IMAGE STRIPS COMPUTED FROM IODP WHOLE CORE PHOTOS

Chris Jenkins

INSTAAR, Univ Colorado at Boulder
CO 80309-0450 USA
8 Dec 2008

Introduction

Image strips for each section of drill core recovered by IODP can be used in applications such as Corewall (Rao et al., 2005) which display and integrate down-hole sensor, descriptive and image data. However, the image-strips often need to be extracted from whole-core photographs. This is true for IODP core recoveries and also for many other piston-core data sets (e.g., Cain, W., III, 2003) and the DSDP recoveries (e.g., IODP-USIO, 2006, 2008). Scanned core imagery however, is generated directly in image-strip format (e.g., Expedition 309/312 Scientists, 2006).

This project created software to extract image strips from the IODP data collection, covering over 36,000 whole-core photos. The software, in Python v.2.5, has been tested on a range of generations of photographs, section and void space arrangements, and lithology types.

The project was initiated by Drs David Divins and Sean Higgins of Ocean Leadership (while formerly at JOI, Joint Oceanographic Institutions) under a program to improve the useability of Ocean Drilling Program legacy data. It is part of contract 54505 (UCOLBO).

Data sources

Whole-core photographs

Whole core photographs of the archive core-halves were taken aboard ship (JOIDES Resolution) during core description and testing stages.

Some markings on background frame were relatively unchanged throughout the program: the positions of text “LEG”, “SITE”, “CORE”, “HOLE”, the physical holes for the tubes, and graticule marks. This was an important aid to creating the image strips.

However there were difficulties. The number of sections was variable and images were cropped at right margin to suit. Some photos (or parts of photos) were clockwise oblique by 0.5 degrees, some were wider, many were cropped at the base, and field distortion was also an issue. Other practical difficulties were: flow-out of materials at the section ends (past 150cm mark), and some recovery had internal - pull-aparts created during core handling. A number of photographs had had text stamped on later. The zero-metres line was removed in others.

