THE WINCH-BOT: A CABLE-SUSPENDED, UNDER-ACTUATED ROBOT UTILIZING PARAMETRIC SELF-EXCITATION

by

Daniel Philip Cunningham

B.S., Mechanical Engineering, Massachusetts Institute of Technology, 2007

Submitted to the Department of Mechanical Engineering in partial fulfillment of the requirements for the degree of
Master of Science in Mechanical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2009

Author .....
Department of Mechanical Engineering
May 22, 2009

Certified by.....
H. Harry Asada
Ford Professor of Mechanical Engineering
Thesis Supervisor

Accepted by .....
David E. Hardt
Chairman, Department Committee on Graduate Students
Department of Mechanical Engineering

THE WINCH-BOT: A CABLE-SUSPENDED, UNDER-ACTUATED ROBOT UTILIZING PARAMETRIC SELF-EXCITATION

by

Daniel Philip Cunningham

B.S., Mechanical Engineering, Massachusetts Institute of Technology, 2007

Submitted to the Department of Mechanical Engineering in partial fulfillment of the requirements for the degree of
Master of Science in Mechanical Engineering

Abstract

A simple, compact, yet powerful robotic winch, called "Winch-Bot," is presented in this thesis. The Winch-Bot is an underactuated robot having only one controllable axis. Although hanging a load with merely one cable, it is capable of moving it in a large workspace by swinging the load dynamically based on parametric self-excitation. The generated trajectories can be used for a variety of tasks, from moving material to cyclic inspection of surfaces. The basic principle and design concept of the Winch-Bot are first described, followed by dynamic modeling and analysis. Two trajectory generation problems are solved. One is point-to-point transfer of a load, and the other is the tracking of a continuous path. It will be shown that the system can track a given geometric trajectory, although the tracking velocity cannot be determined arbitrarily due to the underactuated nature of dynamics. A prototype Winch-Bot is designed and built, and point-to-point, continuous path, and parametric excitation control are implemented.

Thesis Supervisor: H. Harry Asada
Title: Ford Professor of Mechanical Engineering

ACKNOWLEDGEMENTS

First and foremost, I would like to thank Professor Asada for his priceless guidance; his vast knowledge and experience has helped to guide me along the correct path for the past two years at MIT.

Secondly, I would like to extend my eternal thanks to my wife, Marjorie, for always being there for me, even when my work takes me to places I never thought I'd go. Her constant support has always been a beacon to sanity and happiness in my life.

I would like to thank my parents for their love and support, even as I was the first to leave the proverbial nest. They have been the kind of parents who keep in close contact even from a thousand miles away.

Special thanks should be given to my undergraduate professors at MIT, who helped guide me to the field of study I now know I love.

Finally, thanks to all my friends at the d'Arbeloff lab for their sanity checks, their late-night theoretical discussions, and their watchful eye in the machine shops. So much was accomplished because of your help.

TABLE OF CONTENTS

TABLE OF FIGURES

Chapter 1. INTRODUCTION

1.1. The Need for an Underactuated Pendulum for Motion Control

There is an increasing need for a simple, compact, and flexible solution to transportation of material and end-effecter in a *large* workspace. These needs include the manufacturing of aircraft and ships, construction of buildings and infrastructures, monitoring and inspection over broad areas, and security checks of large structures. Traditional overhead cranes are slow and bulky, and need a costly infrastructure to install and use the machine. Long arm manipulators are inefficient for covering a large space, since the actuators and arm structure become significantly larger as the required workspace increases. Mobile robots, on the other hand, are advantageous in terms of workspace to cover, but complex environments having many obstacles and rough terrain often make it difficult to deploy mobile robots.

1.2. Goal

The objective of my thesis is to fill the technological gap between mobile robotics and the fixed crane and manipulator technologies. A simple, compact, winch-like robot called a "Winch-Bot" will be developed to transport a heavy end-effecter swiftly in a large workspace. The Winch-Bot is fixed to a ceiling, a mobile platform with a high attachment

point, or any structure above the workspace. There it hangs an end-effecter at the tip of a cable, and swings the end-effecter by controlling the length of the string in real-time. The swing amplitude can be modulated by parametric self-excitation. With only one or two axes of servoed joints, the Winch-Bot can control the end-effecter trajectory in a large workspace.

1.3. Previous Work

In the robotics literature, multi-cable cranes have been studied extensively [1-3]. These advanced cranes can transport heavy objects by coordinating multiple winches that control the lengths of the multiple cables. Despite the dexterity and stability, the multi-cable cranes need numerous winch systems placed at different locations in the space. The usable workspace is substantially smaller than that of the envelope inside the multiple winches. The casting manipulator [4], on the other hand, can project an end-effecter in a long range by swinging a fly-fishing-rod-type manipulator and retrieve it by tugging the string [4]. For light payload throwing tasks, the casting manipulator approach is effective particularly for long-range projection. There, though, the real-time control of the thrown trajectory is only variable by small amounts, so is unsuitable for tracking an arbitrary geometric path.

1.4. Schematic description

The Winch-Bot consists of only one axis of winch placed at a fixed point over a large workspace. It does not need a manipulator arm to swing the end-effecter; rather, it oscillates the end-effecter suspended with a cable by simply extending and contracting the cable's length. This type of parametric self-excitation has a form similar to the well-

known partial differential equation, Mathieu's Equation, and will be discussed in more detail in Section 4.1.2.1.1 on page 33. The simple, compact structure of the Winch-Bot composed of only one smart winch is advantageous for dealing with a heavy payload. Furthermore, the Winch-Bot can generate both point-to-point motion with soft-landing capabilities *and* continuous path motion for tracking a geometric trajectory. This allows us not only to throw an object but also to track along a surface, search a wide area, and perform a variety of tasks that require path control capabilities.

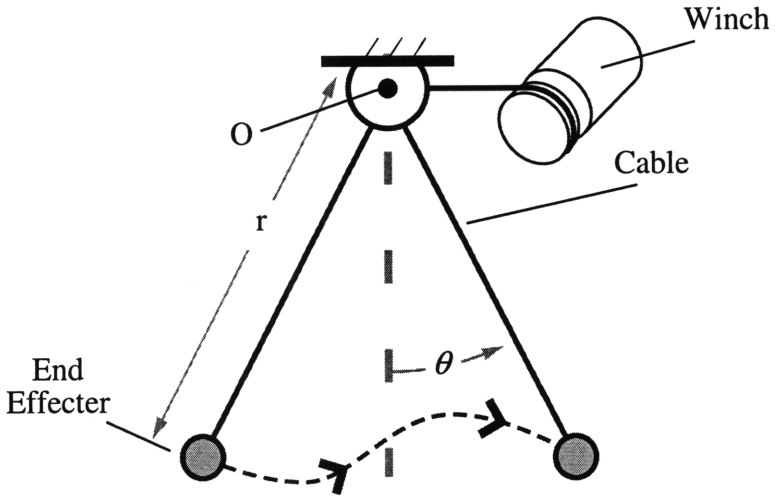

Fig. 1. Shown are the parameters important in our simple pendulum whose length is controlled by a smart winch.

In the following, the basic principle and design concept of the Winch-Bot under-actuated system will be presented, followed by dynamic modeling and trajectory generation algorithms.

Chapter 2. PRINCIPLE

2.1. Parametric Excitation

Consider a two-dimensional pendulum system within a fixed vertical plane. The system consists of a single-axis winch fixed to a ceiling or any fixed point above a workspace, as shown in Fig. 1. A cable hangs down into the workspace, and at the end of the cable is attached the end-effecter, whether it be a basket, a hook, a sensor platform, or any fixed volume with a concentrated mass. Let r be the length of the cable, i.e. the distance between the end-effecter and the fixed point O at the ceiling, and θ be the angle of the cable measured from the vertical line. The length of the cable, r, is actively controlled. No other means of actuation is needed for transporting the end-effecter in the two-dimensional space. Hence the system is under-actuated.

When the cable length is fixed, the system is a simple pendulum oscillating at a specific amplitude and frequency. Now as the winch varies its cable length dynamically in relation to the cable angle θ, the oscillation amplitude is increased or decreased depending on the "phase" between the two time profiles for θ and r, a well-known dynamic phenomenon, "parametric self-excited vibration". Imagine a child on a playground swing set. The child can "pump" his/her legs back and forth in time with the swinging. In this way the thrust the child is using to pull his/her legs back and forth is being transferred to the motion of the swing. This is the same principle of our system,

where the energy used to change the length of the cable in time with the swinging motion, is increasing the amplitude of the swing. As will be discussed later, the change in amplitude can be synthesized more deliberately than just for swinging, so that a desired end-effecter trajectory can be generated.

With only a single actuator, only in-plane motion is affected through parametric self-excitation. With the addition of another actuator, out-of-plane motions can be controlled. Several ways to achieve these out-of-plane motions include introducing a rotation axis to the fixed winch, or by placing the wince on a sliding rail. By driving the rotation or out-of-plane motion as a function of the in-plane swing motion, motion perpendicular to the swing plane can be induced. Moving the out-of plane motion will induce Coriolis forces in the end mass, so much more research into its control would be required, but is not covered here. Examples of this implementation are shown in Fig. 2.

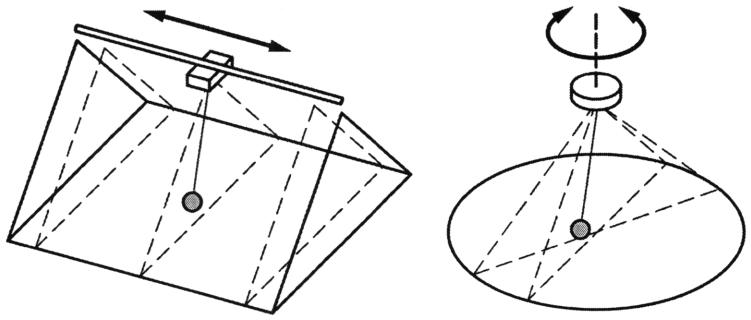

Fig. 2. Two ideas for controlling out-of-plane motion of the swing include introducing a linear or a rotary axis.

2.2. Motion Tasks

This Winch-Bot can generate a variety of controlled oscillatory movements for transporting an end-effecter in a large workspace. In contrast to traditional gantry robots

and long-arm manipulators, this Winch-Bot is much more compact and simple, yet applicable to a variety of tasks.

Two types of transportation tasks appropriate for the Winch-Bot are illustrated in Fig. 3 and Fig. 4.

2.2.1. POINT-TO-POINT MOTION

In one common task, the end-effecter must be moved from a specified start point in space to another specified point. We call this Point-to-Point motion.

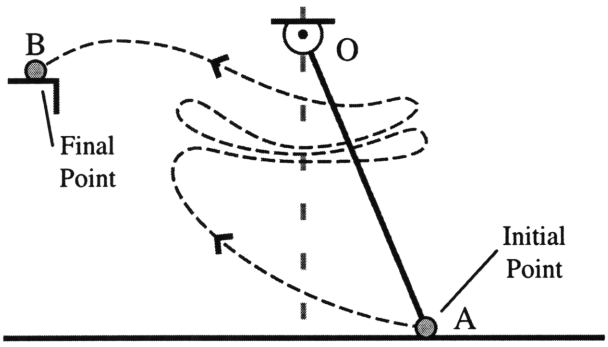

Fig. 3. The most common endpoint control scenario is the lift, leap, and land. Here, only the initial and final endpoints are specified.

Fig. 3 shows a typical trajectory of the end-effecter leaping from initial point A to destination point B. As long as the initial point A is not *exactly* beneath the winch, point O, but at a point offset from it, the end-effecter begins to oscillate as soon as it is lifted by the winch. Accommodating the oscillation amplitude by means of the parametric self-excitation, the Winch-Bot can accumulate enough energy to swing and reach the specified final point. The trajectory may be elaborated so that the end-effecter may land on the destination at zero velocity, i.e. a soft landing.

2.2.2. CONTINUOUS PATH MOTION

Fig. 4 illustrates another class of tasks: continuous path (CP) motion. In the figure, the end-effecter carrying an inspection instrument is moving along a wide surface, an aircraft body, for example. The task is to move the end-effecter a few centimeters above the surface at an unspecified velocity. The geometric trajectory is specified and the end-effecter must keep the distance from the surface. It must be noted, however, that the speed of the end-effecter along the trajectory cannot be specified due to the underactuated, self-exciting nature of the system dynamics. There are many tasks where speed control is not a stringent requirement, but *geometric* trajectories are. For a class of trajectories, there exists an input pattern to the winch control system that drives the end-effecter to follow the specified trajectory.

