

Geometric Image
Processing Lab

YOGA MASTER

By: Noa Wengrowicz & Barr Assenheimer

Supervisor: Ron Slossberg



PROBLEM STATEMENT

YOGA MATSER is a platform for yoga practicing anywhere you want. With the **Jetson Nano** you can take the yoga teacher to the park, to the beach, or just stay at home. Although the teacher is not near you can still get the feedback you need to improve your poses.



PROJECT SCOPE

BACKGROUND

- **NVIDIA® Jetson Nano™** Developer Kit is a small, powerful computer that lets you run multiple neural networks in parallel for applications like image classification, object detection, segmentation, and speech processing.
- **TensorFlow** is an open-source library for numerical computation and large-scale machine-learning.
- **TensorFlow Posenet** is a real-time pose estimation model.
- **TensorFlow Object Detection API** is an open-source framework built on top of TensorFlow for constructing, training and deploying object detection models.



PROJECT SCOPE

BACKGROUND



PROJECT SCOPE

MOTIVATION

- In the last few years there is an increasing interest in fitness apps and at home training. The main downside of these apps is the lack of real time feedback.
- This need was emphasized in the last few months, during the COVID-19 outbreak, forcing people to find alternatives for live fitness classes.



PROJECT SCOPE

HYPOTHESIS

- The real-time yoga poses detection on top of the posenet skeleton tracker can give the trainers the feedback they need to practice from home and improve their yoga poses.