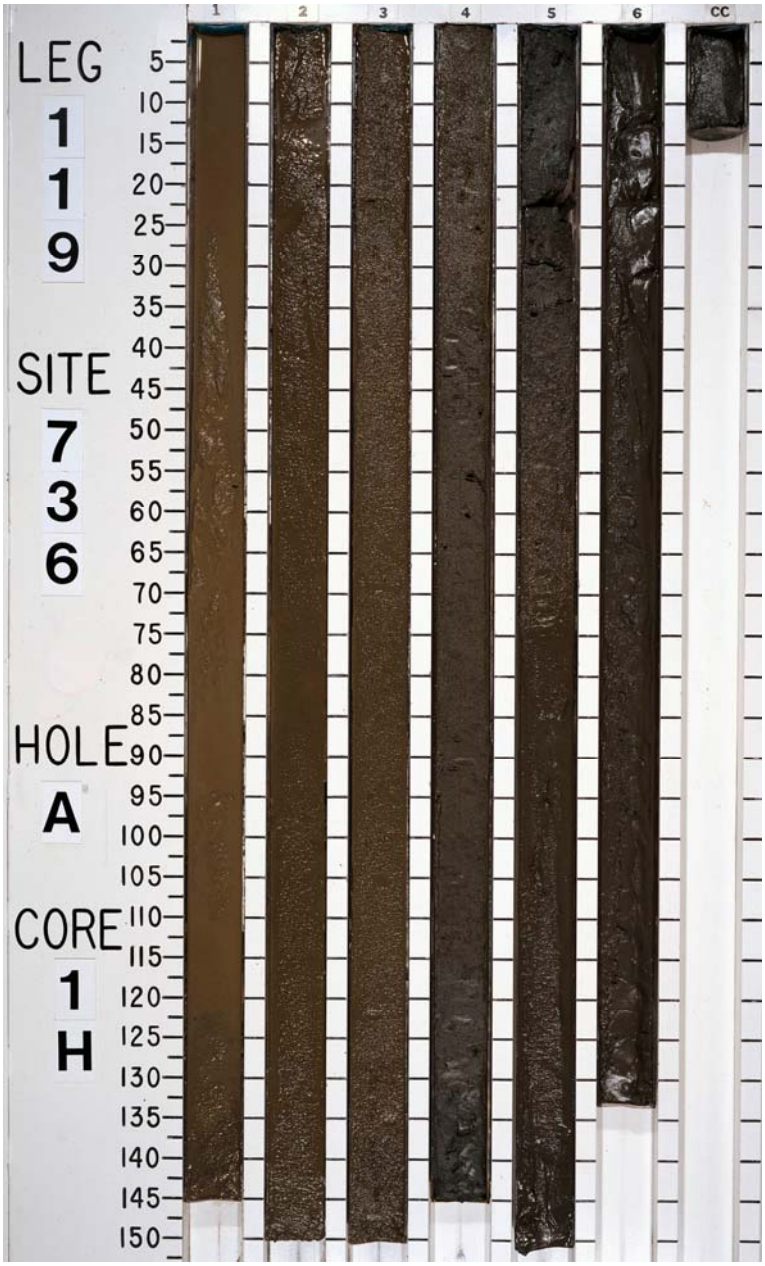

Fig. 1. An IODP whole-core photograph.

Methods

TIFF or PDF digital images of the whole-core photographs were obtained from NGDC (NGDC 2008) and the JANUS database (IODP-USIO, 2008). All other formats were transferred to TIF using an Imagemagick script. The native images had varying image characteristics:

a. The photos obtained from JANUS as TIFFS or via PDF files were usually 2700 pixels (long), nominally 300dpi and 24bit colour. The widths varied by the count of non-vacant sections. File sizes were typically 9MB.

b. The images from NGDC were all TIFFs. They varied in the range 5060-5140 pixels long by minor cropping at the base. Due to cropping of vacant section tubes, widths varied from 1024 pixels (when a CC only, 14MB) to 3404 pixels (when 7 sections plus CC, 50MB). They had 24 bit colour, and nominal 1200 dpi.

Software description

The software "imgSTRIPS_*.py" (currently version 3) performed almost all steps to create the image strips. The program was used in 2 stages: A. preparation of templates; B. creation and finishing of the image strips.

A. Preparation

i. From real imagery (photo 0736A1H.tif), the function *getFIXpnts* prepared small images of fixed features in the frame which later could be used to measure each image's scale, rotation and distortion – as well as dpi. The features were "LEG", "50", the "E" in "CORE", and "150".

ii. The function *profileLINR* prepared a measured grayscale profile of the bottom 10cm of each tube in the photo rack in terms of grayscale average and RMS per x-pixel. Because of shadow and usage effects, a profile was made for each tube of the rack. The profile was based on averages of travelling 8-row zones, similar to the later matching process (B-viii, below). The profile is used to detect the bottom of recovered materials in each tube for each image strip.

B. Creation and finishing

i. Metadata for the whole core photo was constructed (in function *main*) from the imagefile name and by reference to the JANUS Hole, Core, Section data listings "site_hole_summary.txt" and "core_section_summary.txt" (downloaded 18 Nov 2008). Typical image file names held the essential site/hole/core/coretype data. Where the site code was <1000, a leading zero was added to make names predictable.

ii. The function *makeCML* makes an XML-style Corewall Markup Language (CML, Jenkins & Chen 2007) file for each core.

iii. The colour image was copied to a grey image of 2700 pixels length, for the working.

iv. The zones of the photos to seek the fix points in can be defined quite closely because some parts of the frame layout were constant through IODP. Those

'lookin' areas are defined by fractional locations across and down the photo (relative to photo pixel length).

v. The 'seek' images were cross-correlated (in function *xCORRELATE*) with the 'lookin' images using Scipy correlate2d function, with wrap at boundaries. Some difficulty was had obtaining a sharp location from this process, especially at the lower right of images where no distinctive text was available.

Figure 2. xCORRELATE results, to obtain fix coordinates on text featues in the photo frame. The method worked with the LEG, 50, COR-E, 150 features.

vi. The absolute coordinates of the fixes were calculated as follows: $X_{fix}=X0_{lookin}+W_{lookin}/2+X_{detect}$, $Y_{fix}=Y0_{lookin}+L_{lookin}/2+Y_{detect}$, where $(X,Y)_{fix}$ is the fix absolute coordinate, $(X0,Y0)_{lookin}$ is the size of the 'lookin' subimage, and $(X,Y)_{detect}$ is the coordinate of the *xCORRELATE* maximum (white spot in Figs 2 for LEG, CC, 50, 150.

vii. Zero level was also determined by looking for first definite horizontal dark line at the top of section 1, and at the far right of the photo. These points were also added to the list fixes, along with a bottom measure at 1.56m level.

