Monocular depth estimation

1.Introduction

Depth sensing is useful for tasks such as map drawing, navigation, and obstacle detection, but historically it requires stereo cameras or RGB D cameras. Nowadays, DNN can infer relative depth (also known as monocular depth) from a single monocular image. Please refer to the FastDepth paper from the Massachusetts Institute of Technology for a method of using fully convolutional networks (FCNs) to achieve this goal.

The depthNet object accepts a monochrome image as input and outputs a depth map. The depth map is colored for visualization, but the original depth field can also be used to directly access depth. DepthNet can be used from Python and C++. As an example of using the depthNet class, we provide sample programs for C++and Python:

2.Mono depth on image

Firstly, let's try running the depthnet example on some sample images. In addition to the input/output path, there are also some additional command-line options that are optional:

- --Network changes the network flag of the depth model being used

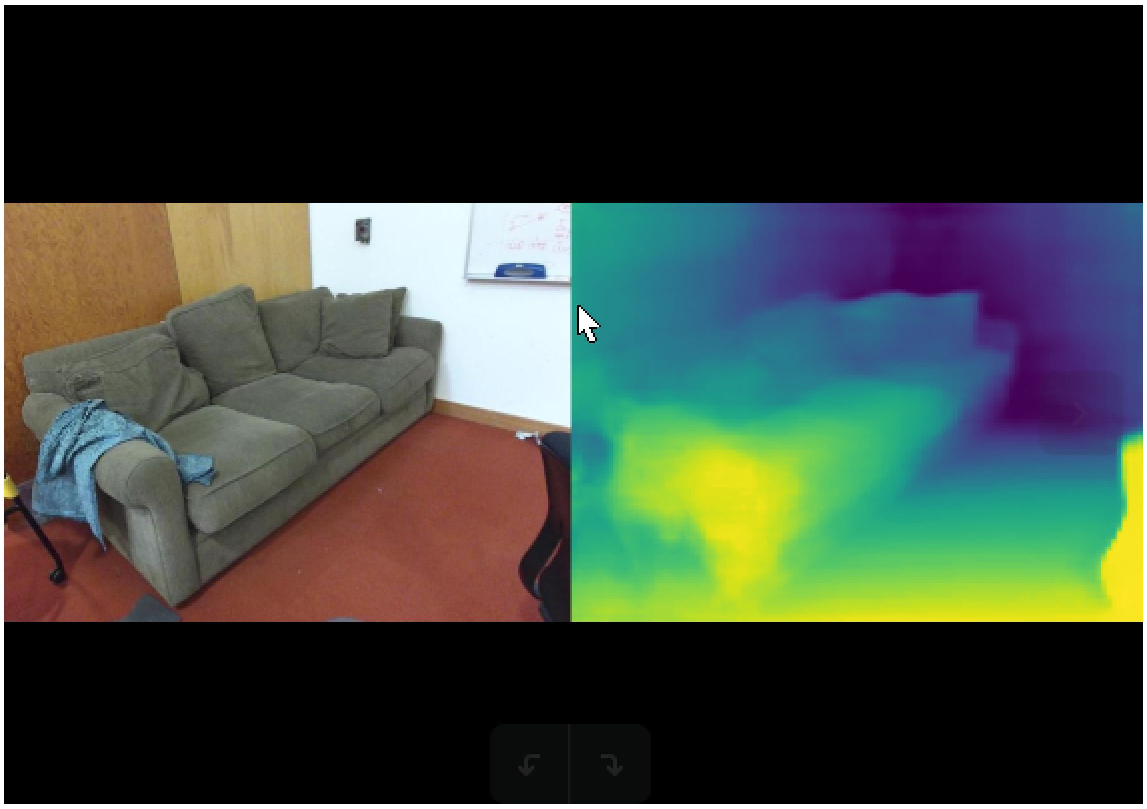
It is recommended to save the output image to the directory where images/test is mounted. Then, under Jetson inference/data/images/test, these images can be easily viewed from the host deviceAfter building the project, please ensure that your terminal is located in the aarch64/bin directory:

```
cd jetson-inference/build/aarch64/bin
```

Here are some examples of mono depth estimation in indoor scenes:

```
# C++
$ ./depthnet images/room_1.jpg images/test/depth_room_%i.jpg

# Python
$ ./depthnet.py images/room_1.jpg images/test/depth_room_%i.jpg
```



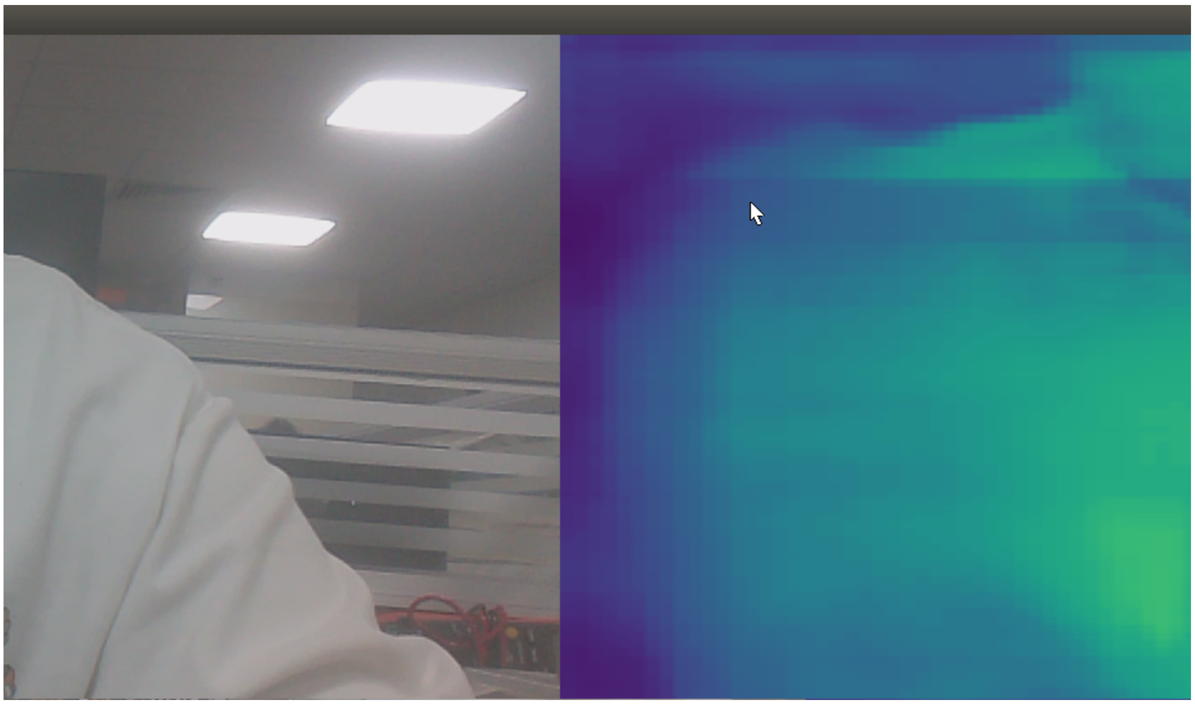
Note: TensorRT will take a few minutes to optimize the network when running each model for the first time. This optimized network file is then cached on disk, so future runs using this model will load faster.

3.Video Mono Depth

To run mono depth estimation on real-time camera streaming or video, enter the device or file path from the "Camera Streaming and Multimedia" page.

```
# C++
$ ./depthnet /dev/video0     # csi://0 if using MIPI CSI camera

# Python
$ ./depthnet.py /dev/video0  # csi://0 if using MIPI CSI camera
```



Note: If the screen is too small to accommodate the output, you can use -- depth scale=0.5 to reduce the size of the depth image, or reduce the size of the camera, where -- input width=X -- input height=Y