

ラズベリー・パイで画像認識

目次

1 ウェブカメラの接続

Raspberry Pi にウェブカメラを接続します。

1.1 FSWEBCAM のインストール

一般的な USB の Web カメラを使ってラズベリーパイ上で写真や動画を撮るために、fswebcam パッケージをインストールします。SSH で接続した端末から以下のコマンドを実行します。

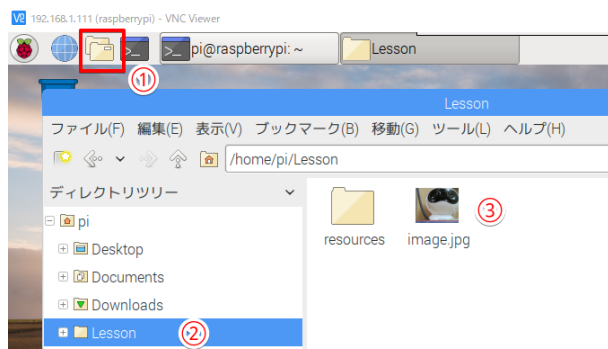
```
sudo apt-get install fswebcam
```

1.2 ウェブカメラのテスト

ウェブカメラで画像が取得できるか確認します。以下のコマンドでウェブカメラから画像が取得できます。

```
cd ~/Lesson  
fswebcam image.jpg
```

取得した画像（ /Lesson/image.jpg ）を VNC ビューワで確認し、画像が取得できているかどうかを確認します。



①クリックすると、ファイルマネージャーが開きます。

②「Lesson」フォルダを指定します。

③クリック（又は右クリックで「イメージビューワ」を選択）すると、画像を確認できます。

※アイコンはイメージの画像でない場合がありますので、イメージビューワで開いて確認します。

（参考）オプションの指定により、画像の解像度の指定や、画像下部のバナーの消去ができます。

```
fswebcam -r 1280x720 --no-banner image2.jpg
```

2 音声出力先の設定

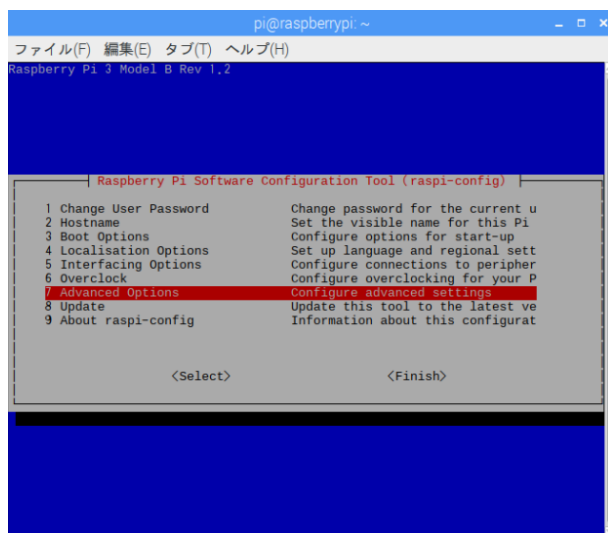
Raspberry Pi にスピーカーを接続します。電源が必要なタイプのスピーカーの場合には、スピーカーに電源を供給します。

