Computation of Air-Vortices Based on GPU Technology - Optimizing and Parallelizing a Model for Wake-Vortex Prediction Using OpenCL

Erik Peldan

Master of Science Thesis
Stockholm, Sweden 2013
Computation of Air-Vortices Based on GPU Technology - Optimizing and Parallelizing a Model for Wake-Vortex Prediction Using OpenCL

ERIK PELDAN

Master’s Thesis in Numerical Analysis (30 ECTS credits)
Master Programme in Mathematics (120 credits)
Royal Institute of Technology year 2013
Supervisor at KTH was Olof Runborg and Elias Jarlebring
Supervisor at AVTECH was David Rytter
Examiner was Michael Hanke

TRITA-MAT-E 2013:07
ISRN-KTH/MAT/E--13/07--SE
Abstract

This thesis details the refinement and numerical solution of a preexisting model for predicting the strengths and positions of so-called wake-vortices that are generated from the lift of heavy aircraft. The ultimate objective is to implement a numerical scheme for the model that is fast enough to allow for probabilistic methods, such as Monte Carlo-simulations, in order to deal with the inherent uncertainty in input parameters for wake-vortex predictions.

The differential equation system of the wake-vortex model is stated clearly, which has not been done before. The refinement consists in reducing the number of necessary state variables in the differential equation system.

A numerical algorithm based on the mathematical properties of the model is implemented and different ways of optimizing the computations are considered, e.g. through parallelization.

Finally, a study will be made trying to assess the validity of the results through analyses of the accuracy and of the model’s sensitivity to small input parameter variations.
Referat

Beräkning av Luftvirvlar med hjälp av GPU-Teknologi


Modellens differentialekvationssystem kommer redogöras för, något som inte gjorts tidigare. Modellförfiningen består i en minskning av antalet tillståndsvariabler i differentialekvationssystem.

En numerisk lösare baserad på modellens inneboende matematiska egenskaper utvecklas och olika sätt att optimera beräkningarna kommer övervägas, t ex genom parallellisering.

Slutligen så kommer även en studie där resultatens giltighet valideras göras genom att undersöka noggrannheten i lösningen samt känsligheten för variationer i inputparametrar.
# Contents

1 Introduction .......................................................... 1
   1.1. Background ................................................. 1
   1.2. Literature .................................................. 2
   1.3. Purpose ..................................................... 2
   1.4. My contribution ............................................. 2
   1.5. Acknowledgement ............................................ 3

2 Physical Model ...................................................... 5
   2.1. Preliminaries ............................................... 6
   2.2. Ancillary Physical Concepts ................................. 8
       2.2.1. Atmospheric Conditions ............................ 8
       2.2.2. Vortex Circulation Strength ....................... 9
       2.2.3. Buoyancy ............................................. 11
   2.3. The Wake-Vortex Model .................................... 11
       2.3.1. Vortex Movement ..................................... 12
       2.3.2. Biot-Savart Interactions ........................... 12
       2.3.3. The Ground Effects ................................. 13

3 ODE System .......................................................... 21
   3.1. Mathematical Definitions .................................. 21
       3.1.1. The Vortex Index Functions ......................... 22
       3.1.2. Representing the $Z$ and $G$ functions ............ 23
       3.1.3. The Ground Effect Conditions ....................... 23
       3.1.4. The Vortex Circulation Strengths .................. 24
       3.1.5. The Vortex Activation Functions ................... 24
       3.1.6. The Biot-Savart Tensors ............................ 25
   3.2. Formulating the ODE System ................................. 25
       3.2.1. The $x$-Coordinate of the Gate ..................... 26
       3.2.2. The Buoyancy Factors ............................... 26
       3.2.3. The Secondary Vortex Angles ....................... 28
       3.2.4. The Vortex Coordinate Pairs ....................... 28
       3.2.5. Initial Values ....................................... 29
       3.2.6. Ground Effect-Specific Dependencies ............... 29
4 Implementation

4.1. Solving on the Graphics Card ........................................ 33
4.2. Solver Algorithm ...................................................... 33
4.3. The Weather Grid ...................................................... 34
4.4. The Memory Setup ...................................................... 35
  4.4.1. Weather Grid Data .............................................. 35
  4.4.2. Simulation Parameters .......................................... 36
  4.4.3. Runtime Parameters ............................................. 36
  4.4.4. The Solution Data .............................................. 36
4.5. Accuracy vs Performance Trade-offs ................................. 36
  4.5.1. Native Math ....................................................... 36
  4.5.2. An $r_c(\Gamma(t))$-Lookup Table ................................. 37
  4.5.3. Circulation ........................................................ 37
4.6. GPU Idiosyncrasies .................................................... 37
  4.6.1. Debugging on the GPU .......................................... 37
  4.6.2. Vectorization Speedups ......................................... 37
  4.6.3. Floating Point Accuracy ........................................ 38
  4.6.4. Data Transfer .................................................... 38
  4.6.5. Memory Buffer Limitations ..................................... 39

5 Analysis of the Rate of Convergence ................................ 41
  5.1. Convergence in Out of Ground Effect .............................. 42
  5.2. Convergence in Near Ground Effect ............................... 43
  5.3. Convergence In Ground Effect ..................................... 44
  5.4. Analysis and Proposed Improvements ............................... 46

6 Analysis of the Parameter Sensitivity ................................. 47
  6.1. Sensitivity in Out of Ground Effect ............................... 48
  6.2. Near/In Ground Effect Sensitivity ................................ 49
  6.3. Tuning the Variations .............................................. 61
  6.4. A Sample Monte Carlo-Simulation ................................ 61

7 Final Thoughts and Suggestions ....................................... 65
  7.1. Reliability .......................................................... 65
  7.2. Further Work ....................................................... 67

Bibliography 69
Chapter 1

Introduction

1.1. Background

With air traffic growing at an ever-increasing rate, airports are hard pressed to accommodate as many take-offs and landings as possible without compromising safety requirements.

Currently, the number one factor limiting airport capacities is the mandatory delay between consecutive landings incurred by hazardous vortices from large aircraft. Safety regulations require that landing aircraft be separated a prescribed amount of time to ensure that they not be caught up in the vortices of previous aircraft, with potentially severe consequences.

Previously, it has been unclear what the adequate amount of time should be between approaching aircraft. Most airports use lookup tables based on the make and weight of the aircraft to determine the delay, but this fails to take into account local weather conditions and can lead to flight separations becoming unnecessarily long or even dangerously short depending on if the weather is favorable or not. This in turn can lead either to airports not being able to reach their full potentials because of unnecessary delays or to an elevated risk of accidents because of inadequate delays.

Because of this, researchers are currently putting an effort into trying to model these so-called wake-vortices. A model based on empirical data that relies on the solution of a differential equation system has been progressively developed over the course of 30 years.

AVTECH, which is the company where this thesis was written, have developed their own model derived from the 30-year-old one. However, due to the many assumptions in the model made purely on empirical studies and the number of empirical constants, it is clear that there remains a great deal of uncertainty and it is still hard to correctly predict all vortices’ positions and strengths.

It is thus of interest to assess the effect that the uncertainty has on the outcome of the predictions. AVTECH want to improve their calculations using probabilistic methods in order to investigate the effects of small variations in the input data.
CHAPTER 1. INTRODUCTION

The use of probabilistic methods, however, necessitates quick calculations. That is the purpose of this Master’s Thesis.

1.2. Literature

Three important papers have been used frequently throughout this thesis for describing the vortex model. Rytter [1] is the Master’s Thesis conducted by David Rytter at AVTECH that this thesis is based on. The vortex model presented in this thesis is the same as the one in Rytter [1] except for a few very minor modifications. Rytter, in turn, relied heavily on the two reports Holzäpfel [2] and NASA [4]. NASA [4] is where the concept of modeling vortex interactions through the use of a differential equation system was introduced. Holzäpfel [2] on the other hand, provided descriptions of most of the ancillary physical phenomena including modeling of stratification, turbulence, circulation and so on.

1.3. Purpose

The main purpose of this thesis is to construct a fast algorithm to compute the existing vortex-model at AVTECH. The overall goal is to utilize the algorithm for performing Monte Carlo-simulations in order to quantify the uncertainty in the model.

1.4. My contribution

In the process of writing this thesis, I have contributed by

- Restating the vortex-model as an Ordinary Differential Equation instead of a Delay Differential Equation. This simplifies the mathematical work since ODEs are easier to solve in general.

- Reducing the required number of state-variables down to 19. The original implementation used 33. I also suggest an alternative implementation requiring only 15 variables in Section 3.2.3.

- Actually presenting the ODE in full. Rytter [1] explains the principles behind the ODE, and the formulae can be seen in the code, but the ODE itself is never spelled out in precise form. I have used Rytter’s MATLAB code to restate the ODE and also attempted to simplify some of the formulae by use of vector notation.

- Translating the implementation first from MATLAB to C++, then from C++ to OpenCL.

- Selecting and implementing a suitable numerical scheme for solving the differential equation system.
1.5. Acknowledgement

I would like to acknowledge the invaluable guidance in the form of the many ideas, comments and suggestions I received from my three supervisors: Olof Runborg and Elias Jarlebring at KTH, and David Rytter at AVTECH. Their continuous support throughout the process of this thesis has been greatly appreciated. I am very proud of how the thesis turned out, and I am truly grateful to them for helping me achieve it.

I also would like to thank AVTECH as a company for giving me this opportunity. It has been a very interesting project, and I am thankful that I was entrusted to do it. I specifically enjoyed learning about programming on the graphics card and the fact that I implemented the numerical scheme myself.

Finally, I would also like to acknowledge that AVTECH assisted me in supplying the hardware for performing the Monte Carlo-simulations, and for granting me access to their weather data that was used in the Monte Carlo-simulations.
Chapter 2

Physical Model

A central concept in this thesis will be that of a wake-vortex. A wake-vortex is a vortex in the atmosphere that is formed as a result of the turbulence generated from a large aircraft during flight. Typically there is one wake-vortex generated per wing of an aircraft. Wake-vortices can be dangerous to encounter. For this reason,

![Figure 2.1. Vortices are generated because an aircraft pushes air downward in order to fly. The initial vortex separation is $b_0$ meters.](image)

we want to construct a model for how they behave. Primarily, we are interested in predicting their strengths and positions so that subsequent aircraft can avoid colliding with them. However, in order to do this, we must also construct a model for other physical phenomena that affect vortex behavior.

Thus, after discussing some notation and conventions in Section 2.1 we begin by explaining these physical concepts in Section 2.2. We then move on to explain the principles governing wake-vortex interactions in Section 2.3 before finally formulating an Ordinary Differential Equation (ODE) system from those principles in Chapter 3.

As we will see, the system will amount to an $n$-body-simulation with some additional variables for incorporating weather conditions. Moreover, the number of
vortices will grow as we approach the ground. Figure 2.2 illustrates a typical vortex trajectory. The vortices descend until they get close to the ground, after which their behavior is more complex. It is this complex behavior that the vortex model aims to predict.

The ODE and the modeling of the physical phenomena that are presented in this thesis are those developed by Rytter in [1] at AVTECH.

The use of this ODE together with the physical model has been given the name P2Pm reflecting that it is derived from the P2P model of Holzäpfel [2] (the ‘m’ is short for ‘modified’). P2P is based largely on empirical data and as such does not always justify many of its formulae theoretically. In fact, according to Holzäpfel [2], section Model Concept:

P2P is designed to include as much knowledge as possible gained from both experimental and numerical wake vortex research with a focus on operational needs.

2.1. Preliminaries

In the model we will use a concept called gates. A gate is a 2D-plane perpendicular to the flight trajectory at a given time $T$ with a normal pointing in the flight direction. It should match the position of the aircraft at time $T$. We illustrate how gates are related to aircraft trajectories in Figure 2.4.

The whole model is built around describing vortex movements within a gate. Thus vortex movements and interactions are essentially 2D, but the gate itself is allowed to move along its normal direction to simulate 3D-movement. The normal
2.1. PRELIMINARIES

direction is always fixed, so that regardless of how the gate has moved from its original position it will always point in the same direction.

