Getting Started with i2k Align and i2k Retina

Support Staff (support@dualalign.com)
DualAlign LLC
Sept 30, 2018

Overview

Welcome to users of the i2k Align and i2k Retina software. This document provides instructions on how to run the software and how to activate it once the user has received an activation code. The document assumes the software has been downloaded and installed. All functionality of i2k Align is included in i2k Retina, but i2k Retina adds additional functionality for aligning and montaging retinal mages. The description that follows uses the name i2k Align generically, and references i2k Retina as needed.

The software may be run in three different ways:

- Using the graphical-user-interface (GUI) version of the software.
- Using the command-line interface (included with the API version).
- By running your own customized software that calls the functions of the library or API. The API for i2k Align and i2kRetina products is available upon request.

On-line documentation for the i2k Align GUI is available through the help menus. The current document focuses on the command-line tool and the API interfaces.

Software Location

On Microsoft Windows, `i2k Align (or i2k Retina)` is by default installed under `"c:\ Program Files"`. On a Mac, the executable programs for i2k Align (or i2k Retina) are put in /Applications. Developers who download the API version of the software, will receive header files such as da_error_codes.h and da_i2k_align.h, the required dlls (library code) and other supporting files and documents in a single archive (.zip , .tar.bz2, .dmg or similar depending on your platform).

Running the Command-Line Executable

The command-line executable honors switches (flags) for exercising many features of the API library, and for activating the software once the user has purchased an activation code. A list of the options may be obtained by typing the name of the command line tool without any switches (flags) :

```
da_i2k_retina_exec_v2
```

The command line executable and the library do extensive error checking on the form of the inputs, with as much of the error checking being performed by the library as possible. Here are some examples using images provided with the installation. These instructions assume the current working directory contains a .list file and images referenced in the .list file and also assumes that da_i2k_align_exec_v2 is in the PATH (environment variable). In each of the examples where da_i2k_align_exec_v2 is used, da_i2k_retina_exec_v2 can be used in its place.

1. Basic montaging

```
da_i2k_align_exec_v2 -list hawaii\hawaii.list
```

Builds a montage of the images whose names are listed in a text file hawaii.list. The output montage will be stored in the current working directory.

2. Specifying the output location

```
da_i2k_align_exec_v2 -list hawaii\hawaii.list  -dir results
```

With the additional switch -dir the montage will be put in subdirectory results (created as necessary) in the hawaii directory.

3. Basic alignment:

```
da_i2k_align_exec_v2 -list hawaii\hawaii.list -output_type 1 -layout 2 -dir results
```

Output type of 1 aligns the images, outputting the mapped images to separate files in the same folder as hawaii.list. The -layout 2 parameter indicates that cylindrical layout is used. Remember, the layout must be specified (a) for alignment or (b) for montaging of non-retinal images.

4. Montaging thermal images:

```
da_i2k_align_exec_v2 -list thermal\thermal.list -output_type 0 -image_type 2 -layout
5
```

(Note that -output_type 0 was not strictly necessary since its the default) The image type here is “nonphotographic” which works for thermal and multimodal image sets. Layout option 5 is for affine transformations, which works well for most thermal images because of their low resolution.

5. Aligning multimodal brain MRI's

```
da_i2k_align_exec_v2 -list brain\brain.list -dir results -output_type 1 -image_type
2 -layout 6 -save_xforms
```

(this should all be on one line). Here the homography layout is used, but planar and affine would have worked just as well. DualAlign's convex mask calculation algorithm is applied to restrict attention to the primary regions of the images. The estimated transformations and correspondences are output to two files in the output directory results, which is created if it does not already exist.

6. Aligning and cropping images for HDR applications

```
da_i2k_align_exec_v2 -list hdr\hdr.list -output_type 1 -layout 1 -align_crop 2 -dir
results
```

Aligns the images and crops them to their overlap.

7. Aligning retinal red-free images and fluorescein angiograms

```
da_i2k_retina_exec_v2 -list angio\angio.list -output_type 1 -image_type 0 -
align_crop 3 -target IMG0038.jpg
```

Here, we have moved down into the angio folder prior to running the exec, and therefore the specification of the file containing the list of images is simpler. The image type is retinal (the 0) and the software produces a set of aligned images, cropped to fit inside the mapped version of IMG_0038.jpg.

8. Creating a retinal montage

```
da_i2k_retina_exec_v2 -list retina_montage\F.list -image_type 0
```

Running i2k Viewer

Currently, i2k Viewer can only take image file(s) supplied at the command line. I2k Viewer comes with the GUI software packages.

