What you need to know to use *Silbido*

Release date: 2011-10-07

Note that the setup instructions must be followed before instructions in any of the guides will work. Most of the instructions in this guide need only be done once, except for calling “triton; close all” which must be done each time Matlab is started if *silbido* is to be used.

Introduction

Silbido is a software system that consists of several components:

1. An automatic contour detector developed for odontocete whistle detections.
2. Annotation tools for manual annotation of tonal signals and/or viewing/editing automatically detected tonals.
3. A scoring module to permit comparison between a reference “ground truth” set of detections and another set of detections.

Silbido is the Spanish word for whistle. The name for the system came about when several of John Hildebrand’s lab members thought that I needed a snappier name than “graph search whistle detector.” I’ve since started thinking of *silbido* as referring to the collection of algorithms we have developed for annotating and detecting frequency modulated tonal calls as well as analyzing the performance of detectors of . Please note that in its current form *silbido* is not appropriate for closely spaced pulsed calls that exhibit closely spaced harmonics or harmonic artifacts. In other words, *silbido* will not work well on burst pulses or other calls that exhibit rich harmonic patterns in a spectrogram such as many primate vocalizations.

The basic ideas used in this collection of algorithms can be found in the publication:

Roch, M.A., T.S. Brandes, B. Patel, Y. Barkley, S. Baumann-Pickering, M.S. Soldevilla (**2011**) Automated extraction of odontocete whistle contours. *J. Acous. Soc. Am.*, **130**(4), 2212-2223.

Please note that the version that you downloaded contains changes since the system was published. A couple of changes have been introduced that have resulted in a reduction of performance and we are working on rectifying this. As a number of people have expressed interest in the scoring utilities used in the 2011 Detection, Classification, and Localization of Marine Mammals Using Passive Acoustic Monitoring ground-truth information and scoring tools, we are releasing the current version as is and will provide periodic updates.

If you would like to compare the performance of *silbido* to your algorithm, you are welcome to contact us for the version that was published. I elected to distribute the most recent version as it contains a number of changes that are useful, better comments, and also has a beta version of a new annotation tool that we believe is easier to use than the previous version.

Best regards,
Marie Roch
e-mail: usefirstnameDOTuselastname @ sdsuDOTedu

For the short of patience

You will need Matlab 2008b or later, the DSP and image processing toolboxes. You will need to increase your Java heap space and must invoke a software tool called triton or manually set up both Matlab and Java paths before using any of the tools. Continue reading for a gentler introduction to the tool set.

Software requirements

Silbido is written in a combination of Matlab and Java. Some of the noise routines using the Matlab API for C/C++ and require either a C/C++ compiler or a precompiled version which is provided with this distribution.

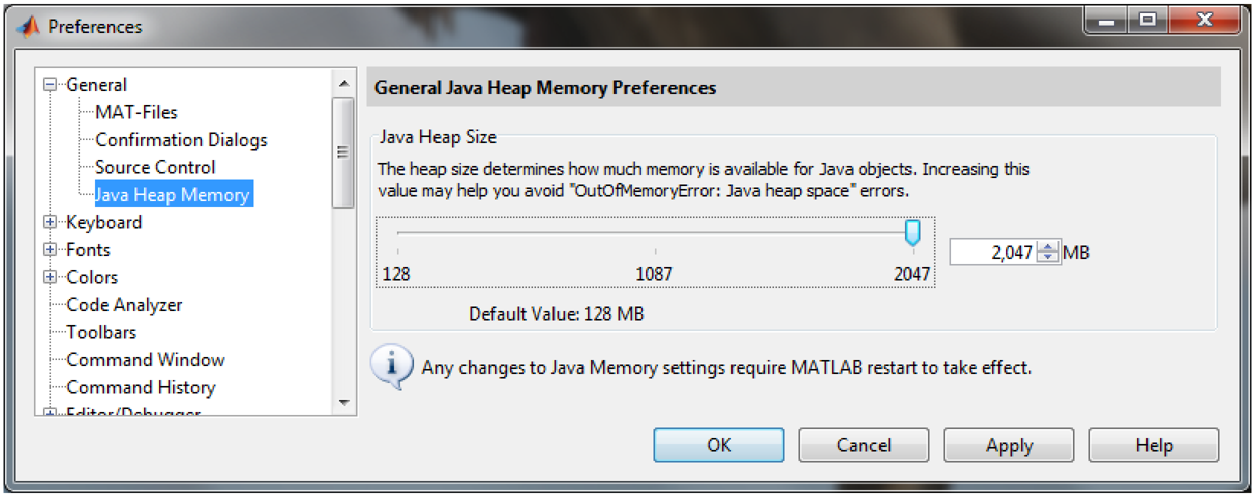
Silbido has been tested with Matlab 2008b, 2010a and to a lesser extent with 2011a. The signal and image processing toolboxes are required; although the detector can be used without the image processing toolbox if a different noise removal technique than the default median filter is specified.

