3-2016

Dynamics and Power Generation Potential from a Tethered Kite Moving in a Horizontal Flightpath

Glenn Romo Gavi
grg5393@rit.edu

Dynamics and Power Generation Potential from a Tethered Kite Moving in a Horizontal Flightpath

by

Glenn Romo Gavi

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of
Master of Science
in Mechanical Engineering

Supervised by

Assistant Professor Dr. Mario W. Gomes
Department of Mechanical Engineering
Kate Gleason College of Engineering
Rochester Institute of Technology
Rochester, New York
March 2016

Approved by:

Dr. Mario W. Gomes, Assistant Professor
Thesis Advisor, Department of Mechanical Engineering

Dr. Kathleen Lamkin-Kennard, Associate Professor
Committee Member, Department of Mechanical Engineering

Dr. Margaret Bailey, Professor
Committee Member, Department of Mechanical Engineering

Dr. Agammemnon Crassidis, Associate Professor
Department Representative, Mechanical Engineering

Thesis Release Permission Form

Rochester Institute of Technology
Kate Gleason College of Engineering

Title:

Dynamics and Power Generation Potential from a Tethered Kite Moving in a Horizontal Flightpath

I, Glenn Romo Gavi, hereby grant permission to the Wallace Memorial Library to reproduce my thesis in whole or part.

Signature

Date

Acknowledgments

A huge thanks to Dr. Gomes for his dedication to his students and his willingness to teach. It is greatly appreciated, and this project would not have been possible without your guidance. Also to Dr. Bailey, Dr. Lamkin-Kennard, and Dr. Crassidis for your support. Matt Douglas was always there to provide technical knowledge when aerodynamic concepts did not make sense, and Shriya Gosavi challenged me to become a better student overall. A special thanks is due to Kathleen Ellis for lighting a fire that continues today and to Manuel Hidalgo for bearing through all of the struggle. I am very much appreciative of the long hours spent in the lab with Joseph Lee and Samket Dhami, along with the short but great time with Caleb Weeks. You all made my thesis possible and I will cherish this moment. Thank you all.

Abstract

Dynamics and Power Generation Potential from a Tethered Kite Moving in a Horizontal Flightpath

Glenn Romo Gavi

Supervising Professor: Dr. Mario W. Gomes

Tethered-wing power systems are a viable possibility for collecting energy from stronger, more consistent winds found in the upper regions of the atmosphere where conventional wind turbines are incapable of reaching. To date, all of the tethered-wing systems fly with the tether oriented down-wind of the ground attachment point. Examined here are the dynamics and performance of a novel system where the tether is oriented both upwind and downwind of the ground attachment point during normal operation of the device. Certain prototypes built by Makani and Ampyx Power are considered to have motions analogous to the motions of the blade tips on conventional horizontal-axis wind turbines. If true, this system has motions that are analogous to conventional vertical-axis wind turbines. The system has a ground-based generator which is mechanical coupled to the aircraft and energy is generated on the reel-out phase of each cycle while a smaller amount of energy is consumed during the reel-in phase of each cycle. A simple model was developed which captures the dominant dynamics of this system and shows, via simulation, that the proposed system is viable and capable of stable and unstable periodic motions with a simulated closed-loop tether tension controller or a simple open loop reel-rate controller. In addition, it is capable of motions which produce net positive power. The small system examined, where parameter optimization was not performed, predicts an average cycle power of more than 500 watts in a 10 m/s wind.

Contents

List of Tables

List of Figures

Nomenclature

$\hat{i}$	Inertial Coordinate Frame, X-Axis
$\hat{j}$	Inertial Coordinate Frame, Y-Axis
$\hat{k}$	Inertial Coordinate Frame, Z-Axis
$\hat{e}_\theta$	Coordinate Frame,
$\hat{e}_r$	Coordinate Frame,
$\hat{e}_\phi$	Coordinate Frame,
m	Mass of the Kite
b	Wingspan of the Kite
c	Chord Length of the Kite
S	Effective Surface Area of the Kite
AR	Aspect Ratio of the Kite
ρ	Air Density
μ	Dynamic Viscosity
θ	$\hat{i}\hat{j}$-Plane Tether Angular Position
$\dot{\theta}$	$\hat{i}\hat{j}$-Plane Tether Angular Velocity
$\ddot{\theta}$	$\hat{i}\hat{j}$-Plane Tether Angular Acceleration
l	Tether Length
$\dot{l}$	Tether Length Change Velocity
$\ddot{l}$	Tether Length Change Acceleration
ϕ	Altitude Angular Position
$\dot{\phi}$	Altitude Angular Velocity
$\ddot{\phi}$	Altitude Angular Acceleration
T	Tether Tension
L	Airfoil Lift Force
D	Airfoil Drag Force
F_p	Constant Perpendicular Force
$\hat{\lambda}_L$	Lift Force Unit Vector
$\hat{\lambda}_D$	Drag Force Unit Vector
$\vec{V}_\infty$	Wind Velocity
V_∞	Magnitutde of Wind Velocity
$\vec{V}_{kite}$	Kite Velocity
$\vec{V}_A$	Apparent Wind Velocity
β	Tether to Bridle Angle
γ	Tether to Apparent Wind Velocity Vector
α	Angle of Attack
ζ	$\hat{e}_\phi$ Coor. Frame to Kite Span Line Angle
η_{power}	Power Ratio
g	Gravity

Chapter 1

Introduction

1.1 Motivation

Energy provides a certain standard of living that society is eager to keep. The sources of energy currently maintaining this standard (Coal, Petroleum, Natural Gas, and Nuclear Materials) are detrimental to the biosphere and are not sustainable in practice nor supply. In 2014, the U.S. Energy Information Administration released projections predicting how the country will satisfy its energy demand through 2040. Renewable energies and biofuels contributed 9% of total U.S. energy demand in 2014 and will only increase 3% by 2040 [11]. In order to facilitate a shift of demand to renewable energies, multiple systems must be created that are applicable to different locations and environments (e.g. urban, off-shore, climates). The goal should be to diversify the energy collection portfolio with multiple large scale production sites [2].

A major supplier of renewable energy, Wind Energy, is commercially captured by modern wind turbines with onshore and offshore sites for turbine arrays. Some of the largest and highest capacity wind turbines are located offshore because of steadier winds [16]. For example, the AMSC SeaTitan is capable of producing 10MW at max capacity and has a operating wind speed range from 4-30 m/s.

Modern wind turbine rotors have a maximum energy extraction due to hub height,

geographic location, rotor diameter, wind behavior, and turbine mechanical properties. However, there is a theoretical maximum called Betz Limit. Extracting energy from the wind will cause the downstream velocity of the wind to decrease compared to the upstream velocity since the wind is transferring kinetic energy to the turbine. The downwind velocity must also stay >0 m/s because the air must pass through the turbine and never become stagnant. Therefore, Betz Limit is an estimated maximum of energy extraction from a wind system due to the necessary upwind and downwind velocity changes. Betz determined 1/3 of the original velocity must remain downwind of the system, and that rotor efficiency can only gather 59.3% of the total energy present in the wind. Using the equation for kinetic energy and mass flow rate, the energy available in the wind is found in Eq. (1.1).

$$P = \frac{1}{2}\dot{m}V_\infty^2 = \frac{1}{2}\rho A V_\infty^3 \qquad \bar{P} \approx \frac{1}{2}\rho V_\infty^3 \tag{1.1}$$

where P is the total power of the wind, $\dot{m}$ is the mass flow rate, ρ is the density of air, A is the cross-sectional area, $\bar{P}$ is the energy density, and V_w is the velocity of the wind.

For the system proposed in this paper, Betz Limit still governs, however, crosswind motion systems are slightly different because they only remove a fraction of the total energy in one pass and may pass through the same cross-sectional area again depending on altitude of the kite. This is possible since the crosswind area profiles are not circular and the path through the blades of are better represented as a vertical axis wind turbine.

With a maximum energy extraction efficiency available for all wind systems a standard called the capacity factor (CF) is used to compare wind energy systems. CFs are ratios between actual energy extracted from the wind and the energy extracted from the wind if the system performed at full capacity during the same time period. Modern wind turbines will have a CF of 20-40% [1]. The low capacity factors are due

to ambient wind speeds outside the range of operating conditions, typically higher than the cutoff speed for modern wind turbines or energy dissipation by tilting the blades to reduce damage on the unit.

Currently, modern systems produce energy with low CFs, and there is a high demand to improve the performance. The increased performance in CF might therefore be found in High Altitude Wind energy systems. High altitude winds are more consistent and are hypothesized to increase CFs. Canale, whom performed a simulation and experimental tests of a high-altitude tethered kite system, believes these systems might double the CFs of current wind systems [10].

Therefore, crosswind kite systems designed with kite-tether combinations have a high potential to increase the performance of wind energy systems. It is estimated they will require 90% less material compared to modern wind turbines [13], they are not restricted to the same height limitations of roughly 250m, and they will perform in a wider range of wind speeds where modern turbines have a high cut-in and low cut-out wind speed. The proposed kite system will reach heights >1000m and will hopefully generate higher amounts of energy compared to its counterparts while reducing material costs [13].

1.2 System Description

The proposed system is unique because rather than creating a system which pumps back and forth with changing horizontal direction and altitude commonly found in current ground generation systems, the proposed system will rotate around an anchored point, where a drum houses a tether, while keeping a relatively constant altitude. Eventually, the system will adapt to variable changes in wind vectors and array installations of these smaller systems will be possible. Figure 1.1 depicts the basic setup and concept flightpath with increasing tether length downwind of the anchor and decreasing tether length as the wing flies upwind of the anchor.

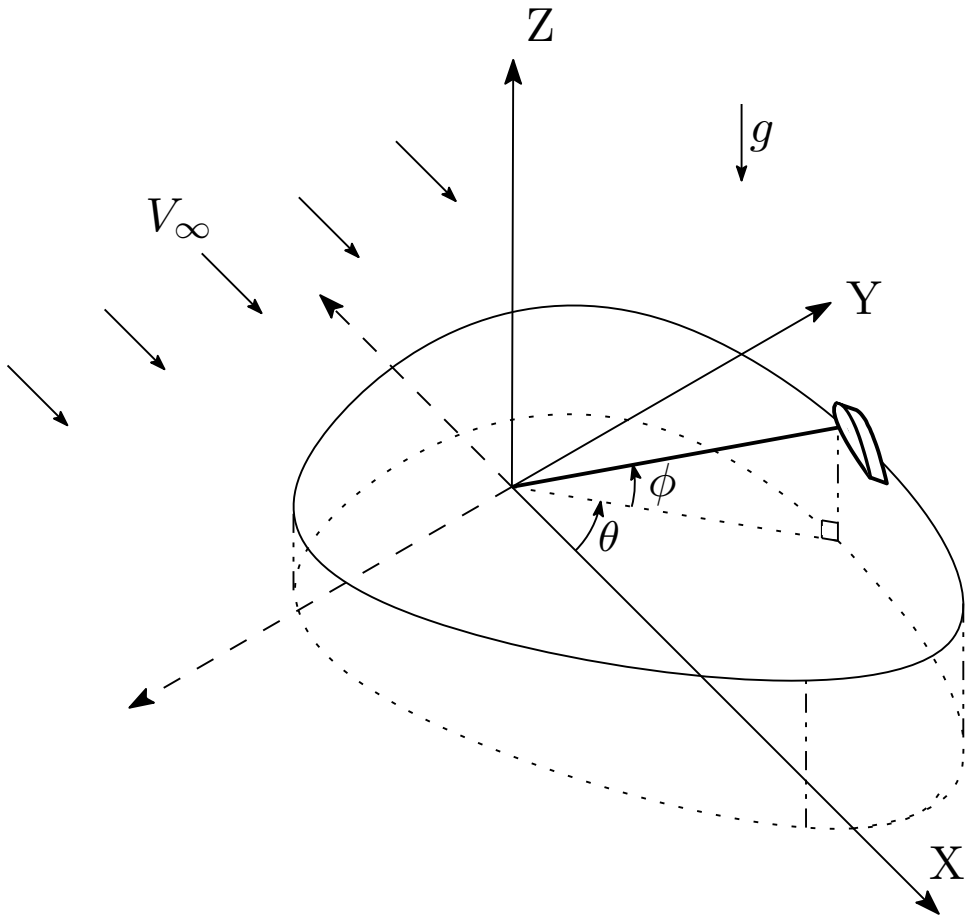


Fig. 1.1: Isometric schematic of our variable tether-length airborne wind energy system which moves in a horizontal flight path with the tether both upwind and downwind of the base station.

Gravity acts in the negative Z-direction and the simulated wind travels in the positive X-direction. The XY-plane is tangential to the surface of the earth at the location of the anchor. As the wing begins to fly upwind of the anchor point, tether tension is increased by reeling in the aircraft and, therefore, changing the apparent velocity vector to keep the wing in flight.

A controller will govern the reel-in and reel-out rates of the tether along with the range of angles for both phases shown in Fig. 1.2. The flightpath projection of a single pass (wave) onto the XY-plane of the system then is an asymmetrical, oblong circle. The plane of the flightpath likely will not lie on the tangential plane to the surface at the anchor point but will rather have an adjusted planar angle due to the reel-in and reel-out applied tension.

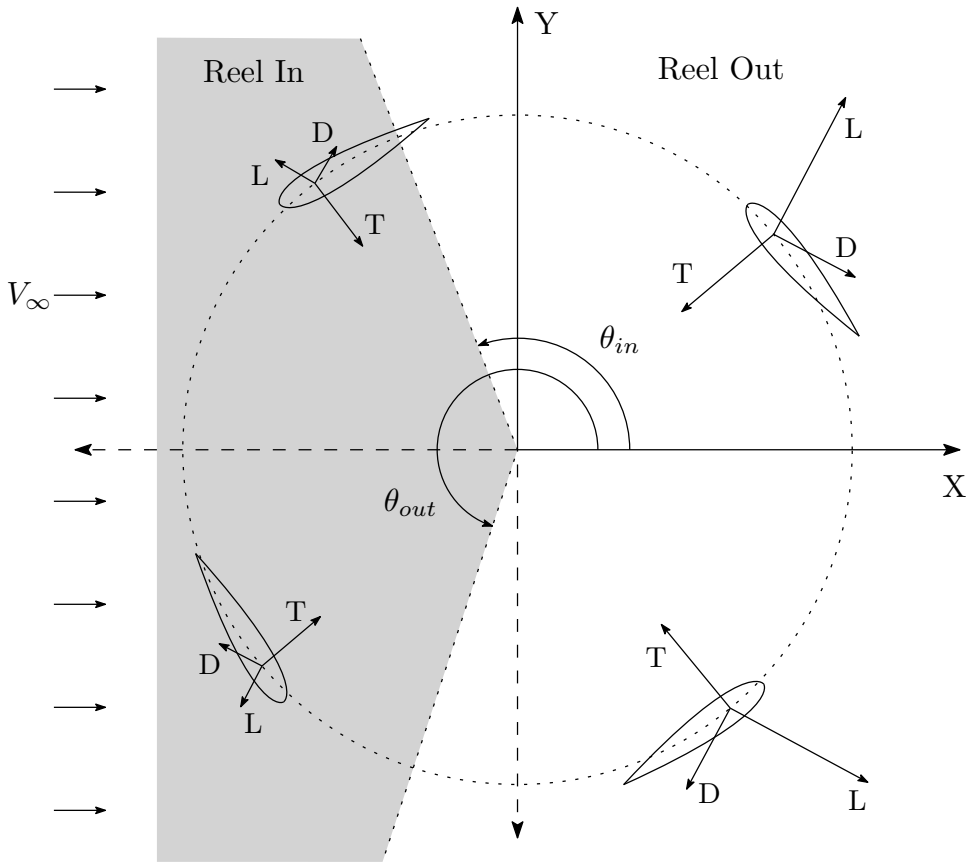


Fig. 1.2: Phase Diagram of reel-in region for tether retraction and reel-out for tether release

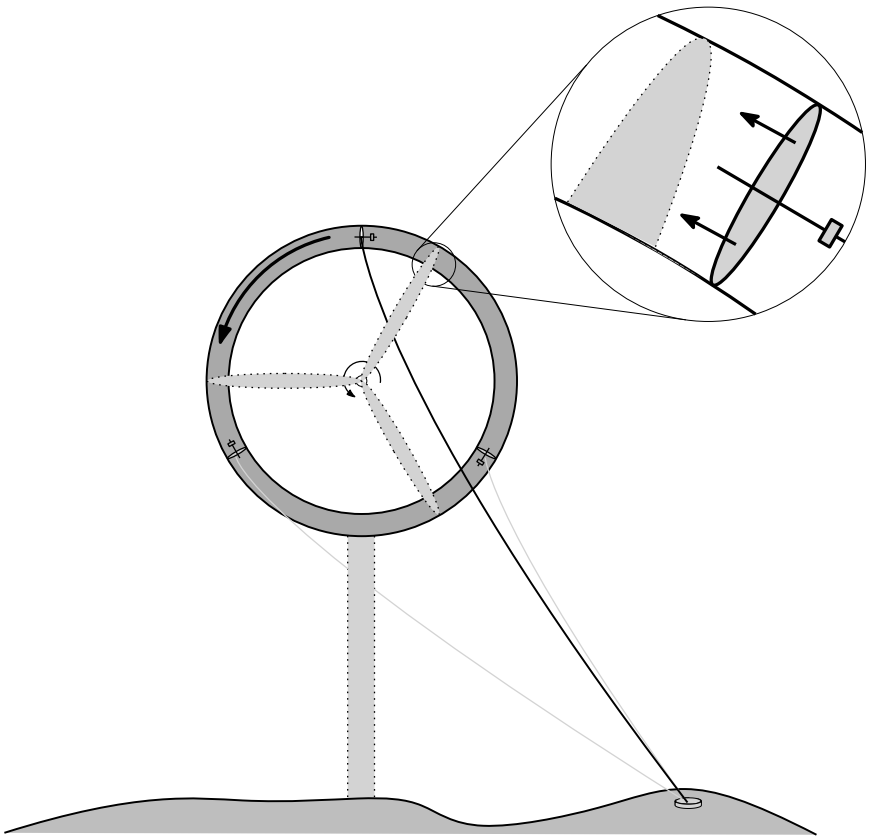


Fig. 1.3: Conventional horizontal axis wind turbine (HAWT) configuration[left] and a down-wind tethered-wing airborne wind energy system. Note that the swept area of the tip of the HAWT system is a scaled down version of the swept area of the airborne wind energy system (both annular regions).

Modern wind turbines typically use an internally located generator to produce energy from wing blades rotating in front of the main hub. Figure 1.3 shows a Horizontal Axis Wind Turbine's (HAWTs) front face with counter-clockwise rotating blades. Current Airborne Wind Energy (AWE) systems are analogous to the tip of a single HAWT blade rotating in the same counter-clockwise motion with mini-generators located on-board the aircraft. The remaining wind turbine materials are removed and reduced to a single aircraft and tether in AWE systems. The shaded, annular region is the effective swept area of the AWE system displayed.

Tethered-wing systems can generate energy from the wind with generators located on-board the aircraft or with generators located on the ground. The AWE system

described in Fig. 1.3 is an on-board generating system and the anchoring tether is simultaneously an electrically conductive line to transmit power from the aircraft to the ground. As the aircraft sweeps around in the same fashion as the tip of a modern wind turbine blade, wind passes through mini-turbine blades mounted on the aircraft similarly to propeller planes. This motion creates rotations and torques on internal generators to produce energy on-board the aircraft to be transferred along the tether.

Airborne wind energy systems generating power without the use of on-board turbines create energy by mechanical coupling a ground station and the aircraft. The ultimate result is a pumping motion on the tether from small or large wind force on the aircraft (alternately lengthening and shortening the tether). As the aircraft travels further away from the anchoring point, a drum unwinds while releasing cable to increase tether length. The unwinding of the drum during reel-out is the positive energy generation action taken to harvest wind energy. During reel-in, a motor is used to wind the tether back onto the drum while using energy and is referred to as the negative energy generation action. The goal is to reduce the wind force acting on the aircraft during reel-in and increase the wind force on the aircraft during reel-out to obtain a positive net energy generation.

As discussed, on-board generation systems are analogous to HAWT systems. Making a similar argument, the proposed system is analogous to a Vertical Axis Wind Turbine (VAWT) system. As shown in Fig. 1.4, the wing of the AWE system is essentially a portion of a single VAWT blade with a circular, horizontal trajectory. Removing the majority of the VAWT and replacing it with a tether and anchor point will provide similar energy generation dynamics and effective cross sectional area to that of a portion of a VAWT blade.

This AWE system was chosen because of the abilities to harvest energy from air currents found in the upper regions of the atmosphere. These currents are stronger and more consistent which will lead to increased capacity factors for wind energy

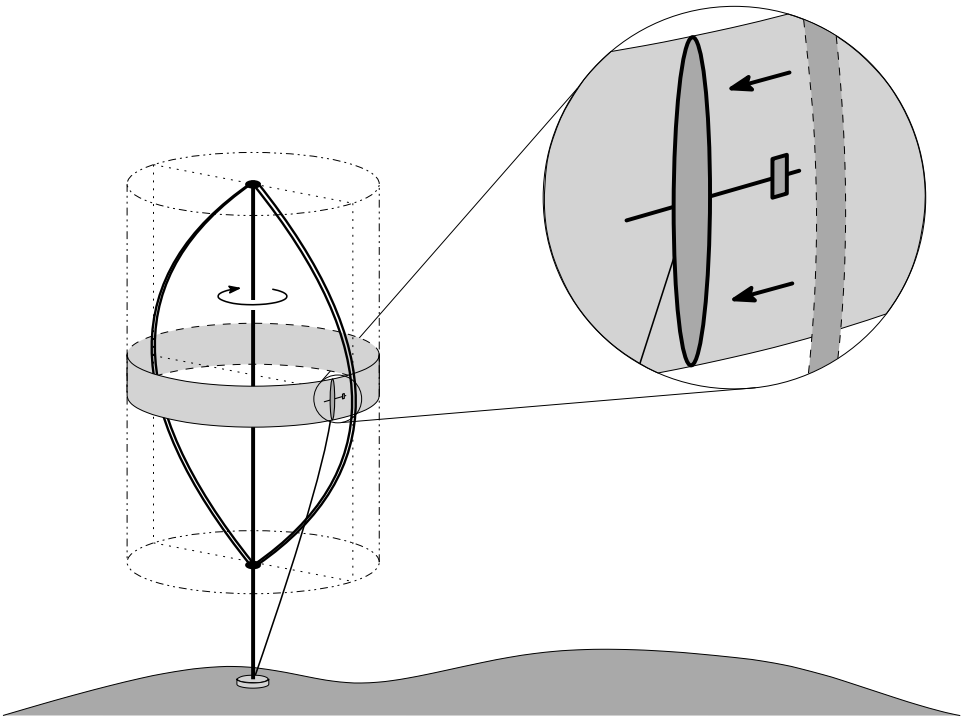


Fig. 1.4: Conventional vertical axis wind turbine (VAWT) configuration [bottom left] with a top view of the path of the blade sections furthest from the axis of rotation shown in the top left of the figure. A schematic of an analog tethered-wing airborne wind energy system is shown on the bottom right, with the path of the wing shown in the top right of the figure.

harvesting devices [21]. Not only are the upper air currents a benefit in switching to AWE systems over conventional wind turbines, AWE systems reduce the amount of materials by 90% [13], have a larger availability for site installation, including remote locations, and increase the range of cut-in and cut-out wind speeds for energy production. Drawbacks include the increase in complexity for steady flight operation and competing airspace limitations.

The proposed system also is not without competition. Current prototypes from both on-board and ground-based systems have been developed by several companies. Consequently, there is an inherent disadvantage in pumping AWE systems versus on-board generation since it is necessary to consume energy during the tether retraction phase, but, as Loyd discusses in [14], systems with on-board turbines and those with ground generators are capable of producing comparable amount of power in identical wind conditions. This system will have several advantages over current prototypes: 1) The tether will not transfer energy and a high-voltage line from the ground to the aircraft is unnecessary; 2) Mechanical subsystems are not needed to reduce lift force during tether retraction; 3) There is a decrease in tether tension during tether retraction because of kite orientation with the incoming wind.

1.3 Research Goals

1.3.1 Numerical Two-Dimensional Model

The primary goal is to develop a model to numerically integrate dynamic equations to simulate the horizontal flightpath of the proposed variable-tethered kite. The model will calculate energy generated by unraveling a tether as the wind force pushes the kite away from the ground station, and energy consumed while the wind assists with the reset (i.e. reel-in). The average net power production of the system is positive if the system generates more energy during the reel-out phase of the kite than it uses for the

reel-in phase. A controller, the ODE event trigger, will monitor the proposed system by regulating reel-in and reel-out rates by determining when to switch reel-phases. Net power is determined for a single cycle when the system operates at steady state. The model will simulate the dynamic behavior of the system as the wing interacts with the wind that is traveling horizontally across the face of the flightpath, as shown in Fig. 1.2. A motor will retract the tether once the lift from the air stream is no longer sufficient to keep the kite in flight, therefore a small amount of generated energy is consumed during operation. This model is intended to analyze the flight path, kite forces, and net average cycle power production from reel-out and reel-in phases of the kite.

1.3.2 Steady, Periodic Motion and Cycle Power Production

Producing energy with a turbine is to design or discover environments that foster periodic motion. This simulation will be a tool to model the dynamic behaviors of the wing and tether combination. It will be used to determine which parameters lead to periodic motions that result in net positive energy generation. Solutions will consist of repeating patterns that consistently generate energy and will keep the tether in constant tension. The net average cycle power will be calculated for each viable solution found.

1.4 Literature Review

1.4.1 Energy Potential

Harvesting energy from the wind is a well-known concept. Wind mills and simple mechanical turbines are commonly distributed pieces of technology. A less-known approach becoming more popular today uses kites in high altitudes as a method to extract energy from the wind. There are several 'High Altitude Wind Kite' designs

that generate electricity with turbine rotation. These systems can passively use the speed of the wind, airbone wind energy systems (AWEs), or use crosswind velocities that are faster.

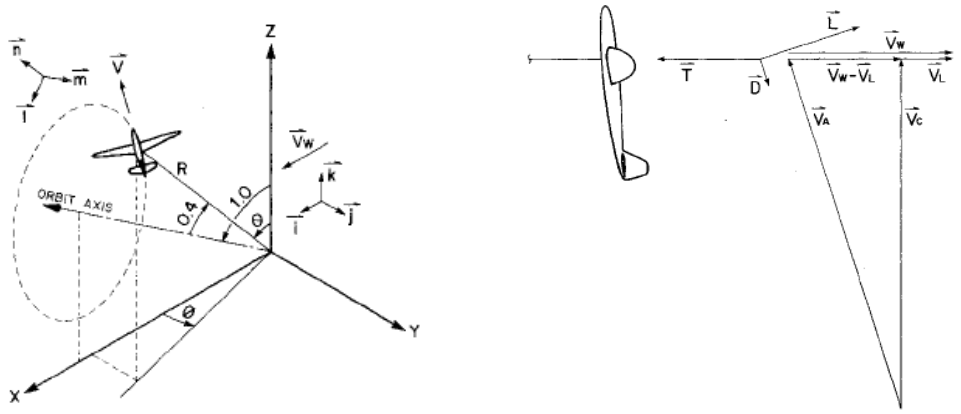
Crosswinds occur when an airfoil placed inside a stream of air begins to cross perpendicular to the direction of the air-stream flow. A new velocity vector is created that describes the magnitude and direction of the wind the kite is experiencing while flying through the air stream. The apparent velocity vector ($\vec{V}_A$) is found from the difference of the vectors of the air stream ($\vec{V}_\infty$) and the wing velocity ($\vec{V}_{kite}$), shown in Eq. (1.2).

$$\vec{V}_A = \vec{V}_\infty - \vec{V}_{kite} \tag{1.2}$$

Modern wind turbine designs are fabricated to take advantage of crosswinds because of the rotated tips on the ends of each blade. The tip of the blades are twisted to maximize lift in the new 'apparent wind direction' instead of the plane of motion of the air stream. Using cross-wind velocities will provide much higher wind velocities, thus, creating potential to generate larger amounts of energy with similar swept areas by a crosswind system.

Loyd, in [14], analyzed the potential energy gains from crosswind velocities by analytically modeling several systems in a downwind, tethered kite system. Figure 1.5a contains Loyd's diagrams depicting the flight pattern and orientation of a tethered kite anchored to a ground station.

The air stream flows in the positive x-direction and crosswind occurs as the kite moves orthogonal to the air stream flow. Figure 1.5b represents the forces placed on the kite and the characteristics of the air stream at the kite.As the kite flows through crosswinds, lift and drag vectors are created opposite the tether vector from the apparent wind velocity ($\vec{V}_A$) mentioned above. Achieving higher lift to drag ratios will benefit power production by the square of the ratio [14]. As the lift to drag ratio



(a) (b)

Fig. 1.5: Orbiting Pattern for system outlined in [14] to have an tethered kite downwind of a base stating and the velocities or forces associated with the kite.

doubles, the power produced will quadruple. According to the analysis of a C-5A aircraft, a single system comprised of a plane, ground station, and crosswind motion with a lift to drag ratio of 20 would produce three times more power than other wind turbines of that time, with the potential to reach twenty times the power production with newer technology [14].

Yet where in the atmosphere should these systems be placed? According to Tieleman in [21], above the atmospheric boundary layer which separates inconsistent winds and more reliable, stronger air streams. This boundary is usually found above 1300m taking into consideration the landscape and regional wind pattern. The altitude of the layer will occasionally decrease during the day while strong winds are prevalent, but will rise again because they are not consistent at night. Therefore, depending ont he time of the day, the optimal system might change altitude but it is expected to have energy generation year round. Another study completed in Southeastern Europe by Marko Ban *et al.*, in [8], found a large amount of the region is classified as potential deployment locations for kite systems. Maps are provided in the paper, but the largest available regions are offshore installations and several on-land locations away from airports, railways, and regional roads.

The study found the energy density above the atmospheric boundary layer and

below 2500m to average 350 W/m^2. As a reference, the energy density from the sun is referenced around 1000 W/m^2 on Earth's surface during a cloudless day while current marketable solar panels have efficiencies around 12-20% with high end caps around 30-35%. [16]. The studies altitude cap of 2500m is introduced to decrease interference with low flying aircraft and FAA regulations.

1.4.2 Kite Systems and Controllers

Several kite system prototypes are available today and researchers are maximizing the energy generation from these systems to have commercially competitive applications. The energy generation mechanisms vary between on-board generation and mechanical coupling for ground generation. The mechanical energy transformations also vary with each system: downwind rotating kites, stationary flight with rotating turbines, tether tension from crosswind flight-paths, or tether tension from reel-in and reel-out pumping.

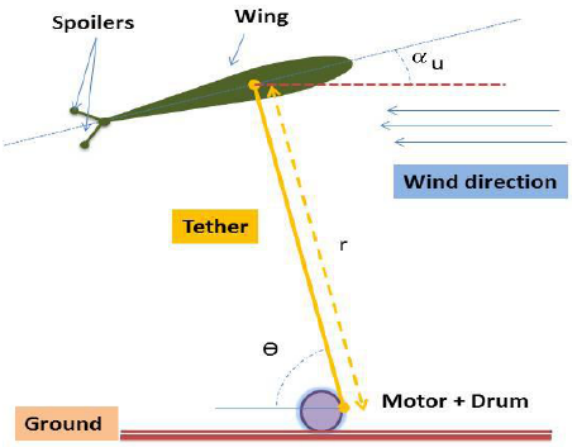
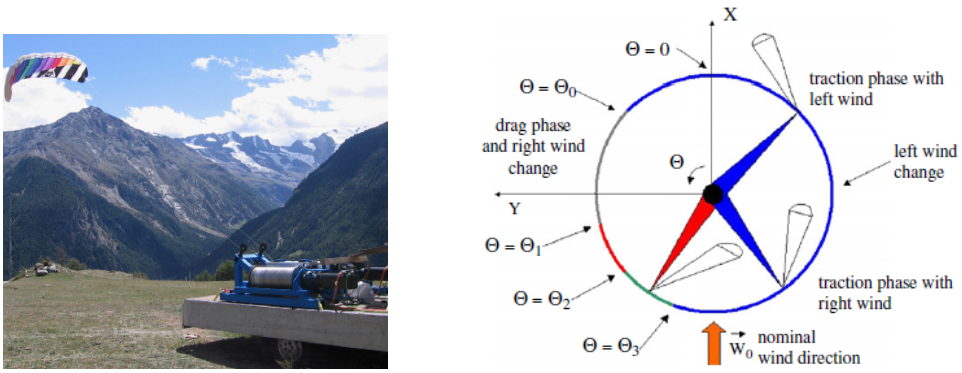


Fig. 1.6: Pumping Kite System in [15] with release and retraction in the same plane while maintaining stable flight control.

Tether tension tended to dominate research for alternative wind energy systems in the late 1990s and early 2000s. Fuel savings were analyzed for tugging ships and rigs with kite systems, and the concept of pumping mills/laddermills was rationalized. Pumping mills are kites attached at intervals along the length of a tether and have the appearance of a ladder which induce a pumping effect by increasing or decreasing lift with variable angles of attack. A simplified diagram is provided in Fig. 1.6 containing only one kite. The tether unwinds a drum as it is drawn out with the wind until a

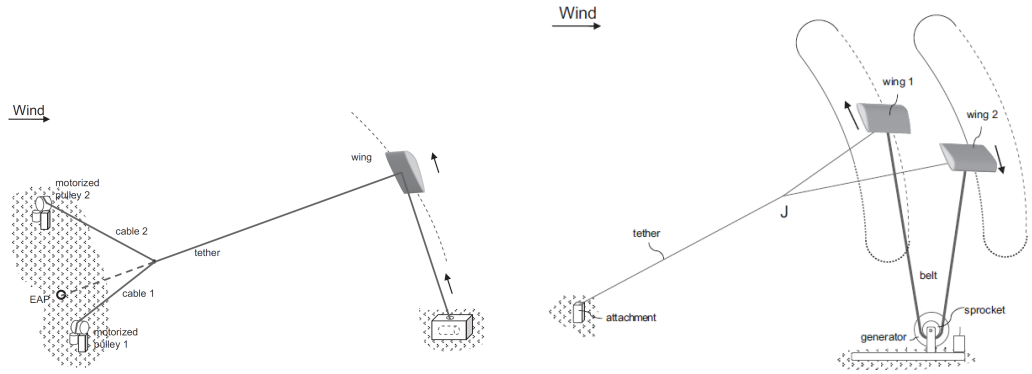


(a) (b)

Fig. 1.7: Physical testing system to verify kite behavior from [10] and the multi-kite proposed carousel structure for large scale energy generation.

motor rewinds the drum at the end of the cycle.

Another system, KiteGen, was developed by Canale *et al.* to create rotational motion of a carousel using multiple kites. Moment forces are induced by the tug of the kites as they travel through a 'traction' phase, shown in Fig. 1.7b. As the kites reach a point where traction counters the desired rotation, they are placed into a neutral 'drag' phase. With data obtained from the experiment of a single kite, shown in Fig. 1.7a, feasibility to control two tether lines directing the kite between drag and traction phases was achieved [10]. Discussed in the paper were estimates drawn from data collected in the simulation for a full carousel unit. Potentially, 450kWs of generation is capable from a single kite with possible gains up to 1GW with increased kites and larger carousel. The advantage of this system is a large decrease in land mass and an increase in nominal power. Variable wind speeds were used in the simulation as linear functions of height, shown in Eq. (1.3) [10]. Capacity factor of the proposed carousel was also shown to double that of a system similar in energy output capabilities. The capacity factor of KiteGen was found by comparing power curves of KiteGen and a 2MW, 90m diameter wind turbine along with wind speed data of locations throughout Italy.



(a) (b)

Fig. 1.8: Theorized system from [13] to provide pumping action similar to other downwind, tethered kite designs but with a separate tether for energy generation directly beneath the kite. Second system contains two kites working in unison to reduce power loss during tether retraction.

$$W_0(z) = \begin{cases} 0.04z + 8 & z \leq 100 \\ 0.0172(z - 100) + 12 & z > 100 \end{cases} \tag{1.3}$$

Based off the concept of Loyd, Goldstein proposed a system with a circular flight-path but instead placed the energy generation on the ground, as shown in Fig. 1.8a. The kite is connected to a separate tether wrapped around a drum underneath the flightpath. As the kite gains height, the tether is drawn out and unwinds the drum, generating energy. As the kite loses height, the motors of the drum retract the line, consuming energy. A second concept placed two kites with flight-paths at 180° offset, the tether connected the two kites together but would loop around a pulley on the ground, shown in Fig. 1.8b. As one kite gained altitude, the other would lose height and energy loss due to the motor rewind would not be necessary. The power generation from the system, through numerical analysis, is comparable to a modern wind turbine and would cost a tenth of the amount in material costs [13].

Besides flying kites to create tension in a tether and using the pull to generate energy, kites are flown in steady state operation in conjunction with crosswind velocities. Modeling and experiments conducted at the University of California, Santa

Barbara, with the same kite design as Canale's carousel, Fagiano managed a stable flight pattern with a controller and two tether lines, Fig. 1.9. The infinity pattern reduces tether tangling and yields the greatest traction forces [12]. Figure 1.9 depicts a loose, steady state operation for the kite with two motor controllers to determine direction. Finer steady state operation is possible, but requires more energy from the motors and reduces possible total energy production. Another advantage to the experimental controller, which is based on angle and height of the kite, is the ability to achieve stable infinity, curve flightpath without wind speed or the calculating the apparent wind velocity vector.

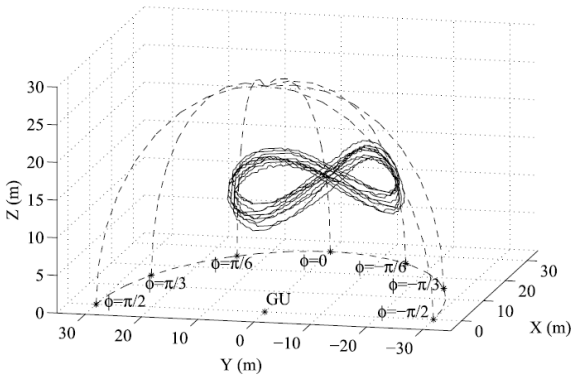


Fig. 1.9: University of California, Santa Barbara, flightpath of kite model to produce infinity shaped pattern [12]

Most research thus far has used steady winds because of the expectation that the systems will operate in the steady air stream above the atmospheric boundary layer. However, testing takes place below this layer and wind gusts, changes in steady air stream state, or zero wind conditions are possible. Lozano and other researchers from Gipsa-Lab in France developed a controller to monitor wind variations and manipulate the tethered kite to manage the new conditions [12]. The system models a typical pumping kite but with a rigid wing and is similar to the design for Ampyx Power discussed under Section 1.4.4. The experimental values of perturbations in response to wind changes were captured using motion capture systems and image processing.

1.4.3 Tether Dynamics

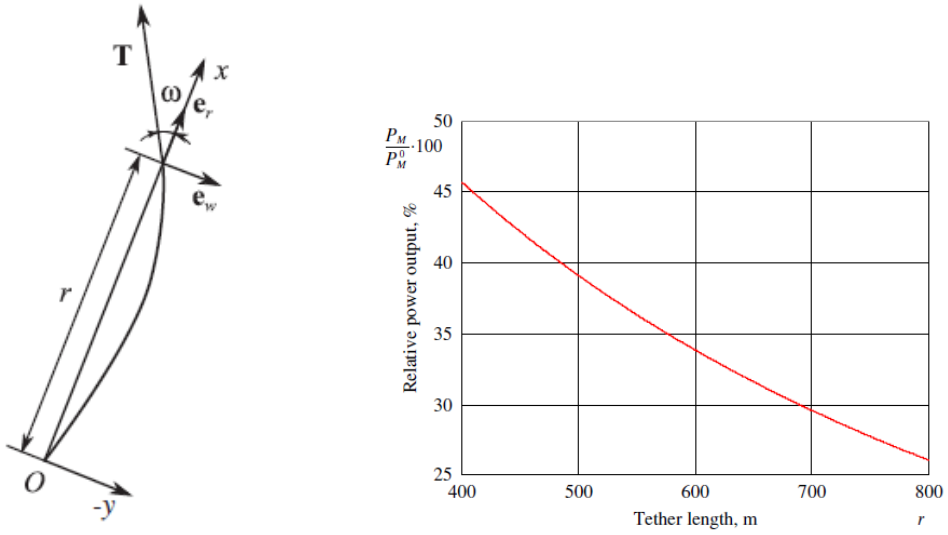
High altitude wind kites use one or more tethers to anchor the system to the ground and, depending on the system, may use the cable to transfer electrical energy into a storage device. Argatov sought to analyze the energy loss and dampering of the system from the dynamics of a dual-line tether model. Figure 1.10a shows the dimensions of the tether in reference to an origin and rotational coordinate system. The tether lines bow due to weight causing the kite to vary in location than if the tether was perfectly straight. The change in location would result in lower altitude and shorter radial distance from the base station. The tether will also change the angle of attack based on the angle of tether attachment (ω) on the rigid wing, shown in Eq. (1.4) which can be modified with a simple bridle system.

$$\alpha = \alpha_0 - \omega + \Delta\alpha \tag{1.4}$$

The base angle of attack is α_0 and $\Delta\alpha$ is the additional angle between the base angle and the apparent wind vector. Simplifying the formulas with an approximation assuming the tether is straight, therefore assuming $\omega \ll 1$, then the drag is decreased by a negligible percentage from the actual value [7] and the simplified model is assumed an accurate solution.