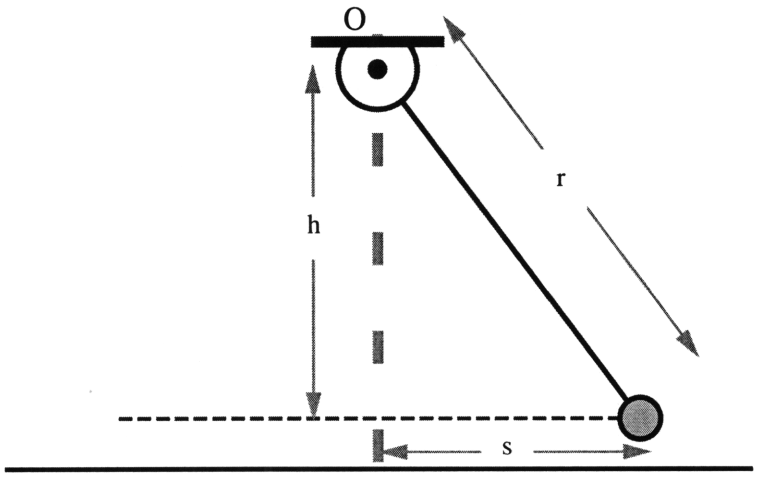

Fig. 4. Continuous path control is useful when the tracking velocity is unspecified, but an entire trajectory within a range is desired.

Input pattern synthesis is a central issue for this type of underactuated robots. The following sections will describe dynamic modeling and input syntheses for each type of tasks: point-to-point and continuous path.

Chapter 3. System Modeling

3.1. Equations of Motion

The governing dynamic equations of the Winch-Bot are derived in this section. We assume that the end-effecter is a point mass and that the cable is mass-less. We also ignore the longitudinal elasticity of the cable and aerodynamic effects on the end-effecter and the cable, justified by the fact that when measured, were small in comparison to the other forces and velocities.

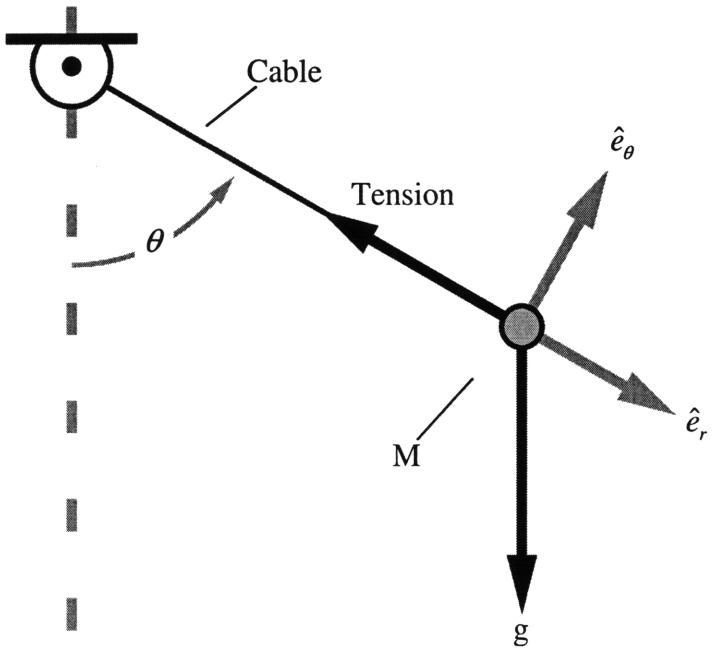

Fig. 5. The well-known free-body diagram of a point on a rigid cable is relevant to our system's design.

Starting with the basic free-body diagram for a point mass, one can derive the two equations of motion for our fixed point/variable-length cable system, moving only in a

single plane. Let m be the mass of the end-effecter and T be the tension of the cable. See Fig. 5. Assuming that the cable is taut, we can obtain the following equations of motion:

$$(mg\cos\theta - T)\hat{e}_r - (mg\sin\theta)\hat{e}_\theta = m\frac{d^2}{dt^2}(r\hat{e}_r) \tag{3.1}$$

where $\hat{e}_r$ and $\hat{e}_\theta$ are unit vectors pointing in the radial and tangential directions, respectively. Expanding the right-hand side and taking a time derivative, we get

$$(mg\cos\theta - T)\hat{e}_r - (mg\sin\theta)\hat{e}_\theta = m\left(\ddot{r} - r\dot{\theta}^2\right)\hat{e}_r + m\left(2\dot{r}\dot{\theta} + r\ddot{\theta}\right)\hat{e}_\theta. \tag{3.2}$$

From the $\hat{e}_r$ direction we get

$$g\cos\theta - \frac{T}{m} = \ddot{r} - r\dot{\theta}^2, \tag{3.3}$$

and from the $\hat{e}_\theta$ direction we get

$$-g\sin\theta = 2\dot{r}\dot{\theta} + r\ddot{\theta} \tag{3.4}$$

or

$$\ddot{\theta} + \frac{2\dot{r}}{r}\dot{\theta} + \frac{g\sin\theta}{r} = 0. \tag{3.5}$$

Equation (3.5) is our main dynamic equation to be used for control synthesis. Equation (3.3) is needed for two purposes:

- To check whether a trajectory under consideration violates the actuator torque limit or not, and
- To ensure that the tension is always positive, meaning the cable stays taut.

When the cable length r is varied periodically, the small-angle oscillatory behavior of the Winch-Bot can be described with a Mathieu Equation, which has the form

$$\ddot{\theta} + \gamma\dot{\theta} + \left(\omega_0^2 + \alpha\cos(2\pi ft)\right)\theta = 0 \tag{3.6}$$

and whose properties have been well documented. We will use the known properties of the Mathieu Equation for trajectory planning, in particular, for accommodating the swing amplitude and energy level. However, for more general trajectory synthesis we will explore a broader class of input patterns based on the full dynamic equations, (3.3) and (3.5).

Chapter 4. POINT-TO-POINT TRAJECTORY CONTROL

4.1. Parametric Formulation

One general idea for this type of system is that of lift, swing, and land, all in a single vertical plane. In order to generate a valid trajectory to get the endpoint from one arbitrary location to another, we must describe the problem in such a way that a solution can be systematically found. Optimal control is a way of rigorously obtaining a valid trajectory. However, it often ends up with an impractically complex numerical method and heavy computational load. The following three-step algorithm following the lift-swing-land sequence provides us with a practical, yet efficient solution method utilizing a bidirectional planning algorithm first demonstrated by Nakamura et al [5].

4.1.1. LIFT

The first segment, the Lift, consists of the dynamic motion between an initial state and a higher-energy state with a shorter string, through the shortening of the cable. See Fig. 6. In general, we would like to make this lift trajectory to be as smooth as possible, to prevent jostling the end mass, which would cause unwanted, uncontrollable oscillations. Because of the smooth nature (and its computationally simple

differentiation), we chose to parameterize and use the sigmoid curve for our smooth trajectories.

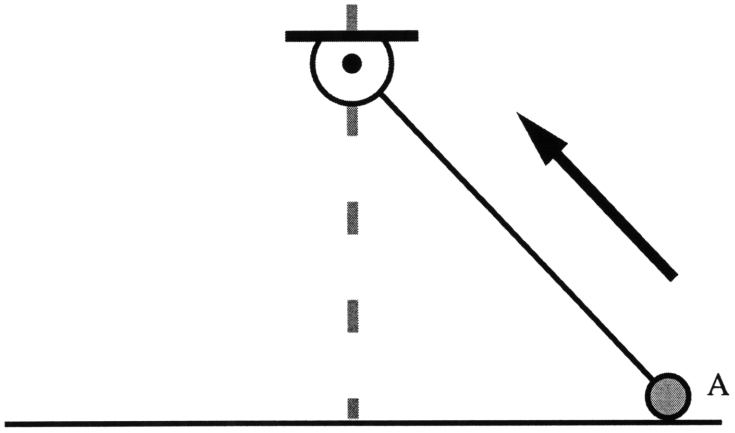

Fig. 6. The first step in the sequence of calculated events is the pull, or where the string is shortened to clear the floor, to have a higher frequency, and to drive up the energy in the swing to a desired state.

4.1.1.1. SIGMOID GENERATION

There are nigh-infinite trajectories connecting the initial and final cable lengths. To facilitate computation, let us parameterize the sigmoid string-length curve with a few simple parameters. As shown in Fig. 7, the sigmoid trajectory generates a smooth pull from the initial point to a higher height. The change of length of the string is Δr_P. If Δr_P previously decided or determined, then the only parameter to be varied in this step is the the amount of time it takes to pull the cable, Δt_P. That makes the adjustment of the parameter Δt_P the only free variable.

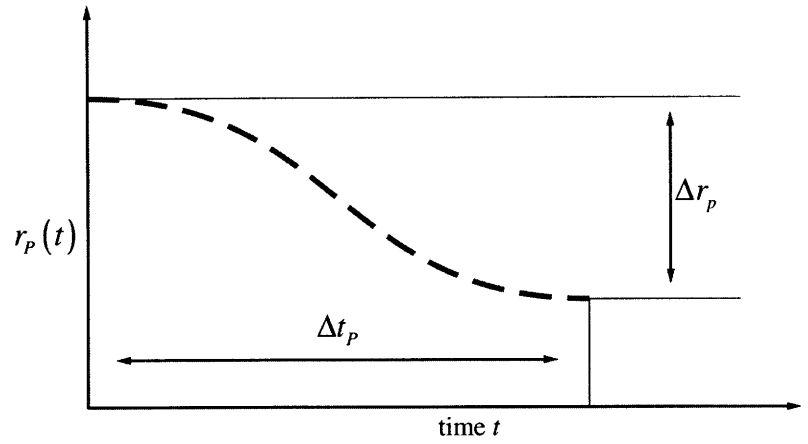

Fig. 7. A smooth sigmoid would be an ideal lift trajectory, as it both accelerates and decelerates smoothly.

Another option with the Lift would be to use a half-sigmoid, as shown in Fig. 8. This shape has the added benefit of a quick initial pull, which in implementation is important for establishing the zero-time location. One can imagine that in order to start a sigmoid trajectory moving at exactly time-zero, there would be no indication as to when the zero occurs, as it is a smooth transistion. With a quick pull, the first moment of motion indicates the zero time.

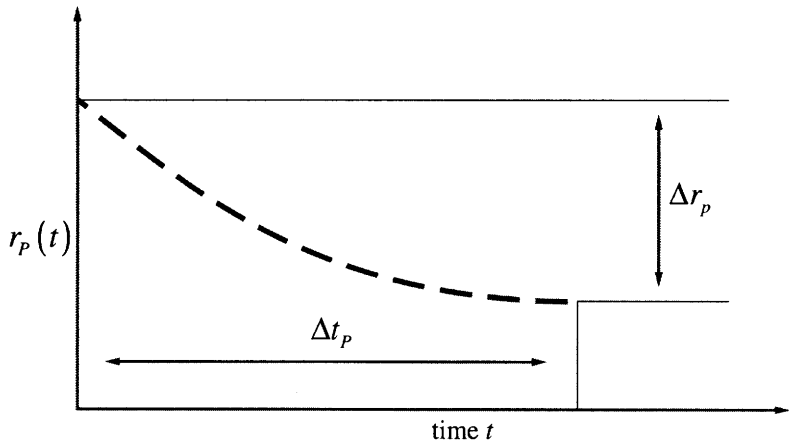

Fig. 8. In implementation, it turns out that a half-sigmoid is much easier to use, as the zero-time is much easier to establish.

4.1.1.2. *Resulting Energy Modulation*

Changing this length of time parameter has the effect of varying the resulting energy in the pendulum's swing cycle. If one imagines pulling a string from one length to a shorter length at a speed that finishes the pull as the pendulum passes through one half-period, the amount of resulting energy is much higher than if the pulling occurred over a full period. Fig. 9 shows two overlaid examples of starting with the same initial conditions, pulling the same Δr_P, how varying the time parameter Δt_P affects maximum amplitude of the final swing. Fig. 10 shows the range that the effect the time parameter has on the final energy of a system going from one initial point and pulled through a specific Δr_P. In implementation, this time parameter Δt_P is varied to best match the energy at the beginning of the Landing segment. This technique is similar to the principle of Time Scaling used by Arai et al [7].

