# The Cricket-Tracking Project
# a case study

J Christmas

May 28, 2012

## Contents

# 1 Introduction

This document describes a case study for the archiving of a project to the Exeter Data Archive (EDA). The project described here presents a number of challenges for the archive and for the process of recording its information. The first challenge is that some of the information is too big to upload into a website (of the order of 20Tb). Other challenges are the number of different types of information and the dependencies between them.

We start, in section 2, by describing some terms and definitions that will be used in the document. In particular the words *data* and *dataset* have particular meanings in the context of a Computer Science project which may differ from those used within EDA and in other university departments. In section 3 the project is described, including what sorts of content it has generated, and section 3.3 lists the different file formats used. The process by which the project's content has been grouped together for entry into EDA is described in section 3.5, which also details how the EDA entries have been constructed.

# 2 Terms and definitions

To avoid confusion with words and phrases used within EDA and its forms and documentation, I define here the meanings that should be attached to some terms as they are used within this document:

| | |
|---|---|
| file | - a computer file; the basic unit of content. |
| data | - a specific type of content: quantitative (or possibly qualitative) measurements obtained by experimentation. |
| dataset | - a file that contains data. |
| program | - a computer program (as opposed to a television *programme*, for example). |
| EDA item | - an EDA item record, including the metadata and its attached file(s). The word *item* on its own has the usual English meaning. |
| content | - files that are attached to an EDA item. |

In addition is is useful to record some definitions for the building blocks of EDA. A **Collection** is a container for one or more EDA items that have related content. **Communities** and **Sub-communities** are descriptors which provide a hierarchical structure for the content and each of them may contain Collections and/or other Sub-communities. In EDA the Communities are generally named after the colleges. For the College of Engineering, Mathematics and Physical Sciences (CEMPS) Community, the Sub-Communities are currently named after the research institutes and the Collections after the departments, but this is likely to change; a more likely hierarchy is college–department–research group–project.

Files may be attached to more than one EDA item, an EDA item may by linked to more than one Collection and a Collection may be linked to more than one Community or Sub-community. This enables EDA to capture the structure of cross-discipline projects such as mine.

# 3 The cricket-tracking project

## 3.1 Overview

The project is concerned with automatically locating and tracking wild field crickets (*Gryllus campestris*) in digital videos. It is a NERC-funded collaboration between the Centre for Ecology and Conservation in the College of Life and Environmental Sciences (CLES) and the Computer Science department in the College of Engineering, Mathematics and Physical Sciences (CEMPS). The team members are as follows:

| | | |
|---|---|---|
| Prof Tom Tregenza | primary investigator | CLES |
| Prof Richard Everson | co-investigator | CEMPS |
| Dr Rolando Rodríguez-Muñoz | research fellow | CLES |
| Dr Jacqueline Christmas | research fellow | CEMPS |

The crickets inhabit a field in Spain where motion-triggered digital video cameras have been sited above the individual burrows, recording any movement that happens in the near vicinity of those burrows. The result is hundreds of thousands of hours of video segments which may or may not show cricket activities. The project is to develop programs that will automatically (i.e. without human intervention) identify, locate and track crickets in these videos.

## 3.2 Content

The project has produced a large number of files that need to be archived. These include, but are not limited to

1. The original videos recorded over a number of years (2006-2011 so far). They run to 20Tb, which is too big to be uploaded into EDA via the web front end, and are stored on about 15 internal and 3 external hard drives. The videos are in a proprietary format, which means that anyone who wishes to view them will need a licensed software package and a version of Microsoft Windows XP to be installed. The project holds licences and we would wish to store versions of this software for security. A new set of videos will be generated each year, but these are likely to be in non-proprietary formats.
   The proprietary format keeps together videos generated by one camera, but avoids creating enormous files firstly by separating video from audio and secondly splitting them into smaller-sized chunks. All of these files must be present in order to view the video; it is not possible, for example, to view the content of just one of these files.

2. Each year's videos have associated with them some manually recorded data detailing, for example, how and where they were collected, local weather conditions, etc..

3. Since the proprietary video format is not able to be read by anything other than the licensed software, the project paid a third-party company to convert selected video segments into the non-proprietary jpeg format, which resulted in one jpeg file per video frame. Each of these files can be viewed as separate images, but it makes sense to keep them together.

