

# Making videos

*Disclaimer: I know very little about ffmpeg! Cory Wolff provided scripts used by NCAR EOL to create flight videos that use ffmpeg, and what I've learned after that has been through trial and error, internet searches, and reading the ffmpeg website (<https://www.ffmpeg.org/>). Again, I do not claim to know much about ffmpeg. If you find any errors or know of better ways to do things, please let me know.* I've tested the code on a Win10 PC running Igor 8 and a Mac running Igor 6.

To use the Igor script, you will first need to install `ffmpeg`, and then get the path to `ffmpeg`. On the PC, I installed the `ffmpeg` folder in C. For Macs, the preferred way apparently is to use HomeBrew. I don't use the admin account on my Mac for everyday use, and I ran into some issues installing HomeBrew as a non-admin user. I wasn't in the mood to find the solution to this, so I just downloaded a static build and installed it in `usr/local/bin`. To get to that folder, I opened Finder, pressed `Command+Shift+G`, and entered `/usr/local/bin` into the dialog box. I needed admin permission to install there, but that wasn't an issue.

There are two `ffmpeg` paths already in the ipf. If you installed in the same places I did, then you don't need to change anything. Otherwise, change the string constants at the top of the ipf to the correct paths for your setup. Make sure to change the correct string constant for the OS you are using. You can also add the `ffmpeg` path to your system variables or bash profile, but I decided that just passing the full path is a little nicer. I could be wrong.

To make the videos, you will want to create a series of images that `ffmpeg` will then stitch together to make the movie. The names of the images should have an index at the end that increases by 1 for each new image. There is a command in `ffmpeg` that you can pass that doesn't require the indices to increase by 1 (`-pattern_type glob`), but it doesn't seem to work on Windows, so I haven't had a chance to test it. The image names should be of the form `[Base name]_[index]`, where the index is padded with zeros so the number of characters is always the same. Sample code is provided below.

```
FUNCTION Sample_Video()
    Make/O/D/N=100 wx=p, wY=2*p
    Variable vMake_Cells=0

    String strCel_Base_Name="Movie_Frame_"
    String strCel_Path="My Computer:Pictures:Pictures_For_Movie:" //Put a real path here to try this out
    String strCel_Number

    Variable iDex
    FOR(iDex=0;iDex<numpts(wx);iDex+=1)
        IF(iDex=0)
            Display wY[0,iDex] vs wx[0,iDex]
        ELSE
            AppendToGraph wY[0,iDex] vs wx[0,iDex] //The trace name for this will be wY#1
            RemoveFromGraph wY //Removes the old trace, and now the new trace has the name wY
        ENDIF //End checking to see if we are on the first point or not

        //You'll probably want code here to make the graph look pretty

        DoUpdate //VERY important! Must have this for the code to work

        IF(vMake_Cells)
            sprintf strCel_Number, "%08d", iDex //Using %08d so we (hopefully) don't have to worry about running out of digits
            SavePICT/O/E=6/B=144 as strCel_Path+strCel_Base_Name+strCel_Number+".jpg"
        ELSE
            Sleep/T 20 //If not making the cells, pause execution briefly
        ENDIF //End checking to see if we are making cells or not
    ENDFOR //End looping over the points in the wave

END
```

*Sample code for making the cells*

There cannot be anything else in the images folder except the images. I'm not sure why you would do this, but if you do, the code will probably run into some issues.

Once you have made the cels, run `Make_Video_Panel`. This creates a little GUI. Browse to the folder that contains the images ("Images Folder Path"). If your images have nicely formatted names, the "Images Name" box will fill in. The default bit rate and framerate seem to work well, though I've used lower framerates when I have a small number of frames and want to slow the movie down. Select the folder to hold the completed video ("Video Folder Path"), and a name for the video. `ffmpeg` has several encoders, but I've hardcoded H.264 into the script. I think some of the other encoders are more efficient, but H.264 has been good enough for me. The file type of mp4 is also hardcoded in the ipf, but can be changed if desired.

After you click the "Make Video" button, Igor will use a command called `ExecuteScriptText` to create the video. On a PC, it will execute a batch file with the proper command, while on a Mac it passes the command to AppleScript, which is why the commands are a little different. On PCs, the Igor code will create the batch file in the same folder as the video destination. If everything was entered correctly, `ffmpeg` should run and create a nice video. The command that was executed will show up in the box below the "Make Video" button if you run into issues and want to call the commands from the terminal prompt.

The second tab has controls for changing the speed of the video. I've used

```
-filter:v "setpts=X*PTS"
```

for this, where "X" is the factor by which you want to slow down the movie. More details are given here: <https://trac.ffmpeg.org/wiki/How%20to%20speed%20up%20/%20slow%20down%20a%20video>.