

A Scalable Solution for Integrating Illustrated Parts Drawings into a Class IV Interactive Electronic Technical Manual

Molly L. Boose, David B. Shema, Lawrence S. Baum
The Boeing Company, Seattle, WA

Molly.Boose@boeing.com, David.Shema@boeing.com,
Larry.Baum@boeing.com

Abstract

This paper discusses a scalable solution for integrating legacy illustrated parts drawings into a Class IV Interactive Electronic Technical Manual (IETM) [1,2]. An IETM is an interactive electronic version of a system's technical manuals such as for a commercial airplane or a military helicopter. It contains the information a technician needs to do her job including troubleshooting, vehicle maintenance and repair procedures. A Class IV IETM is an IETM that is authored and managed directly via a database. The end-user system optimizes viewing and navigation, minimizing the need for users to browse and search through large volumes of data.

The Boeing Company has hundreds of thousands of illustrated parts drawings for both commercial and military vehicles. As Boeing migrates to Class IV IETM systems, it is necessary to incorporate existing illustrated parts drawings into the new systems. Manually re-authoring the drawings to bring them up to the level of a Class IV IETM is prohibitively expensive. Our solution is to provide a batch-processing system that performs the required modifications to the raster images and automatically updates the IETM database¹.

1. Introduction

Illustrated Parts drawings are used extensively in the maintenance and repair of commercial and military aircraft. A typical aircraft may have from several hundred to many thousands of illustrated parts drawings that identify hierarchies of assemblies and parts. These assemblies and parts are labeled to show how they fit together (Figure 1). Reference labels and pointer-lines are used to identify specific parts. Supporting parts information is stored in the IETM database and is indexed by the reference labels on the illustrated parts drawings. The drawings may also show multiple pictures, or details, because the drawings were authored for paper (i.e. the authors wanted to fill the page).

¹ Patent Pending, U.S. Application No. 10/318,921. Dec 13, 2002

There are three details on Figure 1: detail G, detail H and Detail J.

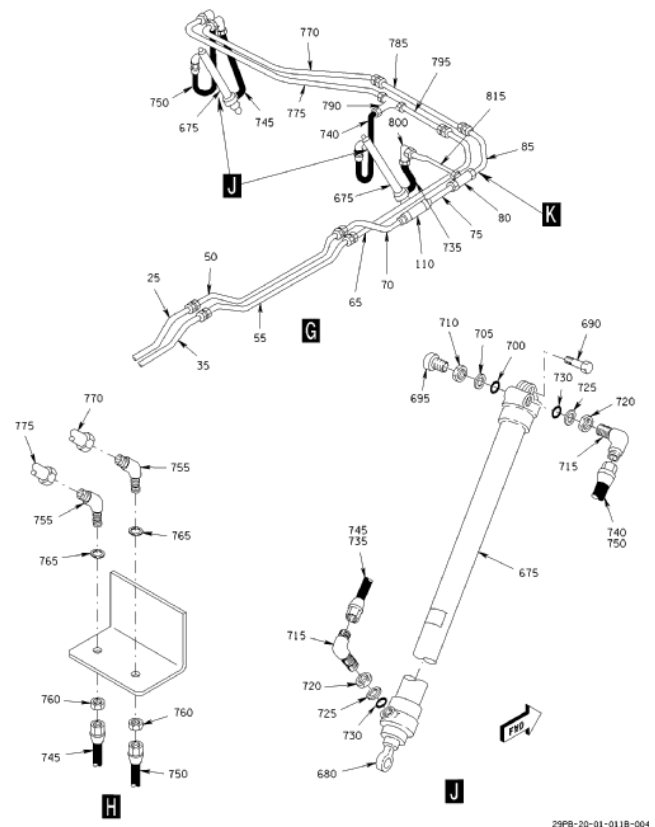


Figure 1. Sample raster illustrated parts drawing.

Figure 2 shows the three types of reference labels for the illustrated parts drawings in our case study. There are, **locator, detail and item number** reference labels. *Locator* reference labels point to a portion of a detail that is linked to another detail (i.e. a zoomed-in view of the area identified by the locator pointer-line). *Detail* reference labels identify an individual detail that a locator references. *Item Number* references are indexes to the parts records stored in the IETM database (or in the legacy parts database).