The induced drag on the tether will significantly affect the system where the tether is traveling faster (i.e. at the point connected to the kite) and not where the tether is relatively stationary at the base. After numerical analysis, Argatov predicted the tether would cause the capacity factor of the kite system to decrease dramatically as the tether length increased [7]. Figure 1.10b contains a graph depicting the results from the analysis. The system analyzed saw a loss of 20% mechanical output when increasing the length of the tether from 400m to 800m.

Figure 1.10a shows the possible catenary orientation for a tethered wing. To



(a) (b)

Fig. 1.10: Tether vector diagram from [7] showing tether catenary behavior and power reduction from increasing tether drag

determine the catenary behavior in the tether, force vectors are calculated along the line, without over-complicating the analysis, by breaking the tether into smaller line segments along the length. By reducing the length of the line segments to infinitely small lengths would result in a perfect representation of the catenary orientation. Another method is to visualize the tether as a length of links (similar to bike chains) pinned together at joints and hinging to accommodate and analyze catenary behavior.

Milan Milutinovic analyzed a similar tethered system as a straight line without catenary behavior by using smaller segments along the cable [17]. Beginning with a simple model with one segment and zero tension, the cord length was compared to the stretched cord length to determine stretch speed to determine spring and damping forces in the cable. For the simple model, the weight is assumed to sit at both ends of the cable rather than distributed along the cable length.

From the simple rope model, Milutinovic modeled a tether with multi-mass dynamics and rested the mass of each segment at the connection point between each segment. Segment 1 started at the airborne unit and increased segment numbers

moved down the cable towards the winch. Each segment was equidistant from the previous segment and contained the same natural length except for the kth segment closest to the winch. This segment is different because it might not be fully released by the winch.

Using the multi-mass rope model, tether drag is calculated with a linear wind profile with respect to altitude and with the equation for aerodynamic drag [17]. Drag force components from the oncoming wind and the cable segment vector provides a force curve along the segment length and total drag force for the segment is found by integrating this curve. Milutinovic ultimately determined that the catenary behavior is accurately modeled with the multi-mass system but the drag is accurately determined using the simple rope method when considering simplicity and computational power.

1.4.4 Industrial Prototypes

Four companies creating prototypes are outlined below. The four encompass the range of systems used by industry to harvest high-altitude wind energy. Altaeros and Sky Wind Power demonstrate static-altitude operation and passively generate wind energy on-board the kite. Makani and Ampyx were chosen based on relevance to Loyd's concept of dynamic-altitude on-board and ground station energy generation.

Fig. 1.11: Altaeros Energies 35ft Diameter Prototype [3]

Altaeros Energies, shown in Fig. 1.11, uses a lighter-than-air vehicle to place a wind turbine directly into the motion of the air stream at higher altitudes than current

wind turbines. The system is constructed from an inflatable balloon structure filled with gases that are lighter-than-air to have buoyancy forces lift the system into higher altitudes [3]. Fixed airfoils are placed on the top and sides for directional stability and wires connected to three points on the lower, frontal face of the vehicle guide the system to achieve steady flight. Steady state is achieved through use of a complex controller algorithm and maneuvering of the three guide-wires. The turbine placed at the center of the vehicle generates energy from the incoming flow of the air stream and transfers the energy down one of the conductive lines that tether the system to a ground station. The turbine, however, does not take advantage of crosswind motion and is limited by the speeds of upper air currents. If deployed to higher altitude, the system can achieve greater energy generation. The system also has the potential to scale to larger dimensions but there must be a balance to keep the buoyancy force greater than the weight of the system.

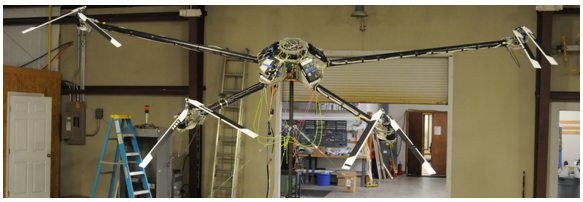


Fig. 1.12: Sky Wind Power Prototype [5]

Sky Wind Power is a stationary unit placed at a specific height to passively use the wind traveling through the four rotary units, as shown in Fig. [5] to generate energy passed down through a high voltage line. Each rotary unit produces lift for the unit to keep the Flying Electric Generator (FEG) at the designated height, and the excess energy is transferred into electrical energy. According to Roberts and contributing authors, the capacity factor of a single FEG averages 80% with factors as high as 90% [9]. Current prototypes demonstrate feasibility with power outputs of 6-15kW. Projections to scale the prototype from a 10ft rotor diameter to 35 ft. would increase power output to 240kW. Power output is calculated using an 18.4 m/s wind speed while the operating height of the unit will reach an altitude 15,000 ft. (4600m). Arrays are possible with the option to lower kites to perform maintenance

and relocate based on direction of seasonal wind patterns.

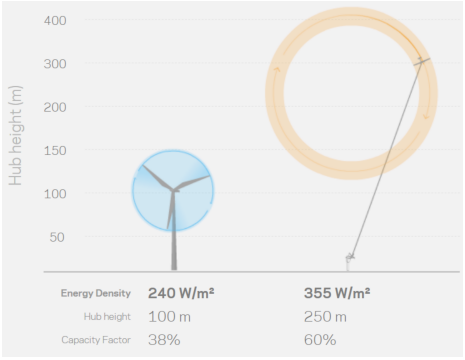


Fig. 1.13: Makani Power Operating Height for downwind tethered kite prototype [4]

Makani Power, a company now owned by Google X, and Ampyx Power, founded by Bas Lansdorp, first created tethered kite systems to produce infinity-curve flight-paths. Makani has since shifted to circular patterns mimicking the paths of Loyd's system in Fig. 1.5a using a tethered kite containing on-board turbines. The system uses crosswind velocities, as outlined in [14], and delivers energy to a ground-based storage device by means of a conductive tether line. Currently, the system is not yet available on the market. The average operating 'hub height' for the Makani Power tethered system is 150% higher than typical land-based wind turbine systems, as shown in Fig. 1.13.

Capacity factor of the system coincides with research and is higher than competing wind turbines. Makani's goal is to reach higher altitude winds, generate more power from these winds, and considerably increase the deployable locations compared to modern wind turbines [4]. The increase in deployable locations is due to the availability of high altitude winds around the globe. Winds below 250m that are sufficient for modern wind turbines are not common and are non-existent in many regions.

Ampyx Power, found in Fig. 1.14, uses mechanical coupling to transfer the kite lift forces into a ground-based turbine where energy is produced on the ground. A rigid wing is launched downwind of the air stream and traces infinity-curve patterns within crosswind velocities. The tether reels-out and pulls on a drum housing the tether to produce energy on the ground until reaching the maximum system tether length [19]. After maximum length is reached, the angle of attack is reduced to reduce lift, thereby reducing power consumption during reel-in, and allowing the wing to glide

back towards the ground station. The difficulty for this particular type of system is to reach a maximum energy generation where the net average cycle power is positive when comparing the energy gain from reel-out and energy loss from reel-in.

1.5 Possible Research Areas of Improvement

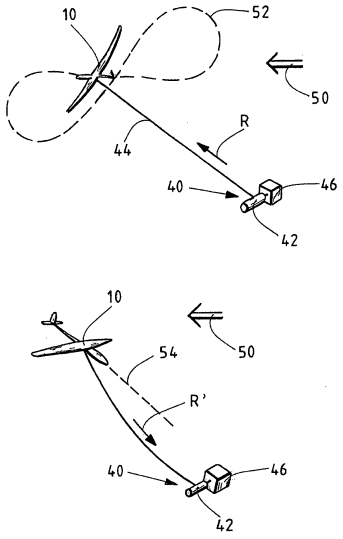


Fig. 1.14: Ampyx Power patent images for infinity kite flightpath pattern and mechanical coupling to a ground energy generation turbine [19]

Loyd left several questions outlined in his paper from the early 1980's. Question such as, "How large can kites be made? What ratios of strength to weight and lift to drag can be achieved? How do the costs vary with such factors? What are the relative site and land-use costs?" [14]. Answers to these questions were attempted and answers hold to current standards regulated by market limitations. The market and researchers have also not decided which systems are best when balancing between simplicity, energy generation, and ecological impact. There is still room for testing new designs and ideas to find better systems.

Significant headway was achieved with improvements to current systems in the areas of material properties, manufacturing capabilities, construction advancements, and types of systems. The rate at which advancement is progressing is indicative that some limitations were previously lifted and more understanding of the topic is now possible. There is also a need to develop newer systems and until a consensus in met for consumers satisfy the need for a practical unit.

To add to the current literature, the proposed research will analyze a new system

to determine if a new configuration is better in regards to energy generation. The simulation will incorporate aspects of flight dynamics, control theory, and newtonian dynamics to model the flight and power generation. The new system, also, does not need a live wire to electrically couple the system to the ground, bypassing significant ecological impact driven concerns.

Chapter 2

Model Description

The 2D System Model is designed to model the behavior of a tethered kite without including altitude. The initial model is necessary to determine if periodic motions are possible and if these periodic motions are capable of producing a net positive energy average. Several assumption were made to model the system with sufficient accuracy to deem the 2D system feasible. The assumptions are listed in Table 2.1.

Table 2.1: Assumptions to simplify model to determine system feasibility and simplify areas to reduce numerical computation

	Simulation Assumptions
1	Kite is a Point Mass
2	Wind has constant velocity and direction with uniform profile
3	Power production is approximated by mechanical power
4	Well established airflow over the wing
5	2D Coefficients of lift and drag from experiments in [20]
6	Use Anderson's Method to Transform 2D c_L, c_D to 3D C_L, C_D
7	Coefficients of lift and drag are at a Reynold's number of 10^6
8	Moment coefficients are neglected (does not contain a bridle)
9	Finite wing with fixed geometry
10	Air density is $1.225\frac{kg}{m^3}$
11	Air viscosity is $1.79(10^{-5})$ $\frac{kg}{m \cdot s}$
12	Flightpath is Restricted to 2D plane
13	Angle between tether and chordline of the wing (β) is constant
14	Tether cannot twist
15	Tether is straight and massless
16	Tether can support non-zero compression forces but this is monitored and results in a non-viable solution
17	Tether drag is neglected

2.1 Flight Dynamics

Equations of motion were derived to serve as the state functions for a numerical integration sequence in MATLAB. The system is based on a point mass with three forces acting on the mass. Equation (2.1) states that the sums of tension, drag, and lift forces multiplied by their respective unit vectors are equal to the mass of the kite multiplied by the acceleration of the kite.

$$-T\hat{e}_r + D\hat{\lambda}_D + L\hat{\lambda}_L + F_p\hat{e}_\theta = m_{kite}\vec{a}_{kite} \tag{2.1}$$

T is the tether tension; D is the magnitude of Drag; $\hat{\lambda}_d$ is the unit vector of drag; L is the magnitude of lift; $\hat{\lambda}_l$ is the unit vector of lift; F_p is an optional propulsion force; $\hat{e}_\theta$ is the rotational coordinate system unit direction; m is the mass of the kite; and $\vec{a}_{kite}$ is the acceleration of the kite, as shown in Fig. 2.1.

Figure 2.2 shows the Free Body Diagram of the kite. The forces include lift, drag, and tether tension. Figure 2.3 contains the velocity vectors from the wind, the kite, and the apparent wind velocity vector from Eq. (1.2) along with rotational and polar coordinate systems.

Equation (2.2) is the radius vector ($\vec{r}_{P/O}$), or kite position vector, of the tether from the base drum (Point O) to the mass in the air (Point P), shown in Fig. 2.1, with l as the length of the tether and $\hat{e}_r$ as the directional vector of the rotational coordinate system.

$$\vec{r}_{P/O} = l\hat{e}_r \tag{2.2}$$

Velocity and acceleration of the kite are found by taking the first and second derivatives of the kite position vector with respect to time, $\frac{\partial \hat{r}}{\partial t}$ and $\frac{\partial^2 \hat{r}}{\partial t^2}$ respectively.

$$\vec{V}_{kite} = \dot{l}\hat{e}_r + l\dot{\theta}\hat{e}_\theta \tag{2.3}$$

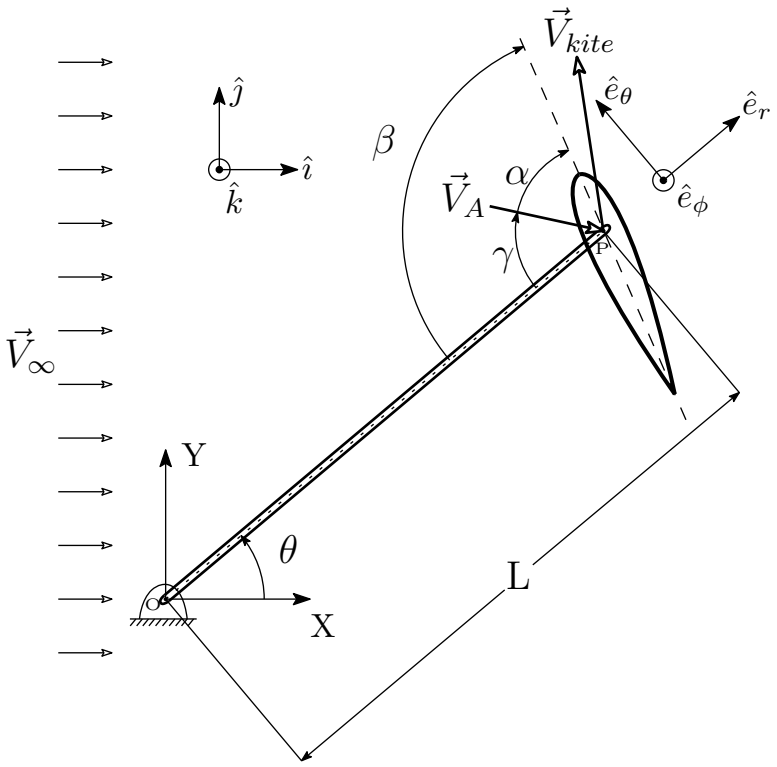


Fig. 2.1: Overall System Schematic for 2D model

$$\vec{a}_{kite} = (2\dot{\theta}\dot{l} + \ddot{\theta}l)\hat{e}_\theta + (-\dot{\theta}^2 l + \ddot{l})\hat{e}_r \tag{2.4}$$

$\dot{\theta}$ is the angular velocity of the tether; $\ddot{\theta}$ is the angular acceleration of the tether; $\dot{l}$ is the rate of change of the tether length, and $\ddot{l}$ is the acceleration of tether length.

By defining the apparent wind velocity vector, the unit vectors for lift and drag can be found. The apparent wind velocity is found in Section 1.4.1, Eq. (1.2).

The unit drag vector is found by dividing the apparent wind velocity vector by its magnitude since the vector for drag is parallel with the apparent wind velocity vector. The unit vector of drag, $\hat{\lambda}_D$ is shown in Eq. (2.5).

$$\hat{\lambda}_D = \frac{\vec{V}_A}{|\vec{V}_A|} = \frac{V_\infty[cos(\theta)\hat{e}_\theta + sin(\theta)\hat{e}_r] - \dot{\theta}l\hat{e}_\theta + \dot{l}\hat{e}_r}{|V_\infty[cos(\theta)\hat{e}_\theta + sin(\theta)\hat{e}_r] - \dot{\theta}l\hat{e}_\theta + \dot{l}\hat{e}_r|} \tag{2.5}$$

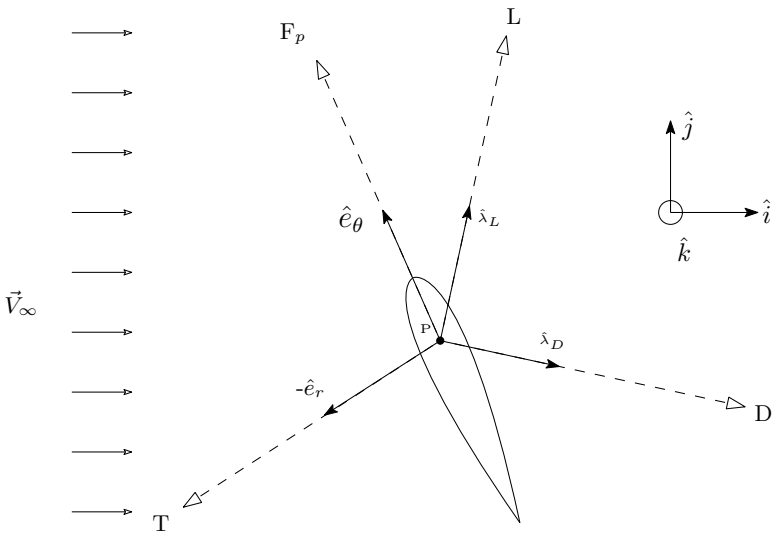


Fig. 2.2: Wing Free Body Diagram with Velocity Vectors

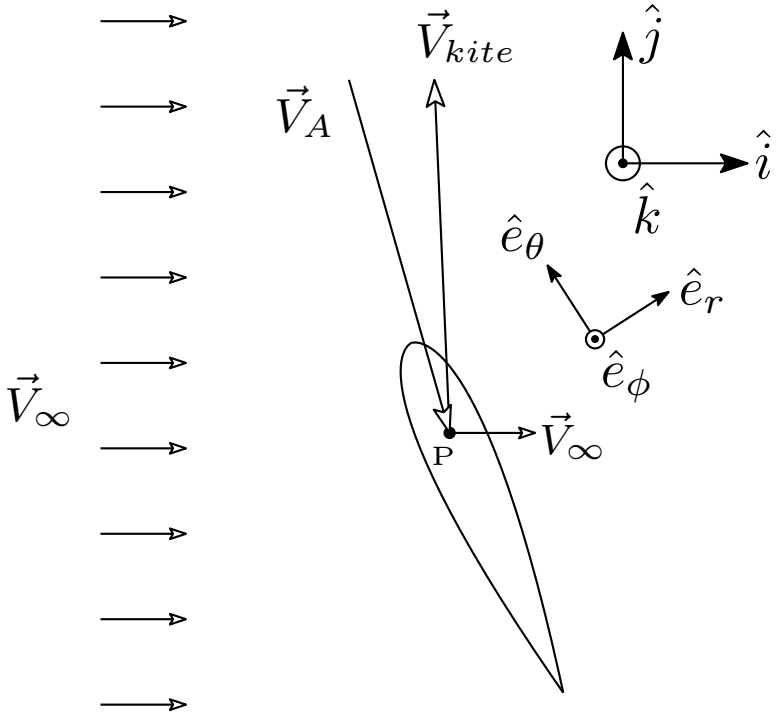


Fig. 2.3: Velocity Vectors and Coordinate System Unit Vectors

Because lift and drag are orthogonal, the unit vector for lift can be found by the cross product of $\hat{k}$ and $\hat{\lambda}_D$ and dividing by its magnitude.

$$\hat{\lambda}_L = \frac{\hat{k} \times \hat{\lambda}_D}{|\hat{k} \times \hat{\lambda}_D|} \tag{2.6}$$

Magnitudes of lift and drag are dependent upon aerodynamic derivations shown in Eq. (2.7),

$$L = \frac{1}{2} S\rho(V_{app})^2 c_L \qquad D = \frac{1}{2} S\rho(V_{app})^2 c_D \tag{2.7}$$

where S is the effective area of the kite, ρ is the density of air $=1.225\frac{kg}{m^3}$ (Wind Power Data Standard) [16], c_L is the coefficient of lift, and c_D is the coefficient of drag. Equation (2.7) determines the coefficients for a wing of infinite length. The kite lift and drag coefficients are found in [20] for a 0-360^o range in angles of attack at 10^o intervals after the air flow as been well established.

Equation (2.8) and (2.11) adjust the infinite wing coefficients of lift and drag based on an induced angle of attack (α_i) [6]. Downwash, the downward flow of air caused by the kite traveling through the fluid, causes the geometric angle of attack to change slightly and α_i is the corrected angle of attack. To correct the for the finite wing,

$$C_l = a(\alpha - \alpha_{L=0}) \tag{2.8}$$

where a is the slope of the lift curve for a finite wing, α is the angle of attack for the current orientation of the kite, and $\alpha_{L=0}$ is the angle of attack when lift is equal to 0.

$$a = \frac{a_0}{1 + \frac{180 a_0}{\pi^2 e_1 AR}} \tag{2.9}$$

with a_0 as the slope of the lift curve for an infinite wing, e_1 is a coefficient based on the geometry of the kite called the Oswald Efficiency Factor, and

$$AR = \frac{b^2}{S} \tag{2.10}$$

is the aspect ratio between the kite length and width. In Eq. (2.10), b is the length (m) and S is the surface area of the wing, which is $S = b * c$ where c is the width of the kite (m). Using C_L,

$$C_d = c_d + \frac{C_L^2}{\pi e_1 AR} \tag{2.11}$$

the corrected drag coefficient, C_D, is found for a finite wing length. Substituting into (2.7)

$$L = \frac{1}{2} S\rho(V_{app})^2 C_L \qquad D = \frac{1}{2} S\rho(V_{app})^2 C_D \tag{2.12}$$

will produce the corrected lift and drag forces for a finite wing.

2.2 Numerical Integration and Equations of Motion

2.2.1 Integration Tolerance Convergence

The tolerance convergence is set to determine the integration tolerance level acceptable to calculate accurate results from the numerical solver. Different minimum tolerances will calculate different results that are inaccurate up to a minimum tolerance level.

The convergence plot in Fig. 2.4 shows the numerical integration value for angular velocity ($\dot{\theta}$) with varying tolerance factors. Once the system reached a tolerance factor of 10^{-6}, the integration provided the same result for tolerance factors below the minimum tolerance. The convergence plot both verifies that accurate results are found and saves computer processing time by allowing the computer to use a higher minimum tolerance when deemed capable to still provide accurate results.

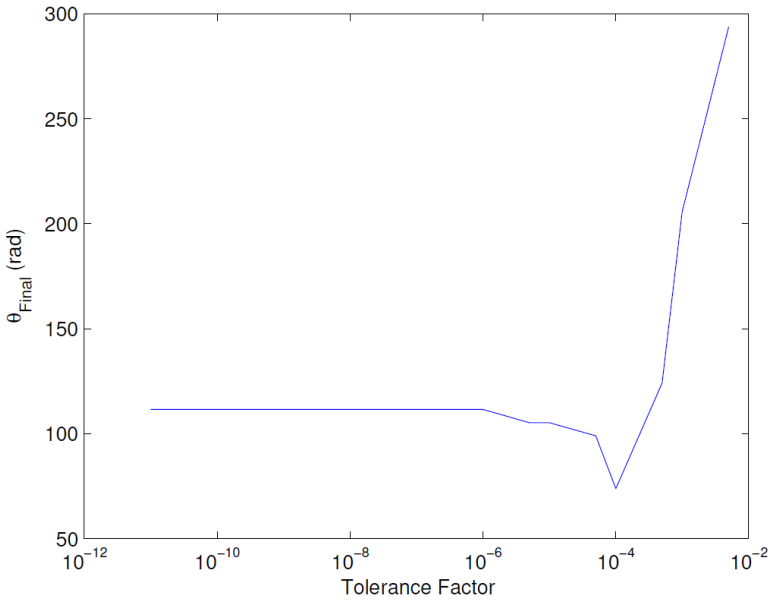


Fig. 2.4: Tolerance convergence plot for the solution of angular position (θ) after one pass through the ODE45 solver during the reel-in phase to determine minimum solver tolerance

The simulations in this paper are at a tolerance level of 10^{-10}, because the simulation is fast enough to allow a smaller minimum tolerance level.

2.2.2 Reel-Rate Step Controller

The ODE45 Solver will calculate subsequent numerical solutions using a state space technique. The equation for angular acceleration of the kite, Eq. (2.13) is used to integrate the next time step in the series provided by the previous time-step value as an initial condition for the lower-order derivative.

Equations (2.13), (2.14), and (2.15) were derived from a symbolic solver in MATLAB outlining the derivations in this chapter. The simulation code is found in Appendix A.3.1.

$$\ddot{\theta} = \frac{1}{ml}\left[F_p - 2m\dot{l}\dot{\theta} - \frac{L\dot{l} + Dl\dot{\theta} + V_\infty(D\sin\theta - L\cos\theta)}{\sqrt{(\dot{l}\sin\theta + l\dot{\theta}\cos\theta)^2 + (V_\infty - \dot{l}\cos\theta + l\dot{\theta}\sin\theta)^2}}\right] \tag{2.13}$$

Numerically integrating angular acceleration provides angular velocity, and integrating angular velocity provides the angular position.

The same process is used to calculate tether length (l). Tether length is increased based on a constant reel-rate set as an initial condition. The second order derivative, tether length acceleration, is set to zero, and the first order derivative, tether length rate of change, is set to a positive integer for reel-out and a negative integer for reel-in. The tether length will then increase as the integral of the tether length rate of change. For the tension controller, the tether length acceleration is used to change the tether length rate of change and is discussed further in Section 2.2.3.

After the integration for kite position and tether length, the tether tension, Eq. (2.14), is calculated from the state variables. The tether tension is not an integral process and therefore must be calculated aside from the state variables.

$$T = ml\dot{\theta}^2 - m\ddot{l} + \frac{L l\dot{\theta} - D\dot{l} + V_\infty(D\cos\theta + L\sin\theta)}{\sqrt{(\dot{l}\sin\theta + l\dot{\theta}\cos\theta)^2 + (V_\infty - \dot{l}\cos\theta + l\dot{\theta}\sin\theta)^2}} \tag{2.14}$$

2.2.3 Tether Tension Step Controller

Another method for controlling the system is to specify a tension for the reel-in and reel-out phases. In a physical system, this could be achieved by use of a closed-loop controller for line tension. Airfoil properties, airborne dynamic equations, and free body diagram remain the same for analysis. However, the reel-in and reel-out rates are no longer constant values and must update based on tether tension, found in Eq. (2.15).

$$\ddot{l} = \frac{1}{m}\left[ml\dot{\theta}^2 - T + \frac{L\dot{l} + Dl\dot{\theta} + V_\infty(D\sin\theta - L\cos\theta)}{\sqrt{(\dot{l}\sin\theta + l\dot{\theta}\cos\theta)^2 + (V_\infty - \dot{l}\cos\theta + l\dot{\theta}\sin\theta)^2}}\right] \tag{2.15}$$

2.3 Steady State Convergence

The simulation will assume steady, periodic motion when the system error satisfies the steady state (SS) criteria. The SS_{error}, absolute percent difference between the tether length (l) at the end of the wave and at the beginning of the wave summed with the percent difference between the tether angular velocity ($\dot{\theta}$) at the end of the wave and at the beginning of the wave, must converge to a value less than the convergence criteria (ε). ε is a predefined percent difference $< (10^{-5})$.

$$SS_{error} = \left| \frac{l_{waveend} - l_{waveinit}}{l_{waveinit}} \right| + \left| \frac{\dot{\theta}_{waveend} - \dot{\theta}_{waveinit}}{\dot{\theta}_{waveinit}} \right| \leq \varepsilon \tag{2.16}$$

2.4 Power Generation

To calculate the mechanical power generated during a single cycle, the tether tension and velocity profiles are needed. Tension from a single periodic cycle is used to find the net average energy generated per second. First, an instantaneous power (J) curve for the cycle is calculated by multiplying the tension of the tether during a cycle and the rate of reel-out or reel-in at each unit of time during the cycle.

$$P_{instant} = T\dot{l} \tag{2.17}$$

The integral over time of the instantaneous power curve is the net energy generated per cycle (J) in periodic motion, and, if this value is positive, the system is generating more energy in the reel-out phase than consuming in the reel-in phase. The integral is approximated with a trapezoidal approximation MATLAB function (trapz).

$$E = \int_{t_0}^{t_f} P_{instant} dt \tag{2.18}$$

Net average energy generated per second, power (W), is calculated by dividing

the average energy per cycle by the amount of time to complete the cycle.

$$\bar{P} = \frac{E}{t_f - t_0} \tag{2.19}$$

2.5 Root Finding Methods

The Multidimensional Newton-Rhapson Method [18] is a mathematical solver to find solutions to functions determined by the user. The two functions found in Eq. (2.20) are the convergence equations modified to reduce the error of $\dot{\theta}$ to 0 rad/s or l to 0 m. Defining the before and after waves difference as 0 m or 0 rad/s will find periodic motions in the system model.

$$F(\dot{\theta}) = \dot{\theta}_{waveend} - \dot{\theta}_{waveinitial} \qquad G(l) = l_{waveend} - l_{waveinitial} \tag{2.20}$$

The standard simulation is the numerical integrating main file which allows the dynamics of the system to control behavior if an experimental unit is created. Initial conditions are given to the standard simulation to observe the behavior of the system after a predetermined amount of time. For root finding purposes, one cycle is performed with measurements for $\dot{\theta}$ and l at the beginning and end of the cycle.

After the standard simulation differences are calculated, the initial conditions are altered by a small Δ to find the slope change in function differences. A one-dimensional root find uses either derivatives found in Eqs. (2.21) and (2.22) depending which initial condition is governing the root find.

$$\dot{\theta}_{der} = \frac{F(\dot{\theta} + \Delta\dot{\theta}) - F(\dot{\theta})}{\Delta\dot{\theta}} \tag{2.21}$$

$$l_{der} = \frac{G(l + \Delta l) - G(l)}{\Delta l} \tag{2.22}$$

The definition of the derivative is then used in Eqs. (2.23) and (2.24) to calculate the next value closer to a solution of Eq. (2.20). Each calculation of the root find will place the root find closer to the solution and will stop when a minimum tolerance within the range of a solution is met, typically 10^{-7} or 10^{-9} for Eq. (2.20).

$$\left[x^*_{\dot{\theta}}\right] = \frac{-1}{\dot{\theta}_{der}}\left[\dot{\theta}_{waveend} - \dot{\theta}_{waveinitial}\right] + \left[\dot{\theta}_{waveinitial}\right] \tag{2.23}$$

$$\left[x^*_{l}\right] = \frac{-1}{l_{der}}\left[l_{waveend} - l_{waveinitial}\right] + \left[l_{waveinitial}\right] \tag{2.24}$$

The multidimensional root find uses a Jacobian matrix in lieu of single variable derivatives. The Jacobian specifically for the $\dot{\theta}$ and l root find is located in Eq. (2.25).

$$J = \begin{bmatrix} \frac{F_{\Delta\dot{\theta}}(\dot{\theta}) - F(\dot{\theta})}{\Delta\dot{\theta}} & \frac{F_{\Delta l}(\dot{\theta}) - F(\dot{\theta})}{\Delta l} \\ \frac{G_{\Delta\dot{\theta}}(l) - G(l)}{\Delta\dot{\theta}} & \frac{G_{\Delta l}(l) - G(l)}{\Delta l} \end{bmatrix} \tag{2.25}$$

Each derivative inside the Jacobian uses the same definition of a derivative as Eqs. (2.23) and (2.24), but it mixes the numerators and denominators as outlined in [18]. The function differences for each beginning and end cycle for the Jacobian are found in Eqs. (2.26) and (2.27).

$$F_{\Delta\dot{\theta}}(\dot{\theta}) = \dot{\theta}_{\Delta\dot{\theta}_{waveend}} - \dot{\theta}_{\Delta\dot{\theta}_{waveinitial}} \qquad F_{\Delta l}(\dot{\theta}) = \dot{\theta}_{\Delta l_{waveend}} - \dot{\theta}_{\Delta l_{waveinitial}} \tag{2.26}$$

$$G_{\Delta\dot{\theta}}(l) = l_{\Delta\dot{\theta}_{waveend}} - l_{\Delta\dot{\theta}_{waveinitial}} \qquad G_{\Delta l}(l) = l_{\Delta l_{waveend}} - l_{\Delta l_{waveinitial}} \tag{2.27}$$

Once the Jacobian is calculated it is placed into Eq. (2.28), inversed, and used to calculate the solution to Eq. (2.20).

$$\begin{bmatrix} x^*_{\dot{\theta}} \\ x^*_l \end{bmatrix} = -J^{-1} \begin{bmatrix} \dot{\theta}_{waveend} - \dot{\theta}_{waveinitial} \\ l_{waveend} - l_{waveinitial} \end{bmatrix} + \begin{bmatrix} \dot{\theta}_{waveinitial} \\ l_{waveinitial} \end{bmatrix} \tag{2.28}$$

2.6 Simulation Verification

Several reality checks, simple test cases, were used to determine if the simulation is operating within physical limitations, such as, static behavior, ball and string, zero wind conditions, external propulsion, Reynold's number, and zero value reel-out/reel-in rates and phase regions.

TEST CASE 1: The simulation remains static while there is a lack of wind and the initial angular velocity is set to zero. Without wind, there is also a lack of lift and drag forces. Figure 2.5 shows the system without wind and with an initial angular velocity equal to 0 m/s. The wing, blue asterisk, stays in the same place and remains

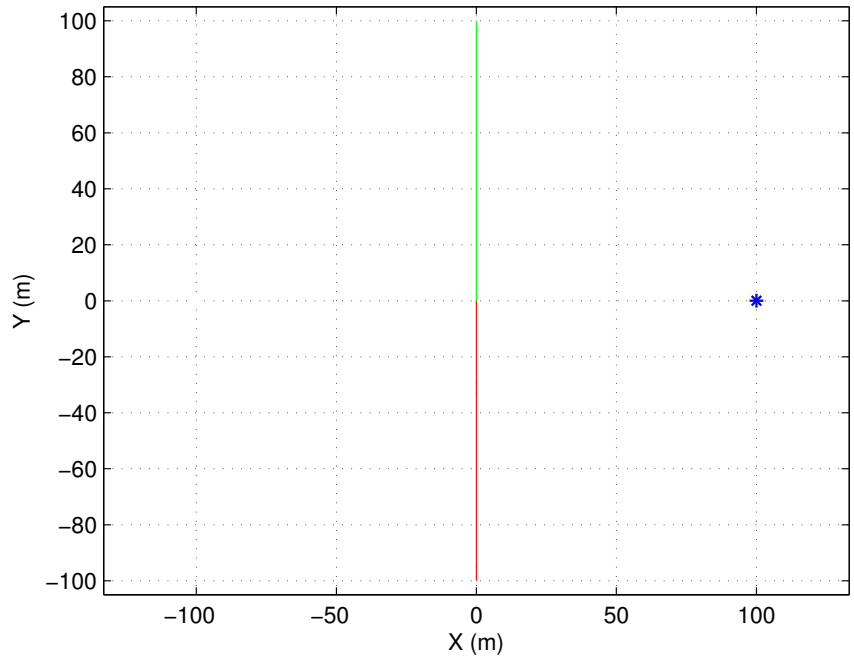


Fig. 2.5: Simulation verification to determine behavior with the wind speed at 0 m/s and 0 m/s initial angular velocity

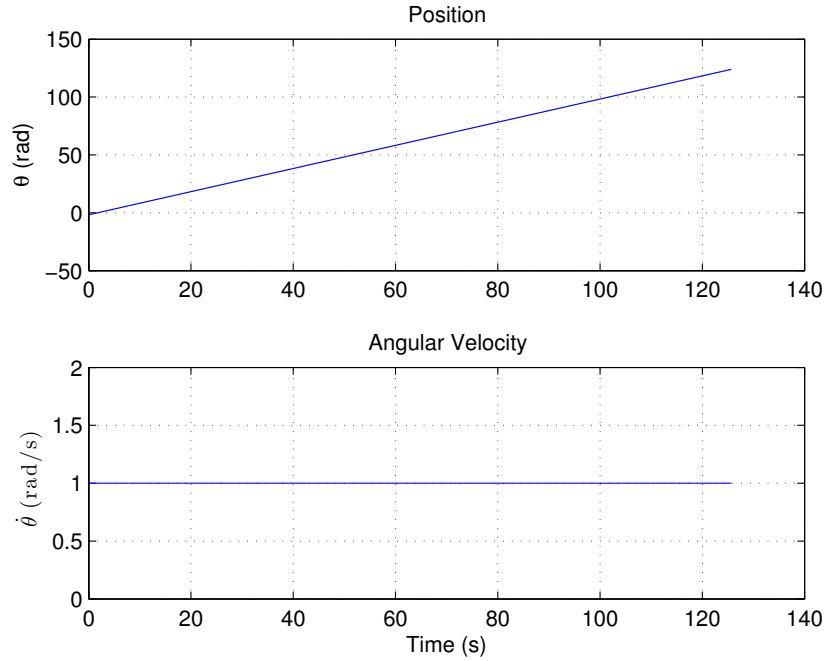


Fig. 2.6: Simulation verification to determine behavior with the wind speed at 0 m/s, >0 m/s initial angular velocity, and lacking lift or drag forces

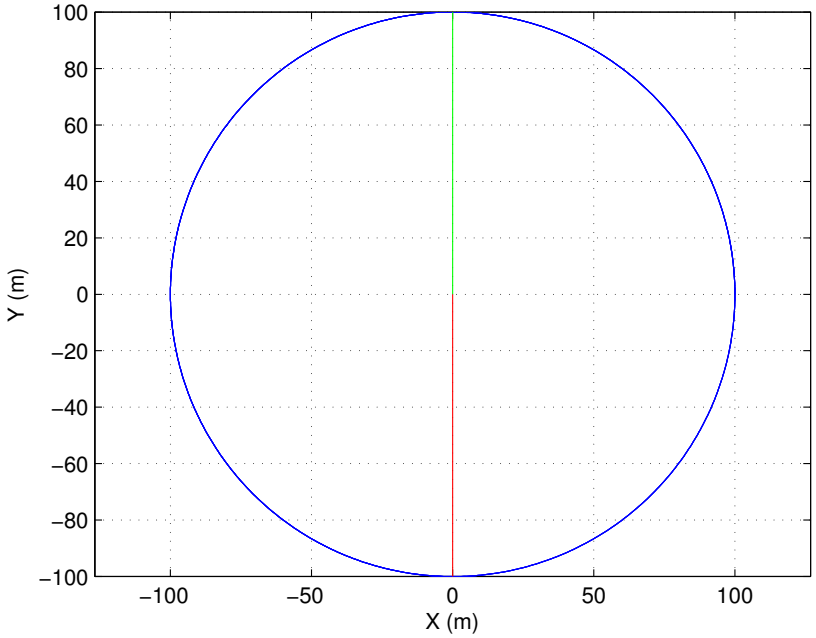


Fig. 2.7: Simulation verification to determine transient behavior with the wind speed at 0 m/s, >0 m/s initial angular velocity, and lacking lift or drag forces

at a constant angular velocity equal to 0 m/s. The green and red lines indicate the reel-in and reel-out trigger, respectively.

TEST CASE 2: The ball and string scenario represents the system with an initial angular velocity, lacking lift and drag forces, and the reel-rates equal to 0 m/s. In the top graph of Fig. 2.6, the angular position of the wing consistently increases, showing a constant rotation of the kite around the origin. The bottom graph is the angular velocity and shows the wing rotating with a constant angular velocity similar to a system with zero gravity and zero external forces. The top view of the kite path for the ball and string scenario is found in Fig. 2.7 which shows the flightpath of the wing. The length remains constant with the reel-in and reel-out rates equal to 0 m/s. The system also shows that the 2 set ODE45 solvers are working correctly with the reel-in and reel-out phase triggers to produce a full rotation about the origin.

TEST CASE 3: If the drag coefficient is non-zero, the system should lose energy, and, as time increases, the system should decay to a zero angular velocity. Figure 2.8 shows the system with an initial angular velocity with the wind speed at 0 m/s. In the top graph, the angular position increases and very slowly becomes constant while the angular velocity of the kite decays towards zero as time approaches infinity. The angular velocity never becomes equal to zero because drag is the only force causing the system to slow down since drag is dependent upon the speed of the kite. Therefore, the angular velocity will converge upon zero but never equal zero. Unfortunately, since drag is the only counteracting force and it's magnitude is correlated to $(\vec{V}_a)^2$, the system needs a large amount of time to slow. The check verifies that the wing will eventually stop while there is a lack of wind.

