AUTOMATION OF CHANGE DETECTION IN ENGINEERING DRAWINGS

A Project Report

submitted by

RENUKA N

in partial fulfilment of the requirements
for the award of the degree of

MASTER OF TECHNOLOGY

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY MADRAS.

MAY 2013

THESIS CERTIFICATE

This is to certify that the thesis titled **AUTOMATION OF CHANGE DETECTION IN ENGINEERING DRAWINGS**, submitted by **RENUKA N**, to the Indian Institute of Technology, Madras, for the award of the degree of **Master of Technology**, is a bona fide record of work done by her under my supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Place: Chennai

Date:

Dr. A. N. Rajagopalan
Professor
Dept. of Electrical Engineering
IIT-Madras, 600036

ACKNOWLEDGEMENTS

I take this opportunity to express my deepest gratitude to my project guide Dr. A.N. Rajagopalan for his valuable guidance and motivation throughout the project. I am very grateful to him for providing his valuable time to guide me during the project.

It is a privilege to be a student in IIT Madras. I express special thanks to all my teachers for all the academic insight obtained from them. I also acknowledge the excellent facilities provided by the institute to the students.

I am grateful to Caterpillar India Private Ltd for providing me this opportunity to work with them and for providing all the necessary help during the project.

I would like to thank all my lab members with whom I had various fruitful discussions. Special thanks to all my friends for making my stay in IIT Madras an enjoyable one.

I am indebted to my parents for their unconditional love, support and guidance. I dedicate this thesis to them.

ABSTRACT

Engineering drawings play a very important role in the field of Engineering Design and Production. They provide a graphical means of understanding the information required for manufacturing a certain component such as dimensions, tolerance and geometry.

The changing technology is reflected in the change of machinery, infrastructures and various engineering equipments. One of the basic steps in applying these changes is by changing the engineering designs. When an engineer makes these changes, he generally does not document the changes, though it is very much necessary to document the changes in any design from one stage to another. As the changes are not readily available to the manufacturer, he has to go through all the dimensioning details in order to manufacture the new product which increases the time of delivery.

In order to avoid this delay, another intermediate person is involved to document the changes from one stage to another. Manually going through each and every dimension and documenting the changes is a very tedious task and also erroneous. This thesis is an attempt at proposing an algorithm to automatically detect the changes and document the same.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

Introduction

To adapt to changing technology, engineering designs are often modified. In general, only few modifications are made in the design to obtain the new design. If at all these changes are readily available to the manufacturer, he then needs to just do only few modifications over the previous settings to manufacture the new product. Otherwise, he will have to go through each and every dimension in order to manufacture the new product, which will delay the process of manufacturing by a significant amount of time. Generally the engineer who makes the changes in the drawing does not document those changes; a checker is the one who manually checks each and every dimension and indicates whether there was a change or not. This is a very slow, tedious and an error-prone process and also requires considerable amount of manual involvement.

To avoid all the delay and manual intervention, it necessary to have an automated system which does the comparison efficiently. Currently, Computer Aided Drawings (CAD) and Computer Aided Manufacturing (CAM) are used extensively in the process of making engineering drawings and manufacturing the products. Though these automate a lot of processes and help reduce human effort and intervention, the tools for comparing two drawings efficiently are not readily available.

A typical engineering drawing is shown in Fig. 1.1. Fig. 1.2 is obtained by modifying some of the dimensions of the engineering drawing shown in Fig. 1.1 and the dimensions in the green bounding boxes indicate the changed dimensions.