We will represent a gate by the $y$-$z$-plane as shown in Figure 2.3. Thus vortices

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{figure2.3.png}
\caption{The coordinate system of a gate. We align the gate with the $y$-$z$ plane. $e_z$ points in the plane normal.}
\end{figure}

will have individual $y$- and $z$-coordinates, but share the same $x$-coordinate, \textit{i.e.}, the $x$-position of the gate. Since we will be dealing with multiple vortices, we will refer to them by enumeration. For instance, vortex $i$ will have the position

$$r_i = (x, y_i, z_i).$$

(2.1)

Also useful will be the vector $r_{ij}$ pointing from vortex $j$ to vortex $i$:

$$r_{ij} = r_i - r_j.$$  

(2.2)

The axis unit vectors will be denoted by $(e_x, e_y, e_z)$, and are illustrated in Figure 2.3.

The idea is to create a large number of gates for different points in time and apply the wake-vortex model to each one of them. By choosing a sufficient amount of gates, one can get an understanding of how the vortices behave throughout a flight trajectory. See Figure 2.4 for an illustration.

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{figure2.4.png}
\caption{Gates are generated from flight trajectories at different time points, and their normals point in the direction the aircraft is heading at the time point they correspond to. Each gate’s normal is fixed, so that it will always face the same direction regardless of if the gate has moved from its initial position or not.}
\end{figure}

We select the units so that the formulae are ‘aircraft-agnostic’: That is, we do not want equations to depend on the aircraft size nor weight.
Let the initial vortex spacing $b_0$ (illustrated in Figure 2.1) be equal to 1 unit-length in the coordinate system:

$$1 \text{ unit length} = b_0 \text{ meters}. \quad (2.3)$$

Furthermore, let

$$1 \text{ unit time} = t_0 \text{ seconds}. \quad (2.4)$$

The scaling factor $t_0$ depends on the aircraft’s initial root circulation $\Gamma_0$:

$$t_0 = \frac{2\pi b_0^2}{\Gamma_0} \quad (2.5)$$

That is, $\Gamma_0$ is the value of the vortex strength (to be described in Section 2.2.2) at time $t = 0$. It depends on the size and weight of the aircraft, so by scaling by it, we remove that dependency.

### 2.2. Ancillary Physical Concepts

Above, we stated that vortex behavior will be governed by a number of physical phenomena. These phenomena are: atmospheric conditions, vortex circulation and atmospheric buoyancy. We will now present how they are modeled.

#### 2.2.1. Atmospheric Conditions

Atmospheric conditions are an important part of the model. They affect both vortex movement and circulation.

There are three atmospheric properties that influence the vortex behavior; the wind vector $W(x, y, z)$, the turbulence $\epsilon(x, y, z)$ and the stratification $N(x, y, z)$ and they are all considered to be known a priori in the wake-model.

By stratification, we mean the Brunt-Väisälä or buoyancy frequency. For our purposes, it will affect all vertical movement in the atmosphere and also circulation lifetime. It is defined by

$$N = t_0 \sqrt{\frac{g}{\theta}} \frac{d\theta}{dz}. \quad (2.6)$$

where $\theta$ denotes the potential temperature in the atmosphere and $g$ is the gravitational acceleration. Stratification will affect vortex decay rate and buoyancy factors, as we will see in Sections 2.2.2 and 2.2.3.

Turbulence, also known as Eddy Dissipation Rate (EDR), is a measure of how quickly the kinetic energy within a vortex will dissipate. It will affect the lifetime of the vortices in our model, as explained in Section 2.2.2.

The wind vector $W$ represents the direction and strength that the wind is blowing. In the P2Pm we will make the assumption that the wind vector is always in the x-y plane. That is, there is no z-component of the wind vector,

$$W \cdot e_z = 0. \quad (2.7)$$
2.2. ANCILLARY PHYSICAL CONCEPTS

Typically, the $y$-component (cross-wind) will constitute part of the total time-derivative of each vortex’s $y$-coordinate. The $x$-component (tail-/head-wind) of the wind vector will affect the movement of the gate.

It will be convenient later on to refer to the mean wind, taken over both main vortices:

$$\bar{W} = \frac{1}{2} (W_{\text{left main vortex}} + W_{\text{right main vortex}}) \quad (2.8)$$

or, using the vortex-enumeration of Table 2.1 in Section 2.3:

$$\bar{W}(x, y_1, y_2, z_1, z_2) = \frac{1}{2} (W(x, y_1, z_1) + W(x, y_2, z_2)). \quad (2.9)$$

2.2.2. Vortex Circulation Strength

Vortex circulation is what determines the strength of a vortex. In physical terms, it is a measure of how fast the air molecules are moving within the vortex. We intend to describe the circulation $\Gamma$ as a function of the distance $r$ from the vortex centre and of time $t$.

The basic idea is to represent $\Gamma(r, t)$ using the formula

$$\Gamma_{\text{N-S}}(r, t) \overset{\text{def}}{=} 1 - \exp \left( -\frac{r^2}{4\nu t} \right), \quad (2.10)$$

which constitutes an analytical solution of the Navier-Stokes equations for a non-stationary, plane, rotating flow [2]. That is, it is the analytical solution to a very specific case. Note that, $\Gamma_{\text{N-S}}(r, 0) = 1$ is always the maximum due to the normalization discussed in Section 2.1.

However, it is clear from LES (Large Eddy Simulations, see [2]) that wake-vortices do not behave exactly as predicted by $\Gamma_{\text{N-S}}$. Instead, there appear to be two distinct phases in vortex decay. Initially, the vortices are in what is known as diffusion phase, where decay is relatively slow. Eventually, they will move on to the rapid decay phase, which sees a much faster decrease.

To account for this, we define $\Gamma(r, t)$ using $\Gamma_{\text{N-S}}(r, t)$, but in such a way that the two decay phases can be discerned. Let $T_1$ be the time when diffusion phase starts and $T_2$ be the time when rapid decay starts. Then we set

$$\Gamma(r, t) = \begin{cases} 
A - \exp \left( -\frac{r^2}{4\nu_1(t-T_1)} \right), & t \leq T_2, \\
A - \exp \left( -\frac{r^2}{4\nu_1(t-T_1)} \right) - \exp \left( -\frac{r^2}{4\nu_2(t-T_2)} \right), & T_2 < t < T_3, \\
0, & T_3 \leq t, 
\end{cases} \quad (2.11)$$

where $T_3$ is the earliest time satisfying

$$A - \exp \left( -\frac{r^2}{4\nu_1(T_3-T_1)} \right) - \exp \left( -\frac{r^2}{4\nu_2(T_3-T_2)} \right) = 0. \quad (2.12)$$
The constants $A$ and $T_1$ have fixed numerical values

$$T_1 = -2.22, \quad (2.13)$$
$$A = 1.162, \quad (2.14)$$

because they have been adapted [2] to reflect the vortex structure at time $t = 0$.

The viscosity, $\nu_1$ is also constant. It is found through LES [2],

$$\nu_1 = 1.78 \times 10^{-3}, \quad (2.15)$$

whereas both $T_2$ and $\nu_2$ are functions of the turbulence $\epsilon$ and stratification $N$.

Let $W(x)$ denote the Lambert W-function, i.e., the inverse function of $f(W) = W \exp(W)$. Then

$$T_2(\epsilon, N) = \begin{cases} 
5 \exp \left( -\frac{37}{40} N \right), & \epsilon \leq 0.0235, \\
-\frac{5}{4} \mathcal{W} \left( -\frac{14}{5} \epsilon^4 \right) \exp \left( -N \frac{37}{560} \mathcal{W} \left( -\frac{14}{5} \epsilon^4 \right) \right), & 0.0235 \leq \epsilon \leq 0.2535, \\
[0.8039/\epsilon^{3/4} - 1] \exp \left( -\frac{37}{200} N [0.8039/\epsilon^{3/4} - 1] \right), & \epsilon > 0.2535, 
\end{cases} \quad (2.16)$$

For $\nu_2$ there are different definitions depending on the model you want to use. Holzäpfel [2] defined a lower and an upper bound:

$$\nu_2^{\text{lower}} = \frac{9}{5000} + \frac{13}{1000} N \quad (2.17)$$
$$\nu_2^{\text{upper}} = \frac{1}{40} \left[ 1 - \exp \left( -N - \frac{13}{25} \right) \right]. \quad (2.18)$$

and performed a study with different values of $\nu_2$ within that range. Rytter [1] elected to use the mean value. For all the numerical computations in this report, we will use $\nu_2 = \nu_2^{\text{lower}}$.

Nevertheless, even using $\Gamma(r,t)$ above it is still very difficult to correctly determine the velocity distribution within an actual vortex in the atmosphere. This is due to the fact that e.g. ambient turbulence and or neighboring vortices can drastically change the local circulation values in the atmosphere. For this reason, we make use of the simplification introduced in Holzäpfel [2] in which we compute the circulation as an average over the radii 5 to 15 meters as a means to reduce the sensitivity to modeling errors:

$$\Gamma_{5-15}(t) \overset{\text{def}}{=} \frac{1}{11} \int_5^{15} \Gamma(r/b_0, t)dr. \quad (2.19)$$

We need to scale the radius by $1/b_0$ above because the integration limits are given in meters instead of the model’s length units (defined in Section 2.1).

Note that Equation (2.19) implies that the vortex circulation model we will be using only depends on $t$ and not on $r$. 

10
2.3. THE WAKE-VORTEX MODEL

2.2.3. Buoyancy

Buoyancy forces are present in all types of fluids (and the atmosphere can be seen as such). They are the result of the fluid reacting to gravity and pressing downwards, causing whatever objects present in the fluid to experience a slight lifting force. In the atmosphere, buoyancy forces are only existent during high stratification (=high stability), i.e., when there is a steep temperature gradient.

The P2Pm model represents buoyancy through buoyancy factors $B_i$. The subscript $i$ indicates which vortex the buoyancy factor represents. The $B_i$ work by scaling all vertical movement. That is, if $d$ is the expected vertical displacement without buoyancy, then the new displacement will be $dB_i$. Initially we put $B_i(t=0) = 1$.

To compute $B_i$ for other times, we integrate the time-derivative:

$$\frac{dB_i}{dt} = -\frac{181}{400} N(x,y,z_i) \sqrt{2} (z_0 - z_i) \tag{2.20}$$

where $z_0$ is the initial altitude of the vortex, i.e., $z_0 = z_i(t=0)$, and $N(x,y,z)$ is the stratification defined in Section 2.2.1.

2.3. The Wake-Vortex Model

We are now ready to describe the wake-vortex model itself. The general idea is that we want perform an $n$-body simulation where the vortices act on each other through the so-called Biot-Savart interactions that will be described shortly. Each vortex has an assigned number $i$, a position $r_i$ and a circulation strength $\sigma_i$. Refer to Table 2.1 where each vortex’s number is listed.

<table>
<thead>
<tr>
<th>Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>left main vortex</td>
</tr>
<tr>
<td>2</td>
<td>right main vortex</td>
</tr>
<tr>
<td>3</td>
<td>left first secondary vortex</td>
</tr>
<tr>
<td>4</td>
<td>right first secondary vortex</td>
</tr>
<tr>
<td>5</td>
<td>left second secondary vortex</td>
</tr>
<tr>
<td>6</td>
<td>right second secondary vortex</td>
</tr>
</tbody>
</table>

Table 2.1. This will be the numbering convention for the vortices. The concept of a ‘secondary vortex’ is introduced in Section 2.3.3.

As was mentioned in the beginning of this chapter, the number of active vortices varies. The model is split up into three different phases, called ground effects, which determine the nature and complexity of the simulations. The most complex phase, known as In Ground Effect, can comprise up to 12 vortices (counting also mirror images), whereas in Out of Ground there are only two.

For the main vortices 1 and 2 from Table 2.1, the circulation strength is given by:

$$\sigma_i(t) = (-1)^{i+1} \Gamma_{5-15}(t), \quad i \in \{1, 2\}.$$
where $\Gamma_{5-15}(t)$ is the circulation function defined in Section 2.2.2. However, the strengths of the remaining 4 so-called secondary vortices will first be defined in Section 2.3.3 when we explain the ground effects.

