Spectrum Technology

V1.0 2020/6/1

抜粋版

Jetsonを使った人認識・物体認識用AI開発キット

～社会的距離のAI分析から自動運転用物体認識までGPU搭載Edgeデバイスの完全活用～

設定・開発編（Jetson版）

スペクトラム・テクノロジー株式会社

https://spectrum-tech.co.jp

sales@spectrum-tech.co.jp

Spectrum Technology

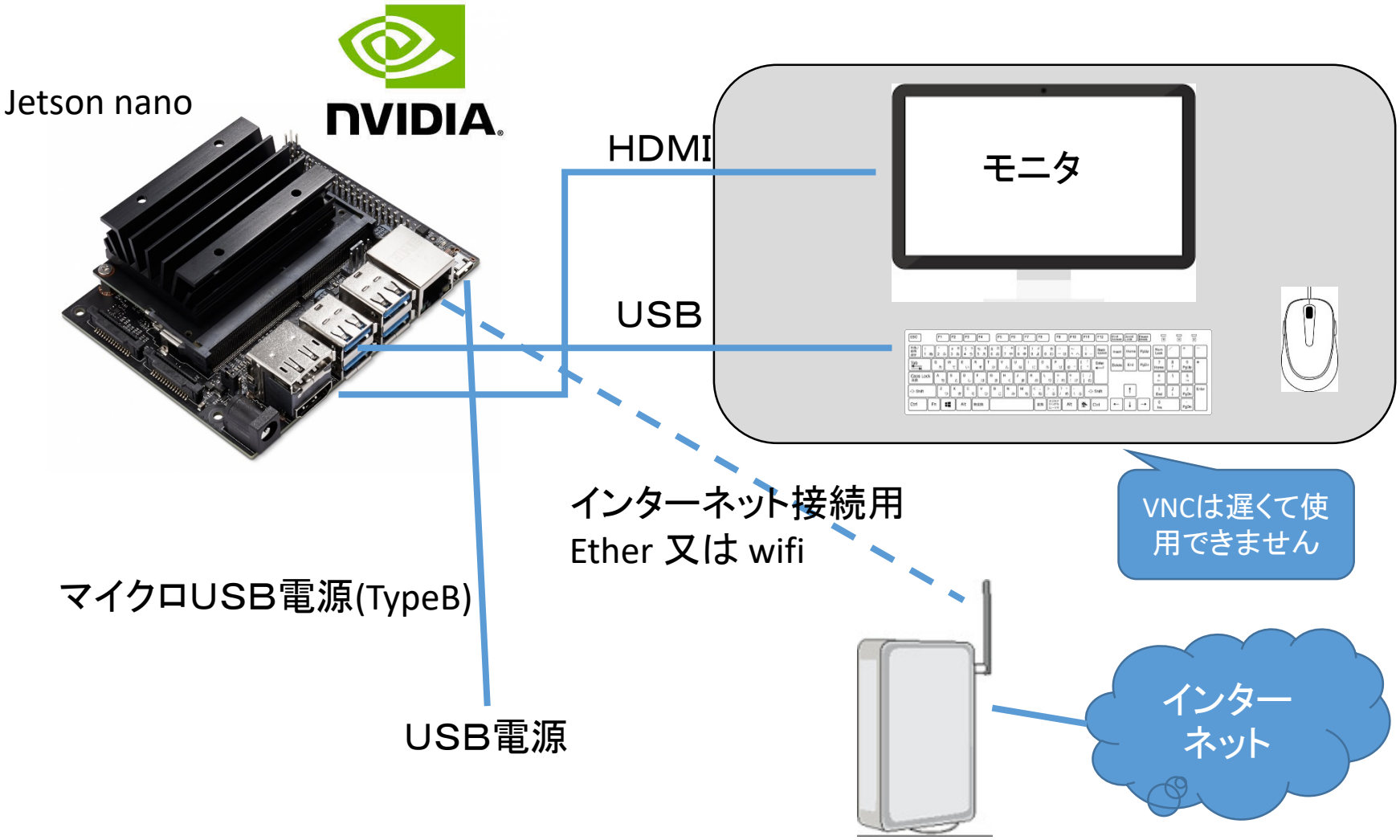
目次

ST Spectrum Technology

目次

ページ

開発キット（Jetson版） 接続構成

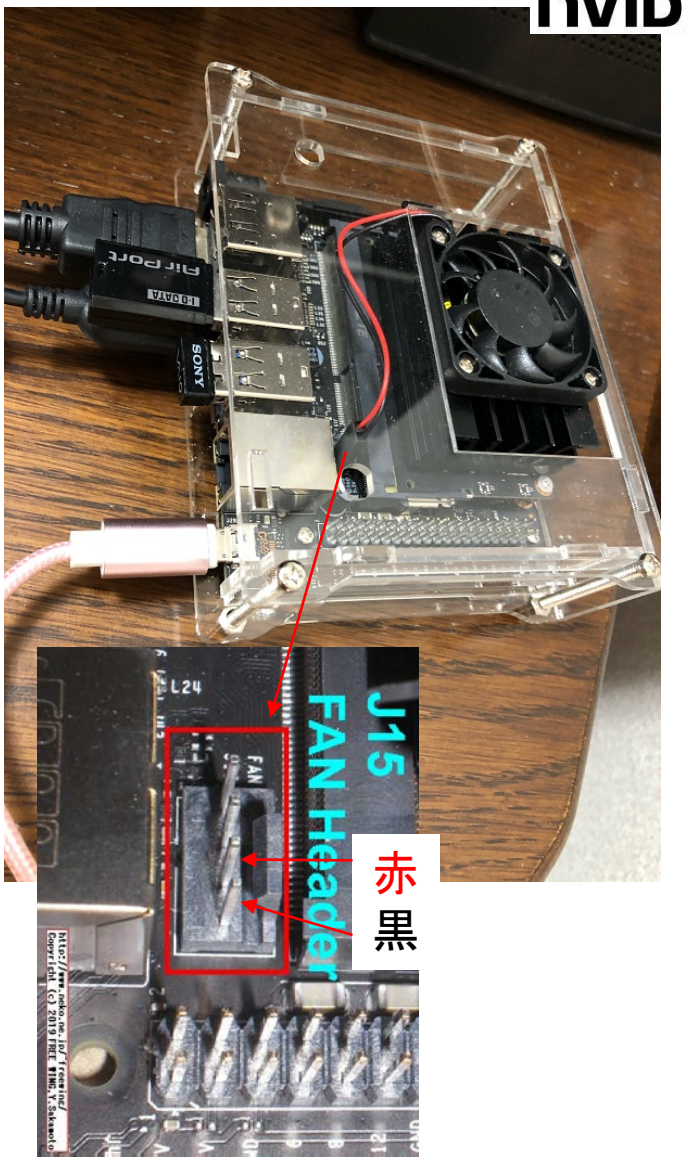


開発キット設定

1. Jetson組立

① FAN取付

- 添付のFANをネジ4本で接続
- メス側は、ラジオペンチ等ではさんで、オスのネジを挿入して、ネジ止めします。
- 電源ケーブルを左側から黒、赤で接続します。



① ケース組立

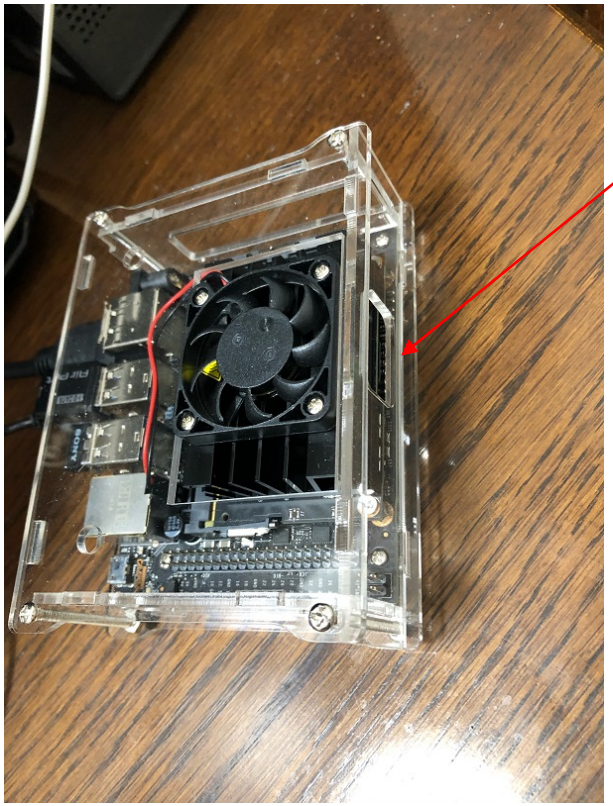
- 内側のネジ4本を先に、スペーサを入れて固定
- 内側のネジ4本が足になります。