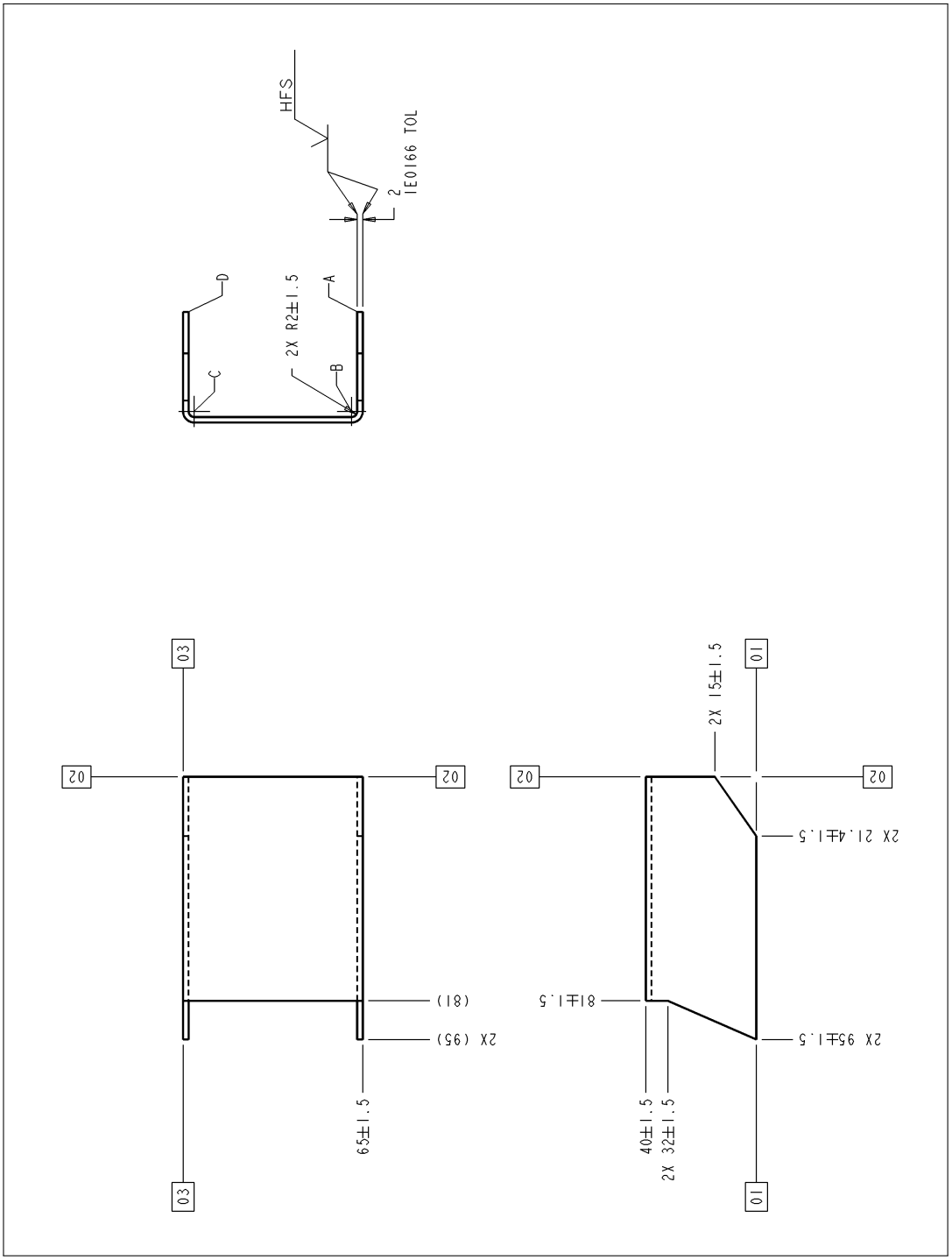


Figure 1.1: A typical engineering drawing.

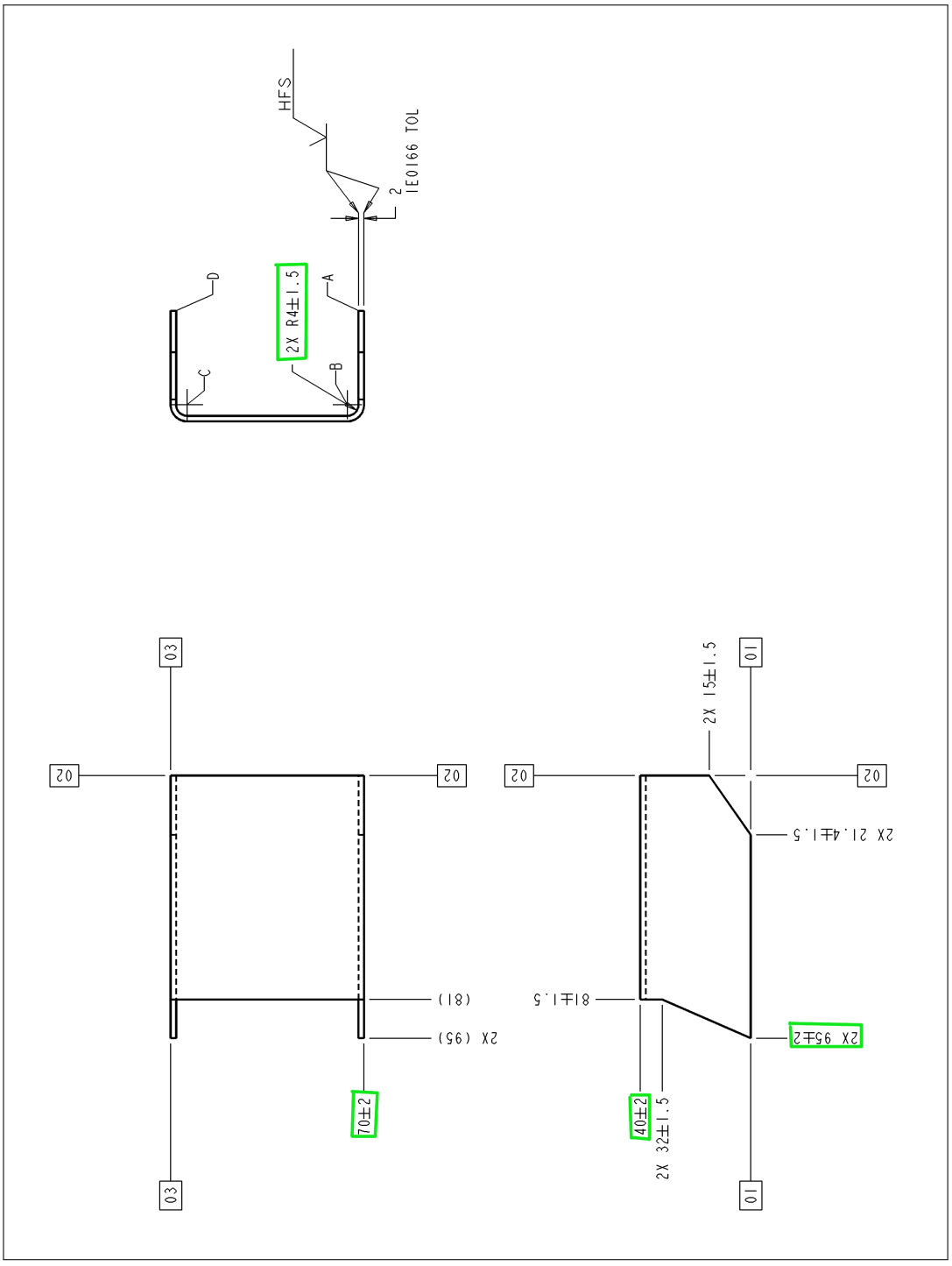


Figure 1.2: Engineering drawing obtained by modifying the drawing shown in Fig. 1.1. The dimensions which are modified are shown in green bounding boxes.