### 2.3.1. Vortex Movement

Let $r_i = (x_i, y_i, z_i)$ denote the position of a vortex $i$, let $B_i$ be its buoyancy factor, let $W$ be the wind with

$$W(x, y_1, y_2, z_1, z_2) = \frac{1}{2}(W(x, y_1, z_1) + W(x, y_2, z_2))$$

being its average value as defined in Section 2.2.1, and let $v_i$ be the Biot-Savart interaction-velocity (see below). Then vortex $i$’s spatial time-derivatives are given by:

\[
\begin{align*}
\frac{dx}{dt} &= \bar{W}(x, y_1, y_2, z_1, z_2) \cdot e_x \\
\frac{dy_i}{dt} &= v_i(\ldots) + W(x, y_i, z_i) \cdot e_y \\
\frac{dz_i}{dt} &= v_i(\ldots) + W(x, y_i, z_i) \cdot B_i(t) e_z.
\end{align*}
\]

For brevity, we omitted the arguments to $v_i$ and replaced them by ’$\ldots$’ because, as we will see in the next section, $v_i$ depends on all vortex positions $r_i$ and vortex strengths $\sigma_i$.

However, Formulae (2.21)-(2.23) can be further simplified. Since $W(\ldots) \cdot e_z = 0$ by definition (Section 2.2.1) we get the simpler system:

\[
\begin{align*}
\frac{dx}{dt} &= \bar{W}(x, y_1, y_2, z_1, z_2) \cdot e_x \\
\frac{dy_i}{dt} &= v_i(\ldots) + W(x, y_i, z_i) \cdot e_y \\
\frac{dz_i}{dt} &= v_i(\ldots) \cdot B_i(t) e_z.
\end{align*}
\]

Note that $\frac{dx}{dt}$ is lacking a subscript because, as was mentioned, all vortices share $x$-coordinate with the gate.

### 2.3.2. Biot-Savart Interactions

We mentioned that vortices affect each other through the so-called Biot-Savart interactions. They work as follows: The effect one vortex has on another vortex’s velocity is proportional to the strength of the affecting vortex, and inversely proportional to the distance between them. That is, the velocity contribution from vortex $j$ to vortex $i$ is given by:

$$v_{ij} = \sigma_j(t, \ldots) \frac{e_{ij} \times e_x}{r_{ij}}$$

(2.27)
2.3. THE WAKE-VORTEX MODEL

where $\sigma_j$ is the circulation strength of vortex $j$. It is closely related to the circulation function $\Gamma(t)$ from Section 2.2.2, but its exact definition will have to wait until Section 2.3.3 because it depends on the vortex. The unit vector $e_{ij}$ points from vortex $j$ to vortex $i$. Equation (2.27) is called the Biot-Savart law applied to aerodynamics and is further explained in Rytter [1]. Moreover, vortex $i$ will be affected by all other vortices in the simulation, so the total ‘Biot-Savart contribution‘ experienced by vortex $i$ is thus:

$$v_i = \sum_{j \neq i} v_{ij}. \tag{2.28}$$

That is, the sum of the contributions from each individual vortex.

Special Case - Out of Ground

We do not explicitly calculate the Biot-Savart interactions if there are only two vortices participating in the simulation. This is because the vortices are then assumed to be fixed relative to each other, which allows us to use a different formula:

$$v_i = \left[1 - \exp\left(-\frac{(0.4b_0)^21.257}{r_c(\Gamma(t))^2}\right)\right] e_z \tag{2.29}$$

where $r_c$ is the so-called effective core radius. It was first introduced by Holzäpfel [2], and is given implicitly by:

$$\Gamma_{5-15}(t) = \frac{\Gamma_{5-15}(t = 0)}{11} \sum_{r=5}^{15} \left[1 - \exp\left(-\frac{1.257(r/b_0)^2}{r_c(t)^2}\right)\right]. \tag{2.30}$$

By equating Equation (2.30) and Equation (2.19) one can find $r_c$ as an implicit function of $\Gamma$.

Equation (2.29) is related to Equation (2.27), but uses the fact that the vortex pair has a fixed separation when descending together. A simulation with only two active vortices is known as Out of Ground Effect, see below.

The rest of this chapter will be devoted to explaining the ground effects.

2.3.3. The Ground Effects

As mentioned, there are different phases of the model. For historical reasons, we call these phases ground effects. Ground effects are intended to capture the fact that vortices behave differently depending on which altitude they are generated. Thus, different ground effects will be present on different altitudes. Vortices do not necessarily need to go through all of the phases during their lifetime. For instance, a typical scenario for vortices that originate close to the ground is that they will never enter Phase I but instead start in Phase II or III.

Generally, the rule is that the closer to the ground the more complex the ground effect you have to use. This is because the proximity to the ground will require the
addition of secondary vortices that change the behavior of the system. We will clarify how secondary vortices work when describing In Ground Effect, see below.

In addition to secondary vortices, another key concept in explaining the ground effects is the concept of mirror vortices. Mirror vortices are imaginary vortices introduced below ground-level in the simulation with the purpose of stopping their real companions from falling through the ground. A mirror vortex always has the same \( y \)-coordinate as its companion but a \( z \)-coordinate of opposite sign. It also always rotates in the opposite direction of the vortex it is mirroring.

![Figure 2.5. Illustration of a mirror vortex. It has the same altitude as its real counterpart, except that its sign is negative. Also, the mirror vortex rotates in the opposite direction.](image)

The ground effects will be presented in ascending order of complexity. Thus, Effect I is the simplest and operates at the highest altitude. In the P2Pm model there are three distinct effects. The first two of these are analogous to Phase I and Phase II of NASA [4] respectively, whereas the third one corresponds to Phases III and IV.

Note however that since there are only three effects, it will be easier to assign a descriptive name to each of them rather than referring to them using numbers.

**Phase I - Out of Ground Effect - OGE**

*Out of Ground* is active whenever we are modeling the system far above the ground. Typically, this will be the ‘fallback’-effect in the sense that there is no explicit criterion for when to run this effect. Instead we use Out of Ground whenever none of the other effects is active.

During OGE there are just the two initial main vortices participating in the simulation. Because of that, one wants to avoid calculating the Biot-Savart contributions and instead uses an explicit formula, as already described in Section 2.3.2.

Figure 2.6 gives a schematic illustration and Figure 2.7 gives an example of vortex movement during OGE.

**Phase II - Near Ground Effect - NGE**

*Near Ground Effect* is initiated once both vortices fall below the predefined height \( h = 1.5 \). The vortices are not supposed to fall through the ground, so two
2.3. THE WAKE-VORTEX MODEL

Figure 2.6. Illustration of Out of Ground Effect. The two vortices are descending together toward the ground.

Figure 2.7. Illustration of a typical vortex time-trajectory during OGE. We see both main vortices (there are two graphs, one for each vortex) starting off at $z \approx 15$ and descending together until their circulation has died after which they just float with the wind.

Mirror vortices are introduced; one for each real vortex. Each real vortex is affected through Biot-Savart by the three other vortices (the other real one and the two mirror vortices). The result is that the descent of the real vortices slows down and instead they are pushed apart along the $y$-axis. Figure 2.8 illustrates the idea behind NGE.

NGE is a ‘transition’ ground effect, i.e., vortices seldom stay in NGE during a whole simulation. For instance, in Figure 2.2, the vortices start in Out of Ground Effect, transition to Near Ground Effect when they fall below $z = 1.5$ and finally also enter In Ground Effect (can be seen in the figure as the point when they start moving upwards). Note, however, that it is very hard to visually distinguish NGE in Figure 2.2.
CHAPTER 2. PHYSICAL MODEL

Figure 2.8. Illustration of Near Ground Effect. The two vortices are approaching the ground and give rise to two mirror vortices below ground level. The effect of the mirror vortices is to slow down the vertical movement and also to push the vortices apart along the $y$-axis.

**Phase III - In Ground Effect - IGE**

*In Ground Effect* is by far the most complicated effect. There are different criteria for each main vortex to initiate IGE. Let

$$\bar{W}_y \overset{\text{def}}{=} \bar{W} \cdot e_y$$  \hspace{1cm} (2.31)

be the $y$-component of the mean wind defined in Section 2.2.1. For the first primary vortex the criterion is

$$z \leq Z_1 \overset{\text{def}}{=} \begin{cases} 1.0 & \bar{W}_y \leq -1 \\ 0.6 - 0.4\bar{W}_y & -1 < \bar{W}_y < 0 \\ 0.6 & 0 < \bar{W}_y \end{cases} \hspace{1cm} (2.32)$$

For the second primary vortex we have similarly

$$z \leq Z_2 \overset{\text{def}}{=} \begin{cases} 0.6 & \bar{W}_y < 0 \\ 0.6 + 0.4\bar{W}_y & 0 < \bar{W}_y < 1 \\ 1.0 & 1 \leq \bar{W}_y \end{cases} \hspace{1cm} (2.33)$$

These conditions are intended to emulate the so-called *adapted rebound*, where the wind causes the vortices to enter IGE at different heights. It is further discussed in Rytter [1].

Once a main vortex has entered IGE it will stay in that state. When just one vortex is IGE the other vortex will give rise to only a mirror vortex while still interacting with *all* the other (mirror) vortices in the simulation.

In IGE we introduce not only mirror vortices, but also *secondary vortices* that orbit their main vortex companions. Secondary vortices are virtual. That is, they do not exist physically, but are there to 'aid' the real vortices in behaving as they should. The result of adding secondary vortices to the simulation is that they will push the main vortices upward. Every secondary vortex also gives rise to a
2.3. THE WAKE-VORTEX MODEL

There are multiple secondary vortices as well as mirror vortices, causing complex behavior. The secondary vortices are orbiting the main ones, pushing them further up in the atmosphere.

In Figure 2.9 we have illustrated IGE. From Figure 2.9 it is easy to see why we have mentioned that IGE is the most complex ground effect. Also refer to Figure 2.12 where the characteristic upwards vortex-movement of IGE can be seen.

There are two secondary vortices to be introduced for each main vortex. The first one is introduced once IGE is entered and the second one is introduced once the first one has rotated $180^\circ$ around its main vortex.

When referring to a secondary vortex’s angle $\alpha$, we mean the angle where $\alpha = 0^\circ$ corresponds to a position directly below its respective main vortex and $\alpha = -90^\circ$ positions the secondary vortex between the two main vortices (so $\alpha = -90^\circ$ for a left-side secondary vortex puts it to the right of its main vortex whereas a right-side secondary vortex ends up to the left of its main companion), as shown in Figure 2.10.

Let $z_0$ denote the initial altitude of the main vortices, i.e., at time $t = 0$. Then the starting position for every secondary vortex is at an angle of $\alpha_0 = -45^\circ$ and at the distance $R$ from its main vortex, where

$$R = \begin{cases} 0.4 + (0.4/0.5)(z_0 - 0.5) & z_0 < 0.5 \\ 0.4 & z_0 \geq 0.5 \end{cases} \quad (2.34)$$

The strength of a secondary vortex $i$ is different depending on its angle and whether it belongs to vortex 1 or 2. Let $\alpha_i$ be its angle and $W_y = \mathbf{W} \cdot \mathbf{e}_y$ be the $y$-component of the mean wind as defined in Section 2.2.1. Then

$$\sigma_i(t, \mathbf{W}) = -f(\alpha_i)G_1(W_y)\Gamma_{5-15}(t), \quad i \in \{3, 5\} \quad (2.35)$$

$$\sigma_i(t, \mathbf{W}) = f(\alpha_i)G_2(W_y)\Gamma_{5-15}(t), \quad i \in \{4, 6\} \quad (2.36)$$

where

$$G_1 = \begin{cases} 0.2 & W_y \leq -1 \\ 0.3 + 0.1W_y & -1 < W_y < 0 \\ 0.3 & 0 < W_y \end{cases} \quad (2.37)$$
Figure 2.10. Illustration of how angles are measured for secondary vortices. In the picture, both secondary vortices have an angle of approximately $45^\circ$.

are factors that emulate the adapted rebound discussed in Rytter [1]. The sign difference in Equations (2.35) is due to the fact that secondary vortices rotate in the opposite direction of their primaries. The function $f(\alpha)$ was introduced in NASA [4] (where it is called ‘thetafac’) in order to model the fact that the secondary vortices affect their primary companions differently depending on their relative angles. The function $f(\alpha)$ is defined as the piecewise linear $360^\circ$-periodic function given by Figure 2.11 below.

