华章程序员书库

Apress®

HZ BOOKS
华章IT

Java图像处理

基于OpenCV与JVM

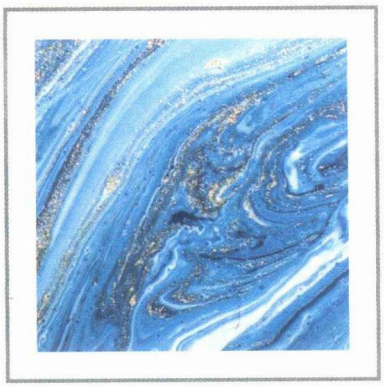
Java Image Processing Recipes

With OpenCV and JVM

[法] 尼古拉斯·莫德奇克（Nicolas Modrzyk）著

魏兰 潘婉琼 译

机械工业出版社
China Machine Press



Java Image Processing Recipes

With OpenCV and JVM

Java图像处理

基于OpenCV与JVM

[法] 尼古拉斯·莫德奇克（Nicolas Modrzyk）著

魏兰 潘婉琼 译

机械工业出版社
China Machine Press

图书在版编目（CIP）数据

Java 图像处理：基于 OpenCV 与 JVM /（法）尼古拉斯·莫德奇克（Nicolas Modrzyk）著；魏兰，潘婉琼译 . —北京：机械工业出版社，2019.4
（华章程序员书库）
书名原文：Java Image Processing Recipes: With OpenCV and JVM

ISBN 978-7-111-62388-5

I. J… II. ① 尼… ② 魏… ③ 潘… III. JAVA 语言 – 程序设计 – 应用 – 图像处理 IV. ① TP312.8 ② TP391.413

中国版本图书馆 CIP 数据核字（2019）第 057647 号

本书版权登记号：图字 01-2018-8094

Java 图像处理：基于 OpenCV 与 JVM

出版发行：机械工业出版社（北京市西城区百万庄大街 22 号 邮政编码：100037）
责任编辑：刘 锋 责任校对：殷 虹
印 刷：中国电影出版社印刷厂 版 次：2019 年 4 月第 1 版第 1 次印刷
开 本：186mm × 240mm 1/16 印 张：14.5
书 号：ISBN 978-7-111-62388-5 定 价：99.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换
客服热线：（010）88379426 88361066 投稿热线：（010）88379604
购书热线：（010）68326294 读者信箱：hzit@hzbook.com

内容简介

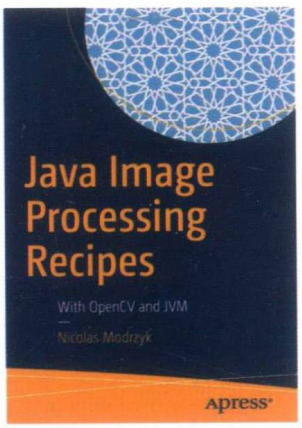
OpenCV作为计算机视觉库，一直被广泛应用于各种图像和视频相关的项目中，更在推动人工智能和神经网络的发展中发挥了重要作用。但是学习和使用OpenCV并不简单，总是需要些时间选择合适的类库、合适的编译工具、合适的编译配置等。

本书第1章将介绍使用Java、Scala和Kotlin在Java 虚拟机(JVM)运行OpenCV，同时介绍它们的一些缺点。

第2章将介绍使用Clojure语言的OpenCV封装：Origami，同时介绍如何使用它进行简单的图像操作。

第3章将用优雅的语法进阶地介绍更多先进的图像处理技术，如形状检测等。

最后，第4章转向视频处理，结合形状检测、图像变换等各种技术对实时视频流进行简单的分析。



原书封面

译者序

计算机视觉是在计算机技术、生物视觉系统、神经科学以及感知心理学等学科基础上发展起来的学科。近年来，随着机器学习技术的大力发展，以及计算机本身算力的提高，越来越多的计算机视觉问题得以解决并大规模应用，如图像检索、人脸识别、文字识别、自动驾驶等。计算机视觉技术在无形中扩展了我们探索物质世界边界的能力，并潜移默化地变革着我们的生活方式。