TEST CASE 4: The reel-in and reel-out rate scenario, found in Fig. 2.9 and 2.10, show the system with reel-in set to -10 m/s, reel-out set to 10 m/s, and with lift and drag forces set to 0 N. The top graph of Fig. 2.9 shows the tether length as it is triggered between the two phases and corresponds with the bottom graph by reeling

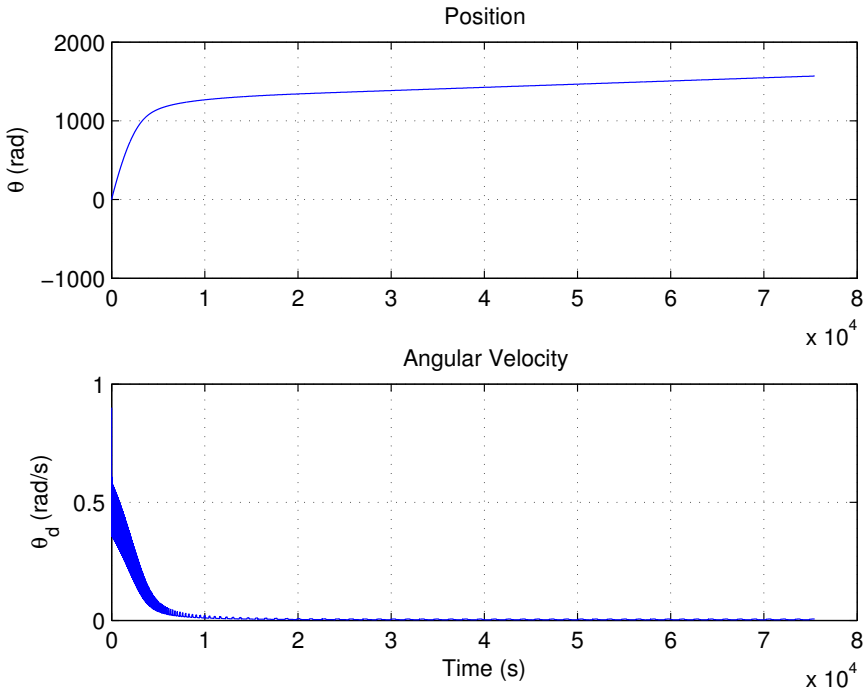


Fig. 2.8: Simulation verification to determine angular velocity behavior with the wind speed at 0 m/s

out when set to reel-out and reeling-in when set to reel-in. The value of reel-in and reel-out when multiplied by the time spent in the phase also equals the correct length. Figure 2.10 shows the two semi-circular paths during motion. The length differences are equal due to the equal time spent in each phase and the symmetrical reel-out triggers. The triggers are therefore shown to operate as expected which begins with the reel-out phase at -90^o, or an angular equivalent of the reel-out trigger (360^{o}n with n as the cycle number), and continues to shift between phases at the correct trigger locations.

TEST CASE 5: Along with the zero wind velocity check, an external force was applied to the wing along the $\hat{e}_\theta$ unit vector to simulate propulsion from the wing. This force would tend to increase the speed of the kite. Figure 2.11 shows the angular velocity ($\dot{\theta}$) with the external force equal to 0 N, and Fig. 2.11 shows the angular velocity with an external force equal to 200 N. Without the external force, the angular

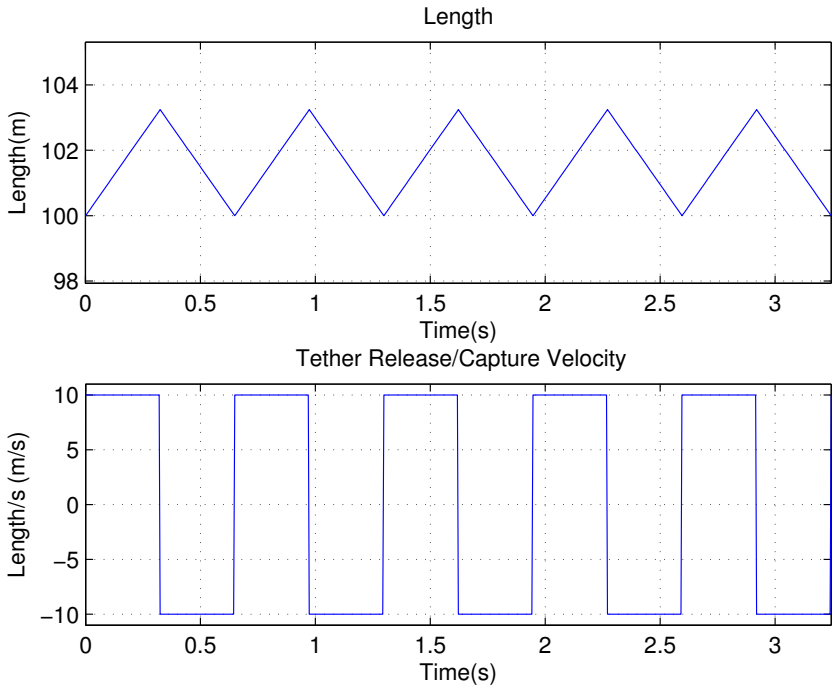


Fig. 2.9: Simulation Verification to verify reel-in and reel-out rates/triggers perform as expected

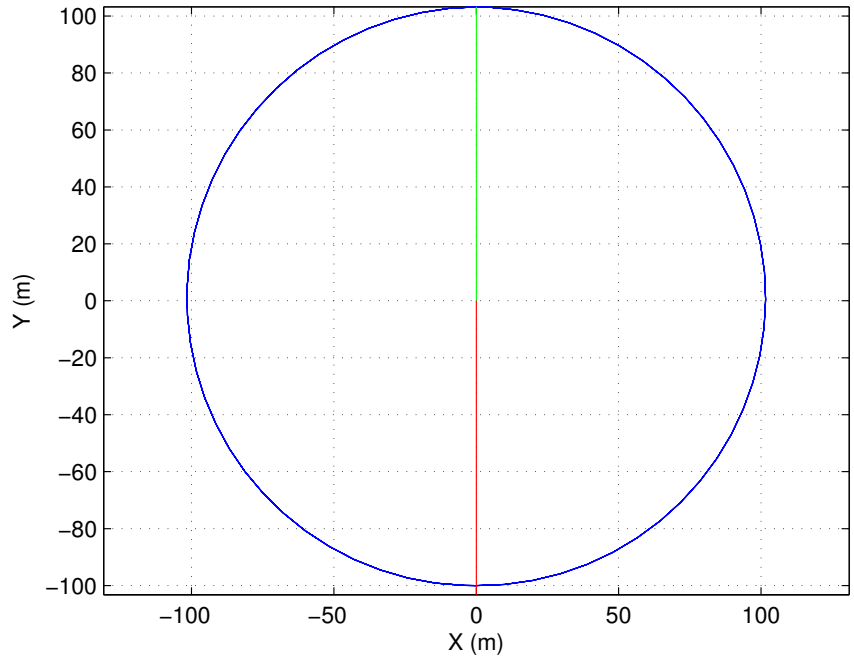


Fig. 2.10: Simulation Verification to verify reel-in and reel-out rates/triggers transient performs as expected

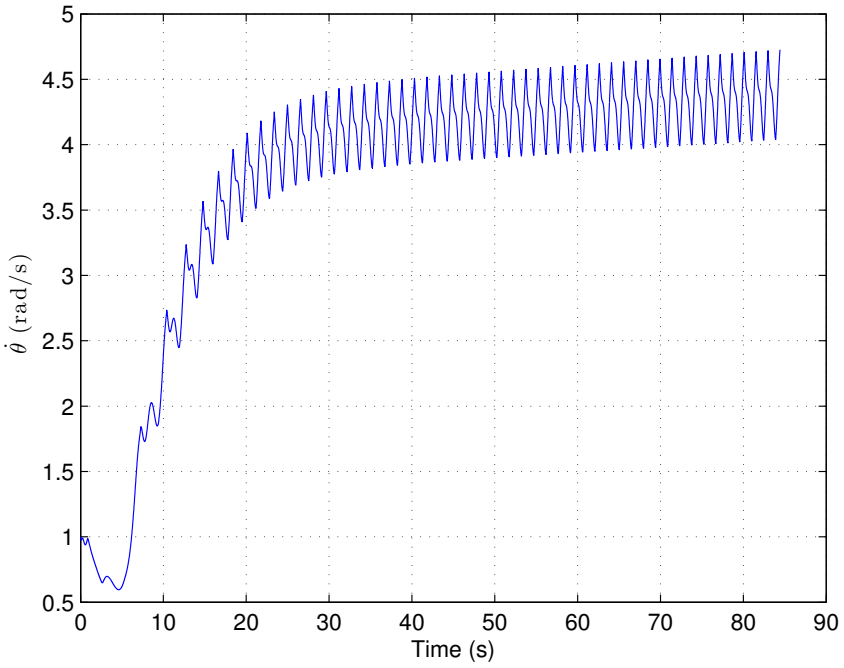


Fig. 2.11: Simulation Verification with a 0 N force applied to the wing

velocity converges to an average of 4.8 m/s and will converge to and average of 22 m/s with the external force. This shows that the angular velocity and position are logically adjusting due to external factors.

REYNOLD'S NUMBER ASSUMPTION CHECK: The Reynold's number curve for a typical solution, found in Fig. 2.13, ranges between $2\text{x}10^5$-$15\text{x}10^5$. The coefficients of lift and drag used from [20] are valid between the ranges of 10^4-10^7 with the data specific coefficients in the described model as $10\text{x}10^5$. The straight line in Fig. 2.13 reflects the Reynold's number of the coefficients of lift and drag from the Sandia National Labs experiments [20]. The exponential curve leading towards a stabilized Reynold's number is the Reynold's number from the simulation and is contrasted by the assumed Reynold's number at $10\text{x}10^5$. Therefore, the Reynold's Numbers achieved in the model are different than the Reynold's Number used in coefficients of lift and drag. Yet, the coefficients are assumed sufficiently accurate for

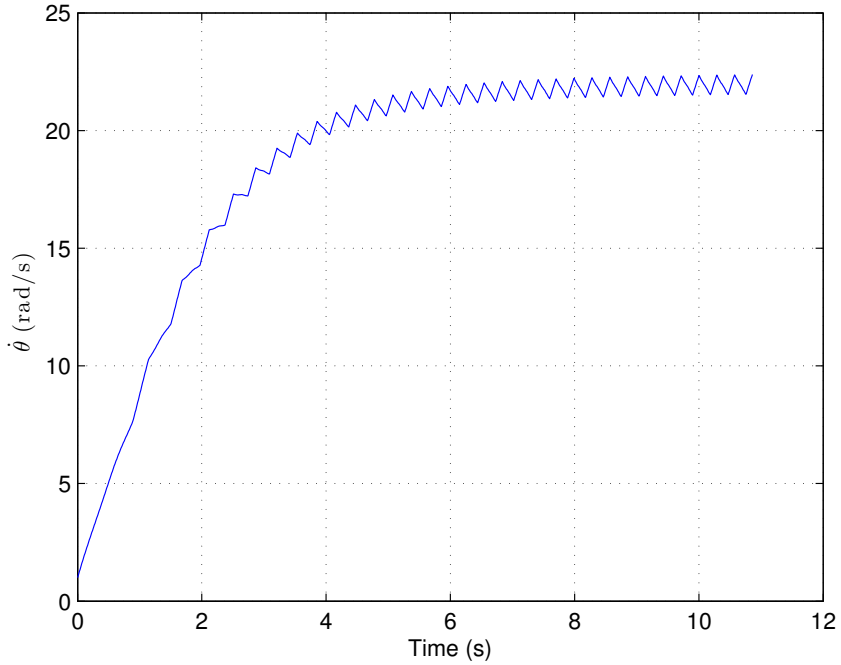


Fig. 2.12: Simulation Verification with a 200 N force applied to the wing

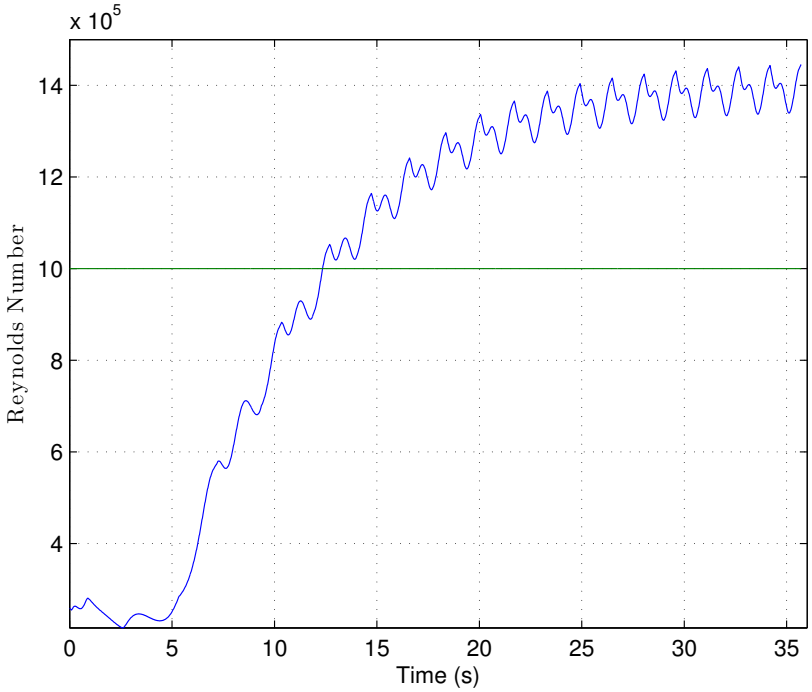


Fig. 2.13: Simulation Verification that Reynold's number is within approximate range for data set used from [20]

simulation purposes compared to adding an additional interpolation for data sets with $1x10^5$ or $20x10^5$ since faster simulation calculations outweighs the benefit of slightly more accurate coefficients (since the 2D model approach is largely insufficient when considering a detail such as the Reynold's number inconsistencies).

FORCE VECTORS DIRECTION CHECK: The force vectors for lift and drag were verified using an animation window. The airfoil shape in Fig. 2.14 shows the kite orientation based on the angular position of the tether and the beta angle between the tether and the kite chord line. The three lines signify the apparent wind velocity vector (cyan), the lift force vector (green), and the drag force vector (red). Each vector is a ratio of the current value divided by the largest value to scale in that simulation is between 0 and 1. The animation shows that the drag force is in the same direction as the apparent wind velocity vector, the apparent wind vector is pointing into the kite and the drag vector is pointing away from the kite, while the lift vector is orthogonal to the drag vector. Several initial conditions were evaluated to verify forces and apparent wind velocities, such as, 90^o, 180^o, 270^o, and 360^o to check behavior with and without reel-rates equal to or greater than 0. Each condition verified the vectors were in the appropriate directions.

The tether is shown to follow the same trajectory as the kite while the animation is progressing as shown in Fig. 2.15. The tether tip location corresponds to the location of the tether and kite connection point and follows the same path. Axes are shown in meters. The red line indicates compression in the tether and visually shows the violation of a solution.

2.7 Conclusion

The test cases were designed to test for inconsistencies and determine flaws in the 2D model or simulation behavior. With the simulation behaving consistently with real world validity checks, the simulation is assumed appropriate to model the behavior

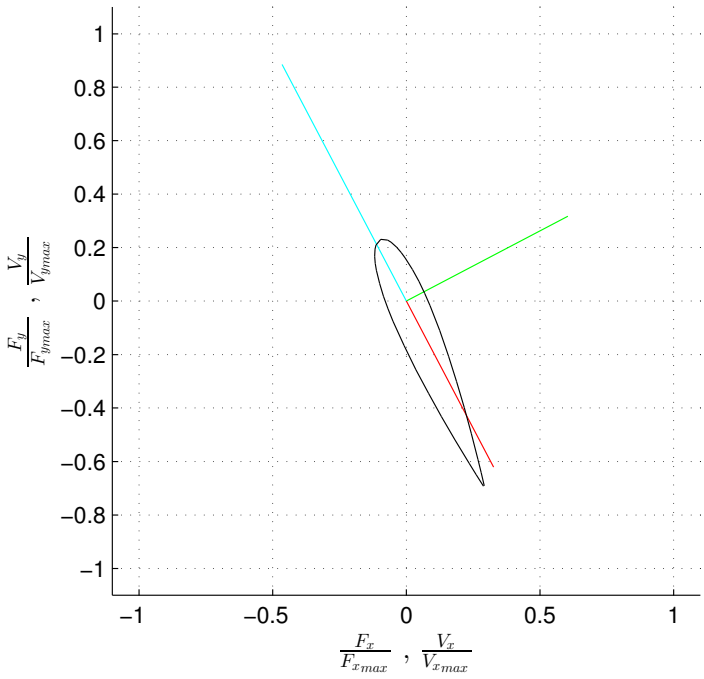


Fig. 2.14: Simple animation to verify lift and drag vectors by post-processing the simulation

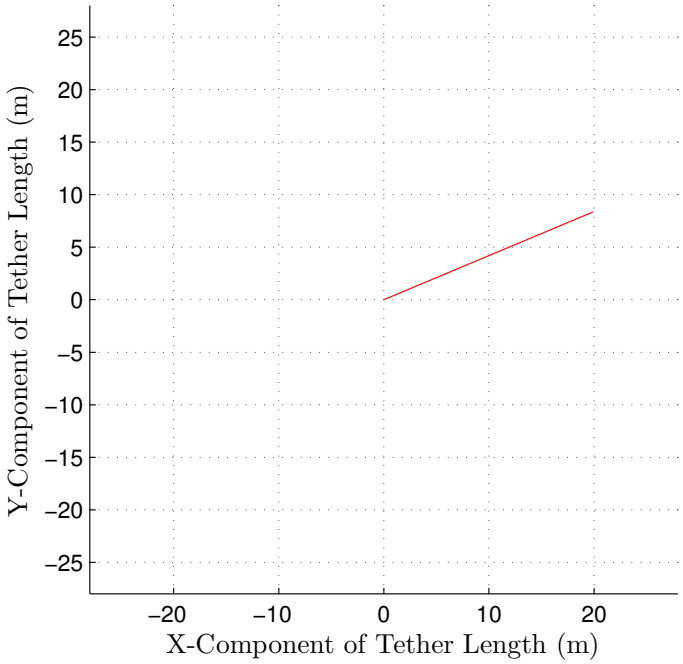


Fig. 2.15: Simple animation to verify tether orientation by post-processing the simulation

of the tethered kite system and provide basic observations of system feasibility.

Based on assumptions and simplicity of the model, the real world viability of the system for a 3D application cannot be proven with the 2D model. The system does not consider kite moments, tether twist, or variable wind speeds. Therefore, this model can only provide a basic level understanding for the behaviors of a tethered kite system rotating upwind of a base station and using a crosswind velocity vector to have the wind assist with kite reel-in.

Chapter 3

2D Model Behavior

3.1 Periodic Motion

3.1.1 Basic Periodic Solution

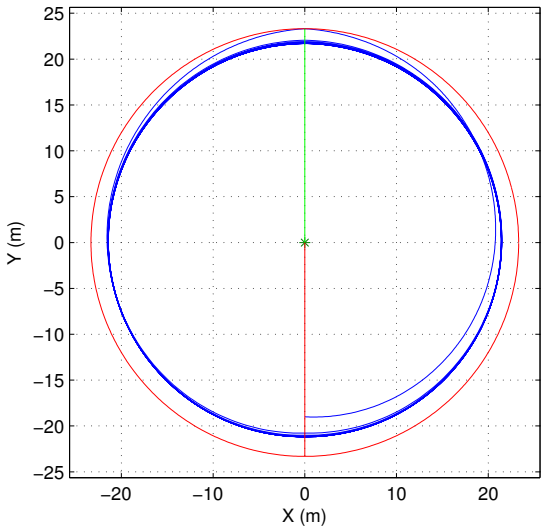
The purpose of this system is to achieve periodic flight around a base station and to use the wind for two purposes. The main purpose is to push the wing and tether downwind for energy generation and then to use the wind to push the system towards the base station while upwind. The wind will assist with reeling-in the wing/tether combination to reduce the energy consumed during reel-in. The 2D model behavior is observed through a standard simulation which begins with initial conditions that start the transient phase, and, depending upon system parameters, will approach steady state behavior.

The set variable values in Table 3.1 are specifically set to the constant values shown and are unchanged for all simulation results unless specifically stated otherwise.

Figure 3.1 depicts the transient response leading to steady state operation of a

Table 3.1: Simulation parameter constants for the results of parameter set unless specifically stated otherwise

$V_\infty = 10$ m/s	c = 0.2 m	$\beta = 90^o$
m = 1.2 kg	b = 1.2 m	$F_p = 0$ N



$\theta_{in} = 90^o$	$\dot{\theta} = \frac{57.3^o}{s}$	$\dot{l}_{in}$ = -1 m/s
$\theta_{out} = 270^o$	l = 19m	$\dot{l}_{out}$ = 1 m/s

Fig. 3.1: Transient Behavior leading to Steady State Operation of the Proposed Kite System in a parallel trigger orientation

system with reel-in and reel-out event triggers parallel at $\theta = 90^o$ + n·360^o and $\theta = 270^o$ + n·360^o (with n as the cycle number). The base station is located at (X = 0,Y = 0) and is Point O in Fig. 2.1, about which the wing and tether rotate. The larger circle is drawn at the max length of the system. The circle provides a reference to see the flightpath is not completely circular and has varying length throughout flight.

The region from θ = -90^o to 90^o defines the reel-out phase while the region from $\theta = 90^o$ to 270^o defines the reel-in phase shown in Fig. 1.2. The transient response is shown with blue lines leading towards a darker blue, semi-circular region showing the steady state operation of the system. This particular system is a viable solution by satisfying the convergence criteria discussed in Section 2.3 and the positive tension criteria. The blue circular flightpath defines the location of the wing at a given angle and tether length during the flight of the system. As the wing rotates about the base point, the length of the tether increases during reel-out and decreases during reel-in

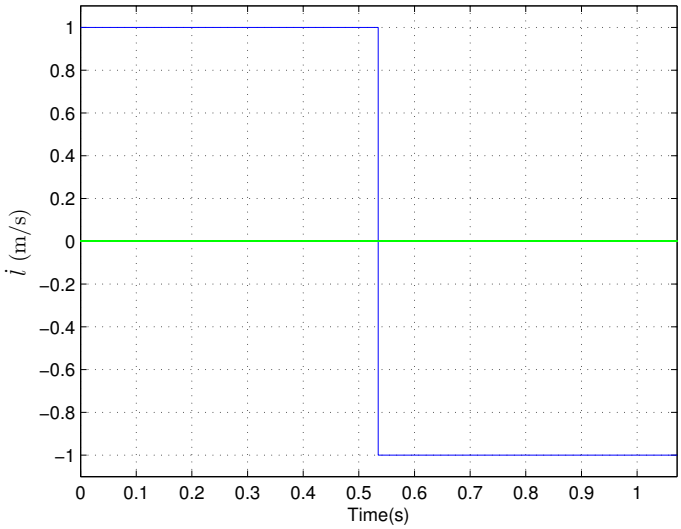


Fig. 3.2: Reel-rate input for the parameter set in Fig. 3.1 showing that the trigger switches between reel-out rate at 1 m/s to the reel-in rate at -1 m/s

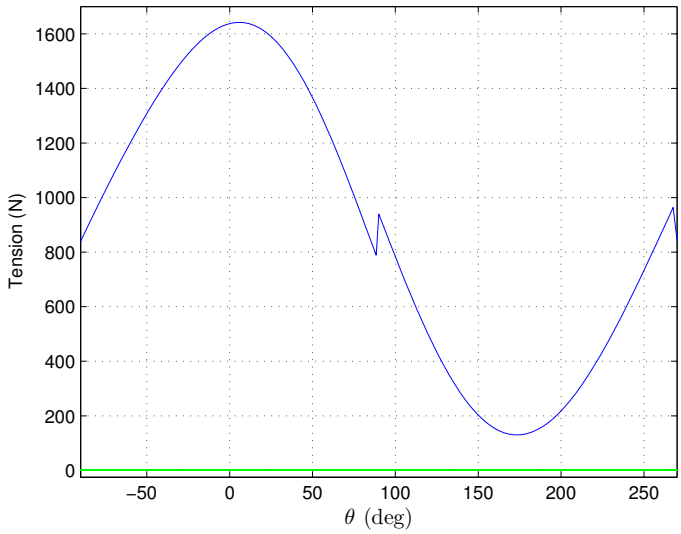


Fig. 3.3: Tether tension curve for the parameter set in Fig. 3.1 with tension switching after reel-rate change

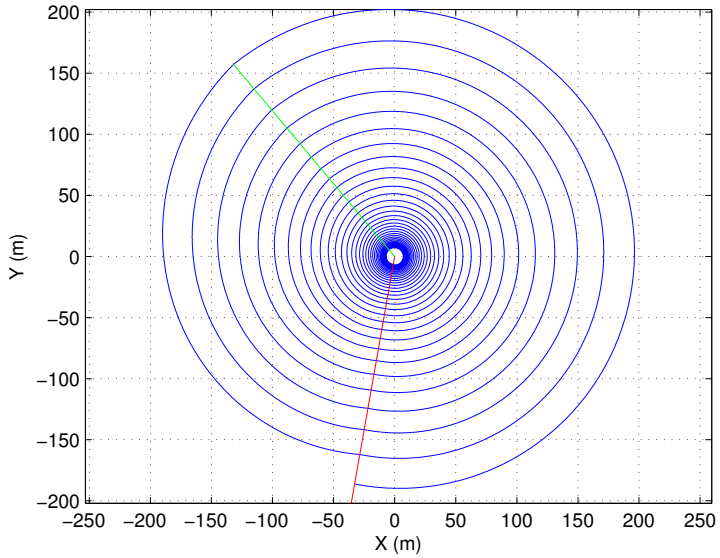
with the longest length occurring at $\theta = 90^o$ and the shortest length occurring at $\theta = 270^o$. The system will slowly increase or decrease the tether length during each cycle until the change in location is small enough to trigger convergence, as shown in Fig. 3.2. The system begins with a reel-out rate of 1 m/s and ends with a reel-in rate of -1 m/s. The trigger happens almost directly in the center of the graph showing the times in each phase are almost equal for these parameters. However, this is based on a time scale and the trigger locations are better seen with reference to angular position (θ).

The flight path for this set of parameters is a nearly symmetrical, oblong circle since predefined angles for initializing the reel-in or reel-out phases are 180^o out of phase. The cycle time for this case is $\sim$1.1 secs, shown in Fig. 3.2, and the Δl is also small at 0.15m, shown in Fig. 3.1. Both contribute to a low energy output and average cycle power. This system operates at steady state with a positive net average cycle power of 441.6W. This is mainly due to the low reel-out rate which is increased in later solutions.

Tension stays positive, shown in Fig. 3.3, and depicts the discontinuity in tension due to the step change in tether length rate as specified by the assumed tether length controller at 90^o. The maximum tension is 1.6 kN, but the system keeps positive tension in the line throughout the cycle. Higher tether tension occurs during the reel-out phase while the lower tension occurs during reel-in.

3.1.2 Convergence Determination and Inviability

Inviability must be determined when choosing periodic motions capable of providing sustained flight. Two criterion are present to determine inviability. The first is the ability to converge based on the steady state error found in Eq. (2.16) which provides insight if the system is capable of sustaining periodic flight, and the second is to maintain positive tether tension.

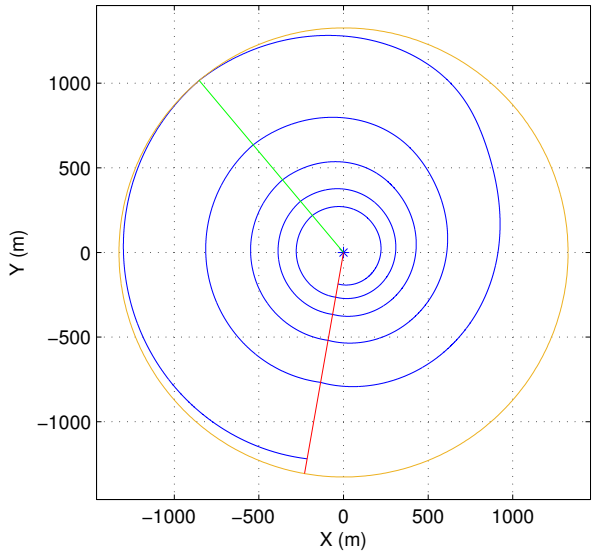


$\theta_{in} = 130^o$	$\dot{\theta} = \frac{30.93^o}{s}$	$\dot{l}_{in}$ = -9 m/s
$\theta_{out} = 260^o$	l = 189.055m	$\dot{l}_{out}$ = 2.07 m/s

Fig. 3.4: Transient Behavior leading to non-convergence of the proposed kite system.

Figure 3.4 shows a solution that spiraled towards the base station and decreased the tether length during a set number of cycles (50 minimum cycles in this case). The position of the wing rotates about the base station and neither the tether length nor the angular velocity return to the previous value needed to stop the simulation. The system tether would rapidly converge to a zero length if the simulation were to continue. Figure 3.5 travels too far out and will also fail to converge at the set parameters. After 5 cycles, the wing was near 1500 m long and would continue to diverge. Neither inviable case warrant a power calculation since the parameters do not satisfy convergence.

The flightpath has two main contributing parameters controlling convergence. Trigger locations must provide a proper combination of timing in the cycle during the phases and reel rates need to work in conjunction with trigger locations to provide logical parameters for convergence. For example, a reel-in phase region much less



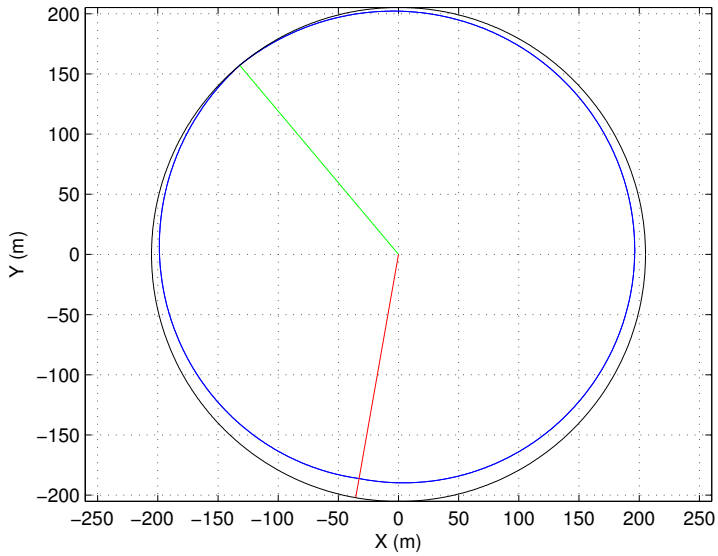
$\theta_{in} = 130^o$	$\dot{\theta} = \frac{30.93^o}{s}$	$\dot{l}_{in}$ = -3.75 m/s
$\theta_{out} = 260^o$	$l = 189.055$m	$\dot{l}_{out} = 9$ m/s

Fig. 3.5: Transient Behavior leading to non-convergence of the proposed kite system.

than the reel-out phase region cannot have a reel-in rate much less than the reel-out rate because the system needs time to reel-in the system to the original length.

Also, upwind of the anchor point, the kite will travel slower than downwind because the kite is experiencing a smaller apparent wind velocity vector from $\theta = 90^o$ to 180^o with an incoming wind along the $\hat{i}$-direction. The kite velocity is decreased since the vector is maintained by the tether length multiplied by tether angular velocity or by the reel-in rate. In order to make the system travel faster during the reel-in phase, the tether must remain constant or increase to satisfy the first term (which will reduce the apparent wind velocity vector and cause the kite to lose lift) or retract the tether faster. However, this will cause a higher consumption of power and will reduce the overall net average cycle power.

Another important behavior will occur when the reel-out angle is small. In the downwind region, air flows at an appropriate angle of attack for the design of the



$\theta_{in} = 130^o$	$\dot{\theta} = \frac{30.93^o}{s}$	$\dot{l}_{in}$ = -3.75 m/s
$\theta_{out} = 260^o$	l = 189.055m	$\dot{l}_{out}$ = 2.07 m/s

Fig. 3.6: A system which has negative tension during the periodic cycle

airfoil without the need to use the reel-out rate to adjust the apparent wind velocity vector (*i.e.* compared to the necessary larger reel-in rate to adjust the angle of attack in the reel-in phase). The ability to increase tether length causes the velocity vector to increase, accelerating the kite forward and greatly increasing angular velocity about the anchor point. If the reel-out phase region were decreased, the reel-out rate would need to be increased to balance convergence with a reel-in rate comparable to those found for solutions in this section while the upper bound for reel-out rate is the speed of the wind for the symmetrical airfoil, NACA0015, and with a tether to kite chord line angle (β) of 90 ^{o}.

The second criteria declares that tension must remain positive to keep the tether in tension. If the solution provides tension below zero, at any point, the tether is essentially in compression (would lose tautness) and, for a physical system, would result in a slack tether line. The solution in Fig. 3.6 is inviable because it violates the

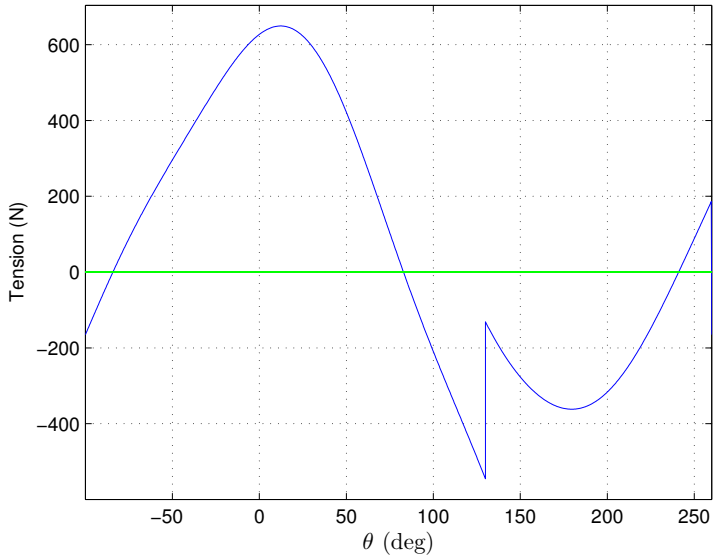
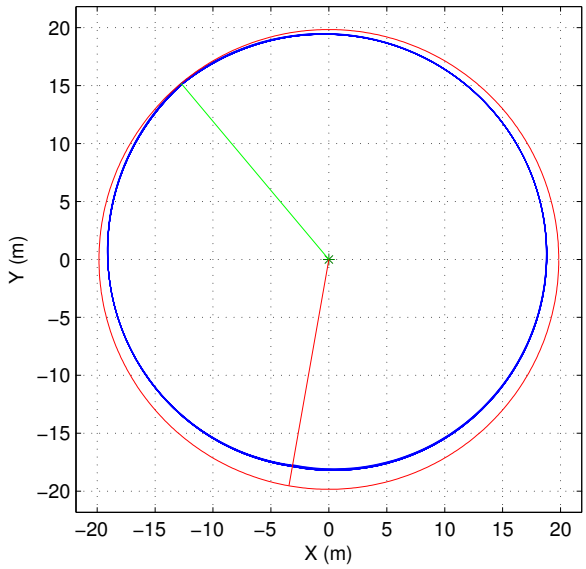


Fig. 3.7: Negative tether tension behavior for parameter set in Fig. 3.6

tension criteria but otherwise is indistinguishable from a viable solution. The tension plot for this parameter set is found in Fig. 3.7.

The minimum tension is most likely to occur upwind since the wind will push the wing towards the anchor point. Therefore, reel-in rate is the driving force to control negative tension. In the case that negative tension occurs upwind, the reel-in rate would need to increase in order to pull the wing faster than the wind would like to push it and change the apparent wind velocity vector towards a proper airflow over the wing.

Another possible scenario is reeling-out the kite too fast for the wind to establish flow and assist pushing the system downwind and away from the anchor point. This scenario is not as likely but will occur as the wing lift force is perpendicular into the wind or approaching perpendicularity to the oncoming wind (around -90^o or 90^o). This scenario is also the reason why the wind speed is the maximum reel-out rate for this airfoil design and β angle. The wind will not establish flow over the kite, and the tether, in this simulation, will push the kite outward instead of the wind.



$\theta_{in} = 130^o$	$\dot{\theta} = \frac{286.5^o}{s}$	$\dot{l}_{in}$ = -3.75 m/s
$\theta_{out} = 260^o$	l = 18m	$\dot{l}_{out}$ = 2.114 m/s

Fig. 3.8: Transient Behavior leading to Steady State Operation of the Proposed Kite System

3.1.3 Viable Solution

A parameter set satisfying all conditions is found in Fig. 3.8. The trigger locations are the same for the non-converging tether length and negative tension parameters. The transient phase is very small and convergences within a few cycles because a solution was found through the multidimensional root find and initial conditions set close to the solution (calculations for determining solution locations is discussed in Section 3.3). If initial conditions started further away, the solution would slowly reach convergence and have a larger transient response such as Fig. 3.1.

The net average cycle power for this parameter set is 559.6W. The periodic wave is also very fast at 1.2 sec as shown in Fig. 3.9. The tension remains positive throughout the simulation as is shown in Fig. 3.10 and there is a clear discontinuity at 130^o and 260^o to show distinct curves for reel-out versus reel-in.