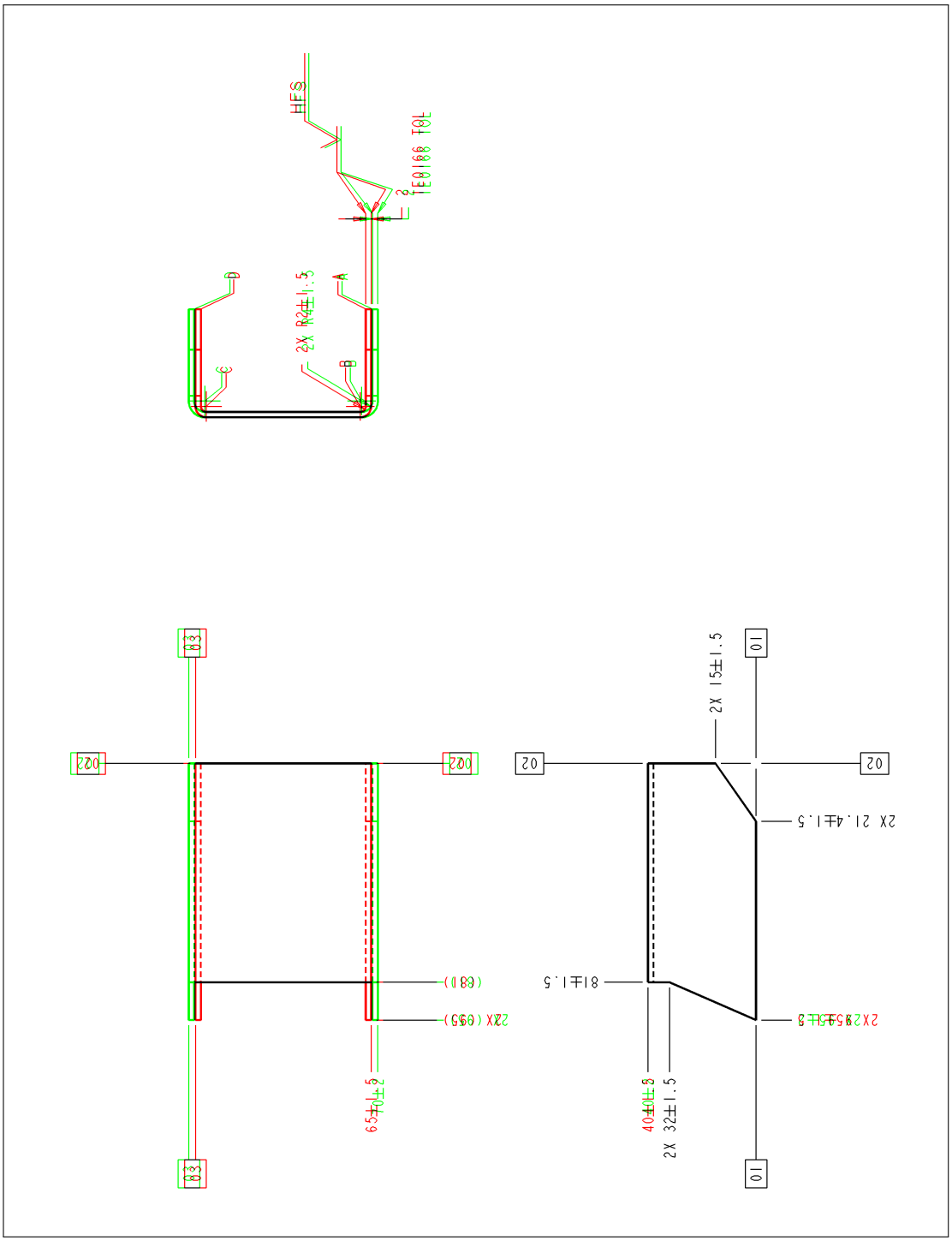


Figure 1.3: Result of comparison of the drawings shown in Fig. 1.1 and Fig. 1.2. Black lines indicate the unchanged regions, red indicates the deleted regions and the green indicates the added regions.

There are some image processing tools which are based on image subtraction currently available to compare such drawings. Fig. 1.3 shows the output when the drawings shown in Fig. 1.1 and 1.2 are compared using one such tool. From the Fig. 1.3, it is evident that such tools are not very useful in detecting the changes since they work only when the two drawings are perfectly aligned which generally need not be the case. This thesis presents an automated algorithm to compare two engineering drawings. The goal is to propose an algorithm the will be robust to alignment differences between the two engineering drawings.

It is to be noted that any change in the dimension is reflected in the corresponding label marked in the drawing. The labels are nothing but the textual content present in the drawing. Hence, comparison of two drawings can be done by comparing the labels of the two drawings. The first step here is to separate the graphics and the textual part in the drawing. Most of the text extraction algorithms proposed in the literature are based on the texture and color based features. These algorithms fail in this case as we deal with only binary images. Some algorithms which are used for text extraction in documents use the knowledge of the layout of the document, but in this case no such information is available. Fletcher and Kasturi [2] proposed an algorithm for text string separation from mixed text/graphics images. It is based on generation of connected components and application of Hough Transform to group together the components in logical character strings. Lai and Kasturi [3] proposed a system for detecting dimensioning sets in engineering drawings. Even this method is based on connected component generation and later composing them into strings which are associated with dimensioning lines. Lu [4] presented a rule-based method for text/graphics separation based on features of text and graphics in engineering drawings. In this thesis, we propose a method based on connected components generation similar to the method mentioned in [2].

Each drawing will have multiple sub drawings corresponding to various views of the object. Before comparing, the sub-drawings in the two drawings have to be associated with each other. Since we deal with binary images, texture or intensity information is not available; it is the shape of the drawings that can be exploited in order to achieve the matching. We propose a method in which matching is achieved by using SIFT-features [5]. Since SIFT fails on binary images, the engineering drawings are blurred before the SIFT features are obtained.

Once the sub-drawing associations are known, the labels of two corresponding sub-drawings are compared with each other to find the difference between the two drawings. Every label of a sub-drawing is matched with every label in the corresponding sub-drawing of the other drawing. If a label does not have any match, then it is reported as a change. Initially, the number of characters in the label, Euler number [6] and dimensional similarity are used to reduce the search space for matching. The labels are finally matched using a Hausdorff Distance [7] based measure. Flowchart for the proposed method is shown in the Fig. 1.4 and the detailed explanation is provided in the further chapters.

