Video Input Driven Animation (VIDA)

Meng Sun

A thesis submitted in conformity with the requirements
for the Degree of Doctor of Philosophy,
Graduate Department of Computer Science
University of Toronto

Abstract

Video Input Driven Animation (VIDA)

Meng Sun

Doctor of Philosophy

Graduate Department of Department of Computer Science

University of Toronto

2003

There are many challenges associated with the integration of synthetic and real imagery. One particularly difficult problem is the automatic extraction of salient parameters of natural phenomena in real video footage for subsequent application to synthetic objects. Can we ensure that the hair and clothing of a synthetic actor placed in a meadow of swaying grass will move consistently with the wind that moved that grass? The video footage can be seen as a controller for the motion of synthetic features, a concept we call *video input driven animation* (VIDA). We propose a schema that analyzes an input video sequence, extracts parameters from the motion of objects in the video, and uses this information to drive the motion of synthetic objects. To validate the principles of VIDA, we approximate the inverse problem to harmonic oscillation, which we use to extract parameters of wind and of regular water waves. We observe the effect of wind on a tree in a video, estimate wind speed parameters from its motion, and then use this to make synthetic objects move. We also extract water elevation parameters from the observed motion of boats and apply the resulting water waves to synthetic boats.

Acknowledgements

My parents' unconditional support is one of the things which helped me to focus and finish this thesis in full speed within the last two years. At the same time, they taught me the meaning of a balanced life.

My supervisor Eugene's passion and enthusiasm for computer graphics had profound influence on me ever since the first time we met. I would like to thank him for his encouragement and guidance, as well as being an excellent role model for me.

I have the best committee members I could have ever asked for.

Prof. Allan Jepson and his student Thomas F. El-Maraghi kindly did the video feature tracking for the video stream in my thesis, which is a crucial part of the experiment. Their most recent paper on tracking won the runner-up for best paper award at CVPR 2001. Prof. Allan Jepson's amazing mathematical intuition helped me tremendously in formulating the building blocks of this thesis. I have always come to his office with crazy ideas. No matter how far fetching the idea might be, he could always find a positive and sensible way to bringing out the potential in it. I am very lucky to be working with him.

James, who is now a professor at Queen's University, was the first person who pointed me in the direction of image based rendering (IBR). I owe him many thanks for introducing me to the vast research area of computer vision. His enthusiasm encouraged and pushed me to experiment with combining vision and graphics research.

Prof. Ken Sevcik have always been extremely supportive of my research. I thank him for many interesting suggestions for potentially applications of my thesis and for all the hours he spent tirelessly reading and correcting my thesis.

Dr. Rick Sziliski is my external examiner. His ground breaking research work in computer vision and image based rendering was an inspiration to my thesis. I am very thankful that, despite of his busy schedule, he kindly spent so much time on giving very insightful comments to my thesis and flew down to Toronto to attend my PhD defense.

Though Michiel van de Panne is not on my committee, I have always had the blessing of having him as a mentor. He meant so much to so many of us – his wisdom, intuition, creativity, his no nonsense approach to things, and most of all his kindness and sincerity as a person.

Sara Burns had lent me a helping hand time after time. She has always been so patient with us, and her sense of humor is cherished by all of us. Thanks for Linda Chow's diligence, she helped me a lot in making sure that I get things prepared by deadlines. Joan Allen is so much fun to work with.

The wonderful graduate students, professors and system administrators at the dgp lab are the ones who made my graduate school years so memorable and full of joy. They helped me a great deal in my studies. More than that, their friendship have always made me feel like dgp was my second home. I would like to thank Joe Laszlo, Victor Ng, Petros Faloutsos, David Mould, Glenn Tsang, Jeff Tupper, Chris Trendall, Michael Neff, Jos Stam, Michael McGuffin, Dave Torre, Anastasia Bezerianos, Maciej Kalisiak, Alan Rosenthal, Ania Lipka, Xiaohuan Wang, Joanna McGrenere, Paulo Pacheco, Qinxing Yu, Sageev Oore, Ravin Balakrishnan, Karan Singh, Alejo Hausner, Daniel Taranovsky, Cathy Jansen, Reimar Schubert, Jimmy Talbot, Theophanis Tsandilas, William Hunt, Wael Aboelsaadat, Michael Tsang, Gonzalo Ramos, Yan Wang, Jingrui Zhang, Michael Wu, Nigel Morris, Cathy Jansen, Dr. Baining Guo, Prof. Hiroshi Ishii, Naomi Friedlan-

der, Nick Torkos, Opal Downer, Tom Bellman, Yuyan Liu.

I also owe many thanks to my fellow student friends at the department – Thomas F. El-Maraghi, Hai Wang, Bowen Hui, Yongmei Liu, Zongpeng Li, Wei Tjioe Chakra Chennubhotla, George Giakkoupis, Michalis Faloutsos, Wayne Hayes, Diana (Zaiu) Inkpen, Daniela Rosu, Cristiana Chitic, Vladimir Kolesnikov, Ding Wei, Jingjing Lu, Yuxing Zhu, Xi Wang, Panayiotis Tsaparas, Ray Ortigas, Nick Koudas, Tiffany Lai, Tasso, Tao Hu.

I would also miss the nice faculty members at the department – Jim Clarke, Diane Horton, Tom Fairgrieve, Faith Fich, Paul Gries.

Contents

List of Figures

Chapter 1

Introduction

The success of computer animation as an expressive medium has put increasing demands on integrating computer animation with real video or film footage. Techniques from augmented reality and inverse rendering now permit the rendering of synthetic models within still images [FGR93, DRB97, LFT97, SWI97, YDH99, BG01]. However, inserting objects that interact with or are affected by forces and real objects present in the video is a difficult open problem. How do we ensure that the hair on a synthetic actor placed in a meadow of swaying grass will move consistently with the real wind field that moved that grass?

In this case, raw video footage can be seen as a kind of controller for the motion of the synthetic hair. We call this concept *video input driven animation*, or VIDA. The schema we propose for VIDA is to analyze an input video sequence, to extract relevant parameters of the motion of objects in the video, and to use these parameters to drive the motion of synthetic objects which are introduced into the real environment (Figure 1.1). One aim of VIDA is to allow synthetic objects to be natural participants in a video.

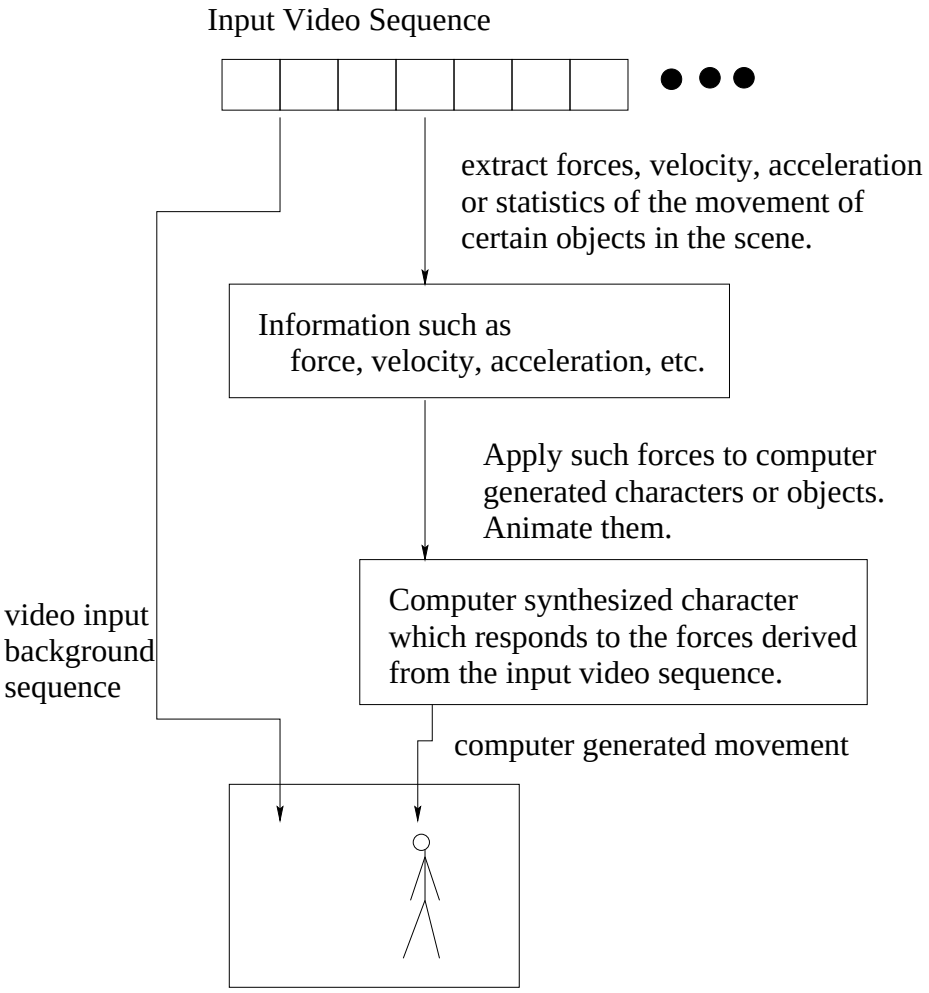

Figure 1.1: Using video input to drive computer animation.

This opens up a huge and fascinating range of problems. Our goal in this thesis is to explore some of these problems, to validate the approach through the development of a nontrivial VIDA application, and to point the way to further results.

1.1 Scope

Computer graphics has enjoyed rapid growth over the past fifty years. With dramatic technological advances in all areas of computer graphics such as geometric modelling, rendering, animation, we are able to model more and more complicated scenes, environ-

ment, motion and situation. However, the dazzling beauty and amazing complexity of the natural real world still has much more to offer in computer graphics applications in addition to or in comparison to what we can artificially generate by computer simulation. Increasingly, researchers are exploring different possibilities of sampling the real world to extract useful information to enrich computer graphics applications.

1.1.1 Computer Augmented Reality

In computer augmented reality, researchers are interested in integrating synthetic and real imagery. There are several interesting research directions that arise from this problem:

- Recover common viewing parameters. Whenever one wants to mix three dimensional computer graphics with footage of the real world, one needs to match the computer graphics camera with the actual camera that took the footage. This process ensures that the computer graphics elements match the perspective and movement of the real objects in the real camera shot. This is called *camera tracking* or *match moving*. There are several commercially available software packages that automate this task, such as boujou from 2d3 (www.2d3.com) and SceneGenie from 3D Studio MAX.

- Approximate common global illumination for real video and computer generated objects. A number of researchers [FGR93, DRB97] proposed different approaches using different assumption and constraints about surface reflectance properties of objects in the real video.

- Recover surface reflectance function from one photograph or multiple photographs.

This enables us to change the lighting condition and correctly render the surface of objects under different illumination. Many researchers [YDH99, BG01] have explored this topic in depth.

In this thesis, we propose to extract parameters of forces and the parameters of natural phenomena which caused object motion, then use the inferred information to drive the animation of computer synthesized object, making them appear to be natural participants of the real video. The different tasks in augmented reality naturally link computer vision and computer graphics research together.

1.1.2 Sampling, Analysis, Synthesis

The methodology behind VIDA falls into the category of vision directed sampling-analysis-synthesis. By vision directed sampling-analysis-synthesis, we are referring to a three step process. First, sample the real world using equipments such as camera, video camera, etc. Second, analyze the images we obtained for different purposes such as extracting properties and parameters of objects in the scene. Then, use the inferred information to synthesize new images or scenes or scenarios. In fact, as different variations of sampling-analysis-synthesis approach emerge in a number of research areas in computer graphics, there is a growing interest in the strong bond between computer vision and computer graphics research.

For instance, researchers realized that texture mapping and computer-synthesized texture based on mathematics formulas have their limitations. Recently, several researchers have experimented with different techniques for synthesizing textures that match the ap-

pearance of a given texture sample [HB95, Bon97, EL99, WL00, Tur01, WL01]. Most of the approaches use statistical models while others rely on deterministic structural models. Soatto et al. [SDW01] studied dynamic textures, which are sequences of images of moving scenes that exhibit certain stationarity properties in time. These include sea-waves, smoke, foliage, whirlwind, talking faces, traffic scenes, etc. They presented a novel characterization of dynamic textures by learning models of the dynamic textures. Once learned, a model can be used to extrapolate synthetic sequences of arbitrary length.

1.1.3 Causal Information Inference

In computer graphics, we are always fascinated with making objects move realistically. Both artificial intelligence and psychology researchers have argued for the need to represent "causal" information about the world in order to make inferences. In particular, understanding motion sequences requires the observer to postulate forces on objects and force transfer between interacting objects or within an object. VIDA naturally lands itself in this research area. To push the recovery of "causal" information one step further, VIDA poses questions not just about objects in the scene but also about the natural phenomena we can observe in the scene.

There are a number of interesting research results in causal inference of information based on motion understanding. Bregler et al. [BCS97] propose the idea of video rewrite, where they use existing video footage to create automatically new video of a person mouthing words that she did not speak in the original footage. In the paper "Voice Puppetry" [Bra99], Brand introduced a method for predicting a control signal

from another related signal, and applied it to voice puppetry: Generating full facial animation from expressive information in an audio track. Popović and Witkin [PA99] introduced an algorithm for transforming character animation sequences that preserve essential physical properties of the motion. Hoshino and Saito [HYS01] proposed a new technique for merging computer generated clothes onto the human in a video sequence. Black and Yacoob [BY95b, BY95a] track and recognize rigid and non-rigid facial motions using local parametric models of image motion. They showed how expressions such as anger, happiness, surprise, fear, disgust and sadness can be recognized from the local parametric motions in the presence of significant head motion. Mann et al. [MJ97] reasoned about qualitative scene dynamics based on an analysis of the Newtonian mechanics of a simplified scene model.

1.1.4 How VIDA Relates to the Overall Scope

VIDA has a close connection to the landscape we described in section 1.1, and it aims at extending and expanding these existing research topics into a new territory. As an application, VIDA can provide solutions to tasks in augmented reality. As a methodology, VIDA is based on the vision directed sampling-analysis-synthesis approach. In terms of recovering parameters or characteristics of natural phenomena and object interaction, VIDA opens up more challenges in this area.

1.2 What is VIDA?

Computer-based image synthesis transforms an input model to an output, such as computer animation or rendered images. Both the transformation and the model are described and controlled by some parameters. VIDA couples the inverse problem to such a synthesis process. The inverse problem is to observe the motion of objects and environments in a scene, fit a model to the components in the scene, such as objects or wind, and extract parameters of the input to a system, such as forces and wind velocity. One of the immediate benefits of such a parameter extraction process is that we can use the parameters that are estimated from the input video sequence to indirectly drive computer animation of synthetic objects.

The complexity of this task is open ended. In any given scene, there could be different types of objects involved, such as fluids, deformable objects, rigid objects, articulated objects, particles, etc. There are also different types of forces among objects or within each object. It is difficult to tackle these problems in a unified and comprehensive way. Rather, it would appear that different approaches are needed for difference classes of phenomena. To make the process tractable, we propose to use simple models for solving this inverse problem. Physics provides a consistent way to account for the motion we see in real world video sequences, whether the motion is caused by natural phenomena such as wind and water, the ballistic motion of objects in flight, or the biomechanical motion of animate beings, etc. In a forward computer animation or computer synthesis problem, people might apply control inputs such as forces or a wind field to computer generated objects. A physical model allows these objects to react to internal and external forces in

a natural way. The output is the resulting computer simulation. For an inverse problem, we are only given the observable motion of the object. Since real life object movement obeys the laws of physics, fitting models to the objects in the scene allows us to link the observed movement to the input which caused the components to move, as shown in Figure 1.2.

Figure 1.2: VIDA process pipeline.

Just like the forward animation process, which consists of specific approaches for fluid dynamics, facial expression, articulated figure motion, etc, we envision that the inverse problem (VIDA) will also have solutions which are like tools in a toolbox. There might be different classes of dynamic analysis for different types of objects and phenomena. As part of a long term research effort, we intend to study many more of the different useful instances of the VIDA problem in the future. They would be powerful tools for computer animators. Our work here will demonstrate the plausibility of VIDA.

In a broader sense, video input driven animation does not necessarily have to be physics based. For instance, the animator might want to use VIDA to simply track the movements of flowers on a hillside, and add some autonomous virtual butterfly or bees to interact with the video captured swinging flowers. In this case, we may not need to use physics. This is similar to conventional or digital rotoscoping. Or in some other instances, say, in a games setting, a real physical simulation might be too time consumming, then pseudo-physics or motion based method can be used to allow the video input to indirectly drive the computer animation of virtual objects.

In this thesis, to demonstrate the importance and exploit the usefulness of the VIDA methodology, we will explore some natural scenes and estimate parameters of natural phenomena, such as wind and regular water wave elevation. We use a physics based approach for our analysis, and we explain the reasoning behind this choice later in the chapter. The examples are intended to illustrate the strength of VIDA, in particular its application in indirect animation. We will show how to recover wind speed parameters by observing objects in a scene and how to estimate regular water wave elevation parameters by observing boat motion in a scene. These example VIDA techniques are interesting, important, and novel in their own right.

1.3 Motivation

As a simple example of a possible VIDA application, we will contrast it with the back projection technique traditionally used in cinematography. Back projection is commonly used for creating a background environment as shown in Figure 1.3. The background is the projection of video footage of a surrounding environment at some distance far away or medium range, such as open outdoor scenery or a street scene shot from a moving vehicle. To make the foreground objects and characters look like they are placed in the scene and are consistent with the background, people sometimes create artificial wind using a fan or rock a car to given audiences the illusion that the car is being driven on a bumpy road.

When we try to use such real life video background in computer animation and place computer generated objects in it, typically, the artists/animators are responsible for mak-

ing sure that the video background and the synthetic foreground objects are compatible. For example, if the background is a windy scene, the synthesized objects/characters should be animated compatibly as if the wind is also blowing on them. In general, animators need to take care of similar scene continuity and consistency problem as in regular cinematography. This type of effort for maintaining scene consistency largely relies on the animator's intuition and interpretation of the real life video input. The task is time consuming and tedious for an animator.

The question we ask ourselves is: "Which scene consistency relations can we automatically enforce?" In other words, can we analyze the input video sequence, extract parameters and information from the motion of objects in the scene and use these parameters to drive a certain class of motion for the computer generated objects/characters? As we have mentioned before, we categorize this class of problem as **Video Input Driven Animation (VIDA)**.

For example, in a windy scene, we might want to "reverse engineer" a natural phenomenon such as wind, in the video sequence, and use the extracted wind field parameters to drive new computer animated objects. When the animator does not have to take care

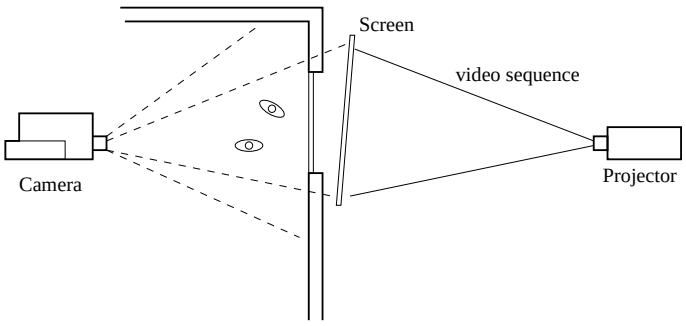

Figure 1.3: Back projection set-up in cinematography. Video footage is used as the background.

of such tedious details, they can concentrate more on the artistic and creative components of the scene and character animation. Moreover, this does not have to be restricted to just the background-foreground relation of the scene. In computer animation, there are many creative ways that the computer generated objects can interact with and be driven by the environment in the video input.

There are many applications for VIDA, such as movie making, animation prototyping, virtual reality simulations, and landscape design, to name a few. It is challenging because it requires that we think about things in a new way. We need to capitalize on the existing computer vision tools, and push ourselves to invent new computer vision tools to satisfy the requirements of computer graphics applications. For computer graphics, VIDA introduces a rich new source for animation creation and animation control. It allows us to use video to give indirect inputs, such as forces and constraints, to computer animation. VIDA combines the strength of computer vision and computer graphics, and at the same time, pushes the envelopes in both fields. VIDA is inherently phenomenological, which lies at the core of both computer graphics and computer vision.

1.4 Objects and Forces in VIDA

When a VIDA application takes a physics based approach, we might be concerned with several types of forces (Figure 1.4):

- the forces that natural phenomena such as wind exert can exert on objects;
- the forces an object can exert on other objects, and this force can be observed without knowing a lot about the inner structure of the object;

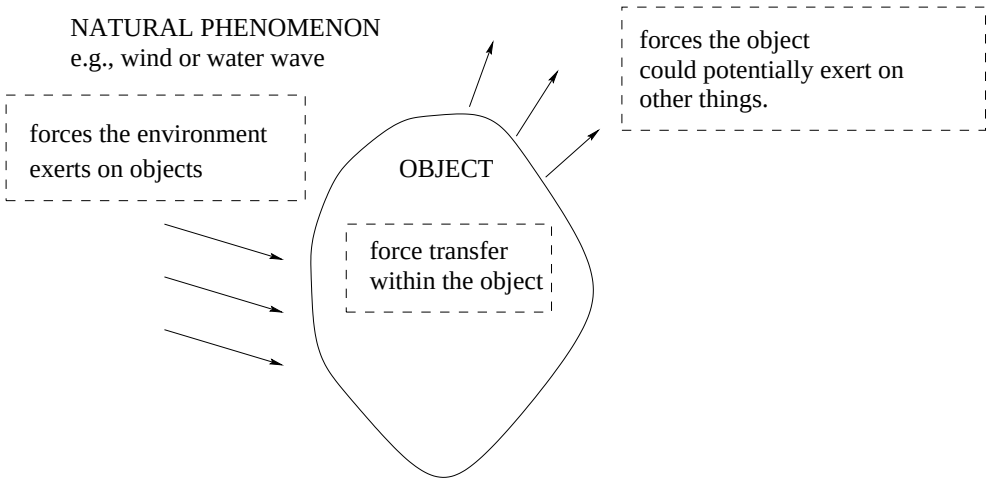

Figure 1.4: The forces in the video sequence which might be of interest to us.

- the forces an object exerts on itself.

In terms of types of object primitives that VIDA might be dealing with, here we list some of the primitives:

- fluids such water and wind.
- rigid objects such as the body of a car.
- articulated objects/figures such as animals.
- deformable objects such as tree branches and hair.
- particles such as snow.

There is a great deal of computer vision research which deals with rigid and articulated objects. The list above does not exhaust every type of primitive. There might be other primitives people might want to deal with and add to this list.

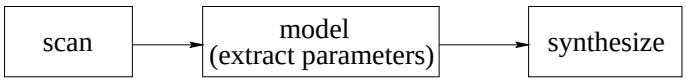

Figure 1.5: Three stages of our problem.

1.5 Goal and Tasks

The goal of VIDA is to extract parameters from a background video sequence, and to use these parameters to drive the animation of the foreground computer generated objects. One important application of VIDA is to make the computer generated foreground objects appear as natural participants of the video captured background.

The process consists of three stages as depicted in Figure 1.5: scan, model, synthesize. The **scan** stage analyzes the input video sequence. It would focus on the part of the scene or objects in the scene which we are interested in, and perhaps ignore much of the rest. Scanning will often employ techniques from computer vision. The **model** stage establishes reasonable models of observed objects or natural phenomena in which we are interested, and a model might have unknown parameter values initially. We analyze the information obtained in the scanning stage according to the model. This allows us to estimate and extract a model's parameters. Some examples of the parameters in which we are interested may be kinematic, such as position or velocity of objects, while other parameters may be dynamic, such as determining the forces exerted on objects. The **synthesize** stage uses the parameters we extracted in the modelling stage to drive the animation of computer generated objects.

Due to the richness and complexity of the natural as well as man-made environment surrounding us, there are many aspects to this problem. Hence there are a great many interesting, important and potentially difficult sub-topics of VIDA which await study.

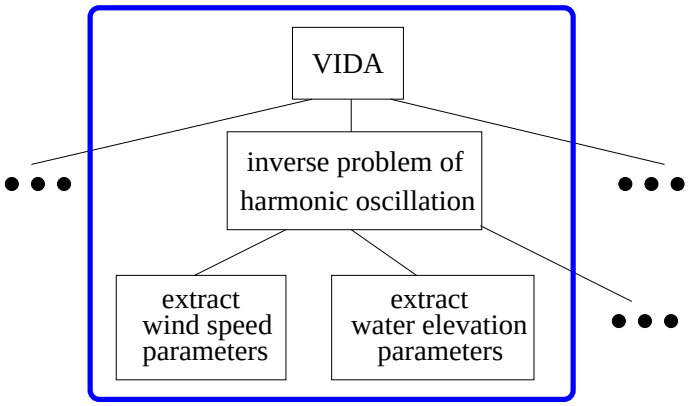

Figure 1.6: The layout of the components of this thesis.

We can not hope to cover all these problems in a single thesis. We instead shall establish the importance of the idea and demonstrate it through several key contributions. In particular, we identify the principle of harmonic oscillation as an extremely effective way of modelling the passive motion of many objects when disturbed by external forces.

1.6 Focus of This Thesis

In this thesis, we would like to demonstrate the basic principles behind VIDA by studying the inverse problem to harmonic oscillation and use it to "reverse engineer" natural phenomena such as wind or regular water waves. Here, by "reverse engineer" natural phenomena, we meant the following process: we observe harmonic oscillation of objects in the scene, infer the driving force, and deduce the natural phenomenon which induced the driving force, as shown in Figure 1.8. The layout of this thesis is as shown in Figure 1.6. Two examples will be used to demonstrate the strength and applications of this methodology. The first example is related to wind-object interaction. We observe the effect of the wind on a tree in the scene, estimate the wind speed from the tree's motion,

(a) Wind (b) Water wave

Figure 1.7: Two examples of inverse problems to harmonic oscillation.

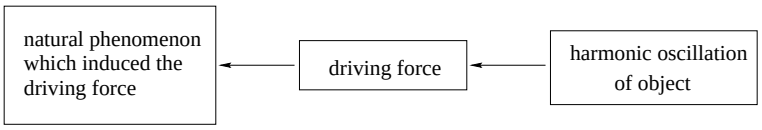

Figure 1.8: Going back in the harmonic oscillation process pipeline to "reverse engineer" parameters of natural phenomenon.

then use the wind information to make computer generated objects move, as shown in Figure 1.7(a). The second example is related to boat-water interaction. We study the motion of boats anchored to the shore as shown in Figure 1.7(b), estimate the water wave elevation which drives the boat motion, then use the water wave parameters to control a computer synthesized floating structure.

We have chosen such a focus for a number of reasons. First, it is an interesting computer vision problem with useful applications. Second, it is an interesting computer graphics problem and application. Third, our inverse of harmonic oscillation examples nicely demonstrate the principles behind VIDA. Next, we will explain each point in turn.

First, it is an interesting computer vision problem with useful applications. The problem VIDA poses requires us to combine the strengths of computer vision and computer graphics. Traditionally, the computer vision analysis of real world video

sequences has been closely linked to the development of artificial intelligence (AI) and robotics research. A great deal of progress has been made in vision which is both driven by and somewhat limited to satisfying the demands of AI/robotics research. Many computer vision experiments are based on man-made environments, and some are related to a natural environment. There is a limited amount of vision literature which direct targets the natural phenomena around us, such as wind and water. By vision problems targeting the natural phenomena, we mean trying to understand the parameters and characteristics of the natural phenomena by analyzing image sequences of a natural phenomenon. For example, we could be interested in problems such as trying to derive the wind parameters by observe a windy scene or trying to recover parameters of the water wave by observing the movement of the water. In recent years, Soatto et al.'s work in dynamic texture [SDW01] took an interesting step in such a direction. Soatto et al. studied sequences of images of a moving scene that exhibit certain stationarity properties in time. These include sea-waves, smoke, foliage, whirlwind, talking faces, traffic scenes etc. They present a novel characterization of dynamic textures that poses the problems of modelling, learning, recognizing and synthesizing dynamic textures on a firm analytical footing. Their algorithms learn models of the dynamic textures. Once learned, a model can be used to extrapolate synthetic sequences of similar phenomena to infinite length. Our idea of "reverse engineering" wind information from, say, wind-plant interaction and estimating water wave information from water-boat interaction are interesting attempts at analyzing natural phenomena via visual observation. We are taking the parameter extraction process one step further to infer the cause of the phenomenon, so that synthetic objects which are not similar to the observed objects can also be added to the scene.

In terms of computer vision inspired graphics research, in recent years, there is quite a lot of work on texture analysis in natural settings [HB95, Bon97, WL00, WL01, Tur01]. A number of recent algorithms create synthetic texture images by clever re-sampling from the original texture image. Although these do not provide an explicit model for texture, the results are visually stunning. The topics of texture re-sampling and synthesis serves as a good example of using computer vision research for computer graphics applications.

Second, it is an interesting computer graphics problem and application. It is useful in computer graphics to have synthetic objects be driven by video input. In the case of wind, we can use extracted wind parameters we estimate to drive a grass field, trees, snow, synthetic actor's hair and clothes, a flag, etc. Such details in the scene can be taken care of automatically fashion, relieving animators of tedious work. This provides a kind of indirect puppetry in which indirectly observed influences are consistently applied to all objects in a scene. In the case of water wave, we can use inferred wave elevation to animate other boats and floating structures. Using traditional techniques, the animator would have to calibrate each boat's motion by hand to match the background. They would have to make sure the boats' motions are not completely synchronized, but at the same time not completely unrelated, because the boats are moving in roughly the same water wave. VIDA provides a physically sound tool for automating this type of task.

Third, our inverse of harmonic oscillation examples nicely demonstrate the principles behind VIDA. Specifically, we show how to find the force or the cause of the observed motion and use the cause or the correct force to drive a new object. One application of VIDA is to maintain the continuity and consistency of the (foreground) synthetic objects and the (background) video input. This takes the guess work away

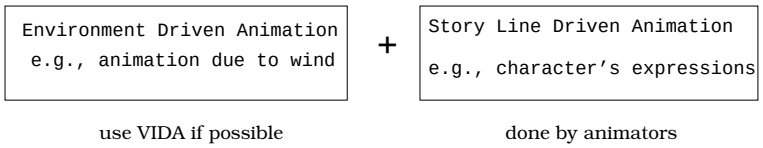

Figure 1.9: This is one possible way of using VIDA.

from the artists or animators, but leaves them a high degree of creative control.

There can be two types of animation in the scene as shown in Figure 1.9. One type is closely related to the environment the objects are in, so it has to be environment driven, which is where VIDA could be very useful. The other type has more to do with the story being told by the animation, which is usually best done by the animators. The inversion of harmonic oscillation is by no means the only way to demonstrate the strength of VIDA, but it is certainly an interesting and novel way to bring out the principles and explain the processes involved in VIDA. Further, it is a novel technical contribution in its own right.

1.7 Physics Based Approach

As we mentioned in section 1.6, we will demonstrate the basic principles behind VIDA using the inverse harmonic oscillation problem and its applications. Depending on the VIDA application, we can choose to use a physics based approach or some other approach. In the case of the inverse harmonic oscillation problem, we made an important decision to use a physics based approach. We think, in this case, physics is both important and it helps to simplify the task.

If we look around in our natural environment, there are many circumstances in nature in which something is oscillating and in which the resonance phenomenon occurs [Fey63].

Harmonic oscillation is studied in marine engineering, because a ship's swaying, heaving and vibration are oscillations. It is studied in material engineering, because the motion of beam under wind force or during an earthquake is harmonic oscillation. Many more examples of harmonic oscillation can be found in everyday life, and its ubiquity is reflected in textbooks for many other disciplines such as mechanical engineering, electrical engineering, chemical engineering, etc.

There is a unified and simple physics formulation which explains the phenomenon of harmonic oscillation. We will give a concise mathematical description of the physics principle in section 2.

People might ask: when you add computer generated objects to the video captured scene, why not just animate the virtual objects by hand? Physics provides a consistent way to account for the motion of objects we see in real world video sequences. If we can extract a number of parameters and infer the cause of the motion based on physics principles, then it makes the animation of virtual object more automatic, more consistent and simpler. For instance, when we want to generate the motion of many boats influenced by the same waves, animators would have to ensure that the boats' motions are not completely synchronized, but at the same time not completely unrelated. This is hard to do by hand, but it is easy to do using physics.

Another question people might ask is why not cheat? For instance, when you have trees moving in the wind in the original video, why not link the real tree's movement to the virtual tree by an invisible spring, so that the new tree's movement is influenced by the real tree's motion? We would like to point out that this trick is not simpler or computationally cheaper than our inverse harmonic oscillation solution. First, to figure

out the motion of the original tree automatically, one still has to use computer vision to track the original tree in the video. Second, the physics formulation of harmonic oscillation is very simple and easy to compute, so the idea of linking the virtual tree to the real tree by an invisible spring is not necessarily computationally cheaper.

In general, we can try to "cheat" and create each different "cheating" trick for a different application. So for a hundred different situations, we would have a hundred different tricks. Such an ad hoc approach makes things more complicated than necessary. Instead of using different tricks, the physical formulation of harmonic oscillation can be used in a consistent way in each situation which falls into this category. It is a principled approach. It is simple, straight forward and consistent.

As we will discuss later in section 2, there are two key parameters in our parameter extraction process — the natural frequency of the oscillation ω_0, the damping coefficient γ. This helps us to capture the characteristic of the oscillating system in a very simple and concise fashion. It simplifies the problem. The physics principle captures the essence of harmonic oscillation in a very clean way.

1.8 Potential Limitations

There are potential limitations and challenges surrounding VIDA.

We want the computer generated objects to appear as natural participants in the video. Sometimes, we might need to alter the video sequence slightly to create a more realistic effect. For example, we might have a sandy beach video sequence into which a virtual character steps. The character should leave footprints and cast shadows on the

beach. It might require slight alteration of the scene to create footprints on the sand. While this phenomenon itself is not hard to model and simulate, the problematic aspect is its identification.

Furthermore, the introduction of synthetic objects might have a large influence on the video captured scene. In this case, it is no longer a simple case of dealing only with video input driven animation. The reverse problem also needs to be studied, which is how the synthetic objects would change the video sequence. In that case, we would need to augment the VIDA concept to accommodate and create more interaction between the captured video sequence and the synthetic objects. The captured video sequence might also need to be modelled, parameterized and changed. This is by no means a simple task.

For the time being, we would like to concentrate our effort on studying video input driven animation (VIDA) where the synthetic character does not have too much influence on the video captured sequence.

1.9 Overview

In this section, we give an overview of the organization of this thesis. In chapter 2, we describe the VIDA inverse problem of harmonic oscillation. In chapter 3, we review related work. In chapter 4, we review background physics related to forced harmonic oscillation. In chapter 5, we focus on the modelling part of VIDA, and explore some of the basic ingredients in extracting useful parameters by observing harmonic oscillation. In chapter 6, we discuss the example of "reverse engineering" wind speed parameters by observing object motion due to the wind. In chapter 7, we look at the example of

estimating water wave elevation parameters by studying the motion of boats anchored to shore. In chapter 8, we validate our approach and provide methods for calculating the error in our estimation. In chapter 9, we summarize the VIDA problem and our methodology for approaching it. We lastly discuss our contributions and future work.

Chapter 2

The Inverse Harmonic Oscillation Problem

To demonstrate the principles and strength of VIDA, we will develop a methodology for the inverse harmonic oscillation problem, and use our approach to extract parameters of natural phenomena such as wind and water waves from relevant video sequences.

Before we start, we would like to say a few words about why we use the harmonic oscillation problem as our example. If we look around in our natural environment, there are many circumstances in nature in which something is oscillating and in which the resonance phenomenon occurs [Fey63].

In many situations when an object is slightly disturbed from its equilibrium or resting position, there might be a lone or combined push and pull force that tends to restore it to the equilibrium position. For example, if a marble is rolled up the side of a bowl, gravitational attraction to the Earth pulls it down, and the walls of the bowl confine it, so it moves back towards the bottom of the bowl. If an atom of, say, the table, is pushed

slightly out of place, its chemical bonds to its neighbours pull it back in place. If air molecules gather in a crowd of higher density, the added intermolecular collisions drive them apart again. If a mass hanging from a spring is pulled down, the spring pulls the mass back up. When a weeble toy is pushed to the side, it tends to rotate back towards the upright position, due to its round bottom and its very low center of gravity. A tree branch pulled away from its resting position will try to return to its resting position due to the strain, stress and the elasticity of the tree branch. If a soap bar floating in our bath tub is pushed down on one side and released, it will try to get back to its original upright facing state. A boat floating on the water works in similar ways as that of the soap floating on the water. In each of these cases, if no external forces were added to the system, due to the interplay of the inertia, restoring force and frictional force, the object oscillates back and forth about its equilibrium position and eventually stops at its resting position.

Feynman [Fey63] points out many more examples of harmonic oscillation. For instance, the atmosphere which we suppose surrounds the earth evenly on all sides is pulled to one side by the moon or, rather, squashed prolate into a double tide, and if we could then let it go, it would go sloshing up and down. It is an oscillator. This oscillator is driven by the moon, which is revolving about the earth. Not surprisingly, the idea of measuring and inferring the parameters of harmonic oscillation has existed and been used by scientists in many areas of research. In the case of the earth's atmosphere for example, initially, people inferred the oscillation parameters of the atmosphere by observing the size of the atmospheric tides and the phase, i.e., the amount of delay. Another way to confirm this inferred information is by hoping that someone disturbs the atmosphere,

then it would oscillate with its natural frequency ω_0. It turns out that there was such a sharp disturbance in 1883, when the Krakatoa volcano exploded and half of the island was blown away. It made such a terrific explosion in the atmosphere that the period of oscillation of the atmosphere could be measured. The period of oscillation came out to $10\frac{1}{2}$ hours.

There are many more examples of harmonic oscillation which we can observe in everyday life. In harmonic oscillation, the resting state of the system corresponds to the state with the lowest energy. Almost everything we see around us tends to return to and rest at the configuration with the lowest energy. That is why harmonic oscillation is so ubiquitous.