The reference labels are vestiges of the paper world used for cross-referencing. In Class IV IETMs only information

that is relevant to the current step or task being performed may be displayed to the user, so we must transform the drawings to meet the Class IV requirements.

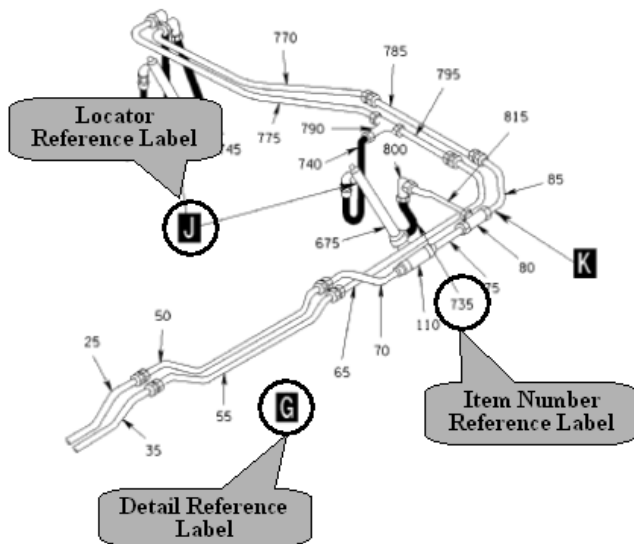


Figure 2. Detail, locator and item number labels.

The first requirement for our automated tool is to separate Figure 1 into 3 separate details (G, H and J for Figure 1). We do this not only to meet the Class IV requirement that only relevant information be displayed, but to significantly improve the resolution for the end user by displaying only the relevant detail on the computer screen. The second requirement is to erase each reference label and associated pointer-lines from the individual details. A detail image with all reference labels and pointer-lines erased is called a *base graphic*.

For the base graphic, we create a layer (overlay) for each of the reference labels and any associated pointer-lines that we are erasing. We also create a hotspot, or click-able region, over the reference label location. When authoring a procedure or step for the technician, the author only needs to select a base graphic and specify which layers are to be visible. The tool automatically builds the electronic link from a hotspot to the associated parts information or detail.

When a reference label is made visible by the author, all of its associated pointer-lines are automatically made visible and its hotspot is activated. The IETM user can click on a hotspot to retrieve valuable parts information or to automatically zoom to a new detail.

We store all necessary information about the layers in the IETM database. This includes the layer's graphic id and the reference label content (i.e. the item number or locator letter for a reference). When the IETM author is writing the instructions for a particular procedure, she can choose a base graphic and then select exactly which layers should be made visible in the IETM. The base graphic is reused in the IETM for as many different steps or tasks as

the author deems appropriate, but with a different combination of reference labels visible each time. This allows the IETM to display relevant information only, eliminating extraneous information that has historically made technicians jobs more difficult.

Figure 3 shows a Class IV IETM with an Illustrated Parts Drawing displayed. There are three reference labels visible on the drawing (i.e. three layers are visible). These three reference labels are the only ones that are relevant to the current step the technician is performing. The original item numbers or locators are not shown because the user has no need for this information. The user simply clicks on the hotspot (shaded circle) to display related information, which could be parts data (for an item number reference) or another detail (for a locator reference). Our tool automatically updates the IETM database so that it knows the item number or locator for each layer and thus knows what to display when a hotspot is clicked.

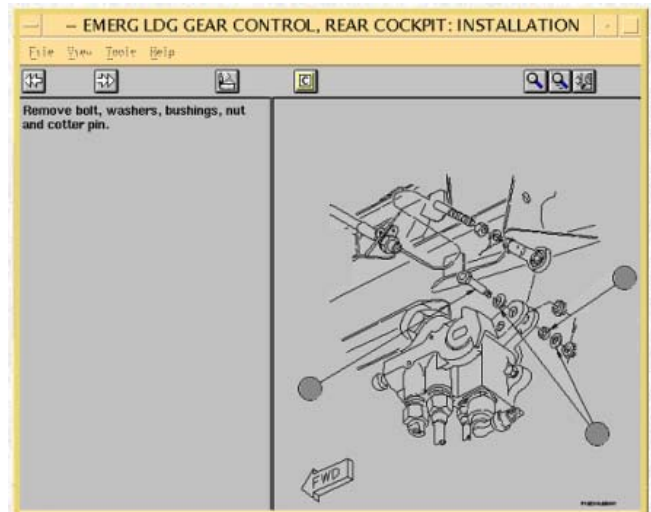


Figure 3. A Class IV IETM displaying an Illustrated Parts Breakdown drawing with three reference labels visible (and selectable).