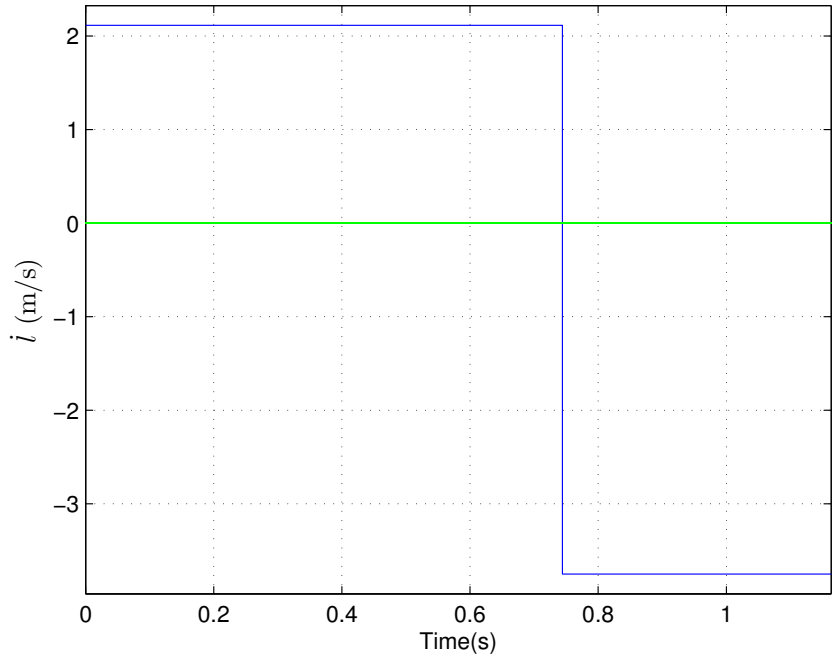


Fig. 3.9: Reel-rate for the viable parameter set in Fig. 3.8

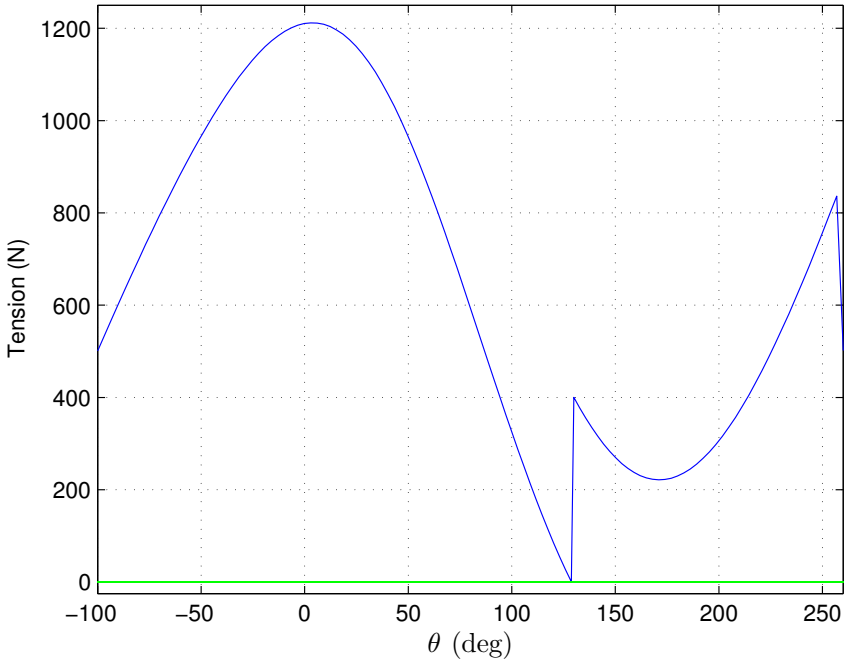


Fig. 3.10: Tether tension plot for the viable parameter set in Fig. 3.8 showing that the minimum tension is > 0 N during the cycle

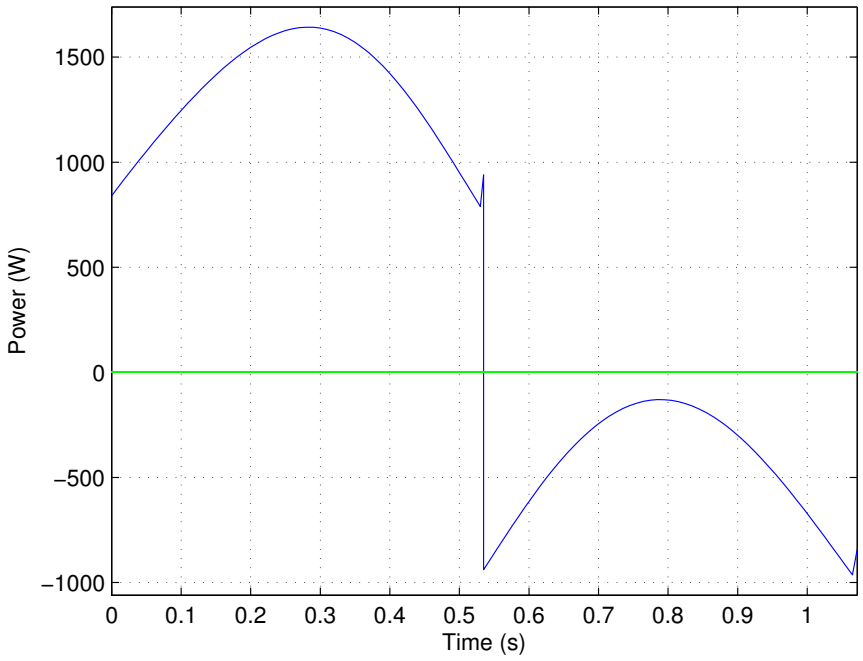


Fig. 3.11: Instantaneous power for a steady state, periodic cycle from the parameter set in Fig. 3.1

3.2 Positive Net Power

An instantaneous power curve is generated for each solution that fulfills the criteria of convergence and tension for a viable solution. As stated in Eq. (2.17), the power curve is equal to the reel-rate multiplied by the tension in the tether. Figure 3.11 is the power curve for the symmetric solution found in Fig. 3.1. The phase triggers for reel-in and reel-out are 90^o + n·360^o and 270^o + n·360^o, respectively (with n as the cycle number). The curve depicts the instantaneous power for the particular parameter set after one cycle about the origin for the periodic motion. The power curve defines the initial time for the cycle (t_0) at the reel-out trigger (-90^o) to begin producing positive energy and proceeds until the reel-in trigger (270^o) where energy consumption begins and the tether is retracted.

The switch between the two phases is shown at the 0.53 sec marker as a discontinuity and again at the end of the cycle with final time (t_f). The curve flips from a

positive area above the zero boundary and is a negative area below. The switch at the beginning/end of the wave is shown with the small increase at the end of curve. The parameter set for the symmetric, viable solution produces 447.6W of mechanical power.

A baseline is set at zero joules to contrast the difference between producing and consuming energy. The positive energy generation for each cycle is apparent from the size of area underneath each portion of the curve. A larger positive area than negative will lead to a positive net average cycle power, and vice versa.

Power curves are not generated for solutions that do not meet convergence criteria because they are inconsistent and, therefore, not accurate to produce or quantify a steady amount of energy. Parameter sets that lead to negative tensions are inviable so average cycle powers for these motions are not calculated.

The power output for the asymmetric trigger location, viable solution from Fig. 3.8 is found in Fig. 3.12. It is similar to the power curve for the symmetric trigger location solution in Fig. 3.11. During the reel-out phase, energy production is highest when the tether tension is largest (where the kite chord line is perpendicular to the air stream at $\theta = 0^o$) since the reel-out rate is always a constant. Both negative and positive power have parabolic relationships in their respective phase regions with the wind consistently traveling in the same direction. This relationship is obvious because of the approximate location of the relative minimum and maximum of tether tension occurring while the wing is within close proximity to a perpendicular geometry with the wind, and increases or decreases as the wing is pushed away or pulled towards the anchor point.

The net average cycle power of the asymmetric, viable solution with this particular set of parameters is 559.6 W. A positive net average cycle power shows that the system is producing more power during reel-out than consuming during reel-in.

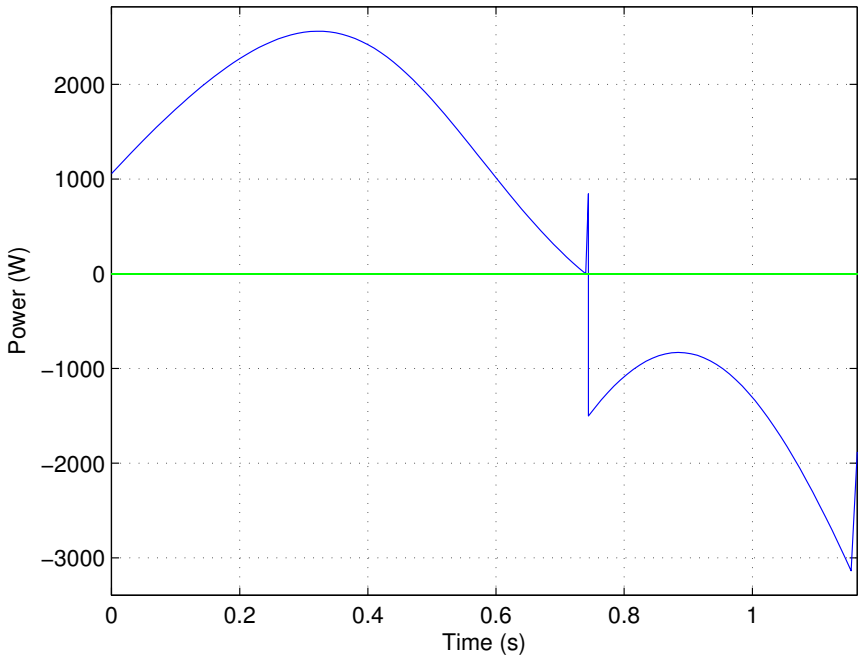


Fig. 3.12: Instantaneous Power for a Steady State Cycle for the asymmetric, viable solution found in Fig. 3.8

3.3 Periodic Convergence Patterns

The convergence criteria described in Section 2.3 states that the difference between the beginning and end wave angular velocity ($\dot{\theta}$) and tether lengths (l) must fall below a convergence criterion (ϵ). Therefore, there must be solution along a certain range of $\dot{\theta}$ and l initial conditions that satisfy when either condition is below the ϵ. For convergence calculations, the $\dot{\theta}$ and l are saved at the end of each cycle, considered the reel-out trigger since the initial conditions begin each simulation at the same trigger.

A single variable root find, Newton's Method, determines where the difference between beginning and end wave $\dot{\theta}$ and the difference between beginning and end wave l were equal to 0 rad/s or 0 m, discussed in Section 2.5 and found in Eq. (2.20). The results for each difference were located and used to create error curves depicting where both differences of $\dot{\theta}$ and l were close to 0 rad/s or 0 m.

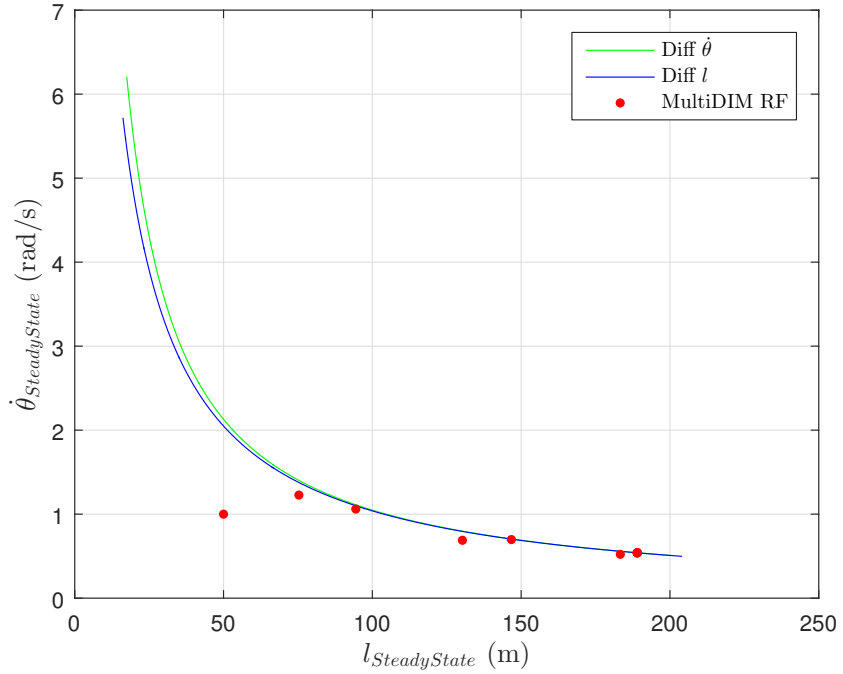


Fig. 3.13: Solution curves for Eq. (2.20) from single variable root finding routines and the multidimensional root finding routine path towards the zero error crossing of both variables

The two curves shown in Fig. 3.13 which are the solution curves as $\dot{\theta}$ and l fall below ϵ are from the inviable, asymmetric, negative tension parameter set used in Fig. 3.4.

A periodic solution satisfying Eq. (2.16) is located where the two curves cross. A 2D root find, using a Multidimensional Newton-Raphson Routine, is used to find convergence solutions with varying initial conditions for $\dot{\theta}$ and l. Figure 3.13 contains the results from a multidimensional root find to determine where the solution for the particular parameter set occurs to satisfy both difference equations used in the single variable root finds. The red bullets denote each step the root find moves to reach a tolerance of (10^{-7}) difference with the convergence factor in Eq. (2.16). From the initial conditions, the root find calculates the steady-state error and adjusts the initial conditions for the next calculation which would be closer to a convergence solution. The solution found in the parameter set for Fig. 3.13 are $\dot{\theta} = 0.5399$ rad/s and $l =$

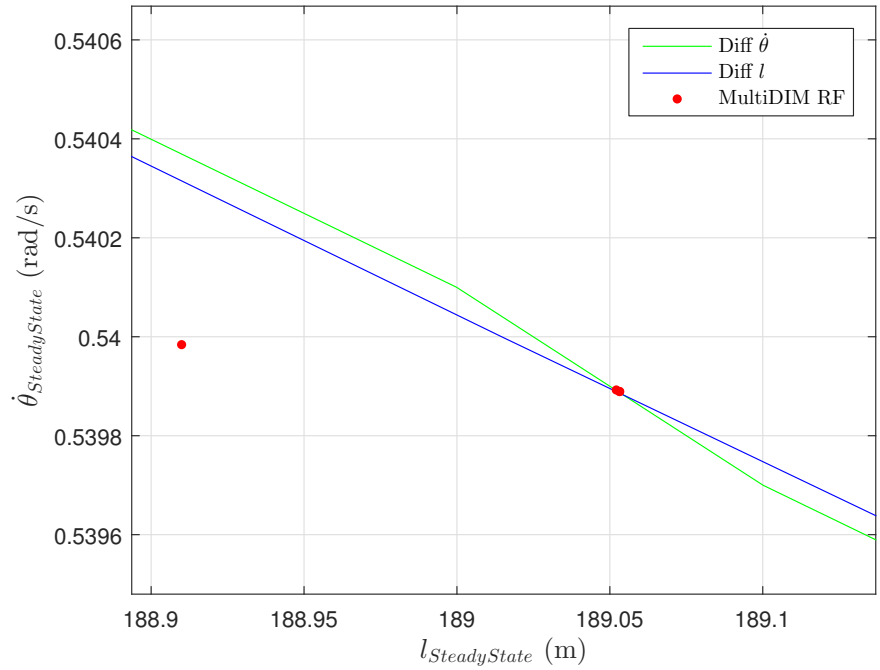


Fig. 3.14: A close-up of Fig. 3.13 showing the solution to the difference curves crosing and the multidimensional root finding routine leading towards a zero error crossing are consistent from both methods

189.055 m and Fig. 3.14 shows a close up of Fig. 3.13. Both methods converge at the same location.

As periodic motions are found with respect to $\dot{\theta}$ and l using the multidimensional routine, viable solutions are found by introducing variations to a third variable that affects minimum tension. This is discussed further in Section 3.4.

3.4 Parameter Variation

Periodic, positive tension solutions for the 2D system varies according to several key factors, such as, wing characteristics, wing orientation, reel-rates, wind speed, wind direction, and phase trigger locations. Wing design and the wind vector tend to hold stronger effects on minimum tension since they have the largest effects on tension. Reel-rates and trigger locations will affect convergence and at which angle

Table 3.2: Parameters set at constant values for the parameter sensitivity study unless stated otherwise

V_∞ = 10 m/s	c = 0.2 m	$\theta_{in} = 130^o$	$\dot{\theta} = \frac{286.5^o}{s}$	$\dot{l}_{in}$ = -3.75 m/s
m = 1.2 kg	b = 1.2 m	$\theta_{out} = 260^o$	l = 19 m	$\dot{l}_{out}$ = 2.07 m/s

the minimum tension occurs. Reel-rates can influence the minimum tension greatly, but, as previously discussed, they have a limited range according to wind speed during reel-out and reducing energy consumption during reel-in. A select amount of variables were chosen and varied at certain intervals to determine their effects on the system. The parameters in Table 3.2 are held constant unless varied at the ranges found in this section. Note, the variables are set at the same initial condition values as the viable solution from Fig. 3.8.

Three kite wing characteristics were analyzed to determine the relationship with minimum tension: mass, chord length, and wingspan. The three kite wing characteristic parameter variations were performed using the standard simulation. Minimum tension was calculated for periodic motions (determined by the convergence criteria) by varying the kite characteristics. All three characteristics have linear relationships with minimum tether tension, however, minimum tether tension will increase by increasing mass, and increasing chord length or wingspan causes the minimum tension to decrease. The viable, asymmetric system found in Fig. 3.8 requires a mass of 1.2 kg to keep a minimum tension at or above 0N, as shown in Fig. 3.15.

The same parameters were used to determine chord length and wingspan. The chord length of the kite in Fig. 3.16 and wingspan in Fig. 3.17 show there is a balance between mass and these characteristics to maintain a minimum tension $>$ 0 N. Defining the mass lower will cause the necessary chord length or wingspan to decrease compared to having a higher mass. In Fig. 3.16, the necessary chord length with a mass of 1.2 kg is 0.2 m. As the mass is increased to 3 kg, the minimum

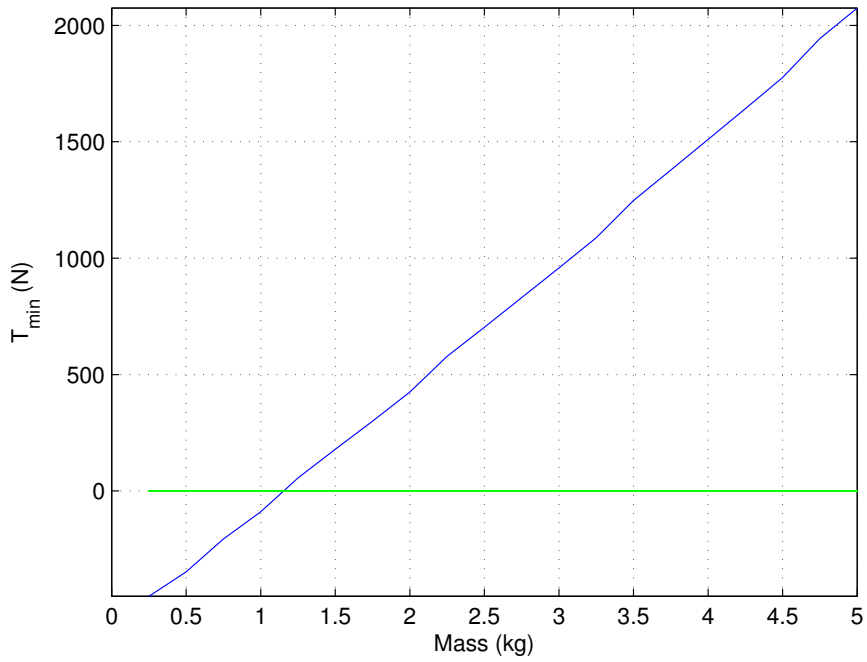


Fig. 3.15: Minimum Tension varying mass of the kite

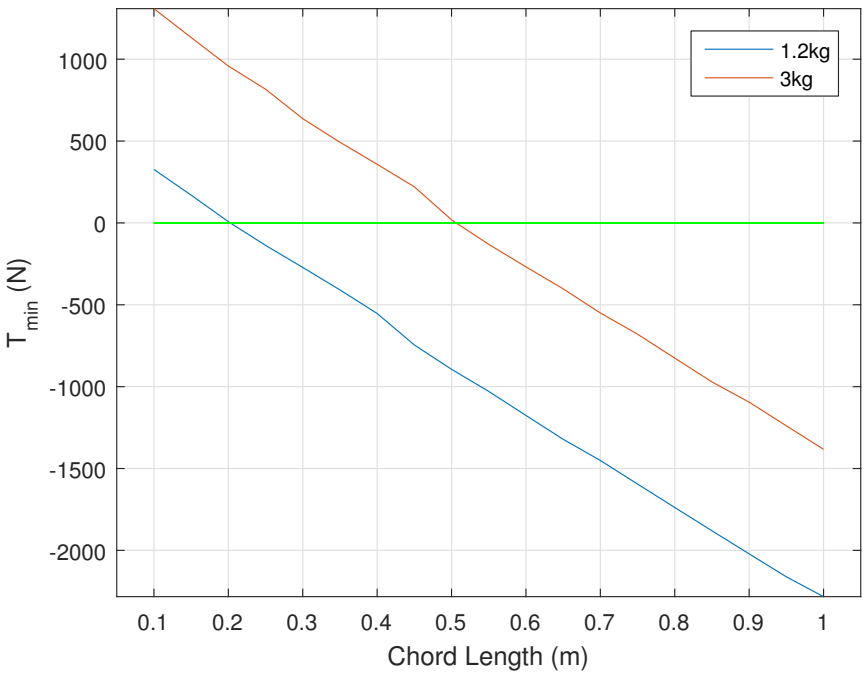


Fig. 3.16: Minimum Tension varying chord length of the kite

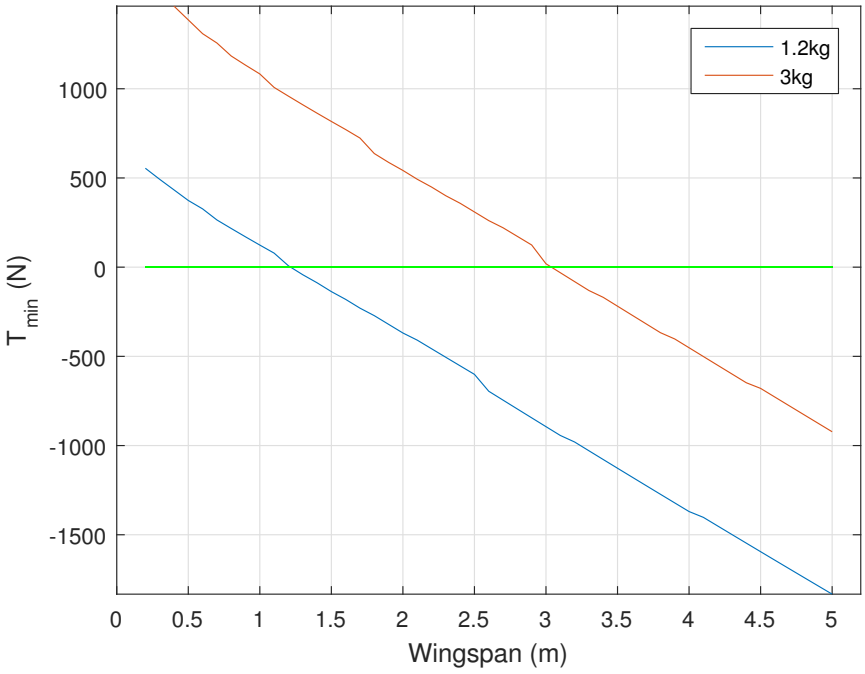


Fig. 3.17: Minimum Tension varying wingspan of the kite

chord length necessary to achieve a positive minimum tension is increased to 0.5 m. Respectively, the wingspan in Fig. 3.17 increased from 1.3 m to 3.1 m with the same change in mass. Therefore, an increase in the size of the wing will create a necessary mass increase to achieve a positive tension without modifying the other variables parameters affecting minimum tension. Essentially, any increase in the dimensions of the wing will necessitate an increase in mass to offset the change and place minimum tension in the positive region. To standardized the project analysis, mass is set at 1.2 kg, chord length is 0.2 m, and wingspan is 1.2 m to mimic the design of an R/C plane.

Previously discussed in Section 3.3, a third variable is varied with the multidimensional root find to determine effects on minimum tension. In Fig. 3.18, the reel-out rate is varied for a specific range and the multidimensional root find is performed to find the solutions for convergence. Solutions have distinct areas where positive

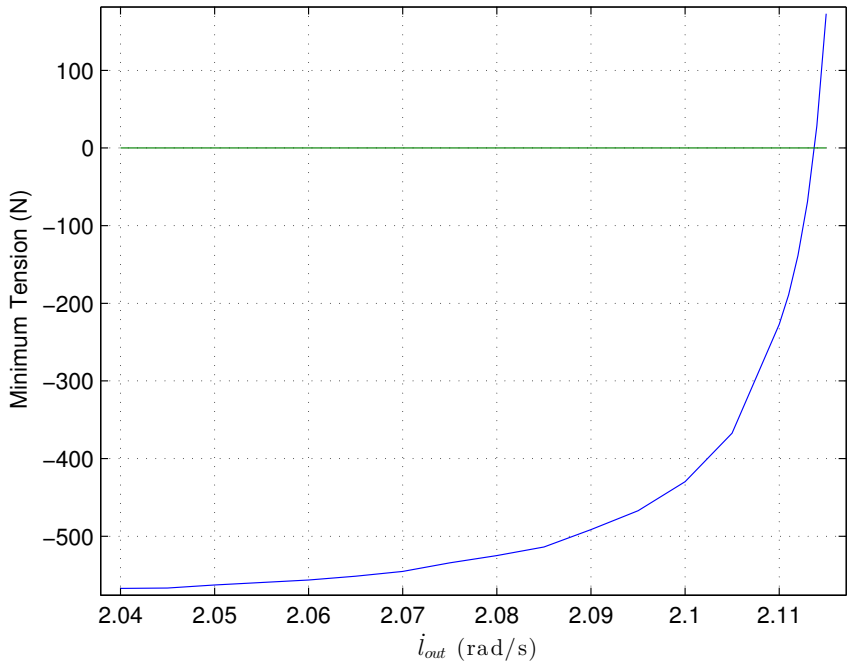


Fig. 3.18: Minimum Tension varying reel-out rate of the parameter set

tension was found and provide information where false solutions are located for select parameter sets. This method provides a direct method to quickly find viable solutions.

The reel-out rate ($\dot{l}_{out}$) shown in Fig. 3.18 is negative in the majority of converging parameter sets found with the multi-dimensional root find. This does not state that most converging or periodic solutions will have negative tension, but that solutions found with the root find are more likely to have a short range for positive tension. Figure 3.18 shows that the $\dot{l}_{out}$ must be $\geq$2.114 m/s to achieve a positive minimum tension for this particular parameter set. The figure also shows that the minimum tension will quickly increase the farther away $\dot{l}_{out}$ is set from the minimum value. As minimum tension exponentially grows, a physical limit in the tether is set and an increase in $\dot{l}_{out}$ will not be possible.

Similar curves are produced for reel-in rate ($\dot{l}_{in}$), tether to kite chord line angle

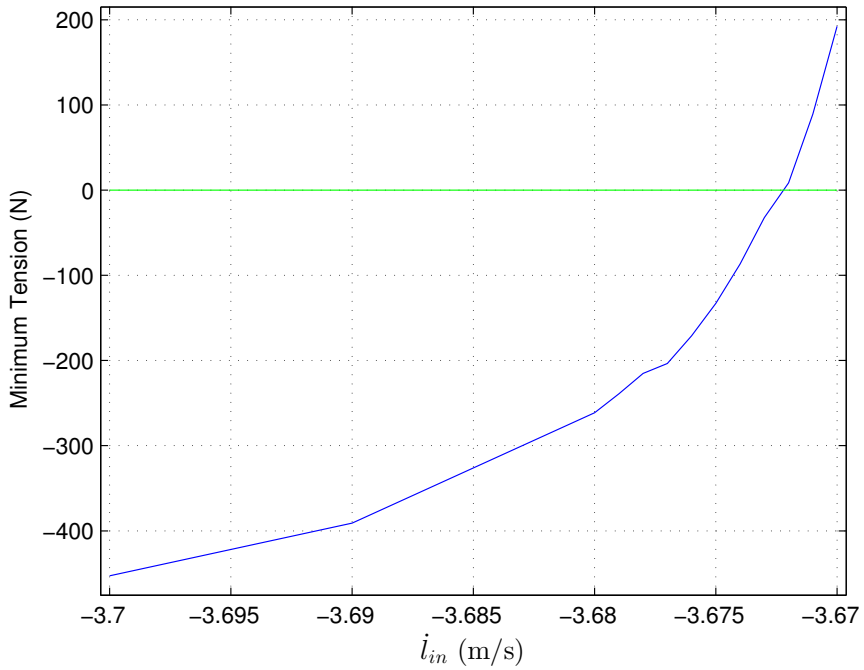


Fig. 3.19: Minimum Tension varying reel-in rate of the parameter set

(β), the reel-out trigger (θ_{out}), and the reel-in trigger (θ_{in}) in Figs. 3.19, 3.20, 3.21, and 3.22 respectively. In Figs. 3.18, 3.19, and 3.21, the ranges of $\dot{l}_{out}$, $\dot{l}_{in}$, and θ_{out} are very small with variations of the variables on the order of 10^2 which cause large changes in minimum tension. Yet, in Fig. 3.20, the changes are so small that it will not help drive the system into positive tension by changing this variable alone. If the systems expands beyond the region shown, convergence is not met.

The most interesting variable is θ_{in}. Figure 3.22 shows θ_{in} has a moderate affect on minimum tension. The order of magnitude changes to 10^{-1} and there is a larger range of values for power calculations. Figure 3.23 contains the average cycle powers for the positive tension solutions from Fig. 3.22. They are compared to the steady state lengths of the system to show that comparable power outputs are possible from multiple tether lengths. Steady state tether lengths ranging from 8.5-18m tend to produce average cycle powers from 500-600W. As the tether length becomes shorter

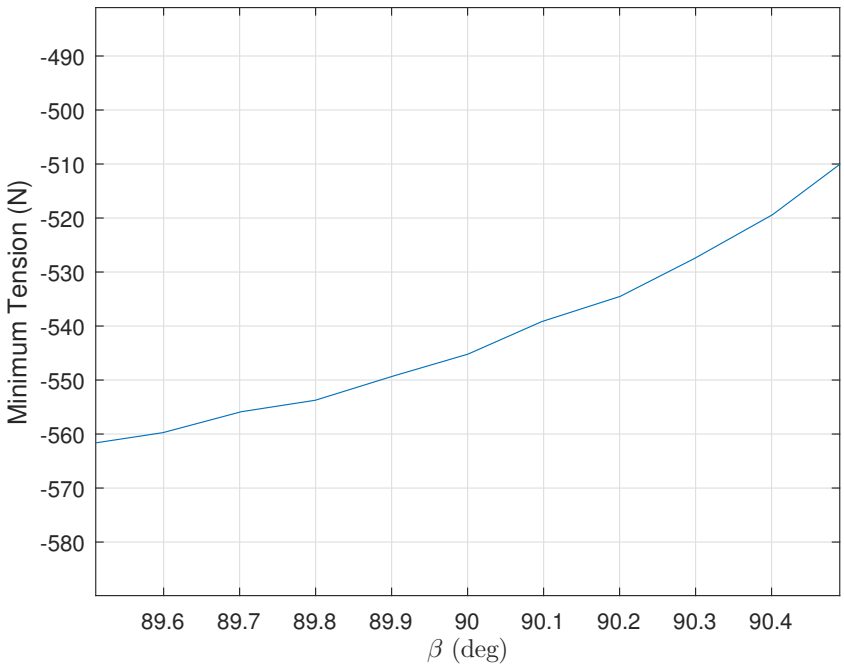


Fig. 3.20: Minimum Tension varying the angle between the tether and the chord line of the kite

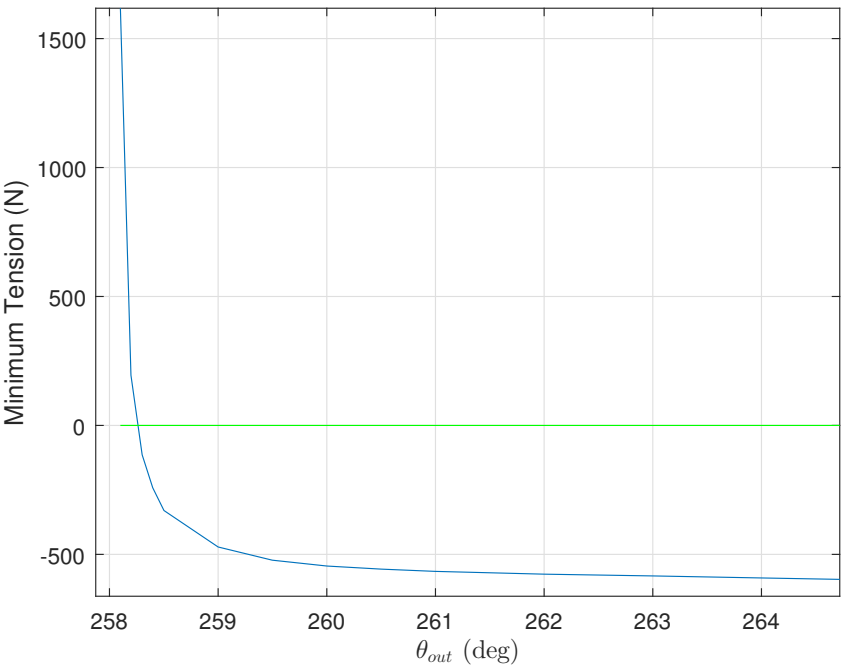


Fig. 3.21: Minimum Tension varying reel-out trigger of the parameter set

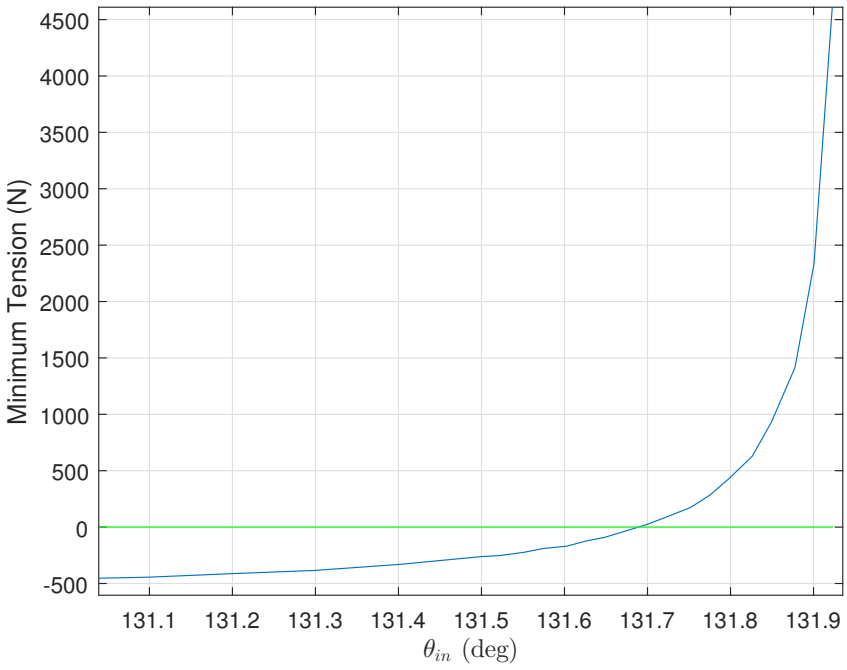


Fig. 3.22: Minimum Tension varying reel-in trigger of the parameter set

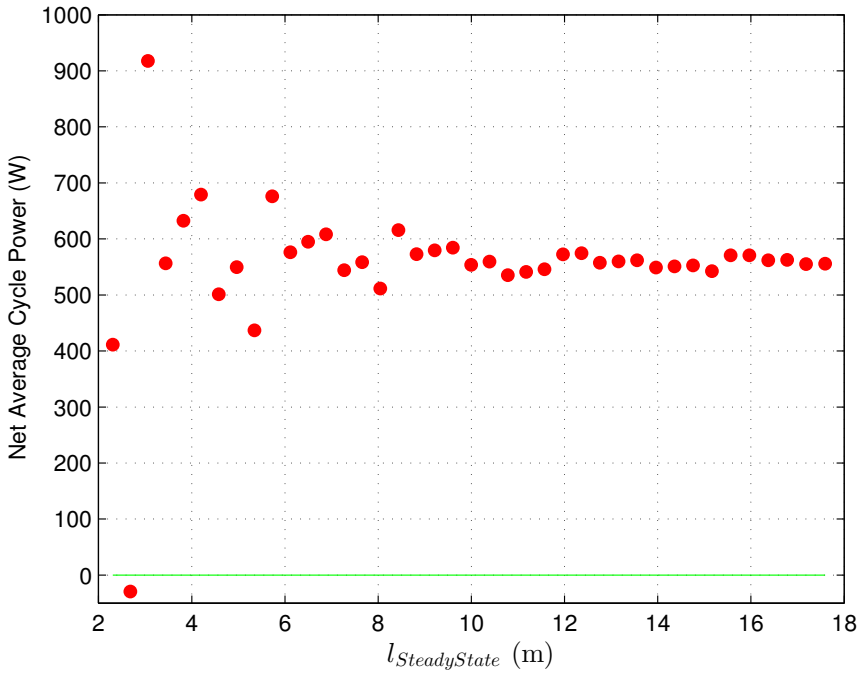


Fig. 3.23: Average Cycle Power for the multidimensional routine zero crossing solutions from varying the reel-in trigger location

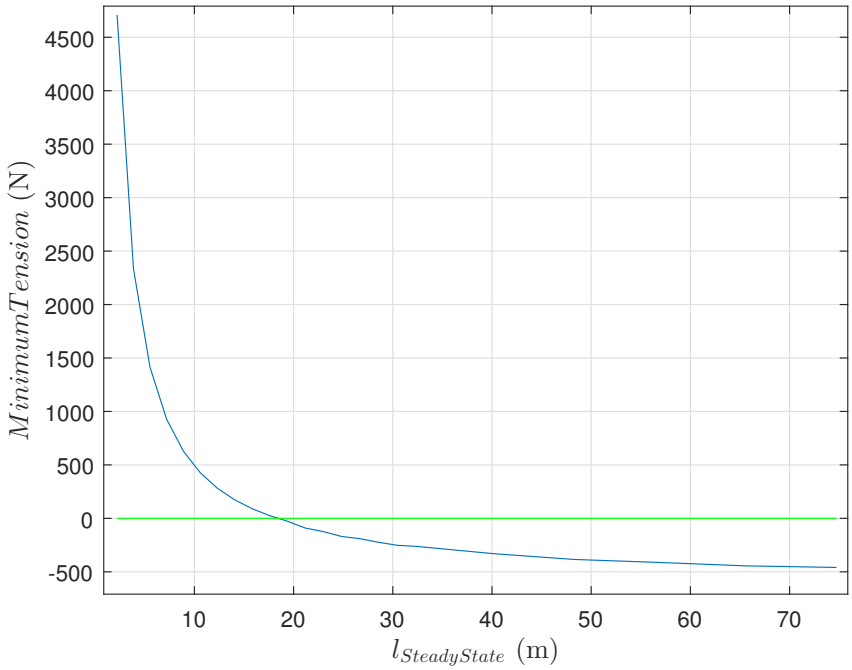


Fig. 3.24: Steady state tether lengths for the multidimensional routine zero crossing solutions compared to minimum tether tension from varying the reel-in trigger location

than 8.5m, the average cycle powers become sporadic and convergence is harder to achieve. Figure 3.24 compares the minimum tether tension to the steady state tether length and states that they are inversely related. As the initial length is increased, the minimum tension decreases. The tether tension is much higher at lower tether lengths due to the relationship between steady state variables discussed in the next section.

3.5 Single Parameter Family

Periodic motions are not constrained to specific initial conditions and are not difficult to achieve. The initial condition difference curves found in Fig. 3.13 are close to one another for certain parameter sets, and when are sitting as close together as they are (to within the distance of triggering convergence based on a small ϵ) the curves can

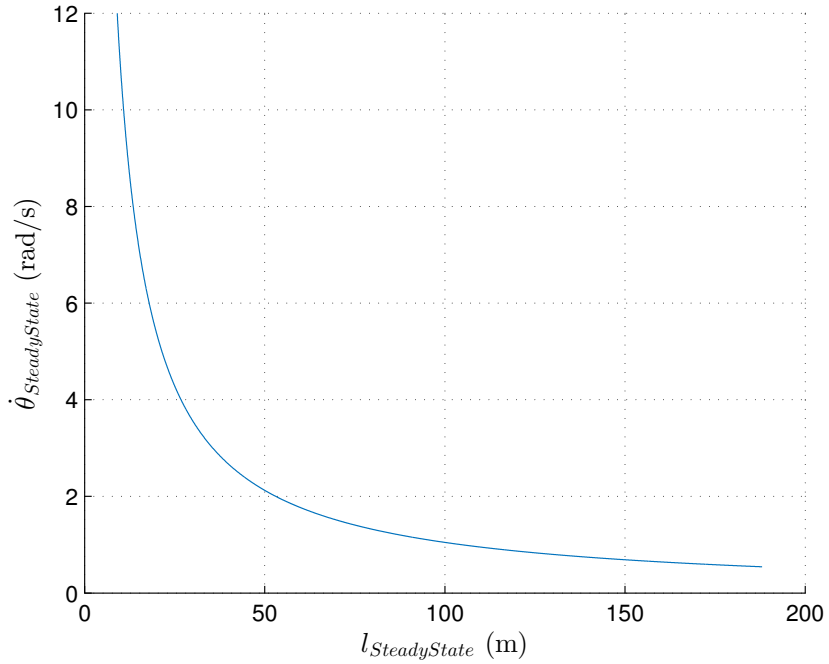


Fig. 3.25: Steady state conditions for $\dot{\theta}$ and l if the behavior for the two curves are considered close enough to provide a family of unstable solutions

be considered to lie on top of each other. Considering this approach, periodic motions are found for several cases when varying the initial conditions angular velocity ($\dot{\theta}$) and tether length (l), as shown in Fig. 3.25. The figure contains results from a standard simulation with 10,000 cycles as it began on a diverging path from the multidimensional zero crossing solution decreasing tether length. As the simulation progressed, the $\dot{\theta}$ and l of each cycle were recorded with the path following the exact curves found in 3.13 for the range of l shown.

As the initial l is increased or decreased, the transient phase, such as those found in Fig. 3.1, will attain periodic motion along the curve generally in less than 4 cycles. The particular initial initial l will adjust to a neighboring steady state $\dot{\theta}$ to produce a periodic motion with a specific steady state l near the initial l. In Fig. 3.25, each l contains a unique periodic motion with a unique $\dot{\theta}$. This behavior shows that the system is capable of attaining periodic motion dependent upon one initial condition

that close to the difference curves. Figure 3.25 also shows that l has an inverse relationship with $\dot{\theta}$ which as the tether angular velocity increases, the l decreases.

This relationship affects the minimum tension of the system. Figure 3.24 shows that the minimum tether tension remains negative for a larger portion of l's found in Fig. 3.25. The minimum tether tension becomes positive for a smaller range of l's found below 20 m. The minimum tension rapidly increases as l is reaching the lower bound of 0 m and will create a vertical asymptote at $l = 0$ m.