The idea of observing an harmonic oscillation and inferring parameters of the oscillating system is commonly used in different scientific disciplines such as mechanical engineering, electrical engineering, naval architecture, etc [Fey63]. In this chapter, we will explain the basic idea behind inferring parameters of harmonic oscillation, and defining the inverse harmonic oscillation problem. This is the basic component of this thesis. It will be used again and again in more specific applications such as retrieving wind and water wave parameters later in this thesis.

Of course, the natural environment surounding us is very complex. There are many natural phenomena that can not be modeled using harmonic oscillation. The related VIDA problems would be interesting to investigate in the future. However, in this thesis, we will only focus on the inverse harmonic oscillation problem, and use it to demonstrate the principles behind VIDA. The ubiquity of harmonic oscillation makes it an attractive class of problems for the purpose of our demonstration, and it can be widely used in

many harmonic oscillation situations which we encounter in our daily lives.

2.1 Forward Problem vs. Inverse Problem

The forward problem of harmonic oscillation is well-studied. A simple example would be the forced harmonic oscillation of a spring-mass system as shown in Figure 2.1, in which the point mass with mass m is placed at the top of a horizontal surface, and it is connected to the wall by a spring. As depicted in Figure 2.2, typically, we are given the external force $f(t)$ as a function over time and other properties of the oscillating system. The properties of the oscillating system includes the mass, the initial position and velocity, the damping characteristic, and the stiffness properties. We are looking for the response of the system, such as its displacement from the equilibrium position over time.

We would like, however to consider the inverse problem to harmonic oscillation. As shown in Figure 2.3, in the inverse problem, the only information we are given is the observed displacement of the oscillator. We do not know the properties of the oscillating system, the external driving force, and parameters of the phenomenon that cause forces to be exerted on the oscillating system. The solution to the inverse problem of harmonic oscillation uses the known observed displacement of the oscillator to recover the unknown

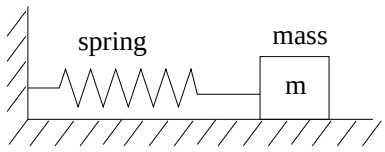

Figure 2.1: An example oscillator: a spring-mass system.

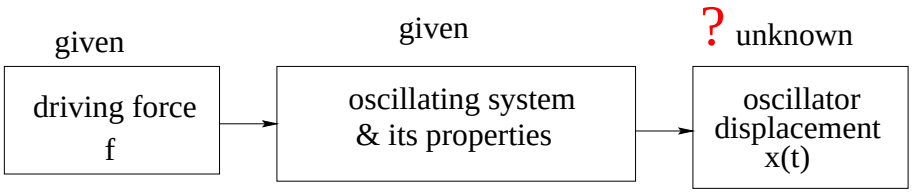

Figure 2.2: Harmonic oscillation, the forward problem.

parameters which caused the motion. By observing the motion of an oscillating system, if we are able to extract the parameters of the natural phenomenon which determines the forcing function of such a harmonic motion, we might be able to re-use these parameters in the forward simulation of computer synthesized objects. This inverse problem, which is a technical challenge on its own to solve, also nicely demonstrates the principles and methodology behind VIDA.

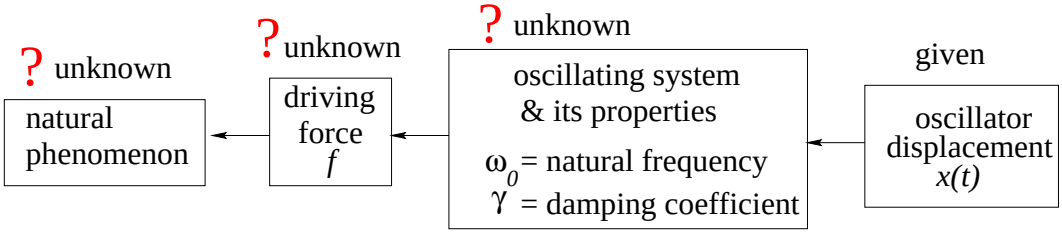

Figure 2.3: Estimating the driving force by observing the displacement.

2.2 Problem Formulation

In a harmonic oscillation system, the speed at which the system oscillates is heavily influenced by the natural frequency ω_0 of the oscillating object. When the outside driving force dies down, the oscillation should also decay. This decay depends on the damping characteristic γ as some proportion of the energy in the system is expended in order to overcome friction and other resistance.

In general, the physical relationship between the displacement of an object and the external force that caused the motion is described by Newton's law:

$$mass \times acceleration = net\ force.$$

For a harmonic oscillator, the net force is the sum of the external driving force f, the restoring force $-kx$, and the frictional force $-c\frac{dx}{dt}$. That is,

$$m\frac{d^2x}{dt^2} = -c\frac{dx}{dt} - kx + f,$$

or

$$m\frac{d^2x}{dt^2} + c\frac{dx}{dt} + kx = f, \tag{2.1}$$

where f is the force which drives the oscillator, x is the displacement, m, c and k are the mass, damping characteristic and stiffness of the oscillator respectively. The term $m\frac{d^2x}{dt^2}$ accounts for the force due to inertia, $c\frac{dx}{dt}$ accounts for the force due to friction or resistance, and kx accounts for the restoring force, say due to the stiffness of the spring in a spring-mass system. To make Equation 2.1 easier to analyze, we divide both sides by m, where $m > 0$. Let $\gamma = c/m$ and $k/m = \omega_0^2$, we have

$$\frac{d^2x}{dt^2} + \gamma\frac{dx}{dt} + \omega_0^2 x = \frac{f}{m}. \tag{2.2}$$

We call γ the damping coefficient and ω_0 the natural frequency of the oscillator.

As we will show in chapter 4, we can treat the driving force as an input signal, the

oscillating system as a filter and the displacement of the oscillator as the output of the system, as shown in Figure 2.4. We want to go backwards in this chain, starting from the observed output and trying to estimate the filter and the input.

2.3 Suggested Steps

In general, there are three stages to the process: scanning raw data; modelling; and synthesis. In preparation for their solution later in this thesis, we shall summarize the steps. More explicitly, the information flow is as shown in Figure 2.5:

Step 1: Extract motion information from the video sequence.

Step 2(a): Use the oscillator's displacement over time to estimate the natural frequency ω_0 and the damping coefficient γ of the oscillating system.

Step 2(b): Use the displacement data $x(t)$ and the estimated information about the properties of the oscillating system to compute the external driving force $f(t)$ which caused the motion.

Step 2(c): Use the estimated driving force $f(t)$ to infer the parameters of the natural phenomenon which caused the motion.

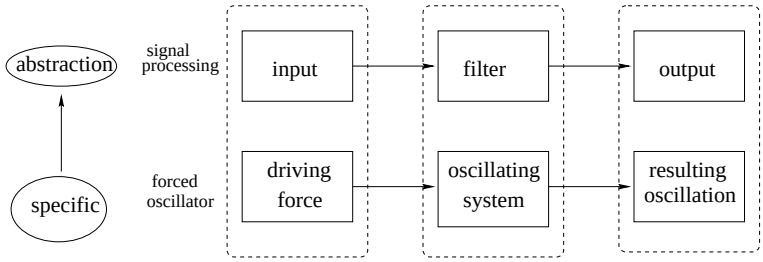

Figure 2.4: Forced harmonic oscillation in terms of input, filter and output relation.

Step 3: Once we have extracted the parameters of the natural phenomenon, we use these parameters to apply forces to computer generated objects.

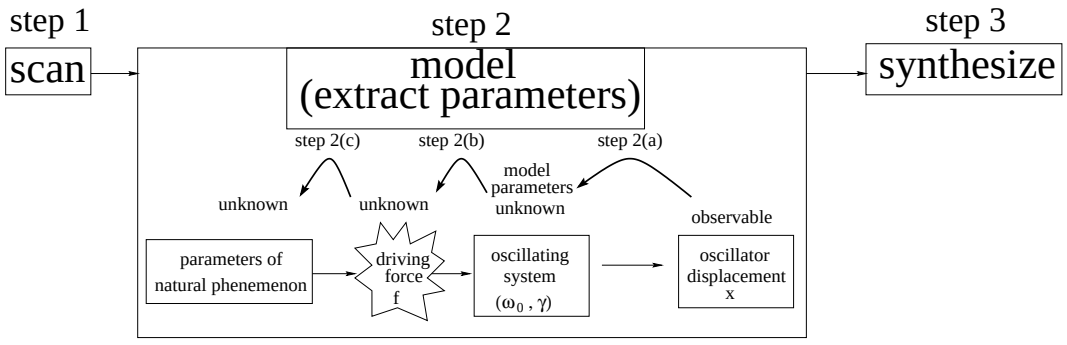

Figure 2.5: The process for the VIDA inverse problem of harmonic motion: suggested steps.

In chapter 3, we will talk about related work. In chapter 4, we introduce some background information about harmonic oscillation analysis. In later chapters, we will cast two different situations as instances of inverse problems of harmonic oscillation. One is related to wind-object interaction, and the other one is related to water-boat interaction in regular water waves.

Chapter 3

Related Work

We break related work into two categories. One category consists of work which has a similar flavor to our idea, i.e., using one type of input signal to control another type of output signal or to create a new object/character. The other category consists of work related to the subtasks of our VIDA inverse problem of harmonic oscillation, namely, scan, model, synthesize.

3.1 Indirect Puppetry

In this section, we talk about the work which has a similar flavor to ours which can be viewed as some kind of indirect puppetry.

Bregler et al. [BCS97] propose the idea of video rewrite, where they use existing video footage to automatically create new video of a person mouthing words that she did not speak in the original footage. Video rewrite uses computer-vision techniques to track points on the speaker's mouth in the training footage, and it uses morphing techniques to

combine these mouth gestures into the final video sequence. The new video combines the dynamics of the original actor's articulations with the mannerisms and setting dictated by the background footage. In the paper "Voice Puppetry" [Bra99], Brand introduced a method for predicting a control signal from another related signal, and applied it to voice puppetry: generating full facial animation from expressive information in an audio track. Using techniques from computer vision, the voice puppet learns a facial control model from estimates of real facial behavior, automatically incorporating vocal and facial dynamics such as co-articulation. Animation is produced by using audio to drive the model, which induces a probability distribution over the manifold of possible facial motions. Brand presents a linear-time closed-form solution for the most probable trajectory over this manifold. The output is a series of facial control parameters, suitable for driving many different kinds of animation ranging from video-realistic image warps to 3D cartoon characters. This is a nice example of using one type of signal to control another type of signal by studying their correlation. Though it has a similar flavor to VIDA, voice puppetry work and VIDA explore quite different research areas. Video rewrite [BCS97] and Voice puppetry [Bra99] are based on correlation, whereas our work is based on physics.

Popović and Witkin [PA99] introduced an algorithm for transforming character animation sequences that preserves essential physical properties of the motion. They take the approach of motion transformation as the underlying paradigm for generating computer animations. They take physically consistent motion, such as captured motion, as an input. Their transformation algorithm first constructs a simplified character model and fits the motion of the simplified model to the captured motion data. From this

fitted motion, they obtain a physical spacetime optimization solution that includes the body's mass properties, pose and footprint constraints, muscles and an objective function. To edit the animation, they modify the constraints and physical parameters of the model and other spacetime optimization parameters. From this altered spacetime parameterization, they compute a transformed motion sequence. Lastly, they map the motion change of the simplified model back onto the original motion to produce a novel animation sequence. To some extent, this approach could be used by VIDA to study the forces or the spacetime optimization parameters within one object and eventually adapt them to create or drive other objects. While they are interested in the forces and mechanics within an articulated figure, we are interested in the forces surrounding us in the natural environment, such as forces generated by wind and water waves.

Hoshino and Saito [HYS01] proposed a new technique for automatic registration of virtual objects with the human body images. As an example, they merge computer generated clothes onto the human in a video sequence. First, they track current 3D pose of human figure using the spatio-temporal analysis and the structural knowledge of human body. Then they take computer generated clothes and merge them with the human in the video. Both their work and our work takes a physics based approach. Their work is based on analyzing human body motion, whereas in this thesis, we study the motion of objects due to natural phenomena such as wind and water waves.

3.2 Vision Based Facial Motion Modelling

Researchers have studied using video and computer vision to capture more subtle details of real human facial expressions and use this information to improve computer facial motion modelling.

Williams [Wil90] describes a means of acquiring the expressions of real faces and applying them to computer-generated faces. To track facial expressions of live performers, Williams applied Scotchlite spots to a performer's face to mark the important facial features. A spot tracking routine is used to keep track of the spot as the performer makes different facial expressions. The tracked facial expression is directly mapped to a computer generated 3D facial model. This work is concerned with the motion itself, whereas we are interested in the cause of the motion.

Some particularly interesting work in vision based on models of natural phenomena is that of Black and Yacoob [BY95b, BY95a] in tracking and recognizing rigid and non-rigid facial motions using local parametric models of image motion. Parametric flow models are popular for estimating motion in rigid scenes. They observed that within local regions in space and time, such models not only accurately model non-rigid facial motions but also provide a concise description of the motions in terms of a small number of parameters. These parameters are intuitively related to the motion of facial features during facial expressions and they showed how expressions such as anger, happiness, surprise, fear, disgust and sadness can be recognized from the local parametric motions in the presence of significant head motion. Their idea of extracting a concise description of motion in terms of a small number of parameters is similar to ours. We try to extract

parameters of harmonic oscillation. Their work is concerned more with facial motion modelling, whereas this thesis studies the parameter extraction for natural phenomena such as wind and water waves.

3.3 Computational Perception of Scene Dynamics

The understanding of the dynamics in a scene is an interesting problem. Mann et al. [MJ97] point out that understanding observations of interacting objects requires one to reason about qualitative scene dynamics. For example, on observing a hand lifting a can, we may infer that an "active" hand is applying an upwards force by grasping to lift a "passive" can. They implemented a computational theory that derives such dynamic descriptions directly from camera input. Their approach is based on an analysis of the Newtonian mechanics of a simplified scene model. Interpretations are expressed in terms of assertions about the kinematic and dynamic properties of the scene. The feasibility of interpretations relative to Newtonian mechanics is determined by a reduction to linear programming. To select plausible interpretations, multiple feasible solutions are compared using a preference hierarchy. This is a useful approach for understanding the forces between objects. In this thesis, our investigation focuses on the forces that natural phenomena such as wind and water exert on objects.

3.4 Video Based Animation

Increasingly, computer animation has turned to video input for inspiration. Schödl et al. [SSSE00] introduced a video based animation technique called *video texture*. They take a source video with fixed camera position, and the video has a finite set of images. They randomly rearrange and possibly blend original frames from the source video to synthesize a continuous infinitely varying stream of images. Some of their example videos include a candle flame, a clock, a flag fluttering in the wind, a campfire, a video of a woman posing for a portrait, a waterfall and blowing grass. Their work is based on visual continuity. Our work is based on physics and we are also interested in the cause of the object movement. Schödl and Essa [SE01] present techniques for rendering and animation of realistic scenes by analyzing and training on short video sequences. This work extends the video texture idea. Schödl and Essa create video sprites which are a special type of video texture. With video sprites, instead of storing whole images, the object of interest is separated from the background and the video samples are stored as a sequence of alpha-matted sprites with associated velocity information. Schödl and Essa can be rendered anywhere on the screen to create a novel animation of the object. They present methods to create such animations by finding a sequence of sprite samples that is both visually smooth and follows a desired path. To estimate visual smoothness, they train a linear classifier to estimate visual similarity between video samples. Both their work and ours are video based animation. Their work studies the visual smoothness and how an object can follow a desired path, whereas we are interested in the force which caused the motion of the object. In the video texture paper [SSSE00], the camera is static

for all the video sequences. If a video sequence is taken using a moving camera, it poses a problem. Fitzgibbon [Fit01] considers the registration of sequences of images where the observed scene is entirely non-rigid, for example, a camera flying over water, a panning shot of a field of sunflowers in the wind, or footage of a crowd applauding at sports event. The problem is that, in these cases, it is not possible to impose the constraint that world points have similar colour in successive views, so existing registration techniques can not be applied. The relationship between a point's colours in successive frames is essentially a random process. Fitzgibbon proposes a technique for automatically searching for the camera registration parameters for such nowhere-static scenes. Fitzgibbon's work is focussed on the problem of recovering the camera registration parameters related to video based animation. We can eventually use their work when we experiment settings where we video tape the scene with a moving camera. For the time being, our experimental setup consists of only a static camera.

Our work differs from the above, in that we are interested in the cause of the object motion in the scene and the parameters of the natural phenomenon which produced the objects' motion. Once we extract the parameters of the natural phenomenon, we can use them to drive the animation of objects that are significantly different from the objects captured. For instance, after estimating the wind speed in the scene by observing the tree movement, we can apply the wind to snow, hair, clothes, etc.

3.5 Vision Directed Sampling and Synthesis

In recent years, the general idea of vision directed sampling and synthesis has been used in many computer graphics research areas such as augmented reality, and texture synthesis from example.

3.5.1 Augmented Reality

The ability to combine real video with computer generated objects enhances the usefulness of both. Many researchers are interested in the subproblem of combining real video with computer generated objects and computer generated lighting conditions. The result is sometimes referred to as *computer augmented reality*. Some of the problems people have looked into are recovering common viewing parameters and common illumination, recovering surface reflectance function, etc.

Fournier, Gunawan and Romanzin [FGR93] presented a technique for approximating the common global illumination for real video and computer generated object, assuming some elements of the scene geometry of the real world and common viewing parameters are known. They approximate the global illumination of the merged environment by making the real video part of the solution to the common global illumination computation. The objects in the real scene are replaced by a set of covering boxes. The image intensity of the real video is used as the initial surface radiosity of the visible part of the boxes; the surface reflectance of the boxes is approximated by subtracting an estimate of the illuminant intensity based on the concept of ambient light; finally, global illumination using a classic radiosity computation is used to render the surface of the computer

generated objects with respect to their new environment and for calculating the amount of image intensity correction needed for surfaces of the real image. This approach uses only a single image. However, this technique is limited to perfectly diffuse environments and is not able to take specular surfaces into account.

In [DRB97], Drettakis, Robert and Bougnoux pointed out that providing common illumination between the real and synthetic objects can be very beneficial. They proposed a new frame work for solving this problem. They addressed three specific aspects of the common illumination problem for computer augmented reality: simplification of camera calibration and modelling of the real scene; efficient update of illumination for moving computer generated objects; and efficient rendering of the merged world. Their approach allows interactive update rates on mid-range graphics workstation.

In [YDH99], Yu et al. presented a method for recovering the reflectance properties of all surfaces in a real scene from a sparse set of photographs, taking into account both direct and indirect illumination. The result is a lighting-independent model of the scene's geometry and reflectance properties, which can be rendered with arbitrary modifications to structure and lighting via traditional rendering methods. Their technique models reflectance with a low-parameter reflectance model, and allows diffuse albedo to vary arbitrarily over surfaces while assuming that non-diffuse characteristics remain constant across particular regions. The method's input is a geometric model of the scene and a set of calibrated high dynamic range photographs taken with known direct illumination. The algorithm hierarchically partitions the scene into a polygonal mesh, and uses image-based rendering to construct estimates of both the radiance and irradiance of each patch from the photographic data. The algorithm computes the expected location of specular

highlights, and then analyzes the highlight areas in the images by running a novel iterative optimization procedure to recover the diffuse and specular reflectance parameters for each region. Lastly, these parameters are used in constructing high-resolution diffuse albedo maps for each surface.

Boivin and Gagalowicz [BG01] presented a new method for recovering an approximation of the bidirectional reflectance distribution function (BRDF) of the surfaces present in a real scene. This is done from a single photograph and a three dimensional geometric model of the scene. The result is a full model of the reflectance properties of all surfaces, which can be rendered under novel illumination conditions with, for example, viewpoint modification and the addition of new synthetic objects. Their technique produces a reflectance model using a small number of parameters. These parameters nevertheless approximate the BRDF and permit the recovery of the photometric properties of diffuse, specular, isotropic or anisotropic and textured objects. The input data are a geometric model of the scene including the light source positions and the camera properties, and a single image captured using this camera. Their algorithm generates a new synthetic image using classic rendering techniques and a Lambertian reflectance model of the surfaces. It then iteratively compares the original image to the new one, and chooses a more complex reflectance model if the difference between the two images is greater than a user-defined threshold.

These pieces of work have similar flavor as VIDA in the sense that they are trying to analyze a scene and recover properties of objects or parameters of material, then use this information to add new objects or modify the scene in a consistent way. Their work is focussed on rendering and we are interested in animation.

3.5.2 Texture Sampling and Synthesis

Many researchers have proposed interesting methods of sampling images to produce synthesized textures.

Heeger and Bergen [HB95] presents a technique for synthesizing an image or solid texture that matches the appearance of a given texture sample. The key advantage of this technique is that it works entirely from the example texture, requiring no additional information or adjustment. The technique starts with a digitized image and analyzes it to compute a number of texture parameter values. Those parameter values are then used to synthesize a new image of any size that looks in its color and texture properties like the original. The analysis phase is inherently two-dimensional since the input digitized images are two dimensional. The synthesis phase, however, may be either two- or three-dimensional. For the three dimensional case, the output is a solid texture such that planar slices through the solid look like the original scanned image. In either case, the two or three dimensional texture is synthesized so that it tiles seamlessly.

De Bonet [Bon97] described a two-phase process for analysis and synthesis of texture images. The input texture is first analyzed by computing the joint occurrence, across multiple resolutions, of several of the features used in psychophysical models. In the second phase, a new texture is synthesized by sampling successive spatial frequency bands from the input texture, conditioned on the similar joint occurrence of features at all lower spatial frequencies. The sampling methodology is based on the hypothesis that texture images differ from typical images in that that there are regions more discriminable at certain resolutions than at others. By rearranging textural components at locations

and resolutions where the discriminability is below threshold, new texture samples are generated which have similar visual characteristics

Efros and Leung [EL99] proposed a non-parametric method for texture synthesis. The texture synthesis process grows a new image outward from an initial seed, one pixel at a time. A Markov random field model is assumed, and the conditional distribution of a pixel given all its neighbors synthesized so far is estimated by querying the sample image and finding all similar neighborhoods. The degree of randomness is controlled by a single perceptually intuitive parameter. The method aims at preserving as much local structure as possible and produces good results for a wide variety of synthetic and real-world textures.

Wei and Levoy [WL00] present a very simple algorithm that can efficiently synthesize a wide variety of textures. The inputs consist of an example texture patch and a random noise image with size specified by the user. The algorithm modifies this random noise to make it look like the given example. This technique is flexible and easy to use, since only an example texture patch is required. New textures can be generated with little computation time, and their tileability is guaranteed. The algorithm is also easy to implement; the two major components are a multiresolution pyramid and a simple searching algorithm.

Turk [Tur01] pointed out that many natural and man-made surface patterns are created by interactions between texture elements and surface geometry. He suggested that the best way to create such patterns is to synthesize a texture directly on the surface of the model. Given a texture sample in the form of an image, he created a similar texture over an irregular mesh hierarchy that has been placed on a given surface.

A texture created this way fits the surface naturally and seamlessly. Wei and Levoy [WL01] address the same problem by extending their own texture synthesis method in [WL00].

In [SDW01], Soatto et al. studied dynamic textures, which are sequences of images of moving scenes that exhibit certain stationarity in time. These include sea-waves, smoke, foliage, whirlwind and also talking faces, traffic scenes, etc. They presented a novel characterization of dynamic textures that poses the problems of modelling, learning, recognizing and synthesizing dynamic textures on a firm analytical footing. They borrowed tools from system identification to capture the "essence" of dynamic textures. They do so by learning and identifying models that are optimal in the sense of maximum likelihood or minimum prediction error variance. For the special case of second-order stationary processes, they identified the model in closed form. Once learned, a model has predictive power and can be used for extrapolating synthetic sequences to infinite length with negligible computational cost.

These research works are similar to our idea in the sense that we are both trying to synthesize new image or image sequences which are consistent with the original input. Their work is concerned with texture synthesis, and we are interested in inferring the cause of the motion of the observed objects in a video footage.

3.6 Subtasks of The Inverse Harmonic Oscillation Problem

There are three stages to the inverse harmonic oscillation problem: scanning, modelling and synthesis. Next, we discuss related work at each stage.

3.6.1 Scanning

Scanning an input video sequence so as to analyze the motion of an object in the scene is a typical vision problem. There is a great deal of literature on it. In our application, we would like to track particular features in an image sequence from frame to frame.

Shi and Tomasi show how to monitor the quality of image features during tracking using a measure of feature dissimilarity that quantifies the change of appearance of a feature between the first and the current frame [ST94]. The idea is straightforward: dissimilarity is the feature's root mean square (rms) residue between the first and the current frame, and when dissimilarity grows too large, the feature should be abandoned. They provide experimental evidence that pure translation is not an adequate model for image motion when measuring dissimilarity, but affine image changes are adequate. They propose a numerically sound and efficient way of determining affine changes by a Newton-Raphson style minimization procedure. In addition, they propose a more principled way to select features than the more traditional "interest" or "cornerness" measures. Specifically, they show that features with good texture properties can be defined by optimizing the tracker's accuracy.

Tracking features in an image involves computing the optical flow of the region we

want to track. The computation of optical flow relies on merging information available over an image patch to form an estimate of 2D image velocity at a point. This merging process raises a host of issues, which includes the treatment of outliers in component velocity measurements and the modelling of multiple motions within a patch which arise from occlusion boundaries or transparency. Jepson and Black [JB93] present an approach that allows them to deal with these issues within a common framework. Their approach is based on the use of a probabilistic mixture model to explicitly represent multiple motions within a patch. They use a simple extension of the expectation maximization algorithm to compute a maximum likelihood estimate for the various motion parameters. From the wide range of different measurement strategies for component velocities, they chose a phase-based approach. The approach they use is based on only two consecutive frames, and the actual component velocity measurement method is similar to the phase-based stereo disparity measurement scheme discussed by Jepson and Jenkin [JJ89]. Experiments have indicated that this approach is computationally efficient and can provide a robust estimates of the optical flow values in the presence of outliers and multiple motions.

Jepson, Fleet and El-Maragh [JFEM01] proposed a framework for learning robust, adaptive, appearance models to be used for motion-based tracking of natural objects. The approach involves a mixture of stable image structure, learned over long time courses, along with 2-frame motion information and an outlier process. An on-line EM-algorithm is used to adapt the appearance model parameters over time. They developed an implementation of this approach for an appearance model based on the filter responses from a steerable pyramid. This model is used in a motion-based tracking algorithm to provide

robustness in the face of image outliers, such as those caused by occlusions. It is also provides the ability to adapt to natural changes in appearance, such as those due to facial expressions or variations in 3D pose. They show experimental results on a variety of natural image sequences of people moving within cluttered environments. In this thesis, the automatic tracking of objects in the scene was done using their system. Their work is very much drawing its strength from all the papers we have discussed in this section (section 3.6.1).

3.6.2 Modelling

Mathematical modelling generally allows one to characterize the behavior of a phenomenon and to make predictions about it. In our case, modelling is intended to extract parameters from the observed motion sequences in the input video. We can then use these parameters to drive the animation of synthesized objects. Even more specifically, in our VIDA inverse problem of harmonic oscillation, we want to model the object motion in terms of forced oscillation. Such oscillations are resonance phenomena. The estimation of parameters of a resonance phenomenon has been studied by physicists and researchers in many other disciplines [Fey63]. For instance, the response of the earth's atmosphere to the tidal pull of the moon is that of oscillation. From the size of the atmospheric tides, and from the phase, the amount of delay, one can estimate the atmosphere's oscillation properties, such as its natural frequency and damping coefficient. Such estimation is very similar in spirit to our task of estimating the vibrational properties of a harmonic oscillation system.

In this thesis, we discuss two applications of the inverse problem to harmonic oscillation. In the first application, we model the tree or stem of a plant as a cantilever beam structure. The properties of such a structure are well studied in structural engineering [CP93, Smi88, Tim49, Tim47, TL25, TM49]. In the second application, we model a boat's oscillating motion due to regular water waves. This type of dynamics is studied in Marine Engineering and Naval Architecture [Bha78, GJ82, Mor90, MT87, Pat89, PS78, RC42]. We used many of the mathematical formulas in these books.

3.6.3 Synthesis

In this thesis, we will use the parameters extracted by the modelling process to drive computer animation.

3.6.3.1 Synthesizing Plant Motion

The wind speed estimation problem is to take a video footage of a windy scene, analyze the motion of the objects moving in the wind, try to estimate the parameters of the wind force on the objects, and infer the parameters for large scale wind velocity fluctuation. For the wind speed estimation problem, we re-apply the extracted wind field parameters to synthetic plants and then simulate the result. The synthesizing process of a plant or a tree moving in the wind has been studied by several researchers.

Shinya and Fournier [SF92] synthesized the motion of trees and plants by modelling a cascade system of three components: wind model, dynamic model and deformation model. Wind models produce spatio-temporal wind velocity fields using the power spectrum and auto-correlation of wind. Dynamic models describe the dynamic response of

the systems, using equation systems or response functions. Deformation models produce deformed shapes of objects according to the geometric models of the object and the results of the dynamic systems. The simulation is done by integrating dynamical equations over time.

Stam [Sta97] addresses the problem of realistically simulating the motion of tree-branches subjected to turbulence. Since the resulting motion is random in nature, it is modelled as a stochastic process. He synthesizes this process directly by filtering white noise in the Fourier domain. The filter is constructed by performing a modal analysis of the tree. He uses a sophisticated numerical technique to compute the first few significant modes of large trees. This enables him to compute complicated motions without the necessity of integrating dynamical equations over time. A user can view and manipulate tree-motions in real-time.

Shinya, Mori and Osumi [SMO98] propose a simple, robust and efficient method to compute periodic motion from systems of linear equations. They applied their method to model the swaying motion due to wind of bamboo in a given wind field.

These pieces of work are strictly those of synthesizing the tree's movement. There is no attempt at "reverse engineering" the wind speed by observing the tree's movement in the wind. After we use our approach to infer wind speed parameters, one can use their work to make elaborate simulations of complicated tree structure moving in the wind field.

3.6.3.2 Synthesizing Boat Motion

The dynamics of the motion of boats in regular water waves are studied mainly in naval architecture and marine engineering. There is commercially available software package such as SHIPMA [Hyd] developed by Delft Hydraulics which uses a mathematical model to compute the track and course angle of a vessel. The model takes into account the influences of wind, waves, currents, shallow water and bank suction. The mathematical model is used in port design and inland waterway studies to give the designer an insight into the inherent possibilities and restrictions of vessels, infrastructure and environmental conditions.

In this thesis, we implemented our own procedure which calculates the motion of a boat under water waves. We use the same mathematical principles as the commercial software. We use a simple boat model for fast computation and a reasonable computer graphics visual effect. The commercial software is overkill for our application, because for port design or vessel design, engineering accuracy due to the shape and design of the boat is critical, but it is not essential for our application.

Chapter 4

Oscillatory Motion

It is well-known that motion around an equilibrium is well approximated by harmonic oscillation. In this chapter, we shall discuss harmonic oscillators and, in particular, the forced harmonic oscillator [Fey63, Fre71]. Readers who are familiar with this subject area in physics can skip this chapter.

With the help of complex numbers, we can present the physics behind forced harmonic oscillation cleanly and clearly. We will first briefly explain how we intend to use the complex exponential method. Then we will use it to solve Newton's equation of motion.

A complex number a can be written as $a = a_r + ja_j$, where a_r denotes the real part of a, and a_j denotes the imaginary part of a. Referring to Figure 4.1, we see that we may also write a complex number $a = x + jy$ in the form $x + jy = re^{j\theta}$, where $r^2 = x^2 + y^2 = (x + jy)(x - jy) = aa^*$. (The complex conjugate of a, written a^*, is obtained by reversing the sign of j in a.) We represent a complex number in either of two forms, a real plus an imaginary part, or a magnitude r and a phase angle θ. Given r and θ, then x and y are clearly $r\cos\theta$ and $r\sin\theta$ and, in reverse, given a complex number

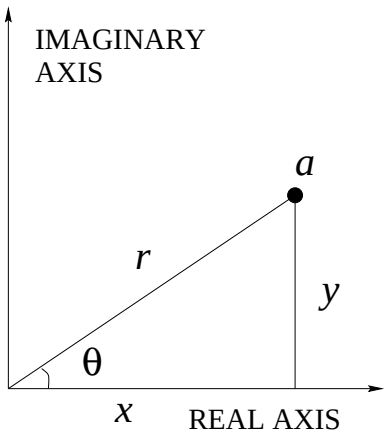

Figure 4.1: A complex number may be represented by a point in the "complex plane."

$x + jy$, then $r = \sqrt{x^2 + y^2}$ and $\tan\theta = y/x$, namely, the ratio of the imaginary to the real part.

We apply complex numbers to our analysis of physical phenomena by the following trick. We have examples of things that oscillate; the oscillation may have a driving force which is a certain constant times $\cos\omega t$. Now such a force, $p = p_0\cos(\omega t)$, can be written as the real part of a complex number $\tilde{p_c} = p_0 e^{j\omega t}$ because $e^{j\omega t} = \cos(\omega t) + j\sin(\omega t)$. We write a little tilde (˜) over the variable, in this case p_c, to remind ourselves that this quantity is a complex number. The reason we do this is that it is easier to work with an exponential function than with a cosine. So the whole trick is to represent our oscillatory functions as the real parts of certain complex functions. The complex number $\tilde{p_c}$ that we have so defined is not a real physical force, because no force in physics is really complex; actual forces have no imaginary part, only a real part. We shall, however, speak of the "force" $p_0 e^{j\omega t}$, but of course the actual force is modelled by the real part of that expression.

Let us take another example. Suppose we want to represent a force as a cosine wave

with a phase delay of Δ. This, of course, would be the real part of $p_0 e^{j(\omega t+\Delta)}$, but exponentials being what they are, we may write $e^{j(\omega t+\Delta)} = e^{j\omega t}e^{j\Delta}$. Thus we see that the algebra of exponentials is much easier than that of sines and cosines; this is the reason we choose to use complex numbers. We shall often write

$$\tilde{p_c} = p_0 e^{j\Delta} e^{j\omega t} = \tilde{p} e^{j\omega t}, \tag{4.1}$$

where $\tilde{p}$ represents a function which depends on the magnitude and phase shift and stands for

$$\tilde{p} = p_0 e^{j\Delta}.$$

Now let us solve an equation using complex numbers to see whether we can work out a problem for some real case. For example, let us try to solve a simplified version of Newton's equation Eq. 2.2 which does not involve the damping term:

$$\frac{d^2x}{dt^2} + \omega_0^2 x = \frac{p}{m} = \frac{p_0}{m}\cos(\omega t), \tag{4.2}$$

where p is the force which drives the oscillator and x is the displacement. Now, let us suppose for mathematical convenience that x and p are complex. That is to say, x can be written as $x_r + jx_j$, and similarly for p. If we had a solution of Eq. 4.2 with complex numbers, and substituted them in the equation, we would get

$$\frac{d^2(x_r + jx_j)}{dt^2} + \omega_0^2 k(x_r + jx_j) = \frac{p_r + jp_j}{m}$$

or

$$\frac{d^2x_r}{dt^2} + \omega_0^2 x_r + j(\frac{d^2x_j}{dt^2} + \omega_0^2 x_j) = \frac{p_r}{m} + \frac{jp_j}{m}.$$

Since two complex numbers are equal only if their real parts are equal and their complex parts are equal, we deduce that the real part of x satisfies the equation with the real part of the force. We must emphasize, however, that this separation into a real part and an imaginary part is not valid in general, but is valid only for equations which are linear, that is, for equations in which x appears in every term only in the first power or the zeroth power. For instance, if there were in the equation a term λx^2, then when we substitute $x_r + ix_j$, we would get $\lambda(x_r + jx_j)^2$, but when separated into real and imaginary parts this would yield $\lambda(x_r^2 - x_j^2)$ as the real part and $2j\lambda x_r x_j$ as the imaginary part. So we see that the real part of the equation would not involve just λx_r^2, but also $-\lambda x_j^2$. In this case we get a different equation than the one we wanted to solve, with x_j, the completely artificial thing we introduced in our analysis, mixed in.

Let us now try our new method for the problem of the forced oscillator, which we already know how to solve. We want to solve Eq. 4.2 as before, but we say that we are going to try to solve

$$\frac{d^2\tilde{x_c}}{dt^2} + \omega_0^2 \tilde{x_c} = \frac{\tilde{p}e^{j\omega t}}{m}, \tag{4.3}$$

where $\tilde{p}e^{j\omega t}$ is a complex number. Of course $\tilde{x_c}$ will also be complex, all the while remembering that the real component is the only relevant term. Now we will discuss how to solve Eq. 4.3 for the forced solution. The forced solution has the same frequency as the applied force, and has some amplitude of oscillation and some phase, and so it can be represented also by some complex number $\tilde{x}$ whose magnitude represents the swing

of x and whose phase represents the time delay in the same way as for the force:

$$\tilde{x_c} = \tilde{x}e^{j\omega t},$$

A wonderful feature of an exponential function is that $d(\tilde{x}e^{j\omega t})/dt = j\omega\tilde{x}e^{j\omega t}$. When we differentiate an exponential function, we bring down the derivative of the exponent as a simple multiplier. The second derivative does the same thing, it brings down another $j\omega$. Thus, it is very simple to write immediately, by inspection, the equation for $\tilde{x}$: every time we see a differentiation, we simply multiply by $j\omega$. Thus Equation 4.3 becomes

$$(j\omega)^2\tilde{x} + \omega_0^2\tilde{x} = \tilde{p}/m. \tag{4.4}$$

We have divided out the common factor $e^{j\omega t}$. Differential equations of this form are immediately converted, by inspection, into mere algebraic equations; we virtually have a solution by sight, that since $(j\omega)^2 = -\omega^2$. Hence,

$$\tilde{x} = \frac{\tilde{p}}{m(\omega_0^2 - \omega^2)}. \tag{4.5}$$

This, of course, is the solution we had before; for since $m(\omega_0^2 - \omega^2)$ is a real number, the phase angles of $\tilde{p}$ and of $\tilde{x}$ are the same (or perhaps 180° apart, if $\omega^2 > \omega_0^2$). As advertised previously, the magnitude of $\tilde{x}$, which measures how far the oscillator oscillates, is related to the size of the $\tilde{p}$ by the factor $1/m(\omega_0^2 - \omega^2)$, and this factor becomes enormous when ω is nearly equal to ω_0. So we get a very strong response when we apply the right frequency ω. This phenomenon is called resonance.

4.1 The Forced Oscillator with Damping

Equation 4.5 tells us that if the frequency ω were exactly equal to ω_0, we would have an infinite response. Actually, no such infinite response occurs because some other factors, like friction, limit the response. Let us therefore add to Eq. 4.2 a friction term to bound the response.

Ordinarily such a problem is very difficult because of the character and complexity of the frictional term. There are, however, many circumstances in which the frictional force is *proportional to the speed* with which the object moves. An example of such friction is the friction for slow motion of an object in oil or a thick liquid. There is no force when it is just standing still, but the faster it moves the faster the oil has to go past the object, and the greater is the resistance. So we shall assume that there is another term which models a resistance force proportional to the velocity, namely: $p_f = -m\gamma dx/dt$, where γ is the damping coefficient. Thus our equation will be Newton's equation of motion:

$$\frac{d^2x}{dt^2} + \gamma\frac{dx}{dt} + \omega_0^2 x = \frac{p}{m}. \tag{4.6}$$

Now we have the equation in the most convenient form to solve. If γ is very small, it represents very little friction; if γ is very large, there is a tremendous amount of friction. How do we solve this new linear differential equation? Suppose that the driving force is equal to $p_0\cos(\omega t + \Delta)$; we could put this into Eq. 4.6 and try to solve it, but we shall instead solve it using complex exponentials. Thus we write p as the real part of $\tilde{p}e^{j\omega t} = p_0 e^{j\Delta}e^{j\omega t}$ and x as the real part of $\tilde{x}e^{j\omega t}$, and substitute these into Eq. 4.6. It is

not even necessary to do the actual substitution, for we can see by inspection that the equation would become

$$[(j\omega)^2\tilde{x} + \gamma(j\omega)\tilde{x} + \omega_0^2\tilde{x}]e^{j\omega t} = (\tilde{p}/m)e^{j\omega t}. \tag{4.7}$$

If we divide by $e^{j\omega t}$ on both sides, then we can obtain the response $\tilde{x}$ to the given force $\tilde{p}$; it is

$$\tilde{x} = \tilde{p}/[m(\omega_0^2 - \omega^2 + j\gamma\omega)]. \tag{4.8}$$

Thus again $\tilde{x}$ is given by $\tilde{p}$ times a certain factor called the frequency response function $\tilde{R}$, namely

$$\tilde{R} = \frac{1}{m(\omega_0^2 - \omega^2 + j\gamma\omega)} \tag{4.9}$$

and

$$\tilde{x} = \tilde{p}\tilde{R}. \tag{4.10}$$

This factor $\tilde{R}$ can be written as

$$\tilde{R} = \rho(\omega)e^{j\theta(\omega)}, \tag{4.11}$$

where $\rho(\omega)$ and $\theta(\omega)$ are the magnitude and the phase of of $\tilde{R}$. Let us see what it means. We know the complex representation of the force is

$$\tilde{p}e^{j\omega t} = p_0 e^{j\Delta}e^{j\omega t},$$

and the actual force p is the real part of this, that is, $p_0\cos(\omega t + \Delta)$. Eq. 4.10 tells us

that $\tilde{x}$ is equal to $\tilde{p}\tilde{R}$, so we can rewrite it as

$$\tilde{x} = \tilde{R}\tilde{p} = \rho(\omega)e^{j\theta(\omega)}p_0 e^{j\Delta} = \rho(\omega)p_0 e^{j(\theta(\omega)+\Delta)}. \tag{4.12}$$

Finally, going even further back, we see that the physical x, which is the real part of the complex $\tilde{x}e^{j\omega t}$, is equal to the real part of $\rho(\omega)p_0 e^{j(\theta(\omega)+\Delta)}e^{j\omega t}$. But $\rho(\omega)$ and p_0 are real, and the real part of $e^{j(\theta(\omega)+\Delta+\omega t)}$ is simply $\cos(\omega t + \Delta + \theta(\omega))$. Thus

$$x = \rho(\omega)p_0 \cos(\omega t + \Delta + \theta(\omega)). \tag{4.13}$$

This tells us that the amplitude of the response is the magnitude of the force p multiplied by a certain magnification factor, $\rho(\omega)$; this gives us the "amount" of oscillation. It also tells us, however, that x is not oscillating in phase with the force, which has the phase Δ, but is shifted by an extra amount $\theta(\omega)$. Therefore $\rho(\omega)$ and $\theta(\omega)$ represent the size of the response and the phase shift of the response.

Now let us work out what $\rho(\omega)$ is. If we have a complex number, the square of the magnitude is equal to the number times its complex conjugate; thus

$$\begin{aligned} \rho^2 &= \frac{1}{m^2(\omega_0^2 - \omega^2 + j\gamma\omega)(\omega_0^2 - \omega^2 - j\gamma\omega)} \\ &= \frac{1}{m^2[(\omega^2 - \omega_0^2)^2 + \gamma^2\omega^2]}. \end{aligned} \tag{4.14}$$

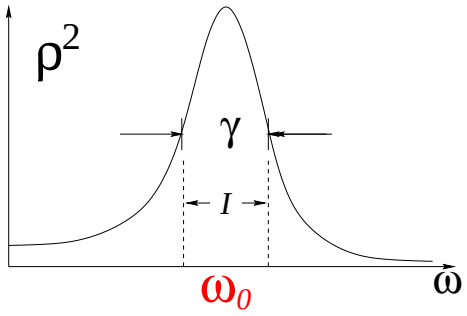

Figure 4.2: Plot of ρ^2 versus ω.

In addition, the phase angle θ is easy to find, for if we write

$$\frac{1}{\tilde{R}} = \frac{1}{\rho(\omega)e^{j\theta(\omega)}} = \frac{1}{\rho(\omega)}e^{-j\theta(\omega)} = m(\omega_0^2 - \omega^2 + j\gamma\omega),$$

we see that

$$\tan\theta = -\frac{\gamma\omega}{\omega_0^2 - \omega^2}. \tag{4.15}$$

The right hand side is negative because $\tan(-\theta) = -\tan\theta$. There are multiple solutions for θ. For a given ω, we choose a solution of θ such that $\theta < 0$ and it corresponds to the displacement x lagging the force p.

Figure 4.2 shows how $\rho^2(\omega)$ varies as a function of frequency. The function ρ^2 is physically more interesting than ρ, because ρ^2 is proportional to the square of the amplitude, which is proportional to the *energy* that is developed in the oscillator by the force. We see that if γ is very small, then $1/(\omega_0^2 - \omega^2)^2$ is the most important term, and the response is large for ω near ω_0 (see Figure 4.2). The phase shift varies as shown in Figure 4.3.

In certain circumstances we get a slightly different expression for the resonance formula in Eq. 4.8. In situations for which γ is very small, the most interesting part of the curve is near $\omega = \omega_0$, and we may replace Eq. 4.8 by an approximate formula which is

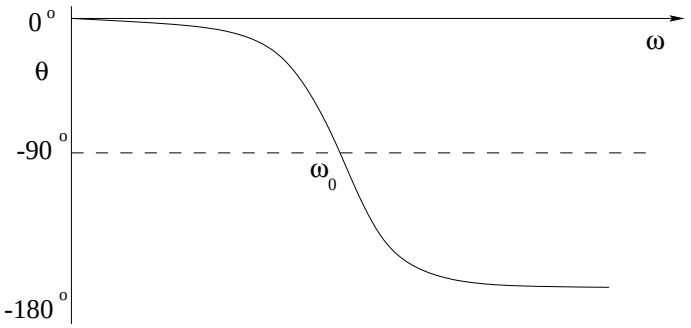

Figure 4.3: Plot of θ versus ω.

very accurate if γ is small and ω is near ω_0. Since $\omega_0^2 - \omega^2 + j\gamma\omega \approx 2\omega_0(\omega_0 - \omega + j\gamma/2)$, so that

$$\tilde{x} \approx \frac{\tilde{p}}{2m\omega_0(\omega_0 - \omega + j\frac{\gamma}{2})} \qquad \text{if } \gamma << \omega_0 \text{ and } \omega \approx \omega_0. \tag{4.16}$$

It is easy to find the corresponding formula for ρ^2. It is

$$\rho^2 \approx \frac{1}{4m^2\omega_0^2[(\omega_0 - \omega)^2 + \frac{\gamma^2}{4}]}. \tag{4.17}$$

In this case, the information about γ is embedded in the curve $\rho^2(\omega)$. The maximum height of the curve $\rho^2(\omega)$ is at $\omega = \omega_0$, and it equals

$$M_0 = \rho^2(\omega_0) = \frac{1}{4m^2\omega_0^2\frac{\gamma^2}{4}}$$

by Eq. 4.17. It turns out that the interval over ω for which $\rho^2(\omega) \geq \frac{1}{2}M_0$ is of great interest. If we ask for the width $\Delta\omega$ of this interval I, we have

$$\begin{aligned} \rho^2(\omega_0 + \frac{\Delta\omega}{2}) &= \frac{1}{2}\rho^2(\omega_0) \\ \frac{1}{4m^2\omega_0^2(\frac{\Delta\omega^2}{4} + \frac{\gamma^2}{4})} &= \frac{1}{2}\frac{1}{4m^2\omega_0^2\frac{\gamma^2}{4}} \end{aligned}$$

$$\begin{aligned} \frac{\Delta\omega^2}{4} + \frac{\gamma^2}{4} &= 2\frac{\gamma^2}{4} \\ \Delta\omega &= \gamma. \end{aligned} \tag{4.18}$$

So the full width of I, namely the length of the interval between which $\rho^2(\omega) \geq \frac{1}{2}M_0$ is $\Delta\omega = \gamma$, supposing that γ is small. The resonance is sharper and sharper as the frictional effects are made smaller and smaller.

4.2 Transient Behavior in Damped Oscillation

We now turn to the discussion of transients. By a *transient* we mean a solution of the differential equation when the force is $f(t) \equiv 0$, but when the system is not simply at rest. Suppose when the oscillation starts it was driven by a force for a while, and then we turn off the force. What happens then? The system stores energy and there is a certain amount of work done to maintain it. Now when we turn off the force, and no more work is being done, the losses consume the energy that the system has stored. Let us suppose that the system moves so nicely, with hardly any force, that if we let go it will oscillate at essentially the same frequency all by itself. So we will guess that ω is the resonant frequency ω_0, and the stored energy will decrease.

Our goal is thus, starting with the following equation,

$$\frac{d^2x}{dt^2} + \gamma\frac{dx}{dt} + \omega_0^2 x = 0, \tag{4.19}$$

solve for x. Let us try as a solution an exponential curve, $x = Ae^{j\gamma t}$. We put this into

Eq. 4.19 with $F(t) = 0$, using the rule that each time we differentiate x with respect to time, we multiply by $j\alpha$. Thus our equation is of the form

$$(-\alpha^2 + j\gamma\alpha + \omega_0^2)Ae^{j\alpha t} = 0. \tag{4.20}$$

The net result must be zero for *all times*, which is impossible unless either $A = 0$, which is the rest state, or

$$-\alpha^2 + j\alpha\gamma + \omega_0^2 = 0. \tag{4.21}$$

If we can solve this and find an α, then we will have a solution in which A need not be zero. We find

$$\alpha = j\gamma/2 \pm \sqrt{\omega_0^2 - \gamma^2/4}. \tag{4.22}$$

For now we shall assume that γ is fairly small relative to ω_0, so that $\omega_0^2 - \gamma^2/4$ is definitely positive, and there is a real valued square root. Then we get *two solutions*:

$$\alpha_1 = j\gamma/2 + \sqrt{\omega_0^2 - \gamma^2/4} = j\gamma/2 + \omega_\gamma, \tag{4.23}$$

and likewise

$$\alpha_2 = j\gamma/2 - \omega_\gamma, \tag{4.24}$$

where $\omega_\gamma = \sqrt{\omega_0^2 - \gamma^2/4}$. Let us consider the first one. Then we know that one solution for x is $x_1 = Ae^{j\alpha_1 t}$, where A is any constant whatever. Thus $j\alpha_1 = -\gamma/2 + j\omega_\gamma$, and we

get $x = Ae^{(-\gamma/2+j\omega_\gamma)t}$, or what is the same,

$$x_1 = Ae^{-\gamma t/2}e^{j\omega_\gamma t}. \tag{4.25}$$

First, we recognize this as an oscillation at a frequency ω_γ, which is not *exactly* the frequency ω_0, but is rather close to ω_0 if the damping coefficient γ is small. Second, the amplitude of the oscillation is decreasing exponentially! If we take the real part of Eq. 4.25, we get

$$x_1 = Ae^{-\gamma t/2}\cos\omega_\gamma t. \tag{4.26}$$

This is like our guess, except that the frequency really is ω_γ.

Now let us consider the other solution, which is α_2 in Eq. 4.24. We see that the difference is only that the sign of ω_γ is reversed:

$$x_2 = Be^{-\gamma t/2}e^{-j\omega_\gamma t}. \tag{4.27}$$

What does this mean? Since Eq. 4.19 is a linear equation, if x_1 and x_2 are each a possible solution of Eq. 4.19 with $f(t) = 0$, then $x_1 + x_2$ is also a solution of the same equation. So the general solution x is of the mathematical form

$$x = e^{-\gamma t/2}(Ae^{j\omega^\gamma t} + Be^{-j\omega^\gamma t}). \tag{4.28}$$

In order for x to be real, $Be^{-j\omega_\gamma t}$ will have to be the complex conjugate of $Ae^{j\omega_\gamma t}$, so that the imaginary parts disappear. So it turns out that B is the complex conjugate of A,

and our real solution is

$$x = e^{-\gamma t/2}(Ae^{j\omega_\gamma t} + A^* e^{-j\omega_\gamma t}). \tag{4.29}$$

We can write the complex number A as $A = \rho_A e^{j\theta_A}$. Then

$$\begin{aligned} x &= e^{-\gamma t/2}(\rho_A e^{j(\omega_\gamma t+\theta_A)} + \rho_A e^{-j(\omega_\gamma t+\theta_A)}) \\ &= 2e^{-\gamma t/2}\rho_A \cos(\omega_\gamma t + \theta_A) \end{aligned}$$

So our real solution is an oscillation with a *phase shift* and a damping.

In this chapter, we introduced the background mathematics and physics for modelling harmonic oscillation. The basic formula will be used again and again in different application in the next few chapters. We will use it to model tree branch swaying motion, as well as the heaving, pitching and rolling motion for boats in regular water waves.

Chapter 5

Modelling Inverse Harmonic Oscillation

There are three stages to the inverse problem of harmonic oscillation, namely, scan, model and synthesize, as shown in Figure 5.1. The scan and synthesize stages are dependent on the specific application. In this chapter, we describe the basic parameter extraction steps which are common to many applications of the inverse problem of harmonic oscillation, namely, steps 2(a) and 2(b) at the modelling stage in Figure 5.1. These two steps translate

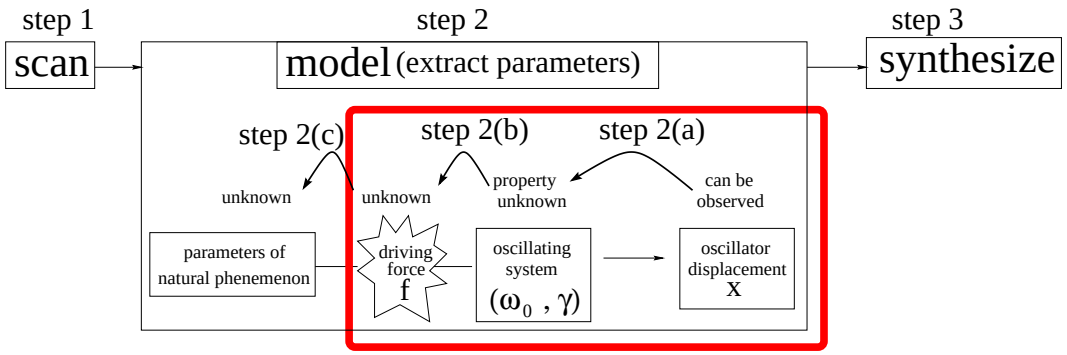

Figure 5.1: Steps 2(a) and 2(b) are the basic parameter extraction subtasks of the inverse problem of harmonic oscillation.

observed displacement to effective force parameters. Step 2(c), which relates the effective force to the natural phenomenon, is again dependent on the specific application. We defer the discussion of step 2(c) until later chapters when we deal with actual applications.

5.1 x and f in Terms of Fourier Series

Before we discuss steps 2(a) and 2(b) in detail, we will briefly describe the representation which we use to express the displacement $x(t)$ and the driving force $f(t)$.

Our goal is to represent x and f in their Fourier series expansion. First, we will talk about the case where x and f are smooth functions. Then, we will apply the theory to our sampled data.

5.1.1 The General Case

We do not know the driving force, $f(t)$, but we can express it as a Fourier transform:

$$f(t) = f_0 + \sum_{i=1}^{\infty} f_i \cos(i\omega_1 t + \Delta_i),$$

where $\omega_1 = 2\pi/(NT)$ is the fundamental frequency, NT is the duration of $f(t)$, i.e., $f(t)$ is defined for $t \in [0, NT)$, i is an index, the f_i's are the amplitude coefficients and the Δ_i's are the phase delays.

We know the impact of each part of this force. Recall that, in chapter 4, we discussed an example oscillating system where the driving force is a simple cosine curve $p(t) =$

$f_d \cos(\omega t + \Delta)$. From that example, we know for the driving force

$$p_i(t) = f_i \cos(i\omega_1 t + \Delta_i) = \text{ real part of } \tilde{f}_i e^{ji\omega_1 t},$$

where $\tilde{f}_i = f_i e^{j\Delta_i}$. In complex exponential form, the resulting displacement is $\tilde{x}_i e^{ji\omega_1 t}$, where

$$\tilde{x}_i = \tilde{R}\tilde{f}_i.$$

Since Newton's equation of motion Eq. 4.6 is a linear equation, the displacement caused by the driving force

$$f(t) = \sum_{i=0}^{\infty} p_i(t) = f_0 + \sum_{i=1}^{\infty} f_i \cos(i\omega_1 t + \Delta_i),$$

is just the sum of the real parts of the individual displacements $\tilde{x}_i e^{ji\omega_1 t}$,

$$\begin{aligned} x(t) &= \sum_{i=0}^{\infty} \text{ real part of } \tilde{x}_i e^{ji\omega_1 t} \\ &= \sum_{i=0}^{\infty} \text{ real part of } \tilde{R}\tilde{f}_i e^{ji\omega_1 t} \end{aligned}$$

From Eq. 4.12, we know that the complex number $\tilde{R} = \rho e^{j\theta}$ serves the purpose of amplifying the magnitude of $\tilde{f}_i$ by ρ and shifting its phase by θ. So we can rewrite $x(t)$ as

$$\begin{aligned} x(t) &= \rho^2(0) f_0 + \sum_{i=1}^{\infty} \rho(i\omega_1) f_i \cos(i\omega_1 t + \Delta_i + \theta_i) \\ &= x_0 + \sum_{i=1}^{\infty} x_i \cos(i\omega_i t + \phi_i), \end{aligned} \tag{5.1}$$

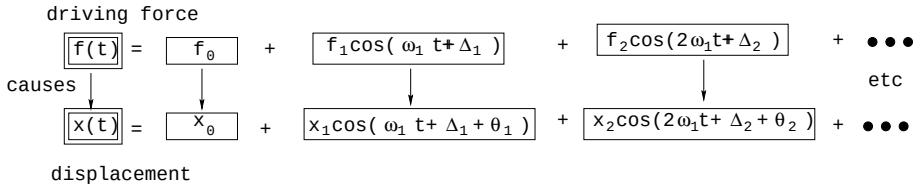

Figure 5.2: At each frequency, the driving force at that frequency determines the displacement at that frequency.

where

$$x_i = \rho(i\omega_1) f_i \tag{5.2}$$

and

$$\phi_i = \Delta_i + \theta_i. \tag{5.3}$$

The relationship between the Fourier series $f(t)$ and $x(t)$ is as shown in Figure 5.2.

5.1.2 Applying the Theoretical Analysis to Sampled Data

Now we need to apply the theoretical analysis above to our observed data. This time, instead of going from the force to the displacement, we reverse the process to use the observed displacement to predict the driving force.

First, we need to find the Fourier series expansion for our observed data. In our experiment, we sampled the displacement function $x(t)$ at N sample points,

$$t = t_0 + \tau T,$$

where $\tau = 0, 1, \ldots, N-1$ is the sample point index, T is the sampling period. This gives us a discrete representation of $x(t)$, namely $\hat{x}(\tau)$, The Fourier series expansion of $\hat{x}(\tau)$

has only $M+1$ terms,

$$\hat{x}(\tau) = \hat{x}_0 + \sum_{\nu=1}^{M} \hat{x}_\nu \cos(2\pi(\nu/N)\tau + \hat{\phi}_\nu), \tag{5.4}$$

where M is roughly half of N

$$M = \left\lfloor \frac{N}{2} \right\rfloor,$$

ν is a measure for the number of cycles over the N samples, and the fundamental frequency corresponds to $\nu = 1$, i.e., one cycle per N samples. The fundamental angular frequency is

$$\omega_1 = 2\pi/(NT).$$

Therefore, we expect our estimated force $\hat{f}(\tau)$ to have only M terms as well, such as

$$\hat{f}(\tau) = \hat{f}_0 + \sum_{\nu=1}^{M} \hat{f}_\nu \cos(2\pi(\nu/N)\tau + \hat{\Delta}_\nu), \tag{5.5}$$

where

$$\hat{x}_\nu = \hat{\rho}(\nu)\hat{f}_\nu \tag{5.6}$$

and

$$\hat{\phi}_\nu = \hat{\Delta}_\nu + \hat{\theta}_\nu. \tag{5.7}$$

Here, $\hat{\rho}(\nu)$ and $\hat{\theta}_\nu$ are sampled from $\rho(\omega)$ and $\theta(\omega)$ at $\omega = 2\pi\nu/(NT)$.

Next, we will discuss the driving force parameter extraction steps 2(a) and 2(b).

5.2 Example Input Displacement x

In addition to the general explanation of the process of inferring a driving force from the observed displacement, we will use an example to demonstrate how information can be estimated and analyzed at each step. The data is obtained by observing the top of a tree swaying in the wind (Figure 6.16). We will discuss this in more detail in chapter 6. For the time being, this data set will be used as an example to demonstrate the basic data analysis process. Figure 5.3 is the example observed oscillator displacement $\hat{x}(\tau)$ plotted against sample point index τ, where $\tau = 0, 1, \ldots, N-1$, the total number of samples is $N = 1047$, and the sampling period is $T = \frac{1}{6}$ second. This data is based on the observed displacement of an oscillator due to natural wind force.

5.3 Obtaining Properties of the Oscillating System

In order to extract the driving force of a harmonic oscillation by observing its displacement, we need to learn the properties of the system, such as its natural frequency ω_0 and its damping coefficient γ. The harmonic oscillation is a resonance phenomenon, in which case, information about ω_0 and γ are embedded in the displacement data $\hat{x}$. To extract this information, we can study $\hat{x}$'s behavior in frequency domain.

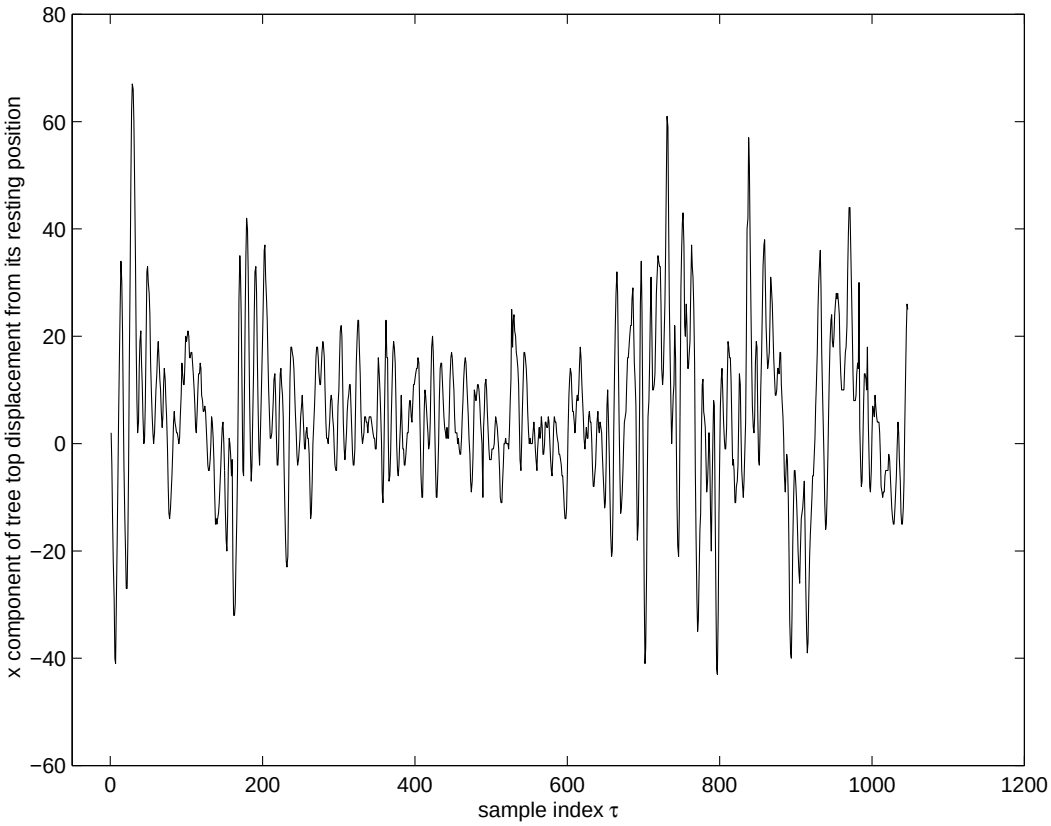

Figure 5.3: An example plot of the oscillator displacement x component of the oscillator's movement plotted against sample index τ.

5.3.1 Transforming $\hat{x}(\tau)$ to Frequency Domain

We transform $\hat{x}(\tau)$ to the frequency domain using the discrete Fourier transform (DFT):

$$\hat{X}(\nu) = \frac{1}{N} \sum_{\tau=0}^{N-1} \hat{x}(\tau) e^{-j2\pi(\nu/N)\tau}.$$

where the quantity ν/N is analogous to frequency measured in cycles per sampling interval.

Taking our example displacement data $\hat{x}$ in Figure 5.3, we apply the DFT to it to obtain its frequency domain response. The amplitude of the DFT, $|\hat{X}(\nu)|$ is shown in Figure 5.4.

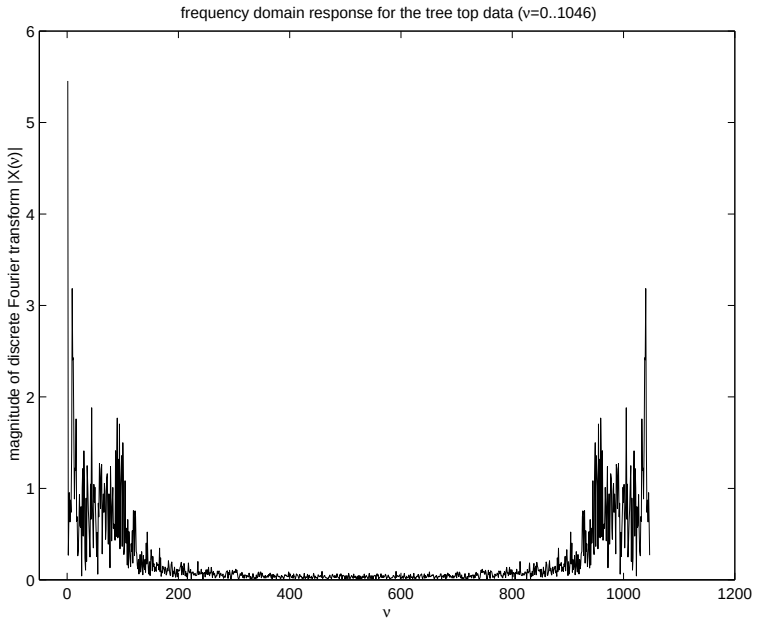

Figure 5.4: Frequency domain analysis of the oscillator's displacement data for $\nu = 0 \ldots 1046$.

As explained in Appendix A, in Figure 5.4, for $\nu = 1 \ldots N-1$, $|\hat{X}(\nu)|$ is symmetric. Also, since the highest frequency needed corresponds to $\nu = \lfloor N/2 \rfloor$, the second half of Figure 5.4 contains redundant information. We thus only need to analyze the first half of it, as shown in Figure 5.5

5.3.2 The Natural Frequency and Other Phenomena

Every structure has what is called a natural frequency, or rate at which it tends to vibrate. This is determined by its height, shape, material and other details [Fey63]. An oscillator is like a type of signal filter which amplifies the signal at around the neighbourhood of its natural frequency and it tends to "annihilate" the signal at other frequencies. This is the so-called resonance phenomenon. We can estimate the natural frequency of such an oscillator by analyzing its response in the frequency domain and look for peaks in the

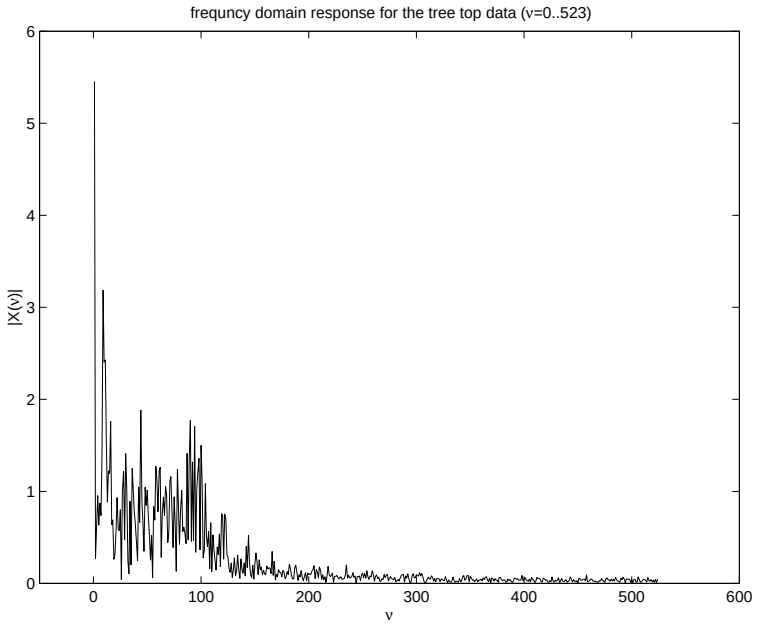

Figure 5.5: Frequency domain analysis of the oscillator's spatial domain data for $\nu = 0 \ldots 523$.

plot.

In our example, there are several significant peaks in the frequency domain analysis in Figure 5.5. For the time being, we intend to have the user observing the pattern in the plot. No method is provided to analyze the peaks in the graph automatically by the computer. Identification of the peak in the graph relies on the user's intuition. This is not a difficult task. The peaks are normally quite prominent.

- There is a dominating spike at $\nu = 0$ in Figure 5.5, which corresponds to angular frequency $\omega = 0$. This spike shows the average displacement of the oscillator.

- There is a peak at $\nu = 8$ in Figure 5.5. This corresponds to the angular frequency $\omega = 2\pi\frac{\nu}{NT}$, where

$$\frac{\nu}{NT} = \frac{8}{1047 * (1/6)s} = \frac{48}{1047}s^{-1} \approx \frac{1}{22s}.$$

The object from which we acquired this data was oscillating due to the wind force exerted on it, and this frequency is roughly related to the large scale wind movement. Once every 22 seconds or so, a gust of wind starts, blows stronger and stronger, then slowly dies down.

- There is a peak at $\nu = 89$ in Figure 5.5. This corresponds to angular frequency $\omega = 2\pi\frac{\nu}{NT}$, where

$$\frac{\nu}{NT} = \frac{89}{1047 * (1/6)s} \approx \frac{1}{2s}.$$

When we watch the object oscillate, it tends to move left and right once every 2 seconds or so. This clearly corresponds to the natural frequency of the observed object, which is

$$\omega_0 = 2\pi\frac{1}{2s} = \pi. \tag{5.8}$$

5.3.3 Damping Coefficient

Another interesting thing about a resonance phenomenon is that sometimes it is possible to infer the damping property of the system by observing the frequency domain response $X(\omega)$ of the displacement $x(t)$ near the oscillator's natural frequency ω_0 [Fey63]. Here, we will briefly explain how this is done.

First, we take a close look at the behavior of $\hat{x}(\tau)$ near the natural frequency of the oscillating system, and check to see what it depends on. This information will help us to extract the damping coefficient. Since the driving force is like an input signal, the oscillator is like a filter and the displacement is like an output. The relationship between the frequency domain response of the input and the output will give us insights about

the filter.

We have written down the Fourier series expansion for $\hat{x}(\tau)$ in Eq. 5.4 as

$$\hat{x}(\tau) = \hat{x}_0 + \sum_{\nu=1}^{M} \hat{x}_\nu \cos(2\pi(\nu/N)\tau + \hat{\phi}_\nu).$$

In the previous section, we used the DFT to transform $\hat{x}(\tau)$ to the frequency domain to study its behavior at near the natural frequency of the oscillator,

$$\hat{X}(\nu) = \frac{1}{N} \sum_{\tau=0}^{N-1} \hat{x}(\tau) e^{-j2\pi(\nu/N)\tau}.$$

The coefficients in function $\hat{x}(\tau)$ are related to $\hat{X}(\nu)$ in the following way:

$$\begin{aligned}
\hat{x}_0 &= \hat{X}(0), \\
\hat{x}_\nu &= 2|\hat{X}(\nu)|, \text{ for } \nu = 1 \,..\, M-1, \\
\hat{x}_M &= \begin{cases} 2|\hat{X}(M)|, & \text{when } N \text{ is odd} \\ |\hat{X}(M)|, & \text{when } N \text{ is even} \end{cases} \\
\hat{\phi}_\nu &= \text{phase angle of } \hat{X}(\nu), \text{ for } \nu = 1 \,..\, M.
\end{aligned} \tag{5.9}$$

We analyze the unknown $\hat{f}(\tau)$ function in a similar way. Recall that we had predicted that the Fourier series expansion of $\hat{f}(\tau)$ is

$$\hat{f}(\tau) = \hat{f}_0 + \sum_{\nu=1}^{M-1} \hat{f}_\nu \cos(2\pi(\nu/N)\tau + \hat{\Delta}_\nu).$$

Its DFT shows the driving force's frequency domain response,

$$\hat{F}(\nu) = \frac{1}{N} \sum_{\tau=0}^{N-1} \hat{f}(\tau) e^{-j2\pi(\nu/N)\tau}.$$

Similarly, the coefficients in function $\hat{f}(\tau)$ are related to $\hat{F}(\nu)$ in the following way:

$$\begin{aligned}
\hat{f}_0 &= \hat{F}(0), \\
\hat{f}_\nu &= 2|\hat{F}(\nu)|, \text{ for } \nu = 1 \,..\, M-1, \\
\hat{f}_M &= \begin{cases} 2|\hat{F}(M)|, & \text{when } N \text{ is odd} \\ |\hat{F}(M)|, & \text{when } N \text{ is even} \end{cases} \\
\hat{\Delta}_\nu &= \text{phase angle of } \hat{F}(\nu).
\end{aligned} \tag{5.10}$$

In Section 5.1, Eq. 5.6 expresses the relationship between the amplitude coefficients of $\hat{x}(\tau)$ and $\hat{f}(\tau)$

$$\hat{x}_\nu = \hat{\rho}(\nu)\hat{f}_\nu. \tag{5.11}$$

Using Equations 5.9 and 5.10, we can rewrite the equation above as

$$|\hat{X}_\nu| = \hat{\rho}(\nu)|\hat{F}_\nu|, \text{ for } \nu = 0 \,..\, M. \tag{5.12}$$

Thus, in terms of magnitude, the frequency domain spectrum of the displacement and that of the driving force are related by the amplification factor $\hat{\rho}(\nu)$.

For the example, from our analysis in Section 5.3.2, we notice that the dominant frequencies of the wind force is near $\nu = 8$, whereas the natural frequency of the oscillator

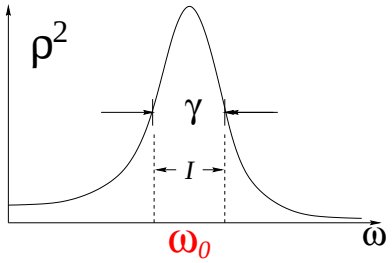

Figure 5.6: Plot of ρ^2 versus ω.

is near $\nu_0 = 89$. In the plot of $|X(\nu)|$ in Figure 5.4, the two peaks at those two values are clearly separated. Since the dominating wind force frequency is much lower, we assume that the frequency spectrum of the wind has no significant features near the natural frequency of the oscillator.

Therefore, the peak in $|\hat{X}(\nu)|$ near the oscillator's natural frequency is mainly due to the resonance effect, which is mathematically due to the the amplification factor $\hat{\rho}(\nu)$. In other words, near the tree's natural frequency, $|\hat{X}(\omega)|$ is roughly proportional to $\hat{\rho}(\nu)$.

