**Python environment setup guide**

1. **Install miniconda**

Miniconda is a python package and environment manager. You will need some form of the Anaconda management system, and Miniconda is a “light weight” version of this that doesn’t install much overhead.

It will be easiest if you have one consistent “user” account on the Windows machine that will be processing the data. That is, the installation and following steps will be easiest if you have one logon that anyone who is doing this can use. If you don’t want to tie it to an existing person on the machine (like, if you want multiple people to be able to log on and do it), it may be worth creating a Windows user account specifically for doing this, log on under this user name, and then do all of the installation and subsequent work.

[Go to the miniconda page and download and execute the installer package.](https://docs.conda.io/en/latest/miniconda.html) You can accept the default settings.

1. **Set up the folder structure on your hard drive where you’ll do the work**

Create a folder on your hard drive where you’ll save the python scripts, as well as all of the data files for the project. It’s easiest to put this in the root directory of your user profile. So, if your login name is ‘datanner’ it would be in C:/Users/datanner, but you can put it anywhere you’d like.

You can call the folder anything you like as well. E.g., “video\_data\_processing” or whatever.

You will need a few files in the folder you create (environment.yml, utils.py, and process\_videos.py), and some sub-folders. The sub-folders are where you will need to put the raw mp4 files, and the other sub-folders will be where the script saves the processed files (mp4 files with the skeleton tracked on top, the csv files with the tracking coordinates, and the tracing image with the ankles and wrists).

The sub-folders need to have the names:

* raw\_videos
* tracked\_csvs
* tracked\_images
* tracked\_videos

I will email you the ‘environment.yml,’ ‘utils.py,’ and ‘process\_videos.py’ scripts that you will need. Put them in the root of the folder you create.

Within that root folder (call it whatever you want), this is what the contents should look like:  
Graphical user interface

Description automatically generated with medium confidence

Sub-folder spellings/names are important. The script won’t work if the files/folders aren’t exactly like this. You can have extra stuff in there if you want, but you need these files and folders as a minimum requirement.

1. **Move some data to their respective folders (and manage files as you do more processing batches after the initial setup)**

The root folder from above should be on a hard drive that is large enough to a decently large amount of data for each set of raw files you want to process.

I have processed all of the videos that were available on the blob storage around the end of August, so the ones that were there already have a csv, tracing image, and tracked video file in the respective folder on the blob storage under the main folder `processed-data`. E.g., `processed-data/tracked\_images` contains all of the tracing images from the videos that have been processed, etc.

I have made a csv list of all of the files that have been processed. You don’t need to reprocess these at all. This csv will be in the email I send you.

You should keep all the tracing image files that have already been processed in the `tracked\_images` folder that you created above. You can download the ones I have already done from the blob storage. These are small and shouldn’t take up too much hard drive space.

These are what will tell the scripts what to do and what to ignore when you add more raw files, which is why you should keep them around even after you run the script to process video batches.

You will want to add any raw mp4 files that do not have corresponding traced images to the `raw\_videos` folder you created. You don’t need to add any of the raw mp4s from the csv of already-processed videos to the `raw\_videos` folder. Any raw mp4 files not in that list will need to be processed.

After you run the script, you will be able to move the raw files that have already been processed off of the hard drive to save space. As you get more raw mp4s you can add them to the folder, run the script, and then pull them off to save storage if you want.

After each run of the script, you can move the mp4s with the skeleton tracking on top of the baby out of the folder if you want to save storage space on your computer. They should be backed up to the blob storage (and other backups too – multiple backups for everything is always a good idea… stuff can get deleted from the blob storage and isn’t recoverable). So, if there are space issues getting all of the raw mp4s on at once, you can process them in batches. Process some, move them off, process more, etc.

The csv files and images should be backed up (to blob, and elsewhere) as well. The images should stay in the folder you’ve created on the computer for reasons described below, but the csvs and anything else can be removed if you want or need to free up space.

The logic for how the script will decide for what videos to process is this:

1. The script will look in the ‘raw\_videos’ folder and see the names of all of the files in the folder and make a list of the files with the extension ‘mp4’

2. The script will look in the ‘tracked\_images’ folder to see what raw files have a corresponding image file already processed in that folder with an extension of ‘png’. The png files will have the same file name as the mp4 files; only the extension will differ.

3. The script will then process any file in the ‘raw\_videos’ folder that does not have a corresponding image in the ‘tracked\_images’ folder.

4. It will save the tracing image, csv file, and skeleton tracked video file for each raw video it processes into the corresponding sub-folder. The tracked mp4 file

So, if you don’t keep the tracked images files around, the script may do extra work and re-process any raw videos that are still laying around in the raw videos folder. This isn’t a problem, it just means more time for the script to run. The traced image files will be small, so they won’t be a storage problem.