2.1 音声出力先の設定

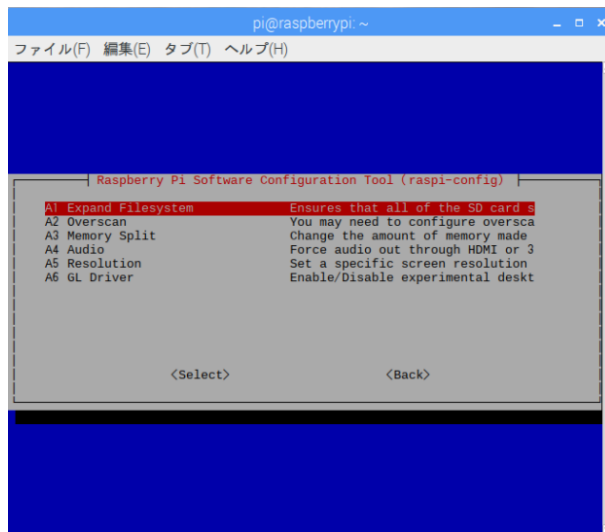
ターミナルを起動して、

```
sudo raspi-config
```

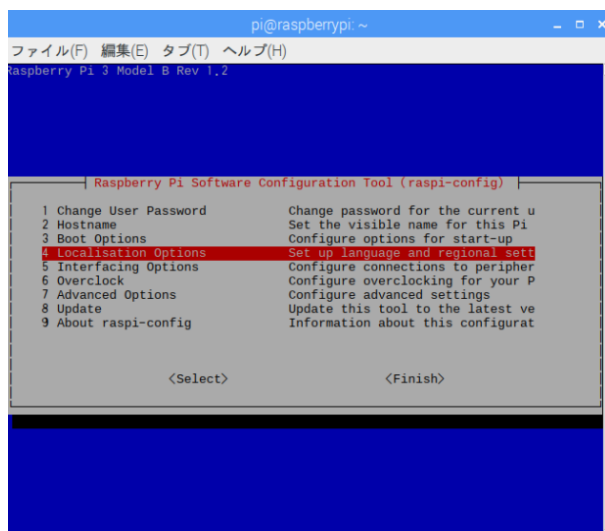
と入力します。



「Advanced Options」を選択して改行します。



「Audio」を選択して改行します。



「Auto」の場合には、HDMI → HeadPhone の順で接続を確認し出力を行います。HeadPhone に外部スピーカーをつなげている場合には、必要に応じて「Force 3.5mm ('headphone') jack」を選択します。

2.2 音声出力先のテスト

```
speaker-test -t sine -f 1000
```

1kHz の正弦波を出力されます。指定したデバイスから音が鳴るかを確認します。外部スピーカーにボリュームが付いている場合には、ボリュームを調整して音がでるようにします。

内蔵の WAV ファイルを用いてテストすることもできます。

```
speaker-test -t wav
```

3 パッケージのインストール

3.1 pip のインストール

python のパッケージ管理ツールである「pip」をインストールします。pip を用いて python のパッケージのインストール等の管理を行います。

```
wget https://bootstrap.pypa.io/get-pip.py
sudo python get-pip.py
```

3.2 virtualenv のインストール

プロジェクト毎に、python を実行する環境を作成し、管理するために、virtualenv と virtualenvwrapper をインストールします。

```
sudo pip install virtualenv virtualenvwrapper
sudo rm -rf ~/.cache/pip
```

エディタで ~/.profile を開いて、WORKON_HOME という環境変数を設定します。※ nano を使っていますが、vim や emacs でもかまいません。

```
nano ~/.profile
```

nano の画面での移動は、上下左右のカーソルキーで行えます。

文字を入力すると、ファイルの編集を行うことができます。

「ctrl+g」でヘルプを表示します。

「ctrl+o」でファイルを保存します。

「ctrl+x」で編集を終了します。

./profile の最後の行に以下を追加します。

```
# virtualenv and virtualenvwrapper
export WORKON_HOME=$HOME/.virtualenvs
source /usr/local/bin/virtualenvwrapper.sh
```

変更を反映させます。

```
source ~/.profile
```

3.3 仮想環境の作成

「cv」という python の仮想環境を作成します。

```
mkvirtualenv cv -p python2
```

仮想環境の作成は、最初の1回のみです。一度環境を作成したら、作業は、その作成した環境に入っていきます

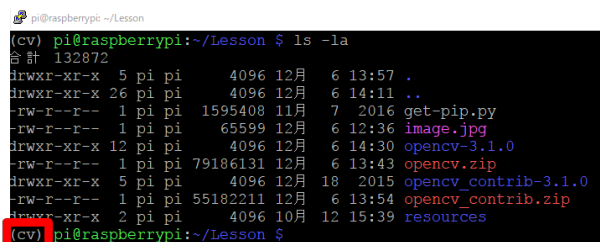
3.4 仮想環境に入る

作成した仮想環境に入ります。。新しいターミナルを開いたり、コンピュータを再起動した場合にも、作成した環境に入ってから作業を行います。同じです。再度、環境を作成する必要はありません。

```
source ~/.profile
workon cv
```

仮想環境に入ると、コマンドラインの先頭に括弧つきで仮想環境の名前が表示されます。

仮想環境 cv 内でインストールされた python パッケージは、cv 内でのみ有効です。またグローバル環境でインストールされた python パッケージは、仮想環境からはアクセスできません。



次のように表示されていることを確認します。「cv」という表示がない場合には、仮想環境 cv に入っていない。仮想環境に入ってから操作します。

3.5 依存関係のあるパッケージのインストール

インストールされているパッケージを更新します。

```
sudo apt-get update
```

ビルドに必要な cmake 等のツールをインストールします。

```
sudo apt-get install build-essential cmake pkg-config
```

画像の入出力に必要なパッケージをインストールします。

```
sudo apt-get install libjpeg-dev libtiff5-dev libjasper-dev libpng12-dev
```

ビデオの入出力に必要なパッケージをインストールします。

```
sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev
libv4l-dev
sudo apt-get install libxvidcore-dev libx264-dev
```

OpenCV は、highgui というサブモジュールを利用して画像の表示などを行っています。このモジュールを利用するのに、libgtk2.0-dev が必要となります。

```
sudo apt-get install libgtk2.0-dev
```

OpenCV 内での行列演算などの最適化のために、以下のモジュールをインストールします。

```
sudo apt-get install libatlas-base-dev gfortran
```

python2.7 と python3 のヘッダーファイルをインストールします。

```
sudo apt-get install python2.7-dev python3-dev
```

3.6 numpy のインストール

仮想環境 cv 内で、Numpy をインストールします。Numpy は、Python の数値計算用パッケージです。インストールには多少時間（12～13分程度）がかかります。

```
pip install numpy
```

3.7 OpenCV のインストール

3.7.1 インストール

OpenCV のインストールを行い、共有ライブラリの依存関係情報を更新します。

```
wget https://github.com/mt08xx/files/raw/master/opencv-rpi/libopencv3_3
.3.1-20171126.2_armhf.deb
sudo apt install -y ./libopencv3_3.3.1-20171126.2_armhf.deb
sudo ldconfig
```

3.7.2 仮想環境内からのリンクの作成

(python2) OpenCV のインストールにより、/usr/local/lib/python2.7/dist-packages にパッケージがインストールされます。

```
ls -l /usr/local/lib/python2.7/dist-packages/
-rw-r--r-- 1 root staff 4294256 11月 27 2017 cv2.so
```

仮想環境 cv 内からもパッケージを参照できるように、リンクを張ります。

```
cd ~/.virtualenvs/cv/lib/python2.7/site-packages/
ln -s /usr/local/lib/python2.7/dist-packages/cv2.so cv2.so
```

3.7.3 インストールの確認

新しいターミナルを開きます。

```
source ~/.profile
workon cv
python
>>> import cv2
>>> cv2.__version__
'3.3.1'
>>> quit()
```

import をしてエラーが出なければ、無事にインストールが来ています。

3.8 TensorFlow のインストール

依存関係のあるモジュールと TensorFlow をインストールします。

```
sudo apt-get install libblas-dev liblapack-dev python-dev libatlas-base
-dev gfortran python-setuptools
pip install http://ci.tensorflow.org/view/Nightly/job/nightly-pi/
lastSuccessfulBuild/artifact/output-artifacts/tensorflow-1.7.0-cp27-
none-any.whl
```

3.9 h5py のインストール

h5py と次の scipy のインストールは、かなり時間がかかります。待っている間に、テキスト「画像データを集める」を試してみてください。

```
sudo apt-get install libhdf5-dev
pip install h5py
```

3.10 scipy のインストール

```
pip install scipy
```

3.11 Keras のインストール

Keras は、TensorFlowなどをバックエンドとして利用する、Deep Learning のフレームワークです。

```
pip install keras
```

3.12 PIL のインストール

```
pip install Pillow
```

以下は、次回の授業の内容です。時間のある方は、試してみてください。

4 InceptionV3 のインストール

inception_v3 のモデルをインストールします。

```
cd ~/Lesson
git clone https://github.com/fchollet/deep-learning-models.git
```

keras のアップデートに伴い、一部修正が必要になっています。inception_v3.py の以下の部分を修正します。

プログラムの修正 inception_v3.py 157行目
(修正前)

```
input_shape = _obtain_input_shape(
    input_shape,
    default_size=299,
    min_size=139,
    data_format=K.image_data_format(),
    include_top=include_top)
```

(修正後) 「include_top=」を「require_flatten=」へ修正

```
input_shape = _obtain_input_shape(
    input_shape,
    default_size=299,
    min_size=139,
    data_format=K.image_data_format(),
    require_flatten=include_top)
```

5 InceptionV3 を使った画像認識

inception_v3 のモデルを使って画像認識を行います。

```
cd ~/Lesson/deep-learning-models
wget https://upload.wikimedia.org/wikipedia/commons/6/63/
    African_elephant_warning_raised_trunk.jpg -O elephant.jpg
python inception_v3.py
Using TensorFlow backend.
Predicted: [[('n02504013', 'Indian_elephant', 0.87686646), ('n01871265',
    'tusk', 0.044712357), ('n02504458', 'African_elephant',
```



```
0.02874627), ('n02398521', 'hippopotamus', 0.0072720782), ('n02092339', 'Weimaraner', 0.0020943223)]]
```

実行すると、学習済みの InceptionV3 のモデルをダウンロードします。デフォルトでは、象の画像の認識を行います。「elephant.jpg」というファイルを用意して、認識させます。画像を解析すると、「ラベル番号・ラベル名・予測値」が多次元配列で返ってきます。

```
wget <画像のURL> -O elephant.jpg
```

で画像を「elephant.jpg」というファイル名で保存できます。プログラムはデフォルトで、「elephant.jpg」というファイル名の画像を認識するようになっています。他の画像を探して、試してみましょう。

6 (参考) motion の停止

他の授業で motion というサービスがインストールされて、自動的に起動している場合があります。その場合、WEB カメラからの画像取得時にリソースが競合してエラーとなります。それを避けるために、サービスを停止します。

サービスの停止

```
sudo systemctl stop motion
```

再起動時に起動しないようにする

```
sudo systemctl disable motion
```

また、取得された画像や動画でディスク容量があふれる場合があります。画像や動画を削除します。

```
sudo rm /var/lib/motion/*
```