Adding new tag data to the database

Mark Johnson
markjohnson@st-andrews.ac.uk
24 June 2007

0. Set up the correct directory structure on your PC and set the clock time to the correct local time. Make sure you have the latest copy of the tag tools in a directory that is listed on the Matlab path variable (use File -> Set Path -> Add with Subfolders to include the toolbox and all of its sub-directories to the Matlab path). All tools are known to work on Matlab version 6 and 7. Please report any problems.

1. collect the *.dtg* files in a directory called after the tag deployment name e.g., *pw06_091d*. Place this directory in a directory named after the species and year e.g., *pw06*. From hereon the full name of the deployment as a Matlab string, e.g., '*pw06_091d*' will be referred to as ***tag***.

2. run *ffsrdall* on the *.dtg* files you want to access. Check that the last samples of each chip match the first samples of the following chip. Enter the first and last sample values on the offload data sheet. If bad chunks are reported, note the block number of bad chunks and the reason for failure e.g., bad header CRC, bad data CRC etc., on the offload data sheet (CRC stands for cyclic redundancy check, a kind of error detection code or parity). If bad matches are reported there is no need to record these on the offload data sheet (they are almost always due to periodic changeovers between mono and stereo recording enabled with the 'Conditional' option in the tag *aset* menu).

3. Notify Matlab of the location of data. Startup Matlab and enter:

```
settagpath('audio',path,'cal',caldir) ;
```

where: ***path*** is the absolute path to the base directory of the audio data, e.g., *'g:/data'*. The subdirectory *'g:/data/pw06/pw091d'* will automatically be added when forming file names. ***caldir*** is the directory to put the calibration files e.g., *'c:/tag/tag2/matlab/cal'*.

4. Generate the cue table for the deployment:

```
N = makecuetab(tag) ;              % read and process the .txt log files
savecal(tag,'CUETAB',N) ;          % add the cue table to the cal file
```

You will be prompted to enter the tagon time as a 6 element vector of:
[year month day hour minute second]
corresponding to the tag start time reported by the DTAG *scan* function.

5. Add location dependent information to the cal file:

```
savecal(tag,'GMT2LOC',hours) ;     % add the GMT time offset to the cal file
savecal(tag,'TAGLOC',position) ;   % add the tag-on position to the cal file
savecal(tag,'DECL',decl) ;         % add the local magnetic field declination
                                   % in decimal degrees to the cal file
```

where: ***hours*** is the number of hours from GMT time to local time.
position = *[latitude,longitude]* is the tag-on location in decimal degrees.
decl is the local magnetic field declination value in degrees (see magnetic field maps or local charts for this value).

6. Make the raw decimated sensor data file:

```
        settagpath('raw',rawdir)           % add raw directory to paths
        [s,fs] = swvread(tag) ;            % read the .swv files
        saveraw(tag,s,fs)                  % save a raw file
```

where ***rawdir*** is the directory to put the raw sensor files e.g., *'c:/tag/data/raw'*

7. Make the tag frame prh file:

```
        settagpath('prh',prhdir)           % add prh directory to paths
        [s,fs] = loadraw(tag) ;            % if workspace was cleared after step 5
        CAL= tagxxx ;                      % read the calibration for the device used
```

where ***tagxxx*** is the tag name of the device used for this deployment e.g., *tag212*.
prhdir is the directory to put the prh files e.g., *'c:/tag/data/prh'*

7.1 Pressure calibration:

```
        [p,tempr,CAL] = calpressure(s,CAL,'full') ; %follow screen directions
```

You will be asked to pick the data points on the display that correspond to times when the whales is at the surface. These are the lowest points in the graph and usually form a distinctive broken line that is roughly horizontal but will often have some curvature. Select points that follow this line by drawing a rectangle over the desired points using the left mouse button. Make sure to highlight points over the entire length of the broken line.

7.2 Magnetometer and accelerometer calibration:

```
        [M,CAL] = autocalmag(s,CAL) ;              % accept or reject test results
        [A,CAL] = autocalacc(s,p,tempr,CAL) ;      % accept or reject test results
```

Repeat the above steps until the reported standard deviations are satisfactory (e.g., for M try to get below 0.5; for A try to get below 0.04). Bear in mind that some tag placements and especially smaller animals lead to large A standard deviations.

When you are comfortable with the results run:

```
        savecal(tag,'CAL',CAL)                     % save calibration results
        saveprh(tag,'p','tempr','fs','A','M')      % save tag frame results
or:     saveprh tagname p tempr fs A M
```

where ***tagname*** is the same as ***tag*** but without the string designators e.g., if ***tag*** = *'pw05_199a'*, then ***tagname*** *= pw05_199a*.