2. Approach

The illustrated parts drawings we are working with are in **raster** format. In some cases they may be scanned paper images and in other cases they may have been authored electronically as raster images. Our automated solution employs Optical Character Recognition (OCR), page layout analysis and line finding techniques.

Our approach is to exploit *domain-specific knowledge* about the set of drawings to be transformed. Our tool reads a file with domain-specific parameters such as the range of valid pointer-line lengths and maximum distances between pointer-lines and their associated reference labels. The parameter values are used extensively by the tool and provide a flexible method for handling unique sets of

drawings, minimizing the need to develop additional software.

We use DAFSLIB [3] to segment a drawing into blobs, or connected (adjacent) pixels and to perform optical character recognition. We use attributes about the blobs such as their bounding-box size or their saturation to identify them as characters or shapes.

We identify reference labels using OCR. The reference labels found are grouped into one of three categories based on physical attributes of the label such as size. We then use the categorizations to perform the modifications to the images.

We automatically discover and separate the individual details within a single drawing. This is done with proximity-based calculations using the detail labels as a starting point to identify and assign pixels to each of the individual detail images.

We automatically find pointer-lines for each of the individual detail images using the locator and item number reference labels as a starting point and line following techniques to trace the pointer-line from the tail to the head [4]. We erase the reference labels and pointer-lines from the separated image to create a reusable base graphic.

Figure 4 shows the **base graphic** “G” after all processing is complete. If you carefully examine the base graphic in Figure 4 you will notice imperfections where some of the pointer-lines have been erased. In a small percentage of cases, an illustrator must edit a newly created base graphic to ensure that the resulting detail is substantially identical to the original detail.

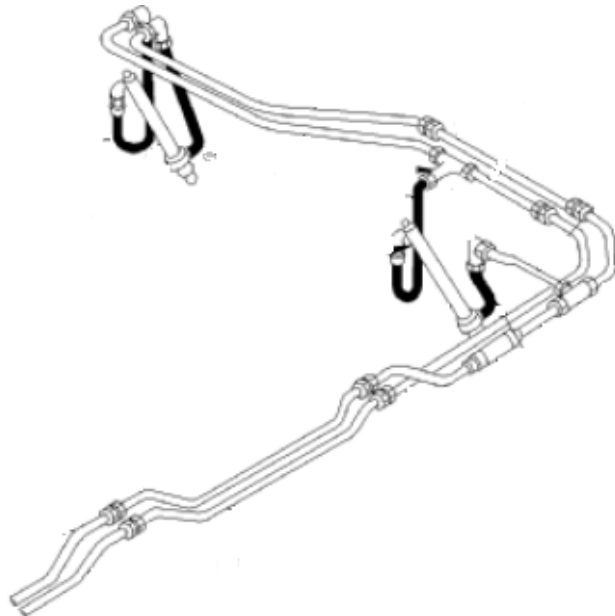


Figure 4. Base graphic “G” generated from processing the image shown in Figure 1.

2.1 Finding Reference Labels

In order to find the reference labels in our collection of raster drawings, we first build character templates unique to our set of drawings. We then match blobs of the right size against the characters. Our label recognition software forms words from the characters based on proximity characteristics (domain-specific parameter values). The label recognition algorithms also filter out any words that are not legitimate reference labels. The detail and locator labels can be distinguished from each other because detail labels do not have pointer-lines emanating from them as locator references do.

2.2 Separating Images

Once the reference labels have been found and categorized, we use the **detail** labels to begin separating our figure into separate detail images. We write one image file for each detail that is found. For the set of drawings in our case study, the detail labels appear below their associated pictures. We use this fact to find blobs whose pixels are contained inside a narrow search region as shown in Figure 5. We can dynamically enlarge the search region as necessary to successfully find at least one blob to assign to each detail by changing values in our parameter file.