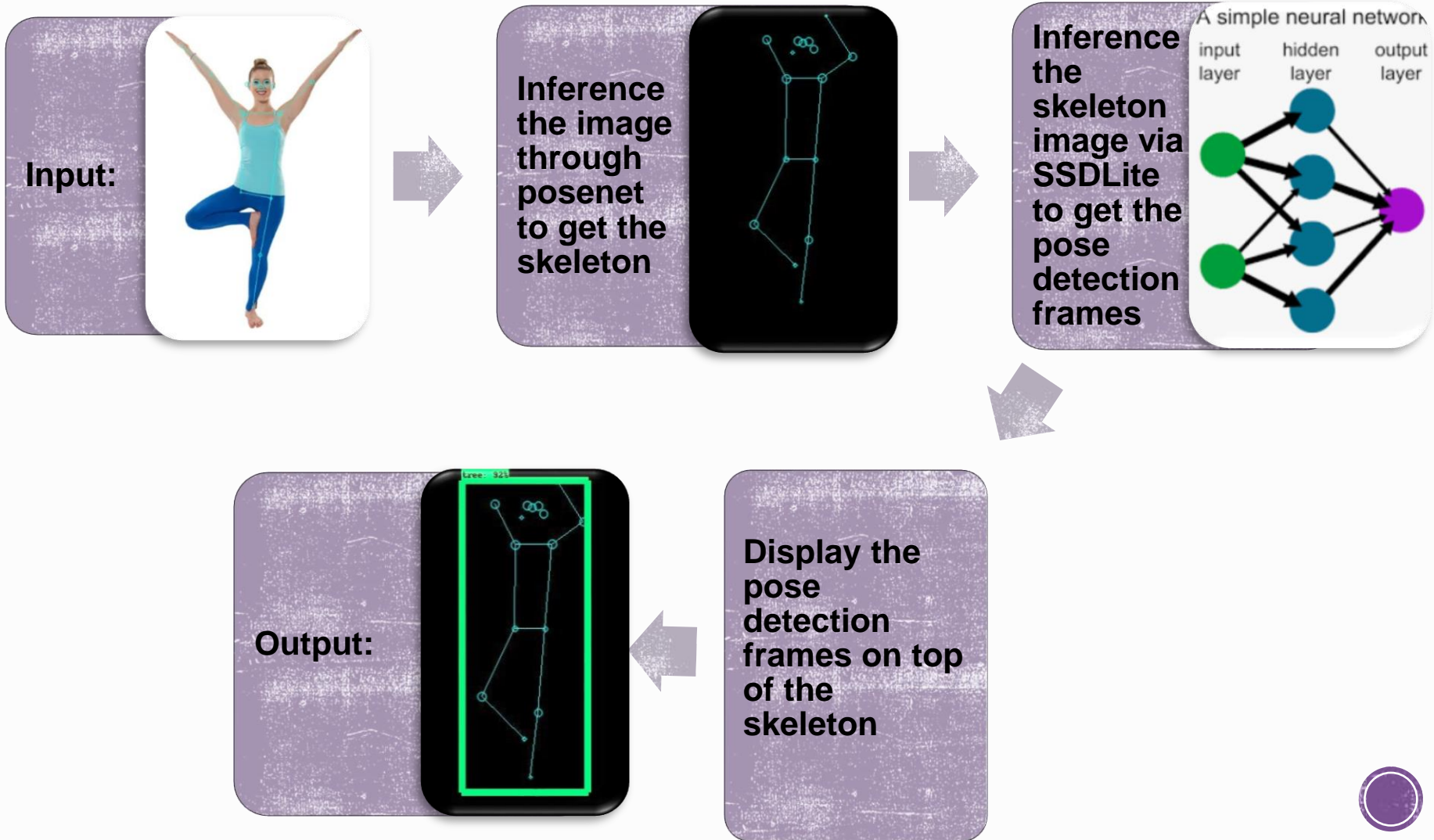
PROJECT OBJECTIVES

- **YOGA MASTER** will detect yoga poses in real-time.
- **YOGA MASTER** will display the pose detection frames with the skeleton on top of the live video stream.
- **YOGA MASTER** will let the user know whether he did the pose correctly.
- **YOGA MASTER** will keep track of the poses the user did correctly.