1.1 Organization of Thesis

Chapter 2 deals with text and graphics separation using connected component generation and characteristics of the textual information in engineering drawings to classify the components into text and graphics.

Chapter 3 explains a procedure followed to segment the given drawing into sub-drawings corresponding to various views of the component. It also explains how to obtain SIFT descriptors and use the same to match binary images.

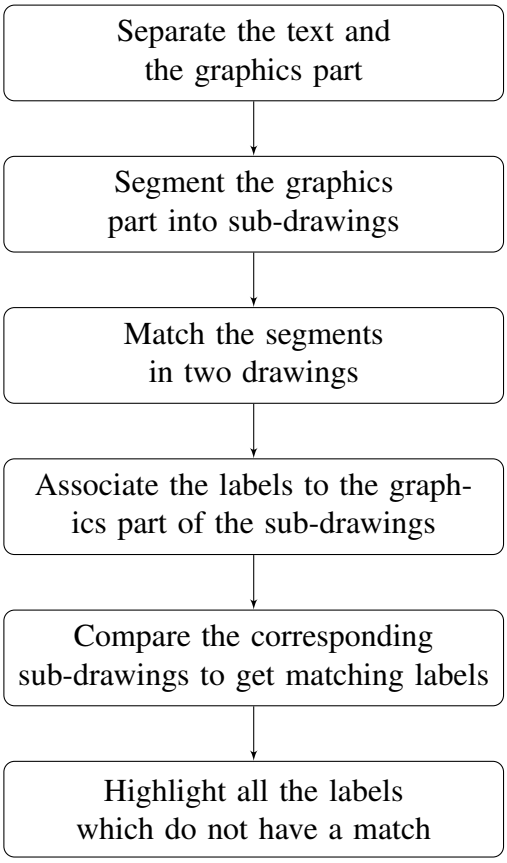


Figure 1.4: Flowchart of the proposed method.

Chapter 4 discusses about grouping the dimensioning information to the respective segments. It deals with grouping of text logically to form labels and then assigning the labels and the dimensioning lines to the corresponding segments based on distance transform [8].

Chapter 5 deals with the comparison of the labels of the corresponding sub drawings, thereby performing the comparison of engineering drawings. A series of parameters like number of characters in the label, Euler number and dimensions of the characters in the label reduce the space for searching the possible matching labels for each label. Finally, Hausdorff distance-based measures are used to compare the labels.

Chapter 6 presents some experimental results and discusses the issues involved.

Chapter 7 concludes the thesis.

CHAPTER 2

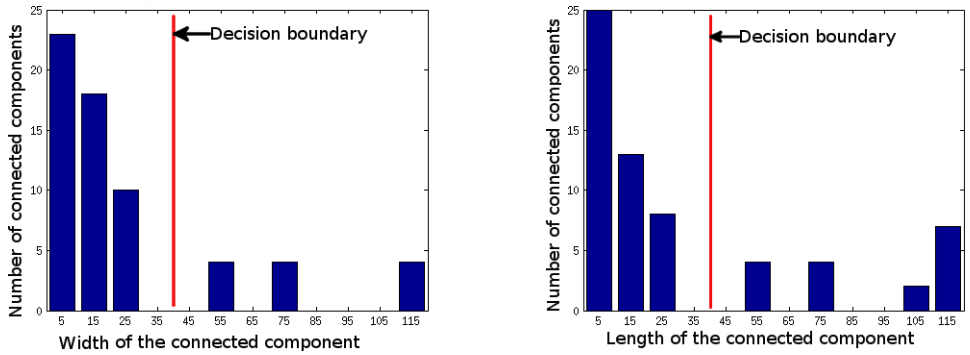
Text and Graphics

Any engineering drawing will consist of a lot of dimensioning information related to the drawing in the form of labels and the corresponding labeling lines. Fig. 1.1 is a typical engineering drawing shown at a scale of 0.15. In order to efficiently interpret such drawings, it is necessary to separate the text and the graphics. This chapter explains the technique to separate the dimensioning information from the given drawing.

2.1 Text Extraction

Text extraction is a crucial step in separating the text and graphics as most of the dimensioning information present in the drawing is textual. Once this textual part is removed from the drawing, only the labeling lines have to be eliminated to obtain the graphics part. Connected component based algorithm is used for extracting the text. Often the size of characters is much smaller compared to that of graphics. This property can be used as a classifier to distinguish the text and graphics.

Firstly, the connected components are to be generated. They are obtained by grouping all the 8-connected black pixels against the white background. These connected components are analyzed to know the characteristics of the text in the drawing. From the histogram of length and width of connected components, the



(a) (b)

Figure 2.1: (a) Indicates the histogram of width of the connected components and, (b) indicates the histogram of length of the connected components.

average range in which the characters occur is evident. Figs. 2.1a and 2.1b indicate the histogram of width and length of the connected components extracted from the drawing shown in Fig. 1.1. From these histograms it is evident that all the characters have length and width less than 35 pixels. Based on this range, all the characters can be extracted from the drawing to obtain an image which contains only the text. Using only this constraint will group even dashed lines as characters, since the dashed lines generally are thin and are shorter than characters. Hence, an additional constraint such as length of the line if the the width is too small can be included accordingly to eliminate the dashed lines. Also, when the characters are logically grouped to form labels, if many characters with similar characters get grouped they can be eliminated as they will belong to dashed lines.