OpenCV 作为开放的计算机视觉源代码库，逐步成为一个通用的基础研究和产品开发平台，它不仅提供了许多前沿的图像视频处理算法，同时最大限度地进行跨平台支持，让每个对计算机视觉感兴趣的学生、科研人员以及应用开发人员参与其中。越来越多的优秀研发人员自发投身于 OpenCV 的开发和维护工作中，并致力于 OpenCV 的推广，有关 OpenCV 的书籍也如雨后春笋，日益繁多。权威广博如《 Learning OpenCV 3》，结合视觉算法原理介绍 OpenCV 函数约定及正确的使用方法；生动有趣如《 Mastering OpenCV with Practical Computer Vision Projects 》以及《 OpenCV 2 Computer Vision Application Programming Cookbook 》，通过实践探索计算机视觉的应用开发；独树一帜如《 Learning OpenCV3 Computer Vision with Python 》，介绍基于 Python 的快速上手实践。而本书无疑是对基于 JVM 的 OpenCV 实践的有力补充。本书介绍了使用 Java、Scala 和 Kotlin 在 JavaVM 上运行 OpenCV，随后引入 Clojure 语言的 OpenCV 封装——Origami，并提供了各种纷繁有趣的实践项目。读者可以依次学习这些小项目，也可以直接跳到感兴趣的章节，更可以充分发挥你的想象力，将学到的内容应用于日常生活中，如马赛克拼图、实时视频检测等。

学以致用，实践出真知，相信本书的问题和样例会有助于你快速应用计算机视觉的诸多算法。

本书的翻译工作由我和潘婉琼合作完成，我们穿插着完成了每章不同节的翻译，最后共同进行了终稿的合并以及审校。在翻译过程中，我们力求忠实原文。但限于译者的知识水平、翻译经验和翻译时间，必定会有诸多不足，恳请广大读者谅解。

最后，祝愿所有读者能够在阅读和实践中享受本书带来的乐趣！

魏兰

2019 年 1 月于北京

前 言

我的父亲是一名牙科医生。在我的记忆里，童年时期父亲曾经一遍又一遍地重复同一句话，现在翻译过来应该是：

“儿子，工欲善其事，必先利其器。”

当他看着我试图用错误的洗涤产品事倍功半地洗车时，在内心深处我是认可他的这句话的。

我的父亲不会用螺丝刀从患者口中拔牙，相反，他用 20 多种形状各异的刷子去清洗不同类型的牙齿，当时我还觉得这是很搞笑的事情。

时光飞逝，30 年后，我和父亲谈起这本书，他补充道：

“嗯，儿子，你知道，这不仅仅是关于工利其器，而且是关于对的时间选择对的利器。”

我想这基本可以作为对本书哲学观的总结。

作为计算机视觉库，OpenCV 一直被广泛应用于各种图像和视频相关的项目中，更在推动人工智能和神经网络的发展中发挥了重要作用。但是学习和使用 OpenCV 并不简单，总是需要些时间选择合适的类库、合适的编译工具、合适的编译配置等。

由 Clojure 封装 Origami 的愿景是用令人愉悦的语法，直截了当地发挥 OpenCV 的巨大威力。我希望这样能帮助你集中更多的时间和精力在项目本身而非工具上。