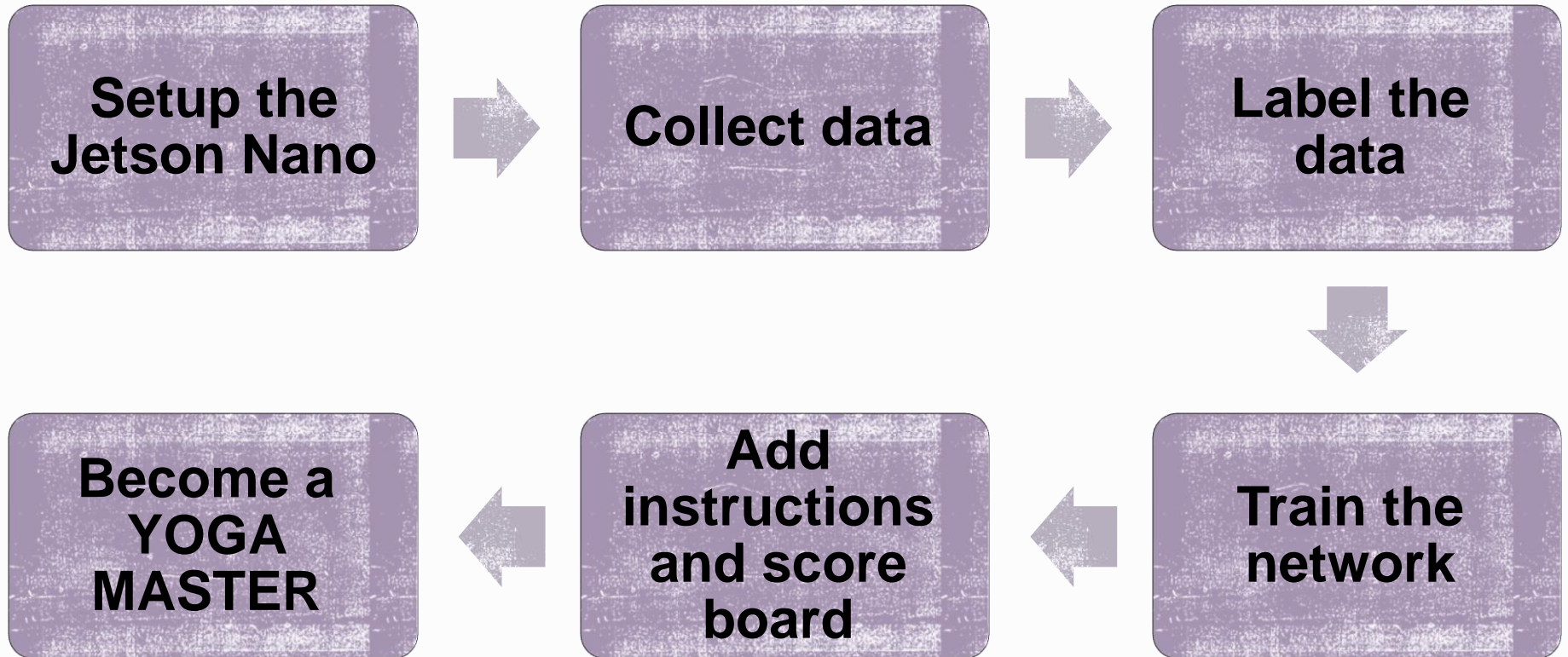


YOGA MASTER

USES DOUBLE INFERENCE

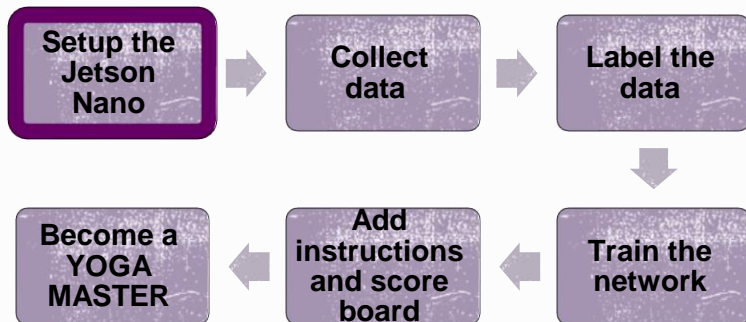


SOLUTION PROCEDURE

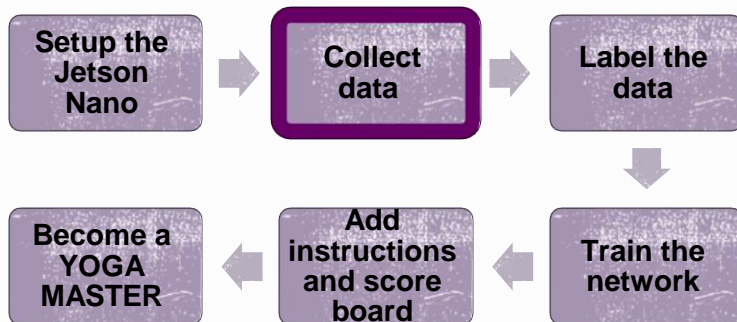
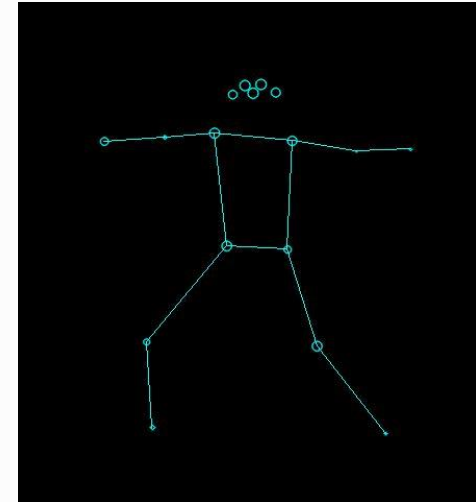
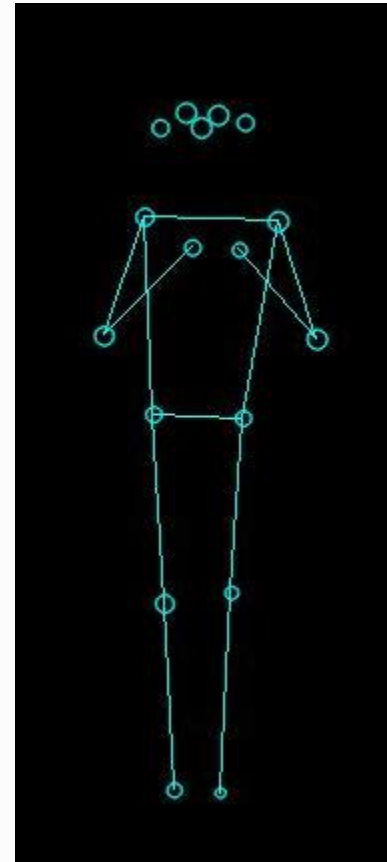
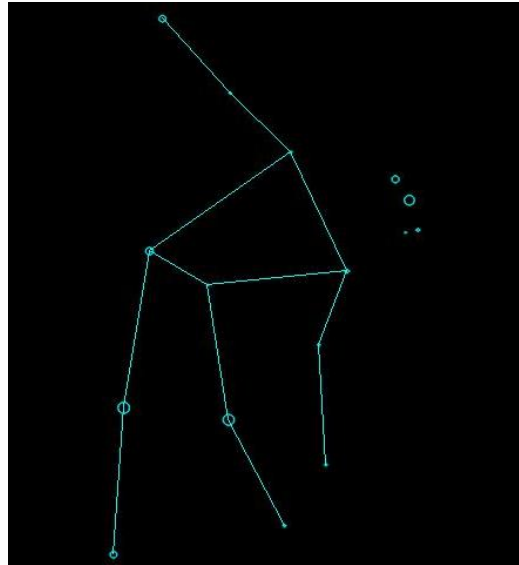
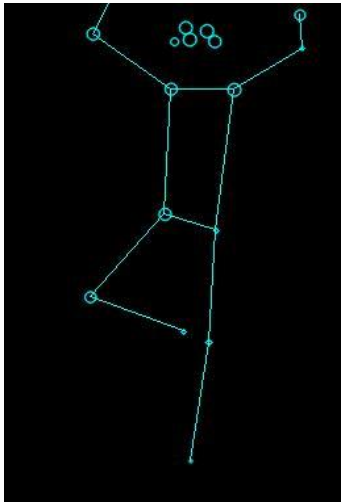


SOLUTION PROCEDURE

- The Jetson Nano environment is challenging. Setting up the required packages for the project took longer than we anticipated.

