

Computer Vision in Processing

The following workshop shows some very simple computer vision examples in Processing using the BlobDetection library from v3ga.

1. Download and install the library in your Processing library folder, following the instructions at: <http://www.v3ga.net/processing/BlobDetection/index-page-download.html>
2. Download the Webcam example sketch from v3ga at: <http://www.v3ga.net/processing/BlobDetection/index-page-examples.html>
3. Run the example sketch in Processing – on the new iMacs in our lab the sketch automatically takes the camera images from the inbuilt iSight camera.

For good tracking results you should always start with the best possible video input to your computer. Since the BlobDetection library is aimed at doing computer vision by finding ‘blobs’ on an image, that is to say areas whose brightness is above or below a particular value, a black and white camera is sufficient. If possible make sure to be able to focus the camera and adjust the iris manually. Avoid “auto-everything” cameras, such as webcams if possible. In the demos in class I used the following configuration:

- SONY PC164CEX-2 Super Low Light High Resolution Monochrome Video Camera with Elmo, T2812W 1/3" CS-Mount 2.8-12mm Vari-Focal Lens, (B&H #ELT2812W)
- 100ft. BNC/power cable, 12V DC wall adapter
- Imaging Source DFG/1394-1e Video-to-FireWire converter, 12V DC wall adapter, 6ft. firewire 6pin to 6 pin cable.

The advantage of using an analog camera like the SONY PC164CEX is that the image quality is better than that of a regular miniDV camera and that the analog signal can be send much longer distances than a digital one (through a firewire cable for example). In the above setup the 100ft. BNC cable does not decrease the quality of the camera signal at all.

A further improvement would be the use of an Infrared illuminator and IR filter for the camera which would allow to project on the same surface that is also used as the background for the computer vision application.

4. Take a look at the documentation of the library at: <http://www.v3ga.net/processing/BlobDetection/index-page-documentation.html> which explains which methods and fields you can access and work with.

In the following I have listed some basic Processing sketches that are based on the standard webcam example sketch that you can download from the v3ga website. They show how you can extend this sketch quite easily by changing/adding only a few things.

In general, you can change the following lines of code to make the blob detection more precise:

```
cam = new Capture(this, 40*4, 30*4, 15);
```

This line determines the pixel resolution of the camera image to be displayed, the DFG/1394-1e Video-to-FireWire converter can handle a maximum resolution of 640x480 pixels for NTSC cameras, so

```
cam = new Capture(this, 640, 480, 15);
```

would be the highest resolution camera image for display purposes.

```
img = new PImage(80,60);
```

is the actual pixel size of the image that is being sent to the blob detection function.

```
img = new PImage(320,240);
```

still runs fine on the Intel iMacs and results in a much more precise blob detection.

```
img = new PImage(640,480);
```

slows down the visualization noticeably.

```
theBlobDetection.setPosDiscrimination(true);
```

If the passed boolean value is true, the blob detection will attempt to find bright areas. Otherwise, it will detect dark areas (default mode).

```
theBlobDetection.setThreshold(0.2f);
```

this will detect bright areas whose luminosity is greater than 0.2f, experiment with different thresholds!

Examples

1. The first example extends the original webcam sketch by some code that shows you how, in addition to the outline of a blob and its bounding box, you can also easily get a blob's center:

http://web.ics.purdue.edu/~fwinkler/590E_S08/bd_webcam_Centers.zip

2. The second example show how you can easily create "trigger" areas in your video image that can be activated by blobs entering them. In this example the word "triggered" is printed to the console in the Processing window whenever a blob's bounding box enters the trigger area on the left hand side of the screen.

http://web.ics.purdue.edu/~fwinkler/590E_S08/blob_detection_trigger_zone.zip

3. Example three is an extension of example 2 only now we turn on output pin 13 (and the onboard LED) on an Arduino board whenever a blob's bounding box enters the trigger area in the video frame. Obviously, the changing output on pin 13 can trigger any kind of physical output you wish. In the example in class I connected a relay to it which turned on a 60W household light bulb when somebody or something entered the trigger area.

http://web.ics.purdue.edu/~fwinkler/590E_S08/blob_detection_trigger_zone_serial.zip
http://web.ics.purdue.edu/~fwinkler/590E_S08/Arduino_serial_receive.zip

4. The next example illustrates one way of finding coordinates of a blob's maximum X or Y values and attaching a simple geometric shape to it. This shape can easily be replaced by an image or any other graphics element. In this concrete example the program finds the coordinate of the vertical maximum of all blobs and attaches an aqua colored square to it.

http://web.ics.purdue.edu/~fwinkler/590E_S08/blob_detection_get_max_Y.zip

5. The last example is still quite sketchy and resulted from a short experiment in class. It takes the apex point from the previous example but uses it to draw lines connecting the apex point from the previous frame to the one of the current frame. Unfortunately I had to turn off the video visualization since it would erase all the previously drawn lines on top of each video frame.

http://web.ics.purdue.edu/~fwinkler/590E_S08/blob_detection_draw_with_maxY.zip

Further Reading

Golan Levin: "Computer Vision for Artists and Designers: Pedagogic Tools and Techniques for Novice Programmers"

http://www.flong.com/texts/essays/essay_cvad/

More Examples (not blob detection related!)

Make your Processing sketch run fullscreen using the fullscreen library

http://www.superduper.org/processing/fullscreen_api/

This is a modified version of the simple continuous lines example: click and drag the mouse to draw lines.

http://web.ics.purdue.edu/~fwinkler/590E_S08/ContinuousLines_fullscreen.zip

Smoothing motion, de-acceleration in the process of moving a shape from one point to another. This example makes a circle follow the position of the mouse with a de-accelerated motion.

http://web.ics.purdue.edu/~fwinkler/590E_S08/follow_mouse_objdest.zip

Sending more than one sensor value from the Arduino board to Processing – a simple etch-a-sketch like application that moves a dot on the screen in X and Y position based on the values of two potentiometers.

Arduino code:

http://web.ics.purdue.edu/~fwinkler/590E_S08/Arduino_serial_two_analog_sensors.zip

Processing code:

http://web.ics.purdue.edu/~fwinkler/590E_S08/etch_a_sketch.zip