第 1 章将介绍使用 Java、Scala 和 Kotlin 在 Java 虚拟机（JVM）上运行 OpenCV，同时介绍它们的一些缺点。

第 2 章将介绍使用 Clojure 语言的 OpenCV 封装——Origami，同时介绍如何使用它进行简单的图像操作。

第 3 章将用优雅的语法介绍更多先进的图像处理技术，如形状检测等。

第 4 章转向视频处理，结合形状检测、图像变换等各种技术对实时视频进行简单的分析。

作者简介

Nicolas Modrzyk 目前是 Karabiner Software 的首席技术官和开发团队的领导者。

他还是开源软件社区的活跃贡献者。作为开发人员和技术顾问，他多年来一直在一家视频会议公司参与设计大型服务器应用程序，通过从零开始编写高性能中间件来管理庞大的数据库集群，使日本领导者能够使用内容管理和流程管理系统，并为亚洲领先公司拓宽了业务流程的界限。

他也是敏捷方法的热情倡导者，专注于及时完成工作以满足客户需求。他还喜欢鼓励朋友和团队成员挑战自我并实现目标。他在不同国家拥有专利授权收入，包括法国、美国、爱尔兰、日本、中国和印度。他还是其他几本有关 Clojure 编程语言的书的作者，这些书以英语和日语出版。

他目前居住在日本东京，在那里他常常利用业余时间踢足球、徒步旅行、在现场音乐会演奏吉他，并与朋友和同事一起享受生活。

技术审校者简介

Aakash Kag 是 Manacola Private 公司的人工智能开发人员。他有两年大数据分析工作经验，是一名计算机科学专业的研究生，专业方向是大数据分析。Aakash 曾开发过微软机器人。

目前，Aakash 致力于解决有关聊天机器人和自然语言理解方面的难题。

他热衷于参加机器学习交流会，而且经常在交流会上发表演讲。

译者简介

魏兰

联系方式：weilan.cqu@gmail.com

重庆大学软件工程专业本科毕业，北京大学计算机科学与技术专业硕士毕业。现就职于Google北京，任软件开发工程师。主要感兴趣领域为Android移动开发、计算机视觉、图像处理。CSDN专家博主，博客地址：https://blog.csdn.net/xiaowei_cqu。

潘婉琼

联系方式：poemqiong@gmail.com

本科就读于北京邮电大学计算机科学与技术专业，毕业后前往北京大学攻读计算机应用技术硕士学位，研究方向为文字图像处理，发表多篇相关论文和专利。现就职于Google美国总部，从事客户端开发和前端基础架构。

JAVA IMAGE PROCESSING RECIPES
With OpenCV and JVM

目 录

基于 JavaVM 的 OpenCV

第1章

几年前，在去上海的旅途中，一位好友送给我一本很厚的书，是介绍 OpenCV 的。书中包含了海量的图像处理方法、实时视频分析例子和引人入胜的深度解析，于是我迫不及待地配置好环境来测试书中的程序。

众所周知，OpenCV 是开源计算机视觉（Open Source Computer Vision）的英文简写。作为一个开源库，OpenCV 提供可直接使用的高级图像处理算法，既包括简单易用的高级图像操作，也包括形状识别以及实时视频监测和分析功能。

OpenCV 中最核心的内容是多维矩阵对象，叫作 Mat。通过本书的学习，Mat 将成为我们最熟悉的朋友。在许多攻略中，输入的对象是 Mat，处理的内容是 Mat，输出的结果也是 Mat。

虽然 Mat 即将成为我们的好朋友，但是作为一个 C++ 对象，它并不是很好相处。你必须重新编译、安装和小心地配置任何使用 Mat 的新环境。

但是 Mat 可以被打包。

Mat 虽然在本地运行，但它可以被神不知鬼不觉地加载到 Java 虚拟机中运行。

第 1 章将通过介绍 Java 虚拟机中的多种语言让你开始上手使用 OpenCV，当然包括 Java 语言，也包括通俗易懂的 Scala 语言和谷歌最爱的 Kotlin 语言。

