

# Kinect Open Source Programming Secrets

Hacking the Kinect with OpenNI, NITE, and Java



Andrew Davison

# Kinect Open Source Programming Secrets

Hacking the Kinect  
with OpenNI, NITE, and



Java 章

Andrew Davison



New York Chicago San Francisco Lisbon  
London Madrid Mexico City Milan New Delhi  
San Juan Seoul Singapore Sydney Toronto

**Cataloging-in-Publication Data is on file with the Library of Congress**

McGraw-Hill books are available at special quantity discounts to use as premiums and sales promotions, or for use in corporate training programs. To contact a representative, please e-mail us at [bulksales@mcgraw-hill.com](mailto:bulksales@mcgraw-hill.com).

## **Kinect Open Source Programming Secrets: Hacking the Kinect with OpenNI, NITE, and Java**

Copyright © 2012 by The McGraw-Hill Companies. All rights reserved. Printed in the United States of America. Except as permitted under the Copyright Act of 1976, no part of this publication may be reproduced or distributed in any form or by any means, or stored in a database or retrieval system, without the prior written permission of publisher, with the exception that the program listings may be entered, stored, and executed in a computer system, but they may not be reproduced for publication.

McGraw-Hill, the McGraw-Hill Publishing logo, TAB™, and related trade dress are trademarks or registered trademarks of The McGraw-Hill Companies and/or its affiliates in the United States and other countries and may not be used without written permission. All other trademarks are the property of their respective owners. The McGraw-Hill Companies is not associated with any product or vendor mentioned in this book.

12 34567890 DOC DOC 1098765432

ISBN 978-0-07-178317-0

MHID 0-07-178317-2

### **Sponsoring Editor**

Roger Stewart

### **Editorial Supervisor**

Patty Mon

### **Project Manager**

Rohini Deb,  
Cenveo Publisher Services

### **Acquisitions Coordinator**

Molly Wyand

### **Copy Editor**

Marilyn Smith

### **Proofreader**

Claire Splan

### **Indexer**

Karin Arrigoni

### **Production Supervisor**

George Anderson

### **Composition**

Cenveo Publisher Services

### **Illustration**

Cenveo Publisher Services

### **Art Director, Cover**

Jeff Weeks

### **Cover Designer**

Jeff Weeks

Information has been obtained by McGraw-Hill from sources believed to be reliable. However, because of the possibility of human or mechanical error by our sources, McGraw-Hill, or others, McGraw-Hill does not guarantee the accuracy, adequacy, or completeness of any information and is not responsible for any errors or omissions or the results obtained from the use of such information.

# **Kinect Open Source Programming Secrets**

## About the Author

**Andrew Davison** (Hat Yai, Thailand) received his Ph.D. from Imperial College in London, and was a lecturer at the University of Melbourne for six years, before moving to Prince of Songkla University in Thailand. He has also taught in Bangkok, Khon Kaen, and Hanoi. His research interests include scripting languages, logic programming, visualization, and teaching methodologies. These led to an interest in teaching game programming and natural user interfaces. He is the author of two game programming books: *Killer Game Programming in Java* (O'Reilly, 2005) and *Pro Java 6 3D Game Development* (Apress, 2007). He is also the coauthor with Carol Hamer of *Learn BlackBerry Games Development* (Apress, 2010).

# Acknowledgments

As always, I thank my wife, Supatra, and son, John, who have again been a source of strength.

I must acknowledge my department and faculty members at Prince of Songkla University in Thailand, who have given me the time and resources for completing this work. I heartily recommend Thailand as a wonderful place to live.

Finally, regards to Roger Stewart, Patty Mon, Rohini Deb, Marilyn Smith, Claire Splan, and the rest of the team at McGraw-Hill and Cengage Publisher Services, who helped me focus on the important things.

# Introduction

It was at my university's 2011 Open Day that I realized the importance of the Kinect sensor. Returning from lunch, I saw a very long line of people waiting to get into one of our computer labs. A lab of PCs is nothing special to today's computer-savvy kids, so I walked over to see what the fuss was about. People wanted to dance in a virtual disco, boogying on down in front of a Kinect.

