Feature Extraction and Classification of Skiing/Snowboarding Jumps with an Integrated Head-mounted Sensor

by

Tien Jung (Matthew) Lee

B.A.Sc., Simon Fraser University, 2012

Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of
Master of Applied Science

in the
School of Mechatronic Systems Engineering
Faculty of Applied Sciences

SIMON FRASER UNIVERSITY

Summer 2015

Approval

Name:	**Tien Jung (Matthew) Lee**
Degree:	**Master of Applied Science**
Title:	***Feature Extraction and Classification of Skiing/Snowboarding Jumps with an Integrated Head-mounted Sensor***
Examining Committee:	**Chair:** Krishna Vijayaraghavan Assistant Professor

Edward J. Park
Senior Supervisor
Professor

Greg Mori
Supervisor
Associate Professor
School of Computing Science

Flavio Firmani
Internal Examiner
Lecturer

Date Defended/Approved: Aug. 24, 2015

Ethics Statement

The author, whose name appears on the title page of this work, has obtained, for the research described in this work, either:

a. human research ethics approval from the Simon Fraser University Office of Research Ethics,

or

b. advance approval of the animal care protocol from the University Animal Care Committee of Simon Fraser University;

or has conducted the research

c. as a co-investigator, collaborator or research assistant in a research project approved in advance,

or

d. as a member of a course approved in advance for minimal risk human research, by the Office of Research Ethics.

A copy of the approval letter has been filed at the Theses Office of the University Library at the time of submission of this thesis or project.

The original application for approval and letter of approval are filed with the relevant offices. Inquiries may be directed to those authorities.

Simon Fraser University Library
Burnaby, British Columbia, Canada

update Spring 2010

Abstract

With advancing technology in miniature MEMS sensors, wearable devices are becoming increasingly popular, facilitating convenient activity detection. One particular application is in sports performance monitoring. This thesis presents novel real-time jump detection and classification algorithms in skiing and snowboarding using a head-mounted MEMS-based inertial measurement unit (MEMS-IMU), which is integrated with a barometric pressure sensor. The key performance variables of the jump are extracted and evaluated for training and/or entertainment purposes. In contrast to the existing jump detection algorithms based on acceleration signals, the proposed algorithm uses vertical velocity and air time in addition to acceleration in the vertical direction. A support vector machine (SVM) is applied to generate a classification model. The jumps are classified into four different groups – Ollie, Standard, Drops, and Step up jumps. The experimental results show that by incorporating the velocity and air time into the detection algorithm, the sensitivity and specificity increase dramatically to 92% and 93%, respectively. In addition, the proposed classification model achieved 80.5% accuracy.

Keywords: Real-time skiing/snowboarding jump classification; wearable technology; MEMS sensors; sensor fusion; barometric pressure sensor

to my loved ones

Acknowledgements

Attaining a MASc degree is a great experience and accomplishment in my life journey. Most of all, I want to thank my professor, Dr. Edward Park, for giving me this opportunity to participate in this project and support me throughout my learning process. He has been a great mentor to me since 2009 when I first started my co-op with him. He has always supported and assisted me when I have questions and problems. In addition, I am really privileged to have worked with Reynald Hoskinson, Shaghayegh Zihajehzadeh, Darrell Loh, and Paul Yoon. They have helped me with many useful feedback and suggestions. Special thanks to Shaghayegh and Darrell, with whom we have been through all the ups and downs when we went out to the field to collect the data. They have been willingly supporting and assisting me in every single field test we went. I also want to thank my parents, my family, and my friends who faithfully supported me though prayers and encouraged me from time to time to keep me moving forward. It has been a great journey and I am very honoured to be blessed with many important people through this journey. Without them, I wouldn't have gone this far. Thank you all for everything you have done for me.

Table of Contents

List of Tables

List of Figures

List of Acronyms

ENU	East, North and Up
FMC	Fusion Motion Capture
FN	False Negative
FP	False Positive
FPF	False Positive Fraction
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
IMU	Inertial Measurement Unit
KPV	Key Performance Variable
MEMS	Microelectromechnical Systems
PDF	Probability Distribution Function
PFAD	Preceding and Following Acceleration Difference
ROC curve	Receiver Operating Characteristic curve
SVM	Support Vector Machine
TN	True Negative
TP	True Positive
TPF	True Positive Fraction
WMCM	Windowed Mean Canceled Multiplication

Nomenclature

${}^{S}\mathbf{a}_{\varepsilon}$	External acceleration error in sensor frame.
a_i	Lagrange multipliers.
h_{offset}	The difference in height measured by the GPS and that measured by the barometric pressure sensor.
${}^{S}\mathbf{a}$	Acceleration measurement from 3-axis accelerometer.
ξ_i	Slack variable of SVM model.
σ^2_{baro}	Variance of barometric pressure sensor noise.
$\mathbf{A}_i$	State transition matrix of the Kalman filter. i can be any integer to differentiate this matrix from different Kalman filters.
$\mathbf{B}_i$	A second state transition matrix of the Kalman filter. i can be any integer to differentiate this matrix from different Kalman filters.
$\mathbf{C}_i$	Observation matrix of the Kalman filter. i can be any integer to differentiate this matrix from different Kalman filters.
$\mathbf{I}_n$	n x n identity matrix.
$\mathbf{Q}_i$	Process noise covariance matrix of the Kalman filter. i can be any integer to differentiate this matrix from different Kalman filters.
$\mathbf{R}_i$	Measurement noise covariance matrix of the Kalman filter. i can be any integer to differentiate this matrix from different Kalman filters.
$\mathbf{n}_A$	Accelerometer measurement noise.
$\mathbf{n}_G$	Gyroscope measurement noise.
$\mathbf{n}_M$	Magnetometer measurement noise.
$\mathbf{y}_G$	Angular velocity measurement from 3-axis gyroscope
P_0	Standard atmospheric pressure equal to 101,325 Pa
P_{baro}	Pressure value measured by the barometric pressure sensor.
a_{th_1}	First acceleration threshold for jump detection algorithm.
a_{th_2}	Second acceleration threshold for jump detection algorithm.
t_{th}	Air time threshold for jump detection algorithm.
v_{th}	Velocity threshold for jump detection algorithm.
$\boldsymbol{a}_z$	Acceleration on z-axis in the navigation frame (vertical acceleration).

$\boldsymbol{j}_z$	The rate of change in acceleration on z-axis in the navigation frame (vertical jerk).
$\boldsymbol{p}_z$	Position on z-axis in the navigation frame (vertical position).
$\boldsymbol{v}_z$	Velocity on z-axis in the navigation frame (vertical velocity).
ε_{acc}	The constraint on the acceleration at a given time if the subject is at rest.
ε_{jerk}	The constraint on the change in acceleration between two data update if the subject is at rest.
σ_A^2	Variance of accelerometer measurement noise.
$\mathbf{K}$	Kalman filter gain.
$\mathbf{P}$	Process error covariance matrix of the Kalman filter.
$\mathbf{v}$	Measurement noise vector of the Kalman filter
$\mathbf{w}$	Process noise vector of the Kalman filter
$\boldsymbol{w}$	Weight vector of the features for the classification model.
Δh_{baro}	Change in the vertical height measured by the barometric pressure sensor.
α	Angle of the sensor measured about z-axis in navigation frame (yaw).
β	Angle of the sensor measured about y-axis in navigation frame (pitch).
γ	Angle of the sensor measured about x-axis in navigation frame (roll).
ϕ	Features space of the classification model.

Chapter 1.

Introduction

1.1. Overview

Skiing and snowboarding are two of the most popular winter sports. Ski jumping is one of the earliest competitive winter sports, being part of the Winter Olympics since it began in 1924 [1]. In 1998, snowboarding, which combines the element of surfing, skateboarding, and skiing, made its first debut in the Winter Olympics [2]. While competitions are expanding in scope and depth, amateur skiers and snowboarders around the world continue to stretch their abilities, striving for greater achievement in skill, technique, and difficulty of stunts and tricks. One way to determine athlete performance is to define key performance variables (KPVs) for sports. In skiing and snowboarding, the basic KPVs used to describe jumps are: 1) air-time, which is the duration of the jump from take-off to landing; 2) horizontal distance achieved from take-off to landing; 3) maximum vertical height achieved during the jump with respect to the ground level where the jump begins, and 4) the total rotation [3] achieved during the jump. Calculation of these KPVs can be used to provide feedback to these individuals and motivate them to improve their performance [4].

With the latest advances in wearable sensor technology, measuring these KPVs in portable and automatic fashion is becoming possible. With the advent of the microelectromechanical systems (MEMS) sensor technology, the sensors have become smaller in size and lighter in weight. In the wearable technology, the miniature MEMS sensors are used to measure kinematics of body segments that they are mounted to. The KPVs can then be extracted from these kinematic information. This project utilizes an integrated MEMS sensors to capture the jump KPVs of the skier or snowboarder in real-time, which then can be used as features to classify these jumps.

1.2. Literature survey

Motion capture technology has been extensively used in sports performance analysis. Over the course of time, athletes are pushing the envelope in their skills and abilities, striving to go beyond the limitation of gravity for greater achievement in more difficult jump tricks. As a result, camera-based image processing has been introduced to aid in analysis of the athletes' performance [5]-[7]. Camera-based optical motion tracking systems are widely used due to their measurement accuracy. However, they present the limitation of requiring an external source (i.e. cameras) and thus work best in controlled/confined areas (e.g., indoor settings). Hence, they are not suitable for outdoor motion tracking applications in active sports such as skiing, snowboarding, and biking that take place over long distances. On the other hand, an inertial measurement unit (IMU) is self-contained, and thus capable of providing unconstrained accessibility to kinematic motion information in outdoor/on-field settings [8]-[10]. Recently, researchers are exploring the potential in performance analysis with wearable sensors [3],[11]-[14]. With the advances in MEMS technology, wearable sensors have become smaller in size and more precise in accuracy. These sensors have the advantage of real-time feedback, minimal setup time, and minimal space limitation [15]. A MEMS-IMU-based motion tracking system can be worn by the athlete for the purpose of providing direct information feedback to them with minimal time delay in both recreation and professional training [9]-[10],[16].

Motion capture technology is useful to athletes for training purposes [15],[17] and to judges for evaluation [16]. Previous research applying IMU-based systems for winter sports applications include Harding et al. [8], which utilizes a torso-mounted IMU consisting of a tri-axial accelerometer and tri-axial gyroscope for jump detection applications in half-pipe competitions. A threshold-based three-state filter is applied on the acceleration signal to determine whether or not the athlete is in the air. Another threshold-based filter is then applied to the air time to validate the jumps. With IMU sensors attached to the lower back of the athletes, the air time and degree of rotation could be calculated. This information can be used by judges in scoring and ranking of the athlete performance in half pipe competition [16]. In Brodie et al. [15], the authors applied fusion motion capture (FMC) to capture 3D motion of alpine ski racing through a

10-gate slalom course. In the study, 13 IMUs were attached to the athlete's head, torso, pelvis, upper and lower arms, thighs, shanks, and ski boots. The data from IMU, GPS, and pressure sensitive insoles were combined to improve the position and orientation accuracy. The computed results were presented in animation and validated with video images for the purpose of improving athlete performance in the future. In Sadi and Klukas [18], a head-mounted IMU, consisting of tri-axial accelerometer, tri-axial gyroscope, and tri-axial magnetometer is used for jump detection in general skiing and/or snowboarding maneuver. The authors applied two methods, Windowed Mean Canceled Multiplication (WMCM) and Preceding and Following Acceleration Difference (PFAD), on the accelerometer data. This algorithm claims 92% sensitivity and 8% false detection rate. Our work presents novel algorithms for detecting and classifying snowboarding and ski jumps with a head-mounted IMU system. In addition to the typical acceleration signals, a vertical velocity signal is introduced to aid in jump detection. Kalman filter-based sensor fusion on accelerometer, gyroscope, magnetometer, and barometer pressure sensor is used to obtain the vertical velocity to improve the detection rate.

Commercially, interactive ski goggles have emerged as notable wearable devices for the alpine sports market. For example, Recon Instruments Inc., a local wearable devices company, has developed sport action goggles, Snow2, in which a 9-axis IMU, barometric pressure sensor, and GPS are embedded [19]. With its onboard sensors, Snow2 allows the users to monitor their speeds, vertical drop during a jump, distance travelled, and the altitude achieved, in real-time as well as share this information with friends. Also, GPS along with the onboard resort map allows the users to track where they are and set up a meet up location with a group of friends. Another group of entrepreneurs has recently developed augmented reality goggles, RideOn [20]. RideOn allows the users to communicate with each other through text messages and to track friends' location through GPS. A HD camera is also integrated in the goggles to record different activities the user does. In addition, RideOn creates a virtual environment for games and challenges, such as a virtual slalom or chasing after an image of your friend or a famous skier, that adds an extra layer of fun in the mountain. Still another start-up created a project called GogglePal [21]. In addition to the above mentioned features, GogglePal helps the individuals to get fit and stay healthy by

providing them their personal fitness stats, such as calories burned, elapsed time, distance traveled, etc. These devices not only enrich the users' alpine experience, but also provide the users real-time feedback for entertainment purposes.

1.3. Objective

The overall objective of this research is to develop accurate jump detection and classification algorithms with an integrated head-mounted sensor, consisting of 9 axes IMU, barometric pressure sensor, and GPS, which is embedded in a ski goggle. Key technical considerations involved in the research are: a) sensor fusion, b) online processing, c) algorithm efficiency, d) online learning, e) sensor placement, f) noise, and g) sensor accuracy.

Sensor Fusion

Sensor fusion is the primary technique used in this project. Several sensors, such as 3-axis accelerometer, 3-axis gyroscope, 3-axis magnetometer, barometric pressure sensor, and GPS, are involved and each sensor has different update rate and accuracy. How the various sensor data are combined to complement each other for the purpose of jump detection and classification is the fundamental problem to solve.

Online Processing

In addition, the algorithm should be able to process the data online in order to provide the instant feedback to the individuals. The data collected by IMU, barometer, and GPS, will be processed online and the KPVs will be extracted and displayed on a head-up display to provide the instant information to the users. Also, the processor has to be powerful enough to process the data.

Algorithm Efficiency

This also leads to better algorithm efficiency due to limited energy resource. Since the device is portable and is worn by the users, the only power resource is through the battery and the size of the battery should be light and small. Efficient algorithm is desired so the device will last longer when the individuals are using it.

Online Learning

Lastly, the algorithm should support online learning. Since each individual performs jumps differently. As a result, the detection and classification parameters should be tuned to fit each individual accordingly. Therefore, online learning is desired to tune the device according to the individuals.

Sensor Placement

Besides the considerations on the algorithm, other considerations are related to the hardware. The sensor is placed and fixed on top of the helmet in order to simulate the effect of the sensor embedded in commercial sport action goggles, such as Recon's Snow2. The advantage of embedding the sensor in the goggles is that the individuals do not have to wear extra equipment. In addition, the head-up displace can provide the feedback to the users instantly.

Noise

However, because the sensor is mounted on the head, it captures the motion of the head instead of the main body. Some head movements create dynamic noises that will affect the detection algorithm. The magnetometer is also susceptible to magnetic disturbance. These noises may cause false detection and affect the accuracy of the algorithm dramatically.

Sensor Accuracy

Also, sensor bias will affect the algorithm performance if not dealt correctly. Calibration should be done precisely when the sensor is used. The accuracy of the sensor should also be taken into consideration. If the range of motion is greater than the capacity of the sensor, the KPVs will not be accurate either.

1.4. Contributions

The main contributions of this project are (i) a real-time jump detection algorithm, and (ii) a real-time jump classification algorithm, which can be implemented into the

sports action goggles with embedded MEMS sensors, for skiing and snowboarding applications. The algorithms utilize sensor fusion to extract useful information, such as orientation of the athlete, the vertical velocity the athlete travels, the air time the athlete achieves, and the vertical position information of the athlete. These information can potentially be used for both recreational and professional training purposes, as well as for competition scoring purpose.

This thesis provides novel solutions to improve the accuracy of jump detection and classification. In the conventional method, acceleration is the main source for detecting the jumps in skiing/snowboarding. In this thesis, two additional parameters, change in vertical velocity and air time, are introduced for the first time to improve the detection accuracy. Also, for jump classification, characteristic features, such as height, drop, and cumulative yaw rotation, are also included in the feature set in addition to typical time-domain features to increase the classification accuracy.

1.5. Thesis outline

This thesis is divided into the following chapters. In Chapter 2, a threshold-based jump detection algorithm on vertical acceleration, vertical velocity, and air time is proposed, which is presented in T.J. Lee [22]. After the jumps are detected, the classification algorithm is discussed in Chapter 3, which is the work presented in my second paper [23]. The multi-class SVM classifier is applied. Chapter 4 conclude his thesis and provides suggestions for future research.

Chapter 2.

Automatic Jump Detection

2.1. Introduction

The first stage of the proposed application is to detect the jump windows. A novel skiing and snowboarding threshold-based jump detection algorithm is proposed in this chapter. The remainder of the chapter is organized as follows. In Methodology, the theory behind the proposed jump detection algorithm is explained in detail. The experimental protocol and setup are described in the following section. After that, experimental results consisting of snowboarding jumps are presented and discussed. Finally, suggestions and future works are provided in the Conclusions section.

2.2. Methodology

The proposed algorithm for jump detection is presented in this section. The IMU consists of a tri-axial accelerometer, tri-axial gyroscope, and tri-axial magnetometer. The IMU is mounted on the helmet to collect the inertial/magnetic data. The head may not be the best place to mount the IMU as the measurement from the head-mounted IMU will get affected by head movements, which might not represent true motion of the athletes' center of mass [24]. However, it is more practical and convenient to the athletes in a sense that they do not need to put on extra equipment other than a helmet and “smart” goggles (e.g., Recon’s Snow2). The algorithm utilizes measurements from the IMU, and a MEMS barometric pressure sensor. These measurements are passed into the proposed jump detection algorithm illustrated in Figure 2.1. As shown in this figure, the first step in the jump detection is to estimate the vertical velocity and acceleration with the “orientation and position/velocity Kalman filter”. Then, the acceleration, velocity and

air time are validated with pre-defined threshold values. Finally, the jump detection is performed based on the results of these comparisons. The above-mentioned steps are explained below.

2.2.1. Orientation and Position/Velocity Kalman Filter

The orientation and velocity Kalman filter uses inertial and barometric pressure measurements in the sensor frame to estimate the vertical component of acceleration ($\boldsymbol{a}_z$) and velocity ($\boldsymbol{v}_z$) in the navigation frame. The structure of this filter is shown in Figure 2.2. The sensor frame is the coordinate system fixed to the sensor and the navigation frame is the global coordinate system pointing to the East, North and Up (ENU) directions. Because the sensor is rigidly mounted on the wearer's body, for simplicity, the body frame and the sensor frame are assigned in a way that they coincide with each other.

To find the vertical component of free acceleration, i.e. the gravity compensated acceleration in the navigation frame, the two orientation angles of the body frame with respect to the horizontal plane, i.e. roll and pitch angles, need to be estimated. As shown in Figure 2.2, a tilt Kalman filter based on our previous work in [25]-[26] is used for this purpose. This filter fuses the tri-axial accelerometer and gyroscope measurements to estimate the two tilt angles.

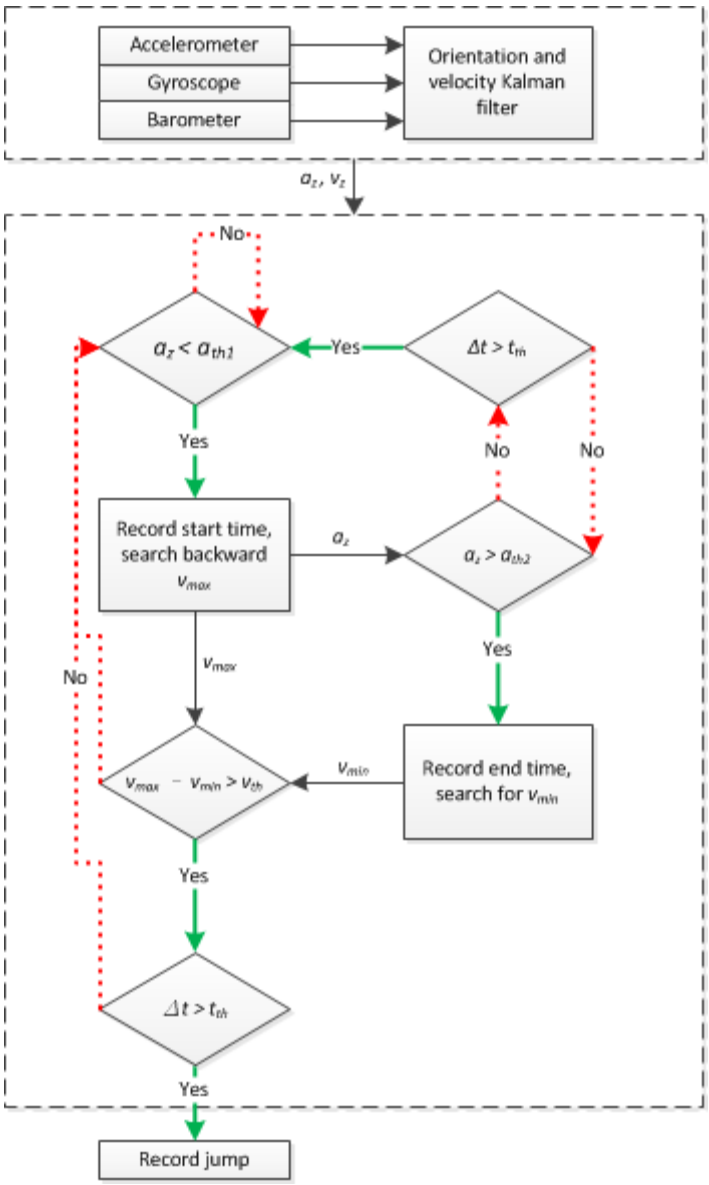

Figure 2.1. A flowchart of the proposed jump detection algorithm.

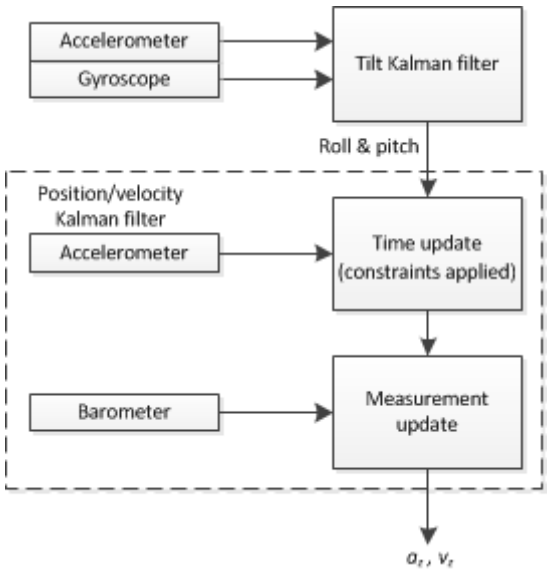

Figure 2.2. Structure of orientation and position/velocity Kalman filter.

As illustrated in Figure 2.2, the output of the tilt Kalman filter is passed to the position/velocity Kalman filter, which estimates the vertical position and velocity of the moving athlete in the navigation frame. This Kalman filter fuses the accelerometer measurements with barometric pressure data to overcome the otherwise drift-prone inertial navigation due to bias instability and noise in IMU measurements [27]. To further reduce the drift, kinematic constraints are applied. These constraints are set on acceleration, ε_{acc}, and on the change in acceleration of the subject, ε_{jerk}. Based on (2.1), when the acceleration and the change in acceleration are less than the pre-defined thresholds, the signal is assumed to be caused by noise [28]. As a result, a damping factor, $\alpha < 1$, is applied to the velocity update equation in the previous time step as the velocity decreases to zero. On the other hand, if the acceleration data was not due to noise, the velocity is updated with the Euler's method:

$$\boldsymbol{v}_z[k] = \begin{cases} \alpha \boldsymbol{v}_z[k-1] \\ \boldsymbol{v}_z[k-1] + \Delta t \boldsymbol{a}_z[k-1] \end{cases} \quad \begin{matrix} if\, |\boldsymbol{a}_z[k]| \leq \varepsilon_{acc} \cap |\boldsymbol{j}_z[k]| \leq \varepsilon_{jerk} \\ otherwise \end{matrix} \tag{2.1}$$

where $\boldsymbol{v}_z[k]$ is the velocity, $\boldsymbol{a}_z[k]$ is the acceleration, and $\boldsymbol{j}_z[k]$ is the jerk, which is the rate of change in acceleration, at time k, all in the vertical direction. Based on the Euler's method, the following model is used in the position/velocity Kalman filter:

$$\boldsymbol{p}_z[k] = \boldsymbol{p}_z[k-1] + \boldsymbol{v}_z[k-1]\Delta t + \frac{1}{2}\boldsymbol{a}_z[k-1]\Delta t^2 \tag{2.2}$$

$$\boldsymbol{v}_z[k] = \boldsymbol{v}_z[k-1] + \boldsymbol{a}_z[k-1]\Delta t \tag{2.3}$$

where $\boldsymbol{p}_z$ is the position. The above equations can be written in the state-space format as:

$$\boldsymbol{x}[k] = \mathbf{A}\boldsymbol{x}[k-1] + \mathbf{B}\boldsymbol{a}_z[k-1] + \mathbf{w}[k-1] \tag{2.4}$$

$$\boldsymbol{y}[k] = \mathbf{C}\boldsymbol{x}[k] + \mathbf{v}[k-1] \tag{2.5}$$

where $\boldsymbol{x} = \begin{bmatrix}\boldsymbol{p}_z \\ \boldsymbol{v}_z\end{bmatrix}$, $\mathbf{A} = \begin{bmatrix}1 & \Delta t \\ 0 & 1\end{bmatrix}$, $\mathbf{B} = \begin{bmatrix}\frac{\Delta t^2}{2} \\ \Delta t\end{bmatrix}$, $\boldsymbol{y}[k] = h_{baro}$ and $\mathbf{C} = [1 \quad 0]$. $\mathbf{w}$ and $\mathbf{v}$ are the process and measurement noise vectors respectively [26]. h_{baro} is the relative altitude calculated from barometric pressure data using [29]:

$$\Delta h_{baro} = 44330\left(1 - \left(\frac{P_{baro}}{P_o}\right)^{0.19}\right) + h_{offset} \tag{2.6}$$

where P_0 is the standard pressure equal to 101,325 Pa and P_{baro} is the measured pressure. h_{offset} is the bias in barometric altitude that can be adjusted by the altitude information from the Global Navigation Satellite System (GNSS) when absolute altitude tracking is of interest. Since GNSS tends to be less accurate in the vertical direction [30]-[31], in relatively stable weather conditions barometric altitude tracking is more accurate [26]. However, barometric pressure data is noisy, therefore a few seconds of averaging is necessary in order to achieve sufficient accuracy [32]. In this study, a moving average of 0.5 second square window is applied on the barometric pressure data to filter out the noise. The following Kalman filter equations are used for state estimation at each step [33]:

In the time update, the state is updated with

$$\boldsymbol{x}^-[k] = \mathbf{A}\boldsymbol{x}[k-1] + \mathbf{B}\boldsymbol{a}_z[k-1] \tag{2.7}$$

and the process error covariance ($\mathbf{P}$) is updated with

$$\mathbf{P}^{-}[k] = \mathbf{A}\mathbf{P}[k-1]\mathbf{A}^{T} + \mathbf{Q} \tag{2.8}$$

$$\mathbf{Q} = \begin{bmatrix} \frac{1}{2}\Delta t^2 \\ \Delta t \end{bmatrix} \sigma_A^2 \begin{bmatrix} \frac{1}{2}\Delta t^2 \\ \Delta t \end{bmatrix}^T \tag{2.9}$$

where $\mathbf{Q}$ is the process noise covariance matrix and σ_A^2 is the accelerometer noise variance in each axis. The measurement updates contain three equations: (i) Kalman gain update equation, (ii) a posteriori state update equation, and (iii) a posteriori measurement error covariance update equation [33]:

$$\mathbf{K} = \mathbf{P}\mathbf{C}^{T}(\mathbf{C}\mathbf{P}\mathbf{C}^{T} + \mathbf{R})^{-1} \tag{2.10}$$

$$\mathbf{R} = \sigma_{\text{baro}}^2 \mathbf{I}_3 \tag{2.11}$$

$$\boldsymbol{x} = \boldsymbol{x}^{-} + \mathbf{K}(h_{baro} - \boldsymbol{C}\boldsymbol{x}^{-}) \tag{2.12}$$

$$\mathbf{P} = \mathbf{A}\mathbf{P}^{-}\mathbf{A}^{T} + \mathbf{Q} - \mathbf{A}\mathbf{P}^{-}\mathbf{C}^{T}\mathbf{R}^{-1}\mathbf{C}\mathbf{P}^{-}\mathbf{A}^{T} \tag{2.13}$$

where $\mathbf{R}$ is the measurement noise covariance matrix, σ_{baro}^2 is the barometer noise variance, and $\mathbf{I}_3$ is the 3 x 3 identity matrix. The selected outputs of the orientation and position/velocity Kalman filter, which are the acceleration and velocity data on the vertical axis, are subsequently used to detect potential jump occurrences with a threshold-based algorithm.

2.2.2. Thresholds

Jumps are detected using a threshold-based method. Linthorne [34] plotted the relation between acceleration, velocity and vertical trajectory during a typical jump as shown in Figure 2.3. These curves are marked with cursors to show the corresponding points between each curve. As shown, before the jump, the acceleration increases to a threshold value (point d). After that, the acceleration drops to a negative constant value when the subject leaves the ground (point f) and returns to positive value when the subject lands (point h). Also, the velocity reaches its maximum just before the jump

starts (point e), and its minimum right after the jump ends (point h). Consequently, the following four thresholds were selected to verify a jump: (i) a_{th1} on acceleration at jump start, (ii) a_{th2} on acceleration at jump end, (iii) v_{th} on the difference between maximum and minimum velocity right before jump start and right after jump end, and iv) t_{th} on the duration of jump from start to finish.

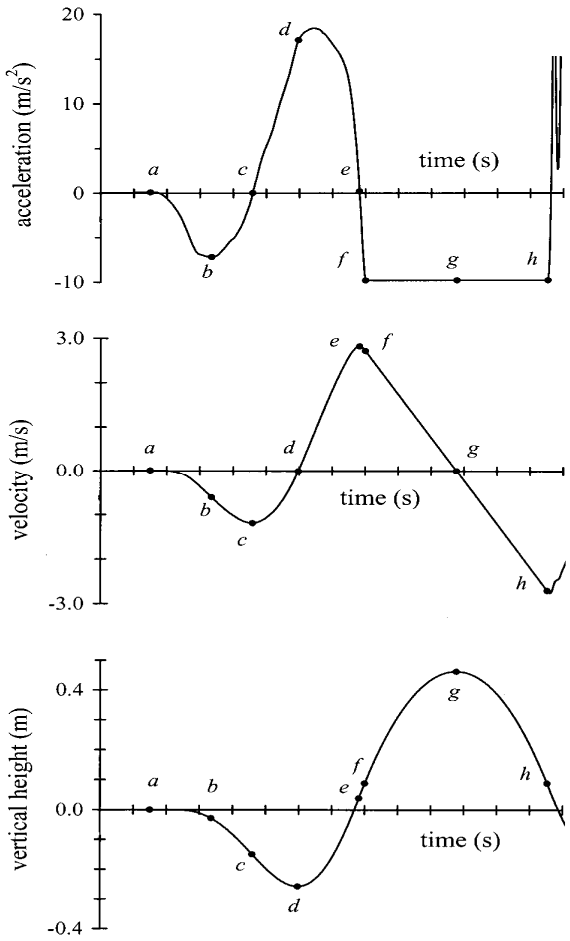

Figure 2.3. Dynamic and kinematic curves of a jump from force plate [34][1].

Figure 2.4 shows the vertical acceleration and velocity data of a typical jump measured with a head-mounted IMU sensor. Three of the thresholds used to detect a

[1] Reproduced with permission from Linthorne NP. Analysis of standing vertical jumps using a force platform. Am. J. Phys.2001; 69(11): 1198-1204. Copyright 2001, American Association of Physics Teachers.

jump are illustrated in the figure. Two thresholds were used on acceleration to detect the start and end of jumps. In addition to these thresholds applied on the acceleration, a threshold was defined as the difference in vertical velocities before the jump started and after the jump ended to verify whether or not a jump occurred. The acceleration values in the figure were normalized with respect to gravity. When the subject is standing on the ground, the accelerometer has a positive value in the vicinity of 1g in the vertical direction. This is caused by the ground reaction force exerted on the body in order to counter the gravity force. As the subject is about to jump, acceleration increases gradually due to the movement of subject's center of gravity. When the center of gravity moves downward, the legs act as springs and store this energy, ready to be exploded on the jumping instance. As the subject starts to move back up from the squat position, there js an increase in acceleration caused by the release of this energy. The acceleration reaches its peak when the energy is fully released. As the subject leaves the ground, the acceleration decreases to zero, because no external force is applied on the subject. After the subject returns to the ground, a peak is generated in the acceleration readout caused by the impact between the subject and ground. The acceleration, then, gradually returns to 1g contributed by the normal force exerted on the body from the ground.

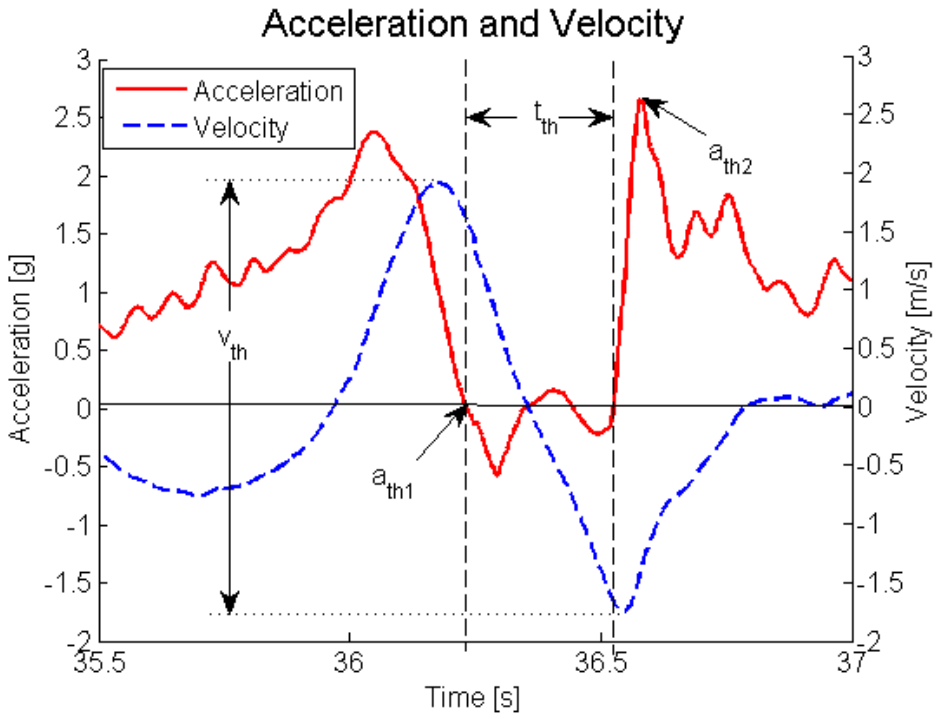

Figure 2.4. Vertical acceleration and velocity curves of a typical jump and the respective thresholds.

Knowing that, ideally, the acceleration will reach zero when the subject leaves the ground, the algorithm declares a potential jump whenever the acceleration drops to zero. Nevertheless, acceleration alone cannot determine a jump occurrence reliably. IMUs are susceptible to high dynamic noise, which could cause the acceleration to drop below zero from time to time. In addition, because the sensor is mounted on the head, any vertical head movement may cause the acceleration to drop to zero for a short time period independent of a jump. Therefore, vertical velocity estimation is utilized for jump occurrence validation. Figure 2.5 shows the acceleration and velocity when the subject squats. This is a falsely detected jump when acceleration signal alone is used to detect jumps. However, as can be seen in Figure 2.5, the difference between the maximum and minimum velocity achieved in the vertical velocity curve is less than 0.5 m/s, while in Figure 2.4, the velocity difference is close to 4 m/s. Consequently, a threshold on the velocity, v_{th}, is introduced for verification and to filter out falsely detected jumps.

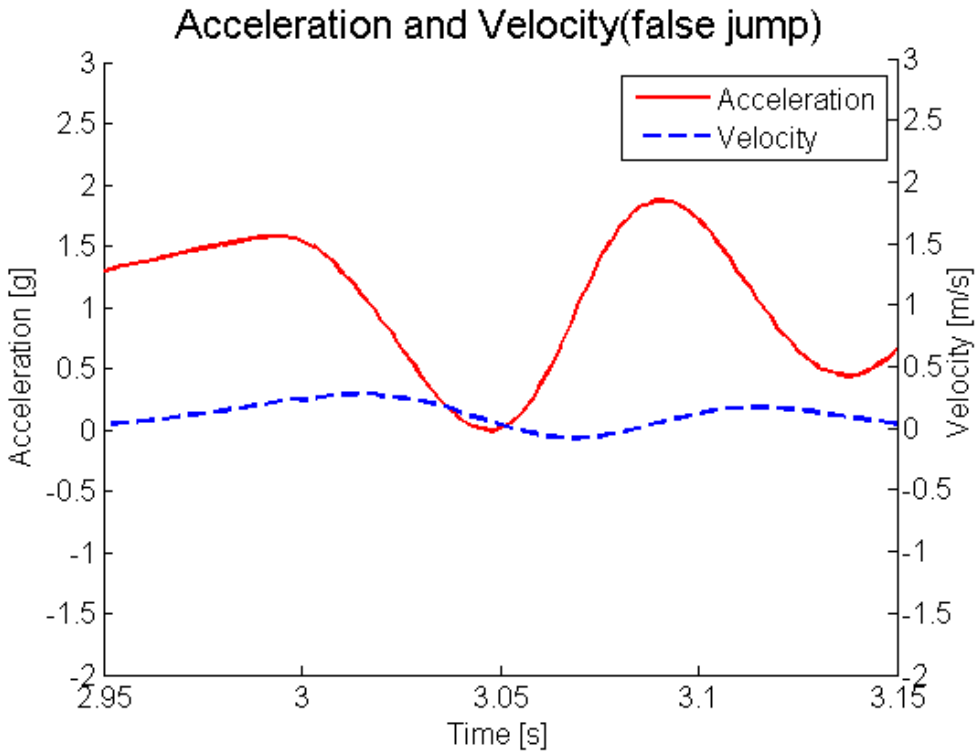

Figure 2.5. Acceleration and velocity of a falsely detected jump.

2.2.3. Detection

Data is passed to the algorithm in chronological order. The algorithm declares a potential jump start once the acceleration data falls below a_{th1}, and it continues monitoring the acceleration signal until the data crosses a_{th2}, when a jump end is detected. The maximum and minimum velocities achieved in the vicinity before the jump start and after the jump end are recorded accordingly. The difference in the vertical velocities is then compared to v_{th}, and the duration between the jump start and jump end is compared to t_{th}. Sometimes, the algorithm detects potential jumps when the acceleration signal drops below zero caused by head movement noise or vibration. Usually the duration of these false positives is much smaller than that of an actual jump. Therefore, the threshold on the air time (t_{th}) is applied on the signal to remove false positives caused by head movement noise. If both thresholds are satisfied, a jump occurrence is confirmed and the jump window is recorded for further processing (e.g., jump classification).

2.2.4. Performance Evaluation

In order to evaluate how well the algorithm performs, three evaluation parameters – performance, true positive fraction, and false positive fraction – are defined. It is important to understand the concept of true positive (TP), false positive (FP), true negative (TN) and false negative (FN) in relation to jump detection. True positives are defined as jumps correctly detected by the algorithm for actual jumps that were performed. A false positive, on the other hand, is defined as a jump falsely detected by the algorithm when there was no jump performed. A true negative means no jump was detected by the algorithm given that no jump was indeed performed, and false negative means no jump was detected by the algorithm when actually a jump was performed. The three performance evaluation parameters are then defined as follows:

$$\text{Performance} = \frac{\text{TP}}{\text{TP} + \text{FP} + \text{FN}} \tag{2.14}$$

$$\text{True Positive Fraction}(\text{Sensitivity}) = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2.15}$$

$$\text{False Positive Fraction}(1 - \text{Specificity}) = \frac{\text{FP}}{\text{TN} + \text{FP}} \tag{2.16}$$

True Positive Fraction (TPF), or sensitivity, is the measure of how well the jumps are detected with the proposed algorithm. It is defined as the ratio of the number of jumps detected to the total number of jumps performed. False Positive Fraction (FPF) is the measure of how well the algorithm distinguishes from other activities. It is a measure of falsely detected jumps out of all the activities other than jumps. Performance is used to determine the overall jump detection accuracy of the algorithm. Since other activities correctly identified are not of concern, TN is not included in the definition of Performance. On the other hand, the parameters that are directly related to the detection of jumps, i.e. TP, FP, and FN, are included in the equation.

2.3. Experimental Setup

2.3.1. Experimental Setup and Protocol

To verify the performance of the proposed algorithm, field tests were conducted on local ski hills (Cypress Mountain, British Columbia, Canada and Whistler, British Columbia, Canada). An Xsens MTi-G-700 was used to collect sensor data, consisting of an IMU sampling at 100Hz, a GPS sampling at 1Hz, and a barometric pressure sensor sampling at 20Hz. The Xsens MTi-G-700 was rigidly mounted on a helmet worn by the subject, approximating the placement of sensors in consumer wearable computing smart goggles for alpine sports with a built-in onboard processor with IMU sensor, GPS, and barometric pressure sensor (e.g., Recon's Snow2). Data was transmitted wirelessly to a laptop at the base station. A GoPro Hero3+ camera was also used to keep track of the subjects' activities for future reference. The sensors and GoPro camera were synchronized with GPS timestamps. The sensors output GPS timestamps when there was GPS reception. A device displaying the GPS timestamp was placed in front of the camera and was recorded for 2 seconds before each test. The subjects were asked to perform various types of jumps listed as follows in a predefined route in the terrain park.

a. Ollie: Jump in which the test subject springs off a fairly flat surface.

b. Standard Jump: Jump in which the test subject springs off a ramp.

c. 180^{o} Jump: Jump in which the test subject springs off a ramp with 180^{o} rotation about the vertical axis.

d. Drop: Jump in which the test subject slides off a ramp or small cliff.

The accelerometer noise variance σ_{A}^{2} is obtained from static measurements and is set to $10^{-4}\,\mathrm{m}^{2}/\mathrm{s}^{4}$. The barometer noise variance $\sigma_{\mathrm{baro}}^{2}$ is set to $1\,\mathrm{m}^{2}$.

2.3.2. Participants

Seven healthy male subjects aged between 19 to 28 participated in the field experiment. The participants were recruited through the University's internal posting and invitation on site. These participants have had several years of skiing or snowboarding

experience. Some of the participants have acquired their skiing or snowboarding instructor license while others were proficient in skiing or snowboarding and comfortable with jumping or doing tricks in terrain parks. They were informed of the experimental protocol (No. 2013s0790), which was approved by the Research Ethics Board of Simon Fraser University. The type and quantity of jumps each subject performed are tabulated in Table 2.1. Figure 2.6 is a picture one of the test subjects with head-mounted sensor on the helmet and Figure 2.7 shows an example of a jump performed by one of the test subjects.

Table 2.1. **Jumps performed by each subject**

	Ollie	**Standard**	**180°**	**Step Up**	**Drop**
Subject 1		5	5	4	4
Subject 2		6			2
Subject 3		5	6	5	5
Subject 4	7				
Subject 5		4	1		5
Subject 6	5	8			4
Subject 7	8		1		
TOTAL	**20**	**28**	**13**	**9**	**20**

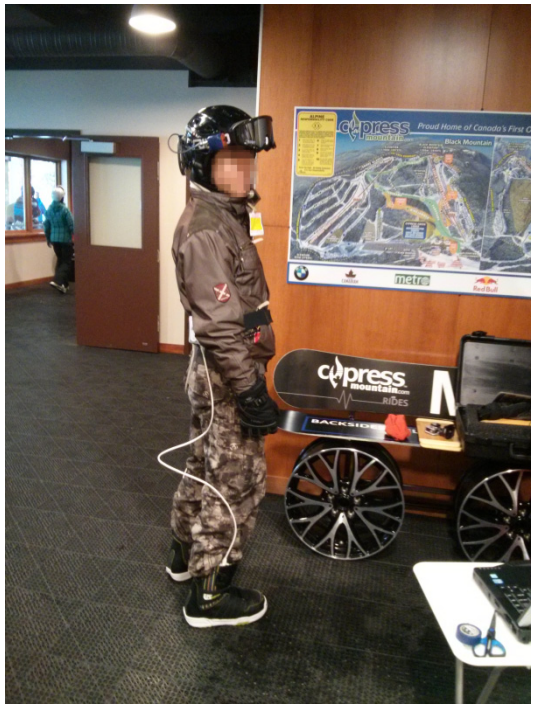

Figure 2.6. The experimental setup of the test subject with head-mounted IMU sensor.

Figure 2.7. An 180^{o} jump performed by the test subject.

2.4. Results and Discussion

The algorithm detected 242 potential jumps from 90 actual jumps when no thresholds were set on v_{th} and t_{th}. These potential jumps were manually inspected and

the peak acceleration at the end of jump, the difference between maximum and minimum velocities, and the air time were extracted. Optimal thresholds for these parameters were chosen by comparing the results of different combinations of threshold sets generated from the following two methods. The first method involved a grid search over selected values for each parameter. For each set of parameters, the true positive fraction and false positive fraction were calculated and then receiver operating characteristic (ROC) curves were plotted. A coarse grid search, in which the thresholds increase on the order of a power of 2, was carried out followed by a fine grid search over the values around the best set of parameters found in the coarse grid search. The ROC curves for the coarse grid and fine grid searches are shown in Figure 2.8 and Figure 2.9, respectively. Each point in the ROC curves represents a set of threshold values that generate the specific TPF and FPF values. In a ROC curve, the top left corner, which represents a 100% TPF and 0% FPF, is the desired value. Because it is physically impossible to achieve the desired point, Performance, defined in (2.14), is used to determine the best set of the thresholds. In the coarse grid search, the set of parameters that gave best Performance value were found to be 1.2g for the peak acceleration, 3 m/s for the velocity difference, and 0.1 second for the air time. After the fine grid search, these values were refined to be 1.1g, 3.1 m/s, and 0.12 seconds, respectively.

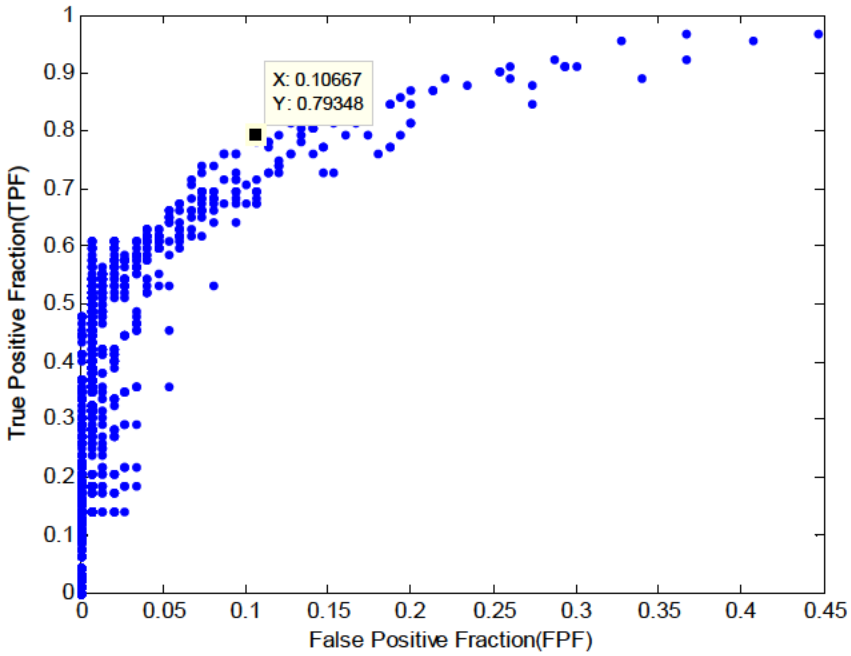

Figure 2.8. ROC curve of coarse grid search.

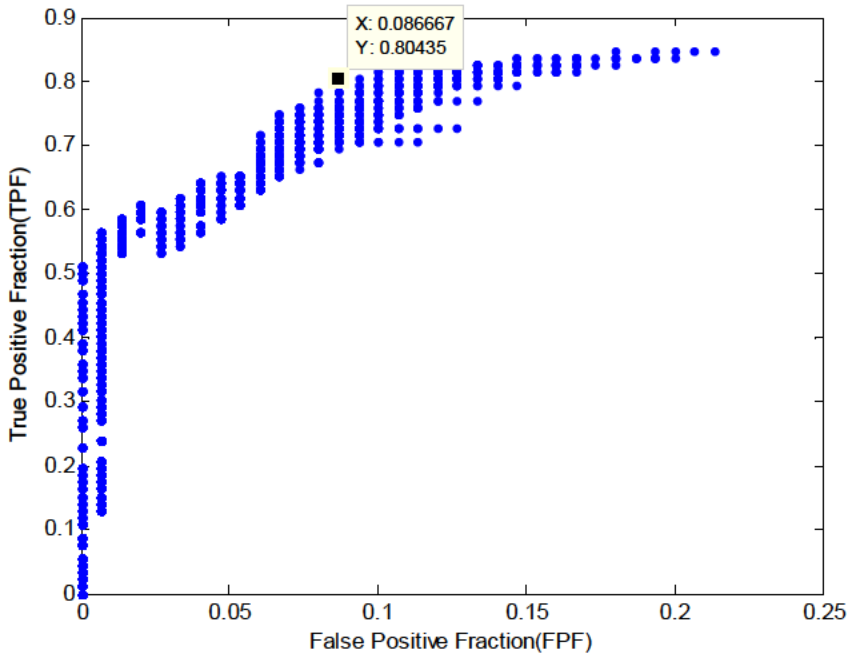

Figure 2.9. ROC curve of fine grid search.

Decision boundary was the second method used to set the thresholds for these parameters. The potential jumps were manually classified into two groups, *jump* and *false detection*, based on whether or not jumps occurred. Each parameter was assumed to have a normal probability distribution function (PDF). Figure 2.10, Figure 2.11, and Figure 2.12 show the probability distribution function of the peak acceleration, velocity differences, and air time, respectively. The red curves are probability distribution functions of the *jump* group and the blue curves are that of the *false detection* group. The optimal threshold values to separate the two groups are the intersecting points between these curves [35]. The values were found to be 1.25g, 4.29 m/s, and 0.30 second for the peak acceleration, velocities difference, and air time, correspondingly.

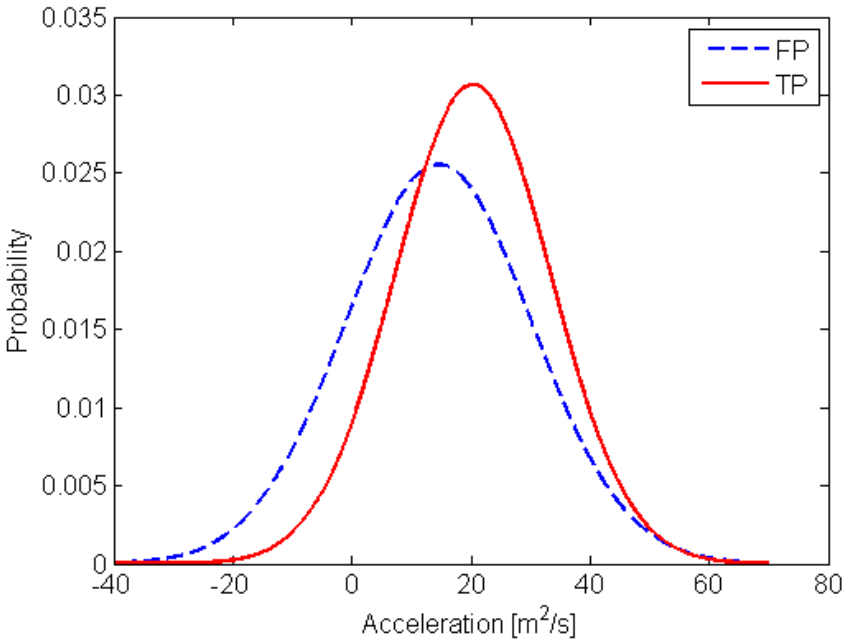

Figure 2.10. Probability distribution functions of peak acceleration.

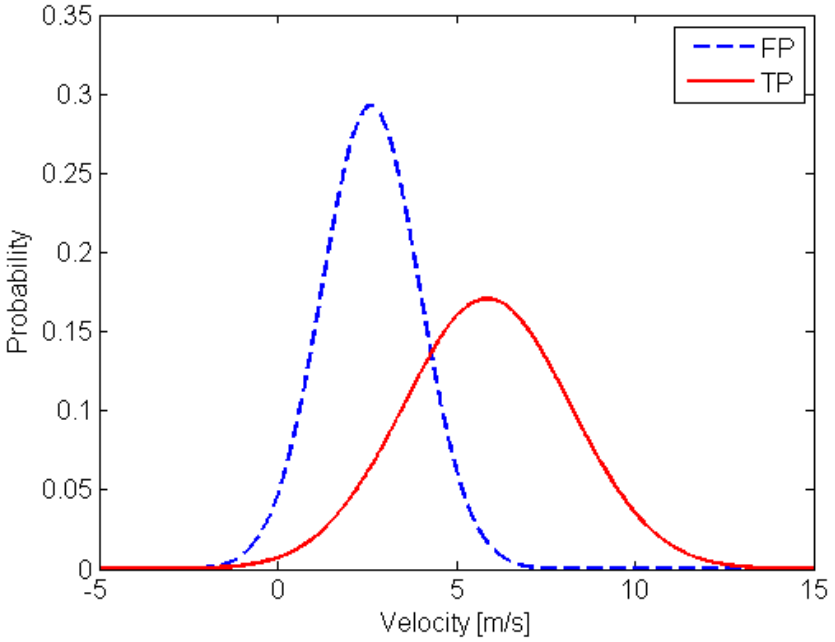

Figure 2.11. Probability distribution functions of velocities difference.

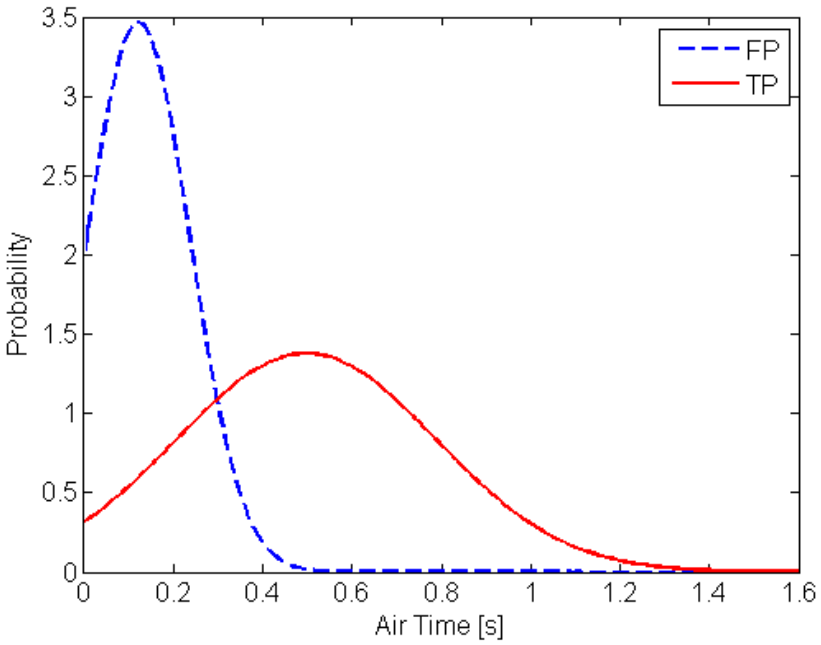

Figure 2.12. Probability distribution functions of air time.

The thresholds found were then applied to the algorithm in different combinations. Five different combinations of threshold sets were used, as shown in Table 2.2. Threshold Sets 1 and 2 were derived from the two methods mentioned above while Sets 3, 4, and 5 are variations of Set 1, designed to observe the effectiveness of each parameter.

Table 2.2. Threshold parameter values of five threshold sets

Sets	Peak Acc.	Vel. Diff.	Air Time
Threshold Set 1	1.1g	3.1m/s	0.12s
Threshold Set 2	1.25g	4.29m/s	0.30s
Threshold Set 3	1.1g	0m/s	0s
Threshold Set 4	1.1g	3.1m/s	0s
Threshold Set 5	1.1g	0m/s	0.12s

Figure 2.13 shows the performance of the proposed algorithm on the five types of jumps when the above threshold sets were applied.

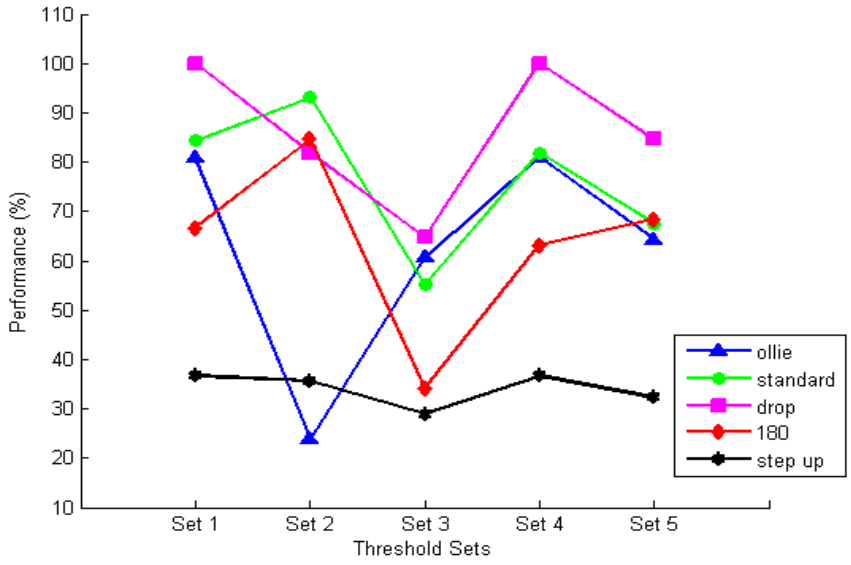

Figure 2.13. Algorithm performance under different threshold sets.

The overall performance, sensitivity, and specificity corresponding to each threshold set are shown in Figure 2.14, and tabulated in Table 2.3.

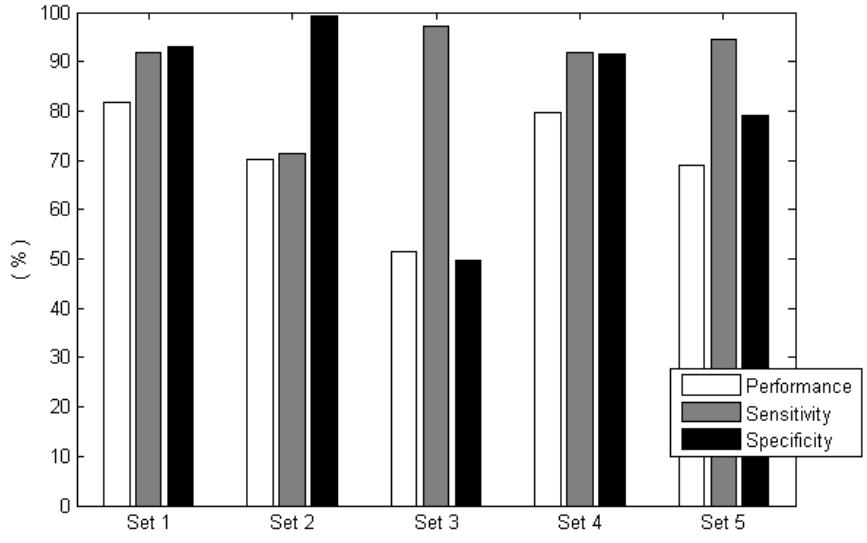

Figure 2.14. Overall algorithm performance, sensitivity, and specificity for different threshold sets.

Table 2.3. Overall performance, sensitivity, and specificity

	Performance	Sensitivity	Specificity
Threshold Set 1	81.7%	91.8%	93.0%
Threshold Set 2	70.3%	71.2%	99.2%
Threshold Set 3	51.4%	97.3%	49.6%
Threshold Set 4	79.8%	91.8%	91.5%
Threshold Set 5	69.0%	94.5%	79.1%

By varying the threshold value for each parameter, different sensitivities and specificities were achieved. The idea is to attain perfect sensitivity and specificity; however, in most cases, sensitivity and specificity are inversely proportional [36]. Thus higher sensitivity or higher specificity can be chosen to be more favorable. For instance, the detection algorithm can be used for two purposes in skiing/snowboarding. For training, one can repeat jumps if they were not detected and missed. Therefore, specificity, which is the ability to distinguish jumps from other movements, is more important in this case. For entertainment or competition, the athlete usually performs a jump or a series of jumps once and would like it to be not missed. As a result, sensitivity, which is the ability to detect as many jump occurrences as possible, is more favorable in this case. In our study, performance, which is a measure of both sensitivity and specificity, is used to evaluate the results.

Threshold Sets 1 and 2 were derived directly from the method above. The first set of thresholds gave 91.8% in sensitivity and 93.0% in specificity. On the other hand, with the second set of thresholds, specificity was improved to 99.2% but sensitivity dropped to 71.2%, which was to be expected as the thresholds were stricter. From the probability distribution graphs, Figure 2.10 to Figure 2.12, the thresholds are set at the intersecting points of the true positive and false positive curves. Any value above the thresholds is classified as a true positive and any value below the threshold is classified as a false positive. It can be observed that as the thresholds increase, some false positives would be correctly identified while a lot more true positives would be incorrectly classified. In other words, the performance of the algorithm is obviously sensitive to the threshold values, and finding the optimal thresholds for each parameter is important.

The probability distribution graphs can also be an indicator of how efficient the parameters can be used to separate the two groups, TP and FP. From the pdf curves, the farther apart the peaks are, the less the overlapping region under the two curves, and thus, the fewer the number of misclassified cases. In Figure 2.10, most of the region under the two curves overlapped. This indicated that the peak acceleration after the jump, alone, was not good enough to separate the two groups. On the other hand, the pdf curves of the velocity difference and that of the air time were fairly well separated from one group to another. Therefore, in addition to the peak acceleration, the velocity difference and air time can be used to improve the performance of the system.

Threshold Sets 3, 4, and 5, which are variations of Set 1, are used to observe the effectiveness of each parameter to the algorithm. Threshold Set 3 used only the peak acceleration to distinguish jumps from other activities. Because the criteria were less strict, it produced highest sensitivity, but on the contrary, lowest specificity. Because of the low performance on specificity, the overall performance of Set 3 was 51.4% even though the sensitivity reached 97.3%. In addition, when the velocity difference and air time were introduced in the decision making criteria, as illustrated in Threshold Sets 4 and 5, respectively, the sensitivity dropped a little bit while the specificity increased dramatically. Among these two parameters, the threshold on the vertical velocity showed a significant improvement on the overall performance, which is a key finding of our work. When all of the parameters were introduced into the decision criteria, the best performance was achieved.

The proposed algorithm utilizes only one sensor attached to the head. Any head movement would create noise in measurements that can affect the performance of the algorithm. To improve performance, one option is to move the sensor to the lower back/waist area because the sensor would be less susceptible to unwanted head movements. Another option is to increase the number of sensors used (at multiple locations) so they can complement each other if any of them is suffering from noises or disturbances.

2.5. Conclusion

This chapter presents a real-time jump detection algorithm using head-worn sensors specifically for skiing and snowboarding applications. Vertical direction information was decoupled from information in other directions in order to filter out the errors caused by noise and other uncertainties, such as vibrations of the head from breaking or skidding over the hard snow or the skin artefact caused by the helmet on which the sensor is attached to is not secured tightly to the head [14]. Four thresholds are applied to the algorithm for detecting and verifying jump occurrences. The primary signal used is acceleration, while vertical velocity and air time are used as supplemental information to verify the jumps. By setting thresholds on vertical velocity and air time, the overall performance improved by more than 30% compared to acceleration alone.

However, this threshold-based algorithm is highly dependent on how well the thresholds are set. A small change in the threshold value may have a dramatic effect on the overall performance. Determining the threshold values can be challenging, because IMU signals vary between individuals. In addition, different type of jumps may cause different impacts on the ground before the jump or after the landing, and thus, generate various IMU signal values. Nevertheless, with proper selection of threshold values, the algorithm showed 91.8% sensitivity with 93.0% specificity in our field experiments.

After jump detection, features in these jump windows can be extracted for jump classification, which will be discussed in next chapter. Furthermore, post processing techniques, such as filtering and smoothing, can be applied to these jump windows to get more accurate KPVs for both entertainment and training purposes.

Chapter 3.

Automatic Jump Classification

3.1. Introduction

This chapter presents a two-stage ski/snowboarding jumps classification algorithm for both training and entertainment purposes. First, a jump is detected with the threshold based algorithm described in the previous chapter, and it is, then, classified by a classification model. A support vector machine (SVM) is used to formulate the classification model. The remainder of this chapter is organized as follows. In Section 3.2, the proposed ski/snowboard jump classification algorithm is explained in detail. The experimental protocols and setup are explained in Section 3.3. Section 3.4 presents the classification results and provides suggestions to improve the performance. Finally, Section 3.5 concludes the paper.

3.2. Methodology

Our proposed approach to activity classification is illustrated in Figure 3.1. As shown, activity classification is typically a multi-stage process. The sensor signal is first divided into small segments of windows and this step is called signals segmentation. In the proposed jump classification algorithm, sensor fusion with three cascaded Kalman filters was used to detect the jumps and divides the signals into jump windows. After selecting the windows, feature extraction is applied to each segment. These features are then used for activity classification with machine learning algorithms. The following sessions discuss each component of the jump classification method in Figure 3.1 in details.

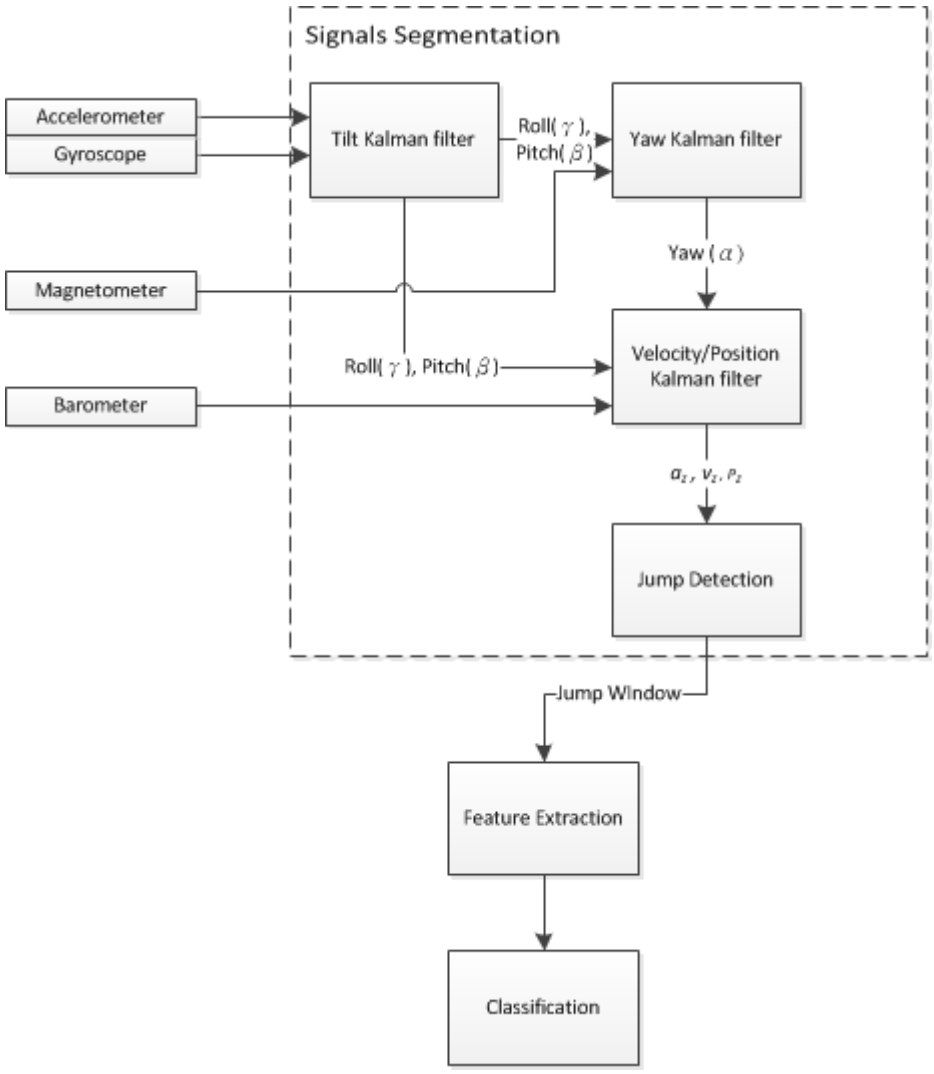

Figure 3.1. Proposed jump classification method.

3.2.1. Segmentation Techniques

The segmentation techniques can be divided into three categories: (i) sliding windows, (ii) activity-defined windows, and (iii) event-defined windows [37]. Sliding window segmentation is the most widely used approach in activity classification. The windows size is predefined and is fixed throughout the entire classification process. In some applications, these windows are in conjunction with consecutive windows [38]-[40], while in other applications, these windows are overlapping with one another [41]-[42]. These sliding windows cover the entire set of data and are very easy to implement. In activity-defined windows segmentation, the data is segmented into different windows

based on the activities performed [43]. Usually a simple classification algorithm is implemented to divide the data into different activity groups based on the intensity level of the activities [44]. The entire data set is divided into different window segments, which vary in length depending on the duration of the activities performed. The event-defined windows segmentation uses similar approach. An algorithm is first implemented to detect the start of the event and the end of the event. In Nyan et al. [45], the windows are segmented by gait cycles. The lengths of the windows are different because of the difference in gait duration.

Because the duration of the jumps vary from person to person and from style to style, a fixed-length sliding window is not appropriate for this application. If the window size is too big, it may contains two or more jumps in one window. On the other hand, if the window size is too small, it may not be enough to cover the entire jump. Furthermore, the activity-defined window is not suitable for signal segmentation, because jumps usually happen in a short period of time and the intensity level of jumps may or may not be very different from other skiing/snowboarding activities. The event-defined windows are thus best suited for jump classification. The jump windows are detected as described in our recent work [22]. Novel Kalman filtering-based sensor fusion is applied on IMU and barometer pressure data to extract vertical acceleration, which is the primary data for jump detection, and vertical velocity, which is used as supplementary information to validate the jumps. Three cascaded Kalman filters - Tilt Kalman filter, Yaw Kalman filter, and Velocity/Position Kalman filter - are applied on the raw data to obtain the desired signals mentioned above. The details of the Tilt Kalman filter and Yaw Kalman filter are summarized in the Appendix and the Velocity Kalman filter in the previous chapter. Based on this, a threshold-based algorithm is implemented to detect the start and the end of the jumps. The vertical acceleration is the primary detection signal. Once the vertical acceleration exceeds the threshold value, a potential jump start is recorded and, when the vertical acceleration drops to zero and goes back up to the standard value, a potential jump end is recorded. Other thresholds set on the vertical velocity and the air time are used to validate the jump window. Following that, the window is defined from half a seconds before the start of the jump to half a seconds after the end of the jump. After the windows are selected, features are extracted from these windows for classification as described below.

3.2.2. Features Extraction and Selection

Feature extraction and selection are used to reduce the signals into features that depict the activities within the given window. Previous classification studies used different approaches to generate features, which are then used as input to the classification models. Generally, these features can be grouped into different categories: (i) time-domain features, (ii) frequency-domain features, and (iii) characteristic features. The time-domain features are typically statistical features, such as the maximum, minimum, average, variation, skewness, kurtosis, autocorrelation, etc., which are normally derived directly from the given windows. On the other hand, to derive frequency domain features, the data is transformed into the frequency domain. Since the data is discrete, the fast Fourier transformation (FFT) is normally used to convert the windows into the frequency domain. The frequency-domain features, which are often used to differentiate activities with different intensity levels, include peak frequencies, sum of the frequency coefficients, frequency-domain entropy, and so on. Besides these two categories, characteristic features portray the essence of the activities performed. Examples include the degree of rotation of the body segment, the inclination of the posture, the difference in height of the body segment, and other characteristics of the repetitive movement sequences.

Because in some applications, tens and thousands of features can be generated, variables and feature selection have been studied to improve the processing time and the prediction performance [46]. Three common feature selection methods are (i) filter method, which selects the features based on scoring function, (ii) wrapper method, which assess the usefulness of a subset of features according to the prediction performance, and (iii) embedded methods, which incorporate feature selection into the training process [47]. For multi-label classification, because the classes are not usually mutually exclusive, the subsets of features may affect the performance differently [48]. As a result, a variation of the wrapper method is implemented in our algorithm. The initial set of features are the first peak and the last peak of vertical acceleration data within the window, the maximum and minimum value in vertical velocity and gyroscope data, the average, variance, skewness, and kurtosis in vertical acceleration, vertical velocity, and gyroscope data, the cumulative yaw angle from start of the window to the end of it, the

height from the starting ground to the maximum height reached, and the drop from maximum height to landing. These features, which can be divided into time-domain and characteristic features, are calculated as follows:

- Time-domain features:

$$\mathrm{Peak1}_{\mathrm{acc}} = \max\left(\boldsymbol{a}_{\mathrm{z}}(start: start + \Delta\mathrm{t})\right) \tag{3.1}$$

$$\mathrm{Peak2}_{\mathrm{acc}} = \max\left(\boldsymbol{a}_{\mathrm{z}}(end - \Delta\mathrm{t}: end)\right) \tag{3.2}$$

$$\mathrm{Max}_{\mathrm{vel}} = \max\left(\boldsymbol{v}_{\mathrm{z}}(start: start + \Delta\mathrm{t})\right) \tag{3.3}$$

$$\mathrm{Min}_{\mathrm{vel}} = \min\left(\boldsymbol{v}_{\mathrm{z}}(end - \Delta\mathrm{t}: end)\right) \tag{3.4}$$

$$\mathrm{Max}_{\mathrm{gyrox,y,z}} = \max\left(\mathbf{y}_{\mathrm{G_{x,y,z}}}(start: start + \Delta\mathrm{t})\right) \tag{3.5}$$

$$\mathrm{Min}_{\mathrm{gyrox,y,z}} = \min\left(\mathbf{y}_{\mathrm{G_{x,y,z}}}(end - \Delta\mathrm{t}: end)\right) \tag{3.6}$$

$$\mathrm{average(x)} = \frac{1}{\mathrm{N_x}}\sum_{\mathrm{i=1}}^{\mathrm{N_x}} \mathrm{x_i} = \bar{\mathrm{x}} \tag{3.7}$$

$$\mathrm{variance(x)} = \frac{1}{\mathrm{N_x}}\sum_{\mathrm{i=1}}^{\mathrm{N_x}} (\mathrm{x_i} - \bar{\mathrm{x}})^2 = \sigma^2 \tag{3.8}$$

$$\mathrm{skewness(x)} = \frac{1}{\mathrm{N_x}\sigma^3}\sum_{\mathrm{i=1}}^{\mathrm{N_x}} (\mathrm{x_i} - \bar{\mathrm{x}})^3 \tag{3.9}$$

$$\mathrm{kurtosis(x)} = \frac{1}{\mathrm{N_x}\sigma^4}\sum_{\mathrm{i=1}}^{\mathrm{N_x}} (\mathrm{x_i} - \bar{\mathrm{x}})^4 \tag{3.10}$$

- Characteristic features:

$$\text{cumulative yaw angle} = \sum_{\mathrm{i=2}}^{\mathrm{N_x}} (\alpha_i - \alpha_{i-1}) \tag{3.11}$$

$$\text{height} = \boldsymbol{p}_{z_{max}} - \boldsymbol{p}_{z_{start}} \tag{3.12}$$

$$\text{drop} = \boldsymbol{p}_{z_{max}} - \boldsymbol{p}_{z_{end}} \tag{3.13}$$

where $\boldsymbol{a}_{\mathrm{z}}$ is the acceleration data on the vertical direction, $\boldsymbol{v}_{\mathrm{z}}$ is the velocity data on the vertical direction, $\mathbf{y}_{\mathrm{G_{x,y,z}}}$ are the tri-axial gyroscope data, $start$ is the very first point of the window, end is the last point of the window, $\Delta\mathrm{t}$ is a small interval of time that is smaller

than the window size, $\mathbf{x}_\mathrm{i} = \{\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_{\mathrm{N_x}}\}$ are the data points in the window, and $\boldsymbol{p}_z$ is the vertical position data.

In order to apply the wrapper method for features reduction, the initial set of features is divided into two subsets. The first subset consists of statistical features that include average, variance, skewness, and kurtosis. The second subset is the characteristic features, which are cumulative yaw angle, height, and drop. These two subsets of features are passed into the classification model and its performance is evaluated based on classification accuracy. Since characteristic features capture the essence of different activities, it generally produced higher classification accuracy than statistical features. Therefore, characteristic features, such as cumulative yaw angle, height, and drop, are used as a base subset. The statistical features are then added to the base subset one by one and the classification performance is re-evaluated every time when addition features are included. If the additional feature has no effect on the accuracy, the feature is removed to increase the efficiency of the classification model. Features such as kurtosis of acceleration, the statistical features of gyroscope data on x, y, and z axes do not affect the classification result and are thus removed from the feature set.

3.2.3. Classification

After features are selected, they are passed into the training algorithm to derive a classification model. A support vector machine (SVM), one of the most efficient classification algorithms, is implemented in our work. The objective of the SVM classifier is to find an optimal hyper-plane between different classes in the feature space. Figure 3.2 is an example of a linearly separable data set in a two-dimensional (2-D) feature space. The nature of the SVM classifier can be illustrated with two linearly separable classes in a 2-D feature space. The training data is represented by $\{\mathbf{x}_\mathrm{i}, t_\mathrm{i}\}, \mathrm{i} = 1, 2, \ldots \mathrm{N}, t_\mathrm{i} \in \{1, -1\}$, where $\mathbf{x}_\mathrm{i}$ is the feature set of the i^{th} training data point and t_i is class label for the corresponding data point. Let $\boldsymbol{w}$ be the weight of the features and b be an arbitrary constant, the simplest representation of a linear boundary in feature space can be modeled as

$$y(\mathrm{x}) = \boldsymbol{w}^{\mathrm{T}}\phi(\mathbf{x}_{\mathrm{i}}) + b, \tag{3.14}$$

where $\phi(\cdot)$ is the feature vector in the feature space.

As shown in Figure 3.2, the data can be separated by infinite number of lines that fit into the space between the two classes, but there is only one optimal plane such that the margin, which is defined as the closest distance between the decision boundary and any of the data points in both classes, is as large as possible. Figure 3.3 shows the definition of a margin. The data point lies on the margin is known as support vector.

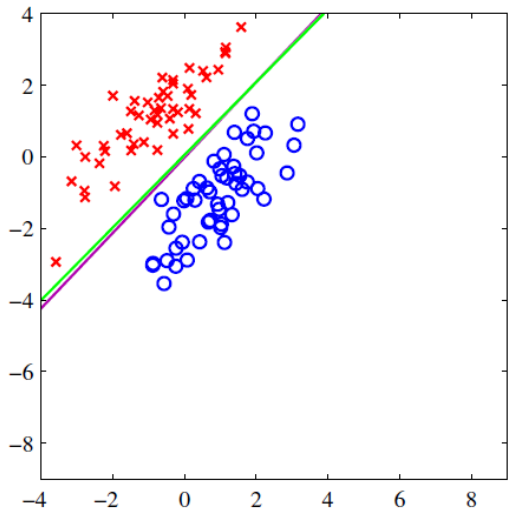

Figure 3.2. Linear separable data set in 2-D feature space [35].

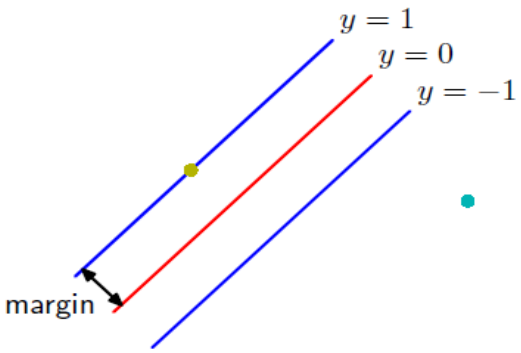

Figure 3.3. Illustration of margin, the distance between decision boundary to the closest data point in the class [35].

As illustrated in Figure 3.4, the distance, D(x), of the data point to the decision boundary can be represented by

$$\mathrm{D(x)} = \frac{|y(\mathrm{x})|}{\|\boldsymbol{w}\|} = \frac{t_{\mathrm{i}} y(\mathbf{x}_{\mathrm{i}})}{\|\boldsymbol{w}\|} = \frac{t_{\mathrm{i}}(\boldsymbol{w}^{\mathrm{T}}\phi(\mathbf{x}_{\mathrm{i}}) + b)}{\|\boldsymbol{w}\|}, \tag{3.15}$$

knowing that $t_{\mathrm{i}} y(\mathbf{x}_{\mathrm{i}}) > 0$ because only the correctly classified data points will be used in the support vector.

A SVM seeks the optimal decision boundary by maximizing the margin with the following equation:

$$\arg\max_{\mathbf{w},b}\left\{\frac{1}{\|\boldsymbol{w}\|}\min_{\mathrm{i}}[\mathrm{t_i}(\boldsymbol{w}^{\mathrm{T}}\phi(\mathbf{x}_{\mathrm{i}}) + b]\right\}. \tag{3.16}$$

Since w and b are arbitrary, they can be rescaled so that $\boldsymbol{w} \to \zeta\boldsymbol{w}, b \to \zeta b$, where $\zeta = \min\left(\mathrm{t_i}\left(\boldsymbol{w}^{\mathrm{T}}\phi(\mathbf{x}_{\mathrm{i}}) + b\right)\right)$ is the minimum distance from the data points to the decision boundary. The equations become

$$\arg\min_{\mathbf{w},b}\frac{1}{2}\|\boldsymbol{w}\|^2, \tag{3.17}$$

subject to the following constraint:

$$\mathrm{t_i}(\boldsymbol{w}^{\mathrm{T}}\phi(\mathbf{x}_{\mathrm{i}}) + b) \geq 1, \qquad \mathrm{i} = 1 \ldots \mathrm{N}. \tag{3.18}$$

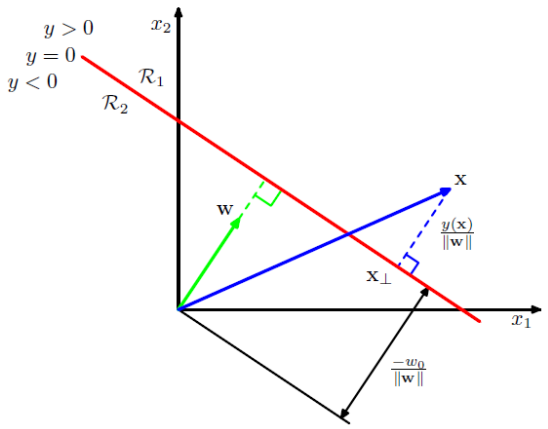

Figure 3.4. The geometry of the linear classier boundary and a data point x [35].

However, in most of the cases, the data points are not linearly separable. Therefore, slack variables, ξ_i, and penalty parameter, C, is introduced to (3.17), i.e.

$$\min_{\mathbf{w},b,\boldsymbol{\xi}} \left\{ \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N} \xi_i \right\}, \tag{3.19}$$

subject to the constraint,

$$\mathrm{t_i}(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x_i}) + b) \geq 1 - \boldsymbol{\xi}_i, \qquad \mathrm{i} = 1 \ldots \mathrm{N}. \tag{3.20}$$

Figure 3.5 is an illustration of the slack variables. The slack variables are defined such that $\xi_i = 0$ if the data points lie either on the margin or on the correct side of the margin. If the data points lie inside the margin and is on the correct side of the decision boundary, then $0 < \xi_i < 1$. Data points that lie on the wrong side of the decision boundary and are misclassified have slack variables $\xi_i > 1$. The penalty parameter, $C > 0$, is a constant that controls the trade-off between slack variable penalty and the margin. With a low penalty parameter, the model may be less accurate since the penalty for the falsely classified data points is negligible compared to the margin size. On the other hand, if the penalty parameter is too high, the SVM model may over fit to the training data and thus have lower generalization ability.

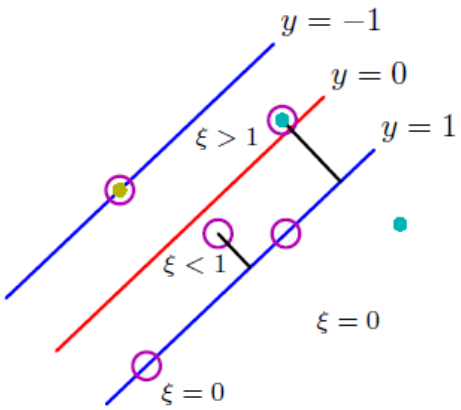

Figure 3.5. Illustration of slack variables [35].

Typically, (3.19) and (3.20) are rearranged to a dual form, by introducing the Lagrange multipliers $a_i \geq 0, i = 1, \ldots, N$, which simplifies learning nonlinear SVMs [49]. The dual form is represented as

$$W = \min_{a_i} \left\{ \frac{1}{2} \sum_{i,j=1}^{N} a_i \boldsymbol{Q}_{ij} a_j - \sum_{i=1}^{N} a_i + b \sum_{i=1}^{N} t_i a_i \right\}, \tag{3.21}$$

where $\boldsymbol{Q}_{ij} = t_i t_j \phi(\mathbf{x}_i) \phi(\mathbf{x}_j) = t_i t_j \boldsymbol{k}(\mathbf{x}_i, \mathbf{x}_j)$ and subjected to the following constraints

$$0 \leq a_i \leq C, i = 1, \ldots, N. \tag{3.22}$$

A positive definite kernel function, $\boldsymbol{k}(\mathbf{x}_i, \mathbf{x}_j)$ is used to implicitly map the examples into a higher dimensional feature space and compute the dot product. The geometry of the nonlinear decision boundary is determined by the dimensions of the kernel function employed. Some common kernels are linear, polynomial, Gaussian radial basis, and sigmoid [50]. The performance of the SVM classification model depends on the selection of the kernel and its parameters. A grid search is carried out on the testing data to find the best set of penalty parameter and the parameters for the kernel, C and $\boldsymbol{\gamma}$ respectively.

As discussed so far, SVM is designed for binary decisions. In the jump classification problem, however, there are more than two classes of jumps. Different methods, such as one-against-one, one-against-all, and directed acyclic graph SVM (DAGSVM), have been proposed to construct a multi-label classifier by combining multiple binary SVM classifiers [51]. Generally, one-against-one method produces better classification accuracy and requires less training and testing time. Therefore. it is used in the proposed jump classification SVM model.

3.2.4. Performance Evaluation

The evaluation of classification performance depends greatly on the problem itself. A confusion matrix is commonly used to evaluate the accuracy of a multiclass classification problem [52]. Table 3.1 is an example of a binary class classification confusion matrix. Any size of multiclass classification confusion matrix can be decoupled into several binary class classification matrices. The first column from the left is the

actual label while the first row is the predicted label. The confusion matrix has four categories. True Positive (TP) is defined as positive examples correctly predicted as positive. True Negative (TN) is the negative examples correctly predicted as negative. When the positive examples are incorrectly predicted as negative, they are counted toward the False Negative (FN). Conversely, False Positives (FP) are negative examples incorrectly predicted as positive.

Table 3.1. Confusion matrix

	Positive (Predicted)	**Negative (Predicted)**
Positive (Actual)	True Positive (TP)	False Negative (FN)
Negative (Actual)	False Positive (FP)	True Negative (TN)

From the confusion matrix, sensitivity (also known as Recall), Specificity, Precision, True Positive Rate (TPR), and False Positive Rate (FPR) can be found with the following equations [53]:

$$\text{Sensitivity(Recall)} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3.23}$$

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}} \tag{3.24}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{3.25}$$

$$\text{True Positive Fraction(TPF)} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3.26}$$

$$\text{False Positive Fraction(FPF)} = \frac{\text{FP}}{\text{TN} + \text{FP}} \tag{3.27}$$

$$\text{Performance} = \frac{\text{TP}}{\text{TP} + \text{FP} + \text{FN}} \tag{3.28}$$

In the skiing/snowboarding jump classification, the sensitivity is a measure of "within class" accuracy. The correctly classified jumps are assessed out of all the jumps in that particular class. On the other hand, the specificity is a measure of "between class" accuracy, which evaluates how well one particular class is distinguished from other classes. The precision calculates the correctly classified jumps out of all the jumps labeled to that particular class. Usually when the data is skewed, i.e. TN >> TP, precision is used rather than sensitivity to evaluate an algorithm [54]. Another evaluation

parameter, performance, is introduced by combining TP, FP, and FN to evaluate the overall performance. TN is not included in the equation because TNs of one group fall in TPs, FPs, or FNs of other groups. All these parameters are desired to be as close to 1 as possible.

3.3. Experimental Setup

3.3.1. Experimental Setup and Protocol

Field tests were carried out at local ski resorts (Cypress Mountain, British Columbia, Canada and Whistler Mountain, British Columbia, Canada). An integrated sensor consisting of an IMU, a GPS, and a barometric pressure sensor, sampling at 100Hz, 1Hz, and 20Hz, respectively, was mounted on the top of the helmet as shown in Figure 3.6. The helmet was worn by the test subjects and the data was transmitted wirelessly through Bluetooth connection to a laptop. Also, the tests were recorded by GoPro Hero 3+ at 100 fps to keep track of the activities of subjects.

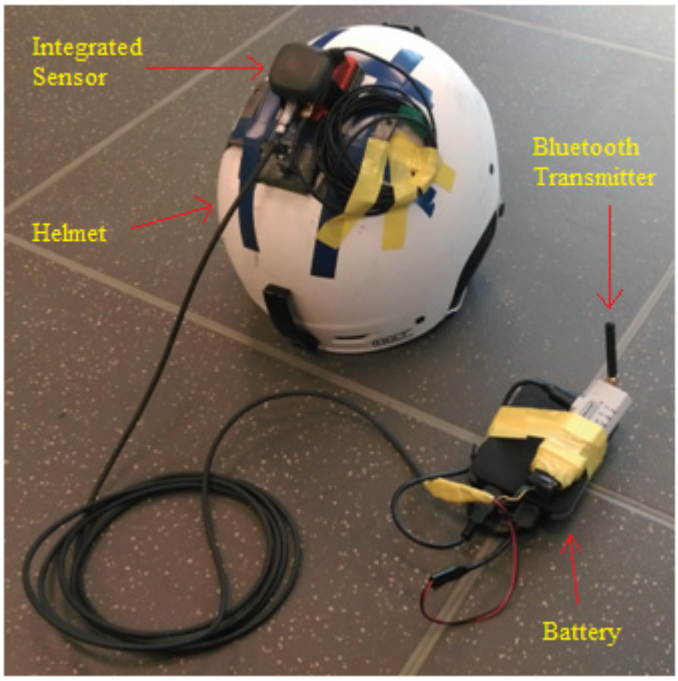

Figure 3.6. Integrated sensor setup.

The subjects were asked to perform the following types of jumps listed as follows:

1. Ollie: The subject springs off from the level ground and lands.
2. Standard Jump: The subject springs off from a ramp and lands.
3. Drop: The subject slides off from a ramp, box, or rail.
4. Step-up Jump: The subject hops on to a box or a ramp.

The IMU sensor is calibrated before the start of each test. The accelerometer is calibrated by asking the subjects to stay still for about 30 seconds before performing the jumps. In addition, the magnetometer is periodically calibrated by asking the subject to rotate the IMU sensors in all directions for another 30 seconds.

3.3.2. Participants

Nine healthy male subjects aged between 19 and 28 participated in the field experiment. The participants were recruited through internal postings at the University and invitations on site. All participants had at least several years of skiing or snowboarding experience. Some of them have acquired their skiing or snowboarding instructor license while others were proficient in skiing or snowboarding and comfortable with jumping or doing tricks in terrain parks. They were informed of the experimental protocol (No, 2013s0790), which was approved by the Research Ethics Board at Simon Fraser University in advance. Signed consent forms were also obtained. The subjects were asked to perform the jumps that they were most comfortable with. Table 3.2 shows jumps completed by the test subjects. All the jumps are labeled manually, so the accuracy of the classification algorithm is independent of that of the jump detection algorithm. The fifth subject carried out the jumps on skis while the rest of them used snowboards, hence there were one skier (s05) and eight snowboarders in total.

Table 3.2. **Jumps performed by subjects**

	Ollie	Standard	Step Up	Drop
s01		10	4	4
s02		6		2
s03		11	5	5
s04	7			
s05		*5*		*5*
s06	5	8		4
s07	8			
s08	9	9	5	6
s09	12	8	4	5
Total	**36**	**41**	**18**	**31**

3.4. Results and Discussion

The classification model is trained with LIBSVM [55], which uses one-against-one approach for multi-labeled SVM classification. In order to simulate the real-time application, the jumps of one subject are used as testing data, while the rest is used to train the classification model. Before training the classification model, feature selection was carried out to obtain better classification accuracy and efficiency. Intuitively, two important features used to classify the jumps are height and drop, defined in (3.12) and (3.13). Figure 3.7 shows the relationship of height and drop values for different group of jumps.

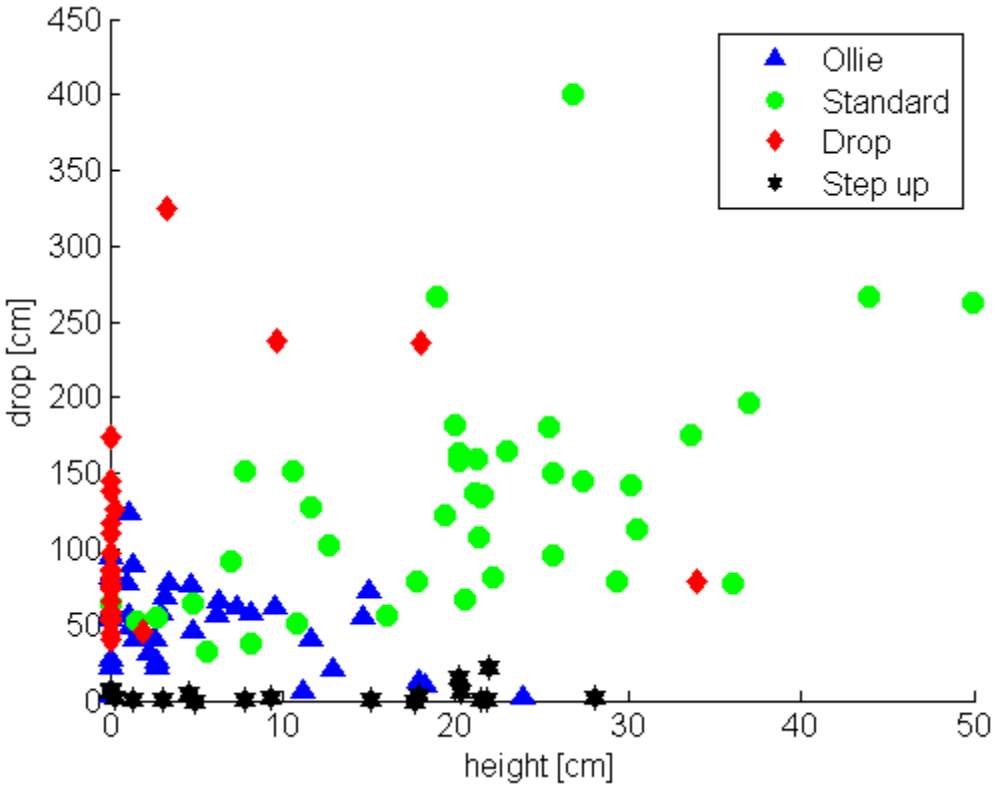

Figure 3.7. 2-D features plot of height and drop values.

Figure 3.8 and Figure 3.9 show the probability distribution curve of drop and height feature for different jump categories. The drop feature is a good indicator for the Step up jumps. From Figure 3.8, the Step up jumps are very distinctive, because most of the time, the drop of the Step up jumps is much smaller than that of the other jumps. On the other hand, the height feature may not be as distinct from one category to another, because height is dependent on the size of the jump performed. Sometimes, the height of a big Drop may be very similar to the height of a small Ollie jump. Nevertheless, the peak acceleration, achieved right before the jump starts, can be used to differentiate Drops from other type of jumps.

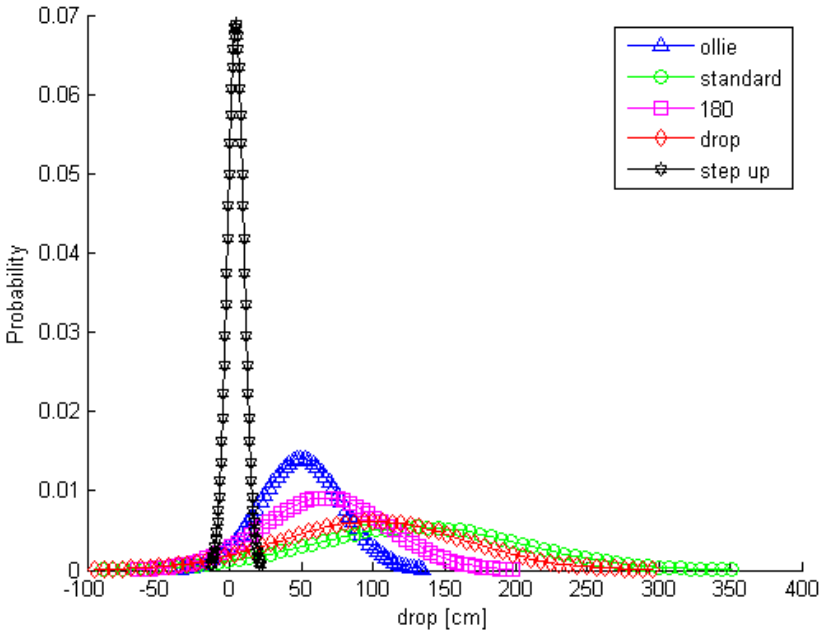

Figure 3.8. **The probability distribution curve of the drop feature for different jump groups.**

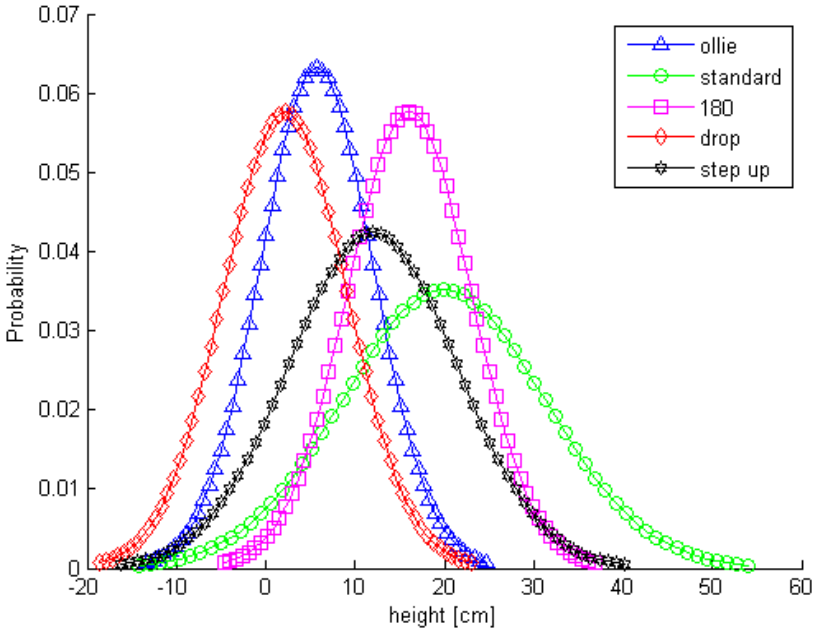

Figure 3.9. **The probability distribution curve of the height feature for different jump groups.**

Figure 3.10 is the probability distribution curve of the first acceleration peak feature of the jump windows. Usually, when Drops are carried out, the test subjects do not have to exert too much force on the ground. Instead, they slide off the ramp or the box to complete the Drops. As a result, the initial acceleration of Drops is smaller than that of other classes of jumps. By combining different features together, the jump classification can be more efficient and accurate. Having said that, it does not follow that more features always improves the overall performance of the classification model. As the size of the features increase, the processing time for the classification decision increases as well. Not all the features are useful in classifying these jumps. Examples are the average of gyroscope data on the x axis and y axis, which are shown in Figure 3.11 and Figure 3.12 respectively. From the figures, the curves are overlapping each other from group to group. Including these features does not add any benefit to the classification result, but decreases the decision efficiency.

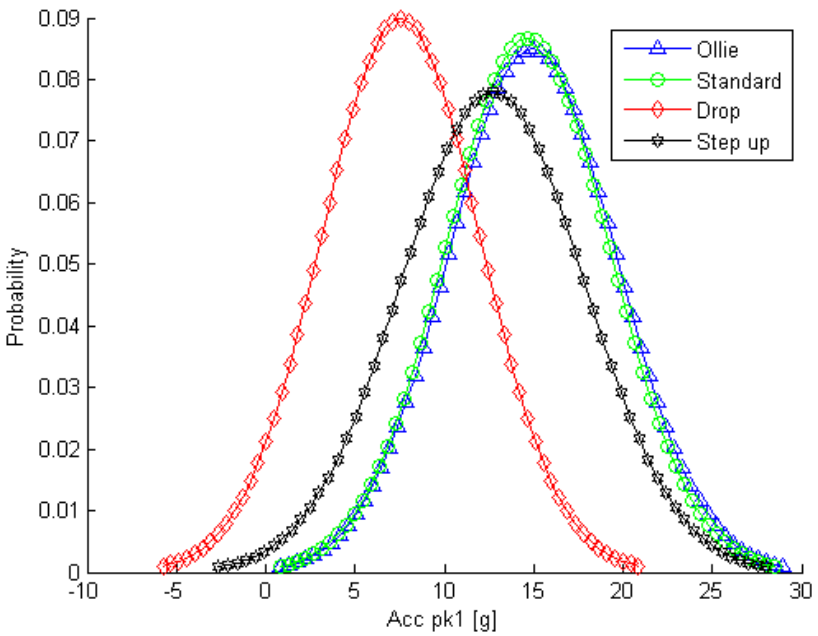

Figure 3.10. The probability distribution curve of the first acceleration peak feature for different jump groups.

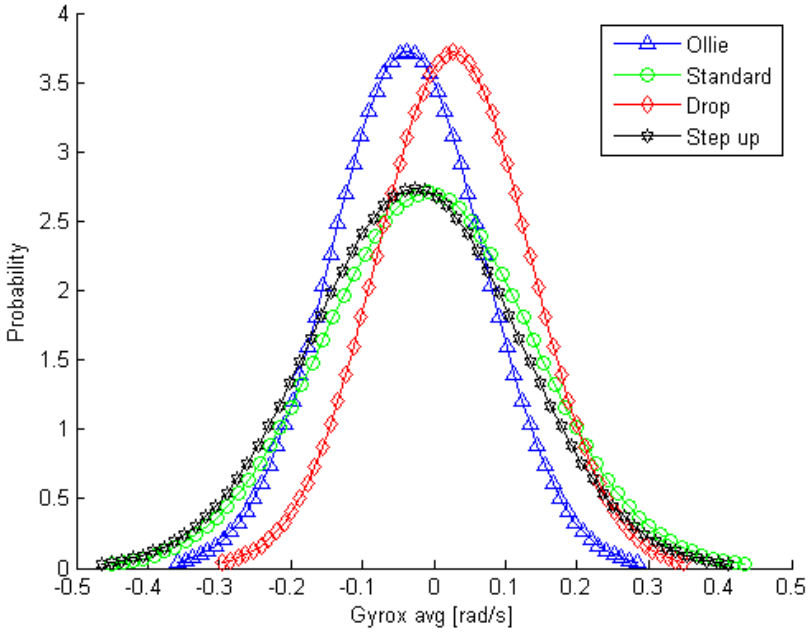

Figure 3.11. The probability distribution curve of the average of gyroscope x-axis feature for different jump groups.

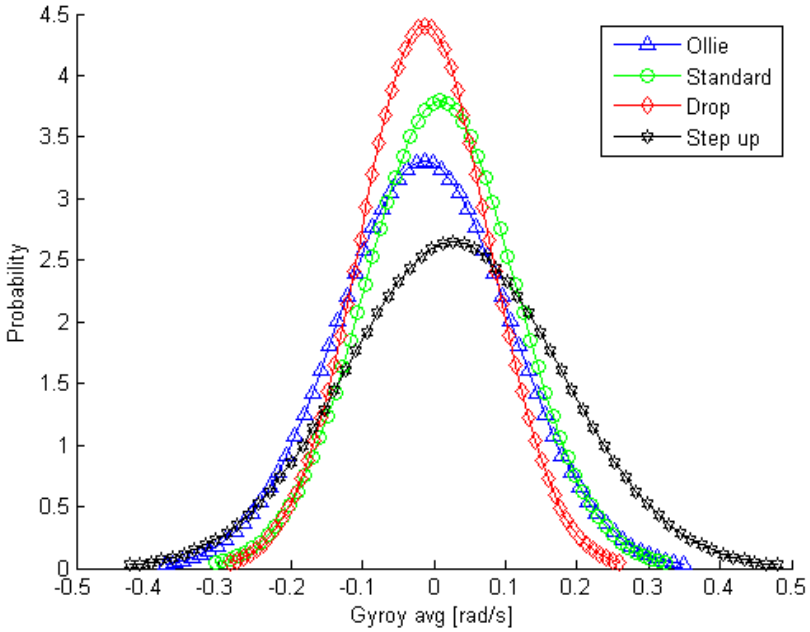

Figure 3.12. The probability distribution curve of the average of gyroscope y-axis feature for different jump groups.

In summary, the final features that were included in the classification model to produce the results in this paper are first and second peaks of vertical acceleration, the maximum and minimum values of vertical velocity and tri-axis gyroscope data, the yaw rotation, the height, and the drop.

As mentioned above, the test subjects consisted of one skier and eight snowboarders. The data analysis is carried out by *A*) training and testing the snowboarding jumps with the characteristic features only, *B*) training and testing the snowboarding jumps with both the characteristic features and the time-domain features, and *C*) including the skiing jumps to the previous test. For the sake of simplicity, these tests will be referred to as *Test A*, *Test B*, and *Test C*, respectively. The SVM parameters, C and $\boldsymbol{\gamma}$, are tuned by using the grid search method from 2^0 to 2^5, and from 2^{-15} to 2^3, respectively, on the training data. Since the training data for each test subject is different, the values of these parameters are different as well. From *Test A* and *Test B*, improvement of the classification model by including the time-domain features in the feature set is observed. The results of *Test A* and *Test B* are tabulated in Table 3.3. By introducing some of the time-domain features into the feature set, the overall sensitivity, specificity, precision, and performance were improved by 16.1%, 4.0%, 16.1%, and 25.9%, respectively.

Table 3.3. The overall sensitivity, specificity, precision, and performance of *Test A* and *Test B*

	Test A	*Test B*	Improvement
Sensitivity	65.4%	75.9%	16.1%
Specificity	88.5%	92.0%	4.0%
Precision	65.4%	75.9%	16.1%
Performance	0.486	0.612	25.9%

Table 3.4 shows the SVM parameters used for each test subject and the accuracy achieved for *Test B* and *Test C*. When including the ski jumps into the training data, the overall accuracy was 80.5% (compared to 75.9%). The accuracy is the ratio of correctly classified jumps to the total number of jumps performed. It does not provide any information on the specific type of jumps. The sensitivity, specificity, precision and performance for each test and jump class are show in Table 3.5.

Table 3.4. Parameters used for each testing subjects and the accuracy achieved for *Test B* and *Test C*

No.	*Test B*			*Test C*		
	C	γ	Accuracy	C	γ	Accuracy
s01	2	1.221E-04	72.22%	2	1.221E-04	94.44%
s02	2	1.221E-04	87.50%	2	1.221E-04	87.50%
s03	2	2.883E-04	80.95%	8	1.221E-04	76.19%
s04	2	9.766E-04	100.00%	2	9.766E-04	100.00%
s05				2	1.221E-04	70.00%
s06	8	1.221E-04	76.47%	2	9.766E-04	82.35%
s07	8	1.221E-04	87.50%	8	1.221E-04	87.50%
s08	2	1.221E-04	86.21%	2	1.221E-04	89.66%
s09	2	1.221E-04	48.00%	2	2.441E-04	52.00%
All			75.9%			80.5%

Table 3.5. Sensitivity, specificity, precision, and performance for different types of jumps

	Test A	*Test B*	*Test C*	
Ollie	51.4%	70.3%	70.3%	Sensitivity
Standard	75.0%	78.8%	82.7%	
Drop	61.5%	69.2%	84.6%	
Step up	72.2%	88.9%	88.9%	
All	65.4%	75.9%	80.5%	
Ollie	80.2%	85.4%	91.7%	Specificity
Standard	96.3%	96.3%	95.1%	
Drop	86.0%	91.6%	92.5%	
Step up	92.2%	94.8%	94.8%	
All	88.5%	92.0%	93.5%	
Ollie	50.0%	65.0%	76.5%	Precision
Standard	92.9%	93.2%	91.5%	
Drop	51.6%	66.7%	73.3%	
Step up	59.1%	72.7%	72.7%	
All	65.4%	75.9%	80.5%	
Ollie	0.339	0.510	0.578	Performance
Standard	0.709	0.745	0.768	
Drop	0.390	0.514	0.647	
Step up	0.481	0.667	0.667	
All	0.486	0.612	0.673	

The purpose of *Test* B and *Test C* is to observe if there is any discrepancy in ski and snowboarding jumps. When jump data of the ski subject is included in the training data, the performance for Ollies, Standard jumps, and Drops increases while that for Step up jumps remains the same. By definition in (3.28), performance is a combined evaluation of TP, FP, and FN. Improving either of these parameters results in an improvement on the performance of the classification result. From Table 3.5, both sensitivity and specificity for Ollies and Drops increased while those of Step up jumps are unchanged. For Standard jumps, even though the specificity went down, the

improvement on the sensitivity is considerable, resulting in a better performance value. In other words, ski jump data seems to improve the detection of Ollies Standard jumps, and Drops by adding distinctiveness to the Ollies and Drops. This indicates that the ski jump data has a positive effect on the classification result; meaning that the ski jumps and snowboarding jumps of these categories – Ollies, Standard jumps, Drops, Step up jumps - are very similar to each other and can be used to support the decision boundaries. By including the ski jumps into training data, the performance of Ollies and Drops were improved by 13.3% and 25.8%, respectively, and that of Standard jumps was improved by 3.0%.

Generally, the Standard jumps, which scored 82.7%, 95.1%, 91.5% and 0.768 on sensitivity, specificity, precision, and performance respectively, are more successfully classified than other classes of jumps. Table 3.6 is the confusion matrix of the jump classification result that manifests how the jumps are classified. From Figure 3.7, the Standard jumps data, which clusters in the middle area of quadrant one in the figure, can be easily separated from other jump data. This explains why Standard jumps scored highest in specificity, which is a measure of how distinctive it is from the other groups.

From the confusion matrix, the Ollies are sometimes misclassified into the Standard jumps, the Drops, and the Step up jumps. An Ollie is essentially a smaller version of a Standard jump. From Figure 3.7, it can be clearly seen that the Ollies data scatters to the bottom-left corner of the Standard jumps data. Since the subjects are different in their jump proficiency level, some subjects could complete Ollies as big as others' Standard jumps. In addition, some of the Ollies have similar height and drop values as that of the Drops and the Step up jumps when jumping on a slope. As a result, the Ollies are, occasionally, misclassified into other groups, which results in the least score, 0.578, in terms of the performance.

Table 3.6. Confusion matrix of the jump classification result

	Ollie	Standard	Drop	Step up
Ollie	26	3	3	5
Standard	3	43	5	1
Drop	3	1	22	0
Step up	2	0	0	16

The Drops and Step up jumps are distinctive to each other. None of the Drops are misclassified into Step up jumps and vise versa. Theoretically, the Drops will have a height value close to zero. On the other hand, the Step up jumps will have a drop value close to zero. As shown in Figure 3.7, the Drops data scatters along the y-axis, which represents a height value of zero, while the Step up jumps data scatters along the x-axis, which represents a drop value of zero. Therefore, drop and height features are good indicators for Drops and Step up jumps.

3.5. Conclusion

In activity classification, raw data is segmented into small windows and these windows are then passed into the classification model for activity classification. This paper presents a quasi real-time skiing and snowboarding jump classification algorithm. A single head-mounted integrated sensor, consisting of IMU, GPS, and barometric pressure sensor, is used to capture the jump windows with a threshold-based jump detection algorithm. After the jump window is selected, features are extracted from the window and are passed into an offline trained SVM decision model which classifies the jump into Ollies, Standard jumps, Drops, or Step up jumps.

The accuracy of the classification result is highly dependent on the amount of the training data, the consistency of the training data and the testing data, and the features selected for classification. Generally, height and drop defined in our work are good features to distinguish Standard jumps, Ollies, Drops, and Step up jumps. Nevertheless, different type of jumps may have similar height and/or drop value depending on where and who performs it. In addition to features such as height and drop, other time-domain features such as peaks value of vertical acceleration, maximum and minimum value of vertical velocity and tri-axle gyroscope data were introduced to the classification model to map the data points into higher dimensional feature space to better cluster the data together. By properly selecting the features and the parameters, the overall sensitivity, specificity, precision and performance were improved by 16.1%, 4.0%, 16.1%, and 25.9%, respectively.

The SVM classification model achieves an average of 80.5% precision. In other words, four out of five jumps can be accurately classified into their corresponding classes. Because only one ski test subject was recruited and the type of jumps performed by the test subjects were limited to Ollie, Standard jump, Drop, and Step up, it is difficult to generalize that all skiing jumps and snowboarding jumps behave similarly for classification purposes. For example, skiers and snowboarders perform 180^{o} rotation jumps differently. The dominant feature for 180^{o} jumps is the degree of yaw angle rotation of the sensor mounted on the helmet. For snowboarders, to achieve 180^{o} jumps, their heads do not have to rotate with the bodies. On the other hands, in order to achieve 180^{o} jumps in skiing, skiers rotate their heads along with the bodies. Therefore, in a follow up pilot study, when 180^{o} jumps of the skiers were included in the training data, the classification model misclassified the 180^{o} jumps of the snowboarders into other groups because the yaw degree of rotation is considerably small compares to that of the 180^{o} jumps of the skiers. In our future work, more ski test subject will be recruited and more categories of jumps should be included in the classification model. Also, since the characteristics of the jumps are different from person to person, to further improve the algorithm, on-line SVM learning can be implemented and the classification model can be customized to best fit the individual skier or the snowboarder.

Chapter 4.

Conclusion

The overall objective of this research has been to develop a jump detection and classification algorithm with a head-mounted integrated sensor, consisting of 9 axes IMU, barometric pressure sensor, and GPS. For this purpose, a novel threshold-based jump detection algorithm was proposed and the jumps were classified in a pre-trained SVM classifier. In this chapter, the obtained results and performance of the proposed algorithms are summarized, followed by recommendations on possible future work to further explore and advance the technology.

4.1. Objective and Algorithm Performance

A jump detection and classification algorithm was presented in this thesis. The two-stage algorithm can provide real-time or quasi real-time feedback when the individual performs a jump in skiing or snowboarding. First, the jumps were detected with a threshold-based detection method. The detected jumps were then classified into one of the following groups - Ollie, Standard jump, Drop, or Step up jump - with a SVM classifier. The work presented in this thesis was done in post processing, and further work is needed for online processing of the proposed algorithm.

4.1.1. Jump Detection

A threshold-based detection algorithm was developed, implemented, and tested. Vertical acceleration was the primary source to determine whether or not a jump occurs. Sensor fusion was applied by combining the barometric pressure data with IMU data to extract vertical acceleration, vertical velocity, and vertical position. Once the jump was detected, the difference between the maximum and minimum vertical velocity achieved

during the jump was evaluated. Thresholds were set on this velocity difference as well as the air time of the jump. By setting thresholds on vertical velocity and air time, the overall performance improved by more than 30% compared to acceleration alone. However, this threshold-based detection method depends highly on the threshold values. The values were selected with gird search method, and were found to be 1.1g, 3.1m/s, and 0.12s for vertical acceleration, difference in vertical velocity, and air time, respectively for our field experiments data. By properly select the threshold values, the algorithm showed 91.8% sensitivity with 93.0% specificity in our field experiments, which consist Ollies, Standard jumps, 180^{o} Standard jumps, Drops, and Step up jumps. Ollies, Standard jumps, and 180^{o} Standard jumps have higher success rate than Drops and Step up jumps.

4.1.2. Jump Classification

Jump classification was carried out with a multi-labeled SVM classifier. Features were extracted from the jump windows detected in the first stage of the proposed algorithm. The features extracted from the jump windows were height of the jump, drop of the jump, the total degree of rotation about z-axis during the jump, first and second vertical acceleration peaks of the jump, maximum and minimum vertical velocity of the jump, and the maximum and minimum gyroscope data. These features were passed into the SVM classifier and the jumps were classified into one of the following groups - Ollie, Standard jump, Drop, Step up jump. 180^{o} Standard jumps were not included in the groups because the characteristic features of these jumps performed by skiers and snowboarders were inconsistent. The accuracy of the classification result depends highly on the amount of the training data, the consistency of the training data and the testing data, and the features selected for classification. With the features stated above, 80.5% accuracy was achieved. Among these jumps, Standard jump achieved the highest precision, 91.5%, because other jumps rarely misclassified into this group. On the other hand, the precision of Drop and Step up jump, 73.3% and 72.7%, respectively, were considerably lower than that of the Standard jump, because some of the Ollies were misclassified into these groups due to similar characteristic features. Nevertheless, Drop and Step up jump had relatively higher sensitivity values than Standard jumps, 84.6% and 88.9%, respectively, because their features were more defined, meaning, a

Drop or a Step up jump have certain features that other jumps have not. For example, the drop value of a Drop is significantly greater than the height value of a Drop and the height value of a Step up jump is considerably greater than the drop value of a Step up jump.

4.1.3. Practicality of the Proposed Algorithm

The accuracy of the proposed jump detection/classification algorithm is satisfactory for entertainment purposes. Sensitivity measures how many jumps are detected out of all the jumps, while specificity measures how well the jumps are distinguished from all the falsely detected jumps. In the jump detection stage, for entertainment purpose, the specificity is more important than the sensitivity. The individuals could perform the jumps again and again were the jumps not detected. However, once a jump is detected, it should be an actual jump and nothing else, i.e. 100% specificity. Our algorithm can be tuned by increasing the threshold values on vertical acceleration, vertical velocity and air time, that it achieves 100% on specificity while sacrificing the accuracy on the sensitivity. The accuracy of the classification stage is not perfect but is still acceptable for entertainment purpose.

On the other hand, for competition purposes, the athletes only perform the jumps once. Therefore, sensitivity is more important than specificity. The algorithm must capture all the jumps during the competition, i.e. 100% on sensitivity. Also, all the jumps should be classified into the correct groups (100% accuracy on the classification). As a result, this algorithm is not suitable for competition yet. However, for competition, the professional skiers or snowboarders performs bigger and higher jumps. In other words, the thresholds can be set higher. In addition, the features of these jumps, such as height, drop, degree of ration, may be more distinguishable between different groups of jumps. Further study needs to be carried out to see how well the algorithm performs on professional skiers and snowboarders.

4.2. Future Recommendations

During this research, several challenges were encountered and some recommendations are made for future work.

Sensor mounting location

In future, different mounting location should be considered. The mounting location of the sensor is very important. The sensor captures the motion of the segment where it is mounted. In this project, the sensor is mounted on the helmet in order to simulate embedding the sensor in the ski goggles. Therefore, any head movement will affect the accuracy of the detection method. Also, the head movement cannot represent the body motion of the individuals. The better place for mounting the sensor is the torso. However, by mounting a sensor on the torso of the individuals may restrict the range of motion or make them feel uncomfortable. Another place to mount the sensor would be the snowboard or the ski. Mounting the sensor on the snowboard or the ski does not require the individuals to wear extra equipment. At the same time, the sensor captures the motion of the snowboard or the ski, which is tightly bound to their feet. The vertical height and the heading information will be more accurate if the sensor is mounted on the snowboard or the ski.

Online learning

Also, online learning can be integrated into the system in the future. The proposed jump detection method depends highly on the threshold values. Three of the thresholds introduced in the jump detection method require proper tuning. The primary source of detection signal is the vertical acceleration, vertical velocity, and the air time. Determining thresholds for these values is difficult, because these values vary from person to person and from jump to jump. The threshold values should be high enough, that any false alarm caused by dynamic noise will be removed. However, if the threshold is set too high, some of the jumps will not be detected because they are considered as false alarm. In addition, the accuracy of the classification results is also individual dependant. Some of the characteristic features of the jumps, such as height and drop, differ in values depends on how the jumps was carried out. Individuals may have

different proficiency in jump skills, and results in bigger or smaller jumps. All these defences affect the accuracy of the jump detection and classification algorithm. As a result, online learning is desired so that the threshold values and the SVM classifier best suit for the each individual. The classification accuracy depends on the number of the training data and the consistency between the training data and the testing data. Online learning not only allows the system to train itself with more data, but also more consistent data.

Application

This jump detection and classification algorithm is developed for winter sports such as skiing and snowboarding for, mainly, entertainment purpose. As a result, the accuracy of the KPVs is not too critical. However, if we want to apply this algorithm for training the athlete or for competition scoring, the accuracy of the KPVs, such as height, drop, air time, degree of rotation, etc, should be investigated. Higher accuracy is desired when the system is used for automatic scoring. In the next phase of this project, this algorithm can also be implemented for sports such as skateboarding, BMX biking, and rollerblading, where freestyle jumps are desired.

References

[1] International Olympic Committee. (2014). Ski jumping. [Online]. Available: http://www.olympic.org/ski-jumping.

[2] International Olympic Committee. (2014). Snowboard. [Online]. Available: http://www.olympic.org/snowbo-ard.

[3] J.W. Harding, K. Toohey, D.T. Martin, C. Mackintosh, A.M. Lindh, and D.A. James, “Automated inertial feedback for half-pipe snowboard competition and the community perception,” In: *The Impact of Technology on Sport II*, Taylor & Francis, 2007, pp. 845-850.

[4] G. Baker, “Are pedometers useful motivational tools for increasing walking in sedentary adults,” In: *Proc. 6th International Conference on Walking in the 21st Century*, Zurich, Switzerland, September 22–23, 2005, pp. 1–12.

[5] A. Nes, “Capturing the motion of ski jumpers using multiple stationary cameras,” *Proceedings of Norsk Informatikkonferanse*, 2005.

[6] P. Kannan, R. Ramakrishnan, “Development of human pose models for sports dynamics analysis using video image processing techniques,” *International Journal of Sports Science and Engineering*, vol. 6, no. 4, pp. 232–238, 2012.

[7] R. Poppe, “Vision-based human motion analysis: an overview,” *Computer Vision and Image Understanding*, vol. 108, no. 1-2, pp. 4-18, Oct. 2007.

[8] Harding JW, Small JW and James DA. “Feature extraction of performance variables in elite half-pipe snowboarding using body mounted inertial sensors,” In:

BioMEMS and Nanotechnology III. Proceedings of SPIE 2007, vol. 6799, pp. 679917–1-679917–12, 2007.

[9] Q. Yuan and I.-M. Chen. “Localization and velocity tracking of human via 3 IMU sensors,” *Sensors and Actuators A: Physical*, vol. 212, no. 1, pp. 25-33, 2014.

[10] P. Esser, H. Dawes, J. Collett, and K. Howells, “IMU: inertial sensing of vertical CoM movement,” *Journal of Biomechanics*, vol. 42, no. 10, pp. 1578-1582, 2009.

[11] M. Bächlin, M. Kusserow, G. Troster, and H. Gubelmann, “Ski jump analysis of an Olympic champion with wearable acceleration sensors,” in *2010 International Symposium on Wearable Computers (ISWC)*, 2010, pp. 1-2.

[12] Y. Ohgi, K. Seo, N. Hirai, and M. Murakami, “Measurement of jumper's body motion in ski jumping,” in *The Engineering of Sport 6*, E. F. Moritz and S. Haake, Eds. Springer New York, 2006, pp. 275-280.

[13] A. Krüger, and J. Edelmann-Nusser, “Biomechanical analysis in freestyle snowboarding: application of a full-body inertial measurement system and a bilateral insole measurement system,” *Sports Technology*, vol. 2 , no. 1–2 , pp. 17–23, Apr. 2009.

[14] S. Patel, H. Park, P. Bonato, L. Chan, and M. Rodgers, “A review of wearable sensors and systems with application in rehabilitation,” *Journal of NeuroEngineering and Rehabilitation*, vol. 9, p. 21, 2012.

[15] M. Brodie, A. Walmsley, and W. Page, “Fusion motion capture: a prototype system using inertial measurement units and GPS for the biomechanical analysis of ski racing,” *Sports Technol*ogy, vol. 1, no. 1, pp. 17-28, 2008.

[16] J.W. Harding, C.G. Mackintosh, D.T. Martin, A.G. Hahn, and D.A. James, “Automated scoring for elite half-pipe snowboard competition: important sporting

development or techno distraction," *Sports Technology*, vol. 1, no. 6, pp. 277-290, 2008.

[17] K. Hirose, H. Doki, and A. Kondo, "Dynamic motion analysis of snowboard turns by the measurement of motion and reaction force from snow surface," *Procedia Engineering*, vol. 34, pp. 754-759, 2012.

[18] F. Sadi and R. Klukas, "Reliable jump detection for snow sports with low-cost MEMS inertial sensors," *Sports Technology*, vol. 4, no. 1–2, pp. 88–105, 2011.

[19] R. Instruments, http://www.reconinstruments.com, 2015.

[20] RideOn, http://www.rideonvision.com/, 2015.

[21] GogglePal, http://gogglepal.com/, 2015.

[22] T.J. Lee, S. Zihajehzadeh, D. Loh, R. Hoskinson, and E.J. Park, "Automatic jump detection in skiing/snowboarding using head-mounted MEMS inertial and pressure sensors", *Proceedings of IMechE, Part P: Journal of Sports Engineering and Technology*, published online before print on 10 July 2015 (DOI: 10.1177/1754337115594501).

[23] T.J. Lee, R. Hoskinson, G. Mori, and E.J. Park, "Automatic jump classification in skiing/snowboarding using an integrated head-mounted MEMS sensor," submitted to *Proceedings of IMechE, Part P: Journal of Sports Engineering and Technology*.

[24] D. Roetenberg, "*Inertial and magnetic sensing of human motion,*" PhD Thesis, University of Twente, Netherlands, 2006.

[25] J.K. Lee, E.J. Park, and S.N. Robinovitch, "Estimation of attitude and external acceleration using inertial sensor measurement during various dynamic conditions," *IEEE Transactions on Instrumentation and Measurement*, vol. 61, no. 8, pp. 2262–2273, 2012.

[26] S. Zihajehzadeh, D. Loh, M. Lee, R. Hoskinson, and E.J. Park “A cascaded two-step Kalman filter for estimation of human body segment orientation using MEMS-IMU,” In: *Proceedings of IEEE Engineering In Medicine and Biology Society (EMBC)*, Chicago, IL, USA, August 2014, pp. 6270-6273.

[27] A.J. Ruiz, F.S. Granja, J.C. Prieto Honorato, and J.I.G Rosas “Accurate pedestrian indoor navigation by tightly coupling foot-mounted IMU and RFID measurements,” *IEEE Transactions on Instrumentation and Measurement*, vol. 61, no. 1, pp. 178–189, 2012.

[28] J.K. Lee, S.N. Robinovitch, and E.J. Park, “Inertial sensing-based pre-impact detection of falls involving near-fall scenarios,” *Transactions on Neural Systems and Rehabilitation Engineering,* vol. 23, no. 2, pp. 258–266, 2015.

[29] M. Tanigawa, H. Luinge, L. Schipper, P. Slycke, “Drift-free dynamic height sensor using MEMS IMU aided by MEMS pressure sensor,” In: *Proceedings of 5th Workshop on Positioning, Navigation and Communication (WPNC),* Hannover, Germany, 27 March 2008, pp. 191–196.

[30] M. Bevermeier, O. Walter, S. Peschke, R. Haeb-Umbach, “Barometric height estimation combined with map-matching in a loosely-coupled Kalman-filter,” In: *2010 7th Workshop on Positioning Navigation and Communication (WPNC)*, Dresden, Germany, 11-12 March 2010, pp. 128–134.

[31] J. Seo, J.G. Lee, and C.G. Park, “A new error compensation scheme for INS vertical channel,” In: *Proceedings of 16th IFAC Symposium on Automatic Control in Aerospace*, 2004, pp. 1119–1124.

[32] S. Zihajehzadeh, T.J. Lee, J.K. Lee, R. Hoskinson, E.J. Park, “Integration of MEMS inertial and pressure sensors for vertical trajectory determination,”

Transactions on Instrumentation and Measurement, vol. 64, no. 3, pp. 804-814, 2014.

[33] D.J. Simon, “Kalman filtering,” *Embedded System Programming*, vol. 14, no. 6, pp. 72–79, 2001.

[34] N.P. Linthorne, “Analysis of standing vertical jumps using a force platform,” *American Journal of Physics*, vol. 69, no. 11, pp. 1198-1204, 2001.

[35] Bishop CM. *Pattern Recognition and Machine Learning.* Springer, 2006.

[36] C.E. Metz, “Basic principles of ROC analysis,” *Seminars in Nuclear Medicine*, vol. 8, no. 4, pp. 283–298, 1978.

[37] S.J. Preece, J.Y. Goulermas, L.P. Kenney, D. Howard, K. Meijer, R. Crompton, “Activity identification using body-mounted sensors—a review of classification techniques,” *Physiological Measurement*, vol. 30, no. 4, pp. R1–R33, Apr. 2009.

[38] F.-C. Chuang, J.-S. Wang, Y.-T. Yang, and T-P Kao, “A wearable activity sensor system and its physical activity classification scheme,” in *The 2012 International Joint Conference on Neural Network (IJCNN)*, 2012, pp. 1-6.

[39] A. Sorvala, E. Alasaarela, H. Sorvoja, and R. Myllyla, “Activity classification using a state transition diagram and activity levels,” in *2012 6th International Symposium on Medical Information and Communication Technology (ISMICT)*, 2012, pp. 1-4.

[40] Y. Kim and H. Ling, “Human activity classification based on micro-Doppler signatures using a support vector machine,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 47, no. 5, pp. 1328-1337, May 2009.

[41] S. Chernbumroong, A.S. Atkins, and H. Yu, “Activity classification using a single wrist-worn accelerometer,” in *2011 5th International Conference on Software, Knowledge, Information Management and Applications (SKIMA)*, 2011, pp. 1-6.

[42] G. Panahandeh, N. Mohammadiha, A. Leijon, and P. Handel, “Continuous hidden markov model for pedestrian activity classification and gait analysis,” *IEEE Transactions on Instrumentation and Measurement*, vol. 62, no. 5, pp. 1073-1083, May 2013.

[43] B. Logan and J. Healey, “Sensor to detect the activities of daily living,” *Conference Proceedings of IEEE Engineering in Medicine and Biology Society*, vol. 1, pp. 5362-5365, 2006.

[44] E. Guenterberg, S. Ostadabbas, H. Ghasemzadeh, and R. Jafari, “An automatic segmentation technique in body sensor networks based on signal energy,” in *Proceedings of the Fourth International Conference on Body Area Network*, ICST, Brussels, Belgium, 2009, pp. 21:1-21:7.

[45] M. N. Nyan, F. E. H. Tay, K. H. W. Seah, and Y.Y. Sitoh, “Classification of gait patterns in the time-frequency domain,” *Journal of Biomechanics*, vol. 39, no. 14, pp. 2647-2656, 2006.

[46] I. Guyon and A. Elisseeff, “An introduction to variable and feature selection,” *Journal of Machine Learning Research*, vol. 3, pp. 1157-1182, Mar. 2003.

[47] O. Chapelle, and S. Keerthi, “Multi-class feature selection with support vector machines,” in *Proceedings of the American Statistical Association*, 2008.

[48] G. Doquire and M. Verleysen, “Feature selection for multi-label classification problems,” in *Advances in Computational Intelligence*, J. Cabestany, I. Rojas, and G. Joya, Eds. Springer Berlin Heidelberg, 2011, pp. 9-16.

[49] C.P. Diehl and G. Cauwenberghs, “SVM incremental learning, adaptation and optimization,” in *Proceedings of the International Joint Conference on Neural Networks*, vol. 4, 2003, pp. 2685-2690.

[50] R. Sangeetha and B. Kalpana, “Performance evaluation of kernels in multiclass SVM,” *International Journal of Soft Computing and Engineering (IJSCE)*, vol. 1, no. 5, pp. 138-143, 2011.

[51] C.-W. Hsu and C.-J. Lin, “A comparison of methods for multiclass support vector machines,” *IEEE Transactions on Neural Networks*, vol. 13, no. 2, pp. 415-425, Mar. 2002.

[52] G.M. Foody and A. Mathur, “A relative evaluation of multiclass image classification by support vector machines,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 42, no. 6, pp. 1335-1343, Jun. 2004.

[53] T. Fawcett, “An introduction to ROC analysis,” *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861-874, Jun. 2006.

[54] J. Davis and M. Goadrich, “The Relationship Between Precision-Recall and ROC Curves,” in *Proceedings of the 23rd International Conference on Machine Learning*, New York, NY, USA, 2006, pp. 233-240.

[55] C.-C. Chang and C.-J. Lin, LIBSVM: a library for support vector machines. ACM Transactions on Intelligent Systems and Technology, 2:27:1--27:27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm

Appendix A.

Cascaded Orientation Kalman Filters

Two Kalman filters are used in finding the orientation: a) Tilt Kalman filter, and b) Yaw Kalman filter [32]. These two Kalman filters are used to orient the sensor frame (S) to the inertial frame (I) so the data acquired from the IMU sensor, the barometric pressure sensor, and GPS are in the same inertial frame. The orientation of the sensor frame with respect to the inertial frame can be represented as

$$^{\mathrm{I}}\mathbf{x} = {}^{\mathrm{I}}_{\mathrm{S}}\mathbf{R}\ {}^{\mathrm{S}}\mathbf{x} \tag{A.1}$$

where $\mathbf{x}$ is an arbitrary 3 x 1 vector being rotated and ${}^{\mathrm{I}}_{\mathrm{S}}\mathbf{R}$ is the 3 x 3 rotation

$${}^{\mathrm{I}}_{\mathrm{S}}\mathbf{R} = \begin{bmatrix} c\alpha c\beta & c\alpha s\beta s\gamma - s\alpha c\gamma & c\alpha s\beta c\gamma + s\alpha s\gamma \\ s\alpha c\beta & d\alpha s\beta s + c\alpha c\gamma & s\alpha s\beta c\gamma - c\alpha s\gamma \\ -s\beta & c\beta s\gamma & c\beta c\gamma \end{bmatrix} \tag{A.2}$$

where c and s are short for cos and sin respectively. α (yaw), β (pitch), and γ (roll) are rotation about z-axis, y-axis, and x-axis respectively.

Tilt Kalman filter

The *Tilt Kalman filter* uses the following system model equations:

$$\mathbf{x}_1(\mathrm{k}) = \mathbf{A}_1(\mathrm{k}-1)\mathbf{x}_1(\mathrm{k}-1) + \mathbf{w}_1(\mathrm{k}-1) \tag{A.3}$$

$$\mathbf{z}_1(\mathrm{k}) = \mathbf{C}_1(\mathrm{k})\mathbf{x}_1(\mathrm{k}) + \mathbf{v}_1(\mathrm{k}) \tag{A.4}$$

where $\mathbf{x}_1(\mathrm{k}) = [-s\beta \quad c\beta s\gamma \quad c\beta c\gamma]^{\mathrm{T}}$ is the state vector and $\mathbf{z}_1(\mathrm{k})$ is the measurement vector for the *Tilt Kalman filter* at step k. $\mathbf{A}_1$ is the state transition matrix. $\mathbf{C}_1$ is the observation matrix. $\mathbf{w}_1$ and $\mathbf{v}_1$ are process model noise vector and measurement model noise vector respectively. These matrices and vectors can be calculated with the following equations:

$$\mathbf{A}_1(\mathrm{k}-1) = \mathbf{I}_3 - \Delta\mathrm{t}\tilde{\mathbf{y}}_{\mathrm{G}}(\mathrm{k}-1) \tag{A.5}$$

$$\mathbf{w}_1(\mathrm{k}-1) = \Delta\mathrm{t}\left(-\tilde{\mathbf{x}}_1(\mathrm{k})\right)\mathbf{n}_{\mathrm{G}} \tag{A.6}$$

$$\mathbf{C}_1(\mathrm{k}) = \mathrm{g}\mathbf{I}_3 \tag{A.7}$$

$$\mathbf{v}_1(\mathrm{k}) = -\,{}^{\mathrm{S}}\mathbf{a}_{\varepsilon}^{-}(\mathrm{k}) + \mathbf{n}_{\mathrm{A}} \tag{A.8}$$

$${}^{\mathrm{S}}\mathbf{a}_{\varepsilon}^{-}(\mathrm{k}) = {}^{\mathrm{S}}\mathbf{a}^{-}(\mathrm{k}) - {}^{\mathrm{S}}\mathbf{a}(\mathrm{k}) \tag{A.9}$$

$${}^{\mathrm{S}}\mathbf{a}^{-}(\mathrm{k}) = c_a\ {}^{\mathrm{S}}\mathbf{a}^{+}(\mathrm{k}-1) \tag{A.10}$$

where $\tilde{\mathbf{y}}_{\mathrm{G}}$ is the 3 x 3 skew-symmetric matrix of tri-axial gyroscope measurements and $\tilde{\mathbf{x}}_1$ is the skew-symmetric matrix of $\mathbf{x}_1$. $\mathbf{I}_3$ is the 3 x 3 identity matrix. ${}^{\mathrm{S}}\mathbf{a}_{\varepsilon}^{-}$ is the external acceleration error in sensor frame and ${}^{\mathrm{S}}\mathbf{a}$ is the acceleration measurement from the

accelerometer. The superscripts + and − represents the *a posteriori* and *a priori* estimates in the Kalman filter respectively. $\mathbf{n}_{\mathrm{G}}$ and $\mathbf{n}_{\mathrm{A}}$ are gyroscope and accelerometer measurement noise and are assumed to be uncorrelated and zero-mean white Gaussian. $0 \leq c_a \leq 1$ is a dimensionless constant and g is the gravity constant. Once all these matrices are calculated, the Kalman filter is process with the following steps:

Step 1: compute the *a priori* state estimate:

$$\mathbf{x}_1^-(\mathrm{k}) = \mathbf{A}_1(\mathrm{k}-1)\mathbf{x}_1^+(\mathrm{k}-1) \tag{A.11}$$

Step 2: compute the *a priori* error covariance matrix:

$$\mathbf{P}^-(\mathrm{k}) = \mathbf{A}_1(\mathrm{k}-1)\mathbf{P}^+(\mathrm{k}-1)\mathbf{A}_1^{\mathrm{T}}(\mathrm{k}-1) + \mathbf{Q}_1(\mathrm{k}-1) \tag{A.12}$$

where $\mathbf{Q}_1(\mathrm{k}) = \mathrm{E}[\mathbf{w}_1(\mathrm{k})\mathbf{w}_1^{\mathrm{T}}(\mathrm{k})]$ is the process noise covariance matrix and $\mathrm{E}[\ \bullet\ ]$ denotes the expected value of the vector within the bracket.

Step 3: compute the Kalman gain:

$$\mathbf{K}(\mathrm{k}) = \mathbf{P}^-(\mathrm{k})\mathbf{C}_1^{\mathrm{T}}(\mathrm{k})[\mathbf{C}_1(\mathrm{k})\mathbf{P}^-(\mathrm{k})\mathbf{C}_1^{\mathrm{T}}(\mathrm{k}) + \mathbf{R}_1(\mathrm{k})]^{-1} \tag{A.13}$$

where $\mathbf{R}_1(\mathrm{k}) = \mathrm{E}[\mathbf{v}_1(\mathrm{k})\mathbf{v}_1^{\mathrm{T}}(\mathrm{k})]$ is the measurement noise covariance matrix.

Step 4: compute the *a posteriori* state estimate:

$$\mathbf{x}_1^+(\mathrm{k}) = \mathbf{x}_1^-(\mathrm{k}) + \mathbf{K}(\mathrm{k})[\mathbf{z}_1(\mathrm{k}) - \mathbf{C}_1(\mathrm{k})\mathbf{x}_1^-(\mathrm{k})] \tag{A.14}$$

Step 5: compute the *a posteriori* error covariance matrix:

$$\mathbf{P}^+(\mathrm{k}) = [\mathbf{I}_3 - \mathbf{K}(\mathrm{k})\mathbf{C}_1(\mathrm{k})]\mathbf{P}^-(\mathrm{k}) \tag{A.15}$$

After the *a posteriori* state estimate, $\mathbf{x}_1^+(\mathrm{k}) = [\mathrm{x}_{1,\mathrm{x}} \quad \mathrm{x}_{1,\mathrm{y}} \quad \mathrm{x}_{1,\mathrm{z}}]^{\mathrm{T}}$ is acquired, the roll and the pitch can be found with the following equations:

$$\gamma(\mathrm{k}) = \tan^{-1}\frac{\mathrm{x}_{1,\mathrm{y}}}{\mathrm{x}_{1,\mathrm{z}}} \tag{A.16}$$

$$\beta(\mathrm{k}) = \tan^{-1}\frac{-\mathrm{x}_{1,\mathrm{x}}}{\mathrm{x}_{1,\mathrm{y}}/s\gamma} \tag{A.17}$$

This two angles are used as input to the *Yaw Kalman filter*.

Yaw Kalman filter

With the roll and pitch values available from the previous Kalman filter, the *Yaw Kalman filter* uses the following system mmodel equations:

$$\mathbf{x}_2(\mathrm{k}) = \mathbf{A}_2(\mathrm{k}-1)\mathbf{x}_2(\mathrm{k}-1) + \mathbf{w}_2(\mathrm{k}-1) \tag{A.18}$$

$$\mathbf{z}_2(\mathrm{k}) = \mathbf{C}_2(\mathrm{k})\mathbf{x}_2(\mathrm{k}) + \mathbf{v}_2(\mathrm{k}) \tag{A.19}$$

where $\mathbf{x}_2(\mathrm{k}) = [c\alpha c\beta \quad c\alpha s\beta s\gamma - s\alpha c\gamma \quad c\alpha s\beta c\gamma + s\alpha s\gamma]^{\mathrm{T}}$ is the state vector at step k and $\mathbf{A}_2$, $\mathbf{w}_2$, $\mathbf{C}_2$, and $\mathbf{v}_2$ can be calculated as

$$\mathbf{A}_2(\mathrm{k}-1) = \mathbf{I}_3 - \Delta\mathrm{t}\tilde{\mathbf{y}}_{\mathrm{G}}(\mathrm{k}-1) \tag{A.20}$$

$$\mathbf{w}_2(\mathrm{k}-1) = \Delta\mathrm{t}\big(-\tilde{\mathbf{x}}_2(\mathrm{k})\big)\mathbf{n}_{\mathrm{G}} \tag{A.21}$$

$$\mathbf{C}_2(\mathrm{k}) = \mathbf{I}_3 \tag{A.22}$$

$$\mathbf{v}_2(\mathrm{k}) = \mathbf{n}_{\mathrm{M}} \tag{A.23}$$

where $\mathbf{n}_{\mathrm{M}}$ is the magnetometer noise and is assumed to be a white noise.

In order to calculate the measurement vector, $\mathbf{z}_2$, magnetometer data, $\mathbf{y}_{\mathrm{M}}$, is first rotated with the following equation:

$$^{\mathrm{tilt}}\mathbf{y}_{\mathrm{M}} = {}^{\mathrm{I}}_{\mathrm{S}}\mathbf{R}_{\gamma,\beta}\mathbf{y}_{\mathrm{M}} \tag{A.24}$$

where ${}^{\mathrm{I}}_{\mathrm{S}}\mathbf{R}_{\gamma,\beta} = \begin{bmatrix} c\beta & 0 & s\beta \\ 0 & 1 & 0 \\ -s\beta & 0 & c\beta \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 \\ 0 & c\gamma & -s\gamma \\ 0 & s\gamma & c\gamma \end{bmatrix}$. After that, let $^{\mathrm{tilt}}\mathbf{y}_{\mathrm{M}} = [\mathrm{y}_{\mathrm{M,x}} \quad \mathrm{y}_{\mathrm{M,y}} \quad \mathrm{y}_{\mathrm{M,z}}]$, α can be calculated.

$$\alpha = \tan^{-1}\frac{y_{M,y}}{y_{M,x}} \tag{A.25}$$

Once all these matrices are calculated, the *Yaw Kalman filter* is calculated with the following steps:

Step 1: compute the *a priori* state estimate:

$$\mathbf{x}_2^-(\mathrm{k}) = \mathbf{A}_2(\mathrm{k}-1)\mathbf{x}_2^+(\mathrm{k}-1) \tag{A.26}$$

Step 2: compute the *a priori* error covariance matrix:

$$\mathbf{P}^-(\mathrm{k}) = \mathbf{A}_2(\mathrm{k}-1)\mathbf{P}^+(\mathrm{k}-1)\mathbf{A}_2^{\mathrm{T}}(\mathrm{k}-1) + \mathbf{Q}_2(\mathrm{k}-1) \tag{A.27}$$

where $\mathbf{Q}_2(\mathrm{k}) = \mathrm{E}[\mathbf{w}_2(\mathrm{k})\mathbf{w}_2^{\mathrm{T}}(\mathrm{k})]$.

Step 3: compute the Kalman gain:

$$\mathbf{K}(\mathrm{k}) = \mathbf{P}^-(\mathrm{k})\mathbf{C}_2^{\mathrm{T}}(\mathrm{k})[\mathbf{C}_2(\mathrm{k})\mathbf{P}^-(\mathrm{k})\mathbf{C}_2^{\mathrm{T}}(\mathrm{k}) + \mathbf{R}_2(\mathrm{k})]^{-1} \tag{A.28}$$

where $\mathbf{R}_2(\mathrm{k})$ is the measurement noise covariance matrix. Here, conditions are set on $\mathbf{R}_2(\mathrm{k})$ to filter out the potentially disturbed magnetic field data. If the norm of the magnetometer data is greater than the local earth's magnetic field, h, by a threshold value, ε_{M}, a large number is assigned to $\mathbf{R}_2(\mathrm{k})$ so the filter excludes the potentially disturbed data and rely less on magnetometer measurement.

$$\mathbf{R}_2(\mathrm{k}) = \begin{cases} \mathrm{E}[\mathbf{v}_2(\mathrm{k})\mathbf{v}_2^{\mathrm{T}}(\mathrm{k})] & \|\mathbf{y}_{\mathrm{M}} - h\| < \varepsilon_{\mathrm{M}} \\ \infty & otherwise \end{cases} \tag{A.29}$$

Step 4: compute the *a posteriori* state estimate:

$$\mathbf{x}_2^+(\mathrm{k}) = \mathbf{x}_2^-(\mathrm{k}) + \mathbf{K}(\mathrm{k})[\mathbf{z}_2(\mathrm{k}) - \mathbf{C}_2(\mathrm{k})\mathbf{x}_2^-(\mathrm{k})] \tag{A.30}$$

Step 5: compute the *a posteriori* error covariance matrix:

$$\mathbf{P}^+(\mathrm{k}) = [\mathbf{I}_3 - \mathbf{K}(\mathrm{k})\mathbf{C}_2(\mathrm{k})]\mathbf{P}^-(\mathrm{k}) \tag{A.31}$$

With the *a posteriori* state estimate, $\mathbf{x}_2^+(\mathrm{k}) = [\mathrm{x}_{1,\mathrm{x}} \quad \mathrm{x}_{1,\mathrm{y}} \quad \mathrm{x}_{1,\mathrm{z}}]^{\mathrm{T}}$, the yaw angle can be calculated:

$$\alpha(\mathrm{k}) = \tan^{-1}\frac{-c\gamma \mathrm{x}_{2,\mathrm{y}} + s\gamma \mathrm{x}_{2,\mathrm{z}}}{\mathrm{x}_{2,\mathrm{x}}/c\beta} \tag{A.32}$$

After the roll, pitch, and yaw are calculated, these angles are substituted into (A.2) and the rotation matrix is derived.

Appendix B.

Ethics Approval

[redacted] OFFICE OF RESEARCH ETHICS

Street Address
Simon Fraser University
Discovery 2
Room 230, 8900 Nelson Way
Burnaby, BC Canada V5A 4W9

Mailing Address
8888 University Drive
Discovery 2
Burnaby, BC Canada
V5A 1S6

[redacted]
dore@sfu.ca
http://www.sfu.ca/ore

Annual Renewal Approval

Study Number: 2013s0790

Study Title: Advanced body motion and location tracking in active sports goggles

Annual Renewal Date: Nov. 21, 2014 **Expiry Date:** Nov. 21, 2015

Principal Investigator: Park, Edward **Supervisor:** n/a

SFU Position: Faculty **Faculty/Department:** School of Mechatronic Systems Engineering

Co-Investigators/Research Personnel: Zihajehzadeh, Shaghayegh; Hoskinson, Reynald; Lee, Matthew; Loh, Darrel; Musngi, Magnus; and Yoon, Paul

Funding Source 1: NSERC
Grant Title 1: Advanced body motion and location tracking in active sports goggles

Approved in this Application:

- Annual Renewal/Progress Report Form
- Annual Renewal Approval Expiry Date to Nov. 21, 2015

The approval for the above-referenced study expires on the **Expiry Date**. **Failure to submit an annual renewal form will lead to your study being suspended and potentially terminated.** If you intend to continue your protocol to collect data past the term of approval, you must submit an annual renewal/progress report at least 4 weeks before the expiry date at dore@sfu.ca.

Please notify the Office of Research Ethics at dore@sfu.ca once you have completed the data collection portion of your project so that we can close the file.

This Notification of Status is your official Annual Renewal Approval documentation for this project. Please keep this document for reference purposes.

Sincerely,

[redacted]

Dina Shafey, MBA, PhD
Acting Associate Director,
Office of Research Ethics