottom subplot, the

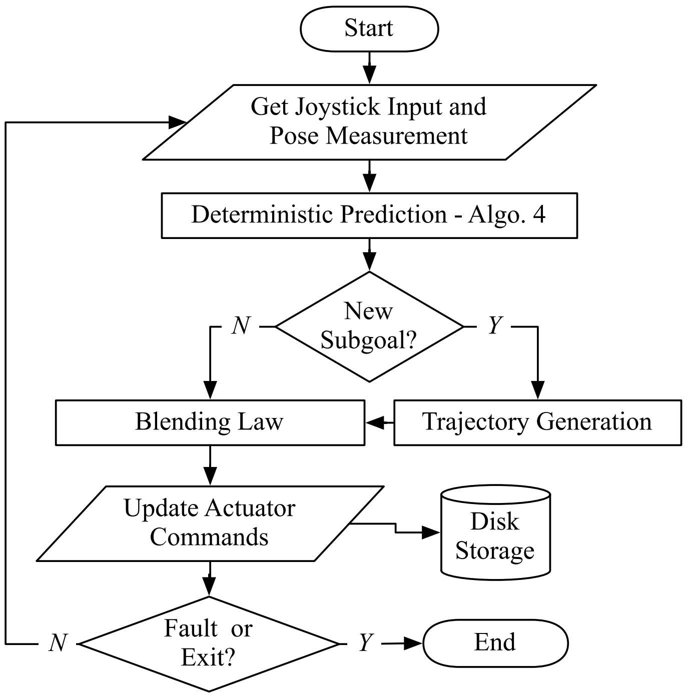


Figure 5.8: Logical overview of BSC software

autonomous controller demonstrates a fairly consistent cycle time, nearly half the length of the manual and blended architectures.

Occasionally during the trials, an operator would bypass a termination set. On one hand, defining the termination sets in the actuator space eliminates the need for computationally expensive forward kinematics in real-time. However, since the position of the tip of the bucket is not uniquely defined by the actuator displacements, operator subgoals in the end-effector workspace may not be unique in the actuator space. This means that the distribution of subgoals locations in the actuator space of the training data may be multi-modal.

Cycle times for manual, blending, and autonomous modes are shown in Fig. 5.10. Blending does not have a consistent effect on the cycle time, which suggests that this advantage may vary with the operator style and experience.

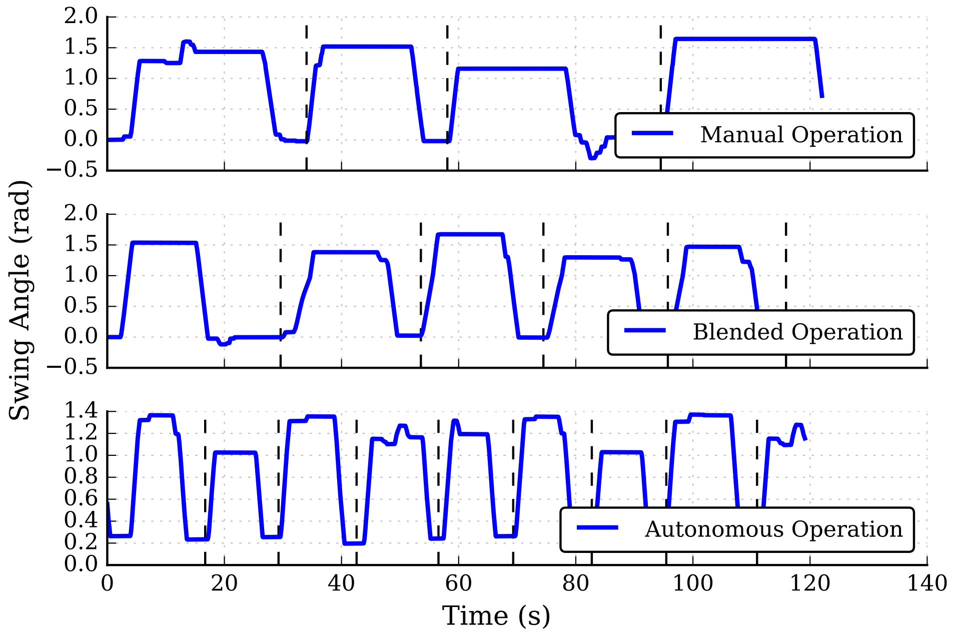


Figure 5.9: Sampled data for the swing actuator in the three operating modes. Vertical dotted lines denote the end of a cycle, i.e., the end of subgoal 6.

The cycle efficiency is defined as the mass of material moved per cycle time, for each task cycle performed. An increase in this cycle efficiency was observed for each operator under BSC. Fig. 5.11 shows the effect of blending on cycle efficiency. Fig. 5.12 illustrates the effect of BSC on cycle time and cycle efficiency across all operators. The performance of BSC is superior in both metrics, and the cycle time distribution is notably more narrow.

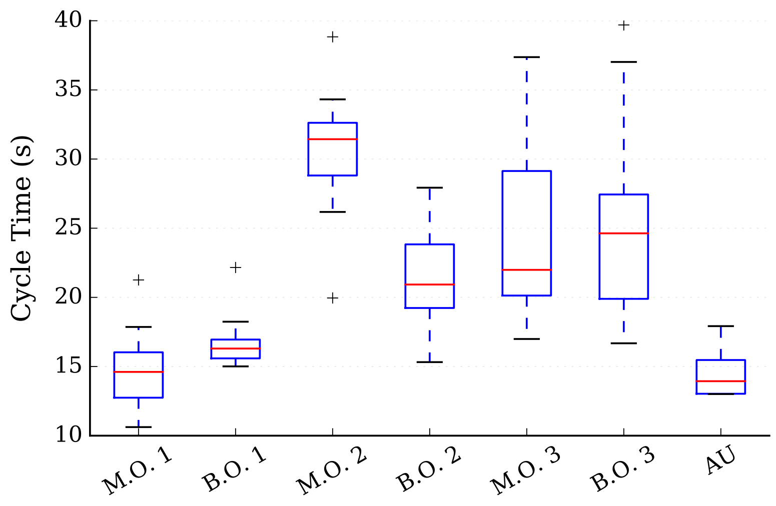


Figure 5.10: The effect of blending on cycle time varies with the operator style and experience. (M: Manual, B: Blended, O: Operator, AU: Autonomous)

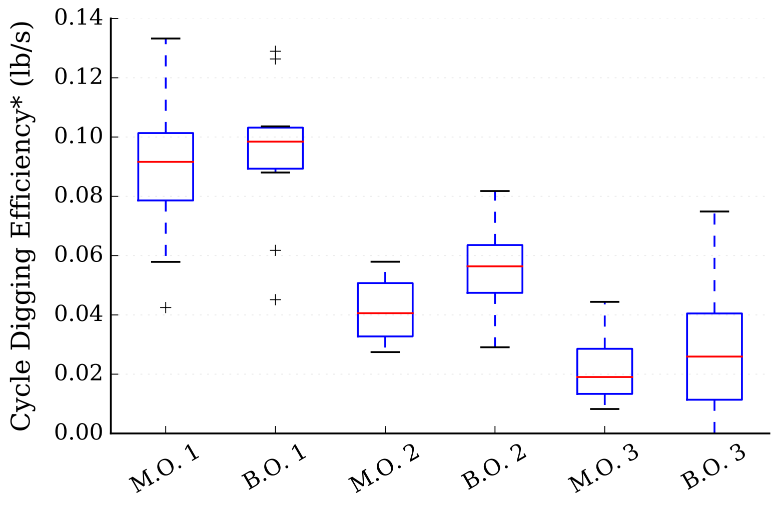


Figure 5.11: Comparison of cycle efficiency for different operators and operating modes. Cycle efficiency is measured as the mass of material moved in a cycle divided by the cycle time. Higher efficiency is desirable. (M: Manual, B: Blended, O: Operator)

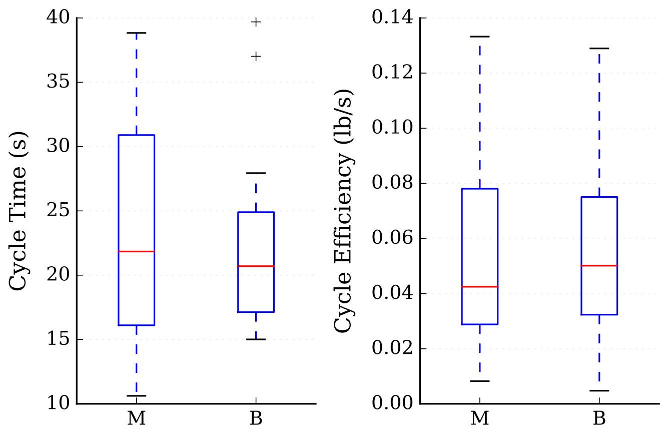


Figure 5.12: Comparison of cycle time and cycle efficiency across all operators. (M: Manual, B: Blended)

Operators were able to become comfortable fairly quickly with the blended control scheme, with most users practicing only a few minutes. Some considerable task specific advantages were observed and reported during the trials. In particular, during a swing operation, which in this case corresponds to shifting operator focus from the pile to the truck, adjustments of the bucket position and stick actuator allowed operators to be instantly prepared for the ensuing dump or dig operation. The effect of these adjustments is that each dig operation can be initiated from the same state. Also, the material in the bucket was notably better secured after the dig operation with blending enabled. The detailed parameters of these experiments are provided in Table 5.1, and a more thorough listing of the kinematic characteristics of the scale experimental excavator is provided in the tables of Appendix A.

Task Learning	Subgoal Prediction
DP-GMM*	FSM-P
6 subgoals	$T_k : \{\lvert\mu_i - \zeta_i\rvert \leq 2\sqrt{\Sigma_{ii}}\}^*$
$\boldsymbol{\mu}_{1:K}, \Sigma_{1:K}$	I_k: 30% of full range*
	$\alpha_0 = 0$

Table 5.1: Experimental parameters for learning and prediction. The trials were run prior to the development of the changepoint filtering process, so a set of points were selected from the data and clustered using a DP-GMM. The termination sets were defined by the mean of the subgoal distribution plus or minus 2 standard deviations on each actuator, i.e., $2\sqrt{\Sigma_{ii}}$. The initiation sets were identified by observing the common movements between subgoals and applying a threshold on those actuators in the subgoal direction.

6. SUMMARY AND FUTURE WORK

A series of methods have been presented for designing shared control architectures for hydraulic excavator equipment. The relevant academic literature was presented, highlighting some of the historical challenges in automating hydraulic earth-moving equipment and some of the promising new techniques for learning and assisting in manipulation tasks. The forward and inverse kinematics of an excavator manipulator with fixed tracks have been developed, along with the dynamic equations from the valve spool position to the equations of motion of the excavator manipulator. Next, three algorithms for learning task representations from demonstration data have been detailed and compared. Those same task representations can then be used to propose methods for on-line prediction of operator subgoal. Given a predicted subgoal, smooth point to point trajectories can be generated and tracked by a low-level controller. The magnitude of assistance provided is mediated by a blending law, which is simply a weighted average of the operator and controller inputs, for which the weighting can be static or dynamic. Linking a dynamic blending parameter to a probabilistic measure of confidence from our prediction step has the potential to provide an intelligent and situationally aware control assistance. Finally, an integrated BSC system with a deterministic prediction component was implemented on a scale model excavator, and tested with human operators, showing improvements in digging efficiency on each operator and cycle time on some operators.

Some dynamic characteristics of hydraulic actuation were given less emphasis in this project, for a few key reasons: the scale model hydraulic system is significantly less complex than a full-scale excavator system, and certain traits and limitations of the scale-model are not present or already addressed in commercial equipment.

However, in order to optimize the benefits of a shared control approach, future experiments may need to place more strict requirements on the tracking performance of low-level controllers. In such a case, the dynamic equations may be useful in developing and testing more advanced controllers for both the scale model and future experimental platforms.

More trials with various operator experience levels are required to determine whether one representation of the task is sufficient for assisting many different operators. Future iterations of the control approach should also address the possibility of subgoal locations moving during the task execution. However, care should still be taken to distinguish *learning the task* from *locating the subgoals*, so that the latter step (locating) can exploit the ordering of subgoals and fixed structure of the former objective (learning). That said, the motivation for learning to be performed off-line was largely computational, and the issues of changing subgoal locations and learning completely new tasks could possibly be better addressed by developing new learning techniques which can be executed on-line.

Each prediction scheme is conditioned on knowledge of the last subgoal visited by the operator. This is addressed by defining a region about each subgoal in which that subgoal can be confirmed, as in the definition of termination sets. However, it is much more difficult to specify in real-time *when* or *where* specifically the subgoal was reached or the operator switched to a new subgoal. Early attempts at solving this problem have used complex logic and computed velocities and accelerations, but will need to be refined for implementation.

Of note, the dynamic blending derivations of Sec. 4.3 were not tested on the physical apparatus, but only in the virtual wireframe environment. It remains to be seen how various operators will respond to dynamic blending, particularly with different bounds on the blending parameter and different action thresholds. Unfortunately, there are few generally accepted metrics for evaluating performance in shared control domains, and researchers almost exclusively use specific task performance metrics (e.g., digging efficiency). Since operator receptiveness is a crucial component in the success of shared control technologies, analyses will need to be developed which quantify the interaction between the human and computer agent at a more fundamental level over large trials. Through operator feedback and careful experimentation, the effectiveness and responsiveness of blended shared control can be improved even further in hydraulic excavation tasks, and potentially reach new application domains.

REFERENCES

[1] C. T. Jarhen, "Transportation construction equipment," in *Transportation in the New Millenium*, National Academy of Science, Engineering and Medicine, 2000.

[2] G. V. Research, "Earthmoving equipment market analysis by product (excavators, loaders), by application (construction, mining) and segment forecasts to 2022," Feb 2016.

[3] D. A. Bradley, D. W. Seward, J. E. Mann, and M. R. Goodwin, "Artificial intelligence in the control and operation of construction plant-the autonomous robot excavator," *Automation in Construction*, vol. 2, no. 3, pp. 217–228, 1993.

[4] P. S. Rowe, *Adaptive Motion Planning for Autonomous Mass Excavation.* PhD thesis, Carnegie Mellon University, Pittsburgh, PA, USA, 1999. AAI9936891.

[5] R. Parasuraman, T. B. Sheridan, and C. D. Wickens, "A model for types and levels of human interaction with automation," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 30, no. 3, pp. 286–297, 2000.

[6] H. E. Merritt and V. Pomper, *Hydraulic Control Systems*, vol. 35. John Wiley & Sons, 1968.

[7] N. Manring, *Hydraulic Control Systems.* John Wiley & Sons, 2005.

[8] P. H. Chang and S.-J. Lee, "A straight-line motion tracking control of hydraulic excavator system," *Mechatronics*, vol. 12, no. 1, pp. 119–138, 2002.

[9] N. Sepehri, P. D. Lawrence, F. Sassani, and R. Frenette, "Resolved-mode teleoperated control of heavy-duty hydraulic machines," *Journal of Dynamic Systems, Measurement, and Control*, vol. 116, no. 2, pp. 232–240, 1994.

[10] P. Vähä and M. Skibniewski, "Dynamic model of excavator," *Journal of aerospace engineering*, vol. 6, no. 2, pp. 148–158, 1993.

[11] A. Koivo, M. Thoma, E. Kocaoglan, and J. Andrade-Cetto, "Modeling and control of excavator dynamics during digging operation," *Journal of Aerospace Engineering*, vol. 9, no. 1, pp. 10–18, 1996.

[12] T. V. Alekseeva, *Machines for earthmoving work: theory and calculations*, vol. 30. Intl Public Service, 1985.

[13] M. R. Sirouspour and S. Salcudean, "On the nonlinear control of hydraulic servo-systems," in *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, vol. 2, pp. 1276–1282, IEEE, 2000.

[14] M. R. Sirouspour and S. E. Salcudean, "Nonlinear control of hydraulic robots," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 2, pp. 173–182, 2001.

[15] R. Fales, *Robust control design for an electrohydraulic wheel loader system with a human-in-the-loop assessment in a virtual environment.* PhD thesis, Iowa State University, Ames, IA, USA, 2004.

[16] R. Fales and A. Kelkar, "Robust control design for a wheel loader using hâĹđ and feedback linearization based methods," *ISA transactions*, vol. 48, no. 3, pp. 312–320, 2009.

[17] M. H. Raibert and J. J. Craig, "Hybrid position/force control of manipulators," *Journal of Dynamic Systems, Measurement, and Control*, vol. 102, no. 127,

pp. 126–133, 1981.

[18] N. Hogan, "Impedance Control: An Approach to Manipulation," *American Control Conf., 1984*, vol. 107, no. March 1985, pp. 304–313, 1984.

[19] F. Ha., Q. P., Nguyen Q. H.; Rye D. C. & Durrant-Whyte, "Impedance control of a hydraulically-actuated robotic excavator," *Journal of Automation in Construction*, vol. 9, no. 5, pp. 421–435, 2000.

[20] S. E. Salcudean, K. Hashtrudi-Zaad, S. Tafazoli, S. P. DiMaio, and C. Reboulet, "Bilateral matched-impedance teleoperation with application to excavator control," *IEEE Control Systems Magazine*, vol. 19, no. 6, pp. 29–37, 1999.

[21] S. Tafazoli, S. E. Salcudean, K. Hashtrudi-Zaad, and P. D. Lawrence, "Impedance control of a teleoperated excavator," *IEEE Transactions on Control Systems Technology*, vol. 10, no. 3, pp. 355–367, 2002.

[22] T. B. Sheridan and W. R. Ferrell, *Man-Machine Systems.* MIT, 1974.

[23] T. B. Sheridan, *Telerobotics, Automation, and Human Supervisory Control.* MIT Press, 1992.

[24] M. Itoh, T. L. Gibo, and E. R. Boer, "IEEE systems, man, and cybernetics technical committee on shared control," 2016.

[25] M. Mulder, D. A. Abbink, and T. Carlson, "Introduction to the Special Issue on Shared Control: Applications," *Journal of Human-Robot Interaction*, vol. 4, no. 3, p. 1, 2015.

[26] C. J. Kessel, C. D. Wickens, and M. Shelves, "Transfer of failure-detection skills between monitoring and controlling," *Human Factors*, vol. 24, no. 1, pp. 49–60, 1982.

[27] D. B. Kaber and M. R. Endsley, "Out-of-the-loop performance problems and the use of intermediate levels of automation for improved control system functioning and safety," *Process Safety Progress*, vol. 16, no. 3, pp. 126–131, 1997.

[28] B. Lorenz, F. Di Nocera, S. Röttger, and R. Parasuraman, "The effects of level of automation on the out-of-the-loop unfamiliarity in a complex dynamic fault-management task during simulated spaceflight operations," *Proceedings of the Human Factors and Ergonomics Society*, vol. 28, pp. 44–48, 2001.

[29] J. S. McCarley and C. D. Wickens, *Human factors implications of UAVs in the national airspace.* University of Illinois at Urbana-Champaign, Aviation Human Factors Division, 2005.

[30] D. B. Kaber and M. R. Endsley, "The effects of level of automation and adaptive automation on human performance, situation awareness and workload in a dynamic control task," *Theor. Issues in Ergon. Sci*, no. January, pp. 1–40, 2003.

[31] M. D. Elton and W. J. Book, "Comparison of human-machine interfaces designed for novices teleoperating multi-dof hydraulic manipulators," pp. 395–400, 2011.

[32] R. C. Winck, M. Elton, and W. J. Book, "A practical interface for coordinated position control of an excavator arm," *Automation in Construction*, vol. 51, no. C, pp. 46–58, 2015.

[33] B. Domingue, "Coordinated rate control user interface and task identification of an excavator," Master's thesis, Georgia Institute of Technology, Atlanta, GA, USA, 1 2017.

[34] Komatsu, "Intelligent machine control," 2014.

[35] P. Aigner and B. McCarragher, "Contrasting potential fields and constraints in a shared control task," in *Intelligent Robots and Systems, 1997. IROS '97.,*

Proceedings of the 1997 IEEE/RSJ International Conference on, vol. 1, pp. 140–146 vol.1, Sep 1997.

[36] A. Poncela, C. Urdiales, E. J. Perez, and F. Sandoval, "A new efficiency-weighted strategy for continuous human/robot cooperation in navigation," *IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans*, vol. 39, no. 3, pp. 486–500, 2009.

[37] A. R. Enes, *Shared control of hydraulic manipulators to decrease cycle time*. PhD thesis, Georgia Institute of Technology, Atlanta, GA, USA, 2010.

[38] J. G. Storms and D. M. Tilbury, "Blending of human and obstacle avoidance control for a high speed mobile robot," in *2014 American Control Conference*, pp. 3488–3493, June 2014.

[39] V. Dimitrov and T. Padır, "A shared control architecture for human-in-the-loop robotics applications," in *IEEE International Symposium on Robot and Human Interactive Communication*, pp. 1089–1094, Aug 2014.

[40] A. D. Dragan and S. S. Srinivasa, "A policy-blending formalism for shared control," *The International Journal of Robotics Research*, vol. 32, no. 7, pp. 790–805, 2013.

[41] A. Y. Ng and S. Russell, "Algorithms for inverse reinforcement learning," in *in Proc. 17th International Conf. on Machine Learning*, pp. 663–670, Morgan Kaufmann, 2000.

[42] B. Michini, T. J. Walsh, A.-A. Agha-Mohammadi, and J. P. How, "Bayesian nonparametric reward learning from demonstration," *IEEE Transactions on Robotics*, vol. 31, no. 2, pp. 369–386, 2015.

[43] M. V. Mark W. Spong, Seth Hutchinson, *Robot Modeling and Control.* John Wiley & Sons, Inc., 2006.

[44] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*, vol. 1. MIT Press Cambridge, 1998.

[45] B. Michini, T. J. Walsh, A. A. Agha-Mohammadi, and J. P. How, "Bayesian nonparametric reward learning from demonstration," *IEEE Transactions on Robotics*, vol. 31, pp. 369–386, April 2015.

[46] B. A. Frigyik, A. Kapila, and M. R. Gupta, "Introduction to the Dirichlet Distribution and Related Processes," Tech. Rep. 206, 2010.

[47] H. Maske, E. Kieson, G. Chowdhary, and C. Abramson, "Can Co-robots Learn to Teach?," tech. rep., University of Illinois Urbana-Champagne, Department of Agricultural and Biological Engineering, 11 2016.

[48] B. Kulis and M. I. Jordan, "Revisiting k-means: New algorithms via Bayesian nonparametrics," *arXiv preprint arXiv:1111.0352*, 2011.