3.6 Unstable Periodic Solutions

The zero difference solutions found with the multidimensional root find are placed back into a standard simulation model as initial conditions to generate the figures to show transient phases. The standard simulation model is used to determine system behavior while only affected by flight dynamics and system inputs. The standard simulation stops only when the convergence criteria are met, the minimum number of cycles is reached without convergence, or the integration becomes exponential and a solution cannot be found.

Solutions come in different forms with this paper discussing stable and unstable solutions. Stable solutions have a convergence area surrounding their locations to drive the system towards the system point and stability. So if the initial conditions were to start within the convergence area, the system would work towards the stable solution point and begin to repeat the pattern (the same $\dot{\theta}$ and l each cycle) at that solution.

However, this system seems to correspond with solutions that are unstable meaning that there is not a convergence zone surrounding the solution point and the system tends to drive away from the solution while starting slow and slightly increasing the speed of divergence over time. Figure 3.26 shows the pattern of divergence for initial

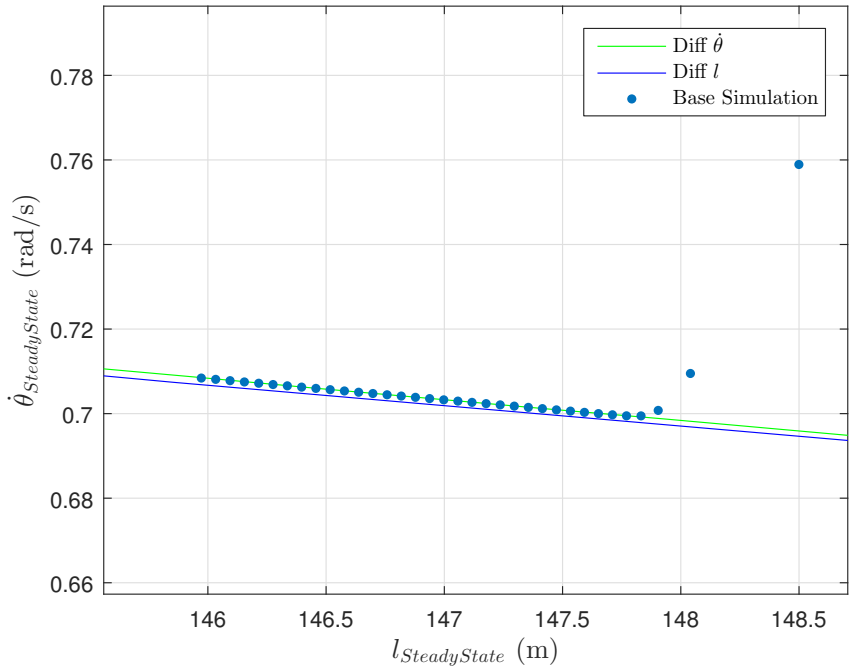


Fig. 3.26: Standard simulation end wave values for $\dot{\theta}$ and l measured at the reel-out trigger to show the path of transient along the single root find difference curves

conditions starting at a $\dot{\theta}$ of 57.3 rad/s (1 rad/s) and a l of 150m. The standard simulation shows the $\dot{\theta}$ and l at the end of each reel-out trigger (the location considered the beginning and end of a cycle) lead directly into the error curves found in Fig. 3.13 and slowly move along the error path with a steady and consistent change, as shown in Fig. 3.27, without finding a solution. Figure 3.27 is the difference between the $\dot{\theta}$ and l at the end of each cycle. The rates are almost constant with very small trends showing that $\dot{\theta}$ is slowly accelerating faster with each cycle and tether length is decreasing faster with each cycle.

Figures 3.28, 3.29, and 3.30 will explain this effect in more detail. The standard simulation began with initial conditions directly on the multidimensional root find solution for the error curves found in Fig. 3.13. The simulation was then set to run a minimum of 5000 cycles while sampling the $\dot{\theta}$ and l at the end of each cycle. Figure 3.28 shows that the system diverges away from the solution and works along the

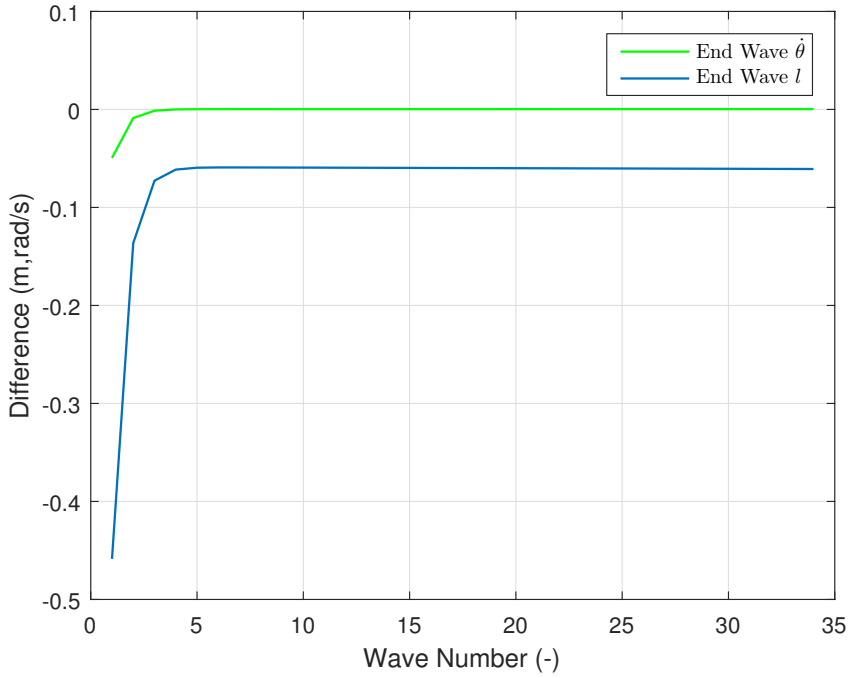


Fig. 3.27: The end wave value differences for $\dot{\theta}$ and l measured at the reel-out trigger between each successive cycle

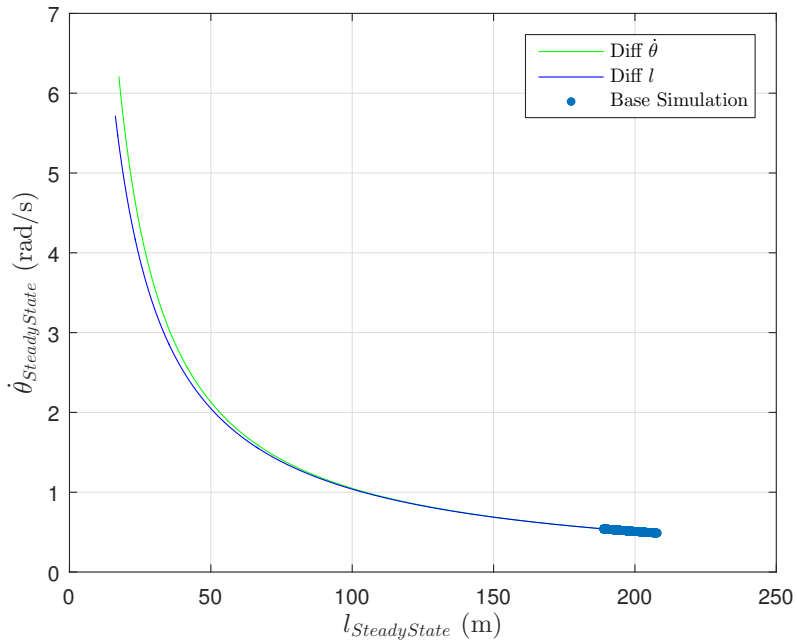


Fig. 3.28: The end wave values for $\dot{\theta}$ and l for 5000 cycles showing a divergence from the zero error crossing solution

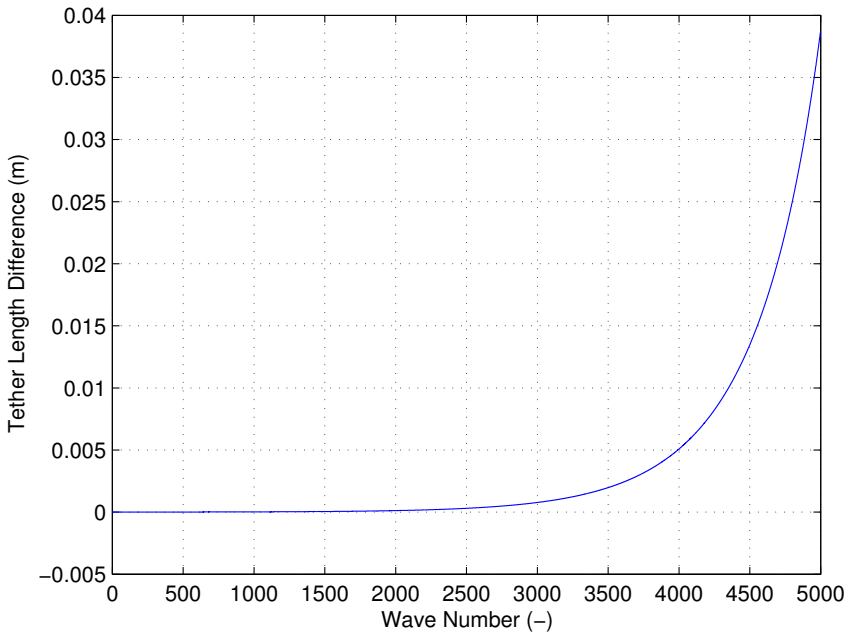


Fig. 3.29: The end wave values for the l difference during 5000 cycles showing an increase in divergence rate from the zero error crossing solution

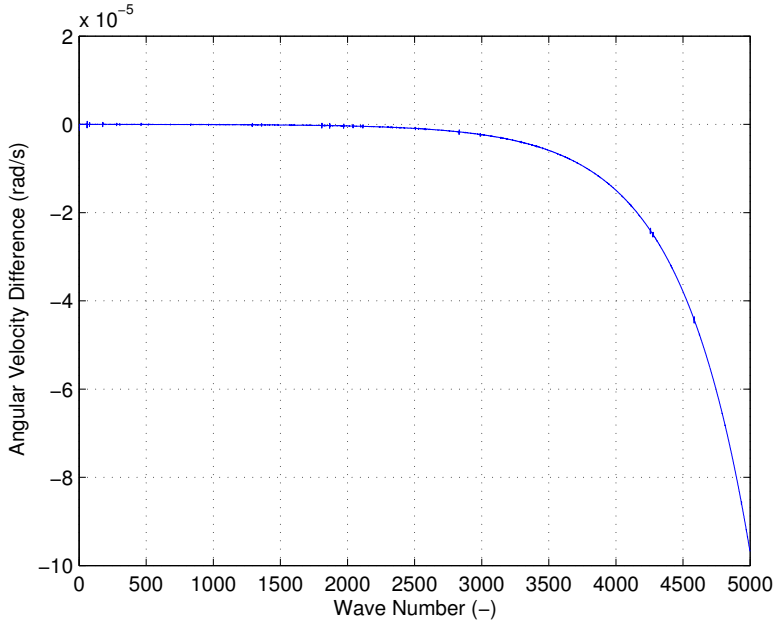


Fig. 3.30: The end wave values for the $\dot{\theta}$ difference during 5000 cycles showing an increase in divergence rate from the zero error crossing solution

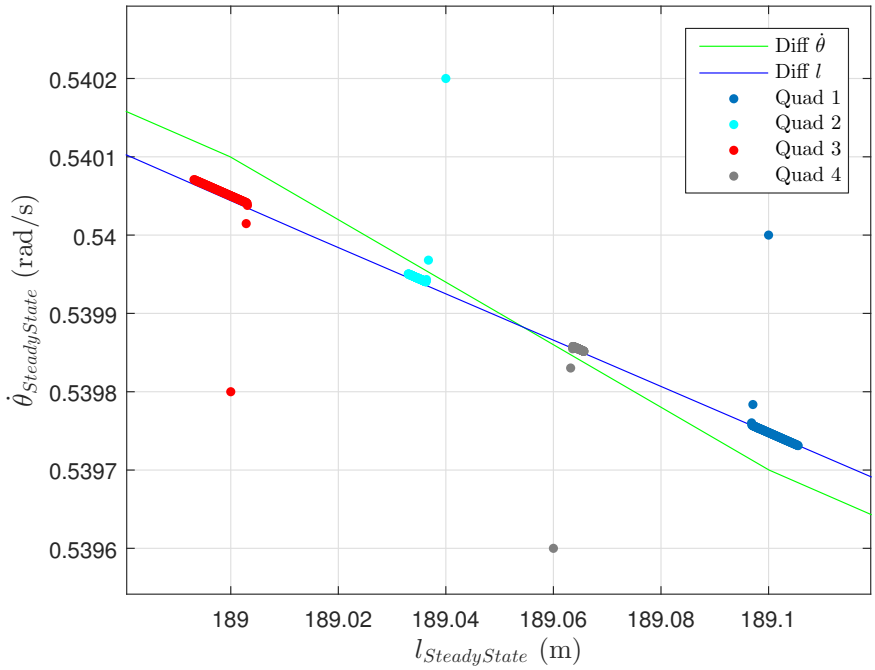


Fig. 3.31: Divergence plot showing the divergence changes per cycle after 35 cycles for initial conditions starting at varying values

error curves in a predictable pattern. Figures 3.29 and 3.30 show that the difference between each $\dot{\theta}$ and l difference at the ends of each cycle increases with each rotation. The $\dot{\theta}$ absolute difference is larger, but Fig. 3.30 shows that the $\dot{\theta}$ is slowing down faster with each cycle. This is due to where the initial conditions are located in relation to the zero difference solution provided by the multidimensional root find. The system overall shifted from a l = 189 m to a l = 207 m and slowed down from $\dot{\theta}$ = 0.54 rad/s to $\dot{\theta}$ = 0.48 rad/s after 5000 cycles.

The multidimensional root find provides an exact location where the system will have a zero difference between $\dot{\theta}$ and l at the end of each location. Figure 3.31 contains the solution with four standard simulation initial condition parameter sets starting in the four quadrants surrounding the solution. The four quadrants surrounding the solution are labeled the same as a Cartesian coordinate system and signify the patterns of: Quadrant 1) Higher $\dot{\theta}$ and longer l; Quadrant 2) Higher $\dot{\theta}$ and shorter l;

Quadrant 3) Lower $\dot{\theta}$ and shorter l; and Quadrant 4) Lower $\dot{\theta}$ and longer l.

Each parameter set begins at the furthest point from the error curves for that set. Each set quickly moves directly to the error curves and begin to diverge from the solution. Quadrants 1 and 4 moves away from the solution towards the right with a decreasing $\dot{\theta}$ and an increasing l. Quadrants 2 and 3 move in the opposite direction with an increasing $\dot{\theta}$ and a decreasing l.

The difference in $\dot{\theta}$ and l for each cycle increases the same as if the parameter set started on the solution itself but begins with a higher divergent rate depending on the distance from the multidimensional root find solution. This is apparent from the distance traveled by each quadrant parameter set since the minimum number of cycles was set at 35. Each parameter set was set to the same amount of cycles, however, quadrant 1 and 3 were diverging much faster away from the solution while quadrant 2 and 4 were much slower.

3.7 Phase Energy Ratio

A ratio of generated and consumed power will provide a basic comparison between the magnitudes of power during each reel-phase of periodic motion. A value of 1 will be a parameter set with equal generation to consumption, and a percentage >1 or <1 will be a parameter set with higher generation than consumption or vice versa, respectively. The power ratio is the absolute value of the quotient generated power by consumed power and is found in Eq. (3.1).

$$\eta_{power} = \left| \frac{W_{generation}}{W_{consumption}} \right| \tag{3.1}$$

The power ratio for the asymmetric solution found in Fig. 3.8 is 2.06 with an average cycle power = 559.6 W. This value shows there is twice as much generation than consumption. The power ratio equal to 2.06 and average cycle power of 559.6

W is calculated from a $W_{generation} = 1088.7$ W and $W_{consumption} = 529.1$ W.

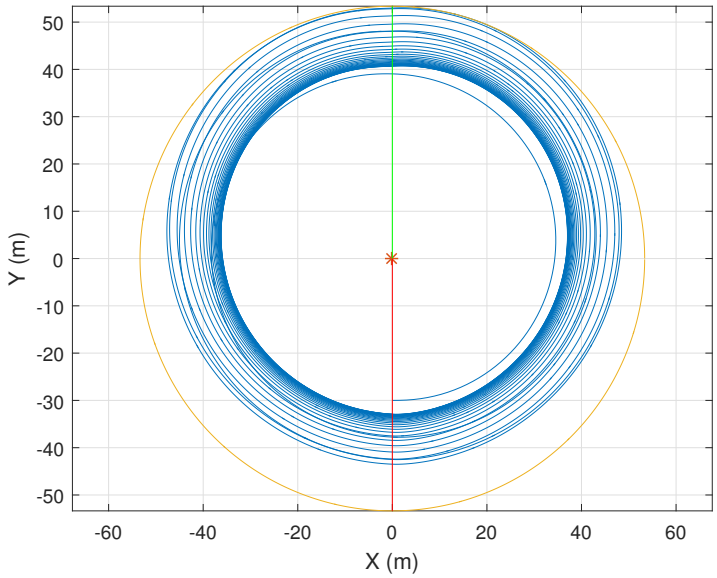
The power ratio and average cycle power pair provide a little more information when comparing magnitudes of power generation and consumption for pumping kite systems. A power ratio closer to 1 with a larger average cycle power shows that the magnitudes of generation or consumption are very large compared to the magnitude of the difference between generation and consumption. An example is a system generating a net of 550 W of power but needs to generate 1,000,500 W and consumes 1,000,000 W to produce the same difference. The ratio would be closer to 1 but with the same average cycle power. Therefore, a power ratio much greater than one and a larger average cycle power is desired to maintain a lower magnitude of power generation and consumption during each phase while generating a comparable power difference to those magnitudes.

3.8 Tether Tension Controller

3.8.1 System Behavior and Power Generation

In all the previous results shown, tether length rate was specified by the simple controller $\dot{l}_{in} = c_1$ and $\dot{l}_{out} = c_2$. In this section, we examine a closely related system where tether tension is specified instead of tether length rate. The controller for tether tension is specified at a constant value compared to the previous method of specifying the reel-rates. Therefore, any solution converging with the tension controller that has a set minimum tension >0 N is a viable solution for this particular altitude wind energy system.To achieve this type of tension control in a physical system, a closed-loop, step controller with tension feedback could regulate tether tension during the reel-in and reel-out regions.

The system in Fig. 3.32 begins with an initial length of 30 m, reel-out tension of 200 N, and reel-in tension of 1 N. The transient phase leads into convergence with



$\theta_{in} = 90^o$	$\dot{\theta} = \frac{286.5^o}{s}$	$T_{in} = 1$ N
$\theta_{out} = 270^o$	$l = 30$m	$T_{out} = 200$ N

Fig. 3.32: Transient of tether controller system leading into a steady, periodic motion around 30-32m with a power generation of 365.5W

a minimum length of 32 m and a maximum length of 41 m. Most tether tension convergence solutions, with a wind speed of 10 m/s, converged to a similar tether length as the solution shown in Fig. 3.32. This configuration converged within a range of reel-in tension from 0-50 N and reel-out tension from 400-1000 N.

From Fig. 3.32, the steady state convergence of this system is comparable to the solutions in previous sections. The large maximum and minimum tether distance is an oddity because the system has a low overall net power, discussed below. This particular system produces 365.5W, which is comparatively low to the second solution provided below and solutions in previous sections. A comparable power production to other solutions for the reel-rate controller is found below in Fig. 3.36.

Tether tension as a step response results in a square wave as the input, shown in

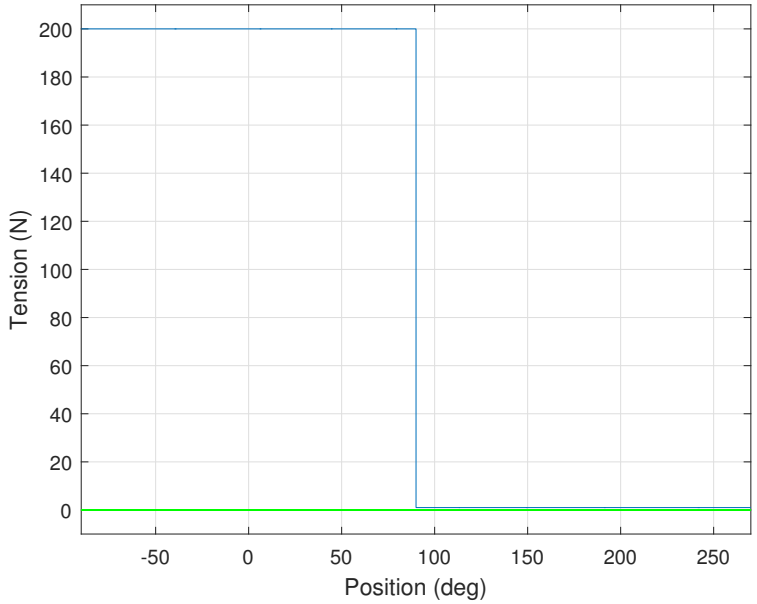


Fig. 3.33: Tether tension for the symmetric reel-phase, tether controller in Fig. 3.32

Fig. 3.33. The input for tension is low for the reel-in phase and high for the reel-out phase to minimize consumption and maximize production. The axis is shown in degrees to compare phase regions instead of timing, which corresponds with the symmetric solutions discussed previously in Section 3.1. The tension input is now based on the $\theta_{in} = 90^o$ and θ_{out} 270^o reel-in and reel-out triggers. The tension magnitude is also as discussed at 1 N for reel-in and 200 N for reel-out.

The curve for reel-rates is now similar to previous tension curves found in Section 3.1. The major parabola in Fig. 3.34 found in the first 2.2s of the curve is the reel-out region. It is associated with the major portion of generated power. As the wing rotates and is facing into the wind before the 90^o mark, the reel-out rate turns negative and begins to reel-in the system. The following reel-in has a positive reel-rate, and is reeling-out the system to generate a small portion of energy.

The same phenomena is found at the beginning of the next phase switch at the location of the two reel triggers. In all parameter combinations studied, the phenomenon were the same and had a greater affect in other tether tension controller

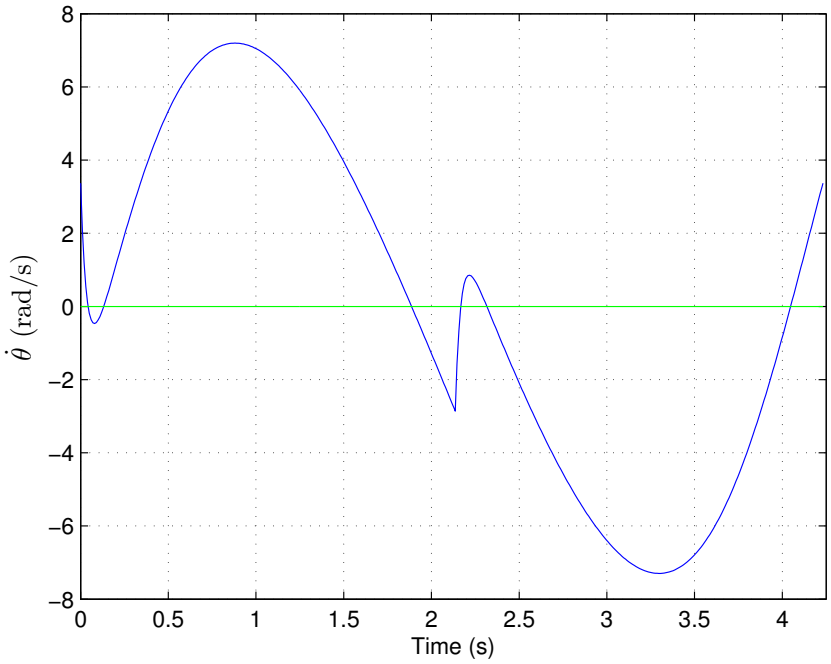


Fig. 3.34: Reel-in and reel-out rates for the symmetric reel-phase, tether controller in Fig. 3.32

tension curves.

Finishing off the last section of the reel-rate curve, the parabola in the graph shown during 2.3-4.3s is the reel-in rate to keep the tension at 1 N. A high reel-in rate with low tether tension is possible due to the decreased wind resistance upwind of the base station and provides evidence that the wind is assisting with decreasing power consumption during reel-in.

The power curve for the symmetric tension controller found in Fig. 3.35 is similar to the power curves found in Section 3.2. The power curve is the multiplication of the reel-rate and the tension force. The advantage of the tension controller is shown the massive reduction in power consumption during the reel-in phase. The power curve is minor due to the 1N force multiplied by the reel-in rate. The curve is close to 0 W because of the difference in scale compared to the power generation section of the curve. The power curve also has the phenomena found during phase change and is a

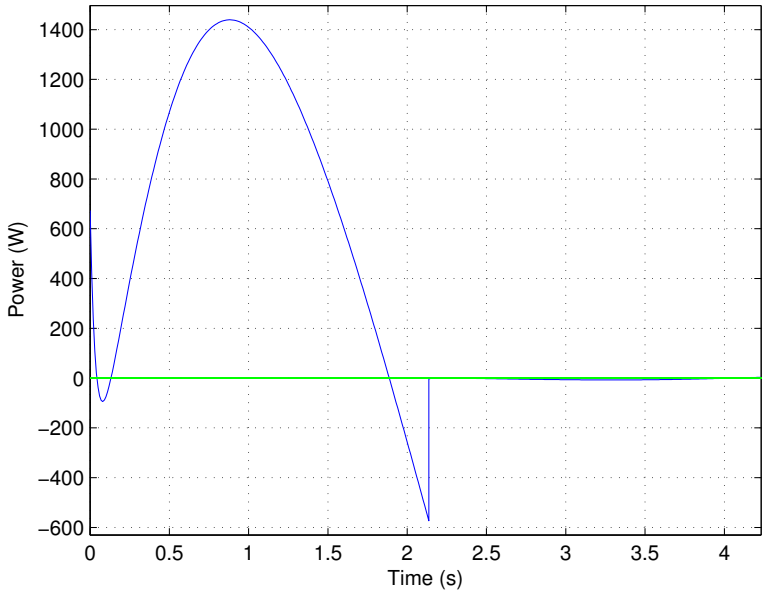
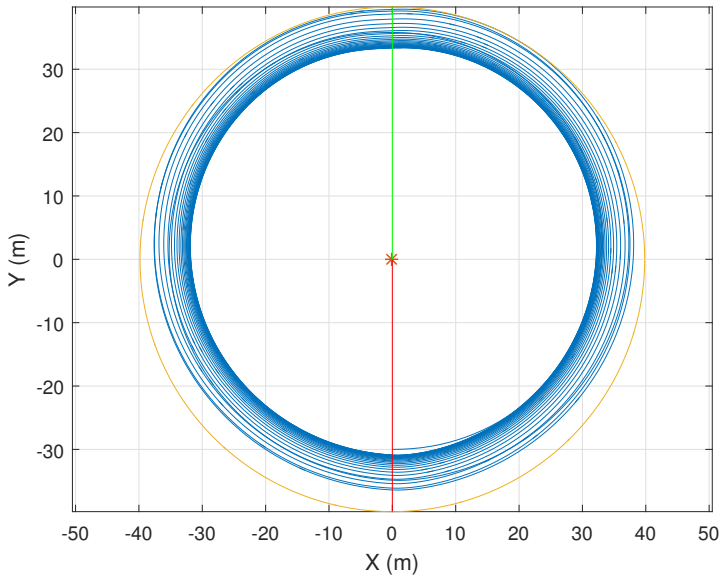


Fig. 3.35: Instantaneous power for the symmetric reel-phase, tether controller in Fig. 3.32

major consumption of energy just before the reel-in phase.

The drawback of the tension system is that much higher tensions are not achieved such as those in Section 3.1. This factor is important because it is a multiplier for the power curve and higher tensions in reel-in would increase the reel-in rate and, more importantly, decrease the time needed per wave. If the tensions were increased, they would begin to consume much larger amounts of energy during the phenomena, which lowers the overall net power generation. It is more beneficial to keep lower tensions during reel-out due to both reasons. However, a comparable system to those found with the reel-rate controller is found in Fig. 3.36.

The initial condition for angular velocity in this system causes a large transient phase, but the steady, periodic motion settles along the same 30-32m tether length range with a power output of 537.5W. As stated before, every parameter set that converges with tensions set higher than 0N is a viable solution for this type of system. Therefore, it is very simple to find solutions along the entire parameter spectrum.



$\theta_{in} = 90^o$	$\dot{\theta} = \frac{286.5^o}{s}$	$T_{in} = 35$ N
$\theta_{out} = 270^o$	$l = 30$m	$T_{out} = 540$ N

Fig. 3.36: Transient of tether controller system leading into a steady, periodic motion around 30-32m with a higher power generation of 537.5W

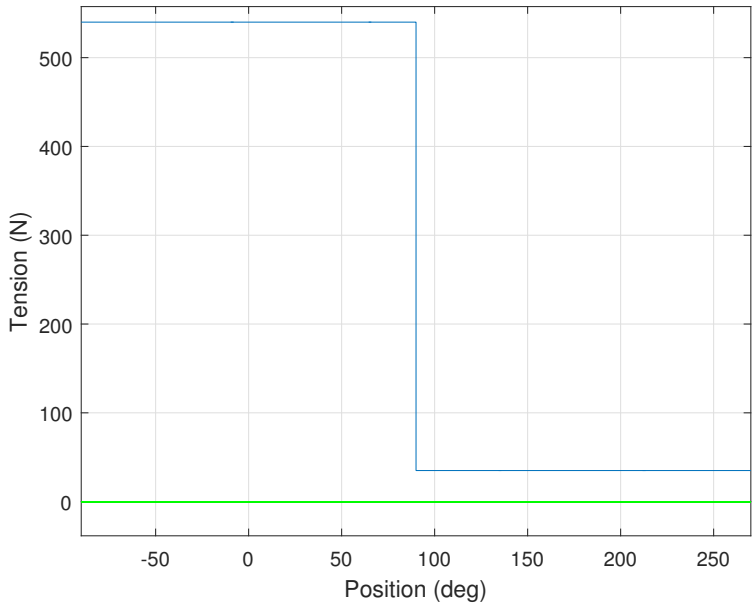


Fig. 3.37: Tether tension for the asymmetric, tether controller in Fig. 3.36

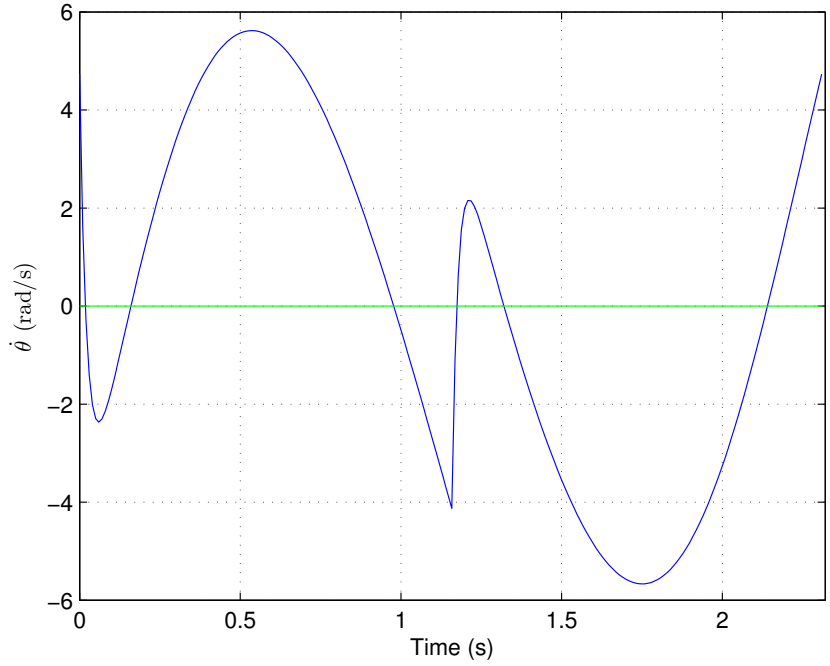


Fig. 3.38: Reel-rate curve for the asymmetric, tether controller in Fig. 3.36

Varying the phase triggers and tension in each phase with a "steepest descent" optimal pattern produced the current solution with a comparable power to other solutions found.

The trigger locations are symmetrical with the reel-in trigger at 90^o while the reel-out trigger is at 270^o. From the tension input found in Fig. 3.37, the reel-out tension is 540N, and the reel-in tension is 35N.

The reel-rate curve, shown in Fig. 3.38, follows the same pattern as the previous reel-rate curve with a symmetrical parameter set. The phenomenon is much more apparent during each reel-rate change at the trigger locations.

As shown in Fig. 3.39, the power curve for the higher power system has a much higher amount of consumption compared to the symmetrical system. The power ratio for the system is 3.98. The cause for a higher average cycle power is due to the higher instantaneous power achieved from a higher tension during reel-out, but also from

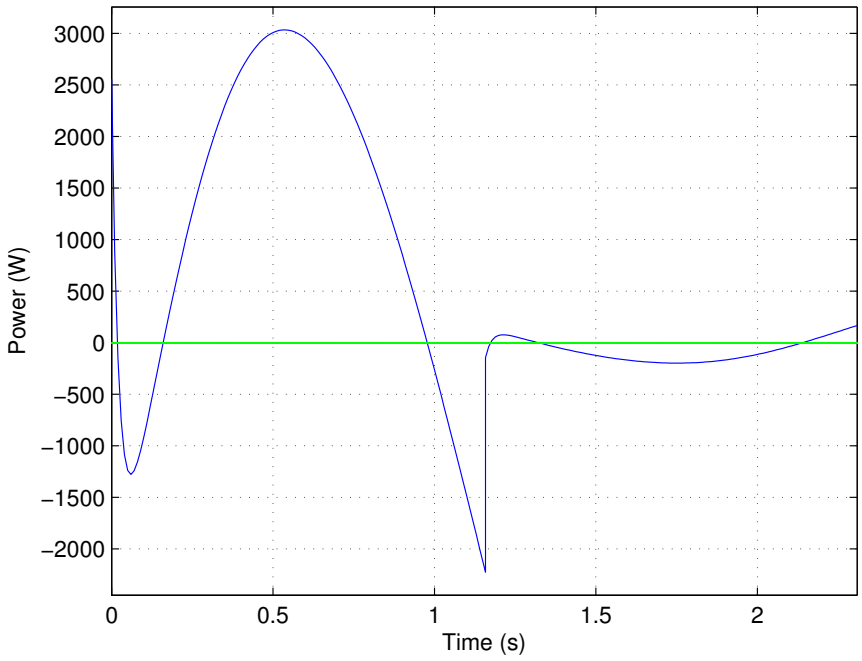


Fig. 3.39: Instantaneous Power for the asymmetric, tether controller in Fig. 3.36

the decreased time needed to complete a steady, periodic cycle because of increased tension during reel-in. The power ratio and average cycle power are consistent with desirable values. The power ratio is much higher than 1 and the average cycle power is large. This ratio and net cycle power pair show a large disparity between generated and consumed power while the net power is comparable to the magnitudes of power during each phase.

3.8.2 Stable Periodic Solutions

The tension controller is very similar to the initial reel-rate controller, but tends to find periodic solutions that are stable. Figure 3.40 shows the initial condition difference curves for Eq. (2.20), as discussed in Sections 2.5 and 3.6. The solution is found at $\dot{\theta} = 2.97$ rad/s and $l = 30.56$ m while the differences curves lie side-by-side from $l = 15$ m to $l = 35$ m.

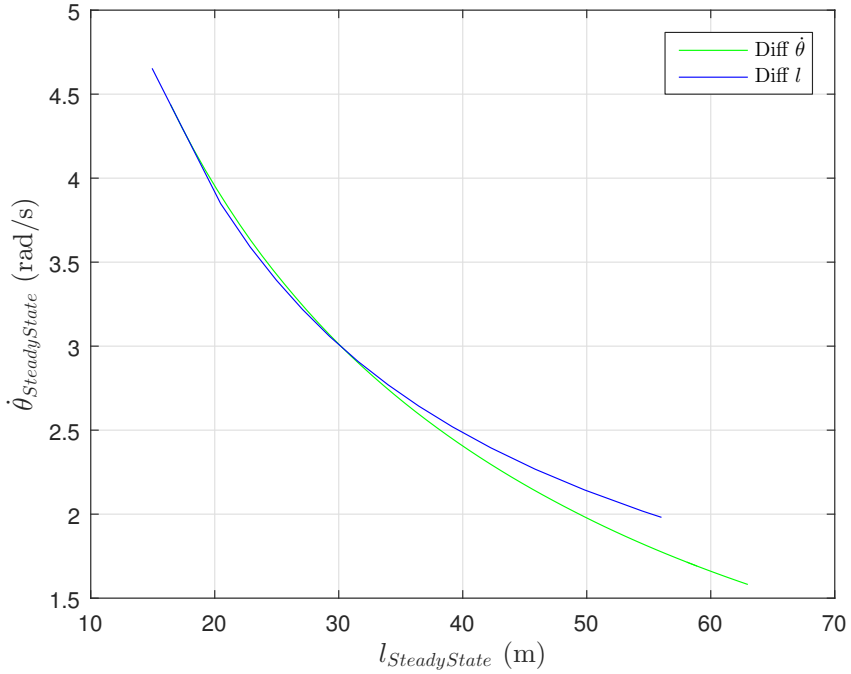


Fig. 3.40: Single variable difference solution curves for specific steady state tether length domain that shows family of solutions behavior

Figure 3.41 shows that the difference between $\dot{\theta}$ and l for each cycle, up to 750 cycles, is 0 m or 0 rad/s after the initial transient period and while operating in steady state motion. The zero initial condition difference shows that the system will retain the periodic motion while system variables are unaltered.

The system also shows stable behavior by creating an "attraction" area surrounding the periodic motion solution. In Fig. 3.42, the same four quadrant solution area procedure as discussed in Section 3.6 was performed for the tether tension controller solution in Fig. 3.36. Each quadrant shows there is a convergence area surrounding the periodic motion solution verified by the multidimensional root find. Depending on the initial $\dot{\theta}$, the system would shoot towards the tether length associated with the $\dot{\theta}$ achieved after the first cycle, as seen in Fig. 3.42. Then with each passing cycle, the transient phases converged upon the same periodic motion.