Note also that the $\sigma_i$ possess a property we will be using in Chapter 6: they only depend on the wind during weak cross-wind. That is, $G_1$ and $G_2$ are both constant whenever $|\bar{W}_y| \geq 1$. As long as that condition holds we can be sure that the wind will not affect $\sigma_i$. 

\begin{equation}
G_2 \overset{\text{def}}{=} \begin{cases} 
0.3 & W_y < 0 \\
0.3 - 0.1W_y & 0 < W_y < 1 \\
0.2 & 1 \leq W_y 
\end{cases}
\end{equation} (2.38)
2.3. THE WAKE-VORTEX MODEL

Figure 2.11. Plot of the 360°-periodic ‘thetafac’ [4] function $f(\alpha)$.

Figure 2.12. Illustration of a vortex time-trajectory during IGE. We see both main vortices influenced by the orbiting secondary vortices (dashed lines), rendering their trajectories slightly unpredictable. The vortices start at $z \approx 0.7$ and are pushed upward by the secondary vortices.
Chapter 3

ODE System

Having described the physical model, we now turn to the task of defining the system of ordinary differential equations that will be used to calculate the vortices’ movements. The ODE system is quite complex and contains many different state variables with dissimilar purposes. Therefore, we will start out by constructing some auxiliary mathematical tools that will simplify the presentation of the system.

In Chapter 2 we explained that the vortex coordinates can be referred to by e.g.

\[ r_i = (x_i, y_i, z_i) \]  

(3.1)

where \( i \) is the number assigned to a specific vortex, see Table 2.1. From now on we will also need to refer to the coordinates of the mirror image of vortex \( i \). These will be denoted \( r'_i \) and are defined by:

\[ r'_i = (x_i, y_i, -z_i) \]  

(3.2)

Also let \( r'_{ij} \) denote the vector from the mirror of vortex \( j \) to vortex \( i \):

\[ r'_{ij} \overset{\text{def}}{=} r_i - r'_j \]  

(3.3)

3.1. Mathematical Definitions

As mentioned previously, we want to make some mathematical definitions in order to allow the ODE to be formulated more elegantly. For instance, we will be able to ’encode’ each ground effect condition using mathematical expressions. In doing this, we will make heavy use of the Heaviside function, which we will denote by \( H(x) \). Its definition is

\[ H(x) = \begin{cases} 0 & x < 0 \\ 1 & x \geq 0 \end{cases} \]  

(3.4)

The Kronecker delta \( \delta_{ij} \) will also be important:

\[ \delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases} \]  

(3.5)
Figure 3.1. Graph of all the dependencies of the intermediate functions defined in Section 3.1.

Figure 3.1 provides an overview of all the functions that will be defined in this section and how their interdependencies work.

3.1.1. The Vortex Index Functions

Table 2.1 displays the numbering scheme used for the vortices. It will often be the case that certain functions are the same for all left-side vortices, and vice versa on the right-hand side. It will therefore be useful to translate a vortex’s index number to its corresponding main vortex’s index number.

Define the function \( m : \{1, 2, \ldots, 6\} \to \{1, 2\} \) to be the map from each vortex index to the corresponding main vortex index for that side. That is, left-side vortex indices are mapped to 1 and right-side ones are mapped to 2. Using the enumeration from Table 2.1 we can formulate this as
3.1. MATHEMATICAL DEFINITIONS

\[ m(i) = \begin{cases} 
  1 & \text{if } i \text{ is odd} \\
  2 & \text{if } i \text{ is even} 
\end{cases} \] (3.6)

This will allow us to define e.g. only \( G_1, G_2 \) below instead of \( G_1, G_2, G_3, \ldots, G_6 \) because \( G_1, G_3, G_5 \) would all be the same.

Furthermore, let \( s(i) = m(i) + 2 \) (3.7)

so that \( s(i) \) is the index of the first secondary vortex on the same side as vortex \( i \).

3.1.2. Representing the \( Z \) and \( G \) functions

Using the Heaviside-function, we can define mathematical expressions for \( Z_i \) and \( G_i \) that we discussed in Section 2.3.3, specifically in Equations (2.35) and (2.37)-(2.38).

\[
Z_1 = 0.6 + 0.4H(-\bar{W}_y) \left[ |\bar{W}_y|H(1 - |\bar{W}_y|) - (|\bar{W}_y| - 1)H(|\bar{W}_y| - 1) \right] 
\] (3.8)

\[
Z_2 = 0.6 + 0.4H(\bar{W}_y) \left[ |\bar{W}_y|H(1 - |\bar{W}_y|) - (|\bar{W}_y| - 1)H(|\bar{W}_y| - 1) \right] 
\] (3.9)

\[
G_1 = 0.3 - 0.1H(\bar{W}_y) \left[ |\bar{W}_y|H(1 - |\bar{W}_y|) - (|\bar{W}_y| - 1)H(|\bar{W}_y| - 1) \right] 
\] (3.10)

\[
G_2 = 0.3 - 0.1H(-\bar{W}_y) \left[ |\bar{W}_y|H(1 - |\bar{W}_y|) - (|\bar{W}_y| - 1)H(|\bar{W}_y| - 1) \right] 
\] (3.11)

As we see, these are all functions of the mean wind, which depends on \( x, y_1, y_2, z_1 \) and \( z_2 \).

3.1.3. The Ground Effect Conditions

Given the complex logic for determining e.g. when In Ground Effect is active, we will formulate indicator functions that will be 1 whenever the conditions are satisfied and 0 otherwise.

We denote the NGE-condition function by \( \mathcal{N}(z_1, z_2) \) (not to be confused with the stratification \( N_i(x, y_i, z_i) \)) and define it:

\[
\mathcal{N}(z_1, z_2) = H(h - z_1)H(h - z_2) 
\] (3.12)

which thus evaluates to 1 only when both main vortices are below \( h \), in accordance with Section 2.3.3. Note that \( \mathcal{N} \) will evaluate to 1 even when the vortices have fallen so low that IGE should activate.

The task of writing an indicator function for IGE is a little bit trickier. Remember that both main vortices can enter IGE independently. Thus we will have to define two indicator functions; one for each side. Another caveat is that the logic for IGE states that we should not ever leave IGE once it has commenced. Since
the Biot-Savart interactions in IGE invariably cause the vortices to move relative to each other, we can restate that condition as ‘the secondary vortices have rotated away from their initial positions’. Thus, let $\alpha_3$ and $\alpha_4$ represent the secondary vortex angles of vortices 3 and 4 respectively. Then the IGE indicator functions are

\[ I_1(x, y_1, y_2, z_1, z_2, \alpha_3) = H(H(Z_1 - z_1) + \alpha_3 - \alpha_0) \]  
(3.13)

\[ I_2(x, y_1, y_2, z_1, z_2, \alpha_4) = H(H(Z_2 - z_2) + \alpha_4 - \alpha_0). \]  
(3.14)

where $\alpha_0 = -45^\circ$ is the initial angle of the secondary vortices. The $I_1, I_2$-functions are 1 only if the secondary vortex angles changed or if the vortices are under their respective altitude-threshold for initiating IGE. See Section 2.3.3.

Finally, we will also need an expression for when the second secondary vortices activate:

\[ S_1(\alpha_3) = H(\alpha_3 - 180^\circ - \alpha_0) \]  
(3.15)

\[ S_2(\alpha_4) = H(\alpha_4 - 180^\circ - \alpha_0) \]  
(3.16)

that is, we are checking ‘has the first secondary vortex rotated 180° relative to its initial angle $\alpha_0$ yet?’.

### 3.1.4. The Vortex Circulation Strengths

The vortices have different circulation strengths depending on whether they are secondary or not, and depending on which side they are on. It will be cumbersome to write out each vortex’s circulation directly in the ODE. Hence, we define the functions $\sigma_i$ representing the circulation strength of each vortex $i$:

\[ \sigma_i(t, x, y_1, y_2, z_1, z_2, \alpha_3, \ldots, \alpha_6) = (-1)^i \Gamma(t) \times \begin{cases} 1 & i \leq 2 \\ f(\alpha_i)G_{m(i)} & i > 2 \end{cases} \]  
(3.17)

where the factor $(-1)^i$ represents the fact that left and right vortices rotate in opposite directions. The $\alpha_i$ are the secondary vortex angles introduced above. The function $f(\alpha)$ used here is the ‘thetac’-function, see Figure 2.11. The map $m(i)$ was defined in Section 3.1.1 and $\Gamma(t)$ is the $\Gamma_{5-15}(t)$ of Section 2.2.2.

### 3.1.5. The Vortex Activation Functions

Different vortices have different conditions for when they are to activate. Just as for circulation strength, it is cumbersome to incorporate this logic directly into the ODE formulation. Thus, let

\[ A_i(x, y_1, y_2, z_1, z_2, \alpha_{s(i)}) = \begin{cases} 1 & 1 \leq i \leq 2 \\ I_{m(i)} & 3 \leq i \leq 4 \\ I_{m(i)}S_{m(i)} & 5 \leq i \leq 6 \end{cases} \]  
(3.18)

be indicator functions evaluating to 1 whenever their respective vortices are to activate. The $\alpha_i, I_i$ and $S_i$ were all defined in Section 3.1.3.
3.2. FORMULATING THE ODE SYSTEM

3.1.6. The Biot-Savart Tensors

The Biot-Savart interactions were explained in Section 2.3.2. Now that we have made suitable definitions for denoting vortices and their circulation strength, we are ready to define $v_{ij}$. Its definition will be very similar to that of the physical model:

$$v_{ij} = v_{ij}(x, y_i, y_j, z_i, z_j) = \sigma_j(\ldots) \frac{r_{ij}}{|r_{ij}|^2} \times e_x$$  \hspace{1cm} (3.19)

where $\sigma_j$ was defined in Section 3.1.4 and the notation $r_{ij}$ was discussed in the beginning of Chapter 2. However, we also have to deal with mirror vortex contributions. As such, let

$$v'_{ij} = v'_{ij}(x, y_i, y_j, z_i, z_j) = -\sigma_j(\ldots) \frac{r'_{ij}}{|r'_{ij}|^2} \times e_x$$  \hspace{1cm} (3.20)

represent the contribution from the mirror of vortex $j$ to vortex $i$. The minus sign is there because mirror vortices rotate in the opposite direction of their companions.

Just as in the physical model, $v_i$ will be the total Biot-Savart contribution to vortex $i$. However, because of the special case in OGE (see Section 2.3.2), the expression will be a little more complicated than in Section 2.3.2. We have to define it so that it chooses the right $v_i$ depending on the prevailing Ground Effect. Let $N$ be the NGE indicator function that was defined in Section 3.1.3. Then

$$v_i = Nv_i^{BS} + (1 - N)v_i^{OGE},$$  \hspace{1cm} (3.21)

where

$$v_i^{BS} = \sum_{j=1}^{6} A_j \left( (1 - \delta_{ij})v_{ij} + v'_{ij} \right)$$  \hspace{1cm} (3.22)

$$v_i^{OGE} = \left[ 1 - \exp\left( -\frac{(0.4b_0)^2 1.257}{r_c(\Gamma(t))} \right) \right] e_z.$$  \hspace{1cm} (3.23)

We neglected to write out all the dependencies above because it would clutter up the notation. In fact $v_i$ depends on all coordinate pairs $(y_k, z_k)$ where $k$ is an index of an active vortex. If $k$ is an index of a secondary vortex, then $v_i$ will also depend on its secondary angle $\alpha_k$ as discussed in Section 3.1.4. Furthermore, $v_i$ depends on the time $t$ and the $x$-coordinate of the gate (explained in Section 2.1).