Fix\Fix	0 "LEG"	1 "50"	2 "COR-E"	3 "150"	4 UL Zero	5 UR Zero	6 LL 1.56m
0 "LEG"	-	43	44	44	25	25	43
1 "50"	43	-	44	45	41	41	43
2 "COR-E"	44	44	-	46	43	43	41
3 "150"	44	45	46	-	43	43	12
4 UL Zero	25	41	43	43	-	-	42
5 UR Zero	25	41	43	43	-	-	42
6 LL 1.56m	43	43	41	12	42	42	-

Table 1. Example DPI calculation from cross-correlated fixes. These are the DPI for the processing-stage reduced greyscale images only.

viii. The fixes' y- and pixel-coordinates were then used to calculate median and average vertical (y) dpi from a matrix (Table 1), where values are dpi.

ix. The number of strips was predicted from a calculation involving the width of the left-side approx 384-pixel wide lead-in area of the photo, plus allowance for about 180 pixels per section.

x. A more exact method of determining the section edges was required, since photo fields could be shifted. A photo is scanned down each y-column to determine 3 characteristics; average grey level, mean absolute deviation of grey level, and number of dark pixels. The latter is a good identification of the vertical graticule zones between the sections. The results on these characteristics were differenced between adjacent y-columns, in effect making a vertical line detection method (Fig. 3).

The 3 characteristics were combined into one summed measure (called pk3VAL) which was smoothed over 3 adjacent columns. The peaks of pk3VAL mark the edges of the sections no matter the colour or texture of the recovered materials, and when a section tube is almost empty. Complexities in the peaks were resolved using a distance- and height-weighting around the expected x-pixel location. From the unified peaks, the strip x-coordinates were derived.

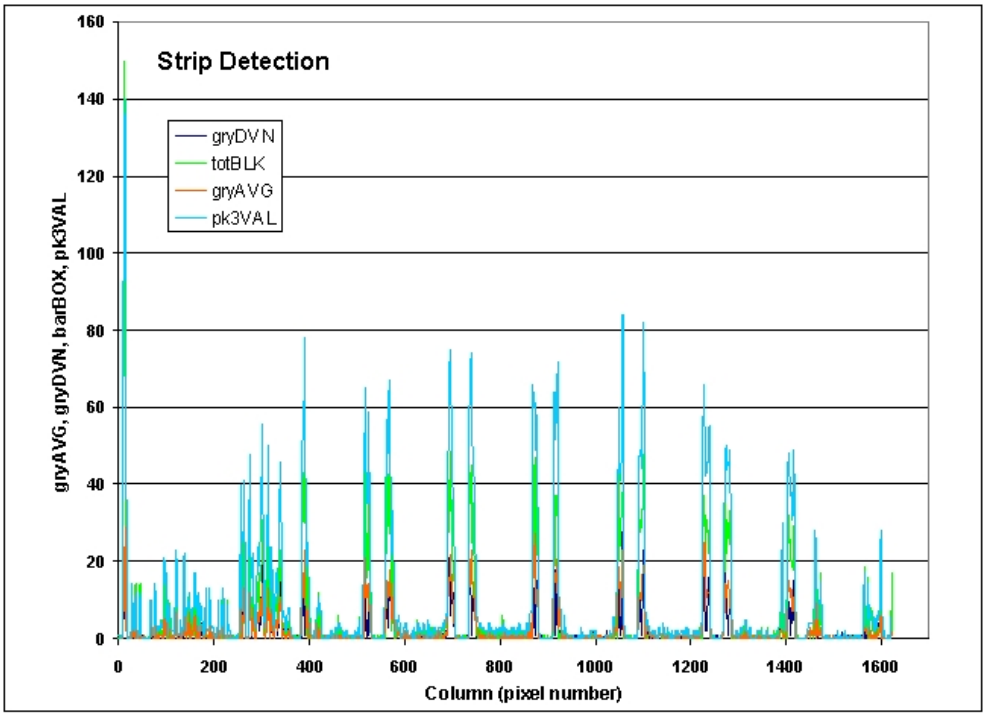

Figure 3. Graphic showing method of more exact detection of section edges.

xi. The initial cut of image strips is based on their top Y-coordinate, four (UL clockwise) X-coordinates, and bottom Y-coordinate (function *maskCUT*).

xii. Each strip is scanned bottom-up (in function *deTUBE*) to find the first level where the greyscale across-strip profile deviates in RMS from a previously measured profile. The measured profile is distinctive for each tube because of shadowing, grime and wear. The function *trimSTRIP* then crops the image strip down to that level. The scanning is done using a travelling 8-row zone. Shadow from the base of the d-tube or recovered material may still cause problems with this procedure.