Setup

Silbido is distributed with a modified copy of Triton, a software package used at the Scripps Whale Acoustics Lab. Triton is authored primarily by Sean Wiggins, but the version distributed here has been heavily modified by Melissa Soldevilla and myself. *Silbido* is not integrated into triton but this is expected to happen in the future. You are welcome to use triton, but the mainline code maintained by Sean Wiggins may have features not supported in our derivative version and vice versa.

Silbido requires a larger Java heap space than Matlab provides by default and it is recommended to provide at least 1 gigabyte of heap space (preferably two if running on a 64 bit operating system) to Java. Note that Java shares the process space with Matlab and memory allocated to Java is memory that Matlab cannot use for storing data.

In modern versions of Matlab, this can be done via the General/Java Heap Memory dialog under File→Preferences and requires a restart of Matlab before changes take effect:



General Java Heap Memory Preferences

Java Heap Size

The heap size determines how much memory is available for Java objects. Increasing this value may help you avoid "OutOfMemoryError: Java heap space" errors.

128 1087 2047 2,047 MB

Default Value: 128 MB

Any changes to Java Memory settings require MATLAB restart to take effect.

Alternatively, one can increase Java heap memory by editing a java.opts file. To do this for all users of a machine, type the following:

```
>> edit(fullfile(matlabroot, 'bin', computer('arch'), 'java.opts'))
```

Note that you must have appropriate permissions to modify this file.

For a single user, create the java.opts file in the Matlab startup directory. If you wish to change the directory where Matlab starts in Windows, you can create a shortcut and edit the Start in property of the shortcut. No special permissions are required for this.

Contents of java.opts file:

Type “ver java” at the Matlab prompt to determine your version of Java.

For Java virtual machines 1.2.2 and newer, use the following:

```
-Xmx1024m
```

For earlier Java virtual machines (up to 1.1.8) use:

```
-mx2048m
```

`java.lang.Runtime.getRuntime.maxMemory/(1024^2)` will display the amount of memory allocated to Java’s heap in megabytes.

Using silbido

To use *silbido*, you need to add the triton folder to your Matlab path (you need not add its children). You will need to start triton once each Matlab session which will set up remaining paths for both Matlab and Java code. If you do not plan on using Triton, a good way of doing this is to type

```
triton; close all
```

at the Matlab prompt. This will add the appropriate paths for both Java and Matlab to your instance of Matlab. Afterwards, follow the instructions in one of the following instruction manuals:

- Annotation – Second generation annotation tool for producing ground truth detections or viewing/editing detections. This one is not as well tested as the first generation tool and likely to still contain serious flaws. We use it in production mode for viewing annotations, but still use the older dtPlotUIGroundTruth tool for serious annotation. Use it for annotating at your own risk. If you find reproducible problems, we are interested in fixing them. If you find fixes, even better.
- GroundTruth –First generation tool for producing ground truth tonals.
- Detector – Automated detection of tonal signals.
- ManipulatingTonals – Information on accessing the data contained in a set of detections.

There is also a slightly out of date copy of the Triton manual modified in the triton directory. For the most recent copy of Triton (that does not contain our modifications, but has some new functionality and bug fixes), visit http://cetus.ucsd.edu and follow the technologies link.