2.2 Elimination of Dimensioning Lines

After all the text is extracted from the drawing, all that remains is the graphics part and the labeling lines as shown in Fig. 2.3. In general, the thickness used to draw

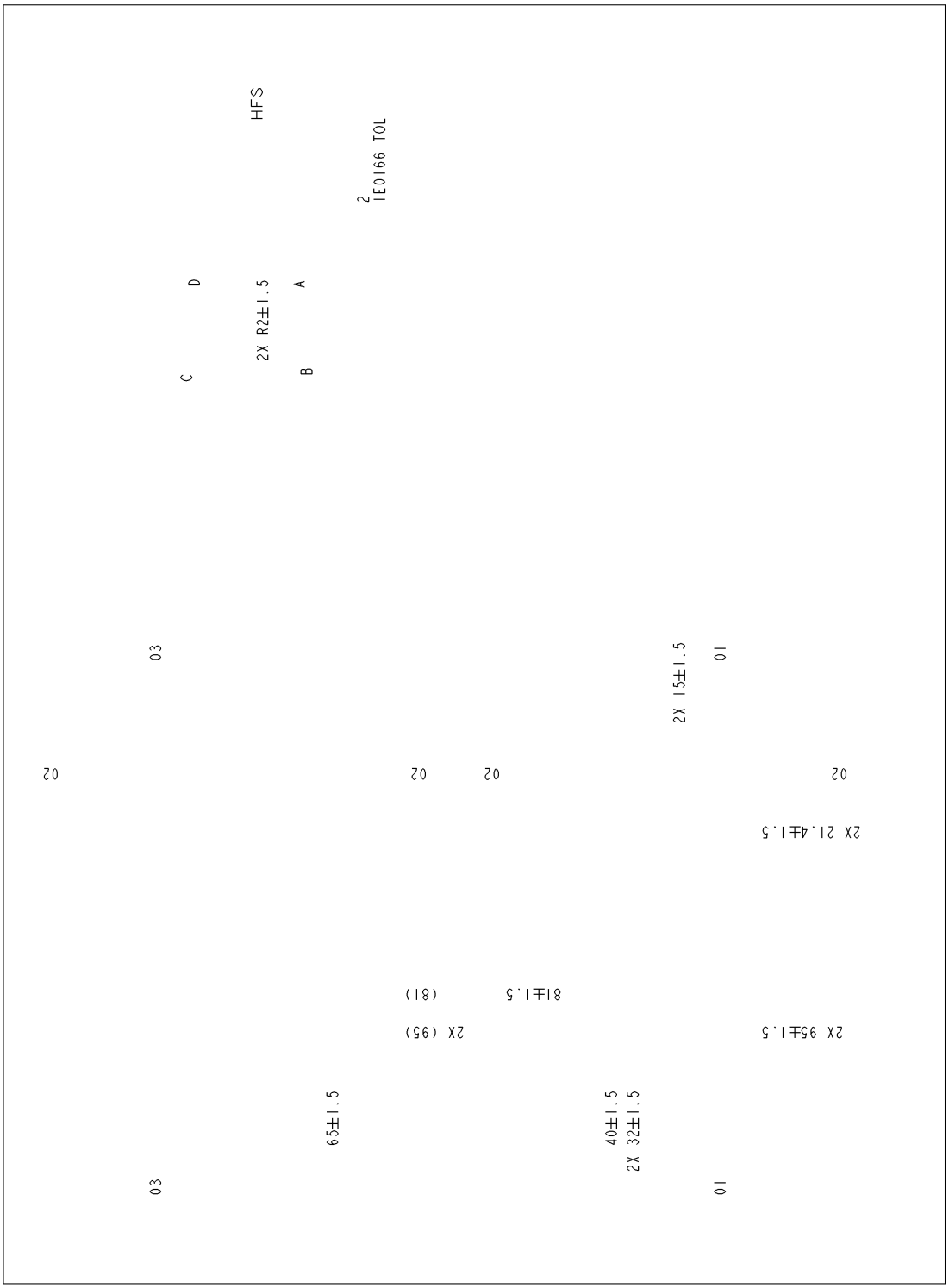


Figure 2.2: Result of text extraction on the drawing shown in Fig. 1.1.

labeling lines is less than that used for the drawing itself. Thus, morphological operations can be used to eliminate the labeling lines from the drawing.

2.2.1 Dilation and erosion

Dilation and erosion [9] are two basic morphological operations. These operations take two inputs; the first is the image to be operated upon and the second is the structuring element also known as a kernel.

Dilation gradually enlarges the boundary region of the foreground image. Mathematically, dilation is performed by laying the structuring element B on the image A and sliding it across in a manner similar to convolution. The steps involved include:

- If the origin of the structuring element coincides with a white pixel in the image then there is no change and we move on to the next pixel.
- If the origin of the structuring element coincides with a black pixel in the image, then we make all the pixels covered by the structuring element black.

Erosion is the dual of dilation. It gradually erodes away the boundary region of the foreground image. The process of erosion is similar to that of dilation, but the pixel turns to white instead of black. As before, the structuring element is made to slide across the image and the following steps are performed.

- If the origin of the structuring element coincides with a white pixel in the image, there is no change and we move on to the next pixel.
- If the origin of the structuring element coincides with a black pixel in the image, and atleast one of the black pixels in the structuring element falls

over a white pixel in the image, then we change the black pixel corresponding to the origin of the structuring element to white.

2.2.2 Thinning

Thinning is yet another morphological operation which is similar to erosion. Through thinning, binary regions can be reduced to their center lines also called skeletons [10]. This is a composition of morphological operations and works as follows:

- Perform erosion with the mask given below
$$\begin{pmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$
- Remove pixels such that it does not split the region. The following masks can be used for this purpose.
$$\begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$
- Remove pixels such that endpoints are retained using the following masks.
$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$
- The above steps are performed iteratively till there are no changes in the image.