为了使用同样的方法来运行不同的语言，你会首先（重新）认识一种 Java 编译工具，叫作 Leiningen，之后利用它来运行简单的 OpenCV 函数。

第 1 章是第 2 章的入门基础。第 2 章的内容是相似的基于 JVM 的 Clojure 语言，可以为富有创造性的 OpenCV 代码带来即时的视觉反馈。

1.1 初识 Leiningen

问题定义

有一句名言是“一次编写，随处运行”，也就是说，在不同的机器上，可以用同样简单便捷的方法来编译和运行 Java 程序。当然，你总是可以使用最原始的 javac 命令来编译 Java 代码，然后使用单纯的 Java 在命令行中运行编译过的代码，但现在已经是 21 世纪了，我们应该寻找更有效的方法。

无论使用何种编程语言，手动配置工作环境都是一项大工程。而且当你完成配置之后，很难与他人分享胜利果实。

使用编译工具，可以用简单的方法定义项目所需的依赖，同时也可帮助其他用户更快地上手。

接下来，我们介绍一个简单易用的编译工具。

解决方法

Leiningen 是（主要）面向 JavaVM 的编译工具。它与一些知名的工具有些相似，例如 Ant、Maven 和 Gradle。

当 Leiningen 命令行安装完成之后，就可以基于模板来创建 JavaVM 项目，并且毫无顾虑地运行程序了。

本攻略将介绍如何快速安装 Leiningen，以及如何使用它运行你的第一个 Java 程序。

工作原理

首先，把 Leiningen 安装在你需要的地方，然后用它来创建一个空的 Java 项目。

注意　安装 Leiningen 之前，需要在你的电脑上安装 Java 8。由于 Java 9 通过破坏现有方法来解决旧的问题，我们目前还是选择使用 Java 8。

安装 Leiningen

Leiningen 的网站主页是 https://leiningen.org/。

在主页上方，可以找到手动安装 Leiningen 的四个简单步骤。

在 MacOS 和 Unix 环境中：

1. 下载 lein 脚本。https://raw.githubusercontent.com/technomancy/leiningen/stable/bin/lein
2. 把它放在你的 $PATH 变量中，以便 shell 可以找到它（例如～ /bin）。
3. 设置脚本为可运行（chmod a+x ~/bin/lein）。

4. 在终端运行 lein，然后它会下载一个安装包。

在 Windows 环境中：

1. 下载 lein.bat 批处理脚本。https://raw.githubusercontent.com/technomancy/leiningen/stable/bin/lein.bat

2. 用管理员权限把它放在你的 C:/Windows/System32 文件夹下。

3. 打开命令提示符运行 lein，然后它会下载一个安装包。

在 Unix 环境中，你可以使用包管理器。在 MacOS 中，Brew 有 Leiningen 的包。

在 Windows 中，也有一个很好的安装器在 https://djpowell.github.io/leiningen-win-installer/。

如果你是一个 Chocolatey 粉丝，Windows 也有 Chocolatey 的包：https://chocolatey.org/packages/Lein。

如果你在终端或命令提示符中安装成功，那么你就可以看到已安装工具的版本号。在第一次运行的时候，Leiningen 会下载它的内部依赖（internal dependecy），但是之后的运行通常会非常快。

```
NikoMacBook% lein -v
Leiningen 2.7.1 on Java 1.8.0_144 Java HotSpot(TM) 64-Bit
Server VM
```

用 Leiningen 创建新的包含 OpenCV 的 Java 项目

Leiningen 通常使用一个文本文件，叫作 **project.clj**，里面有一个简单的图，存储着元数据（metadata）、依赖（dependencies）、插件（plug-ins）和配置（settings）。

当你通过调用 lein 命令来运行项目时，lein 会去 project.clj 文件中查找该项目的相关信息。

Leiningen 自带可直接使用的项目模板，但是为了更好地理解，我们来一步步地学习第一个例子。

对于一个 Leiningen Java 项目，你需要两个文件：

- 一个用来描述项目的 project.clj 文件
- 一个包含 Java 代码的 .java 文件，例如 Hello.java

第一个项目的目录结构看起来是这个样子的：

```
.
├── java
│   └── Hello.java
└── project.clj
```

一个目录，两个文件。

为了简化问题，第一个 Java 例子十分简单。

```
public class Hello {
    public static void main(String[] args) {
            System.out.println("beginning of a journey");
    }
}
```