- 周りのケース4個と上のケースをはめ込みネジ止めします。
- 注意：USBなどを収容する側のケースは無理すると割れます。

開発キット設定

2.　Jetson起動

①　マイクロSDカードを挿入

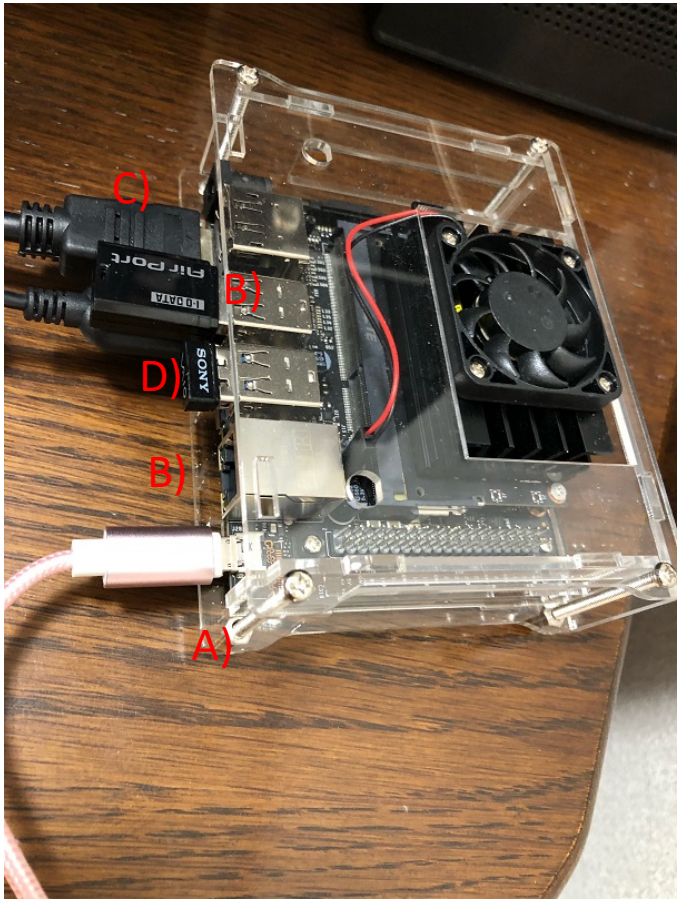


・既にマイクロSDカードは挿入済。
・取り出しは、押すと少し飛び出します。
・挿入時は、金属面を上にして差します。

開発キット設定マニュアル

2. Jetson起動

② 電源、LANケーブル、モニター、キーボード接続



ノートPCの場合は、表示されません

A) USB電源
- TypeBを接続

B) LANケーブル又はWiFiドングル接続
- 左図は、WiFi接続例

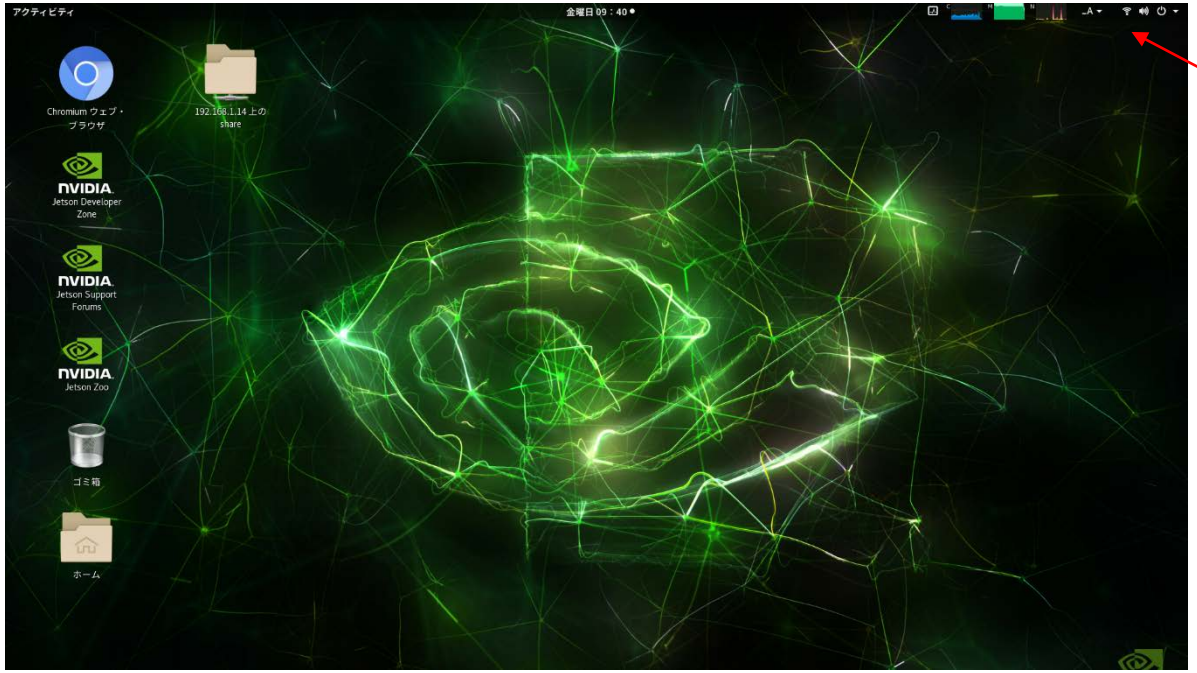
C) モニタ接続
- モニター（TV、PCでHDMI端子のあるもの）を準備します。
- HDMIケーブルにより、Jetsonとモニタを接続します。

D) マウス、キーボード接続
- マウス、キーボードがBluetoothで接続されている場合は、Bluetooth USBと接続します。

開発キット設定

3. Jetson初期設定

② デスクトップ画面



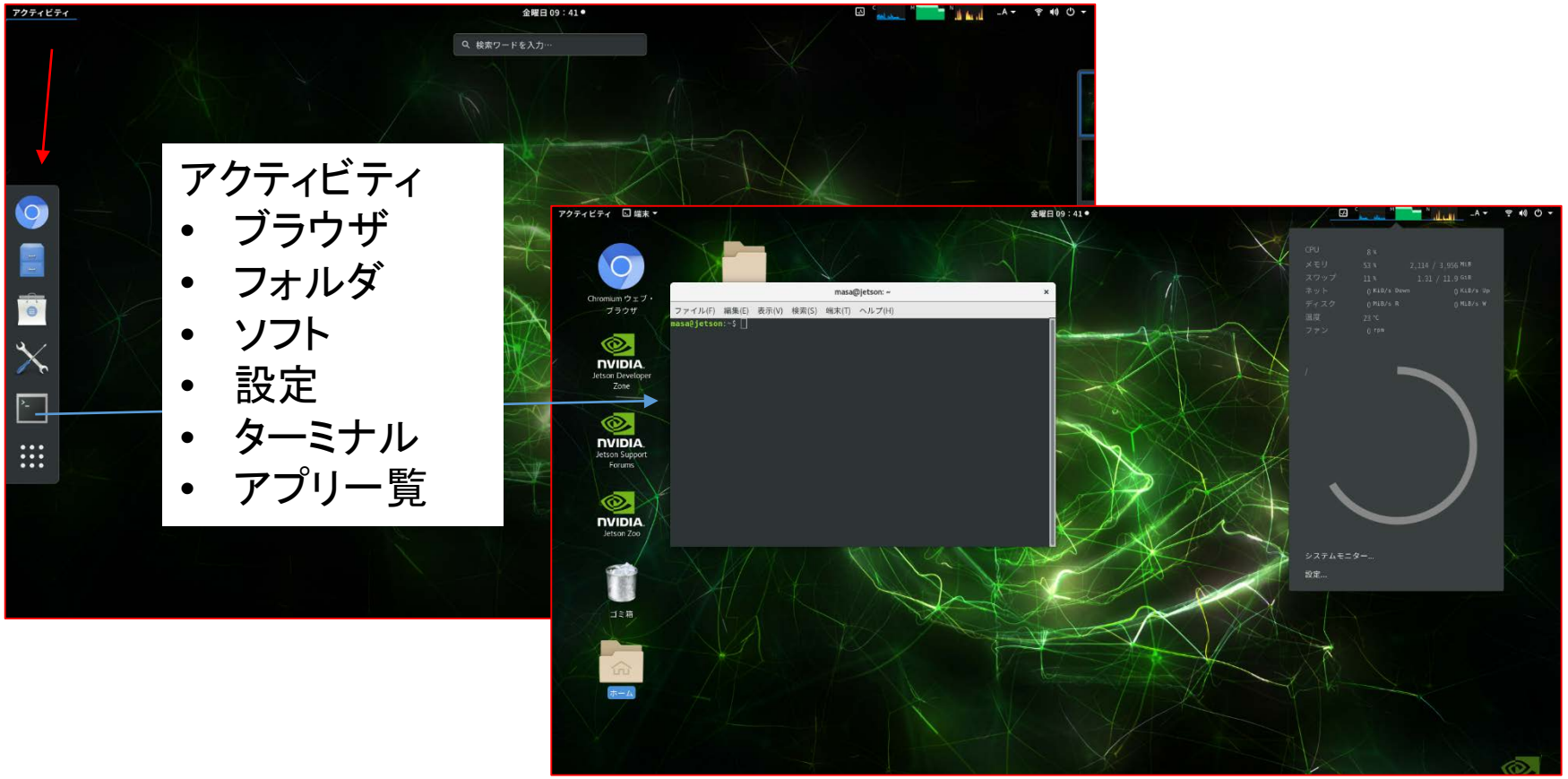
上部アイコン説明

- 左：アクティビティ
  - 設定、ターミナルなど
- 中央：時計、通知
- 右：システムモニタ、日本語入力、回線接続、スピーカ、電源

開発キット設定

3. Jetson初期設定

② デスクトップ画面



開発キット設定

3. Jetson初期設定

② デスクトップ画面

日本語入力

・アイコンで入力方法の切替可能

・全角／半角：英数、日本語切替

・Capslock／英数（Aのとなり）：日本語のかな、英数切替

・変換はスペース、選択はtab

Spectrum Technology

開発キット設定
4. ハードウエア概要

品名	項目	内容	備考
Jetson nano	CPU	Quad-core ARM A57 @ 1.43 GHz	
	GPU	128-core Maxwell	
	メモリ	4GB RA	
	OS	Ubuntu 18.04 LTS	
	インターフェース	1000base-T, USB 3.0x4, 2.0x1, HDMIx2, microSDカード, 40 GPIO pin,I2C, SPI,UART	WiFiは別ドングルが要
	電源／消費電力	Micro USB typeB　2A 5V (通常の使い方では、ダウンすることはない)	別Jackは5V 4A供給可能
	サイズ	115x90x50mm	ケース収納時
付属品		**内容**	**備考**
ケース		透明のみ	
ファン		小型ファン、速度調整はありません。	
microSD 64GB		Ubuntu OS及び必要なプログラム Python3.6, 関連pip 、opencv、tensorRT	
マニュアル		設定・開発編	

