Computer Vision Pipeline For Object Recognition

Kevin Zhang and Felipe Hofmann
November 17th, 2017

Commitment

NTSC Decoder / NTSC to ZBT / ZBT Memory / XVGA / Display

These modules make up the camera-video pipeline. In order, they decode NTSC signals, write to ZBT memory, generate VGA display signals, and write to the display.

Demo: The monitor displays the camera input.

Line Buffer

This module is responsible for caching 8-line subimages and feeding each 8x8 patch to the linear scorer by pulling ready high when the data is available.

Demo: We can achieve over 8fps object tracking due to this buffer.

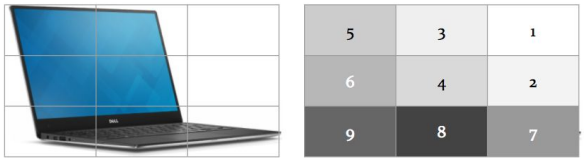
Linear Scorer

This passes the subimages to the individual detector modules and returns a linear combination of the scores returned by the detectors. In addition, this module is responsible for interpreting the user input - i.e. button down means that the user is training the module to recognize object X, where X is determined by the switches - and pulling *train* high.

Demo: The switches and enter button are functional and objects can be tracked.

Brightness Ordinal

This module accepts *subimage[64*8-1:0]* and train as inputs and returns a *score[3:0]*. It computes a brightness ordinal which splits the subimage up into an even grid and returns a sorted list of cells from brightest to darkest. This gives us a coarse measure of the morphology of the object.



As described above, this detector module also trains in 1 clock cycle and produces scores with purely combinatorial logic.

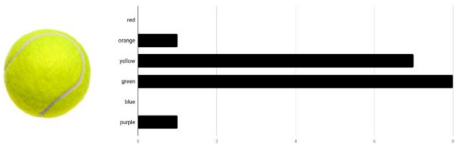
Demo: Large bright or dark spots can be tracked.

Goal

Hue Histogram

When *train* is pulled high, the *score* is set to 16 and the hue histogram module stores the current histogram; when train is pulled low, this module returns a score indicating how similar the current histogram is to the stored histogram.

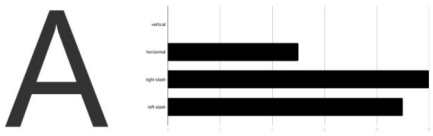
An example of a histogram is shown below; as you may expect, a tennis ball will have high yellow and green values but low counts for the other colors.



Demo: Colorful balls can be tracked.

Edge Histogram[1]

It computes a simplified "histogram of oriented gradients" in the subimage. For example, the below diagram shows the 4 primary types of edges found in the letter A.



Demo: Printed letters and objects with straight edges (i.e. iPhones) can be tracked.

Linear Scorer w/ Feedback

The key distinction between this module and the original linear scorer is that it offers rotational invariance as well as support for non-rigid objects. The model is partially refit to the new maximum likelihood location, so rotating/non-rigid objects can be tracked.

Demo: An object which rotates can be tracked as it moves through the frame.

Stretch Goal

Linear Scorer w/ Learned Parameters

The key distinction between this module and the linear scorer w/ feedback is that the weights for the linear combination of scores is now learned through approximate gradient descent rather than hardcoded.

Demo: Empirically demonstrate that accuracy is higher when sw[7] is high.

Model Mover

This module interfaces with a computer over a serial RS232 port and allows models to be pushed in both directions. After training your detector on the FPGA, you can push the up button to transfer the model to the computer where a Python script allows you to give it a name and save it. When you press the down button, the model mover waits for the Python script to send a model back and loads it into the appropriate object id.

Demo: You can train a model, copy it to a laptop, reset the FPGA, and reprogram it from the laptop.

---

[1] A variant of this is the histogram of oriented gradients (HoG) found in computer vision literature.