Maneuver Based Design of Passive Assist Devices for Active Joints

by

William Robert Brown

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Mechanical Engineering)
in the University of Michigan
2013

A C K N O W L E D G M E N T S

This research was supported in part by the Ground Robotics Reliability Center at the University of Michigan, with funding from government contract DoD-DoA W56H2V-04-2-0001 through the US Army Tank Automotive Research, Development, and Engineering Center.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF APPENDICES

LIST OF ABBREVIATIONS

DAQ data acquisition board

DCMC DC motor and controller

dFVCF derivative of Force Velocity Curve Fit

dFVCF+ weighted derivative of Force Velocity Curve Fit

DOF Degree of Freedom

FDCF Force Displacement Curve Fit

FDCF+ weighted Force Displacement Curve Fit

NBD non-backdrivable

PAD Passive Assist Device

SB statically balanced

UGV Unmanned Ground Vehicle

COM center of mass

ABSTRACT

Maneuver Based Design of Passive Assist Devices for Active Joints

by

William Robert Brown

Chair: A. Galip Ulsoy

This thesis describes a novel, general methodology for designing a Passive Assist Device (PAD) (e.g., spring) to augment an actuated system using optimization based on a known maneuver of the active system. Implementation of the PAD can result in an improvement in system performance with respect to efficiency, reliability, and/or safety. The methodology is experimentally demonstrated with a parallel, torsional spring designed to minimize energy consumption of a prototypical, single link Unmanned Ground Vehicle (UGV) robot arm. The method is extended to series systems as well as dual PAD systems that contain both a series and a parallel component. We show that the proposed method is not limited to robot manipulator joints, can be applied to multi-DOF systems, and can be used to design PADs that are robust against variation in the maneuver. Furthermore, for certain situations a significant increase in performance can be realized if the maneuver is redesigned considering that a PAD will be added to the system. The addition of properly designed energy minimizing springs can lead to a decrease in energy consumption, as shown in various engineering examples, by as much as 60–80% while also improving reliability and/or safety.

CHAPTER 1

Introduction

This thesis presents a novel method of achieving energy efficiency by augmenting active components (e.g., motors) with optimized Passive Assist Devices (PADs) (e.g., springs). The method considers the loading profile of the active system over the course of a known trajectory and determines the design of the passive element that would best counteract those forces or velocities. If this passive device were added to the active-only system the energy required to execute the maneuver (i.e., trajectory and loading) would be reduced. Furthermore, implementing such a device could also improve reliability, utility, and/or safety of the original system.

1.1 Motivation and Potential Impact

The potential economic impact of this proposed method is large. Today there are over 190,000 industrial robots operating in the United States and each typically consumes over 300 kWh per day [6] corresponding to \$1.5 billion annually. A mere 1% improvement in efficiency yields the following annual benefits:

- A reduction in energy consumed by 200,000,000 kWh.
- Electrical cost reduction of \$15,000,000.
- Greenhouse gas emissions reduced by 140,000 tons [7].
- SO_2 and NO_x emissions reduced by 720 tons [7].

Implementing a parallel PAD designed with our approach has experimentally reduced the energy required for a single link robot arm to execute a simple lifting maneuver by 25% [8] and in simulation shown savings as high as 79% for multilink robot arms executing typical maneuvers.

Furthermore, every year 45 billion kWh are spent in the fabricated metal products sector (e.g., machining, stamping) in the United States [9] with an annual electric cost of \$3.15 billion. This quantity not only includes the cost of directly shaping each work piece, but also auxiliary costs including positioning the tool and/or workpiece, waste material collection, cooling, etc. Because the energy costs are so large a great deal of work has gone into making the entire manufacturing process more efficient. This can be done by streamlining the entire manufacturing system as well as by making each individual operation more efficient [10, 11].

Positioning machinery is particularly well suited to being augmented with a PAD because, in a mass production environment, they undergo the same periodic motions thousands or millions of times. Positioning costs can vary dramatically depending on the operation. If there are no external forces, positioning may only account for 4% of energy consumption [11]. However, if external forces are large (e.g., moving the workpiece while milling), positioning costs could exceed 60% [12]. Furthermore, the faster these processes occur the larger the energy cost [13]. Our approach can reduce these losses in positioning manufacturing machinery by up to 79% [14].

The benefits of our approach may not be realized for all motions on all machines. However, for machines repeatedly executing identical or similar maneuvers, an optimized PAD could provide a substantial improvement in performance. Furthermore, in many instances it may be possible to redesign the trajectory of the workpiece (and/or tool) in order to fully take advantage of a PAD without affecting the final product.

Passive assist devices may also be designed to improve the performance of UGVs. Unlike traditional industrial robots, which perform preprogrammed tasks in a well-structured environment and are among the most reliable machines available, the current generation of UGVs exhibit a mean time between failure on the order of 10 hours. UGVs are machines with numerous actively controlled degrees of freedom, and joint or motor problems are a common failure mode. This is in part due to the varied environments in which they are asked to operate (e.g., hot sandy deserts vs. carpeted floors in an air conditioned building), in part due to operators using them in ways they are not designed for, and in part because of new unproven technologies (e.g., sensor hardware and AI software) [15]. In a UGV manipulator, a passive energy storage device (i.e., spring) can be placed in parallel with active devices (i.e., motors) and designed to reduce the maximum motor torques required for a particular trajectory, thereby increasing efficiency and reliability. A series spring may also be used to augment a manipulator joint in order to improve efficiency. The addition of compliance from a series spring also improves the safety of the system and can improve reliability by reducing shock or impact loads. Finally, if the entire joint powertrain is redesigned for the addition of a PAD, the same utility can be achieved with a smaller motor. This will further reduce cost and weight making the system even more efficient.

On a higher level, the concept of a PAD is similar to that of using a battery in a hybrid electric vehicle. An internal combustion engine in a traditional vehicle has to be large enough to execute the most extreme tasks within the drive cycle but much of the time it is being underused (e.g., to maintain constant speed). The addition of the battery and electric motor effectively level the demands of the drive cycle, allowing for the engine to operate at its "sweet spot" where efficiency is high and allowing for a significantly smaller engine to be used [16, 17]. The use of a smaller engine reduces cost and further improves efficiency. In this research, the PAD is designed for a particular trajectory and load (analogous to a vehicle drive cycle) such that the required energy consumption is minimized. Another example of a PAD being used to improve efficiency and reduce active component size would be the case of a water tower [18]. To satisfy peak consumer water demand a very large pump would be required. By adding a water tower to the distribution system, pumps only need to be large enough to supply the average load and can therefore be much smaller.

In the case of the water tower and hybrid electric systems the energy storage devices are used to level demand so a smaller energy source can be used. Energy storage systems are being considered to do the opposite in the case of renewable energy sources. Power generated by both solar and wind systems tend to vary much more than consumer demand, thus requiring that excess power be wasted and having traditional power plants as backup. In order for wind and solar to become a larger part of electricity generation there needs to be an accompanying increase in energy storage that can take the variable power generation and supply a more constant amount of power to the consumer [19, 20, 21].

1.2 Passive Assist Device Overview

1.2.1 Fundamental Concept

An actuated system can be augmented with a passive device in two fundamentally different ways: a parallel configuration and a series configuration (see Fig. 1.1). If the designer has knowledge about the external forces being applied to the system or the trajectory the system will execute, the passive device in either configuration could improve the performance of the system from an energy standpoint. However, the different configurations are best suited for different tasks. A series configuration can assist the actuator by reducing the actuator velocity but has no effect on the required actuator forces. A parallel configuration does the opposite in that the required actuator forces can be reduced but the actuator velocities remain unchanged.

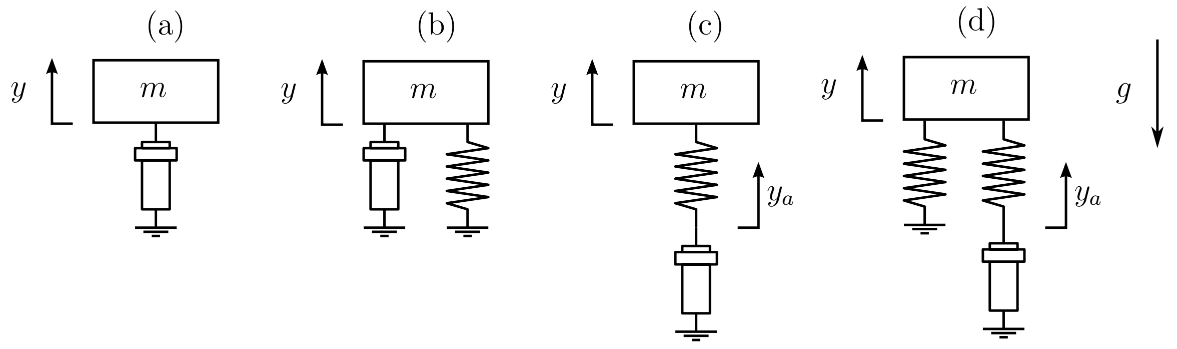


Figure 1.1: Depicts an unsprung, actuated mass (a), the same system augmented with a parallel spring (b), the same system as (a) augmented with a spring in a series configuration (c), and the system augmented with a dual spring configuration (d).

Consider a set of simple mass-spring-actuator systems as depicted in Fig. 1.1. In maneuvers where the system is operating about a steady velocity and the primary energy losses are due to inertial forces, a series configuration can provide a greater increase in performance than a parallel configuration. Conversely, in maneuvers where the energy losses are primarily caused by overcoming external forces a parallel configuration can provide significant advantages by balancing these forces, while the series configuration cannot provide much (if any) assistance. In the limit where the system is executing a harmonic maneuver either configuration could be used to eliminate ideal actuator power consumption. As a maneuver becomes more arbitrary, the corresponding optimal spring design will have a spring constant that approaches zero for a parallel system or infinity for a series system, showing that the best design for a completely arbitrary maneuver is a system with no spring. In Chap. 3 we explore which design (i.e., parallel, series, or dual) offers superior potential performance with respect to energy savings.

The concept of using a passive device to assist an actuated system is similar to hybrid electric vehicles, where a battery and motor are used to supplement an internal combustion engine and to maintain its operation at its “sweet spot” where efficiency is high. Here, the PAD is designed for a particular trajectory and load (analogous to a vehicle drive cycle) such that the required energy is reduced. While most automated systems (e.g., robot arms, machine tools) perform a variety of tasks, such tasks can often be restricted to a family of similar trajectories and loads. Consequently, properly designed passive assist systems can significantly improve performance.

For the purposes of PAD design, the translational systems shown in Fig. 1.1 are perfectly analogous to the rotational joints analyzed elsewhere in the thesis.

1.2.2 Literature Review

Most of the literature on relevant devices containing spring loaded joints come from the robotics community and can be broadly divided into two categories depending on their function: static balancing and biomechanics/walking robots.

Much of the energy used to manipulate a robot arm is spent overcoming the weight of the arm itself. Parallel springs (or countermasses) can be added to an existing manipulator in such a way that the potential energy of the arm is constant regardless of configuration. This results in an arm that is effectively weightless, drastically reducing peak torques and energy consumption. A system designed this way is considered to be statically balanced [22, 23]. Applications for static balancing include industrial robots [24, 25], exoskeletons [26, 27, 28, 29], as well as non-robotic devices including an architect's desk lamp, a garage door assembly, and even a pressurized pneumatic piston to assist the hydraulic lift system on the boom assembly of earth-moving loading machines [30].

Although our design method cannot make a robot arm effectively weightless over its entire workspace, it provides two major advantages over static balancing designs. There are many ways to achieve a statically balanced system but they are difficult to implement on an existing robot arm because at least one of the following mechanisms are required: nonlinear springs, additional masses [31], extension springs attached to external points on the arm [32], additional links [33], cables running through the arm, or noncircular pulleys [34]. Our method only adds a PAD (e.g., torsion spring) at a joint, thereby simplifying the design and implementation without compromising certain regions of the workspace with potential hardware collisions. Our method also has the potential to outperform a statically balanced system because it considers the dynamics and external loading that define a typical maneuver in addition to the weight distribution and kinematics of the arm itself.

A series PAD cannot be used for static balancing applications but can effectively improve energy efficiency in systems executing repeated dynamic motions. Animal skeletal muscle systems are well modeled by springs acting in parallel (internal to muscle tissue) and in series (tendons) to linear actuators (muscles) [35]. Therefore it is logical that biologically inspired robots often make use of springs. Walking gaits are, by definition, cyclic and sufficiently harmonic such that either a parallel or a series configuration can be effectively used to reduce the kinetic energy losses.

For example, there are many different series PAD designs that reduce the kinetic energy losses in walking robots. These are often designed by attempting to mimic biologically equivalent systems [36, 37] or by choosing the PAD stiffness such that the natural frequency is typical of efficient natural gaits [38]. By contrast, we consider a prescribed maneuver (e.g., gait) and design a PAD that minimizes the energy required to execute such a maneuver without any of the approximation of other methods and without altering the final maneuver.

Vanderborght provides an overview of variable stiffness series elastic actuators [39]. Much like non-linear parallel PADs, variable stiffness elements have the potential to outperform a linear stiffness device but are more complicated to design, fabricate and implement. Series PADs can also be designed to reduce backlash and shock loads while maintaining a necessary degree of control bandwidth [40]. These objectives are different from energy minimization but elements of this design approach could be incorporated into our methodology in the form of optimization constraints.

There are four papers of particular interest to our research. The robot ERNIE [41] uses springs acting parallel to the knee joints to reduce average power consumption. The design of ERNIE's springs is a three step process: (a) optimizing the walking gait for a variety of spring stiffnesses and gait speeds, (b) selecting commercially available springs with stiffnesses corresponding to low-cost walking, and (c) simultaneously optimizing the gait and spring offset for different speeds and the selected spring stiffness. By contrast, we simultaneously optimize the spring stiffness and preload based on a known maneuver in one step.

Harper designed a robotic fish that makes use of a series spring configuration to reduce energy consumption while swimming by aligning the phase of the harmonic tail velocity with the harmonic external hydrodynamic forces [42]. We consider a similar design approach except the trajectory and forcing we are considering are more general than out of phase harmonic oscillations.

Mettin et al describe a parallel spring design process for augmenting active joints based on a prescribed trajectory and external forcing [43]. The spring design is a Force Displacement Curve Fit (FDCF) — a linear least squares approximation to the required force-displacement data. Adding a spring designed this way to a system performing this maneuver will reduce the required effort of the actuator, resulting in an increase in efficiency. Their approach differs from ours primarily in the fact that their spring design is not based on energy minimization.

There are four main advantages of our proposed optimization approach over the current, state-of-the-art, FDCF design approach [43]:

1. **Guarantee of Superior Performance:** The optimization approach is guaranteed to perform at least as well as the FDCF design. For certain maneuvers incorporating an FDCF designed spring can actually increase energy consumption. An energy minimizing design will, at worst, provide no energy savings. Furthermore, because the problem is decoupled prior to optimization in the design methodology, optimization can be done quickly and cheaply even in cases where gains in performance are small [44].

2. **Flexible Passive Assist Device Topology:** The optimization approach can be used to find designs for a wide variety of non-linear PADs. The curve-fit approach can also find non-linear designs but is practically restricted to a few common classes of functions (e.g., polynomials).

3. **Flexible Objective Function:** It is convenient that a FDCF design often does a good job of reducing motor energy consumption. However, if the design goals were to change (e.g., minimize peak power) the objective function in the optimization problem can easily be changed to match; yielding a new design. Furthermore, when considering a distribution of possible maneuvers, the optimization approach can be easily modified to find robust designs that exhibit less variance in a metric of interest. This last advantage will allow us to find designs that are robust with respect to changes in trajectories and loads [45].

4. **Limited to Feasible Designs:** A FDCF can be used to fit any data set and can have a negative slope. However, linear springs cannot have negative stiffness. Optimization can enforce this condition, ensuring a feasible PAD.

The maneuver-based PAD design process described by Mettin is the state-of-the-art but it is not necessarily a competitor to the approach presented in our research; it can be complementary and used to initialize the optimization step in our design process.

Wang et al discuss the possible uses of springs in exoskeleton design. They also provide a comparison of the effects of adding parallel and series springs in simulation [46]. They use a FDCF design for their parallel springs and a peak power minimizing design for their series springs. By contrast we derive a series analog to FDCF (dFVCF), develop weighted curve fits, and compare energy minimizing springs for both parallel and series designs (see Chap. 3).

1.2.3 Non-energy Benefits of Passive Assist Devices

While either a parallel or a series PAD has the potential to improve energy efficiency, implementation of a PAD provides other benefits that are specific to each class. Reliability problems in electric motors are often associated with high temperatures [47, 48] which are caused by high current operation and current is proportional to the required torque. Therefore, implementation of a parallel PAD can improve motor reliability by reducing required torques. Because series PADs have no effect on required torque they cannot be used to improve reliability problems based on temperature. On the other hand, if the powertrain is exposed to impact loading, only a series PAD can be used to effectively absorb shocks.

Safety is a major concern in industrial robotics where it is possible that a human and a robot arm could share the same workspace. Industrial robot arms are powerful, massive machines and if they were to impact a human operator there is potential for injury or death. A number of techniques have been developed to reduce these risks including isolating the robot [49], smart control systems that quickly stop the arm when an unexpected collision is detected [50], or compliant joints that are nominally stiff but are designed to become very soft if a torque threshold is exceeded [51]. An energy minimizing series PAD designed with a maneuver based methodology will increase the performance of a robot arm with respect to human safety. A parallel PAD would have no effect on collision safety.

The severity of a human-robot collision is roughly proportional to the change in velocity of the human, ΔV, as a result of the collision [52, 53, 50]. A large robot arm with a well designed motor position controller and a very stiff joint will behave as if it were much much more massive than a person during the collision. Based on a momentum analysis of a linear collision between a moving robot arm and a stationary human, the resulting change in human speed is found to be:

$$\Delta V = 2V_a \tag{1.1}$$

Where V_a is the velocity of the robot mass immediately before impact. If the same robot had a very soft spring between the motor and the joint the resulting collision would result in a change of human speed of:

$$\Delta V = \frac{2V_a m_a}{m_a + m} \tag{1.2}$$

Where m_a is the mass of the robot between the spring and the collision point and

m is the mass of the human. Therefore, the maximum decrease in human ΔV due to a series PAD designed based on our methodology is $\frac{m}{m+m_a} \times 100\%$. Although it is intuitive that a softer spring will be safer than a stiffer one it is difficult to say how much a spring of moderate stiffness would increase safety as the severity of a collision depends on a lot of other factors including angle of impact, location and type of impact (end effector to head vs robot joint to torso), and other safety features (padded robot arms, hard hats) that might be present.

An advantage of a parallel PAD over a series one is the ease of implementation. The introduction of a parallel PAD does not make the overall system any less rigid. Therefore, it has a minimal impact on the controller — torques being contributed to the joint from the PAD are well known (3.2) and can easily be compensated for. The addition of a series PAD turns an otherwise rigid system into a compliant one. This introduces a number of control challenges. Suppose a controller is set up to control end effector position of a rigid system (as would be the case in a machining operation) and the loading requires a sudden jump in joint torque. In the original system or one augmented with a parallel PAD, the jump in load can be matched by a similar jump in motor current. To achieve the same result with a system augmented by a series PAD, the motor would need to suddenly change its position — requiring massive motor accelerations with huge power demand. If instead, the controller of a system augmented in series is set up to control motor position (as would be the case in the safety illustrating example above), disturbance forces acting on the end effector could not be compensated for. Because of these two limitations, when end effector positioning is very important, a parallel PAD is a much better choice than a series one.

The packaging problem is also simpler for a parallel PAD compared to a series one. This is especially important if the purpose of the PAD is to augment an existing system. Placing a PAD between powertrain components (i.e., in series) that had previously been directly connected is almost certainly going to be more challenging than attaching the PAD in a parallel configuration [44].

1.3 Original Contributions

This thesis presents a number of original contributions:

- The most significant contribution is a general 6-step methodology for designing passive assist devices based on a prescribed maneuver. This method is detailed below (see Sec. 1.4) and referenced throughout the remainder of the thesis.

- We used this method to design an energy minimizing parallel PAD, compared its performance to a PAD based of the state-of-the-art Force Displacement Curve Fit (FDCF) design [43] and to a statically balanced (SB) PAD design, and showed the advantages of our proposed approach [8]. This is presented in Chap. 2.

- We validated our approach by comparing simulated results to experimental values of a non-backdrivable (NBD) single link robot arm executing a lifting maneuver [44, 8]. We considered a PAD in parallel with a motor and NBD worm gear. We also considered simultaneous optimization of the PAD as well as the gear ratio. This is presented in Chap. 2.

- We broadened our approach to the design of series PADs and developed a series analog to the FDCF: a derivative of Force Velocity Curve Fit (dFVCF) [54]. This is presented in Chap. 3.

- For some maneuvers, FDCF and dFVCF designs perform poorly and/or provide a poor initial guess for optimization. We developed alternative designs: a weighted Force Displacement Curve Fit (FDCF+) and a weighted derivative of Force Velocity Curve Fit (dFVCF+) that outperform their unweighted counterparts, allowing for the optimization routine to converge more quickly [54]. This is presented in Chap. 3.

- The system can be augmented with a dual PAD design which has both parallel and series components. We showed the potential of such designs from an energy standpoint and provided an engineering discussion on when each PAD type (parallel, series or dual) is most effective [54]. This is presented in Chap. 3.

- We demonstrated that this approach can effectively be applied to any augmented joint or axis executing a known, repeated maneuver, including multi-Degree of Freedom (DOF) robot manipulators [45] and manufacturing positioning machinery [14]. The 3 DOF robot examples are presented in Chaps. 3 and 5. The manufacturing system example is presented in Chap. 4.

- We showed that in certain situations (e.g., manufacturing machinery executing the same task thousands or millions of times) it is possible to modify the maneuver so that the addition of an optimized PAD can effectively increase total system performance without affecting the product [14]. This is presented in Chap. 4.

- The design methodology may also accommodate machines that execute a distribution (or family) of maneuvers. Given the distribution, a PAD can be designed to be robust against variation in the maneuver [45]. This is presented in Chap. 5.

1.3.1 Related Research Publications

The research presented in this thesis has been published in the following journals:

[8] Brown, W. R., and Ulsoy, A. G., 2013. “A maneuver based design of a passive-assist device for augmenting active joints”. *Journal of Mechanisms and Robotics.* (in press) This is the basis for Chap. 2 in the disertation.

[55] Brown, W. R., and Ulsoy, A. G., 2014. “Multi-dof and robust design of passive assist devices”. *Journal of Mechanisms and Robotics.* (submitted) This is based on Chaps. 4 and 5 in the dissertation.

The research has also been presented at the following conferences:

[1] Brown, W. R., and Ulsoy, A. G., 2011. “A passive-assist design approach for improved reliability and efficiency of robot arms”. In Proc. IEEE International Conference on Robotics and Automation (ICRA), pp. 4927–4934 This is related to Chap. 2 in the disertation.

[44] Brown, W. R., and Ulsoy, A. G., 2012. “Experimental verification of a passive-assist design approach for improved reliability and efficiency of robot arms”. In Proc. ASME Dynamic Systems and Control Conference (DSCC) This is related to Chap. 2 in the disertation.

[14] Brown, W. R., and Ulsoy, A. G., 2013. “Maneuver based design of a passive-assist device for augmenting linear motion drives”. In Proc. American Control Conference (ACC) This is the basis for Chap. 4 in the disertation.

[45] Brown, W. R., and Ulsoy, A. G., 2013. “Robust maneuver based design of a passive-assist device for augmenting active robotic joints”. In Proc. ASME Dynamic Systems and Control Conference (DSCC). (submitted) This is the basis for Chap. 5 in the disertation.

[54] Brown, W. R., and Ulsoy, A. G., 2013. “Maneuver based design of passive-assist devices: a comparison of parallel and serial systems”. In Proc. ASME Dynamic Systems and Control Conference (DSCC). (submitted) This is the basis for Chap. 3 in the disertation.

1.4 General Maneuver Based Design Methodology

Our goal is to develop a method to improve the performance of active systems with optimized, passive assist devices. The key aspect of our approach is that the design is based on a known maneuver (i.e., trajectory and loading). This general method can be applied to a wide range of specific systems performing specific tasks by following key steps. Depending on the complexity of the system being optimized some of the steps may be trivial but for a general case they are all important. These steps are described below:

1.4.1 Define Machine Architecture

Define the architecture of the machine that you wish to augment with a PAD: number of DOFs, link or axis mass and inertia information, and link or axis dimensions.

Throughout this thesis we will apply this design approach to a variety of systems. In Chap. 3 we consider the 1-DOF mass-spring-actuator systems depicted in Fig. 1.1. Chapter 2 investigates a 1-DOF rotational system — a non-backdrivable single link robot arm. The 1-DOF positioning machinery analyzed in Chap. 4 has a very similar architecture as the aforementioned mass-spring-actuators but Chap. 4 also examines a 2-DOF X-Y table and includes cutting forces. The additional axis allows for more complex maneuvers to be preformed and examined. Finally, a three link, three rotational joint, robot arm with a design similar to that of an iRobot Packbot is considered extensively in Chap. 5 and briefly at the end of Chap. 3.

1.4.2 Define Trajectory and External Forcing

The trajectory and external forcing of the machine (collectively referred to as the maneuver in this thesis) must be known in order to design PADs with this method. This method is most effective when the maneuver is precisely repeated thousands or millions of times — as is often the case with dedicated manufacturing machines. However, a substantial increase in performance may still be realized if the maneuver is not precisely known so long as the prescribed maneuver is typical of actual maneuvers [44]. Furthermore, this approach can also accommodate situations where a specific maneuver is not known but the expected maneuvers can be characterized by a known distribution [45].

Every chapter considers this design approach with a single prescribed maneuver. These maneuvers may be as simple as simple harmonic motion (Chap. 3) to a complex sequence of poses and lifting maneuvers (Chap. 5). In Chap. 4 we show that for some maneuvers it may not be possible to design an effective energy reducing PAD but that the maneuver can be modified for use with a PAD so that the final product is the same but the energy consumed is drastically reduced. There are many situations where the precise motions of the machine may not be known beforehand, e.g., UGV manipulator arms. In these situations it may be possible to consider a distribution of possible maneuvers (Chap. 5) to achieve a design that is robust against variation to the maneuver.

1.4.3 Perform Inverse Dynamics / Decouple Joints

Given the machine's architecture, trajectories of all joints (or axes), and the external forcing, a set of joint torque-angle (or axis force-displacement) profiles can be calculated using inverse dynamics. This step transforms an n-DOF system into n single DOF systems. PADs can then be designed for each joint independently. This step is skipped for single DOF systems and is trivial for the Cartesian machines presented in Chap. 4 but is critical for more complex machines such as the 3-DOF robot arm presented in Chap. 5.

1.4.4 Select Passive Assist Device Topology

There are many different PAD configurations that could be considered including springs, pistons, pressurized tanks, capacitors, etc. For a moderately sized mechanical system a spring is a practical choice for a PAD. However, there is still a large degree of different spring topologies (torsional; extension; some combination of strings, pulleys, and springs, etc.). Different topologies provide different inherent strengths and weaknesses (e.g., as topological complexity increases absolute performance will likely increase but so will design complexity, cost, and implementation difficulty). Selecting a topology should be done by taking into consideration the machine architecture (e.g., rotational joints should use torsion springs while prismatic joints should use extension springs) and packaging constraints. Once a topology is selected the variables that define the behavior of that PAD become the design variables in the optimization step.

All the PADs examined in this thesis are linear springs. Chapter 3 investigates when a parallel, series or a dual PAD setup should be used. Chapters 2 and 5 consider parallel torsion springs while parallel extension springs are used in Chap. 4.

1.4.5 Model Each Joint's Powertrain

In order to design a PAD that minimizes an objective function (see below) the parameters defining the joint powertrain must be known beforehand. There are many designs that could be considered and we present representative cases. Chapter 3 examines mass-spring-actuator systems with ideal powertrain components. Chapters 2 and 5 consider rotational joint(s) with a NBD worm gear, gear head and DC motor. Chapter 4 investigates linear axes powered via a ball screw, gear head, and DC motor.

1.4.6 Optimize Passive Assist Device

The final step of this methodology is optimization of the PAD design. We define an objective function, $f(\mathbf{x})$, corresponding to some performance metric that we wish to minimize with the addition of an optimal PAD, where $\mathbf{x}$ is a vector containing the design variables of the PAD (e.g., spring stiffness and preload). In Chap. 2, in addition to spring stiffness and preload, we also consider the gear ratio as a design variable.

In every chapter we design PADs that minimize energy consumption. In Chap. 2 we also examine a PAD that minimizes the maximum required motor torque in order to improve the reliability of the system. In Chap. 5 we consider a distribution of possible maneuvers. With this distribution we design a PAD that is robust against the variations in the maneuver by maximizing guaranteed energy savings at a certain confidence level.

In Chap. 3 we consider multiple PAD designs to accomplish energy minimization (parallel, series or dual). The selection of one of these designs can be considered to be part of the optimization process, though it likely will include some qualitative decisions on the part of the designer (see Sec. 3.2.5). For example, the designer may discover after optimization that the benefits of adding a PAD are not worth the costs of implementation.

1.5 Thesis Organization

The remainder of this thesis is organized as follows:

In Chap. 2 we provide experimental verification of our design process on a non-backdrivable (NBD) single link robot arm executing a lifting maneuver. We then discuss the key results including the primary modes of energy savings; the significance of powertrain design on energy savings; a preliminary analysis of alternative maneuvers; and a simulated comparison between energy minimizing, FDCF, and SB designs.

In Chap. 3 we set up an illustrative example of linear mass-spring-actuator systems executing a prescribed, parameterized maneuver. We then summarize how to design a parallel PAD; apply an analogous approach to a series PAD; illustrate the limitations of existing FDCF designs and propose an alternative; and extend the work to a dual PAD design. We examine and discuss when different designs would be preferable from an energy standpoint as well as the most efficient way to select them. We then apply our enhanced PAD design methodology to a more complex system — a 3-link robot arm.

In Chap. 4 we provide two examples of using our approach to design PADs for manufacturing machinery. In the first example we illustrate the process on a simple single-axis linear motion drive. In a second example we show how the general method can be applied to more complicated systems and maneuvers such as a two-DOF X-Y table performing a milling operation. In this second example we show the potential benefits of redesigning the maneuver prior to designing the PAD.

In Chap. 5 we illustrate our design approach on a 3-link UGV manipulator arm performing a single trajectory, emphasizing the inverse dynamics/decoupling step of the design process (see 1.4.3). We show how the performance of this design is affected as the variability of the maneuver increases. We then consider a family of maneuvers and design a PAD to be robust against maneuver variation.

Chapter 6 provides a summary, conclusions, and a brief discussion of future work.

Appendix A provides details of our experimental setup. Appendix B is a proof that the NBD worm gear model we use always dissipates energy. Appendix C catalogues the MATLAB code used for this research.

CHAPTER 2

Single DOF Design Example and Experimental Verification

The primary contribution of this research is a novel, general methodology for designing a Passive Assist Device (PAD) to augment an active system using optimization based on a known maneuver of the active system (see Sec. 1.4). Implementation of the PAD can result in an improvement in system performance with respect to efficiency, reliability, and/or utility. In this chapter, the methodology is experimentally demonstrated on a prototypical UGV robot arm and compared to other state-of-the-art design approaches.

There are two significant original contributions described in this chapter:

1. We validated our design approach by comparing simulated results to experimental values for a non-backdrivable (NBD) single link robot arm executing a lifting maneuver [44, 8].

2. We used this method to design an energy minimizing parallel PAD, compared its performance to a PAD based on the state-of-the-art Force Displacement Curve Fit (FDCF) design [43] and to a statically balanced (SB) PAD design, and showed the advantages of our proposed approach [8].

The remainder of this chapter is organized as follows: First we provide an example of how the general six-step design process can be used to design an energy minimizing (or maximum motor torque minimizing) PAD to augment a NBD single link robot arm. The modeled system is then validated experimentally. We then discuss the key results including the primary modes of energy savings; the significance of powertrain design on energy savings; a preliminary analysis of alternative maneuvers; and a simulated comparison between energy minimizing, FDCF, and static balancing designs. Finally, we provide a summary, conclusions, and a brief discussion of future work.

This chapter is based on the work published in [8].

2.1 Passive Device Design Methodology

Our goal is to develop a method to improve the performance of active systems with parallel optimized passive devices and to demonstrate this approach by designing an approximate counterbalancing spring for a single link robot arm. There are six main steps to the proposed design method (see Sec. 1.4). However, because the system under consideration is a single DOF joint there is nothing to decouple by performing inverse dynamics.

2.1.1 Define Machine Architecture

The machine we wish to augment with a PAD in this example is a one DOF, single-link, single-joint robot arm. The experimental arm is a LEGO beam with length, l, and mass, m_a. The arm has a near uniform mass distribution and is stiff enough to be effectively modeled as rigid. The joint end is fixed to the output shaft of the worm gear transmission, also constructed of LEGOs. LEGOs are an easy medium to work with as they are precise [56] and can be easily modified without any need to enter a machine shop. Furthermore, LEGO gears do not require the use of a lubricant during operation. The arm will raise a load mass, m_l, as part of the maneuver. The experimental values of the arm parameters are listed in Tab. 2.1. The basic arm parameters can be combined into an effective inertia and mass of the combined arm and load, respectively:

$$J_a = (m_l + \tfrac{1}{3}m_a)l^2 \tag{2.1a}$$

$$M_a = m_l + \tfrac{1}{2}m_a \tag{2.1b}$$

Table 2.1: A summary of the key parameters used in the single link arm experiments. The single link parameters are the arm length, l, arm mass, m_a, and load mass, m_l. The spring parameters are stiffness, k, and preload, T_0.

Single Link Parameters	Value	Spring Parameters	Value
l	127 mm	k	0.0194 N-m/rad
m_a	5.8 g	T_0	-0.0339N-m
m_l	45 g		

2.1.2 Define Trajectory and External Loading

Exact counterbalancing methods only consider the mass distribution of the robot and add springs (or other masses) in appropriate ways such that the robot is statically balanced in all possible configurations within its workspace. Our approximate design approach considers the mass of the robot arm as well as any load it may be carrying and its trajectory. Our design will only be able to provide exact balancing in specific loading conditions and configurations but will provide superior performance over an entire maneuver so long as the maneuver being executed is close to the one that the spring was designed for.

The selection of a particular trajectory will affect the analysis and results. Because there is no accepted standard maneuver for a robot arm (analogous to a drive cycle for vehicles), a simple task is proposed: using the arm to lift an object from a horizontal position to a vertical one. To make this task fair we make it a cyclic trajectory. This guarantees that the spring has no net energy change over the course of the maneuver. The maneuver depicted in Figs. 2.1 and 2.2, indeed, represents one of the few practical motions that a single link robot arm can do. Furthermore, this motion ensures that the worm gear is always operating under the conditions of left engagement (see Sec. 2.1.5.1). This prevents backlash from occurring, which was not included in the model for simplicity.

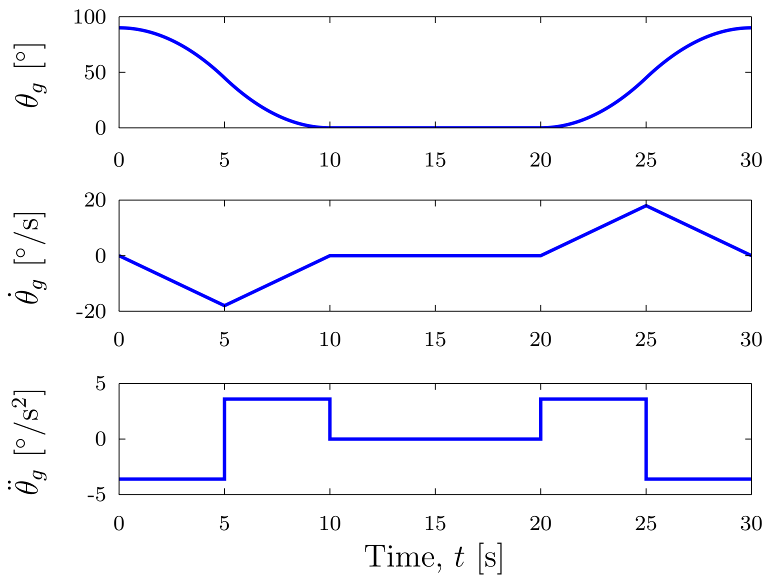


Figure 2.1: The prescribed (a) angular position, θ_g, (b) velocity, $\dot{\theta}_g$, and (c) acceleration, $\ddot{\theta}_g$, of the robot arm over the course of the experimental maneuver

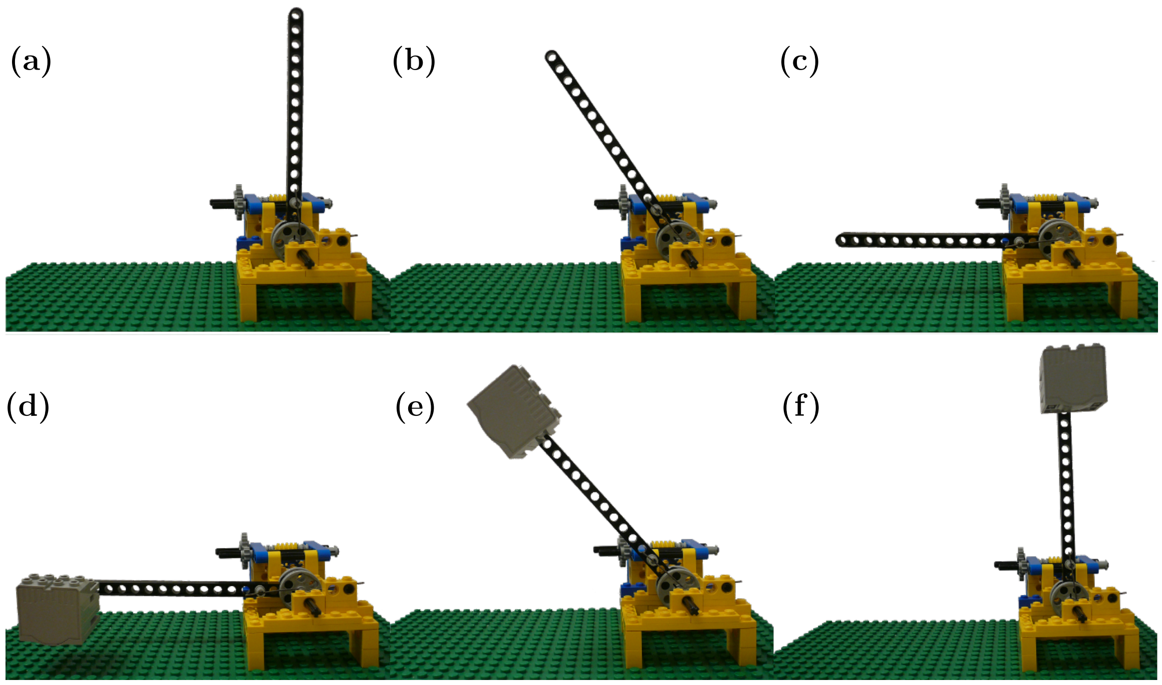


Figure 2.2: Depicts how the arm progresses through the maneuver prescribed in Fig. 2.1. The arm starts unloaded, at rest, and in a vertical position (a), lowers (b), stops and remains at rest in a horizontal configuration (c), is loaded while still at rest (d), raises the load (e), and finishes the maneuver loaded, at rest, and back in a vertical configuration (f).