USB電源ケーブル(typeB)、HDMIケーブルは付属しておりません。
別途オプション品等を購入ください

開発キット設定

7. オプションハードウエアの設定

①Webカメラ搭載

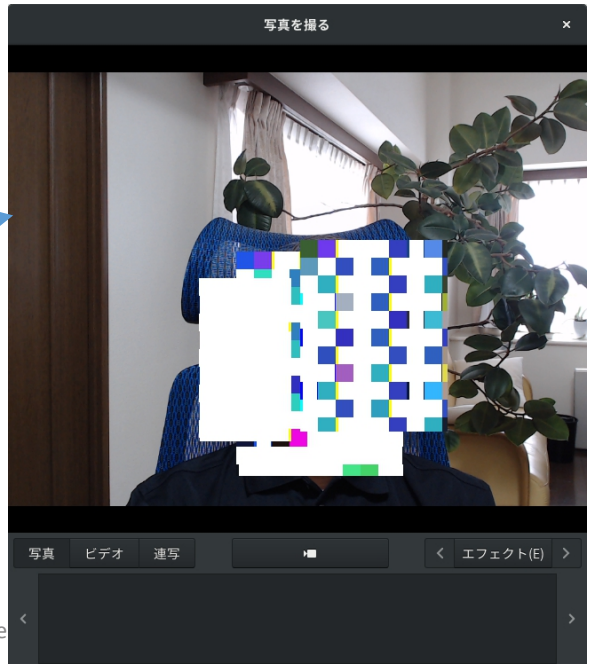
- WebカメラをJetsonにUSBで接続します。
- 設定済です。
- ターミナルで設定確認

$ lsusb

- Cheeseのアプリでカメラを確認

```
ファイル(F)  編集(E)  表
Copyright (C) 2017
This is free softwa                    conditions.  There is NO
warranty; not even                     FOR A PARTICULAR PURPOSE.

masa@jetson:~/openpose/build/caffe/bin$ v4l2-ctl -d /dev/video0 --all
bash: v4l2-ctl: コマンドが見つかりません
masa@jetson:~/openpose/build/caffe/bin$ v4l2-ctl --list-devices
bash: v4l2-ctl: コマンドが見つかりません
masa@jetson:~/openpose/build/caffe/bin$ lsusb
Bus 002 Device 002: ID 0bda:0411 Realtek Semiconductor Corp.
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 008: ID 054c:0688 Sony Corp.
Bus 001 Device 004: ID 046d:082d Logitech, Inc. HD Pro Webcam C920
Bus 001 Device 003: ID 04bb:0947 I-O Data Device, Inc. WN-G150U Wireless LAN Ada
pter
Bus 001 Device 002: ID 0bda:5411 Realtek Semiconductor Corp.
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
masa@jetson:~/openpose/build/caffe/bin$ cheese
```



Ubuntu運用

1. Ubuntuについて

   Linuxの中でも一番シェアの高いOSです。2004年にDebian系から派生。

2. Linux基本コマンド

   ① システム関係

   - 起動：電源を入れると自動で起動します。
   - 再起動：$ reboot

     又は、左上のメニューの「ゲストを再起動
   - 終了： $ shutdown

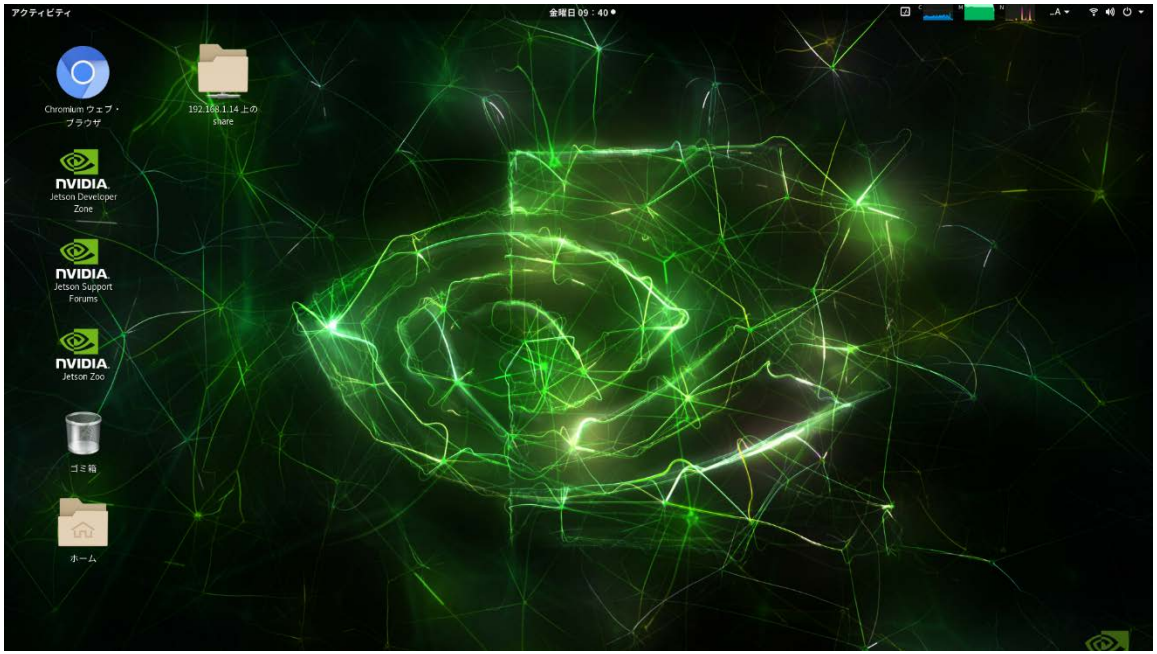
     又は、左上のメニューの「ゲストをシャットダウン」
   - ログアウト $ exit

     ルートからログアウトします
   - 日本語／英語の入力切替：半角／全角のボタン（ESCボタンの下）

Ubuntu運用

3.　基本操作

①　表示画面と内容



主に使用するもの
・ブラウザ：Chrome
・フォルダ：Documents内に必要なファイルがあります。
・コマンド：コマンド画面を立ち上げて、python3のプログラムを動作させます。

Jetson　AI開発

1．概要

- Jetsonは、Nvidiaが提供する、GPUを搭載し、高速化された並列処理用のビジュアルコンピューティングプラットフォームで画像認識、機械学習や自動運転等を想定した組込み用として開発された。CUDAに対応し、Edgeコンピューティング代表格である 。
- また、NVIDIA Jetson Nano は、Jetsonの中でも最小のモデルで、小型で低コスト、かつ低消費電力で、AI デバイス開発を可能にします。エントリレベルのネットワーク ビデオ レコーダー (NVR)、ホーム ロボット、分析機能をフルに搭載したインテリジェント ゲートウェイなど、組み込み IoT アプリケーションの新しい世界を開きます。
- Jetsonプロジェクト：
  https://developer.nvidia.com/embedded/community/jetson-projects

Jetson AI開発
2. 開発メニュー

① Hello AI world
- 画像分類
- 物体認識
- ライブカメラ物体認識
- 物体認識（色分け）
- ライブカメラ物体認識（色分け）
- 転移学習（Pytorch）
- 再学習（TensorRT）
  - 犬・猫編
  - 植物編

② 機械学習事例
- MNIST（Tensorflow）

③ リアルタイム・ポーズ(Trt_pose)
- Jupyter notebook

④ ポーズ（tf-pose-estimation)

⑤ ポーズ（openpose)

⑥ Yolo（Darknet）：写真、動画 物体認識

⑦ コロナ関連
  ① 社会的距離 事例1（ビデオ）
  ② 社会的距離 事例2（ビデオ、webカメラ、Liveカメラ）

Jetson AI開発
3. 開発内容
① Hello AI world

コマンド
$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin
$ python3 imagenet-console.py --network=googlenet images/orange_0.jpg output_0.jpg