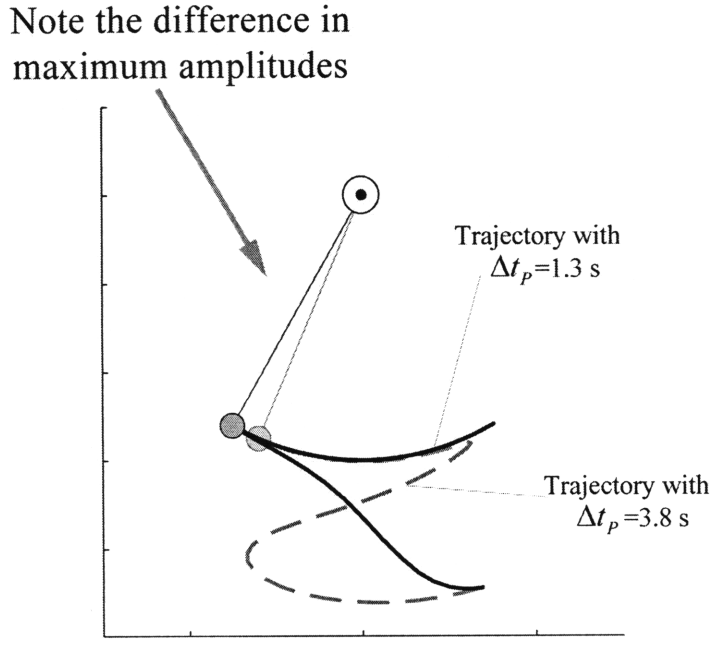

Fig. 9. How quickly one decreases the length of the string affects the final maximum amplitude, and thus the energy, in the eventual stable swing.

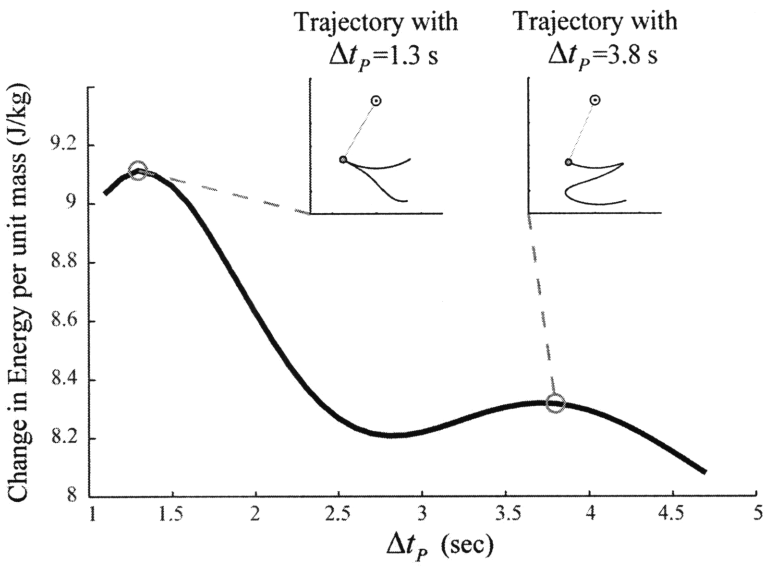

Fig. 10. This is the resulting plot of a pendulum with a starting length of 2.33 m and a Δr_P of 0.83 m as the string length change Δt_P is varied. The insert plots show the pendulum trajectories shown in Fig. 9.

4.1.2. SWING

The second segment, or Swing, is necessary if the energy in the pendulum at the end of the Lift does not match the energy required for the final landing. This could be caused by errors in measurement, or only because the desired energy cannot be achieved with the Lift alone. In this segment the energy within the pendulum's swing is adjusted using parametric oscillation to generate self-excitation (or damping). These parametric oscillations are calculated utilizing the general solutions to the Mathieu Equation.

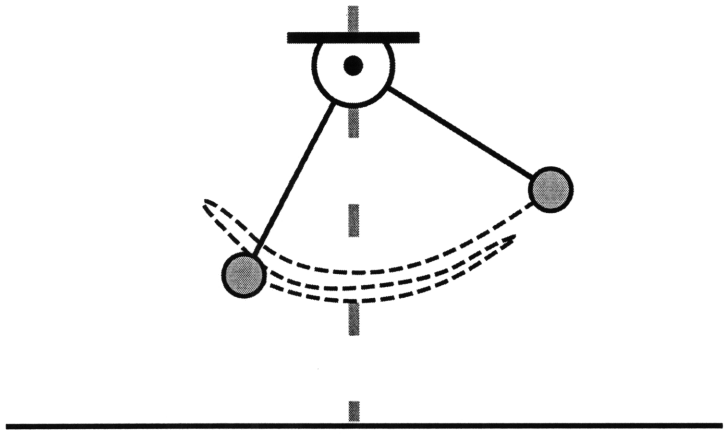

Fig. 11. The second step in the sequence is not always needed. When required, this step uses parametric self-excitation to adjust the energy within the swing.

4.1.2.1. PARAMETRIC EXCITATION

By making small changes to the string's length locked in phase with the swinging of the pendulum, energy can be pumped into the system, as discussed in Section 2.1 on page 19. The amplitude of the small changes to the length can be made to be proportional to the difference between the desired and the current swing amplitudes, or the amplitude error. Thus the amplitude can be made the output to a closed-loop system, like in Fig. 12. This results in the speed of excitation to be proportional to the error in amplitude.

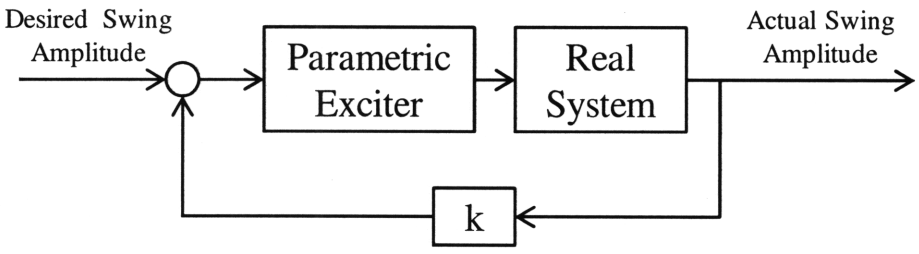

Fig. 12. This sample closed-loop controller could be used to regulate the maximum amplitude of a periodic swing by driving the parametric excitation proportional to the error in amplitude.

4.1.2.1.1. MATHIEU EQUATION

As described, our dynamic equation closely resembles Mathieu Equation,

$$\ddot{\theta} + \gamma\dot{\theta} + \left(\omega_0^2 + \alpha\cos\left(2\pi ft\right)\right)\theta = 0.$$

By linearizing our dynamic equation (3.5),

$$\ddot{\theta} + \frac{2\dot{r}}{r}\dot{\theta} + \frac{g}{r}\theta = 0,$$

we see that the structure of our equation is the same as Mathieu's, namely

$$\gamma = \frac{2\dot{r}}{r}$$

and

$$\omega_0^2 + \alpha\cos\left(2\pi ft\right) = \frac{g}{r}$$

As mentioned in Section 3.1, we can then use the well-known solutions to the Mathieu Equation to drive r sinusoidally, and thus excite our system to higher energies.

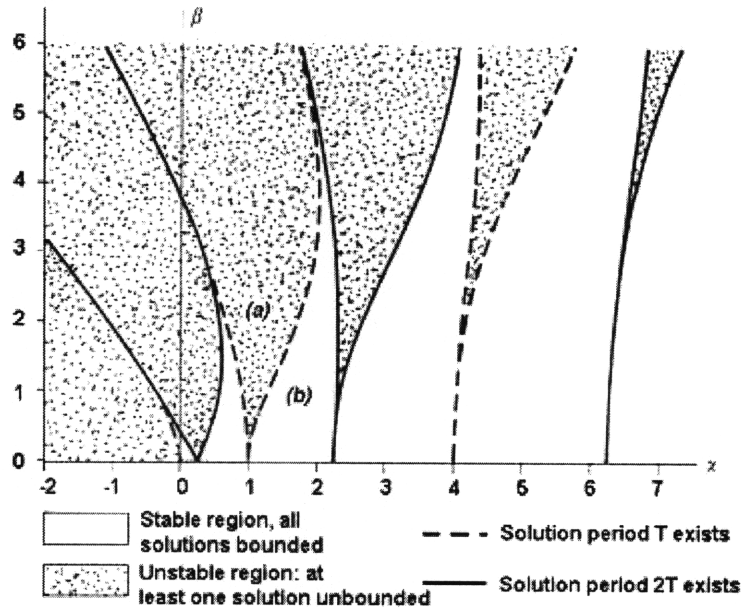

Fig. 13. The Mathieu Equation has both unstable and stable regions. Here we typically drove the excitation at resonance. [6]

4.1.2.1.2. NONLINEAR EFFECTS

Because the amplitudes with which we deal are generally large enough that the linear approximation fails, we cannot parametrically drive our system at a single frequency and expect to stay at resonance. This is because the resonance point, or eigenfrequency, is a nonlinear function of the amplitude. Because the eigenfrequency shifts, if you simulate parametrically driving our system at a single frequency, peaks and troughs in the amplitude appear, as shown in Fig. 14.

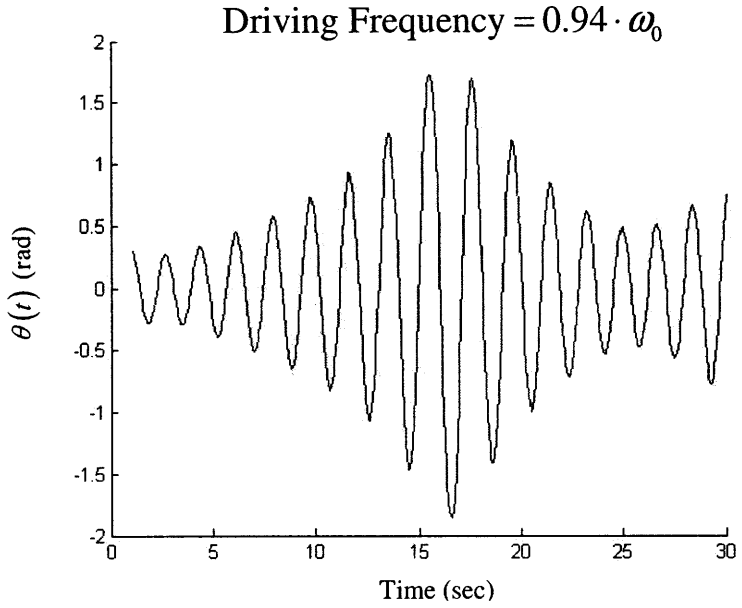

Fig. 14. Nonlinear eigenfrequency shift causes the swing angle to reach a maximum when parametrically exciting at a constant frequency.

4.1.3. LAND

The third and final segment is Land, which can be thought of as the Lift step in reverse time. In forward time it starts from the trajectory achieved by either the Lift or the Swing, and smoothly extends out to the final point, again parameterized as a sigmoid curve. In most cases, the final point has zero velocity, so it is obvious how this is

basically a reverse-time version of the Lift segment; instead of going from a fixed point with no velocity to periodic pendulum motion, the system goes from a periodic pendulum motion to a fixed point with no velocity. Not surprisingly, this step is solved mathematically just as one would expect: set the final point as initial conditions, and solve from time $t=0$ to $t=-\infty$. (In reality, going backward, the system reaches the desired energy rapidly, so it is unnecessary to solve for t of large negative values.)

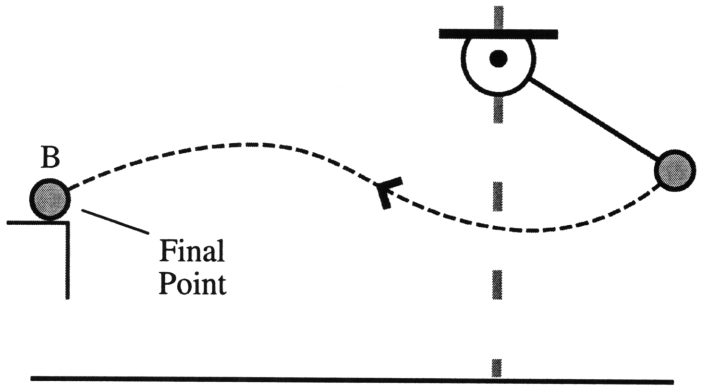

Fig. 15. The third step takes the system from a stable swing cycle, lengthening the string to softly land with a zero (or arbitrary) velocity at the endpoint.

In total, our strategy is to pick an arbitrary middle height (from where the Land starts the Lift ends), then to calculate the Land in reverse time. This solution gives us the energy required from the first two steps. From there we take the initial conditions, adjust the pull time so that the Lift reaches the desired middle energy. If this is impossible, the Swing step will be necessary, and controlled in real-time.

In the case of the endpoint being at rest, the initial conditions can be measured by the system itself. As long as the cable is not slack, the measured angle of the cable combined with the current length fully defines the location of the endpoint. This is most useful when the endpoint is a tool holder and can be placed in a constant and fixed location.

Other situations may arise where the initial conditions are not at rest, such as in a stable oscillatory trajectory. If that is known, the system can use that as initial conditions, because it can measure the progress around the oscillations and trigger when those measurements meet certain criteria.

Specifying the final conditions includes more than just specifying a final endpoint location with zero velocity. If this system were used to apply force at a specific location, like for "hammering" in a series of nails or impact-testing at multiple points, one can specify a final location and a non-zero end velocity, which would translate to a force dependent on the mass.

