Getting started with position calculation

Preliminary remarks

Position calculation can be performed with three different programs that will be referred to as Tapad (developed by Andreas Zierdt, Munich), Kalman (developed by Korin Richmond, Edinburgh, with interface to the matlab system by Phil Hoole), and Calcpos (the program provided with the AG500 by Carstens).
Here is some basic information on each:

Calcpos

Advantages

No special procedures required to make it available

Easy to use, because currently no user intervention is required

Disadvantages

Originally, my understanding was that Calcpos can only be run on the specific computers (the so-called 'control server' and 'lida') provided by Carstens as part of the AG500 installation. I recently discovered that this was not correct. Procedures for running Calcpos on other Linux machines are given in the footnote[1].

If, however, the control server and lida are the only linux machines you have, then some fiddling around may be necessary to incorporate the data into our matlab framework:

If the experimental data is moved to a normal network server or work-station at the end of the recording session (highly recommended!) then it will have to be moved back to the AG500 control server (particularly if filtering is carried out as outlined above) for position calculation, and the results retrieved again afterwards.
(Alternative: try and mount a network location to be visible to the local file system of the AG500 control server)

In addition, if the results are to be further processed within my matlab framework then they have to converted from the raw binary format produced by Calcpos to mat files (see do_rawpos2mat.m below).

The ease of use is actually also a disadvantage. The position calculation sometimes produces unstable results, and in these difficult cases Tapad offers more flexibility. This is the reason why we have used Calcpos very little. The other reason is that we have had access to the Kalman program as an alternative procedure. Experience has shown that it is extremely useful to be able to compare the results of different procedures (even though it means more work), because it makes it much easier to

[1]To run calcpos on additional linux machines the first step is to copy the executable files from the control server. The easiest way is to copy the /opt/age directory (with all its subdirectories) to a convenient location on the linux network. If necessary, use the linux command chmod to make sure that all files in age are still marked as executable after the copying. The just-mentioned steps only need doing once. Assuming you have matlab available on the linux network then a convenient way to run calcpos is to use the matlab function calcpos4mat. An example of a wrapper script showing how this function is used is do_calcpos4mat (note that this version of the wrapper script automatically includes the conversion of the binary position files to matlab outlined for the wrapper script do_rawpos2mat in the main part of the text below).

identify the unstable cases mentioned above.
Thus, in labs without access to Kalman there is a lot to be said for using Calcpos to get a first quick impression of the data, and as a cross-check on Tapad.
We will explain below how this could look in practice.
A parallel version of this document explains the procedures when Tapad and Kalman are used in parallel (the usual case at IPS over the last few years).

Tapad

Advantages

Can be used on any machine where matlab is available (any operating system)
Completely integrated into my matlab processing framework.
More flexible than Calcpos, but this means that the user has to be prepared to devote some thought to issues such as choosing the most appropriate start positions (more discussion of this below).
The underlying optimization algorithm is probably more robust than that used by Calcpos.
Tapad uses the matlab optimization toolbox, so in principle any enhancements to this toolbox could be used by Tapad if they appeared useful.
Source code is available.

Disadvantages

Very slow! (but calculation can be easily spread over several machines or instances of matlab)
The matlab optimization toolbox has to be purchased.
(Note that, in addition, my matlab functions make a lot of use of the matlab signal processing and statistics toolboxes.)

Kalman

Advantages

Very fast (close to real time for 12 sensors at 200Hz on current machines)
Uses a fundamentally different algorithm from Tapad and Calcpos. Thus, makes a particularly good combination with Tapad. We have had cases both where Kalman provided a reasonable solution when Tapad failed, as well as vice-versa.
Less likely than Tapad (or Calcpos) to produce really wild errors, because it inherently preserves the continuity of human movements (but this means bad results may be less obvious, because the resulting movement patterns usually look plausible)

Disadvantages

Not yet freely available
Only runs on Linux

Using Calcpos and Tapad in parallel

We will now illustrate a typical kind of work-flow using both Calcpos and Tapad.