To my mind, the Kinect has achieved two remarkable things. One is that it introduced the wonders of computer vision to a public waiting for the next Big Thing. Also, the Kinect is the first consumer-level (affordable) device that makes it comparatively easy to program with motion sensing, skeletal tracking, and natural user interfaces using gestures. That's why the Kinect has caught the imaginations of the hacker/developer community, won a slew of awards for best gadget and innovative engineering, and been awarded a Guinness World Record for the fastest selling consumer electronics device.

But notice that I said "comparatively easy to program." It's fair to say that there has been a rather dismal lack of documentation on the Kinect's APIs. This book has grown out of my online tutorials (at <http://fivedots.coe.psu.ac.th/~ad/kinect>), which aim to remedy that problem.

## Why Buy This Book?

I can suggest four reasons for buying this book:

- It offers a unique choice of Kinect programming tools.
- It explains the official Java wrappers for those tools.
- It covers topics not found elsewhere.
- It provides depth, but with brevity.

## Unique Programming Tools

This is the only book on programming the Kinect using the OpenNI library, NITE, and Java (as of April 2012, when this book went to press). Of course, this claim won't mean much if you don't know OpenNI or NITE, so let me briefly describe them.

OpenNI (or more correctly, the OpenNI framework) is open source software for accessing the Kinect's sensors. OpenNI has been ported to multiple platforms, such as Windows, Mac OS X, and Linux. It was developed by PrimeSense, the company that worked with Microsoft on building the Kinect. NITE adds higher-level features to OpenNI, including user tracking and hand gestures. It was also developed by PrimeSense, and although it isn't open source, it is available for free. OpenNI and NITE have been around since the early days of the Kinect, and a vast number of fun projects use their APIs. The OpenNI and NITE developer forums are very active, providing a lot of help.

## Official Java Wrappers

This is the only book that explains the official Java wrappers for OpenNI and NITE (again, as of April 2012). By "official," I mean that these bindings were developed by PrimeSense. Obvious advantages of Java include object-orientation, cross-platform support, availability for free, and many people (including you, probably) knowing how to program with it. Most important, programming in Java gives you access to a massive number of libraries—for graphics, networking, and beyond—that can be linked to the Kinect without much effort. For example, I'll demonstrate how to use the Java 3D graphics library and the Java binding for the OpenCV computer vision package.

Some Kinect books employ the Processing language, a simplified version of Java aimed at new programmers interested in generating graphics. Processing has many interesting libraries, but nothing like Java's vast collections.

There are other Java bindings for OpenNI, created by third-party developers. These APIs tend to have limited functionality and are not consistently maintained.

The main drawback of using the PrimeSense Java wrappers is their lack of documentation. As I explain in Chapter 1, I had to decompile the libraries' JAR files, and work out the correspondences between the Java source and the somewhat better documented C++ OpenNI/NITE APIs. (This is why including *Secrets* in the book's title isn't too excessive.)

## A Wide Range of Topics

This book covers programming topics not found elsewhere. I start off with the basics, of course, with chapters on depth, infrared, and RGB imaging, point clouds, skeletal user tracking, hand tracking, and gesture support. Moving beyond that, I cover several novel and unusual features, including the following:

- Kinect gaming based around a version of the classic Breakout video game.
- Controls for the Kinect motor, LED, and accelerometer, which are not part of the standard OpenNI API. In fact, their absence is often held up as a serious drawback of the API. It's actually quite easy to add these capabilities using a custom-built USB driver.



- 3D graphics programming in the point cloud and skeletal tracking examples, using Java 3D.
- A computer vision example that demonstrates how to link the Kinect to the popular (and powerful) OpenCV library.
- The creation of new body gestures (inspired by the FFAST system), which are not part of the limited NITE repertoire.
- A new type of GUI component controlled by hand gesturing, illustrated with three examples: a button, dial, and slider. These components are controlled without the help of mouse or keyboard.

## Depth with Brevity

This book describes a lot of complicated code but, unlike some rather hefty programming tomes, you won't find all the code tediously printed on these pages. Instead, you can download it from the book's website at <http://fivedots.coe.psu.ac.th/~ad/kinect/>. In addition, I've been adding supplementary chapters to the website, including ones discussing speech recognition and the Kinect microphone array.