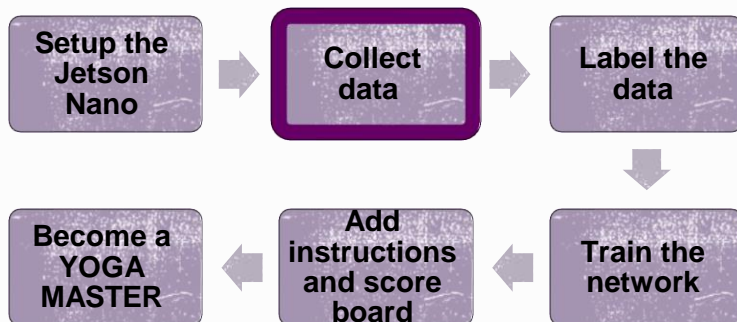


SOLUTION PROCEDURE



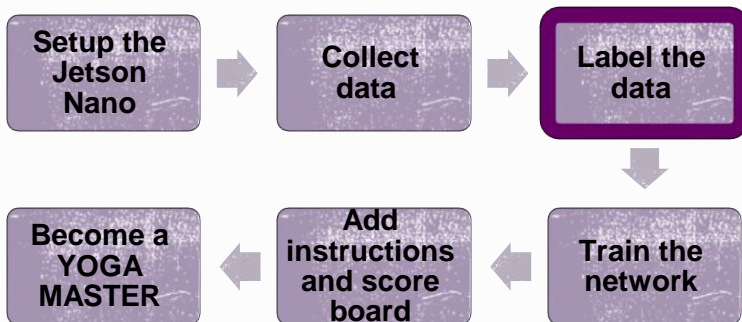
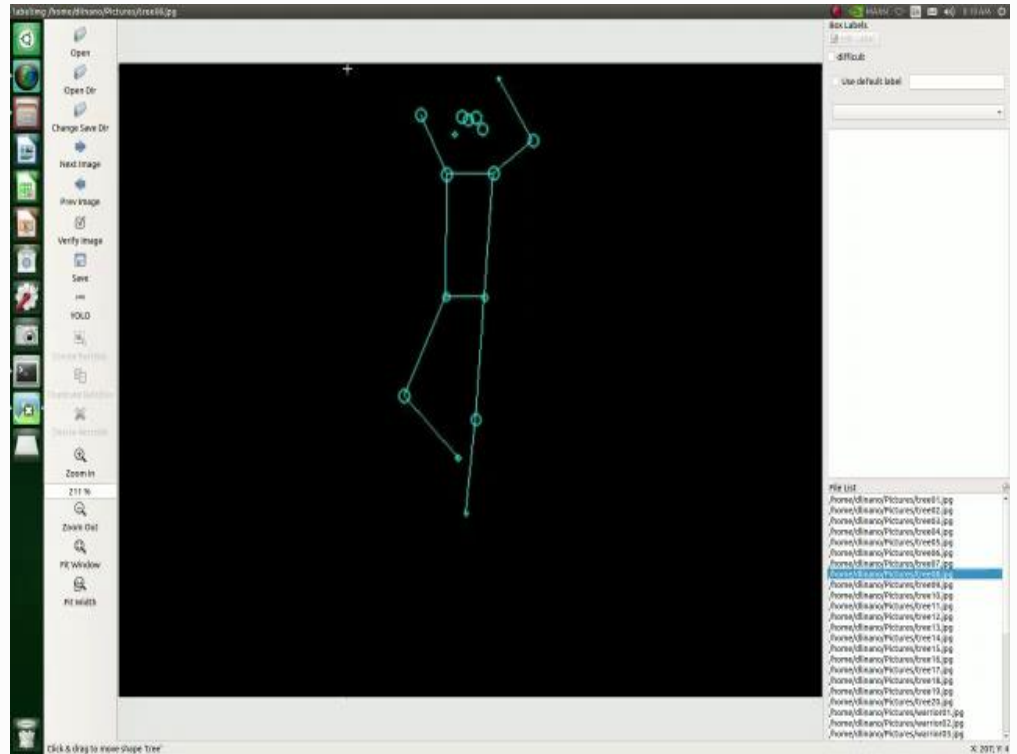
SOLUTION PROCEDURE

- We collected 130 images for each of the 4 poses.
- We tried to create a variety of images:
 - Different people (different body shapes)
 - Different locations in the image
 - Different angles



