Getting Started with Computer Vision Proposal

by Greg Borenstein

3/1/2013

Introduction

The field of computer vision has a deceptively simple mission: build up information from images to make conclusions about the world. In a time when we're more and more surrounded by increasingly high resolution cameras hooked up to powerful computers (cf. the More Pixels Law) and huge online databases of images (the world's largest photo libraries), computer vision is taking its place as one of the foundational skills necessary to invent the next generation of technology: from user interfaces, to mobile applications, to search engines, to social networks.

Until now, computer vision has mainly been the domain of academic researchers, computer science wizards, and a few adventurous interactive artists and designers. However the tools of the field have never been easier to use. Specifically, OpenCV, a huge open source project implementing a tremendous range of computer vision algorithms and techniques, has reached an advanced level of maturity where it is widely available (including on both iOS and Android) and integrated into creative coding toolkits that are accessible to beginners.

This book attempts to create an easy entry point for a much broader set of people to get started with computer vision. It introduces the code and concepts necessary to build computer vision projects with the Processing creative coding environment. It is designed to be accessible to a beginner audience, but also to instill the fundamental concepts of the field, enabling readers to not just follow recipes, but build their own projects.

To achieve these goals, this book is organized around the six stages that constitute most computer vision applications:

- Capture
- Filter
- Calibrate
- Track
- Train
- Combine

The plan for this book is to integrate these techniques into fun and motivating projects such as:

- A Caroonifier
- Object tracking for a simple game
- Augmented Reality
- Identifying toys
- Recognizing faces

Note: Due to the inherently rich multi-media nature of computer vision applications, this book is a prime candidate to use O'Reilly's new web-based publishing tools that include video and interactive pieces. It should be planned for that format from the start with the appropriate modules designed as scripts for videos, interactive pieces, quizzes, etc. at the writing stage rather than waiting for a written book to be completed first.

Timing and Code Development

A brief note about some practical issues related to timing of the book's release and the development process. I am currently negotiating with the Processing Foundation to work for them on updating the Processing OpenCV library for Processing 2.0. This work will pair very well with this project as I'll be able to include all the necessary functions in the library for the projects here and this book will act as comprehensive documentation for that project which can be maintained along with the project going forward.

Given the goals of producing rich online multi-media content for this project, I believe that the individual pieces should be released as they are completed so we can get feedback from the community and possibly even participation and collaboration from other core Processing members who teach with OpenCV.

The release of Processing 2.0 (probably this summer) will be the target release date for this new OpenCV library and would also be a good target for having at least the first components of this project publicly available to draft on the attention from those announcements.

Chapter Outline

1. First steps: Cartoonifier

This first chapter introduces the basics of how to work with OpenCV in Processing. We'll learn how to access images and live camera feeds as well as how to do the basic filtering on these images that makes all other applications possible.

Project: build an application that renders a person's face in a cartoon-y style.

- Install OpenCV for Processing
- Access camera, load still images
- Filter image: brightness, contrast, convert to grayscale, blur, invert, etc.
- Background subtraction with remember() and absDiff()
- Contour finders (canny, scharr, sobel)
- Face detection with HAAR finder
- flood fill

2. Tracking an Object

The ability to locate and track objects in images enables all kinds of new ways for people to interact with computers. In this chapter we'll explore one of the most basic and reliable ways to track objects: based on their color. We'll start with the simplest possible color tracker and then we'll improve it to make it more robust. Finally, we'll explore some other, more complex ways of tracking objects.

- Basic color tracking: inRange, mixerRGBGray, thresholding, blobs
- Connected components with cvBlob
- Adaptive color tracking using mean-shift/cam-shift
- template matching
- preview the idea of image features from AR chapter

3. Augmented Reality

Augmented Reality has been a hot area of interaction design research and experimentation for the last few years. Creating AR interfaces involves adding computer graphics to live camera input so that those graphics match the perspective of the scene, creating the illusion that they are actually present. In order to accomplish this, we must track an object in the scene (as in the previous chapter) and also determine its size and orientation so we know how to position our graphics.

There are two main approaches to AR: using markers and markerless.

"AR markers" are flat objects with visible patterns that are designed make it easy to detect them and to find their orientation. They also frequently contain information uniquely identifying each marker. There are standard AR marker sets with accompanying code that makes them relatively easy to work with. We'll explore the basic OpenCV functions used to detect and identify these markers. Then we'll learn how to use the ARToolkit, which makes it easy to work with a particular set of markers.

Manual

- Camera Calibration: creation and loading
- image binarization
- Contour detection
- polygon approximation
- marker detection
- pose estimation

ARToolkit

- Install NyARToolkit (or SimpleARToolkit)
- Detect markers
- Determine orientation
- Display 3D cube, display 3D model

Sometimes, you don't want to or can't use an AR marker, either to avoid the appearance of the ugly markers or because you want to track some pre-existing markerless object. OpenCV enables markerless object tracking using "image features", unique parts of objects that can be reliably found in multiple images even as the object moves and rotates. We'll see how to track such an object in an AR application.

Markerless

- Find image features (using corner detection) in a source image
- FLANN search for similar features
- outlier filtering with RANSAC
- pose estimation

4. Object and Face Recognition with Machine Learning

Recognition is the task of identifying a person or an object in a never-before-seen image using data extracted from a pre-processed set of images of many people or objects. Face recognition is how Facebook can automatically tag you in new photos. Recognition can let us detect hand gestures and specific toys.

All recognition applications are based on machine learning: the process of training an algorithm based on data. In this chapter we'll use OpenCV techniques to extract data from images, we'll use its machine learning tools to train classifiers based on this data, and then we'll use these classifiers to recognizes, faces, toys, and hand gestures.

- Building a feature vector: color histogram
- Training a Support Vector Machine (libsvm vs OpenCV implementation)
- Using SVM for matching
- Militarizing your backyard with Python example (video)
- Histogram of Oriented Gradients feature vector
- Toy detection and hand gesture detection
- Windowed search
- Eigenfaces