In the previous section, the method to extract the text was mentioned. After removing the the textual part from the drawing, all that remains is the labeling

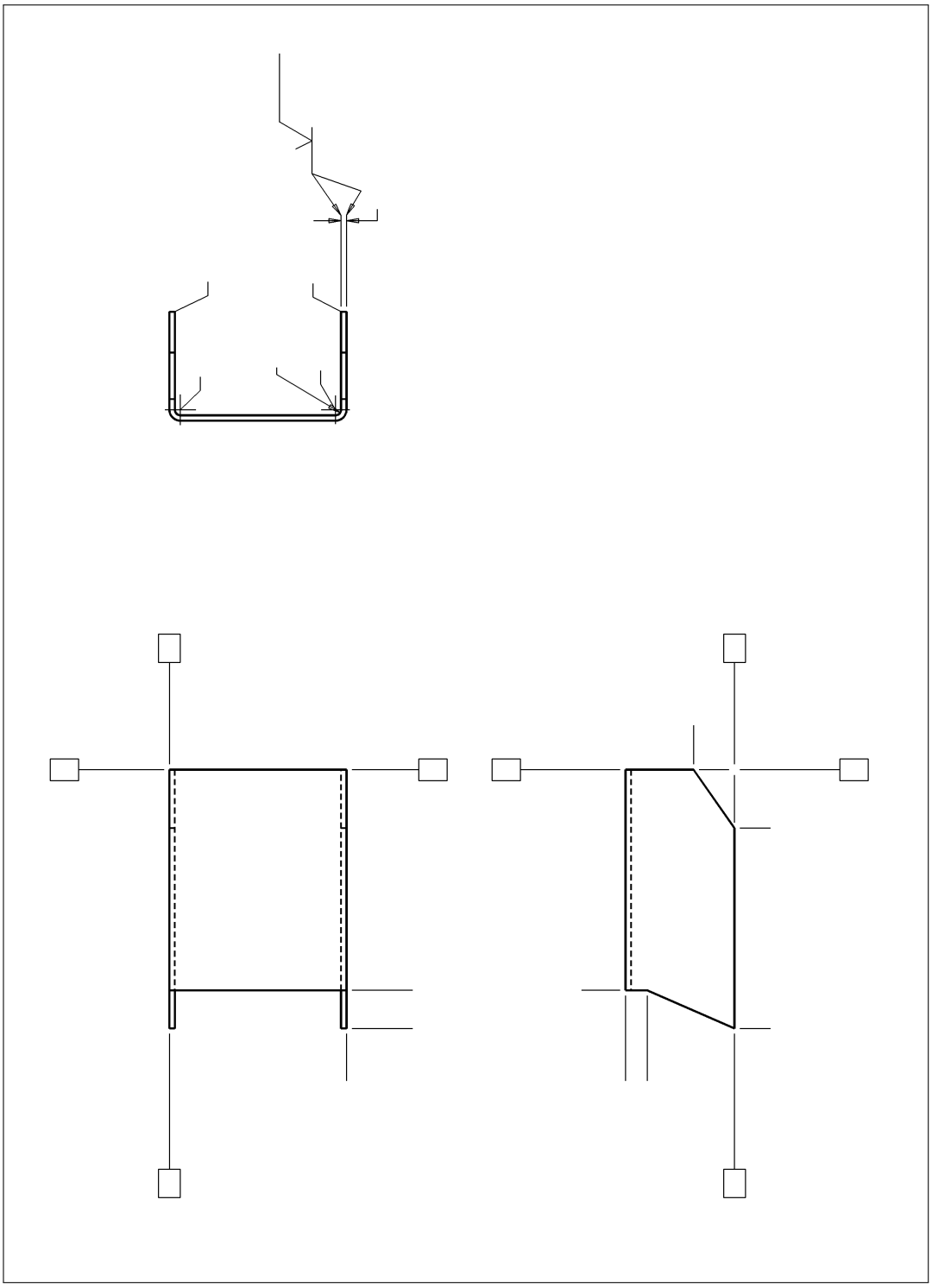


Figure 2.3: Image after the removal of textual information from the drawing shown in Fig. 1.1.

