A Ground-Truthing Engine for Proofsetting, Publishing, Re-Purposing and Quality Assurance

Steven J. Simske, Margaret Sturgill
Imaging Systems Laboratory
HP Laboratories Palo Alto
HPL-2003-234
November 18th , 2003*

layout, region management, print-on-demand, templates

We present design strategies, implementation preferences and throughput results obtained in deploying a UI-based ground truthing engine as the last step in the quality assurance (QA) for the conversion of a large out-of-print book collection into digital form. A series of automated QA steps were first performed on the document. Five distinct zoning analysis options were deployed and the PDF output thence generated was used to regenerate TIFF files for comparison to the originals. Regenerated TIFFs failing automated QA or a separate visual QA were tagged for ground truthing. Less than 3% of the pages in a $1.2x10^6$ -page corpus required ground truthing, resulting in a throughput rate of “fully-proofed” pages of $2x10^5$ pages/manweek. Among the design advantages crucial for this throughput rate was the use of the identical zoning engine for the original production workflow and for the ground truthing engine.

A Ground-Truthing Engine for Proofsetting, Publishing, Re-Purposing and Quality Assurance

Steven J. Simske
Margaret Sturgill
Imaging Systems Laboratory
HP Laboratories
October 10, 2003

ABSTRACT

We present design strategies, implementation preferences and throughput results obtained in deploying a UI-based ground truthing engine as the last step in the quality assurance (QA) for the conversion of a large out-of-print book collection into digital form. A series of automated QA steps were first performed on the document. Five distinct zoning analysis options were deployed and the PDF output thence generated was used to regenerate TIFF files for comparison to the originals. Regenerated TIFFs failing automated QA or a separate visual QA were tagged for ground truthing. Less than 3% of the pages in a 1.2x10^6-page corpus required ground truthing, resulting in a throughput rate of "fully-proofed" pages of 2x10^5 pages/man-week. Among the design advantages crucial for this throughput rate was the use of the identical zoning engine for the original production workflow and for the ground truthing engine.

Keywords: Layout, Region Management, Print-on-Demand, Templates

1 Introduction

This report is a follow-up of the Hewlett-Packard Technical Report HPL-2002-272 [6]. Recently, the authors and colleagues completed work on a large print-on-demand system for MIT Press [1]. Its purpose was to convert raster scanned documents (TIFF) into printable and web-readable files (PDF). As a consequence, many out-of-print books in the MIT Press collection were made available to users, both electronically and via print-on-demand (POD). More than 1.2x10^6 pages were automatically converted, with all appropriate region types (text, drawing, photograph) appropriately classified for their boundaries (segmentation), type (classification as text, drawing or photograph) and bit-depth (1-bit for text, rules, line drawings, bounding rectangles, etc.; 8-bit for gray-scale graphics and photos; and 24-bit for color graphics and photos). This process is referred to as "zoning analysis", and it has a key role in quality assurance (QA).

The "ground truthing engine", or GTE, is a user-interface (UI) controlled application for proofing, template definition, ground truth definition, and repurposing that was developed using Java and, separately, Visual Studio.NET. The application was designed and deployed to complete the quality assurance requirements of an otherwise fully-automated POD process for the MIT Press Classics book collection [1]. Any pages failing automated QA ("AutoQA") were proofed for printing, publishing and later metadata tagging. Slight expansion of the original design provided templating, ground truthing and repurposing capabilities by allowing the specification of borders and region bit depth and classification. The UI implementation comprised providing a view of the image along with various (hot key, mouse, menu) built-in commands (and various automated zoning engines) to facilitate zoning and proofing of the image. Its output is in XML in accordance with a schema/DTD written to provide sufficient metadata for publishing, repurposing and template definition. This user-defined data set also comprises "ground truth" of the viewed image and is thus theoretically an 100% accurate representation of the image layout (and a full solution to any residual QA errors), assuming no user errors. The XML file so specified is then used as an input file (together with the original, high-resolution, scanned TIFF files) to guide the overall analysis engine in its generation of a PDF output file.

Forms, templates, specialized scanning adapters, and specific re-purposing applications (i.e. for POD, web page generation, content delivery to specialized display devices, etc.) require 100% accuracy for layout definition. A 100% accurate description of a document/image/etc. layout is termed "**ground truth**". Ground truth specifies exactly where all of the regions are on a (possibly composite) image. It also describes what they are (e.g. "text", "photo", "drawing"), termed "classification" or "region typing". Furthermore, ground truth may describe how the region is to be treated (e.g. as binary "BW", or black and white; "GRAY" or 8-bit typically, and "COLOR" or 24-bit typically), termed "bit depth". Ground truth can be used to ensure the proper rendering of a complex scanned document (PDF, RTF, DOC, etc.). It can also be used to define web page specifications (HTML, XML, etc.). In defining a template for all other documents in a set, ground truth can be used for forms and other like-document processing (e.g. automatic document feeder slide templates, predefined layouts, etc.).