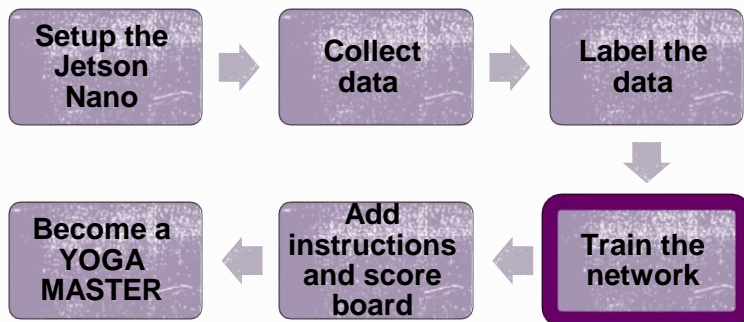
SOLUTION PROCEDURE

- We used **'labelling'** to label all the images



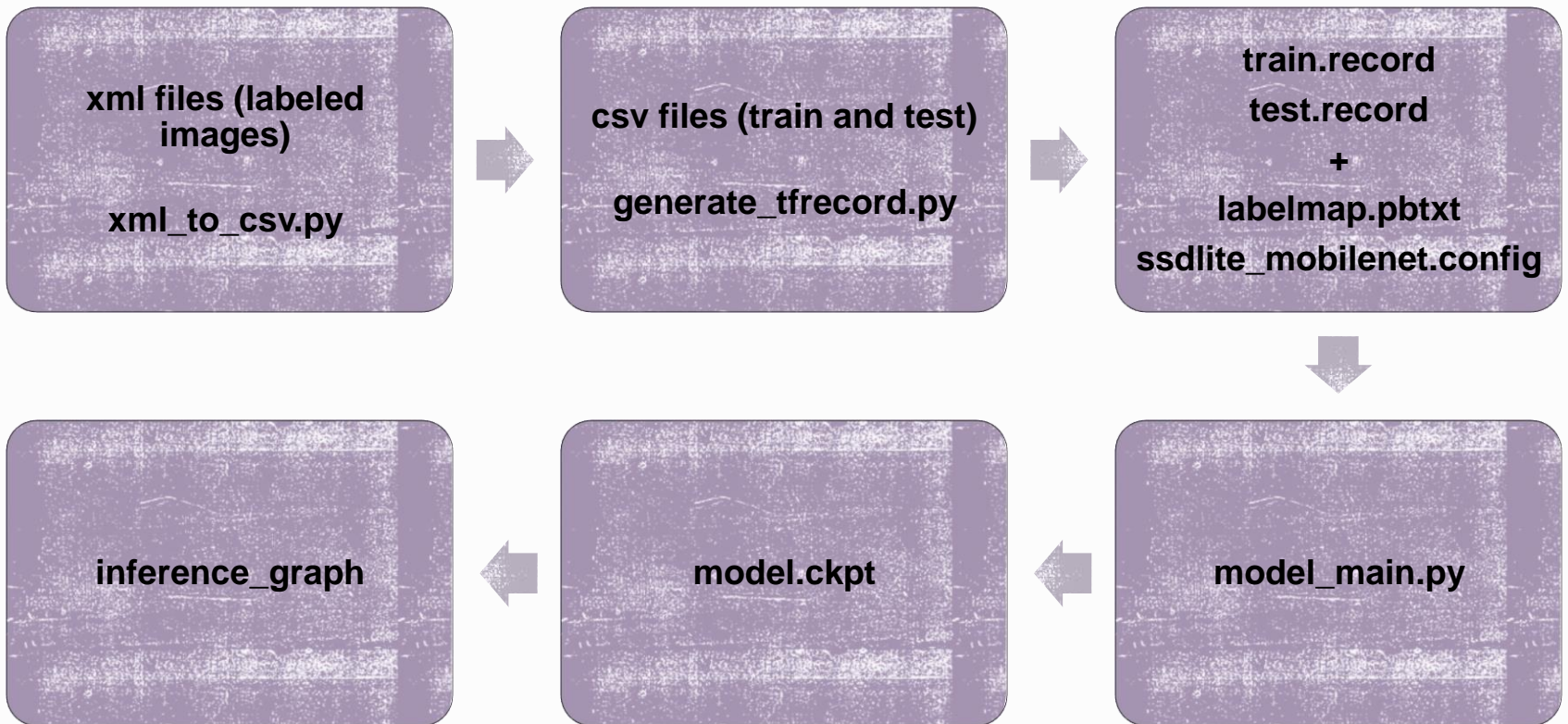
SOLUTION PROCEDURE

- We discovered the training process is too slow on the **Jetson Nano**, so we needed to find an alternative.
- We decided to do training using the **Google Colab GPU**. It was much faster, but not so easy to work with.