Based upon the single variable routine and verified with the standard model in

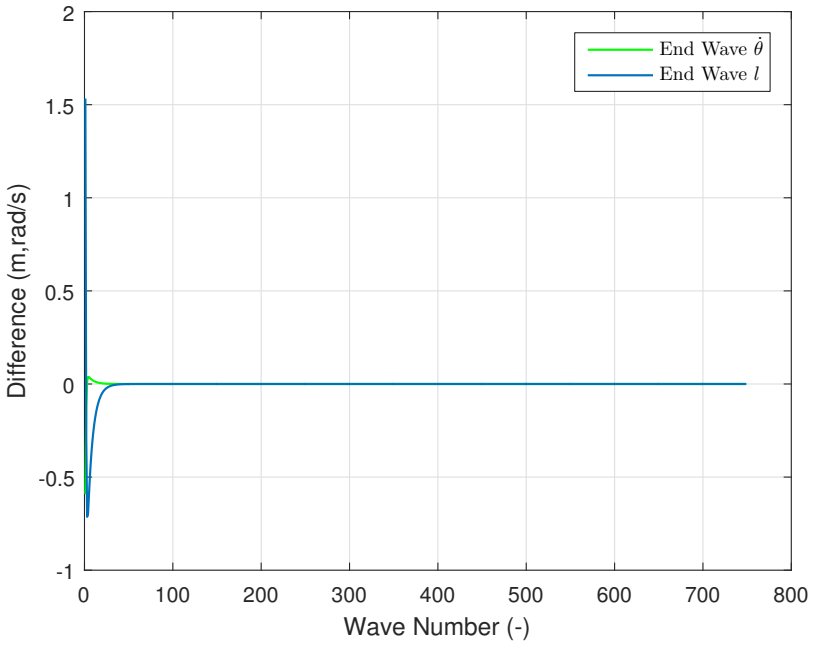


Fig. 3.41: The end wave value differences for $\dot{\theta}$ and l measured at the reel-out trigger between each successive cycle showing that the system maintains constant behavior for up to 750 cycles

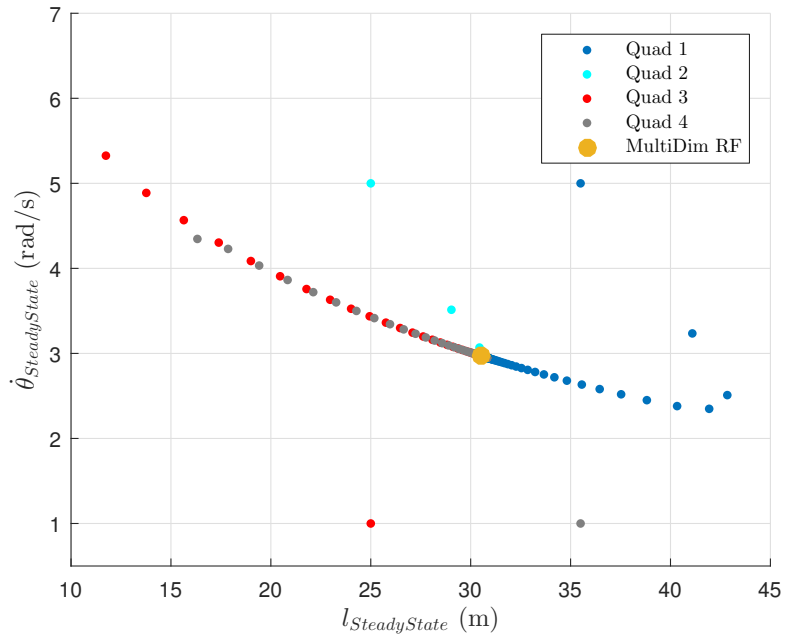


Fig. 3.42: The end wave value for $\dot{\theta}$ and l of four different simulations with varying initial conditions leading into convergence towards the multidimensional zero error crossing solution

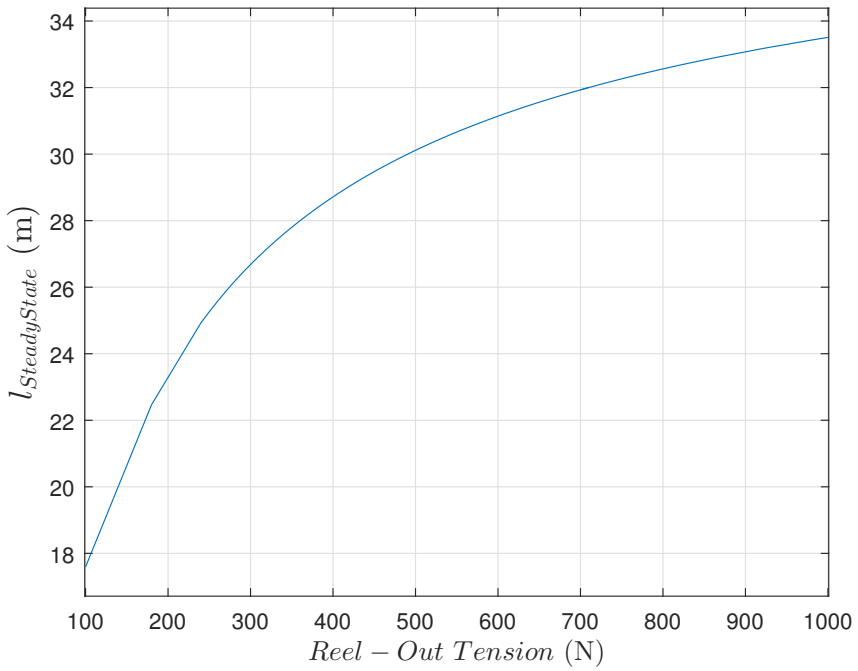


Fig. 3.43: Parameter variation of tether controller system showing tether length while varying reel-out tether tension

Fig. 3.43, the single parameter family of solutions behavior is similar with the tether tension controller. For each reel-out tension, there is a specific steady state length for periodic motion and there is a family of solutions with varying initial conditions. However, usually within two cycles the system $\dot{\theta}$ and l lie on the single parameter curve to converge towards the stable periodic motion. The periodic motion relationship between $\dot{\theta}$ and l is the same (*i.e.* $\dot{\theta}$ increases as the l decreases).

Minimum tension is specified as an initial condition and, therefore, any solution found with a minimum tension >0 N is a viable solution for the system. Figure 3.44 shows that net average cycle power has a parabolic relationship when varying reel-out tension. T maximum for net average cycle power can then be used to determine the best combinations for power generation. Other parameter sensitivity graphs for the tether tension controller can be found in Appendix B.

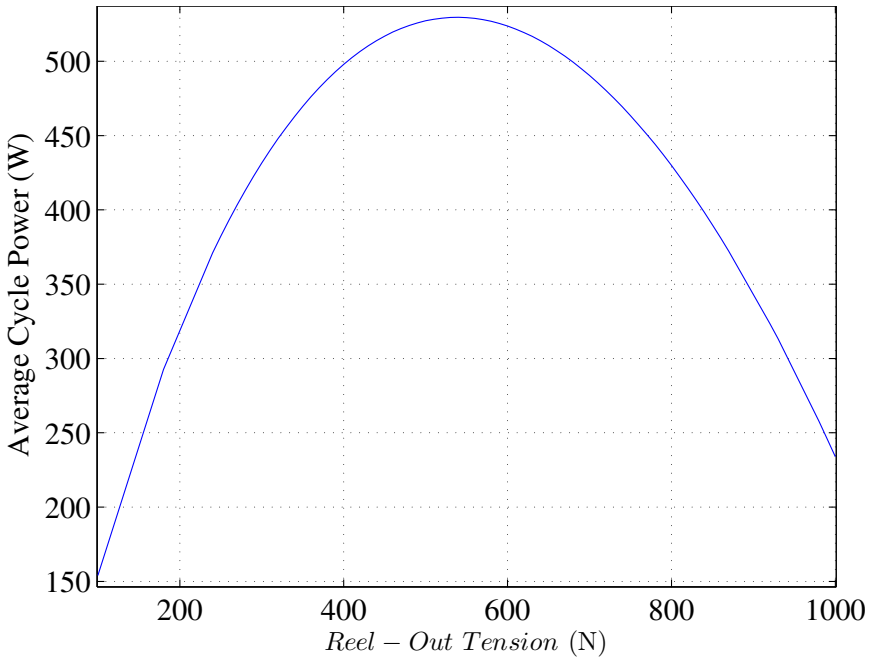


Fig. 3.44: Parameter variation of tether controller system showing net average cycle power while varying reel-out tether tension

3.9 Conclusions

The 2D simulation reel-rate controller and line tension controllers are capable of providing periodic and, in some cases, stable, periodic motion while generating a net positive average cycle power. Stable and unstable convergence solutions are located throughout the parameter space where, at those initial condition locations, changes are non-existent in the initial conditions of angular velocity ($\dot{\theta}$) and tether length (l), or there is very little change over time. It is also shown that it is possible to easily achieve periodic motion at system initialization or if the system motions were to be interrupted. Several parameters are available to manipulate the system to behave in manners desired by designers, and vary in magnitude with their affect on the system behavior. The results are promising and suggest that further research is warranted to provide a conclusion for system feasibility.

Chapter 4

Altitude Inclusion

The results from the 2D model are promising and provides the opportunity to observe the dynamic behavior of the system with regard to basic operation during crosswind motion upwind of the base station, tether reel-in and reel-out phases for energy generation, basic knowledge of possible periodic motions through use of different control methods, and possible forces applied to the tether from the kite during flight.

However, the 2D model does not portray the system while in flight above ground and the possibility of the kite to lose altitude from lack of lift. Therefore, this 3D model is pursued to provide an understanding of the basic operation of flight. The 3D model resembles many aspects of the 2D model and, in most aspects, it is the 2D model directly propped up from the ground (*i.e.* the tether dynamics are not considered, tether twist is neglected, the kite is fixed with the same angles from the tether throughout simulation, and tether can support non-zero compression forces.

In other aspects, the models are much different. Altitude is controlled by a separation of the lift vector, which is now broken into two components to create vertical and horizontal directional forces, and a new angle between the kite span line and tether will allow a rotation of the kite to experience greater vertical lift.

4.1 Model Description

4.1.1 Flight Dynamics

With many factors from the 2D model considered the same for the 3D model, the derivations for coefficients of lift and drag and the finite wing correction factor from Section 2 are valid, reel-rates are set to invoke crosswind motion, and phase regions are regulated by tether angular position θ in the $\hat{i}\hat{j}$-directions. All other derivations that differ are found below with assumptions listed in Table 4.1.

The system schematic is found in Fig. 4.1 and shows the tether angular velocity (θ), the altitude angle (ϕ), and the rotational coordinate system of the kite ($\hat{e}$). The system will rotate about the origin in the same positive θ direction measured from the X-axis. The system will rotate about a the XY-plane with an altitude angle (ϕ). Figure 4.2 shows the back edge of the kite with the leading edge hidden from view. As the kite span line is rotated clockwise about $\hat{e}_\theta$, the angle ζ between the span line and $\hat{e}_\phi$ grows larger to bank the plane. The tether vector is aligned with $\hat{e}_r$ and is directed toward the origin. The forward direction of the kite, $\hat{u}_\theta$, will align with $\hat{e}_\theta$ in this model due to the axis locking and neglecting the angle between the tether and the chordline (β).

The apparent wind velocity vector ($\vec{V}_A$) is shown in Fig. 4.3 with the chord wise flow velocity vector ($\vec{V}_{A_{aero}}$). The vector is the apparent wind velocity parallel with the chord line of the kite. Span wise flow disrupts the chord wise air flow over the kite, affects the assumed established flow over the airfoil, and changes the Reynold's

Table 4.1: Assumptions to simplify model to determine system feasibility and simplify areas to reduce numerical computation

	3D Simulation Assumptions
1	Assumptions from Table 2.1 except #12
2	Angle between tether and spanwise line ($\hat{u}_s$ in Fig. 4.2) is constant
3	Spanwise flow is neglected (Lift and Drag Components calculated by chordwise flow)

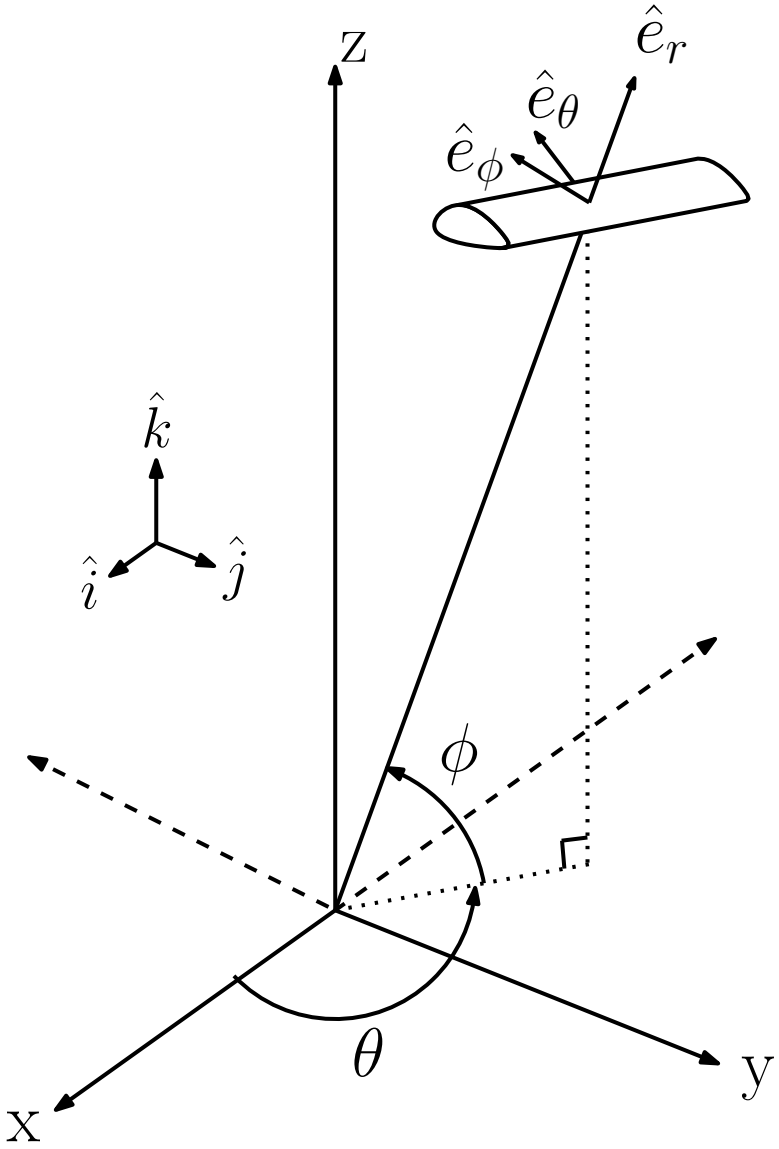


Fig. 4.1: System schematic for a 3D model

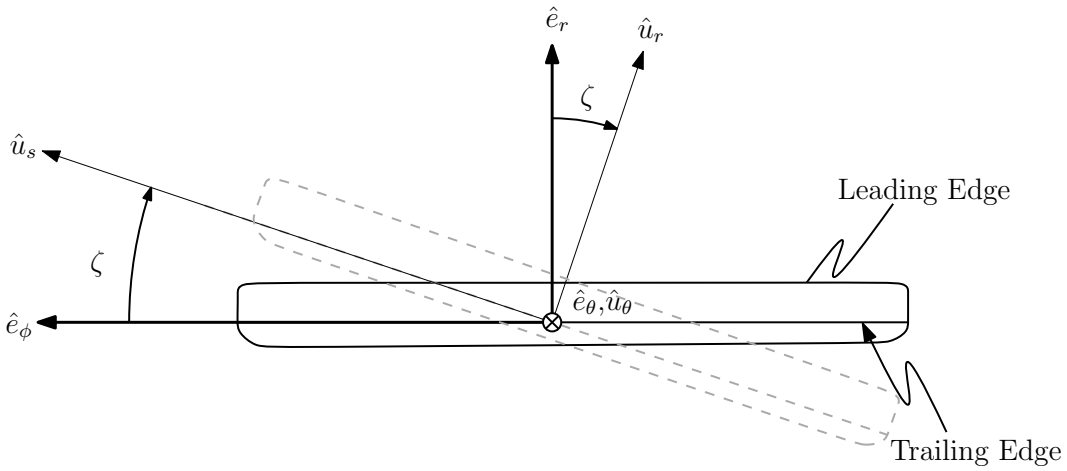


Fig. 4.2: Back view of the kite designating difference between $\hat{e}$ coordinate system and $\hat{u}$

numbers from our assumed values from [20].

The Equation of Motion for the 3D system is shown in Fig. (4.1). The forces, shown in Fig. 4.4, consist of an optional propulsion force in the direction of the rotational coordinate system vector ($\hat{e}_\theta$), the tether tension (T) in the negative rotational coordinate system vector ($\hat{e}_r$), lift (L) and drag (D) forces in their respective unit vector directions ($\hat{\lambda}_L$ and $\hat{\lambda}_D$), and the gravity (g) force applied to the mass (m) of the kite. These forces are summed to the mass and acceleration of the kite ($\vec{a}_{kite}$).

$$F_p\hat{e}_\theta - T\hat{e}_r + D\hat{\lambda}_D + L\hat{\lambda}_L - mg\hat{k} = m_{kite}\vec{a}_{kite} \tag{4.1}$$

The tether position or kite position vector, Eq. (4.2) is the same for the 2D model which is the tether length (l) multiplied by the tether vector ($\hat{e}_r$).

$$\vec{r}_{P/O} = l\hat{e}_r \tag{4.2}$$

The rotational component for the kite velocity vector, Eq. (4.3) now contains the cross product between rotational velocity and the tether vector ($\vec{\omega}x\hat{e}_r$).

$$\vec{v}_{kite} = \dot{l}\hat{e}_r + l(\vec{\omega} \times \hat{e}_r) \tag{4.3}$$

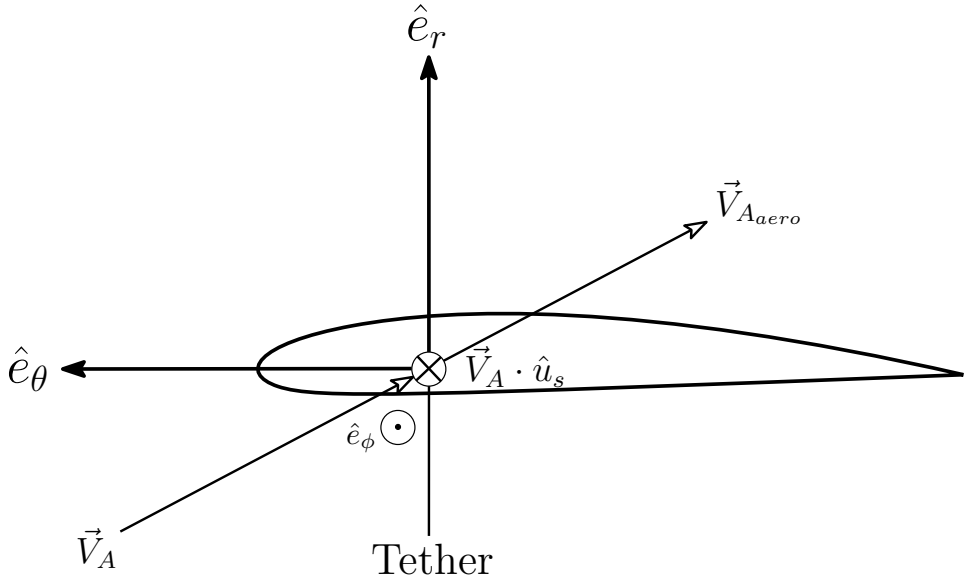


Fig. 4.3: Side view of the kite designating difference between $\hat{e}$ coordinate system and $\hat{u}$

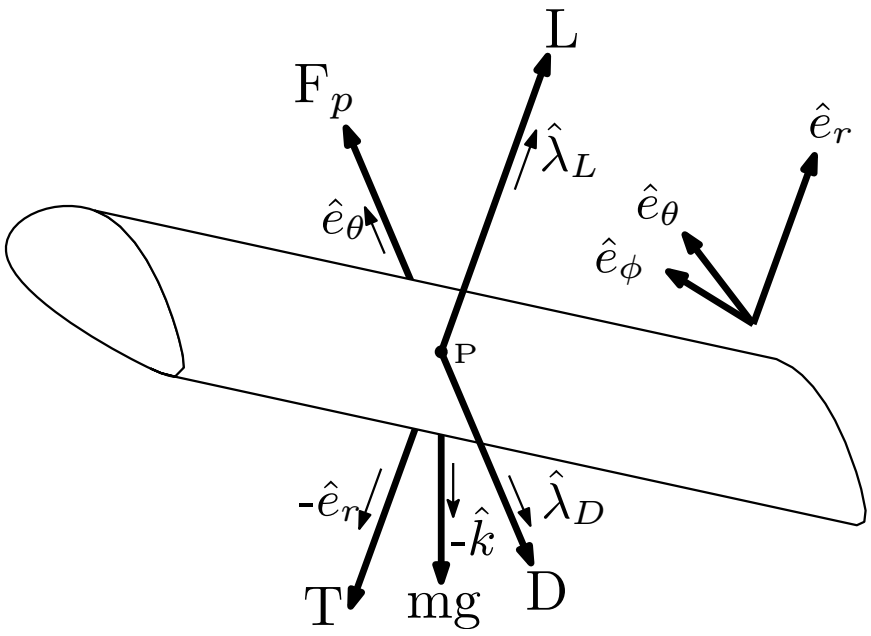


Fig. 4.4: Free body diagram of the kite and associated unit vectors

The cross product is the derivative of $\hat{e}_r$, found in Eq. (4.4).

$$\frac{\partial}{\partial t}(\hat{e}_r) = \vec{\omega} \times \hat{e}_r \tag{4.4}$$

The rotational velocity is the multiplication of the XY-plane tether angular velocity $\dot{\theta}$ and the $\hat{k}$-coordinate direction subtracted by the multiplication of tether altitude angular velocity ($\dot{\phi}$) and e_θ rotational coordinate system unit vector, as shown in Eq. (4.5).

$$\vec{\omega} = \dot{\theta}\hat{k} - \dot{\phi}\hat{e}_\theta \tag{4.5}$$

Kite velocity, Eq. (4.3), and kite acceleration, Eq. (4.6), include l as tether length; $\dot{l}$ as reel-rate; $\ddot{l}$ as reel-rate rate of change; and $\vec{\alpha}$ as tether angular acceleration in Eq. (4.7).

$$\vec{a}_{kite} = \ddot{l}\hat{e}_r + 2\dot{l}(\vec{\omega} \times \hat{e}_r) + l(\vec{\alpha} \times \hat{e}_r) + l(\vec{\omega} \times (\vec{\omega} \times \hat{e}_r)) \tag{4.6}$$

The tether angular acceleration in Eq. (4.7) includes $\ddot{\theta}$ as tether $\hat{i}\hat{j}$-direction angular acceleration; $\dot{\phi}$ as altitude angular velocity; and $\ddot{\phi}$ as altitude angular acceleration.

$$\vec{\alpha} = \ddot{\theta}\hat{k} - \ddot{\phi}\hat{e}_\theta - \dot{\phi}(\vec{\omega} \times \hat{e}_\theta) \tag{4.7}$$

The cross product between $\vec{\omega} x \hat{e}_r$ is derived from the derivative of e_θ in Eq. (4.8).

$$\frac{\partial}{\partial t}(\hat{e}_\theta) = \vec{\omega} \times \hat{e}_\theta \tag{4.8}$$

The simulation neglects span wise flow, as shown in Eq. (4.9), by subtracting the value of the span wise flow apparent wind velocity ($\vec{V}_A \cdot \hat{u}_s$) from the apparent wind velocity vector ($\vec{V}_A$) in Eq. (1.2) to isolate the chord wise flow apparent wind velocity ($\vec{V}_{A_{aero}}$). $\hat{u}_s$ is the wingspan direction vector as shown in Fig. 4.5.

$$\vec{V}_{A_{aero}} = \vec{V}_A - \vec{V}_A \cdot \hat{u}_s \tag{4.9}$$

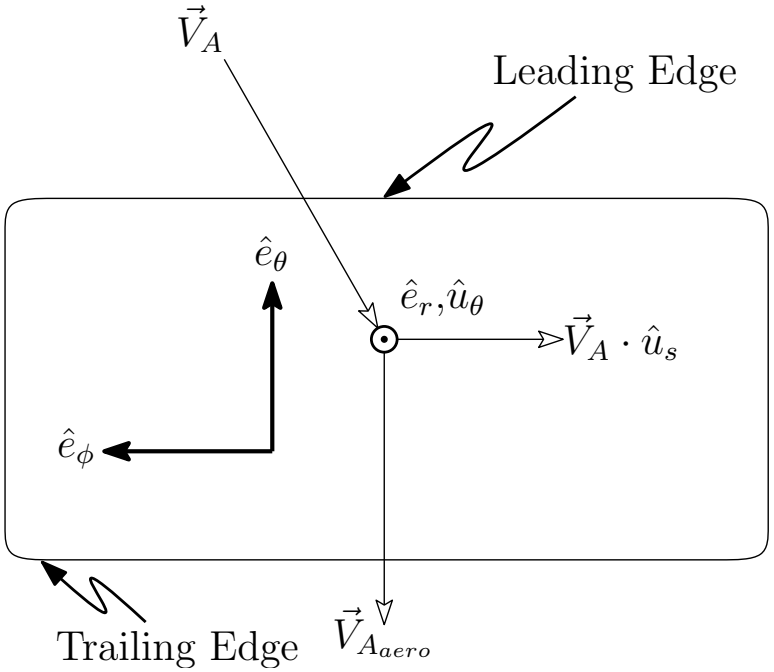


Fig. 4.5: Top view of the kite showing velocity vectors

Equation (4.10) is the unit vector for both the chord wise airflow and kite drag.

$$\hat{\lambda}_{aero} = \hat{\lambda}_D = \frac{\vec{V}_{a_{aero}}}{|\vec{V}_{a_{aero}}|} \tag{4.10}$$

The cross product of $\hat{u}_s$ and $\vec{V}_{a_{aero}}$ provides the unit vector for the kite lift, as shown in Eq. (4.11).

$$\hat{\lambda}_L = \hat{u}_s \times \vec{V}_{a_{aero}} \tag{4.11}$$

The angle of attack is calculated using the vector notation for law of cosines with $\hat{u}_\theta$ as the chord line vector that is analogous to the rotational coordinate vector ($\hat{e}_\theta$).

$$\alpha = \arccos(\frac{\vec{V}_{A_{aero}} \times \hat{u}_\theta}{V_{A_{aero}} \cdot \hat{u}_\theta}) \tag{4.12}$$

The lift and drag equations are considered after the Anderson Method transformation for coefficients of lift and drag. The resultants are the equations in Eq. (4.13).

$$L = \frac{1}{2} S \rho V_{A_{aero}}^2 C_L \qquad D = \frac{1}{2} S \rho V_{A_{aero}}^2 C_D \tag{4.13}$$

Equation (4.14) contains the transformation matrices to change vectors from the $\hat{i}$, $\hat{j}$, and $\hat{k}$ coordinate system into the rotational coordinate system $\hat{e}_r$, $\hat{e}_\theta$, and $\hat{e}_\phi$.

$$\begin{bmatrix} \hat{e}_r \\ \hat{e}_\theta \\ \hat{e}_\phi \end{bmatrix} = \begin{bmatrix} \cos\phi & 0 & \sin\phi \\ 0 & 1 & 0 \\ -\sin\phi & 0 & \cos\phi \end{bmatrix} \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{i} \\ \hat{j} \\ \hat{k} \end{bmatrix} \tag{4.14}$$

The transformation matrix to transform the rotational coordinate system to the kite body coordinate system $\hat{u}_r$, $\hat{u}_\theta$, and $\hat{u}_s$ is found in Eq. (4.15).

$$\begin{bmatrix} \hat{u}_r \\ \hat{u}_\theta \\ \hat{u}_s \end{bmatrix} = \begin{bmatrix} \cos\zeta & 0 & -\sin\zeta \\ 0 & 1 & 0 \\ \sin\zeta & 0 & \cos\zeta \end{bmatrix} \begin{bmatrix} \hat{e}_r \\ \hat{e}_\theta \\ \hat{e}_\phi \end{bmatrix} \tag{4.15}$$

4.1.2 Integration Tolerance Convergence

The convergence plot in Fig. 4.6 shows that the system will approach the acceptable value of $\theta = 1.84259$ rad with a tolerance level of 10^{-5} or below. The 3D simulations were set at a tolerance level of 10^{-8} to maintain accurate results.

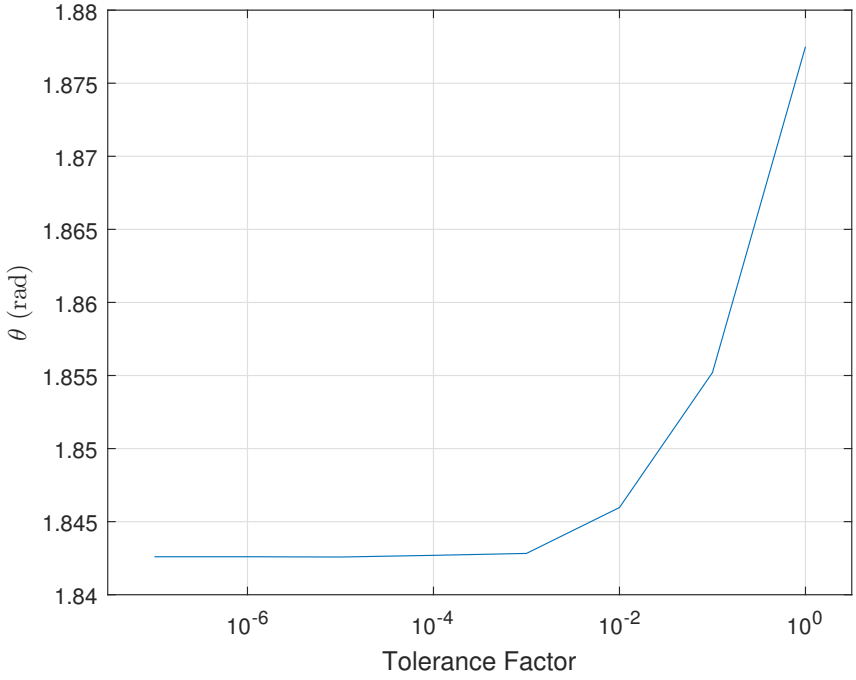


Fig. 4.6: Convergence plot relating to accurate approximation value of *theta* with increasing numerical accuracy

4.1.3 3D Simulation Verification

Test cases to perform reality checks were used to determine the validity of the simulation similar to the methods provided in Section 2.6. The following verifications were analyzed along with several others found in Appendix C.

TEST CASE 1: The system was setup to allow behavior similar to a pendulum. The wind velocity was set = 0 m/s, lift and drag forces were set = 0 N, and gravity was set = 9.81 m/s^2. As shown in Fig. 4.7, the system falls and creates a pendulum while maintaining system height due to the zero loss of height reached on the upswing, as shown in Fig. C.1 in Appendix C.

TEST CASE 2: This test mimics the condition from the previous test case but returns lift and drag forces to their calculated values, as shown in Fig. 4.8. The lift and drag forces cause the airfoil to send the system into an almost erratic behavior.

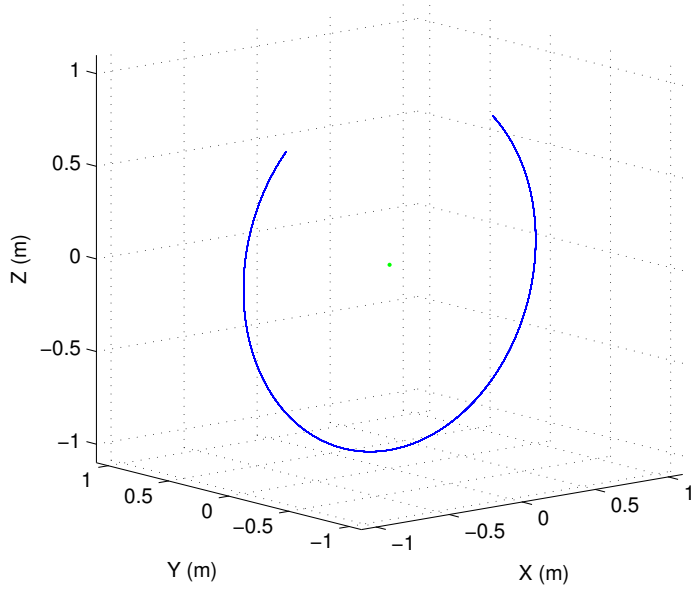


Fig. 4.7: Test case showing pendulum behavior of the model with wind velocity = 0 m/s, lift and drag forces = 0 N, and gravity = 9.81 m/s^2

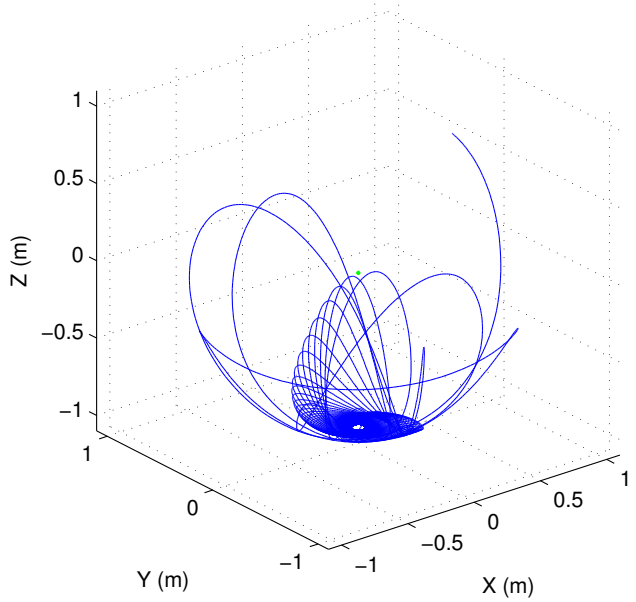


Fig. 4.8: Test case showing pendulum behavior of the model with wind velocity = 0 m/s, allowing lift and drag forces, and gravity = 9.81 m/s^2

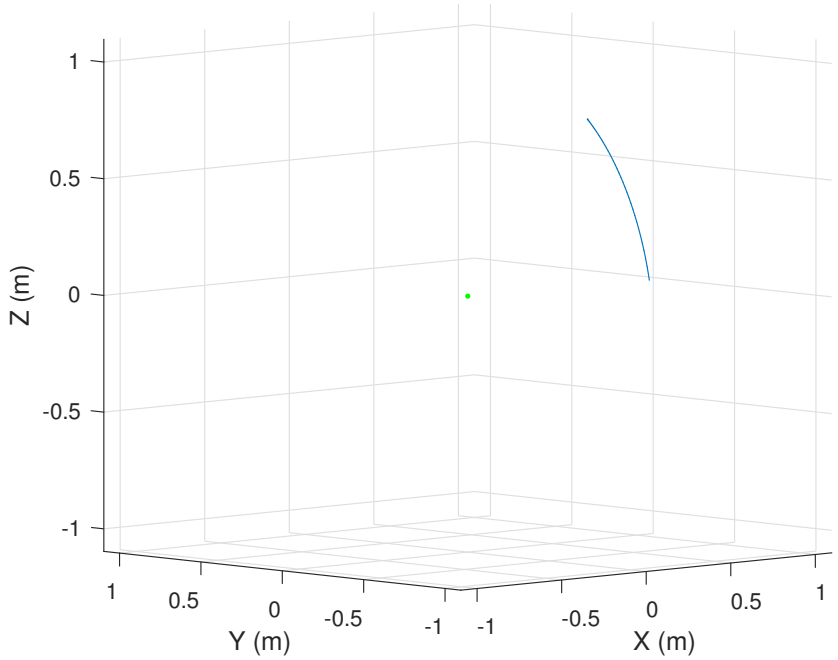


Fig. 4.9: Test case showing event detection of the simulation with wind velocity = 0 m/s, lift and drag forces = 0 N, gravity = 9.81 m/s^2, and the event set at $\phi = 0^o$ (*i.e.* ground level) ending numerical integration due to a crash

However, the radial distance from the center point is maintained and each pass is kept along the spherical boundary of the radial arm. From the initial point at the top of the image, the arm swings down continues to lose movement until spiraling to a stop towards the base of the sphere.

TEST CASE 3: The test case is to show that the system has predictable motion of a point mass on the end of a thin rod to end the simulation with an event trigger once the altitude angle (ϕ) is = 0 rad. The model is capable of ending the simulation after the kite is at ground-level, as shown in Fig. 4.9.

4.1.4 Steady State Convergence

Just as before with the 2D simulation, periodic motion is assumed when the system error satisfies the steady state (SS) criteria. The SS_{error} is now the absolute percent

difference between the tether length of the current and previous wave summed with the percent difference between $\dot{\theta}$ of the current and previous wave summed with the percent difference between $\dot{\phi}$ of the current and previous wave. Convergence occurs when the summation value is less than the convergence criteria (ε). ε is a predefined percent difference $< (10^{-5})$.

$$SS_{error} = \left| \frac{l_{current} - l_{previous}}{l_{previous}} \right| + \left| \frac{\dot{\theta}_{current} - \dot{\theta}_{previous}}{\dot{\theta}_{previous}} \right| + \left| \frac{\dot{\phi}_{current} - \dot{\phi}_{previous}}{\dot{\phi}_{previous}} \right| \leq \varepsilon \quad (4.16)$$

4.2 System Behavior

The tethered kite behavior of the system is unable to generate enough lift to keep the system in flight for the limited simulation parameter sets analyzed. Figure 4.10 shows the system starting at θ = -90^o with an incoming wind of 10 m/s from the $\hat{i}$ direction. The view is upwind of the base station with the tether starting at the red dot located at the origin and extending toward the pink asterisk. The kite begins at the asterisk point and continues to fall in a zig-zag fashion but begins with a counter-clockwise rotation about the base station typically found in the 2D simulations. Once reaching a tether angular position (θ in the $\hat{i}\hat{j}$-plane) = to 45^o, the kite begins a free-fall and rotates the opposite direction. Once establishing higher lift before returning to the original θ, it once again begins to rotate in the counter-clockwise direction to end with a final descent around $\theta = 40^o$.

4.3 Conclusions

The system is performing within expectation for the reality checks and fundamental derivations are proving correct. The model, however, is exaggerating the apparent wind velocity vector. A verification of the values entering and calculated from the

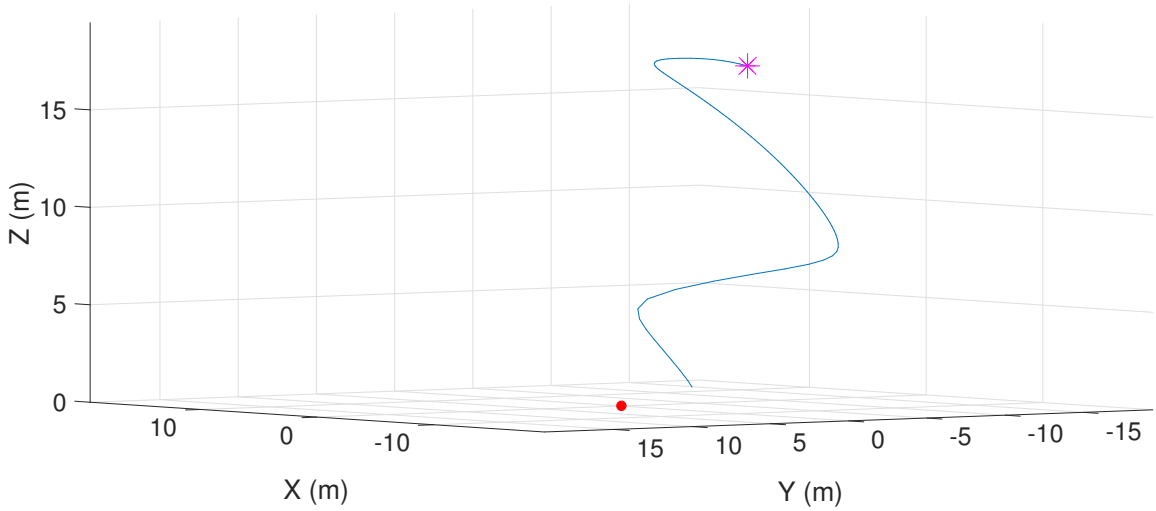


Fig. 4.10: Tethered kite without enough lift to maintain altitude and resulting in a crash

first step of the 3D dynamic equation numerical integration in comparison to the 2D value integration is necessary to determine if there is input error or the problem lies in the derivation of the 3D dynamic equations.

Chapter 5

Final Conclusions

5.1 Overall Conclusions

The model and simulation were designed to discover if periodic motions existed to generate a net positive average cycle power with the tethered kite. Results show that it is capable of producing net positive average cycle power with two different types of control systems, and that the net average cycle power is 500+ W with a kite design similar to an R/C plane. However, more research is needed to simulate the dynamics of a system comparable to a real life system.

Stable and unstable periodic motions were abundant and found for multiple parameters sets. The periodic motions are easily attainable but the observed regions with highest and most consistent power production came from reducing differences between successive cycle tether angular velocities and tether lengths.

Net positive cycle power is achieved by locating periodic motions that are consistent or are predictably deviating from the previous cycles at a relatively small rate for a large amount of cycles. The stable periodic solutions found with the tension controller are capable of converging from a broad range of initial conditions to set the dynamic motions of the system into a periodic pattern. The reel-rate controller dynamics will converge upon a predictable pattern of instability with very small changes in tether length and angular velocity of the kite for each cycle.

Once periodic motion is attained, the net average cycle power is calculated from the system behavior with most periodic solutions producing an average net 500-600 W per cycle.

Even though the model is lacking from certain parameters necessary to predict real-world dynamic behavior (*i.e.* the model is 2D, does not analyze moment forces on the kite, tether drag and weight are neglected, and the wind is stable/consistent at elevations known for variability in wind speed and direction) it still provides feasibility for wind assisted tether retraction, periodic behavior, stable periodic solutions, and a possibility for the system to consume less energy than it generates. The model must account for these conditions before real world system viability can be established, but it warrants more research into the development of a real world system or for a more advanced model.

5.2 Recommendation for Future Work

Several recommendations have been discussed in detail to improve the model. The next step is to analyze a 3D model to search for periodic motions, search if 2D model concepts of stable and unstable solutions carried over to the altitude model, and, if possible, determine if a net average cycle power is possible. If the system is proven to find periodic motions within the 3D model, a working prototype with controller systems to verify simulations should be constructed. The system is also designed to reach altitudes of 6000 ft (2000 m) so scaling issues will have large implications. The periodic motions found are occurring at tether lengths <35 m with the parameter variation showing the affect on minimum tension from certain variables. This might show great constraint on the system and should determined if these periodic solutions are subject to short tether lengths or if it is possible to extend to longer tether lengths and higher locations.