After experimental verification, we can investigate how changes to the trajectory or loading affect the design of the parallel spring.

2.1.3 Perform Inverse Dynamics / Decouple Joints

This step is trivial because the actuated system only has a single DOF. Thus, the torque required to move the arm through the prescribed maneuver is

$$T = J_a \ddot{\theta}_g + M_a l g \cos\theta_g \tag{2.2}$$

where g is gravitational acceleration, J_a is the effective inertia (2.1a), M_a is the effective mass (2.1a), and l is the length of the arm (see Tab. 2.1).

2.1.4 Select Passive Assist Device Topology

There are many different PAD configurations that could be used to augment an actuated joint but in this chapter we will only consider a parallel, torsional spring.

An alternative PAD design could conceivably outperform our torsional spring by taking advantage of the geometric non-linearities to better balance gravity over the workspace (e.g., [34, 32]). However, such a design introduces two key problems: a) spring related hardware would occupy space outside of the original arm structure that could lead to collisions when the arm is in certain configurations, and b) there are multiple design topologies (e.g., should there be offsets or pulleys in the design?).

By contrast, a linear torsional spring can be designed to balance gravity exactly at the angle where gravity produces the most torque and provide some assistance to the motor over the rest of the workspace. The spring is confined to the joint so implementation is straightforward [57]. The spring is characterized by a stiffness, k, and a preload, T_0:

$$T_k = k\theta_g + T_0 \tag{2.3}$$

The obvious disadvantage of adding a spring to an existing design is that it adds to the complexity and cost of the arm. However, parallel spring implantation would be a simple upgrade, requiring no new sensors and only a minor change to any joint control design. This will help mitigate the costs and problems of increased design complexity [58].

In experiments the torsion spring is implemented such that the spring coils wrap around the shaft; one leg of the spring is attached to the joint housing (fixed) while the other is attached to the base of the arm (see Fig. 2.3).

To choose a practical spring for the experiment we configure the model for the motor, worm gear, and arm parameters recorded in Tabs. 2.1 and 2.2, simulate the maneuver depicted in Figs. 2.1 and 2.2, and find the spring design that minimizes the electrical energy consumed (see Eq. (2.10)) by the method described in Sec. 2.1.6. We then purchased the torsional spring closest to the optimal design (see Tab. 2.1). If the model matches the experiment with this spring design as well as without springs one can expect that they would match for other spring designs as well.

2.1.5 Model Powertrain

In order to design a PAD that minimizes energy consumed by a robot arm over the course of a maneuver the parameters defining the robot arm and joint powertrain must be known beforehand. There are many designs that could be considered and we will not attempt to model all of them. Instead we will demonstrate our design methodology on a robot arm and joint powertrain typical of a UGV. UGV arm joints are often non-backdrivable so that the arm can remain in a raised configuration for extended periods without consuming any power. This is typically accomplished using friction drives such as worm gears or lead screws [59, 60]. To capture these main characteristics in a simple setup, we demonstrate our design methodology on a single link arm that is driven by a DC motor through a gear head and worm gear transmission (see Fig. 2.3).

The PAD design methodology described in Sec. 1.4 can be applied to any powertrain including ones where the link is attached directly to the motor. Including the worm gear illustrates how the methodology is be applied to a specific UGV example and that the methodology can work even with nonlinear components in the powertrain.

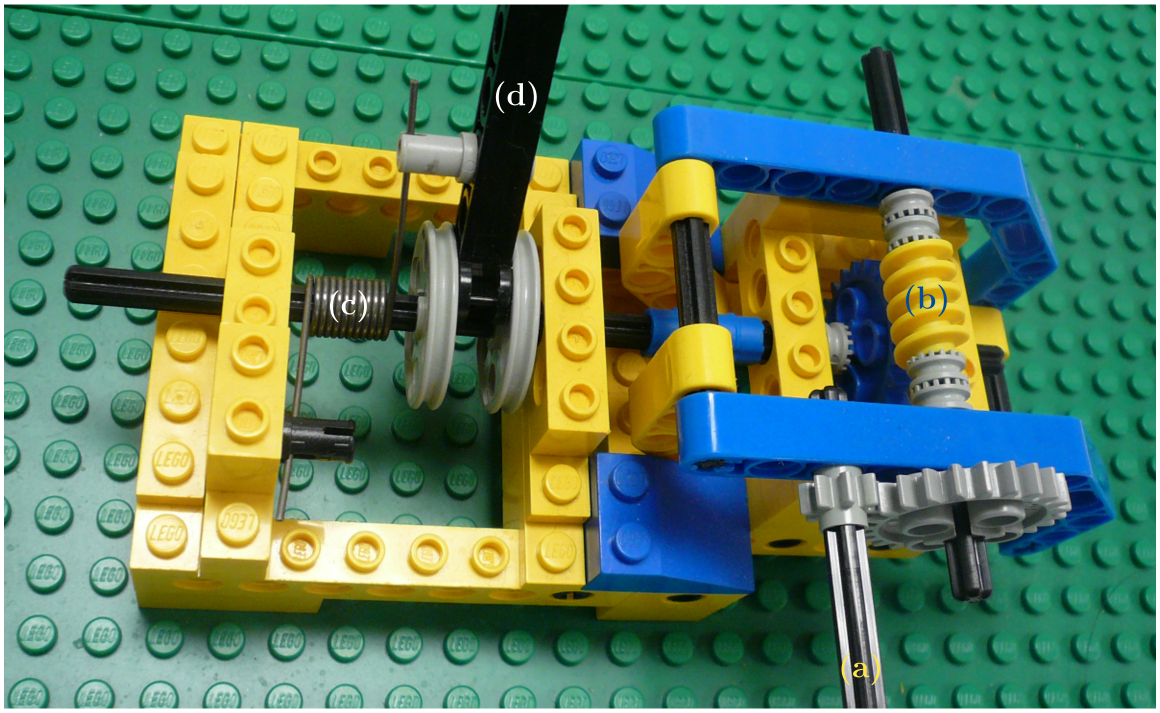


Figure 2.3: Torque is supplied from the motor (not shown) to the input shaft, (a), which drives the worm, (b), via a gear head. For each revolution of the worm, the worm gear advances one tooth. The worm gear applies a moment to the output shaft parallel to a torsion spring, (c), to maneuver the arm, (d).

Dynamics problems are typically solved to find the trajectory that satisfies the equations of motion and prescribed inputs. However, here we are interested in the inverse problem: given a trajectory (prescribed output), we want to determine the necessary inputs. Furthermore, we want to perform an optimization that minimizes the energy and/or motor torque required (see 2.1.6). Given knowledge of the trajectory and its time derivatives, the motor torque and power can be calculated algebraically using the model equations for each time step.

The first step to calculating energy consumption is to determine the torque required at the joint to make the arm motion match the prescribed trajectory and then to work backwards through the powertrain, calculating the torques required at each stage, and ending with the required voltage and current inputs.

For consistency and clarification all terms that contain the subscript g correspond to the load/gear side of the worm gear drive (e.g., $\dot{\theta}_g$ is the rotational speed of the gear) and all terms with the subscript w correspond to the motor/worm side of the worm gear drive (e.g., T_w is the total torque acting on the worm).

We consider the simplest kinematic robot arm that will capture the necessary behavior: a single link, single joint arm, shown in Fig. 2.3. By summing the moments about the joint we find that the torque acting on the worm gear is:

$$T_g = T + T_k \tag{2.4}$$

T_g is then inserted into the worm gear model.

2.1.5.1 Worm Gear Equations

The worm gear model is based on the model derived by Yeh and Wu [2] and has five fundamental parameters: worm radius, r_w, worm gear radius, r_g, worm pitch, λ, pressure angle, α, and coefficient of kinetic friction, μ. These parameters can be combined into more useful ratios; specifically, the speed ratio, i_{wg}, and the steady state torque ratios, C_1 and C_2.

The required worm gear input torque can be calculated for each time step by solving the the following conditional algebraic equation:

$$T_w = \begin{cases} 0 & \text{if stuck} \\ (J_w i_{wg} + C_1 J_g)\ddot{\theta}_g + C_1 T_g & \text{if load driven} \\ (J_w i_{wg} + C_2 J_g)\ddot{\theta}_g + C_2 T_g & \text{if motor driven} \end{cases} \tag{2.5a}$$

The first case is where the arm is stuck – it is not moving, nor is it about to. The arm is stuck if

$$\dot{\theta}_g = 0 \quad \text{and} \quad F_f < \mu F_n \tag{2.5b}$$

where F_f and F_n are the friction and normal forces on the worm-gear interface, respectively (see [2]).

The second case corresponds to a load driven situation, where the energy of the load is decreasing. The arm is load driven if

$$\begin{gathered} T_w \leq \frac{J_w i_{wg}}{J_g} \quad \text{and} \quad \dot{\theta}_g > 0 \\ \text{or} \\ T_w > \frac{J_w i_{wg}}{J_g} \quad \text{and} \quad \dot{\theta}_g < 0 \end{gathered} \tag{2.5c}$$

The third case corresponds to a motor driven situation, where the energy of the load is increasing. The arm is motor driven if

$$\begin{gathered} T_w > \frac{J_w i_{wg}}{J_g} \quad \text{and} \quad \dot{\theta}_g > 0 \\ \text{or} \\ T_w \leq \frac{J_w i_{wg}}{J_g} \quad \text{and} \quad \dot{\theta}_g < 0 \end{gathered} \tag{2.5d}$$

The second and third cases also hold in situations where the arm is currently not moving but is accelerating such that in the immediate future $\dot{\theta}_g^+ \neq 0$. When this is the case, $\ddot{\theta}_g$ can be substituted for $\dot{\theta}_g$ in Eqns. (2.5c) or (2.5d) when determining T_w. A proof is given in Appendix B that this switched system always dissipates energy and is passive.

The speed ratio is easy to measure in a worm gear because for every full rotation of the worm, the gear advances one tooth. The worm gear in the experiment has 24 teeth so the speed ratio for our transmission is 24:1. C_1 (C_2) $= -T_w/T_g$ during steady state, load driven (motor driven) operation. These ratios were found experimentally by applying a constant torque to the gear by removing the arm and lowering (or raising) a weight with a spool at constant velocity. T_w was calculated based on measurements of the motor current, i (see Eqns. (2.8a) and (2.6a)). Experimentally finding the five fundamental worm gear parameters proves that the Yeh and Wu model is accurate and is described in Appendix A.

2.1.5.2 Gear Head & Motor Equations

DC motors typically operate at high speeds and low torques. Thus, a gearhead is needed to reduce the motor shaft speed by a factor of n before driving the worm side of the worm gear transmission. The torque required of the motor is

$$T_m = T_w/n \tag{2.6a}$$

while the corresponding required motor speed is

$$\dot{\theta}_m = \dot{\theta}_g \times i_{wg} \times n \tag{2.6b}$$

We can represent the motor as a standard circuit containing the following features: (a) voltage source, V, (b) electrical resistor, R, (c) electrical inductor, L, (d) motor constant, k_m, (e) shaft and rotor inertia, J_m, (f) frictional torque internal to the motor, T_f, and (g) the motor shaft torque, T_m. The motor is governed by the following two equations [61]:

$$V = iR + k_m\dot{\theta}_m + \frac{di}{dt}L_m \tag{2.7a}$$

$$J_m\frac{d\dot{\theta}_m}{dt} = ik_m - T_f - T_m \tag{2.7b}$$

The effects of inductance and motor inertia are small compared to resistance, friction, and load torque [62] and are, therefore, ignored. This is a common assumption in mechatronic systems when the motor dynamics are much faster than the inertial dynamics. However, the energy losses due to electrical resistance and Coulomb friction are modeled. These are necessary to accurately calculate the efficiency and power requirements of the motor. The motor resistance can be calculated as the ratio of voltage to current during stalled operation. The motor constant and friction internal to the motor can be determined by plotting the voltage against speed and current against speed under no-load conditions, respectively [44]. The friction was experimentally determined to be Coulomb friction such that $T_f = C_f\text{sgn}(\dot{\theta}_m)$. The values of k_m, R, and C_f are recorded in Tab. 2.2.

The necessary current, i, voltage, V, and electrical power, P_e, at each time step can be calculated as follows:

$$i = \frac{T_m + T_f}{k_m} \tag{2.8a}$$

$$V = iR + k_m \times \dot{\theta}_m \tag{2.8b}$$

$$P_e = V \times i \tag{2.8c}$$

Sensors built into the DC motor and controller (DCMC) allow us to measure the motor current as well as the angular position and velocity of the motor shaft. A data acquisition board (DAQ) is used to connect the DCMC to a PC running Labview. A controller is implemented within Labview, which provides a command voltage back to the DCMC via the DAQ. The addition to the experiment of a basic controller is required to compensate for minor disturbances. The controller commands a motor voltage that is the sum of the simulated required voltage (feedforward) and compensates for errors with respect to angular position and velocity (PD feedback).

Table 2.2: A summary of the motor and worm gear parameter values. Each value was found through direct measurement, m, or after a number of experimental trials, e.

Motor Parameters	Value		Worm Gear Ratios	Value	
R	13.22 Ω	e	i_{wg}	24	m
k_m	56.0 mN-m/A	e	C_1	-9.95	e
C_f	0.486 mN-m	e	C_2	4.91	e
n	3	m			

2.1.6 Optimize Passive Assist Device

Given the trajectory in Fig. 2.1, the parameters required to fully describe the model in Tabs. 2.1 and 2.2, and the passive device topology defined in Sec. 2.1.4, two design variables are considered: spring stiffness, k, and preload, T_0. The model is then used to generate profiles of the motor torque needed or electrical power consumed to execute the prescribed maneuver. These profiles are functions of the two design variables, k and T_0. We then formulate a gradient based optimization problem:

$$\begin{aligned} \text{minimize} \quad & f(k, T_0) \\ \text{with respect to} \quad & k, T_0 \\ \text{subject to} \quad & k \geq 0 \\ & i_{max} \geq \max(|i(t)|) \\ & V_{max} \geq \max(|V(t)|) \\ & P_{e_{max}} \geq \max(|P_e(t)|) \\ \text{and} \quad & t_f \geq t \geq 0 \end{aligned} \tag{2.9}$$

Where $f(k, T_0) = f_1$ or f_2 and

$$f_1 = \int_0^{t_f} |P_e(t, k, T_0)| \, \mathrm{d}t \tag{2.10}$$

or

$$f_2 = \max(|T_m(t, k, T_0)|), \quad 0 < t < t_f \tag{2.11}$$

Where P_e is the electrical power consumed by the motor (see Eq. (2.8c)) and T_m is the motor shaft torque (see Eq. (2.6a)). The motor performance limit constraints are typically inactive because (a) the approach is maneuver based, (b) the prescribed maneuver can (usually) be performed by the motor without the spring, and (c) adding an optimized spring allows the motor to operate further away from its performance limits.

The function f_1 is a measure of the total energy consumed to perform the prescribed maneuver. Minimizing f_1 yields the most energy efficient design. The function f_2 is the maximum absolute torque the motor needs to generate to execute the prescribed maneuver. Because high torque operation is associated with reliability problems in electric motors (see Sec. 1.2.3), minimizing f_2 could increase the reliability of a UGV robot arm. We utilize the MATLAB optimization function *fmincon* with its default parameters to solve an optimal design problem for each of the objective functions and find the corresponding optimal values of the design variables, k^* and T_0^*.

Table 2.3: A summary of the simulated results for four different spring designs: no springs, springs optimized for energy efficiency, springs optimized to minimize maximum motor torque, and the springs used in our experiment. For each design the spring variables (k, T_0), total energy consumed (f_1, [mJ]), maximum motor torque (f_2, [mN-m]), and respective improvements [%] over the no spring design are cataloged.

Spring Design	k $[\frac{\text{mN-m}}{\text{rad}}]$	T_0 [mN-m]	f_1 [mJ]	[%]	f_2 [mN-m]	[%]
No Spring	0	0	888.4	—	4.05	—
f_1 Opt	22.2	−35.6	662.1	25.47	2.17	46.3
f_2 Opt	18.4	−33.0	665.6	25.08	2.01	50.4
Experiment	19.4	−33.9	664.2	25.23	2.05	49.3

There are two key observations to be made from the results summarized in Tab. 2.3. First, incorporating a spring designed to reduce the peak torque will also reduce the energy consumed and vice versa. Second, neither performance metric is sensitive to the spring design. All three spring designs improve energy efficiency by $\sim$25% and reduce peak torque by $\sim$50% despite modest changes to the design variables.

2.2 Experimental Validation of the Simulation

The experimental maneuver was run ten times both with and without the spring attached to the arm. The plots in Figs. 2.4–2.6 are only of a single trial but are characteristic of all experimental runs. The average energy consumed without the spring was 0.9087 J while the predicted energy consumed for such a configuration was 0.8884 J. This corresponds to an error of 2.2%. The average energy consumed with the spring attached was 0.6839 J while the predicted energy consumed for such a configuration was 0.6642 J. This corresponds to an error of 2.9%. Furthermore, the fact that less energy is consumed with the spring validates one of our primary assertions that adding a properly designed passive device to an active system can reduce energy consumption. In both the experiment and the simulation the addition of the spring results in a peak torque reduction of $\sim 50\%$ and a reduction in energy consumption of $\sim 25\%$. The $\sim 50\%$ reduction in peak torque is the same as what we found for other maneuvers of this type but the $\sim 25\%$ savings in energy is much higher than the 8% we predicted using a different arm configuration in [1]. The percent peak torque reduction remains nearly the same in both cases because it is dominated by the class of maneuver. However, the energy savings can vary depending on the design of the motor and intermediate gearing (see Sec. 2.3.2).

Fig. 2.4a overlays the simulated motor torque and the experimental motor torque. The lines are qualitatively similar and adding a properly designed torsional spring reduces the maximum motor torque for this type of maneuver by approximately half. There are some discrepancies between experiment and simulation, most notably between 20 and 25 seconds. These are mostly due to the control of the maneuver as discussed in Sec. 2.2.1. Fig. 2.4b overlays the simulated and experimental electrical power profiles. Like the motor torque profiles, we see that the lines are qualitatively similar and that adding the spring results in lower peak power demand and a more balanced power profile. Discrepancies like the ones observed between the torque profiles are also observed between the power profiles. However, the discrepancies are smaller in the power profiles than the torque profiles, most likely due to the accuracy of the experimental measurements. Only measurements from one sensor (current) are needed to calculate power, while calculating motor torque requires two measurements (current and velocity) as well as the value of friction internal to the motor found experimentally.

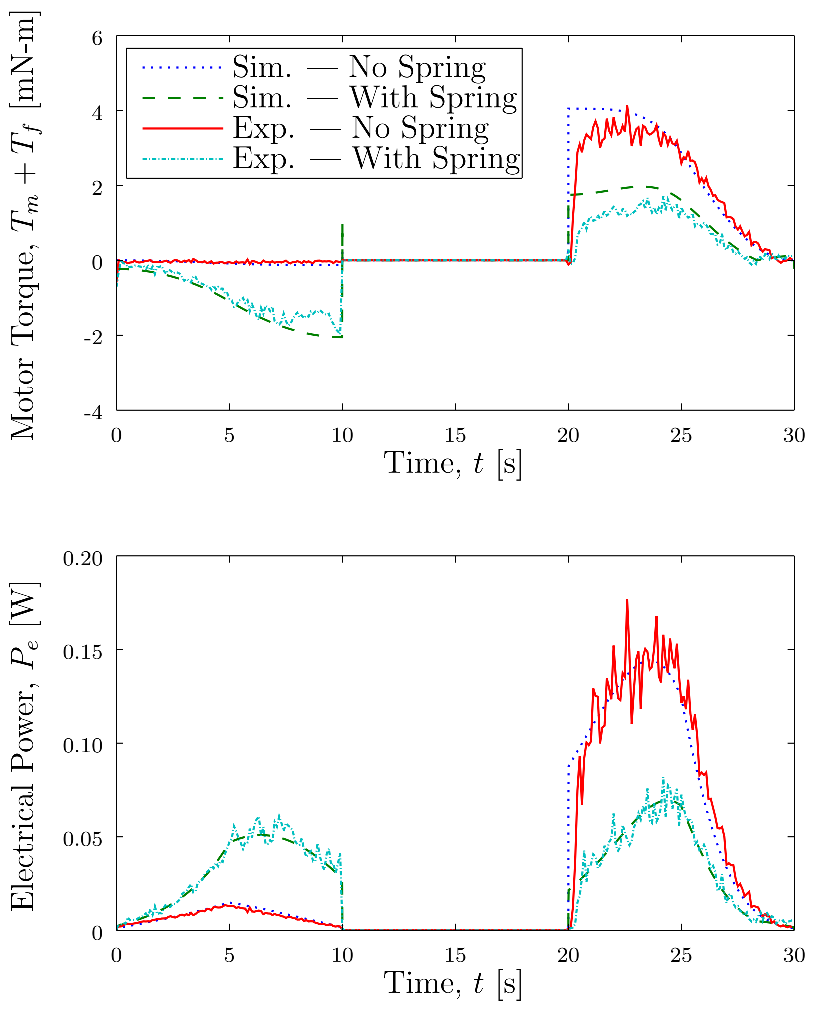


Figure 2.4: Compares the experimental electrical power and motor torque profiles to the simulated profiles with and without a spring for the maneuver in Fig. 2.1. The simulated and experimental results are qualitatively similar and adding a well designed spring can significantly reduce the maximum required motor torque, peak electrical demand, and total energy consumed over the entire maneuver despite an increase in the power consumption at the beginning of the trajectory.

2.2.1 Explanation of Differences

Our model is not perfect and the 2–3% error, although quite small, can be attributed to the following differences between the model and the actual experiment: 1) sensor noise, 2) simulation assumes perfect control, and 3) variation of the friction coefficient value. We mitigate the effects of sensor noise by averaging over ten sample measurements. This results in more accurate measurements but also imparts a brief delay of 0.1 s.

For each time step our model calculates the exact voltage, current and torque that

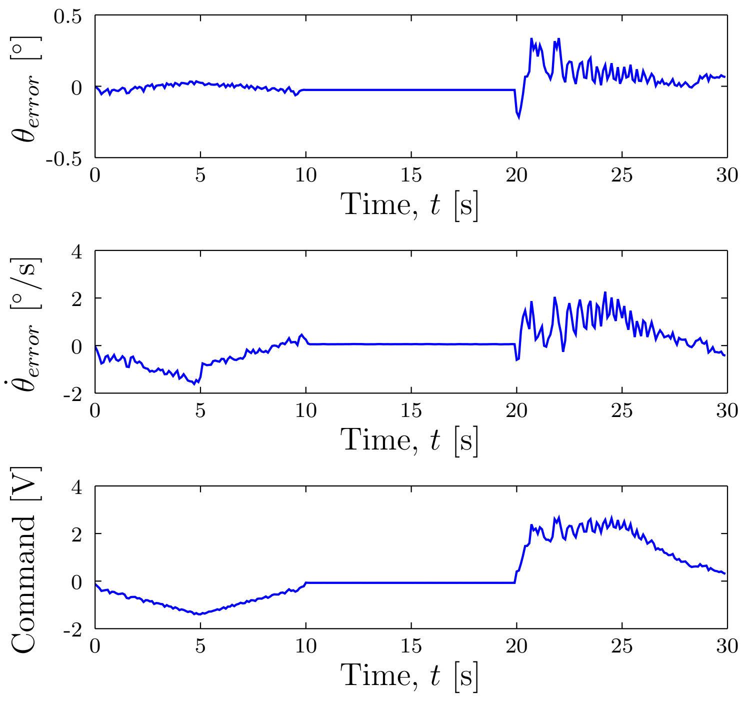


Figure 2.5: The error in (a) position and (b) velocity, and (c) the command voltage of an experimental trial with no spring. The command voltage is comprised of feed-forward information on the prescribed maneuver as well as PD control. Superior controller design could reduce fluctuations in the experimental results and decrease dynamic lag at the beginning of the ascent phase.

must be applied for the arm to perform the specified maneuver. Our experiment is operated with a real controller consisting of a feed-forward simulated voltage profile as well as feedback based on position and velocity errors. This creates a brief dynamic lag that can most clearly be seen at the beginning of the ascent phase at $t = 20$ s (see Fig. 2.4).

The third limitation of the simulation is that it only uses the mean value of kinetic friction. However, based on the experimental results we know that significant variation exists (see Fig. A.1). Figure 2.6 shows that much of the variation within the experimental data could be attributed to the variation of kinetic friction within the worm gear. The data is centered about the mean value of μ (medium dashed blue line) and almost all the data stays within two standard deviations of the mean value (long dashed green and short dashed red lines). The experimental data appears a little high over the last 5 seconds but this can be attributed to a brief temporal delay due to the controller design.

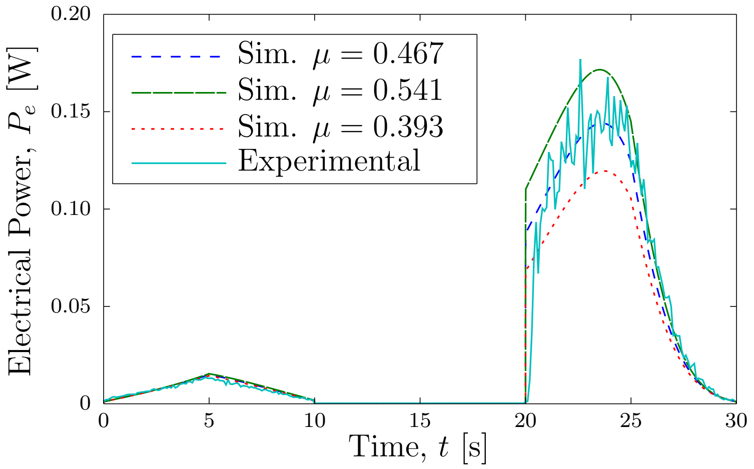


Figure 2.6: Overlays the experimental electrical power profile onto three simulated power profiles that show the effect of variation of kinetic friction within the worm gear transmission. The three simulated profiles depict the predicted power based on the mean value of friction (medium dashed blue), the mean plus two standard deviations (long dashed green), and the mean minus two stand deviations (short dashed red).

2.3 Results & Discussion

2.3.1 Sources of Energy Savings

The addition of optimized springs at the joint, in parallel with the power train, reduces the total energy consumed while executing a maneuver in three ways: (a) storing gravitational potential in the spring, (b) reducing the losses in the worm gear, and (c) reducing the losses in the DC motor. How much savings come from each mode depends both on the maneuver being performed and the parameters describing the system being simulated. The amount of energy saved in each mode can be determined in simulation by subtracting the total energy consumed at each phase along the powertrain of a sprung system from an unsprung system.

First, the addition of a torsional spring introduces an energy storage device at the joint. This can be used to capture gravitational potential energy when the arm is lowered and release that energy when the arm is raised. Without the spring, any potential energy in the system at the beginning of the maneuver would be wasted. This can account for as much as one third of the energy savings. The amount of kinetic energy in the system is always very small due to the slow speed of the maneuver. If inertial effects were more significant, then the spring could be designed to capture kinetic energy as the arm slowed and release that energy as the arm accelerated in the opposite direction in a manner analogous to the energy saved in hybrid cars by regenerative braking.

The second source of energy savings is a slight reduction in frictional losses at the worm gear. Friction power losses across the worm gear are proportional to the torque acting on the worm gear:

$$P_d = (1 - Ci_{wg})\dot{\theta}_g(T_g - J_g\ddot{\theta}_g) \tag{2.12}$$

Where P_d is the power dissipated by the worm gear and C is either C_1 or C_2 depending on the current operational state of the robot arm. The derivation of (2.12) can be found in Appendix B. The springs effectively reduce the total gear side torque over the course of the maneuver. This is typically the smallest source of energy savings, often accounting for less than ten percent of the total energy saved. However, it is important to note that if the worm drive were backdrivable, energy would have been spent to hold the arm in place for the duration when the arm was at rest.

The final, and most significant, source of energy savings is a reduction in energy lost via the DC motor. The motor can dissipate energy via friction or electrical resistance. The addition of parallel springs does not affect Coulomb losses. However, resistive losses (i.e., i^2R) are proportional to the square of the current and the current is proportional to the torque. Therefore, reducing the torque over the course of the maneuver — especially the maximum torques – leads to a significant reduction of resistive losses. This mode of energy savings explains why the spring designed to minimize the maximum torque, f_2 Opt, is almost identical to the spring designed to minimize energy consumed, f_1 Opt (see Tab. 2.3). Nearly two thirds of the energy savings in our previous work [1] and almost 90% in the experiment can be attributed to this source of energy savings. Using a spring to operate the motor in a region of lower torque and higher efficiency is analogous to operating an internal combustion engine at its "sweet spot" in terms of fuel efficiency in a hybrid electric vehicle [63].

2.3.2 Gear Ratio Parameter Study

In the simulation based on experimental parameters we observed a reduction in peak torque of $\sim$50% and a reduction in energy consumption of $\sim$25%. In previous work we simulated similar trajectories to the one depicted in Fig. 2.1 but the parameters describing the system represented a typical UGV arm and the lifted load was much greater [1]. The peak torque savings was very similar in every case ($\sim$50%) but the 8–11% energy savings observed for the UGV arm was much lower than the $\sim$25% observed on the experimental arm. This is because the torque reduction is almost entirely a function of the type of maneuver (e.g., lifting an object) while the energy savings also depend on the parameters of the joint powertrain. In other words, a poorly designed motor may benefit more from the addition of a PAD than a well designed motor. We would like to know how much of the energy savings can be attributed to powertrain design and how much can be attributed to the addition of the optimized spring.

We observed the effect of altering the gear ratio, n, on energy savings. For a wide range of gear ratios we calculated the energy consumed with and without a spring optimized for energy minimization at each particular gear ratio. The results of this parameter study are plotted in Fig. 2.7.

At low gear ratios the motor experiences high torques and consequently a lot of energy is required to perform the maneuver. The addition of an optimized parallel spring has a very large effect in this region because it can reduce the peak torques by $\sim$50% and those torques correspond to the majority of energy consumption. However, even with the spring the system is consuming more energy than it would if it were operating with a higher gear ratio. The experimental system is operating at a gear ratio that is less than optimal, explaining the relatively large 25% energy savings.

At high gear ratios the motor is operating at low torques and high speeds. In such a configuration the power losses are both due to electrical resistance (related to torque squared) and friction (related to speed). The addition of an optimized spring has less influence in this region as the torque experienced by the motor is already small. As the gear ratio continues to increase, the energy consumed will increase as the power needed to overcome motor friction increases. Furthermore, the addition of the spring begins to have a negligible effect on energy consumption. The UGV powertrain simulated in our previous work [1] is operating at a gear ratio higher than optimal thus only 8% energy savings were obtained.

The powertrain is properly geared for a specified maneuver prior to the addition of an optimized spring when the losses due to electrical resistance and motor torque are balanced. This is the gear ratio that corresponds to the minimum point on the no-spring curve. An optimized spring can then be added to the system to further reduce peak torques. The energy savings observed by doing this are $\sim$15% regardless of other design differences.

The system optimal gear ratio corresponds to the minimum point on the optimized spring curve. The system optimal design always has a lower gear ratio than the gear optimized design because the higher torque, lower speed operational range allows for the spring to be more effective compared to the design where the gear ratio is

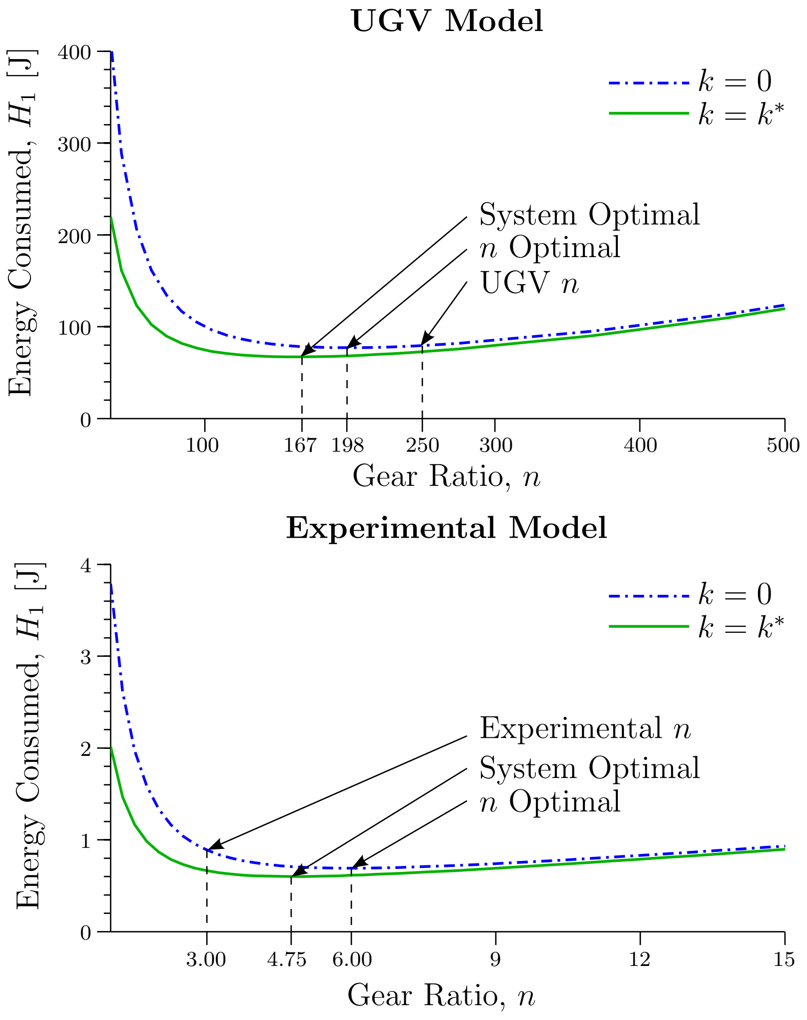


Figure 2.7: Depicts the energy consumed to execute the maneuver defined in Sec. 2.1.2 as a function of the gear ratio, n. Figure 2.7a corresponds to the UGV model simulated in [1] while Fig. 2.7b corresponds to the experimental model described in Sec. 2.1. The dashed blue line is the energy consumed by a no-spring design while the solid green line is the energy consumed with a spring design optimized for the specific gear ratio.

optimized first. This observation illustrates the significant benefits of designing the robot arm, powertrain, and PAD simultaneously, which will be explored in future research.

2.3.3 Preliminary Analysis of Alternative Maneuvers

For the method proposed in this thesis to be of practical value it must work for different trajectories, loads, and robot arm configurations. This is especially true for

UGVs, which operate in unstructured environments. Fig. 2.8 shows that a spring optimized for one maneuver will provide a similar performance enhancement even when executing maneuvers it was not optimized for as long as the difference in maneuver is not too large (e.g., $m_l > 18$ g when $m_l = 45$ g was used for the design).