Assume that the do_filteramps script has been used to generate raw binary versions of the filtered amps (located in the directory ampsfiltraw/amps). These amp files must be made available to the calcpos program. The output of this program consists of new files named 0001.pos, 0002.pos etc. in new directory named rawpos. This directory should then be moved

if necessary from the AG500 control server to the ampsfiltraw directory (the rawpos directory also contains a further subdirectory named posamps which we will be concerned with later).
These new .pos files must then be converted to mat files.
The wrapper script for this purpose is named do_rawpos2mat.m. The base version of this script should be retrieved and placed in the current main working directory, i.e the directory immediately above ampsfiltraw. Normally, the only thing that needs setting is the list of trials to process[2].
(Note: if rawpos2mat finds any data with high rms values (default threshold is 30), this will be set to NaN ('missing data') in the output mat files.)
After this script has been run there will be a new sub-directory within ampsfiltraw named rawposmat. This contains files named 0001.mat, 0002.mat etc.

Assessing the results
Several procedures are available for numerically and graphically assessing the quality of the results. These leads to a new wrapper script 'do_do_comppos.m'. We will start with a very basic version and gradually add to it as we move through each processing stage.

One specific aim will be to use the Calcpos results to get an initial estimate of the typical position and orientation of each sensor, and then provide this information to Tapad as a set of start positions.

Open the script in the editor and make sure that the variable 'doshowtrial' is set to 1
Also check that the variable 'basepath' points to the location of the matlab position files.
For the present example it should be
```
basepath=['ampsfiltraw' pathchar 'rawposmat' pathchar];
```
Make sure that 'triallist' is set to an appropriate range of trials to process (look for '??' in the template file).
If not all channels have actually been used in the experiment change the specification for 'kanallist'.

Run the script and then find the figure that shows values for position, orientation, rms, tangential velocity and nan-count over trials.

Try and determine a range of trials that give stable results, and find out if any sensors have particular problems.
To help with this use the program 'showstats'. This gives a numeric summary of results for selected trials or channels. Use its mode 1 to check that the relative position and orientation of the sensors makes sense.
When a group of stable trials has been identified, run showstats with mode 7. You will be prompted to give a filename in which start positions will be stored.
(As with any matlab function, type 'help showstats' in the command window to get the details of how to call this function and what information it provides.)

---

[2]If the matlab-callable version of calcpos is used (as outlined in the previous footnote) then rawpos2mat is called automatically by the do_calcpos4mat wrapper script.

These start positions will then be used to in effect bootstrap the Tapad algorithm, i.e. to increase the chances that it will easily be able to converge on a reasonable solution.

To set up position calculation with Tapad do the following:
Load the file do_tapad_ds_base.m and save as do_tapad_ds.m. Set the appropriate values for 'startfile' (i.e. the name of the file you just stored above) and 'triallist'. Otherwise, nothing should need changing.
Run the procedure. It will probably take a lot longer than Calcpos, even though we are using a downsampled version of the data here.

Check the results of the Tapad version as done previously for the Calcpos version.
This will mean setting the path in do_do_comppos to[3]

```
basepath=['ampsfiltds' pathchar 'beststartl' pathchar 'rawpos'
pathchar];
```

If there is any suspicion that either the Tapad or Calcpos solution has instabilities then do_do_comppos can be used to compare them.
Set doshowtrial to 0. Set basepath to point to the Calcpos solution, and set altpath to point to the Tapad solution. This displays the two solutions in parallel. The distance between the two solutions is given in the panel labelled 'Euclidean distance'. If the solutions ever diverge widely, i.e by more than a millimeter or two, then one can try and make a preliminary decision as to which version is more plausible. More details on this display mode of do_do_comppos are given later. [4]

[3]The 'beststartl' part of the path is a relic of the time when we were experimenting with different settings in Tapad. 'beststart' refers here not to the spatial start positions but to the fact that Tapad makes a preliminary estimate of the temporal position in the trial where a robust solution seems easy to find, and then works forwards and backwards from there. This technique has proved very useful, but it works best when the trials are fairly short, i.e. no more than one utterance or breath group. For experiments where the trials are much longer then it may be worth using the function newstudy.m to subdivide them as a first processing step (after the amp files have been converted to mat-file format). The 'l' (n.b lower-case 'L') in 'beststartl' refers to the use of the **L**evenberg-Marquardt version of the non-linear optimization.

[4]Since in this example downsampled data (Tapad) is being compared with full data (Calcpos) an absolutely perfect match is not to be expected, particularly in regions where movement velocity is high.