Once we have assigned the initial blobs to the appropriate details, we begin iteratively assigning other blobs (outside the initial search region). We do this by starting with tiny tolerances (distances) and finding blobs whose pixels are very close to only one of the details. We say these blobs are unambiguous and assign them to the only close-by detail. We then iteratively relax the tolerances until every blob is assigned to an individual detail. An error condition occurs when we are unable to unambiguously assign a blob to a single detail. The software reports these errors in an error report, identifying the image that generated the error and manual separation is necessary.

During initial testing, we iterated once with a large distance tolerance yielding very poor results. We found that by varying the number of iterations and the minimum and maximum tolerances we were able to achieve very high accuracies.

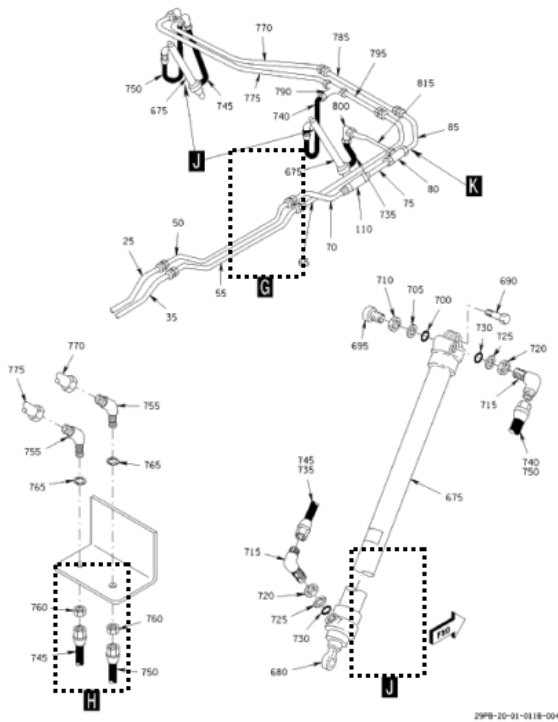


Figure 5. Initial search region for detail blobs shown by the dashed rectangle.

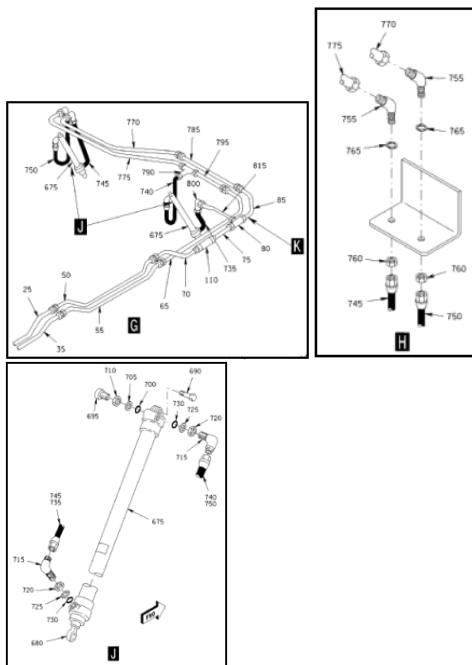


Figure 6. After image separation three images "G", "J" and "K" exist.

2.3 Finding Pointer-lines

When the individual images are separated, we begin looking for pointer-lines. Using the **locator** and **item number** reference labels, we find the closest pixel to the label and use this as our starting point to find an associated pointer-line. The line following algorithm requires that the line be a straight line (i.e. constant slope). We define domain-specific parameters for line thickness and line noise in addition to other parameters that improve our accuracy in finding and erasing the pointer-lines.

Starting from the tail, we begin forming a line until we reach a minimum pointer-line length that has been defined in the domain-specific parameter file. At this point, we calculate the slope and verify that the line is straight. Then we continue to grow the line, regularly recalculating the slope and comparing it to our initial slope. If the slope changes significantly, we end our line. We also end our line if we do not find another touching pixel along the pointer-line's path (a pixel must be found at a calculated location based on the pointer-line's starting point, current length and slope).