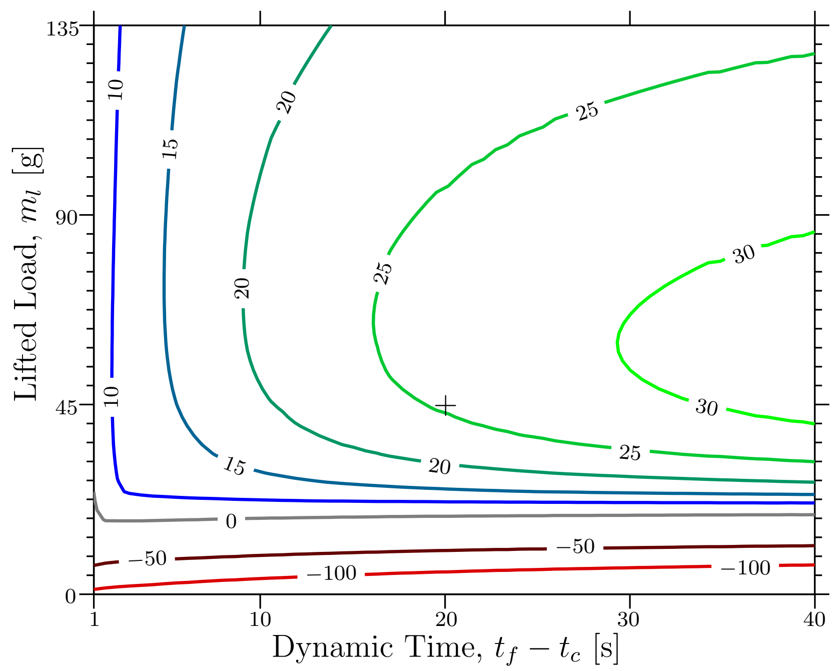


Figure 2.8: The amount of energy saved [%] by the addition of a spring optimized for a specific maneuver varies as the maneuver being performed deviates from the maneuver used for design, +. The spring provides a varying degree of assistance so long as the lifted load does not drop too far below the load the spring was design for, at which point the spring decreases the systems performance.

If the maneuver is not known with certainty it may be beneficial to design the spring so that it minimizes the average energy consumed over a distribution of maneuvers. The trajectory, which is piecewise quadratic in angular position, can be defined by four parameters: time to execute the maneuver, t_f, time at rest in a horizontal configuration, t_c, the ratio of ascent time to descent time, and the ratio of acceleration to deceleration. The four trajectory parameters as well as the mass lifted, m_l, were distributed such that the mean values of each parameter corresponded to the values used in the single maneuver case described in Fig. 2.1 and Tab. 2.1. A spring design was then found that minimized the average energy consumed for 10,000 different maneuvers (see Fig. 2.9 and Tab. 2.4).

The design and performance of the single maneuver and distribution of maneuvers is very similar. This makes sense because the average distributed maneuver is the same as the single maneuver by construction. However, the optimal spring for a

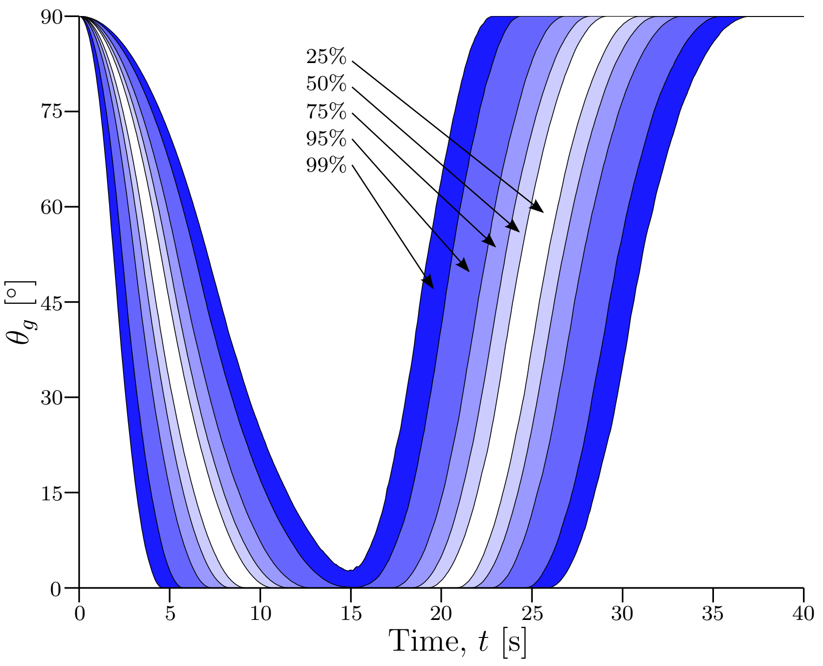


Figure 2.9: The angular position of a distribution of 10,000 trajectories. The contours correspond to deviation from the mean trajectory. The innermost contour contains the central 25% of trajectories, while the outermost contour contains 99% of all trajectories.

distribution of maneuvers was slightly more compliant and on average saved a little less energy than the design optimized for a single maneuver. The differences between the average performance and the single maneuver performance can be attributed to the more extreme maneuvers introduced by the distribution far away from the mean. The decrease in efficiency can be explained by the nonlinear relationship between energy consumed and peak torque, especially due to resistive losses in the motor that are proportional to i^2R (described in Sec. 2.3.1). Consequently, a unit increase in peak torque will correspond to a increase in energy consumption greater in magnitude than the decrease in energy consumption corresponding to a unit decrease in peak torque. This behavior couples with the fact that a spring that is too stiff for a low torque maneuver *increases* the energy required to perform such a maneuver (see Fig. 2.8). However, a soft spring acting on a high torque maneuver will always decrease the energy required, even if not by very much. These two behaviours explain why a spring designed for a distribution is slightly more compliant than, and on average not as efficient as, a spring designed for a single maneuver.

In Chap. 5, we optimize the PADs over a distribution (or family) of maneuvers so that the design will be robust with respect to uncertainties in the trajectories and loads. This can be accomplished by altering the objective function from a minimization of energy to a minimization of a weighted combination of average energy and variation in energy consumed, i.e., a robust design approach [64].

2.3.4 Spring Design Comparison

No work had previously been done to optimize a parallel PAD for a known maneuver. However, it has been known for a long time that the addition of a passive device to an active system can improve performance. Therefore, it is not surprising that our optimized design provided a substantial increase in performance over a springless design. Here we will compare our design methodology against two other state-of-the-art design concepts: a statically balanced (SB) design that exactly compensates for the mass of the arm but ignores the load and trajectory [22], and a design that is a first order approximation of the torque required from a prescribed maneuver (FDCF) [43]. Not surprisingly, we see in Tab. 2.4 that all three spring designs provide an increase in efficiency compared to the spring-less design.

Table 2.4: A comparison of four spring design methods: no spring, statically balanced (SB), maneuver-based Force Displacement Curve Fit (FDCF), and maneuver-based energy minimization (f_1 Opt). For each design method the spring design and performance are recorded for both a single maneuver and an average of 10,000 distributed maneuvers (see Figs. 2.1 & 2.9, respectively). The model parameters match those described in Tabs. 2.1 & 2.2.

Spring Design	Single Maneuver			10,000 Maneuvers		
	k $[\frac{\text{mN-m}}{\text{rad}}]$	T_0 [mN-m]	Energy Saved [%]	k $[\frac{\text{mN-m}}{\text{rad}}]$	T_0 [mN-m]	Energy Saved [%]
No Spring	0	0	—	0	0	—
SB	—	—	8.61	—	—	8.57
FDCF	19.0	−33.0	25.23	19.1	−31.3	23.30
f_1 Opt	22.2	−35.6	25.47	20.2	−33.0	23.35

In general, determining the design of a PAD that will make the system statically balanced is difficult, can be done in a variety of ways [22, 23, 34] and is beyond the scope of this paper. However, it is easy to calculate the effect of such a PAD on a single link robot arm as considered here: $T_k = -m_a lg \cos\theta_g$. The SB design provides significantly less improvement than the two maneuver based designs. This is because of the type of maneuver being performed. Here we showed an example of a relatively light arm lifting a relatively heavy load. Because the SB design does not consider the effects of the loading it cannot achieve the same performance increase as the maneuver based designs despite its non-linear capability. If the situation were reversed so that a massive arm was moving a relatively small load (e.g., an industrial pick-and-place robot) then a SB design would likely outperform a maneuver based, linear design.

The two maneuver based designs are nearly identical, as is their performance. This result suggests that for this simple system, executing a basic maneuver, with the goal of minimizing energy consumption there is no reason to employ optimization as the improvements over a FDCF design are negligible at the expense of greater computational cost. However, there are three major advantages of employing the optimization method proposed in this paper. First, the optimization approach is guaranteed to perform at least as well as the FDCF design. Second, the objective function is easily generalized to handle more complex systems and/or maneuvers, where a FDCF may not be the best design. Finally, when considering a distribution of possible trajectories, the optimization approach can be easily modified to find designs that exhibit less variance in a metric of interest. This last advantage will allow us to find designs that are robust with respect to changes in maneuver in Chap. 5.

2.4 Concluding Remarks

In this thesis we proposed a general approach to increase the performance of active systems by augmenting them with optimized parallel PADs. In this chapter, we experimentally demonstrated this concept on a single link robot arm augmented with a torsional spring. We observed an increase in performance by decreasing the peak motor torque required by $\sim 50\%$. This decrease in required peak torque can improve reliability by increasing the life of the joint motors, improve the utility by allowing heavier loads to be lifted, and significantly improve efficiency. Finally, we analyzed the modes of energy savings, examined the importance of the gear ratio on energy savings, considered a distribution of maneuvers, and compared our results to other applicable state-of-the-art designs.

The remaining chapters in this thesis demonstrate the utility of the proposed PAD design approach. This chapter exclusively examines parallel PADs. In Chap. 3 we examine how the design approach can be used to design series PADs as well as systems that are augmented PADs that contains both series and parallel components. In Chap. 4 we apply this approach to systems other than robotic arms such as an X-Y table performing a milling operation [14] and demonstrate that in certain situations it is both practical and beneficial to redesign a maneuver in order to take advantage of a

PAD. A limitation of the work presented in this chapter is the simple kinematics of the system under consideration. Chapter 5 applies the proposed design approach to a 3-link robot arm by incorporating an inverse dynamics / decoupling step. Furthermore, Chap. 5 examines the impact that maneuver vaiablity has on system performance and designs PADs to be robust against such variation.

CHAPTER 3

Parallel, Series, and Dual PADs

Chapter 2 used the general design method described in Sec. 1.4 to augment an active joint executing a known maneuver with a *parallel* PAD. This chapter extends this methodology to series PADs as well as systems augmented with both a series and parallel PAD. Some of the work presented in this chapter is similar to the work presented in [46]. The work presented here is presented on a more abstract system and is more general. Furthermore, there are three significant original contributions in this chapter:

1. We extend the existing maneuver based design method for parallel PADs to series and dual PAD designs.

2. We provide a superior initial design for optimization beyond the traditional FDCF.

3. We provide insights on how to select a specific class of PAD for a particular engineering application.

The remainder of this chapter is organized as follows: First we provide an overview of the six step design process (see Sec. 1.4 and set up an illustrative example of linear mass-spring-actuator systems executing a prescribed, parameterized maneuver. We then summarize how to design a parallel PAD, apply an analogous approach to a series PAD, illustrate the limitations of traditional FDCF designs and propose an alternative, and extend the work to a dual PAD design. We will examine and discuss when different designs would be preferable from an energy standpoint as well as the most efficient way to select them. We then apply our enhanced PAD design methodology to a more complex system — a 3-link robot arm. Finally, we provide a summary, conclusions, and a brief discussion of future work.

This chapter is based on the work presented in [54].

3.1 Illustrative Example

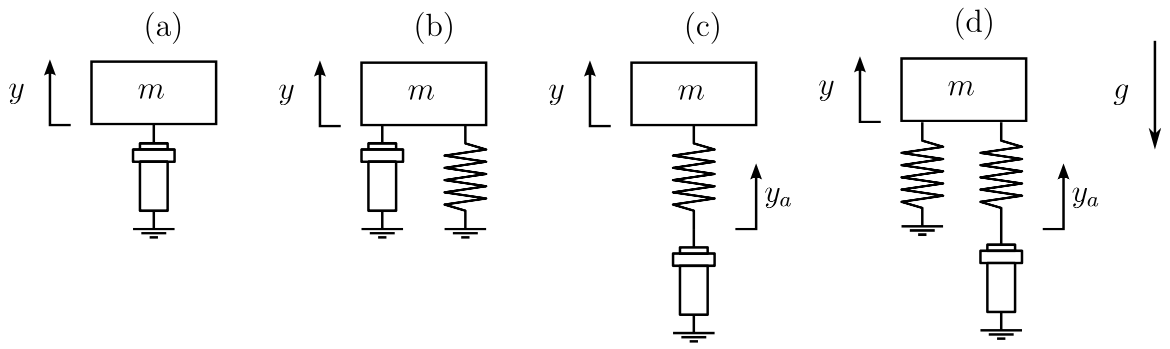


Figure 3.1: Depicts an unsprung, actuated mass (a), the same system augmented with a parallel spring (b), the same system as (a) augmented with a spring in a series configuration (c), and the system augmented with a dual spring configuration (d).

The maneuver based PAD design method described in Sec. 1.4 was used to design a torsional spring acting parallel to the joint actuator in Chap. 2. In this chapter we illustrate the analogous nature of series and parallel design and how to select between them. To do so, we consider a simple example where some of the six steps described in the general method are trivial.

Several subscripts are used throughout this chapter. The subscripts $_a$ and $_k$ correspond to the actuator and spring. The subscripts $_s$, $_p$, and $_d$ refer to specific series, parallel, and dual configurations, respectively. Finally, the subscript $_i$ refers to the i^{th} timestep. For example, F is the force required to move the mass, P_{a_p} is the power required of the actuator in a parallel configuration, and $\dot{y}_{k_s}$ is the spring velocity for a series spring.

1. **Define Machine Architecture**: The machine we wish to augment with a PAD in this example is a one dimensional translational mass-actuator system as depicted in Fig. 3.1a. We consider the mass to be $m = 1$ kg.

2. **Define Trajectory and External Forcing**: The trajectory and external forcing of the machine (collectively referred to as the maneuver throughout the thesis) must be known in order to design PADs with this method. The selected trajectory, $y(t)$, in Fig. 3.2 is a piece-wise smooth mix of sinusoidal acceleration zones (where the mass reverses direction) and constant velocity zones. The trajectory is parameterized by the frequency, $\omega = 1$ rad/s, and amplitude, $a = 1$ m, of the sinusoidal portions and the ratio of constant velocity duration to sinusoidal duration, $n = 1$.

$$y(t) = y_0 + \begin{cases} a\sin(\omega t) & 0 \le t < \pi/\omega \\ -a(\omega t - \pi) & \pi/\omega \le t < (n+1)\pi/\omega \\ a\sin(\omega t - n\pi) - an\pi & (n+1)\pi/\omega \le t < (n+2)\pi/\omega \\ a(\omega t - 2(n+1)\pi) & (n+2)\pi/\omega \le t < 2(n+1)\pi/\omega \end{cases} \tag{3.1}$$

Where $y_0 = a\omega n\pi/2$ so that the position is evenly distributed about the y-axis. Increasing ω or a increases the severity of the acceleration of the mass. As $n \to 0$ the trajectory becomes harmonic. The external forcing is a constant load, C, which could simulate gravitational acceleration. For example, $C = 0$ could be the mass-actuator operating on a horizontal plane (or in space), while $C = 10$ N could be the mass actuator acting vertically.

3. **Perform Inverse Dynamics / Decouple Joints**: This step is trivial because the actuated system only has a single DOF. Thus, the force required from the actuator to execute the prescribed maneuver is $F = m\ddot{y} + C$.

4. **Select PAD Topology**: We will demonstrate the design methodology with parallel, series and dual PAD systems (see Fig. 3.1b-d). However, in each case the PAD will be a linear extension spring with stiffness, k, and preload, F_0:

$$F_k = ky_k + F_0 \tag{3.2}$$

5. **Define Powertrain**: The actuator in the system is considered to be an ideal motor — it can always supply the required force and velocity and has no inefficiencies. When analyzing a real system it would be important to include more realistic powertrain models (e.g., gears, non-backdrivable elements, friction losses) to accurately predict and design PADs for energy saving purposes [8].

6. **PAD Optimization**: The final step of this methodology is optimization of the PAD design. In this example we wish to find a linear extension spring that minimizes actuator energy consumption. Thus, we formulate the following

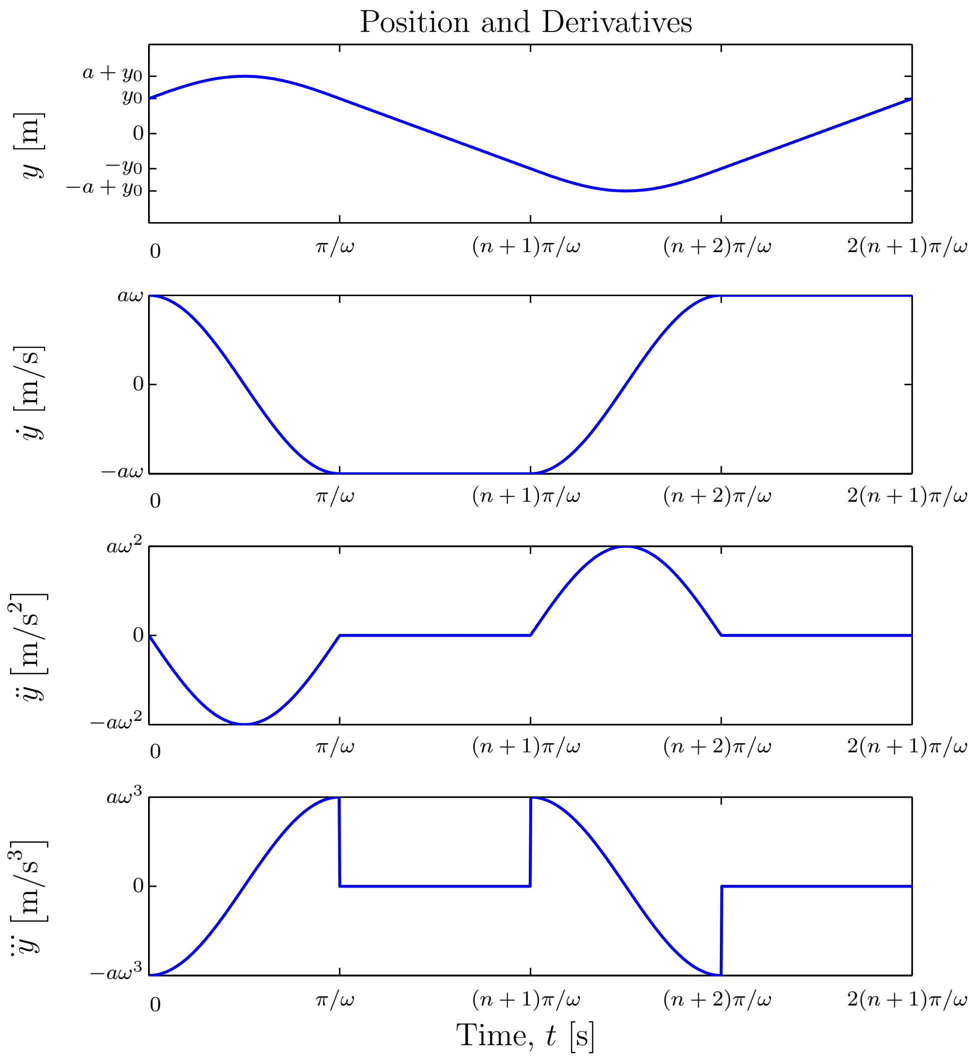


Figure 3.2: The parameterized trajectory of the actuated mass, $y(t)$, and the first three time derivatives. The trajectory and the external loading, C, combine to define the maneuver.

gradient based optimization problem:

$$\begin{aligned} \text{minimize} \quad & f(k, F_0) = \int_0^{t_f} |P_a(t, k, F_0)| \, \mathrm{d}t \\ \text{with respect to} \quad & k, F_0 \\ \text{subject to} \quad & k \geq 0 \\ \text{and} \quad & t_f \geq t \geq 0 \end{aligned} \tag{3.3}$$

Where P_a is the required actuator power. The form of P_a will change depending on the type of PAD being optimized. We solve Eq. (3.3) with MATLAB's built in function *fmincon* initialized with a FDCF design.

3.2 PAD Design Comparison

3.2.1 Parallel Spring Design

In this example, the objective of adding a PAD is to minimize required actuator energy (see Eq. (3.3)) to execute a given maneuver (see Fig. 3.2). Because the actuator is ideal, the power required is:

$$P_a = \dot{y}_a F_a \tag{3.4}$$

For a parallel configuration, the spring extension, y_k, and the actuator position, y_a, are both equal to the mass position, y, while the actuator force, $F_a = F + F_{k_p}$.

$$P_{a_p} = \dot{y}(F + F_{k_p}) \tag{3.5}$$

As y is prescribed, we can reduce the energy required by making $F + F_{k_p}$ as close to 0 as possible as often as possible:

$$-F \sim k_p y + F_0 \tag{3.6}$$

F and y are both prescribed by the maneuver and can be plotted against each other as in Fig. 3.3a. The linear least squares curve fit that best fits the F vs. y data will have a slope of k_p and a constant offset of F_0. We refer to this design as a Force Displacement Curve Fit (FDCF) [43].

An optimal design based on Eq. (3.3) will always outperform a FDCF design at reducing energy consumption because the optimization problem also considers $\dot{y}$. The FDCF treats all force-displacement data points equally but it is more important to reduce the required actuator force when the actuator velocity is high, as discussed in the Series Spring Design section below (Sec. 3.2.2). An optimization routine handles this automatically. However, FDCF often provides a good initial guess to initialize a gradient based optimization.

Implementing an energy minimizing parallel PAD can greatly increase efficiency. Figure 3.4a shows the power profiles of the actuator without a parallel PAD as well as with three different PAD designs for a maneuver with $n = 1$ and $C = 10$ N. Figure 3.4b zooms in to see the effect of the different PAD designs. As expected the optimal design performs the best while FDCF is not quite as good. FDCF+ denotes a weighted force displacement curve fit whose performance is intermediate. When the maneuver is altered such that $n = 0$ the maneuver becomes harmonic and the PAD completely eliminates actuator energy consumption. However, if the maneuver is altered such that $n = 1$ and $C = 0$, a parallel PAD can only achieve modest energy savings. This suggests that parallel PADs are well suited for maneuvers where external forces are large compared to inertial forces. These results are summarized in Tab. 3.1.

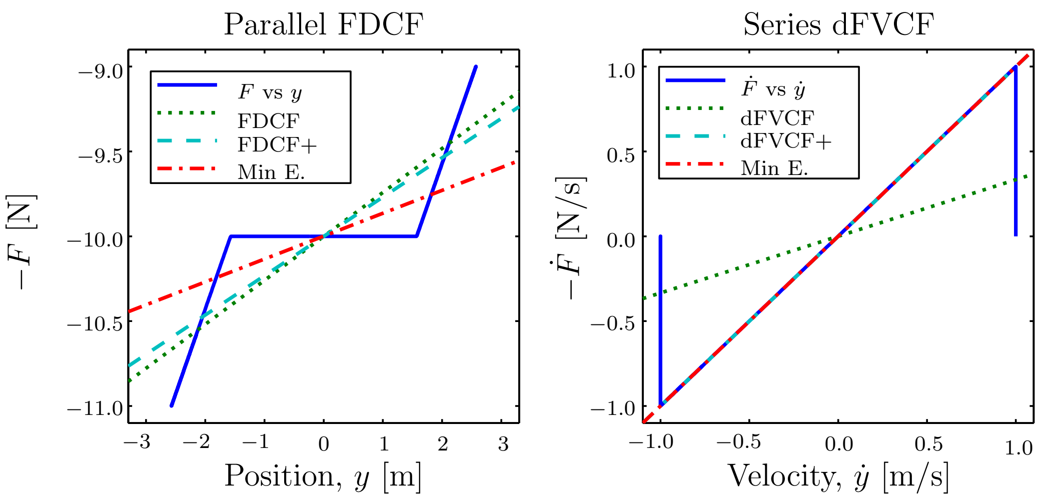


Figure 3.3: a) The required force to move the mass through the prescribed trajectory vs the corresponding position of the mass (solid blue). Three parallel linear extension springs designs are overlaid: a force-displacement curve fit (FDCF; dot green), weighted FDCF (FDCF+; dash cyan), and energy minimization (dash-dot red). b) The analogous data and spring designs for a series configuration.

3.2.2 Series Spring Design

A series PAD can be designed in a manner completely analogous to the parallel design. A series PAD transmits the force required to move the mass, F, through the spring directly to the actuator, and therefore has no affect on actuator force, $F_a = F$. The series spring extension, $y_{k_s} = y - y_a$, thus, the required actuator power is:

$$P_{a_s} = (\dot{y} - \dot{y}_{k_s})F \tag{3.7}$$

As F is prescribed, we reduce the energy required by making $\dot{y}_{k_s}$ as close to $\dot{y}$ as possible as often as possible. To get $\dot{y}_{k_s}$ in terms of the series spring parameters we take the time derivative of the spring constitutive relation (see Eq. (3.2)):

$$\dot{F}_{k_s} = k_s \dot{y}_{k_s} \tag{3.8}$$

Note that the spring preload, F_0, disappears. The extension force in the spring is equal and opposite to the force required to move the mass, $F_{k_s} = -F$. This yields a

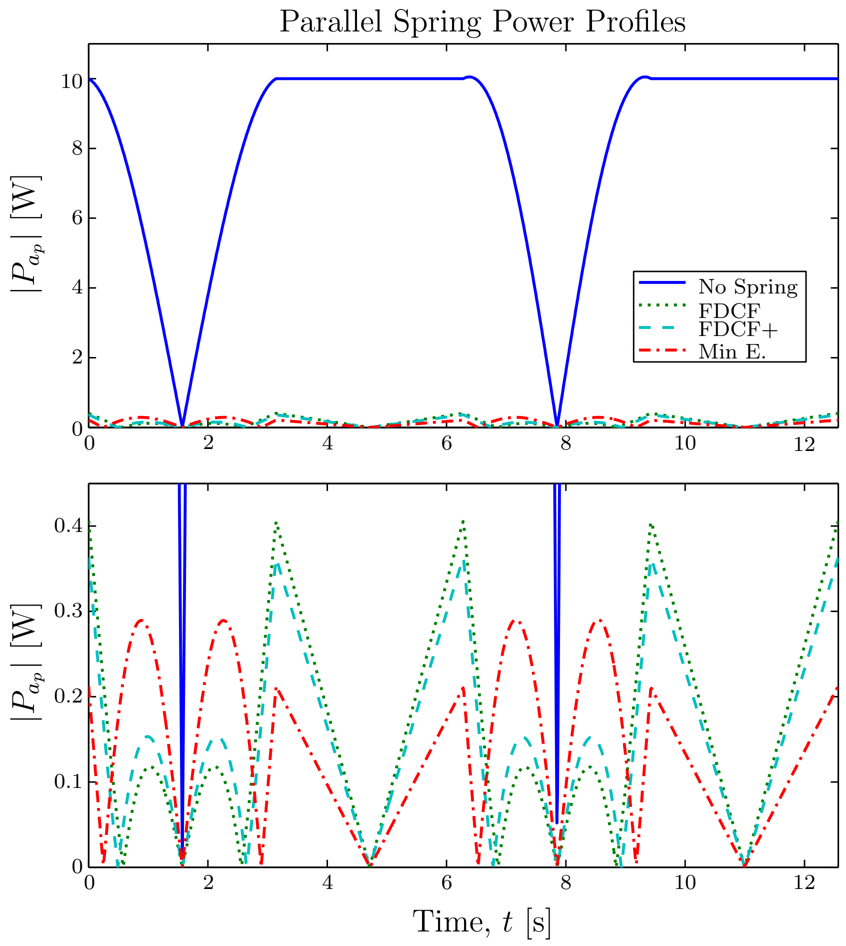


Figure 3.4: Required actuator power to execute the maneuver with no spring (solid blue) and three parallel springs. The top figure depicts the total power while the bottom figure examines the different performance of the three spring designs.

derivative of Force Velocity Curve Fit (dFVCF):

$$-\dot{F} \sim k_s \dot{y} \tag{3.9}$$

This is the same form as Eq. (3.6) except with $\dot{F}$ and $\dot{y}$ instead of F and y and there is no F_0 term. The series spring preload could be important for implementation and packaging reasons but has no effect on actuator power consumption. This reduces the optimization problem from two design variables to one. The dFVCF design will often perform well and can be used as in initial guess an a gradient based optimization routine.

However, for certain maneuvers (such as Fig. 3.2) the dFVCF design is worse

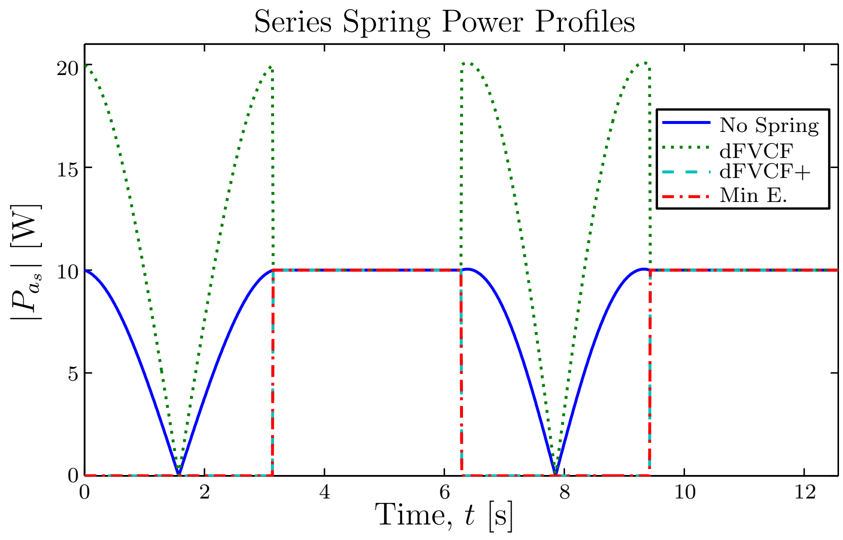


Figure 3.5: Required actuator power to execute the maneuver with no spring (solid blue) and three series PADs. No series PAD can affect the power consumption during periods of constant velocity. Furthermore, implementing the unweighted dFVCF design (dot green) substantially increases energy consumption. The weighted dFVCF+ (dash cyan) and energy minimizing design (dash-dot red) are the same and both eliminate actuator power during the sinusoidal regions of the trajectory.

than having no spring at all. It dramatically increases energy consumption (see Fig. 3.5) and serves as a poor initial guess for optimization. The maneuver explored in this example is simple enough that the effect of a series spring design on actuator energy consumption, $f(k)$, can be calculated analytically by substituting the selected maneuver (3.1) into the series power equation (3.7) and integrating over the duration of the maneuver:

$$f(k_s) = \begin{cases} 2a\left|1 - \dfrac{\omega^2 m}{k_s}\right| \left(\dfrac{C^2}{B} + B\right) + 2aC\pi n & |C| \leq B \\ 4a\left|1 - \dfrac{\omega^2 m}{k_s}\right| C + 2aC\pi n & |C| \geq B \end{cases} \tag{3.10}$$

Where $B \coloneqq a\omega^2 m$. This leads to different performance regimes based on the spring stiffness:

$$\begin{array}{rl} k_s < 0 & \text{infeasible stiffness} \\ k_s = 0 & \text{disconnected joint} \\ 0 < k_s < \omega^2 m/2 & \text{reduced performance} \\ k_s = \omega^2 m/2 & \text{neutral performance} \\ \omega^2 m/2 < k_s < \infty & \text{improved performance} \\ k_s = \omega^2 m & \text{optimal performance} \\ k_s \to \infty & \text{no spring} \end{array} \tag{3.11}$$

The dFVCF design lies within the reduced performance regime. This is because the vertical "spikes" or "tails" in Fig. 3.3b correspond to points in the maneuver where the mass is moving at constant speed ($\ddot{y} = 0$) and experiencing a constant external force ($\dot{F} = 0$). A series spring cannot affect actuator power under these conditions. Therefore, trying to fit the spring design to those data points only has the effect of pulling the design away from regions where the spring could effectively reduce power consumption. An analogous problem could occur for parallel designs if the prescribed maneuver had a stationary segment with changing external loads. In both cases a curve fit spring design would perform much better if it were able to ignore data points where the spring would be ineffective.

The dFVCF design is the linear, least-squares curve fit to the derivative of force vs. velocity data (3.9):

$$\text{minimize with respect to } k_s \qquad \sum_{i=1}^{s} (\dot{F}_i + k_s \dot{y}_i)^2 \tag{3.12}$$

Where s is the number of time steps in the maneuver and $\dot{F}_i$ is the force required to move the mass at the i^{th} time step.

We propose a new curve fit design, dFVCF+ (parallel analogue: FDCF+), which uses a linear, *weighted*, least-squares curve fit:

$$\text{minimize with respect to } k_s \qquad \sum_{i=1}^{s} W_{i_s} (\dot{F}_i + k_s \dot{y}_i)^2 \tag{3.13}$$

By making the weighting, W, proportional to the corresponding spring power, P_k, we can find a PAD design that typically outperforms its unweighted counterpart because the weighting is able to de-emphasize data points where the PAD cannot be as effective (e.g., the "tails" in Fig. 3.3b).

For a series system, the equation for actuator power (3.7) can be expressed as

$$P_{a_s} = P - P_{k_s} \tag{3.14}$$

Where P_{a_s} is the actuator power in the series configuration, P is the power required to move the mass, and $P_{k_s} = \dot{y}_{k_s} F$. Combining this result with Eq. (3.8) yields

$$P_{k_s} = F\dot{F}/k_s \tag{3.15}$$

When P_{k_s} is small it will have little affect on P_{a_s}, therefore, the curve fit should weight the data with

$$W_{i_s} = F_i \dot{F}_i \tag{3.16}$$

The analogous weighting for an FDCF+ design would be

$$W_{i_p} = y_i \dot{y}_i \tag{3.17}$$

We can then use the dFVCF+ design to initialize the series optimization.

Due to the simple nature of this example maneuver the dFVCF+ design is the same as the optimal series PAD design. This is because the dFVCF+ design ignores the constant velocity data ($W_{i_s} = 0$ in those regions) and can completely compensate for the joint velocity in the sinusoidal regions. Regardless of constant external loading, an energy minimizing series spring can completely eliminate actuator velocity (and, thus, power) for a harmonic trajectory ($n = 0$). However, in the example maneuver where $n = 1$ and $C = 10$ N, a series PAD can only achieve modest energy savings. This suggests that series PADs are well suited for maneuvers where inertial forces are large compared to external forces. These results are summarized in Tab. 3.1.

3.2.3 Dual Spring Design

In addition to augmenting a joint with a parallel or series PAD, a joint may be augmented with both types of PADs simultaneously (see Fig. 3.1d). This would be the closest system to animal skeletal muscle systems that are well modeled by springs acting in parallel (internal to muscle tissue) and in series (tendons) to linear actuators (muscles) [35].

We can design the dual PAD in a manner similar to the parallel and series designs.

The actuator power of a dual PAD system is:

$$P_{a_d} = (\dot{y} - \dot{y}_{k_s})(F + F_{k_p}) \tag{3.18}$$

The actuator power, P_{a_d}, is affected by the parallel spring force, F_{k_p} in Eq. (3.2), and by the series spring velocity, $\dot{y}_{k_s}$ in Eq. (3.8). The energy consumed is now a function of three variables: parallel stiffness, parallel preload, and series stiffness. These variables can be initialized for an optimization routine by using the FDCF+ design for the two parallel variables and the dFVCF+ design for the series stiffness.

An optimized dual system can always outperform either an optimized parallel or series design because the dual system has additional variables compared to either single PAD design. As it turns out, for the class of trajectory described in this example (Fig. 3.2) with constant external forcing, an energy minimizing dual PAD can always completely eliminate actuator energy consumption if the parallel PAD is a constant force spring with $F_0 = -C$ and the series stiffness is $k_s = \omega^2 m$. In general this will not be the case.

Despite the energy advantages of the dual PAD, it is the most difficult to design. Because it has as many design variables as the two single PAD designs combined it will typically take more computational time to design the dual system than the total computational time to design the series and parallel PADs. Of course, when the PADs being optimized are linear and only a single trajectory is considered (as in this example) computational time is very quick even for the dual system (a few seconds). However, this could be a point of concern for complex non-linear PADs or when a distribution of maneuvers is considered [45]. The greatest difficulty in the design of a dual PAD is likely packaging constraints and implementation problems, especially if the goal is to augment an existing system.

3.2.4 Results

Table 3.1 summarizes the results of three different methods to design a parallel, series, and dual PAD for three different prescribed maneuvers. For all three maneuvers analyzed here $\omega = 1$ rad/s and $a = 1$ m (see Fig. 3.2 and Eq. (3.1)). The first maneuver analyzed ($n = 0$, $C = 0$ N) is simple harmonic oscillation. For this maneuver $k_p = k_s = \omega^2 m = 1$ N/m. Either the series or the parallel spring to the system would completely eliminate actuator power. A dual PAD also eliminates actuator power but does not provide an energy improvement over either single spring design.

The second maneuver ($n = 1$, $C = 0$ N) introduces periods of constant velocity

Table 3.1: A performance comparison of three PAD types: parallel, series, and dual. Each type is designed by up to four different methods: no spring, standard curve fit (FDCF/dFVCF), weighted curve fit (FDCF+/dFVCF+) and energy minimization. The PAD design variables, k_p, k_s, and F_0, energy consumed, f, and energy savings relative to the corresponding no spring design, S, are cataloged for three translational maneuvers.