SOLUTION PROCEDURE

TRAINING PROCESS



SOLUTION PROCEDURE

TRAINING PROCESS

```

Open labelmap.pbtxt Save
~/project/my-project-en...

item {
  id: 1
  name: 'tree'
}

item {
  id: 2
  name: 'warrior'
}

item {
  id: 3
  name: 'triangle'
}

item {
  id: 4
  name: 'mountain'
}

```

```

Open ssdlite_mobilenet_v3_large_320x320_coco.config Save
~/project/my-project-env/fitstream-jetson-nano/training

model {
  ssd {
    inplace_batchnorm_update: true
    freeze_batchnorm: false
    num_classes: 4
    box_coder {
      faster_rcnn_box_coder {
        y_scale: 10.0
        x_scale: 10.0
        height_scale: 5.0
        width_scale: 5.0
      }
    }
  }
  matcher {
    argmax_matcher {
      matched_threshold: 0.5
      unmatched_threshold: 0.5
      ignore_thresholds: false
      negatives_lower_than_unmatched: true
      force_match_for_each_row: true
      use_matmul_gather: true
    }
  }
  similarity_calculator {
    iou_similarity {
    }
  }
  encode_background_as_zeros: true
  anchor_generator {
    ssd_anchor_generator {
      num_layers: 6
      min_scale: 0.2
      max_scale: 0.95
      aspect_ratios: 1.0
    }
  }
}

```

```

Open mountain01.xml Save
~/project/my-project-env/fitstream-jetson-nano...

<annotation>
  <folder>images</folder>
  <filename>mountain01.jpg</filename>
  <path>/home/dlinano/Desktop/images/
mountain01.jpg</path>
  <source>
    <database>Unknown</database>
  </source>
  <size>
    <width>640</width>
    <height>480</height>
    <depth>3</depth>
  </size>
  <segmented>0</segmented>
  <object>
    <name>mountain</name>
    <pose>Unspecified</pose>
    <truncated>0</truncated>
    <difficult>0</difficult>
    <bndbox>
      <xmin>246</xmin>
      <ymin>36</ymin>
      <xmax>410</xmax>
      <ymax>410</ymax>
    </bndbox>
  </object>
</annotation>

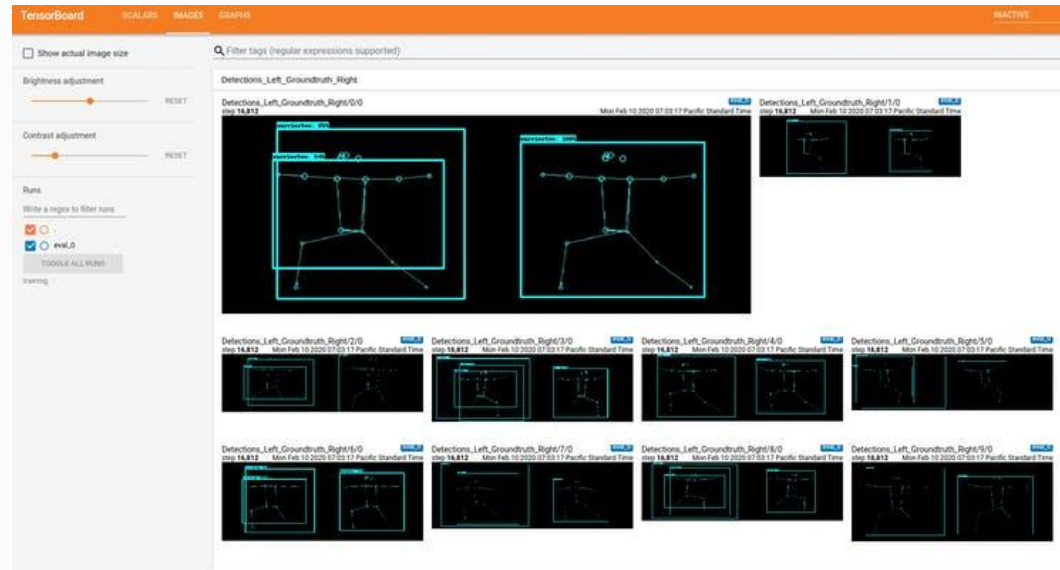
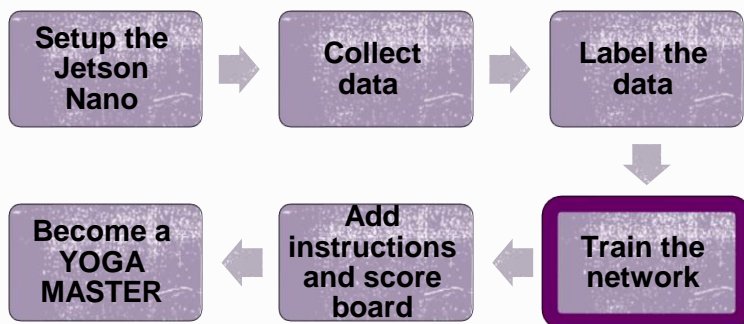
```

