Depth Camera for Mac Users: Setting Up the OAK-D Pro in TouchDesigner

If you've worked with TouchDesigner before you would have surely stumbled across some Kinect tutorial and might have been disappointed to discover it is only compatible with Windows. The Creative Technology Hub now offers students the chance to borrow an OAK-D Pro camera. These IR depth cameras are compatible with both Mac and Windows, making them a great alternative to Kinects. This page will briefly guide you through installing the prerequisites and show you how to set your camera up in TouchDesigner.

1. Installing Dependencies

Before using the OAK-D Pro with TouchDesigner, you'll need to install some dependencies:

- **Python 3.8+**
- **DepthAI Library**

Assuming you've already installed Python and pip, you can check your Python version by opening the terminal and typing:

```
pyhton --version
```

If your version is older than 3.8, make sure to upgrade it. Next, install the DepthAI library by typing:

```
pip install depthai
```

At this point, it's good practice to restart your laptop before launching TouchDesigner.

2. Setting up TouchDesigner

2.1 Connecting the Camera

1. Connect the OAK-D Pro camera to your laptop using a **USB-C to USB-A** or **USB-C to USB-C** cable.
2. Launch **TouchDesigner**.

2.2 Using OAK Device CHOP

1. From the **Operators** panel, find the **OAK Device CHOP**.
2. Activate the CHOP, refresh the sensor list, and find your camera.
3. Click **Initialize**, wait for it to mark "Ready" (1), then press **Start**.

This CHOP runs a script in **Text DAT** that captures RGB data from the camera using the DepthAI library.

2.3 Displaying Data in TouchDesigner

1. From the **Operators** panel, find the **OAK Select TOP**.
2. Open its parameter window.
3. Drag and drop your **OAK Device CHOP** into the OAK Select TOP parameter **'OAK Device CHOP'**.
4. Click the arrow next to the **Stream** parameter and select `'rgb'`.
5. Your camera should now be running!

3. Capturing a Depth Map

Next, we'll program the camera to create a grayscale depth map using it's stereo vision. In fact, this camera has two monochrome sensors that capture images from slightly different angles. By comparing these two images, the camera calculates 'disparity' - the difference in position between matching points in the left and right images - which is then used to determine depth. In the grayscale image the brightness of each pixel represents the distance of objects from the camera. Lighter areas are closer, and darker areas are farther away.

To gather both `'disparity'` and `'rgb'` data, replace the code in **Text DAT** with the following:

```
import depthai as dai

def onInitialize(oakDeviceOp, callCount):
⬚return 0

def onInitializeFail(oakDeviceOp):
⬚parent().addScriptError(oakDeviceOp.scriptErrors())
⬚return

def onReady(oakDeviceOp):
⬚return
⬚
def onStart(oakDeviceOp):
⬚return

def whileRunning(oakDeviceOp):
    return

def onDone(oakDeviceOp):
⬚return
```

```python
def createPipeline(oakDeviceOp):
    # Create pipeline
    pipeline = dai.Pipeline()

    # Define nodes for RGB stream
    camRgb = pipeline.create(dai.node.ColorCamera)
    xoutRgb = pipeline.create(dai.node.XLinkOut)
    xoutRgb.setStreamName('rgb')

    # Define nodes for Stereo Depth stream
    monoLeft = pipeline.create(dai.node.MonoCamera)
    monoRight = pipeline.create(dai.node.MonoCamera)
    depth = pipeline.create(dai.node.StereoDepth)
    xoutDepth = pipeline.create(dai.node.XLinkOut)
    xoutDepth.setStreamName('disparity')

    # Configure RGB camera node
    camRgb.setPreviewSize(1280, 720)
    camRgb.setBoardSocket(dai.CameraBoardSocket.RGB)
    camRgb.setFps(30)

    # Configure Stereo Depth nodes
    monoLeft.setResolution(dai.MonoCameraProperties.SensorResolution.THE_800_P)
    monoLeft.setBoardSocket(dai.CameraBoardSocket.LEFT)
    monoRight.setResolution(dai.MonoCameraProperties.SensorResolution.THE_800_P)
    monoRight.setBoardSocket(dai.CameraBoardSocket.RIGHT)

    # Configure Depth node
    depth.setDefaultProfilePreset(dai.node.StereoDepth.PresetMode.HIGH_DENSITY)
    depth.initialConfig.setMedianFilter(dai.MedianFilter.KERNEL_7x7)
    depth.initialConfig.setConfidenceThreshold(240)
    depth.setLeftRightCheck(True)
    depth.setExtendedDisparity(False)
    depth.setSubpixel(False)

    # Linking RGB camera to output stream
    camRgb.preview.link(xoutRgb.input)

    # Linking Mono cameras to depth node
```

```
    monoLeft.out.link(depth.left)

    monoRight.out.link(depth.right)


    # Linking depth output to disparity stream

    depth.disparity.link(xoutDepth.input)


    return pipeline
```

Once you've replaced the script:

1. Click the arrow next to the **Stream** parameter in **OAK Select TOP**.
2. Select `'disparity'` to visualize depth data.
3. To display the `'rgb'` stream simultaneously, create a second **OAK Select TOP** and select `'rgb'` as the stream.

3. Conclusion

The OAK-D Pro is a fantastic alternative to the Kinect for Mac users working with TouchDesigner. At this stage you should be able to set this camera up and follow along any Kinect tutorial!

Revision #8
Created 10 June 2024 11:42:30 by Marta Ilacqua
Updated 16 January 2025 12:01:58 by Marta Ilacqua