8. Determine the tag orientations on the whale:

```
loadprh(tag)                          % if workspace was cleared
T = prhpredictor(p,A,fs,mindive,1) ;  % estimate tag orientations
```

where ***mindive*** is an optional argument restricting attention to the edges of dives deeper than ***mindive***. If you want to focus attention on deep dives (which tend to provide the best tag orientation estimates), use a value for mindive that is just lower than the shallowest dive you want to analyse. Note that *prhpredictor* is the new name for the script that used to be called *prhpredictor2* (which no longer xists). Use method 2 in *prhpredictor* if the animal actively swims at the surface. If the animal predominantly logs at the surface, use method 1.

9. Produce the orientation table. By examining the prhpredictor results, decide on the orientation(s) of the tag and whether it slides during the deployment. If you suspect that the tag moves during a dive, plot the *A* matrix (or it's 2-norm *norm2(A)*) to see if there is any indication of an impact or rub consistent with a sudden move. Using the tag orientation worksheet, construct the orientation table:

```
OTAB = [move1;move2;move3...]
```

where each row corresponds to a move of the tag and has the form:

```
moven = [t1,t2,p0,r0,h0]
```

where *t1* and *t2* are the start and end times of the move (in seconds-since-tagon), and *p0*, *r0*, *h0* are the new orientation Euler angles after the move (in radians). If the move is instantaneous, use *t2=t1*. If you are not sure if there was a move or you simply want to notify a time at which the tag was at a certain orientation, use *t2=0*. If the move time is uncertain, note this on the orientation worksheet.

Test the *OTAB* by creating the whale frame acceleration signals:

```
[Aw Mw] = tag2whale(A,M,OTAB,fs) ;
```

Inspect *Aw* graphically or using *prhpredictor* to assure that all moves are handled correctly. When you are done, save the *OTAB* to the cal file using:

```
savecal(tag,'OTAB',OTAB)              % save orientation table
```

10. Produce the prh file:

```
makeprhfile(tag)
```

This function implements the full calibration from raw to tag- and whale-frame taking values from the cal file. It is a useful function to read to verify the individual steps.

Directory and file structure

The following directories are recognized by settagpath:

raw contains the raw decimated sensor data files with names *tag*raw.mat. Each file contains: *s*, *fs*.

prh contains the derived sensor data files with names of the form *tag*prh.mat. Each file should contain:

p	depth vector. Unit is meters of H_2O (salt).
tempr	temperature vector. Unit is degrees Celsius
A	tag frame accelerometer matrix (nx3). Unit is $g=9.81ms^{-2}$.
M	tag frame magnetometer matrix (nx3). Unit is micro Tesla.
fs	sampling rate of all derived sensor time series. Unit is Hz.
Aw	whale frame accelerometer matrix (nx3). Unit is $g=9.81ms^{-2}$.
Mw	whale frame magnetometer matrix (nx3). Unit is micro Tesla.
pitch	navigation-to-whale frame pitch angle. Unit is radians.
roll	navigation-to-whale frame roll angle. Unit is radians.
head	navigation-to-whale frame head angle. Unit is radians True (i.e., a heading of 0 implies that the whale is pointing to true north, not magnetic north).

cal contains the calibration values specific to each deployment. Filenames are of the form *tag*cal.mat. At the end of the process, each file should contain at least:

CAL	structure containing deployment specific calibration values for the sensors.
CHIPS	a vector containing the chip numbers recorded in the deployment.
CUETAB	conversion data from cues to sample and file number for audio data.
DECL	local magnetic field declination angle in radians.
GMT2LOC	number of hours between local time and GMT. This is the value that should be added to GMT to obtain local time.
TAGON	[yr month day hr min sec] in local time of the tagon time.
TAGLOC	[latitude,longitude] of the tagging location in decimal degrees.
OTAB	matrix of tag orientations during the deployment.

audio contains subdirectories with the data files unpacked from the archive (.dtg) files. The subdirectories are named according to the species and year, e.g., pw05. Within each sub-directory are sub-sub-directories, one for each tag deployment with names of the form: pw05_199a. Within these directories are the .wav, .swv, and .txt files unpacked from the .dtg files. The .dtg files can be in the same directory if desired. Example file structure:
g:/data contains: pw04, pw05, sw04, sw05 etc.
g:/data/pw05 contains: pw05_199a, pw05_200a, pw05_200b, pw05_201f etc.
g:/data/pw05/pw05_199a contains: pw199ann.wav, pw199ann.swv, pw199ann.txt with nn=01..24.
In this case, audio = 'g:/data'

audit: contains the results from tagaudit. Filenames are of the form *tag*aud.txt.

Hints and troubleshooting

Put all of the *settagpath* instructions in a file called *startup.m* in the same directory as all of the tag tools. Matlab will run this file automatically everytime it starts up saving you the effort.

Is the directory of the tag tools on your Matlab path?
Have you used settagpath to point to the correct directories?
Is the directory structure correct?
Do you have another, older copy of some of the tools in your working directory that are being called instead of the correct tools?