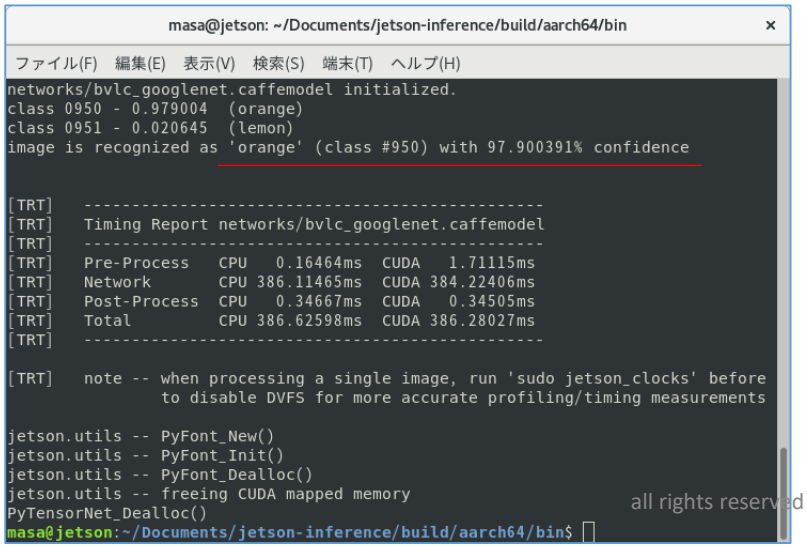
本キットでは、python3のみ利用可能です。

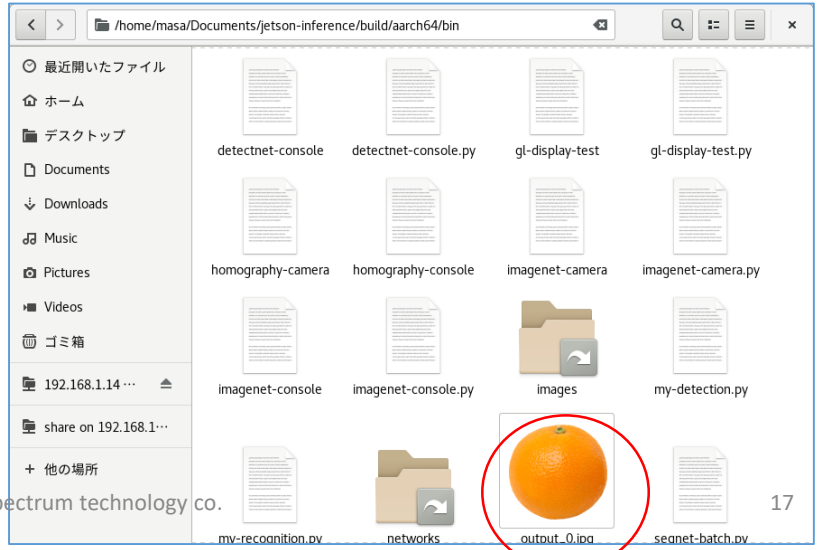
- https://github.com/dusty-nv/jetson-inference#hello-ai-world
- 画像分類 その1 python3
  - モデル：googlenet, ResNet-18を使って画像を分類し、確率を算出します。TensorRT使用
  - 以下のフォルダに移動し、プログラムを動作させます。

$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin

$ python3 imagenet-console.py --network=googlenet images/orange_0.jpg output_0.jpg

  - --network=モデルを指定、images/orange_0.jpg：同じフォルダ内のimageのorange_0.jpgの写真を使い、確率を算出
  - Orangeで97%の確率で出力。1回目のモデル使用には、5分位学習にかかります。





Spectrum Technology

Jetson AI開発
3. 開発内容
① Hello AI world

コマンド
$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin
$ ./imagenet-console --network=googlenet images/orange_0.jpg output_00.jpg

- https://github.com/dusty-nv/jetson-inference#hello-ai-world
- 画像分類 その1 cpp
  - モデル：googlenet, ResNet-18を使って画像を分類し、確率を算出します。TensorRT使用
  - 以下のフォルダに移動し、プログラムを動作させます。

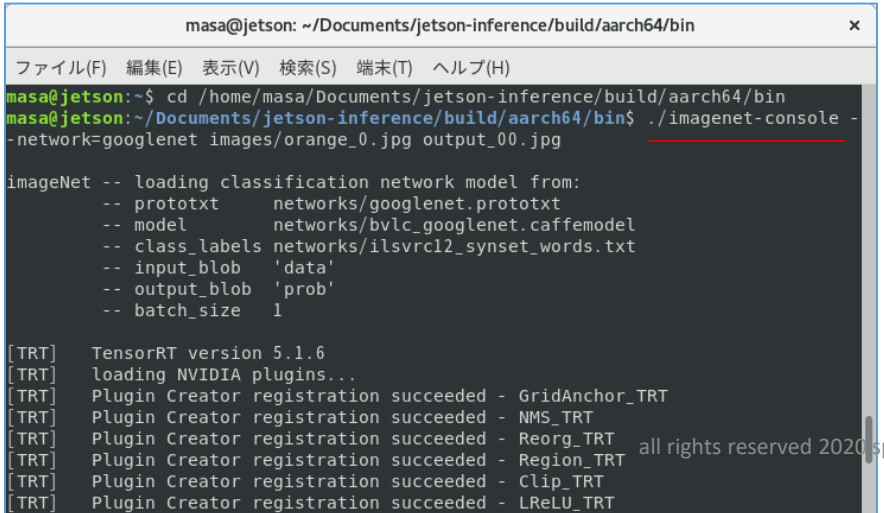
$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin

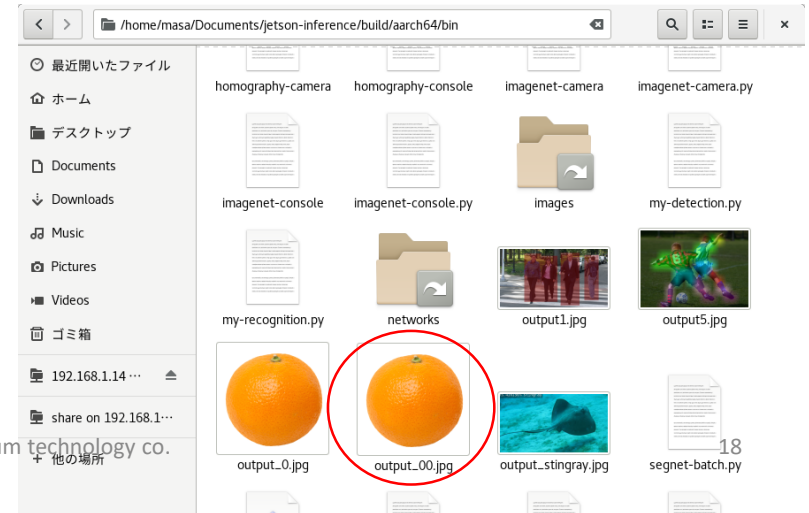
$ ./imagenet-console –network=googlenet images/orange_0.jpg output_00.jpg　CPPの場合、結果は同じ。

プログラム内容を確認する方は、cppの方が簡単です。

ーコンパイル前のimagenet-console.cpp　cd /home/masa/Documents/jetson-inference/examples/imagenet-cosole/ 内にあります。

関連のプログラムは、cd /home/masa/Documents/jetson-inference/build/aarch64/include にあります。





Jetson AI開発
3. 開発内容
① Hello AI world

コマンド
$ cd /home/masa/Documents/jetson-inference/build/
$ ./download-models.sh

- https://github.com/dusty-nv/jetson-inference#hello-ai-world
- 画像分類 その1 モデル
  - モデル:googlenet, ResNet-18を使って画像を分類し、確率を算出します。TensorRT使用
  - 他のモデルを使用したい場合、個別にダウンロードします。ファイルが巨大な場合がありますので、時間がかかることを考慮してください。

  $ cd /home/masa/Documents/jetson-inference/build/

  $ ./download-models.sh

Network	CLI argument	NetworkType enum
AlexNet	alexnet	ALEXNET
GoogleNet	googlenet	GOOGLENET
GoogleNet-12	googlenet-12	GOOGLENET_12
ResNet-18	resnet-18	RESNET_18
ResNet-50	resnet-50	RESNET_50
ResNet-101	resnet-101	RESNET_101
ResNet-152	resnet-152	RESNET_152
VGG-16	vgg-16	VGG-16
VGG-19	vgg-19	VGG-19
Inception-v4	inception-v4	INCEPTION_V4

Spaceで選択して、ダウンロードしてください。
左はimage分だけですが、他に物体識別、市街地、複数人物などたくさんの事前学習済モデルが準備されています。

Spectrum Technology

Jetson AI開発
3. 開発内容
① Hello AI world

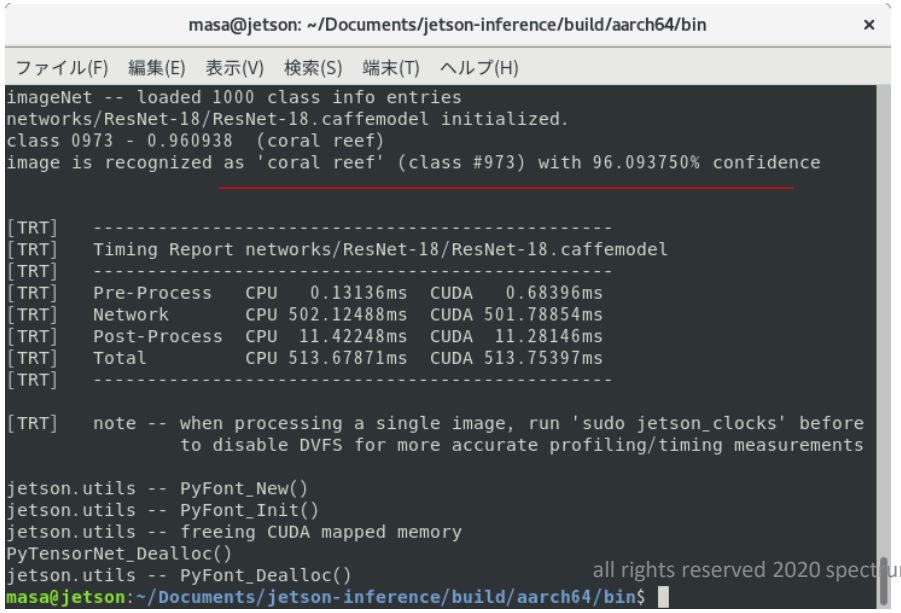
コマンド
$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin
$ python3 imagenet-console.py --network=resnet-18 images/coral.jpg output_coral.jpg