1. **Create the conda environment**

This will download and install the packages that you will need to process the videos.

After you install Anaconda/Miniconda, there will be a program on your computer called the Anaconda prompt. If you go to the windows search bar at the bottom and start typing “anaconda” you’ll see this program pop up:  
  
Graphical user interface, text, application, email

Description automatically generated

Click on that and you’ll see a command prompt window open in your user root directory, like this:

Graphical user interface, text

Description automatically generated

“cd” into the directory you made in step 2 above. So, if you created a directory called “video\_processing” under your username, you could type “cd video\_processing”.

If you made the director in some other place, you’ll want to type “cd [full path to your directory that you created]”.

Make sure you’re in the directory that has the environment.yml, utils.py, and process\_videos.py files as well as the directories you made.

Once you’re in there, you’ll need to type the following line:

conda env create -f environment.yml

This will create the python environment based on the packages listed in the environment.yml file. If it asks you any questions while you’re installing just click “Y”.

Once this is done, try typing the following command to make sure it works:

conda activate nch\_data\_process

The word to the left of the prompt (which said “base” before) should now say “nch\_data\_process”. That means you’re in the correct python environment and the packages you installed will be accessible.

Every time you want to process more videos, you’ll need to open up this anaconda prompt again and cd into the same directory again and type the same conda activate nch\_data\_process line again.

1. **Fix some files**

You will need to fix/add some files in the default installation of the pose tracking packing.

To do this you will need to navigate to the installation directory for the MediaPipe package. On Windows, the directory will generally be nested under your user directory.

First, enable viewing hidden files in your Windows explorer window. Open a Windows explorer window, click on the View tab, then check the box next to “hidden items”:

Graphical user interface, application, Word

Description automatically generated

You will need to click into the directory that has the mediapipe package, and it will be under one of the “hidden” folders you just revealed.

The directory you want will have a path like:

C:/Users/[username]/.conda/envs/nch\_data\_process/Lib/site-packages/mediapipe

where [username] is the username you were logged in under when you installed created the conda environment. E.g., if my username is ‘datanner’, you would replace [username] with datanner.

You’ll need to do two things once you find this file

* Replace the drawing\_utils.py file

I will email you the correct file. It contains some fixes that allow you to plot the tracking skeleton on top of the video with a little more info on it than the default utility.

The file will be in the folder (continue navigating from the one above):

C:\Users\[username]\.conda\envs\nch\_data\_process\Lib\site-packages\mediapipe\python\solutions

Simply copy the drawing\_utils.py file I send to you and paste it into that folder. If it asks you if you want to replace, click “yes”.

* Download the tracking model and drop it into the correct folder

When I was testing the environment out it seems that the default installation of the tracking model no longer points to the correct URL to download the model you need to do this.

First, download the model. Go to this URL at MediaPipe: [Models and Model Cards - mediapipe (google.github.io)](https://google.github.io/mediapipe/solutions/models.html#pose). Click to download “TFLite model (heavy)”.

Take the model you downloaded and drop it into this folder:

C:\Users\[username]\.conda\envs\nch\_data\_process\Lib\site-packages\mediapipe\modules\pose\_landmark

At this point you should hopefully have everything set up and ready to go to do the media processing.

1. **Process videos!**

If everything above has gone according to plan, you should be able to process videos now.

Make sure you have the raw mp4s in the correct folder from step 2.

To process the videos, open up the Anaconda prompt and cd into the correct directory, like you did in step 4. Type:

conda activate nch\_data\_process

to activate the environment, like you did in step 4. To process the raw videos in the raw videos folder (and which don’t have corresponding png files in the tracked images folder), type the following:

python process\_videos.py

This should start everything running. The first time you do this I would run 1 or 2 raw videos, just in case. You’ll see a printout in the console something like “Processing video 1/2" (if you have two videos). And when it’s done, it will print the word “DONE” and you’ll get the command line back.

If you get any error messages, copy the *entire* error message printout and email it to me without changing any formatting. Or, get a screenshot and email me that screenshot. This will help me troubleshoot.

A couple of things that you might want to/can change if you want:

* *Frame resizing*

I’ve set the script to resize the video frames by a factor of 0.8 (make them 20% smaller). They’ll still be plenty big enough to see file sizes will be much smaller than without resizing. If the file sizes are still big (they will be for some videos), you might want to change the resizing factor to something smaller, say 0.6 or 0.5. If you want to go to 0.6, type the following command when you go to process the videos:

python process\_videos.py -fr 0.6

This will override the default 0.8 scaling. “fr” is for “frame resize”.