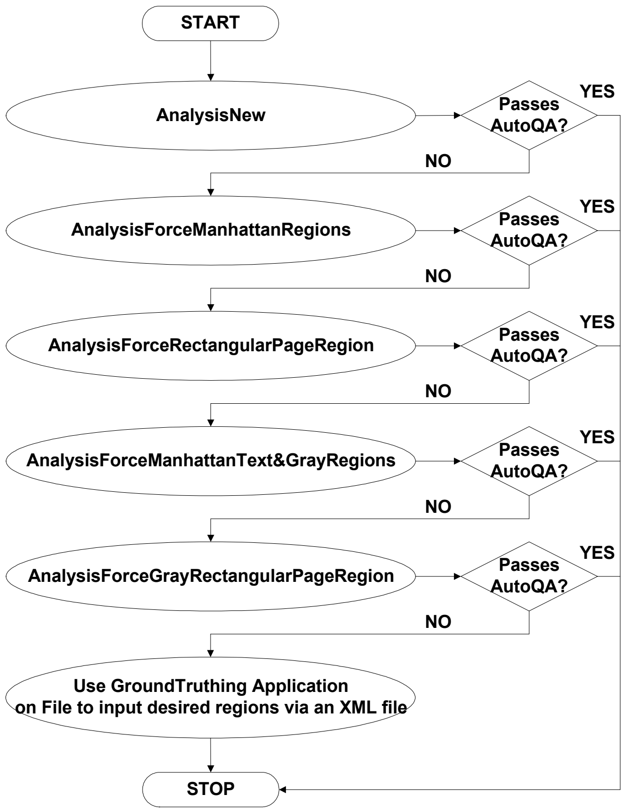

START
AnalysisNew
Passes AutoQA?
YES
NO
AnalysisForceManhattanRegions
Passes AutoQA?
YES
NO
AnalysisForceRectangularPageRegion
Passes AutoQA?
YES
NO
AnalysisForceManhattanText&GrayRegions
Passes AutoQA?
YES
NO
AnalysisForceGrayRectangularPageRegion
Passes AutoQA?
YES
NO
Use GroundTruthing Application on File to input desired regions via an XML file
STOP

Figure 1. AutoQA process exhausted before sending pages to the GTE. If the original zoning analysis (AnalysisNew) has a region error, the next stage (AnalysisForceManhattanRegions) reorganizes the regions into a Manhattan layout. Further attempts at AutoQA follow until finally the file is tagged for the GTE. In this manner, 84% of the original errors are automatically corrected, and only 0.8% of all pages need to be viewed using the GTE application. While this figure represents one specific workflow [1], the GTE can be used in combination with any zoning analysis scheme.

2. Application Overview

The application described here is used to ensure QA for a POD system when the default automatic zoning analysis steps (Figure 1) do not provide output of sufficient quality. In such cases, a human operator will need to examine the document and specify the correct page layout. In the MIT Press project, 99.2% of all pages passed AutoQA after the completion of the method AnalysisForceGrayRectangularPageRegion shown in Figure 1. The assessment is performed by an AutoQA verifier in which the PDF of the page gets converted to a TIFF image, which is then compared to the original scan. Large deviation between the two images implies an error in page processing. The 0.8% of pages remaining after AutoQA have to be processed manually. On our 1.2x10^6-page corpus, this turns out to be nearly 10^4 pages that either have to be looked at by a human operator or else run in "copier" mode (i.e. the page is directly copied to the output PDF with OCR run separately, a visually less appealing option). Another 1-2% of pages may need to be processed manually based on the visual appearance of the reconstructed (output) document. This fast-screening method can be done on a "book at a time" scale by scrolling through the pages of a book on a display. Borderline pages (passing AutoQA, but having intermediate correlation values) can be tagged for this screening, dubbed "VisualQA". This VisualQA is typically only necessary if the quality of the original scans is suspect. VisualQA is performed on pages that pass Auto QA but have some (usually small or subtle) visual defect, often the consequence of scanned originals with poor contrast or poor exposure. Such defects include rendering photos in 1-bit and rendering line drawings in grayscale (8-bit). Most are binarization/gray issues, whereas lost regions and segmentation errors are mainly captured by AutoQA. This means that nearly $3x10^4$ pages in our set of > 10^6 pages had to be reviewed with the GTE application. Of course, given different sizes and complexities of corpuses, these percentages and values will differ for each project. For example, scans made at better original contrast (which generally incur higher operational costs) have much lower error rates, obviating the need for significant user intervention. Even in these cases, however, we have found the GTE to be useful. Our goal, regardless of error rate, was to limit the review process to 2 pages/minute (the authors, after a few training pages, achieved a throughput of 5 pages/minute), allowing for multi-tasking during the review. This means our pages could be processed in $1.5x10^4$ man-minutes (6.25 man-weeks). If this corpus is representative, the manual rate is $1.2x10^6/6.25$ pages/week/man or $2x10^5$ pages/man-week.