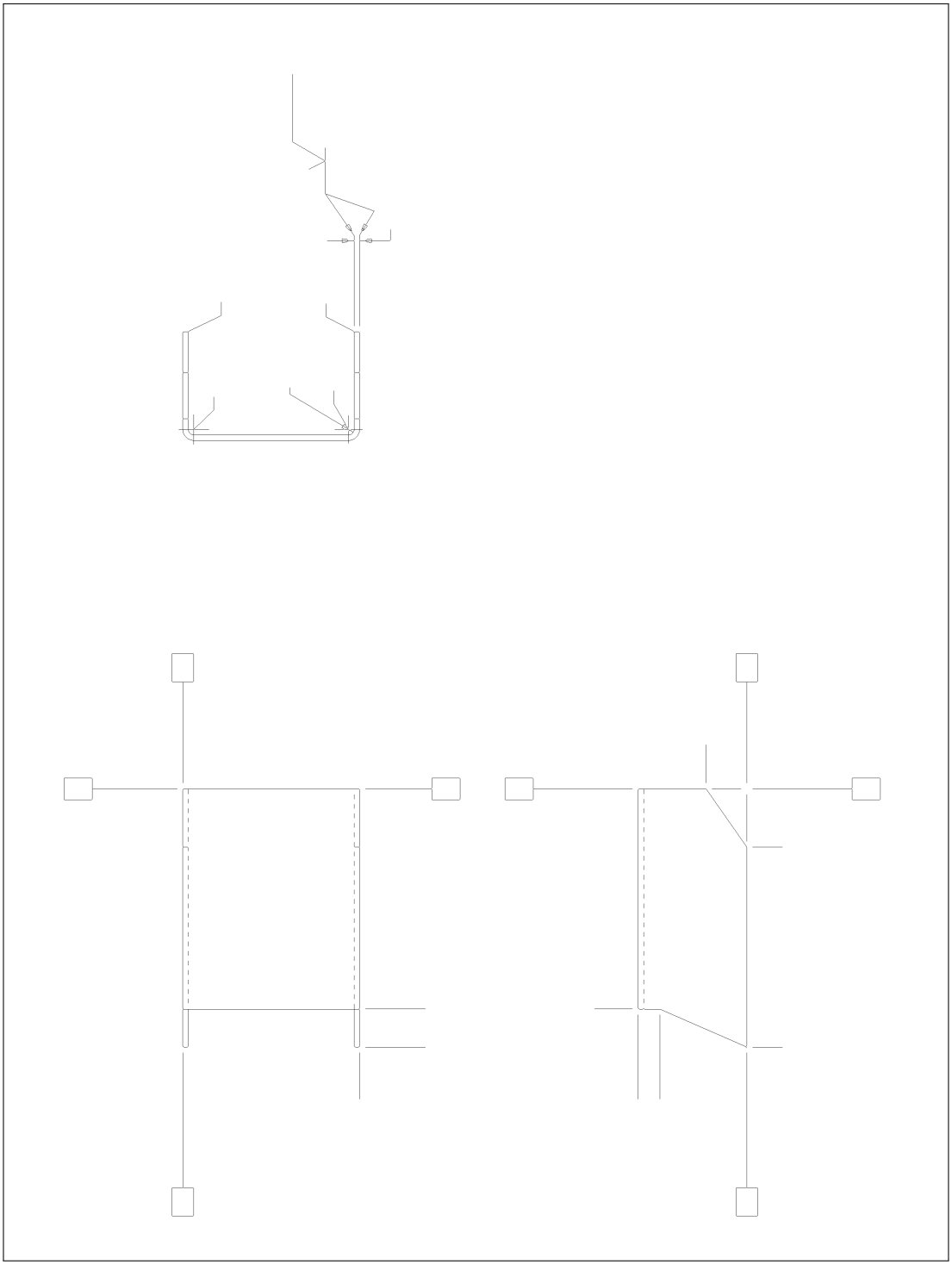


Figure 2.4: Effect of thinning on the drawing shown in Fig. 2.3.

lines and graphics. Fig. 2.3 shows the residue after removing the text content from the drawing shown in Fig. 1.1. From Fig. 2.3, it can be clearly seen that labeling lines are much thinner compared to the lines used for the drawing. We perform thinning operation on the residue image shown in Fig. 2.3 to get the skeleton of the image in Fig. 2.4. The lines in the drawing can be thinned down even by eroding the residue image shown in Fig. 2.3. When a small structuring element is used for erosion, only the label lines will become single pixel wide. Hence, on subtracting the skeleton image from this, label lines can be completely eliminated and the resulting image can be dilated to get better results. Fig. 2.5 shows the graphics part of the drawing shown in Fig. 1.1. Due to image subtraction, the resulting image will not be clean but this can be taken care of during the process of segmentation.

This algorithm fails if the labeling lines and the graphics part of the drawing are both of the same thickness. When the graphics part is thin, the algorithm classifies it as labeling line and eliminates it completely and when the labeling lines are thick the algorithm classifies them into graphics part. Though the second case is acceptable, the first case is not desirable. Hence, for good results the graphics part of the drawing should be thick compared to the labeling lines.