		Parallel PAD				series PAD			Dual PAD				
Maneuver	Description	k_p	F_0	f	S	k_s	f	S	k_p	F_0	k_s	f	S
		[N/m]	[N]	[J]	[%]	[N/m]	[J]	[%]	[N/m]	[N]	[N/m]	[J]	[%]
$n = 0$, $C = 0$ N	No Spring	0	0	2.00	—	∞	2.00	—	0	0	∞	2.00	—
Sinusoidal Motion	FDCF/dFVCF	1.00	0	0.00	100	1.00	0.00	100	1.00	0	1.00	0.00	100
in Horizontal Plane	FDCF+/dFVCF+	1.00	0	0.00	100	1.00	0.00	100	1.00	0	1.00	0.00	100
	Min. Energy	1.00	0	0.00	100	1.00	0.00	100	1.00	0	1.00	0.00	100
$n = 1$, $C = 0$ N	No Spring	0	0	2.00	—	∞	2.00	—	0	0	∞	2.00	—
Sinusoidal Plus	FDCF/dFVCF	0.26	0	2.02	-1.3	0.33	4.00	-100	0.26	0	0.33	2.77	-38.6
Constant Velocity Motion	FDCF+/dFVCF+	0.23	0	1.91	4.3	1.00	0.00	100	0.23	0	1.00	1.14	42.9
in Horizontal Plane	Min. Energy	0.14	0	1.76	12.2	1.00	0.00	100	0.00	0	1.00	0.00	100
$n = 1$, $C = 10$ N	No Spring	0	0	102.8	—	∞	102.8	—	0	0	∞	102.8	—
Sinusoidal Plus	FDCF/dFVCF	0.26	-10	2.02	98.0	0.33	142.8	-38.8	0.26	-10	0.33	2.77	97.3
Constant Velocity Motion	FDCF+/dFVCF+	0.23	-10	1.91	98.1	1.00	62.83	38.9	0.23	-10	1.00	1.14	98.9
in Vertical Plane	Min. Energy	0.14	-10	1.76	98.3	1.00	62.83	38.9	0.00	-10	1.00	0.00	100

motion in between sinusoidal acceleration zones. This maneuver is dominated by inertial forces and thus is not well suited for parallel PADs. The FDCF design decreases performance, the FDCF+ design provides slightly increased performance, and the energy minimizing parallel design only provides a modest energy savings. series PADs perform much better — both the dFVCF+ and the energy minimizing design completely eliminate actuator power. (The dFVCF design, however, performed very poorly and illustrates the importance of using the weighted curve fit design to initialize optimizations.) The energy minimizing dual PAD design includes a parallel spring with $k_p = 0$ meaning that the best dual PAD is actually the same as an energy minimizing series design.

The third maneuver ($n = 1$, $C = 10$ N) adds a constant external load to the second maneuver (Figs. 3.3–3.5 correspond to this maneuver). The external forces are now large compared to the inertial forces making parallel PADs more suited to the task of energy minimization. This is because the parallel spring has a preload component, F_0, that affects actuator power. This preload can exactly compensate for the external load in all three parallel PAD designs, greatly reducing actuator power throughout the maneuver (see Fig. 3.4). The series PAD designs can eliminate power requirements during the sinusoidal portions of the maneuver (except for the dFVCF design) but cannot help during the periods of constant velocity — limiting how much a series PAD can reduce actuator power (see Fig. 3.5 and Eq. (3.10)). The energy minimizing dual PAD combines a constant force parallel PAD with $F_0 = -C$ and a series spring with $k_s = \omega^2 m$ to completely eliminate actuator power throughout the maneuver.

3.2.5 Design Selection & Reduction

We have shown that from an energy perspective different PAD types are preferable for different maneuvers. Series PADs tend to be superior to parallel PADs for maneuvers where the inertial forces dominate the external ones and vice versa. Furthermore, a dual PAD can always outperform a single PAD design. We propose three possible ways to select a PAD type for a general maneuver.

The first decision should be made based on practical constraints. Is this PAD intended to augment an already existing system? And if so, can a series or dual PAD easily be implemented? If the maneuver has a large change in position a parallel spring would encounter wind-up problems. On the other hand, if precise position control is essential, series and dual PADs may provide unacceptable performance.

If any of the PAD types could be acceptable for non-energy reasons then a decision needs to be made considering potential gains in performance against potential costs of design and implementation. The simplest way to do this is to directly compare the three energy minimizing designs (see Tab. 3.1). The obvious advantage of this approach is that it provides a clear summary of effectiveness of each design. The disadvantage of this approach is the computational cost as three optimizations are required. Although this cost is trivial for a simple system and a single maneuver, it could be significant if non-linear PADs are considered, if the system has many joints that need PADs, or if the objective is to make a PAD robust against variations in maneuvers.

For the example presented here, this is a good method for comparing three spring designs because optimization can be done very quickly, even for the dual PAD. An energy minimizing series PAD can reduce energy consumption by 30%, a parallel PAD by 97%, and a dual PAD by 100%. The series PAD is clearly inferior based on this criteria but both the dual and parallel designs perform very well. If energy efficiency is paramount then a dual PAD should be selected. However, the parallel design performs almost as well as the dual spring design and is simpler and easier to implement making it an attractive choice as well.

For more complex systems, PADs, maneuvers, or objective functions, performing three design optimizations in order to select a single design may be cost prohibitive. If this is the case, a PAD type can be selected based on the results of weighted curve fits. The advantage of this approach is only a single optimization is performed. However, this approach introduces estimation and it may be possible that for certain maneuvers a weighted curve fit performs poorly while an energy minimizing design performs well.

To select a PAD type prior to performing optimization the three weighted curve fit designs should be compared (FDCF+ for parallel, dFVCF+ for series, FDCF+ and dFVCF+ for dual). Any design that has a minimal effect on energy can be eliminated from contention. Finally, if the weighted curve fit dual design fails to provide at least a moderate improvement over the superior of the two single spring designs it may not be worth optimizing. Hopefully two of the three PAD types can be eliminated in this manner. However, even if one PAD type is eliminated this way, only two optimizations need to be performed instead of three.

3.3 3-Link Robot Arm Example

This method for selecting a PAD type can be applied to more complex systems such as a 3-link robotic manipulator arm. The arm we will consider is meant to be similar to an iRobot Packbot arm and the prescribed maneuver is intended to be typical of tasks that such an arm is likely to perform. Each joint on the arm is modeled with a non-backdrivable worm gear (which allows the arm to be held stationary without consuming any energy), a gearhead, and a DC motor. As the robot arm is comprised of three rotational joints we consider linear torsional springs for the PADs. The powertrain modeling is described in detail in Sec. 5.1.5 and is omitted here for brevity.

The maneuver under consideration involves the robot arm moving into and out of three typical poses: extending the arm vertically, looking under a vehicle, and lifting an object for closer inspection (see Fig. 5.3). This arm could be used for other maneuvers as well, but a PAD that improves performance for a typical maneuver will often be beneficial for other similar maneuvers [8]. Furthermore, although not discussed here, the PAD can be designed to be robust against variations in maneuver [45]. Once the system parameters and maneuver are defined, the required forces at each joint can be calculated via inverse dynamics. This reduces the problem from a 3-DOF problem to 3 1-DOF problems [45]. This data can then be used to find energy minimizing PADs where the optimization is initialized with the weighted curve fits

(FDCF+/dFVCF+).

3.3.1 Results

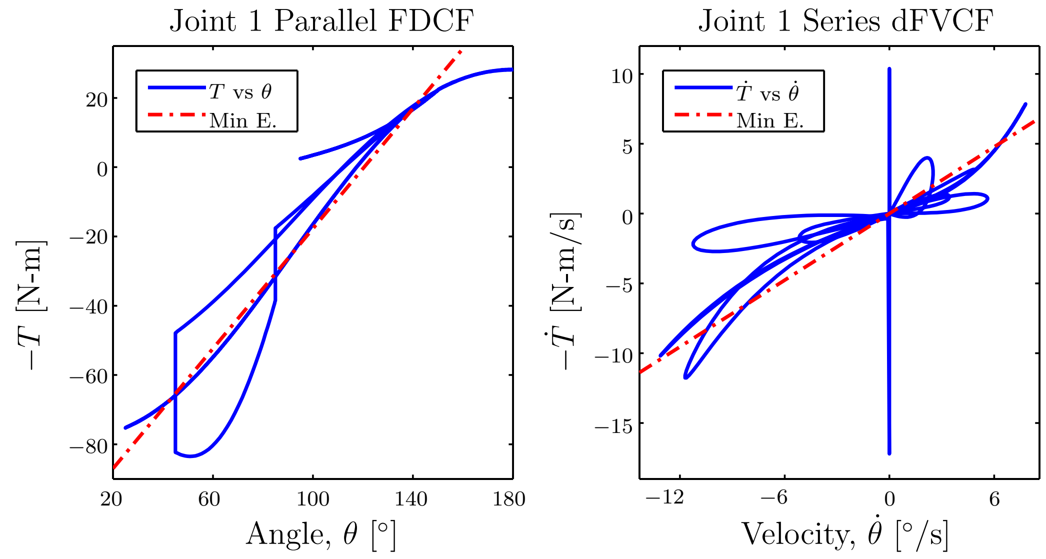


Figure 3.6: a) Required Joint 1 torque vs angle (solid blue) and energy minimizing torsional spring (dash-dot red). b) Derivative of Joint 1 torque vs angular velocity.

Table 3.2 summarizes the potential improvement of adding different energy minimizing PAD types to the various joints of a 3-link manipulator arm. Joint 1 connects the arm to the base. The required torque vs. angle, and derivative of torque vs. angular velocity for Joint 1 are plotted in Fig. 3.6. Qualitatively, the energy minimizing spring appears to be a good fit for both the parallel and series springs, suggesting that either design could substantially reduce energy consumption. The optimized parallel PAD performs very well — reducing energy consumption by almost 80%. The energy minimizing series PAD does not perform as well but can still reduce energy by almost 40%. The dual design combines a parallel and a series spring that are slightly softer than their single spring counterparts and reduce energy consumption by 90%.

Figure 3.6 illustrates a difference between how series and parallel designs are affected by specific maneuvers. The large vertical spike in Fig. 3.6b corresponds to the point in the maneuver where the arm is stationary and grabs a load (and when the load is later released). The height of the spike is inversely proportional to how long it takes to transfer the mass to the arm. In order to remain in the same position, the motor in a series configuration needs to wind the spring. Thus, how quickly the mass is transferred to the arm has a profound effect on the ability of the series spring to reduce motor energy requirements. Furthermore, if this transfer occurs too quickly,

Table 3.2: A performance comparison of how three different PAD types affect motor energy consumption in each joint of a three link manipulator arm. Each design minimizes motor energy consumption. The parallel torsional stiffness, k_p, parallel preload, T_0, series torsional stiffness, k_s, energy consumed, f, and energy saved compared to a springless design, S, are cataloged.

	PAD Type	k_p $\left[\frac{\text{N-m}}{\text{rad}}\right]$	T_0 [N-m]	k_s $\left[\frac{\text{N-m}}{\text{rad}}\right]$	f [J]	S [%]
Joint 1	Parallel	49.6	−104.0	—	219.8	78.9
	Series	—	—	45.6	639.9	38.5
	Dual	46.8	−99.8	39.9	103.4	90.1
Joint 2	Parallel	1.1	−16.5	—	205.4	65.4
	Series	—	—	∞	593.4	0
	Dual	1.1	−16.5	∞	205.4	65.4
Joint 3	Parallel	2.8	0.5	—	98.5	39.8
	Series	—	—	7.1	109.9	32.8
	Dual	2.8	0.9	9.1	51.6	68.4

the motor will not be able to wind the spring fast enough to match the prescribed maneuver. This problem is much less significant for a parallel spring. In order for a motor in a parallel configuration to match the new load the motor current needs to be raised. However, a step increase in current is much easier to implement than a step change in position, as is needed in the series case.

The motor in Joint 2 can be aided substantially by a parallel PAD. However, a series spring is unable to provide a benefit. Thus, the energy minimizing series spring has infinite stiffness, meaning that the best series spring is no spring. The dual spring design is the same as a parallel only design for this joint. The energy required of the Joint 3 motor can be reduced by 39.8% with a parallel PAD, 32.8% with a series PAD, and almost 70% with a dual PAD.

Joint 2 should be augmented with a parallel PAD. Parallel springs are also effective options for Joints 1 and 3 resulting in a total energy savings of 67.9% for the entire manipulator arm. Alternatively, using dual PADs for Joints 1 and 3 results in a total energy savings of 77.9%. This further reduction may be enough to justify the increased complexity and cost of using dual PADs instead of parallel ones in Joints 1 and 3.

3.4 Concluding Remarks

In this chapter we extended the maneuver based, passive assist design approach of our previous work to series and dual systems. We illustrated how the design processes for the series and parallel systems were analogous using simple mass-spring-actuator systems. To make the optimization converge more quickly we introduced a new initial design using a weighted force displacement curve fit FDCF+ for a parallel system or a weighted derivative of force velocity curve fit dFVCF+ for a series system. We provide engineering insight into why different types of PADs perform differently depending on the maneuver and offer guidelines on how to select a specific type based on the application. Specifically, parallel designs in general have greater potential when external forces dominate while series springs are often superior for maneuvers with large inertial forces. We also discuss how the addition of different PAD types can affect a system from a non-energy standpoint. Finally, we demonstrate this design process and selection procedure on a 3-link manipulator arm and found that a combination of parallel and dual PADs could reduce energy consumption by up to 78%.

CHAPTER 4

Application to Manufacturing Machines

In Sec. 1.4 we proposed a general method to design a Passive Assist Device (PAD) to augment an active joint (or axis) based on a prescribed maneuver. In Chap. 2 we demonstrated and experimentally validated this approach on a non-backdrivable single link robot arm using a parallel, torsional spring as a PAD. In Chap. 3 we examined how the PAD design approach could be used to design series and dual PADs discussed when those designs would be preferable to parallel designs. In this chapter we show that our proposed design approach is quite general and may be used for any actuated system performing a periodic dynamic process including linear motion drives used extensively in the manufacturing industry.

4.1 Manufacturing Introduction

Every year 45 billion kWh are spent in the fabricated metal products sector (e.g., machining, stamping) in the United States [9]. At \$0.07/kWh [7], this corresponds to an annual electric cost of \$3.15 billion. This quantity not only includes the cost of directly shaping each work piece, but also auxiliary costs including positioning the tool and/or workpiece, waste material collection, cooling, etc. Because the energy costs are so large a great deal of work has gone into making the entire manufacturing process more efficient. This can be done by streamlining the entire manufacturing system as well as by making each individual operation more efficient [10, 11].

Positioning machinery is particularly well suited to being augmented with a PAD because, in a mass production environment, they undergo the same periodic motions thousands or millions of times. Positioning costs can vary dramatically depending on the operation. If there are no external forces, positioning may only account for 4% of energy consumption [11]. However, if external forces are large (e.g., moving the workpiece while milling), positioning costs could exceed 60% [12]. Furthermore, the faster these processes occur the larger the energy cost [13]. The results in this paper show that our approach can reduce these losses by up to 79%.

The benefits of our approach may not be realized for all motions on all machines. However, for machines repeatedly executing similar maneuvers, an optimized PAD could provide a substantial improvement in performance. Furthermore, in many instances it may be possible to redesign the trajectory of the workpiece (and/or tool) in order to fully take advantage of a PAD without affecting the final product.

Springs are commonly used in manufacturing machines to eliminate backlash in gear transmissions, as thrust bearings, or to ensure secure connections with tools. However, they are not widely used as a means to reduce motor effort or energy costs. There are examples of springs designed into linear actuator systems that open doors [65] or power speakers [66]. The springs in these systems are designed to provide sufficient force to return the mechanism to its base configuration when the actuator is not powered. They are not designed based on a known maneuver nor is their primary function to reduce motor effort or energy costs. PADs designed by our general approach are much more akin to devices used in robotics as described in Sec. 1.2.2.

There are two significant original contributions presented in this chapter:

1. We demonstrate that the proposed design approach can effectively be applied to any augmented joint or axis executing a prescribed, repeated maneuver, including linear motion drives used extensively in the manufacturing industry [14].

2. We show that in certain situations (e.g., manufacturing machinery executing the same task thousands or millions of times) it is possible to modify the maneuver so that the addition of an optimized PAD can effectively increase total system performance without affecting the product [14].

The remainder of this chapter is organized as follows: First we illustrate the process on a simple single axis linear motion drive. In a second example we show how the general method can be applied to more complicated systems and maneuvers such as a two-DOF X-Y table performing a milling operation. In this second example we show the potential benefits of redesigning the maneuver prior to designing the PAD.

This chapter is based on the work presented in [14].

4.2 Example 1 — Single Axis Placement Table

The general methodology described in Sec. 1.4 can be applied to machines used extensively in the manufacturing industry such as linear motion drives. In this example we consider a single axis (or stage) positioning table. Such a table can be used for countless applications including moving a workpiece underneath a tool or sensor, holding the piece in place during the operation, and then pulling the piece back to its original location, where the finished piece is swapped out for a new piece — the cycle can then repeat.

4.2.1 Apply Passive Assist Device General Methodology

The machine architecture is simple in this example. The table has a single prismatic joint and the mass of the moving table and workpiece is m. The system consists of the table driven by a ballscrew via a gear head and DC motor. The trajectory is described as follows:

(a) A workpiece at rest is accelerated to a different location where it comes to rest.

(b) The workpiece is held in place while some operation is performed on it (e.g., inspection, painting, welding, or drilling).

(c) The workpiece returns to its original position.

(d) The table stays in place while the finished workpiece is removed and replaced by a new one.

The trajectory in parts (a) and (c) follow a fifth order polynomial curve in position as shown in in Fig. 4.1. The process occurs in the horizontal plane and no axial forces are imparted on the table in parts (b) and (d) of the trajectory. Therefore, there is no external forcing in this example. Calculating the inverse dynamics in this example is trivial because of the single DOF. Thus, the force required from the prismatic joint to execute the prescribed maneuver is $F = m\ddot{x}$, where x is the prescribed position of the table.

Because the stage translates along an axis we choose a linear extension spring of stiffness, k, with preload, F_0, with the following force characteristic:

$$F_k = kx + F_0 \tag{4.1}$$

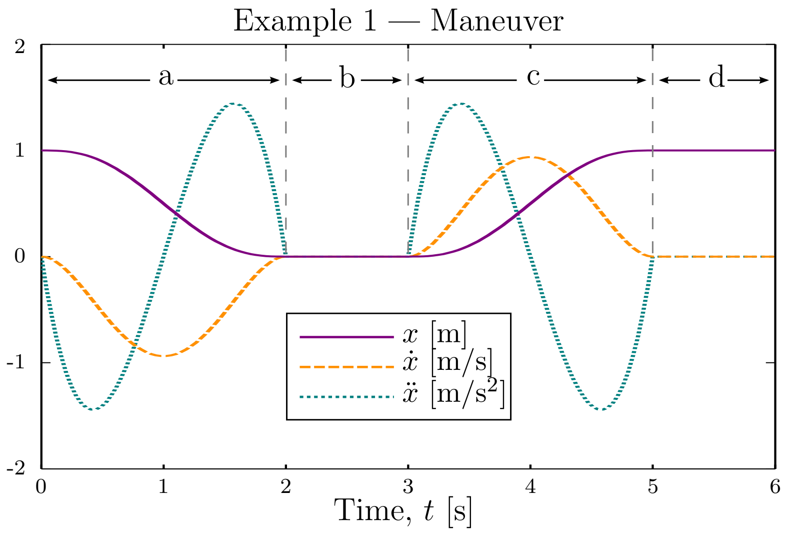


Figure 4.1: Trajectory of the single stage placement table. The displacement (solid purple) begins 1 meter from the tool then follows a fifth order polynomial curve (a) for two seconds until it comes to rest under the tool. The table is held in place under the tool for 1 second (b) before it moves back to its starting location (c) and remains at rest until the next cycle (d). The corresponding velocity (long dashed gold) and acceleration (short dashed teal) are also shown.

The axis powertrain could have any number of designs or components. We consider a representative linear actuator powertrain consisting of a ballscrew, gearhead, and DC motor. Ballscrews can be modeled by the following equation [67]:

$$T = \frac{(F + F_k) \times l}{2\pi\nu} \tag{4.2a}$$

$$\dot{\theta} = \frac{\dot{x} \times 2\pi}{l} \tag{4.2b}$$

Where T and $\dot{\theta}$ are the torque and speed required of the gearhead and l and ν are the lead and efficiency of the ballscrew, respectively.

The gearhead and DC motor can be modeled using equations developed in our previous work for a single joint robot arm [44]:

$$i = \frac{T/n + T_f \times \text{sgn}(\dot{\theta})}{k_m} \tag{4.3a}$$

$$V = iR + k_m \times \dot{\theta} \times n \tag{4.3b}$$

$$P_e = V \times i \tag{4.3c}$$

Where i, V, and P_e are the motor current, voltage, and electrical power required at each time step, n is the gear ratio, and R, k_m, and T_f are the motor resistance, torque constant, and internal friction torque, respectively.

Table 4.1: Key parameters used in Example 1. Table and ballscrew parameters are based on a HIWIN KK130 Single Axis Drive [3], while gearhead and DC motor parameters are based on a Micromo 2642 012 CXR [4].

Parameter	Definition	Value
m	Moving Mass	10 kg
l	Ballscrew Lead	25 mm/rad
ν	Ballscrew Efficiency	0.9
n	Gear Ratio	2
R	Resistance	1.46 Ω
k_m	Torque Constant	54 mN-m/A
T_f	Motor Friction	1.7mN-m

In this example we wish to find a linear extension spring that minimizes motor energy consumption. Thus, we formulate the following gradient based optimization problem:

$$\begin{aligned} \text{minimize} \quad & f(k, F_0) = \int_0^{t_f} |P_e(t, k, F_0)| \, \mathrm{d}t \\ \text{with respect to} \quad & k, F_0 \\ \text{subject to} \quad & -k \leq 0 \\ & \max(|i(t, k, F_0)|) \leq i_{max} \\ & \max(|V(t, k, F_0)|) \leq V_{max} \\ & \max(|P_e(t, k, F_0)|) \leq P_{e_{max}} \\ \text{and} \quad & 0 \leq t \leq t_f \end{aligned} \tag{4.4}$$

We solve this problem with MATLAB's built in function *fmincon* initialized with a Force Displacement Curve Fit (FDCF) design [43]. A weighted Force Displacement Curve Fit (FDCF+) design could be used to initialize the optimizations presented in this chapter and likely would decrease the computational time (see Sec. 3.2.2). However, in this chapter the optimization routine converges quickly even with the unweighted FDCF designs. The motor performance limit constraints are typically inactive, but including these constraints in the problem ensures that we are only testing feasible maneuvers.

4.2.2 Results

Performing the optimization described in (4.4) yields a spring design that greatly reduces the energy required to execute the maneuver in Fig. 4.1. Table 4.2 summarizes the results. The two main observations are that (a) employing our methodology provides a very significant benefit compared to the original system and (b) an optimized PAD can significantly outperform a FDCF based design.

Table 4.2: Simulated results for three different spring designs: no spring, FDCF, and energy minimization. For each design the spring variables (k, F_0), total energy consumed (f, [J]) and improvement ([%]) over the no spring design are cataloged.

Spring	k	F_0	f	
Design	[N/m]	[N]	[J]	[%]
No Spring	0	0	19.7	0
FDCF	8.9	-4.4	14.8	24.6
Min. Energy	34.7	-17.4	4.25	78.4

Figure 4.2 illustrates that the addition of a PAD can improve system performance by handling some of the load that had been required of the motor. The plot shows the required force vs displacement for the axis throughout the maneuver (solid blue line). A force vs displacement plot is particularly useful because the axes represent the fundamental units of springs. An ideal spring would match the required force-displacement curve and completely eliminate the need for the motor. In general this cannot be achieved because the axis can have multiple force values at a given position. However, even with a one-to-one relationship like the one depicted here, only a nonlinear spring could achieve exact force compensation and such a spring is much more difficult to design and implement than a linear one.

The FDCF based spring design (dashed green) does not appear to be a great fit to the curve shown in Fig. 4.2. The reason is that one third of the maneuver time is spent at 0 force and a position of either 0 or 1m. These data points decrease the slope of the FDCF design in Fig. 4.2. However, it is clear in the top plot of Fig. 4.3 that implementing the FDCF design evenly distributes required motor torque throughout the maneuver — decreasing energy consumption. This reasoning is completely analogous to why a series derivative of Force Velocity Curve Fit (dFVCF) design is such a poor fit to the data in Fig. 3.3b. We expect that a FDCF+ would be able to significantly outperform the unweighted FDCF based on the results of Chap. 3.

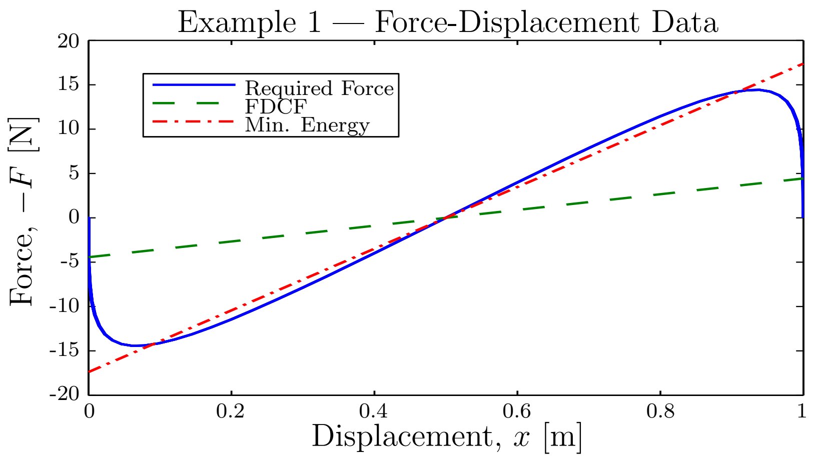
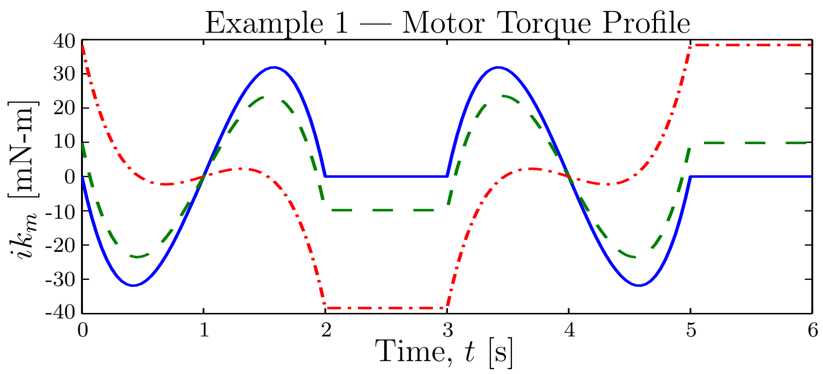


Figure 4.2: Overlays the required motor force as a function of displacement to the force-displacement curves of two spring designs: a FDCF design and a design that minimizes the energy required to execute the prescribed maneuver. This plot helps to illustrate the fundamental idea behind our methodology: a well designed PAD can reduce required motor effort.

The energy minimization design is able to outperform the FDCF design because it has access to more information, specifically the velocity profile of the joint and the parameters and equations that describe the powertrain. Because the table is stopped at the limits of its motion very little energy is required during those phases even if a large torque is being exerted by the motor. This allows for an optimized spring design (short dashed red) to closely match the required forces of the joint during the "expensive" moving parts of the maneuver at the expense of requiring additional energy consumption during the "cheap" stationary parts of the maneuver. These effects are clearly illustrated in the bottom plot of Fig. 4.3.

Even though the system analyzed in this example is simple, such systems are widely used in a manufacturing setting. Thus, employing the PAD design methodology proposed in this paper (with which nearly 80% energy savings were attained) could have a very large industry-wide energy-saving impact. Furthermore, motors have to be sized large enough to execute the most extreme tasks in the maneuver. Often these are accelerations corresponding to reversing direction. Adding PADs can directly reduce these costs meaning a smaller motor could be used to perform the same task. This could ultimately lead to an even greater energy savings due to the weight reduction of using smaller motors. This also could save money as smaller motors are less expensive than larger ones. Including the design of the motors (or other powertrain components) in addition to the PADs is an area of future research.



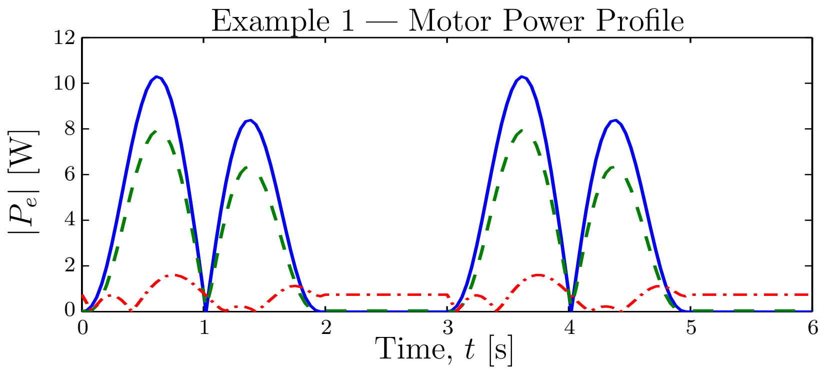


Figure 4.3: Compares the required motor torque (top) and power (bottom) when the system has no spring (solid blue), a FDCF spring (dashed green) and an energy minimizing spring (dash-dot red). These plots provide information not captured in a force-position curve (see Fig. 4.2) and illustrate why there is significant difference in spring designs and performance. It is clear that the total energy consumed is much less with the energy minimizing spring compared to the FDCF spring.

4.2.3 Harmonic Motion Case

Suppose we used the same system but altered the trajectory shown in Fig. 4.1 such that the duration of part (b) and (d) were reduced to zero time and the motion of parts (a) and (c) were sinusoidal instead of polynomial (i.e., simple harmonic oscillation). In such a maneuver the required motor force-displacement curve (Fig. 4.2) would be linear and both the FDCF spring and the energy minimizing spring would overlap the required motor force profile exactly. In a friction-less system, the force and energy required from the motor would be completely eliminated. In a real system the motor still has to exert a small effort to overcome internal friction but most of the effort and energy involved in moving the work piece would be handled by the spring.

4.3 Example 2 — X-Y Table Milling Operation

The general design method proposed in this thesis can also work on more complicated machining systems. For example, consider an X-Y table that moves a workpiece underneath a mill, performs a milling operation, and then returns the work-piece to its original location. This example differs from the example in the previous section because the machine has more actuated DOFs (i.e., x and y) and because there is now external forcing (i.e., cutting forces) present in the maneuver. Two dimensional motion also allows for more generalized trajectories.

4.3.1 Apply PAD General Methodology

The machine in this example is essentially two copies of the machine in the previous example — the second mounted on top of the first and perpendicular to it. The prescribed maneuver in this case is the slot milling of “U M” as shown in Fig. 4.4. When moving without cutting, the trajectory is described by a 5th order polynomial path similar to the one used in Example 1. When slot milling, the motion has a slow constant speed (i.e., feedrate) of 40.1 cm/min but there are significant external forces. We consider a flat end mill with a straight cutting edge insert made of uncoated tungsten carbide spinning at 4010 rpm with a 15.88 mm diameter cutting a 1 mm depth slot through P-20 mold steel. The corresponding average cutting forces tangent and normal to the path are $F_t = 150$ N and $F_n = 200$ N, respectively [68].

The inverse dynamics remains simple in this example due to the orthogonal nature of the axes.

$$\begin{bmatrix} F_x \\ F_y \end{bmatrix} = \begin{bmatrix} m_1 & 0 \\ 0 & m_2 \end{bmatrix} \begin{bmatrix} \ddot{x} \\ \ddot{y} \end{bmatrix} + \begin{cases} \begin{bmatrix} \cos\Theta & -\sin\Theta \\ \sin\Theta & \cos\Theta \end{bmatrix} \begin{bmatrix} F_t \\ F_n \end{bmatrix} & \text{if milling} \\ \mathbf{0} & \text{otherwise} \end{cases} \tag{4.5}$$

Where $m_1 = 10$ kg is the mass of the workpiece and moving parts of the x-stage and $m_2 = 64$ kg is the mass of m_1 plus the masses of the stationary parts of the x-stage and the moving parts of the y-stage. Θ is the current direction of motion: $\Theta = \arctan(\dot{y}/\dot{x})$.

The powertrain parameters for each axis in this example match the parameters described in the single axis example. Furthermore, the same spring topology is used for these axes as in the previous example and the optimization is performed in the same way.

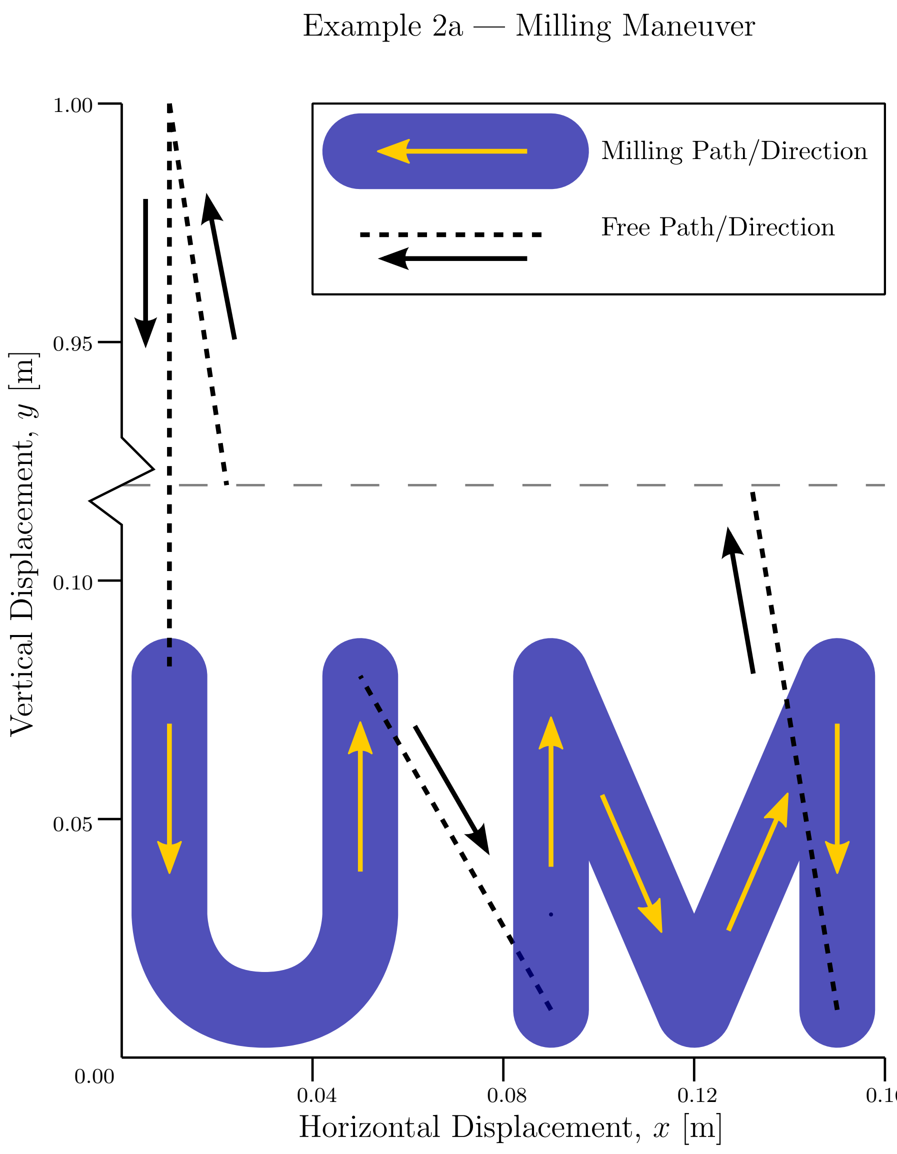
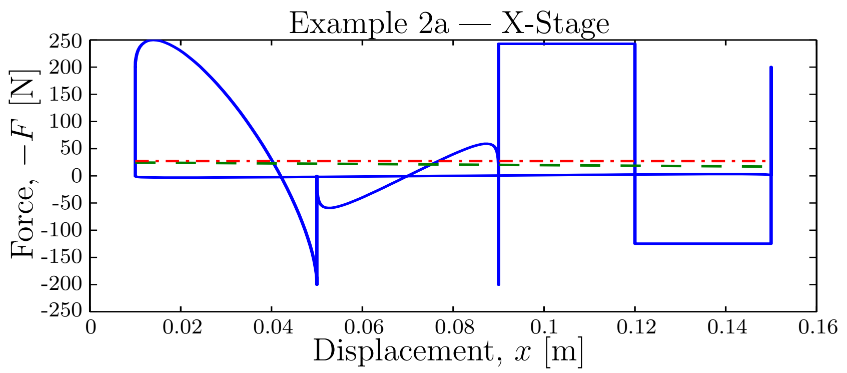


Figure 4.4: Motion of an X-Y table performing a slot milling operation.

4.3.2 Results

Figure 4.5 shows the same type of information as presented in Fig. 4.2 but corresponding to each axis. Because the maneuver has been decoupled, a PAD can be designed for each axis independently. Unlike Example 1, it is no longer clear what is going on by looking only at this plot. The plot is still useful for its ability to provide an FDCF design via a linear least-squares approximation.

