Aruspix Development Set Up

June 10, 2008

Four libraries must be downloaded in order to be able to compile Aruspix. They are the following:

- wxWidgets 2.8.7 (http://www.wxwidgets.org/)
- IM 3.1 for VC 6 (http://www.tecgraf.puc-rio.br/im/)
- Torch 2.0 (http://www.torch.ch/)
- TinyXML (http://www.grinninglizard.com/tinyxml/index.html/)

This document will outline the steps involved in compiling these libraries for Mac OS X and Windows XP operating systems.

1 Windows XP Set Up (Using Microsoft Visual C++ 6)

Note: Microsoft Visual C++ 6 has been the main Windows development tool up to date. However the project has also been compiled and run successfully using Microsoft Visual C++ 8 following the same set up guide. Some of the project properties may need to be adjusted as they are not always maintained when upgrading a project. Refer to the trouble shooting section if issues occur.

1.1 wxWidgets

To compile wxWidgets open

```
wxWidgets-2.8.7\build\msw\wx.dsw
```

using VC 6. Select Batch Build from the Build drop down menu. Now select the following three configurations from each section and build:

- Win32 DLL Debug
- Win32 Release
- Win32 Debug

1.2 IM Library

This library is available pre-compiled. Just ensure that the library you've downloaded has a file structure matches the following example:

```
\imlib
    \include : should contain all source files
    \lib : should contain all library files
```

1.3 TinyXML

To compile this library simply open

```
tinyxml\tinyxml.dsw
```

with VC 6. Select Batch Build from the Build drop down menu and the build the whole project. Now ensure that the file structure matches the following example:

```
\tinyxml : Contains the tinyxml source files.
    \Debug : Contains the debug tinyxml lib file.
    \Release : Contains the release tinyxml lib file.
```

1.4 Torch

After downloading Torch, follow the instructions provided on the Torch website for compiling the library on Windows. Be sure to name the project "Torch3".

```
http://www.torch.ch/matos/w_install.pdf
```

After compilation, make sure the Torch library folder has a file structure that matches the following example:

```
\Torch3
    \Debug : Contains the Torch3.lib (Debug) file
    \Release : Contains the Torch3.lib (Release) file
    \src : Contains the Torch source files
```

Next you need to add the following environment variable to your system:

- TORCH : Path to the Torch folder.

To do this, right click on the My Computer icon and select properties. Then select Environment Variables from the Advanced tab. Add the variable to the "User variables" section.

Now open

```
aruspix\win32\torch.dsw
```

with VC 6. Select batch build and build all the project configurations.

1.5 Aruspix

To compile Aruspix the following environment variables must added to operating system's configuration:

- IMLIB : Path of the IMlib folder
- TINYXML : Path of the TinyXML folder
- WXWIN : Path of the wxWidgets folder
- WXWIN_VERSION : wxWidgets version number (format ex. for 2.8.7: 28)

Now simply open aruspix/win32/Aruspix.dsw and you should be ready to compile Aruspix.

1.6 Troubleshooting

If linking errors occur at any stage in the setup it is likely due to the code generation settings of the projects. To fix the problem ensure that all the libraries and projects are compiled using the same code generation settings (can be found in project settings/properties -¿ C/C++ -¿ Code Generation -¿ Use run-time library:). We have been using "Multithreaded DLL" for release versions and "Multithreaded Debug DLL" for debug versions.

2 Mac OS 10 Set Up (Using Xcode)

2.1 wxWidgets Compilation

Note: wxWidgets must be compiled using the MaxOSX 10.4u sdk to enable Aruspix use with both Mac OS 10.4 and 10.5.

1. Create two directories within the wxWidgets directory: osx-static, osx-static-debug.
2. In the osx-static-debug directory run the following commands to compile for debug mode:

```
../configure --disable-shared --enable-debug --with-libjpeg=builtin --with-libpng=builtin
     --with-macosx-sdk=/Developer/SDKs/MacOSX10.4u.sdk
clean
make
```

3. In the osx-static directory run the following commands to compile for release mode:

```
../configure --disable-shared --enable-universal_binary --with-libjpeg=builtin
    --with-libpng=builtin --with-macosx-sdk=/Developer/SDKs/MacOSX10.4u.sdk
clean
make
```

2.2 Setting Aruspix Environment Variables

In the aruspix/osx directory you will find an XML file named enviroment.plist. This file is used by Xcode to set the following environment variables for linking purposes:

- ARUSPIX: Location of Aruspix project directory.
- ARUSPIX_IMLIB: Location of IM library.
- ARUSPIX_TORCH: Location of Torch library.
- ARUSPIX_WX: Location of wxWidgets library.
- ARUSPIX_WX_VERSION: wxWidgets library version number.

environment.plist prototype:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>ARUSPIX</key>
    <string>/Users/puginl/projects/aruspix</string>
    <key>ARUSPIX_IMLIB</key>
    <string>/Users/puginl/libs/imlib</string>
    <key>ARUSPIX_TORCH</key>
    <string>/Users/puginl/libs/Torch3</string>
    <key>ARUSPIX_WX</key>
    <string>/Users/puginl/libs/wx2.8.7</string>
    <key>ARUSPIX_WX_VERSION</key>
    <string>2.8</string>
</dict>
</plist>
```

1. You must modify the paths within this file to match your directory structure.
2. In the home directory create the following hidden directory: .MacOSX.
3. Now copy the modified environment.plist file into the .MacOSX directory.
4. You must now log out and log back in.

2.3 Compilation of Machine Learning Executables used by Aruspix

1. Open aruspix/osx/torch.xcodeproj with Xcode.
2. Compile the following executables in both release and debug mode: adapt, decoder, ngram.

2.4 Compilation of Aruspix

Open aruspix/osx/aruspix.xcodeproj with Xcode. Aruspix should be ready to be compiled in both Debug and Release mode.

3 Aruspix Source Code Organization

This file describe the source code organization of the project, with the directory tree. A first group of directories contains files that are common to the application, as the other directories are specific to the different workspace environment. Every workspace environment is contained into a single directory that can be excluded from the application by changing the preprocessor definitions.

3.1 Common files

/app : This directory contains the files that constitute the core of the application. The are prefixed with "ax". Most of the a related to GUI classes.

/im : This directory contains the files related to image processing. Classes and function that extend some functionalities of the IMLIB image processing library, such as advanced binarization methods, are to be find here.

/ml : This directory contains the files that relate to machine learning.

/mus : This directory contains the files that implement the music editor and handle the music files.

3.2 Workspace environment files

Workspace environments can be enabled or disabled at compilation by changing preprocessor definitions. The files generated with wxDesigner cannot be disabled by changing the preprocessor definitions and have to be removed by hand (this has to be fixed).

/empty : This directory contains a workspace template (May be out of date).

/superimposition : This directory contains the files that implement the superimposition of digital music on to the original images.

/recognition : This directory contains the files that deal with the music recognition.

/comparison :