接下来，让我们来看一看 project.clj 文件中的细节：

```
(defproject hellojava "0.1"
  :java-source-paths ["java"]
  :dependencies [[org.clojure/clojure "1.8.0"]]
  :main Hello)
```

这其实是 Clojure 代码，不过我们把它看作一种领域专用语言（Domain Specific Language，DSL），即使用特定术语描述项目的语言。

为了方便，每种术语在表 1-1 中有解释。

表 1-1 Leiningen 项目元数据

术语	用法
Defproject	定义项目的入口
Hellojava	项目名称
0.1	版本号
:java-source-paths	用于存放 Java 代码文件的一组目录，关联到项目文件夹
:dependencies	项目所需的外部依赖库，以及所需的版本号
[[org.clojure/clojure “1.8.0”]]	默认情况下，该列表包含 Clojure，这是运行 Leiningen 所必需的。之后会把 OpenCV 库也放在这里

（续）

术语	用法
:main	默认运行的 Java 类名称

接下来，请创建对应的文件夹和文件目录，复制并粘贴每个文件对应的内容。

完成之后，就可以运行你的第一个 Leiningen 命令：

```
lein run
```

这个命令会根据你的环境，在你的终端或控制台中生成以下内容：

```
Compiling 1 source files to /Users/niko/hellojava/target/classes
beginning of a journey
```

太棒啦！我们的旅程开始啦！但是，等一等，刚才到底发生了什么呢？

其中包含了一点魔法。Leiningen 的 `run` 命令会让 Leiningen 运行一个编译过的 Java 类 `main` 函数。这个被运行的类定义在项目的元数据中，你应该记得，叫作 **Hello**。

在运行 Java 类之前，我们需要先编译它。默认情况下，Leiningen 会在运行 `run` 之前进行编译，这也解释了“Compiling...”是从哪里来的。

之后，你可能会注意到，在你的项目中创建了一个叫作 target 的文件夹，里面包含了一个叫作 classes 的文件夹和一个 Hello.class 文件。

```
.
├── dev
├── java
│   └── Hello.java
├── project.clj
├── src
├── target
│   ├── classes
│   │   ├── Hello.class
```

target 和 classes 文件夹是编译过的 Java 字节码（bytecode）的默认存储地址，这个 target 文件夹之后会被加入 Java 运行时的类路径（classpath）。

紧跟着是“lein run”触发的执行阶段，`Hello` 类的主函数中的代码块被执行，并输出

信息。

```
beginning of a journey. 旅途开始
```

你可能会问："如果我有多个 Java 文件，并且想运行非主函数之外的其他函数呢？"

这是一个非常合理的问题，在第 1 章中编写和运行不同代码时，你会经常使用这个技巧。

假如你在同样的 Java 文件夹中写了第二个 Java 类，叫作 Hello2.Java，其中包含更新的旅途内容。

```
import static java.lang.System.out;
public class Hello2 {
    public static void main(String[] args) {
            String[] text = new String[]{
                    "Sometimes it's the journey that ",
                    "teaches you a lot about your destination.",
                    "--",
                    "- Drake"};
            for(String t : text) out.println(t);
    }
}
```

为了运行 Hello2.java 的主函数，在调用 lein run 时，需要加上 -m 选项，这里 m 代表着主函数，之后跟着需要运行的 Java 类的名称。

```
lein run -m Hello2
```

这个命令输出以下内容：

```
Compiling 1 source files to /Users/niko/hellojava/target/
classes
Sometimes it's the journey that
teaches you a lot about your destination.
--
- Drake
```

太好啦！根据这些指导，你可以勇往直前，运行你的第一个 OpenCV Java 程序了。