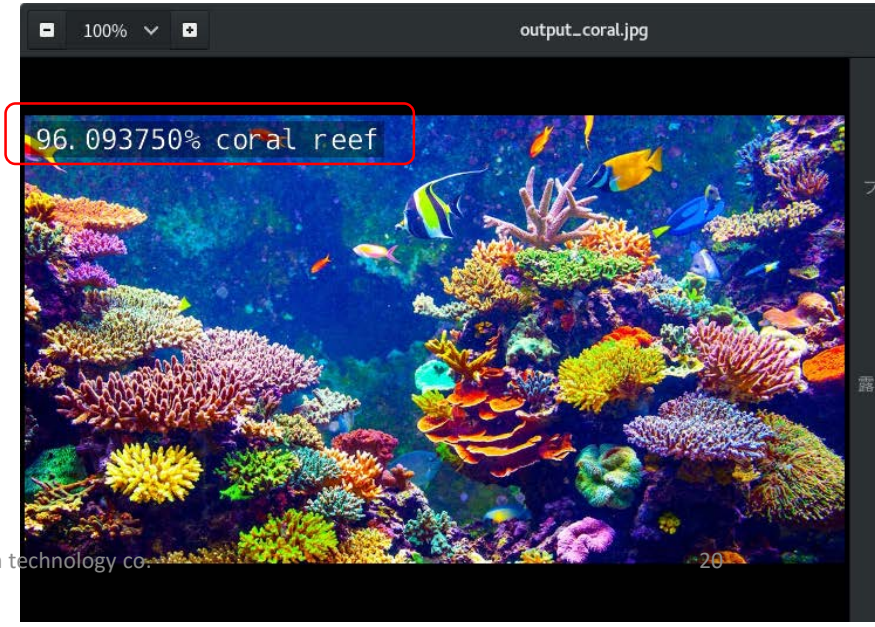
- https://github.com/dusty-nv/jetson-inference#hello-ai-world
- 画像分類 その3 python3、resnet－18
  - モデル：googlenet, ResNet-18を使って画像を分類し、確率を算出します。TensorRT使用
  - 以下のフォルダに移動し、プログラムを動作させます。

$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin

$ python3 imagenet-console.py --network=resnet-18 images/coral.jpg output_coral.jpg

  - --network=モデルを指定、同じフォルダ内のimageのcoral.jpgの写真を使い、確率を算出
  - Coral reefで96%の確率で出力。1回目のモデル使用には、5分位学習にかかります。





Jetson AI開発
3. 開発内容
① Hello AI world

コマンド
$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin
$ python3 detectnet-console.py --network=ssd-mobilenet-v2 images/peds_0.jpg output1.jpg

- https://github.com/dusty-nv/jetson-inference/blob/master/docs/detectnet-console-2.md
- 物体認識 その1 歩行者 python3
  - モデル：ssd-mobilenet-v2（91種類の物体を認識、詳細はこちら）を使って、物体を認識し、確率を算出します。TensorRT使用
  - 以下のフォルダに移動し、プログラムを動作させます。

$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin

$ python3 detectnet-console.py --network=ssd-mobilenet-v2 images/peds_0.jpg output1.jpg

  - 90%以上の確率で人物を認識。

Jetson AI開発
3. 開発内容
① Hello AI world

コマンド
$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin
$ python3 segnet-console.py --network=fcn-resnet18-cityscapes images/city_0.jpg output4.jpg

- https://github.com/dusty-nv/jetson-inference#hello-ai-world
- 物体色分け 市街地景色 python3
  - モデル: fcn-resnet18-cityscapes（市街地の景色）を使って物体を識別し、色分けします。TensorRT使用
  - 以下のフォルダに移動し、プログラムを動作させます。

$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin

$ python3 segnet-console.py --network=fcn-resnet18-cityscapes images/city_0.jpg output4.jpg

  - 右のクラスで画像を色分けします。自動運転などに利用します。

Jetson AI開発
3. 開発内容
① Hello AI world

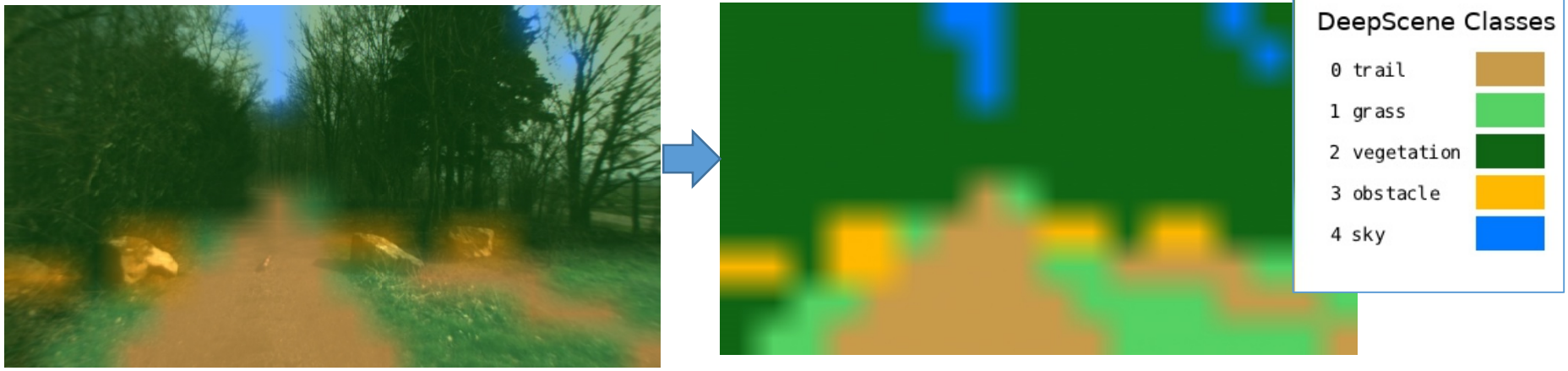
コマンド
$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin
$ python3 segnet-console.py --network=fcn-resnet18-deepscene --visualize=mask images/trail_0.jpg output_mask.jpg

- https://github.com/dusty-nv/jetson-inference#hello-ai-world
- 物体色分け オフロード景色 python3
  - モデル: fcn-resnet18-deepscene(オフロードを走る森の景色)を使って物体を識別し、色分けします。TensorRT使用
  - 以下のフォルダに移動し、プログラムを動作させます。

$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin

$ python3 segnet-console.py --network=fcn-resnet18-deepscene --visualize=mask images/trail_0.jpg output_mask.jpg

  - 右のクラスで画像を色分けします。オフロードの自動運転などに利用します。



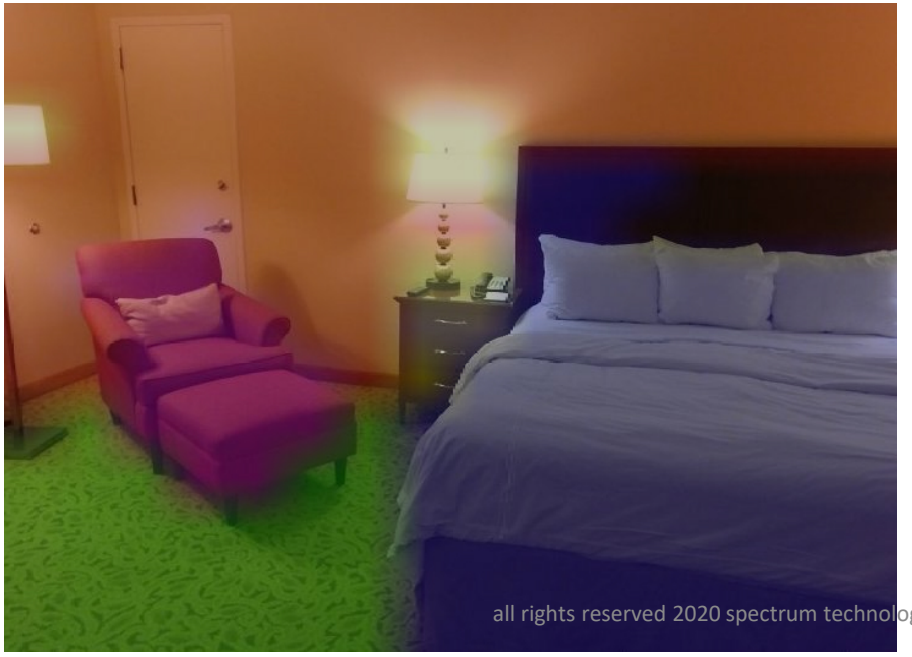
Jetson AI開発
3. 開発内容
① Hello AI world

コマンド
$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin
$ python3 segnet-console.py --network=fcn-resnet18-sun images/room_0.jpg output7.jpg