Recall that in Section 4.1 we talked about the relationship between $\rho^2(\omega)$ and the damping coefficient γ. As shown in Figure 5.6, clearly, the magnification factor is much larger around the neighborhood of the oscillator's natural frequency. As we explained in Section 4.1, the information about γ is embedded in the curve $\rho^2(\omega)$. Let M_0 be the maximum height of the curve $\rho^2(\omega)$ at $\omega = \omega_0$. The length of the interval between which $\rho^2(\omega) \geq \frac{1}{2}M_0$ is $\Delta\omega = \gamma$, supposing that γ is small as shown in Figure 5.6 [Fey63].

For our example, since we expect $|\hat{X}(\omega)|$ to be roughly proportional to $\hat{\rho}(\nu)$ near the oscillator's natural frequency, instead of plotting ρ^2, we can examine $|\hat{X}(\nu)|^2$, as shown in Figure 5.7. The width at half the maximum height is $\Delta\nu = 14$. The damping coefficient

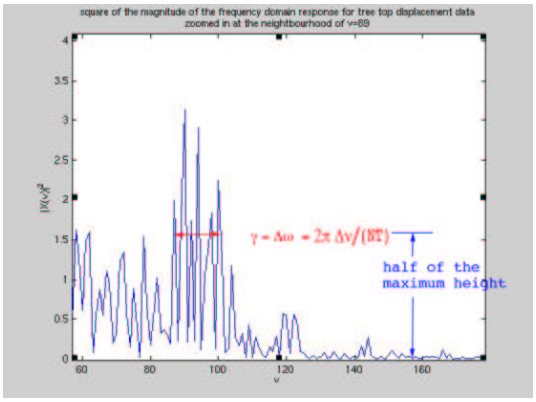

Figure 5.7: Estimate γ from $|\hat{X}(\nu)|^2$.

is thus estimated to be

$$\gamma = \frac{2\pi\Delta\nu}{NT} = \frac{2\pi * 14}{1047 * \frac{1}{6}s} \approx 0.504s^{-1}.$$

5.3.4 Step 2(b): Going from Displacement to Force

Knowing the natural frequency ω_0 and the damping coefficient γ, we can use the equation of harmonic oscillation, Eq. 5.13, to drive the force per unit mass $\hat{f}/m$ from the observed displacement data $\hat{x}$ and its frequency domain transformation $\hat{X}(\nu)$:

$$\frac{d^2x}{dt^2} + \gamma\frac{dx}{dt} + \omega_0^2 x = \frac{f}{m}. \tag{5.13}$$

For our purposes, knowing the fluctuation of the force up to a constant scaling factor $1/m$ is good enough. We do not need to know the absolute value of the force. This is because we intend to apply the wind to virtual objects, for instance a virtual plant. For computer graphics purposes, the virtual object does not have a real mass. So long as the ratio of mass between the virtual object and the observed real object is estimated to be

Table 5.1: Components of the Fourier series of $f(t)$ and $x(t)$.

	force (unknown)	effect of oscillator (due to ω_0 and γ)	displacement (can be calculated)
amplitude	$\hat{f}_\nu$	$\hat{\rho}_\nu$	$\hat{x}_\nu$
phase	$\hat{\Delta}_\nu$	$\hat{\theta}_\nu$	$\hat{\phi}_\nu$

a reasonable value, the absolute value of the mass of the observed object does not have to be known.

The Fourier series expansion of $\hat{x}$ is

$$\hat{x}(\tau) = \hat{x}_0 + \sum_{\nu=1}^{M} \hat{x}_\nu \cos(2\pi(\nu/N)\tau + \hat{\phi}_\nu)$$

and the Fourier series expansion of the unknown wind force $\hat{f}$ is

$$\hat{f}(\tau) = \hat{f}_0 + \sum_{\nu=1}^{M} \hat{f}_\nu \cos(2\pi(\nu/N)\tau + \hat{\Delta}_\nu).$$

The relationship between the Fourier series $f(t)$ and $x(t)$ is as shown in Figure 5.2. The relevant components are listed in Table 5.1.

Here, $\hat{\rho}_\nu$ and $\hat{\theta}_\nu$ are the sampled functions of the force-to-displacement magnification factor ρ and the additional phase shift θ due to the oscillating system. We want to derive the relationship between force per unit mass $\hat{f}/m$ and displacement $\hat{x}$. For $\hat{x}$, the coefficients and phase shifts can be obtained from $\hat{X}$, the frequency domain response of $\hat{x}$ (Eq. 5.9). Hence, to obtain the coefficients and phase delays of the driving force per

unit mass $\hat{f}/m$, we only need to re-arrange Equations 5.6 and 5.7, yielding:

$$\begin{aligned}
\frac{\hat{f}_0}{m} &= \frac{1}{m}\frac{\hat{x}_0}{\hat{\rho}(0)} = \frac{1}{m}\frac{|\hat{X}(0)|}{\hat{\rho}(0)}, \\
\frac{\hat{f}_\nu}{m} &= \frac{1}{m}\frac{\hat{x}_\nu}{\hat{\rho}(\nu)} = \frac{1}{m}\frac{2|\hat{X}(\nu)|}{\hat{\rho}(\nu)}, \\
\hat{\Delta}_\nu &= \hat{\phi}_\nu - \hat{\theta}_\nu = (\text{ phase angle of } \hat{X}(\nu)) - \hat{\theta}(\nu), \text{ for } \nu = 1 \,..\, M-1.
\end{aligned}$$

Note that $m\hat{\rho}(\nu)$ is a function which depends on only the natural frequency and the damping coefficient of the oscillator.

5.3.5 Calculation Using Example Data

In this section, we want to calculate the driving force using the method above (Equation 5.14). First, we will show the estimated phase shift of the driving force at each frequency. Then we will show its estimated magnitude at each frequency.

5.3.5.1 Phase Shifts of Driving Force

For the phase shift calculation, the observed resulting phase shift $\hat{\phi}$ in displacement $\hat{x}$ at each frequency is plotted in Figure 5.8. The phase shift $\hat{\theta}$ due to the effect of the oscillator is plotted in Figure 5.9. Using $\hat{\phi}$ and $\hat{\theta}$, we estimated the initial phase delay $\hat{\Delta}$ in the wind force as shown in Figure 5.10. The phase delay in the wind force looks random, which is what we expected.

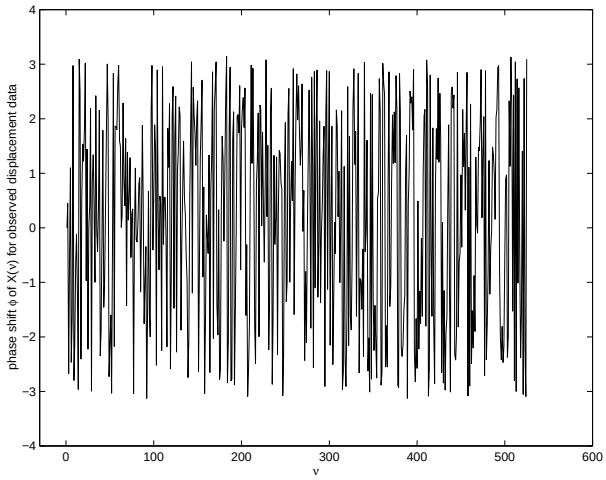

Figure 5.8: Phase shift $\hat{\phi}(\nu)$ for the observed displacement data's Fourier series representation.

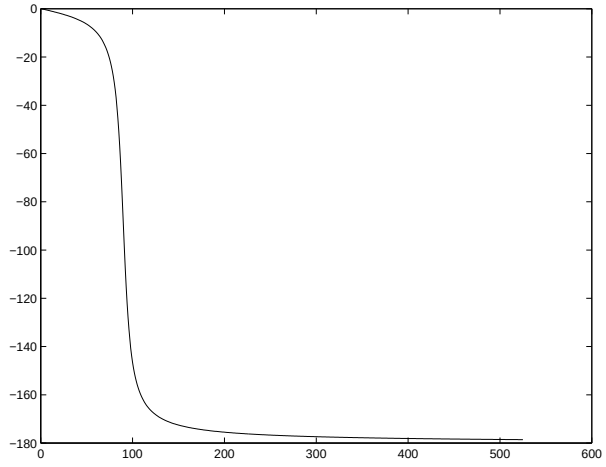

Figure 5.9: Phase shift $\hat{\theta}(\nu)$ due to the oscillator.

5.3.5.2 Amplitudes of Driving Force

For our example data, the amplitude of the DFT of the driving force at each frequency is calculated using Eq. 5.14. In Figure 5.11, we plot the estimated amplitude of the DFT of the driving force against ν.

We would like to be able to say something about the reliability of our estimate. In order to do that, we need to address an important issue, namely, the noise in our input displacement data. Since we are dealing with data obtained from the real world, the data

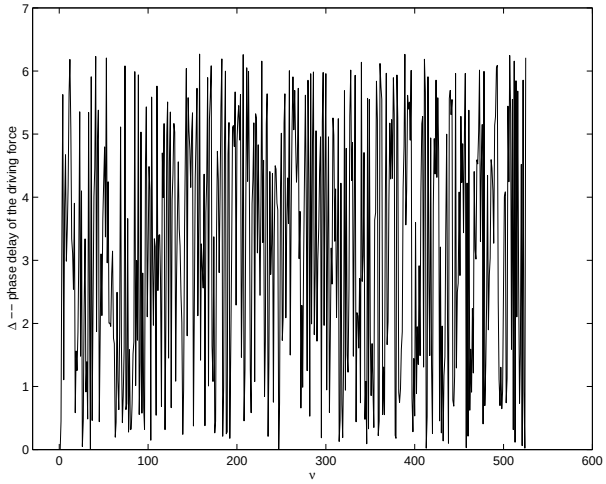

Figure 5.10: Estimated phase delay of the driving force.

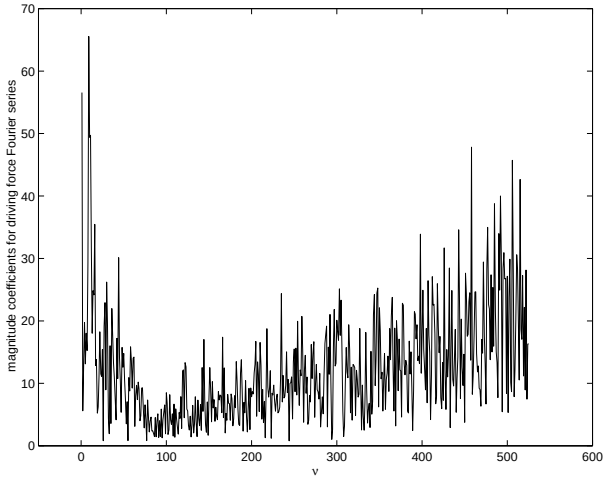

Figure 5.11: Estimated amplitude of the driving force in frequency domain.

will be noisy. Such noise might contaminate the data at certain frequency range so much that the noise would obscure the signal. We should trust the estimated data only in the frequency range where the signal component is dominates the noise component. In the next section, we will talk about how the theoretical white noise analysis is done for our specific problem, namely, extracting the driving force parameter from observed harmonic oscillation. Then, we will demonstrate how to apply this white noise analysis theory to our example data. This will tell us in which frequency range the extracted parameters

contain usable information.

5.4 White Noise Analysis

Since it is likely that our tracking result is not 100% accurate, we shall model the tracking error using Gaussian white noise. Let $x_n(t)$ be the white noise in the displacement data $x(t)$. To characterize white noise's behavior in frequency domain, we need to look at its power density spectrum [Pee87].

5.4.1 White Noise Power Density Spectrum

In this section, we will briefly derive the discrete representation of the expected power density spectrum of white noise given N sampling points in the spatial domain [Pee87].

5.4.1.1 Energy

Let $x_n(t)$ be a Gaussian noise function, where the random variable is drawn from a Gaussian distribution with standard deviation σ_{noise}. Note that here we use the subscript n in $x_n(t)$ to mean noise, so that the reader will not confuse the noise function $x_n(t)$ with the displacement function $x(t)$. The *energy* contained in $x_n(t)$ in the interval $t \in [t_0, t_1]$ is

$$Energy(t_1 - t_0) = \int_{t_0}^{t_1} x_n^2(t)dt.$$

In our applications, we are more likely to deal with the discrete presentation of signal

and noise. Assume that we sampled function $x_n(t)$ at N points,

$$t = t_0 + \tau T.$$

We obtained the sampled function $\hat{x}_n(\tau)$, where t_0 is the starting point of the sampling process, τ is the sampling index, $\tau = 0 \; . \; . \; N-1$, and $T = \frac{t_1 - t_0}{N}$ is the sampling period. The discrete Fourier transform of $\hat{x}_n(\tau)$ is

$$\hat{X_n}(\nu) = N^{-1} \sum_{\tau=0}^{N-1} \hat{x}_n(\tau) e^{-j2\pi(\nu/N)\tau},$$

where the quantity ν/N is analogous to frequency measured in cycles per sampling interval. The discrete representation of the *energy* contained in $\hat{x}_n(\tau)$ is

$$Energy(\tau N) = \frac{t_1 - t_0}{N} \sum_{\tau=0}^{N-1} \hat{x}_n^2(\tau). \tag{5.14}$$

5.4.1.2 Power and Power Density Spectrum

By dividing the energy by the time duration, we obtain the *power* of the white noise function

$$P(\tau N) = \frac{Energy(\tau N)}{t_1 - t_0} = \frac{1}{N} \sum_{\tau=0}^{N-1} \hat{x}_n^2(\tau).$$

By Parseval's theorem [Pee87], we can relate the power of $\hat{x}_n(\tau)$ to its discrete Fourier transform [PM92]:

$$P(\tau N) = \frac{1}{N} \sum_{\tau=0}^{N-1} \hat{x}_n^2(\tau) = \sum_{\nu=0}^{N-1} |\hat{X}_n(\nu)|^2.$$

The sequence $|\hat{X}_n(\nu)|^2$ for $\nu = 0, 1, \ldots, N-1$ is the distribution of power as a function of frequency and is called the *power density spectrum* (PDS).

The variable $\hat{x}_n(\tau)$ is a random variable for each τ, and $P(\tau N)$ is actually a random variable with respect to the random process which generated $\hat{x}_n(\tau)$ [Pee87]. By taking its expected value, we can obtain an average power P_{white_noise} for the random process:

$$P_{white\_noise}(\tau N) = E[P(\tau N)] = E\left[\frac{1}{N}\sum_{\tau=0}^{N-1}\hat{x}_n^2(\tau)\right] = E\left[\sum_{\nu=0}^{N-1}|\hat{X}_n(\nu)|^2\right]. \tag{5.15}$$

Since $\hat{x}_n(\tau)$ is independently sampled Gaussian white noise for each τ, the expected value is

$$E\left[\frac{1}{N}\sum_{\tau=0}^{N-1}\hat{x}_n^2(\tau)\right] = \sigma_{noise}^2,$$

where σ_{noise}^2 is the variance of the Gaussian. We can rewrite Equation 5.15 as

$$P_{white\_noise}(\tau N) = \sigma_{noise}^2 = \sum_{\nu=0}^{N-1} E[|\hat{X}_n(\nu)|^2]. \tag{5.16}$$

5.4.1.3 Band-Limited White Noise

For our applications, we are only dealing with band-limited white noise. This white noise has a nonzero and constant power density spectrum (PDS) over a finite frequency band and zero everywhere else, as shown in Figure 5.12. Here, between $\nu = 0 \ldots N-1$, the white noise power density spectrum's expected value is constant

$$E[|\hat{X}_n(\nu)|^2] = \text{constant} = S_{white\ noise}, \tag{5.17}$$

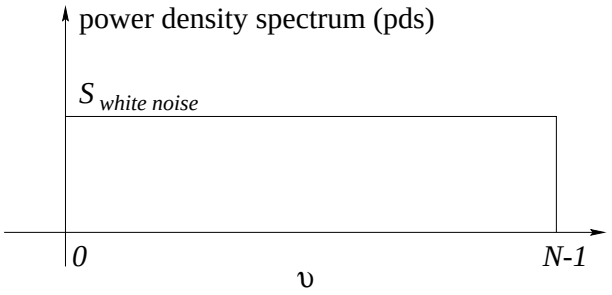

Figure 5.12: Band-limited white noise power density spectrum.

and outside of this range the power density spectrum is zero. Substituting Eq. 5.17 into Eq. 5.16, we have:

$$P_{\text{white noise}}(\tau N) = \sigma^2_{noise} = \sum_{\nu=0}^{N-1} S_{white\ noise} = N S_{white\ noise} \tag{5.18}$$

Re-arranging this equation, we see that the power density spectrum is constant for band-limited white noise:

$$S_{white\ noise} = \frac{\sigma^2_{noise}}{N}, \tag{5.19}$$

for the discrete representation with N samples.

5.4.2 Noise Power Density Spectrum in Force

Given the white noise in the displacement measurement, we would like to know how much noise is introduced into the estimated driving force parameter.

When the displacement is used to extract parameters of the driving force, the white noise $\hat{x}_n(\tau)$ in the displacement introduces noise $\hat{f}_n(\tau)$ into the driving force. Let $\hat{F}_n(\nu)$ be the discrete Fourier transform of $\hat{f}_n(\tau)$. The noise $\hat{f}_n(\tau)$ in the force is related to $\hat{x}_n(\tau)$ in the displacement in the same way as how the force and the displacement are

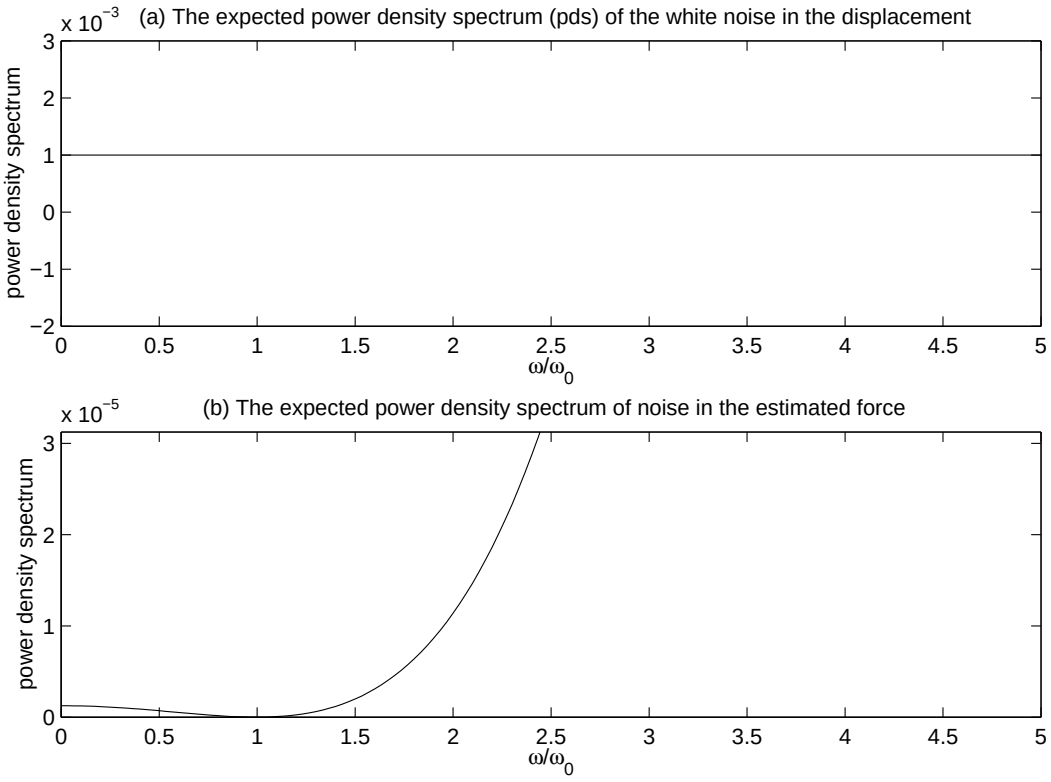

Figure 5.13: (a) The theoretically expected power density spectrum for white noise in displacement; (b) The expected power density spectrum for noise in the estimated force parameter, when $\gamma/\omega_0 = 1/10$.

related. In particular, the amplitudes of their discrete Fourier transforms are related in the following way:

$$|\hat{F}_n(\nu)| \;=\; \frac{|\hat{X}_n(\nu)|}{\hat{\rho}},$$

where $\hat{\rho}$ is a discrete representation of the amplitude magnification function as shown in Figure 5.6. The power density spectrum $|\hat{F}_n(\nu)|^2$ of the noise in the estimated force has an expected value, by Equations 5.17 and 5.19, of:

$$E[|\hat{F}_n(\nu)|^2] \;=\; \frac{E[|\hat{X}_n(\nu)|^2]}{\hat{\rho}^2} \;=\; \frac{S_{white\ noise}}{\hat{\rho}^2} \;=\; \frac{\sigma^2_{noise}}{N}\frac{1}{\hat{\rho}^2}. \qquad (5.20)$$

Figure 5.13(a) shows an example of a white noise power density spectrum where the standard deviation of the white noise is $\sigma_{noise} = 1$ and the number of samples is $N = 1000$. Figure 5.13(b) shows the expected power density spectrum $E[|\hat{F}_n(\nu)|^2]$ of the noise in the estimated driving force. Clearly, near the natural frequency ω_0 of the oscillating system, the amplitude of the noise is suppressed. The expected amplitude of the noise becomes larger as we move away from the neighbourhood of the natural frequency. It is up to the analyst to decide what is an acceptable frequency range where the noise in the estimated force is tolerable. For instance, some user might think a signal to noise power spectrum ratio of more than $10 : 1$ is good enough, whereas someone else might think ratio ought to be higher than $20 : 1$ or $30 : 1$, etc. Wiener filtering can be used to remove noise from the estimated parameters. Since effective use of Wiener filter requires the user to choose appropriate window size for the filter, user intervention would still be necessary.

5.4.3 Application to a Real Example

To demonstrate how this analysis can be used in our applications, we will apply it to our example displacement data in Figure 5.3, which we studied earlier.

We shall assume that the only source of white noise is the discretization error in sampling the video input signal. Each video frame has a finite resolution. When we are extracting features from the video, the discretization causes inaccuracy in our measurement and estimation. The inaccuracy in the observed displacement, which is due to tracking error, is roughly 1 pixel, so we let the standard deviation of the white noise be $\sigma_{noise} = 1$. After extracting the driving force parameter from the displacement informa-

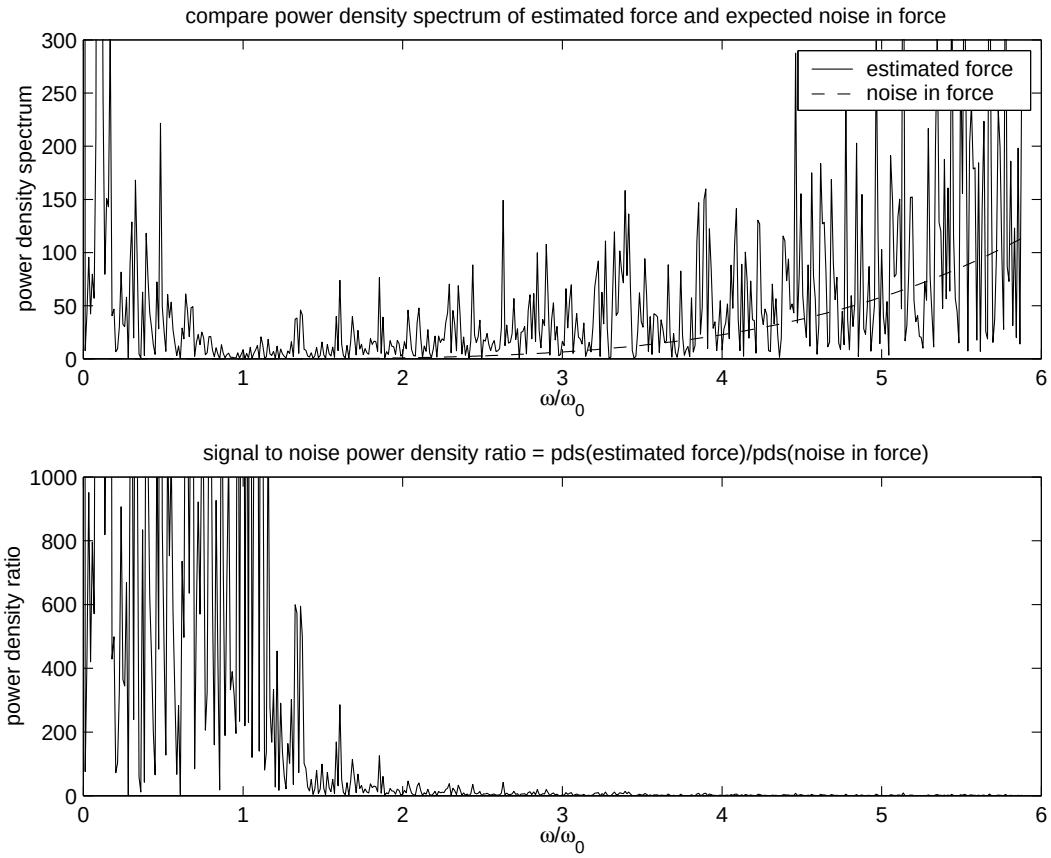

Figure 5.14: Noise in estimated force parameter: (a) compare power density spectrum of estimated force parameter and expected power density spectrum; (b) power density ratio between estimated force parameter and noise in the force.

tion, we plot the power density spectrum of the estimated force and compare it to the expected power density spectrum of the noise in the force, as shown in Figure 5.14(a). By inspection, at high frequencies, the power density spectrum (PDS) of the estimated force oscillates about the expected PDS of the assumed noise. So we expect the higher frequency components to be largely contaminated by noise.

To get a more intuitive idea about how much signal and noise are at each frequency, we investigate the power density spectrum ratio between the estimated force parameter

and the expected noise at each frequency:

$$\begin{aligned} \text{power density spectrum ratio} &= \frac{\text{PDS of estimated force parameter at } \nu}{\text{expected PDS of noise in force}} \\ &= \frac{|\hat{F}(\nu)|^2}{E[|\hat{F}_n(\nu)|^2]}, \end{aligned} \tag{5.21}$$

where the expected PDS $E[|\hat{F}_n(\nu)|^2]$ of noise in the force is determined by Equation 5.20. The result is plotted in Figure 5.14. Clearly, at around the neighbourhood of the natural frequency, the recovered signal (i.e., the estimated force parameter) dominates the power density spectrum, because the signal to noise power density ratio is very high. So we can trust the parameter over this range. A small part of the low frequency and the high frequency components have low signal to noise power density ratio, so the noise is likely to obscure the signal. Once again, a person can look at the graph and use some intuition to decide on a frequency range near the natural frequency where the accuracy of the estimated parameter is considered to be acceptable.

For our example, the signal to noise power density ratio is relatively large for $\omega/\omega_0 = 0 \,..\, 1.15$, which corresponds to $\nu = 0 \,..\, 102$. We are willing to accept the estimated driving force in this frequency range as being usable.

For our example data, based on the information we have analyzed and calculated, we can reconstruct the large scale fluctuation of the driving force. We can check our calculation using our intuition, which we now discuss.

5.5 Driving Force Reconstruction

Since we have estimated the driving force's magnitude $\hat{f}_\nu$ and phase delay $\hat{\Delta}_\nu$ at each frequency, we can use the Fourier series (Equation 5.5) to calculate the estimated driving force function.

For our example data, our white noise analysis in Section 5.4.3 has shown that only approximately the first 102 terms of the Fourier series are relatively reliable. This is fine for our purposes, because we are only interested in the large scale or low-frequency fluctuation of the driving force.

Intuitively, the large scale movement of the oscillator should be due to the large scale fluctuation of the wind force. To capture the large scale wind movement, we normally take the first 30 terms of the $\hat{f}(\tau)/m$'s Fourier series expansion. The plots would be roughly similar with a few more higher frequency details. We want to plot it against the observed oscillator's displacement, to see if our experiment confirms our intuition. Since $\hat{f}/m$ and $\hat{x}$ do not have the same units, we scaled $\hat{f}/m$ by a constant factor to make them fit into the same graph, and this does not change $\hat{f}/m$'s fluctuation pattern. In Figure 5.15, we plot $\hat{f}/m$ using its first 30 terms against the oscillator's displacement. Clearly, the large scale oscillating movement follows the large scale wind fluctuation, as we expected.

5.6 Comparing Simulated and Actual Displacement

In this section, we conduct an experiment to demonstrate the difference between the true driving force and the extracted force, and show their effect on the same oscillating

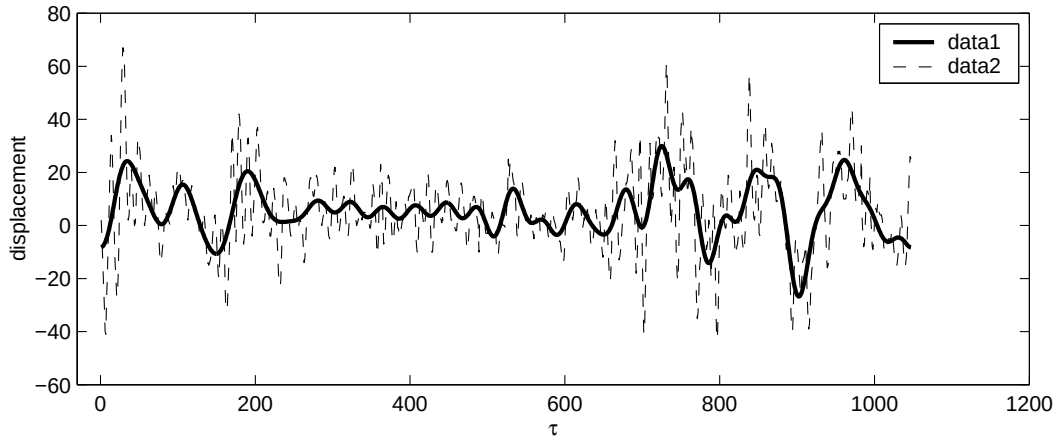

Figure 5.15: The large scale driving force fluctuation (*data1*) compared against the observed oscillator displacement (*data2*).

system.

5.6.1 Experimental Setup

Figure 5.16 shows the experimental setup.

- We create a simple true force f, with sample period $T = 1$ and sample size $N = 513$. We choose the amplitude of the true force in the frequency domain to be constant, say, 1. We let the phase shift of the true force at the frequency domain to be random between $[0, 2\pi]$ as shown in Figure 5.17.

- The true natural frequency ω_0 corresponds to $\nu_0 = 30$ cycles per N samples, and the true damping coefficient γ corresponds to $\Delta\nu = 10$.

- Using the true force and the true parameters, we calculate the true displacement x, as shown in Figure 5.16.

- Since in real life, there is noise in our measurements, the displacement we observed $x^\star$ is modelled as the sum of the true displacement and a white noise component.

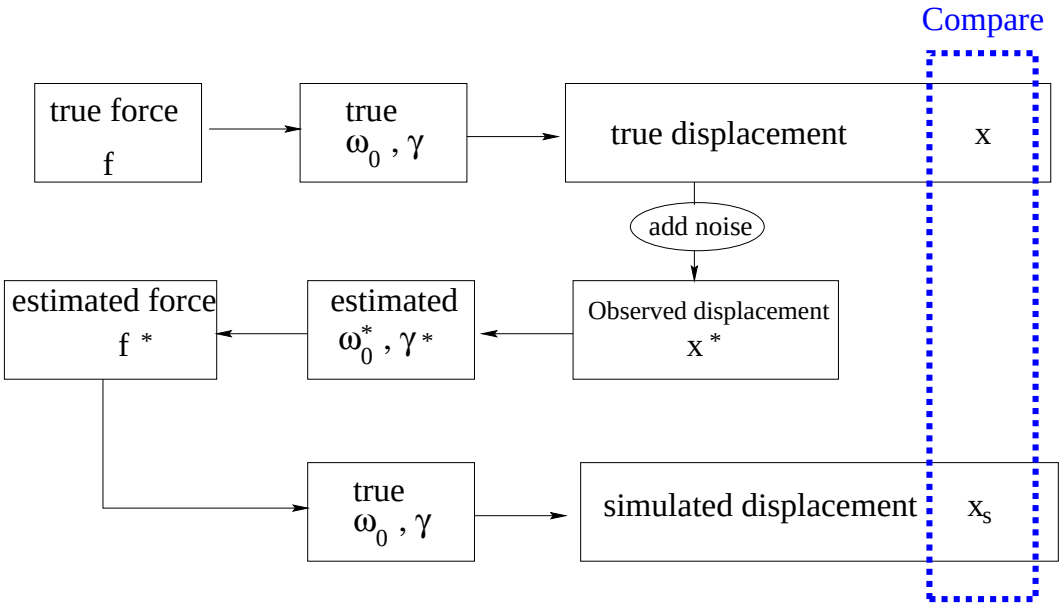

Figure 5.16: Constructing the simulated displacement, then comparing it to the true displacement without added noise.

For this experiment, the Gaussian white noise we added has its standard deviation being $\frac{1}{20}$ of the maximum true displacement amplitude, $\frac{1}{20} \cdot max(|x(\tau)|)$.

- To obtain the estimated natural frequency $\omega_0^\star$ and the estimated damping coefficient $\gamma^\star$, we examine the discrete Fourier transformation (DFT) of the observed displacement $x^\star$. The square of the amplitude of the DFT is plotted in Figure 5.18. The estimated natural frequency corresponds to $\nu_0^\star = 30$, and the estimated damping coefficient $\gamma^\star$ corresponds to $\Delta\nu^\star = 10.76$. There is a 0% error in the natural frequency estimation and a 7.6% error in the damping coefficient estimation.

- Next, we estimate the parameters of the force from the observed displacement. In Figure 5.19, we plot the amplitude of the estimated force $f^\star$ and that of the true force f. The difference between the phase delays of the extracted force and the true force at each frequency is plotted in Figure 5.20. By observing the two figures, we

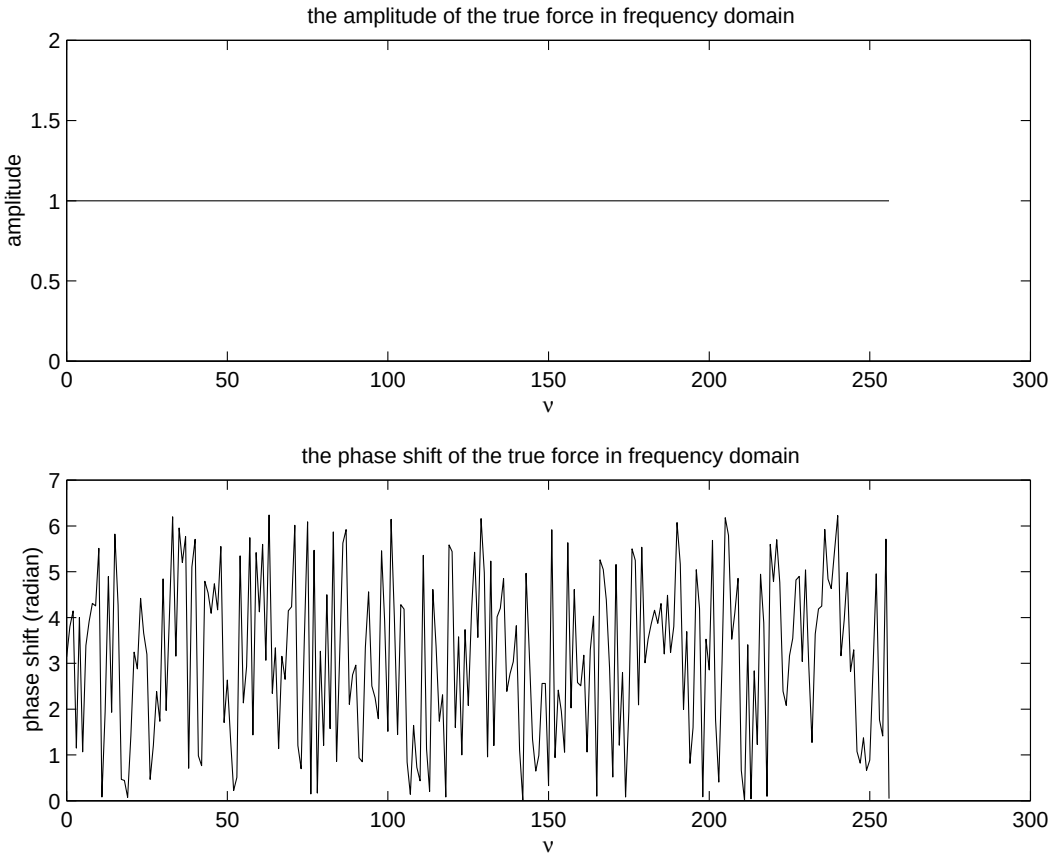

Figure 5.17: The amplitude and phase shift of the true force in frequency domain.

can tell that the estimated values after the first 50 terms are becoming inaccurate.

To obtain a quantitative measure of the error, we calculated the average error for a sequence of frequency windows with window size $\frac{1}{3} \cdot \omega_0$. The average relative errors for the extracted force amplitude and the average absolute errors for the phase delay are listed in Table 5.2. The relative error for the extracted force amplitude is less than 6.9% near the natural frequency of the system. And the error increases as the frequency becomes higher. At frequency $2\omega_0$, the relative error in the force amplitude is near 20%.