* *Number of processing cores*

In order to make the videos process faster, I’ve set the script to do parallel processing – it will use multiple “cores” on your processor, and send one video to each processor so that more than one video can be processed in parallel. You can find out the number of cores on your machine by doing the following:

* Right click on the task bar at the bottom of the screen
* Select Task Manager
* Click on the Performance tab
* Look for a line at the bottom saying either “virtual processors” or “logical processors” or something similar, like this:

A picture containing chart

Description automatically generated

You’ll probably have either 4 or 6 cores/processors. This is the upper-limit on the number of videos you can process in parallel. Generally you won’t want to use all of the processors at once, though. One a laptop using all of them will eat up the RAM really fast (I have a HUGE amount of RAM on my VM, so it’s not an issue for me, but will be on a normal laptop). It’s also good to leave some processing room leftover for other system processes, or else the task manager will forcibly slow things down to manage everything that needs to happen.

By default the script uses your number of cores minus 2. So if you have 6 cores, it will use 4. If you have 4, it will use 2.

If you run into problems with memory issues, you might want to override the default number of cores that the script uses. You can do this by adding a different “flag” to the call to the script. For example, if you only wanted to use 1 processor, you can use the following command when you run the script instead:

python process\_videos.py -nc 1

The -nc flag tells the script to just use 1 core. “nc” is for “number of cores.” You can combine this flag with the frame scaling flag from above if you want. For example, if you want to use 1 core and make the frame scaling factor 0.6, you’d type this:

python process\_videos.py -nc 1 -fr 0.6

If you get an error message saying something like “MemoryError” on the last line of the error output, you’ll want to reduce the number of processors so that the machine doesn’t have to do too much at once. It’s fine to do this – things will just go slower.

**Some places where you might have hiccups and some things to try**

If you get error messages at any of the above steps, and none of the troubleshooting below helps, copy the *entire* error message and email it to me with no formatting changes. Also, get a screen shot and email that as well.

1. Installing miniconda

Unlikely to have problems here to be honest. It’s just a normal windows installer.

1. Creating the conda environment

On rare occasions I’ve had problems creating the environment. If it’s a new miniconda installation there shouldn’t be too many problems, but here are some things to check out:

* If the Anaconda prompt program doesn’t exist when you look for it, make sure you are logged in to the same user account that was used when you installed Miniconda. By default Miniconda only installs for the user who ran the installer, so if another person logs in under a different user profile, they will probably not find the Anaconda prompt. So, it’s easiest if you use a single login to do this. Either pick one person’s to use, or create a username that you can share, then do the installation.
* Make sure you’re using the Anaconda prompt program from Step 4 and not a “normal” command or PowerShell prompt. There are ways to initialize other terminal programs, but to keep things simple for this process, just use that prompt program. If you’re not using the Anaconda prompt, it won’t recognize the “conda” command.

1. Fixing/copying the drawing utils file and the model file

Finding the path to where conda installs the packages can be tricky. It *should* be in the path I described above on Windows. If you follow the default miniconda install instructions and then create the environment in that same user profile, it should install all of the package files in the .conda (hidden) folder.

I’ve had to dig around for this sometimes too. My way is to open a windows explorer window and search the C:/ drive for “drawing\_utils.py”. It may take a while, and a few results will come up. When the search stops. Right click on each one, then click on “Properties” and look at the path. Some might be a shortcut to something or it might link to the file I sent you in my email. Those aren’t the file paths you want. Look for something similar to the file path I copied above… e.g., something with site-packages/mediapipe. You can then use the path stuff in the top of the windows explorer bar to dig around for the correct folders for the two files.

1. Running the script

I’m guessing this is the most likely place for errors. Depending on the error the following things may help.

Things to check for:

* Are the folders you created named/spelled exactly like in the description/image above? Spelling errors will cause errors.
* Do the files in the raw videos folder have the “.mp4” suffix? The script needs that to be able to parse the file names.
* Did you activate the conda environment with the “conda activate nch\_data\_process” command before running the script? Does it say “nch\_data\_process” to the left of the command prompt in the terminal window?
* If you added a flag like “-nc” or “-fr” did you do it exactly like in the above examples? They require specific types of values. “-fr” requires values > 0.0 (decimal). “-nc” requires an integer value > 0 as well. If you put in a huge value it will fix it and just use the number of cores you have minus 1. But if you give it a decimal (like 1.5) it won’t work. I tried to code the error messages for these so they’re intuitive in case they get a bad value.

If you get an error message with the word “MemoryError” at the bottom, try reducing the number of cores using the “-nc” flag described above. Also, make sure all other programs are closed. If you get down to 1 core and you’re still getting the error even with all other programs closed, you may need to find a machine with more RAM. E.g., if your machine has 8GB RAM, you should find a machine with 16GB RAM.