- https://github.com/dusty-nv/jetson-inference#hello-ai-world
- 物体色分け 屋内の家具、壁 python3
  - モデル: fcn-resnet18-sun（屋内の家具、壁）を使って物体を識別し、色分けします。TensorRT使用
  - 以下のフォルダに移動し、プログラムを動作させます。

$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin

$ python3 segnet-console.py --network=fcn-resnet18-sun images/room_0.jpg output7.jpg



Spectrum Technology

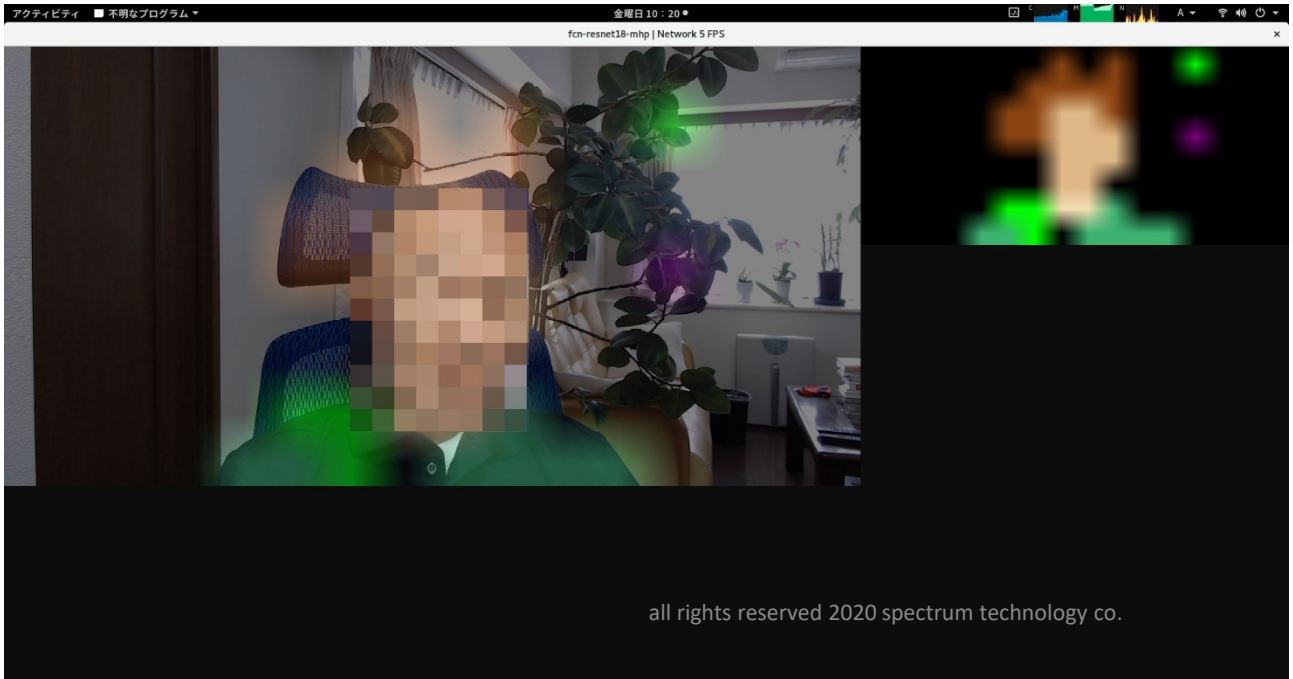
Jetson AI開発
3. 開発内容
① Hello AI world

コマンド
$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin
$ python3 segnet-camera.py --network=fcn-resnet18-mhp --camera=/dev/video0

- https://github.com/dusty-nv/jetson-inference#hello-ai-world
- ライブカメラ物体色分け　複数人数　python3
  - モデル：fcn-resnet18-mhp（複数人数）を使って物体を識別し、色分けします。TensorRT使用
  - 以下のフォルダに移動し、プログラムを動作させます。

$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin

$ python3 segnet-camera.py --network=fcn-resnet18-mhp --camera=/dev/video0



Spectrum Technology

Jetson AI開発
3. 開発内容
① Hello AI world

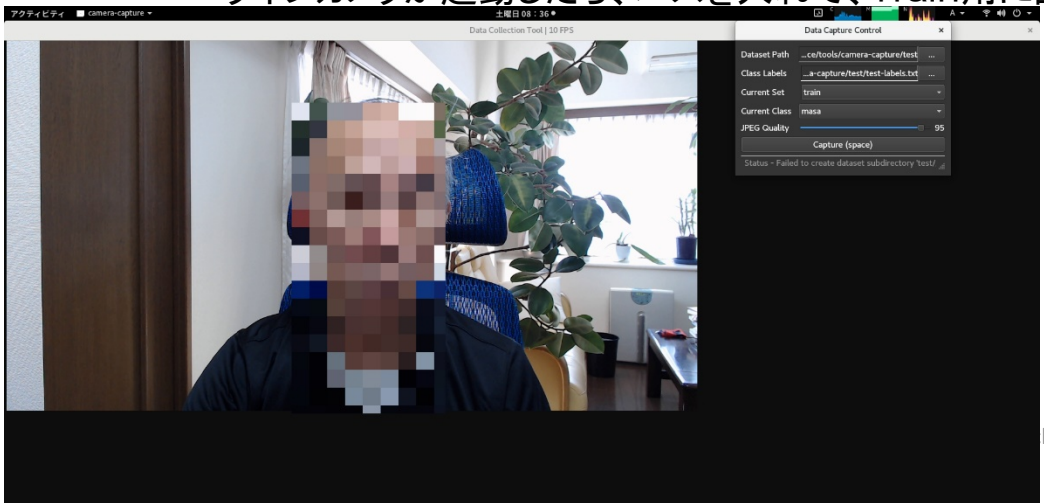
コマンド
$ cd /home/masa/Documents/jetson-inference/tools/camera-capture/
$ camera-capture --camera=/dev/video0

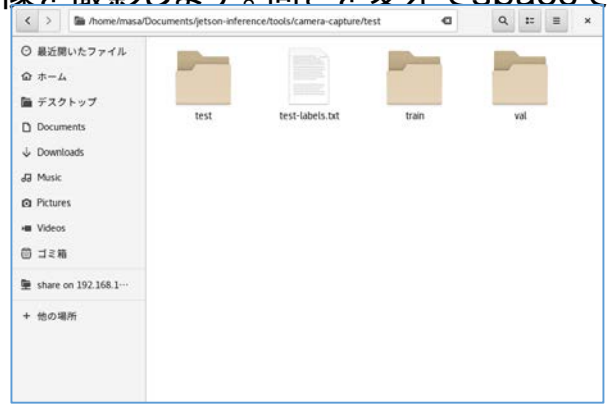
- https://github.com/dusty-nv/jetson-inference/blob/master/docs/pytorch-transfer-learning.md
- 転移学習 Pytorchを使用、自己データ作成
  - 再学習：Pytorchを使って再学習させます。
  - モデルは、ResNet-18。
  - 以下のフォルダに移動し、プログラムを動作させます。

  $ cd /home/masa/Documents/jetson-inference/tools/camera-capture/
  - 自己データ作成方法

  $ camera-capture --camera=/dev/video0
  - 例：testフォルダを事前に作成し、ラベルLabels.txtを作り、ラベル内に分類を作成しておきます。
  - ライブカメラが起動したら、パスを入れて、Train用に画像を撮影します。向きを変えてspaceで作成

少なくとも100枚の画像を取得





コマンド
$ cd /home/masa/Documents/mnist
$

Jetson AI開発
3. 開発内容
② 機械学習事例

TensorFlow

- MNIST Tensorflow－gpu
  - TensorRTの他に本格利用のためのTensorflow－gpuをインストール。
  - MNISTは0から9までの手書き数字を認識するために、学習用：6万枚の画像、試験用：1万枚の画像を準備し、学習した内容でどれくらいの精度で読み取りが可能となるかのプログラム。
  - モデル：ソフトマックス回帰（出力yの10個のうちで近いものに1に近い数字を出力する）
  - 以下のフォルダに移動して、プログラムを動作