When we terminate our line, we apply additional conditions before adding it to our list of pointer-lines. For our case study, pointer-lines must have a filled arrowhead at the head. We construct the expected arrowhead region and then determine if the constructed region is filled (black) on the image. If we do not find the required arrowhead, we back up one pixel at a time, re-running the arrowhead test until we find an arrowhead or determine that the line is not a valid pointer-line.

For other data sets, a pointer-line may have a hollow arrowhead or a solid circle or no special head characteristic at all. These criteria are part of the domain-specific conditions implemented for each unique data set. Fortunately, technical illustrations tend to be highly standardized and we need only implement a few variants for the pointer-line head to support hundreds of thousands of legacy illustrated parts drawings.

Once we have established our list of pointer-lines, we begin the task of erasing them from the new base graphic we are producing. The erasing algorithm does not erase pixels from the pointer-line that touch the base graphic pixels in an attempt to minimize or prevent holes from forming in the base graphic. At times this algorithm incorrectly leaves behind pixels that should have been erased creating noise on the base graphic. There are also circumstances where leaving behind just the touching pixels is not enough and a hole results in the base graphic. We are continuing to improve the algorithms to reduce these adverse effects.

2.4 Generating Intelligent Graphics

The final steps are the generation of an Intelligent Graphic for each base graphic and the update of the IETM database. An Intelligent Graphic is an interactive image that allows an author to specify which labels and pointer-lines are to be displayed to an end-user for each step in the IETM. We embed the newly created base graphic (raster image) in a Computer Graphics Metafile Version 4 (CGM4) file [5]. For each reference label (and its associated pointer-lines), we create a CGM4 layer. We also create a CGM4 hotspot for each locator or item number reference label. Detail reference labels do not link to additional information and therefore do not have a hotspot.

For each layer created, information is stored in the IETM database. The information stored in the database enables the automatic link behavior that occurs when the end user clicks on a hotspot in the IETM. The IETM database contains the index (item number or locator) for every reference label. The IETM retrieves the index when the user clicks on a hotspot, and initiates a database query to display parts information (for an item number) or a detail (for a locator).

In the IETM authoring environment, the author writes a procedure or step and chooses one or more base graphics to associate with the step. Next the author selects the layers that are to be shown when the base graphic is displayed with the step. The base graphic may be selected any number of times for any number of steps. The layers that are displayed will be different for each unique step.

3. Results

By adjusting distance tolerances, the number of iterations and other domain-specific parameter values, results can be improved. Our first production run of the tool fully automated more than 80% of the work required to integrate Illustrated Parts Drawings into a Class IV IETM. Fewer than 20% of the base graphics required any human intervention and those required only minimal touch-ups using a raster-editing tool. More than 95% of the electronic

links were correctly identified and inserted into the Intelligent Graphic. We are continuing to refine the algorithms to achieve higher accuracies.

4. Conclusions

We are investigating how other types of raster technical drawings can benefit from similar approaches. In the immediate future we are interested in developing a tool to vectorize raster wiring diagrams for data mining, circuit identification and integration into Class IV IETMs.

5. References

List and number all bibliographical references in 9-point Times, single-spaced, at the end of your paper. When referenced in the text, enclose the citation number in square brackets, for example [1]. Where appropriate, include the name(s) of editors of referenced books.

[1] DoD Interactive Electronic Technical Manuals, <http://www.ietm.net/DoD.html>.

[2] Interactive Electronic Technical Manuals: General Content, Style, Format, and User-Interaction Requirements, Revision A, MIL-PRF-87268A, October 1, 1995.

[3] Dov Dori, David Doerman, Christian Shin, Robert Haralick, Ihsin Phillips, Mitchell Buchman, and David Ross. *Handbook on Optical Character Recognition and Document Image Analysis*, chapter The Representation of Document Structure: a Generic Object-Process Analysis. World Scientific Publishing Company, 1996.

[4] David S. Doermann. An Introduction to Vectorization and Segmentation. In *Proceedings of the International Workshop on Graphics Recognition*, pages 1-5, 1997.

[5] Computer Graphics Metafile (CGM), Version 4, International Standards Organization IS 8632: 1992