We used the Kronecker delta in Equation (3.22) to omit the term $v_{ii}$ which is undefined. The $A_j$ is meant to filter out vortices that are inactive.

3.2. Formulating the ODE System

We have made all the definitions that we need. We will now show how to construct the ordinary differential equation that incorporates the information in the physical model and describes wake-vortex movement.
It can be very difficult to get a good overview of the ODE we are about to describe because of the many different variables with dissimilar purposes. Thus we would recommend the reader to consult Figure 3.2 below, were we have tried to visualize all the direct inter-variable dependencies.

As has been already stated, the system we formulate is a standard Ordinary Differential Equation:

\[
\dot{\mathbf{u}} = f(\mathbf{u}, t) \tag{3.24}
\]

with the state vector

\[
\mathbf{u} = \begin{pmatrix}
x \\
B_1 \\
B_2 \\
\alpha_3 \\
\vdots \\
\alpha_6 \\
y_1 \\
z_1 \\
\vdots \\
y_6 \\
z_6
\end{pmatrix} \tag{3.25}
\]

where the dot in the left-hand side denotes a time-derivative and \( t \) denotes time.

We will attempt to give physical meaning to the state vector \( \mathbf{u} \), and define the function \( f(\mathbf{u}, t) \) component-wise.

3.2.1. The \( x \)-Coordinate of the Gate

Denoted by \( x \). The time-derivative \( \dot{x} \) is only affected by the wind. Hence it is a function of the main vortices \( y \)- and \( z \)-coordinates as well as of \( x \):

\[
\dot{x} = \bar{\mathbf{W}}(x, y_1, y_2, z_1, z_2) \cdot \mathbf{e}_x \tag{3.26}
\]

that is, it is equal to the \( x \)-component of the mean wind vector \( \bar{\mathbf{W}} \) defined in Section 2.2.1. As mentioned in Chapter 2, each vortex will have the same \( x \)-position.

3.2.2. The Buoyancy Factors

Denoted by \( B_1 \) and \( B_2 \). The buoyancy factors are functions of the main vortex positions and also of the vector \( \mathbf{v}_i \). For \( i \in \{1, 2\} \):

\[
\dot{B}_i = \frac{181}{400} N(x, y_i, z_i)^2 \sqrt{2}(z_0 - z) \tag{3.27}
\]

That is, Equation (3.27) is the same as Equation (2.20).

Even though there can be up to 6 vortices there will only be two buoyancy factors; one for each main vortex. The rationale is that the secondary vortices are
3.2. FORMULATING THE ODE SYSTEM

Figure 3.2. Graph showing how the ODE variables depend on the intermediate functions and on each other. It contains the graph of Figure 3.1 as a subgraph. Note that the graph is so complex because it incorporates the dependencies of each ground effect. For ground-effect-specific dependency-graphs, see the end of this chapter.

merely supposed to ‘aid’ their corresponding main vortex. They are not real physical vortices and as such should use the same buoyancy factor as their counterparts in order to follow their motion.
3.2.3. The Secondary Vortex Angles

Secondary vortices were explained in Section 2.3.3. We introduce one angular-variable per secondary vortex so that we can e.g. keep track of when they have rotated 180° (see Section 2.3.3). There are thus four secondary vortex angles, but we will subscript them with the number of the particular vortex they are representing. Thus the variables are: \( \alpha_3, \alpha_4, \alpha_5, \alpha_6 \), c.f Table 2.1.

For \( i \in \{3,4,5,6\} \) we have:

\[
\dot{\alpha}_i = A_i \left( x, y_1, y_2, z_1, z_2, \alpha_s(i) \right) \frac{\mathbf{r}_{m(i)} \times (\mathbf{v}_i - \mathbf{v}_{m(i)})}{|\mathbf{r}_{m(i)}|} \tag{3.28}
\]

which is a standard formula from solid mechanics for calculating angular velocity, multiplied by the IGE indicator function. Notice that \( \mathbf{v}_i - \mathbf{v}_{m(i)} \) is the relative velocity of vortex \( i \) with respect to its main vortex.

Remark on Redundancy

As explained in Section 2.3.3, during IGE a second secondary vortex is to be introduced once the first secondary vortex rotates 180° around its main companion. This requires us to somehow keep track of the secondary vortices’ angles throughout a simulation. We do this by introducing one angular variable per secondary vortex. Since we also save each vortex’s \( y \)- and \( z \)-coordinates this creates some redundancy.

Had we saved a radius \( r \) per secondary vortex we could just calculate \( y \) and \( z \) instead. Thus it is in fact possible to formulate the same ODE with a smaller number of variables. However, for our purposes the numerical solution time is most important, and the alternative approach requires more trigonometric computations, which would detract from the possible gains of reducing the variable count.

3.2.4. The Vortex Coordinate Pairs

Just as in Section 2.3.1, let

\[
\begin{align*}
\dot{y}_i & = \left[ \mathbf{v}_i + \mathbf{W}(x, y_i, z_i) \right] \cdot \mathbf{e}_y \tag{3.29} \\
\dot{z}_i & = \mathbf{v}_i \cdot \mathbf{e}_z B_i \tag{3.30}
\end{align*}
\]

for the main vortices, i.e., for \( i \in \{1,2\} \).

When dealing with the secondary vortices, the equations become slightly more complicated. The secondary vortices need to follow the movement of their main counterparts until the moment when they themselves become active. Furthermore, they use the same wind as their main vortices [1]. Thus, for \( i \in \{3,4,5,6\} \),

\[
\begin{align*}
\dot{y}_i & = \left[ A_i \mathbf{v}_i + \left( 1 - A_i \right) \mathbf{v}_{m(i)} + \mathbf{W}(x, y_{m(i)}, z_{m(i)}) \right] \cdot \mathbf{e}_y \tag{3.31} \\
\dot{z}_i & = \left[ A_i \mathbf{v}_i + \left( 1 - A_i \right) \mathbf{v}_{m(i)} \right] \cdot \mathbf{e}_z B_{m(i)} \tag{3.32}
\end{align*}
\]
Simplification During OGE

Consider the $y$- and $z$-derivatives defined above. Equations (3.21) and (3.23) show that during OGE, $v_i \cdot e_y = 0$. Hence for $i = 1, 2$, these formulae hold:

$$\dot{y}_i = W(x, y_i, z_i) \cdot e_y \tag{3.33}$$
$$\dot{z}_i = v_i(t, \ldots) \cdot e_z B_i(t) \tag{3.34}$$

Or, by inserting from (3.23):

$$\dot{y}_i = W(x, y_i, z_i) \cdot e_y \tag{3.35}$$
$$\dot{z}_i = 1 - \exp\left(-\frac{(0.4b_0)^21.257}{r_c(\Gamma(t))}\right)B_i(t) \tag{3.36}$$

We will make use of this simplification in Chapter 6.

3.2.5. Initial Values

Given an initial height $z_0$ to run the simulation on, the initial values are as follows:

$$u_0 = \begin{pmatrix} x \\ B_1 \\ B_2 \\ \alpha_3 \\ \vdots \\ \alpha_6 \\ y_1 \\ z_1 \\ y_2 \\ z_2 \\ y_3 \\ z_3 \\ y_4 \\ z_4 \\ y_5 \\ z_5 \\ y_6 \\ z_6 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ \alpha_0 \\ \vdots \\ \alpha_0 \\ -0.5 \\ z_0 \\ 0.5 \\ z_0 \\ -0.5 - R \sin(\alpha_0) \\ z_0 - R \cos(\alpha_0) \\ 0.5 + R \sin(\alpha_0) \\ z_0 - R \cos(\alpha_0) \\ -0.5 - R \sin(\alpha_0) \\ z_0 - R \cos(\alpha_0) \\ 0.5 + R \sin(\alpha_0) \\ z_0 - R \cos(\alpha_0) \end{pmatrix} \tag{3.37}$$

The initial vortex angle $\alpha_0 = 45^\circ$ and $R$ were explained in Section 2.3.3.

Note that the initial conditions are invariant with respect to aircraft size, as explained in Section 2.1.

3.2.6. Ground Effect-Specific Dependencies

Part of the complexity in the ODE seen in Figure 3.2 comes from the fact that it must be able to switch ground effects. Had we dealt with each effect separately,
the graphs would have looked simpler, as shown in Figures 3.3, 3.4 and 3.5. The task of formulating a new model that treats each ground effect as a separate ODE is suggested as a future development in Section 7.2.

Figure 3.3. Graph of all dependencies during OGE. Note that $\alpha_i = 0$ always, which is why it is depicted by an isolated node.
3.2. FORMULATING THE ODE SYSTEM

$B_i \epsilon_{B_i v_{ij}} z_i y_i x_i N v_{BS i} \alpha_i t W_1, W_2 v_i y_i x_i z_i \sigma_i \Gamma \sigma_i v_{ij} v_i v_{i,BS} \dot{v_i} \dot{B_i} \dot{z_i} \dot{\alpha_i}$

Figure 3.4. Graph of all dependencies during NGE. Note that $\alpha_i = 0$ always, which is why it is depicted by an isolated node.
```
\textbf{CHAPTER 3. ODE SYSTEM}

\section*{Figure 3.5.} Graph of all dependencies during IGE.
```
Chapter 4

Implementation

We have defined the physical model and showed how to represent it using an ODE. We will now describe how the ODE is solved numerically. Due to the intention of performing Monte Carlo-simulations, the algorithm that we use needs to execute quickly. In fact, emphasis has been given to speed over accuracy, seeing as we are more interested in studying how the solution changes as a result of the Monte Carlo-variations than in the accuracy of any single solution.

4.1. Solving on the Graphics Card

GPUs excel at performing arithmetic-intense calculations in parallel. The IVPs we wish to solve are all independent, and as such are excellent candidates for parallelization. The numerical scheme is therefore implemented on the GPU using the OpenCL-toolkit.

It should be pointed out that there are multiple ways of achieving parallelism for the problem we are trying to address. For instance, one might attempt to parallelize even the solution of a single IVP, e.g. by calculating each ODE-variable in a parallel thread of execution. The difficulty in this approach, lies in the fact that the variables are far from independent, as is evident from Figure 3.2. The task of achieving parallelism using this approach is thus clearly nontrivial, and is not attempted in this thesis.

4.2. Solver Algorithm

The Runge-Kutta4 (RK4) algorithm is employed to solve the IVPs. It is a well known method of order 4. Note that the implementation uses a fixed stepsize, even though it is possible to write an adaptable RK4 algorithm.

Nevertheless, as we will see in Chapter 5, the main reason for low accuracy in the solution is not due to the algorithm itself but because of discontinuities within the ODE. It is true that an adaptive scheme would be able to bypass this problem by reducing the stepsize repeatedly until the exact point of the discontinuity is found.
However such a process would require a large overhead due to having to reevaluate the points close to a discontinuity so many times.

Additionally, by employing an adaptive scheme, one would run the risk of losing homogeneity. Meaning that solving different IVPs in the Monte Carlo-simulation might require different stepsizes, which in turn requires branching. That would have bad consequences for performance on a GPU device[7].

4.3. The Weather Grid

We mentioned in Section 2.2.1 that weather data is supplied by a large weather grid, from which the weather at any particular position can be calculated through interpolation.

The grid consists of several layers. Layers are stacked almost directly on top of each other so that corresponding grid points from different layers almost align. By ‘almost’ we mean that the grid is typically constructed from real-world measurements, and the position of each measurement can be off by a couple of meters.

Typically, the layers are vertically separated by approximately 300 m, and grid points of the same layer tend to be approximately 10 km apart.

Prior to a simulation, the weather grid is queried for weather data in the area that the simulation is to figure in. A query for weather data usually results in three sets of grid points; two sets of points from the closest layers below the requested position, and another set of points for the closest layer above it.