4.1.3.1. REVERSE-TIME

One complication coming from solving for the Lift by starting at $t=0$ going backwards towards $t=-\infty$ instead of replacing t with $t-\tau$ in the dynamic equations, the task of pairing the Lift with the negative Land. One general way to solve this is to use the Swing step to wait until or drive to a specific set of values of the states, and then continue along the generated string length trajectory. The other option, to be used when both the Lift and Land step middle energies exactly match, is to pair a set of matching state values, one in each generated string length trajectory. Then when the state reaches one value, the controller immediately transitions over to the other value and continues.

Chapter 5. CONTINUOUS PATH CONTROL

5.1. Modeling

Continuous path control synthesis is the problem of finding an input pattern for the winch so that the end-effecter can track a given geometric trajectory. Unlike standard continuous path control for fully actuated systems, we cannot specify a time trajectory for the end-effecter since the Winch-Bot is an under-actuated system. When a geometric trajectory is specified, the time trajectory is to be determined from the conditions that dictate the end-effecter to follow the geometric trajectory. Before solving a general CP control synthesis problem let us work out a special case, which is of practical importance.

5.1.1. HORIZONTAL TRAJECTORIES

Consider a horizontal trajectory, as shown before in Fig. 4 on page 22. The end-effecter is required to track this horizontal line at a distance h from the winch center. Let s be the distance along the horizontal line measured from point C, which is directly beneath the winch center. For the end-effecter to move along the horizontal line, the cable length must be varied to satisfy

$$r(s) = \sqrt{s^2 + h^2} \tag{5.1}$$

and the cable angle must be

$$\theta(s) = \tan^{-1}\left(\frac{s}{h}\right). \tag{5.2}$$

When these conditions are perfectly satisfied, then the end-effecter mass m has no acceleration in the vertical direction. Therefore the cable tension is given by

$$T = \frac{mg}{\cos\theta} \tag{5.3}$$

and its component in the horizontal direction given by

$$F_B = -mg\tan\theta = -\frac{mg}{h}s, \tag{5.4}$$

which works as a braking force on the end-effecter as it tends to move away from point C. Therefore, the work done by the winch upon the end-effecter when it moves from point C to a current position at s is given by

$$Work = -\int_0^s F_B ds = \frac{mg}{2h}s^2. \tag{5.5}$$

Suppose that the end-effecter passes point C at a speed V_C, where it has kinetic energy $T = mV_C^2/2$. The end-effecter can move along the line until the winch absorbs all the kinetic energy. Let s_D be the distance to point D where the kinetic energy becomes zero. Equating *Work* and T yields

$$s_D = V_C\sqrt{\frac{h}{g}} \tag{5.6}$$

On the way back from point D, the end-effecter regains the kinetic energy and continues to move the same distance s_D in the opposite direction, if the process is loss-less. Therefore, the Winch-Bot can perform the continuous pass tracking of the horizontal trajectory of length $2s_D$. The tracking speed for the forward motion is given by

$$V(s) = \sqrt{V_C^2 - \frac{g}{h}s^2}. \tag{5.7}$$

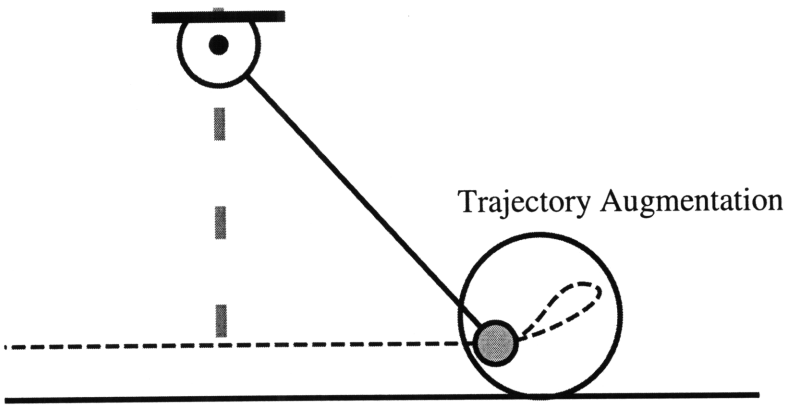

Fig. 16. Using techniques similar to those detailed in the "Swing" step in the Point to Point control scheme, the ends of each controlled path can be augmented to replace any losses with excitation.

To accelerate the end-effecter so that it possesses the necessary velocity V_C at point C, the Winch-Bot will need to swing the end-effecter a few times. We can apply the same technique described in the previous section to this initial acceleration.

The true system has some friction and energy dissipative component. Therefore, the distance of horizontal motion, $2s_D$, decreases if the same horizontal trajectory tracking must be repeated many times. To resolve this diminishing distance problem, energy must be pumped into the end-effecter. As shown in Fig. 16, we can increase the energy level by using the same techniques described in the previous section to augment the trajectory.

5.1.2. GENERAL CASE

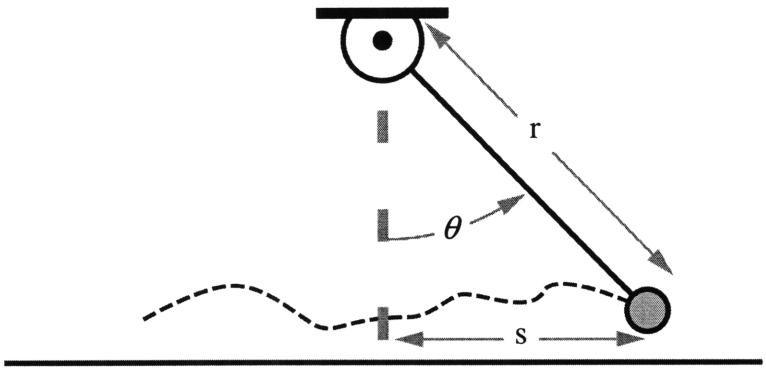

Fig. 17. The general case for CP control makes the variable s arbitrary, and both and r are functions of that variable s.

Based on the insight gained from the above special case, let us now formulate a CP control synthesis problem for general trajectories. This formulation is similar to that shown by Dahl and Nielsen [8]. To facilitate to describe a geometric trajectory, consider a path length s along a given trajectory. See Fig. 17. Given a geometric trajectory, the cable length and the cable angle must satisfy functional relationships so that the end-effecter may track the trajectory:

$$r = r(s), \quad \theta = \theta(s). \tag{5.8}$$

The tracking speed is given by

$$\dot{s} = \frac{ds}{dt} \tag{5.9}$$

and the time derivatives of r and θ are given by

$$\dot{r} = \frac{dr}{ds}\frac{ds}{dt} = r'\dot{s}, \quad \dot{\theta} = \frac{d\theta}{ds}\frac{ds}{dt} = \theta'\dot{s}. \tag{5.10}$$

Furthermore, the second derivative of θ is given by

$$\ddot{\theta} = \theta''\dot{s}^2 + \theta'\ddot{s}. \tag{5.11}$$

Substituting these into (3.5), our main dynamic equation, yields

$$\ddot{s} + A(s)\dot{s}^2 + B(s) = 0, \tag{5.12}$$

where

$$A(s) = \frac{\theta''}{\theta'} + \frac{2r'}{r} \tag{5.13}$$

and

$$B(s) = \frac{g \sin\theta}{r\theta'}. \tag{5.14}$$

Given a geometric trajectory and initial conditions, Eq. (5.12) can be solved numerically for time function $s(t)$. Once the time profile of path length $s(t)$ is obtained, the cable length and the cable angle as well as their time derivatives can be obtained. Based on these results, we examine whether the solution is eligible. Typically we have to check the following items:

- The solution $s(t)$ covers the entire geometric trajectory.
- The tracking speed $\dot{s}(t)$ is within an acceptable range.
- The cable tension T is always positive, i.e. no slack of the cable.
- The cable tension T and the winch speed is within the maximum actuator parameters.

These conditions depend on the initial conditions of the end-effecter. To find an acceptable solution, the above numerical procedure may need to be repeated for varied initial conditions. Note that the initial conditions can be tuned by augmenting the trajectory and applying the same techniques described previously.

5.1.3. PROPERTIES OF THE CP PROBLEM

In order to prove the properties of the Continuous Path problem, we must start by formally defining certain terms.

Firstly, we use the term *path* to define a geometric, time-invariant curve in the r,θ state space, as opposed to a time-varying progression of those states, which we call a *trajectory*. Secondly, a system is said to *trace* a path if the endpoint is on some geometric curve defined by $r=f(\theta)$ or by a set of parametric equations $r=f(s)$, $\theta=f(s)$ for all time. Finally, a geometric path is considered *strictly-positive traceable* if there exists a time-trajectory of the states that can trace the given path where the tension T remains strictly positive.

Proposition:

We can fully define our Winch-Bot system with the following dynamic equations first derived in Section 3.1:

$$T=-m\left(\ddot{r}-r\dot{\theta}^2-g\cos\theta\right) \tag{5.15}$$

$$\ddot{\theta}=-\frac{2\dot{r}}{r}\dot{\theta}-\frac{g\sin\theta}{r}. \tag{5.16}$$

Given a geometric path, a winch-bot system governed by the above equations of motion has a repetitive, periodic trajectory (which depends on the initial velocity alone) that perfectly traces that path while performing zero net-work on the system after each period. Additionally, this trajectory can be implemented on the winch-bot as long as the geometric path is strictly-positive traceable. In other words, the total work done by the winch in one part of the swing is cancelled by the work done on the winch in another, which results in a cyclic trajectory that neither diverges nor converges. The strictly-positive traceability constraint is required due to the cable or string only supporting a tension.

Proof:

Firstly, we can define the geometric path r as a function of θ alone. We can then write the time derivatives as

$$r = f(\theta),\ \dot{r} = f'\dot{\theta},\ \ddot{r} = f''\dot{\theta}^2 + f'\ddot{\theta}. \tag{5.17}$$

By substituting this result into the first dynamic equation (5.15), we find

$$T = -m\left(f''\dot{\theta}^2 + f'\ddot{\theta} - f\dot{\theta}^2 - g\cos\theta\right), \tag{5.18}$$

and then substituting the value of $\ddot{\theta}$ found from (5.16), we find

$$T = -m\left[\left(f'' - f\right)\dot{\theta}^2 - g\cos\theta - \frac{f'}{f}\left(2\dot{r}\dot{\theta} + g\sin\theta\right)\right] \tag{5.19}$$

and again substituting for time derivatives it becomes

$$T = -m\left[\left(f'' - f\right)\dot{\theta}^2 - g\cos\theta - \frac{f'}{f}\left(2f'\dot{\theta}^2 + g\sin\theta\right)\right]. \tag{5.20}$$

Finally, by rearranging to a cleaner form, we arrive at our equation for the tension:

$$\frac{T}{m} = \left(f + \frac{2f'^2}{f} - f''\right)\dot{\theta}^2 + g\cos\theta + \frac{f'}{f}g\sin\theta. \tag{5.21}$$

By solving for the work done on the system by the winch, we can find the complete energy of the system. Here we can define work as the force times a distance, in this case there are two: the change in string length r times the string tension, and the change in θ times the force of gravity. See Fig. 18. Because we know that the second term results in a potential energy, as we are returning to the same point we know the net change in potential energy is zero, so we can remove that term. So because the change in θ is perpendicular to the tension, the only remaining term is the tension and the change is string length.

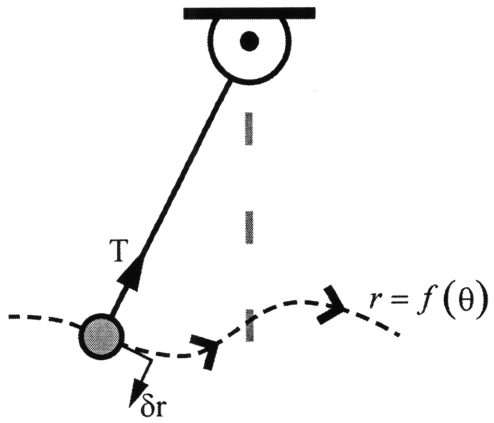

Fig. 18. The only net work over a cycle would be performed by the Tension because the gravity will only change the potential energy of the system.