The energy minimizing springs designed to assist the actuators in these axes are not very effective and improve performance by a mere 2.1% on the x-stage and 3.6% on the y-stage. There is no linear spring that could perform better because any assistance provided in a certain part of the maneuver would be more than offset by



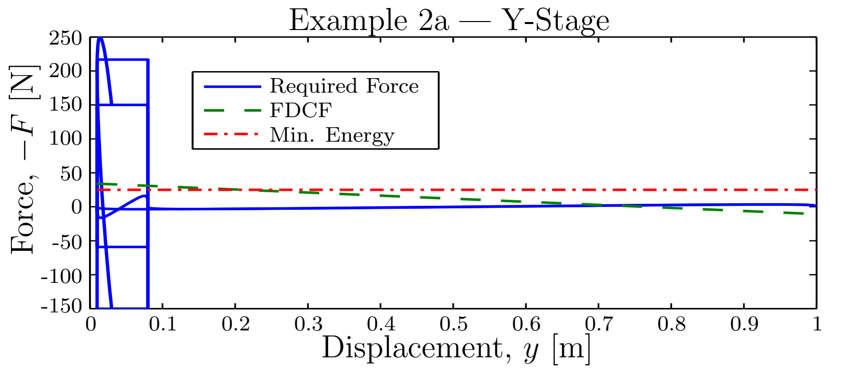


Figure 4.5: Force-displacement curves for the x-stage (top) and y-stage (bottom) actuators. Once the trajectory and required forcing are known for each stage, PADs can be designed one axis at a time. The required axis force-displacement data (solid blue) are overlaid with the force deflection curves of springs based on FDCF (dashed green) and energy minimization (dash-dot red) methods.

an increased hindrance during a different part. If the spring were unable to help at all it would provide zero force over its entire range (i.e., the best spring would be no spring). In Fig. 4.5 it is clear that the optimal designs are close to a zero-force spring, and therefore cannot be effective at reducing motor torques or energy consumption.

4.3.3 Maneuver Redesign

In the above example an optimized spring can only achieve a small reduction in energy. In general, the closer the force-displacement profile can be approximated by a linear spring, the better the spring will aid the system. One way more energy can be saved is if the maneuver and PAD are designed together. Significant research has gone into the simultaneous optimization of springs and gait designs in walking robots [41, 69]. These works demonstrate the potential advantages of simultaneous maneuver and spring optimization including the performance of simultaneous optimization vs sequential optimization [69]. However, the nature of simultaneous maneuver and PAD optimization introduces a number of complications that we leave for future work.

Table 4.3: A comparison of the effectiveness of spring designs on energy consumption and the significance of redesigning a maneuver prior to designing springs. For each design the spring variables (k, F_0), total energy consumed (f, [J]) and improvement ([%]) over the original, no-spring design are cataloged. Note: the FDCF design for the original maneuver is unfeasible because k has a negative stiffness.

Maneuver	Spring Design	X Stage k [N/m]	X Stage F_0 [N]	X Stage f [J]	X Stage f [%]	Y Stage k [N/m]	Y Stage F_0 [N]	Y Stage f [J]	Y Stage f [%]
Original (Fig. 4.4)	No Spring	0	0	2253	—	0	0	1352	—
	FDCF	−53.9	25.1	—	—	−45.3	34.6	—	—
	Min. Energy	0	27.2	2205	2.1	0	25.0	1304	3.6
Redesign (Fig. 4.6)	No Spring	0	0	2257	−0.2	0	0	1356	−0.3
	FDCF	35.9	92.0	769	65.9	104.4	−82.2	581	57.0
	Min. Energy	276.7	136.9	466	79.3	176.2	−127.1	476	64.8

Although simultaneous optimization is a difficult problem, we demonstrate that an intuitive redesign of the maneuver performed in Fig. 4.4 followed by an optimization and incorporation of a PAD can result in much greater energy savings than in the preceding case. One cause of the small energy savings was that the force-displacement data was relatively evenly distributed about the horizontal (displacement) axes. If we were to mill the same shape as before but perform all the cuts in a downward (i.e., negative y) direction (see Fig. 4.6), the data for each stage's force-displacement plot would be shifted away from the horizontal axes. Therefore, the addition of an energy minimizing PAD should be able to have a much larger impact.

A comparison of Figs. 4.5 and 4.7 shows that the maneuver redesign achieved the desired effect. Instead of the force-displacement data being nearly evenly distributed about the horizontal axes, the data is now predominantly on one side of the axes. This distribution allows for springs designs that are more effective at assisting the motor over the entire course of the maneuver.

In this example, it is clear that both the maneuver redesign and PAD implementation are critical to achieve significant energy savings while neither is effective by itself. Without the maneuver redesign, a mere 2.7% energy savings can be achieved. On the other hand, redesigning the maneuver with the intention of adding a PAD actually increases energy consumption by 0.2% if a PAD is not implemented. Performing all cutting in the downward direction necessitates additional steps be added to the maneuver where the table would backtrack to a previous position without milling. Some additional energy is needed to perform these extra steps. However, the extra energy spent executing the additional steps is very small compared to the additional

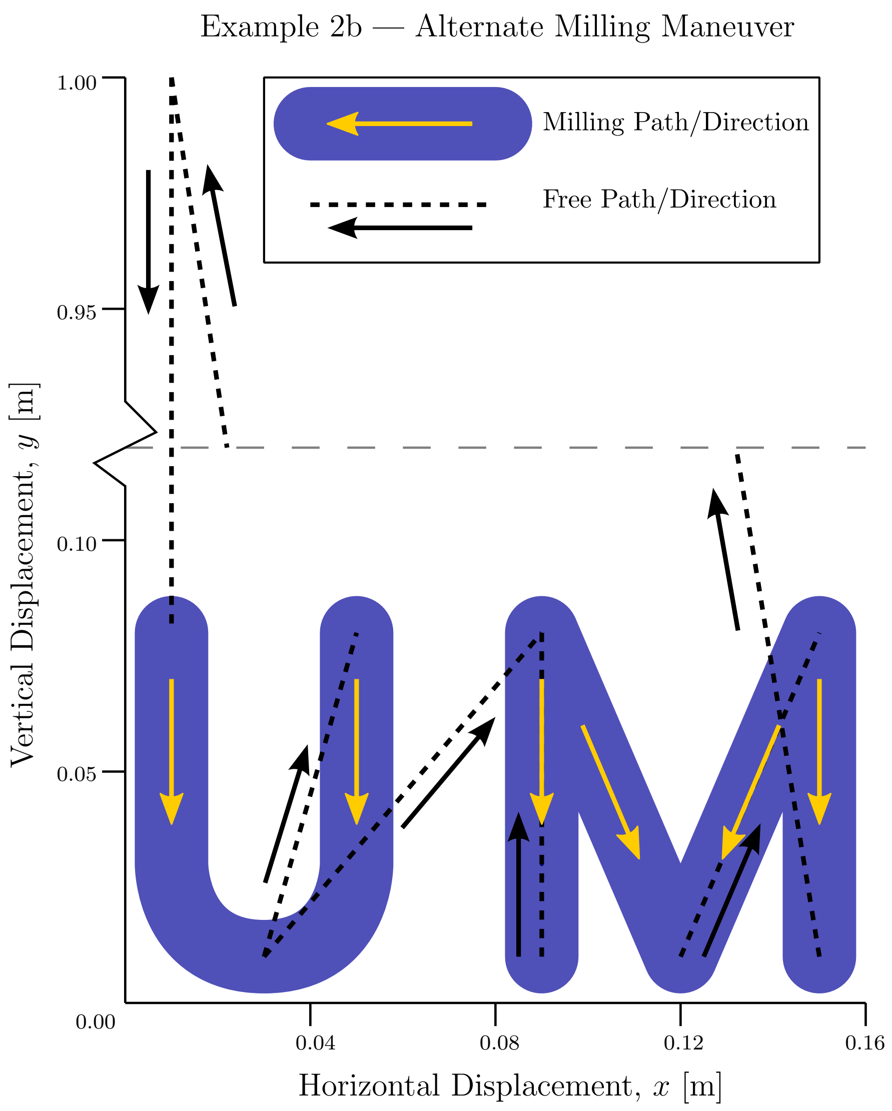


Figure 4.6: Alternative path to cut the same final “U M” shape. In this maneuver the cuts always occur in the negative y direction. Changing the maneuver in anticipation of adding a PAD can provide a huge increase in performance even though additional non-milling steps need to be added to the trajectory.

savings that can be achieved with PADs optimized for a maneuver designed with the addition of PADs in mind (73.9% total energy saved). Furthermore, Fig. 4.8 shows that because of the redesign, springs are able to greatly reduce motor power during most of the maneuver. These savings far outweigh the increase in energy required to perform the additional non-milling operations because those operations are brief compared to the milling ones.

We demonstrated that alternating the original maneuver to take advantage of a PAD can drastically reduce positioning costs. We could reduce costs even further

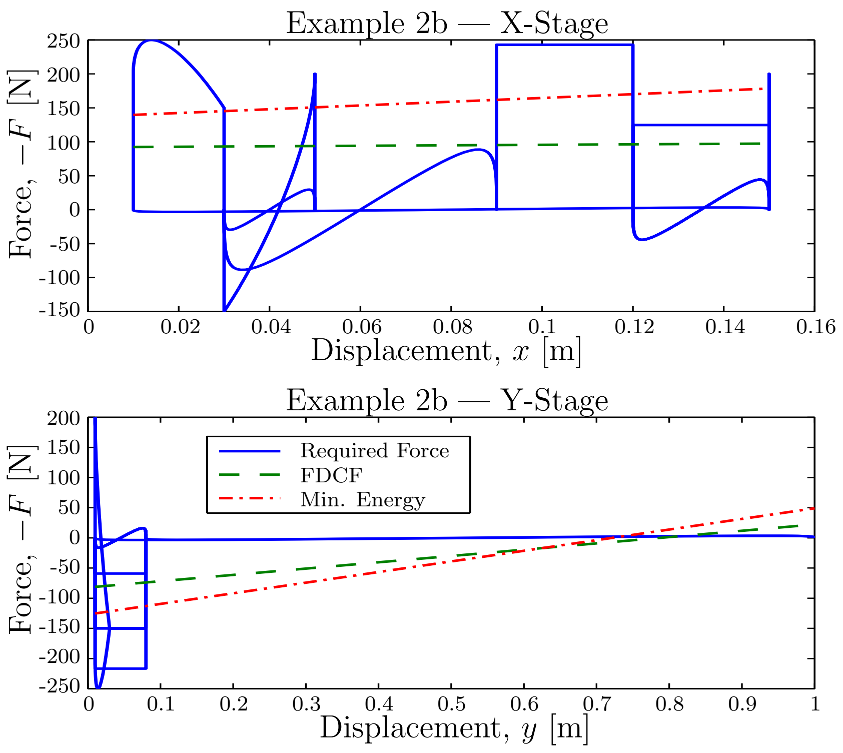
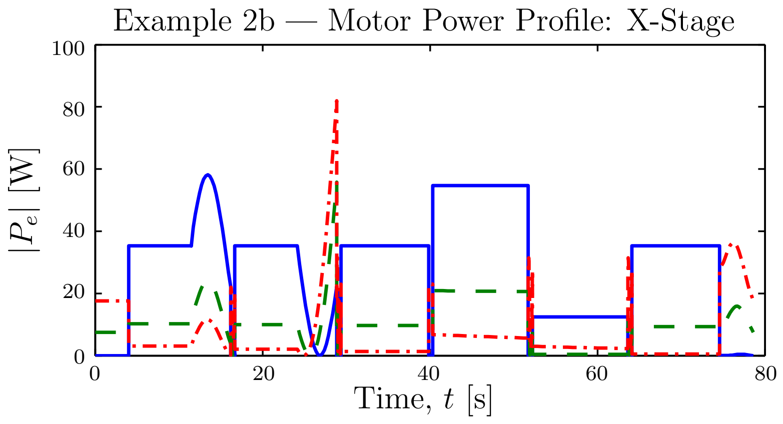


Figure 4.7: Provides the same information as Fig. 4.5 but for the alternative maneuver where cuts are always milled in the negative y direction (see Fig. 4.6). The spring profiles are now much farther away from the horizontal axes suggesting that they will have a much larger impact on the required motor torque and energy consumption.

by optimizing the maneuver and PAD simultaneously. It should be noted that we only considered reducing positioning costs and and other differences between the two maneuvers should be considered in a more complete analysis. For example, in the original maneuver the mill only needs to plunge into the metal plate twice (once to start each letter) but in the alternate maneuver four plunges are required. These additional plunges will require additional time and energy and should be accounted for when optimizing the entire system.

Maneuver redesign may not be practical in many situations. For example, if a machine is used in a flexible manufacturing setting and frequently performs different operations, redesigning the maneuver and spring each time will not be cost effective. On the other hand, if a machine is going to perform the same task thousands of times — as in a dedicated machining line — redesigning the maneuver to maximize the performance of an energy minimizing PAD could be quite beneficial.



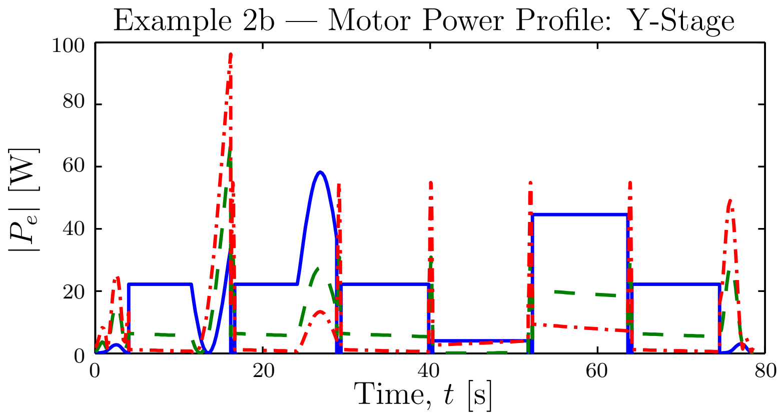


Figure 4.8: Power profiles of each stage executing the maneuver depicted in Fig. 4.6. For each stage, the addition of a PAD greatly reduces power requirements during milling, while increasing power consumption during non-milling. However, it is clear that the net effect is a significant reduction in energy consumption for each stage.

4.4 Concluding Remarks

In this thesis, we have proposed a general methodology to increase the performance of active systems by augmenting them with optimized, PADs. This concept was experimentally demonstrated on a single link robot arm augmented with a torsional spring in Chap. 2. In this chapter we showed that the concept can effectively be applied to other machines performing prescribed periodic motions by simulating a reciprocating single axis table and an X-Y table performing a slot milling operation. The addition of energy minimizing springs yielded a decrease in energy consumption of 65–79%. Finally, we showed that a significant increase in performance can be realized if the maneuver is redesigned considering that a PAD will added to the system.

CHAPTER 5

Robust Design Example

In Sec. 1.4 we introduced a method that considers the loading profile of an active system over the course of a known, prescribed maneuver and determines the design of a Passive Assist Device (PAD) that would best counteract those forces. In Chap. 2 we experimentally demonstrated that an optimized spring augmenting an active joint in a single link robot arm executing a known maneuver can significantly reduce energy consumption [1, 44]. This chapter presents two significant original contributions:

1. The general design process is demonstrated on multi-DOF systems, specifically multi-link robotic manipulator arms.
2. The design methodology may also accommodate machines that execute a distribution (or family) of maneuvers. Given the distribution, a PAD can be designed to be robust against variation in the maneuver [45].

Our general PAD design approach could be particularly useful for Unmanned Ground Vehicle (UGV) manipulator arms that have two major mechanical shortcomings that can both be improved with the addition of PADs. First, unlike traditional industrial robots, which perform preprogrammed tasks in a well-structured environment and are among the most reliable machines available, the current generation of unmanned ground vehicles exhibit a mean time between failure on the order of 10 hours. UGVs are machines with numerous actively controlled degrees of freedom, and joint or motor problems are a common failure mode. Second, because UGVs are battery powered the energy available on any given mission is limited. Reducing the energy requirements of arm manipulation allows for more power to be spent traversing ground, communicating, performing additional arm maneuvers, or increasing mission length. In a UGV manipulator a PAD can be placed in parallel with motors and designed to reduce the maximum motor torques required for a particular maneuver, thereby increasing efficiency and reliability.

A UGV manipulator arm also distinguishes itself from its industrial cousins because a UGV arm operates in an open environment and performs a variety of tasks instead of precisely repeating maneuvers. However, such tasks can often be restricted to a family of similar trajectories and loads. Consequently, in this chapter we show that properly designed PADs can significantly reduce peak torques and, thus, improve reliability and energy efficiency so long as the maneuver being performed is close to the maneuver the PAD was designed for. Furthermore, we present a robust design approach where, if the maneuver can be characterized by a known distribution, a PAD can minimize the effects of maneuver variation on system performance.

The remainder of this paper is organized as follows: First we illustrate the PAD design process for a 3-link UGV manipulator arm performing a single trajectory. We then show how the performance of this design is affected as the variability of the maneuver increases. Finally, we consider a family of maneuvers and design a PAD to be robust against maneuver variation.

This chapter is based on the work presented in [45].

5.1 Apply PAD Design Methodology

In this chapter we apply the general, six step design method described in Sec. 1.4 to improve the performance of a 3-link UGV manipulator arm with optimized, PADs.

1. **Define Machine Architecture**: We consider a 3-link manipulator arm similar to the arm on a standard iRobot Packbot. Parameters characterizing this system are listed in Table 5.1.

2. **Define Trajectory and External Forcing** The trajectory and external forcing of the machine must be known in order to design PADs with this method. In previous chapters we have shown that this method is effective when the maneuver is precisely repeated thousands or millions of times — as is often the case with dedicated manufacturing machines [8, 14]. However, a substantial increase in performance may still be realized if the maneuver is not precisely known so long as the prescribed maneuver is typical of actual maneuvers [44].

   In this chapter we explore the effect of maneuver variations on PAD design and performance. The nominal maneuver considered in the chapter is described in Sec. 5.2. We also consider variations about this maneuver.

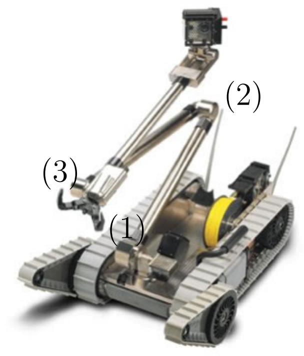


Figure 5.1: A Packbot UGV with a 3-link manipulator arm. The arm has three joints: Joint 1 connects the arm to the chassis, Joint 2 is the first elbow, and Joint 3 is the second elbow. The arm has a camera at the end of the third link and a gripper located at Joint 3. Photo: iRobot.com

3. **Perform Inverse Dynamics / Decouple Joints** Given the machine's architecture, trajectories of all joints, and the external forcing, a set of joint torque-angle profiles can be calculated using inverse dynamics. This step transforms an n-DOF system into n 1-DOF systems (see Fig. 5.2). PADs can then be designed for each joint independently. This step is simple for Cartesian machines [14], but can be complex for other machines such as the robot arm presented here.

   The equations of motion of a generic serial robot manipulator can be written as

$$\underline{Q} = \mathbf{M}(\underline{q})\underline{\ddot{q}} + \mathbf{C}(\underline{q}, \underline{\dot{q}})\underline{\dot{q}} + \mathbf{G}(\underline{q}) \tag{5.1}$$

   Where $\underline{q}$ is a vector of joint angles, $\underline{Q}$ is a vector of joint torques, and $\mathbf{M}$, $\mathbf{C}$, and $\mathbf{G}$ are the mass, centripetal/Coriolis forces, and external forcing matrices, respectively. Inverse dynamics solves for $\underline{Q}$ given $\underline{q}$. Solving such an equation analytically with a Lagrange formulation is cumbersome and computationally expensive. A solution can be found much more quickly and easily using a Newton-Euler recursion algorithm [70, 71], which has been compiled in software packages such as Corke's robotics toolbox for MATLAB [72].

   The Recursive Newton Euler formulation is comprised of three main steps. The first step is the outward recursion. Starting at the base and for each time step in the trajectory, the motion of the center of mass (COM) of each link can be determined based on the COM of the previous link and motion of the previous joint. The second step is to calculate the forces and moments acting on the each links' COM. The third step is to perform an inward recursion. Starting from the end effector, the torque and force supplied by each joint has to be sufficient to support the torque and force of the next most outward joint plus the forces and moments acting on the COM of the corresponding link [72].

The 3-link UGV arm described in Tab. 5.1, executing the maneuver described in Tab. 5.2 and Fig. 5.3 needs to produce the torque-angle profiles at each joint (solid blue) shown in Fig. 5.2. The PAD can supply some of the torque required to execute the maneuver (dash-dot red), reducing the load required of the motor and the energy consumed. However, the total torque acting on the joint remains the same and, thus, the PAD design of one joint has no effect on the design of PADs for other joints.

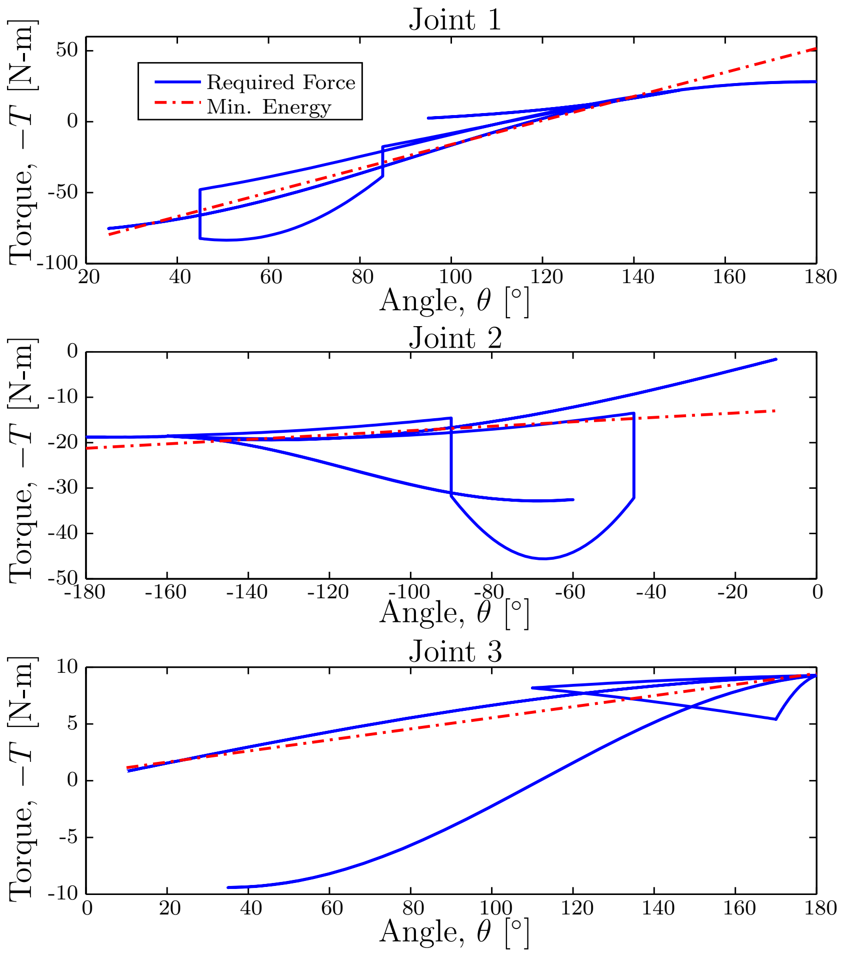


Figure 5.2: For the 3-link UGV arm described in Tab. 5.1 to execute the maneuver described in Tab. 5.2, the joints must follow the torque-angle profiles (solid blue). The PADs that minimize the energy consumed by each motor (dash-dot red) can be designed for each joint independently.

4. **Select Passive Assist Device Topology** There are many different PADs that

could be considered including springs, pistons, pressurized tanks, capacitors, etc. For a moderately sized mechanical system a spring is a practical choice for a PAD.

Because each joint is rotational we will design PADs that are linear torsion springs of stiffness, k, with preload, T_0, with the following torque characteristic:

$$T_k = k\theta + T_0 \tag{5.2}$$

5. **Model Each Joint's Powertrain** In order to design a PAD that minimizes an objective function, $f(k, T_0)$ (see (5.5) below), the parameters defining the joint powertrain must be known beforehand. There are many designs that could be considered and we present a representative case. A typical UGV arm joint powertrain could include the following components: non-backdrivable (NBD) worm gear, gear head, and DC motor. In addition to providing a speed reduction the worm gear allows the arm to remain stationary in a raised configuration without expending any energy. The torque needed to drive the worm, T_w, of a NBD worm gear transmission given the required torque at the joint, T, can be calculated from the following equation of motion [2]:

$$T_w = \begin{cases} 0 & \text{if stuck} \\ (J_w i_{wg} + C_1 J_g)\ddot{\theta} + C_1(T + T_k) & \text{if load driven} \\ (J_w i_{wg} + C_2 J_g)\ddot{\theta} + C_2(T + T_k) & \text{if motor driven} \end{cases} \tag{5.3}$$

Where J_w, J_g are the inertia of the worm and worm gear, respectively, i_{wg} is the speed ratio, and C_1, C_2 are the torque ratios of the load driven and motor driven cases, respectively. Which operating condition is active can be determined at each time step [44].

The gearhead and DC motor can be modeled as in our previous work for a single-joint robot arm [44]:

$$i = \frac{T_w/n + T_f \times \operatorname{sgn}(\dot{\theta})}{k_m} \tag{5.4a}$$

$$V = iR + k_m \times \dot{\theta} \times i_{wg} \times n \tag{5.4b}$$

$$P_e = V \times i \tag{5.4c}$$

Where i, V, and P_e are the motor current, voltage, and electrical power required at each time step, n is the gear ratio, and R, k_m, and T_f are the motor resistance, torque constant, and internal friction torque, respectively.

Table 5.1: Mass, length, and powertrain parameters for the worm gear and gear head were chosen by reverse engineering a single link robot arm. Motor parameters were chosen by examining datasheets for DC motors and selecting one that was big enough to handle the peak power, torque and speed requirements [5]. The same parameters are used for each link and joint in the robot arm.

Parameter	Definition	Value
m_l	Link Mass	3.1 kg
l	Link Length	62 cm
i_{wg}	Worm Gear Speed Ratio	18.7
C_1	Load Driven Torque Ratio	$-1/128$
C_2	Motor Driven Torque Ratio	1/8
n	Gear Head Speed Ratio	30
R	Resistance	0.41 Ω
k_m	Torque Constant	52 mN-m/A
T_f	Motor Friction	4.9 mN-m

6. **Optimize Passive Assist Device** The final step of this methodology is optimization of the PAD design. In this example we wish to find a linear torsion spring that minimizes motor energy consumption. Thus, we formulate the following gradient based optimization problem:

$$\begin{aligned} \text{minimize} \quad & f(k, T_0) = \int_0^{t_f} |P_e(t, k, T_0)| \, \mathrm{d}t \\ \text{with respect to} \quad & k, T_0 \\ \text{subject to} \quad & -k \leq 0 \\ & \max(|i(t, k, T_0)|) \leq i_{max} \\ & \max(|V(t, k, T_0)|) \leq V_{max} \\ & \max(|P_e(t, k, T_0)|) \leq P_{e_{max}} \\ \text{and} \quad & 0 \leq t \leq t_f \end{aligned} \tag{5.5}$$

We solve this problem with MATLAB's built in function *fmincon* initialized with a Force Displacement Curve Fit (FDCF) design [43]. A weighted Force Displacement Curve Fit (FDCF+) design could be used to initialize the optimizations presented in this chapter and likely would decrease the computational time (see Sec. 3.2.2). However, in the example presented in this chapter the unweighted FDCF designs were able to serve as acceptable initial designs for optimization. The motor performance limit constraints are typically inactive, but including these constraints in the problem ensures that we are only testing feasible maneuvers.

5.2 PAD Design for a Nominal Maneuver

UGVs operate in uncertain environments and their manipulator arms are expected to be able to execute many different maneuvers. However, some types of maneuvers are much more common than others. For example, the arm is always stored in the same configuration (see (A) in Fig. 5.3) and the arm is typically raised slightly (B) while driving as to provide a better camera angle for the operator. The arm may also be used to lift the camera as high as possible (D) to achieve a superior point of view while stationary, to check under vehicles (C), and to lift an object from the ground (E_1) onto a higher surface (E_2). Of course the arm could be used to perform other tasks, but it usually will be performing a task similar to the ones listed above. We define a nominal maneuver that progresses through the following configurations: A-B-C-B-D-B-E_1-E_2-B-A. When at each stage the arm is at rest and when moving between stages the joint angles follow a 5th order polynomial trajectory. The time to execute each submaneuver and the lifted mass are listed in Tab. 5.2. For this nominal maneuver only the mean values of each parameter are used.

Table 5.2: The maneuver is described by 10 parameters: 9 parameters describe the time to move from one stage to another (e.g., t_{AB} is the time to move from storage (A) to driving (B)) while the last parameter, m, is the mass lifted from stage (E_1) to (E_2).

Parameter	Mean	Std. Dev.	Parameter	Mean	Std. Dev.
t_{AB}	5 s	0.5 s	$t_{\mathrm{BE_1}}$	15 s	3 s
t_{BC}	20 s	3 s	$t_{\mathrm{E_1E_2}}$	30 s	5 s
t_{CB}	30 s	4 s	$t_{\mathrm{E_2B}}$	25 s	4 s
t_{BD}	20 s	3 s	t_{BA}	10 s	1 s
t_{DB}	30 s	4 s	m	4 kg	1 kg

For this nominal maneuver, energy minimizing PADs can aid Joints 1, 2, and 3 by 78.9%, 65.4% and 39.8%, respectively. These values are significantly higher than

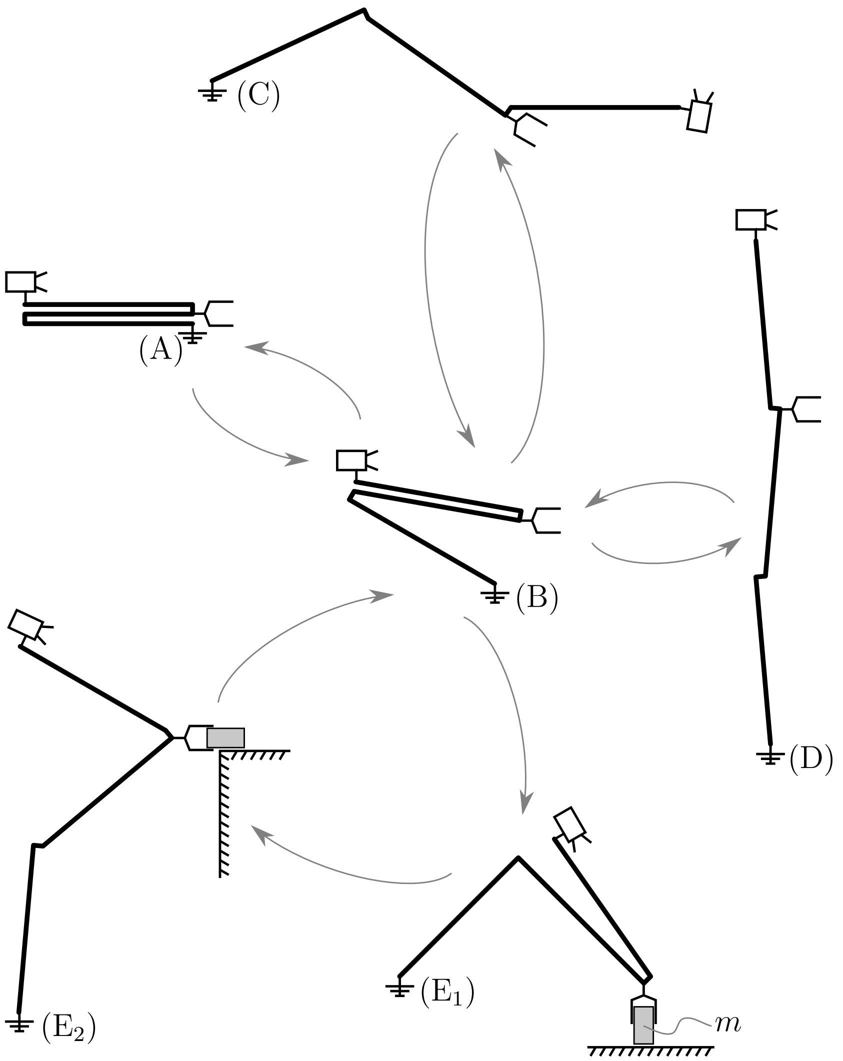


Figure 5.3: Typical operating states of a UGV arm including storage (A), driving (B), looking under an object (C), looking from a raised vantage point (D), and lifting an object from (E$_1$) to (E$_2$).

values found in previous work [44]. This is mostly because the torque angle data for the robot arm joints in this paper arm can be more closely approximated by a linear torsion spring than the corresponding data in previous work where a single joint was unloaded during descent and loaded during ascent. Different powertrain values also account for some of the difference. Optimizing the PAD and powertrain simultaneously will be the subject of future work.

Fig. 5.4 illustrates the effect that adding an energy minimizing PAD has on the motor of Joint 1. The spring is able to supply much of the required torque, reducing the effort required from the motor. Consequently, the required power and energy consumed also decrease. The other joints can be analyzed the same way.

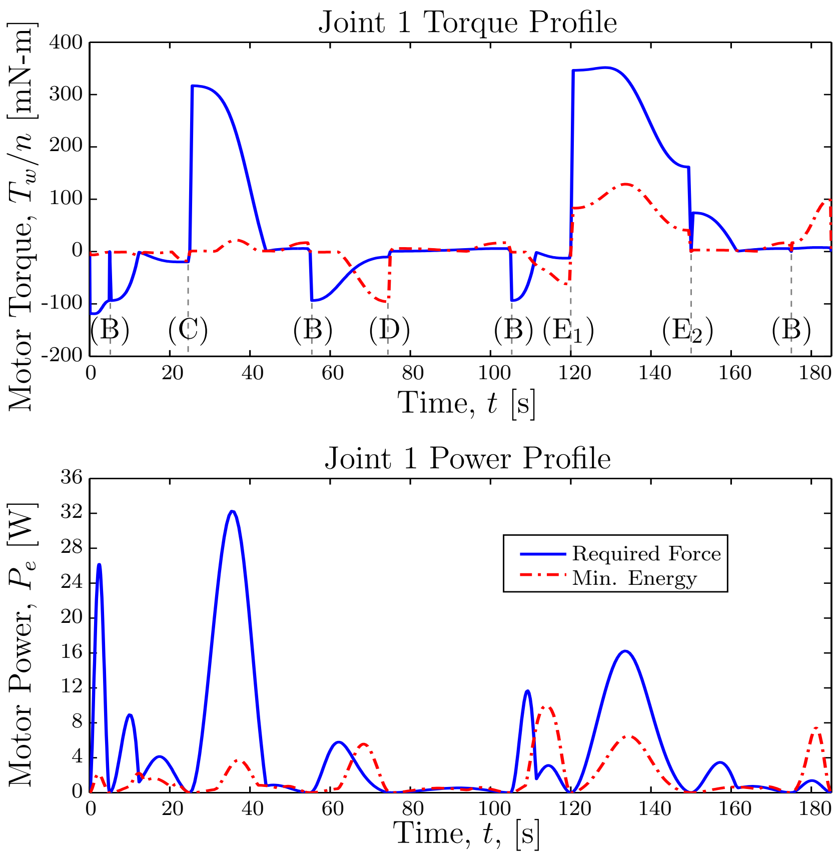


Figure 5.4: Required motor torque (top) and power (bottom) to execute the nominal maneuver with no spring (solid blue) and with an energy minimizing spring (dash-dot red). The maneuver starts and ends in a stored configuration (A) (see Fig. 5.3) and progresses between the other configurations as shown on the motor torque plot.

5.3 Robust PAD Design

A robust design is one where the expected outcome is not sensitive to variance. The design could be made robust against internalities (e.g., machine tolerance) or against externalities (e.g., changes in maneuver). Both of these are important, but which is more important depends on the application. Designing to be robust against internalities is important if a variation in the optimally designed component could violate a design constraint. For example, when designing a pipe it may be desirable to make the walls as thin as possible to reduce weight while satisfying a maximum allowable stress constraint. If the pipe is then manufactured slightly thinner than optimal (perhaps due to machine tolerance), it would be too weak, resulting in a poor product. This could be avoided by accounting for such possible variations in the optimization process. This is making the design robust against internal variance and is at the core of reliability based design optimization (RBDO) [73, 74].

In the case of a UGV manipulator arm being augmented with a linear spring, variations in the spring and robot design itself are likely insignificant compared to variations in the maneuver (i.e., trajectory and loading). Therefore, we expect any changes in performance to be primarily caused by changes to the maneuver and will only focus on these effects for the remainder of this chapter.

5.3.1 Effect of Increased Maneuver Variability on Nominal Energy Minimizing Design

In order for a PAD to be of real use it needs to provide significant assistance for all likely maneuvers, not just one. In general as maneuver variability goes up, the number of maneuvers where the nominal energy minimizing PAD provide assistance will decrease. To demonstrate this point we consider two maneuver distributions: the first maneuver distribution — Variable Parameters — is the nominal maneuver altered such that each parameter is described by a normal distribution with mean and standard deviation as listed in Tab. 5.2. The second maneuver distribution — Random Submaneuvers — considers the same distribution of parameters and also considers a different sequence of stages than in the nominal case. Instead of executing one raised arm submaneuver (B-C-B), one lowered arm submaneuver (B-D-B), and one lifting submaneuver (B-E_1-E_2-B); the arm executes three submaneuvers and each one has a 1/3 chance to be a raised arm, lowered arm, or lifting submaneuver (e.g., 1/27 of maneuvers in this distribution involve lifting a load three times). Each distribution is implemented via a Monte Carlo approach, where performance metrics are calculated based on 1,000 individual maneuvers.

To compare PAD performance across different maneuvers or maneuver distributions we consider several performance metrics. First, the energy required to execute a single maneuver:

$$E = \int_0^{t_f} |P_e(t, k, T_0)| \, \mathrm{d}t \tag{5.6a}$$

Similarly, $\mathbf{E}$ is a vector containing the energy required to execute each simulated maneuver within a distribution such that

$$\mathbf{E}_i = \int_0^{t_f} |\mathbf{P}_{\mathbf{e}i}(t, k, T_0)| \, \mathrm{d}t \tag{5.6b}$$

The objective of our nominal optimization, (5.5), is to minimize E. $\bar{\mathbf{E}}$ and $\sigma(\mathbf{E})$ are the average and standard deviation of energy consumed over a distribution of maneuvers, respectively.