Each grid point contains information about the wind vector, turbulence and stratification at that position. The wind vector at a specific point within the grid is calculated as a weighted average of all the requested grid points, where the weight is taken to be the reciprocal of the squared distance to each point. Thus, if we call the \( k \) requested grid points \( p_1, p_2, ..., p_k \) we can write the formula for the wind at a specific point \( p = (x, y, z) \):

\[
W(x, y, z) = \frac{\sum_{i=0}^{k} \frac{W(p_i)}{|p - p_i|^2}}{\sum_{i=0}^{k} \frac{1}{|p - p_i|^2}}
\]  

(4.1)

In contrast, turbulence is calculated as a standard average across all grid points. The reason is that turbulence affects the vortex circulation. If it were allowed to change as a function of position, the circulation would no longer be guaranteed to be continuous. Thus turbulence \( E \) has the formula:

\[
E = \frac{1}{k} \sum_{i=0}^{k} E(p_i)
\]  

(4.2)

As for the stratification, we will actually have two different formulae. The rationale is that stratification, just as turbulence, affects vortex lifetime and as such

34
4.4. THE MEMORY SETUP

must be kept constant in the circulation formula. Thus the circulation function uses the constant stratification:

\[ N_{\text{circulation}} = \sum_{i=0}^{k} \frac{S(p_i)}{k} \]  

(4.3)

However, the other use of stratification is in the buoyancy factors, and in this case we really want its value to vary. Thus in this case we employ the weighted average:

\[ N_{\text{buoyancy}}(x,y,z) = \frac{\sum_{i=0}^{k} S(p_i) |p - p_i|^2}{\sum_{i=0}^{k} 1 |p - p_i|^2} \]  

(4.4)

Finally, in order to use the weather data within the ODE one must convert it to use the local coordinate system. This includes rotating the wind according to its relative bearing to the gate, as well as converting to use the right units [1]:

\[ W \, [\text{length units/time units}] \leftarrow b_0/t_0 W \, [ms^{-1}] \]  

(5.5)

\[ N \, [\text{time units}] \leftarrow Nt_0 \, [s^{-1}] \]  

(4.6)

\[ \epsilon \, [\text{length units}^2/time \, \text{units}^3] \leftarrow \epsilon^2 t_0 b_0 \, [m^2s^{-3}] \]  

(4.7)

4.4. The Memory Setup

We now explain the memory setup in the implementation. Choosing a memory setup has performance implications. For instance, excessive use of global memory should be avoided when other, faster memory types are available. It is nevertheless necessary to use global memory occasionally because it is the only memory type suitable for storing large amounts of data. Note that this discussion will require some understanding of the OpenCL memory types: private, local, global and constant. For information on these, see e.g. Gaster et al [5].

4.4.1. Weather Grid Data

Weather data is stored in constant memory since it is shared throughout all concurrent simulations. Each weather point requires six parameters: wind strength, wind direction, stratification and position \((x,y,z)\). The total required amount of memory using single precision storage is thus \((1 + 1 + 1 + (1 \times 3)) \times 4 = 24\) bytes. As there are typically around 10 weather points per Monte Carlo-simulation this requires a total of approximately 240 bytes. The small amount of memory required makes it perfect for storage in constant memory.
4.4.2. Simulation Parameters

This is data specific to a single initial value problem. Thus the memory required for this data grows linearly as a function of the number of simultaneous simulations. Presently, the simulation parameters include the precomputed constants $\nu_1, \nu_2, T_1, T_2$ and also some parameters for the Monte Carlo-simulations: turbulence perturbation, stratification perturbation, wind strength perturbation and wind direction perturbation. Thus a total of $(1 \times 8) \times 4 = 32$ bytes per Initial Value Problem.

The simulation parameters reside in constant memory. However, there is no particular reason for storing them there, except for the fact that they are constant. They could just as well be stored in other memory types. For a thousand simulations in total, the memory requirements start approaching the maximum capacity of constant memory for some OpenCL implementations. So in order to increase the number of simulations, one would likely have to relocate this data elsewhere.

4.4.3. Runtime Parameters

These are parameters that are constant throughout a whole Monte Carlo-simulation, just like the weather data. They include only three parameters of which only one pertains to the physical model itself (the rest are implementation-specific constants required e.g. for storing memory at correct addresses). Thus it requires approximately 12 bytes and can easily be stored in constant memory.

4.4.4. The Solution Data

The solution data is a large buffer allocated in global memory where all the parallel solutions can be written. Typically, the GPU was given approximately 1000 IVPs at a time to solve. Thus, 19 variables per 1000 executions per approximately 900 time steps requires around 9 MB. Note that if increasing the number of simultaneous executions one will have to break this buffer into smaller chunks because some graphics cards (and OpenCL enabled CPUs) have a maximum limit on the size of a continuous chunk in global memory.

4.5. Accuracy vs Performance Trade-offs

Seeing as the purpose is a Monte-Carlo simulation, and that the maximum error is allowed to be fairly large (see Section 5.4), a number of performance optimizations are employed in order to gain execution speed.

4.5.1. Native Math

The GPU’s native implementations of common math functions are used in favor of the standard math library. These are accessible in OpenCL using the prefix ‘native_’, e.g. ‘native_exp’ or ‘native_cos’.
4.6. GPU IDIOSYNCRASIES

4.5.2. An $r_c(\Gamma(t))$-Lookup Table

The computations in OGE feature the calculation of an implicit function, namely:

$$r_c(\Gamma(t)).$$

This is a relatively costly operation to the otherwise very simple OGE formulae, see Section 2.3.2. Thus, we use a lookup table to gain speed, as in Rytter [1].

Note, however, that the type of interpolation can impact the accuracy of the algorithm, since it relies on the fact that the integrated function is continuous and differentiable.

4.5.3. Circulation

The circulation is given in (2.10) as an integral. However, in the implementation, we approximate the integral by a relatively crude Riemann sum:

$$\Gamma(t) \approx \frac{1}{11} \sum_{r=5}^{15} \Gamma(r/b_0, t) \quad (4.8)$$

The same ‘optimization’ is performed in both Rytter [1] and Holzäpfel [2].

4.6. GPU Idiosyncrasies

There were a number of interesting obstacles encountered throughout the development of the numerical scheme which pertain to the fact that we are developing on a GPU. It would be worthwhile to outline them here so that future work need not repeat the same trouble-shooting.

4.6.1. Debugging on the GPU

At the time of writing, there is practically no available support for debugging the OpenCL code. Some error reporting can be achieved by specifying slightly larger differential equation system than required. For instance, we require 19 variables to formulate the ODE, but when implementing, 24 variables can be used instead so as to have 5 variables that the numerical scheme completely ignores but that could be useful for outputting various data that could be helpful in trouble-shooting.

4.6.2. Vectorization Speedups

Modern GPUs and CPUs support so called Single Instruction Multiple Data (SIMD) operations, which allow for a single thread to do many computations in parallel. OpenCL supports these operations through the use of its vector datatypes, e.g. float4, float16 etc.

It was found that the recommended approach to using these kinds of operations varies depending on the device. For instance, Intel [10] recommends that vector
operations be explicitly used as much as possible, whereas NVIDIA recommends that they only be used at leisure. From *NVIDIA OpenCL Best Practices* [7], chapter Instruction Optimizations:

[...] The CUDA architecture is a scalar architecture. Therefore, there is no performance benefit from using vector types and instructions. These should only be used for convenience. [...] 

It should be pointed out that using vector operations in the ODE appeared to increase CPU performance but to decrease it on the GPU.

### 4.6.3. Floating Point Accuracy

Presently, even the most cutting edge GPUs struggle with double precision arithmetic efficiency, and single precision accuracy can be orders of magnitude faster[6]. Therefore, NVIDIA recommends that single-precision arithmetic be used at all times[7]. However, single precision accuracy means that the numerical analysis of the ODE becomes more difficult, e.g. because we get round-off errors. This is important to bear in mind, especially in applications where numerical stability is of the essence. It might therefore be worthwhile to define something like

```c
#if SINGLE_PRECISION
    typedef float real;
#else
    typedef double real;
#endif
```

and using `real` throughout all OpenCL code so as to easily be able to change precision for doing e.g. convergence studies.

### 4.6.4. Data Transfer

One has to keep in mind that everything the GPU calculates will have to reside in GPU memory, which is separate from the RAM accessible to the CPU. Before starting execution, one must thus transfer all initial data, including physical parameters and weather data, to the GPU. Once the solutions have been computed, one must also either transfer all the solutions back to main memory or carry out all necessary work on the GPU. It can be preferable to run code on the GPU which would run equally fast (or even faster!) on the CPU simply due to the data transfer overhead and lack of parallelism [7].

In our case, we are interested in e.g. computing the mean and variance of all the solutions. This is a typical instance were it is more efficient to do the work on the graphics card. That is, instead of transferring all the solutions to the main memory and then performing the calculations in serial on the CPU, we can parallelize the computations and then just transfer the results, saving both execution- and transfer-time.
4.6. GPU IDIOSYNCRASIES

4.6.5. Memory Buffer Limitations

It was discovered that even though modern GPUs can have up to 4 GBs of memory, there is still a restriction on the size of a contiguous memory buffer. Thus it is not quite as straight-forward as one might think to increase the number of concurrent threads. Presently, the required number of threads that are enqueued on the GPU (one thousand), appears to be just enough so that the solutions can fit nicely inside a single buffer in GPU memory. However, simply increasing the number to 10 000 will not work unless one splits up the solution buffer into say 10 different parts.
Chapter 5

Analysis of the Rate of Convergence

Above, we described the implementation of an RK4-algorithm. RK4 should, by virtue of being a fourth order method, have an absolute error proportional to the stepsize to the fourth power. We will now discuss under which circumstances this holds true when solving the wake-vortex ODE.

In performing the numerical convergence studies in this chapter, we have investigated how the solution converges to a reference solution. That is, we compute a high accuracy solution and use that as a good approximation of the ‘true’ solution, so that we can compute the absolute error of less accurate solutions e.g by taking the absolute value of the difference from the reference solution. The reference solutions all have a timestep of approximately $1 \times 10^{-2}$ time units.

Note that in using a reference solution, we can only ever give reasonable approximations of the absolute error for cases when the numerical solution converges. There is no way of telling how great the error is when there is no convergence, because in that case the reference solution is not a good approximation of the exact solution.

The RK4-algorithm assumes that the differential equation be at least 5th order differentiable [8]. It is not clear that this condition is satisfied under all circumstances by the wake-vortex ODE, and this is the reason why the theoretical rate of convergence cannot always be achieved.

Also, note that the existence and uniqueness of any ODE, $y = f(t, y)$, can only

| $\epsilon$ | 0.143 |
| $N$       | 0.700 |
| wind strength | 2.000 |
| wind bearing      | 228.2° |
| gate bearing     | 298.6° |

\textit{Table 5.1.} The values of the parameters during the convergence study. There is a single value given per atmospheric property. In other words, all points in the weather grid have the same value and the weather will be constant.
be guaranteed when \( f(t, y) \) is continuous in \( t \) and Lipschitz continuous in \( y \) \([9]\). If \( f \) is discontinuous, solutions may not exist, but nevertheless they often do. Given that we are modeling a physical phenomenon it appears likely that there is a well-defined solution to the wake-vortex ODE.

Within the ODE, all ground effect transitions generally cause problems. Since the ODE changes behavior abruptly once the vortices fall below the altitude-threshold for a new effect, there will be discontinuous derivatives that disrupt the convergence.

For instance, when changing from NGE to IGE, we will see discontinuities in the derivatives of the secondary vortex angles, which throughout NGE have always been 0. This causes slower convergence in the angles, which quickly propagates to the rest of the variables through the Biot-Savart interactions. Figure 5.1 illustrates that the rate of convergence suffers as a result.

![Figure 5.1. Slow convergence in a simulation that transitions from NGE to IGE. Note that since convergence is so slow, the approximation of the absolute error cannot be trusted. See the discussion in the beginning of this chapter.](image)

Likewise, when changing from OGE to NGE the vortices will suddenly start to repel in the \( y \)-direction whereas previously the \( y \)-derivatives were zero (disregarding the wind), causing slower convergence, as shown in Figure 5.2.