The complete differential work term can be written as

$$\delta Work = -T\delta r\,, \tag{5.22}$$

and around one complete cycle we can perform a closed-loop integration to find

$$Work = -\oint T\left(\theta,\dot{\theta}\right)dr\,. \tag{5.23}$$

Substituting the results from Eq. (5.21) and $dr = f'd\theta$, we write work as

$$Work = -\oint\left[f'\left(f+\frac{2f'^2}{f}-f''\right)m\dot{\theta}^2+mgf'\cos\theta+\frac{f'^2}{f}mg\sin\theta\right]d\theta\,. \tag{5.24}$$

This can be abbreviated as

$$Work = -\oint\left[A(\theta)\dot{\theta}^2+B(\theta)\cos\theta+C(\theta)\sin\theta\right]d\theta \tag{5.25}$$

where

$$A = f'\left(f+\frac{2f'^2}{f}-f''\right)m\,,\ B = mgf'\,,\ C = \frac{f'^2}{f}mg\,. \tag{5.26}$$

This loop integral starts from a θ_1 and monotonically increases to θ_2, then reverses back monotonically to θ_1. Because of this, we can separate the loop integral into two line integrals, namely

$$-Work = \int_{\theta_1}^{\theta_2} A(\theta)\dot{\theta}^2 d\theta + \int_{\theta_1}^{\theta_2} B(\theta)\cos\theta d\theta + \int_{\theta_1}^{\theta_2} C(\theta)\sin\theta d\theta$$
$$+\int_{\theta_2}^{\theta_1} A(\theta)\dot{\theta}^2 d\theta + \int_{\theta_2}^{\theta_1} B(\theta)\cos\theta d\theta + \int_{\theta_2}^{\theta_1} C(\theta)\sin\theta d\theta \quad (5.27)$$

By swapping the limits of integration (and negating the $\dot{\theta}$ term), we can rewrite it as

$$-Work = \int_{\theta_1}^{\theta_2} A\dot{\theta}^2 d\theta - \int_{\theta_1}^{\theta_2} A\left(-\dot{\theta}\right)^2 d\theta + \int_{\theta_1}^{\theta_2} B\cos\theta d\theta - \int_{\theta_1}^{\theta_2} B\cos\theta d\theta$$
$$+\int_{\theta_1}^{\theta_2} C\sin\theta d\theta - \int_{\theta_1}^{\theta_2} C\sin\theta d\theta \quad (5.28)$$

which by cancelling terms can be reduced to

$$Work_{loop} = 0,$$

Q.E.D.

5.1.4. CLOSING THE LOOP

If we decide on a certain controller with a known functionality, it can be insightful to look at the closed-loop system of the pendulum and the winch being controlled by some function in order to perform our CP tracking. We will first look at the specific case of the continuous path tracking of a horizontal trajectory that we derived in section 5.1.1 on page 37.

5.1.4.1. HORIZONTAL TRAJECTORIES

For tracking a straight-line geometric trajectory at distance h from the fixed point, like in Fig. 4, one can relate the length of the string r simply as a function only of the angle θ. For this case, we know

$$r(\theta) = \frac{h}{\cos\theta} = h\sec\theta \quad (5.29)$$

and by taking the time derivative, we also know

$$\frac{dr}{dt} = \frac{dr}{d\theta}\frac{d\theta}{dt} = h\sec\theta\tan\theta\frac{d\theta}{dt}. \tag{5.30}$$

By plugging the results of Eq. (5.29) and Eq. (5.30) into our second dynamic equation (5.16), we arrive at an equation for the dynamics of our closed-loop system,

$$\ddot{\theta} + 2\dot{\theta}^2\tan\theta + \frac{g}{h}\sin\theta\cos\theta = 0. \tag{5.31}$$

Given initial conditions, this result can be solved uniquely for the angle θ and its derivatives. That solution provides the closed-loop performance of the system.

5.1.4.2. GENERAL CASE

Expanding on our results from the previous section, we can generalize the system to any geometric path defined in the x-y plane as

$$y = f(x) \tag{5.32}$$

that has the derivative

$$\dot{y} = f'(x)\cdot\dot{x} \tag{5.33}$$

(where $f'(x) = df/dx$) by invoking a coordinate transform, namely

$$y = r\cdot\cos\theta,\ x = r\cdot\sin\theta.$$

From there, we can substitute the derivative of x and y into Eq. (5.33) and find

$$\dot{r}\cos\theta - r\dot{\theta}\sin\theta = f'(x)\left[\dot{r}\sin\theta + r\dot{\theta}\cos\theta\right]. \tag{5.34}$$

Rearranging, we arrive at our controller equation,

$$\dot{r} = \frac{-f'(x)\cos\theta - \sin\theta}{f'(x)\sin\theta - \cos\theta}r\dot{\theta}. \tag{5.35}$$

This form is one such that if we know the complete r,θ state we can know the desired $\dot{r}$ that will cause the system to follow a geometric path with slope matching that of the desired trajectory. This can also be looked at as

$$\dot{r} = \mathbf{J}\dot{\theta}, \tag{5.36}$$

which we can view at as a pseudo-Jacobian controller. As discussed before, this control law defines $\dot{r}$ as a function of θ and its derivatives, but it does not uniquely define r. To do this, we can think of the entire set of states, both r and θ, as lying within a flow field similar to that shown in Fig. 19(a). In order to make the trajectory converge to a line with not only the correct slope, but also the correct absolute geometric path, we need a term in the $\dot{r}$ value that pushes all paths toward the correct path, shown in Fig. 19(b). We do this by implementing a restoring term into (5.36), shown as

$$\dot{r} = \mathbf{J}\dot{\theta} + K\Delta r, \tag{5.37}$$

where K is a negative constant. In this way, this can be shown to converge to tracing the given path from any initial starting conditions.

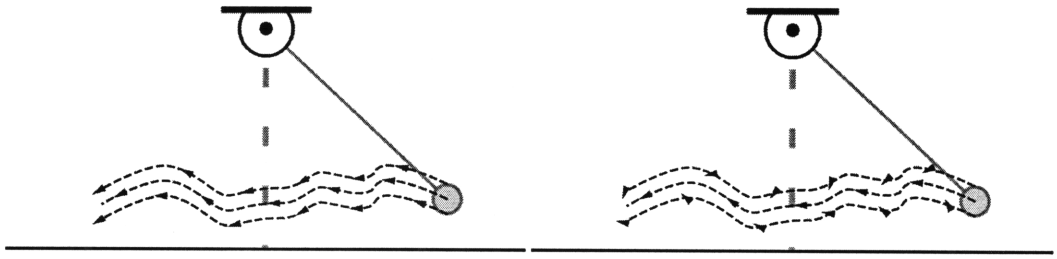

Fig. 19(a) Fig. 19(b)

Fig. 19. Shown is the difference when an absolute restoring term Δr is prescribed into the $\dot{r}$ equation.

This result was simulated in MATLAB using the ODE45 solver with the state vector defined as $\begin{bmatrix}\theta & \dot{\theta} & r\end{bmatrix}^T$. The state equation to be solved was a variation of our main dynamic equation ($\ddot{\theta}+\frac{2\dot{r}}{r}\dot{\theta}+\frac{g\sin\theta}{r}=0$), with the additional state variable, or

$$\begin{bmatrix}\dot{\theta}\\ \ddot{\theta}\\ \dot{r}\end{bmatrix}=\begin{bmatrix}\dot{\theta}\\ -\frac{2}{r}A\cdot\dot{\theta}-\frac{g\sin\theta}{r}\\ A\end{bmatrix} \tag{5.38}$$

where

$$A=\dot{r}=\frac{-f'(x)\cos\theta-\sin\theta}{f'(x)\sin\theta-\cos\theta}r\dot{\theta}+K\left(r_{desired}-r\right), \tag{5.39}$$

found in Eq. (5.35) and Eq. (5.37). Here we define only the values of $r_{desired}$ as a function of θ, and $f'(x)$ as a function of both θ and r. Solving to follow the simple geometry of a straight line a distance 2 below the zero point, we can easily see that the system with an initial angle of zero, positive initial velocity, and initial length of 3 converges to the steady line-following pattern, as shown in Fig. 20. The angle and length responses are shown in Fig. 21.

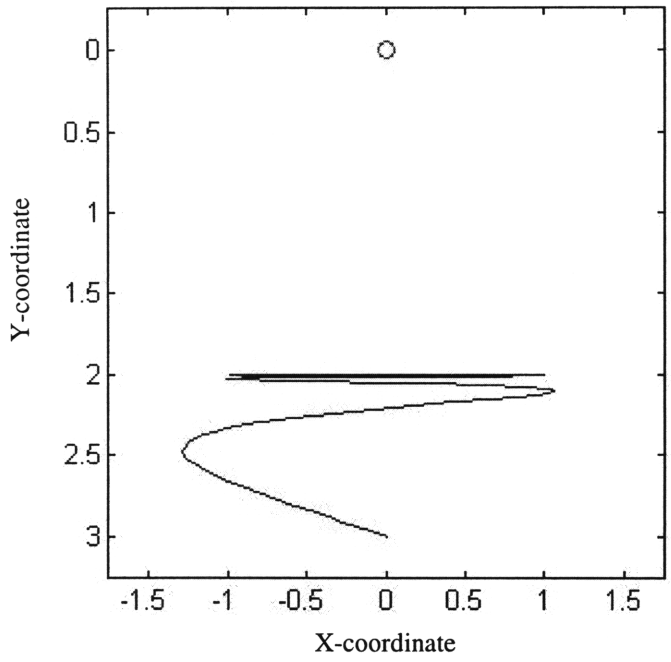

Fig. 20. Starting from an initial conditions of zero-angle, nonzero-velocity, length 3, and driven to follow a geometric line at length 2, the system converges quickly.

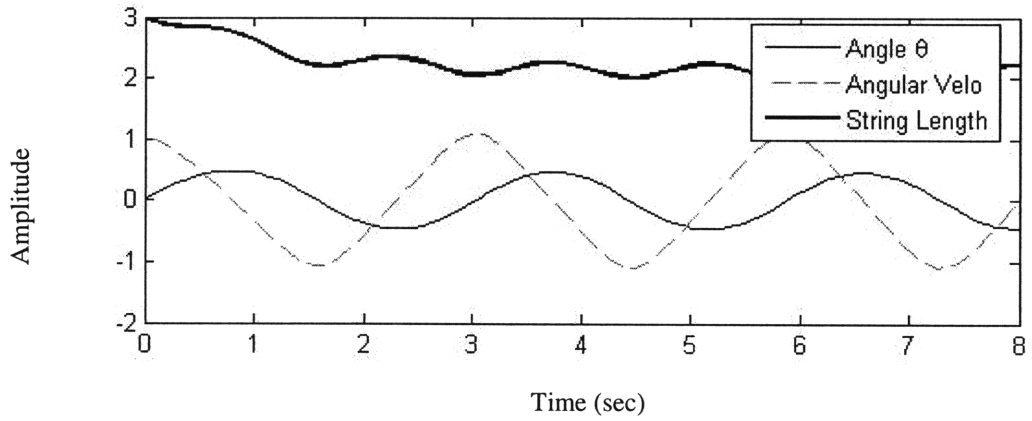

Fig. 21. The responses quickly converge to periodic motion on the prescribed geometric path.

Even with this simple pseudo-Jacobian controller with a correction term, trajectories converge to the desired geometric path. But what geometric paths $y = f(x)$ are valid for tracking? How can we determine the validity of these paths?

5.1.5. PATH VALIDATION

Equation (3.3), the first dynamic equation derived from the free-body diagram of a point in section 3.1, includes the states of our system as well as the tension term T. We explained that the equation was used mainly for path validation, namely to ensure that $T > 0$ for all time (for the equations of motion to remain valid), and that $T < T_{max}$ defined by the physical actuator. The computation of Eq. (3.3) requires the assumption that these criteria are true. But is there a way, given a certain geometric path, to determine if the path is valid before computation?

At any point in time, one can fully define the state of a point mass in space by knowing its position and velocity. The position depends solely on θ and r, and its velocity on $\dot{\theta}$, $\dot{r}$. If we define r and $\dot{r}$ as our inputs, the state can be defined with only θ and $\dot{\theta}$. Knowing this, we understand that the only accelerations applied to our point are along two directions – one straight down due to gravity, and the other in a direction pointing toward the fixed pivot, which depends on θ. Because the T vector (in the direction of the fixed pivot) is a function of our input r and its derivatives, we can call T a *pseudo-input*, and its amplitude can be adjusted as we choose, within the range of $0 < T < T_{max}$. Because the total acceleration vector applied to the mass is the vector sum of the two acceleration terms, we have some control over what direction it points, *within some triangle of actuation*. This provides a limit to the geometric paths we can track.

Given a geometric path, we can define r solely as a function of θ. First, we can define the first and second time-derivatives of r as

$$\frac{dr}{dt} = \frac{dr}{d\theta} \cdot \frac{d\theta}{dt}, \tag{5.40}$$

and

$$\frac{d^2r}{dt^2} = \frac{d}{dt}\left(\frac{dr}{d\theta} \cdot \frac{d\theta}{dt}\right) = \frac{dr}{d\theta} \cdot \frac{d^2\theta}{dt^2} + \frac{d^2r}{d\theta^2} \cdot \frac{d\theta}{dt}. \tag{5.41}$$

Then by defining

$$r' \equiv \frac{dr}{d\theta}, \; r'' \equiv \frac{d^2r}{d\theta^2} \tag{5.42}$$

and by defining the inequality $T > 0$, we can substitute into and rearrange Equation (3.3) to our desired form:

$$-r'\ddot{\theta} + r\dot{\theta}^2 - r''\dot{\theta} + g\cos(\theta) > 0. \tag{5.43}$$

Each of the components of this equation plays a role in determining the outcome of the inequality. In order to discover a generalized space of valid paths, let us consider a fixed, swinging pendulum. That pendulum has a set of time trajectories for θ and its derivatives. Given that set of time-trajectories, what can we do with the first and second spacial derivatives of r, or Eq. (5.42)? By simulating a range of these variables in MATLAB, the results are shown in Fig. 22.

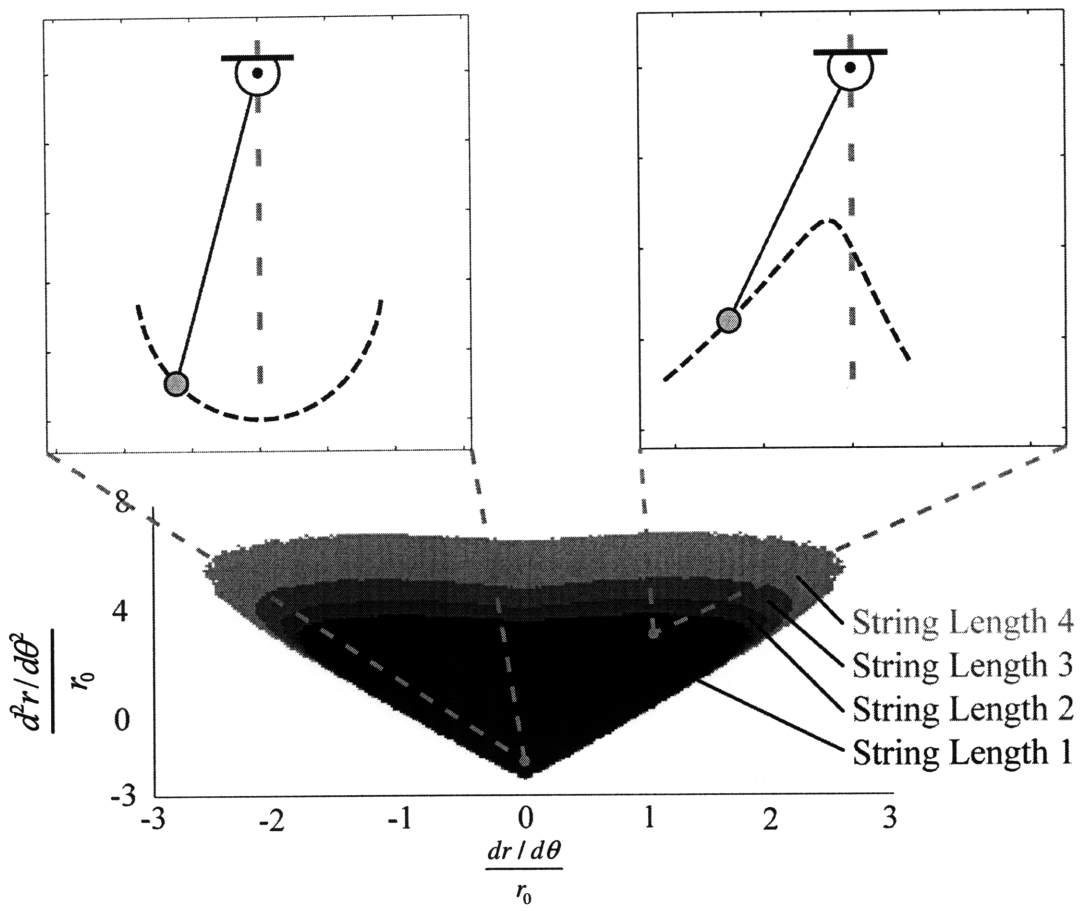

Fig. 22. Shown are the simulation results of a string-tension validity test for a system of multiple string lengths and nonzero velocity / zero angle initial conditions, following a path with varying curvature parameters. Insert graphs show sample geometric trajectories at specific points. Values of derivatives were normalized with respect to an initial string length r_0.

By determining the maximum $\frac{dr/d\theta}{r_0}$ and $\frac{d^2r/d\theta^2}{r_0}$ of a certain geometric path within a desired span, the validity of that path can be roughly determined by calculation of a generic differential equation. This is useful to roughly decide the viability when calculating the full differential equation is impossible or infeasible.

Chapter 6. IMPLEMENTATION

To test the system in question, we built a pendulum with a winch-controlled string. The mass at the end was a steel ball encased in plastic, and the string was Kevlar, to reduce the elastic effects of a string. The winch was a coil of the string, driven by a DC motor. We instrumented the motor with a precision encoder that would feed back the string's actual length in real-time. To measure the string's angle we used a light source to cast a shadow of the string onto an optical sensor, which translated the signal to the string's angle relative to the structure. Given that these two parameters were determined with a high enough frequency, we could take the derivatives of the measured values and know the entire state of the system within the plane of desired swing motion. We used a simple closed-loop controller to specify the desired position of the winch, which drove the length of the string to our desired length.

Our implementation is shown in Fig. 23. Included is a rotary motion stage and actuator for further study into the modes of swinging with this additional degree of freedom. In this thesis's primary topic, the trajectory planning algorithms, this degree of freedom was not used. Implementation of the optical angular measuring device required the addition of a light-tight box to eliminate the effect of the environment's ambient light, and will be detailed to follow.

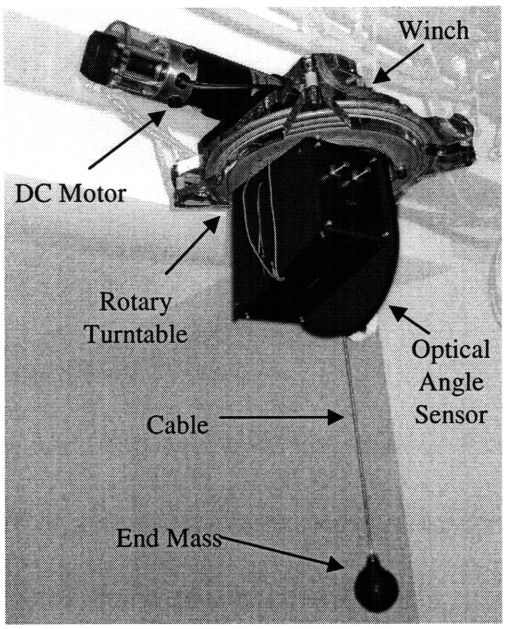

Fig. 23. A light-tight box encasing a single LED was needed to reduce the effects of ambient lighting on the optical sensor.

6.1. Field Programmable Gate Array (FPGA)

In our testing, we used at National Instruments programmable digital I/O and processor, the NI cRIO-9074. The unit contained a host processor capable of running LabVIEW loops at about 100 Hz, and a FPGA with an internal clock running at 40 MHz. Our code was written in LabVIEW v8.6, compiled for use on the FPGA, and transferred to the processor.

Our code was broken up into two interacting blocks – one running at 1 MHz on the FPGA (to be called the FPGA code) and one running at 100 Hz on the cRIO's internal processor (to be called the Host PC code). The FPGA code was written in LabVIEW and compiled to machine-code to run natively on the FPGA at the requested loop frequency. Because of this, all user-generated commands must be entered into the Host PC code and made available to the FPGA code when it requests.

6.2. Measuring Angle

Initial work at measuring the state of the system dealt with obtaining the angle of the string in real-time. While saving a stream of video and batch-processing the frames to get the string angles is helpful in plotting and analyzing trajectories after they have been made, it helps very little with the problem of closed-loop control. Knowing this, the next attempt was running the string along a stiff rod, so the rod matched the angle of the string. From there it was simple to attach an encoder to the rod and measure the angle. What we discovered was that while this worked for a rough estimate of the angle, the non-zero weight of the rod was weighing it down and pulling it off the string's angle, creating error. On top of that, any quick dynamics of the system would excite the rod's own dynamics, generating far more complex oscillations than if there had been no rod.

Eventually we decided the only way to accurately and non-destructively measure the angle would be optically – in real time. This would mean we would need a way to quickly separate the string from its background. We decided that by casting the shadow of the string onto a linear optical sensor array, we could deduce the location of the string without interfering with the string itself or requiring a high computational load. See Fig. 24.

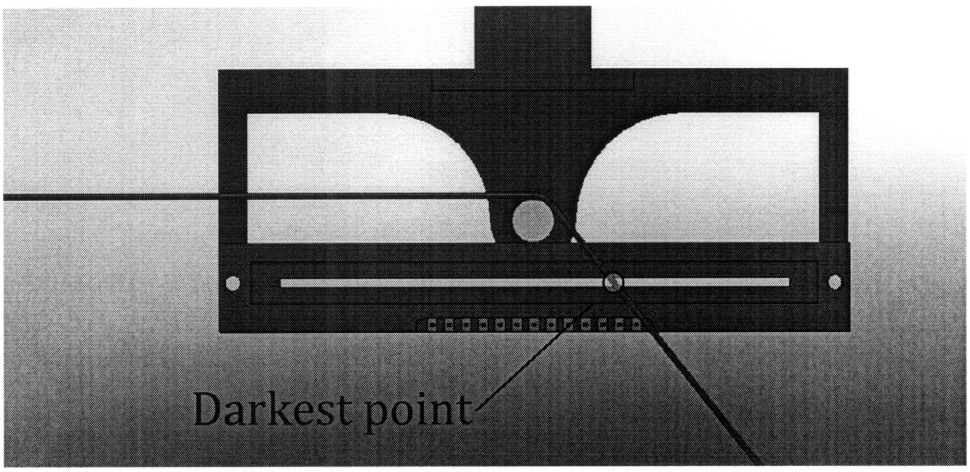

Fig. 24. A string passes in front of a linear optical array, which can feed back the darkest spot to determine the angle of the string.

The linear optical array used was the TAOS TSL1410R, which is a 1280-pixel photodiode array. The pixels have a pitch of 0.0635 mm, and so give a working length of 8.128 cm.

Due to the geometry of a string passing in front of a linear array, we had a few options when it came to placing the array relative to the pivot of the string. With a single array, the closer the array is to the pivot, the larger the angles we could measure, but that also means that the sensor is the least sensitive to angle changes at the center, or the straight-down state, and most sensitive at large angles. Also, this configuration can only approach π radians. See Fig. 25(a). Adding a second sensor can alleviate the sensitivity-location problem as well as increasing the maximum measured-angle problem by shifting the straight-down state to the edges of the sensors. See Fig. 25(b). Despite this, in our initial testing, a single sensor's accuracy and angle-extent was sufficient, and so we decided that the added complexity of coordinating two sensors was not worth the increase in angle and sensitivity.

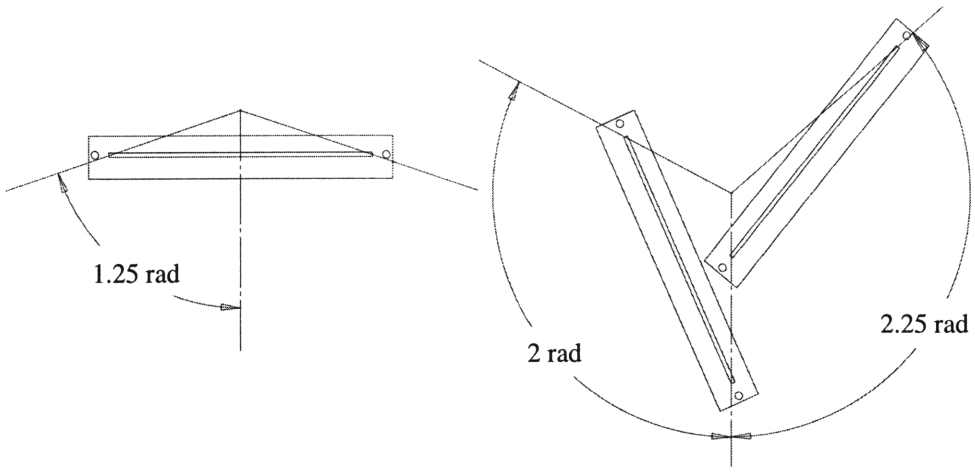

Fig. 25(a) Fig. 25(b)

Fig. 25. An arrangement of two optical arrays gives better sensitivity in the small-angle range, but increases the complexity of signal transmission.

In order to use the angle measurement as part of our closed-loop control, the frequency of the measurement would have to be sufficiently high, i.e. at least 100 Hz, especially if we would like to take the pure derivative as a state. Knowing this, we also had to develop a way to read in all 1280 optical array pixels, compare them together, and choose a single value as the location of the string. The only hardware running fast enough was the FPGA, and with that running at 1 MHz, reading a pixel value in even one clock cycle would mean the final rate would be at most less than 1 kHz. Additionally, comparison or of the darkest sensor had to take place on the FPGA, where it is generally difficult to perform higher-order computations. All these challenges were overcome, and the process was streamlined to an angle measurement frequency of 200 Hz.

In total, the data exchange between the Host PC code, the FPGA code, and the optical sensor's on-board circuitry was complex, as shown in Fig. 26.

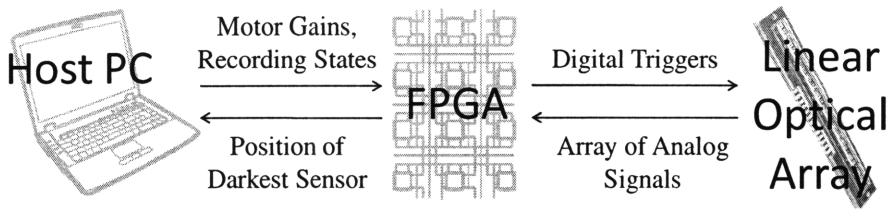

Fig. 26. Data transfer between host, FPGA, and the optical array was complex.

6.3. String Length Control

In order to experiment with either Point-to-Point or Continuous Path motion, we needed a way to control the length of the string in real-time. To do this, we used two simple controllers around the DC winch motor, as shown in Fig. 27.

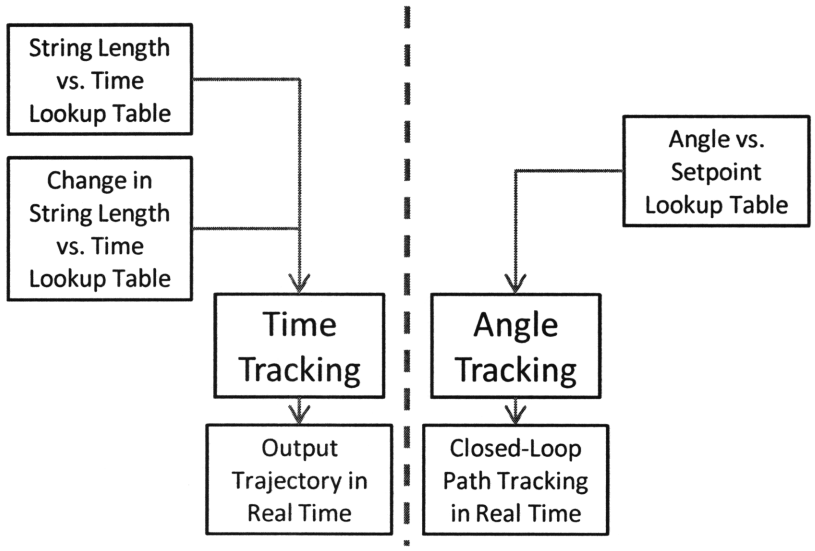

Fig. 27. There are two primary modes of running the LabVIEW software: Time Tracking and Angle Tracking.

6.3.1. *Simple Controller – Time Tracking*

The first controller, Time Tracking, essentially used a closed-loop PD controller around the servoed winch motor, and manually changed the winch setpoint with time, without including the angle in any feedback loop. The reason this worked was because the dynamics were simulated in order to generate the trajectory, so any delay suffered from using this open-loop was a constant delay suffered at all points in time. In this way, we can call this a feed-forward controller, and it can be seen in Fig. 28.

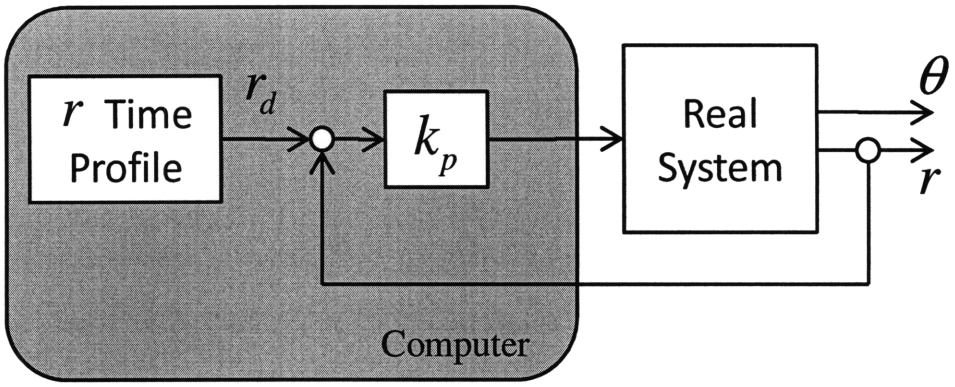

Fig. 28. The closed-loop block diagram of our P2P Controller is an example of feed-forward control.

6.3.2. *Simple Controller – Angle Tracking*

The second controller, Angle Tracking, is more naïve than the first, and this fact can be seen in the experimental results. This controller is essentially the previous controller, but instead of a predetermined time-trajectory for the string length setpoint, the setpoint depends completely on the measured angle of the string. While this seems like it should have worked, we found that because the setpoint was not the same as the current length, there was always an error between the setpoint and the actual length. A short time later, the closed-loop controller had moved the actual string length toward the setpoint, but because the angle had changed, so had the setpoint. What resulted was a severe delay in

the tracking due to the fact that no dynamics were incorporated into this controller. Because of this, it cannot be called a feed-forward controller, but can only be classified as a simple proportional controller around the state-dependent string length. Its block diagram can be seen in Fig. 29.

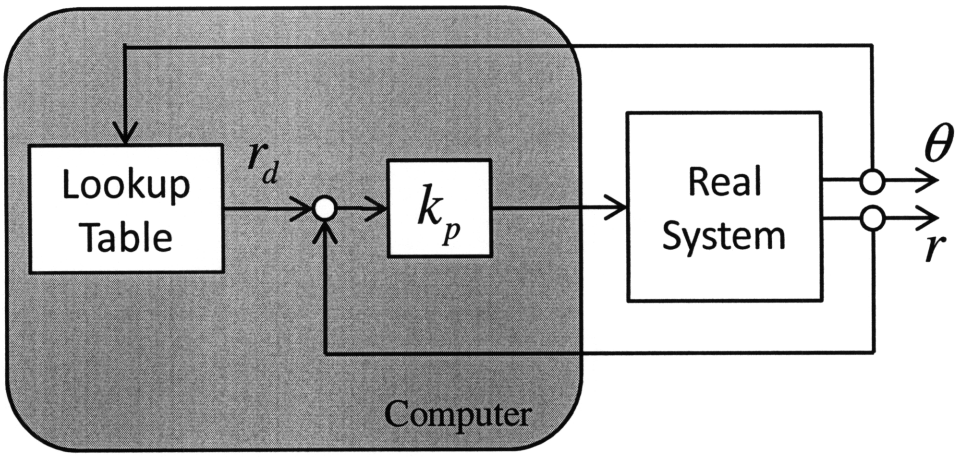

Fig. 29. The closed-loop block diagram of our CP Controller is simple, but not robust, and thus issues arose.

One solution to the problem of the delay in the simple proportional controller would be to incorporate a derivative term, making it a PD controller. This could be achieved by calculating not only the desired string length, but also the rate of change of the string length, similar to that shown in simulation in Section 5.1.4.2 on page 46. This way we incorporate the dynamics of the system into the controller, ensuring that the actual trajectory converges on the desired geometric path. An example block diagram is shown in Fig. 30.

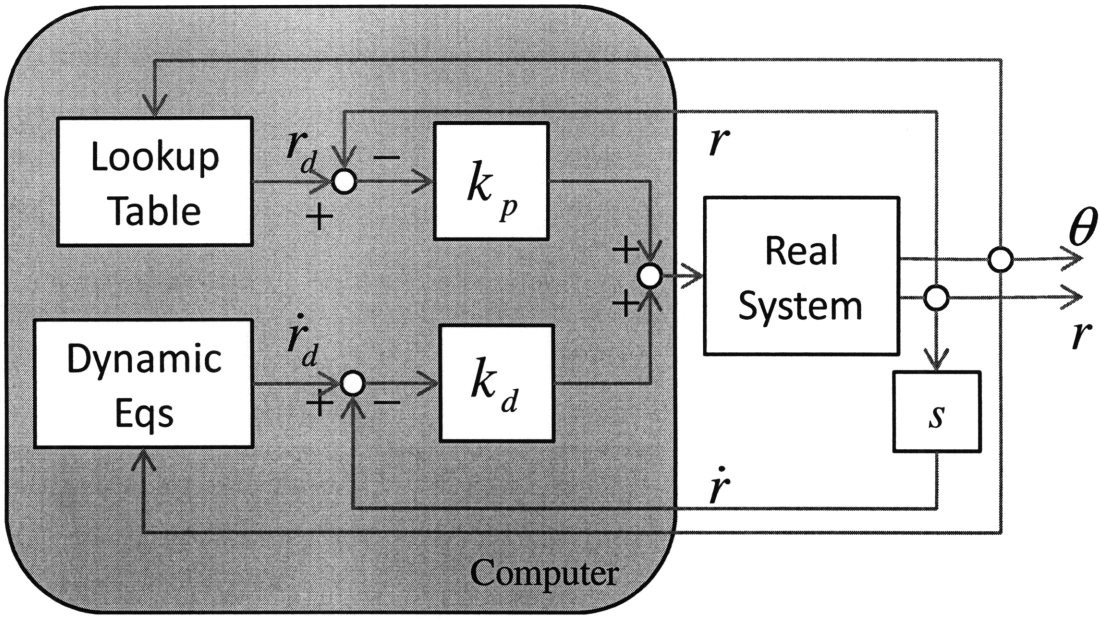

Fig. 30. An improved block diagram would have improved the performance of our Path Tracking system.

Chapter 7. EXPERIMENTATION

In our initial experimentation, we looked at three particular segments of the trajectory generation: the parametric excitation, a point-to-point trajectory, and path tracking using angular feedback.

To parametrically excite the system, we designed a trajectory with a smooth pull like in Fig. 7 on page 29, and then small osciallations with a driving frequency determined by the Mathieu solutions. The intent was to start with a small perturbation in the swing direction, and have the system parametrically drive itself to states of much higher energy. In fact, we were able to give the mass a small push (to provide an initial angle offset), and the system would quickly drive itself to large swinging angles.

Using the technique described in Chapter 4, we chose a start and an endpoint, and varied the pull time to match the energies of the two pulls. Then we joined the trajectories (with a proper amount of time between to allow for an integer swing period), and used the trajectory as a moving setpoint for a proportional controller. The tracking results are shown in Fig. 31. The actual measured angle is presented as points alongside the simulated angle.

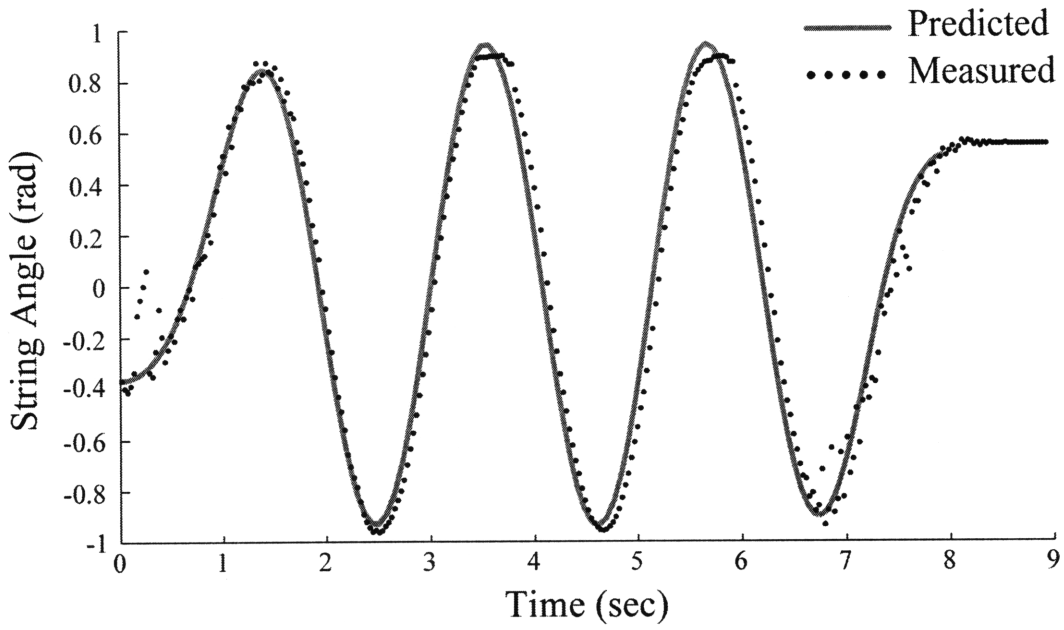

Fig. 31. A trajectory was designed to lift from and land at initial and final points with zero velocity. Shown here is the predicted and measured change in the angle of the string as a function of time.

By combining the recorded data from the actual length and actual angle of the string, plotting the X-Y position is more visibly indicative of the success of the trajectory generation. Fig. 32 shows the actual X-Y trajectory against the simulated, as would be seen from a point away from the swing plane.

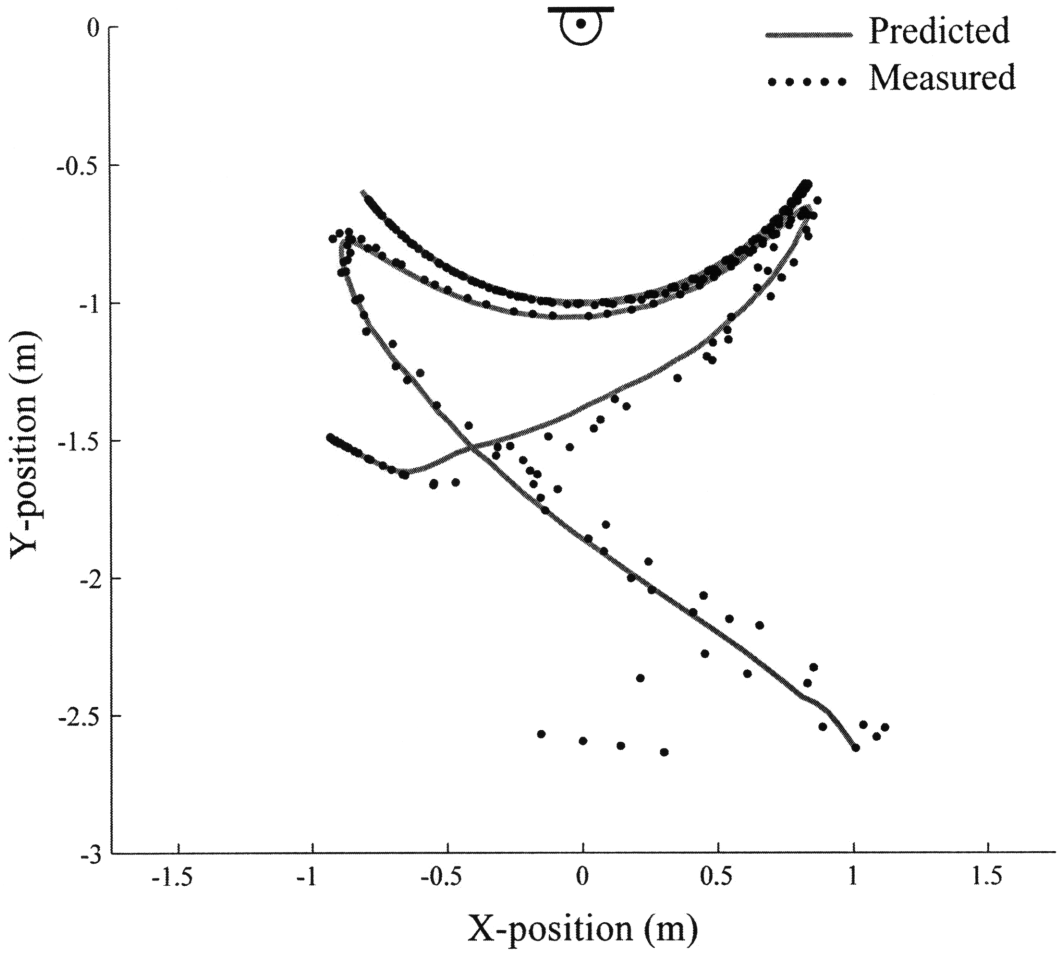

Fig. 32. The simulated X-Y position of the system is shown as a solid grey line, the actual trajectory as points. The noise at the beginning is mostly due to end mass wobble.

In general, the system followed the generated and simulated trajectory well. The initial state of the end mass would always slightly tilted, and so the first pull would always wobble the mass, which created errors in string angle measurement, as can be seen in the lower-right part of Fig. 32. Despite this, the final position and velocity matched the desired state nicely. This likely implies that the system is less sensitive to disturbances to or improper measurements of the initial state than previously expected.

To experiment with path tracking, we calculated a fairly smooth path geometry that resembles a wavy line, shown in grey in Fig. 33. We then created a closed-loop controller that would measure the angle of the string and calculate the string length that would place

the endpoint on the prescribed curve. That newly calculated string length would become the new setpoint of the length controller, as discussed in Section 6.3.2. Fig. 33, an X-Y plot of the desired geometric path and the measured trajectories, shows that the controlled trajectory generally tracked the prescribed path, with the only error introduced because the low proportional gain, required due to noise in the angle measurement. If the gain was increased without making the string lurch (shown in Fig. 35), the trajectory would track much closer.

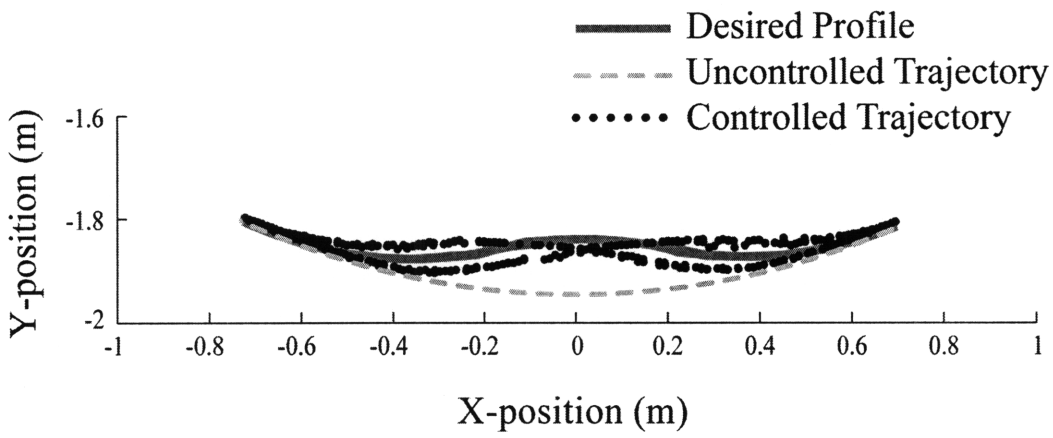

Fig. 33. A simple proportional controller is almost sufficient to generally track a prescribed endpoint trajectory.

Fig. 34 shows the absolute error in length between the prescribed geometric path and both a typical free-swinging pendulum and one actively tracking the path, shown first in Fig. 33.

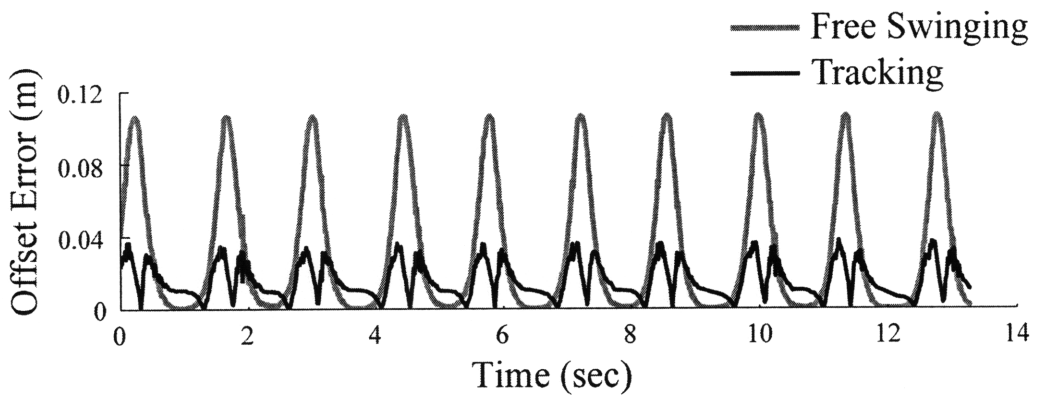

Fig. 34. While an uncontrolled pendulum has large errors toward the middle portion of Fig. 33, the simple controlled pendulum reduces those errors dramatically.

As discussed in section 5.1.5 on page 50, there are classes of geometric trajectories that cannot be tracked. In addition to this, in experimenting we found that with this technique of measuring the angle optically, there was another constraint; as long as the angle-measurement function only commanded small step changes in the setpoint, the path was smooth. If the geometric curve was too steep or a large disturbance affected the measurements, a large step change in the setpoint caused the controller to accelerate the endpoint mass too quickly, causing the mass to "lurch." This was mainly due to the naïve controller implemented. When a lurch occurred, because the endpoint was not a point mass, the mass began to wobble about its own center, shaking the string. As the angle was being measured optically at a high frequency, any small wobble of the angle would change the setpoint dramatically, and quickly. Fig. 35 shows an example of the lurch and the subsequent errors in string measurement. Additionally, these lurches caused very quick setpoint changes, which created accelerations larger than that of gravitational acceleration, which caused the mass to free-fall, violating the positive-tension constraint. When this happened, the free string moved out of line, causing more lurching and thus disturbing all further string measurements and losing the ability to control.

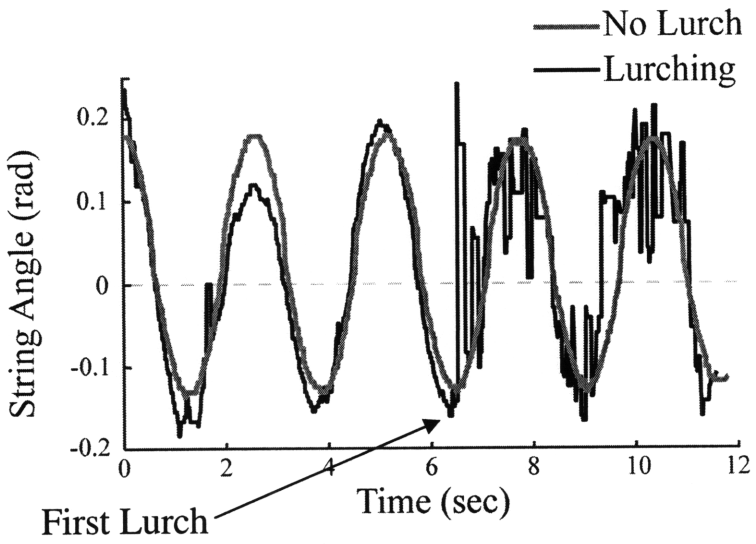

Fig. 35. In these two trials of string-feedback path-tracking, the first (grey) tracks well, but the second (black) lurches at 6.1 seconds, likely from an outside disturbance.

Chapter 8. CONCLUSION

With a system as simple as the Winch-Bot, looking at the theory behind the motion can open doors to many different types of desired motion. Here, by studying two simple types, Point-to-Point and Continuous Path, we dip our feet into all that is the Winch-Bot.

In theory, we've shown that the techniques of using these planned dynamic trajectories to increase the workspace of our small Winch-Bot succeeded and were controllable enough to specify both initial and final points and geometric paths. In experimentation, we've shown that even using only a feed-forward method of control was successful in moving the end-effecter to the final point using our Point-to-Point motion parameterization. It was also observed that real-time control of the trajectories could be done by varying the driving self-excited oscillations, to change the energy. This allows us to make up for errors in measurement, modeling, and implementation in real-time, which is important for controllability of the system. Additionally, initial implementation of Continuous Path control is promising, and we've shown that, even with a naïve controller, it is achievable for many classes of trajectories.

BIBLIOGRAPHY

[1] J. Albus, R. Bostelman and N. Dagalakis, The NIST Robocrane. Journal of Robotic Systems 10 5 (1992), pp. 709–724.

[2] S. Kamamura, W. Choe, S. Tanaka, S.R. Pandian, Development of an ultrahigh speed robot FALCON using wire drive system, in: IEEE International Conference on Robotics and Automation, Nagoya, Japan, 1995.

[3] M. Yamamoto, N. Yanai and A. Mohri, "Inverse Dynamics and Control of Crane-Type Manipulator," Proc. Of the 1999 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp.1228-1233, 1999.

[4] H. Arisumi, K. Yokoi, and K. Komoriya, "Casting manipulation (breaking control for catching motion)," in Proc. 2000 IEEE Int. Conf. Robotics Automation, 2000, pp. 1961–1968.

[5] Y. Nakamura and R. Mukherjee, "Nonholonomic path planning of space robots via a bidirectional approach," IEEE Trans. Robot. Automat., vol. 7, pp. 500–514, Aug. 1991.

[6] McLachlan, N. W. *Theory and Applications of Mathieu Functions*. New York: Dover, 1964.

[7] H. Arai, K. Tanie, and N. Shiroma, "Time-scaling control of an underactuated manipulator," J. Robot. Syst., vol. 15, pp. 525–536, Sept. 1998.

[8] Dahl, O.; Nielsen, L., "Torque-limited path following by online trajectory time scaling," Robotics and Automation, IEEE Transactions on , vol.6, no.5, pp.554-561, Oct 1990