filename	width	height	class	xmin	ymin	xmax	ymax
tree06.jpg	640	480	tree	257	51	388	329
tree_61.jpg	640	480	tree	17	51	220	480
tree_42.jpg	640	480	tree	126	115	294	395
mountain16.jpg	640	480	mountain	322	28	488	429
tree_102.jpg	640	480	tree	22	27	240	467
tree_121.jpg	640	480	tree	312	42	503	358
warrior_82.jpg	640	480	warrior	240	111	392	331
mountain02.jpg	640	480	mountain	121	1	282	458
tree03.jpg	640	480	tree	437	3	590	294
triangle_98.jpg	640	480	triangle	129	153	284	378
tree_90.jpg	640	480	tree	228	59	348	353
mountain_87.jpg	640	480	mountain	213	61	389	480
warrior21.jpg	640	480	warrior	126	78	359	382
tree_46.jpg	640	480	tree	99	65	293	470
tree_44.jpg	640	480	tree	142	103	311	438
tree_75.jpg	640	480	tree	74	40	218	338
warrior_47.jpg	640	480	warrior	166	81	473	455
warrior_81.jpg	640	480	warrior	226	110	396	341
warrior16.jpg	640	480	warrior	328	11	623	375
triangle_62.jpg	640	480	triangle	369	57	570	402
warrior_76.jpg	640	480	warrior	263	96	509	364
tree30.jpg	640	480	tree	187	7	421	397
mountain_127.jpg	640	480	mountain	126	23	332	478
triangle_68.jpg	640	480	triangle	264	62	464	385
warrior_56.jpg	640	480	warrior	170	100	452	393
tree_91.jpg	640	480	tree	208	50	335	353
warrior11.jpg	640	480	warrior	127	69	432	347
mountain38.jpg	640	480	mountain	222	109	329	344
triangle_83.jpg	640	480	triangle	237	52	452	368
triangle25.jpg	640	480	triangle	376	74	581	379
warrior_122.jpg	640	480	warrior	305	71	640	480
warrior_45.jpg	640	480	warrior	3	78	358	480
mountain36.jpg	0	0	warrior	145	71	289	381
triangle_59.jpg	640	480	triangle	305	59	490	366
mountain32.jpg	640	480	mountain	218	1	382	451



SOLUTION PROCEDURE

- We used **TensorBoard** to visualize the training process (model_main.py).
- To get the best results we tried several options of:
 - Batch Sizes
 - Partitions of the train and test data.
 - Training duration.



SOLUTION PROCEDURE

INSTRUCTIONS

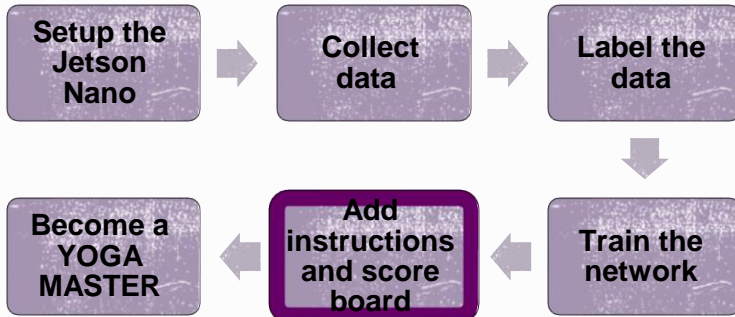
Try to do one of these YOGA poses:

TREE
WARRIOR
TRIANGLE
MOUNTAIN

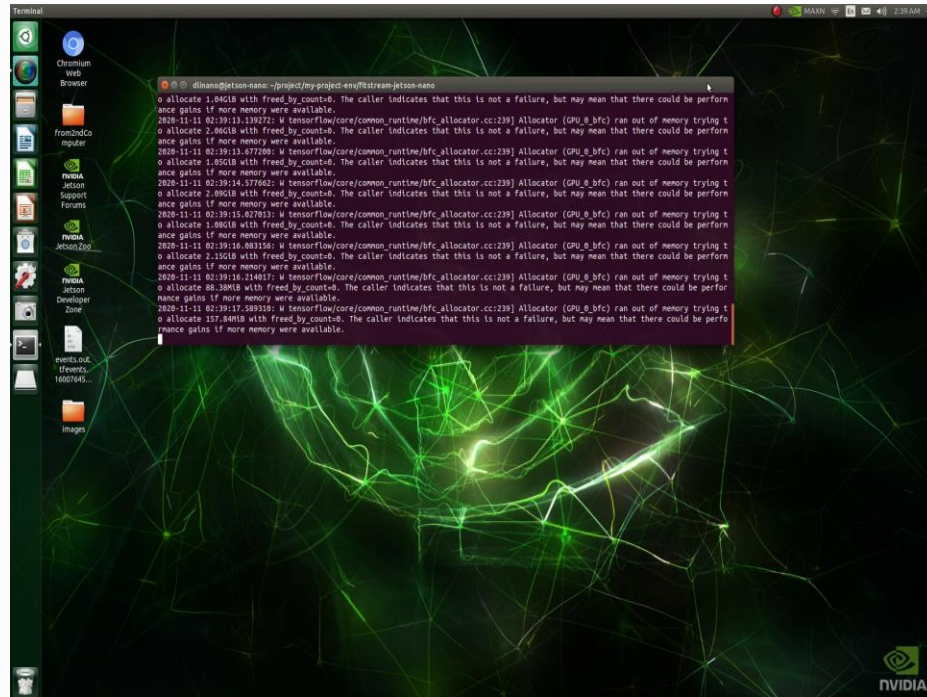
GO!