There are also a number of ground-effect-specific issues to be discussed below.

### 5.1. Convergence in Out of Ground Effect

The ODE has a relatively simple structure during OGE, as can be seen from Equations (3.35)-(3.36), \( i.e., \)

\[
\dot{y}_i = W(x, y_i, z_i) \cdot e_y \\
\dot{z}_i = \left[ 1 - \exp\left( -\frac{(0.4b_0)^2 1.257}{r_c(\Gamma(t))} \right) \right] B_i(x, y_i, z_i)
\]  

(5.1)  

(5.2)
5.2. CONVERGENCE IN NEAR GROUND EFFECT

Both of these formulae are comprised of smooth continuous functions (the implicit function \( r_c(\Gamma) = \Gamma^{-1}(\Gamma(r_c)) \) is continuous by the inverse function theorem). Thus we should expect to get accurate results. Consult Figure 5.3 where the convergence is shown.

Note that in Section 4.5.2 we mentioned that the implementation uses lookup-tables and interpolation to compute \( r_c(\Gamma) \). The choice of this interpolation will impact the error convergence. Figure 5.3 displays the convergence when using Hermite Cubic Splines.

5.2. Convergence in Near Ground Effect

In NGE, the equation system is dominated by the Biot-Savart interactions:

\[
\dot{y}_i = \left[ v_i + W(x, y_i, z_i) \right] \cdot e_y \\
\dot{z}_i = v_i \cdot e_z B_i. \tag{5.3}
\]

The wind function \( W(x, y_i, z_i) \) should always be smooth since it is a model of a physical phenomenon. The Biot-Savart speed \( v_i \) in NGE is given by Equation (3.22):

\[
v_i^{RS} = \sum_{j=1}^{6} A_j \left( (1 - \delta_{ij})v_{ij} + v'_{ij} \right) \tag{5.5}
\]
where $A_3 = A_4 = A_5 = A_6 = 0$ since only vortices 1 and 2 and their mirror images are active, as explained in Section 2.3.3. Clearly, $v_i$ is a sum, and the summands

$$v_{ij} = \sigma_j (\ldots) \frac{r_{ij}}{|r_{ij}|^2} \times e_x$$

(5.6)

and

$$v'_{ij} = -\sigma_j (\ldots) \frac{r'_{ij}}{|r'_{ij}|^2} \times e_x$$

(5.7)

are all smooth and continuous (with respect to the state variables). Thus, the equation system consists entirely of continuous functions and we can expect accurate results. Figure 5.4 illustrates this.

### 5.3. Convergence In Ground Effect

In Ground Effect features the same set of equations as in Near Ground Effect, except that there are more vortices participating. However, problems will arise because the number of participating vortices increases abruptly. In adding a new vortex to the simulation we destroy the convergence. For instance, by letting $A_5 = 0$ change to $A_5 = 1$ in order to activate vortex 5, we are creating a discontinuity in the fifth summand of Equation (5.5).

Thus, the equation system (5.3)-(5.4) is comprised of discontinuous functions and Figure 5.5 shows that the accuracy suffers due to this.

Note that by making sure that the sum in Equation (5.5) is never discontinuous we can achieve just as good convergence as in Near Ground Effect. In Figure 5.6 the convergence of a modified version of IGE is shown where vortices 5 and 6 never
5.3. CONVERGENCE IN GROUND EFFECT

Figure 5.4. Convergence during NGE. Error convergence is adequate.

Figure 5.5. Convergence during IGE. Convergence does not agree with theory. Therefore we cannot expect the approximation of the absolute error to be accurate. See the discussion at the beginning of this chapter.

activate. That is, the simulation is run from beginning to end with only vortices 1, 2, 3 and 4. As can be seen, this preserves the accuracy of the numerical solution, but at the cost of never being able to change the number of participating vortices.
CHAPTER 5. ANALYSIS OF THE RATE OF CONVERGENCE

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{fig5_6.png}
\caption{Convergence during a modified IGE without second secondary vortices. Convergence is better than in Figure 5.5 because there are no discontinuities in the equation system (5.3)-(5.4) anymore.}
\end{figure}

5.4. Analysis and Proposed Improvements

We have studied how the numerical error behaves as a function of the timestep when solving the ODE using the RK4-algorithm. The RK4 method works well for ODEs without discontinuous derivatives. The wake-vortex ODE can be seen as essentially a set of continuous ODEs that have been ‘joined together’ with the help of the Heaviside function $H(x)$ (see e.g Chapter 3 where we used Heaviside to encode all the logic). Thus, we can only achieve 4th order accuracy as long as we apply the RK4-algorithm at a point where no Heaviside function makes its jump. Seeing as all ground effect transitions are governed by Heaviside functions, these must be avoided. Furthermore, it is not practical to solve the In Ground Effect with RK4 since the activation of secondary vortices within IGE is also governed by Heaviside-logic.

In order to remedy the accuracy problem, the most viable option is to amend the ODE itself. For instance, one could introduce smooth transitions between ground effects and smoothly add secondary vortices. Granted, such an approach would change the model but from a physical point of view it nevertheless makes more sense to represent physical phenomena using continuous functions.

Another option is to implement an adaptive numerical scheme which would be able to traverse the discontinuities in the Heaviside functions without losing accuracy. However, note that such a scheme would have to repeatedly decrease and recalculate its stepsize in order to deal with the discontinuities, causing a dramatic increase in computation time.
Chapter 6

Analysis of the Parameter Sensitivity

We have mentioned that the wake vortex model is to be used for simulations of real-world situations. We now want to clarify how the outcomes of these simulations are affected by small variations in the input parameters, so as to estimate the model's sensitivity to measuring errors.

We will study the effect of four parameters, namely: turbulence $\epsilon$, stratification $N$ and the wind vector's magnitude $|W|$ and direction $W/|W|$. Since the main vortices’ $y$- and $z$-coordinates are our main interest we will focus the study on only those variables.

We want to draw our conclusions through referring to plots from two sets of simulations with authentic weather data. Both setups use data from Thursday, 06 Sep 2012 00:00:00 GMT, but one is from $59^\circ39'7''$N, $17^\circ55'7''$E (Stockholm, Arlanda) whereas the other is from $25^\circ15'10''$N, $55^\circ21'52''$E (Dubai, Dubai International Airport). All parameters are varied separately, linearly from 0 up to $+20\%$, except for the wind direction which is varied from 0 up to $10^\circ$ instead. The plots display the difference (due to the parameter variations) in the left main vortex’s coordinates, $y_1$ and $z_1$, versus time. That is, the plots display the difference between a perturbed solution and a reference (unperturbed) solution:

$$\Delta y = y_1^{\text{unperturbed}} - y_1^{\text{perturbed}}$$
$$\Delta z = z_1^{\text{unperturbed}} - z_1^{\text{perturbed}}.\quad (6.1)$$

Each plot displays multiple graphs, where different graphs correspond to perturbations of different magnitude for the varied parameter.

All the simulations have a gate bearing of $90^\circ$ and initial vortex separation of $b_0 = 26.93$ m. IGE simulations run at the altitude 20 m while OGE simulations run at 400 m. Since it is very difficult to run a whole simulation in only NGE, we will not study that ground effect.

Refer to Table 6 for values of $t_0$, which are simulation-specific and required in order to convert the weather to the model’s length- and time units, as explained in Sections 2.1 and 4.3.
CHAPTER 6. ANALYSIS OF THE PARAMETER SENSITIVITY

Simulation Setup | \( t_0 \)
-----------------|-------------------
IGE, Dubai       | 18.457            
OGE, Dubai       | 17.797            
IGE, Stockholm   | 18.457            
OGE, Stockholm   | 17.797            

Table 6.1. Values of \( t_0 \) for each simulation setup

While plots can be useful in explaining ideas it is impossible to cover all cases that way. Hence, we will also need to examine the ODE directly to draw more general conclusions. In the analysis, we have chosen to respect the implementation and consider the stratification as two separate functions: \( N(x, y, z) \) and \( N_{\text{circulation}} \).

Furthermore, it will be hard to say anything concrete about the ODE behavior unless we make assumptions, e.g. on the weather functions. For instance, a useful assumption is that they only depend on the altitude and not on \( y \) or \( x \). This assumption is plausible seeing as the vortices move at most a couple of kilometers during strong wind.

Additionally, we would like to point out that even the altitude-dependence should be very weak. That is, in typical scenario, wake-vortices descend only approximately 200 m. From the physical meaning of the weather we know that such small altitude differences do not cause drastic changes in wind direction nor strength.

### 6.1. Sensitivity in Out of Ground Effect

As was stated above, we need to assume that the weather functions only vary as a function of \( z \), i.e.,

\[
W(x, y, z) = \hat{W}(z) \quad (6.3)
\]

\[
\epsilon(x, y, z) = \hat{\epsilon}(z) \quad (6.4)
\]

\[
N(x, y, z) = \hat{N}(z) \quad (6.5)
\]

Otherwise, the complexity of the equations prohibits us from drawing any qualitative conclusions with regards to the variables under study.

Nevertheless, when that assumption holds, Equations (3.35)-(3.36) become:

\[
\dot{y}_i(z_i) = \hat{W}(z_i) \cdot e_y \quad (6.6)
\]

\[
\dot{z}_i(z_i, t) = B_i(t, z_i, \hat{N}(z_i)) \left[ 1 - \exp\left(-\frac{(0.4b_0)^2 1.257}{r_c(\Gamma(t, \epsilon, N_{\text{circulation}}))}\right) \right]. \quad (6.7)
\]

Figure 6.1 depicts the dependency graph resulting from Equations (6.6)-(6.7). Compare it to the original OGE dependency graph in Figure 3.3.

Consulting Figure 6.1 and Equations (6.6)-(6.7) we can draw the conclusion that \( z_i \) do not depend on \( W \) at all. On the other hand, \( y_i \) has a direct dependence on \( W \).
6.2. NEAR/IN GROUND EFFECT SENSITIVITY

As a matter of fact, we see in Figures 6.3 and 6.5 that the dependence is essentially linear. That is, the difference between two solutions \((y_1^I, z_1^I), (y_1^II, z_1^II)\) with slightly different winds \(W_1^I, W_1^II\) is:

\[
\Delta y_i(t) = y_i^I(t) - y_i^II(t) = \int_0^t (\dot{W}_i^I(z_1^I(\tau)) - \dot{W}_i^II(z_2^I(\tau))) \cdot \mathbf{e}_y d\tau \approx tC \quad (6.8)
\]

where \(C = \dot{W}_i^I(t = 0) - \dot{W}_i^II(t = 0)\). The approximation in the last step is due to the presumed weak altitude-dependence of \(W\) discussed earlier.

Furthermore, notice in Figure 6.1 that \(y_i\) depends indirectly on \(z_i\). This means that \(y_i\) should, in theory, be affected by perturbations in \(\epsilon\) and \(N\) (since \(z_i\) is). However, as Figures 6.2 and 6.4 show, this dependence is so weak in practice that it can be neglected.

So, in general we can say that wind parameters affect only \(y\) whereas circulation parameters affect \(z\).

Figure 6.11 illustrates an OGE-trajectory and how different perturbations affect it. Compare with Figures 6.3 and 6.5.