Bibliography

[1] Electric power monthly. Tech. rep., United States Dept. of Energy and Energy Info Administration.

[2] Energy technology perspectives - pathways to a clean energy system. Tech. rep., Organisation for Economic Co-operation and Development and International Energy Agency.

[3] Altaeros energies: Altaeros lighter than air vehicle. http://www.altaerosenergies.com/, 2015.

[4] Google x: Advanced wind energy: Energy kites. http://www.google.com/makani/solution/, 2015.

[5] Sky wind power: Higher airborne wind energy. http://www.skywindpower.com/, 2015.

[6] Anderson, J. D. J. *Introduction to Flight, Third Edition*. McGraw-Hill, Inc, New York, 1989.

[7] Argatov, I., Rautakorpi, P., and Silvennoinen, R. Apparent wind load effects on the tether of a kite power generator. *Journal of Wind Engineering and Industrial Aerodynamics 99*, 10 (Oct. 2011), 1079–1088.

[8] Ban, M., Perkovic, L., Duic, N., and Penedo, R. Estimating the spatial distribution of high altitude wind energy potential in southeast europe. *Energy 57* (Aug 2013), 24–29.

[9] Bryan W., R. Harnessing High-Altitude Wind Power. *IEEE TRANSACTIONS ON ENERGY CONVERSION 22*, 1 (Mar. 2007), 136–144.

[10] Canale, M., Fagiano, L., and Milanese, M. KiteGen: A revolution in wind energy generation. *Energy 34*, 3 (Mar. 2009), 355–361.

[11] Conti, J. J. Annual energy outlook 2014. Tech. rep., Office of Integrated and International Energy Analysis, United States Dept. of Energy and Energy Info Administration.

[12] Fagiano, L., Zgraggen, A. U., Morari, M., and Khammash, M. Automatic crosswind flight of tethered wings for airborne wind energy: Modeling, control design, and experimental results. *IEEE Transactions on Control Systems Technology 22*, 4 (2014), 1433–1447.

[13] Goldstein, L. Theoretical analysis of an airborne wind energy conversion system with a ground generator and fast motion transfer. *Energy 55* (June 2013), 987–995.

[14] Loyd, M. Crosswind kite power. *Journal of Energy 4*, 3 (May 1980), 106–111.

[15] Lozano, Jr., R., Dumon, J., Hably, A., and Alamir, M. Energy production control of an experimental kite system in presence of wind gusts. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2013), 3-7 Nov. 2013* (2013), 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2013), IEEE, pp. 2452–9.

[16] Masters, G. M. *Renewable and Efficient Electric Power Systems*. John Wiley and Sons Inc., Hoboken, New Jersey, 2013.

[17] Milutinovi, M., Kranjevi, N., and Deur, J. Multi-mass dynamic model of a variable-length tether used in a high altitude wind energy system. *Energy Conversion and Management 87* (Nov. 2014), 1141–1150.

[18] Press, Teukolsky, V. F. *Numerical Recipes in C++: The Art of Scientific Computing, Second Edition*. Press Syndicate of the University of Cambridge, 40 West 20th Street, New York, NY 10011-4211, 2002.

[19] Ruiterkamp, R. Glider for airborne wind energy production (ampyx), Feb. 2014.

[20] Sheldahl, R. E., and Klimas, P. C. Aerodynamic characteristics of seven symmetrical airfoil sections through 180-degree angle of attack for use in aerodynamic analysis of vertical axis wind turbines. Tech. rep., U.S. DOE Sandia National Laboratories.

[21] Tieleman, H. W. Strong wind observations in the atmospheric surface layer. *Journal of Wind Engineering and Industrial Aerodynamics 96*, 1 (Jan. 2008), 41–77.

Appendix A

Simulation Code

A.1 Kite Dynamic Functions

A.1.1 Numerical Integrating Solver

```
%-------------------------------------------------------%
%*MainFile*.m                                           %
%                                                       %
%Kite Simulation of a horizontal path, microns off the  %
%ground. The simulation solves for theta, theta_d, l,   %
%l_d, and tension. The simulation involves lift and drag%
%forces and has one animation for the top view of a     %
%flight path with a simple FBD.                         %
%-------------------------------------------------------%

clear all;
close all;
clc;

global D L l_dd Vinfin m Fp C_LiftDragFile B density Vinf_mag S AR u;
global arccomp k wave_count T SYS_time SYS_theta SYS_theta_d SYS_l SYS_l_d;
global LiftVect DragVect Vrel magg_Vkite VkiteVect theta_in theta_out;

%------------------------------ODE Solution---------------------------------

i = [1,0,0];  %Basic Vector assignment for unit
j = [0,1,0];  %vectors i,j,k
k = [0,0,1];

%Excel Document that contains the lift and drag
%coefficients at 5degree intervals
C_LiftDragFile = xlsread('NACA0015.xlsx');
```

```
28
29
30 t_scale = 0:0.01:1000;      %ODE TimeScale
31 n_waves = 100;
32
33 %System Variables
34 m = 1.2;    %kg          : Mass of the "Kite" (Point Particle).
35 B = pi()/2; %deg         : Angle of Chord Vector w/ Tether Vector.
36 density = 1.225; %kg/m^3 : Wind Power Data Standard [15C @ 1atm].
37 b = 1.2;    %m           : Wing Span (Based on SenDesign AirFoil)
38 c = 0.2;    %m           : Chord Length (Based on SenDesign AirFoil)
39 S = b*c;    %m^2         : Effective Area of Wing
40 AR = b^2/S; %             : Aspect Ratio (dimensionless)
41 u = 1.7976*10^(-5); %kg/(m*s)    : Dynamic Viscosity
42 Fp = 0;     %N           : A perendicular force to the kite
43
44
45 %Defined Parameters
46 Vinfin = 10*i;  %m/s    : Wind Velocity.
47 Vinf_mag = norm(Vinfin); %     : Magnitude of the wind velocity
48 l_d_out = 2.07;          %m/s  : Length Velocity of Tether during Reel-Out
49 l_d_in = -3.75;          %m/s  : Length Velocity of Tether during Reel-In
50 theta_in = 130*pi/180;   %rad  : Theta-In Trigger Location
51 theta_out = 260*pi/180;  %rad  : Theta-Out Trigger Location
52
53 %Initial Conditions
54 theta = theta_out - 2*pi;%rad  : Angle Position at t=0
55 theta_d = 0.8761;        %rad/s: Angle Velocity at t=0
56 l = 118.07;              %m    : Length of Tether at t=0
57 l_d = l_d_out;           %m/s  : Length Velocity of Tether at t=0
58 l_dd = 0;                %m/s^2: Acceleration of Tether Release at t=0
59 D = 0;                   %N    : Drag Force Initial Definition
60 L = 0;                   %N    : Lift Forces Initial Definition
61
62 %System Arrays for Entire Simulation Period
63 SYS_time = 0;
64 SYS_theta = theta;
65 SYS_theta_d = theta_d;
66 SYS_l = l;
67 SYS_l_d = l_d;
68 [~,sizeSYS] = size(SYS_theta);
69
70 %Relative and Absolute Tolerances for ODE45 and Event Function Call
71 my_RelTol = 1e-10;
72 my_AbsTol = 1e-10;
73
```

```
74 %SS Tolerance Criteria
75 ODE_kill = 0;
76 ODE_kill_end = 2;
77 SS_tol = 1e-6;
78
79 fprintf('theta_d_init: %g l_init: %d \r WaveCount: ', theta_d, l)
80
81 %Initial Definition for End Wave Transient Response
82 End_Wave_l_Trans = l;
83 End_Wave_theta_d_Trans = theta_d;
84
85
86 %ODE45 Solution Iterations for Theta and L using Event Detection
87     % ODE45 Solver with Time and 4 Arc Components: theta, theta_dot,
88     % length, and length_dot. The function calls KITE_SS where states
89     % are used to incrementally iterate the listed Arc Components
90
91 for wave_count = 1:n_waves
92
93
94     fprintf('%d ', wave_count);
95
96
97
98     %1st Event ODE45 Solver
99         options = odeset('RelTol',my_RelTol,'AbsTol',my_AbsTol,'Events',@PI_2);
100        [t1,arccomp] = ode45(@KITE_SS,t_scale,[SYS_theta(sizeSYS),SYS_theta_d(
               sizeSYS),SYS_l(sizeSYS),SYS_l_d(sizeSYS)],options);
101
102        %Partial Wave Fragments from 1st Event Solver
103        t2 = t1(:) + SYS_time(sizeSYS);
104        time = transpose(t2);theta = transpose(arccomp(:,1));theta_d = transpose
               (arccomp(:,2));l_wave = transpose(arccomp(:,3));l_d = transpose(
               arccomp(:,4));
105
106        %Last Wave (Pass) Array for State Components
107        if ODE_kill > (ODE_kill_end-1)
108            LAST_WAVE_time = transpose(t1);
109            LAST_WAVE_theta = theta;
110            LAST_WAVE_theta_d = theta_d;
111            LAST_WAVE_l = l_wave;
112            LAST_WAVE_l_d = l_d;
113        end
114
115        %Total System Arrays addition of Partial Wave Fragment
116        SYS_time = [SYS_time time];
```

```
117         SYS_theta = [SYS_theta theta];
118         SYS_theta_d = [SYS_theta_d theta_d];
119         SYS_l = [SYS_l l_wave];
120         SYS_l_d = [SYS_l_d l_d];
121
122         [~,sizeSYS] = size(SYS_theta);
123         [~,sizeL_D] = size(l_d);
124
125         %New Controller Input - Change Length Velocity for Reel-In
126         SYS_l_d(sizeSYS) = l_d_in;
127
128     %Steady State Tolerance Error Definition for Start Wave
129         Start_Wave_l(wave_count) = l_wave(1);
130         Start_Wave_theta_d(wave_count) = theta_d(1);
131
132
133     %2nd Event ODE45 Solver
134         options = odeset('RelTol',my_RelTol,'AbsTol',my_AbsTol,'Events',@PI3_2);
135         [t1,arccomp] = ode45(@KITE_SS,t_scale,[SYS_theta(sizeSYS),SYS_theta_d(
                sizeSYS),SYS_l(sizeSYS),SYS_l_d(sizeSYS)],options);
136
137         %Partial Wave Fragments from 2nd Event Solver
138         t2 = t1(:) + SYS_time(sizeSYS);
139         time = transpose(t2);theta = transpose(arccomp(:,1));theta_d = transpose
                (arccomp(:,2));l_wave = transpose(arccomp(:,3));l_d = transpose(
                arccomp(:,4));
140
141         %Last Wave (Pass) Array for State Components
142         if ODE_kill > (ODE_kill_end-1)
143             [~,sizeLAST_WAVE] = size(LAST_WAVE_time);
144             t3 = t1(:) + LAST_WAVE_time(sizeLAST_WAVE);
145             LAST_WAVE_time = [LAST_WAVE_time transpose(t3)];
146             LAST_WAVE_theta = [LAST_WAVE_theta theta];
147             LAST_WAVE_theta_d = [LAST_WAVE_theta_d theta_d];
148             LAST_WAVE_l = [LAST_WAVE_l l_wave];
149             LAST_WAVE_l_d = [LAST_WAVE_l_d l_d];
150         end
151
152         %Total System Arrays addition of Partial Wave Fragment
153         SYS_time = [SYS_time time];
154         SYS_theta = [SYS_theta theta];
155         SYS_theta_d = [SYS_theta_d theta_d];
156         SYS_l = [SYS_l l_wave];
157         SYS_l_d = [SYS_l_d l_d];
158
159         [~,sizeSYS] = size(SYS_theta);
```

```
160         [~,sizeL_D] = size(l_d);
161
162         %New Controller Input - Change Length Velocity for Reel-Out
163         SYS_l_d(sizeSYS) = l_d_out;
164
165         %End Wave Transient Response
166         End_Wave_l_Trans = [End_Wave_l_Trans  l_wave(sizeL_D)];
167         End_Wave_theta_d_Trans = [End_Wave_theta_d_Trans  theta_d(sizeL_D)];
168
169      %Steady State Tolerance Error Calculation and End Wave Definition
170         End_Wave_l(wave_count) = SYS_l(sizeSYS);
171         End_Wave_theta_d(wave_count) = SYS_theta_d(sizeSYS);
172
173         SS_error_l(wave_count) = abs((End_Wave_l(wave_count) - Start_Wave_l(
                wave_count)) / Start_Wave_l(wave_count));
174         SS_error_theta_d(wave_count) = abs((End_Wave_theta_d(wave_count) -
                Start_Wave_theta_d(wave_count)) / Start_Wave_theta_d(wave_count));
175         Total_error(wave_count) = SS_error_l(wave_count) + SS_error_theta_d(
                wave_count);
176
177         %Steady State Convergence Function Kill Adder
178         if Total_error(wave_count) <= SS_tol
179             ODE_kill = ODE_kill+1;
180         elseif ODE_kill == 1
181             ODE_kill = 0;
182         end
183
184         %Steady State Convergence Function Kill
185         if ODE_kill > ODE_kill_end
186             break
187         elseif ODE_kill > 1
188             ODE_kill = ODE_kill+1;
189         end
190
191 end
192
193 %Determine Minimum Tension of the System
194 Min_Tension = Tension(1);
195
196 %Net Average Cycle Power
197 LAST_WAVE_Power_avg = Power(1);
198
199 %Animation
200 b = aniMATION(1);
```

A.1.2 State Space

```
%-------------------------------------------------------%
%KITE_SS.m                                              %
%                                                       %
%This file takes an input from *MainFile*.m to          %
%produce a numerical solution for four variables: theta,%
%theta_d, length, and length_d through the use of a     %
%State Space Model.                                     %
%-------------------------------------------------------%


function [myoutput,T] = KITE_SS(t,state)

    global D L l_dd Vinfin m Fp ang_attack B density Vinf_mag S;



    %Initial State Declarations
    theta = state(1);
    theta_d = state(2);
    l = state(3);
    l_d = state(4);

    % The angle of attack function is called sending the state-variables to
    % find the angle of attack in the current state. Angle must also be
    % sent in radians.

    %Read the Excel Sheet for Coeff of Lift and Drag
    output = angleOfAttack(theta,theta_d,l,l_d);
    ang_attack = output(1);
    [c_DL] = CoeffLiftDrag(ang_attack);

    L = 0.5*S*density*(theta_d*l)^2*c_DL(2);
    D = 0.5*S*density*(theta_d*l)^2*c_DL(1);

    % State Variables that depend on derivatives. These are taken from the
    % KiteSimul file as initial conditions and then incrementally found by
    % plugging the initials values into the equations for derivatives and
    % original with respect to the ODE45 solver. This solver will find the
    % value evaluated at variable stepsizes depending on activity of the
    % function.

    %State Changes Based on EOM
```

```
44      state_dot(1) = theta_d;
45      state_dot(2) = -((D*Vinf_mag*sin(theta))/((l_d*sin(theta) + l*theta_d*cos(
            theta))^2 + (Vinf_mag - l_d*cos(theta) + l*theta_d*sin(theta))^2)^(1/2)
            - Fp*sin(theta)^2 - (L*Vinf_mag*cos(theta))/((l_d*sin(theta) + l*theta_d
            *cos(theta))^2 + (Vinf_mag - l_d*cos(theta) + l*theta_d*sin(theta))^2)
            ^(1/2) - Fp*cos(theta)^2 + (L*l_d*cos(theta)^2)/((l_d*sin(theta) + l*
            theta_d*cos(theta))^2 + (Vinf_mag - l_d*cos(theta) + l*theta_d*sin(theta
            ))^2)^(1/2) + (L*l_d*sin(theta)^2)/((l_d*sin(theta) + l*theta_d*cos(
            theta))^2 + (Vinf_mag - l_d*cos(theta) + l*theta_d*sin(theta))^2)^(1/2)
            + 2*l_d*m*theta_d*cos(theta)^2 + 2*l_d*m*theta_d*sin(theta)^2 + (D*l*
            theta_d*cos(theta)^2)/((l_d*sin(theta) + l*theta_d*cos(theta))^2 + (
            Vinf_mag - l_d*cos(theta) + l*theta_d*sin(theta))^2)^(1/2) + (D*l*
            theta_d*sin(theta)^2)/((l_d*sin(theta) + l*theta_d*cos(theta))^2 + (
            Vinf_mag - l_d*cos(theta) + l*theta_d*sin(theta))^2)^(1/2))/(l*m*(cos(
            theta)^2 + sin(theta)^2));
46      state_dot(3) = l_d;
47      state_dot(4) = 0;
48
49      myoutput = [state_dot(1);state_dot(2);state_dot(3);state_dot(4)];
```

A.1.3 Angle of Attack Calculation

```
1  %-------------------------------------------------------%
2  %angleOfAttack.m                                       %
3  %                                                      %
4  %An input is taken from either *MainFile.m* or Kite_SS.m%
5  %to find the angle of attack based on theta, theta_d, l,%
6  %and l_d. The angle is calculated by subtracting gamma  %
7  %from our angle Beta the kite makes with the tether.    %
8  %-------------------------------------------------------%
9
10 function [output] = angleOfAttack(theta,theta_d,l,l_d)
11
12 global Vinfin B k;
13
14 %Er and Etheta components in terms of i,j,k
15 e_r = [cos(theta),sin(theta),0];
16 e_theta = [-sin(theta),cos(theta),0];
17
18 %Velocity and Acceleration Components
19 Vi = Vinfin;
20 Vkite = theta_d*l*e_theta+l_d*e_r;
21 Vrel = Vi-Vkite;
22
```

```
23 %Solving for Angle of Attack
24 k_vrelteth = (-Vrel(1))*(-e_r(2)) - (-Vrel(2))*(-e_r(1));
25 Vrel_unit = norm(-Vrel);
26 Teth_unit = norm(-e_r);
27 gamma = acos(dot(-Vrel,-e_r)/(Vrel_unit*Teth_unit));
28
29 %Unit Vectors of Lift and Drag
30 lambda_D = Vrel/Vrel_unit;
31 lambda_L = cross(k,lambda_D);
32
33 %Gamma Correction
34 if k_vrelteth < 0
35     gamma = -gamma;
36 end
37
38 %Solver for Gamma
39 ang_attack = B - gamma;
40 ang_attack = wrapTo2Pi(ang_attack);
41
42 if ang_attack > pi
43     ang_attack = ang_attack - 2*pi;
44 end
45
46 output = [ang_attack,Vrel(1),Vrel(2),lambda_D(1),lambda_D(2),lambda_L(1),
       lambda_L(2),Vkite(1),Vkite(2),Vkite(3)];
```

A.1.4 Lift and Drag Coefficient Interpolation

```
1  %-------------------------------------------------------%
2  %CoeffLiftDrag.m                                       %
3  %                                                      %
4  %An input for angle of attack is used to interpolate the%
5  %excel sheet opened in *MainFile*.m to find the        %
6  %coefficients of lift and drag. The excel sheet has    %
7  %corrected lift and drag coefficients based on the     %
8  %airfoil NACA0015.                                     %
9  %-------------------------------------------------------%
10
11
12 function [myoutput] = CoeffLiftDrag(ang_attck)
13         %Accept Angle of Attack in Radians
14
15 global C_LiftDragFile
16
```

```
%Interpolation for Coefficient of Drag and Lift
c_drag = interp1q(C_LiftDragFile(:,2),C_LiftDragFile(:,6),ang_attck);
c_lift = interp1q(C_LiftDragFile(:,2),C_LiftDragFile(:,5),ang_attck);

myoutput = [c_drag,c_lift];
```

A.1.5 Tension Calculation

```
%-------------------------------------------------------%
% Tension.m                                             %
%                                                       %
% For-Loop desinged to calculate tension corresponding %
% to values from the ODE45 found in the Arc Components %
% array. Tension isn't calculated in the state function%
% because it isn't based on a derivative of itself and %
% is only dependent upon the Arc Components.            %
%-------------------------------------------------------%

function [ output ] = Tension( input )

 global SYS_time SYS_theta SYS_theta_d SYS_l SYS_l_d T S density Vinf_mag;
 global l_dd m c u Re LAST_WAVE_T;

    %Matrix Size Declarations
    [~,t_size] = size(SYS_time); % : Size of Counter used in For-Loop
    T = zeros(1,t_size);         % : Declare Size of Tension
    Re = zeros(1,t_size);        % : Declare Size of Reynold's Number

    [f,g] = size(SYS_time);
    ang_attack = zeros(1,f);     % : Declare Size of Angle of Attack Array
    gamma = zeros(1,f);          % : Declare Size of Gamma Array

    %Determine Last Wave Dynamics if Simulation Converged
    if ODE_kill > ODE_kill_end
        [~,sizeLAST_WAVE] = size(LAST_WAVE_time);
        LAST_WAVE_count = 1;
    end

    %Iterative For-Loop for Size of System Arrays to calculate Tension
    for ind=1:t_size

        %Initial Conditions
        % Define theta, theta-dot, length, length-dot in For-Loop as
        % the respective value from ODE45 Calculations
```

```
37          theta = SYS_theta(1,ind);
38          theta_d = SYS_theta_d(1,ind);
39          l = SYS_l(1,ind);
40          l_d = SYS_l_d(1,ind);
41
42          % Angle of Attack Calculations are performed a second time to find
43          % values that correspond with theta, theta_d, length, and length_d
44          % since the angle isn't stored in an array when evaluated in the
45          % variable timestep ODE45 function
46          [output] = angleOfAttack(theta,theta_d,l,l_d);
47
48          ang_attack(ind) = output(1);
49          Vrel(1,ind) = output(2);
50          Vrel(2,ind) = output(3);
51
52          DragVect(1,ind) = output(4);
53          DragVect(2,ind) = output(5);
54          LiftVect(1,ind) = output(6);
55          LiftVect(2,ind) = output(7);
56
57          %Coefficient of Lift and Drag Calculation
58          [c_DL] = CoeffLiftDrag(ang_attack(ind));
59
60          L(1,ind) = 0.5*S*density*(theta_d*l)^2*c_DL(2);    %N
61          D(1,ind) = 0.5*S*density*(theta_d*l)^2*c_DL(1);    %N
62
63          %Kite Velocity Calculation - Necessary for the Animation
64              %Component Vectors of Vkite
65          magg_Vkite(ind) = norm([output(8),output(9),output(10)]);
66          VkiteVect(ind,:) = [output(8),output(9),output(10)]/magg_Vkite(ind);
67
68          %Tension and Reynolds Number Calculation
```

```
69          T(1,ind) = ((L(1,ind)*Vinf_mag*sin(theta))/((l_d*sin(theta) + l*theta_d*
                cos(theta))^2 + (Vinf_mag - l_d*cos(theta) + l*theta_d*sin(theta))
                ^2)^(1/2) - l_dd*m*sin(theta)^2 - l_dd*m*cos(theta)^2 + l*m*theta_d
                ^2*cos(theta)^2 + l*m*theta_d^2*sin(theta)^2 - (D(1,ind)*l_d*cos(
                theta)^2)/((l_d*sin(theta) + l*theta_d*cos(theta))^2 + (Vinf_mag -
                l_d*cos(theta) + l*theta_d*sin(theta))^2)^(1/2) - (D(1,ind)*l_d*sin(
                theta)^2)/((l_d*sin(theta) + l*theta_d*cos(theta))^2 + (Vinf_mag -
                l_d*cos(theta) + l*theta_d*sin(theta))^2)^(1/2) + (D(1,ind)*Vinf_mag
                *cos(theta))/((l_d*sin(theta) + l*theta_d*cos(theta))^2 + (Vinf_mag
                - l_d*cos(theta) + l*theta_d*sin(theta))^2)^(1/2) + (L(1,ind)*l*
                theta_d*cos(theta)^2)/((l_d*sin(theta) + l*theta_d*cos(theta))^2 + (
                Vinf_mag - l_d*cos(theta) + l*theta_d*sin(theta))^2)^(1/2) + (L(1,
                ind)*l*theta_d*sin(theta)^2)/((l_d*sin(theta) + l*theta_d*cos(theta)
                )^2 + (Vinf_mag - l_d*cos(theta) + l*theta_d*sin(theta))^2)^(1/2))/(
                cos(theta)^2 + sin(theta)^2);
70          Re(1,ind) = density*(theta_d*l)*c/u;
71
72          %If Convergence was Reached
73          if ODE_kill > ODE_kill_end
74              if ind >= (t_size - sizeLAST_WAVE + 1)
75                  LAST_WAVE_T(LAST_WAVE_count) = T(1,ind);
76                  LAST_WAVE_count = LAST_WAVE_count + 1;
77              end
78          end
79
80      end
81
82      %Minimum and Maximum Tensions
83      MAX_Tension = max(T);
84      MIN_Tension = min(T);
85
86      fprintf('\nTension - Max: %.2f N  Min: %.2f N\r', MAX_Tension, MIN_Tension);
87
88      output = MIN_Tension;
89
90  end
```

A.1.6 Average Net Cycle Power Calculation

```
1  %-------------------------------------------------%
2  % Power.m                                        %
3  %                                                %
4  % The net average cycle power for a single wave is %
5  % wanted to know what the system will produce once %
```

```
% reaching steady state, periodic motion. This     %
% function will run only if the system convergences%
%-------------------------------------------------%


function [ output ] = Power( input )

  global LAST_WAVE_T LAST_WAVE_l_d LAST_WAVE_time LAST_WAVE_P;

    if ODE_kill > ODE_kill_end

        [~,sizeLAST_WAVE] = size(LAST_WAVE_time);

        LAST_WAVE_P = times(LAST_WAVE_T,LAST_WAVE_l_d);
        LAST_WAVE_Energy_total = trapz(LAST_WAVE_time,LAST_WAVE_P);
        LAST_WAVE_Power_avg = LAST_WAVE_Energy_total / LAST_WAVE_time(
            sizeLAST_WAVE);

        fprintf('Last Wave Energy Average: %.2f W\r', LAST_WAVE_Power_avg);

    end

    output = LAST_WAVE_Power_avg;

end
```

A.2 Tension Controller Dynamic Functions

A.2.1 Numerical Integrating Solver

```
%-------------------------------------------------------%
% *MainFile*.m                                          %
%                                                       %
%Kite Simulation of a horizontal path, microns off the  %
%ground. The simulation solves for theta, theta_d, l,   %
%l_d, and tension. The simulation involves lift and drag%
%forces and has one animation for the top view of a     %
%flight path with a simple FBD.                         %
%-------------------------------------------------------%

clear all;
close all;
clc;
```

```
14
15 global D L l_dd Vinfin m Fp C_LiftDragFile B density Vinf_mag S AR u k;
16 global SYS_time SYS_theta SYS_theta_d SYS_l SYS_l_d LiftVect DragVect Vrel;
17 global theta_in theta_out arccomp wave_count Tension magg_Vkite VkiteVect;
18 global SYS_l_dd;
19
20 %-----------------------------ODE Solution---------------------------------
21
22 i = [1,0,0];  %Basic Vector assignment for unit
23 j = [0,1,0];  %vectors i,j,k
24 k = [0,0,1];
25
26 %Excel Document that contains the lift and drag
27 %coefficients at 5degree intervals
28 C_LiftDragFile = xlsread('NACA0015.xlsx');
29
30
31 t_scale = 0:0.01:1000;  %ODE TimeScale
32 n_waves = 50;           %Maximum Number of Passes for AWE
33
34 %System Variables
35 m = 1.5;     %kg          : Mass of the "Kite" (Point Particle)
36 B = pi()/2;  %deg         : Angle of Chord Vector w/ Tether Vector.
37 BIN = pi/2;  %deg         : Angle of Chord and Tether for Reel-In
38 BOUT = pi/2; %deg         : Angle of Chord and Tether for Reel-Out
39 density = 1.225; %kg/m^3 : Wind Power Data Standard [15C @ 1atm].
40 b = 1.2;     %m           : Wing Span (Based on SenDesign AirFoil)
41 c = 0.2;     %m           : Chord Length (Based on SenDesign AirFoil)
42 S = b*c;     %m^2         : Effective Area of Wing
43 AR = b^2/S;  %            : Aspect Ratio (dimensionless)
44 u = 1.7976*10^(-5); %kg/(m*s)    : Dynamic Viscosity
45 Fp = 0;      %N           : A perendicular force to the kite
46
47 %Defined Parameters
48 Vinfin = 10*i;           %m/s : Wind Velocity.
49 Vinf_mag = norm(Vinfin); %m/s : Magnitude of the wind velocity.
50 l_d_out = 0;             %m/s : Length Velocity of Tether during Reel-Out
51 %l_d_in = -1;            %m/s : Length Velocity of Tether during Reel-In
52 theta_in = 80*pi/180;    %rad : Theta-In Trigger Location
53 theta_out = 270*pi/180;  %rad : Theta-Out Trigger Location
54 TensionOUT = 540;        %N   : Set Force in Tether during Reel-Out
55 TensionIN = 35;          %N   : Set Force in Tether during Reel-In
56
57 %Initial Conditions
58 theta = theta_out - 2*pi; %rad      : Angle Position at t=0
59 theta_d = 1;              %rad/s    : Angle Velocity at t=0
```

```
60 l = 50;                    %m        : Length of Tether at t=0
61 l_d = l_d_out;             %m/s      : Length Velocity of Tether at t=0
62 l_dd = 0;                  %m/s^2    : Acceleration of Tether Release at t=0
63 D = 0;                     %N        : Drag Force Initial Definition
64 L = 0;                     %N        : Lift Force Initial Definition
65
66 %System Arrays for Entire Simulation Period
                                                                          %
                  components.
67 SYS_time = 0;
68 SYS_theta = theta;
69 SYS_theta_d = theta_d;
70 SYS_l = l;
71 SYS_l_d = l_d;
72 SYS_Tension = Tension;
73 [~,sizeSYS] = size(SYS_theta);
74
75 %Relative and Absolute Tolerances for ODE45 and Event Function Call
76 my_RelTol = 1e-10;
77 my_AbsTol = 1e-10;
78
79 %SS Tolerance Criteria
80 ODE_kill = 0;
81 ODE_kill_end = 2;
82 SS_tol = 1e-3;
83
84 %Initial Condition Figure Title Creation
85 figThIN = num2str(theta_in*180/pi);
86 figThOUT = num2str(theta_out*180/pi);
87 figThD = num2str(theta_d);
88 figL = num2str(l);
89 figBIN = num2str(BIN*180/pi);
90 figBOUT = num2str(BOUT*180/pi);
91
92 SSFigure_TITLE = strcat({'SteadyState__ThIN_'},figThIN,{'_ThOUT_'},figThOUT,{'
       _ThD_'},figThD,{'_L_'},figL,{'_BIN_'},figBIN,{'_BOUT_'},figBOUT);
93 SSFigure_TITLE = char(SSFigure_TITLE(1));
94 fprintf('%s \r WaveCount: ',SSFigure_TITLE)
95
96
97 %ODE45 Solution Iterations for Theta and L using Event Detection
98     % ODE45 Solver with Time and 4 Arc Components: theta, theta_dot,
99     % length, and length_dot. The function calls KITE_SS where states
100    % are used to incrementally iterate the listed Arc Components
101
102 for wave_count = 1:n_waves
```

```
103
104
105     fprintf('%d ', wave_count);
106
107     Tension = TensionOUT;
108
109     %1st Event ODE45 Solver
110         options = odeset('RelTol',my_RelTol,'AbsTol',my_AbsTol,'Events',@PI_2);
111         [t1,arccomp] = ode45(@KITE_SS,t_scale,[SYS_theta(sizeSYS),SYS_theta_d(
                sizeSYS),SYS_l(sizeSYS),SYS_l_d(sizeSYS)],options);
112
113         %Partial Wave Fragments from 1st Event Solver
114         t2 = t1(:) + SYS_time(sizeSYS);
115         time = transpose(t2);theta = transpose(arccomp(:,1));theta_d = transpose
                (arccomp(:,2));l = transpose(arccomp(:,3));l_d = transpose(arccomp
                (:,4));
116         [~,sizeWAVE] = size(time);
117
118         %Last Wave (Pass) Array for State Components
119         if ODE_kill > (ODE_kill_end-1)
120             LAST_WAVE_time = transpose(t1);
121             LAST_WAVE_theta = theta;
122             LAST_WAVE_theta_d = theta_d;
123             LAST_WAVE_l = l;
124             LAST_WAVE_l_d = l_d;
125
126             LAST_WAVE_Tension = ones(1,sizeWAVE)*Tension;
127         end
128
129         %Total System Arrays addition of Partial Wave Fragment
130         if wave_count == 1
131             SYS_time = time;
132             SYS_theta = theta;
133             SYS_theta_d = theta_d;
134             SYS_l = l;
135             SYS_l_d = l_d;
136
137             [~,sizeSYS] = size(SYS_theta);
138
139             SYS_Tension = ones(1,sizeWAVE)*Tension;
140         end
141
142         if wave_count > 1
143             SYS_time = [SYS_time time];
144             SYS_theta = [SYS_theta theta];
145             SYS_theta_d = [SYS_theta_d theta_d];
```

```
146             SYS_l = [SYS_l l];
147             SYS_l_d = [SYS_l_d l_d];
148 
149             [~,sizeSYS] = size(SYS_theta);
150 
151 
152             Wave_Tension = ones(1,sizeWAVE)*Tension;
153             SYS_Tension = [SYS_Tension Wave_Tension];
154         end
155 
156 
157     %Steady State Tolerance Error Definition for Start Wave
158         Start_Wave_l(wave_count) = l(1);
159         Start_Wave_theta_d(wave_count) = theta_d(1);
160 
161     %Tension Switch for New Controller Input
162     Tension = TensionIN;
163 
164     %2nd Event ODE45 Solver
165         options = odeset('RelTol',my_RelTol,'AbsTol',my_AbsTol,'Events',@PI3_2);
166         [t1,arccomp] = ode45(@KITE_SS,t_scale,[SYS_theta(sizeSYS),SYS_theta_d(
                sizeSYS),SYS_l(sizeSYS),SYS_l_d(sizeSYS)],options);
167 
168         %Partial Wave Fragments from 2nd Event Solver
169         t2 = t1(:) + SYS_time(sizeSYS);
170         time = transpose(t2);theta = transpose(arccomp(:,1));theta_d = transpose
                (arccomp(:,2));l = transpose(arccomp(:,3));l_d = transpose(arccomp
                (:,4));
171         [~,sizeWAVE] = size(time);
172 
173         %Last Wave (Pass) Array for State Components
174         if ODE_kill > (ODE_kill_end-1)
175             [~,sizeLAST_WAVE] = size(LAST_WAVE_time);
176             t3 = t1(:) + LAST_WAVE_time(sizeLAST_WAVE);
177             LAST_WAVE_time = [LAST_WAVE_time transpose(t3)];
178             LAST_WAVE_theta = [LAST_WAVE_theta theta];
179             LAST_WAVE_theta_d = [LAST_WAVE_theta_d theta_d];
180             LAST_WAVE_l = [LAST_WAVE_l l];
181             LAST_WAVE_l_d = [LAST_WAVE_l_d l_d];
182 
183             Tension_WAVE2 = ones(1,sizeWAVE)*Tension;
184             LAST_WAVE_Tension = [LAST_WAVE_Tension Tension_WAVE2];
185         end
186 
187         %Total System Arrays addition of Partial Wave Fragment
188         SYS_time = [SYS_time time];
```

```
189         SYS_theta = [SYS_theta theta];
190         SYS_theta_d = [SYS_theta_d theta_d];
191         SYS_l = [SYS_l l];
192         SYS_l_d = [SYS_l_d l_d];
193
194         [~,sizeSYS] = size(SYS_theta);
195
196         Wave_Tension = ones(1,sizeWAVE)*Tension;
197         SYS_Tension = [SYS_Tension Wave_Tension];
198
199      %Steady State Tolerance Error Calculation and End Wave Definition
200         End_Wave_l(wave_count) = SYS_l(sizeSYS);
201         End_Wave_theta_d(wave_count) = SYS_theta_d(sizeSYS);
202
203         SS_error_l(wave_count) = abs((End_Wave_l(wave_count) - Start_Wave_l(
                wave_count)) / Start_Wave_l(wave_count));
204         SS_error_theta_d(wave_count) = abs((End_Wave_theta_d(wave_count) -
                Start_Wave_theta_d(wave_count)) / Start_Wave_theta_d(wave_count));
205         Total_error(wave_count) = SS_error_l(wave_count) + SS_error_theta_d(
                wave_count);
206
207         %Steady State Convergence Function Kill Adder
208         if Total_error(wave_count) <= SS_tol
209             ODE_kill = ODE_kill+1;
210         elseif ODE_kill == 1
211             ODE_kill = 0;
212         end
213
214         %Steady State Convergence Function Kill
215         if ODE_kill > ODE_kill_end
216             break
217         elseif ODE_kill > 1
218             ODE_kill = ODE_kill+1;
219         end
220
221 end
222
223 Min_Tension = AngularAccel(1);
224
225 LAST_WAVE_Power_avg = Power(1);
```

A.2.2 State Space

```
1 %-------------------------------------------------------%
```

```
%KITE_SS.m                                                    %
%                                                             %
%This file takes an input from KiteSimul_0216.m to            %
%produce a numerical solution for four variables: theta,      %
%theta_d, length, and length_d through the use of a           %
%State Space Model.                                           %
%-------------------------------------------------------------%


function [myoutput,T] = KITE_SS(t,state)

    global D L l_dd Vinfin m Fp ang_attack B density Vinf_mag S Tension;

    %Initial State Declarations
    theta = state(1);
    theta_d = state(2);
    l = state(3);
    l_d = state(4);

    % The angle of attack function is called sending the state-variables to
    % find the angle of attack in the current state. Angle must also be
    % sent in radians.
    output = angleOfAttack(theta,theta_d,l,l_d);
    ang_attack = output(1);

    %Read the Excel Sheet for Coeff of Lift and Drag
    [c_DL] = CoeffLiftDrag(ang_attack);

    L = 0.5*S*density*(theta_d*l)^2*c_DL(2);
    D = 0.5*S*density*(theta_d*l)^2*c_DL(1);

    % State Variables that depend on derivatives. These are taken from the
    % KiteSimul file as initial conditions and then incrementally found by
    % plugging the initials values into the equations for derivatives and
    % original with respect to the ODE45 solver. This solver will find the
    % value evaluated at variable stepsizes depending on activity of the
    % function. The main difference is that Angular Acceleration is no
    % longer zero and is a calculable number.

    %State Changes Based on EOM
    state_dot(1) = theta_d;
```

```
44      state_dot(2) = -((L*l_d*cos(theta)^2)/(abs(l_d*sin(theta) + l*theta_d*cos(
            theta))^2 + abs(Vinf_mag - l_d*cos(theta) + l*theta_d*sin(theta))^2)
            ^(1/2) - Fp*sin(theta)^2 - Fp*cos(theta)^2 + (L*l_d*sin(theta)^2)/(abs(
            l_d*sin(theta) + l*theta_d*cos(theta))^2 + abs(Vinf_mag - l_d*cos(theta)
             + l*theta_d*sin(theta))^2)^(1/2) - (L*Vinf_mag*cos(theta))/(abs(l_d*sin
            (theta) + l*theta_d*cos(theta))^2 + abs(Vinf_mag - l_d*cos(theta) + l*
            theta_d*sin(theta))^2)^(1/2) + (D*Vinf_mag*sin(theta))/(abs(l_d*sin(
            theta) + l*theta_d*cos(theta))^2 + abs(Vinf_mag - l_d*cos(theta) + l*
            theta_d*sin(theta))^2)^(1/2) + 2*l_d*m*theta_d*cos(theta)^2 + 2*l_d*m*
            theta_d*sin(theta)^2 + (D*l*theta_d*cos(theta)^2)/(abs(l_d*sin(theta) +
            l*theta_d*cos(theta))^2 + abs(Vinf_mag - l_d*cos(theta) + l*theta_d*sin(
            theta))^2)^(1/2) + (D*l*theta_d*sin(theta)^2)/(abs(l_d*sin(theta) + l*
            theta_d*cos(theta))^2 + abs(Vinf_mag - l_d*cos(theta) + l*theta_d*sin(
            theta))^2)^(1/2))/(l*m*(cos(theta)^2 + sin(theta)^2));
45      state_dot(3) = l_d;
46      state_dot(4) = (l*m*theta_d^2*cos(theta)^2 - Tension*sin(theta)^2 - (D*l_d*
            cos(theta)^2)/(abs(l_d*sin(theta) + l*theta_d*cos(theta))^2 + abs(
            Vinf_mag - l_d*cos(theta) + l*theta_d*sin(theta))^2)^(1/2) - Tension*cos
            (theta)^2 - (D*l_d*sin(theta)^2)/(abs(l_d*sin(theta) + l*theta_d*cos(
            theta))^2 + abs(Vinf_mag - l_d*cos(theta) + l*theta_d*sin(theta))^2)
            ^(1/2) + l*m*theta_d^2*sin(theta)^2 + (D*Vinf_mag*cos(theta))/(abs(l_d*
            sin(theta) + l*theta_d*cos(theta))^2 + abs(Vinf_mag - l_d*cos(theta) + l
            *theta_d*sin(theta))^2)^(1/2) + (L*Vinf_mag*sin(theta))/(abs(l_d*sin(
            theta) + l*theta_d*cos(theta))^2 + abs(Vinf_mag - l_d*cos(theta) + l*
            theta_d*sin(theta))^2)^(1/2) + (L*l*theta_d*cos(theta)^2)/(abs(l_d*sin(
            theta) + l*theta_d*cos(theta))^2 + abs(Vinf_mag - l_d*cos(theta) + l*
            theta_d*sin(theta))^2)^(1/2) + (L*l*theta_d*sin(theta)^2)/(abs(l_d*sin(
            theta) + l*theta_d*cos(theta))^2 + abs(Vinf_mag - l_d*cos(theta) + l*
            theta_d*sin(theta))^2)^(1/2))/(m*(cos(theta)^2 + sin(theta)^2));
47
48      myoutput = [state_dot(1);state_dot(2);state_dot(3);state_dot(4)];
```