SCORE BOARD

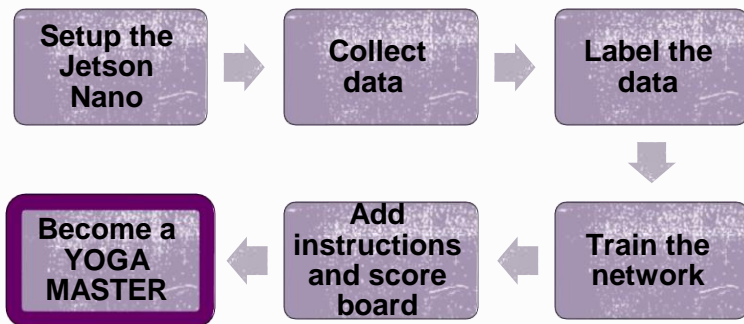
TREE	WARRIOR	TRIANGLE	MOUNTAIN
0	0	2	0



SOLUTION PROCEDURE



```
Terminal
di@jetson-nano:~/project/my-project-env/llstream-jetson-nano
o allocate 1.86GiB with freed_by_count=0. The caller indicates that this is not a failure, but may mean that there could be perform
ance gains if more memory were available.
2020-11-11 02:39:13.130727: W tensorflow/core/common_runtime/bfc_allocator.cc:239 Allocator (GPU_0_bfc) ran out of memory trying t
o allocate 2.86GiB with freed_by_count=0. The caller indicates that this is not a failure, but may mean that there could be perform
ance gains if more memory were available.
2020-11-11 02:39:13.677288: W tensorflow/core/common_runtime/bfc_allocator.cc:239 Allocator (GPU_0_bfc) ran out of memory trying t
o allocate 1.85GiB with freed_by_count=0. The caller indicates that this is not a failure, but may mean that there could be perform
ance gains if more memory were available.
2020-11-11 02:39:14.577662: W tensorflow/core/common_runtime/bfc_allocator.cc:239 Allocator (GPU_0_bfc) ran out of memory trying t
o allocate 2.86GiB with freed_by_count=0. The caller indicates that this is not a failure, but may mean that there could be perform
ance gains if more memory were available.
2020-11-11 02:39:15.827933: W tensorflow/core/common_runtime/bfc_allocator.cc:239 Allocator (GPU_0_bfc) ran out of memory trying t
o allocate 1.86GiB with freed_by_count=0. The caller indicates that this is not a failure, but may mean that there could be perform
ance gains if more memory were available.
2020-11-11 02:39:16.881356: W tensorflow/core/common_runtime/bfc_allocator.cc:239 Allocator (GPU_0_bfc) ran out of memory trying t
o allocate 2.35GiB with freed_by_count=0. The caller indicates that this is not a failure, but may mean that there could be perform
ance gains if more memory were available.
2020-11-11 02:39:16.214077: W tensorflow/core/common_runtime/bfc_allocator.cc:239 Allocator (GPU_0_bfc) ran out of memory trying t
o allocate 88.38MiB with freed_by_count=0. The caller indicates that this is not a failure, but may mean that there could be perform
ance gains if more memory were available.
2020-11-11 02:39:17.589338: W tensorflow/core/common_runtime/bfc_allocator.cc:239 Allocator (GPU_0_bfc) ran out of memory trying t
o allocate 157.84MiB with freed_by_count=0. The caller indicates that this is not a failure, but may mean that there could be perform
ance gains if more memory were available.
```



CONCLUSION

- The video streaming gives better result when running **YOGA MASTER** directly on the **Jetson Nano** and not via SSH.
- The **Jetson Nano** environment is challenging and setting it up takes long time.
- Training the model on the **Jetson Nano** is too slow. It is better to use a powerful GPU.
- Best training results were obtained when:
 - The batch size was 8.
 - We ended the training is when the 'DetectionBoxes_Precision mAP' is very close to 1.
 - We used 80% of the data for training, and the other 20% for testing.



FUTURE WORK

- Add more poses.
- Add different fitness activities.
- Augment exciting images.



THANK YOU

Them: The AI takeover is incoming.

The AI:

