digisub

by timecop@efnet

required software

- VirtualDub
- vobsub (includes required textsub and dvobsub)
- sub station alpha
- msmpeg4v2 (or any other high-performance light codec)

0. setting up the software

This step obviously only need to be done once, and you only need to repeat it each time you reinstall your system or when new versions of the software above come out. Use google to find latest versions of the software. As of this writing, latest VirtualDub is 1.4.10, latest Vobsub is 2.18, and latest SSA is some 3-years old version (but that's all we have). MSMPEG4 is a free download from somewhere on microsoft.com. You are looking for "Windows media player codecs" pack.

UPDATE Since I've noticed a lot of hits to this guide were referrals from google for WMP codecs, I decided to find the microsoft download page for them: msmpeg4 + wmv8 codecs download from microsoft.com

Setup VirtualDub, run vobsub setup, make sure to check the "install textsub for virtualdub" and "install dvobsub", finish install. Use the ssa installer to install that. Check the video codecs list in VirtualDub (Video->Compression), if you don't already have Microsoft MPEG4V2 install the wmp codecs pack from the link above.

1. preparing the project directory

You should organize your current sub project in one directory, keeping the .avi and all the support files in one place.

Name it however you want, by name of your current project or something, but important thing here is to keep it all in one place. Hint, "Desktop" is NOT a good place to keep your current project, especially if you are concerned about your username / last name showing up in the script (and it will if your login name is your last name or something). At the point when you are reading this, you already have a untranslated .avi file and a translated script, probably a plain text file in the style of:

```
B - Where do you want to go today?
```

or

```
Bill:   Where do you want to go today?
```

or

```
I'm a stupid translator who didn't put character names in!
Yes, you are.
```

Where B would be a character name (perhaps expanded at the top of the script), followed by what Bill says. Translations which have timestamps are also common, I don't like them, but if you don't know Japanese at all, having timestamps might make it easier to time the script later on.

First things first, you need to grab the audio out of the .avi and prepare it for SSA's stupid requirements.

2. saving the audio

Open up the avi in virtualdub, goto Audio menu, select Full processing mode, then goto "Conversion" dialog. Change sampling rate to whatever you want (11k doesn't sound too bad), and make precision and channels set to 8bit, mono. By default, audio compression should be set to "no compression", double-check in Audio->Compression menu. Goto File->Save Wav to save the resulting .wav in your current subbing project directory.

SSA is a few years old, and is quite buggy. One of the problems, is sometimes it refuses to accept the .wav file you just created by claiming that "The file is too short to do anything sensible with". Sometimes it'll load it, but lock up right after. To solve both of these problems, (they might only appear on 11k sampling rate files, that's what I use), simply add about 5 minutes of silence at the end of

the file (I use SoundForge 4, any half-decent .wav editor should allow you to do that). Then SSA will happily accept the file.

If you choose a higher sampling rate, such as 22k or 44k, another SSA bug will demonstrate itself by not letting you time past the first half (at 22k) or quarter (at 44k) of the .wav, thinking it ends there. Solution to this problem is similar to the "too short" problem - load your .wav into your favorite text editor and append extra silence at the end. For 22k, you'll need at least 15 minutes, and more for 44k. This is a SSA bug, as far as I can tell.

If the sound in the video you are processing was digitally recorded and preprocessed you might find out that converting it to 11k 8 bit mono makes it almost impossible to hear anything. In this case, you want to use a real sound processing program such as SoundForge or similar to combine two stereo channels into mono. The key here is use the "Stereo to Mono - Use Difference between channels" preset in SoundForge. Find whatever option does this in your sound editing app.

3. mixing your translated script into a dummy ssa file

There's two ways to do this, depending on the layout of the script you receive.

In general, you want to have a .ssa file with the same name as the .avi file in your current project directory. For example, if you are subbing "microsoft.avi", your SSA should be called "microsoft.ssa".

The easy way is to try and use "Open" function in the SSA. If the script is more or less consistent, SSA *might* be able to figure things out and automatically fill in names and each line. If you receive a script that looks like this:

```
Bob:            Where do you want to go today?
                How about my house?
Jeff:           Sure.
```

You will have to pre-process it and repeat the missing name for each line in the script, so that it looks like:

```
Bob:            Where do you want to go today?
Bob:            How about my house?
Jeff:           Sure.
```

SSA doesn't seem to have too many problems with this format. After load is finished, you should have a script full of default timings. Check it out to see if there are any screwups and fix the .txt if necessary and try loading again. I don't like the default timings this method creates because they make script editing slower because SSA tries to redraw the script grid each time you grab timing. Setting the timings to zero is left as an excersize to the reader.

If the script is really messed, you'll have to do it the hard way. Open the .ssa in some text editor which that you know how to use which supports macros or has an easy way to repeat a lot of text operations. What you are going to do, is add default timing line in front of every line in your translated script. In the editor, you will see the text commands of the ssa file (its all text, nothing special).

Startup Sub Station Alpha, and look at its first screen. In the large textbox where you are expected to put in dialog, type in something random. In the name combobox, type in the default character name (Don't use default, use something else, you will need to search/replace it later on). Press enter to save the only line in this sub (SSA wants you to press enter after most modify operations, otherwise the change does not take effect). That's all you need for now. Press F2 to save the .ssa into your current project directory. A dialog box will come up asking you to enter script credits. Do whatever you want with that. I just press enter. Next up the save dialog comes up. Save the ssa with the same name as the .avi file in your current project directory, and open it up with your favorite text editor.

The only useful lines near the top are "PlayResY" (more on this later), followed by styles section, followed by Events section. In the Events section, you should have the line you just added earlier, with default timings followed by random stuff you typed in. It should look like this:

```
Dialogue: Marked=0,0:00:05.00,0:00:07.00,YourStyle,Bill Gates,0000,0000,0000,,test
```

Change all the numbers to 0, and remove the random text you typed in, so that it looks like this

```
Dialogue: Marked=0,0:00:00.00,0:00:00.00,YourStyle,Bill Gates,0000,0000,0000,,
```

What you do now, depends on what kind of text editor you are using. But basically, you need to prepend the above line to each line in the translated script, so it looks something like:

```
Dialogue: Marked=0,0:00:00.00,0:00:00.00,YourStyle,Bill Gates,0000,0000,0000,,B - Where do you want to go today?
Dialogue: Marked=0,0:00:00.00,0:00:00.00,YourStyle,Bill Gates,0000,0000,0000,,S - Your house!
```

Repeat that for each translated line in the script. Don't worry about "B - " etc just yet, and definitely don't worry about changing "Bill Gates" to correct characters name. All done? Allright. Now, try loading the .ssa into Sub Station Alpha. It should load, if it says something like "x lines not parsed" it means you fucked up and did something wrong. Try again. Usually, if you just wrote a macro and repeated it for x lines in your translated script, everything should be fine.

Now that either of these methods successfully loaded the entire script into SSA, you have a dummy .ssa file ready for wav-timing, with all the script lines already entered, with zero or default timings.

4. wav-timing

This is the most boring part of the process. You listen to the audio, determine where dialog starts and ends, and set times accordingly. First off, switch to Wav timing mode. Goto Timing->Time From WAV File. Then press alt-o to get the "Load WAV" dialog, and load the wav you made in step 2. If SSA complains about "too short file to do anything sensible with", or complains about incorrect sampling rates or anything of that sort, double-check if you followed step 2 correctly and insert silence at the end of the file if necessary. Now you should have the file loaded, and see the waveform in the middle black preview area. Notice the y-scale and zoom sliders near the top. I don't know about you, but I like to keep my zoom level at around 128x and y-scale approximately at 66%.

All the stuff related to timing is on the keyboard, in fairly easy to use locations. In the wav-preview window, you have a yellow and red vertical line. Those two determine start and end, respectively, of the current line you are subtitling. Left-click to set the start time, right-click to set the end time. Press Alt-S to preview what it will sound. If you messed up, click until you get it right. After you are done, press Alt-G to grab the timings for the current line. SSA will auto-advance to the next line, moving the yellow/red lines 2 seconds away from your current sub. Adjust them again, and press Alt-G to grab the next timings. Repeat as necessary.

Other useful keys while in wav-timing mode:

Alt-R	plays current row. Use this to position yellow/red lines correctly when you want to fix up timings for a row that you already timed. You can then move the start/end times, and press Alt-G again to grab new timings.
Alt-V	play previous row. You can compare the speech flow and see if you cut off anything with your next sub.
Alt-X	play next row. Same thing, you can use it to quickly navigate around the timings.
Alt-S	the most important one. After setting start/end time, preview what the clip will sound with these timings.
Alt-G	once you are satisfied with the timings end/start, this saves the timing and advances to next line.
F	scrolls the currently timing wav forward a bit so that you don't have to move the mouse away from timing setting to scroll.
A	scrolls the currently timing wav backwards.
F2	save your current project. SSA will crash, and you will be pissed off if you just lost last 30 minutes of work.

Keep going line by line until you time all of your current project. If you need to split some lines (stupid translator makes them too long, too much shit on the screen, etc), select the row you want to split, and press Ctrl-S. Edit both lines of text to make the split logical (i.e. dont split in the middle of a sentence, etc). Don't forget to press enter on each change, otherwise SSA will lose it. Select the first split row, press Alt-R, set correct timings, and Alt-G them. Then press Alt-X and set correct timings for the split row. Joining lines is done by selecting one line, holding Shift, clicking on adjacent line, and pressing Ctrl-J. Save before you do this, SSA likes to crash while joining lines. If it crashes, cut & paste text from the line you want to join into the other line, and use Ctrl-X to delete the old line. If you notice, all the keys required for this process are on the left side of the keyboard in QWERTY layout, allowing you to perform all timing related operations with left hand, while selecting start/end times with the mouse with the right hand. This leads to improved timing efficiency especially once you get the hang of it. Timing about 15 minutes of video in 30 minutes during the first pass is quite common.

When you are done first-pass timing (this is what you just did), you are ready to get rid of SSA and start timing scene changes.

5. timing scene changes

Here, you need to make sure you followed step 0 and setup all the software correctly. Make sure you have a big screen, or know how to use your computer efficiently.

You will need to open 3 programs for this project, and be able to quickly multitask between them. Here, make sure that your .ssa file is named the same as your video. For example, if you are subbing "microsoft.avi", your SSA should be called "microsoft.ssa". Open up the .ssa in your favorite text editor that you are comfortable with. Do NOT open it in Microsoft Word or other "word processor". Near the top of the script, you should see the line such as:

```
PlayResY: 768
```

Modify that according to the actual horizontal resolution of the sub (For example, for a 640x480 source, change that line to 480). Keep in mind, SSA will over-write this value all over again if you ever open this file in Sub Station Alpha, (you shouldn't have to use SSA after the 1st pass timing though), so make sure that value matches with the video. If you are subtitling material which is not 4:3 (for example, wide-screen stuff) you need to set "PlayResX" as well. That option isn't there by default, you will have to add it. Set it to something like:

```
PlayResX: 720
PlayResY: 416
```

For subbing 16:9 dvds. Again, just use the X and Y values from your video source. The PlayRes? values will be very important later on during sub positioning.

Next, start VirtualDub, and load the video into it. (File->Open, etc) A lot of the raws these days come in two-pass Divx5, which uses quite a bit of CPU time during seeking. Since during this part you might have to seek quite a bit, and especially if you have a slower system, it might be beneficial to re-encode your project .avi into something with a lighter codec, such as say, MS-MPEG4V2. This is out of the scope of this document, other than the fact that you should have at the most 2 seconds between keyframes (more keyframes = faster to seek). Also, if you plan on doing any subs positioning later on, get the crop/resize settings from your encoder and use these to re-encode your video. Also, if the raw is 29fps and final version will be released as 23fps, ask your encoder which method will be used to decimate and duplicate it. Otherwise, you might get scene change bleeds of a couple frames which will look ugly and waste unnecessary bits in the final encode.

Now that VirtualDub is loaded, go to the Video->Filters menu, and press Alt-A (or click "Add" button). Scroll the list of plugins until you see "TextSub 2.18" (or some newer version), and press enter to load it (or double-click the plugin name). In the dialog that comes up, click Open, browse to the ssa file you just saved, set the FPS drop-down box to the framerate of the video you have loaded (use File/File Information in VirtualDub to find the framerate), and select Character encoding if necessary (I dont think it's necessary on Windows NT). Close all these dialogs, by clicking OK or whatnot. You should be back to the virtualdub main screen. Default setting of showing the processed video on the right side of the screen is dumb, so you want to go to Options->Swap Input/Output Panels to switch that order. Resize VirtualDub until all you see is a full frame border for your current project and the virtualdub seek bar. Now you should be able to get frame-accurate view of your subs.

Go back to your project directory, and use "mplayer2" (on WindowsNT) to load the .avi file. If you installed software from step 0 correctly, you should have dvobsub load and you should see dvobsub green arrow in the tray. Don't play the file yet, but go to File/Properties of the video player, click "Advanced" tab, select "DirectVobSub (auto-loading version)", and change its properties (double-click or click on properties button). Nothing really interesting here except one setting on the "Misc" page. Go there, and make sure that "Pre-buffer subpictures" checkbox is OFF. Otherwise, vobsub will cache the subs and you won't see some faster effects. Also make sure the checkbox below it is ON, that is, you want vobsub to auto-reload the ssa as you modify it. You only need to change these settings once, all consequent passes will re-use these settings.

At this point, you should have mplayer2, virtualdub, and your .ssa file loaded into your favorite text editor. Here comes the fun part. Without forcing too many rules, I'd like to say a couple things the way *I* time stuff. It might not be the way you want to time it or the way your group wants you to time it, so decide for yourself. It's been mentioned elsewhere before that small, short, clean lines are better than a sub that stays on for a minute and takes up 5 display lines. Another important thing, is not make subs appear for a realy short time for no reason. This is what I call "scene changes timing". You play the video in mplayer, and watch for subs cutting over a scene change and disappearing a split second after, another words, things like:

```
-----|sub|---------------|scene change|-|sub gone|-----
       <----------duration of the sub----->
```

Unless absolutely necessary, these are better be gone directly at scene change. When you see this mistake, hit "space" on your keyboard to pause the mplayer2. Note the timestamp in lower-right. Alt-tab over to VirtualDub, and press Ctrl-G to bring up the seek dialog. Enter the timestamp from media player in there. If you are not fast enough, you might have to enter the timestamp minus one second. You'll get better soon. Use right arrow and alt-right arrow to go to the scene change frame. You want subs to disappear exactly at scene change. Move to the frame where it changes to new scene. Not the frame right before the change. Note the timestamp at the bottom of virtualdub. In your text editor, find the sub which needs correcting (the one you are currently editing). Change the end time of it to the number you just got from virtualdub. Save the .ssa. Step back a couple frames in virtualdub to make sure your change worked. If you fucked up, re-do this step again. Rewind your preview video in mplayer2 and make sure that looks OK. It will. Proceed playing the preview video in mplayer2 until you see another mistake. Fix it. This process will take as long as the show is + about 20% for fixing errors etc. After this, scene change timing is done.

6. character name fixup

If you used the other method of loading your translated script into .SSA, you want to take your script which still reads like "B - Where do you want to go today?", and put those codes as character names in the script. Hopefully you picked the "default character" (in this case, Bill Gates), for the character who talks the most in this episode. Now you can search/replace "B - " with "" to fix Bill's quotes. Next, you'll need to write a macro in your text editor to do something like this:

```
Dialogue: Marked=0,0:00:00.00,0:00:00.00,YourStyle,Bill Gates,0000,0000,0000,,S - Your house!
Dialogue: Marked=0,0:00:00.00,0:00:00.00,YourStyle,Bill Gates,0000,0000,0000,,Your house!
Dialogue: Marked=0,0:00:00.00,0:00:00.00,YourStyle,John Smith,0000,0000,0000,,Your house!
```

So basically, search for "S - ", delete it. Search back for "Bill Gates", retype with "John Smith". Whichever way you do it, my text editor handles it fine. Write a couple macros for the rest of the characters. If you can save the macros, you can reuse them later in sequential timing projects. After this replace, make sure the subs still play, and you should be done with timing completely.

7. intro to typesetting

The same boring ugly yellow arial is no way to win yourself fans. Arial is a great font, but yellow and that huge black border is hardly ever a good combination (commercial subs uses it because there isn't much of a choice with DVD sub pictures). Watch the show for a while and come up with a good color/font combination you can use that doesn't make your viewers sick. Font size is pretty important. There's a difference between readable and too big, and that difference is around the number 26. Arial at 26pt on a 640x480 video looks just about right. Not too big, not too small. If you forgot to edit PlayResY though, it'll look too small. That's your hint you fucked up. I warned you. Avoid too small text, it'll be harder to encode. Avoid too large text for obvious reasons. Vobsub will come with docs which contain ass-quickref.txt, which you should read and understand if you are going to do any serious typesetting. It's all explained in that .txt file, no sense to repeat it here.

The best reference on the effects available in TextSub is installed as part of the VobSub documentation. In vobsub install directory you should have a docs/ directory with ass-quickref.txt and ass-specs.doc which describe all the control codes and functions in detail. Read those files and experiment, and figuring things out should be pretty easy.

Many people somehow think that all the cool text rotation, alpha, moving, etc effects are done from inside the SSA using some hidden menu options that only gurus know about. Well, that isn't true. None of the latest TextSub effects can even be modified from inside SSA, and I am pretty sure these days the only use for SSA is only for timing, and not too many people actually still use it for typesetting of any kind. The way it is now, the best choice for typesetting is still a text editor with a copy of ass-quickref.txt open in another window.

And now you are done. Pack your script and send it along to a typesetter or a encoder (depending on how much typesetting you did), and relax. With practice, the entire process outlined here should not take more than a few hours for a 20-22 minute show.

8. encoding basics

Now, encoding process itself deserves the whole guide which I am not going to waste time with, there are plenty of other places describing the process in complete detail. What I'll focus on here is some preparations and useful things you can do before/during encoding to make your life easier.

Avisynth - The scripting "language" of avisynth is easy to learn, and you can do basic things like cropping/resizing/frame duplication/fixing, as well as more advanced stuff such as decimation/deinterlacing/etc. If you have a choice of doing it in Avisynth, do it. There's some avisynth native filters for image processing and you can even load VirtualDub filters (though not too efficiently since VirtualDub filters operate in RGB space where most of Avisynth filters operate in YUV. Doing all your processing in the avisynth script allows you to use fast recompress during encoding.

Fast recompress - If you are not processing your video source or only have to crop and resize, do it from the avisynth script.

TextSub avisynth version - Already have a clean source video or pre-filtered huffy source and only adding subs? Create a avisynth script with AVISource() and TextSub() and use Fast Recompress during encoding.

Huffyuv codec - If you are running a lot of cleanup filters on the source video, decimating, or performing a lot of processing on the source, there is no sense to run these filters twice, if each pass takes 5 hours. Do a "filter" pass saving results to a huffy-compressed .avi then do fast-recompress two-pass encoding. Use the jobs control in virtualdub/nandub for this.

Last updated: 2002/10/11 contact