We also consider the percent energy saved by adding a PAD compared to a no-spring design for a single maneuver:

$$S = 1 - E(k, T_0)/E(0, 0) \tag{5.7a}$$

$\mathbf{S}$ is the analogous vector for a distribution of maneuvers:

$$\mathbf{S}_i = 1 - \mathbf{E}_i(k, T_0)/\mathbf{E}_i(0, 0) \tag{5.7b}$$

Finally, we measure the n^{th} percentile of energy saved:

$$C(\mathbf{S}) = \bar{\mathbf{S}} + z(n)\sigma(\mathbf{S}) \tag{5.8}$$

For example, 10^{th} percentile of energy saved corresponds to $z(0.10) = -1.28$. A high value of $C(\mathbf{S})$ corresponds to a PAD design that is effective at increasing system performance over a wide range of maneuvers.

Table 5.3: The performance of energy minimizing springs as variance in maneuver distribution increases. $\bar{\mathbf{S}}$ is the average energy saved compared to a springless design (5.7), $\sigma(\mathbf{E})$ is the standard deviation in energy consumed, and $C(\mathbf{S})$ is the minimum expected energy savings in 90% of maneuvers (5.8).

	Nominal	Variable Parameters			Random Submaneuvers		
Joint	S [%]	$\bar{\mathbf{S}}$ [%]	$\sigma(\mathbf{E})$ [J]	$C(\mathbf{S})$ [%]	$\bar{\mathbf{S}}$ [%]	$\sigma(\mathbf{E})$ [J]	$C(\mathbf{S})$ [%]
1	78.9	77.9	15.6	76.1	78.9	89.7	63.7
2	65.4	64.2	11.8	63.0	65.3	47.9	60.1
3	39.8	39.7	0.83	38.8	38.4	38.3	11.7

Table 5.3 shows that the average energy saved stays nearly constant as variance increases because the two distributed cases vary evenly about the nominal maneuver by construction. A smaller $\sigma(\mathbf{E})$ corresponds to a narrower energy distribution — allowing for the energy consumed when executing a maneuver to be predicted more accurately. As expected, $\sigma(\mathbf{E})$ increases as the distribution of maneuvers becomes more varied. An alternative measure of variability, $C(\mathbf{S})$, is the energy saved in the 10^{th} percentile (5.8). In other words, we can guarantee with 90% confidence that implementing an energy minimizing spring on Joint 1 executing the Random Submaneuvers distribution will use at least 63.7% less energy than a springless design.

5.3.2 Robust Optimization Objective Function

Table 5.3 shows that PAD performance is susceptible to variability in the maneuver. A robust PAD design would perform more consistently despite such variability. The traditional robust design objective looks to simultaneously minimize both an average objective and the variance in that objective [64, 75]. For our purposes we would seek to minimize both average energy and variance in energy:

$$f(k, T_0) = (1 - w) \times \bar{\mathbf{E}} + w \times \sigma^2(\mathbf{E}) \tag{5.9}$$

where w is the weighting of the two objectives. If $w = 0$, then the goal is only to minimize the average energy, while $w = 1$ corresponds to minimizing variance in energy only. Although minimizing variance makes system performance more predictable, it is often at the expense of the average, resulting in a system that is predictably mediocre (or even bad).

When executing maneuvers that differ from the nominal design case, a PAD can save more energy on some maneuvers and less on others. If a maneuver being performed is much less demanding than the maneuver for which the PAD was designed, implementation of the PAD can actually decrease the performance of the system. Saving more than the average energy is fine, but we want to avoid situations where the PAD performs poorly or is a detriment to the system. This is similar to finding the n^{th} percentile robust shortest path when estimating travel times [76]. Thus, we formulate the following robust objective function that balances minimizing mean energy consumption and maximizing the energy we are guaranteed to save with 90% confidence (a higher confidence level would result in a more robust system):

$$f(k, T_0) = (1 - w) \times \bar{\mathbf{E}} - w \times C(\mathbf{S}) \tag{5.10}$$

Table 5.4 shows how increasing the weight, w, of the robust objective function shifts the emphasis from minimizing average energy consumption to maximizing the guaranteed energy savings at a 90% confidence level. The set of optimal designs form a Pareto curve. This trade-off is illustrated for Joint 3 in Fig. 5.5. For Joint 3, a PAD can be designed to be significantly more robust against variations in the maneuver. In Joints 1 and 2, the energy minimizing design is, coincidentally, almost exactly the same as the most robust design.

Fig. 5.6 illustrates why we chose an alternative robust design objective function (5.10) instead of the standard one (5.9). If we had used (5.9) for our robust design

Table 5.4: The performance of five PAD designs: no spring, an energy minimizing design based on the nominal maneuver (5.5), and three robust designs of various weights (5.10) based on the Random Submaneuvers distribution. E (5.6) and S (5.7) are the energy consumed and percent saved, respectively. $C(\mathbf{S})$ is the minimum energy savings in 90% of maneuvers (5.8).

	PAD Design			Nominal		Random Submaneuvers		
Joint	Description	k [N-m/rad]	T_0 [N-m]	E [J]	S [%]	$\bar{\mathbf{E}}$ [J]	$\bar{\mathbf{S}}$ [%]	$C(\mathbf{S})$ [%]
1	No Spring	0	0	1041	—	1037	—	—
	Min. Energy	49.6	−104	219.8	78.9	218.7	78.9	63.7
	Robust, $w = 1/3$	49.3	−103	219.8	78.9	218.7	78.9	63.7
	Robust, $w = 2/3$	48.4	−101	221.0	78.8	219.8	78.8	63.7
	Robust, $w = 1$	43.7	−89	248.9	76.1	248.6	76.0	63.8
2	No Spring	0	0	593.4	—	589.6	—	—
	Min. Energy	1.1	−16.5	205.4	65.4	204.8	65.3	60.1
	Robust, $w = 1/3$	1.3	−15.9	205.5	65.4	204.9	65.2	60.8
	Robust, $w = 2/3$	1.6	−15.3	205.7	65.3	205.1	65.2	61.2
	Robust, $w = 1$	1.6	−15.2	205.7	65.3	206.0	65.1	61.2
3	No Spring	0	0	163.5	—	164.7	—	—
	Min. Energy	2.8	0.5	98.5	39.8	101.5	38.4	11.7
	Robust, $w = 1/3$	3.9	−3.0	101.8	37.7	104.0	36.8	22.2
	Robust, $w = 2/3$	4.6	−4.8	103.9	36.5	105.8	35.7	24.0
	Robust, $w = 1$	4.8	−5.5	104.9	35.9	106.6	35.3	25.1

we would have generated a pareto curve that begins at the minimum point of the left plot (average energy) and goes to the minimum point on the right plot (standard deviation of energy). Minimizing avarage energy is a useful objective. However, minimizing standard deviation corresponds to an average energy savings of a mere 3.2% for Joint 3. Forcing the design in this direction effectively pulls the energy savings down to low uniform level. By contrast, our robust design attempts to pull the performance of the poor performing maneuvers up to the average level.

5.4 Concluding Remarks

In this chapter we showed that our design approach can effectively be applied to more complicated machines performing known periodic motions by simulating a 3-link UGV manipulator arm executing a representative sample of typical missions. The torques and energy required of the motor at each joint can be decoupled from the rest of the manipulator arm prior to optimization. This simplifies the problem from an n-DOF problem to n 1-DOF problems. We then found that by adding energy

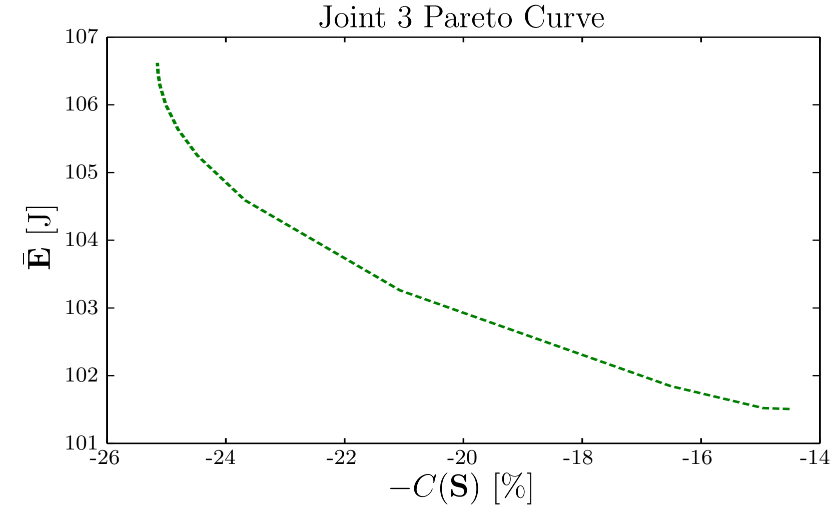


Figure 5.5: A Pareto curve depicts a set of optimal robust designs and the trade-off between minimizing average energy consumption (y-axis) and maximizing the guaranteed energy savings at a 90% confidence level (x-axis). As the weight, w, increases from 0 to 1 the curve moves up and to the left.

minimizing PADs (torsion springs) we achieved joint energy savings of 39–79% and a total system energy savings of 70.9%. This energy savings is substantial; a UGV that may have only had enough battery energy to perform one arm maneuver could now perform three, greatly increasing utility. Most importantly, we showed that PADs can be designed to be robust against variations in the maneuver — substantially increasing the guaranteed energy savings at a 90% confidence level.

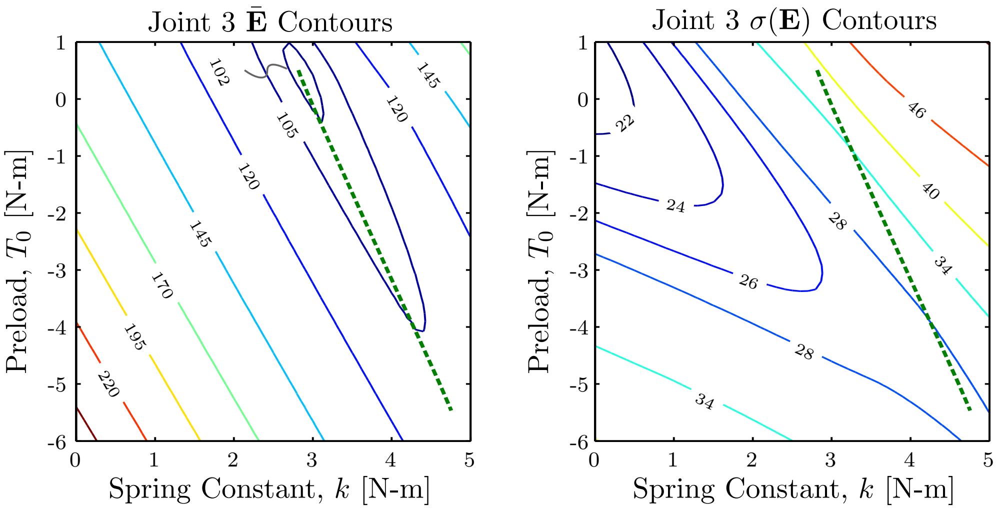


Figure 5.6: Required average energy (left) and standard deviation in energy (right) contours as functions of the two spring variables: k, T_0. The set of optimal robust designs form a Pareto curve (dashed green) with weights ranging from 0 (minimize $\bar{\mathbf{E}}$) to 1 (maximize $C(\mathbf{S})$).

CHAPTER 6

Summary, Conclusions, and Future Work

In Sec. 1.4 we proposed a general methodology to increase the performance of active systems by augmenting them with optimized Passive Assist Devices (PADs). Given the ever-growing number of automated machines (robots, machining systems, material handling, etc.) the approach proposed here can have a major impact by increasing energy efficiency. Some machines execute maneuvers that vary so much that the addition of a PAD designed by this approach will have negligible impact on performance. However, while many automated systems perform a variety of tasks, such tasks are often similar; that is, the range of trajectories and loads is limited. Therefore, PADs optimized for a particular maneuver can be effectively deployed on these systems so long as the particular maneuver is typical of the trajectories and loads of the augmented system [8]. Furthermore, we have presented a robust PAD design approach for systems where the maneuver can be described by a known distribution [45].

In Chap. 2 we experimentally demonstrated this concept on a single link robot arm augmented with a torsional spring. We observed an increase in performance by decreasing the peak motor torque required by $\sim 50\%$. This decrease in required peak torque can improve reliability by increasing the life of the joint motors, improve the utility by allowing heavier loads to be lifted, and significantly improve efficiency. Finally, we analyzed the modes of energy savings, examined the importance of the gear ratio on energy savings, considered a distribution of maneuvers, and compared our results to other applicable state-of-the-art designs.

In Chap. 3 we extended the maneuver based, passive assist design approach to series and dual systems. We illustrated how the design processes for the series and parallel systems were analogous using simple mass-spring-actuator systems. To make the optimization converge more quickly we introduced a new initial design using a weighted force displacement curve fit FDCF+ for a parallel system or a weighted derivative of force velocity curve fit dFVCF+ for a series system. We provide engineering insight into why different types of PADs perform differently depending on the maneuver and offer guidelines on how to select a specific type based on the application. Specifically, parallel designs in general have greater potential when external forces dominate while series springs are often superior for maneuvers with large inertial forces. We also discuss how the addition of different PAD types can affect a system from a non-energy standpoint. Finally, we demonstrate this design process and selection procedure on a 3-link manipulator arm and found that a combination of parallel and dual PADs could reduce energy consumption by up to 78%.

In Chap. 4 we provide two examples of using our approach to design PADs for manufacturing machinery. In the first example we illustrate the process on a a reciprocating single axis table. In a second example we show how the general method can be applied to more complicated systems and maneuvers such as a two-DOF X-Y table performing a milling operation. The addition of energy minimizing springs yielded a decrease in energy consumption of 65–79%. Finally, we showed that a significant increase in performance can be realized if the maneuver is redesigned considering that a PAD will added to the system.

In Chap. 5 we we showed that our design approach can effectively be applied to more complicated machines performing known periodic motions by simulating a 3-link UGV manipulator arm executing a representative sample of typical missions. The torques and energy required of the motor at each joint can be decoupled from the rest of the manipulator arm prior to optimization. This simplifies the problem from an n-DOF problem to n 1-DOF problems. We then found that by adding energy minimizing PADs (torsion springs) we achieved joint energy savings of 39–79% and a total system energy savings of 70.9%. This energy savings is substantial; a UGV that may have only had enough battery capacity to perform one arm maneuver could now perform three, greatly increasing utility. Most importantly, we showed that PADs can be designed to be robust against variations in the maneuver — substantially increasing the guaranteed energy savings at a 90% confidence level.

There are four obvious directions to explore for future work:

1. Throughout this thesis we have only considered ideal linear springs for our PADs. We could consider more complex PADs such as nonlinear springs. If our approach is applied to larger systems than the one we used for experimental verification in Chap. 2 it could be important to include spring mass and spring losses as well. Finally, this approach could be applied to non-mechanical passive devices including electrical components like capacitors.

2. We observed in Sec. 2.3.2 that optimizing the spring and powertrain simul-

taneously can provide further performance enhancements. This increases the scope of the problem from the improvement of an existing arm (by the addition of springs) to full arm design. We can continue in this direction by adding a control system and eventually incorporating co-design optimization of the arm and its controller [77].

3. We demonstrated in Sec. 4.3.3 that it may be possible to design a more effective PAD if the maneuver is redesigned. In many situations where a machine will perform the same task many times, performance can be further enhanced by simultaneously optimizing the maneuver and PAD.

   Maneuver redesign may not be practical in many situations. For example, if a machine is used in a flexible manufacturing setting and frequently performs different operations, redesigning the maneuver and spring each time will not be cost effective. On the other hand, if a machine is going to perform the same task thousands of times — as in a dedicated machining line — redesigning the maneuver to maximize the performance of an energy minimizing PAD could be quite beneficial.

4. We demonstrated in Sec. 5.3 that it is possible to design a PAD that is robust against variations in the maneuver. For applications like a UGV manipulator arm we expect variations in the PAD and robot itself are insignificant compared to the variations in the maneuver (i.e., trajectory and load). However, if the same maneuver is repeated with high precision (e.g., manufacturing machinery) then the effects of variation due to the maneuver and due to the internal design may be on the same order. For such situations we could make the system robust against internalities as well. This concept could be extended even further to include the concept of reliability based design optimization.

APPENDIX A

Experimental Methods

The worm gear speed ratio, i_{wg}, and steady state torque ratios, C_1, C_2, are sufficient for most simulation calculations. However, there are several reasons to experimentally find the five fundamental worm gear parameters: worm radius, r_w, worm gear radius, r_g, worm pitch, λ, pressure angle, α, and coefficient of kinetic friction, μ. First, if the worm gear is backdrivable, neither C_1 nor C_2 are valid torque ratios during no speed operation. Second, experimentally finding the fundamental parameters provides an experimental justification for using the higher level ratios derived from the model (i.e., C_1, C_2). Finally, much of the error between the simulation and experimental results can be explained by observed variation in the coefficient of kinetic friction on the worm gear interface, μ. This appendix will describe the procedure used to experimentally find the five worm gear parameters.

It is difficult to accurately measure the exact point where the teeth of the gears mesh, therefore we will calculate r_g, r_w, and λ by measuring the distance between the axes of rotation, d, the length of the worm, l_w, the number of loops in the worm helix, m, and the speed ratio, i_{wg}, of the worm gear transmission. We have the following relation between speed ratio, gear radii, and pitch [2]:

$$i_{wg} = \frac{\dot{\theta}_w}{\dot{\theta}_g} = \frac{r_g}{r_w} \cot \lambda \tag{A.1}$$

Our worm gear transmission is constructed of LEGOs, which are manufactured with high precision and have a standard length unit of 8 mm [56]. When properly assembled the axles running through the worm and the gear are two standard LEGO units apart, or $d = 16$ mm. This distance is the sum of the gear radius and worm radius.

$$d = r_g + r_w \tag{A.2}$$

The length of the worm and the number of turns of the helix can be directly measured yielding, $l_w = 16$ mm and $m = 5$, respectively. We can imagine the helix being unwound forming a right triangle where the height is l_w, the base is proportional to $r_w \times m$, and the hypotenuse is the length of the helix. The pitch of the helix can be found from the following relation:

$$\cot\lambda = \frac{2\pi \times r_w \times m}{l_w} \tag{A.3}$$

By simultaneously solving Eqs. (A.1), (A.2), and (A.3) we accurately determine three key parameters of the worm drive transmission. Once calculated, the radii and pitch angle can be double checked with a ruler and protractor respectively.

The fourth worm transmission parameter of interest is the pressure angle, α. If theoretical gear efficiency were the only concern in worm construction, the tooth profile would have vertical edges corresponding to a pressure angle of zero. However, such a design would be difficult to machine and the mechanical stresses within the worm would be enormous. Thus, the tooth profile on real worm gears typically has a triangular or trapezoidal shape, where the slope of the slanted face is the pressure angle. This angle can be directly measured with a protractor with an accuracy of about 1°. For our worm we measured the pressure angle to be 14°.

Determining the last worm gear parameter — the coefficient of kinetic friction, μ — requires experimentation. μ can be calculated from the torque transmission ratio and the other four worm gear transmission parameters [2]:

$$\mu = \begin{cases} -\cos\alpha \dfrac{C_1 r_g \cos\lambda - r_w \sin\lambda}{C_1 r_g \sin\lambda + r_w \cos\lambda} & \text{if load driven} \\ +\cos\alpha \dfrac{C_2 r_g \cos\lambda - r_w \sin\lambda}{C_2 r_g \sin\lambda + r_w \cos\lambda} & \text{if motor driven} \end{cases} \tag{A.4}$$

The worm gear has four operating states (not including the non-backdrivable state): motor driven left engagement, load driven left engagement, motor driven right engagement, and load driven right engagement. The two motor driven cases are governed by the C_2 torque transmission ratio while the two load driven cases are governed by the C_1 torque transmission ratio. C_1 or C_2 can be calculated by dividing the worm shaft torque by the gear shaft torque: $C_1, C_2 = T_w/T_g$.

We apply a known constant torque, $T_g = \pm m_l \times g \times r_s$, to the load side by connecting a spool to the worm gear transmission output and lifting (or lowering) a mass at constant speed. The torque is positive for left engagement and negative for right engagement, m_l is the mass of the load, and r_s is the radius of the spool.

For each of the four moving operational states a variety of voltages are applied to the experiment resulting in the experiment being run at a variety of different speeds. The mass of the load is also varied. The torque on the worm side, T_w, is calculated by measuring current and speed and solving Eq. (2.8a). ($T_w = T_m$ for this experiment because the intermediate gear ratio is set to $n = 1$.) A value of μ is calculated for each trial run. If the model is accurate the value of kinetic friction will be the same regardless of operating conditions.

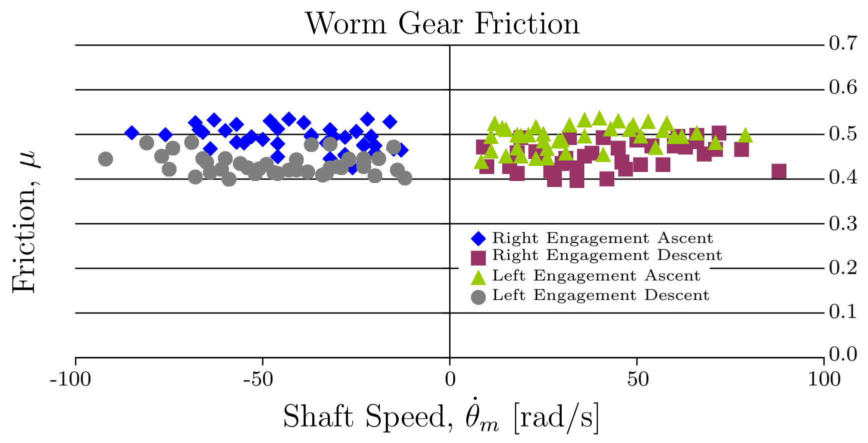


Figure A.1: Plots the experimentally found values of the coefficient of kinetic friction within the worm gear transmission. The data are organized by operating state. There is no statistical difference to the mean value of μ based on speed, direction of motion, or side of engagement.

By plotting the friction coefficient against velocity (see Fig. A.1) it is apparent that the friction coefficient is not dependent on velocity. The average coefficient of kinetic friction over all trial runs is 0.467 with a standard deviation of 0.037. There is no statistical evidence that the mean value of μ of any specific operating condition is different than the average value of μ over all operating conditions. This result verifies Yeh's theoretical model [2].

There is variation within each operating state, which can be attributed to the fact that friction – a very complex phenomenon — is being modeled by a simple equation. This variation accounts for much of the observed differences between experimental and simulation results (see Sec. 2.2.1).

APPENDIX B

Passivity of the Worm Gear Model

We begin with the equations of motion provided by [2] and consider the system shown in Fig. B.1.

Definitions of variables and ratios:

λ: The angle of the pitch helix of the worm. $0 \leq \lambda < 45°$

α: The pressure angle of the gear. $0 \leq \alpha < 90°$

μ: The coefficient of static and kinetic friction in the worm gear interface (they are assumed equal). $\mu \geq 0$

$\theta_w,\ \theta_g$: The angles about the axis of rotation of the worm and the gear, respectively.

$T_w,\ T_g$: The torques applied to the worm and the gear, respectively.

$J_w,\ J_g$: The moments of inertia of the worm and the gear, respectively.

$r_w,\ r_g$: The radii of the worm and gear, respectively.

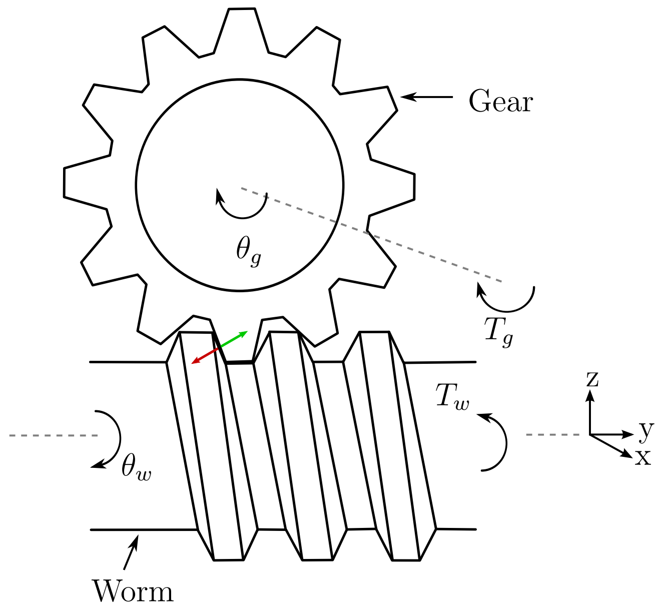
i_{wg}: The speed ratio of the worm gear.

$$i_{wg} = \frac{\dot{\theta}_w}{\dot{\theta}_g} = \frac{r_g}{r_w} \cot \lambda \tag{B.1}$$

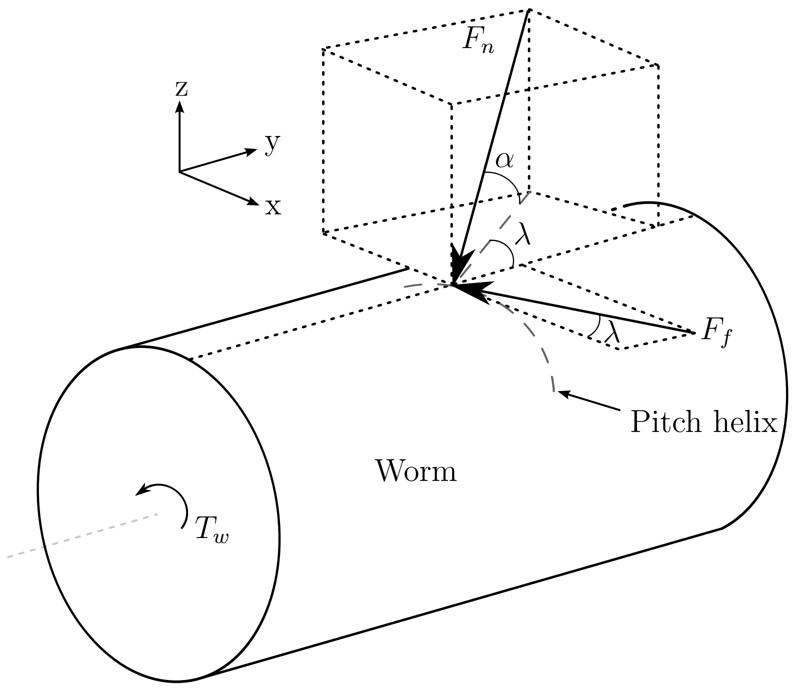
C_1: The ratio of torques during forward (backward) steady state operation during left engagement (right engagement).

$$C_1 = \frac{r_w(\cos \alpha \sin \lambda - \mu \cos \lambda)}{r_g(\cos \alpha \cos \lambda + \mu \sin \lambda)} \tag{B.2}$$

$$C_1 > -\frac{J_w i_{wg}}{J_g} \tag{B.3}$$



(a) Schematics of left engagement



(b) Force diagram of left engagement

Figure B.1: Schematics of the worm gear drive (after [2]).

C_2: The ratio of torques during forward (backward) steady state operation during right engagement (left engagement). The model presented by [2] is only valid if $C_2 > 0$. A situation where $C_2 \leq 0$ corresponds to a worm gear that is worm-locking; a configuration where the motor cannot drive the load regardless of the input torque.

$$C_2 = \frac{r_w(\cos\alpha\sin\lambda + \mu\cos\lambda)}{r_g(\cos\alpha\cos\lambda - \mu\sin\lambda)} \tag{B.4}$$

$$C_2 > 0 \tag{B.5}$$

Because these proofs will be instrumental to show power dissipation later, we will show that $i_{wg}C_1 \leq 1$ and $i_{wg}C_2 \geq 1$. We begin with the product of i_{wg} and C_1:

$$i_{wg}C_1 = \frac{\cos\alpha\cos\lambda\sin\lambda - \mu\cos^2\lambda}{\cos\alpha\cos\lambda\sin\lambda + \mu\sin^2\lambda} \tag{B.6}$$

We substitute the trig identity $\sin^2\lambda = 1 - \cos^2\lambda$ into the denominator of (B.6):

$$i_{wg}C_1 = \frac{\cos\alpha\cos\lambda\sin\lambda - \mu\cos^2\lambda}{\cos\alpha\cos\lambda\sin\lambda - \mu\cos^2\lambda + \mu} = \frac{m}{m+\mu} \tag{B.7}$$

We know $\mu \geq 0$ by definition and that $m+\mu \geq 0$ because it is equal to the denominator of (B.6) and every term the denominator of (B.6) is positive for $0 < \lambda < 45°$ and $0 \leq \alpha < 90°$. Thus

$$\mu \geq 0 \tag{B.8a}$$

$$m + \mu \geq 0 \tag{B.8b}$$

$$1 \geq \frac{m}{m+\mu} \tag{B.8c}$$

$$i_{wg}C_1 \leq 1 \quad \blacksquare \tag{B.8d}$$

We will now show that $i_{wg}C_2 \geq 1$. We begin with the product of i_{wg} and C_2:

$$i_{wg}C_2 = \frac{\cos\alpha\cos\lambda\sin\lambda + \mu\cos^2\lambda}{\cos\alpha\cos\lambda\sin\lambda - \mu\sin^2\lambda} \tag{B.9}$$

We substitute the trig identity $\cos^2\lambda = 1 - \sin^2\lambda$ into the numerator of (B.9):

$$i_{wg}C_2 = \frac{\cos\alpha\cos\lambda\sin\lambda - \mu\sin^2\lambda + \mu}{\cos\alpha\cos\lambda\sin\lambda - \mu\sin^2\lambda} = \frac{n+\mu}{n} \tag{B.10}$$

We know $\mu \geq 0$ by definition and that $n + \mu > 0$ because it is equal to the numerator of (B.9) and every term in the numerator of (B.9) is positive for $0 < \lambda < 45°$ and $0 \leq \alpha < 90°$. Finally, we know that $n > 0$ because $n + \mu > 0$ and $C_2 > 0$ and $i_{wg} > 0$.

Thus

$$\mu \geq 0 \tag{B.11a}$$

$$n + \mu \geq n \tag{B.11b}$$

$$1 \leq \frac{n+\mu}{n} \tag{B.11c}$$

$$i_{wg} C_2 \geq 1 \quad \blacksquare \tag{B.11d}$$

Considering that the law of conservation of energy holds even over infinitesimal periods of time, we know that the net flow of power into the worm gear is equal to the time rate of change of the kinetic energy of the worm gear.

$$\dot{W}_{in} - \dot{W}_{out} - P_d = \dot{KE} \tag{B.12}$$

Where P_d is the power dissipated by the worm gear and

$$\dot{W}_{in} = T_w \dot{\theta}_w \tag{B.13a}$$

$$\dot{W}_{out} = T_g \dot{\theta}_g \tag{B.13b}$$

$$KE = \tfrac{1}{2}(J_w \dot{\theta}_w^2 + J_g \dot{\theta}_g^2) \tag{B.13c}$$

$$KE = \tfrac{1}{2}(J_w i_{wg}^2 + J_g)\dot{\theta}_g^2 \tag{B.13d}$$

$$\dot{KE} = (J_w i_{wg}^2 + J_g)\ddot{\theta}_g \dot{\theta}_g \tag{B.13e}$$

Yeh and Wu derived four different explicit dynamic equations for the worm gear system, which can be described as one equation subject to one of four different conditions:

$$(J_w i_{wg} + C J_g)\ddot{\theta}_g = T_w - C T_g \tag{B.14}$$

Where

$$C = \begin{cases} C_2 & \text{if } T_w > -\frac{J_w i_{wg}}{J_g} T_g \text{ and } \dot{\theta}_g > 0 & \text{(B.15a)} \\ C_1 & \text{if } T_w \leq -\frac{J_w i_{wg}}{J_g} T_g \text{ and } \dot{\theta}_g > 0 & \text{(B.15b)} \\ C_2 & \text{if } T_w \leq -\frac{J_w i_{wg}}{J_g} T_g \text{ and } \dot{\theta}_g < 0 & \text{(B.15c)} \\ C_1 & \text{if } T_w > -\frac{J_w i_{wg}}{J_g} T_g \text{ and } \dot{\theta}_g < 0 & \text{(B.15d)} \end{cases}$$

Starting with (B.14) we multiply both sides by $\dot{\theta}_w$ in order to transform a torque balance equation into a power balance equation:

$$(J_w i_{wg} + CJ_g)\ddot{\theta}_g \dot{\theta}_w = T_w \dot{\theta}_w - CT_g \dot{\theta}_w \tag{B.16}$$

The term for $\dot{W}_{in}$ has already appeared. Equation (B.16) can be further rearranged so that the terms for $\dot{W}_{out}$ and $\dot{KE}$ also appear.

$$\dot{W}_{in} - \dot{W}_{out} - \dot{KE} = (Ci_{wg} - 1)\dot{\theta}_g (T_g + J_g \ddot{\theta}_g) \tag{B.17}$$

Thus

$$P_d = (Ci_{wg} - 1)\dot{\theta}_g (T_g + J_g \ddot{\theta}_g) \tag{B.18}$$

We will now show that for each one of the four operating conditions, power dissipated by the worm gear is always greater than or equal to zero. First, we will consider P_d under the conditions of (B.15a). We already know that $C_2 i_{wg} - 1 \geq 0$ from (B.11d) and $\dot{\theta}_g > 0$ from (B.15a). This leaves the other condition from (B.15a):

$$T_w > -\frac{J_w i_{wg}}{J_g} T_g \tag{B.19}$$

Next, we use (B.14) to replace T_w in (B.19)

$$(J_w i_{wg} + C_2 J_g)\ddot{\theta}_g + C_2 T_g T_w > -\frac{J_w i_{wg}}{J_g} T_g \tag{B.20}$$

We note that $\frac{J_w i_{wg}}{J_g} > 0$ and $C_2 > 0$ from (B.5), which allows us to manipulate (B.20) into

$$\frac{J_w i_{wg} + C_2 J_g}{\dfrac{J_w i_{wg}}{J_g} + C_2} \ddot{\theta}_g > -T_g \tag{B.21}$$

$$T_g + J_g \ddot{\theta}_g > 0 \tag{B.22}$$

Thus, because every term that makes up P_{d_a} is greater than or equal to zero we know that

$$P_{d_a} \geq 0 \quad \blacksquare \tag{B.23}$$

Next, we will consider P_d under the conditions of (B.15b). We already know that $C_1 i_w g - 1 \leq 0$ from (B.8d) and $\dot{\theta}_g > 0$ from (B.15b). This leaves the other condition from (B.15b):

$$T_w \leq -\frac{J_w i_{wg}}{J_g} T_g \tag{B.24}$$

Following a procedure analogous to what was used for P_{d_a} and recalling that $C_1 + \frac{J_w i_{wg}}{J_g} > 0$ from (B.3) we can show that

$$T_g + J_g\ddot{\theta}_g \leq 0 \tag{B.25}$$

Two terms of P_{d_b} are less than or equal to zero and one term is greater than zero. Thus

$$P_{d_b} \geq 0 \quad \blacksquare \tag{B.26}$$

For P_{d_c} we can show $C_2 i_{wg} - 1 \geq 0$, $\dot{\theta}_g < 0$ and $T_g + J_g\ddot{\theta}_g \leq 0$; thus

$$P_{d_c} \geq 0 \quad \blacksquare \tag{B.27}$$

Finally, for P_{d_d} we can show $C_1 i_w g - 1 \leq 0$, $\dot{\theta}_g < 0$ and $T_g + J_g\ddot{\theta}_g > 0$; thus

$$P_{d_c} \geq 0 \quad \blacksquare \tag{B.28}$$

Because $P_d \geq 0$ for every operating condition, we know that the worm gear never generates energy, and is thus always passive. $\blacksquare$

APPENDIX C

MATLAB Code and Documentation

The following code is used to design PADs based on the maneuver based method described in Sec. 1.4 for the different case studies examined in each chapter of this thesis. The code is naturally divided into two files:

The first file, `steps1through3.m` (see p. 99), is responsible for defining the machine architecture, defining the trajectory and external loading, and performing inverse dynamics. Details of the position (and its derivatives) as well as the required forcing to achieve such motion for each joint and saves it to a file. Saving this information to a file before proceding to optimization is particularly useful for complex or distributed maneuvers which can take a long time to generate. This file makes use of a number of functions provided by Corke's robotics toolbox including *link()* and *robot()*, which are used to define the architecture using a general robotics framework; *jtraj()*, which produces a 5^{th} order polynomial in position given start and end points; and *rne()*, which performs inverse dynamics [72].

The second file, `steps4through6andresults.m` (see p. 119), loads the motion and required forcing variables, defines the PAD topology (always linear springs in this thesis), defines the powertrain, and finds energy minimizing PAD designs. Finally, the code provides a performance summary as well as several useful plots including FDCF, trajectory information, and power profiles. The code presented here can be modified to display more complicated data such as a distributed trajectory (see Fig. 2.9), parameter study (see Fig. 2.7), or pareto curve (see Fig. 5.5).

