Computer Vision on the Beagleboard-xM

Verify Beagleboard-xm with the angstrom test image

The Beagleboard-xm comes with angstrom pre-loaded. Here is the minimum set-up:

- USB Keyboard
- USB Mouse
- HDMI-male to DVI-male cable.
- A DVI monitor (HDMI-only won't work, for this you would need a DVI to HDMI adapter which has active circuitry.) The HDMI connector for the Beagleboard was only chosen for its size; digital output is DVI format.
- A 5V (minimum 1A, maximum 3A) DC power supply with 2.1mm center hot connector, 5.5mm outside diameter

You should also have a null-modem serial cable and rs-232 serial interface, or a usb-to-serial adapter. Use putty or another terminal client at baud rate 115200.

Make all these connections except power. Do not power on the beagleboard until your HDMI video cable is connected and the monitor is turned on. Don't turn off the monitor without first turning off the Beagleboard.

To verify the Beagleboard hardware, insert the micro-sd card and connect the power. The BB should boot right up into a Linux desktop interface. If you don't get an OS, get the demo image and set up the SD card according to instructions here. If you still can't boot using that SD card, you may have a hardware issue.

I had some problems with overscan on the monitor. The max resolution the BB-xm will put out is 1280×1024. All of the test driver output modes are in 4:3 aspect-ratio. For monitors with native resolutions other than these, the BB’s video output might result in clipped edges at all resolutions.

Useful parts from Digikey

T946-P5P-ND 5VDC Adapter (highly recommended over using USB power.)
WM19082-ND Video cable, HDMI-Male to DVI-Male

Ubuntu on the Beagleboard-xM

Originally, Angstrom was chosen because it was the demo image and runs faster. Unfortunately, it has poor linux webcam support. Ubuntu is the clear leader in this arena; most webcams will be automatically detected.

There are problems with the pre-built images provided by Canonical (producers of Ubuntu.) Thus, a 'hacked' "Demo Image" must be used.

Instructions are here: elinux.org

A condensed version is reproduced here for convenience. A linux machine with a card-reader is required for this step.

- Insert the micro-sd card and note the device (this can be found by examining the output of the mount command.) In my example, the sd card-reader is /dev/sdb
- Wipe out the card:
pay careful attention to use of **parted** because you can wipe out your linux file system with careless use
first, unmount all partitions on the card. in this example, two partitions were mounted.

```bash
umount /dev/sdb1
umount /dev/sdb2
```
(you might need to unmount more if Ubuntu mounted more than two partitions from your sd card.)

```bash
parted /dev/sdb rm 1 removes the first partition
parted /dev/sdb rm 2 removes the second partition
```

- Verify Image with: `md5sum ubuntu-11.04-r1-minimal-armel.tar.xz`
  output should be `6ae3f2ac6d96dc42b4682a98bb0488ba ubuntu-11.04-r1-minimal-armel.tar.xz` (check website for newer hashes if you're using a newer build.)
- Unpack Image: `tar xf ubuntu-11.04-r1-minimal-armel.tar.xz`
- Change directory to newly created image dir: `cd ubuntu-11.04-r1-minimal-armel`
- Install the image using the provided script: `sudo ./setup_sdcard.sh -mmc /dev/sdX -uboot beagle`
  (this may take 10 or so minutes.)
- type `sync` once the script has finish to make sure all data is written to the SD card.
- eject the card, insert into the BB-xM and boot (make sure you have an ethernet cord plugged in during the boot process.)

Credentials to log in are:
- user: `ubuntu`
- pass: `temppwd`
Change the password
- `passwd`

Get online:
- `sudo ifconfig -a` to view connections
- choose one of the following commands:
  - `sudo dhclient usb1` IF you see a usb1 interface present
  - `sudo dhclient eth0` IF you do not see usb1, but instead see eth0
  (do not execute `sudo dhclient usb0`)
- you should be able to execute `sudo ifconfig -a` again and see that the ipv4 inet address is one handed out by your router or DHCP server.
Update the package management system (apt)
- `sudo apt-get update`
  Install SSH (to allow remote access and a host of other useful linux networking features such as scp)
- `sudo apt-get install ssh`
  (this may take a while)
  Install man and info so you can read man pages
- `sudo apt-get install man info`
  Install camera utilities
- `sudo apt-get install libwebcam0`
  (note, as time proceeds, the package name may change. make sure to take a look at)
- `sudo apt-cache search libwebcam` to make sure you're getting the latest version.
- `sudo apt-get install uvcdynctrl`
- `sudo apt-get install streamer`
  Finally, plug in the web cam and query the system to check whether it is present and detected
- `ls /dev/ | grep video` (you should see /dev/video0)
• **lsusb** shows all connected USB devices; the webcam should be among them. Manufacturer and device ID's are available from this output.

## Configuring the operating system environment

It is convenient to set up a couple scripts for commonly used commands. This mainly applies to the minimal beagleboard-xM image that was used in this set of documents.

- **cd ~** (change to home directory)
- **mkdir scripts;cd script** (create scripts directory and cd)
- Use an editor (such as **nano** to create the text files. Use **chmod goa+x filename**

### Ethernet

Ethernet needs to be set up each time. This can be added to ~/.bash_profile instead to have it run every time.

```
sudo dhclient eth0
```

### Ramdisk

Sets up a ramdisk of 64M and mounts it at /tmp/ramdisk

```
sudo mkdir /tmp/ramdisk
sudo chmod 777 /tmp/ramdisk
sudo mount -t tmpfs -o size=64M tmpfs /tmp/ramdisk
```

### USB Camera Linux Command-line

Webcams are handled by a linux library/api called video4linux, v4l for short. Devices which conform to this standard are called "video4linux" devices. When plugged in, if video4linux detects a compatible device, it will mount the webcam under /dev/video0 if it is the first video device, /dev/video1 if it is the second video device, etc.

### streamer

**streamer** records images and video from the webcam.

A sample command to grab one frame and save a jpeg:

```
streamer -c/dev/video1 -s 1920x1080 -j 100 -o /tmp/ramdisk/picture.jpeg
```

- `-c[video4linux device]`: a path to the device. video1 is the second webcam installed in the system.
- `-s [size]`: resolution to capture
- `-j [quality: 1-100]`: specifies the quality of the jpeg. 100 is very little compression
- `-o [filename.jpeg]`: important to name the file with extension .jpeg, the streamer program decides output format based on this.

### uvcdynctrl

**uvcdynctrl** is a command-line interface to adjust the parameters of the webcam.

- **uvcdynctrl -c** lists available controls for a device. eg., **uvcdynctrl /dev/video1 -c**
- **uvcdynctrl -g [control]** gets the value of a control
- **uvcdynctrl -s [control] [value]** sets the value of a control eg., **uvcdynctrl /dev/video1 -s Contrast 64**

Default values will be restored upon reboot.
Example Application: Particle Algorithms Analysis on the Beagle Board

OpenCV
OpenCV is an open sourced computer vision library. It has a large wiki, support-base, and several introductory texts are on sale on Amazon.com. OpenCV is an extensible framework, and algorithm source code examples for all library functions are provided. They can be modified to suit any application.

Survey of OpenCV library functions:

University of Pennsylvania has an Open CV Reference Manual that provides a topical index of the library functions. Code examples are given in C++, which is great for running on the Beagleboard (the GNU toolchain for OMAP includes a C++ compiler.)

The “Image Segmentation, Connected Components and Contour Retrieval” section lists algorithms for the following:

- Contour finding
- Connected component detection
- Pyramid segmentation
- Watershed segmentation