Remastering a Technical Book for Web Distribution

Don Lancaster
Synergetics, Box 809, Thatcher, AZ 85552

http://www.tinaja.com
don@tinaja.com
(928) 428-4073

Back in **GuruGram #110**, we looked at some of the insider secrets of **re-editing older VHS videos for web distribution**. An even larger problem involves updating older technical books that were "precomputer" and had no source code or cyber records available. It turns out you can now get "fairly decent" results with minimal effort, or a "surprisingly good" product with considerable extra time and effort.

Our primary focus here will be taking an older technical book having mixed text, graphic line art, and perhaps a few halftones. And then attractively presenting it as an on-web .PDF file. Done so with the minimum file sizes, fully searchable, and the highest possible text and graphics quality. A key tool in the process is the **ClearScan OCR Optical Character Recognition** found in Acrobat 9 and newly upgraded in **Acrobat 10**.

Examples of still-being-improved works in process can be found **here** and **here**.

Addressing Economics

Here are some real-world guidelines to the crucial issue of who owns what…

> **It is by far best if you are the current and sole copyright owner of the material being remastered.**
>
> **NEVER remaster anything a current corporate entity is presently profiting from or expecting to profit from.**
>
> **In cases where ownership is questionable or ambiguous, carefully evaluate your possible risks and rewards.**

Many classic technical books may be long out of print and their publishing houses merged or resold many times. Which raises issues of who owns the copyright and what, if anything, they are likely to do about your potential infringement.

In my own case, **I have specifically reserved any and all electronic rights from all of my current print publishers**. More guidelines…

> **Extensive rewrites or rework can sometimes give you some "wiggle room"**
>
> **AND …**
>
> **A major revision of a work can usually be issued a new copyright.**
>
> **BUT …**
>
> **Minor revisions can create a DERIVATIVE WORK whose legal limbo can be fraught with peril.**

And most important…

> **NEVER include DRM Digital Rights Management in ANY ebook in ANY form!**
>
> **If your material is too expensive, it WILL be stolen.**
>
> **All DRM does is piss off legitimate users.**

Before you begin any eBook project, the key question to ask is "**How will you be paid for your time and effort?**" The days of getting filthy rich just by getting "into print" are long gone. And **the perceived value of books and any book-like objects is plummeting**. These days, more often than not, an **indirect** approach may prove best.

I feel it is best to define two possible levels of eBook effort…

> **LEVEL I eBook — Spending the minimum time and effort needed to "just barely" get your project on to the web.**
>
> **LEVEL II eBook — Spending the time and effort needed to do the job right, including minimum file sizes, clearest possible typography, full text searching, the best possible images, vector converted art, and very carefully optimized (or even redone) halftones.**

Very often, a "two stage" approach can be used. In which you improve things later on should interest and funding warrant. This can be most effective when you have more than one eBook you want to get online as quickly as possible.

Getting an ISBN

The "official i.d." of any book (printed or electronic) is called an ISBN, short for **International Standard Book Number**. While older ISBN's were ten digits long, current variations require a full thirteen digits…

> **It is ABSOLUTELY ESSENTIAL that you have an ISBN for ANY book, printed, web, or electronic.**
>
> **The ISBN should be clearly identified on the title page and on the back cover. Also on the spine of any printed version of the book.**

A secondary reason for getting an ISBN is to enter any .JPG cover art into an easily accessed master directory data base. While there are a number of web soures for single ISBN's, the issuing agency is **Bowker** and they will be normally released in blocks of varying sizes. Per details found **here**.

Note that getting an ISBN does **not** register it. **Registration is a separate process you should go through once your eBook is ready for distribution**. Issued ISBN's are good for years, and **Bowker** internally keeps track of your numbers for you.

Printed books also require a bar code on their rear cover. This is not quite as essential for eBook only web distribution.

Sourcing Your Text

There are two normal ways to input your book pages: **Scanning or photography**. In either case, you'll want to get all of the pages as well aligned and as precisely rotated as possible. This is particularly important for the next stage OCR **optical character recognition** needed to convert your input images into capturable and searchable ebook text. OCR conversion, of course, is needed for minimum file sizes, maximum layout flexibility, best searchability, and handicapped aides.

Your simplest and best source is often to…

> **Purposely destroy a copy of the book to extract flat pages for scanning or high resolution photography.**

A used book copy can often be found at **Amazon** or **eBay** at low cost.

Your easiest way to reduce your book to its individual pages is to use a power paper cutter. Typical charges for a single cut at a quick printer should only be a dollar or two.

If you absolutely must preserve an only copy of your input book, then fancier and more costlier techniques may be needed. As you are likely to quickly discover, **an ordinary copier is totally useless here**.

Instead, one popular route can be called the **Vee 45 Method**. In which the book is carefully held open at two 45 degree angles off horizontal and then alternately photographed with one or two high resolution digital cameras.

Preferable at least 12 Megapixels or better.

A wide variety of Vee 45 machines is available with prices ranging from nearly free to many tens of thousands of dollars. The usual tradeoff is processing time when lots of books have to be input with minimum labor. A very interesting collection appears **here**. A very low cost do-it-yourself cardboard version can be found **here**.

In what follows, we will assume destruction of an input book into its individually scanable pages.

Some Scanning Details

The game plan is to scan each page into Paint, and then use the **ClearScan OCR** of **Acrobat 10** to convert as much of the page as possible into as readable and as searchable text as can be done.

While you can quickly and conveniently scan directly into Acrobat, going into **Paint** first gives you all sorts of additional control and pre-editing advantages.

ClearScan has an amazing feature in which is creates "new" fonts that can end up a **credible average** of the existing fonts being scanned. These new fonts often look quite good, are easily improved, and fully preserve the "look and feel" of the original. Even including fill justification. All done without any restrictions on the original font rights.

The result of a ClearScan is a .PDF document page of mixed text and bitmaps. The text part is usually amazingly good, very compact, easily searched, and easily read out loud or reformatted. And, yes, your text portions can simply be further improved using details we'll look at shortly.

There will be four types of bitmaps remaining: Glitches and stuff that ClearScan could not recognize; Bitmaps representing line art; Similar line art bitmaps convertable to vectors; and Bitmaps representing halftones. Again, all of these can be dramatically improved with our upcoming details.

One alternate to ClearScan is the **Scan Tailor**. I overwhelmingly prefer Acrobat.

Here are some scanning guidelines…

SOME SCANNING GUIDELINES

Make sure you have at least five gigabytes of RAM available. Bitmaps gobble up memory at a prodigious rate.

Keep your scanner as clean as possible and reclean it often, at least once every five pages.

Scan at the highest feasible rate, preferably 600 DPI. The higher the resolution, the better the character recognition.

Scan into Paint, rather than directly into Acrobat. This can give you "edge cleanup" options, easy page cropping by using "attributes", and the ability to pre-repair any text portions that may include blobs or dropouts or any hard to recognize characters.

Scan as a black and white image when and where applicable. Otherwise file sizes will become outrageous and your OCR results may not be nearly as good.

While the deskewing in Acrobat 10 is outstanding, avoid using it! Align everything as carefully as you possibly can. This will minimize "stepping" problems with remaining bitmaps and especially any halftones.

The IrfanView Utility can be most useful to deskew or rotate in one-tenth degree intervals. Or to change the size of any bitmap by altering the resolution in its header.

Always check your ClearScan results by searching for an obvious text word or two.

After optimizing your individual pages, the insertion code in Acrobat can be used to form a complete book. Followed by one or more of the speed and size optimization selections.

Naturally, **it is very important to proof the final book assembly for any missing, repeated, misaligned, or wrongly sized pages.**

It may not be immediately obvious how to check ClearScan to see how good a job it did. In older Acrobat versions, the **find OCR Suspects** selection did not work at all with ClearScan.

Here are four different routes to find out which text is nice looking and compact and searchable, and which still remain sloppy unsearchable huge bitmaps…

HOW TO TEST CLEARSCAN RESULTS

RETOUCH METHOD—

Select Tools--> Advanced Editing-->Touchup Text. Then mouseover all available text. Solid blue words will be both smooth and searchable. White letters are still bitmaps. Words with white stripes are smooth but not yet serchable as a single word.

RESIZE FONTS METHOD—

Select Tools--> Advanced Editing-->Touchup Text. Then mouseover all available text. As above. Then use Properties to switch to a one point type size. OCR candidates will remain in their original positions, while small converted fonts will move.

MAGNIFY METHOD—

Select extreme magnification and then mouse scan the full page. Rounded but slightly erratic characters are compact and searchable. Crude bitmaps remain just that and likely will need fixed.

POSTSCRIPT METHOD—

Export PDF as PostScript and view in an editor or word processor. Smaller bitmaps using the "sepimg" operator may be text that still needs repair or adjustment.

Improving Your Text

At this point, you have to decide whether additional time and effort is needed for a full Level II remastering. Sometimes it is best to defer making things as good as possible in favor of getting additional books online faster.

The ClearScan method produces both searchable text and remnant bitmaps. These will need separate treatment for later improvement.

At first glance, the new generated fonts seem to have their own encryption and appear to be locked out from your revision. But the searchAble text portions in fact can be surprisingly easy to further edit…

HOW TO EDIT CLEARSCAN RESULTS

1. **Export your .PDF page as a bitmap to the same resolution as your original scan. Be sure Paint is in its Black and White mode.**
2. **Improve any problem characters by clarifying them, straightening stems, adjusting serifs, eliminating external and internal collisions, making them slightly narrower, etc...**
3. **It is very important to correct each and every instance of any particular character. Otherwise the "good" ones will once again get averaged in with the "bad" ones.**
4. **A separate paint page can be used to stash reference alphabet characters when and as they are generated for later use.**
5. **Should bitmaps end up the wrong size, the resolution values in their header can be changed as needed using the IrfanView Utility**
6. **Send your improved bitmap back to Acrobat for ClearScan reconversion. The process can be repeated as often as needed.**

Editing Remnant Bitmaps

There often may be four types of smaller bitmaps remaining on any ClearScan page. **The smallest of these bitmaps will usually be something that ClearScan did not have the faintest clue how to deal with**. Such as a really unrecognized character or an ink blob. You should attack these first with a goal of eliminating them all. The same process you used above to edit ClearScan text should work just fine.

A very useful method of separating bitmaps for later analysis and use is to **export your .PDF file as PostScript and then view the new file in a suitable editor or word processor**.

Your second type of remnant bitmap will be some **line art that can be converted to vector form**. Vector figures in **PostScript** will typically be much shorter, look a lot better, have ridiculously better typography, and should magnify much more smoothly. But lots of time and effort may be involved in the vector conversion.

While going from Acrobat 9 to **PostScript** back to Acrobat can give you dramatic size and quality improvements, it does seem to have one gotcha: If you try this directly, **any ClearScan text searches will be lost!** It turns out that ClearScan will only work (and only generate its glyph equivalents) using a bitmap input. I am still exploring a suitable workaround for this.

The third form of bitmaps represent **line art that is good enough or can often be improved upon while still in its bitmap format**. Once again, placing the bitmap in Paint should let you make lines more uniform width, eliminating any stairsteps from deskewing, and improving small fonts. But remember **you will want to stay in black and white if possible**. Full RGB bitmaps will be considerably larger.

An interesting alternate method is to purposely go to full RGB, trace all of your improvements in red, and then use some **custom code** to create a new bitmap that is black if the input was red and white otherwise. This greatly simplifies your getting rid of any "noisy" lines or other "sugar".

Our fourth bitmap represents a halftone and usually needs special treatment…

Dealing with Halftones

In general, scanned halftones tend to look awful…

> **A "mesmerizingly awful" halftone is much easier to create than one that is "almost not half bad".**
>
> **Decent electronic halftones demand extreme effort.**

A useful rule…

> **If at all possible, go back to the original photo source. Or otherwise seek out a better halftoning method.**

One cause of the halftone problem is scanning to black and white, rather than RGB or grayscale. This tends to make halftones extremely dark and murky. A second problem can be caused by excessive deskewing that tends to move black dots further on top of each other. And a third issue may involve **Moire** patterning.

A good rule might be to **try and improve your halftones without expecting any superb results**. Unless extreme measures are taken.

Here are some ploys that sometimes may help. Especially on images of technical items…

SOME HALFTONE IMPROVEMENT PLOYS

1. **Rescan the halftone to RGB or grayscale. Then use it to replace the original.**
2. **Use Imageviewer32 or IrfanView to change the gamma to somewhere between murky dark and a washed out gray.**
3. **Color tints may help. Knock out backgrounds to white. Untilt to architect's perspective.**
4. **Outline edges gently with thin grays or blacks. Chase details along their length for consistency. Make surfaces more uniform.**
5. **"Average out" areas. Eliminate any shadows that are unneeded or murky.**
6. **Make sure scans are horizontal! Deskewing does very bad things to halftones.**
7. **Make open holes white. Use repetition to make any pins or legs more consistent.**
8. **Symmetry via mirroring or flipping can replicate your best part overall.**
9. **Always try to do your own scans, rather than relying on others. Use 600 DPI when applicable.**
10. **Use raw PostScript and my Gonzo Utilities.**
11. **Preserve legality and provenance for anything of historical context.**
12. **Replace true circles where needed. Add extra single white or black dots to darken or lighten.**
13. **Make the worst the best. If all else fails, divert attention elsewhere.**
14. **Take half and leave half on the stuff that needs corrections the worst.**

Adding Value Added

The web is a far more powerful distribution media than any printed book could ever hope to be. Additional features and services can easily be included.

Links, of course, are a prime example. Autotracking link code can be found **here**. This will automatically reposition on text edits.

Color or tints at one time were outrageously expensive in a printed book; these are now virtually free on the web. Updates or revisions of a book were also once outrageously expensive. But a remastered web release can now be corrected or expanded at any time.

Generating an index for a traditional book involved major time and effort for only uselessly partial results. But full text searching of an eBook can now be more or less trivial. **JavaScript over Acrobat** offers all sorts of vastly improved navigation possibilities. Among countless other value added features. Not to mention sound, read aloud, video, and animation clips.

Let's look at a value added example. Here is how to replace an image with a new JPG image that you can click expand to a new larger image or send your viewer elsewhere on the web…

```
/jpegimageprocwithlink {
     % hoffset voffset hres vres
save /snap2 exch def
/infilename exch store % passed pix file
/inurllink exch store % link filename
/photoscale exch store
/vpixels exch store
/hpixels exch store
translate % optionally adjust position
inurllink
setareaurl % autolink sizing
/DeviceRGB setcolorspace % pick model
0 0 translate % set page position
hpixels vpixels scale % magnify unit square
photoscale dup scale
/infile infilename (r) file def % establish read file
/Data {infile /DCTDecode filter} def % define source
<< % start image dicationary
/ImageType 1 % always one
/Width hpixels % JPEG width in pixels
/Height vpixels % JPEG height in pixels
/ImageMatrix [hpixels
0 0  vpixels neg 0 vpixels ]
/DataSource Data % proc for filtered JPEG
/BitsPerComponent 8 % color resolution
/Decode [0 1 0 1 0 1] % per red book 4.10 >>
```

```
image % call the image operator
snap2 restore} def

/setareaurl { % for auto include routine
/cururlname exch store
mark % start pdfmark
/Rect [ 0 0
hpixels photoscale mul
        10000 div % remove to activate link
vpixels photoscale mul
       10000 div % remove to activate link
]
/Border [ 0 0 0] % [0 0 0 ] = none;
/Color [ .7 0 0 ]
/Action <</Subtype /URI /URI cururlname>>
/Subtype /Link
/ANN % annotation type
pdfmark % call pdf operators
} def

/new37_image {save /af1 exch store
200 200 5100 6580 0.118
    % xpos ypos hpixels vpixels scale
(http://www.tinaja.com) % url or distant pix link
(C:\localjpgfile\smallpix.jpg) % local pix
jpegimageprocwithlink
af1 restore} store

0 0 new37_image % this does it
```

For More Help

Sourcecode for this **GuruGram** appears **here**.

Custom consulting is available. Additional info on similar topics can be found on our **PostScript** and **Acrobat** libraries. You can also **email us** or you can call (928) 428-4073 for further help.