4. One result of a research project is publications. The publications themselves should already be in EDA as they are uploaded automatically by Symplectic, but we may choose to create an EDA items whose attached content files contain supporting information for

those publications. For this project this supporting information might be videos, graphs, individual images and such like.

5. One output of this project is the tracks identified in each video and classifications of whether the tracks are crickets or other things. This data needs to be associated with some documentation that describes what the data represents and what format it is stored in.

6. The tracks and classifications are produced by a suite of programs. These are predominantly written in Matlab, with one or two Linux shell scripts. Programs generally should be archived as source code. For Matlab (and Linux shell scripts) this is all that is required; for other programming languages, such as C, operating system scripts and/or written instructions for compiling the programs must be recorded alongside the source code.

7. The project has also generated a number of programs for visualising the data, for example by combining videos, tracks and classifications. They provide useful shortcuts for other researchers to view the results and the basis for understanding them.

8. Each program requires some documentation to describe what it does, what it expects its inputs to look like and what its outputs are. In addition the suite of programs needs documentation that describes how to run the suite, for example in which order the programs must be run.

These files may be divided into the following categories: primary data (e.g. the videos), secondary data (e.g. the tracks), publications, programs and documentation.

### 3.3 Files and formats

When archiving the project care needs to be taken in deciding which file formats to use, bearing in mind that the aim is that people should be able to access and use those files perhaps up to 15 years in the future. Sometimes the format is unavoidable, but where possible non-proprietary formats should be used. A basic (ASCII) text file is more likely to survive than a document in Microsoft Word 2007 format, for example. Table 1 lists the file formats used in this project and some comments about them.

### 3.4 Large files

The original videos are much too big and too numerous to be uploaded into EDA through the website. Zipping them into meaningful batches (perhaps based on date and camera) would reduce the number of files but increase the volume in each one to the point where they would not be downloadable from the site. In order to get them into EDA, the hard drives have been connected directly into the EDA server and the files transferred by the technical support team directly onto its disks. What has not yet been decided is how to make them available for attachment to EDA items; new functionality will be required within the submission form.

This size problem is likely to remain an issue as the means and speed of generating vast experimental datasets is increasing all the time, perhaps particularly in Bioscience where things like DNA sequencing and the increasing use of imaging and video analysis is leading to ever-expanding requirements for storage.

Table 1: The formats of files produced by the cricket tracking project.

| *type* | *format* | *comments* |
|---|---|---|
| video | .bix/.box | This is the proprietary format in which the original videos were recorded. The .bix file contains video, while the .box files contains the accompanying audio. |
| | .avi | While this is a standard video format, it is a wrapper that may actually contain videos in many different underlying formats. The underlying format must be selected carefully. |
| image | .jpg | A standard, compressed image format. |
| | .eps | Postscript format. |
| | .pdf | Adobe Acrobat, Portable Document Format. |
| document | .tex/.pdf | Rather than storing the documents in the proprietary Microsoft Word format (.doc or .docx), the documents in this project have been generated using Latex. The .tex file is a text file in which the document text appears along with various formatting commands. The .pdf is the Adobe Acrobat Portable Document Format file generated from the Latex. |
| program | .m | Most of the programs in this project have been written in Matlab. The .m files contain the program code in text format. |
| | .ksh | Some of the programs are written as Linux shell scripts, which are also in a text format. |
| data | .mat | Since most of the programs in this project have been written in Matlab, data produced by them tends to be in Matlab's proprietary (compressed) format. However, these have been shown to be relatively long-lived, with old files able to be read by the most recent versions of Matlab. Key result data has also been extracted into the more standard .csv format (see next format). |
| | .csv | This is a text file in which the data appears in the form of records or rows and the values in each record are separated by a comma. This format is easily viewed in, for example, Microsoft Excel. |
| grouped | .tar | The Linux/Unix equivalent of a zip file. This is a method of storing multiple files of different types within one file. |

## 3.5 Archiving the project in EDA

When considering how to archive the project we need to look first at the EDA item level information. Currently all the content attached to a single EDA item must share the following information:

- language
- title
- owner

- list of contributors
- funder and grant number
- related resources
- rights and restrictions
- embargo date and reason

All of the content for this project share the same information at this level, so could, in theory, all be attached to the same EDA item. At present there is no field in the EDA item form to describe the nature of a contributor's contribution. If this field becomes available then it will result in different EDA items for each different set of contributions. For example, Rolando recorded the original videos, so he is the primary contributor for them, but Jacq developed the programs, so she is the main contributor for them. Since this field is not currently available, each contributor will be recorded equally on all the EDA items.