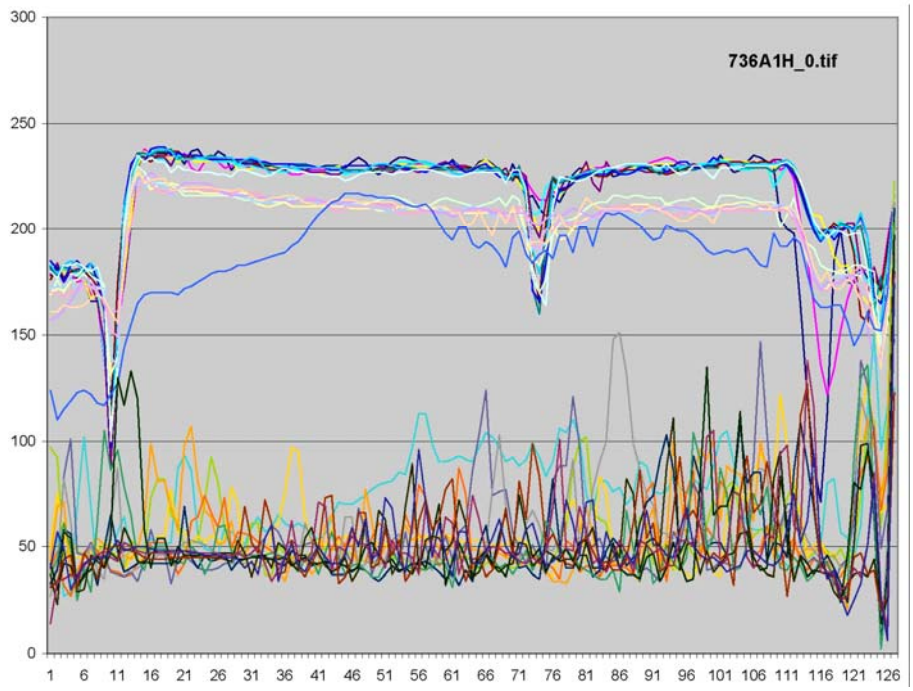

Figure 4. Greyscale profiles across tube 0 (section 1).

The upper family are profiles in full light, those slightly lower are in shadow, the single deviating curve is at the transition from liner to sediment, and the lower highly variable family of profiles is in the sediment recovery. The central dip was caused by abrasion and dirt build-up in the bottom of the liner. The bounding dips are due to tube abrasion and shadowing.

The function deTUBE cuts when the whole-profile median RMS deviation from the in-light or shadow profiles is greater than one tenth of full scale (i.e., > 25.5)

xiii. A table (*_stripCAT.csv*) with metadata on strip size, resolution, origin photo, error reports, processing date/time is produced for each hole.

xiv. Because the image memory buffer would be exceeded for the colour TIFS in the main program (on a PC), stand-alone Python scripts (*_cropCOLOUR.py*) to do the cutting of the colour strips were generated for each hole. Those scripts can be run manually or in batch after the main program has finished. The final strips generated in this way have the same RGB as the original photos, and DPI which for the NGDC photos is about 83 dpi.

xv. Corelyzer project XML files (**.cml*) are produced first of all with local (INSTAAR machine) URLs. A small program *_local2webURNs_*.py* (now version 2) can be run to generate additional CML files with web URLs. Those new CML are named **_wb.cml*.

Results

Fixes on images and DPI calculation

The *xCORRELATE* results were satisfactory for a majority of cases (Fig. 2). The DPI values (e.g., Table. 1) were satisfactory, but not always equal between all fixes, possibly due to photographic field (lense, angle) distortions. As expected dpi's determined from close fix points were often inaccurate. The median of dpi estimates gave a more accurate (and higher) summarizing value than the average.

Cutting and trimming

The trimming process was straightforward. It was decided not to rotate the photos which were slightly (~1º) skewed before cutting the strips, because this would have degraded the image quality. Most strips show only the core recovery, the cut margins of the d-tubes, the liner cap area, and some shadow.

Figure 5. Image strip and trimmed image strip.

Conclusions

Software which can be used to cut image strips from the IODP whole-core photo collection now exists and had been tested and validated against a variety of photo image layouts and lithologies.

The software now needs to be used on the total collection, which is estimated to be about 1TB in size, and is archived at NGDC, NOAA in Boulder.

References

Cain, W., III, 2003. *Classification of surficial sediments: North-Central and Eastern Gulf of Mexico*. TAMU MSc Thesis, [Supervisor] Bryant, William R., August 2003.