A.2.3 Angular Acceleration

```
1  %-------------------------------------------------------%
2  % AngularAccel.m                                       %
3  %                                                       %
4  % For-Loop desinged to calculate tension and angular   %
5  % corresponding to values from the ODE45 found in the  %
6  % Arc Components array. Tension isn't calculated in the%
7  % state function because it isn't based on a derivative%
8  % of itself and is only dependent upon the Arc         %
9  % Components. Angular Acceleration is not saved during %
10 % the ODE solver and is therefore recalculated for     %
```

```
% incremental time-steps in this function.             %
%-------------------------------------------------------%

function [ output ] = AngularAccel( input )

 global SYS_time SYS_theta SYS_theta_d SYS_l SYS_l_d T S density Vinf_mag;
 global l_dd m c u Re LAST_WAVE_T SYS_l_dd;

    %Matrix Size Declarations
    [~,t_size] = size(SYS_time);% : Size of Counter used in For-Loop
    T = zeros(1,t_size);        % : Declare Size of Tension Array
    SYS_l_dd = zeros(1,t_size); % : Declare Size of AngAccel Array
    Re = zeros(1,t_size);       % : Declare Size of Reynold's # Array

    [f,g] = size(SYS_time);
    ang_attack = zeros(1,f);    % : Declare Size of Angle of Attack Array
    gamma = zeros(1,f);         % : Declare Size of Gamma Array

    %Determine Last Wave Dynamics if Simulation Converged
    if ODE_kill > ODE_kill_end
        [~,sizeLAST_WAVE] = size(LAST_WAVE_time);
        LAST_WAVE_count = 1;
    end

    %Iterative For-Loop for Size of System Arrays to calculate Tension
    %and Angular Acceleration
    for ind=1:t_size

        %Initial Conditions
        % Define theta, theta-dot, length, length-dot in For-Loop as
        % the respective value from ODE45 Calculations
        theta = SYS_theta(1,ind);
        theta_d = SYS_theta_d(1,ind);
        l = SYS_l(1,ind);
        l_d = SYS_l_d(1,ind);

        %Initial Conditions
            % Define theta, theta-dot, length, length-dot in For-Loop as
            % the respective value from ODE45 Calculations
        [output] = angleOfAttack(theta,theta_d,l,l_d);

        % Angle of Attack Calculations are performed a second time to find
        % values that correspond with theta, theta_d, length, and length_d
        % since the angle isn't stored in an array when evaluated in the
        % variable timestep ODE45 function
        ang_attack(ind) = output(1);
```

```
57          Vrel(1,ind) = output(2);
58          Vrel(2,ind) = output(3);
59
60          DragVect(1,ind) = output(4);
61          DragVect(2,ind) = output(5);
62          LiftVect(1,ind) = output(6);
63          LiftVect(2,ind) = output(7);
64
65          %Coefficient of Lift and Drag Calculation
66          [c_DL] = CoeffLiftDrag(ang_attack(ind));
67
68          L(1,ind) = 0.5*S*density*(theta_d*l)^2*c_DL(2);    %N
69          D(1,ind) = 0.5*S*density*(theta_d*l)^2*c_DL(1);    %N
70
71          %Kite Velocity Calculation - Necessary for the Animation
72          magg_Vkite(ind) = norm([output(8),output(9),output(10)]);    %Component
                Vectors of Vkite
73          VkiteVect(ind,:) = [output(8),output(9),output(10)]/magg_Vkite(ind);
74
75          %Angular Acceleration System Array
76          SYS_l_dd(1,ind) = (l*m*theta_d.^2*cos(theta).^2 - SYS_Tension(ind)*sin(
                theta).^2 - (D(1,ind)*l_d*cos(theta).^2)./(abs(l_d*sin(theta) + l*
                theta_d*cos(theta)).^2 + abs(Vinf_mag - l_d*cos(theta) + l*theta_d*
                sin(theta)).^2).^(1./2) - SYS_Tension(ind)*cos(theta).^2 - (D(1,ind)
                *l_d*sin(theta).^2)./(abs(l_d*sin(theta) + l*theta_d*cos(theta)).^2
                + abs(Vinf_mag - l_d*cos(theta) + l*theta_d*sin(theta)).^2).^(1./2)
                + l*m*theta_d.^2*sin(theta).^2 + (D(1,ind)*Vinf_mag*cos(theta))./(
                abs(l_d*sin(theta) + l*theta_d*cos(theta)).^2 + abs(Vinf_mag - l_d*
                cos(theta) + l*theta_d*sin(theta)).^2).^(1./2) + (L(1,ind)*Vinf_mag*
                sin(theta))./(abs(l_d*sin(theta) + l*theta_d*cos(theta)).^2 + abs(
                Vinf_mag - l_d*cos(theta) + l*theta_d*sin(theta)).^2).^(1./2) + (L
                (1,ind)*l*theta_d*cos(theta).^2)./(abs(l_d*sin(theta) + l*theta_d*
                cos(theta)).^2 + abs(Vinf_mag - l_d*cos(theta) + l*theta_d*sin(theta
                )).^2).^(1./2) + (L(1,ind)*l*theta_d*sin(theta).^2)./(abs(l_d*sin(
                theta) + l*theta_d*cos(theta)).^2 + abs(Vinf_mag - l_d*cos(theta) +
                l*theta_d*sin(theta)).^2).^(1./2))./(m*(cos(theta).^2 + sin(theta)
                .^2));
77          l_dd = SYS_l_dd(1,ind);
78
79          %Tension and Reynolds Number Calculation
```

```
80          T(1,ind) = (l*m*theta_d^2*cos(theta)^2 - l_dd*m*sin(theta)^2 - (D(1,ind)
                *l_d*cos(theta)^2)/(abs(l_d*sin(theta) + l*theta_d*cos(theta))^2 +
                abs(Vinf_mag - l_d*cos(theta) + l*theta_d*sin(theta))^2)^(1/2) -
                l_dd*m*cos(theta)^2 - (D(1,ind)*l_d*sin(theta)^2)/(abs(l_d*sin(theta
                ) + l*theta_d*cos(theta))^2 + abs(Vinf_mag - l_d*cos(theta) + l*
                theta_d*sin(theta))^2)^(1/2) + l*m*theta_d^2*sin(theta)^2 + (D(1,ind
                )*Vinf_mag*cos(theta))/(abs(l_d*sin(theta) + l*theta_d*cos(theta))^2
                 + abs(Vinf_mag - l_d*cos(theta) + l*theta_d*sin(theta))^2)^(1/2) +
                (L(1,ind)*Vinf_mag*sin(theta))/(abs(l_d*sin(theta) + l*theta_d*cos(
                theta))^2 + abs(Vinf_mag - l_d*cos(theta) + l*theta_d*sin(theta))^2)
                ^(1/2) + (L(1,ind)*l*theta_d*cos(theta)^2)/(abs(l_d*sin(theta) + l*
                theta_d*cos(theta))^2 + abs(Vinf_mag - l_d*cos(theta) + l*theta_d*
                sin(theta))^2)^(1/2) + (L(1,ind)*l*theta_d*sin(theta)^2)/(abs(l_d*
                sin(theta) + l*theta_d*cos(theta))^2 + abs(Vinf_mag - l_d*cos(theta)
                 + l*theta_d*sin(theta))^2)^(1/2))/(cos(theta)^2 + sin(theta)^2);
81          Re(1,ind) = density*(theta_d*l)*c/u;
82
83          %If Convergence was Reached
84          if ODE_kill > ODE_kill_end
85              if ind >= (t_size - sizeLAST_WAVE + 1)
86                  LAST_WAVE_T(LAST_WAVE_count) = T(1,ind);
87                  LAST_WAVE_count = LAST_WAVE_count + 1;
88              end
89          end
90
91      end
92
93      %Minimum and Maximum Tensions
94      MAX_Tension = max(T);
95      MIN_Tension = min(T);
96
97      fprintf('\nTension - Max: %.2f N  Min: %.2f N\r', MAX_Tension, MIN_Tension);
98
99      output = MIN_Tension;
100
101 end
```

A.3 Control Functions

A.3.1 Calculate State Vector Derivative

```
1  %-------------------------------------------------------%
2  %KITE_ThetaSol.m                                       %
```

```
%                                                          %
%This funciton provides function solutions for angular     %
%acceleration and tension. This version has a constant     %
%force applied.                                            %
%----------------------------------------------------------%

i = [1,0,0];
j = [0,1,0];
k = [0,0,1];

%Symbolic Variables
syms T Vinfin D L m Fp B ang_attck
syms l l_d l_dd
syms theta theta_d theta_dd

%Er and Etheta components in terms of i,j,k
e_r = [cos(theta),sin(theta),0];
e_theta = [-sin(theta),cos(theta),0];

%Velocity and Acceleration Components
Vi = Vinfin*i;
Vkite = theta_d*l*e_theta+l_d*e_r;
Vrel = Vi-Vkite;

Akite = (theta_dd*l+2*theta_d*l_d)*e_theta+(l_dd-theta_d^2*l)*e_r;

%Solving for Angle of Attack
k_vrelteth = cross(Vrel,-e_r);
Vrel_unit = norm(Vrel);
Teth_unit = norm(-e_r);
gamma = asin(norm(k_vrelteth)/(Vrel_unit*Teth_unit));

%Unit Vectors of Lift and Drag
lambda_D = Vrel/Vrel_unit;
lambda_L = cross(k,lambda_D);

%Solver for Tension and Theta_dd
eq1 = Fp*e_theta-T*e_r+D*lambda_D+L*lambda_L-m*Akite;
[T,theta_dd] = solve(eq1(1),eq1(2),'T','theta_dd');

%Solver for Gamma
eq2 = B - gamma - ang_attck;
[ang_attck] = solve(eq2(1),'ang_attck');
```

A.3.2 Single Variable Root Finding

```
%-------------------------------------------------------%
%Kite_Simul_0216.m                                     %
%                                                      %
%Kite Simulation of a horizontal path, microns off the  %
%ground. The simulation solves for theta, theta_d, l,   %
%l_d, and tension. The simulation involves lift and drag%
%forces and has one animation for the top view of a     %
%flight path.                                          %
%-------------------------------------------------------%

%***-------------------------------------------------------------------***%
%***------------Insert Initial Conditions and Parameters--------------***%
%***-------------------------------------------------------------------***%

x_star = l_init;

SS_tol = 1e-7; %Tolerance level set for finding a Zero Crossing
del = 0.0001; %Delta Change for Equation Slope

x_star_count = 0; %Completed Passes of Root Find
End_Wave_l = 100000; %Initital Difference between Beginning and End of Wave
Start_Wave_l = 0; %Initial guess for the Root Find
Total_error = 10; %Error Initial Value

%Iterative Process Based on the Difference between the beginning of the
%wave and the end of the wave for Angular Velocity or Tether Length are
%both less than the SteadyState Tolerance Level
while abs(End_Wave_l - Start_Wave_l) > SS_tol
    x_star_count = x_star_count+1;

    %Root Find Initial Conditions
    SYS_time = 0;
    SYS_theta = theta;
    SYS_theta_d = theta_d_init;
    SYS_l = x_star(x_star_count,1);
    SYS_l_d = l_d;

    %Initial Solver for Baseline of Root Find
    baseODE = ODEventFuncDIFF();
    Start_Wave_l = baseODE(1); Start_Wave_theta_d = baseODE(2); l_d_out_base =
        baseODE(3);
    End_Wave_l = baseODE(4); End_Wave_theta_d = baseODE(5); l_d_in_base =
        baseODE(6);
    Total_error = baseODE(7); theta_in_base = theta_in; theta_out_base =
        theta_out;

```

```
44      if abs(End_Wave_l - Start_Wave_l) > SS_tol
45
46          %Define Upper Bound
47          del_l = Start_Wave_l*del;
48          Start_Wave_l_upper = Start_Wave_l + del_l;
49
50          %Del L Integration to determine slope from Baseline and Upper Bound
51          SYS_time = 0;
52          SYS_theta = theta;
53          SYS_theta_d = Start_Wave_theta_d;
54          SYS_l = Start_Wave_l_upper;
55          SYS_l_d = l_d;
56
57          del_l_ODE = ODEventFuncDIFF();
58          Start_Wave_l_del_l = del_l_ODE(1); Start_Wave_theta_d_del_l = del_l_ODE
                (2); l_d_out_del_l = del_l_ODE(3);
59          End_Wave_l_del_l = del_l_ODE(4); End_Wave_theta_d_del_l = del_l_ODE(5);
                l_d_in_del_l = del_l_ODE(6);
60          Total_error_del_l = del_l_ODE(7); theta_in_del_l = theta_in;
                theta_out_del_l = theta_out;
61
62          %Differences between Baseline and Upper Bound start and end lengths
63          Func_l = (End_Wave_l - Start_Wave_l);
64          Func_l_del_l = (End_Wave_l_del_l - Start_Wave_l_del_l);
65
66          %Alpha Coefficient used to calculate the size of the step
67          l_der = (Func_l_del_l - Func_l)/del_l;
68
69          %Stepped Value location fed to next loop iteration
70          stepped_value = (1/l_der)*-(End_Wave_l-Start_Wave_l) + Start_Wave_l;
71          x_star(x_star_count+1,1) = stepped_value(1);
72
73      end
74
75  end
```

A.3.3 Multi-Dimensional Root Finding

```
1  %-------------------------------------------------------------%
2  % 2DRootFind_theta_d_l.m                                      %
3  %                                                             %
4  % Newtonian-Rhapsonian Multi-Dimensional Root Finding Method %
5  % used to find the error plane crossing for the variables    %
6  % angular velocity and tether length at the end of a cycle.  %
```

```
7  %-------------------------------------------------------------%
8
9  %***-------------------------------------------------------------------***%
10 %***------------Insert Initial Conditions and Parameters--------------***%
11 %***-------------------------------------------------------------------***%
12
13 %Initital Difference between Beginning and End of Wave
14 Func_theta_d = 100000;
15 Func_l = 100000;
16
17 SS_tol = 1e-5; %Tolerance level set for finding a Zero Crossing
18 del = 0.0001;  %Delta Change for Equation Slope
19 x_star = [theta_d, l]; %Initial guess for the Root Find
20 x_star_count = 0; %Completed Passes of Root Find
21
22 %Iterative Process Based on the Difference between the beginning of the
23 %wave and the end of the wave for Angular Velocity and Tether Length are
24 %both less than the SteadyState Tolerance Level
25 while abs(Func_theta_d) > SS_tol || abs(Func_l) > SS_tol
26     x_star_count = x_star_count+1;
27
28     %Root Find Initial Conditions
29     SYS_time = 0;
30     SYS_theta = theta;
31     SYS_theta_d = x_star(x_star_count,1);
32     SYS_l = x_star(x_star_count,2);
33     SYS_l_d = l_d;
34
35     %Initial Solver for Baseline of Root Find
36     baseODE = ODEventFuncDIFF();
37     Start_Wave_l = baseODE(1); Start_Wave_theta_d = baseODE(2); l_d_out_base =
           baseODE(3);
38     End_Wave_l = baseODE(4); End_Wave_theta_d = baseODE(5); l_d_in_base =
           baseODE(6);
39     Total_error = baseODE(7); theta_in_base = theta_in; theta_out_base =
           theta_out;
40
41     %Will enter RF if error between variables do not satisfy SS_tolerance
42     if abs(Func_theta_d) > SS_tol || abs(Func_l) > SS_tol
43
44         %Define Upper Bound for Angular Velocity and Tether Length
45         del_l = Start_Wave_l*del;
46         del_theta_d = Start_Wave_theta_d*del;
47
48         Start_Wave_l_upper = Start_Wave_l + del_l;
49         Start_Wave_theta_d_upper = Start_Wave_theta_d + del_theta_d;
```

```
50
51          %Del Theta_d Integration
52              %Initial Conditions for Del Theta_D
53              SYS_time = 0;
54              SYS_theta = theta;
55              SYS_theta_d = Start_Wave_theta_d_upper;
56              SYS_l = Start_Wave_l;
57              SYS_l_d = l_d;
58
59              %Theta_D ODE Solver
60              del_theta_d_ODE = ODEventFuncDIFF();
61              Start_Wave_l_del_t = del_theta_d_ODE(1); Start_Wave_theta_d_del_t =
                    del_theta_d_ODE(2); l_d_out_del_t = del_theta_d_ODE(3);
62              End_Wave_l_del_t = del_theta_d_ODE(4); End_Wave_theta_d_del_t =
                    del_theta_d_ODE(5); l_d_in_del_t = del_theta_d_ODE(6);
63              Total_error_del_t = del_theta_d_ODE(7); theta_in_del_t = theta_in;
                    theta_out_del_t = theta_out;
64
65          %Del L Integration
66              %Initial Conditions for Del L
67              SYS_time = 0;
68              SYS_theta = theta;
69              SYS_theta_d = Start_Wave_theta_d;
70              SYS_l = Start_Wave_l_upper;
71              SYS_l_d = l_d;
72
73              %L ODE Solver
74              del_l_ODE = ODEventFuncDIFF();
75              Start_Wave_l_del_l = del_l_ODE(1); Start_Wave_theta_d_del_l =
                    del_l_ODE(2); l_d_out_del_l = del_l_ODE(3);
76              End_Wave_l_del_l = del_l_ODE(4); End_Wave_theta_d_del_l = del_l_ODE
                    (5); l_d_in_del_l = del_l_ODE(6);
77              Total_error_del_l = del_l_ODE(7); theta_in_del_l = theta_in;
                    theta_out_del_l = theta_out;
78
79
80          %Jacobian, StepSize, and Updated Root
81              %Function Calculations for Differences between Begin and End
82              Func_theta_d = (End_Wave_theta_d - Start_Wave_theta_d);
83              Func_l = (End_Wave_l - Start_Wave_l);
84              Func_theta_d_del_t = (End_Wave_theta_d_del_t -
                    Start_Wave_theta_d_del_t);
85              Func_theta_d_del_l = (End_Wave_theta_d_del_l -
                    Start_Wave_theta_d_del_l);
86              Func_l_del_t = (End_Wave_l_del_t - Start_Wave_l_del_t);
87              Func_l_del_l = (End_Wave_l_del_l - Start_Wave_l_del_l);
```

```
88
89              %Newton-Rhapson Method Jacobian
90              Jacobian = [(Func_theta_d_del_t - Func_theta_d)/del_theta_d, (
                    Func_theta_d_del_l - Func_theta_d)/del_l;
91                      (Func_l_del_t - Func_l)/del_theta_d, (Func_l_del_l -
                            Func_l)/del_l];
92
93              %New Root Step for the Next Iteration
94              stepped_value = inv(Jacobian)*[-(End_Wave_theta_d-Start_Wave_theta_d
                    );-(End_Wave_l-Start_Wave_l)] + [Start_Wave_theta_d;Start_Wave_l
                    ];
95              x_star(x_star_count+1,1) = stepped_value(1); x_star(x_star_count
                    +1,2) = stepped_value(2);
96
97      end
98
99  end
```

A.3.4 Animation of System Position and Orientation

```
1   %-------------------------------------------------------%
2   % aniMATION.m                                           %
3   %                                                       %
4   % Provides a visual representation of the flightpath and  %
5   % and orientation of the airfoil while in flight. This    %
6   % version depicts if the tether is in positive or         %
7   % negative tension by the color of the thin rod. It will  %
8   % also provide a FBD with force vectors in ratio to their %
9   % maximum values.                                        %
10  %-------------------------------------------------------%
11
12  function b = aniMATION(b)
13
14    global SYS_time SYS_theta SYS_theta_d SYS_l SYS_l_d L D LiftVect DragVect;
15    global Vrel T magg_Vkite VkiteVect;
16
17      [~,sizeSYS] = size(SYS_time);
18
19      %Initital Defining of Lines
20
21          %Open a Figure Window to place initial construct of animation
22          verticalFig = figure('Name','2D Tether Animation');
23          %Line for initial tether length and initial theta
```

```
24          tether_line = line('xdata',[0 SYS_l(1,1)*cos(SYS_theta(1,1))],'ydata',[0
                SYS_l(1,1)*sin(SYS_theta(1,1))]);
25          max_l = max(SYS_l);
26          %Tether Figure Margin
27          xlim([-max_l-max_l*.2,max_l+max_l*.2]);
28          ylim([-max_l-max_l*.2,max_l+max_l*.2]);
29          axis equal
30          axis manual
31
32      %Unitization of Lift, Drag, and Velocity Lines
33          M1 = max(D);
34          M2 = max(L);
35
36          if M1 < M2
37              maxLD = M2;
38          else
39              maxLD = M1;
40          end
41
42          maxVEL = max(magg_Vkite);
43
44      %Kite Figure
45          kiteFig = figure('Name','Kite FBD');
46          magg_vrel = norm(Vrel(1,:));
47          lift_line = line('xdata',[0 L(1,1)*LiftVect(1,1)/maxLD],'ydata',[0 L
                (1,1)*LiftVect(1,1)/maxLD]);
48          drag_line = line('xdata',[0,D(1,1)*DragVect(1,1)/maxLD],'ydata',[0,D
                (1,1)*DragVect(1,1)/maxLD]);
49          vrel_line = line('xdata',[0,Vrel(1,1)/magg_vrel],'ydata',[0,Vrel(1,2)/
                magg_vrel]);
50          vkite_line = line('xdata',[0,VkiteVect(1,1)/maxVEL],'ydata',[0,VkiteVect
                (1,2)/maxVEL]);
51
52      %XY-axis limits for Airfoil Figure Window
53          xlim([-1.1,1.1]);
54          ylim([-1.1,1.1]);
55          axis equal
56          axis manual
57
58      %Airfoil Outline Points
59          x_foil = [0.00158000000000000;0.0100800000000000;0.0181000000000000;...
60              0.0327900000000000;0.0458000000000000;0.0570400000000000;...
61              0.0661700000000000;0.0725400000000000;0.0750200000000000;...
62              0.0742700000000000;0.0717200000000000;0.0668200000000000;...
63              0.0585300000000000;0.0525000000000000;0.0444300000000000;...
64              0.0326800000000000;0.0236700000000000;0.00100000000000000;...
```

```
65              -0.0236700000000000;-0.0326800000000000;-0.0444300000000000;...
66              -0.0525000000000000;-0.0585300000000000;-0.0668200000000000;...
67              -0.0717200000000000;-0.0742700000000000;-0.0750200000000000;...
68              -0.0725400000000000;-0.0661700000000000;-0.0570400000000000;...
69              -0.0458000000000000;-0.0327900000000000;-0.0181000000000000;...
70              -0.0100800000000000;-0.0015800000000000];
71          y_foil = [-0.750000000000000;-0.700000000000000;-0.650000000000000;...
72              -0.550000000000000;-0.450000000000000;-0.350000000000000;...
73              -0.251000000000000;-0.150000000000000;-0.0500000000000000;...
74              0;0.0500000000000000;0.100000000000000;0.150000000000000;...
75              0.175000000000000;0.200000000000000;0.225000000000000;...
76              0.237500000000000;0.250000000000000;0.237500000000000;...
77              0.225000000000000;0.200000000000000;0.175000000000000;...
78              0.150000000000000;0.100000000000000;0.0500000000000000;0;...
79              -0.0500000000000000;-0.150000000000000;-0.251000000000000;...
80              -0.350000000000000;-0.450000000000000;-0.550000000000000;...
81              -0.650000000000000;-0.700000000000000;-0.750000000000000];
82
83          [q,r] = size(x_foil);
84
85      %Airfoil Shape Array
86          for ind = 1:q-1
87              airfoil_parts(ind) = line('xdata',[0,0],'ydata',[0,0]);
88          end
89
90      %Draw Dynamics and FBD for the entire simulation
91          for ind=1:sizeSYS
92
93              %Initial Parameters
94              force_scale = 50;
95              theta = SYS_theta(1,ind);
96              theta_d = SYS_theta_d(1,ind);
97              l = SYS_l(1,ind);
98              l_d = SYS_l_d(1,ind);
99              Lift = L(1,ind)/maxLD;
100             Drag = D(1,ind)/maxLD;
101             Vel = magg_Vkite(ind)/maxVEL;
102
103             %Rotate and Draw Airfoil for each iteration
104             for indd = 1:q
105                 [foilCoord1(:,indd)] = [cos(theta),-sin(theta);sin(theta),cos(
                        theta)]*[x_foil(indd,1);y_foil(indd,1)];
106             end
107
108             foilCoord = foilCoord1';
109
```

```
110             for indd = 1:q-1
111                 set(airfoil_parts(indd),'xdata',[foilCoord(indd,1),foilCoord(
                        indd+1,1)],'ydata',[foilCoord(indd,2),foilCoord(indd+1,2)]);
112             end
113
114             % 2-D Line Drawing for the Simulation
115                 %Tether Figure (verticalFig)
116                 if T(1,ind)>0
117                     set(tether_line,'xdata',[0,l*cos(theta)],'ydata', [0,l*sin(
                            theta)]);
118                 else
119                     set(tether_line,'xdata',[0,l*cos(theta)],'ydata', [0,l*sin(
                            theta)],'Color','red');
120                 end
121
122                 %Airfoil Lift and Drag Components Animation (kiteFig)
123                 mag_vrel = norm(Vrel(:,ind));
124                 set(vrel_line,'xdata',[0,Vrel(1,ind)/mag_vrel],'ydata',[0,Vrel
                        (2,ind)/mag_vrel],'Color','yellow');
125                 set(lift_line,'xdata',[0,Lift*LiftVect(1,ind)],'ydata',[0,Lift*
                        LiftVect(2,ind)],'Color','green');
126                 set(drag_line,'xdata',[0,Drag*DragVect(1,ind)],'ydata',[0,Drag*
                        DragVect(2,ind)],'Color','red');
127                 set(vkite_line,'xdata',[0,Vel*VkiteVect(ind,1)],'ydata',[0,Vel*
                        VkiteVect(ind,2)],'Color','blue');
128
129             drawnow
130             grid on
131
132 end
```

Appendix B

Tension Controller Parameter Variation

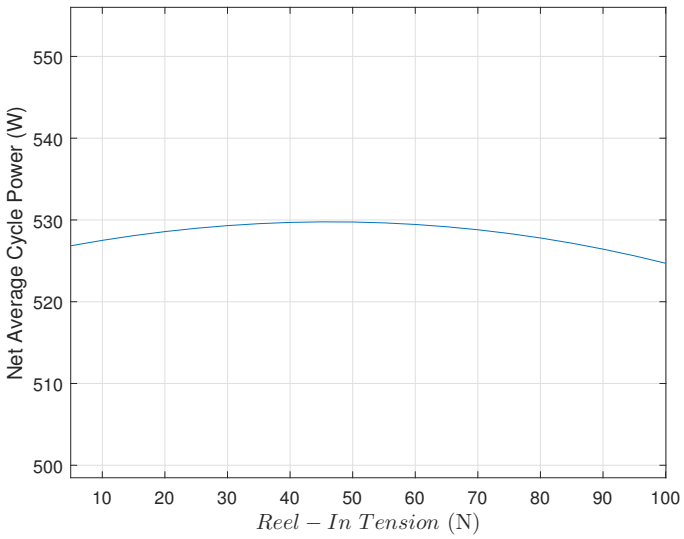


Fig. B.1: Parameter variation of the tether tension controller in Fig. 3.36 to show net average cycle power while varying reel-in tension

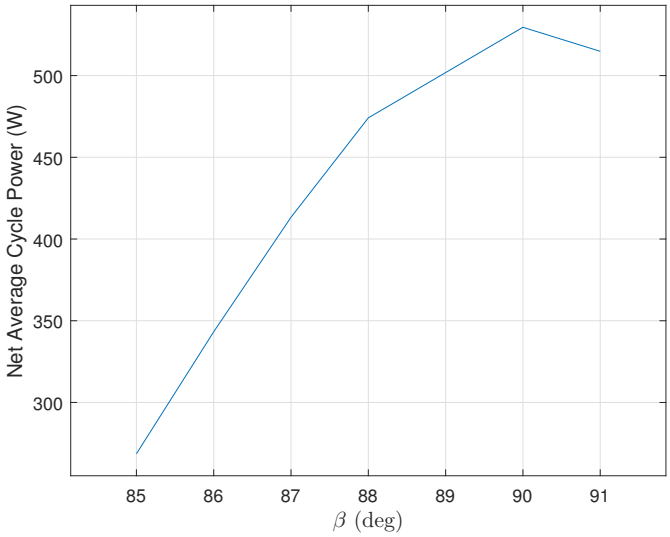


Fig. B.2: Parameter variation of the tether tension controller in Fig. 3.36 to show net average cycle power while varying the angle between the tether and the chord line of the kite

Appendix C

3D Simulation Verification

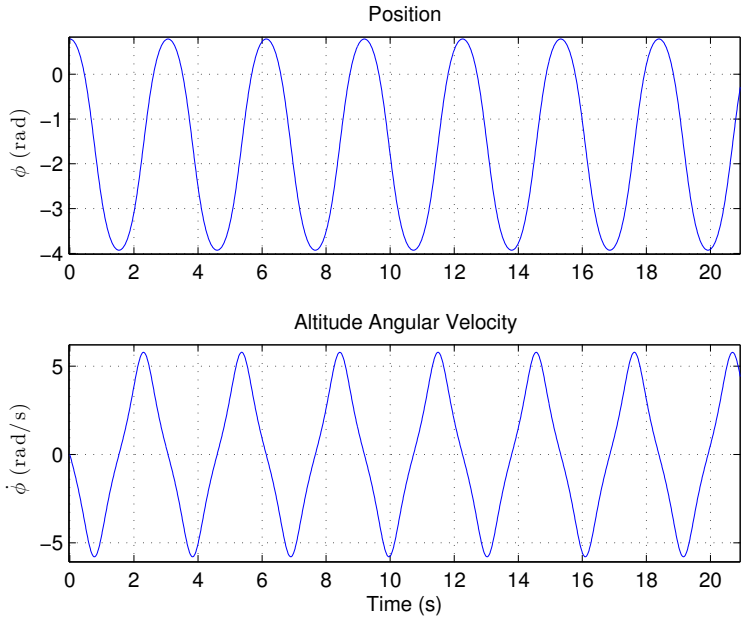


Fig. C.1: Test case showing pendulum behavior of the model as altitude angle oscillates with wind velocity = 0 m/s, lift and drag forces = 0 N, gravity = 9.81 m/s^2 to verify periodic cycle behavior from Fig. 4.7

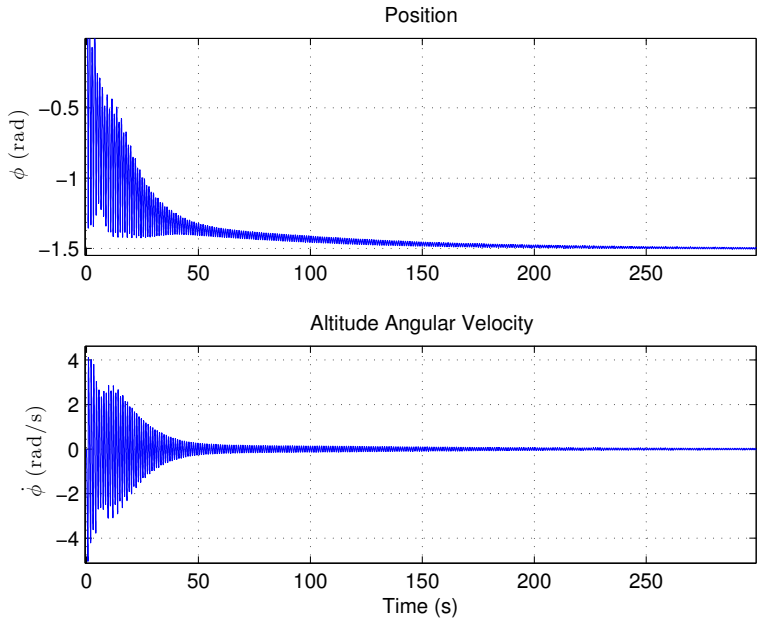


Fig. C.2: Test case showing the eventual dampening of the system with altitude angular velocity reducing to 0 m/s with wind velocity = 0 m/s and lift and drag forces allowed. Associated with Fig. 4.8

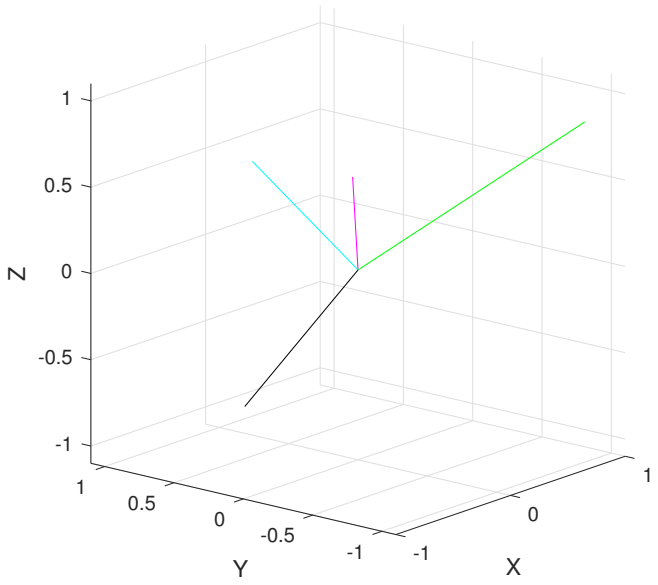


Fig. C.3: Simple animation to verify vector directions of tether (black), lift (green), wing span line (light blue), and apparent wind velocity (magenta)

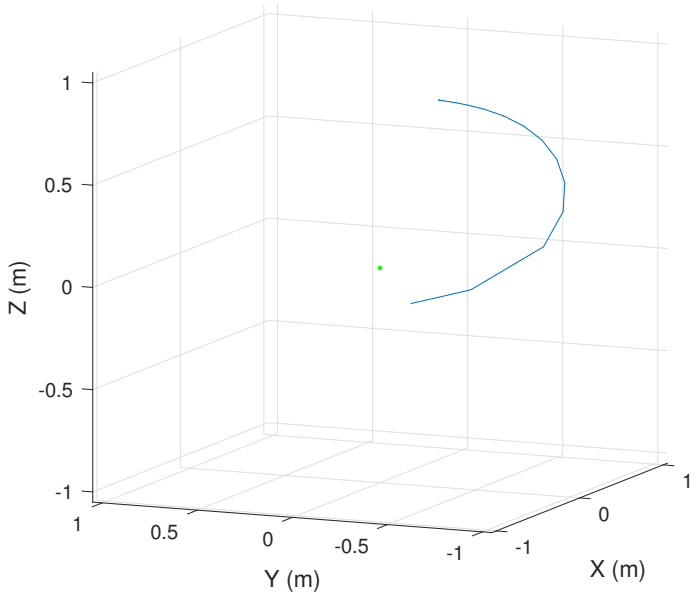


Fig. C.4: Simple animation to verify tether orientation during animation for vector directions in Fig. C.3

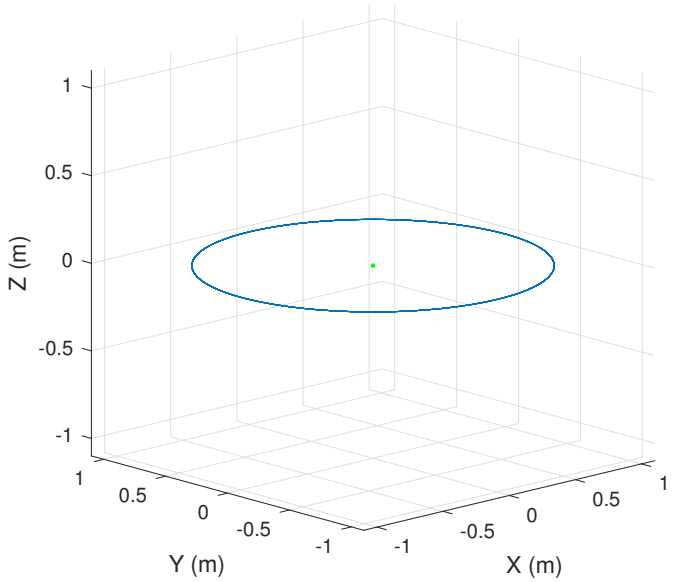


Fig. C.5: Test case to verify ball on a string for horizontal axis (wind speed = 0 m/s, lift and drag forces set to 0N, gravity = 0 m/s^2, and altitude angular velocity/acceleration set to rad/s and 0 rad/s^2)

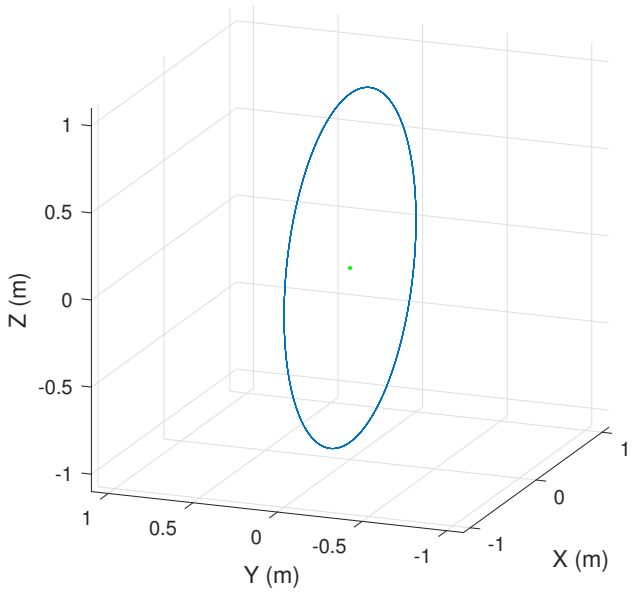


Fig. C.6: Test case to verify ball on a string for vertical axis (wind speed = 0 m/s, lift and drag forces set to 0N, gravity = 0 m/s^2, and tether angular velocity/acceleration set to rad/s and 0 rad/s^2)

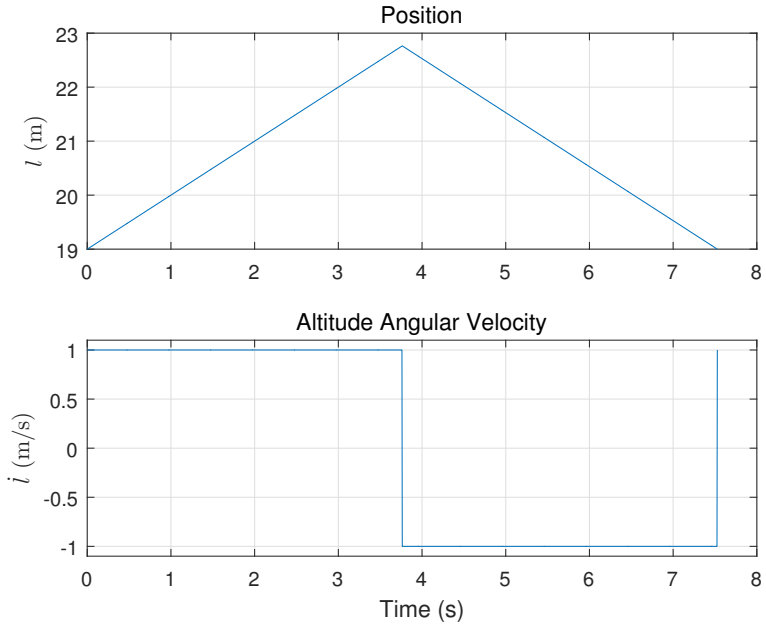


Fig. C.7: Reel-rate for horizontal ball and string to verify reel-in and reel-out adjustments and tether angular position triggers