Maneuver Generation

```
function steps1through3
% This function performs step 1-3 of our six step design process:
% 1) Define system
% 2) Define maneuver
% 3) Perform inverse dynamics / decouple axes (or joints)
% This function then saves the motion and force data of each joint
%
% The user selects the system being analyzed.
%
% The saved system contains up to seven n-dimensional arrays,
% where n is the number of simulated maneuvers:
% t - time at each time step
% y, dy, ddy - position, velocity, and acceleration of the axis/joint
% (dddy) - may also explicitely include jerk information
% F - The total required force/torque acting on the axis/joint
% (dF) - may also explicitely include derivative of force information,

clc; clear all;
global sys_type
sys_type = 4; % <- can alter this value
% 2 -> The experimental LEGO single joint system.
% This is the sytem analyzed in Chapter 2.
% 3 -> A mass-spring-actuator system with ideal actuator.
% This is the sytem analyzed in Chapter 3.
% 4 -> Cartesian manufacturing machinery.
% This is the sytem analyzed in Chapter 4.
% 5 -> 3-link NBD manipulator arm.
% This is the sytem analyzed in Chapter 5.

itrs = 50; % <- can alter this value
        %number of maneuvers to simulate
        %itrs==0 -> mean maneuver

%save trajectory and force profiles
switch sys_type
    case 2  %creates the motion of the Experimental LEGO system
        create_robot_maneuver(1);
    case 3  %mass-spring actuator system
        w = 1; %sinusoid frequency
        a = 1; %sinusoid amplitude
        n = 1; %duration in constant velocity is t_cv = n*pi/w
        m = 1; %load mass
        C = 10; %constant load force
        create_simple_maneuver(w, a, n, m, C);
    case 4  %linear motion drive system
        create_manufacturing_maneuver();
    case 5  %load a single joint maneuver from a decoupled 3-link robot
            %arm maneuver of 3-link robot is generated and decoupled by
            %create_packbot_var_motions.m
        create_robot_maneuver(itrs);
end

% ----------------------------------------------------------------------

function create_simple_maneuver(w, a, n, m, C)
%creates maneuver that is piecewise combination of sinusoidal
%accelerations and constant velocity motions

parameters = [w; a; n; m; C]; %parameters used to define maneuver

%define trajectory
%trajectory is piecewise smooth in velocity
steps = 1000; %<-can change this value
                %number of time steps

y0 = a*w*n*pi/2; %offset ensures position is evenly distributed about
                 %horizontal axis
tf = (2*n+2)*pi/w; %final time

t = linspace(0,tf,steps); %time vector

%piecewise position
y1 = a*sin(w*t)+y0;
y2 = -a*w*(t-pi/w)+y0;
y3 = a*sin(w*t-n*pi) - a*n*pi + y0;
y4 = a*w*(t-(n+2)*pi/w) - a*n*pi + y0;

y = [y1(t<pi/w) y2(t≥pi/w & t<pi/w*(n+1)) ...
    y3(t≥pi/w*(n+1) & t<(n+2)*pi/w) y4(t≥(n+2)*pi/w)];

%piecewise velocity
dy1 = a*w*cos(w*t);
dy2 = -a*w*ones(size(t));
```

```
dy3 = a*w*cos(w*t-n*pi);
dy4 = a*w*ones(size(t));

dy = [dy1(t<pi/w) dy2(t≥pi/w & t<pi/w*(n+1)) ...
    dy3(t≥pi/w*(n+1) & t<(n+2)*pi/w) dy4(t≥(n+2)*pi/w)];

%piecewise acceleration
ddy1 = -a*w^2*sin(w*t);
ddy2 = zeros(size(t));
ddy3 = -a*w^2*sin(w*t-n*pi);
ddy4 = zeros(size(t));

ddy = [ddy1(t<pi/w) ddy2(t≥pi/w & t<pi/w*(n+1)) ...
    ddy3(t≥pi/w*(n+1) & t<(n+2)*pi/w) ddy4(t≥(n+2)*pi/w)];

%piecewise jerk
dddy1 = -a*w^3*cos(w*t);
dddy2 = zeros(size(t));
dddy3 = -a*w^3*cos(w*t-n*pi);
dddy4 = zeros(size(t));

dddy = [dddy1(t<pi/w) dddy2(t≥pi/w & t<pi/w*(n+1)) ...
    dddy3(t≥pi/w*(n+1) & t<(n+2)*pi/w) dddy4(t≥(n+2)*pi/w)];
dddy(end)=dddy(end-1);

%piecewise force
F = m*ddy + C;

%piecewise differential force
dF = m*dddy;

%create save file name
name = ['MSA-' datestr(date)];
save(name, 'y', 'dy', 'ddy', 'dddy', 'F', 'dF', 't', 'parameters')

% ----------------------------------------------------------------------

function create_robot_maneuver(itrs)
%Creates maneuver for experimental LEGO system
%and simulated packbot arm motions.
%Can generate a distribution of maneuvers if itrs>1;

global sys_type dist_type
```

```

% create robot model
bot = createrobot;

n = 50; %number of datapoints (aka timesteps) per submaneuver

%define some parameters describing size and behavior of distribution
rates = [1/3 1/3]; %rates(1)=prob of elevated camera
                   %rates(2)=prob of low camera
%rates = [1/2 1/2]; %use for middle variance
if(sum(rates)>1)
    error('sum(rates) must be ≤ 1')
end

%create trajectory and required joint torques
[QS, QDS, QDDS, TRQS, TS] = make_var_maneuvers(bot, rates, n, ...
    itrs);

% animation of robot motions
% for i=1:length(TS)
%     plot(bot, [0 QS(i,:)])
% end

%create save file name
if sys_type==2
    name = ['LEGO-' datestr(date)];
else %sys_type ==5
    name = ['PACK-' datestr(date)];
    if itrs == 0
        name=[name '-single'];
    else
        name=[name '-' dist_type '-' num2str(itrs)];
    end
end

save(name, 'QS', 'QDS', 'QDDS', 'TRQS', 'TS', 'bot')

% ----------------------------------------------------------------------

function r = createrobot
global sys_type
%this function calls functions provided by Corke's robotics toolbox
clear L
```

```
if sys_type == 4
    %xytable values
    %L{1} corresponds to Joint 0 and has no physical meaning
    %but rotates the other joints into convenient coordinates
    L{1} = link([pi/2 0 pi/2 0.01 1]);
    L{1}.I = [1,1, 1,1,1,1];
    L{1}.m = 1;
    L{1}.r = [0;0;0];
    L{1}.G=1; L{1}.Jm = 0.001;
    L{2}=L{1};
    L{2}.alpha = pi/2; L{2}.A=0; L{2}.theta= pi/2; L{2}.D = 0;
    L{2}.m = 54;
    L{3}=L{2};
    L{3}.alpha = pi/2; L{3}.A=0; L{3}.theta= pi/2; L{3}.D = 0;
    L{3}.m = 10;
else

    if sys_type == 5
        %packbot values
        l = 0.62; %link length
        m=3.1; %link mass
    elseif sys_type==2
        %exp lego values
        l = 0.127; %link length
        m = 0.0058/2; %link mass
        % The experimental LEGO arm can be simulated the same as a
        % 3-link packbot arm if the link masses and lengths are
        % appropriately changed and only the first elbow is actuated
        % while the other joints are held fixed.
    end

    %L{1} corresponds to Joint 0 and has no physical meaning
    %but rotates the other joints into convenient coordinates
    L{1} = link([pi/2 0 0 .01]);
    L{1}.m = 3.1;
    L{1}.r = [0;0;0];
    L{1}.I = [1/4*m*.08^2,1/2*m*.08^2, 1/4*m*.08^2,0,0,0];
    L{1}.G=1; L{1}.Jm = 0.001;

    %L{2} Corresponds to Joint 1 and connects the arm to a fixed base
    L{2} = link([0 l 0 0.08]);
    L{2}.m = m;
    L{2}.r = [-l/2;0;0];
    L{2}.I = [0,1/12*m*l^2, 1/12*m*l^2,0,0,0];
    L{2}.G=1; L{2}.Jm = 0.00001;

    %L{3} Corresponds to Joint 2, the first elbow
    L{3}=L{2};

    %L{3} Corresponds to Joint 3, the second elbow
    L{4}=L{2};
end

%creates the robot model
r = robot(L);

% ------------------------------------------------------------------------

function [QS, QDS, QDDS, TRQS, TS] = make_var_maneuvers(bot, rates, ...
    n, itrs)

m=bot.n; %number of robot joints
%initialize arrays
QS = zeros(n*11, m-1, itrs, 'single');
%positions at all timesteps for of all joints for all simulated
%maneuvers there are 11 submaneuvers, therefore, n*11 timesteps
%the first joint isn't importantant for analysis
%so only joints 2 through m are recorded

QDS = QS; %joint velocities
QDDS = QS; %joint accelerations
TRQS = QS; %joint torques
TS = zeros(n*11, itrs, 'single'); %times

tstart=tic; %timer (not important)
i=1; %initialize loop counter
while i≤max(itrs,1)
    %create path coordinates
    pathdata=createpathdata(itrs, rates);
    disp([num2str(i) '.a - ' num2str(toc(tstart))]) %progress update

    %create trajectory profiles for all joints for single maneuver
    [Q, QD, QDD, T] = makemaneuver(pathdata, n);
    disp([num2str(i) '.b - ' num2str(toc(tstart))]) %progress update

    %calculate joint torque/force profiles for all joints
    TRQ=maketorques(bot, Q, QD, QDD, [0 0 9.81], pathdata(:,end-1));
    disp([num2str(i) '.c - ' num2str(toc(tstart))]) %progress update

    %package trajectory, torque, and time data
    %C = checkmission(Q, QD, QDD, TRQ, TS)
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %When creating distributed maneuvers it is easy to generate a
    %maneuver that cannot be executed even with an optimized PAD.  If
    %this happens, no PAD can be found that satisfies motor constraints
    %and the entire set of trajectories is wasted.  This can be
    %prevented by only appending distributed trajectories that can be
    %executed to the list. checkmission() calls on NLCON() which in turn
    %calls on a number of other functions packaged with the file
    %'steps4through6andresults.m'
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %if C<0
    QS(:,:,i)=Q(:,2:end);
    QDS(:,:,i)=QD(:,2:end);
    QDDS(:,:,i)=QDD(:,2:end);
    TRQS(:,:,i)=TRQ(:,2:end);
    TS(:,i)=T;
    i=i+1; %increment counter
    %end
end
disp(toc(tstart)/itrs) %progress update

% ------------------------------------------------------------------------

function pathpts = createpathdata(itrs, rates)
global sys_type dist_type

dist_type='VP'; % <- can alter this value
% 'VP' -> Variable Parameters distribution
% 'RS' -> Random Submaneuvers distribution

% basic packbot poses
% 4 joints : rotating base, shoulder, elbow1, elbow2
% angles in degrees relative to previous link
stored = [ 0 180 -180 180];      %pose when stored
driving = [0 180-30 -160 180];   %pose when driving
uplook = [0 180-85 -10 10];      %arm extended up
downlook = [0 180-155 -60 35];   %arm extended forward
grab1 = [0 180-135 -90 170];     %grab mass position
```

```
grab2 = [0 180-95 -45 110];     %release mass position

%poses for LEGO experiment
SLup = [0 180 -90 180]; %arm lifted (vertical)
SLdown = [0 180 -180 180]; %arm lowered (horizontal)

%initialize pathpts
%each row corresponds to a pose
%columns refer to joints 1-4, load mass, and duration of submaneuver
pathpts = zeros(11,6);

if sys_type==2
    %create LEGO maneuver
    %exp lego values
    t0 = 0.002; %idling phase time
    td = 10;   %lowering time
    tu = 10;   %lifting time
    m = 0.045; %lifted mass

    pathpts = [SLup, 0, t0; ...
        SLup, 0, t0; SLup, 0, t0; SLup, 0, t0; ...
        SLup, 0, t0; SLup, 0, td; SLdown, 0, td; ...
        SLdown, m, t0; SLdown, m, t0; SLdown, m, t0; ...
        SLdown, m, tu;];

elseif itrs == 0
    %mean packbot maneuver
    t1=5; %time from stored to driving
    t2a = 20; %time from driving to uplook
    t2b = .00200; %time idling at uplook
    t2c = 30; %time from uplook to driving
    t3a = 20; %time from driving to downlook
    t3b = .00200; %time idling at downlook
    t3c = 30; %time from downlook to driving
    t4a = 15; %time from driving to grab1
    t4b = 30; %time from grab1 to grab2
    t4c = 25; %time from grab2 to driving
    mass = 4; %lifted mass
    t5 = 10; %time from driving to stored

    pathpts = [stored, 0, t1; ...
        driving, 0, t2a; downlook, 0, t2b; downlook, 0, t2c; ...
        driving, 0, t3a; uplook, 0, t3b; uplook, 0, t3c; ...
        driving, 0, t4a; grab1, mass, t4b; grab2, 0, t4c; ...
        driving, 0, t5];
else
%randomizable variables:
t1 = max([5 + 0.5*randn(1),2]); %time from stored to driving
t11 = max([10 + 0.75*randn(1),2]); %driving to storage time
%note: randomixed variables are defined in the above method to prevent
%impossibly fast / infeasible submaneuvers from occuring

pathpts(1,:) = [stored, 0, t1]; %stored to driving
switch dist_type
    case 'VP' %create Variable Parameters distribution
        %driving to uplook, hold uplook, to driving
        ta = max([20 + 3*randn(1),2]);
        tb = max([200 + 50*randn(1),2]);
        tc = max([30 + 4*randn(1),2]);
        pathpts(2:4,:) = [driving, 0, ta; uplook, 0, tb; uplook, 0, tc];

        %driving to downlook, hold downlook, to driving
        ta = max([20 + 3*randn(1),2]);
        tb = max([200 + 50*randn(1),2]);
        tc = max([30 + 4*randn(1),2]);
        pathpts(5:7,:) = [driving, 0, ta; downlook, 0, tb; ...
            downlook, 0, tc];

        %driving to grab load, to liftload, to driving
        ta = max([15 + 3*randn(1), 2]);
        tb = max([30 + 5*randn(1), 2]);
        tc = max([25 + 4*randn(1), 2]);
        mass = max([4 + 1*rand(1), 0.1]);
        pathpts(8:10,:) = [driving, 0, ta; grab1, mass, tb; ...
            grab2, 0, tc];

    case 'RS' %create Random Submaneuvers distribution
        for i=2:3:8
            k = rand(1); %random variable
            if k < rates(1) %uplook
                %driving to uplook, hold uplook, to driving
                ta = max([20 + 3*randn(1),2]);
                tb = max([200 + 50*randn(1),2]);
                tc = max([30 + 4*randn(1),2]);
                data = [driving, 0, ta; uplook, 0, tb; ...
                    uplook, 0, tc];
            elseif k < sum(rates) %downlook
                %driving to downlook, hold downlook, to driving
                ta = max([20 + 3*randn(1),2]);
                tb = max([200 + 50*randn(1),2]);
                tc = max([30 + 4*randn(1),2]);
                data = [driving, 0, ta; downlook, 0, tb; ...
                    downlook, 0, tc];
            else %lift
                %driving to grab load, to liftload, to driving
                ta = max([15 + 3*randn(1),2]);
                tb = max([30 + 5*randn(1),2]);
                tc = max([25 + 4*randn(1),2]);
                mass = max([(4 + 1*randn(1)),0.1]);
                data = [driving, 0, ta; grab1, mass, tb; ...
                    grab2, 0, tc];
            end
            pathpts(i:(i+2),:) = data;
        end
end

pathpts(11,:) = [driving, 0, t11];  %driving to stored
end

%convert from deg to rad
pathpts(:,1:end-2)=pathpts(:,1:end-2)*pi/180;

% -----------------------------------------------------------------------

function [Q, QD, QDD, Ts, F] = makemaneuver(pathdata, n, Favg)
%takes a higher level description of the maneuver (pathdata) and returns
%a more detailed description of the motion and forcing at each timestep

%pathdata is the information describing the maneuver

%n is the number of time steps per submaneuver
global sys_type


pts = pathdata(:,1:end-2);  %pts is a set of coordinates that the
                            %trajectory passes through
                            %trajectory ends at beginning point

DTorCS = pathdata(:,end);   %DTorCS is the time to complete each step or
```

```
                               %a constant speed at which the step operates

flag = pathdata(:,end-1);      %if sys_type == 2 or 5,
                               %flag is the lifted mass and not called used
                               %in makemaneuver()

                               %if sys_type ==4,
                               %flag indicates the motion to the next point
                               % 0 -> not milling, 5th order poly
                               % 0.1 -> not milling, sinusoidal
                               % 1 -> milling, direct to next point
                               % 2 -> milling, half circle to next point
                               % 3 -> milling, quarter circle to next point


[m, k] = size(pts); %m is the number of submaneuvers,
                    %k is the number of joints

%initialize
Q=zeros(m*n,k); QD=Q; QDD=Q; %position and derivatives
Ts = zeros(m*n,1); %times
tsbase=0; %temporary time value

for i=1:m %for each submaneuver
    pt1 = pts(i,:); %define begining point
    if i==m
        pt2 = pts(1,:); %define ending point,
                        %end point is first point of first submaneuver
    else
        pt2 = pts(i+1,:); %define ending point
    end

    if sys_type == 2 || sys_type==5 %robot arm system
        %create submaneuver timestep times
        ts = linspace(0, DTorCS(i), n)';
        %create positions at each timestep
        [q, qd, qdd] = jtraj(pt1,pt2,ts);
             %jtraj() from Corke's robotics toolbox
        f=zeros(size(q)); %external forcing for manipulator types not
                          %calculated here, see maketorques()

    elseif sys_type==4 %XY table milling system
        fi = flag(i); %flag for each submaneuver
        if fi == 0 %not milling, 5th order poly
            ts = linspace(0, DTorCS(i), n)'; %create timesteps
            %interpolate 5th order, straight path
            [q, qd, qdd] = jtraj(pt1,pt2,ts);
            f = zeros(n,3); %no external forcing

        elseif fi == 0.1 %not milling, sinusoidal
            ts = linspace(0, DTorCS(i), n)'; %create timesteps
            %interpolate harmonic, straight path
            [q, qd, qdd] = htraj(pt1,pt2,ts);
                %htraj() is a modified version of jtraj() that creates
                %sinusoidal motions instead of 5th order polynomials
            f = zeros(n,3); %no external forcing

        elseif fi==1 %milling, direct to next point
            Vf = DTorCS(i); %constant speed
            %interpolate constant speed, straight path
            [q, qd, qdd] = cstraj(pt1, pt2,n, Vf);
            ts = linspace(0, norm(pt1 - pt2)/Vf, n)'; %timestep times
            f = rotz(atan2(qd(1,3),qd(1,2)))*Favg;  %rotate forcing
            f = ones(n,1)*f';

        elseif fi ==2 %milling, half circle to next point
            Vf = DTorCS(i); %constant speed
            %interpolate constant speed, semi circular path
            [q, qd, qdd] = semicircle(pt1, pt2, n, Vf);
            ts = linspace(0, norm(pt1 - pt2)*pi/2/Vf, n)'; %times
            f=zeros(size(q)); %initialize forcing
            for j=1:n %rotate forcing
                f(j,:) = rotz(atan2(qd(j,3),qd(j,2)))*Favg;
            end
        elseif fi ==3 %milling, quarter circle to next point
            Vf = DTorCS(i); %constant speed
            %interpolate constant speed, quarter circular path
            [q, qd, qdd] = qtrcircle1(pt1, pt2, n, Vf);
            ts = linspace(0, norm(pt1(2) - pt2(2))*pi/2/Vf, n)'; %times
            f=zeros(size(q)); %initialize forcing
            for j=1:n %rotate forcing
                f(j,:) = rotz(atan2(qd(j,3),qd(j,2)))*Favg;
            end
        end
    end

    %packaging
    Ts((i−1)*n+1:i*n)=ts+tsbase; %append to total time vector
    Q((i−1)*n+1:i*n,:)=q;
    QD((i−1)*n+1:i*n,:)=qd;
    QDD((i−1)*n+1:i*n,:)=qdd;
    F((i−1)*n+1:i*n,:)=f;

    tsbase = ts(end)+tsbase; %reset tsbase

    %progress update
    disp(['End of Submaneuver ' num2str(i) ' at time ' ...
        num2str(tsbase)]);
end

% ----------------------------------------------------------------------

function torques = maketorques(bot, q, qd, qdd, gvect, loadmass)
%find changes in mass
%use lifted mass to change robot model for corresponding step


n = length(loadmass); %number of suybmaneuvers
m = size(q,1); %number of timesteps
l = m/n; %needed for proper placement of values in storage arrays
torques = zeros(size(q)); %initialize torque array

%build torque vector one step at a time
for i=1:n %for each submaneuver
    massbot=bot;
    lm = loadmass(i); %what is the lifted mass during this submaneuver

    %following code is based on the packbot arm, which has an end
    %effector at the 2nd elbow
    %basically we redefine the robot temporarily to account for the
    %lifted load mass
    if lm≠0 %need to change robot mass parameters
        linkdata = bot.link{3}; %link data of 3rd link of original robot
        newlink = linkdata; %create replacement link
        L = linkdata.A; %length of link 3

        %new link mass includes lifted load
        newlink.m = linkdata.m + lm;

        %new mass shift COG toward link end, all new mass located at end
        %of the link
        newlink.r = linkdata.r * linkdata.m/(linkdata.m+lm);

        %inertia of new link about new link COM
        dL = norm(linkdata.r - newlink.r);
        newlink.I = linkdata.I + ...
            [0, 0, 0; 0, lm*L^2/4, 0; 0, 0, lm*L^2/4] + ...
            [0, 0, 0; 0, newlink.m * dL^2, 0; 0, 0, newlink.m * dL^2];

        j1 = bot.link{1}; j2 = bot.link{2};
        j3 = newlink; j4 = bot.link{4};
        massbot = robot({j1, j2, j3, j4});
    end

    range = (i-1)*l+1:i*l; %place results in right spot in array
    torques(range,:) = rne(massbot, q(range,:), qd(range,:), ...
        qdd(range,:), gvect); %rne() from Corke's robotics toolbox
    %rne() performs inverse dynamics / decoupling
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %When performing inverse dynamics for multiple trajectories it is
    %important that rne() be a compiled .mex file.  This decreases the
    %computational time enormously.
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

end

% ------------------------------------------------------------------------

function create_manufacturing_maneuver()
%create trajectory [qx qy qdx qdy qddx qddy]
%create external forcing [fx, fy]
Vf = 40.1; %feed rate [cm/min]
Vf = Vf*(1/100)*(1/60); %convert to si
Favg = [-150; -200;0]; %avg milling reaction forces
                       %[feed direction, perp to feed, z]

xyt = createrobot; %create XY table using robot framework
    %for 1DOF reciprocating table only actuate y axis

n=50; %number of timesteps during each submaneuver

path_type = '1c'; % <- can alter this value
```

```
%the path being traced by XY table
% '0a' -> 1D reciprocating linear motion - 5th order poly position
% This is the path analyzed in Chapter 4, first example.
% '0b' -> %1D reciprocating linear motion - sinusoid trajectory
% '1a' -> basic U M milling
% This is the original milling path analyzed in
%Chapter 4, second example.
% '1b' -> alternate |_| M
% '1c' -> alternate U M (2 qtr circle)
% This is the alternate milling path analyzed in
%Chapter 4, second example.
% '1d' -> alternate U M (half circle)
% '2' -> Block M

% pathdata = [pts{z,x,y}, flag, DTorCS]
%flag indicates if the motion to the next point is:
% 0 -> not milling, 5th order poly
% 0.1 -> not milling, sinusoidal
% 1 -> milling, direct to next point
% 2 -> milling, half circle to next point
% 3 -> milling, quarter circle to next point
%DTorCS is the time to next point for non-milling operations
        %and is a constant velocity for milling operations

switch path_type
    case '0a'  %1D reciprocating linear motion - 5th order poly position
        pathdata = [ ...
            0, 1, 100, 0, 2 ;...
            0, 1,  0,  0, 1 ;...
            0, 1,  0,  0, 2 ;...
            0, 1, 100, 0, 1 ;...
            ];

    case '0b' %1D reciprocating linear motion - sinusoid trajectory
        pathdata = [ ...
            0, 1, 100, 0.1, 2 ;...
            %     0, 1,  0,  0.1, 1 ;... %with delays
            0, 1,  0,  0.1, 2 ;...
            %     0, 1, 100, 0.1, 1 ;... %with delays
            ];

    case '1a'% basic U M
        pathdata = [ ...
```

```
            0, 1, 100, 0, 4 ;...
            0, 1,  8, 1, Vf ;...
            0, 1,  3, 2, Vf ;...
            0, 5,  3, 1, Vf ;...
            0, 5,  8, 0, .5 ;...
            0, 9,  1, 1, Vf ;...
            0, 9,  8, 1, Vf ;...
            0, 12, 1, 1, Vf ;...
            0, 15, 8, 1, Vf ;...
            0, 15, 1, 0, 4  ...
            ];

    case '1b' %alternate |_| M
        pathdata = [ ...
            0, 1, 100, 0, 4 ;...
            0, 1,  8, 1, Vf ;...
            0, 1,  1, 0, .5 ;...
            0, 5,  1, 1, Vf ; ...
            0, 1,  1, 0, .5 ;...
            0, 5,  8, 1, Vf ; ...
            0, 5,  1, 0, .5 ;...
            0, 9,  8, 1, Vf ;...
            0, 9,  1, 0, .5 ;...
            0, 9,  8, 1, Vf ;...
            0, 12, 1, 0, .5 ;...
            0, 15, 8, 1, Vf ;...
            0, 12, 1, 0, .5 ;...
            0, 15, 8, 1, Vf ;...
            0, 15, 1, 0, 4  ...
            ];

    case '1c' % alternate U M (2 qtr circle)
        pathdata = [ ...
            0, 1, 100, 0, 4 ;...
            0, 1,  8, 1, Vf ;...
            0, 1,  3, 3, Vf ;...
            0, 3,  1, 0, .5 ; ...
            0, 5,  8, 1, Vf ; ...
            0, 5,  3, 3, Vf ;...
            0, 3,  1, 0, .5 ;...
            0, 9,  8, 1, Vf ;...
            0, 9,  1, 0, .5 ;...
            0, 9,  8, 1, Vf ;...
            0, 12, 1, 0, .5 ;...
            0, 15, 8, 1, Vf ;...
            0, 12, 1, 0, .5 ;...
            0, 15, 8, 1, Vf ;...
            0, 15, 1, 0, 4  ...
            ];

    case '1d' % alternate U M (half circle)
        pathdata = [ ...
            0, 1, 100, 0, 4 ;...
            0, 1,   8, 1, Vf ;...
            0, 1,   3, 2, Vf ;...
            0, 5,   3, 0, .5 ; ...
            0, 5,   8, 1, Vf ; ...
            0, 5,   3, 0, .5 ;...
            0, 9,   8, 1, Vf ;...
            0, 9,   1, 0, .5 ;...
            0, 9,   8, 1, Vf ;...
            0, 12, 1, 0, .5 ;...
            0, 15, 8, 1, Vf ;...
            0, 12, 1, 0, .5 ;...
            0, 15, 8, 1, Vf ;...
            0, 15, 1, 0, 4  ...
            ];

    case '2' % Block M
        pathdata = [ ...
            0, 25, 100, 0, 4 ;...
            0, 25,  20, 1, Vf ;...
            0, 17,  32, 1, Vf ;...
            0, 5,  32, 1, Vf ;...
            0, 5,  23, 1, Vf ;...
            0, 8,  23, 1, Vf ;...
            0, 8,  10, 1, Vf ;...
            0, 5, 10, 1, Vf ;...
            0, 5, 2, 1, Vf ;...
            0, 20, 2, 1, Vf ;  ...
            0, 20, 10, 1, Vf ;  ...
            0, 17, 10, 1, Vf ;  ...
            0, 17, 19, 1, Vf ;  ...
            0, 25, 7, 1, Vf ;  ...
            0, 33, 19, 1, Vf ;  ...
            0, 33, 10, 1, Vf ;  ...
            0, 30, 10, 1, Vf ;  ...
            0, 30, 2, 1, Vf ;  ...
            0, 45, 2, 1, Vf ;  ...
            0, 45, 10, 1, Vf ;  ...
            0, 42, 10, 1, Vf ;  ...
            0, 42, 23, 1, Vf ;  ...
            0, 45, 23, 1, Vf ;  ...
            0, 45, 32, 1, Vf ;  ...
            0, 33, 32, 1, Vf ;  ...
            0, 25, 20, 0, 4 ;  ...
            ];
end

%convert to si units
pathdata(:,1:end-2)=pathdata(:,1:end-2)/100;

%generate the maneuver and external forcing at each time step based on
%the pathdata coordinates
[Q, QD, QDD, TS, F] = makemaneuver(pathdata, n, Favg);

bigF = [F, zeros(size(F))]; %bigF is a matrix which contains the three
                            %external forces and moments acting on the
                            %robot due to milling at each timestep

%calculate joint forces
F=modrne(xyt, Q, QD, QDD, [0 0 9.81], bigF');
%modrne() is a slightly modified version of the rne() function provided
%in Corke's robotics toolbox.  It is altered to accept an external
%loading vector (bigF') that varies at each time step.

%do not save joint "0"
QS=Q(:,2:end);
QDS=QD(:,2:end);
QDDS=QDD(:,2:end);
FS=F(:,2:end);

name = ['XYTAB-' datestr(date) '-' path_type];
save(name, 'QS', 'QDS', 'QDDS', 'FS', 'TS')

% ----------------------------------------------------------------------

function [q, qd, qdd] = cstraj(q0, q1, n, s)
%creates a constant speed, straight path from q0 to q1
```

```
%the path has n points and moves at speed s

%initialization
q=zeros(n,length(q0));  qd=q; qdd=q;

D = norm(q0-q1); %absolute change in position

for i=1:length(q0) %for each axis
    q(:,i)  = linspace(q0(i),q1(i),n)'; %generate positions
    qd(:,i) = ones(n,1)*s/D*(q1(i)-q0(i)); %convert speed to velocity
    % qdd()  = 0  because of constant speed
end

% -----------------------------------------------------------------------

function [q, qd, qdd] = semicircle(pt1, pt2, n, s)
%creates a counter-clockwise, semi-circular from pt1 to pt2
%the path has n points and moves at speed s

%initialization
q=zeros(n,length(pt1)); qd=q; qdd=q;

%relative submaneuver end time and relative time vector
tf=(pt2(2)-pt1(2))*pi/2/s;
t = linspace(0, tf, n);

A = (pt1(2)+pt2(2))/2;      %average x position
B = (pt1(2)-pt2(2))/2;      %x amplitude
C = 2*s/(pt2(2)-pt1(2));    %speed
D = pt1(3);                 %average y position
E = B;                      %y amplitude = x amplitude

%generate motions for each joint at each timestep
q(:,2)  = A + B*cos(C*t);
q(:,3)  = D + E*sin(C*t);
qd(:,2)  = - B*C*sin(C*t);
qd(:,3)  =   C*E*cos(C*t);
qdd(:,2)  = -C^2*B*cos(C*t);
qdd(:,3)  = -C^2*E*sin(C*t);

% -----------------------------------------------------------------------

function [q, qd, qdd] = qtrcircle1(pt1, pt2, n, s)
```

```
%creates a counter-clockwise, quarter-circular from pt1 to pt2
%the path has n points and moves at speed s

%initialization
q=zeros(n,length(pt1)); qd=q; qdd=q;

%relative submaneuver end time and relative time vector
tf=abs(pt2(2)-pt1(2))*pi/2/s;
t = linspace(0, tf, n);

A = pt2(2);                  %x end coordinate
B = pt1(2)-pt2(2);           %change in x
C = s/abs(pt2(2)-pt1(2));    %speed
D = pt1(3);                  %y start coordinate
E = pt2(3)-pt1(3);           %change in y

%generate motions for each joint at each timestep
q(:,2) = A + B*cos(C*t);
q(:,3) = D + E*sin(C*t);
qd(:,2) = - B*C*sin(C*t);
qd(:,3) =   C*E*cos(C*t);
qdd(:,2) = -C^2*B*cos(C*t);
qdd(:,3) = -C^2*E*sin(C*t);
```

Optimization and Results

```
function steps4through6andresults
% This function performs step 4-6 of our six step design process:
% 4) Select PAD topology
% 5) Define powertrain
% 6) Perform PAD optimization
% This function summarizes the energy savings and plots the results.
%
% The user selects the system being analyzed. If the system has more
% than one actuated joint, a single joint must be selected for PAD
% design. The code can be executed again for the remaining joints.
%
% The loaded system contains up to seven n-dimensional arrays,
% where n is the number of simulated maneuvers:
% t - time at each time step
% y, dy, ddy - position, velocity, and acceleration of the axis/joint
% (dddy) - may also explicitely include jerk information, if not this
    %can  be approximated with finite differencing of ddy (see dAdB())
% F - The total required force (or torque) acting on the axis (or joint)
% (dF) - may also explicitely include derivative of force information,
    % if not this can be approximated with finite differencing of F
    %(see function dAdB()

clc; clear all;

global sys_type w opts
% define powertrain/joint type
% the user selects a value for pt_type.  This will load previously
% generated variables that describe the motion and forces of the axis
% (or joint).  It will also define appropriate powertrain components.

sys_type = 2; % <- can alter this value
% 2 -> The experimental LEGO single joint system.
% This is the sytem analyzed in Chapter 2.
% 3 -> A mass-spring-actuator system with ideal actuator.
% This is the sytem analyzed in Chapter 3.
% 4 -> Cartesian manufacturing machinery.
% This is the sytem analyzed in Chapter 4.
% 5 -> 3-link NBD manipulator arm.
% This is the sytem analyzed in Chapter 5.