Running i2k Viewer is simple. It has only two modes, one for viewing a single image (a montage) and one for viewing a sequence of two or more images. The difference between these is simply determined by the number of images provided on the i2kViewer command line. When only one image is provided, the viewer will be in Montage Mode, as in the following example

```
i2kviewer retina_montage\F.jpg
```

When two or more images are provide, the viewer will be in Aligned Images Mode, as in the following example

```
cd retina_montage

i2kviewer F1-2.jpg F3.jpg F4.jpg F5.jpg F6.jpg F7.jpg F8.jpg F9.jpg F10.jpg
```

Running Incremental Montage

Incremental Montage command line tool da_i2k_inc_montage.exec supports most of the basic parameters needed to run da_i2k_align_exec_v2 and/or da_i2k_align_retina_v2. So you may already be familiar with many of them such as -list -layout -license -activate -deactive etc …

The most important new parameter would be the -delay parameter which mimics any delay in the input acquisition process.

The console output from da_i2k_inc_montage should show the incremental processing time for each new image.

Running / Creating a Montage Sequence Alignment

A montage sequence is a tool to visualize changes in the eye over longer timeframesthan a single study session. For example, given the old standard seven fields of diabetic retinopathy imaging, taken over time intervals several months apart, then you can see the change associated with progression of disease across the entire retina.

One thought might be that you would simply create a montage for each image set and then run alignment to align the montages. There are two major problems with this. First, masking out the background becomes that much harder because of the irregular shape. Second, it is harder to get an accurate alignment across the entire retina because of accumulation of minor errors and inconsistencies. The results are both more accurate and more visually pleasing if you solve for the entire set of inter-image transformations at once and then build the montages. There are conceptually straightforward work-arounds for the first problem, but not for the second.

A very basic invocation and performance of montage sequence can be done like this …

```
da_i2k_retina_exec_v2 -license "..\License" -capture_groups "cgroups.txt" -image_type 0 -
output_type 2 -montage_crop 0 -save_alpha -save_layers
```

Above the file cgroups.txt may contain two lines … for example Feb.txt and March.txt where Feb.txt lists images in a February image study and March.txt lists images in a March image study taken 1 month later.

Best results will be obtained when Feb.txt and March.txt files do not have any filename overlap. Filename overlap might happen if someone was trying to mockup a quick test of the above commandline and copies some contents from Feb.txt into March.txt. This is not an acceptable way of testing. It will also be preferred that each study is composed of a single component after images are aligned.

Licensing and Activation

When the software is downloaded and started, it will be in trial mode, which allows the user to run it for up to 14 days and 75 runs. During this time, all functionality is available, but the output images and montages have watermarks on them. A user may purchase an activation code to promote the software from trial to fully function without watermarks. The 16 digit activation code for i2k Align may be purchased at ...

```
https://www.dualalign.com/image-registration/software-buy-now.php
```

while the code for i2k Retina may be purchased at

```
https://www.dualalign.com/retinal/image-registration-montage-software-buy-now.php
```

Each purchased activation code is typically for a single workstation (for more permissive licensing agreements and more permissive activation codes … contact DualAlign).

Once a user has an activation code, it is easy to activate through the GUI software. It can also be done through the command-line executable, as follows:

```
da_i2k_align_exec_v2 -activate xxxx-xxxx-xxxx-xxxx
```

where the -activate switch is followed by the actual code (replace x characters as shown above).

In rare instances, in particular for users behind a particularly stringent fireware, activation will need to be done through a multi-step semi-manual process:

1. Run the command-line exec

```
da_i2k_align_exec_v2 -machine_id
```

2. Email the machine id shown in the output, together with the 16-digit activation code output, (include an appropriate subject line) to

support@dualalign.com. Alternatively … if you are a repeat customer using many activation codes, then you may sign up to use our user webportal to avoid delays.

3. When the return email is received, save the attached license file onto the hard disk. Any name is fine, but to for the example below, assume it has been saved to license.txt.

4. Run the command-line exec (from the folder where license.txt is stored) as follows including dashes as shown:

```
da_i2k_align_exec_v2 -activate xxxx-xxxx-xxxx-xxxx -manual_file license.txt
```

The Library

Developers who want to write their own programs on top of i2k Align and i2k Retina libraries will have to request the one of the DualAlign developer kits. For i2k Retina the URL is

```
http://www.dualalign.com/api_request_form.php?product=i2kretina
```

while for i2k Align the URL is

```
http://www.dualalign.com/api_request_form.php?product=i2kalign
```

Developers will be able to download the developer kit which contains headers, dynamic libraries, and documentation which deliver a straight-forward, C application interface for image alignment and montage creation. Developers will need to pay particular attention to the use the activation functions, since each machine the software is installed on will require its own separate activation code. Fortunately, activation previously performed with the GUI software will also activate the software for the command-line and API versions.

The software runs on the Mac (10.6 and later) and on Windows (Vista, 7, 8, and 10), and versions are also available on Linux machines. The software on the Mac was compiled with XCode. Users compiling and linking against the library must be sure to locate the header files in their include path and link the library correctly. There are different environment variables used on a Mac (vs Windows) to help locate dynamically loaded libraries. Be familiar with the loading rules on your platform.

The software for Windows is compiled with Microsoft Visual Studio. But users may work with other other compilers as needed. Taking the example of including i2k Align into a Visual Studio C++ project, the user must make the following three changes to the project configuration page:

1. Add i2k Align's include directory to Additional Include Directories under General of C/C++ (see (a) in the following figure).