I've also cut down on the page count by assuming that you have some Java programming experience (perhaps you've taken a one-semester course on Java at college). That means that I don't explain how to create a JFrame or search for substrings. I don't describe every line of every program, but concentrate on the important Kinect-specific material.

## What This Book Is Not About

The fact that I'm powering ahead without wasting time explaining about classes, objects, and inheritance could be viewed as a disadvantage. This is not a book for first-time Java programmers.

Also, I don't have the space to seriously explain the topics of 3D graphics or computer vision. I introduce them (in the form of Java 3D and JavaCV) in enough detail so that you can understand my examples, but there is a lot of material that I don't mention. When I get to these topics, I'll suggest resources for finding more information.

# Contents at a Glance

<b>1</b>	Getting Started .....	1
<b>2</b>	Kinect Imaging .....	19
<b>3</b>	A Point Cloud for Depths .....	47
<b>4</b>	Tracking Users in 2D .....	69
<b>5</b>	Viewing Users in 3D .....	93
<b>6</b>	The Tilt Motor, LED, and Accelerometer .....	133
<b>7</b>	NITE Hand Gestures .....	155
<b>8</b>	NITE HandsTracker .....	179
<b>9</b>	Kinect Breakout .....	195
<b>10</b>	Gesture GUIs .....	223
<b>11</b>	Kinect Capture .....	251
<b>11</b>	Motion Detection Using OpenCV .....	261
<b>13</b>	FAAST-Style Body Gestures .....	287
	Index .....	309

# Contents

Acknowledgment .....	xiii
Introduction .....	xv
<b>CHAPTER 1 Getting Started .....</b>	<b>1</b>
The Kinect Sensor Hardware .....	1
Kinect Development Software .....	2
The 800-Pound Gorilla Enters .....	4
Programming the Kinect in Java .....	4
Kinect Software Installation .....	5
Cleaning Up First .....	5
Downloading the Packages .....	6
Installing and Configuring the Software .....	7
Fixing the XML Configuration Files .....	7
Testing OpenNI and NITE .....	9
Documentation and Other Sources of Help .....	11
OpenNI, NITE, and Kinect Documentation .....	11
Java Programming Documentation .....	12
OpenNI Concepts in Java .....	13
Generating Data and Metadata .....	14
Listeners .....	14
Capabilities .....	15
The Simplest Java OpenNI Example .....	16
<b>CHAPTER 2 Kinect Imaging .....</b>	<b>19</b>
An Overview of the OpenNIView Application Versions .....	19
Version 1: Grayscale Depth Map and XML .....	21
Updating the Panel .....	23
From Depth Values to Image Pixels .....	25
Rendering the Depth Map .....	27
Version 2: Grayscale Depth Map Without XML .....	28
Version 3: Camera Image Map .....	29
Updating the Image .....	31
Drawing the Image .....	33

Version 4: Infrared Map .....	34
Updating the Image .....	35
Creating a Grayscale Image .....	36
Version 5: A Colored Depth Map .....	37
Creating a Color Model .....	38
Creating the Depth Map .....	39
Rendering the Image .....	40
Version 6: Combined Depth and Image Maps .....	41
Creating a Color Model .....	42
Initializing the Context .....	43
Updating the Panel .....	44
Rendering the Images .....	45
<b>CHAPTER 3 A Point Cloud for Depths .....</b>	<b>47</b>
An Overview of the PointCloud Application .....	47
What Is Java 3D? .....	50
The HelloUniverse Scene Graph .....	51
Java 3D Documentation and Examples .....	53
Creating the 3D Scene for PointCloud .....	54
Lighting the Scene .....	55
Setting the Scene's Background .....	57
Positioning the Point Cloud .....	57
Viewer Positioning .....	58
Viewer Movement .....	59
Point Clouds in Java 3D .....	60
Initializing the PointArray .....	60
Updating the Points .....	64
Obtaining the Depth Map .....	66
<b>CHAPTER 4 Tracking Users in 2D .....</b>	<b>69</b>
An Overview of the GorillasTracker Application .....	69
Configuring the Tracker .....	72
Updating the Scene .....	73
Drawing the Scene .....	75
Managing Skeletons .....	76
Configuring the User Generator .....	77
Listening for Users .....	79
Using Other Listeners .....	82
Updating the Skeletons .....	83
A More Complex Skeleton .....	86
Drawing the Skeletons .....	87
Drawing a Skeleton .....	88
Drawing a Head Image .....	90
Displaying User Status Information .....	91