- Now, we apply the estimated force $f^\star$ to the original system with the true nat-

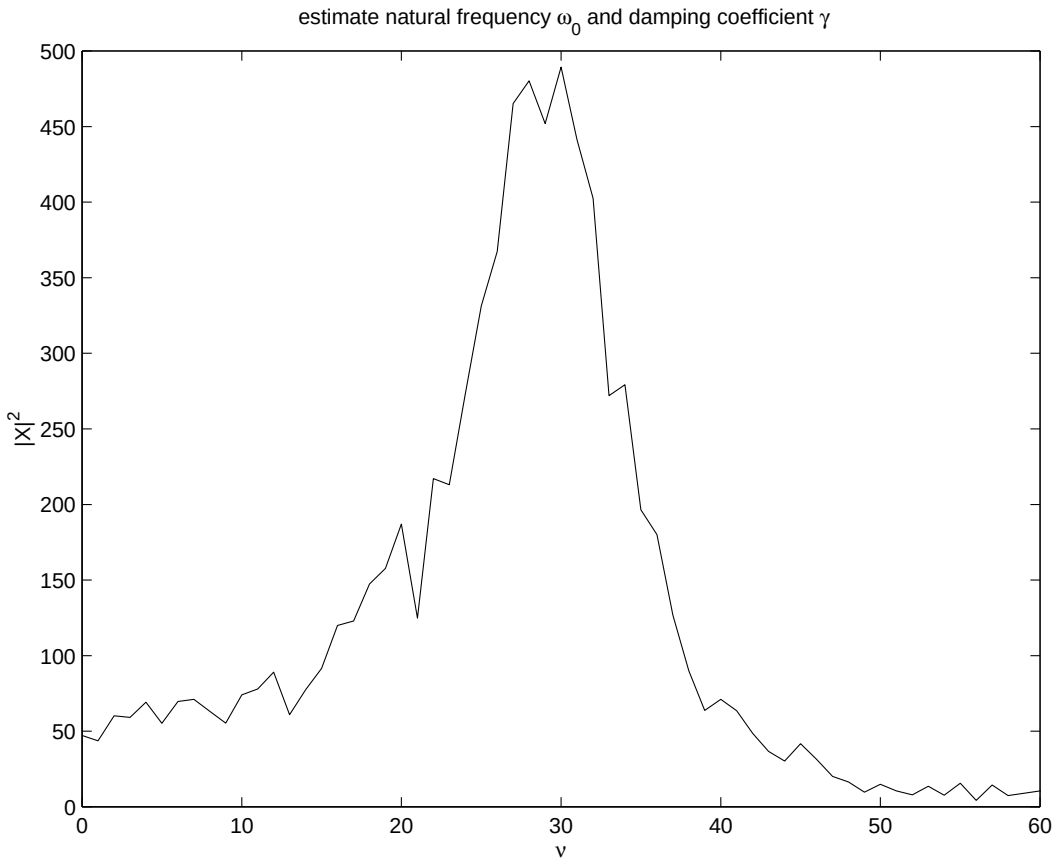

Figure 5.18: The square of the amplitude of the frequency domain response of the observed noisy displacement.

ural frequency ω_0 and the true damping coefficient γ. The resulting simulated displacement is x_s. The true displacement x and the simulated displacement x_s are compared in Figure 5.21. As we can see from the plot, the movements of the two displacements are similar but not exactly the same.

So far, we have discussed how to extract the effective driving force information from observed oscillator displacement. We demonstrated how to analyze the noise in the result and the effect of the noise in the extracted driving force. This is a basic component which will be used in different applications of the inverse problem of harmonic oscillation.

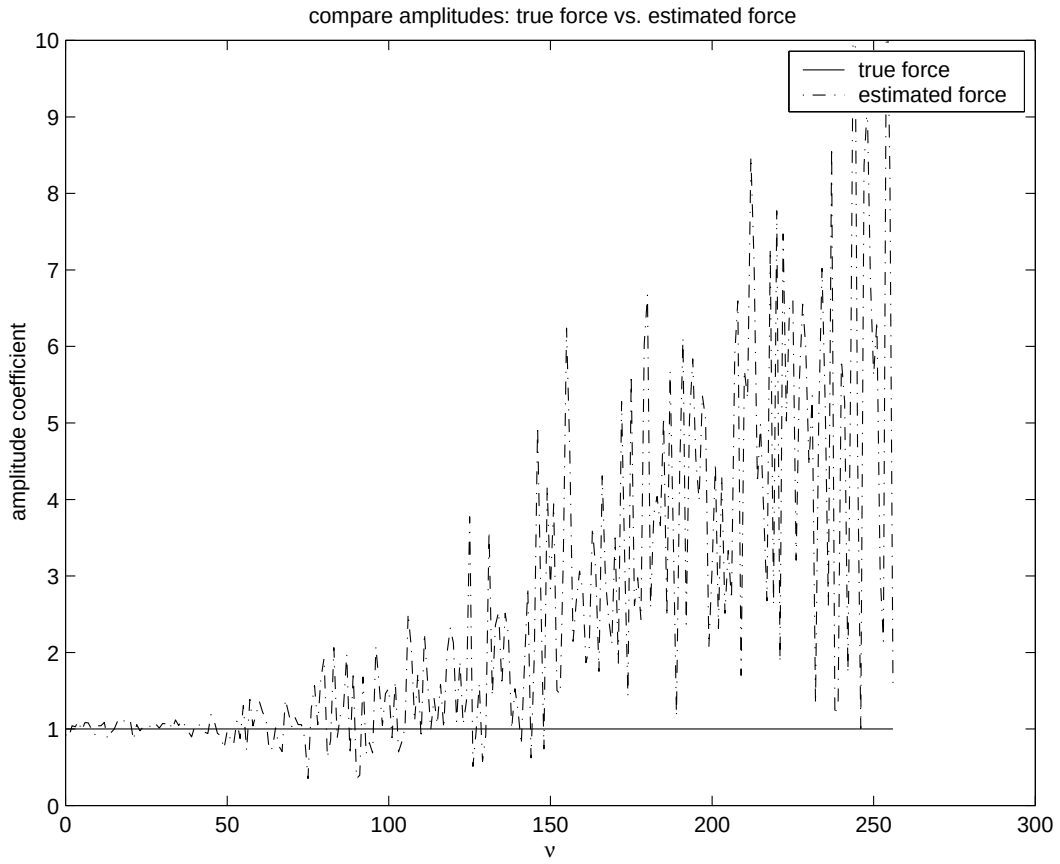

Figure 5.19: A comparison of the amplitude of the actual force and the estimated force.

To demonstrate the usefulness of VIDA, in particular, the inverse problem of harmonic oscillation, we will discuss two applications in the next few chapters. In the first application, we extract wind speed parameters, and in the second, we estimate regular wave parameters. In addition to the basic modelling steps 2(a) and 2(b) discussed above, we will explore the other application dependent parts in the processing stages: scan, model, synthesize.

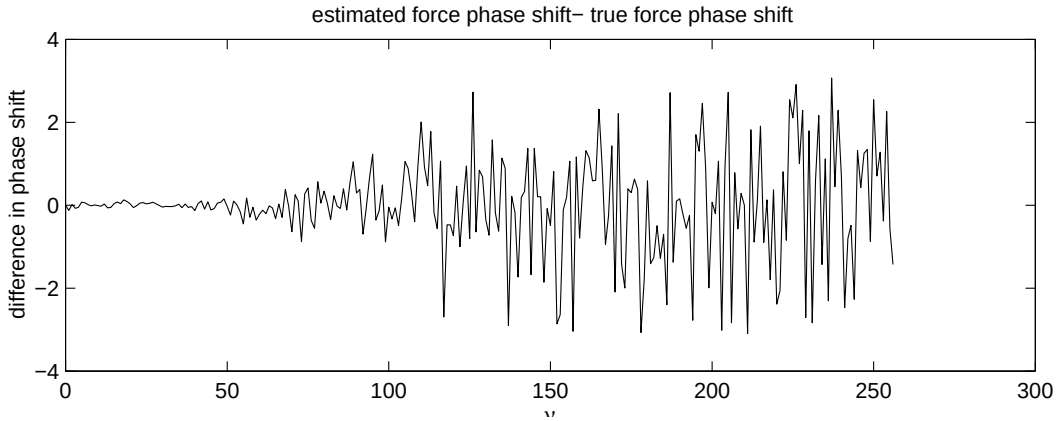

Figure 5.20: The difference between the phase shift of the estimated force and the actual force.

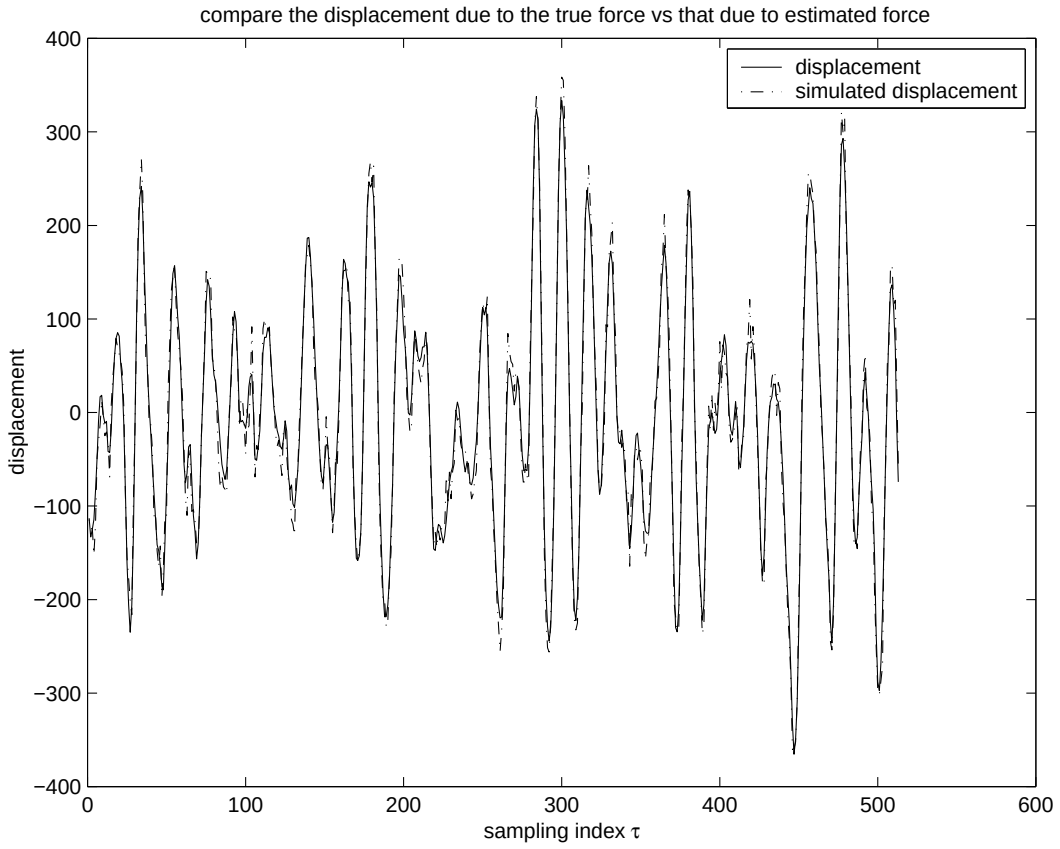

Figure 5.21: A comparison of the simulated displacement due to the estimated force and the true displacement without added noise.

Table 5.2: Experimental result: the average relative error in extracted force for different frequency windows.

Center of window (ω/ω_0)	relative error for amplitude	absolute error for phase delay
0.2000	0.0628	0.0130
0.5333	0.0702	0.0220
0.8667	0.0613	0.0169
1.2000	0.0698	0.0078
1.5333	0.0929	0.0249
1.8667	0.1796	0.0668
2.2000	0.2367	0.0462
2.5333	0.2612	0.1331
2.8667	0.5169	0.0939
3.2000	0.4917	0.1814
3.5333	0.5153	0.1889
3.8667	0.5365	0.4287
4.2000	0.6863	0.2430
4.5333	0.9835	0.2608
4.8667	1.3753	0.4242
5.2000	2.1689	0.4788
5.5333	1.7016	0.3137
5.8667	2.4852	0.4749
6.2000	2.8746	0.3284
6.5333	3.4773	0.3827
6.8667	2.6939	0.3266
7.2000	4.2162	0.4853
7.5333	4.6616	0.5016
7.8667	3.2204	0.4277
8.2000	5.1462	0.4203

Chapter 6

Extracting Wind Speed Parameters

In this chapter, we show how to extract wind speed parameters by observing the motion of objects in the video, such as plants, as shown in Figure 6.1.

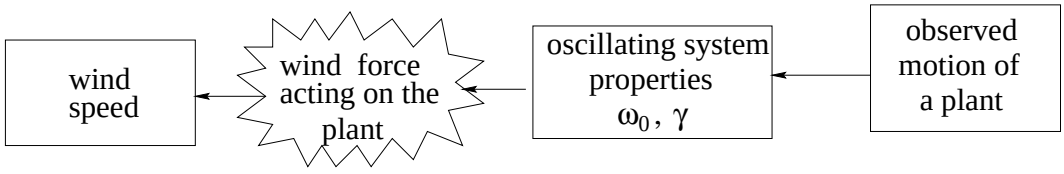

Figure 6.1: Extract wind speed from the motion of a plant in the wind.

6.1 Assumptions about the Input

An example input video sequence could be tree branches moving in the wind as shown in Figure 6.2. The swaying of a plant or a tree branch in the wind is approximated by the motion of a harmonic oscillator.

The video sequence is the 2D projection of the 3D world as shown in Figure 6.3. When our camera is far away from the tree, we assume the projection is orthogonal.

Figure 6.2: An example frame from a video of tree swaying in the wind.

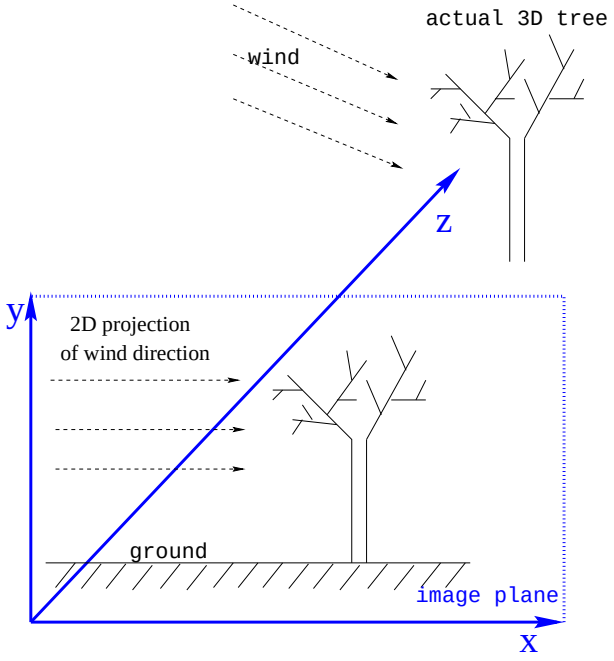

Figure 6.3: The video sequence is a 2D projection of the 3D world.

In the video, we observe the x and y components of the tree movement in the image coordinate system. For the purpose of demonstrating the principle of our wind speed estimation technique, we can assume that the direction of the large scale wind movement is parallel to the ground. For our purposes, we want to recover the large scale wind speed variation over time, so we will consider only the large scale tree motion, ignore the small noise and minor turbulence. Hence, small scale tree movements such as leaves fluttering due to small scale wind turbulence are neglected.

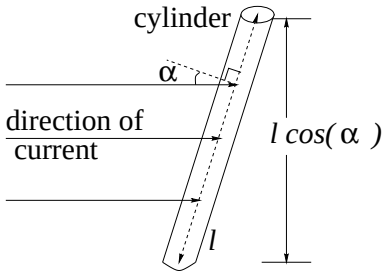

Figure 6.4: Projecting the length of the cylinder to a plane perpendicular to the direction of the current.

6.2 Assumptions about the Model

To simplify the problem, we treat the situation as the air flowing around a cylinder. This simplification is made based on the following argument. The main tree branch we observed has a number of sub-branches. This main branch and its sub-branches roughly move together as a whole. At wind speed V, the force the wind exerts on a cylindrical object is proportional to $V^2 Dl\cos(\alpha)$, where D is the diameter of the cylinder, l is the length of the cylinder and $l\cos(\alpha)$ is the projection of the cylinder length to the plane perpendicular to the direction of the fluid motion as shown in Figure 6.4. We consider the main branch and its sub-branch as consisting of $n_{cylinder}$ segments of cylinders. A cylinder segment would have diameter D_i, length l_i and wind instance angle α_i. So the total amount of wind force the structure experiences is proportional to

$$V^2 \sum_{i=1}^{n_{cylinder}} D_i l_i \cos(\alpha_i).$$

This summation merely estimates the effective cross-sectional area which is directly attacked by the wind. For our purposes, using a single cylinder for this estimate is sufficient,

where for the simple cylinder

$$Dl\cos(\alpha) = \sum_{i=1}^{n_{cylinder}} D_i l_i \cos(\alpha_i), \tag{6.1}$$

for an appropriate D and l. We make this simplifying assumption for the following reasons:

- We are only interested in the wind speed fluctuation along the direction parallel to the image plane, not its magnitude. When we use this fluctuation to drive the computer generated objects, only the relative dimensions and masses of the virtual objects and the original object matters. For instance, if we want to use the wind to drive synthetic snow flakes, the mass ratio between the tree and a snow flake only needs to be a rough and a somewhat reasonable guess, possibly achieved by trial and error. But that is in any event the best we could do, because we have no way of accurately finding out the tree's actual mass versus the snow flake's actual mass.

- When the tree is not moving too wildly, the summation in Eq. 6.1 is roughly constant for a given tree branch. Estimating the main tree branch and its sub-branches together as one cylinder would be a reasonable and efficient approach, and it would not affect our estimate of the wind speed fluctuation pattern.

- For a similar reason as our argument about the relative masses of two objects above,

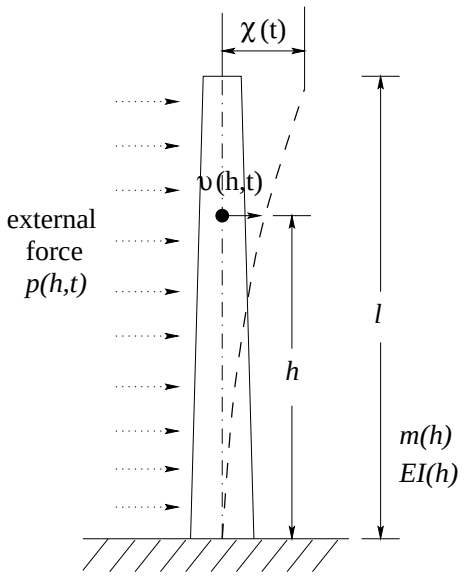

Figure 6.5: Flexure beam treated as a SDOF system.

the gain in parsing the detailed tree branch structure to estimate the summation

$$\sum_{i=1}^{n_{cylinder}} D_i l_i \cos(\alpha_i)$$

is minimal.

6.3 Cantilever Beam Model

In Chapter 5, to make the explanation simple and intuitive, we used a single degree of freedom (SDOF) system with point mass to model an oscillator. In this chapter, we will use a cantilever beam model to approximate flexible beams such as the stem of a plant or the trunk of a tree, as shown in Figure 6.5. We will show the close relationship between the cantilever beam model and the point mass model. We will use our knowledge of the point mass model to solve for the oscillation parameters of the cantilever beam model.

6.3.1 Beam Bending

Since a plant swaying in the wind is similar to a beam bending, we will start our discussion by explaining how to use beam bending to model the forces exerted on a plant or tree like object.

Let us consider a formulation of the equation of motion for the cantilever beam in Figure 6.5 in two dimensional space (2D). The essential properties of the beam (excluding damping) are its flexural stiffness $EI(h)$ and its mass per unit of length $m(h)$. Here, $h = 0\ ..\ l$ is the distance from a point on the beam to the base of the beam measured along the beam, l is the length of the beam, E is the Young's modulus of elasticity and I is the moment of inertia of the cross section of the beam. The beam is assumed to be subjected to external force perpendicular to it with magnitude per unit length $p_{eff}(h,t)$.

When a beam deforms in flexure, the system in principle has an infinite number of degrees of freedom. The shape of the beam at a given point in time can be represented as a linear combination of different mode shapes. A simple single degree of freedom (SDOF) analysis can be used, by assuming that only a single flexure deflection pattern can be developed [CP93].

To approximate the motion of this system with a single degree of freedom, it is necessary to assume that the beam will deform only in a single shape. The deflection function will be designated $\varphi(h)$, and the amplitude of the motion relative to the moving base will be represented by the generalized coordinate $\chi(t)$. Thus, the shape of the

deflection of the beam at a particular time t is

$$\upsilon(h,t) = \varphi(h)\chi(t). \tag{6.2}$$

Typically, the generalized coordinate is selected as the displacement of some convenient reference point in the system, such as the tip displacement of this beam. In this case, the shape function is the dimensionless ratio of the local displacement to this reference displacement [CP93, TL25, TM49]:

$$\varphi(h) = \frac{\upsilon(h,t)}{\chi(t)}. \tag{6.3}$$

Since we need to take the derivatives of a variable with respect to time t or with respect to the length parameter h, we will define some notations to make the mathematical formulas easier to read. We will use the single dot ($\dot{}$) and the double dot ($\ddot{}$) to denote taking the first and second derivatives of a function with respect to t, respectively. We will use the single apostrophe mark ($'$) and the double apostrophe mark ($''$) to denote taking the first and second derivatives of a function with respect to h, respectively. Thus, for example,

$$\frac{\partial \upsilon(h,t)}{\partial t} \equiv \dot{\upsilon}(h,t),$$

and

$$\frac{\partial \upsilon(h,t)}{\partial h} \equiv \upsilon'(h,t).$$

Since the structure we are considering is flexible, internal virtual work δW_I is per-

formed by the real internal moments $M(h,t)$ acting through their corresponding virtual changes in curvature $\delta v'' = \delta[\frac{\partial^2 v(h,t)}{\partial h^2}]$. The virtual-work principle requires that the external virtual work, $\delta W_E(t)$, performed by the external loadings acting through their corresponding virtual displacements be equated to the internal virtual work, i.e.,

$$\delta W_E = \delta W_I. \tag{6.4}$$

To develop the equation of motion in terms of relative displacement $v(h,t)$, the base of the structure can be treated as fixed while an effective loading $p_{eff}(h,t)$ is applied. The inertial loading is then given by

$$f_I(h,t) = m(h)\ddot{v}(h,t). \tag{6.5}$$

Using the full set of external forces, the external virtual work is given by

$$\delta W_E = -\int_0^l f_I(h)dh + \int_0^l p_{eff}(h,t)\delta v(h,t)dh \tag{6.6}$$

and consistent with the above statement regarding internal virtual work,

$$\delta W_I(t) = \int_0^l M(h,t)\delta v''(h,t)dh. \tag{6.7}$$

where $v''(h,t) = \frac{\partial^2 v(h,t)}{\partial h^2}$ is the curvature.

Based on many years of engineering experimentation and practices, engineers find that, for most applications, the damping stresses are developed approximately in propor-

tion to the strain velocity, a uniaxial stress-strain relation of the form

$$\sigma = E[\epsilon + a_1\dot{\epsilon}] \tag{6.8}$$

may be adopted, where E is Young's modulus and a_1 is a damping constant. Intuitively, the strain ϵ is due to the curvature $v''(h,t)$ of the bending beam. Hence, the bending moment M is also related the bending beam. Now, we want to establish the relationship between the bending moment M and the bending curvature $v''(h,t)$. Figure 6.6 shows a bending beam. Let the curve C_1C_2 be the neutral axis where the length of this axis does not change during the bending. Let y be the coordinate which indicates the distance a point on the beam is from the neutral axis C_1C_2. In material engineering, the strain ϵ is related to the curvature $v''(h,t)$ of the bending beam by

$$\epsilon = -yv''(h,t). \tag{6.9}$$

According to Timoshenko [Tim49], the bending moment M and the stress σ are related by

$$M = -\int_A y\sigma dA, \tag{6.10}$$

where A is the area of the cross section of the beam. Substituting Eq. 6.8 into Eq. 6.10, we have the relationship between the bending moment M and the strain ϵ:

$$M = -\int_A yE(\epsilon + a_1\dot{\epsilon})dA, \tag{6.11}$$

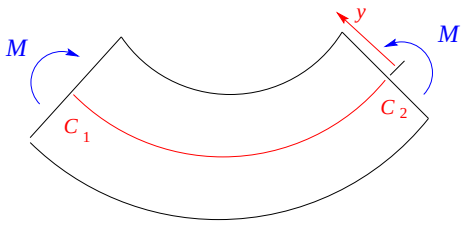

Figure 6.6: The bending moment M is related to the curvature of the bending beam.

Substituting Eq. 6.9 into Eq. 6.11, we have the relationship between the bending moment M and the curvature $v''(h,t)$:

$$\begin{aligned} M &= \int_A yE(yv''(h,t) + a_1 y\dot{v}''(h,t))dA \\ &= E(\int_A y^2 dA)(v''(h,t) + a_1\dot{v}''(h,t)). \end{aligned} \tag{6.12}$$

Notice that the moment of inertia I of the cross section of the beam is exactly

$$I = \int_A y^2 dA.$$

Hence, we can simplify Eq. 6.12 to be

$$M(h,t) = EI(h)[v''(h,t) + a_1\dot{v}''(h,t)]. \tag{6.13}$$

The basic relations may be expressed as follows:

$$\begin{array}{ll} v(h,t) = \varphi(h)\chi(t) & \dot{v}''(h,t) = \varphi''(h)\dot{\chi}(t) \\ v'(h,t) = \varphi'(h)\chi(t) & \delta v(h,t) = \varphi(h)\delta\chi(t) \\ v''(h,t) = \varphi''(h)\chi(t) & \delta v'(h,t) = \varphi'(h)\delta\chi(t) \\ \ddot{v}(h,t) = \varphi(h)\ddot{\chi}(t) & \delta v''(h,t) = \varphi''(h)\delta\chi(t) \end{array}$$

Finally, expressions for the external and internal virtual work may be expressed as

follows:

$$\begin{aligned} \delta W_E &= [-\ddot{\chi}(t)\int_0^l m(h)\varphi^2(h)dh + \int_0^l p_{eff}(h,t)\varphi(h)dh]\delta\chi \\ \delta W_I &= [\chi(t)\int_0^l EI(h)\varphi''(h)^2dh + a_1\dot{\chi}(t)\int_0^l EI(h)\varphi''(h)^2dh]\delta\chi. \end{aligned} \tag{6.14}$$

Putting Eq. 6.14 into Eq. 6.4 yields the generalized equation of motion with respect to the moving base $\chi(t)$:

$$m^\star\frac{d^2\chi(t)}{dt^2} + c^\star\frac{d\chi(t)}{dt} + k^\star\chi(t) = p^\star_{eff}(t). \tag{6.15}$$

Here, we have

$$\begin{aligned} \text{generalized mass} &: m^\star = \int_0^l m(h)\varphi^2(h)dh, \\ \text{generalized damping} &: c^\star = a_1\int_0^l EI(h)\varphi''(h)^2dh, \\ \text{generalized flexural stiffness} &: k^\star = \int_0^l EI(h)\varphi''(h)^2dh, \\ \text{generalized effective load} &: p^\star_{eff} = \int_0^l p_{eff}(h,t)\varphi(h)dh \end{aligned} \tag{6.16}$$

Clearly, the generalized equation of motion for the beam is very similar to the Newton's equation of motion for a single degree of freedom (SDOF) system with a point mass. In fact, the magnitude $\chi(t)$ of the beam motion is proportional to the displacement in the point mass system. We would like to explore the relationship between them in more detail. Before we do that, we would like to say a few words about the beam shape function $\varphi(h)$.

6.3.2 Shape of Deflection for Bending Beam

Under most practical circumstances, there is no analytical solution to the shape of a bending beam. Often a certain approximate deflection that satisfies some reasonable constraints is good enough for engineering purposes [Tim47, TM49]. For the deflection of the bending beam, there are a number of simple deflection often used in engineering textbooks. The one we chose for our example is based on a beam deflection equation in [TM49]. We adapted it to our generalized equation of motion. In our case, the beam shape equation is

$$\varphi(h) = (h^2/24)(6l^2 - 4hl + h^2). \tag{6.17}$$

The maximum deflection φ_{max} at the top of the beam is

$$\varphi_{max} = \varphi(h = l) = \frac{l^4}{8}. \tag{6.18}$$

Moreover, the shape function φ can be calculated from the deflection at the top of the beam φ_{max}. To simplify things, we will reparametrize the beam using

$$h \in cl, \text{ where } c \in [0, 1],$$

so that Eq. 6.17 becomes

$$\begin{aligned} \varphi(h) = \varphi(cl) &= \frac{c^2 l^2}{24}(6l^2 - 4cl^2 + c^2 l^2) \\ &= \frac{l^4}{24} c^2 (6 - 4c + c^2) \end{aligned}$$

$$
\begin{aligned}
&= (\varphi_{max}\frac{8}{l^4})\frac{l^4}{24}c^2(6-4c+c^2) \\
&= \varphi_{max}\frac{c^2}{3}(6-4c+c^2) \qquad (6.19)
\end{aligned}
$$

So knowing the maximum deflection, φ_{max}, of the beam, we can use it to define the deflection at other points on the beam given by parameter c.

6.3.3 Relation to the Point Mass System

A SDOF system with a point mass is very closely related to the SDOF cantilever beam system. In this section, we will show how their parameters are related. Since the generalized equation of motion, Eq. 6.15, is in the same form as the Newton's equation of motion for SDOF system with a point mass, we will use it as the departure point for our discussion.

We rewrite the generalized equation of motion into a form we are familiar with by dividing both sides of Eq. 6.15 by $m^\star$:

$$\frac{d^2\chi(t)}{dt^2} + \gamma^\star\frac{d\chi(t)}{dt} + \omega_0^{\star 2}\chi(t) = \frac{p^\star_{eff}(t)}{m^\star}, \qquad (6.20)$$

where $\gamma^\star = c^\star/m^\star$ and $\omega_0^{\star 2} = k^\star/m^\star$, and $m^\star > 0$.

If we know what $\chi(t)$ is, then the oscillating system pipeline would be as shown in Figure 6.7. We do not have $\chi(t)$, but it is related to the beam's top displacement $x(t)$. Specifically,

$$x(t) = v(h=l,t)$$

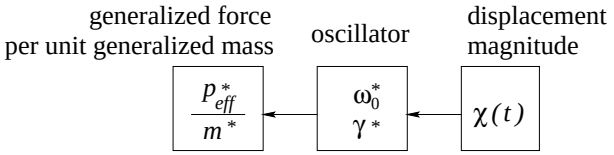

Figure 6.7: Using displacement magnitude to estimate generalized force per unit generalized mass.

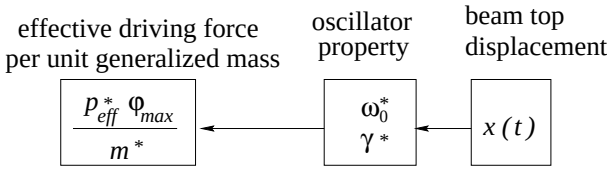

Figure 6.8: Using beam top displacement to estimate generalized force per unit generalized mass.

$$\begin{aligned} &= \varphi(h=l)\chi(t) \\ x(t) &= \varphi_{max}\chi(t), \end{aligned} \tag{6.21}$$

where $\varphi_{max} = \varphi(h = l)$ is a constant for our model of the cantilever beam. To make use of this relation, we multiply both sides of generalized equation of motion Eq. 6.20 by φ_{max}:

$$\varphi_{max}\frac{d^2\chi(t)}{dt^2} + \gamma^\star\varphi_{max}\frac{d\chi(t)}{dt} + \omega_0^{\star 2}\varphi_{max}\chi(t) = \frac{p^\star_{eff}(t)}{m^\star}\varphi_{max}. \tag{6.22}$$

Since $x(t) = \varphi_{max}\chi(t)$, $\dot{x}(t) = \varphi_{max}\dot{\chi}(t)$ and $\ddot{x}(t) = \varphi_{max}\ddot{\chi}(t)$, Eq. 6.22 can be rewritten as:

$$\frac{d^2x(t)}{dt^2} + \gamma^\star\frac{dx(t)}{dt} + \omega_0^{\star 2}x(t) = \frac{p^\star_{eff}(t)}{m^\star}\varphi_{max}. \tag{6.23}$$

So the pipeline now looks like Figure 6.8.

The generalized natural frequency $\omega_0^\star$ and generalized damping coefficient $\gamma^\star$ can thus be estimated from the observable quantity $x(t)$. It makes sense, for in this model the beam shape $\varphi(h)$ is fixed for a given beam, and thus the whole beam moves in the same

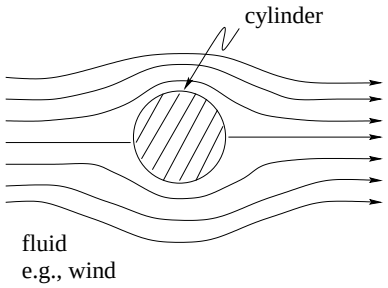

Figure 6.9: A fluid flows around a cylinder.

frequency and with the same damping.

Now we would like to know what $\frac{p^{\star}_{eff}(t)}{m^{\star}}\varphi_{max}$ is related to forces in the real world.

6.3.4 Drag Force

In fluid dynamics, when a fluid flows around a cylinder, as shown in Figure 6.9, it produces a drag force, f_{drag}.

We know

$$\varphi(h) = (h^2/24)(6l^2 - 4hl + h^2)$$

$$p^{\star}_{eff} = \int_0^l p_{eff}(h,t)\varphi(h)dh$$

$$m^{\star} = \int_0^l m(h)\varphi^2(h)dh$$

In our example, for simplicity, we would assume that the beam structure is a cylinder with constant diameter D and uniform mass distribution. Also, the drag force f_{drag} the wind exerts on the beam is assumed to be uniformly distributed along the length of the

beam. So we have the following simplified situation:

$$p^{\star}_{eff} = \frac{f_{drag}}{l}\int_0^l \varphi(h)dh$$
$$m^{\star} = \frac{m_{total}}{l}\int_0^l \varphi^2(h)dh,$$

where m_{total} is the total mass of the cylindrical shape cantilever beam like structure. Therefore,

$$\begin{aligned}\frac{p^{\star}_{eff}}{m^{\star}}\varphi_{max} &= \frac{\frac{f_{drag}}{l}\int_0^l \varphi(h)dh}{\frac{m_{total}}{l}\int_0^l \varphi^2(h)dh}\varphi_{max} \\ &= \frac{f_{drag}}{m_{total}}c_{shape}, \end{aligned} \tag{6.24}$$

where

$$c_{shape} = \frac{\int_0^l \varphi(h)dh}{\int_0^l \varphi^2(h)dh}\varphi_{max} = \frac{81}{52}, \tag{6.25}$$

and this is a constant for any particular shape function.

Hence, we can simplify the generalized equation of motion further and rewrite it as:

$$\frac{d^2x(t)}{dt^2} + \gamma^{\star}\frac{dx(t)}{dt} + \omega^{\star 2}_0 x(t) = \frac{p^{\star}_{eff}(t)\varphi_{max}}{m^{\star}} = \frac{f_{drag}c_{shape}}{m_{total}} \tag{6.26}$$

which is a simple Newton's equation of motion with point mass. So, $\frac{f_{drag}}{m_{total}}c_{shape}$ can be estimated from observed beam top displacement $x(t)$. This makes sense, because the beam top motion is a representative of the beam's motion as a whole. So the pipeline becomes as shown in Figure 6.10.

Note that the terms $\chi(t)$, $m^{\star}$ and $p^{\star}_{eff}$ are merely used for deriving the relationship

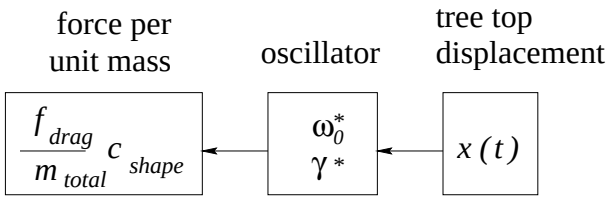

Figure 6.10: Estimate the drag force per unit mass from beam top displacement.

between the point mass model and the cantilever beam model. In practice, they never need to be calculated explicitly.

6.3.5 Wind Velocity

In this section, we will study the relationship between the drag force and the wind velocity.

6.3.5.1 Drag Coefficient

When a fluid flows around a cylinder, the drag force experienced by the cylinder depends on several factors. The drag force depends on many factors. One of the factors is the drag coefficient. Based on experimental data, physicists plotted the so-called *drag coefficient* C_D as a function of Reynolds' number $\mathcal{R}e$ – which is proportional to the air speed V, as shown in Figure 6.11. Reynolds' number $\mathcal{R}e$ is defined as

$$\mathcal{R}e = \frac{\sigma}{\eta} V D,$$

where σ and η are the density and the coefficient of viscosity of the fluid, D is the diameter of the cylinder. The drag coefficient C_D is a dimensionless number equal to the drag force f_{drag} divided by $\frac{1}{2}\sigma V^2 D l \cos(\alpha)$, where l is the length of the cylinder, $l\cos(\alpha)$

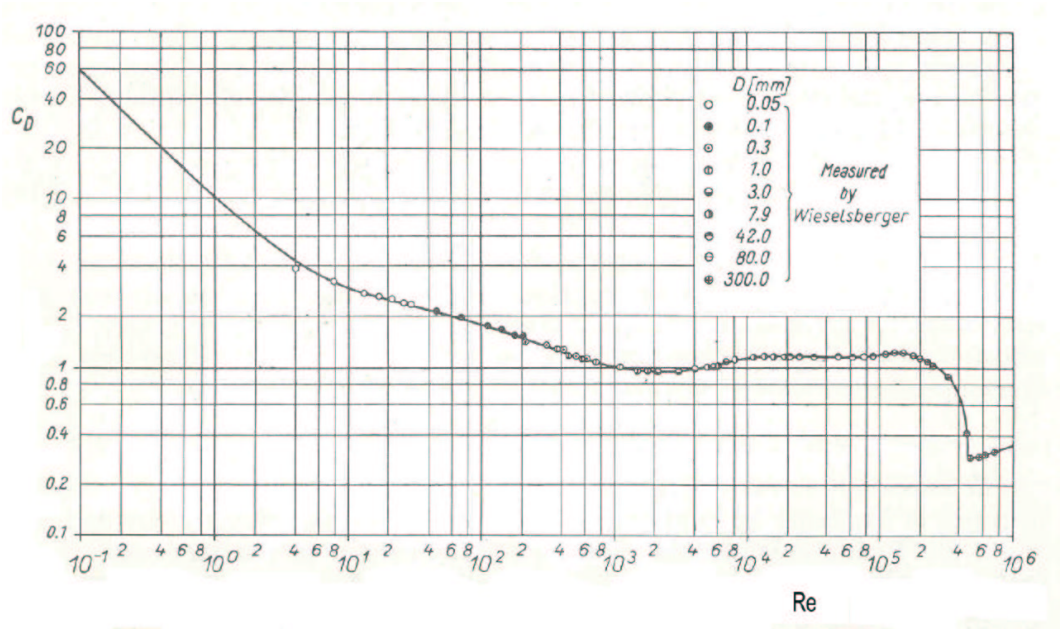

Figure 6.11: The drag coefficient C_D of a circular cylinder as a function of the Reynolds number. Courtesy of Schlichting [Sch60].

is the projection of the cylinder length to the plane perpendicular to the direction of the fluid as shown in Figure 6.4:

$$C_D = \frac{f_{drag}}{\frac{1}{2}\sigma V^2 D l \cos(\alpha)}. \tag{6.27}$$

The drag coefficient is useful, because the experimental points for the drag coefficient of circular cylinders of widely differing diameters fall on a single curve [Ide94] as shown in Figure 6.11.

6.3.5.2 Reynolds Number and the Flow Pattern

The flow changes for different ranges of Reynolds number. The significance of the Reynolds number is that for any two situations which have the same Reynolds number, the flows will be similar. Figure 6.12 shows some of the changes in flow pattern

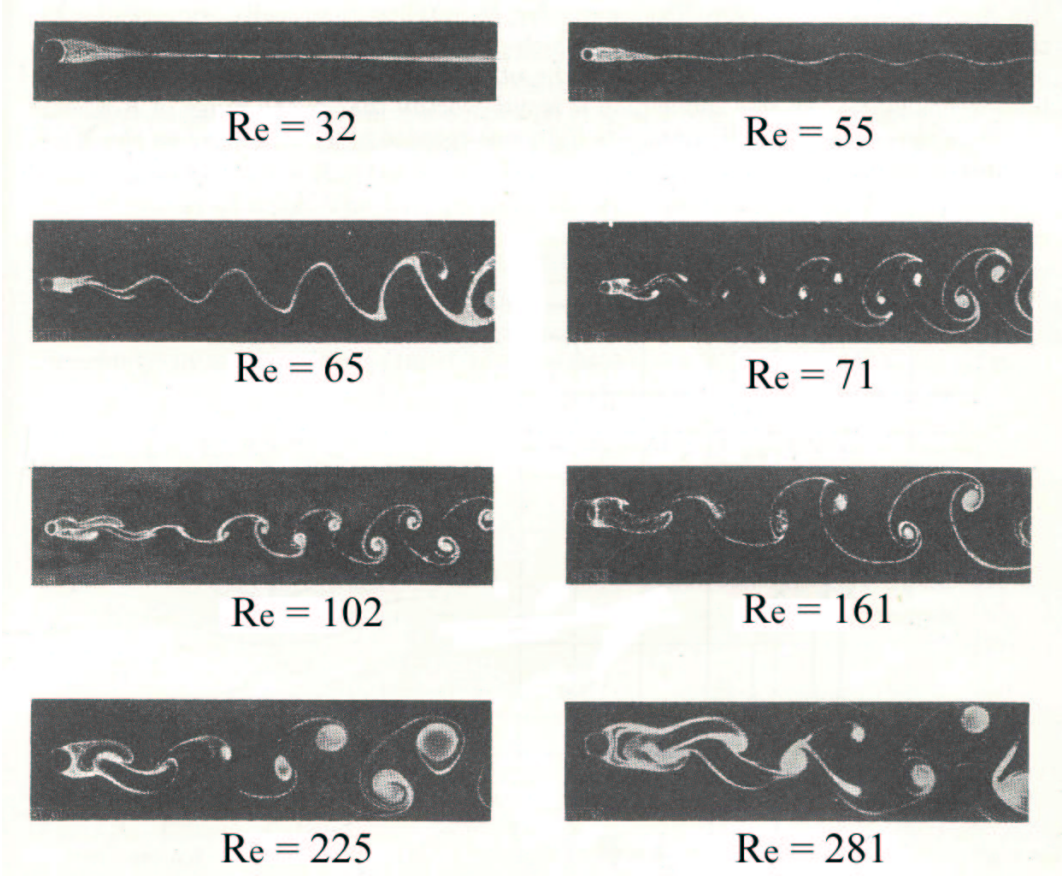

Figure 6.12: Field of flow of oil about a circular cylinder at varying Reynolds numbers from Homann [Hom36]. It shows the transition from laminar flow to a vortex street.

at different Reynolds number. This gives us some intuitive understanding as to how Reynolds number affects the flow around a circular cylinder. Physicists noticed that, as the Reynolds number increases, there are several typical type of flows: steady flow, laminar flow, periodic turbulent flow and turbulent boundary layer flow. The Reynolds number range for each type of flow is shown in Figure 6.13, and it also indicates how the drag coefficient is related to the change of flow pattern. Interested readers are referred to Schlichting [Sch60] and Feynman [Fey63] for details.

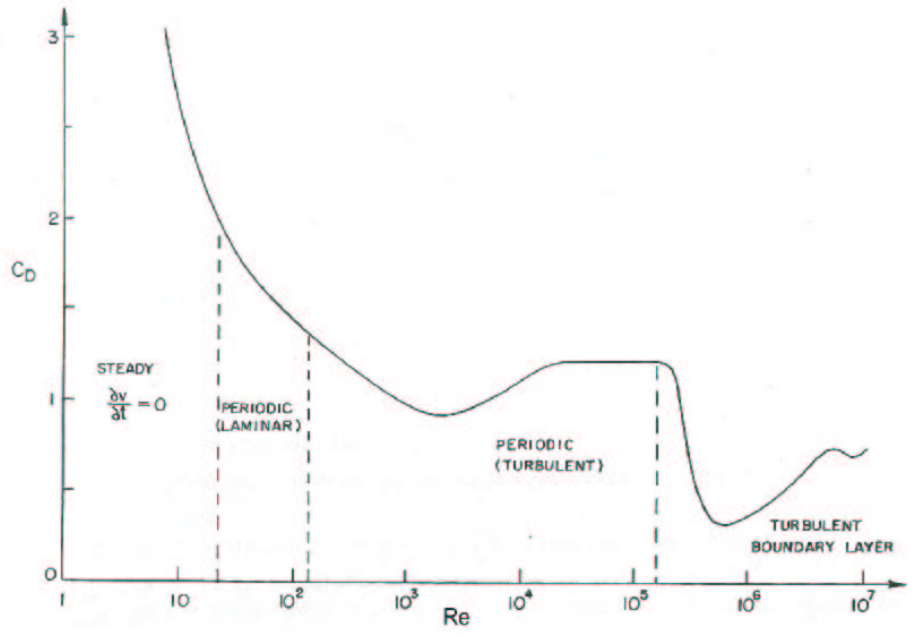

Figure 6.13: The change of flow pattern in relation with the Reynolds number and the drag coefficient.

6.3.5.3 Factors Affecting Wind Speed Estimation

There are many parameters which affect the relationship between the wind velocity and the drag force. Table 6.1 lists those factors that stay constant and those factors that might change:

The relationship between wind speed and wind force on a cylinder is as shown in Figure 6.14:

- The wind speed determines the Reynolds number,
- the Reynolds number determines the drag coefficient C_D,
- the wind speed and the drag coefficient C_D determine the drag force.

Now, we know the estimated drag force up to a scaling factor, and we would like to infer

Table 6.1: Factors which effects the relationship between the wind velocity and the drag force.

input	wind force per unit mass $\hat{f}/m = c_{shape}\hat{f}_{drag}/m$
unknown	wind velocity V up to a scaling factor
constants	air density σ air viscosity η cylinder roughness cylinder diameter D cylinder length l attack angle α
variables	Reynolds number $\mathcal{R}e$ drag coefficient C_D

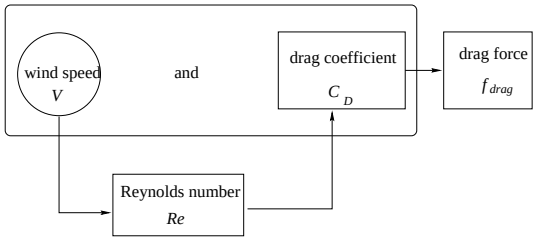

Figure 6.14: The relationship between wind speed and wind force on a cylinder.

the wind speed V. We do not know what drag coefficient C_D is, which might depend on the Reynolds number. In engineering, when the fluid flow around a cylinder is roughly in the categories shown in Figure 6.13, engineers typically treat the drag coefficient as a constant, such as $C_D = 1.2$, which is roughly the average drag coefficient for such flows. In this thesis, we will also use this constant.

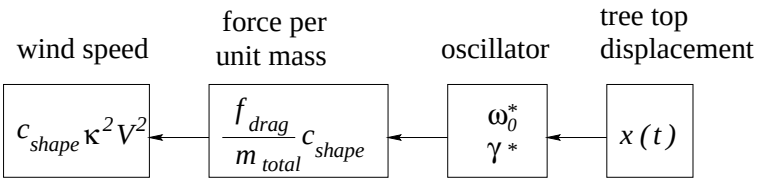

Figure 6.15: Estimate the wind speed from beam top displacement.

6.3.5.4 Estimating Wind Velocity up to a Scaling Factor

Eq. 6.27 gives us the relationship between the velocity V and the drag force f_{drag} it exerted on a circular cylinder. We can rewrite it as

$$V^2 = \frac{f_{drag}}{C_D \frac{1}{2}\sigma D l \cos(\alpha)} = \frac{f_{drag} c_{shape}/m_{total}}{C_D \frac{1}{2}\sigma D l \cos(\alpha) c_{shape}/m_{total}} = \frac{f_{drag} c_{shape}/m_{total}}{c_{shape}\kappa^2}, \tag{6.28}$$

where the κ depends on the parameters of the cylinder:

$$\kappa = \sqrt{\frac{1}{2}\frac{C_D \sigma D l \cos(\alpha)}{m}}, \tag{6.29}$$

and κ is a constant for a given cylindrical beam.

Since $f_{drag} c_{shape}/m_{total}$ can be estimated from the beam top displacement $x(t)$ using Eq. 6.26, we are able to estimate the wind speed fluctuation up to a scaling factor $\sqrt{c_{shape}}\kappa$. Note that we left V^2 in the Equation 6.28, because, for many applications such as re-applying the wind speed to other objects, we do not need to explicitly calculate V. For instance, we can use the relationship between κ_{old} in the original scenario and the κ_{new} in a synthesis scenario to work out the ratio between the estimated force per unit mass $f_{drag} c_{shape}/m_{total}$ and the new force per unit mass for a synthesized plant or tree branch.

Now, our process pipeline can be extended to include wind speed as shown in Figure 6.15.

So far, we have talked about the modelling aspect of the wind speed extraction problem. Next, we will discuss the scanning and synthesis stage of the process. For the synthesis stage, we will show two example applications.

6.4 Scanning Stage

The video input feature tracking process is automatically done using a software implemented by Jepson, Fleet and El-Maraghi using algorithms based on Shi and Tomasi [ST94], Jenkin and Jepson [JJ94], Jepson and Black [JB93], Jepson and Jenkin[JJ89], Jepson, Fleet and El-Maraghi [JFEM01], El-Maragi [EM02]. We want to track the movement of the top of the tree. Figure 6.16 shows one example frame of the tracking result. The regions in the ellipses are tracked from frame to frame. For synthesizing a computer generated object, normally, we need only the tracking result from one of the ellipses. For instance, in this example, we pick the rightmost ellipse. The center of the ellipse is used as an approximation for the top of the tree branch. We record the displacement $(\hat{x}_{image\_plane}, \hat{y}_{image\_plane})$ of the branch top from its resting position in the image coordinate system. The resting position of the top of the tree branch is estimated by the user. During the video sequence, there are periods of time when the wind dies down, and the tree branch comes to rest briefly. Since, most of the time, the change in the y component is negligible, we only need to consider the displacement in the x component. The displacement data is sampled at time interval T. As such, we will use the notation

http://www.dgp.toronto.edu/~meng/animation

Figure 6.16: Trace the movement of the top of tree branches.

$\hat{x}_{image\_plane}(\tau)$ for it, where sample index $\tau = 0 \ldots N-1$, and N is the total number of samples.

The measured displacement value $\hat{x}_{image\_plane}$ in the image coordinate system is related to the corresponding displacement x of the real object by a scaling factor as shown in Figure 6.17,

$$x = c_{distance} w_{pixel} \hat{x}_{image\_plane}, \tag{6.30}$$

where

$$c_{distance} = \frac{|z_{object}|}{|z_{image\_plane}|},$$

$|z_{image\_plane}|$ is the distance from the view point to the image plane, $|z_{object}|$ is the distance from the view point to the observed tree branch, and w_{pixel} is the width of each pixel in the image plane.

Hence, we need to modify the pipeline in Figure 6.15 to take the scaling factor $c_{distance}$ into account. After this modification, our pipeline is as shown in Figure 6.18.

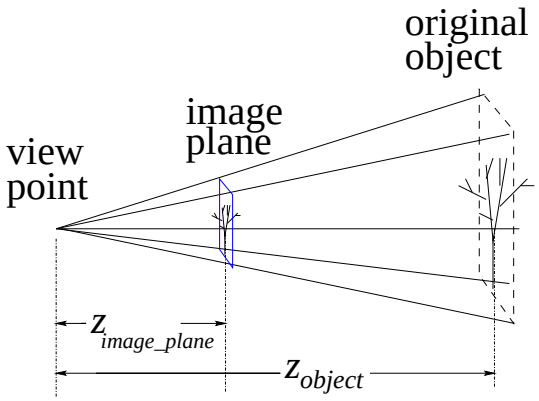

Figure 6.17: The relationship between $\hat{x}_{image\_plane}$ in the image coordinate system and the corresponding x in 3D space.

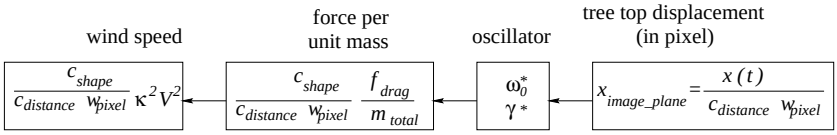

Figure 6.18: Estimate the wind speed parameter from beam top displacement measured in the image coordinate system.

6.5 Synthesis

For the wind-plant interaction scenarios, we can use the wind speed parameter we estimated to drive the animation of synthetic objects, such as trees, plants, snow, hair, clothes, etc.

6.5.1 Example: Wind and Synthesized Plant

In this section, we show an example of using the wind velocity parameter we estimated to drive animation of a computer synthesized plant in the foreground. A frame of the original video is as shown in Figure 6.2. We will refer to this video sequence as "trees-by-the-lake". The sampling parameters are shown in Table 6.2.

The data analysis pipeline is as shown in Figure 6.19. The plots of the data at each

Table 6.2: Sampling parameters for the trees-by-the-lake video sequence.

Number of sample points N	761
Sampling period T (second)	$\frac{1}{15}$

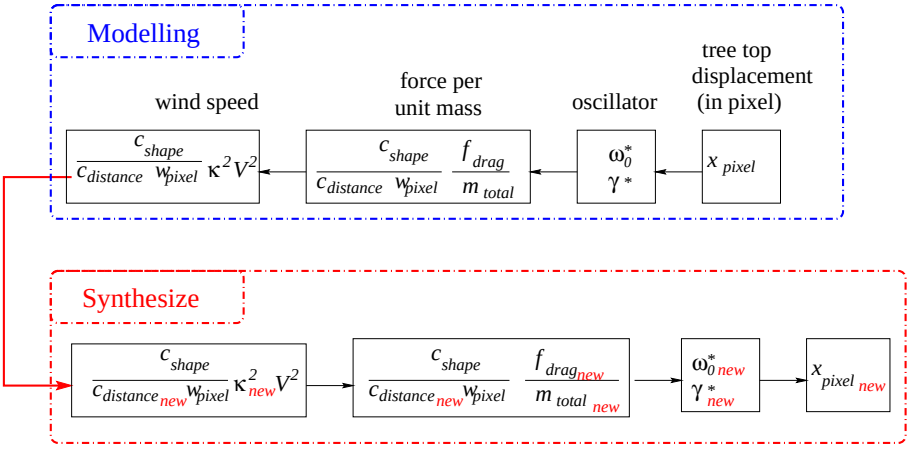

Figure 6.19: Pipeline for applying estimated wind velocity parameter to a new computer synthesized plant.

step are summarized below:

- Figure 6.20 shows the displacement for the top of the observed the plant branch. The resting position of the tree top is selected by user using his/her intuition. There were brief period in the video sequence when the wind is died down for a little while. During this time, the top of the tree is restored to its resting position. It is possible that the human observer could incorrectly pick the resting position of the top of the tree branch wrong. This would result in the displacement values being shifted by a constant value.

- Figure 6.21(top) shows the amplitude of the discrete Fourier transform of the observed displacement. The amplitude squared in Figure 6.21(bottom) is used for estimating the natural frequency and damping coefficient of the oscillation.

- Figure 6.22(top) compares the power density spectrums of the estimated force parameter and the expected noise. Figure 6.22(bottom) is the signal to noise power density spectrum ratio. This ratio is reasonably high for frequency range $\omega/\omega_0 \in [0, 3]$.

- We obtain the large scale fluctuation of the force parameters by applying a low pass filter to the estimated force parameters. Figure 6.23 is the drag force large fluctuation curve to within a scaling factor, $\frac{f_{drag}}{m_{total}} c_{shape}$, which we recovered using Eq. 6.26. In the video, it is clear that the wind blows in one direction, in this case, from left to the right. Hence, we expect the drag force to be having the same sign, say, positive. Note that it is possible for the human observer to choose the resting position of the top of the tree branch incorrectly by a few pixels. As we mentioned earlier, this could result in the displacement values being shifted by a constant value. In turn, the estimated driving force curve could be shifted by some constant value. This could potentially cause the driving force to be negative while we expect it to be positive. The relative fluctuation of the driving force curve remains the same.

Next, we make a computer synthesized plant. Table 6.3 compares the parameters of the observed plant branch and the computer synthesized plant branch.

We computed the displacement for the synthesized plant branch. In Figure 6.24, we compare this displacement to the displacement for the observed plant branch in the original video. We place the synthesized plant in the foreground in front of the captured video video. Figure 6.25 shows one frame of the resulting video sequence.

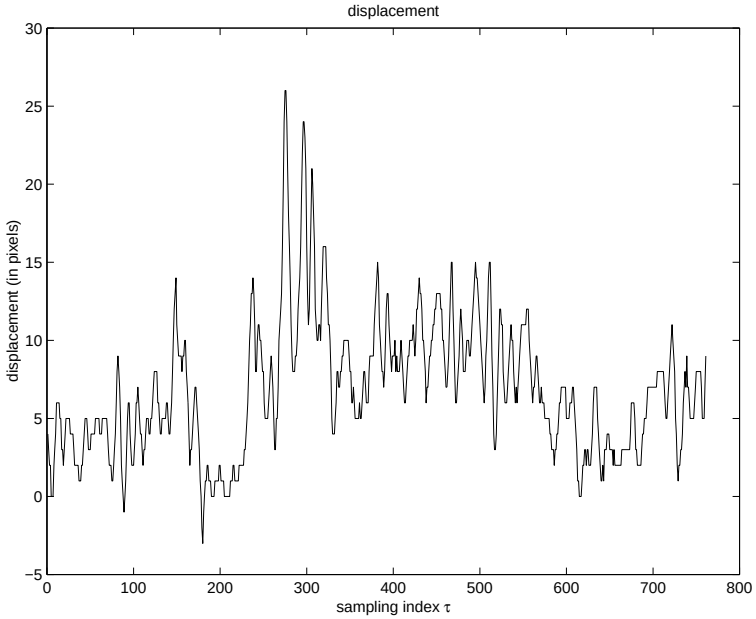

Figure 6.20: The displacement of one tree branch.

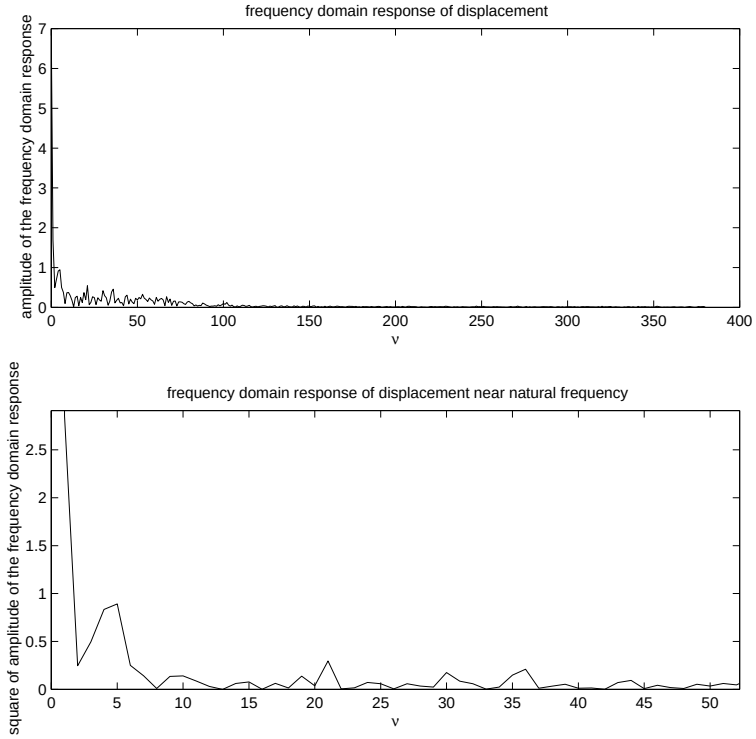

Figure 6.21: Top plot: the amplitude of the discrete Fourier transform (DFT) of the displacement; Bottom plot: the amplitude squared.

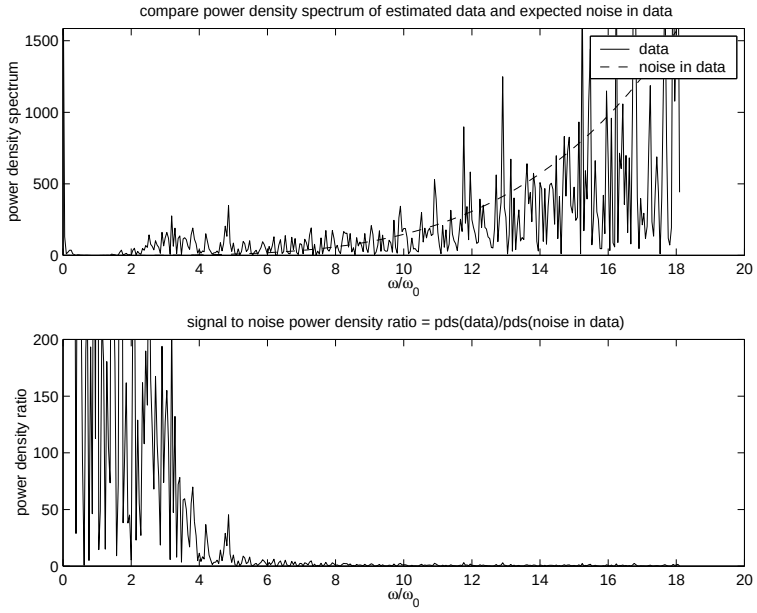

Figure 6.22: Top plot: the amplitude of the discrete Fourier transform (DFT) of the displacement; Bottom plot: the amplitude squared.

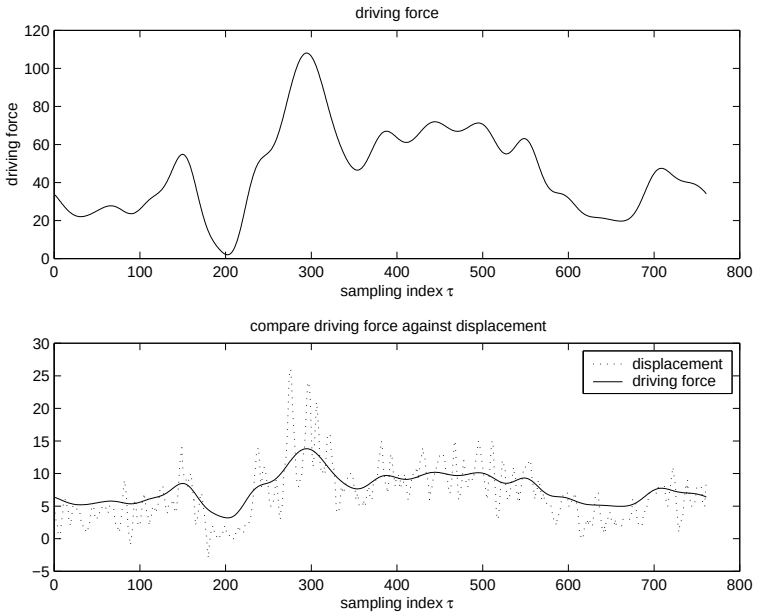

Figure 6.23: Top plot: the large scale fluctuation of the estimated force parameter; Bottom plot: comparing the large scale fluctuation of the estimated force parameter to the observed displacement.

Table 6.3: Comparing the parameters of the observed plant branch and the synthesized plant branch for the trees-by-the-lake example.

	observed plant branch	synthesized plant branch
natural frequency corresponds to ν	21	20
damping corresponds to $\Delta\nu$	1.1818182	1
velocity scaling factor κ (relative)	1	1.1
view point to object scaling factor $c_{distance}$ (relative)	1	0.8
initial displacement		0
initial velocity		0

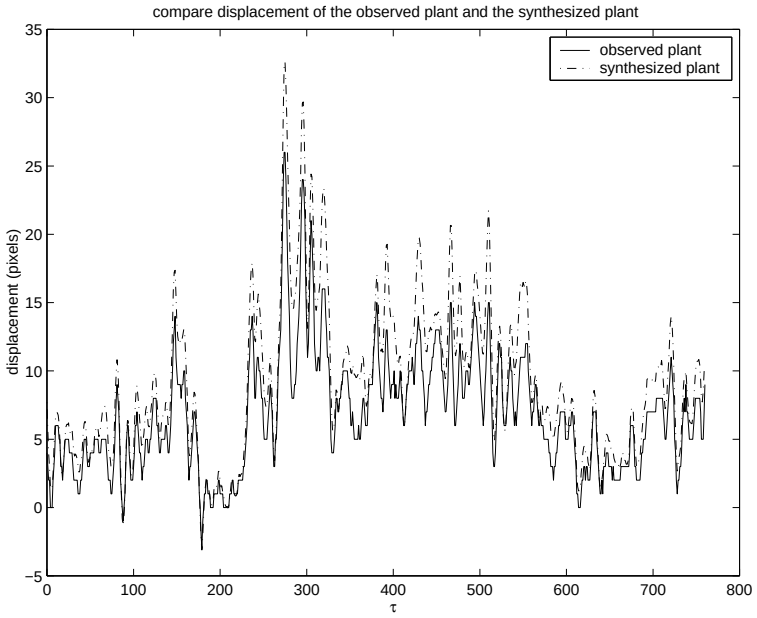

Figure 6.24: Comparing the displacement of the synthesized plant to the displacement of an observed plant.

http://www.dgp.toronto.edu/~meng/animation

Figure 6.25: Adding a computer synthesized shrub in the scene and letting it sway alongside the trees in the video sequence.

Table 6.4: Sampling parameters for the tree-on-the-street video sequence.

Number of sample points N	1047
Sampling period T (second)	$\frac{1}{6}$

6.5.2 Example: Wind and Snow

In this section, we show an example of using the wind velocity parameter we estimated to drive the animation of snow in the foreground.

The input video sequence is the same example we used earlier, as shown in Figure 6.16, when we discussed the feature tracking process. We will refer to this video as the "tree-on-the-street" example.

The sampling parameters for the original video sequence are shown in Table 6.4.

6.5.2.1 Wind Velocity Parameter at One Location

The analysis we have talked about so far would allow us to estimate the wind velocity parameter for the small local region around one branch of a tree or plant over time.

The data analysis process is similar to the trees-by-the-lake example. For the current example, the estimated force fluctuation is plotted in Figure 6.26. The velocity is estimated within a scaling factor, and it is plotted in Figure 6.27.

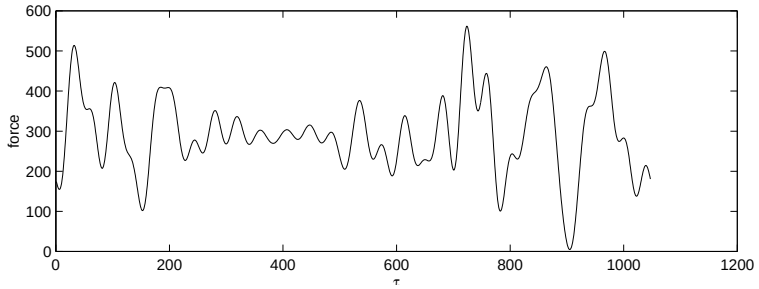

Figure 6.26: The estimated wind force parameter for the tree-on-the-street example.

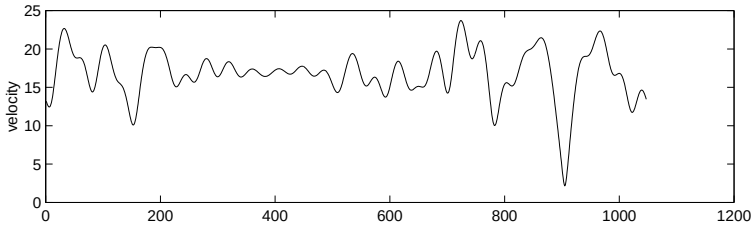

Figure 6.27: The estimated large scale wind velocity within a scaling factor for the tree-on-the-street example.

6.5.2.2 Wind Movement in Space

So far, we have obtained an estimate of the wind velocity parameter at a small local region around the branch we observed. We would like to extrapolate the wind field at the vicinity of the observed object. From a single view point, we do not have an estimate of the distance between the view point and the object. We decided instead to generate snow the region around the image plane, as shown in Figure 6.28, then project the snow on to the image plane. This is equivalent to having an uniformly scaled miniature version of the real world. To blow the snow around, we would like to establish a wind velocity estimate $V_{wind}(x, y, z, \tau)$ for every point in the three dimensional (3D) miniature space, where τ is the sampling index which corresponds to the time $t = \tau T$.

For our example, we make some observations that help us build a simplified wind movement model:

- The wind field is clearly moving from left to right across the image plane and the air moves parallel to the ground.

- We could do an elaborate aerodynamic simulation of the virtual snow particles under the influence of the wind. However, the tree we observed occupies most of the screen space. The wind we captured is strong enough that an air particle near

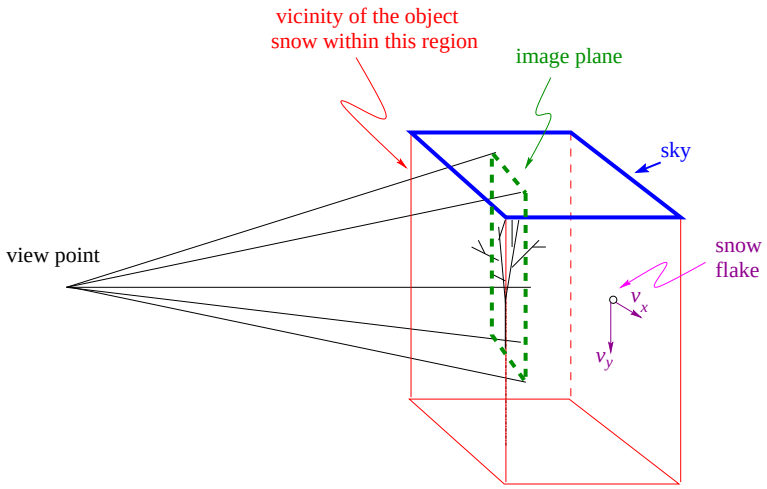

Figure 6.28: Generating snow in the vicinity of the image plane.

the tree passes by the tree within a fraction of a second. Therefore, we decided that a simple naive model of air flow might be sufficient. An elaborate aerodynamic simulation may not improve the visual effect significantly when the lifespan of any particle on a screen lasts for only a fraction of a second.

Based on these assumptions, we use a simple method to assign velocity to each location at a given time. Let $V_{parameter}(\tau)$ be the wind velocity parameter we estimated and is shown in Figure 6.27. We know that, as shown in Figure 6.18,

$$V_{parameter}(\tau) = \sqrt{\frac{c_{shape}}{c_{distance} w_{pixel}}} \kappa V, \tag{6.31}$$

where V is the x component of the air's velocity as it passes by the observed tree top. This velocity is projected onto the image plane. The corresponding velocity we observed along the x axis of the image plane is V_{image_plane} and

$$V = c_{distance} V_{image\_plane}.$$

So Equation 6.31 can be rewritten as

$$V_{parameter}(\tau) = \sqrt{\frac{c_{shape}c_{distance}}{w_{pixel}}}\kappa V_{image\_plane}(\tau). \tag{6.32}$$

Since both $c_{distance}$ and κ need to be estimated, we will simplify things by using just one parameter c_{wind}. Let

$$c_{wind} = \frac{1}{\sqrt{c_{shape}c_{distance}}\kappa},$$

and we have

$$V_{image\_plane}(\tau) = c_{wind}\sqrt{w_{pixel}}V_{parameter}(\tau).$$

In the 2D image plane, the average velocity of air particles passing by the observed tree top is

$$\bar{V}_{image\_plane} = c_{wind}\sqrt{w_{pixel}}\bar{V}_{parameter}.$$

In the synthesized 3D space, near the image plane, let the slab $x = 0$ experience wind fluctuation $V_{image\_plane}(\tau)$ at time $t = \tau T$,

$$V_{wind}(x = 0, y, z, \tau) = V_{image\_plane}(\tau).$$

The slab x experiences the same wind fluctuation with a time delay of

$$\Delta t = \Delta\tau \cdot T = \frac{x}{\bar{V}_{image\_plane}}.$$

Hence,

$$V_{wind}(x, y, z, \tau) = V_{image\_plane}(\tau - \Delta\tau) = c_{wind}\sqrt{w_{pixel}}V_{parameter}(\tau - \frac{x}{\bar{V}_{image\_plane}T}).$$

Values between sampled points are interpolated.

6.5.2.3 Snow Generation

We shall now place computer synthesized snow into a windy scene. We generate snow using a particle system. The system has the following components:

- The sky is modelled by a plane as shown in Figure 6.29. Each new particle is generated at a random location on this plane.
- The particle system has a particle birth rate to control how many snow particles are generated per second.
- Since each snow flake is very light, for its downward motion, we assume the y-component of its velocity is a constant v_y, with no acceleration.
- For each snow flake's sideways motion, the x-component of its velocity $v_x(x, y, z, \tau)$ is assumed to be the same as the wind velocity $V_{wind}(x, y, z, \tau)$ at that location,

$$v_x(x, y, z, \tau) = V_{wind}(x, y, z, \tau). \tag{6.33}$$

To obtain a reasonable visual effect of the snow moving with the wind, an animator using such a system only needs to adjust c_{wind}, so that the wind does not appear to be

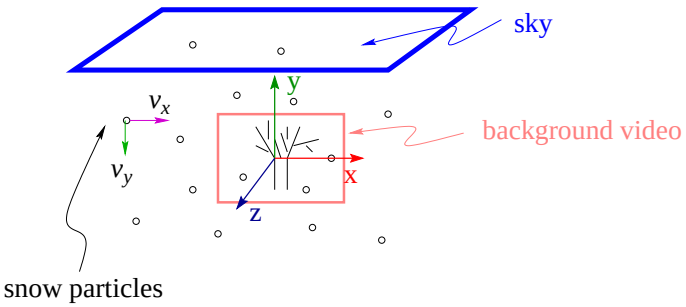

Figure 6.29: The setup for the snow particle system.

too weak or too strong with respect to the background video. Figure 6.30 shows one frame of the resulting video of adding snow to the example background video.

http://www.dgp.toronto.edu/~meng/animation

Figure 6.30: Using the estimated wind velocity parameters to drive the falling snow animation.

Chapter 7

The Regular Water Wave Elevation Problem

In this chapter, we describe our experiments in which we estimate surface water wave elevation parameters by observing a sailboat's motion. Then we will use the these parameters to drive computer synthesized boats. Often we see boats anchored by the harbor leisurely rocking back and forth. People might wonder what caused the boats to heave and sway. It turns out the culprit is the regular water wave.

Regular water waves are unlike wind currents. A regular water wave propagates as a wave, but each individual water particle does not go far. Each water particle merely moves in an elliptical or circular orbit about a local neighbourhood, which we will discuss in more detail in section 7.2.1. Whereas, for wind currents, the air particle does not just orbit about a local neighbourhood. The air particles actually move along with the current. Hence, regular waves are different from wind currents.

The regular water wave's interaction with a boat produces a change of buoyancy

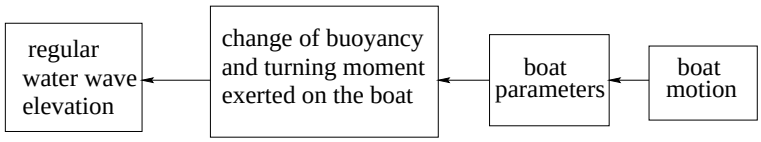

Figure 7.1: Use observed boat motion to estimate regular water wave elevation.

Figure 7.2: An example frame from a video of sailboats doing rolling motion in the water.

force or turning moment exerted on the boat, and this causes the boat's heaving and swaying motion, as shown in Figure 7.1. Since these motions can be approximated by harmonic oscillations, we can use a boat's motion to estimate parameters for its change of buoyancy and turning moment by using our basic modelling approach for the inverse harmonic oscillation problem. However, that is only half of the story. Since regular waves are fundamentally different from wind currents, "reverse engineering" the regular water wave elevation from observed and inferred parameters poses a brand new challenge. This subclass of problems can serve as yet another good example of our use of harmonic oscillation for model extraction.

7.1 Input

Consider an example input to be a sailboat anchored at shore, as shown in Figure 7.2. For the purpose of demonstrating the principle of our approach, we will simplify the

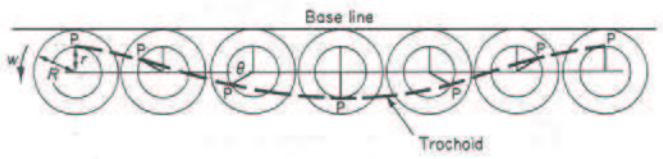

Figure 7.3: Trochoidal water wave. (Courtesy of Muckle and Taylor [MT87])

problem by just considering the video sequence where

- the camera's main axis is parallel to the boat's longitudinal axis for rolling motion,
- or the camera's main axis is parallel to the boat's transverse axis for pitching motion.

7.2 Background: Boats in Regular Water Waves

In this section, we will go over some background information about regular water waves and boat-water interactions.

7.2.1 Regular Water Waves

The periodic nature of the motion of water in ocean waves suggests the possibility of a simple mathematical treatment [RC42]. The water particles in a wave rotate in elliptical or circular orbits, the plane of which is vertical and perpendicular to the line of wave crests, as shown in Figure 7.3. This is the underlying basis for the trochoidal theory of water waves [RC42].

For most engineering purposes, the trochoidal wave and sine wave differ little in profile. Engineers normally assume the profile of the water wave as sine waves in the interest of a convenient mathematical treatment. When the water wave is travelling in

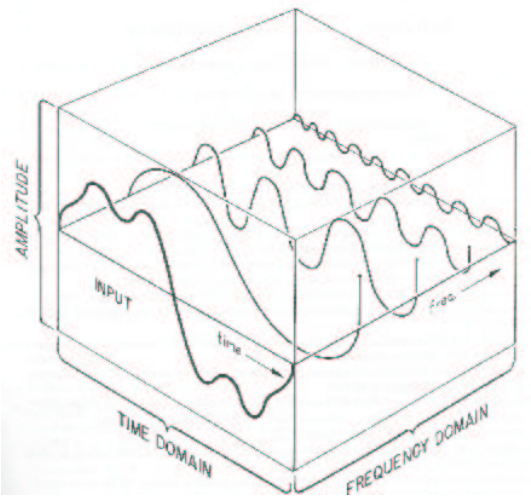

Figure 7.4: Surface wave profile is a sum of a series of sine progressive wave at different frequency. (Courtesy of Gillmer and Johnson [GJ82].)

one direction, say, along the axis x_w, the surface wave profile ζ is the sum of orthogonal progressive sine waves:

$$\zeta = \sum_{i=0}^{\infty} a_i \cos(k_{wi} x_w - \omega_i t - \Delta_i) \tag{7.1}$$

This is as shown in Figure 7.4. The sine wave frequency $\omega_i = i\omega_1$ of the i-th term is a multiple of the fundamental frequency ω_1. The wave number k_{wi} is

$$k_{wi} = \frac{2\pi}{\lambda_i} = \frac{\omega_i^2}{g},$$

where λ_i is the corresponding wavelength, g is the gravitational acceleration, and Δ_i is the wave phase delay. Interested readers are referred to Rossel and Chapman [RC42], and Bhattacharyya [Bha78] for details.

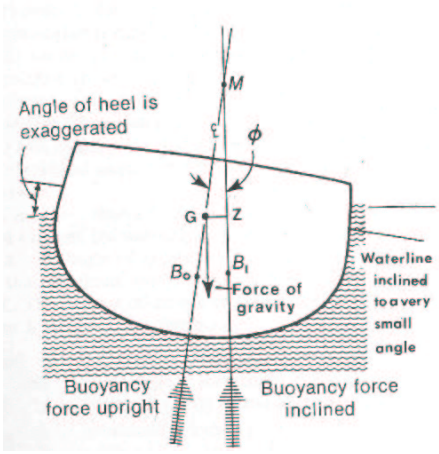

Figure 7.5: Basic forces on a boat: gravitational force and buoyancy. (Courtesy of Gillmer and Johnson [GJ82].)

7.2.2 Basic Forces Acting on a Boat

There are two basic forces acting on a boat floating in water: the force of gravity and the force of buoyancy. The gravity acts vertically downwards through the centre of gravity of the boat, as shown in Figure 7.5. The force of buoyancy acts vertically upward through the centre of buoyancy, where the resultant of all of the buoyancy forces is considered to be acting. The centre of buoyancy is the geometric centroid of the immersed portion of the boat's hull. When the ship is heeled, the shape of the underwater body is changed, thus moving the position of the centre of buoyancy.

The Principle of Archimedes states that, for a wholly or partially submerged free body in a fluid at rest, the gravitational force $m_{boat}g$ is exactly equal to the buoyancy force $\sigma_w g \bigtriangledown$, i.e.,

$$m_{boat}g = \sigma_w g \bigtriangledown,$$

where m_{boat} is the mass of the boat, $\bigtriangledown$ is the volume of the immersed portion of the boat's hull, σ_w is the density of the fluid and g is the gravitational force.

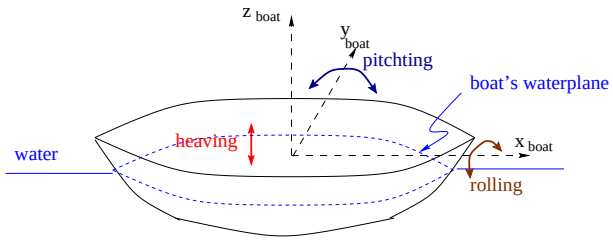

Figure 7.6: Heaving, pitching and rolling of a boat.

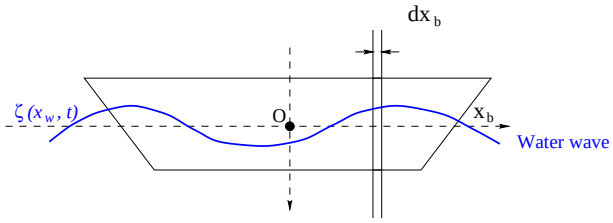

Figure 7.7: Heaving motion of a boat due to water wave.

A ship moving on the surface of open water is almost always in oscillatory motion [Bha78]. There are six different kinds of motions that a ship experiences, three linear and three rotational about its three principal axes. Only three kinds of motion, namely, heaving, rolling, and pitching, are purely oscillatory motions, since these motions act under a restoring force or moment when the ship is disturbed from its equilibrium position. They are shown in Figure 7.6. In this chapter, we will study the three types of oscillatory motions.

7.2.3 Heaving Motion

Heaving refers to a boat's vertical up and down motion. The water waves are the source of boat excitation. The exciting force for heaving can be calculated by assuming that the ship remains still as far as vertical motion is concerned and the waves pass slowly by the ship, as shown in Figure 7.7. Let x_b be the longitudinal axis of the boat. Let x_w be the direction of wave propagation as in Eq. 7.1. We will consider an example where

the boat's longitudinal axis x_b is parallel to the wave direction of propagation x_w. In this case, for simplicity, we assume that these axes coincide, so that $x_w = x_b$. Let $w_b(x_b)$ be the width of the boat. The additional buoyancy $df(x_b)$ on a section of a ship of unit length dx_b at x_b due to the surface wave profile $\zeta(x_w, t)$ is as shown in Figure 7.8(a):

$$df(x_b) = \sigma_w g w_b(x_b) \zeta dx_b. \tag{7.2}$$

In this figure the additional buoyancy due to the wave is negative, because this block is in the wave trough. The exciting force F for the boat's heaving motion can be obtained by integrating such additional buoyancy along the ship:

$$\textit{exciting force} = F = \sigma_w g \int_{-L_b/2}^{L_b/2} w_b(x_b) \zeta dx_b \tag{7.3}$$

where L_b is the length of the boat, $w_b(x_b)$ is the width of the boat. For a simple surface wave profile with angular frequency ω, i.e., for the case where

$$\zeta = a \cos(k_w x_b - \omega t - \Delta), \tag{7.4}$$

the heaving exciting force is

$$F = F_0 \cos(\omega t + \Delta) = \sigma_w g a \int_{-L_b/2}^{L_b/2} w_b(x_b) \cos(k_w x_b) dx_b \cos(\omega t + \Delta). \tag{7.5}$$

The *restoring force* in heaving can be attributed to the boat's vertical movement. In most engineering analysis, we assume the side of the boat as wall-sided, i.e., they are

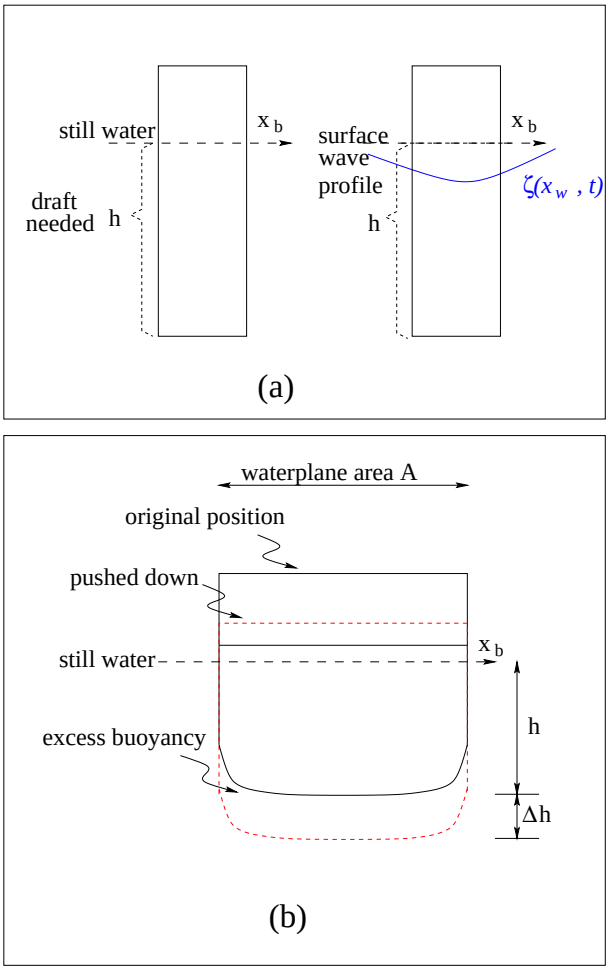

Figure 7.8: (a) Heaving excitation force on a buoy due to water wave, assuming boat is fixed and water is moving. (b) Heaving restoring force due to excessive buoyancy, assuming boat is moving and water is not.

parallel to the boat's upright axis. If we assume now that the boat is moving and the water is still instead, in Figure 7.8(b), the boat shown in solid line is in its the equilibrium position, where the gravitational force $m_{boat}g$ is exactly equal to the buoyancy force $\sigma_w g \bigtriangledown$, where $\bigtriangledown$ is the volume of the immersed portion of the boat's hull. The dotted lines show the boat heaved into the water an amount Δh. The increase in buoyancy is $\sigma_w g A \Delta h$, where the boat's water plane area A is assumed to be constant for the wall-sided boat. This force will tend to restore the boat to its original position. The case when the boat

heaved upwards is similar.

The inertial force for heaving is

$$m_{virtual}\frac{d^2\Delta h}{dt^2}.$$

Here, $m_{virtual}$ is a virtual mass which is the sum of the mass of the boat m_{boat} and the *added mass* m_{added} of the water accelerated by the heaving motion of the boat:

$$m_{virtual} = m_{boat} + m_{added}.$$

The added mass due to the entrained water is usually expressed as a percentage of the mass of the boat. In marine engineering, this percentage is normally determined experimentally for a given boat.

The damping force is

$$m_{virtual}\gamma_{heave}\frac{d\Delta h}{dt},$$

where γ_{heave} is the damping coefficient.

For heaving motion, the change of draft Δh due to the exciting force F can be described using Newton's equation of motion:

$$m_{virtual}\frac{d^2\Delta h}{dt^2} + m_{virtual}\gamma_{heave}\frac{d\Delta h}{dt} + \sigma_w g A\Delta h = F \tag{7.6}$$

It can be rewritten as

$$\frac{d^2\Delta h}{dt^2} + \gamma_{heave}\frac{d\Delta h}{dt} + {\omega_{0heave}}^2\Delta h = \frac{F}{m_{virtual}}, \tag{7.7}$$

where ω_{0heave} is the natural frequency the heaving motion of the boat, and ${\omega_{0heave}}^2 = \frac{\sigma_w g A}{m_{virtual}}$.

7.2.4 Pitching and Rolling Motion

Rolling is a boat's angular motion about the longitudinal axis, and pitching is a boat's angular motion about the transverse axis, as shown in Figure 7.6. They can be modelled using the same approach. Without loss of generalization, in this section, we will use the pitching motion as our example.

The equation of motion for pitching is [Bha78]

$$I_p\frac{d^2\alpha}{dt^2} + I_p\gamma_p\frac{d\alpha}{dt} + m_{boat}g\overline{GQ_M}\alpha = M_p. \tag{7.8}$$

Here, α is the rotational angle from the upright position due to pitching, I_p is the mass moment of inertia of the boat about its transverse rotational axis, so

$$inertia\ moment = I_p\frac{d^2\alpha}{dt^2},$$

and

$$damping\ moment = I_p\gamma_p\frac{d\alpha}{dt}.$$

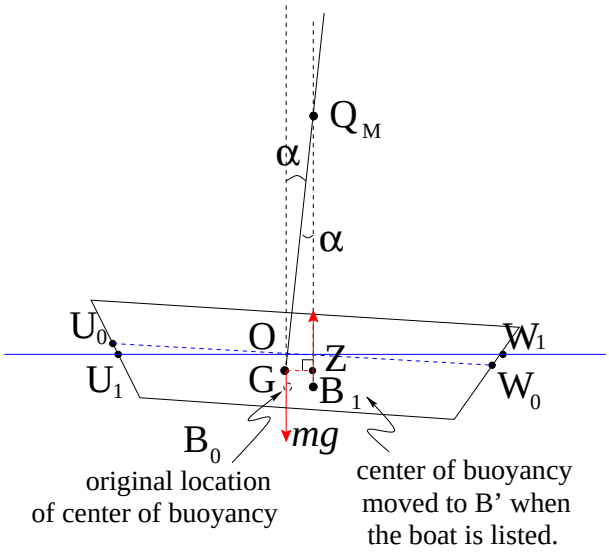

Figure 7.9: The restoring moment in pitching.

The *restoring moment* is due to the boat's inclination angle α [MT87]. As shown in Figure 7.9. The inclination affects the shape of the underwater form and this in turn causes the position of the centre of buoyancy to move from B_0 to B_1. This arises because a volume U_0OU_1 has come out of the water and an equal volume W_0OW_1 has gone into the water to maintain equilibrium. Now, the buoyancy acts upwards through B_1. If the vertical through B_1 is continued upwards, it intersects the original vertical at a point which is denoted by Q_M in the diagram. This point Q_M is called the *metacentre* and for small angles of inclination it is fixed in position. The inclination does not affect the position of G, the centre of gravity, unless there is some weight which is free to move. Now, the weight $m_{boat}g$ acting downwards and the buoyant force of equal magnitude acting upwards are not in the same straight line, but form a couple of magnitudes $m_{boat}g\overline{GZ}$, where $\overline{GZ}$ is the perpendicular on $\overline{B_1Q_M}$ drawn from G. The couple will restore the

body to its original position. Here,

$$\overline{GZ} = \overline{GQ_M} \sin\alpha \approx \overline{GQ_M}\alpha, \tag{7.9}$$

when α is small. The line $\overline{GZ}$ is called the *righting arm* and $\overline{GQ_M}$ is called *metacentric height*. So

$$\textit{restoring moment} = m_{boat} g\overline{GZ} = m_{boat} g\overline{GQ_M}\alpha, \tag{7.10}$$

The *exciting moment* for pitching is due to the unbalanced moment caused by the waves about the transverse axis of the boat. We will consider an example where the boat's longitudinal axis is parallel to the wave's direction of propagation. Once again, in this case, we let $x_w = x_b$. The wave's profile $\zeta(x_w, t)$ is defined in Eq. 7.1. Consider a narrow strip of the boat with width dx_b, as shown in Figure 7.10. The excess buoyancy $df(x_b)$ experienced by this strip due to the wave is as shown in equation 7.2. The resulting turning moment dM_p around the boat's transverse axis would be

$$dM_p(x_b) = df(x_b)x_b = \sigma_w g w_b(x_b)\zeta x_b dx_b.$$

Therefore, we can integrate the contribution from each strip along the boat to obtain the exciting moment:

$$\textit{exciting moment} = M_p = \sigma_w g \int_{-L_b/2}^{L_b/2} w_b(x_b)\zeta x_b dx_b. \tag{7.11}$$

For a simple surface wave profile with angular frequency ω, i.e., for the case where

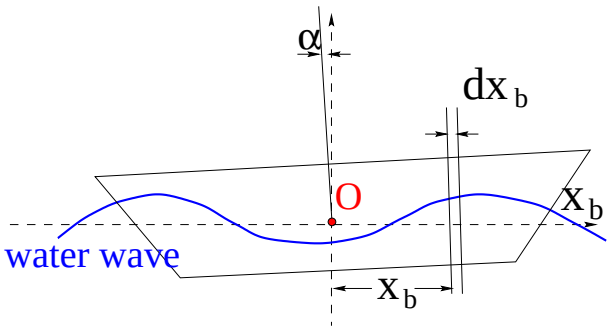

Figure 7.10: Pitching motion of a boat due to water wave.

$$\zeta = a\cos(k_w x_w - \omega t - \Delta), \tag{7.12}$$

the pitching exciting moment is

$$\begin{aligned} M_p &= M_{p_\omega}\sin(\omega t + \Delta) \\ &= [\sigma_w g a \int_{-L_b/2}^{L_b/2} w_b(x_b)\sin(k_w x_b)x_b dx_b]\sin(\omega t + \Delta), \end{aligned} \tag{7.13}$$

where the amplitude of the exciting moment is

$$M_{p_\omega} = a\sigma_w g \int_{-L_b/2}^{L_b/2} w_b(x_b)\sin(k_w x_b)x_b dx_b. \tag{7.14}$$

7.3 Input and Assumptions

We observed sailboats anchored by the shore. We videotaped a sailboat with dominant rolling motion with the forward axis of the camera perpendicular to the rolling motion. In case of dominant pitching motion, we had the camera's forward axis perpendicular to the pitching motion, as shown in Figure 7.11. Since the harbor is a relatively sheltered environment, heaving motion is not prominent enough to be analyzed, but the data for

Figure 7.11: Boats in pitching motion.

pitching was measurable and was thus useful for validating our model.

7.4 Process

In this section, we describe our experiments where we estimate surface water wave elevation by observing a sailboat's motion.

7.4.1 Scan

As with our process for detecting tree displacement due to wind, the video input feature tracking process is automatically done using a software implemented by Jepson, Fleet and El-Maraghi using algorithms based on Shi and Tomasi [ST94], Jenkin and Jepson [JJ94], Jepson and Black [JB93], Jepson and Jenkin[JJ89], Jepson, Fleet and El-Maraghi [JFEM01], El-Maragi [EM02].

For a rolling or pitching video sequence, we decide on a sailboat we want to observe. We want to monitor the inclination of a sailboat by tracking two points of its mast: the top of the mast P_1 and the base of the mast P_2, as illustrated in Figure 7.12. Figure 7.13 shows one frame of the tracking result for the rolling motion example. The centres the the ellipses are the points tracked by the tracking software.

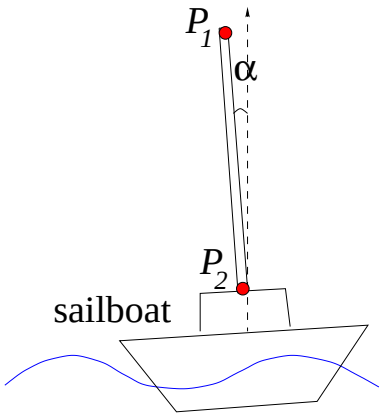

Figure 7.12: Observing the motion of the sailboat.

http://www.dgp.toronto.edu/~meng/animation

Figure 7.13: One frame of the tracking result.

We assume the image's vertical axis is parallel to the upright position of the mast of the sailboat. So the line $\overline{P_1P_2}$ and the image's vertical axis form an angle α, which is the inclination of the sailboat in the viewing plane. Since when the boat is experiencing a pitching dominated motion, we videotaped it with the camera's forward axis perpendicular to the pitching motion, in which case α would be the pitching inclination. Similarly, for rolling, the camera setup would ensure that α is the rolling inclination.

7.4.2 Modelling Pitching

The pitching motion is approximated by a harmonic oscillation. We can rewrite its equation of motion (Equation 7.8) in a form similar to the one for simple spring-mass

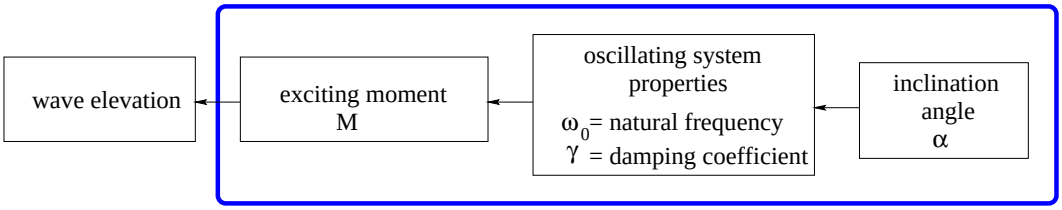

Figure 7.14: Obtain exciting moment by observing the listed angle.

system:

$$\frac{d^2\alpha}{dt^2} + \gamma_p \frac{d\alpha}{dt} + \omega_{0_p}^2 \alpha = \frac{M_p}{I_p}, \tag{7.15}$$

where the natural frequency for the pitching motion is $\omega_{0_p}^2 = m_{boat} g \overline{GQ_M} / I_p$. We can use the basic approach discussed in section 5 to obtain the exciting moment $M_p(t)$ for pitching by analyzing the pitching inclination $\alpha(t)$, as shown in Figure 7.14.

As we can see from equations 7.12 and 7.13, a surface water wave at frequency ω_i can cause pitching exciting moment only at frequency ω_i. Similar to our treatment of converting wind force to wind velocity, we can express both the surface wave profile ζ and the exciting moment M_p as Fourier series expansions.

By Equation 7.14, at each frequency ω_i, the relationship between the wave height a_i and the magnitude of exciting moment M_{p_i} is

$$a_i = \frac{M_{p_i}/I_p}{\frac{1}{I_p} \cdot \sigma_w g \int_{-L_b/2}^{L_b/2} w_b(x_b) \sin(k_{w_i} x_b) x_b dx_b} = \frac{M_{p_i}/I_p}{c_{p_i}}, \tag{7.16}$$

where

$$c_{p_i}(L_b, w_b, k_{w_i}, I_p) = \frac{\sigma_w g \int_{-L_b/2}^{L_b/2} w_b(x_b) \sin(k_{w_i} x_b) x_b dx_b}{I_p} \tag{7.17}$$

is a scaling factor. Note that we keep the form M_{p_i}/I_p as it is in order to emphasize the fact that Eq. 7.15 can only recover the turning moment up to a scaling factor.

In other words, from the inclination angle, we can recover M_{p_i}/I_p, i.e., the fluctuation of the turning moment at a frequency, not the magnitude of M_{p_i}. Let $[c_{p_i}]_{observed}$ be the scaling factor for the observed boat and $[c_{p_i}]_{synthesize}$ be the scaling factor for the synthesized boat. Since we are interested in using the wave elevation parameters to synthesize movement of other floating buoys, the relative value of the scaling factor $[c_{p_i}]_{synthesized}/[c_{p_i}]_{observed}$ is more relevant and useful than the absolute value of $[c_{p_i}] = [c_{p_i}]_{observed}$.

7.4.3 Modelling Rolling

The analysis of the image sequence of rolling motion is similar to that of pitching motion, except that the length and the width are switched in the analysis.

For rolling, let the transverse axis of the boat be the x-axis of the boat. Let the direction the wave travels be the x-axis of the wave. We consider the scenario where the boat's transverse axis is parallel to the direction in which the wave travels. Similar to our analysis of the pitching motion, we let the x-axes of the wave and the boat coincide. At each frequency ω_i, the relationship between the wave height a_i and the magnitude of exciting moment M_{ri} for rolling is

$$a_i = \frac{M_{ri}/I_r}{c_{ri}}, \tag{7.18}$$

where

$$c_{ri}(W_b, l_b, k_{wi}, I_r) = \frac{\sigma_w g \int_{-W_b/2}^{W_b/2} l_b(x_b) \sin(k_{wi} x_b) x_b dx_b}{I_r} \tag{7.19}$$

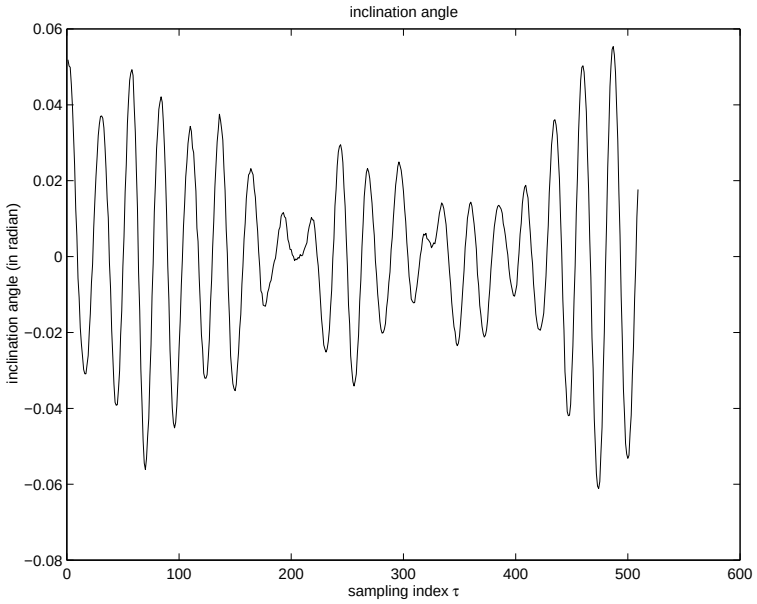

Figure 7.15: The inclination angle of the left most boat in the boat rolling example video sequence.

is a scaling factor for the rolling motion, W_b is the width of the boat at its widest point, $l_b(x_b)$ is the the length of the boat at x_b, and I_r is the boat's moment of inertia for rolling.

7.4.4 Synthesis

We want to use the water wave elevation parameters we extracted to drive the animation of the synthesized boat.

7.4.4.1 Rolling Example

Figure 7.2 shows one frame of an example video sequence of sailboats' rolling motion. Figure 7.15 shows the observed inclination angle α for the left most boat in the video sequence. The number of samples is $N = 509$. The sampling period is $T = \frac{1}{12}$ second. The discrete Fourier transformation of the inclination angle, $DFT(\alpha)$, is calculated. Figure 7.16 shows the square of amplitude of the frequency domain analysis, $|DFT(\alpha)|^2$.

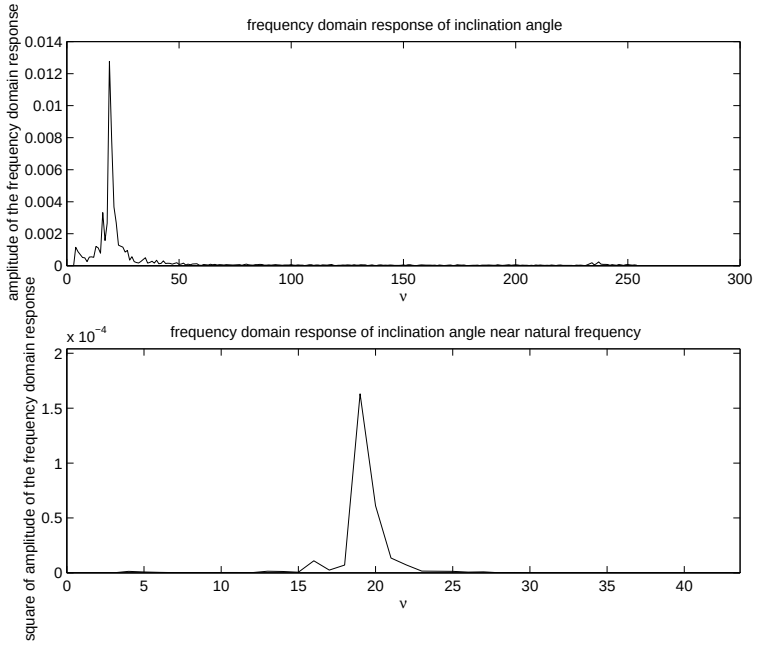

Figure 7.16: The frequency domain analysis of the inclination angle of the left most boat in the boat rolling example video sequence.

The natural frequency is estimated to be corresponding to $\nu_0 = 19$,

$$\omega_0 = \frac{2\pi\nu_0}{NT} = \frac{2\pi 19}{509 \cdot \frac{1}{12} \text{ second}} = 2\pi \frac{1}{2.23 \text{ second}},$$

and the damping coefficient is estimated to be $\Delta\nu = 1.21$.

The exciting moment parameter we extract is M_r/I_r, where M_r is the exciting moment for rolling and I_r is the moment of inertia for rolling. The amplitude of the exciting moment parameter M_r/I_r at the frequency domain is plotted in Figure 7.17. The point tracking result for the rolling sequence is accurate to $\frac{1}{10}$ of a pixel. We can assume that the tracking result might be off by $\frac{1}{10}$ of a pixel for P_1 and off by $\frac{1}{10}$ of a pixel for P_2, In

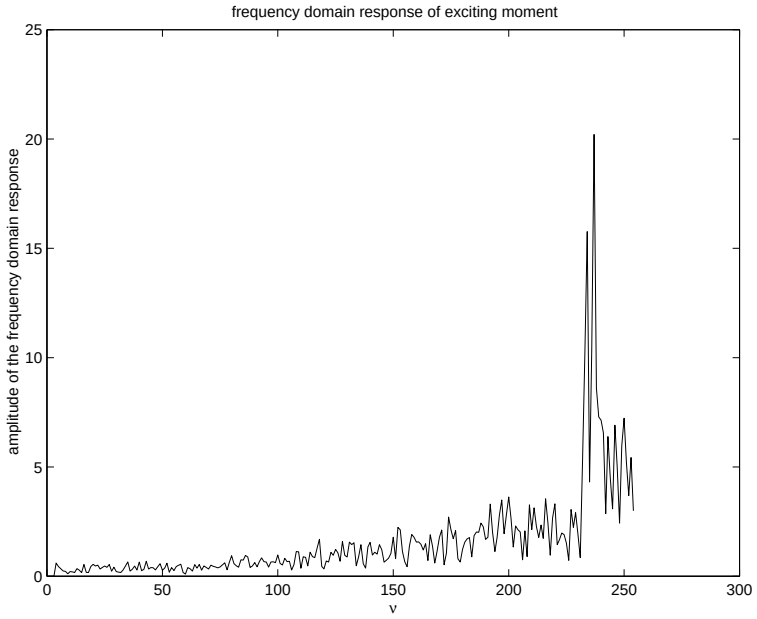

Figure 7.17: The frequency domain analysis of the exciting moment which caused the left most boat to roll in the boat rolling example video sequence.

total, the inclination angle estimation might be off by

$$\arctan(2 \cdot \frac{1}{10}/|P_1P_2|) \approx 2 \cdot \frac{1}{10}/|P_1P_2| = 2 \cdot \frac{1}{10}/159.2 \approx 0.001256 \text{ radian},$$

where $|P_1P_2| = 159.2$ is the length of the mast. We use 0.001256 radian as the standard deviation of the white noise in our measured inclination angle. Hence, this introduces noise in the extracted exciting moment parameter. The power density spectrum of the exciting moment parameter pds_{M_r/I_r} and that of the expected noise in it pds_n are plotted in Figure 7.18(top). The ratio of the two power density spectrums

$$\frac{\text{pds}_{M_r/I_r}}{\text{pds}_n}$$

is plotted in Figure 7.18(bottom). The frequency range where this ratio is above 50 is

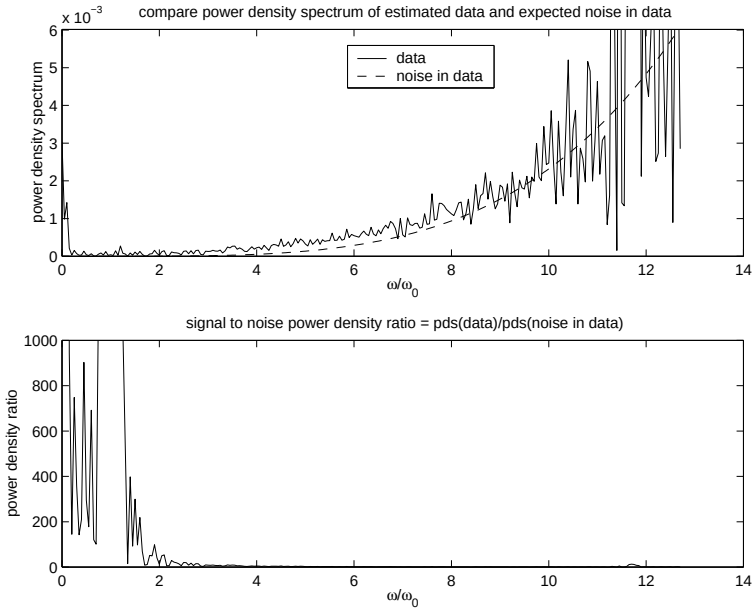

Figure 7.18: Expected white noise in the estimated exciting moment.

$\omega/\omega_0 = 0$ . . 1.6. Since the signal to noise power density spectrum ratio for the exciting moment parameter is high for only $\omega/\omega_0 = 0$ . . 1.6. We reconstruct the exciting moment parameter curve $M_{rreconstructed}$ using only the frequencies within this range. This curve is plotted in Figure 7.19(top). In Figure 7.19(bottom), we scale the exciting moment parameter curve and compare it to the observed inclination angle.

The parameters of the synthesized boat and the observed boat are tabulated in Table 7.1. We use a rectangular block to model the shape of the sailboats. We can measure the width of the observed boat in the video sequence and represent the width in terms of pixels. We assume the boat has a length to width ratio of 4. We use Equations 7.18 and 7.19 to compute the exciting moment parameter $(\frac{M_{ri}}{I_r})_{synthesized}$ for the computer synthesized boat:

$$[\frac{M_{ri}}{I_r}]_{synthesized} = [\frac{M_{ri}}{I_r}]_{observed} \frac{[c_{ri}(W_b, L_b, k_w(_i, I_r)]_{synthesized}}{[c_{ri}(W_b, L_b, k_{wi}, I_r)]_{observed}} \tag{7.20}$$

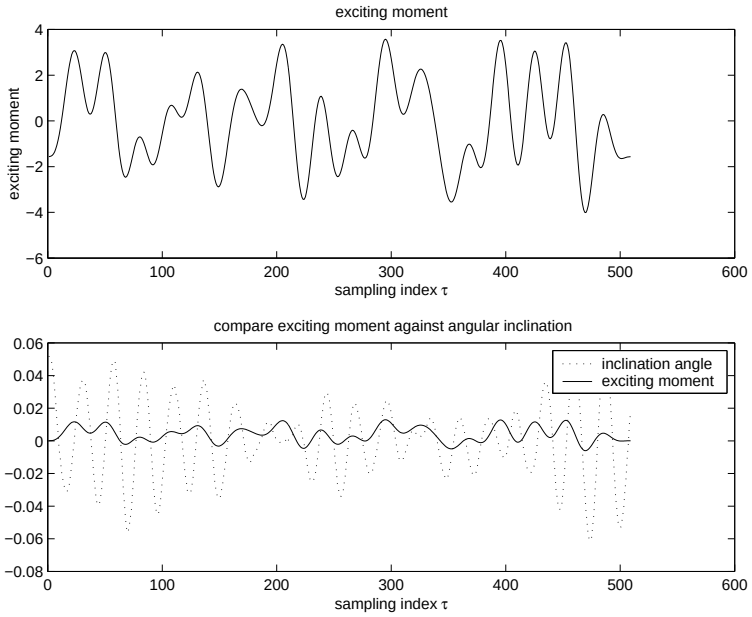

Figure 7.19: The exciting moment which caused the left most boat to roll in the boat rolling example video sequence.

Since the boat is modelled using a solid rectangular block, the moment of inertia for rolling is simply

$$I_r = \frac{m_{boat}}{12}({H_b}^2 + {W_b}^2). \tag{7.21}$$

The exciting moment for the computer synthesized boat is used to calculate its inclination angle for rolling. In Figure 7.20, we plot the inclination angle of the synthesized boat, and compare it to the inclination angle of the observed boat.

We place our computer animated sailboat in the captured video sequence, Figure 7.21 shows one frame of the resulting video sequence.

7.4.4.2 Pitching Example

In this section, we show an example of analyzing the pitching motion of a boat, then use the estimated regular water wave parameters to drive a computer synthesized boat.

Table 7.1: Comparing the parameters of the observed boat and the synthesized boat for the rolling motion example.

	observed boat	synthesized boat
boat width W_b (pixel)	55	55
boat length L_b (pixel)	220	220
boat height without mast H_b (pixel)	20	20
boat mass m_{boat} (relative to each other)	1	1
natural frequency corresponding to ν	19	21
damping corresponding to $\Delta\nu$	1.21	1.3
initial inclination angle		0
initial inclination angular velocity		0

Table 7.2: Sampling parameters for the pitching motion example video sequence.

Number of sample points N	1806
Sampling period T (second)	$\frac{1}{15}$

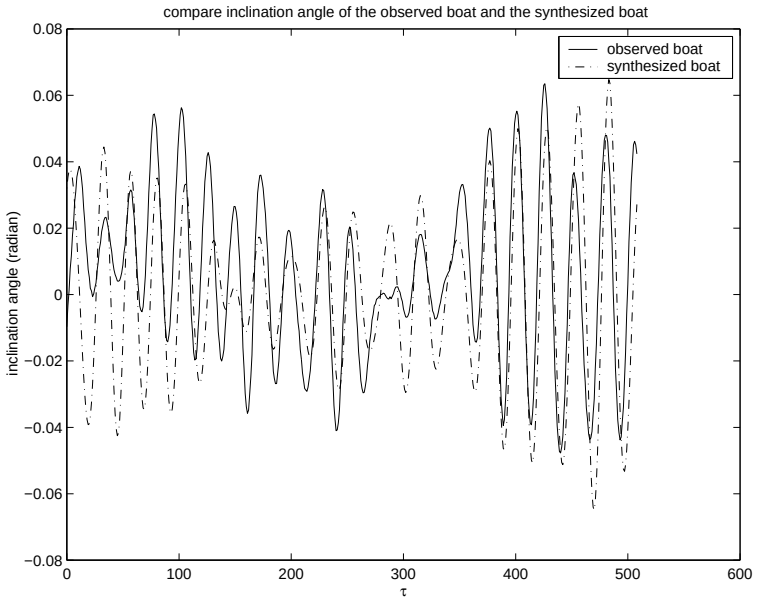

Figure 7.20: Comparing the inclination angle of the observed boat and the synthesized boat for the sailboat rolling motion sequence.

http://www.dgp.toronto.edu/~meng/animation

Figure 7.21: Putting a computer synthesized boat in the scene and letting it move alongside video captured sailboats.

Figure 7.22: An example frame from a video sequence of sailboats in pitching motion while anchored at the harbor.

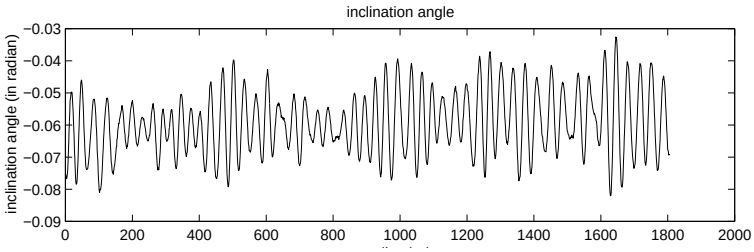

Figure 7.23: The inclination angle of the large boat in the boat pitching example video sequence.

Figure 7.22 shows an example frame from a video sequence of sailboats in pitching motion while anchored in the harbor. The sampling parameters for the pitching motion video are as shown in Table 7.2.

The data analysis process is similar to the rolling example. The plots of the data at each step are summarized below:

- Figure 7.23 shows the inclination angle of the large boat in the boat pitching example video sequence.

- Figure 7.24(top) shows the amplitude of the discrete Fourier transform of the observed inclination angle. The amplitude squared in Figure 7.24(bottom) is used for estimating the natural frequency and damping coefficient of the oscillation.

- Figure 7.25(top) compares the power density spectrums of the estimated exciting moment parameter and the expected noise. Figure 7.25(bottom) is the signal to noise power density spectrum ratio. This ratio is reasonably high for frequency range $\omega/\omega_0 = 0 \ . \ . \ 1.6$.

- Figure 7.26 is the exciting moment fluctuation curve we recovered.

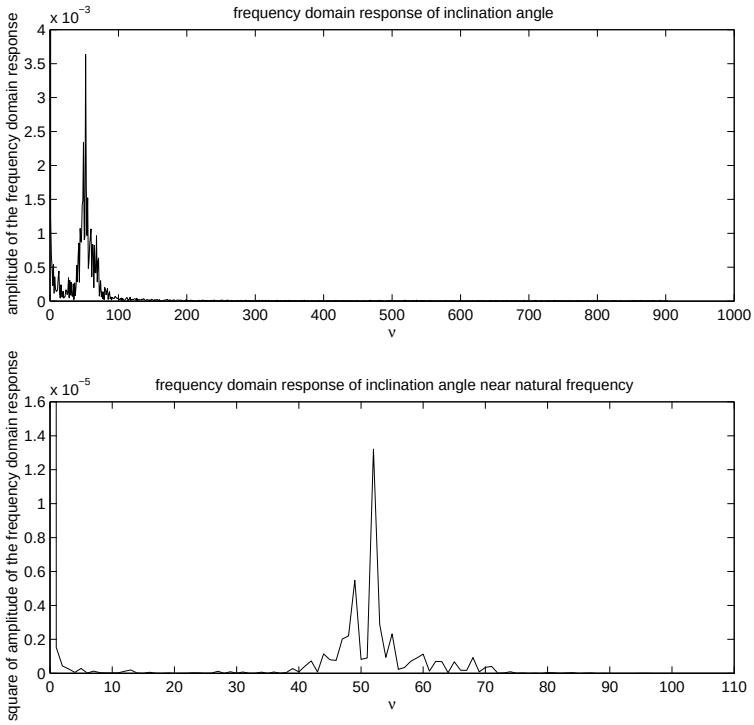

Figure 7.24: The frequency domain analysis of the inclination angle of the large boat in the boat pitching example video sequence.

Next, we make a computer synthesized boat. The parameters of the observed boat and the computer synthesized boat are compared in Table 7.3.

We compute the inclination angle for the synthesized boat's pitching motion. In Figure 7.27, we compare this inclination angle to that of the observed boat.

Finally, we place the synthesized boat in the background video. Figure 7.28 shows one frame of the resulting video sequence.

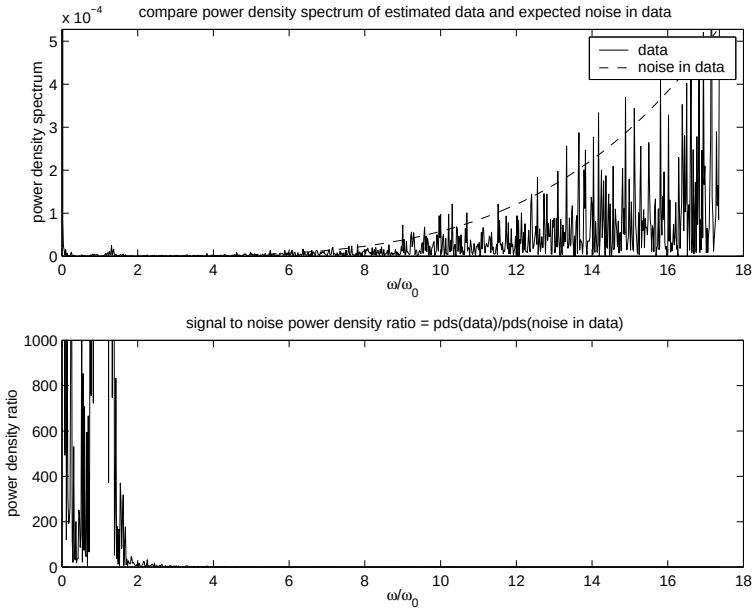

Figure 7.25: Comparing the power density spectrums of the estimated exciting moment parameter and the expected noise.

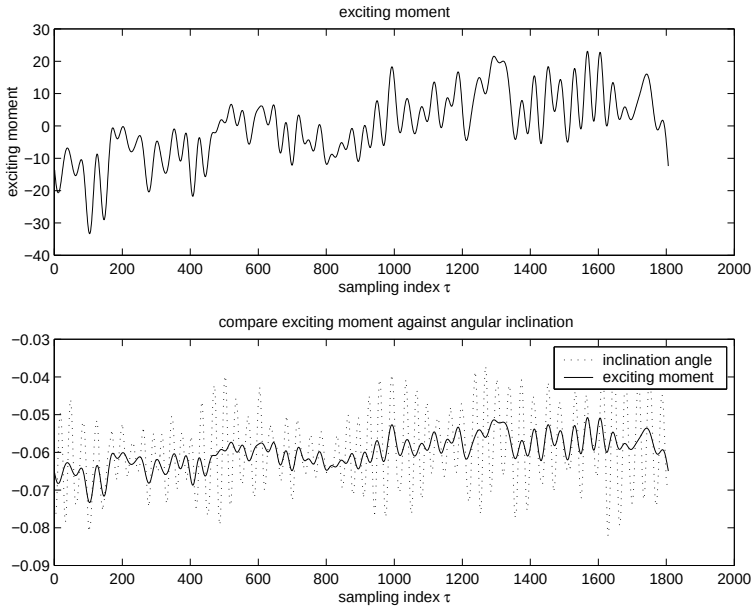

Figure 7.26: The exciting moment which caused the large boat to pitch in the boat pitching example video sequence.

Table 7.3: Comparing the parameters of the observed boat and the synthesized boat for the pitching motion example.

	observed boat	synthesized boat
boat width W_b (pixel)	69	55
boat length L_b (pixel)	276	220
boat height without mast H_b (pixel)	35	20
boat mass m_{boat} (relative to each other)	1	1
natural frequency corresponding to ν	52	50
damping corresponding to $\Delta\nu$	1.16279	2
initial inclination angle		0
initial inclination angular velocity		0

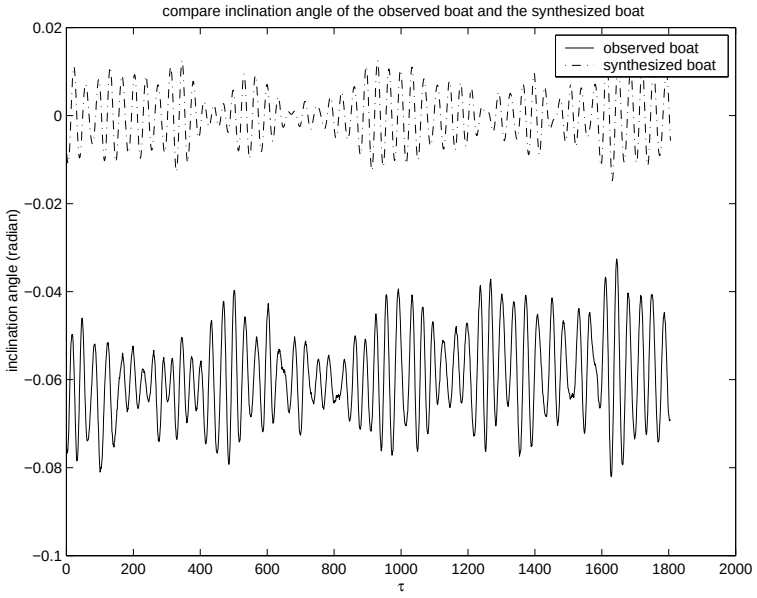

Figure 7.27: Compare the inclination angle of the observed boat and the synthesized boat for the sailboat pitching motion sequence.

http://www.dgp.toronto.edu/~meng/animation

Figure 7.28: Adding the computer synthesized boat in the scene, and animate it. This is an example frame from the resulting video sequence.

Chapter 8

Other Potential Errors

In Chapter 5, we talked about using white noise to account for measurement errors from the feature tracking software. There are other potential errors which could occur that would affect the accuracy of our model. For instance, we could have an error $\Delta\omega_0$ in the natural frequency estimation, and an error $\Delta\gamma$ in the damping coefficient estimation. Since we use ω_0 and γ to estimate the driving force f, as shown in Figure 8.1, errors in ω_0 or γ would introduce error into the estimated force.

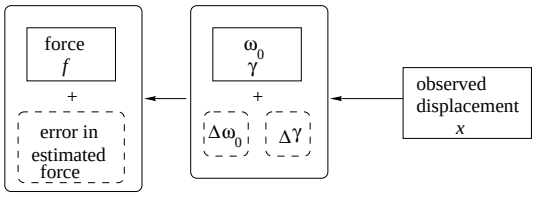

Figure 8.1: Error in natural frequency and damping coefficient estimations introduces error into the estimated force.

For our wind-tree interaction and water-boat interaction experiments, we do not know the actual values of the natural frequency and damping coefficient of the oscillating system. In this chapter, we will merely give a theoretical discussion on the effect of

the errors in ω_0 or γ on the accuracy of our driving force estimation, in particular, the amplitude of the estimated force in frequency domain.

In Chapter 4, Equation 4.13 shows that, given a driving force at frequency ω_i with amplitude f_i and phase delay Δ_i,

$$f_i \cos(\omega_i t + \Delta_i),$$

the resulting displacement x is

$$x_i = \rho(\omega_i) f_i \cos(\omega_i t + \Delta_i + \theta_i).$$

The amplitude of the response is the magnitude of the force multiplied by a magnification factor $\rho(\omega_i)$, where

$$\rho^2(\omega) = \frac{1}{m^2[(\omega^2 - \omega_0^2)^2 + \gamma^2\omega^2]}.$$

Hence, to study the effect of errors in natural frequency and damping estimate on the estimation of driving force amplitude parameter, we only need to investigate how they affect the relative error in $\rho(\omega_i)$.

8.1 Effect of Relative Error in Natural Frequency

First, assuming there is no error in the damping coefficient estimation, we would like to know how the relative error in the estimated natural frequency ω_0 would cause relative error in the magnification factor ρ.

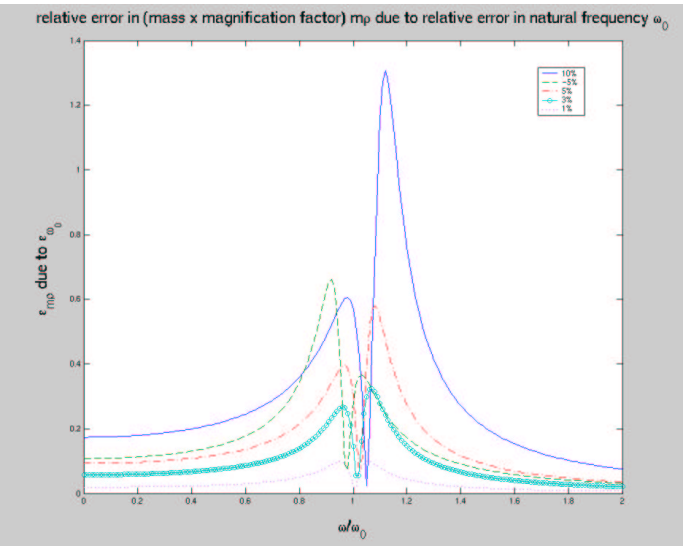

Figure 8.2: Relative error $\varepsilon_{m\rho}$ in magnification factor ρ due to the relative error ε_{ω_0} in natural frequency ω_0 when $\gamma/\omega_0 = 0.1$.

Figure 8.2 is a plot of the relative error in ρ due to the relative error in ω_0. Note that since the mass m is a constant scale factor, the relative error in $\rho(\omega_i)$ is the same as the relative error in $m\rho(\omega_i)$.

Figure 8.2 shows that, when $\gamma/\omega_0 = 0.1$, a 10% relative error in ω_0 can result in large error near the true natural frequency. This is not surprising. Intuitively, as shown in Figure 8.4, when the estimated natural frequency deviates from the actual natural frequency by a factor of 10%, we do expect the relative error in ρ to be somewhat large. The relative error in ρ gets smaller as ω moves further away from ω_0. Figure 8.3 shows that under a similar constraint, when γ/ω_0 is larger, in this case, $\gamma/\omega_0 = 0.2$, the relative error ρ is smaller. Figure 8.2 also shows that as the relative error in ω_0 decreases to 5%, 3% and 1%, the relative error in ρ becomes more reasonable, even when ω is near ω_0.

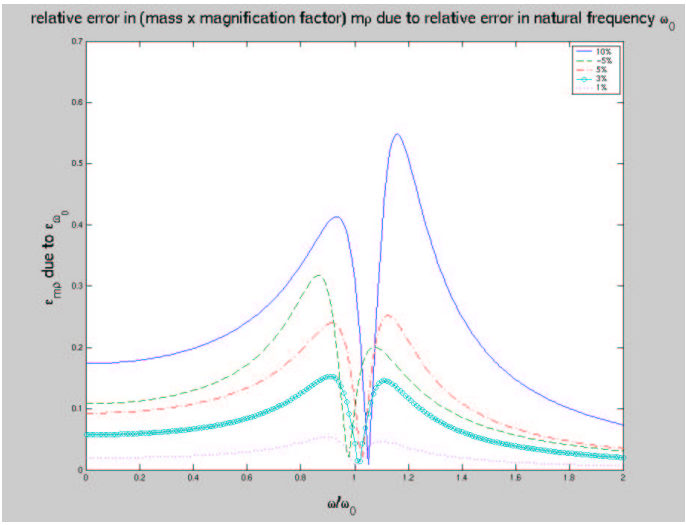

Figure 8.3: Relative error $\varepsilon_{m\rho}$ in magnification factor ρ due to the relative error ε_{ω_0} in natural frequency ω_0 when $\gamma/\omega_0 = 0.2$.

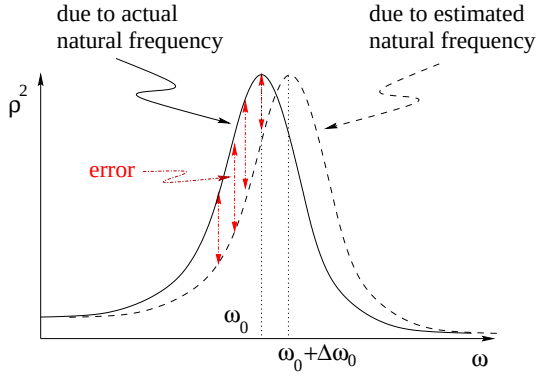

Figure 8.4: Error in the case where the natural frequency estimation is off by 10%. The error in ρ^2 is large near the actual natural frequency.

8.2 Effect of Relative Error in Damping Coefficient

Now, assuming that there is no error in the natural frequency estimation, we would like to know how the relative error in the estimated damping coefficient γ would cause relative error in the magnification factor ρ. Figure 8.5 is a plot of the relative error in ρ when the relative error in γ is 10%. This graph shows that when the relative error in γ is 10%, the resulting relative error in ρ is merely 9% near ω_0 and it gets smaller fairly quickly as ω

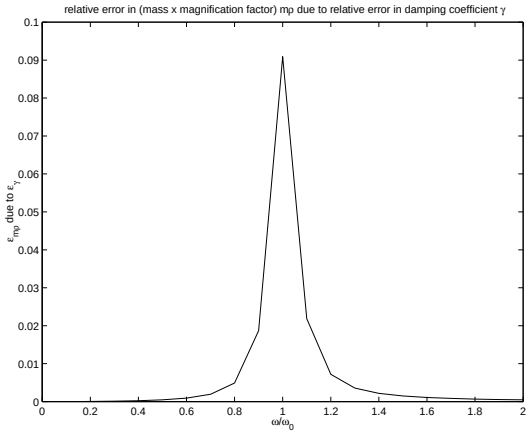

Figure 8.5: Relative error $\varepsilon_{m\rho}$ in magnification factor ρ due to the relative error ε_γ in damping coefficient γ.

becomes further away from ω_0. This tells us that the relative error in γ does not become amplified when we calculate ρ.

Chapter 9

Conclusion

9.1 Contribution

In this thesis, we proposed a novel way to indirectly drive computer animation using real life video input, which we called Video Input Driven Animation (VIDA). We observe the motion of objects in the video captured scene, extract and estimate useful parameters, and re-apply these parameters to drive the animation of computer generated objects/characters. An example application of this idea would be to use real life video as a background, and make computer generated objects/characters appear as natural participants in the video.

To demonstrate this concept, we developed a methodology for extracting the driving force of a harmonic oscillation system by observing the harmonic oscillation motion of objects, and then inferring the parameters of the natural phenomenon which caused the driving force. Two examples are used to show the strength and applications of this methodology. The first example is related to wind-object interaction. We observe the

effect of the wind on a plant in the scene, estimate the wind speed from the plant's motion, then use the wind information to make computer generated objects move. The second example is related to boat-water interaction. We study the motion of boats anchored by the shore, estimate the water wave elevation which drives the boat motion, the use the water wave parameters to control a computer synthesized floating structure. This serves the purpose of validating the principle behind VIDA as well as the processes involved in VIDA. Also, the idea of observing the natural environment and "reverse engineering" natural phenomena such as wind and water waves are novel ideas.

9.2 Limitations

One of the limitations of VIDA is due to the limited resolution of our video camera. When an object of interest is far away, its movement spans merely a few pixels. In this case, measuring accuracy becomes very critical, and we might not be able to capture enough details of the objects motion to make the analysis meaningful.

Another limitation is that currently we are not considering the scenario where the foreground computer synthesized object might create large alteration of the background environment. This situation can be studied in the future.

Since VIDA depends on vision algorithms to provide us with information about the background environment, it is constrained by the robustness and capability of existing computer vision algorithms. The advance of computer vision research will provide us with better tools for exploring VIDA related computer graphics problems.

9.3 Future Work

In this thesis we discussed the VIDA inverse problem to harmonic oscillation in detail. There are many interesting topics in VIDA and applications of it are awaiting to be explored.

There are a number of VIDA projects where we might be able to utilize existing computer vision techniques to extract object orientation and force information from the scene which we could use to drive virtual objects and characters. For instance, we can have virtual characters riding on a bus (Figure 9.1). Or we can have a virtual character riding a deer (Figure 9.2). Such examples explore the forces between a real object and the virtual object.

More difficult projects might require further studies in computer vision and our understanding of nature. For example, we might want to observe the white water flowing down a stream. We would like to be able to estimate the water force by observing the water flow. So far, there is no computer vision literature on this subject.

Since VIDA provides a new linkage between computer graphics and computer vision, it helps to expand the impact of computer vision to applications such as movie making, virtual reality and animation prototyping, to name a few. We need to capitalize on the existing computer vision tools, and push ourselves to invent new computer vision tools due to the demand of computer graphics applications. For computer graphics, VIDA introduces a rich source for animation creation and animation control. It allows us to use video to give indirect inputs, such as forces and constraints, to computer animation. VIDA poses many new research challenges to both computer graphics and computer

Figure 9.1: We can try to put a virtual character on the moving bus.

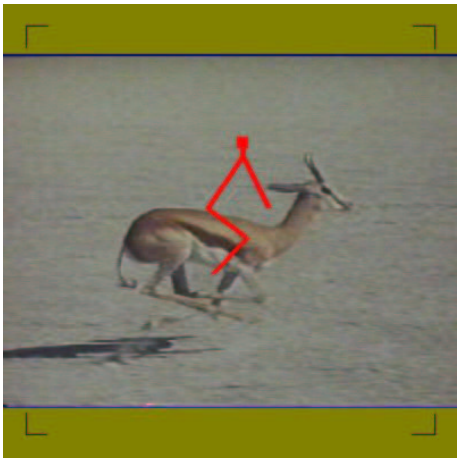

Figure 9.2: A frame from the gazelle riding sequence.

vision.

Appendix A

The Discrete Fourier Transform

If one wishes to obtain the Fourier transform of a given function, often that function is defined in terms of a continuous independent variable. But it may also happen that function values are given only at discrete values of the independent variable as with physical measurements made at regular time intervals. Assume that when we sample continuous function to obtain the discrete values. the sampling period is small enough that there is no aliasing problem. Regardless of the form of the given function, if the transform is evaluated by numerical computing, the values of the transform will be available only at discrete intervals [Bra83]. We often think of this as though an underlying function of a continuous variable really exists and we are approximating it. From an operational viewpoint, however, it is irrelevant to talk about the existence of values other than those given and those computed (the input and output). Therefore, it is desirable to have a mathematical theory of the actual quantities manipulated. Hence, let us think in terms of a signal that is a function of time; but to recognize the discreteness of the independent variable, let us use the symbol τ. Here τ can assume only a finite number N of consec-

utive integral values, say, $\tau \in \{0, \ldots, N-1\}$. Thus, before entering into the realm of the discrete Fourier transform, the independent variable is typically transformed using a change of scale and a change of origin. Given a function $g(t)$, where t is time, it can be converted into a function $f(\tau)$ by sampling the function $g(t)$. In general, if the sampling interval is T and the first sample of interest occurs at $t = t_0$, then

$$f(\tau) = g(t_0 + \tau T), \qquad \tau = 0, 1, 2, \ldots, N-1.$$

By definition, $f(\tau)$, $f(\tau)$ possesses N discrete Fourier transform values $F(\nu)$ given by

$$F(\nu) = N^{-1} \sum_{\tau=0}^{N-1} f(\tau) e^{-j2\pi(\nu/N)\tau},$$

where j denotes the imaginary number. The quantity ν/N is analogous to frequency measured in cycles per sampling interval. The correspondence of symbols may be summarized as follows:

	Time	*Frequency*
Continuous case	t	ω
Discrete case	τ	$(2\pi\nu/N)$

The symbol ν has been chosen in the discrete case, instead of ω, to emphasize that the frequency integer ν is *related* to frequency but is not the *same* as frequency ω.

Given the discrete transform $F(\nu)$, one may recover the time series $f(\tau)$ with the aid of the inverse relationship

$$f(\tau) = \sum_{\nu=0}^{N-1} F(\nu) e^{j2\pi(\nu/N)\tau}.$$

In order to regain our physical feeling for numerical orders of magnitude, let us consider a record consisting of $1,024$ samples separated by 1-second intervals. We expect this to be representable by a Fourier series consisting of a constant term and multiples of a certain fundamental frequency. The fundamental period should be $1,024$ seconds, corresponding to a fundamental frequency $1/1,024$ hertz. The highest frequency needed will be 0.5 hertz, which has two samples per period. This will be the 512th harmonic. The reason that ν assumes $1,024$ values, whereas the number of frequencies is only 513, is as follows. Remember $F(\nu)$ is complex valued, except that $F(0)$ and $F(N/2)$ have no imaginary part and half the remaining values of $F(\nu)$ are complex conjugates of the other half. This is because $f(\nu)$ is real. If $f(\tau)$ were complex, there would be $2,048$ real data values and $F(\nu)$ would require $2,048$ real numbers for its specification. Therefore, there are $2+511\times 2 = 1024$ degrees of freedom in the values of $F(\nu)$, where $\nu = 0\ .\ .\ 512$, which is the same as for $f(\tau)$. The remaining coefficients of $F(\nu)$, i.e., for $\nu = 513\ .\ .\ .\ 1023$, can be obtained from this first block by symmetry, using the fact $f(\tau)$ is real.

Appendix B

Deflection of Cantilever Beam

The question of the deflection of beams is important, because the magnitudes of the deflections are needed frequently [TM49]. In this section, we will use the double integral method to determine the deflection of a two dimensional (2D) cantilever beam with a uniform load of intensity w (Figure B.1). The direction of the load is perpendicular to the length of the beam. The length of the beam is l. The bending moment M at any

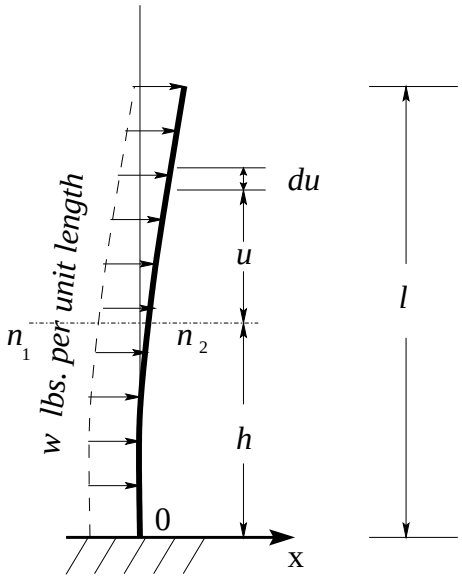

Figure B.1: The deflection of a cantilever beam with a uniform load.

cross section n_1n_2 distance h from the built-in end is the sum of the bending moments at each section du above the cross section n_1n_2:

$$\begin{aligned} M &= \int_0^{l-h} w(l-h-u)du \\ &= \frac{w(l-h)^2}{2}. \end{aligned} \tag{B.1}$$

where u is the distance from the cross section n_1n_2 to a point above n_1n_2. Let us denote the deflection of the beam $\varpi(h)$. Timoshenko and MacCullough [TM49] showed that the bending moment M is related to the flexure stiffness and the curvature of the beam:

$$M = EI\frac{d^2\varpi(h)}{dh^2}, \tag{B.2}$$

where $\frac{d^2\varpi(h)}{dh^2}$ is the curvature of the beam. Putting equations B.1 and B.2 together, we have

$$EI\frac{d^2\varpi(h)}{dh^2} = M = \frac{w(l-h)^2}{2}. \tag{B.3}$$

Integrating Eq. B.3 gives us the slope of the bending beam:

$$EI\frac{d\varpi(h)}{dh} = \frac{w}{2}(l^2h - lh^2 + \frac{h^3}{3}) + C_1. \tag{B.4}$$

Since the slope of the beam $d\varpi(h)/dh$ at $h = 0$ is zero, we have $C_1 = 0$. Integrating Eq. B.4, we obtain the deflection of the beam:

$$EI\varpi(h) = \frac{w}{2}\left(\frac{l^2h^2}{2} - \frac{lh^3}{3} + \frac{h^4}{12}\right) + C_2. \tag{B.5}$$

Since the base of the beam has deflection $\varpi = 0$ at $h = 0$, we have $C_2 = 0$. So, the deflection of the beam is

$$\varpi(h) = \frac{w}{EI}\frac{h^2}{24}(6l^2 - 4lh + h^2). \tag{B.6}$$

Bibliography

[BCS97] Christopher Bregler, Michele Covell, and Malcolm Slaney. Video rewrite: Driving visual speech with audio. *Computer Graphics*, pages 1–8, 1997.

[BG01] Samuel Boivin and André Gagalowicz. Image-based rendering of diffuse, specular and glossy surfaces from a single image. In *Proceedings of ACM SIGGRAPH 2001*, Computer Graphics Proceedings, Annual Conference Series, pages 107–116. ACM Press / ACM SIGGRAPH, August 2001. ISBN 1-58113-292-1.

[Bha78] Bameswar Bhattacharyya. *Dynamics of Marine Vehicles.* John Wiley & Sons, 1978.

[Bon97] J. S. De Bonet. Multiresolution sampling procedure for analysis and synthesis of texture images. In *Computer Graphics*, pages 361–368. ACM SIGGRAPH, 1997.

[Bra83] Ronald N. Bracewell. *The Fourier Transform and Its Applications.* McGraw-Hill Book Company, second edition, 1983.

[Bra99] Matthew Brand. Voice puppetry. *Computer Graphics*, pages 21–28, August 1999.

[BY95a] Michael J. Black and Y. Yacoob. Recognizing facial expressions in image sequences using local parameterized models of image motion. *Int. Journal of Computer Vision*, 25(1):23–48, 1995.

[BY95b] Michael J. Black and Y. Yacoob. Tracking and recognizing rigid and non-rigid facial motions using local parametric models of image motion. *Fifth International Conf. on Computer Vision, ICCV'95*, pages 374–381, June 1995.

[CP93] Ray W. Clough and Joseph Penzien. *Dynamics of Structures.* McGraw-Hill, Inc., second edition, 1993.

[DRB97] George Dretakis, Luc Robert, and Sylvain Bougnoux. Interactive common illumination for computer augmented reality. *Eurographics Rendering Workshop 1997*, June 1997.

[EL99] Alexei A. Efros and Thomas K. Leung. Texture synthesis by non-parametric sampling. In *IEEE International Conference on Computer Vision*, pages 1033–1038, 1999.

[EM02] El-Maraghi. *Robust online appearance models for visual tracking.* PhD thesis, University of Toronto, 2002.

[Fey63] Richard Phillips Feynman. *The Feynman lectures on physics*, volume 1–3. Addison-Wesley Pub. Co., Reading, Massachusetts, 1963.

[FGR93] Alain Fournier, Atjeng S. Gunawan, and Chris Romanzin. Common illumination between real and computer generated scenes. *Graphics Interface '93*, May 1993.

[Fit01] Andrew W. Fitzgibbon. Stochastic rigidity: Image registration for nowhere-static scenes. *IEEE International Conference on Computer Vision*, pages 662–669, July 2001.

[Fre71] A. P. French. *Vibrations and Waves.* W. W. Norton & Company, New York, 1971.

[GJ82] Thomas C. Gillmer and Bruce Johnson. *Introduction to Naval Architecture.* Naval Institute Press, 1982.

[HB95] D. J. Heeger and J. R. Bergen. Pyramid based texture analysis/synthesis. *Computer Graphics*, pages 229–238, 1995.

[Hom36] F Homann. *Einfluss grosser Zähigkeit bei Stroömung um Zylinder*, volume 7. 1936.

[Hyd] Delft Hydraulics. *SHIPMA.* http://www.marin.nl/services/ softwaredevelopment/mscn_shipma.html.

[HYS01] Jun'ichi Hoshino, Masanobu Yamamoto, and Hirofumi Saito. A match moving technique for merging cg cloth and human movie sequences. *The Journal of Visualization and Computer Animation*, 12(1):23–29, 2001.

[Ide94] I. E. Idelchik. *Handbook of Hydraulic Resistance.* CRC Press, third edition, 1994.

[JB93] Allan D. Jepson and Michael J. Black. Mixture models for optical flow computation. *Partitioning Data Sets, DIMACS Workshop*, pages 271–286, April 1993.

[JFEM01] Allan D. Jepson, D. J. Fleet, and T. El-Maraghi. Wsl: Robust online appearance models for visual tracking. *Proc. IEEE CVPR*, pages 415–422, 2001.

[JJ89] Allan D. Jepson and Michael R. M. Jenkin. The fast computation of disparity from phase differences. *Conference on Computer Vision and Pattern Recognition*, 1989.

[JJ94] Michael R. M. Jenkin and Allan D. Jepson. Recovering local surface structure through local phase difference measurements. *CVGIP: Image Understanding*, 59(1):72–93, January 1994.

[LFT97] Eric P. Lafortune, Sing-Choong Foo, and Kenneth E. Torrance. Non-linear approximation of reflectance functions. *Computer Graphics*, August 1997.

[MJ97] Richard Mann and Allan D. Jepson. Computational perception of scene dynamics. *Computer Vision and Image Understanding*, 65(2):113–128, 1997.

[Mor90] Nina Morgan. *Marine Technology Reference Book*. Butterworth & Co. (Publisher) Ltd, 1990.

[MT87] W. Muckle and D. A. Taylor. *Muckle's Naval Architecture*. Butterworth & Co. (Publisher) Ltd, 1987.

[PA99] Zoran Popović and Witkin Andrew. Physically based motion transformation. *Computer Graphics*, pages 11–20, August 1999.

[Pat89] Minoo H. Patel. *Dynamics of Offshore Structures*. Butterworth & Co. (Publisher) Ltd, 1989.

[Pee87] Peyton Z. Peebles, Jr. *Probability, Random Variables, and Random Signal Principles*. McGraw-Hill Book Company, second edition, 1987.

[PM92] John G. Proakis and Dimitris G. Manolakis. *Digital Signal Processing: Principles, Algorithms, and Applications*. Macmillan Publishing Company, second edition, 1992.

[PS78] Tim Poston and Ian Steward. *Catastrophe Theory and its Applications*. Pitman Publishing Limited, 1978.

[RC42] Henry E. Rossell and Lawrence B. Chapman. *Principles of Naval Architecture*. The Society of Naval Architects and Marine Engineers, 1942.

[Sch60] Hermann Schlichting. *Boundary Layer Theory*. McGraw-Hill Book Company, Inc., fourth edition, 1960.

[SDW01] Stefano Soatto, Gianfranco Doretto, and Ying Nian Wu. Dynamic texture. *International Conference on Computer Vision*, pages 439–446, 2001.

[SE01] Arno Schödl and Irfan Essa. Machine learning for video-based rendering. *Advances in Neural Information Processing Systems 13: Proceedings of the 2000 Conference*, 2001.

[SF92] Mikio Shinya and Alain Fournier. Stochastic motion – motion under the influence of wind. *Eurographics '92*, 11(3):119–128, 1992.

[Smi88] J. W. Smith. *Vibration of Structures*. Chapman and Hall, 1988.

[SMO98] Mikio Shinya, Takeaki Mori, and Noriyoshi Osumi. Periodic motion synthesis and fourier compression. *The Journal of Visualization and Computer Animation*, 9:95–107, 1998.

[SSSE00] Arno Schödl, Richard Szeliski, David H. Salesin, and Irfan Essa. Video textures. In *Proceedings of ACM SIGGRAPH 2000*, Computer Graphics Proceedings, Annual Conference Series, pages 489–498. ACM Press / ACM SIGGRAPH / Addison Wesley Longman, July 2000. ISBN 1-58113-208-5.

[ST94] Jianbo Shi and Carlo Tomasi. Good features to track. *IEEE Conference on Computer Vision and Pattern Recognition*, 1994.

[Sta97] Jos Stam. Stochastic dynamics: Simulating the effects of turbulence on flexible structures. *EUROGRAPHICS'97 Proceedings*, 16(3):159–164, 1997.

[SWI97] Yoichi Sato, Mark D. Wheeler, and Katsushi Ikeuchi. Object shape and reflectance modeling from observation. *Computer Graphics*, August 1997.

[Tim47] S. Timoshenko. *Strength of Materials (Part II).* D. Van Nostrand Company, Inc., second edition, 1947.

[Tim49] S. Timoshenko. *Strength of Materials (Part I).* D. Van Nostrand Company, Inc., second edition, 1949.

[TL25] S. Timoshenko and J. M. Lessells. *Applied Elasticity.* Westinghouse Technical Night School Press, East Pittsburgh, PA, first edition, 1925.

[TM49] S. Timoshenko and Gleason H. MacCullough. *Elements of Strength of Materials.* D. Van Nostrand Company, Inc., Princeton, New Jersey, third edition, 1949.

[Tur01] Greg Turk. Texture synthesis on surfaces. *Computer Graphics*, pages 347–354, 2001.

[Wil90] Lance Williams. Performance-driven facial animation. *Computer Graphics*, 24(4):235–242, 1990.

[WL00] Li-Yi Wei and Marc Levoy. Fast texture synthesis using tree-structured vector quantization. *Computer Graphics*, pages 479–488, 2000.

[WL01] Li-Yi Wei and Marc Levoy. Texture synthesis over arbitrary manifold surfaces. *Computer Graphics*, pages 355–360, 2001.

[YDH99] Y. Yu, P. Debevec, and T. Hawkins. Inverse global illumination: Recovering reflectance model of real scenes from photographs. *Computer Graphics*, August 1999.