[49] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[50] A. Genz, "Numerical computation of multivariate normal probabilities," *Journal of Computational and Graphical Statistics*, vol. 1, no. 2, pp. 141–149, 1992.

[51] H. Maske, G. Chowdhary, and P. Pagilla, "Intent aware shared control in off-nominal situations," in *Conference on Decisions and Control*, (Las Vegas, USA), IEEE, 2016. accepted.

[52] A. Enes and W. Book, “Blended shared control of zermelo’s navigation problem,” in *American Control Conference (ACC), 2010*, pp. 4307–4312, IEEE, 2010.

[53] M. Kageyama and N. Nagahashi, “Cylinder stroke position measurement device,” July 20 2010. US Patent 7,757,547.

[54] “String potentiometer kit.” https://www.andymark.com/product-p/am-2618.htm. Accessed: 2016-06-28.

APPENDIX A

KINEMATIC AND ACTUATOR PARAMETERS OF THE REDUCED-SCALE MODEL

Parameter	Value	Units	Parameter	Value	Units
a_1	3.7	cm	$r_{2,E}$	7	cm
a_3	25.3	cm	$r_{3,D}$	33	cm
a_2	48.2	cm	$r_{3,F}$	4	cm
a_4	11.5	cm	$r_{3,G}$	4.7	cm
$r_{B,C}$	5.7	cm	$r_{4,G}$	12.8	cm
$r_{D,E}$	9.4	cm	$^*r_{\text{cyl},1}$	16.3	cm
$r_{D,F}$	29	cm	$^*r_{\text{cyl},2}$	19.4	cm
$r_{E,F}$	21.2	cm	$^*r_{\text{cyl},3}$	15.55	cm
$r_{G,H}$	6	cm	$\theta_{1,2,C}$	0.483	rad
$r_{F,H}$	6	cm	$\theta_{2,3,D}$	0.0870	rad
$r_{1,A}$	7.8	cm	$\theta_{B,1,2}$	0.399	rad
$r_{1,B}$	22	cm	$\theta_{D,2,3}$	2.78	rad
$r_{1,C}$	27.5	cm	$\theta_{D,F,E}$	0.212	rad
$r_{2,B}$	29.2	cm	$\theta_{G,3,4}$	1.659	rad
$r_{2,C}$	26.7	cm	l_{c4}	76.42	cm
$r_{2,D}$	8.1	cm	l_{c2}	24.1	cm
$r_{2,F}$	21.3	cm	l_{c3}	9.62	cm

Table A.1: Kinematic parameters of the reduced-scale excavator model; $^*r_{\text{cyl},1}$ denotes the length of the hydraulic cylinder (excluding the rod extension length), all other parameters are consistent with Sec. 2.2.

Parameter	Value	Units
Boom		
$\dot{\zeta}_{1,\max}$	1.4	cm/s
$\ddot{\zeta}_{1,\max}$	1.2	cm/s^2
u_{db}	1.3	$PWM\ \%$
Stick		
$\dot{\zeta}_{2,\max}$	2.7	cm/s
$\ddot{\zeta}_{2,\max}$	1.2	cm/s^2
u_{db}	1.7	$PWM\ \%$
Bucket		
$\dot{\zeta}_{3,\max}$	2.4	cm/s
$\ddot{\zeta}_{3,\max}$	3.2	cm/s^2
u_{db}	1.7	$PWM\ \%$
Swing		
$\dot{\zeta}_{4,\max}$	0.85	rad/s
$\ddot{\zeta}_{4,\max}$	1.7	cm/s^2
u_{db}	0.6	$PWM\ \%$

Table A.2: Measured actuator parameters; note that u_{db} here is the deadband as a percentage of PWM duty cycle, as in Fig. 5.2, and $\dot{\zeta}_{i,\max}$ and $\ddot{\zeta}_{i,\max}$ are the maximum velocity and acceleration, respectively, of the i^{th} actuator.