2. Add i2k Align's Unicode-libs (or non-Unicode-libs) the installation path to Additional Library Directories under General of Linker (see (b) in the following figure).

3. Add the library name da_i2k_align_v2.lib to Additional Dependencies under Input of Linker (see (b) in the following figure).

After establishing these settings, the user's code may call any functions in the library. The same idea applies to i**2k Retina**, although the library named **da_i2k_retina_v2.lib** must be used instead. Once project compilation is finished, make sure the DualAlign dll files in say Unicode-libs folder are either in the current working directory or in one of the directories specified in the PATH environment variable.

Unicode / Internationalization

On Mac and Linux, Unicode is supported by the i2k Align and i2k Retina in a straightforward manner as these platforms uses UTF-8 Unicode encoding by default. To access a folder or a file that contains Unicode characters, simply provide the name encoded with UTF-8, which is the default encoding on most systems.

However, Windows takes a very different approach — it provides two sets of system API. One set, based on 8-bit character representation (known as char type), is for backward compatability, whereas the other, based on 16-bit character (known as wchar_t type), is for taking full advantage of Unicode functionality on Windows. To access a folder or a file that contains Unicode characters, the best way is to encode the name with UTF-16

(the default Unicode encoding on Windows) and store the string in wchar_t character type. Users may gain an understanding of Unicode support on Windows at

```
https://msdn.microsoft.com/en-us/library/2dax2h36(v=vs.140).aspx
```

Taking a similar approach, we provide unicode-libs and non-unicode-libs versions of the i2k libraries for users to pick from. Each version contains key files, da_i2k_retina_v2.dll and the corresponding import library da_i2k_retina_v2.lib *(or da_i2k_align_v2.dll and da_i2k_align_v2.lib for i2k Align)*. The version based on 16-bit character (wchar_t type) is located in the installation directory. It is used by our GUI to provide support for Unicode file names, and is the recommended version unless backward compatibility is a concern. The other version, based on 8-bit character (char type), is located in the non-Unicode-libs folder under the installation directory. You may use this version if you do not intend to deal with Unicode characters. Both versions behave the same other than the Unicode support.

To convert strings between the 8-bit and the 16-bit character type, we recommend to use the following two functions (available after including windows.h). Please note that the result of conversion from 16-bit to 8-bit depends on the current locale settings and hence Unicode characters may not be preserved.

- WideCharToMultiByte: conversion from 16-bit to 8-bit. The documentation is available at

  `http://msdn.microsoft.com/en-us/library/dd374130%28VS.85%29.aspx`

- MultiByteToWideChar: conversion from 8-bit to 16-bit. The documentation is available at

  `http://msdn.microsoft.com/en-us/library/dd319072%28VS.85%29.aspx`

Compilation with 8-bit version

To compile and link with the 8-bit-character version of the i2k Align library, you must follow the following steps:

1. Make sure you define “NO_DA_USE_WIN_UNICODE” whenever the header file da_i2k_align.h is included. To do so, you may add it to Preprocessor Definitions in the project setting, or use “#define NO_DA_USE_WIN_UNICODE” before “#include da_i2k_align.h”.

2. Link to the library da_i2k_align.lib that is located under non-Unicode-libs.

3. To run the compiled binary code, make sure da_i2k_align.dll located under non-Unicode-libs is either in the current working directory or in one of the directories specified in the PATH environment variable.

I2k Align and Retina Precision and Accuracy

In order to get repeatable results with the i2k products, please take care to provide a consistent operating environment. It will be especially important to provide the image files to i2k Retina / Align in same order on command line or through the API. If you change the order of input images, then there may be very minor differences in the output montage. Naturally if you change any “options” such as montage_blend type (FULL, FEATHERED, NONE, etc.) your output will also vary.

Running Multiple Sessions

The DualAlign i2k align API is multi-session aware, which very means two sessions doing basic register, align and or montage operations can be requested and executed simultaneously (parallel processing) in a shared process space. In a simple case, one session might refer to a single task of aligning images and creating a montage for a one pair of eyes from one individual. Simultaneously, you could create a separate session to align and montage images from eyes from a second individual. If you have sufficient CPU cores and sufficient processing power on your machine … this may help throughput. But the details of exactly how this is done is up to your developers.

API function da_i2k_start_reg_session( ) fills in a reference parameter to a registration identifier (aka session identifier). Subsequently, this registration identifier will be used to manage all session related operations going forward.

If your i2k license has a simultaneous sessions limit of 4, and you then create for example a process with 5 worker threads each running an i2k align registration / montage … DualAlign software will report that you have exceeded the allowed number of simultaneous sessions, and will cease to function properly. In this case you will need to negotiate / purchase a license with more sessions allowed.

When establishing a software license agreement with DualAlign and subsequently when purchasing activation codes, the number of sessions should be indicated in each purchase order.