6.2. Near/In Ground Effect Sensitivity

In NGE and IGE, using the same assumptions as in the OGE-analysis in the previous section, i.e., that Equations (6.3)-(6.5) hold, the ODE Equations (3.29)-
Figure 6.2. Circulation-variations in Dubai, OGE. The perturbations $\Delta y$ and $\Delta z$ were defined in the beginning of this chapter. Considering the relative magnitude of the perturbations, only $\Delta z$ is affected. The changes in $\Delta y$ are 1000 times as small.
6.2. NEAR/IN GROUND EFFECT SENSITIVITY

Figure 6.3. Wind-variations in Dubai, OGE. The perturbations $\Delta y$ and $\Delta z$ were defined in the beginning of this chapter. It is clear that wind only affects $\Delta y$, relatively speaking. Note also that the disturbance grows linearly over time.
Figure 6.4. Circulation-variations in Stockholm, OGE. The perturbations $\Delta y$ and $\Delta z$ were defined in the beginning of this chapter. Again, only $\Delta z$ is affected by the variations, relatively speaking.
6.2. NEAR/IN GROUND EFFECT SENSITIVITY

Figure 6.5. Wind-variations in Stockholm, OGE. The perturbations Δy and Δz were defined in the beginning of this chapter. Relatively speaking, only Δy is affected. Note that the disturbance grows linearly over time.
(3.30) for the main vortex coordinates can be simplified slightly:

\[
\frac{dy_i}{dt} = \mathbf{v}_i(y_1, y_2, \ldots, y_6, z_1, z_2, \ldots, z_i, \epsilon, N_{\text{circulation}}) + \mathbf{W}(z_i) \cdot \mathbf{e}_y \tag{6.9}
\]

\[
\frac{dz_i}{dt} = \mathbf{v}_i(y_1, y_2, \ldots, y_6, z_1, z_2, \ldots, z_i, \epsilon, N_{\text{circulation}}) \cdot \mathbf{B}(z_i, \tilde{N}(z_i)) \mathbf{e}_z. \tag{6.10}
\]

where \(\mathbf{v}_i = \mathbf{v}_i^{\text{BS}}\) from Section 3.1.6. The dependency graph is illustrated in Figure 6.6. As is evident from the graph, strong cross-wind renders \(z_i\) independent of \(W\). Moreover, \(y_i\) displays the same sort of dependency on \(W\) and on \(z_i\) as we saw in Section 6.1 so we should expect a similar behavior in the plots. Figures 6.8 and 6.10 confirm this.

Unfortunately, the dependencies are too complex for drawing any further conclusions without resorting to the numerical results.
6.2. NEAR/IN GROUND EFFECT SENSITIVITY

The effect of different parameter variations on an example IGE-trajectory can be seen in Figure 6.12. Notice that Figure 6.12 displays a weak oscillatory behavior in $z_1(t)$. This oscillatory behavior is characteristic of IGE and can be seen in all plots from that ground effect. The same oscillations can be seen in e.g Figures 6.9 and 6.10 where we only display the perturbations.
Figure 6.8. Wind-variations in Dubai, IGE. The perturbations $\Delta y$ and $\Delta z$ were defined in the beginning of this chapter. Relatively speaking, only $\Delta y$ is affected and its perturbation grows linearly over time. The oscillations we see in the $\Delta z$-plots are briefly discussed in Section 6.2.
6.2. NEAR/IN GROUND EFFECT SENSITIVITY

Figure 6.9. Circulation-variations in Stockholm, IGE. The perturbations $\Delta y$ and $\Delta z$ were defined in the beginning of this chapter. Both $\Delta y$ and $\Delta z$ are affected by the variations. The oscillations that can be seen in the plots are briefly discussed in Section 6.2.
Figure 6.10. Wind-variations in Stockholm, IGE. The perturbations $\Delta y$ and $\Delta z$ were defined in the beginning of this chapter. Relatively speaking, only $\Delta y$ is affected and the perturbation grows linearly over time. The oscillations that can be seen in the $\Delta z$-plots are briefly discussed in Section 6.2.
6.2. NEAR/IN GROUND EFFECT SENSITIVITY

Figure 6.11. Plots of $y_1, z_1$ versus time for all types of parameter variations, Dubai, OGE.
Figure 6.12. Plots of $y_1, z_1$ versus time for all types of parameter variations, Stockholm, IGE.
6.3. Tuning the Variations

Refer to Figures 6.12 and 6.11 which illustrate the effects the perturbations have on both $y_1$ and $z_1$. Notice that many of the perturbations do not visibly impact the outcome of a simulation. Roughly speaking, only the two wind parameters appear to affect it. The other perturbations have no effect because their relative impact on the outcome is minimal.

This suggests that the magnitude of the perturbations needs to be tuned before performing the Monte Carlo-simulations. That is, presently only two parameters ($|W|, W/|W|$) visually affect the outcome, so there is no point in including $N$ and $\epsilon$ in a simulation since that would just extend computations without any noticeable effect on the result.

Table 6.2 displays the approximate impact that each parameter has on the outcome. By 'impact', we mean the maximum perturbation incurred by the variation of that parameter. For best results, the parameter variations should be tuned prior to running a Monte Carlo-simulation so that each parameter’s impact on the outcome is the same.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Approximate impact (in length units)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N$</td>
<td>0.1</td>
</tr>
<tr>
<td>$\epsilon$</td>
<td>0.02</td>
</tr>
<tr>
<td>$</td>
<td>W</td>
</tr>
<tr>
<td>$W/</td>
<td>W</td>
</tr>
</tbody>
</table>

Table 6.2. Approximate impacts caused by varying each parameter, as explained in Section 6.3. Used to scale parameter variations in Section 6.4.

6.4. A Sample Monte Carlo-Simulation

We would like to conclude this chapter by giving an example of a Monte Carlo-simulation in Dubai, IGE. We display only 50 pairs of Monte Carlo-simulated trajectories because the purpose is visualization, and adding more would just clutter up the plot. Each IVP, corresponding to a pair of trajectories, i.e., from the left and right main vortices, uses input parameters perturbed using a gaussian distribution with mean 0. That is, let $p^{\text{unperturbed}}$ be the unperturbed value of a parameter $p$ then

$$p^{\text{perturbed}} = p^{\text{unperturbed}} + N(0, \sigma_p).$$

(6.11)

is the perturbed value. The $\sigma_p$ are given by the formula

$$\sigma_p = 0.02/x_p$$

(6.12)

where $x_p$ is the 'approximate impact'-value corresponding to $p$, in Table 6.2. In this way, all the varied parameters will have some bearing on the outcome. However, one drawback of scaling down e.g the wind-variations like this is that their visual
impact will be diminished. It will be hard to distinguish the individual trajectories in a visualization of a whole vortex-trajectory. For this reason, we first display the initial (unperturbed) vortex trajectories in Figure 6.13 before zooming in on the latter part of the vortex lifetime in Figure 6.14, when the perturbations have had some time to grow larger.
6.4. A SAMPLE MONTE CARLO-SIMULATION

Figure 6.14. Plot of 50 Monte Carlo-simulated vortex trajectory-pairs in the time interval 270-300 seconds. The whole trajectories can be seen in Figure 6.13.
Chapter 7

Final Thoughts and Suggestions

The purpose of this thesis is to perform Monte Carlo-simulations on calculated wake-vortex trajectories. The long-term plan is to introduce this idea at airports to assist them in scheduling flight traffic. However, the concept will only be considered by air traffic management if it is deemed trustworthy. Therefore, we want to conclude this thesis by discussing the reliability and to suggest where in the setup there is still room for improvement.

Note that Rytter [1] already validated the P2Pm model against authentic flight data. We do not want to repeat his work. Instead we will draw upon the results of the analyses in the previous chapters, and on the ODE formulation, to make our conclusions.

7.1. Reliability

Assuming that the P2Pm-model accurately represents wake-vortex behavior, the results should be reliable during OGE. As discussed in Chapter 5, we cannot trust the numerical accuracy of neither IGE nor any simulation containing more than one ground effect. Since a whole simulation is rarely composed solely of NGE we will not be able to safely trust the NGE-results in general either.

However, as mentioned, this is only true provided that we trust the P2Pm to be an accurate model. There are in fact still areas where the trustworthiness of the model itself can be improved.

The obstacle that every empirically-derived model will eventually run into is that all of the credibility must come from agreement with real-world data. Nothing can be said about the trustworthiness of such a model when applied to scenarios where it has not yet been validated. In cases such as wake-vortex prediction this can be troublesome since accurate real-world vortex measurements are hard to come by.

In contrast, a model that is derived completely from physical principles can be reliable even in completely untested scenarios provided that one is aware of all of the simplifications made in the model.
Thus, given that the reliability of the P2Pm will be so crucial to its accept rate at airports it is imperative that an attempt is made to give the model a stronger foundation in theory. Three points in particular stand out where the model could greatly benefit from a stronger theoretical basis: the abrupt transitions, the buoyancy factors and the vortex strengths.

Abrupt Transitions

Real-world phenomena are usually represented by smooth functions. It would therefore be interesting to know why wake-vortex behavior is most accurately described using abrupt transitions between the different states. By ‘states’ we mean not only the ground effect-phases but also the number of different ‘states’ that can occur within the IGE-phase as the number of active secondary vortices goes from 1 to 4.

Furthermore, the motivation for the abrupt transitions is made especially interesting by the fact that they are the precise reason that all numerical solutions in IGE or during a phase-change lack accuracy, as discussed in Chapter 5.

The ground effects were discussed in Section 2.3.3.

Buoyancy Factors

The buoyancy concept, which was discussed in Section 2.2.3, is rather peripheral to the wake-vortex model as a whole. It is nevertheless one of the phenomena which the model incorporates and should therefore be clearly defined. Equation (2.20), which describes how to calculate the buoyancy factors, originated from Holzäpfel [2] but lacked both a proper definition of some of its variables (such as $\Delta z$) as well as a clear derivation from other physical phenomena. Without knowing the definition of $\Delta z$ that Holzäpfel intended it is impossible to know if the formula is used correctly, because there is no physical derivation to fall back on.

Considering that the buoyancy concept is a nonessential part of the wake-vortex model and that its definition is so unclear, it might be advisable to omit it altogether from the model until a time when there is a better understanding of it.

Vortex Strength/Circulation

Vortex strength is closely related to circulation, which was introduced by Holzäpfel [2]. Circulation is a central part of the wake-vortex model. Equation (2.19) for calculating the circulation, although fairly credible in itself, could benefit from some clarification e.g of how the constants $T_1, T_2, \nu_1$ and $\nu_2$ were arrived at.

It would also be interesting to understand the underlying physical motivation for why the strength of the secondary vortices is affected by $|W_y|$, i.e., the absolute cross-wind, as explained in Section 2.3.3.

Circulation was discussed in Sections 2.2.2.
7.2. FURTHER WORK

Final Note on Reliability

Again, the most important issue at present is the RK4-algorithm’s lack of accuracy during IGE and during ground effect-transitions. It is possible to deal with this issue numerically. That is, the RK4-algorithm would need to be replaced by a more sophisticated numerical algorithm capable of solving accurately even ODEs containing discontinuities. We nevertheless wish to stress yet again that a first approach should be to understand why the model needs to feature the discontinuities that are the cause of the problem. One might find that the abrupt transitions (see above) that generate these discontinuities can in fact be replaced by smooth transitions, in which case a numerical treatment of the problem is unnecessary.

7.2. Further Work

In the previous two sections we discussed various ways of making the model more reliable. As a conclusion to this thesis, we would also like to suggest ways in which the method we have presented can be further developed.

Starting with the model, there are three logical ‘next steps’ to take

- Let the wind have a $z$-component, essentially making the wind 3D instead of 2D. This would be a relatively small modification to the model but would for obvious reasons add more realism to it.

- Allow the gates to be tilted, i.e., not always aligned with the $z$-axis. This would probably require that the wake-vortex ODE be reworked. It would nevertheless be a useful refinement since it would allow more accurate treatment of e.g. vortices from an aircraft with a steep downward pitch.

- Allow the initial left- and right main vortex positions to have different altitudes. This would add more realism to wake-vortex predictions for rolling aircraft. The modification would require that the OGE-phase be restated to not use an explicit formula for the vortex descent speed.

As for the numerical treatment, there is really only one thing that matters: execution speed. Thus a logical next step is to move from GPU-computations to even more specialized computing units, e.g Field Programmable Gate Arrays (FPGAs).

Furthermore, in Section 3.2.6 we mentioned that the dependency graph of each individual ground effect put together is smaller than the dependency graph of the whole wake-vortex ODE. Thus, a final idea is to separate the treatment of each ground effect, which might increase performance. This approach is not attempted in the thesis, because the task of implementing such an algorithm efficiently on the GPU is quite complex. Nevertheless, it is something that could be done in the future.
Bibliography