```

```
joint_num=1; %<— can alter this value
% Some precalculated maneuvers have information for multiple axes (or
% joints).  This code only analyzes a single axis at a time, so the axis
% of interest needs to be selected. This value is meaningless for
% sys_types 2 and 3

w = 1; %<—can alter this value
% w is the weight provided to the objective function:
% w=0 —> minimize average energy consumption
% w=1 —> maximize energy saved at 10th percentile
% w not equal to 0 is only significant if analyzing a distribution of
% maneuvers

%load from file the trajectory and force profiles
load_maneuver(joint_num);

%define powertrain parameters
define_pt_parameters();

% find parallel, series, and dual spring designs for defined maneuver
opts = 'p'; %string that can contain upto one of each of the letters
               %'p', 's', and 'd'
KPs=[]; KSs=[]; KDs=[];
if ¬isempty(strfind(opts,'p')) KPs = find_spring_vals('p'); end
if ¬isempty(strfind(opts,'s')) KSs = find_spring_vals('s'); end
if ¬isempty(strfind(opts,'d')) KDs = find_spring_vals('d'); end

% tabulate results and generate figures
results(KPs, KSs, KDs)

% ————————————————————————————————————————————————————————————————————————

function load_maneuver(joint_num)
%load joint motions and required joint forces
%(or a distribution of motions and forces)

global sys_type t y dy ddy dddy F dF
%motion data should all be vertical arrays

delay = 0.02;  %time gap inserted into areas where there is a sudden
             %change of loading put no change in position

switch sys_type
```

```
    case 2  %Experimental LEGO system
        name = 'LEGO-01-Apr-2013';
        load(name, 'QS', 'QDS', 'QDDS', 'TRQS', 'TS', 'bot')
        t = stretch_time(TS,delay);
        y = QS(:,2);
        dy = QDS(:,2);
        ddy = QDDS(:,2);
        dddy = dAdB(ddy,t);
        F = TRQS(:,2);
        dF = dAdB(F,t);

        % animation of robot motions
        %figure(100); clf;
        %for i=1:size(TS,1)
        %    plot(bot, [0 QS(i,:,1)])
        %end

    case 3  %mass-spring actuator system
        name = 'MSA-29-Mar-2013';
        load(name, 't', 'y', 'dy', 'ddy', 'dddy', 'F', 'dF')
        t = t';
        y = y';
        dy = dy';
        ddy = ddy';
        dddy = dddy';
        F = F';
        dF = dF';

    case 4  %linear motion drive system
        %name = 'XYTAB-29-Mar-2013-1a';
        %name = 'XYTAB-01-Apr-2013-0a';
        name = 'XYTAB-01-Apr-2013-1c';
        load(name, 'QS', 'QDS', 'QDDS', 'FS', 'TS')
        t = stretch_time(TS,delay);
        y = QS(:,joint_num);
        dy = QDS(:,joint_num);
        ddy = QDDS(:,joint_num);
        dddy = dAdB(ddy,t);
        F = FS(:,joint_num);
        dF = dAdB(F,t);

        figure(100); clf;
        plot(QS(:,1),QS(:,2))
        axis('equal')
        title('XY Table Maneuver')
        ylabel('y position')
        xlabel('x position')

    case 5  %load a single joint maneuver from a decoupled 3—link robot
            %manipulator arm
        %name = 'PACK—01—Apr—2013—VP—50';
        name = 'PACK—01—Apr—2013—RS—50';
        %name = 'PACK—01—Apr—2013—single';
        load(name, 'QS', 'QDS', 'QDDS', 'TRQS', 'TS', 'bot')
        t = stretch_time(TS,delay);
        y=zeros(size(QS,1),size(QS,3)); dy=y; ddy=y; dddy=y; F=y; dF=y;
        y(:,:) = QS(:,joint_num,:);
        dy(:,:) = QDS(:,joint_num,:);
        ddy(:,:) = QDDS(:,joint_num,:);
        dddy(:,:) = dAdB(ddy,t);
        F(:,:) = TRQS(:,joint_num,:);
        dF(:,:) = dAdB(F,t);
        size(y)

        % animation of robot motions
        %figure(100); clf;
        %for i=1:size(TS,1)
        %    plot(bot, [0 QS(i,:,1)])
        %end
end


figure(1); clf;
subplot(411)
plot(t,y)
title('Position and derivatives')
ylabel('y')
subplot(412)
plot(t,dy)
ylabel('ydot')
subplot(413)
plot(t,ddy)
ylabel('yddot')
subplot(414)
plot(t, dddy)
ylabel('ydddot')
```

```
xlabel('Time, t [s]')

figure(2); clf;
subplot(211)
plot(t,F)
title('Force/Torque and derivatives')
ylabel('F')
subplot(212)
plot(t,dF)
ylabel('Fdot')


figure(3); clf;
subplot(121)
plot(y,-F)
xlabel('Position')
ylabel('-Force')
title('Parallel FDCF')
zoom_out()
subplot(122)
plot(dy, -dF)
xlabel('Velocity')
ylabel('- dForce')
title('Serial dFVCF')
zoom_out()

% ------------------------------------------------------------------------

function newTs = stretch_time(TS,delay)
%This function eliminates instantaneous changes in required force.
%(aka sudden increase in load mass)
%Without this series PAD designs may not work.

dt = diff(TS);              %find differences between time steps
rpts1 = (dt==0);            %find timesteps with the same time
dt(rpts1)= delay;               %add gaps
newTs = [zeros(1,size(TS,2)); cumsum(dt)];   %recompile time vector

% ------------------------------------------------------------------------

function define_pt_parameters()
%Define powertrain parameters
global sys_type bdamp n R b0 b1 k lead nu pmax Tmax wmax ...
```

```
    eps C1 C2 iwg jg jw drive
%all units SI

%pt_type is defined in the parent function SvP()

switch sys_type
    case 2 %Experimental LEGO Parameters

        %worm gear parameters
        eps = 1E-4; %speeds below this value are considered to be stuck
                    %for the purposes of worm gear equations of motion
        drive = 'NBD'; %worm gear is non-backdrivable
        rg = .0122231; %radius of the worm gear
        rw = .0037769; %radius of the worm
        iwg = 24; %speed ratio between worm and gear (thetaw=iwg*thetag)
                  %iwg = rg/rw*cot(lambda)
        lambda = atan(rg/(iwg*rw)); %lead angle of worm, 0<lambda<45
        phi = 14*pi/180;  %pressure angle of gear (angle of gear tooth)
        mu = 0.467; %average coefficient of kinetic friction
        rhoabs = 1024; %density of ABS plastic
        jw=pi*rw^3*rhoabs*rw^2; %inertia of worm
        jg=pi*rg^2*rw*rhoabs*rg^2; %inertia of gear
        %commonly used ratios
        %C1 is the torque transmission ratio through the worm gear
        %ransimssion during left engagement
        C1=rw*(cos(phi)*sin(lambda)-mu*cos(lambda))...
            /(rg*(cos(phi)*cos(lambda)+mu*sin(lambda)));
        %C2 is the torque transmission ratio through the worm gear
        %transimssion during right engagement
        C2=rw*(cos(phi)*sin(lambda)+mu*cos(lambda))...
            /(rg*(cos(phi)*cos(lambda)-mu*sin(lambda)));

        %gear box parameters
        n = 3; %gear ratio
        bdamp= 0; %viscous friction in gearhead
                  %(value included in motor friction)

        %motor parameters
        R=13.22;  %resistance internal to motor circuitry [Ohms]
        b1=0;     %viscous damping internal to motor
        b0=0.000508;    %coulomb damping internal to motor
                        %(includes gearhead friction)
        k=0.05601;   %motor constant
        %motor limits
        %(NOT TRUE motor limits — just to make code happy)
        %true limits for this experiment were not measured
        pmax = 79.2;
        Tmax = 0.531;
        wmax = 5700 *2*pi/60;

    case 3 %mass—spring actuator system
        %contains no powertrain components and an ideal actuator

    case 4 %linear motion drive system
        %(cartesian manufacturing machines)
        % these values are used for both axes in the simulated X—Y table

        % ball screw parameters
        %hiwin kk130
        lead = 0.025;
        nu = 0.9;
        %max speed .55m/s
        %max rail length 1.68 m
        %max load 48kN
        %mass = 53.9 kg

        %vvvv for maneuvers '0*' vvvv
        %gear box parameters
        n = 2; %gear ratio
        bdamp = 0;

        %motor parameters
        %micromo 2642 012 CXR
        R=1.46;     %resistance internal to motor circuitry [Ohms]
        b1=0;       %viscous damping internal to motor
        b0=0.0017;  %coulomb damping internal to motor
        k=.054;     %motor constant
        %motor limits
        pmax = 205; %motor power limit
        Tmax = .1446; %motor torque limit
        wmax = 5800 *2*pi/60; %motor speed limit

        %vvvv for maneuvers '1*' vvvv
        %gear box parameters
        n = 3;
        % bdamp=0.01;
        bdamp= 0;

        %motor parameters
        %micromo 3863 012CR
        R=2.58;    %resistance internal to motor circuitry [Ohms]
        b1=0;    %viscous damping internal to motor
        b0=0.0065;    %coulomb damping internal to motor
        k=.0797;   %motor constant
        %motor limits
        pmax = 205;
        Tmax = 1.424;
        wmax = 5600 *2*pi/60;

    case 5 % packbot joint powertrain parameters
        % these values are used for every joint
        % in the simulated packbot arm

        %worm gear parameters
        eps = 1E-4; %speeds below this value are considered to be stuck
                    %for the purposes of worm gear equations of motion
        drive = 'NBD'; %worm gear is non-backdrivable
        rg = .05; %radius of the worm gear
        rw = .01; %radius of the worm
        lambda = 15*pi/180; %lead angle of worm, 0<lambda<45
        phi = 15*pi/180;  %pressure angle of gear (angle of gear tooth)
        iwg = rg/rw*cot(lambda); %speed ratio between worm and gear
                                 %(thetaw=iwg*thetag)
        mu = 0.3; %coefficient of kinetic friction
        rhosteel=7.8*10^3; %density of steel
        jw=pi*rw^3*rhosteel*rw^2; %inertia of worm
        jg=pi*rg^2*rw*rhosteel*rg^2; %inertia of gear
        %commonly used ratios
        %C1 is the torque transmission ratio through the worm gear
        %ransimssion during left engagement
        C1=rw*(cos(phi)*sin(lambda)-mu*cos(lambda))...
            /(rg*(cos(phi)*cos(lambda)+mu*sin(lambda)));
        %C2 is the torque transmission ratio through the worm gear
        %transimssion during right engagement
        C2=rw*(cos(phi)*sin(lambda)+mu*cos(lambda))...
            /(rg*(cos(phi)*cos(lambda)-mu*sin(lambda)));

        %gear box parameters
        n = 30;  %gear ratio
        bdamp=0.01; %viscous friction in gearhead

        %motor parameters
        %micromo 3257 012CR
        R=0.41;    %resistance internal to motor circuitry [Ohms]
        b1=0;      %viscous damping internal to motor
        b0=0.0049; %constant damping internal to motor
        k=.052;    %motor current constant [A/Nm]
        %motor limits
        pmax = 79.2; %motor power limit
        Tmax = 0.531; %motor torque limit
        wmax = 5700 *2*pi/60; %motor speed limit

    otherwise
        error('invalid powertrain type');
end

% ------------------------------------------------------------------------

function kvals = find_spring_vals(type)

%declare global variables
global y dy F dF spd_type
spd_type = type;
%find optimal spring values
% fmincon settings
options = optimset('Display','iter', 'TolFun', 1e-5, ...
    'DiffMinChange', 1e-3, 'Algorithm', 'active-set');
% matrix/vectors for defining linear constraints (not used)
A=[ ]; b=[ ]; Aeq=[ ]; beq=[ ];

switch type
    case 'p' %parallel system
        kvals=zeros(4,2);  %no spring
        kvals(2,:) = polyfit(y, -F, 1); %FDCF design
        kvals(3,:) = wpolyfit(y, -F, abs(y.*dy), 1); %FDCF+ design

        %determine optimal spring value
        x0 = kvals(2,:); %use FDCF+ for optimization initialization

        lb = [0,-10000];        % lower bounds on the problem
        ub = [1000,10000];       % upper bounds on the problem

    case 's' %series system
        kvals=zeros(4,1);
        kvals(1) = inf;              %no spring
        temp = polyfit(dy, −dF, 1);  %dFVCF design
        kvals(2) = temp(1);          %dFVCF design (stiffness only)
        temp = wpolyfit(dy, −dF, abs(F.*dF), 1);  %dFVCF+ design
        kvals(3) = temp(1); %dFVCF+ design (stiffness only)

        %determine optimal spring value
        x0 = kvals(3); %use dFVCF+ for optimization initialization
        % fmincon settings

        lb = [0.0001];          % lower bounds on the problem
        ub = [inf];       % upper bounds on the problem (not used)

    case 'd' %dual PAD system
        kvals=zeros(4,3);
        kvals(:,3)=inf*ones(4,1); %no spring

        %FDCF/dFVCF
        kvals(2,1:2) = polyfit(y, −F, 1); %parallel FDCF design
        temp = polyfit(dy, −dF, 1);       %series dFVCF design
        kvals(2,3) = temp(1); %series stiffness only

        %FDCF+/dFVCF+
        kvals(3,1:2) = wpolyfit(y, −F, abs(y.*dy), 1); %FDCF+ design
        temp = wpolyfit(dy, −dF, abs(F.*dF), 1); %dFVCF+ design
        kvals(3,3) = temp(1); %series stiffness only

        %determine optimal spring value
        x0 = kvals(3,:); %use FDCF+/dFVCF+ for opt initialization

        lb = [0, −10000, .00001];   % lower bounds on the problem
        ub = [1000, 10000, inf];     % upper bounds on the problem

end
%find optimal spring values
[kvals(4,:),¬,¬,¬] = fmincon(@min_E, double(x0), A, b, ...
    Aeq, beq, lb, ub, @NLCON, options);

% ———————————————————————————————————————————————————————————————————————

function [f, Tdata, E, StDev, C]=min_E(x)
```

```
%returns the electrical energy required by the motor to complete the
%prescribed maneuver
global t y dy F dF spd_type w

switch spd_type
    case 'p'
        %parallel variables
        k = x(1); F0 = x(2);

        %parallel force
        Fk = k*y + F0;
        Fp = F + Fk;
        Tdata=traj2Data(dy, Fp);

    case 's'
        %series variables
        k = x;

        %series speed
        dyk = -dF/k;
        dys = dy - dyk;
        Tdata=traj2Data(dys, F);

    case 'd'
        %dual spring variables
        kp = x(1); F0 = x(2); ks = x(3);

        %parallel force
        Fk = kp*y + F0;
        Fp = F + Fk;
        %series speed
        dyk = -dF/ks;
        dys = dy - dyk;

        %dual spring power
        Tdata=traj2Data(dys, Fp);

end
%required absolute actuator power and energy
Pa = abs(Tdata.apower);
energy = intV(Pa,t);

%average and stdev of energy
```

```
E = mean(energy);
StDev = sqrt(var(energy));

%no spring energy
Tdata0=traj2Data(dy,F);
energy0 = intV(abs(Tdata0.apower),t);

%pct saved by opt PAD
S = (1-energy./energy0)*100;
muS = mean(S);
sigS = sqrt(var(S));

%percent energy saved by PAD at 10th percentile
C = (muS-1.18*sigS);

%objective function is a weighted combination of minimizing average
%energy consumption and maximizing the percent energy saved at the 10th
%percentile
f = E*(1-w) - w*C;

% ------------------------------------------------------------------------

function [Data]=traj2Data(dy, F)
%input: powertrain output velocity, acceleration, and force
%output: required motor current and voltage; also intermediate values
global sys_type
switch sys_type
    case 3 %no powertrain and ideal components
        Data.apower=F.*dy;

    case 4 %X-Y table powertrain

        %shaft speed and torque supplied to the ballscrew
        [dy_w_in, Tw_in] = ballscrew(dy, F);

        %shaft speed and torque supplied to the gearhead
        [dy_gh_in, Tgh_in] = gearhead(dy_w_in, Tw_in);

        %current and voltage supplied to the DC motor
        [current, voltage] = DCmotor(dy_gh_in, Tgh_in);

        %Packaging
        Data.Tw_in=Tw_in; Data.omega_w_in=dy_w_in;
        Data.Tgh_in=Tgh_in; Data.omega_gh_in=dy_gh_in;
        Data.current=current; Data.voltage=voltage;
        Data.apower=current.*voltage;

    otherwise %case 2 or 5, LEGO powertrain or packbot joint

        %shaft speed and torque supplied to the worm gear
        [dy_w_in, Tw_in] = wormgear(dy, F);

        %shaft speed and torque supplied to the gearhead
        [dy_gh_in, Tgh_in] = gearhead(dy_w_in, Tw_in);

        %current and voltage supplied to the DC motor
        [current, voltage] = DCmotor(dy_gh_in, Tgh_in);

        %Packaging
        Data.Tw_in=Tw_in; Data.omega_w_in=dy_w_in;
        Data.Tgh_in=Tgh_in; Data.omega_gh_in=dy_gh_in;
        Data.current=current; Data.voltage=voltage;
        Data.apower=current.*voltage;

end

% ----------------------------------------------------------------------

function [dy_in, T_in] = wormgear(dy_out, T_out)
%wormgear() calculates the angular shaft velocity and torque supplied
%to the worm given information on the wormgear shaft output variables

global eps C1 C2 iwg jg jw drive t
ddy_out = dAdB(dy_out,t);
% T_in = zeros(size(T_out));

%calculate input shaft speed
dy_in=dy_out.*iwg;


%calculate input shaft torque - governing equation varies depending on
%operating conditions

% h2 = dy_out>eps | (abs(dy_out)<eps & dydot_out>eps);

%assume motor driven
```

```
T_in=(jw.*iwg+C2.*jg).*ddy_out+C2.*T_out;

h2a = ((dy_out>eps | (abs(dy_out)<eps & ddy_out>eps)) ...
    & (T_in > (jw.*iwg./jg.*T_out))) | ( (T_in≤jw.*iwg./jg.*T_out) ...
    & (dy_out<-eps | (abs(dy_out)<eps & ddy_out<-eps)));
%assumption valid if h2a(i)==true

%assumption invalid if h2a(i)==false, assume left engagement
T_in(¬h2a)=(jw.*iwg+C1.*jg).*ddy_out(¬h2a)+C1.*T_out(¬h2a);


%The arm is stuck, Tm can be any number of values such that
%friction stays below the friction limit, pick Tm(j)=0 for a NBD
%setup
h1 = abs(dy_out)<eps & abs(ddy_out)<eps & strcmp(drive,'NBD');
T_in(h1)=0;

% ------------------------------------------------------------------------

function [dy_in, T_in] = ballscrew(dy_out, F_out)
%ballscrew() calculates the angular shaft velocity and torque supplied
%to the ballscrew given information on the linear output variables

global lead nu

T_in = F_out*lead/(2*pi*nu);
dy_in = dy_out*2*pi/lead;

% ------------------------------------------------------------------------

function [dy_in, T_in] = gearhead(dy_out, T_out)
%gearhead() calculates the angular shaft velocity and torque supplied to
%the gearhead given information on the gearhead shaft output variables

global n bdamp

%calculate the input shaft speed
dy_in = dy_out.*n;

%calculate the input shaft torque
%equal to the output torque plus the torque from viscous friction
%divided by the gear ratio n
T_in = (T_out + bdamp .* dy_out)./n;
```

```
% ------------------------------------------------------------------------

function [current, voltage] = DCmotor(dy_in, T_in)
%DCmotor() calculates the current and voltage supplied to the motor
%given information on the motor shaft output variables

global R k b0 b1
% Pelec = current * voltage;
% Pmech = T_in * dy_in;
% eff = Pmech/Pelec;

T_motor = T_in + dy_in.*b1 + sign(dy_in).*b0;
current = T_motor./k;
voltage = current.*R + dy_in.*k;

% ------------------------------------------------------------------------

function results(KPs, KSs, KDs)
% tabulate results and generate figures

global t y dy F dF spd_type opts

%unassisted power and energy
Tdata0=traj2Data(dy,F);
P0=abs(Tdata0.apower);

%parallel results ----------------------------------------------
if ¬isempty(strfind(opts,'p'))
spd_type = 'p';

%plot spring designs on FDCF figure
figure(3);
subplot(121)
xrange = get(gca,'xlim');
hold on
%FDCF design
Fk =KPs(2,1)*xrange+KPs(2,2);
plot(xrange,Fk,'--')
%FDCF+ design
Fk =KPs(3,1)*xrange+KPs(3,2);
plot(xrange,Fk,'-.')
%Energy Minimizing (or other optimal) design
```

```
Fk =KPs(4,1)*xrange+KPs(4,2);
plot(xrange,Fk,':')
legend('F vs ydot', 'FDCF', 'FDCF=', 'Min E', ...
    'location','northwest')

%plot power profiles
Pp2 = abs(dy.*(F+KPs(2,1)*y+KPs(2,2)));
Pp3 = abs(dy.*(F+KPs(3,1)*y+KPs(3,2)));
Pp4 = abs(dy.*(F+KPs(4,1)*y+KPs(4,2)));
figure(4)
plot(t,P0,'-',t,Pp2,'--',t,Pp3,'-.',t,Pp4,':')
title('Parallel Spring Power Profiles')
xlabel('time, t [s]')
ylabel('absolute power')
legend('No spring', 'FDFC','FDCF+', 'Min E', 'location','east')

%energy summary
disp('Parallel Spring Energy Summary')
disp(['Stiffness | Preload | Energy | E_svd [%] |' ...
    ' StDev | S_svd [%] | C [%]']);
Ep = zeros(4,1); Sp = Ep; Cp = Sp;
pctEp = Ep; pctSp = pctEp;
for i = 1:4
    [¬, ¬, Ep(i), Sp(i), Cp(i)] = min_E(KPs(i,:));
    pctEp(i) = 100*(1-Ep(i)/Ep(1));
    pctSp(i) = 100*(1-Sp(i)/Sp(1));
end
disp([KPs Ep pctEp Sp pctSp Cp])
end

%series results ------------------------------------------------
if ¬isempty(strfind(opts,'s'))
spd_type = 's';

%plot spring designs on dFVCF figure
figure(3)
subplot(122)
xrange = get(gca,'xlim');
hold on
%dFVCF design
Fk =KSs(2)*xrange;
plot(xrange,Fk,'--')
%dFVCF+ design
```

```
Fk =KSs(3)*xrange;
plot(xrange,Fk,'-.')
%Energy Minimizing (or other optimal) design
Fk =KSs(4)*xrange;
plot(xrange,Fk,':')
legend('dForce-Velocity profile', 'dFVCF', 'dFVCF+', 'Min E', ...
    'location','northwest')

%plot power profiles
Ps2 = abs((dy+dF/KSs(2)).*F);
Ps3 = abs((dy+dF/KSs(3)).*F);
Ps4 = abs((dy+dF/KSs(4)).*F);
figure(5)
plot(t,P0,'-',t,Ps2,'--',t,Ps3,'-.',t,Ps4,':')
title('Serial Spring Power Profiles')
xlabel('time, t [s]')
ylabel('absolute power')
legend('No spring', 'dFVCF','dFVCF+','Min E', 'location','east')

%energy summary
disp('Serial Spring Energy Summary')
disp(['Stiffness |  Energy | E_svd [%] |' ...
    ' StDev | S_svd [%] | C [%]']);
Es = zeros(4,1); Ss = Es; Cs = Ss;
pctEs = Es; pctSs = pctEs;
for i = 1:4
    [~, ~, Es(i), Ss(i), Cs(i)] = min_E(KSs(i,:));
    pctEs(i) = 100*(1-Es(i)/Es(1));
    pctSs(i) = 100*(1-Ss(i)/Ss(1));
end
disp([KSs Es pctEs Ss pctSs Cs])
end

%dual spring results -----------------------------------------------
if ~isempty(strfind(opts,'d'))
spd_type = 'd';

%plot power profiles
Pp2 = abs((dy+dF/KDs(2,3)).*(F+KDs(2,1)*y+KDs(2,2)));
Pp3 = abs((dy+dF/KDs(3,3)).*(F+KDs(3,1)*y+KDs(3,2)));
Pp4 = abs((dy+dF/KDs(3,3)).*(F+KDs(4,1)*y+KDs(4,2)));
figure(6)
% subplot(211) %whole picture
```

```
plot(t,P0,'-',t,Pp2,'--',t,Pp3,'-.',t,Pp4,':')
title('Dual Spring Power Profiles')
xlabel('time, t [s]')
ylabel('absolute power')
legend('No spring', 'CFs','CFs+', 'Min E', 'location','east')

%energy summary
disp('Dual Spring Energy Summary')
disp(['Parallel Stiffness | Preload | Serial Stiffness |' ...
    ' Energy | E_svd [%] | StDev | S_svd [%] | C [%]']);
Ed = zeros(4,1); Sd = Ed; Cd = Sd;
pctEd = Ed; pctSd = pctEd;
for i = 1:4
    [¬, ¬, Ed(i), Sd(i), Cd(i)] = min_E(KDs(i,:));
    pctEd(i) = 100*(1-Ed(i)/Ed(1));
    pctSd(i) = 100*(1-Sd(i)/Sd(1));
end
disp([KDs Ed pctEd Sd pctSd Cd])
end

% ----------------------------------------------------------------------

function [c, ceq] = NLCON(x)
global sys_type  pmax Tmax wmax
switch sys_type
    case 3 %ideal actuator -> no motor constraints
        c = [];
        ceq = [];

    otherwise %packbot motor constraints
        [¬,Tdata]=min_E(x);

        c(1) = max(max(abs(Tdata.current.*Tdata.voltage))) - pmax;
        c(2) = max(max(abs(Tdata.Tgh_in))) - Tmax;
        c(3) = max(max(abs(Tdata.omega_gh_in))) - wmax;
        ceq = [];
end

% ----------------------------------------------------------------------

function result = dAdB(A, B)
%approximate derivate of A wrt B
% A and B must the same size
```

```
dA = [diff(A);zeros(1,size(A,2))];
dB = [diff(B);ones(1,size(B,2))];
result = dA./dB;
```

BIBLIOGRAPHY

[1] Brown, W. R., and Ulsoy, A. G., 2011. "A passive-assist design approach for improved reliability and efficiency of robot arms". In Proc. IEEE International Conference on Robotics and Automation (ICRA), pp. 4927–4934.

[2] Yeh, T. J., and Wu, F. K., 2009. "Modeling and robust control of worm-gear driven systems". *Simulation Modelling Practice and Theory,* **17**(5), pp. 767–777.

[3] HIWIN. *Single Axis Robot Technical Information*, k02te07-0912 ed.

[4] Faulhaber. *DC-Micromotors, Graphite Commutation, 23 mNm, Series 2642 ...CXR*, 2012-2013 ed. Dr. Fritz Faulhaber GMBH & Co. KG.

[5] Faulhaber. *DC-Micromotors, Graphite Commutation, 70 mNm, Series 3257 ... CR*, 2012-2013 ed. Dr. Fritz Faulhaber GMBH & Co. KG.

[6] Bryan, C., Grenwalt, M., and Stienecker, A., 2010. "Energy consumption reduction in industrial robots". In Proc. ASEE North Central Sectional Conference.

[7] U.S. Environmental Protection Agency, 2007. eGRID. ONLINE . http://www.epa.gov/cleanenergy/energy-resources/egrid/index.html.

[8] Brown, W. R., and Ulsoy, A. G., 2013. "A maneuver based design of a passive-assist device for augmenting active joints". *Journal of Mechanisms and Robotics.* (in press).

[9] Devoldere, T., Dewulf, W., Deprez, W., Willems, B., and Duflou, J., 2007. "Improvement potential for energy consumption in discrete part production machines". In *Advances in Life Cycle Engineering for Sustainable Manufacturing Businesses*, S. Takata and Y. Umeda, eds. Springer London, pp. 311–316.

[10] Vijayaraghavan, A., and Dornfeld, D., 2010. "Automated energy monitoring of machine tools". *CIRP Annals - Manufacturing Technology,* **59**(1), pp. 21 – 24.

[11] Özbayrak, M., Akgün, M., and Türker, A., 2004. "Activity-based cost estimation in a push/pull advanced manufacturing system". *International Journal of Production Economics,* **87**(1), pp. 49 – 65.

[12] Dahmus, J., and Gutowski, T., 2004. "An environmental analysis of machining". In ASME International Mechanical Engineering Congress and RD&D Exposition, Anaheim, California, USA.

[13] Gutowski, T., Dahmus, J., and Thiriez, A., 2006. “Electrical energy requirements for manufacturing processes”. In Proc. 13th CIRP International Conference on Life Cycle Engineering.

[14] Brown, W. R., and Ulsoy, A. G., 2013. “Maneuver based design of a passive-assist device for augmenting linear motion drives”. In Proc. American Control Conference (ACC).

[15] Tilbury, D. M., and Ulsoy, A. G., 2011. “A new breed of robots that drive themselves”. *ASME Mechanical Engineering Magazine,* **133**(2), pp. 28–33.

[16] Lukic, S., Cao, J., Bansal, R., Rodriguez, F., and Emadi, A., 2008. “Energy storage systems for automotive applications”. *IEEE Transactions on Industrial Electronics,* **55**(6), pp. 2258–2267.

[17] Schupbach, R., Balda, J., Zolot, M., and Kramer, B., 2003. “Design methodology of a combined battery-ultracapacitor energy storage unit for vehicle power management”. In IEEE 34th Annual Power Electronics Specialist Conference, Vol. 1, pp. 88–93.

[18] Tarquin, A., and Dowdy, J., 1989. “Optimal pump operation in water distribution”. *Journal of Hydraulic Engineering,* **115**(2), pp. 158–168.

[19] Barton, J., and Infield, D., 2004. “Energy storage and its use with intermittent renewable energy”. *IEEE Transactions on Energy Conversion,* **19**(2), pp. 441–448.

[20] Chiang, S. J., Chang, K., and Yen, C. Y., 1998. “Residential photovoltaic energy storage system”. *IEEE Transactions on Industrial Electronics,* **45**(3), pp. 385–394.

[21] Abbey, C., and Joos, G., 2007. “Supercapacitor energy storage for wind energy applications”. *IEEE Transactions on Industry Applications,* **43**(3), pp. 769–776.

[22] Herder, J., 2001. “Energy-free systems: theory, conception, and design of statically balanced spring mechanisms”. PhD Thesis, Delft University of Technology, Delft, Netherlands.

[23] Li, S. C., Qiu, J. X., and Zhu, J. Y., 1990. “The counterbalance design of the articulated robot arms”. *CIRP Annals - Manufacturing Technology,* **39**(1), pp. 455–458.

[24] Akeel, H. A., 1987. Robot with counterbalance mechanism having multiple attachment locations. U. S. Patent 4653975.

[25] Gorman, R. H., and Aggarwal, R. N., 1988. Industrial robot having counterbalanced arms. U. S. Patent 4768918.

[26] Krut, S., Benoit, M., Dombre, E., and Pierrot, F., 2010. "Moonwalker, a lower limb exoskeleton able to sustain bodyweight using a passive force balancer". In Proc. IEEE International Conference on Robotics and Automation (ICRA), pp. 2215–2220.

[27] Lu, Q., McAvoy, J., and Ma, O., 2009. "A simulation study of a reduced-gravity simulator for simulating human jumping and walking in a reduced-gravity environment". In Proc. ASME Dynamic Systems and Control Conference (DSCC), pp. 1675–1682.

[28] Frey, M., Colombo, G., Vaglio, M., Bucher, R., Jorg, M., and Riener, R., 2006. "A novel mechatronic body weight support system". *IEEE Transactions on Neural Systems and Rehabilitation Engineering,* **14**(3), pp. 311–321.

[29] Walsh, C. J., Pasch, K., and Herr, H., 2006. "An autonomous, underactuated exoskeleton for load-carrying augmentation". In Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1410 –1415.

[30] Downer, E. E. J., 2002. Energy conservation system for earth-moving loading machines. U. S. Patent 6497059.

[31] Mahalingam, S., and Sharan, A., 1986. "The optimal balancing of the robotic manipulators". In Proc. IEEE International Conference on Robotics and Automation (ICRA), Vol. 3, pp. 828–835.

[32] Lin, P.-Y., 2012. "Design of statically balanced spatial mechanisms with spring suspensions". *Journal of Mechanisms and Robotics,* **4**(2), p. 021015.

[33] Deepak, S. R., and Ananthasuresh, G. K., 2012. "Perfect static balance of linkages by addition of springs but not auxiliary bodies". *Journal of Mechanisms and Robotics,* **4**(2), p. 021014.

[34] Endo, G., Yamada, H., Yajima, A., Ogata, M., and Hirose, S., 2010. "A passive weight compensation mechanism with a non-circular pulley and a spring". In Proc. IEEE International Conference on Robotics and Automation (ICRA), pp. 3843–3848.

[35] McMahon, T. A., 1984. *Muscles, Reflexes, and Locomotion.* Princeton University Press.

[36] Anderson, F. C., and Pandy, M. G., 1993. "Storage and utilization of elastic strain energy during jumping". *Journal of Biomechanics,* **26**(12), pp. 1413–1427.

[37] Migliore, S., Brown, E., and DeWeerth, S., 2005. "Biologically inspired joint stiffness control". In Proc. IEEE International Conference on Robotics and Automation (ICRA), pp. 4508–4513.

[38] Hutter, M., Remy, C., Hoepflinger, M., and Siegwart, R., 2011. "High compliant series elastic actuation for the robotic leg ScarlETH". In Proc. International Conference on Climbing and Walking Robots (CLAWAR).

[39] Vanderborght, B., Van Ham, R., Lefeber, D., Sugar, T. G., and Hollander, K. W., 2009. "Comparison of mechanical design and energy consumption of adaptable, passive-compliant actuators". *The International Journal of Robotics Research,* **28**(1), pp. 90–103.

[40] Pratt, G., and Williamson, M., 1995. "Series elastic actuators". In Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems., Vol. 1, pp. 399–406.

[41] Yang, T., Westervelt, E., Schmiedeler, J., and Bockbrader, R., 2008. "Design and control of a planar bipedal robot ERNIE with parallel knee compliance". *Autonomous Robots,* **25**(4), pp. 317–330.

[42] Harper, K., Berkemeier, M., and Grace, S., 1997. "Decreasing the energy costs of swimming robots through passive elastic elements". In Proc. IEEE International Conference on Robotics and Automation (ICRA), Vol. 3, pp. 1839–1844.

[43] Mettin, U., La Hera, P. X., Freidovich, L. B., and Shiriaev, A. S., 2010. "Parallel elastic actuators as a control tool for preplanned trajectories of underactuated mechanical systems". *The International Journal of Robotics Research,* **29**(9), pp. 1186–1198.

[44] Brown, W. R., and Ulsoy, A. G., 2012. "Experimental verification of a passive-assist design approach for improved reliability and efficiency of robot arms". In Proc. ASME Dynamic Systems and Control Conference (DSCC).

[45] Brown, W. R., and Ulsoy, A. G., 2013. "Robust maneuver based design of a passive-assist device for augmenting active robotic joints". In Proc. ASME Dynamic Systems and Control Conference (DSCC). (submitted).

[46] Wang, S., van Dijk, W., and van der Kooij, H., 2011. "Spring uses in exoskeleton actuation design". In Proc. IEEE International Conference on Rehabilitation Robotics (ICORR), pp. 1–6.

[47] Dept. of Defense, 1991. *Military Handbook, Reliability Prediction of Electronic Equipment (MIL-HDBK-217F)*, Dec.

[48] Hirschmann, D., Tissen, D., Schroder, S., and De Doncker, R., 2007. "Reliability prediction for inverters in hybrid electrical vehicles". *Power Electronics, IEEE Transactions on,* **22**(6), Nov., pp. 2511–2517.

[49] Nagamachi, M., 1986. "Human factors of industrial robots and robot safety management in japan". *Applied Ergonomics,* **17**(1), pp. 9–18.

[50] Bicchi, A., Tonietti, G., Bavaro, M., and Piccigallo, M., 2005. "Variable stiffness actuators for fast and safe motion control". In *Robotics Research*, P. Dario and R. Chatila, eds., Vol. 15 of *Springer Tracts in Advanced Robotics*. Springer Berlin Heidelberg, pp. 527–536.

[51] Park, J.-J., and Song, J.-B., 2010. "Safe joint mechanism using inclined link with springs for collision safety and positioning accuracy of a robot arm". In Proc. IEEE International Conference on Robotics and Automation (ICRA), pp. 813–818.

[52] Haddadin, S., Albu-Schaffer, A., and Hirzinger, G., 2008. "The role of the robot mass and velocity in physical human-robot interaction — Part I: Non-constrained blunt impacts". In Proc. IEEE International Conference on Robotics and Automation (ICRA), pp. 1331–1338.

[53] Haddadin, S., Albu-Schaffer, A., Frommberger, M., and Hirzinger, G., 2008. "The role of the robot mass and velocity in physical human-robot interaction — Part II: Constrained blunt impacts". In Proc. IEEE International Conference on Robotics and Automation (ICRA), pp. 1339–1345.

[54] Brown, W. R., and Ulsoy, A. G., 2013. "Maneuver based design of passive-assist devices: a comparison of parallel and serial systems". In Proc. ASME Dynamic Systems and Control Conference (DSCC). (submitted).

[55] Brown, W. R., and Ulsoy, A. G., 2014. "Multi-dof and robust design of passive assist devices". *Journal of Mechanisms and Robotics.* (submitted).

[56] Slocum, A. H., and Weber, A. C., 2003. "Precision passive mechanical alignment of wafers". *Journal of Microelectromechanical Systems,* **12**(6), pp. 826–834.

[57] Radaelli, G., Gallego, J. A., and Herder, J. L., 2011. "An energy approach to static balancing of systems with torsion stiffness". *Journal of Mechanical Design,* **133**(9), p. 091006.

[58] Nassiraei, A. A. F., and Ishii, K., 2007. "Concept of intelligent mechanical design for autonomous mobile robots". *Journal of Bionic Engineering,* **4**(4), pp. 217–226.

[59] Chalhoub, N. G., and Ulsoy, A. G., 1986. "Dynamic simulation of a leadscrew driven flexible robot arm and controller". *ASME Journal of Dynamic Systems, Measurement, and Control,* **108**(2), pp. 119–126.

[60] Chalhoub, N. G., and Ulsoy, A. G., 1987. "Control of a flexible robot arm: Experimental and theoretical results". *ASME Journal of Dynamic Systems, Measurement, and Control,* **109**(4), pp. 299–309.

[61] Reyer, J. A., and Papalambros, P. Y., 2002. "Combined optimal design and control with application to an electric dc motor". *ASME Journal of Mechanical Design,* **124**(2), pp. 183–191.

[62] Åström, K. J., Apkarian, J., and Lacheray, H., 2006. *DC Motor Control Trainer Instructor Workbook.* Quanser Inc., 119 Spy Court, Markham, Ontario L3R 5H6 CANADA.

[63] Li, S., Kolmanovsky, I. V., and Ulsoy, A. G., 2012. "Distributed supervisory controller design for battery swapping modularity in plug-in hybrid electric vehicles". *ASME J. Dynamic Systems Measurement and Control,* **134**(4), May.

[64] Phadke, M., 1989. *Quality Engineering Using Robust Design.* Prentice Hall, Englewood Cliffs, NJ.

[65] Current, A., and Lucas, C., 1996. Linear drive power door operator. U. S. Patent 5513467, May 7.

[66] Grumazescu, M., 1998. Electromagnetic linear drive. U. S. Patent 5809157, Sept. 15.

[67] Gregorio, P., Ahmadi, M., and Buehler, M., 1997. "Design, control, and energetics of an electrically actuated legged robot". *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on,* **27**(4), aug, pp. 626 –634.

[68] Özel, T., and Altan, T., 2000. "Process simulation using finite element method prediction of cutting forces, tool stresses and temperatures in high-speed flat end milling". *International Journal of Machine Tools and Manufacture,* **40**(5), pp. 713 – 738.

[69] Scheint, M., Sobotka, M., and Buss, M., 2010. "Optimized parallel joint springs in dynamic motion: Comparison of simulation and experiment". In IEEE RAS and EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob), pp. 485–490.

[70] Lee, C. S. G., and Chang, P. R., 1986. "Efficient parallel algorithm for robot inverse dynamics computation". *Systems, Man and Cybernetics, IEEE Transactions on,* **16**(4), july, pp. 532 –542.

[71] Rodriguez, G., 1987. "Kalman filtering, smoothing, and recursive robot arm forward and inverse dynamics". *Robotics and Automation, IEEE Journal of,* **3**(6), december, pp. 624 –639.

[72] Corke, P. I., 1996. "A robotics toolbox for MATLAB". *IEEE Robotics and Automation Magazine,* **3**(1), Mar, pp. 24–32.

[73] Du, L., Choi, K. K., Youn, B. D., and Gorsich, D., 2006. "Possibility-based design optimization method for design problems with both statistical and fuzzy input data". *Journal of Mechanical Design,* **128**(4), pp. 928–935.

[74] Du, X., and Chen, W., 2002. "Efficient uncertainty analysis methods for multidisciplinary robust design". *AIAA Journal,* **40**(3), p. 8.

[75] Doltsinis, I., and Kang, Z., 2004. “Robust design of structures using optimization methods”. *Computer Methods in Applied Mechanics and Engineering,* **193**(23–26), pp. 2221–2237.

[76] Xing, T., and Zhou, X., 2013. “Reformulation and solution algorithms for absolute and percentile robust shortest path problems”. *Intelligent Transportation Systems, IEEE Transactions on,* **PP**(99), pp. 1–12.

[77] Peters, D. L., Papalambros, P. Y., and Ulsoy, A. G., 2010. “Relationship between coupling and the controllability grammian in co-design problems”. In Proc. American Control Conference (ACC), pp. 623–628.