The application (GTE) was written to graphically specify page layout and then generate an XML description of the page. After starting GTE, the user opens the file with the page image displayed at either low resolution (75 dpi, faster) or the higher resolution (150-300). She can then run automated region analysis (preferably with the same region analysis engine used in Figure 1, AnalysisNew, which is a full-fledged zoning analysis engine used in HP PrecisionScan) to provide a starting point of the page layout, or she can start with no regions. Regions—rectangular or polygonal areas of the page often defining a single object like a paragraph or a picture—can be drawn and their type (text, drawing, photo) and bit depth (bw, gray, color) can be set or modified. The regions should be contiguous and not overlap based on publishing and repurposing requirements. Different region management schemes can be used to replace this "nonoverlapping" region management scheme for other end-applications or purposes. The user can also define the "visible area" of the scanned document, which is the smallest rectangle containing all the regions and whatever border around them is also preferred (this allows margin setting). The visible area is automatically updated when new regions are added.

Once the user is satisfied with the region placement and properties, the page description (metadata) is saved. The metadata generated is sufficient to reproduce the intended layout or reuse it on similar pages as a template. Once the user has generated this region information we need to output it for further use. Since we needed a platform-independent format, XML was a logical choice as the transfer mechanism. While a binary format is more efficient, the amount of region data is small compared to the TIFF size. We also decided not to use a (non-XML) database to limit cost and licensing issues. Reading the (XML) data from a simple text editor or browser also offers advantages.

The user can specify all zoning regions simultaneously or individually. Individual region formation in the GTE uses the "Click and Select" (C&S) UI tool [2] and/or a rubberband tool to quickly generate rectangular regions; and "polygonal" C&S and/or a lassoing tool to generate polygonal regions. Margins are separately specified with a rubberband tool. The region manager used for the GTE application is modular, and for publishing purposes enforces the following: (1) no overlapping regions, (2) no crossover of vertices during polygonal region definition, and (3) no regions extending beyond the boundaries of the image.

3. Benefits of the Approach

Other systems for ground truth exist [3-4], but the GTE offers a number of potential advantages, including the following:

1. The output generated by the GTE—namely, the XML file or other representation of the metadata—can be used as part of a larger system (such as web-page generation, DOC/PDF/RTF destination, or as a publishing/print-on-demand program), or by itself for document comparison, clustering, templating, etc.

2. Because ground truthing can be performed on a lower-resolution version of an image, one logical implementation of the GTE is as a distributed application, where the images and the processing power to render them in reduced resolution exist, combined with an Internet connection of reasonable bandwidth to ship a reduced resolution version of the image. The compression ratio from original TIFF to a ground-truthable JPEG is > 40, meaning a "process locally, distribute globally" architecture for ground truthing is reasonable.

3. The XML output as generated by the GTE is used as input to a POD system that then uses the XML file to "proof" its layout, producing typically, < 200 kB output .PDF files from 8 MB input TIFF files (Using slightly lower quality JPEG, e.g. 75% of full quality JPEG, for the 8-bit and 24-bit regions results in even greater compression of approximately 100x, to 80 kB/PDF page). The print quality is maintained because the regions are fully specified by the GTE, ensuring best possible QA.

4. The GTE can be used to "snap" regions to a template in case of slight shifting between scans.

5. Smart scaling as file is opened—allows maintenance of boundary information without blurring upon re-scaling (searching for integral multiples).

