

TL;DR DWPose distills strong whole-body keypoint estimators into efficient student models (tiny → large). It provides COCO-WholeBody results, plug-and-play checkpoints, and a ControlNet preprocessor replacement for OpenPose.

---

## What is DWPose?

DWPose (ICCV 2023, CV4Metaverse Workshop) is a distilled whole-body 2D pose estimation suite covering body, foot, face, and hand keypoints. It offers a spectrum of model sizes (t/s/m/l) balancing accuracy and speed, along with integration paths for downstream generative/use-cases.

Links:

- Paper: <https://arxiv.org/abs/2307.15880>
  - Repo: <https://github.com/IDEA-Research/DWPose>
  - Models (HF mirror): <https://huggingface.co/lyzd-v/DWPose>
- 

## Highlights

- Two-stage distillation: trains compact students while retaining whole-body accuracy.
- Full-body coverage: body, foot, face, and hands on COCO-WholeBody.
- Model zoo: tiny/small/medium/large, 256×192 and 384×288 input variants.
- ControlNet preprocessor: drop-in replacement for OpenPose in sd-webui-controlnet (dw\_openpose\_full).
- ONNX/OpenCV branch: inference without mmcv/onnxruntime (see repo branches).

Reported results (repo): DWPose-l at 384×288 reaches higher Whole-Body AP than smaller models; see the README table for per-part AP and FLOPs.

---

## Quickstart (MMPose path)

Prereqs: follow MMPose installation (see repo INSTALL.md). Prepare datasets if evaluating (COCO/UBody).

Evaluate on COCO-WholeBody / UBody (examples from README):

```
# UBody
bash tools/dist_test.sh \
  configs/wholebody_2d_keypoint/rtmpose/ubody/rtmpose-l_8xb64-270e_ubody-wholebody-256x192.py \
  $pose_ckpt 8

# COCO-WholeBody
bash tools/dist_test.sh \
  configs/wholebody_2d_keypoint/rtmpose/ubody/rtmpose-l_8xb64-270e_coco-ubody-wholebody-256x192.py \
  $pose_ckpt 8
```

Train (distillation → regular model transfer):

```
# Stage 1 distillation
bash tools/dist_train.sh \
  configs/distiller/ubody/s1_dis/rtmpose_x_dis_l__coco-ubody-256x192.py 8

# Stage 2 distillation
bash tools/dist_train.sh \
  configs/distiller/ubody/s2_dis/dwpose_l-ll__coco-ubody-256x192.py 8

# Convert distilled checkpoint → regular pose model
python pth_transfer.py $dis_ckpt $new_pose_ckpt [--two_dis]
```

ONNX/OpenCV path: see onnx and opencv\_onnx branches in the repo to avoid heavy deps.

---

## Using DWPose in ControlNet

You can swap OpenPose with DWPose as the pose preprocessor for ControlNet:

```
cd ControlNet-v1-1-nightly
# Download DWPose-l_384x288 and place under annotator/ckpts
python gradio_dw_open_pose.py
```

Set preprocessor to `dw_openpose_full` in `sd-webui-controlnet` (v1.1237+).

---

## Practical notes

- Input size trade-offs: 256×192 models are faster; 384×288 improve AP.
  - Detector quality matters: WholeBody AP assumes a person detector; see repo notes for detector AP.
  - Datasets: convert UBody videos to images first (`video2image.py`) and adjust annotations (`add_cat.py`) for evaluation.
  - Export/integration: ONNX branches help embed DWPose in non-PyTorch stacks.
- 

## References

- Paper: <https://arxiv.org/abs/2307.15880>
- Code: <https://github.com/IDEA-Research/DWPose>
- Models: <https://huggingface.co/yzd-v/DWPose>

Notes: Metrics and FLOPs are from the public README at publish time. For downstream use (e.g., generative pipelines), validate detector and preprocessor settings on your data.