<b>CHAPTER 5 Viewing Users in 3D</b>	<b>93</b>
An Overview of the UsersViewer3D Application	93
Configuring the Tracker	96
Managing the Skeletons	98
Configuring the UserGenerator	98
Setting Up the Observers	100
Updating the Skeletons	104
Bringing a Skeleton to Life	105
Creating a Skeleton Scene Graph	105
Building Joints	109
Building Limbs	111
Updating the Skeleton	113
Modifying a Skeleton's Visibility	113
Deleting the Skeleton	114
Creating a 3D Joint	114
Creating the Joint Subgraph	114
Updating the Joint's Position	118
Smoothing a Position	120
Handling Joint Visibility	121
Creating a 3D Limb	121
Creating the Limb Subgraph	122
Updating the Limb	125
Adjusting Limb Position and Length	127
Changing the Limb's Orientation	129
An Interpolation and Cross-Product Problem	131
<b>CHAPTER 6 The Tilt Motor, LED, and Accelerometer</b>	<b>133</b>
Controlling the Kinect's Motor, LED, and Accelerometer	133
A Quick Introduction to USB	134
Becoming a USB Detective	136
Installing USB Support for Java	137
Installing libusb-win32	138
Installing a libusb-win32 Driver for the Kinect Motor	138
Installing and Testing LibusbJava	140
Using a Modified LibusbJava	141
Motor Protocol Discovery	142
Setting the Tilt Motor, LED, and Accelerometer	143
Setting the LED Status Light	146
Tilting the Kinect	147
Reading the Motor's State	148
What Is the Motor's Current Status?	149
Reading the Tilt Angle	150
Reading the Accelerometer	151
Reading the Kinect's Speed	152
Detecting Tilting Limits	152
Testing the MotorCommunicator	152

<b>CHAPTER 7 NITE Hand Gestures</b>	<b>155</b>
An Overview of the GestureDetect Application	155
Focus Detection	156
Gesture Tracking	157
Detecting Gestures	158
Initializing OpenNI	159
Processing Hand Events	160
Processing Gesture Focus Events	162
Initializing NITE	164
Processing Session Events	165
Detecting Hand Points	167
The Wave Detector	170
The Push Detector	172
The Swipe Detector	173
The Circle Detector	175
The Steady Detector	177
Filter Objects	178
<b>CHAPTER 8 NITE HandsTracker</b>	<b>179</b>
An Overview of the HandsTracker Application	179
Configuring NITE	181
Configuring the Kinect	181
Tracking the Session	183
Detecting Hand Points	185
Updates from the Kinect	187
Painting the Panel	188
Storing a Hand Trail	191
Adding a Point	191
Drawing the Trail	192
<b>CHAPTER 9 Kinect Breakout</b>	<b>195</b>
An Overview of the Kinect Breakout Application	195
Creating the Breakout Panel	196
Listening for Keys	199
Configuring the Kinect	199
Moving the Paddle	204
The Game Loop	205
Updating the Game	207
Painting the Game	209
A Sprite	210
The Sprite Constructor	210
Collision Detection	212
Updating a Sprite	212
Drawing a Sprite	213
The Paddle	213
Moving the Paddle	215
Using the Paddle's Direction	216

Managing the Bricks .....	216
Collision Detection .....	218
Drawing the Bricks .....	218
Defining a Brick .....	219
The Ball .....	219
Initializing the Ball .....	219
Updating the Ball .....	220
<b>CHAPTER 10 Gesture GUIs .....</b>	<b>223</b>
An Overview of the TestGestureGUIs Application .....	223
Component States .....	223
TestGestureGUIs Application Classes .....	225
Setting Up the Kinect .....	226
Changing Between Coordinate Spaces .....	230
The Gesture GUI Manager .....	231
Calculating the Component's Rectangles .....	232
Updating the GGUI Components .....	233
A GGUI Component Class .....	234
Component Information .....	237
Rendering the GGUI Component .....	238
The Button GGUI Component .....	240
The Dial GGUI Component .....	241
Updating the Dial .....	242
Drawing the Dial .....	245
The Slider GGUI Component .....	246
Initializing the Slider .....	248
Updating the Slider .....	248
Drawing the Slider .....	250
<b>CHAPTER 11 Kinect Capture .....</b>	<b>251</b>
An Overview of the CapturePics Application .....	252
Displaying Pictures .....	253
Snapping a Picture Again and Again and ... ..	253
Terminating the Application .....	255
Painting the Panel .....	256
Capturing the Kinect's Camera .....	257
Obtaining an Image .....	258
Closing Down .....	259
<b>CHAPTER 12 Motion Detection Using OpenCV .....</b>	<b>261</b>
Detection Tools .....	261
OpenCV .....	261
JavaCV .....	262
A Couple of OpenCV/JavaCV Examples .....	263
Motion and Change Detection in OpenCV .....	267
The CVMotionDetector Test Rig .....	268
Initializing CVMotionDetector .....	270

Detecting Movement .....	271
A Moment Spent on Moments .....	272
From Points to Pixels .....	274
Moments in OpenCV .....	275
The GUI-Based Motion Detector .....	276
The Webcam Display Loop .....	277
Reporting on the Center of Gravity .....	279
Rendering Motion Detection .....	280
Modifying the OpenCV Motion Detection .....	282
Drawbacks to This Approach to Movement Detection .....	284
More OpenCV Examples .....	285
<b>CHAPTER 13  FAAST-Style Body Gestures .....</b>	<b>287</b>
Extending GorillasTracker .....	288
An Overview of the New GorillasTracker .....	291
Creating the Gesture Detectors .....	292
Updating the Skeleton and Detectors .....	292
The Observers .....	293
A Skeletal Reminder .....	294
Detecting Basic Gestures .....	296
Checking for Gestures .....	297
Calculating Skeleton Lengths .....	298
Checking a Gesture .....	299
Turning to the Left .....	302
Moving the Right Hand Up and Down .....	303
Complex Gestures .....	305
Finding a Complex Gesture .....	306
<b>Index .....</b>	<b>309</b>



# Chapter 1

## Getting Started

This chapter introduces the Kinect sensor, explains how to install the developer software on Windows (which can be a bit tricky), looks at some examples that come with the download, and provides an overview of the Java API for programming the Kinect. It even includes a very simple Java program for testing the API.

Later chapters will start programming in earnest, looking at imaging (depth detection, camera, and infrared), point clouds, skeletal user tracking in 2D and 3D, tilt motor control, hand tracking, Kinect games, gesture support, the Kinect and OpenCV, and gesture-controlled graphical user interface (GUI) components.

I'll be programming the Kinect with OpenNI and NITE, not OpenKinect, CL NUI, or Microsoft's Kinect for Windows SDK. The 3D features (in the point cloud and skeletal tracking examples) are implemented using Java 3D. My OpenCV coding will be with its JavaCV binding. I'll be using the Java API released with OpenNI and NITE. I won't be using Processing libraries such as Simple-OpenNI.

You can find detailed code examples at my website: <http://fivedots.coe.psu.ac.th/~ad/kinect/>.

## The Kinect Sensor Hardware

The Kinect sensor was originally intended to be a motion-sensing input device for the Xbox 360, allowing the user to control games via gestures and spoken commands. Key hardware components are an RGB camera, a depth sensor, multiarray microphones, a tilt motor, and a three-axis accelerometer. Figure 1-1 illustrates the components of the Kinect sensor.

Soon after the Kinect's launch in November 2010, third-party developers started releasing software to allow it to be used on platforms other than just the Xbox (such as Windows, Linux, and the Mac). Microsoft eventually came out with a Windows 7-based SDK in June 2011. In February 2012, Microsoft released a new version of the Kinect specifically aimed at desktop PCs running Windows 7 or 8, aptly called "Kinect for Windows," featuring a new "Near Mode" that lets gesture control be used as close as 40 cm. However, all the examples in this book use the older (and cheaper) original Kinect.