Expedition 309/312 Scientists, 2006. Digital imaging in: Methods. In Teagle, D.A.H., Alt, J.C., Umino, S., Miyashita, S., Banerjee, N.R., Wilson, D.S., and the Expedition 309/312 Scientists. Proc. IODP, 309/312: Washington, DC (Integrated Ocean Drilling Program Management International, Inc.), 1–70. [doi:10.2204/iodp.proc.309312.102.2006]

IODP-USIO, 2006. *Core Photographs & Digital Images*. [URL: "http://iodp.tamu.edu/database/coreimages.html", Last modified 08-Dec-2006]

IODP-USIO, 2008. *Core Photos Data Request Form*. [URL: "http://iodp.tamu.edu/janusweb/imaging/photo.shtml", Last modified 05-Mar-2008]

Jenkins, C.J. and Chen J.Y.-C., 2007. *Specification of Corelyzer Files*. INSTAAR, University of Colorado. [URL: "http://instaar.colorado.edu/~jenkinsc/dbseabed/corewall/CmlXml_Syntax_1.pdf"]

NGDC, 2008. *Core Data and Photographs from the Ocean Drilling Program (ODP) JANUS Database: NGDC Metadata Record G03298*. NOAA National Geophysical Data Center (NGDC), Boulder CO USA. [URL: "http://www.ngdc.noaa.gov/geosamples/metadata.jsp?g=G03298", Revised Nov 27 2008]

Python Software Foundation 2008. *Python Programming Language -- Official Website*. [URL: "http://www.python.org/"]

Rao, A., Rack, F., Kamp, B., Fils, D., Ito, E., Morin, P., Higgins, S., Leigh, J., Johnson, A., Renambot, L., 2005. CoreWall: A Scalable Interactive Tool for Visual Core Description, Data Visualization, and Stratigraphic Correlation. *Eos Trans. AGU*,86(52), Fall Meet. Suppl., San Francisco, CA, 12/05/2005 - 12/05/2005

Appendix A
Folder structure

Program	C:\IODPDSDP\CoreImages\work\
Report	C:\IODPDSDP\CoreImages\work\Report\
Program Diagnostics	C:\IODPDSDP\CoreImages\work\
Liner images & data for initializations	C:\IODPDSDP\CoreImages\work_Liner\
Data catalogs from Janus	C:\IODPDSDP\CoreImages\work_Janus\
Fix feature images (with versions) for initializations	C:\IODPDSDP\CoreImages\work_Fixes\
Materials at photo level per Hole	e.g.: C:\IODPDSDP\CoreImages\work\1207A\
Files used in the processing but not important as products	e.g.: C:\IODPDSDP\CoreImages\work\1207A_Work\
Strip images per Core	e.g.: C:\IODPDSDP\CoreImages\work\1207A\1207A_Strips\
Obsolete and backup versions and development materials	C:\IODPDSDP\CoreImages\work_Old\
Zipped deliveries	C:\IODPDSDP\CoreImages\work_Zips\

Appendix B
Run Instruction Notes

1. Note: The code is not yet checked for cross-platform and relative address robustness. For example folder delimiters may be Windows.
2. Note: It is recommended that one hole at a time be processed, because duration is currently 4 mins per photo, or 180 photos overnight.
3. Note: Python may not release even closed output files until the program is closed.
4. Note: Complete consistency with the Janus database hole namings is not assured at this stage.
5. Step 1. Edit the file "_holeLIST.txt" to include the holes which you wish to have processed in this run. Lines starting with "#" or "%" will be ignored in reading. All holes should have a 4-number plus 1-letter name like "0736A".
6. Step 2. Make sure the required colour photos are in the correct folders. We are giving preference to the NGDC 15-50MB hi-res tiff images. They should be in subfolder 0000A_Photos\ from the program location.

   The python module *glob* will locate all the photos of ".tif" image format where the name ends in one of "ABCDEFGHRVWXZ", for the subfolders "_Photos" of the folders for the holes listed in "_holeLIST.txt".
7. Step 3: Start the program "imgSTRIPS_*.py" (now version 6) using Python 2.5 or later (Python Software Foundation 2008). The program will process all the photos found by *glob*.
8. Step 4: After the program is finished with a hole, run the colour-cropping script "_cropCOLOUR.py" (double click is enough) in the Hole folder (e.g., in "C:\IODPDSDP\CoreImages\work\1050C").
9. Step 5: Review the strip images held in each folder "0000ANX_Strips" where "0000A" is the hole and "NX" is the core.
10. Step 6: Send the results to the strip image server.