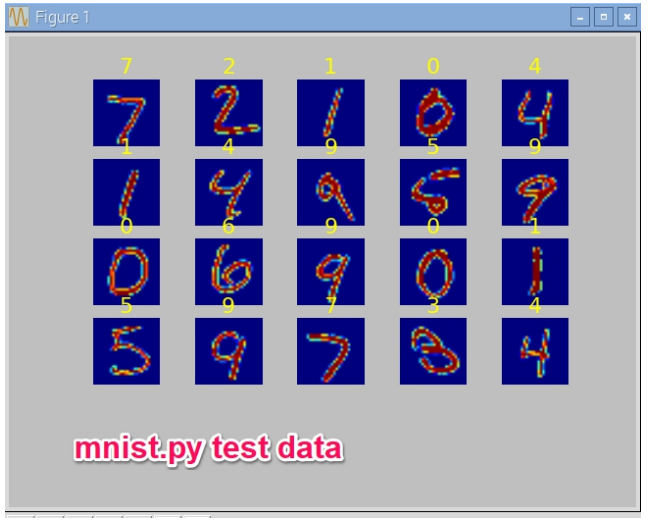
$ cd /home/masa/Documents/mnist

以下の3個のプログラムで1万回、学習し、回答を出します。（本来は、10万回になります。プログラムを修正してください。結果をtensorboardのブラウザで表示

A) mnist.py：正解率0.9 tensorboard --logdir=data1

B) mnist_softmax.py：正解率0.92 tensorboard --logdir=data2

C) mnist_deep.py：正解率：0.99？ tensorboard --logdir=data3



Spectrum Technology

Jetson　AI開発
3. 開発内容
③　リアルタイムポーズ

コマンド
$ cd
/home/masa/Documents/trt_pose/tasks/human_pose
$ python3 main.py

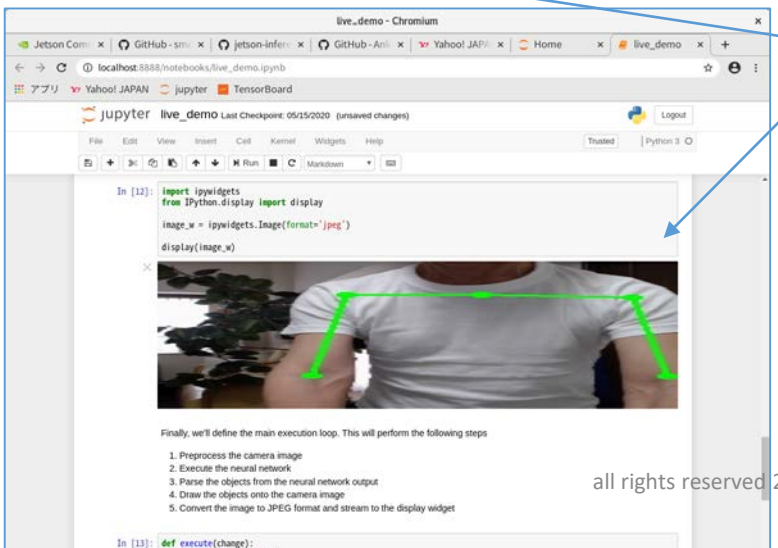
- Trt__pose
  - https://github.com/NVIDIA-AI-IOT/trt_pose
  - TensorRTを使って、人の動きをリアルタイムに表示します。モデルの変換にtorch2trt（Pytorch からtensorRT）を使用、また学習で使用するJupyter　notebookも説明します。
  - モデル：MSCOCOを使用
  - 以下のフォルダに移動して、プログラムを動作

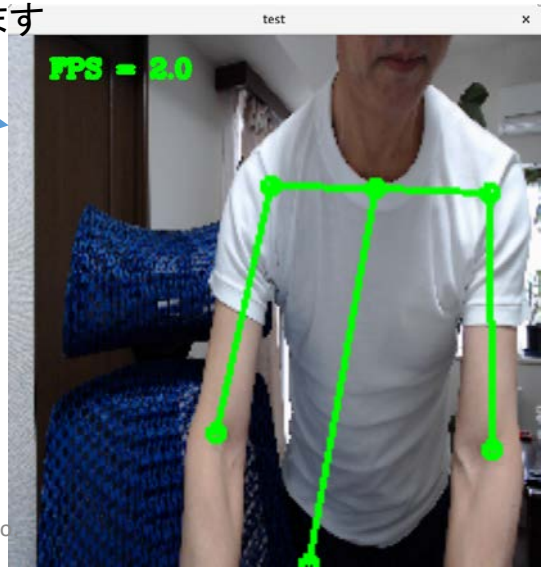
$　cd /home/masa/Documents/trt_pose/tasks/human_pose

$ jupyter notebook

- live_demo.ipynbを起動します。1行ずつrun途中で動画

$ python3 main.py　プログラムで動作、5分位表示にかかります





コマンド
$ cd /home/masa/tf-pose-estimation
$ python3 run.py --model=mobilenet_thin --resize=432x368 --image=./images/p1.jpg

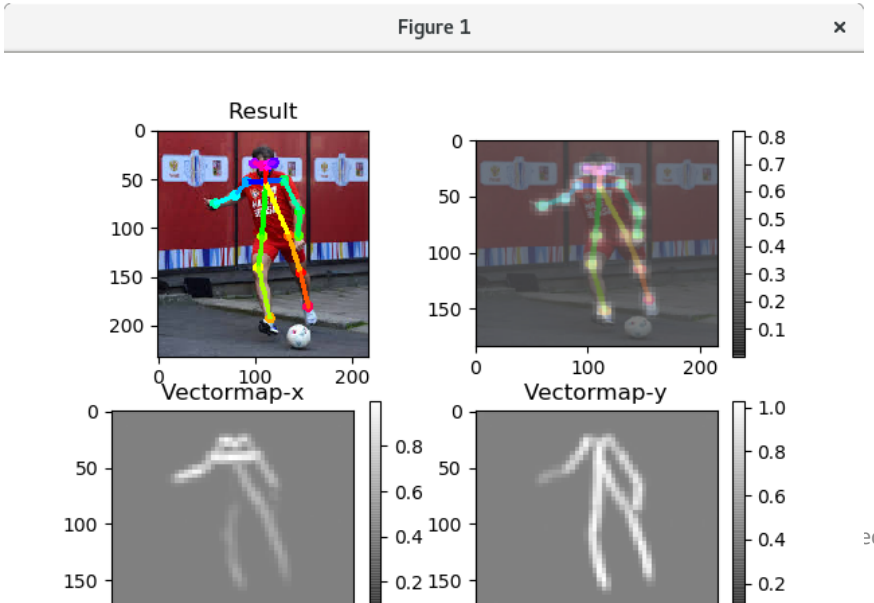
Jetson AI開発
3. 開発内容
④ ポーズ(TF_pose写真)

- TF__pose__estimation
  - https://github.com/ildoonet/tf-pose-estimation
  - Tensorflowが提供するポーズのプログラムで、Apache ライセンスで使用できます。カラーで表示されるのでわかりやすい。
  - モデル:Mobilenet等を使用
  - 以下のフォルダに移動して、プログラムを動作

$ cd /home/masa/tf-pose-estimation

- 写真

$ python3 run.py --model=mobilenet_thin --resize=432x368 --image=./images/p1.jpg



コマンド
$ cd /home/masa/openpose
$ ./build/examples/openpose/openpose.bin --model_folder ./models --net_resolution 320x176

Jetson　AI開発
3．開発内容
⑤　ポーズ（openposeカメラ）

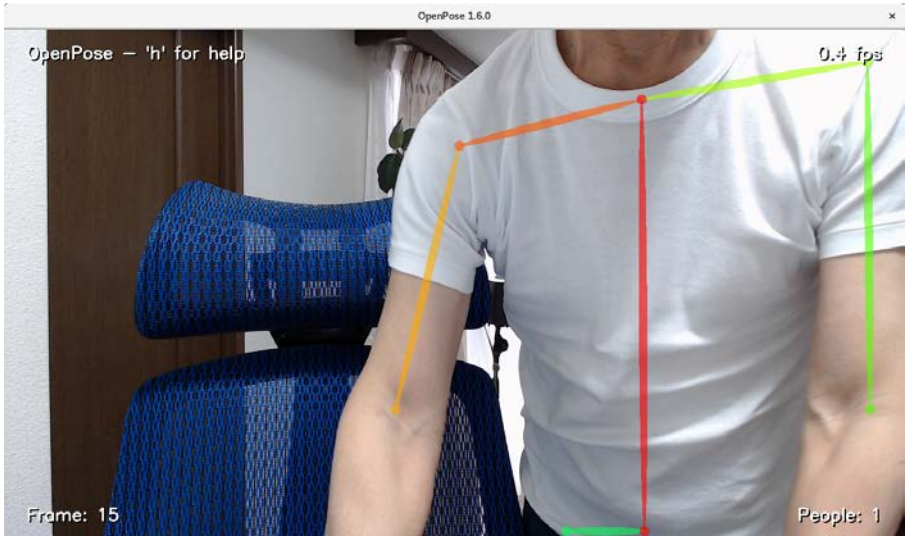
- Openpose
  - https://github.com/CMU-Perceptual-Computing-Lab/openpose
  - 一番メジャなポーズ用のプログラム。複数の人をリアルタイムに処理できる。有料ライセンス。
  - モデル：個別に準備されている
  - 以下のフォルダに移動して、プログラムを動作

$　cd /home/masa/openpose

  - ライブカメラ

$ ./build/examples/openpose/openpose.bin --model_folder ./models --net_resolution 320x176

  - かなり重い



Spectrum Technology

Jetson AI開発
3. 開発内容
⑥ Yolo（Darknet）

コマンド
$ cd /home/masa/darknet
$ ./darknet detector test cfg/coco.data cfg/yolov3.cfg yolov3.weights

- 写真 物体認識
  - https://github.com/AlexeyAB/darknet
  - Darknetが提供するYoloは、一番高速で物体認識をするモデルです。TEDで紹介されています。
  - モデル：Yolov3。前出のJetsonでも使われています
  - 以下のフォルダに移動して、プログラムを動作