4. Conclusions

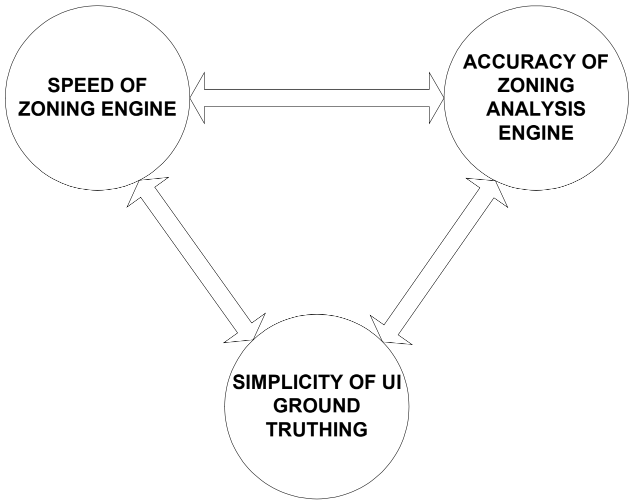
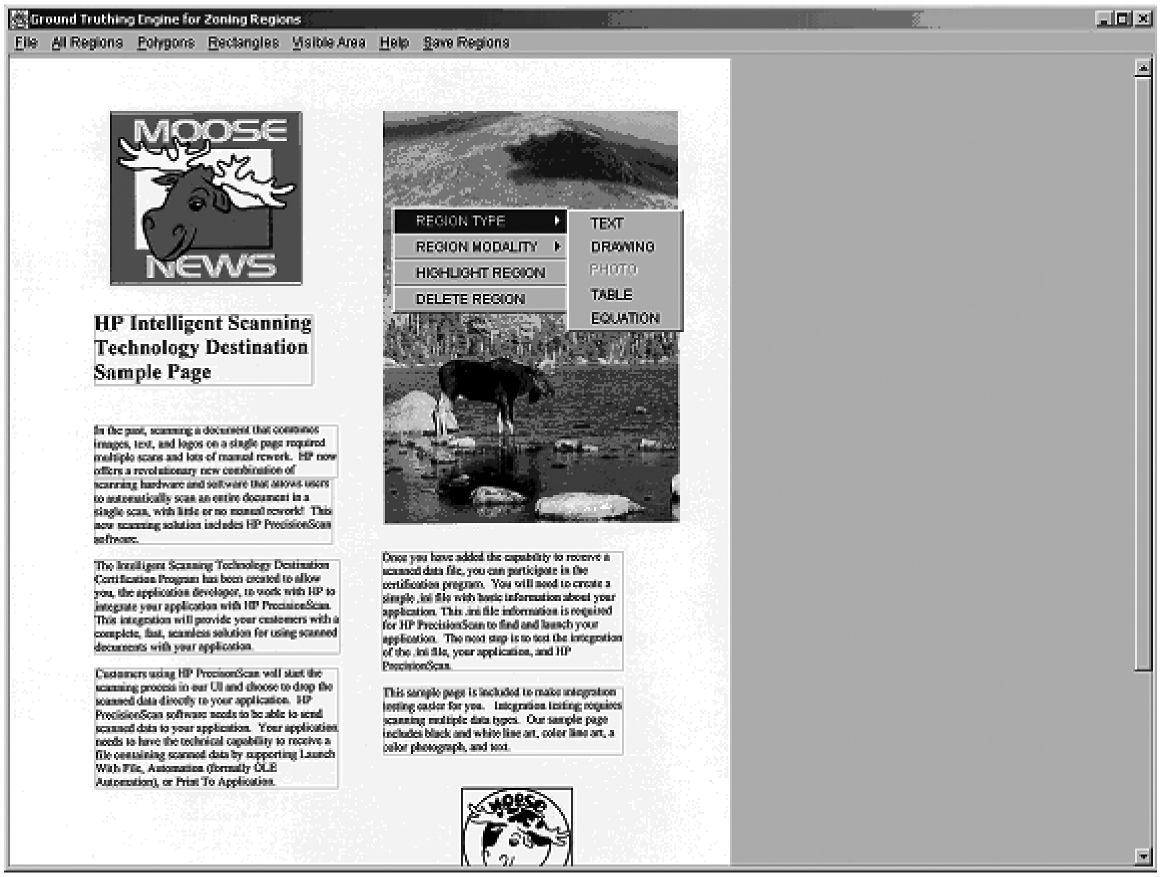

SPEED OF ZONING ENGINE
ACCURACY OF ZONING ANALYSIS ENGINE
SIMPLICITY OF UI GROUND TRUTHING

Figure 2. In the design and development of the ground truthing engine, we considered the trade offs between the speed (performance) of the zoning engine, the accuracy of the zoning engine, and the ease of performing the ground truth. Emphasizing any of these at the expense of the others tends to decrease page throughput in terms of pages/engineer-hour.