Now we must look at the immutability of the content. The primary and secondary datasets and the documentation that described them will never be changed or superseded; likewise published papers and the data and results that are described in them. It is possible that any of the programs could be updated at some later date, perhaps implemented in a different programming language or just updated to include more functionality. Any such changes would require that the documentation for those programs is also updated. In these cases it would not be desirable to create a new EDA item that has attached to it absolutely all the project content again, so this is a reason for archiving the project as more than one EDA item.

Another consideration is the number of files to be attached to an EDA item and how a future user might want to access them. Each year's cricket data is stored in multiple datasets based on which camera and which computer they were recorded on, and each dataset has a maximum size so that long videos are split up into multiple files. In addition, audio and video are stored separately. If all 20Tb of videos were to be stored as a single EDA item either the item would have potentially hundreds of attachments or, if we chose to zip them up into a single attachment, the attachment would be so large that no-one would be able to download it. Equally, we cannot store each file as a separate EDA item; for technical reasons the audio and video channels must be kept together and files that have been split because they exceed the maximum file size limit must also be kept together.

The result is the following set of rules for creating EDA items for this project:

- One EDA item per year of cricket videos for the primary data. Each logical unit (i.e. the audio and video files and all their associated splits) to be zipped and attached as a single content file. The data documentation should also be attached.
- One EDA item per year of cricket videos for the secondary data. Each logical unit (in this case all the images that make up the video for one computer/camera/date) to be zipped and attached as a single content file. The data documentation should also be attached.
- One EDA item for any publication that needs supporting materials. The publication itself will be stored separately as it is automatically uploaded by Symplectic, but all the programs, documentation and results for that publication to be attached separately to one EDA item which is then linked to the publication EDA item.
- One EDA item for each set of related programs and their documentation.
- One EDA item for the results data.

- One EDA item for the other results along with the programs and associated documentation used to generate them.

Since the project is a collaboration between two different departments, each item is linked to both Computer Science and the Centre for Ecology and Conservation.

Each EDA item has generally the same metadata (language, title, etc.), but with different subtitles to reflect the different contents, and they are linked together through the *related resources* links. Additionally, each has a related *identifier* value. The shared metadata is as follows:

| | |
|---|---|
| Language | English (UK). |
| Title | CTP-000: Automatic tracking of wild field crickets in digital videos. |
| Description | This project is concerned with automatically locating and tracking wild field crickets (*Gryllus campestris*) in digital videos. The crickets inhabit a field in Spain where motion-triggered digital video cameras have been sited above the individual burrows, recording any movement that happens in the near vicinity of those burrows. The result is hundreds of thousands of hours of video segments which may or may not show cricket activities. The project is to develop programs that will automatically (i.e. without human intervention) identify, locate and track crickets in these videos. |
| Subtitle | Project header |
| Owner | Tom Tregenza, Centre for Ecology and Conservation |
| Contributor 1 | Rolando Rodríguez-Muñoz, Centre for Ecology and Conservation |
| Contributor 2 | Richard Everson, Computer Science |
| Contributor 3 | Jacqueline Christmas, Computer Science |
| Funder | NERC |
| Grant number | ??? |
| Related resource 1 | www.wildcrickets.org (the project website) |
| Related resource 2 | empslocal.ex.ac.uk/mlg/ (the CompSci research group website) |
| Keywords | crickets, tracking |
| Expiry date | The expiry date is unknown at this point. I anticipate some general policy (perhaps influenced by the requirements of particular research councils). For now I will say current date plus 20 years. |

Rather than trying to enter the same information on multiple EDA items, especially in the *description* field, it would be useful to define a project-level EDA item with all the above details, but with no attached content. The content EDA items, with individual titles and contributors, are then attached to it (bidirectionally) via the *related resources* list. This assumes that each *related resource* is able to have an associated *description* field which describes the nature of the relation (an alternative method would be to allow attached content to include URLs or URIs, enabling us to use the *file description* field to describe the nature of the link). The "CTP-000" text (Cricket Tracking Project, EDA item 000) at the beginning of the project EDA item title (which should also appear in the *identifier* field) is a method of ensuring that the EDA items are listed in a useful order when browsing the collection; the content EDA items might then be

as follows:

CTP-001a: Automatic [...] videos. 2006: videos in proprietary format.
CTP-001b: Automatic [...] videos. 2007: videos in proprietary format.
CTP-001c: Automatic [...] videos. 2008: videos in proprietary format.
CTP-002a: Automatic [...] videos. 2006: selected videos in jpeg format.
CTP-002b: Automatic [...] videos. 2007: selected videos in jpeg format.
CTP-002c: Automatic [...] videos. 2008: selected videos in jpeg format.
CTP-003a: Automatic [...] videos. Programs and results for publications.
CTP-003b: Automatic [...] videos. Tracking programs.
CTP-003c: Automatic [...] videos. Programs and results for publications.
CTP-004a: Automatic [...] videos. 2006: track data.
CTP-004b: Automatic [...] videos. 2007: track data.
CTP-004c: Automatic [...] videos. 2008: track data.

Instead of replicating the project-level *description* (which might be lengthy) to each of these lower-level EDA items, individual descriptions that describe the nature of the attached content may be specified.

A similar sort of identifier to that included at the beginning of the title may usefully be used in the *file description* field to group together attachments. This avoids the problem of requiring a user to read the details in the *file description*. Table 2 shows an example of the content that might be attached to one EDA item for this project.

One very useful new piece of EDA functionality would be to enable the user to select which order the attachments are listed in. This is commonly used on retail websites so that items can be listed in order of, for example, increasing price, decreasing price, description, etc.. In this case if the user elects to display the items in *file description* order, then the project team can design the description to produce a useful ordering, such as placing programs and their associated documentation next to each other.

# 4 Update scenarios

Listed here are a number of example scenarios which would require changes to the EDA items for this project.

**Amending contact details.** The owner of an EDA item leaves the University of Exeter. This should lead to either the contact email address being amended or a new contact found who is at the university.

**Amending incorrect items.** A journal paper has been published and the author has been contacted to point out an error in one of the programs or supporting documents. In this case attachments on an existing EDA item would need to be updated, and possibly new ones added and old ones deleted.

**Updating items.** A new version of one of the programs has been created and its documentation updated. This might happen as a result of further research, or a program may have been rewritten in a new programming language or a later version of the same language. For this a new EDA item should be created and linked to the original version via the *related*

Table 2: An example of the attachments that might be made to a single EDA item.

| *file name* | *type* | *description* |
|---|---|---|
| checkProgress.m | Matlab program | A: Program which checks how far the tracking has progressed for one year's videos and summarises it on the screen in a useful format for the user. |
| doParticles.m | Matlab program | B: Particle filter for the tracking program. |
| instr_trackYear.pdf | Adobe PDF | B: Instructions for running the tracking suite of programs. |
| instr_trackYear.tex | Latex | B: Latex |
| spec_checkProgress.pdf | Adobe PDF | A: Specification and running instructions for the checkProgress.m program. |
| spec_checkProgress.tex | Latex | A: Latex |
| spec_tracking.pdf | Adobe PDF | B: Specification for the tracking suite of programs. |
| spec_tracking.tex | Latex | B: Latex |
| trackCricket_fwdBwd_HA.m | Matlab program | B: The tracking program. |
| trackPrior.m | Matlab program | B: The tracking program with a prior on the ellipse size. |
| trackingPrograms.tar | TAR archive | B: Various other Matlab and Linux shell script programs required for the tracking. |
| videoToMat.m | Matlab program | B: Converts video from the jpeg format to a Matlab .mat file for processing. |

*resource* field. This EDA item should only *supersede* the old record if all the attachments on the old record are attached to the new item too.

**Deleting items** An EDA item has been created to provide supporting information for a paper that has been submitted to a journal (so that the reviewers can access, for example, a video). The paper is rejected due to some flaw in the reasoning, so the EDA item should be deleted or revoked.

**Expiring items** The expiry date of the EDA item has been reached. In this case the owner of the EDA item should be contacted to find out whether (i) the expiry date should be extended or (ii) the item should be deleted or revoked.