$ cd /home/masa/darknet

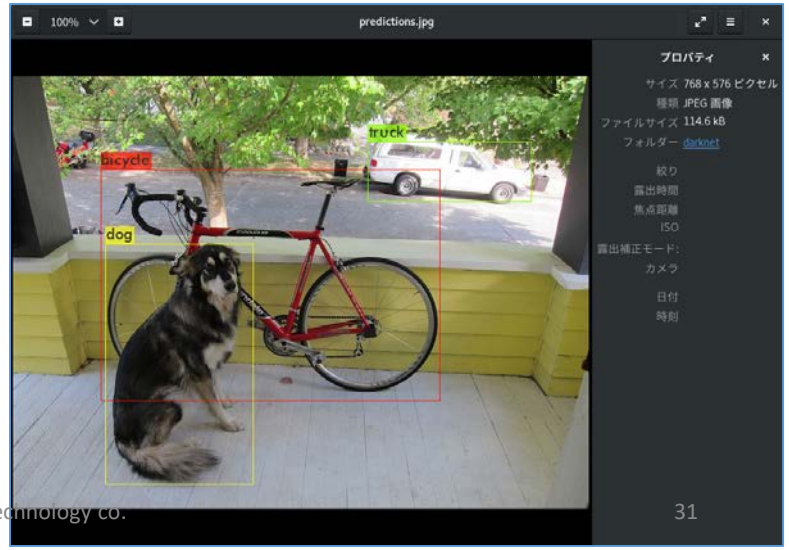
- 写真

$ ./darknet detector test cfg/coco.data cfg/yolov3.cfg yolov3.weights

$ enter imagepath: data/dog.jpgと入力します。

```
masa@jetson: ~/Documents/darknet
ファイル(F)  編集(E)  表示(V)  検索(S)  端末(T)  ヘルプ(H)
 100 conv    256       3 x 3/ 1     52 x  52 x 128 ->   52 x  52 x 256 1.595 BF
 101 conv    128       1 x 1/ 1     52 x  52 x 256 ->   52 x  52 x 128 0.177 BF
 102 conv    256       3 x 3/ 1     52 x  52 x 128 ->   52 x  52 x 256 1.595 BF
 103 conv    128       1 x 1/ 1     52 x  52 x 256 ->   52 x  52 x 128 0.177 BF
 104 conv    256       3 x 3/ 1     52 x  52 x 128 ->   52 x  52 x 256 1.595 BF
 105 conv    255       1 x 1/ 1     52 x  52 x 256 ->   52 x  52 x 255 0.353 BF
 106 yolo
[yolo] params: iou loss: mse (2), iou_norm: 0.75, cls_norm: 1.00, scale_x_y: 1.0
0
Total BFLOPS 65.879
avg_outputs = 532444
 Allocate additional workspace_size = 13.11 MB
Loading weights from yolov3.weights...
 seen 64, trained: 32013 K-images (500 Kilo-batches_64)
Done! Loaded 107 layers from weights-file
Enter Image Path: data/dog.jpg

data/dog.jpg: Predicted in 1933.391000 milli-seconds.
bicycle: 99%
dog: 100%
truck: 94%
Enter Image Path: Cannot load image
```



コマンド
$ cd /home/masa/Documents/SocialDistancingAI
$ python3 main.py --videopath "vid_short.mp4"

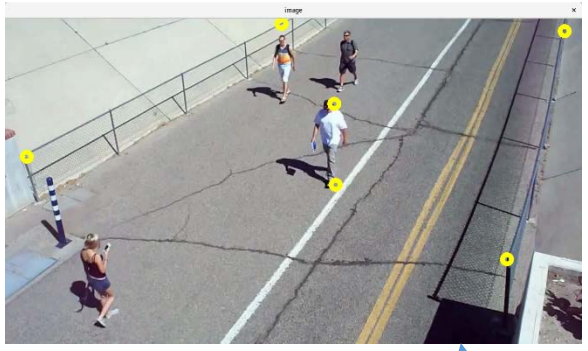
Jetson　AI開発
3. 開発内容
⑦　コロナ関連

- 社会的距離　事例1
  - https://github.com/aqeelanwar/SocialDistancingAI
  - ビデオ内の社会的距離（約2m）を、可視化し、俯瞰図とフレーム内の件数（距離違反、OK）を算出します。
  - モデル：Coco
  - 作成者：GorgiaTech　Aqeel Anwar博士
  - ライセンス：MIT　ライセンス（商用利用可）
  - 以下のフォルダに移動して、プログラムを動作

$　cd /home/masa/Documents/SocialDistancingAI

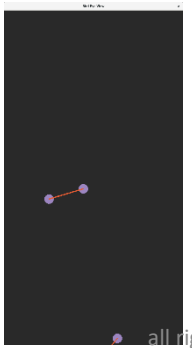
  - ライブカメラ

$ python3 main.py --videopath “vid_short.mp4“　　ビデオパス配下は、分析するビデオ名

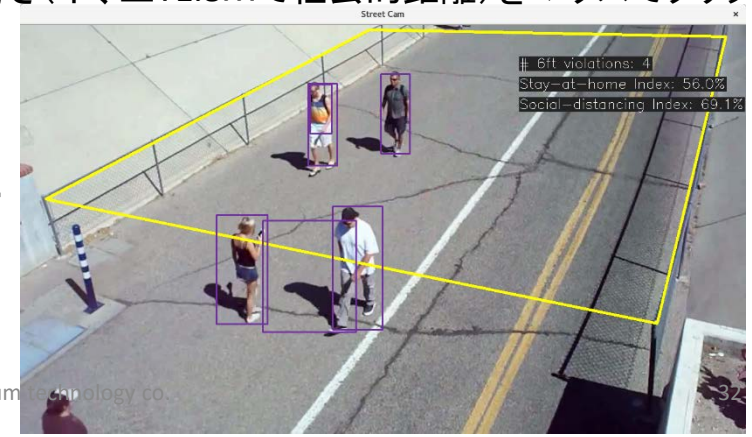


・使い方：6点（左下、右下、左上、右上、人間の大きさ（下、上：1.8mで社会的距離）をマウスでクリック。7回目のクリックは画像のどの位置でもOK。動作開始。

俯瞰図



ビデオ



Jetson AI開発
3. 開発内容
⑦ コロナ関連

コマンド
$ cd /home/masa/Documents/Social-Distancing-Analyser-COVID-19
$ python3 social_distancing_analyser_2.py

- 社会的距離 事例2
  - https://github.com/Ank-Cha/Social-Distancing-Analyser-COVID-19
  - ビデオ、Webカメラ等の社会的距離（約1m，約2m）を、可視化し、接近するとフレームの色を変えます。また、フレーム内の件数（距離違反、OK）を算出します。
  - モデル：Yolov3
  - 作成者：Ankush Chaudhariさん
  - ライセンス：許可が必要（商用不可）
  - 以下のフォルダに移動して、プログラムを動作

$ cd /home/masa/Documents/Social-Distancing-Analyser-COVID-19

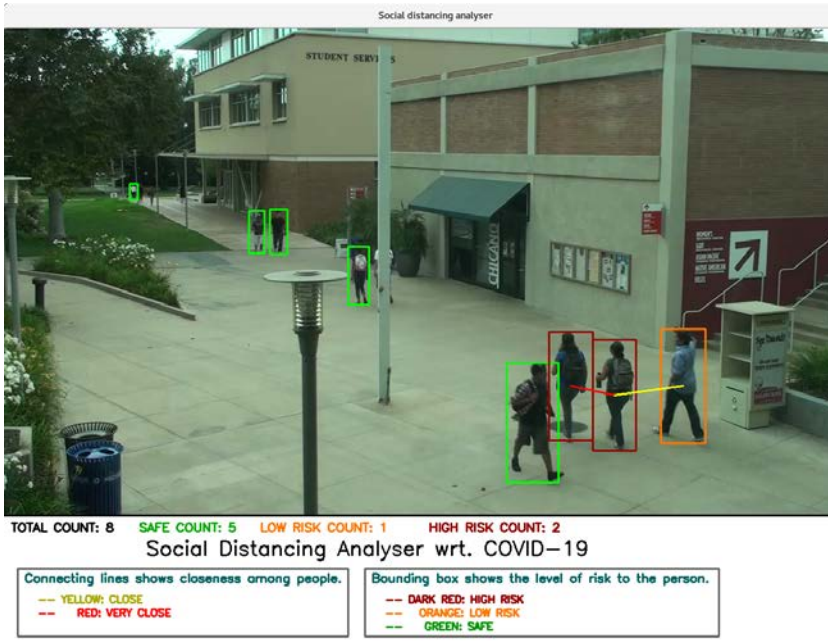
- ビデオ

$ python3 social_distancing_analyser_2.py

Video name： video.mp4 ビデオパス配下は、分析するビデオ名

・分析後op_ビデオ名のファイルがでます。mp4の場合は、PCなどのGOMプレイヤで再生してください。Aviのみ対応しています。処理は、1時間位かかります。

処理を早くするには、プログラムの中段位のtinyを使ってください。但し、人間の大きさが縮小されます。



Jetson AI開発
3. 開発内容
⑦ コロナ関連

コマンド
$ cd /home/masa/Documents/Social-Distancing-Analyser-COVID-19
$ python3 webcam_social_distancing_analyser_2tiny.py

- 社会的距離 事例2
  - https://github.com/Ank-Cha/Social-Distancing-Analyser-COVID-19
  - ビデオ、Webカメラ等の社会的距離（約1m，約2m）を、可視化し、接近するとフレームの色を変えます。また、フレーム内の件数（距離違反、OK）を算出します。
  - モデル：Yolov3
  - 作成者：Ankush Chaudhariさん
  - ライセンス：許可が必要（商用不可）
  - 以下のフォルダに移動して、プログラムを動作

$ cd /home/masa/Documents/Social-Distancing-Analyser-COVID-19

  - Webカメラ

$ python3 webcam_social_distancing_analyser_2tiny.py
立ち上がりに10分位。重い

・ライブカメラ

$ python3 livecam_social_distancing_analyser_2.py 他の高速マシンで実施してください。