In coding and deploying the GTE applications, we utilized five effective design strategies worth noting. (1) The zoning analysis engine was designed independently of the region management model for the system, for modularity and to allow multiple fulfillment paths. (2) The production system benefited from automation wherever possible—including the automatic generation of log files in case of errors. (3) Multiple and redundant methods for ground truthing (menus, left and right clicks, hotkeys, tool tips, templates, etc.) were provided. (4) A simple, clean XML-based description of the layout [5] was generated as part of the ground-truthing process. In spite of larger file size in comparison to binary, these XML files comprise only 0.5% of the size of the original TIFF files, and less than 2% of these TIFF files are ever processed by the GTE application, meaning that these XML files amount to no more than 0.01% as much memory as the original TIFF files, and 1% as much memory as the final PDF files. (5) The ground truthing engine is tuned to increase its throughput rate. In general, trade-offs for throughput depend on the **speed** of the zoning engine (or fulfillment system), the **accuracy** of the zoning analysis engine, and the **ease of ground truthing** from the UI (Figure 2). To increase the **speed** of ground truthing, we used the same zoning analysis engine in the fulfillment (TIFF to PDF transformation) process as in the GTE application. For **accuracy**, we tried to optimize by making the definition of vertices as simple as possible. For **ease of ground truthing**, we made the zoning analysis engine work fast on simple pages, since for complex pages we are more likely to "hand-draw" the exact regions anyway. Figure 3 shows the original Java-based ground truthing engine being used to check the region typing of a photo.



HP Intelligent Scanning Technology Destination Sample Page

In the past, scanning a document that combines images, text, and logos on a single page required multiple scans and lots of manual rework. HP now offers a revolutionary new combination of scanning hardware and software that allows users to automatically scan an entire document in a single scan, with little or no manual rework! This new scanning solution includes HP PrecisionScan software.

The Intelligent Scanning Technology Destination Certification Program has been created to allow you, the application developer, to work with HP to integrate your application with HP PrecisionScan. This integration will provide your customers with a complete, fast, seamless solution for using scanned documents with your application.

Customers using HP PrecisionScan will start the scanning process in our UI and choose to drop the scanned data directly to your application. HP PrecisionScan software needs to be able to send scanned data to your application. Your application needs to have the technical capability to receive a file containing scanned data by supporting Launch With File, Automation (formally OLE Automation), or Print To Application.

Once you have added the capability to receive a scanned data file, you can participate in the certification program. You will need to create a simple .ini file with basic information about your application. This .ini file information is required for HP PrecisionScan to find and launch your application. The next step is to test the integration of the .ini file, your application, and HP PrecisionScan.

This sample page is included to make integration testing easier for you. Integration testing requires scanning multiple data types. Our sample page includes black and white line art, color line art, a color photograph, and text.

Figure 3. Java-based Ground-Truthing Engine with right-click over region pop-up menu shown. Text, black and white line art ("moose" in lower right), color line art ("moose news" in upper left) and photo (REGION TYPE -> PHOTO shown in pop-up menu) regions are all shown here (color-coded in the actual application). Here the user is verifying that the "REGION MODALITY" of the "Moose News" drawing is color (while this will be obvious to most people in the original color-coded application, since it is outlined in purple, this right-click activated ability to check and set region type and modality is necessary—recall that ~4% of all people have some form of color-blindness, and that this figure is in black and white!). Here, the user has verified that the region modality is "color" since the pop-up submenu has "COLOR" grayed out (the user can change it to BW or GRAY if she wants).

5. Acknowledgements

The authors thank Douglas Heins, Teresa Ehling, Sherif Yacoub, Yan Xiong, Xiaofan Lin, Joann Becker, Sheelagh Hudleston and John Burns for their insights, support and intelligent comments throughout the project.

6. References

[1] http://mitpress.mit.edu/main/feature/classics/MITPClassics_release.pdf, MIT Press Classics release web page.
[2] Lee, J.P., Lopez, P.D., and Simske, S.J. "Click and Select User Interface for Document Scanning," U.S. Patent no. 6,151,426, November 21, 2000.
[3] Altamura, O., Esposito, F. and Malerba, D. "Transforming Paper Documents into XML Format with Wisdom++," International Journal of Document Analysis and Recognition, 3(2):175-198, 2000.
[4] Wisdom home page, http://www.di.uniba.it/~malerba/wisdom++/.
[5] Simske, S. "The Use of XML and XML-Data to Provide Document Understanding at the Physical, Logical and Presentational Levels," In Proc. of the ICDAR99 Workshop DLIA, Sept. 1999.
[6] Sturgill, M. and Simske, S.J. "A proofing, templating and purposing engine in Java and C#/.NET," Hewlett-Packard Technical Report HPL-2002-272, 25 pp., 2002.