New functionality in OpenCV 3.0

By Oscar Deniz Suarez, coauthor of the book “OpenCV Essentials”.

The forthcoming version 3.0 represents a major evolution of the OpenCV library for computer vision. At the time of writing this article, OpenCV already includes several new techniques that are not available in the latest official release (2.4.9). The new functionality can be already used by downloading and compiling the latest development version from the official repository. This article provides an overview of some of the new techniques implemented. Other numerous lower-level changes in the forthcoming version 3.0 (updated module structure, C++ API changes, transparent API for GPU acceleration, etc.) are not discussed.

Line Segment Detector

OpenCV users have had the Hough-transform based straight line detector available in previous versions. An improved method called LSD (Line Segment Detector) is now available. LSD is based on the algorithm described in:

> Rafael Grompone von Gioi, Jérémie Jakubowicz, Jean-Michel Morel, and Gregory Randall, LSD: a Line Segment Detector, Image Processing On Line, vol. 2012. `http://dx.doi.org/10.5201/ipol.2012.gjmr-lsd`

This method has been shown to be more robust and faster than the best previous Hough-based detector (the Progressive Probabilistic Hough Transform).

The detector is now part of the `imgproc` module. OpenCV provides a short sample code (`[opencv_source_code]/samples/cpp/lsd_lines.cpp`) which shows how to use the class `LineSegmentDetector`. The following table shows the main components of the class:

Method	Function
<constructor>	The constructor allows to enter parameters of the algorithm, particularly the level of refinements we want in the result

detect	Detects line segments in the image
drawSegments	Draw the segments in a given image
compareSegments	Draw two sets of segments in a given image. The two sets are drawn with blue and red color lines.

Connected components

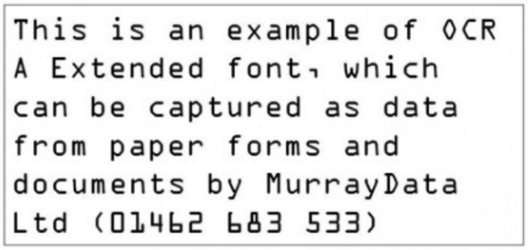
Previous versions of OpenCV have included functions for working with image contours. Contours are the external limits of 'connected components' (i.e. regions of connected pixels in a binary image). The new functions `connectedComponents` and `connectedComponentsWithStats` retrieve connected components as such.

The connected components are retrieved as a labeled image with the same dimensions as the input image. This allows drawing the components on the original image easily. Function `connectedComponentsWithStats` retrieves useful statistics about each component:

CC_STAT_LEFT	The leftmost (x) coordinate which is the inclusive start of the bounding box in the horizontal direction
CC_STAT_TOP	The topmost (y) coordinate which is the inclusive start of the bounding box in the vertical direction
CC_STAT_WIDTH	The horizontal size of the bounding box
CC_STAT_HEIGHT	The vertical size of the bounding box
CC_STAT_AREA	The total area (in pixels) of the connected component

Scene text detection

Text recognition is a classic problem in computer vision. Thus, Optical Character Recognition (OCR) is now routinely used in our society. In OCR, the input image is expected to depict typewriter black text over white background. In the last years researchers aim at the more challenging problem of recognizing text "in the wild": on street signs, indoor signs, with diverse backgrounds and fonts, colors, etc. The following figure shows and example of the difference between the two scenarios. In this scenario, OCR cannot be applied to the input images. Consequently, text recognition is actually accomplished in two steps. The text is first localized in the image and then character or word recognition is performed on the cropped region.



OpenCV now provides a scene text detector based on the algorithm described in:

> Neumann L., Matas J.: Real-Time Scene Text Localization and Recognition, CVPR 2012 (Providence, Rhode Island, USA).

The implementation of OpenCV makes use of additional improvements found in:

> Gomez L. and Karatzas D.: Multi-script Text Extraction from Natural Scenes, ICDAR 2013. Available at `http://158.109.8.37/files/GoK2013.pdf`

OpenCV includes an example (`[opencv_source_code]/samples/cpp/textdetection.cpp`) that detects and draws text regions in an input image.

KAZE & AKAZE features

Several 2D features have been proposed in the computer vision literature. Generally, the two most important aspects in feature extraction algorithms are computational efficiency and robustness. One of the latest contenders is the KAZE (Japanese word meaning "Wind") and Accelerated-KAZE (AKAZE) detector. There are reports that show that KAZE features are both robust and efficient, as compared with other widely-known features (BRISK, FREAK, …). The underlying algorithm is described in:

KAZE Features. Pablo F. Alcantarilla, Adrien Bartoli, and Andrew J. Davison. In European Conference on Computer Vision (ECCV), Florence, Italy, October 2012.

As with other keypoint detectors in OpenCV, the KAZE implementation allows retrieving both keypoints and descriptors (i.e. a feature vector computed around the keypoint neighbourhood). The detector follows the same framework used in OpenCV for other detectors, so drawing methods are also available.

Computational photography

One of the modules with most improvements in the forthcoming version 3.0 is the computational photography module (`photo`). The new techniques include:

Functionality	**Description**
HDR imaging	Functions for handling High-Dynamic Range images (tonemapping, exposure alignment, camera calibration with multiple exposures and exposure fusion)
Seamless cloning	Functions for realistically inserting one image into other image, with an arbitrary-shape region of interest.
Non-photorealistic rendering	Includes non-photorealistic filters (such as pencil-like drawing effect) and edge-preserving smoothing filters (those are similar to the bilateral filter).

New modules

Finally, we provide a list with the new modules in development for version 3.0:

Module name	**Description**
videostab	Global motion estimation, Fast Marching method
softcascade	Implements a stageless variant of the cascade detector which is considered more accurate
shape	Shape matching and retrieval. Shape context descriptor and matching algorithm, Hausdorff distance and Thin-Plate Splines
cuda<X>	Several modules with CUDA-accelerated implementations of other functions in the library