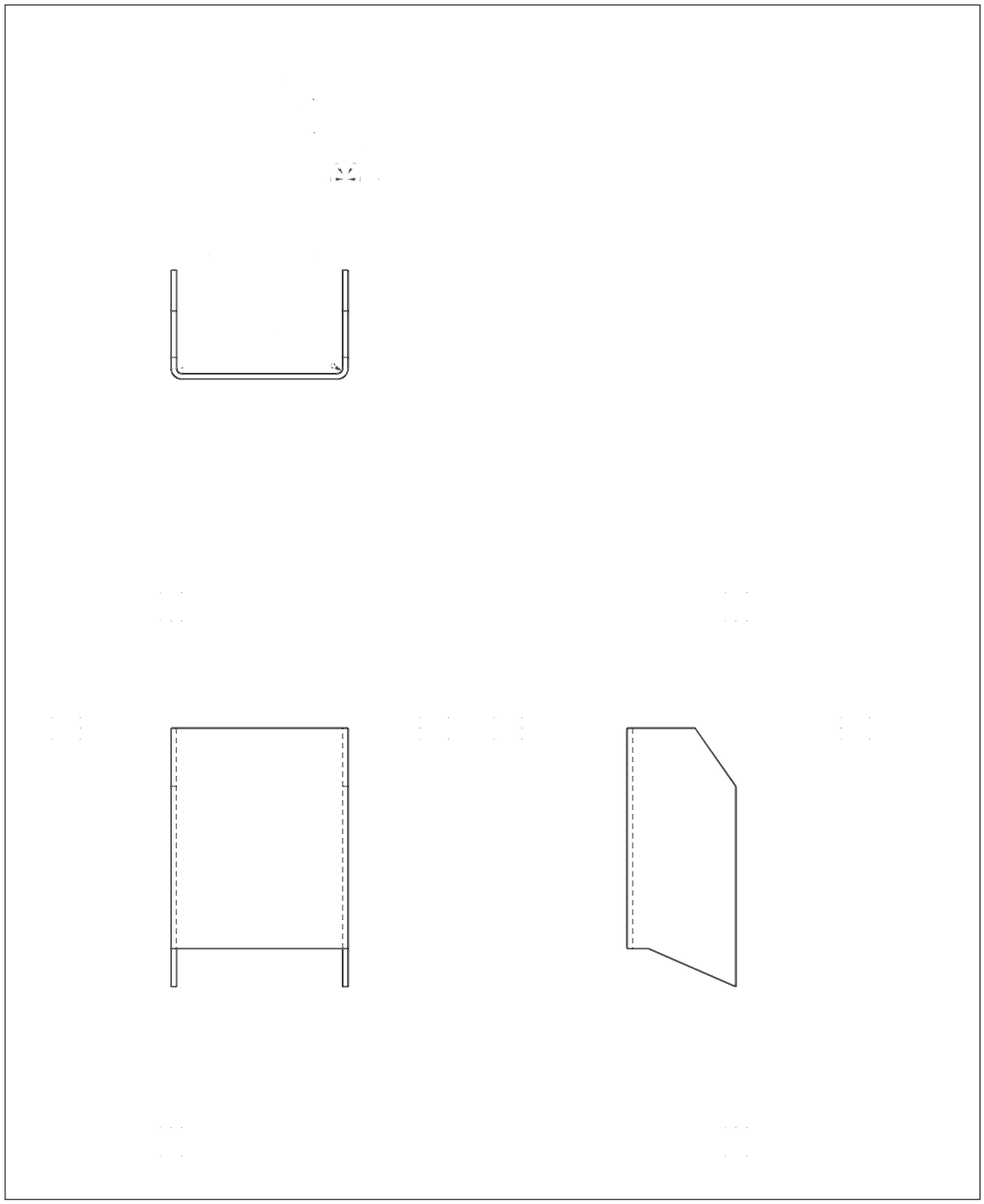


Figure 2.5: Graphics part of the drawing shown in Fig. 1.1.

CHAPTER 3

Segmentation and Matching

3.1 Segmentation

In an engineering drawing, there will generally be several sub drawings. These can be either drawings of various views of the same part or can be of different parts or a combination of both. In order to analyze the engineering drawing, segmentation is an essential step.

Consider the three sub-drawings in the engineering drawing shown in Fig. 1.1, the outer boundary of the graphics part of each of the sub drawings is a closed contour. This is valid for all the sub drawings in any engineering drawing. Thus all the outermost contours should be extracted to get all the sub drawings. To begin with, we extract all the connected components. This includes all the contours, even the ones which are inside the outermost contour. Outermost contours are always bigger than the inner ones. Choosing the contour with the largest number of pixels will give the outermost contour of the largest sub-drawing. The components within this outermost contour can be grouped together to get the first sub-drawing. To get the components within the contour, binary morphological operation called filling [11] can be used. The same procedure can be followed multiple times to extract all the sub-drawings.

Engineering drawings also contain section labels and symbols which are part of labels of the sub-drawings and which might be present outside the contour. These should not be classified as a sub-drawing. Hence, a threshold on the number

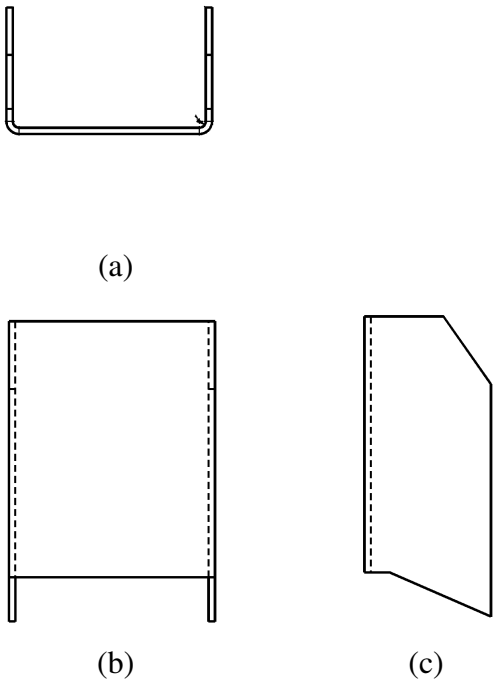


Figure 3.1: (a), (b) and (c) shows the graphics part of the three sub drawings extracted from the drawing shown in Fig. 1.1.

of pixels in the contour can be used to eliminate such outliers. During this process, even the noise which was present in the graphics part of the drawing will get eliminated. The result of segmentation of the graphics part in Fig. 1.1 is shown in Fig. 3.1.

3.2 SIFT

SIFT (Scale Invariant Feature Transform) is an image descriptor for image-based matching developed by David Lowe [5]. The SIFT descriptor is invariant to trans-

lations, rotations and scaling in the image domain and robust to moderate perspective transformation and illumination variations. It has been very useful in the field of object recognition, image stitching, gesture recognition, video tracking and image matching, under real world conditions. The algorithm by David Lowe [5] was designed for gray scale images, and it has been extended to color images by Bosch and Zisserman [12]. Other related image feature descriptors which are inspired from SIFT are Histogram of Gradients (HoG) [13], Gradient Location and Orientation Histogram (GLOH) [14] and Speeded Up Robust Features (SURF) [15].

In this thesis we discuss only the SIFT-descriptor in some detail. The first stage identifies the key locations in the scale space by looking for locations that are extrema of a Difference of Gaussian (DoG) function [16]. Each point is then used to generate a feature vector that describes the local image region. The detailed algorithm for obtaining the SIFT descriptor is explained below:

3.2.1 Detection of scale-space extrema

This is the first step of the algorithm. It is efficiently implemented by using Difference of Gaussians [16, 17] to identify potential interest points that are invariant to scale and orientation.

Octaves and Scales

Several octaves of the image are generated. Each octave's image size is half of the previous one. The number of octaves and scales depend on the size of the original image. It is to be noted that in David Lowe's algorithm [5], the number of octaves is restricted to 3.

Blurring

Blurring is nothing but convolution with a Gaussian kernel,i.e.,

$$\mathbf{L}(x, y, \sigma) = \mathbf{G}(x, y, \sigma) * \mathbf{I}(x, y) \tag{3.1}$$

where,

- **L** is the blurred image
- **G** is the Gaussian kernel
- **I** is the input image
- σ is the scale factor which decides the amount of blur.

Difference of Gaussians (DoG)

Two consecutive images in the octave are picked up and one is subtracted from the other. Then the next consecutive pair is taken and the process repeats. This is done for all octaves. This is a scale invariant version of Laplacian of Gaussians. The results are minima and maxima which are very good key features. Fig. 3.2 shows a pictorial representation of DoG.

3.2.2 Keypoint Localization

Locate extrema in DoG images

The first step here is to coarsely locate the extrema. This is a simple procedure. At every pixel, check for all the neighboring pixels and check if that location was

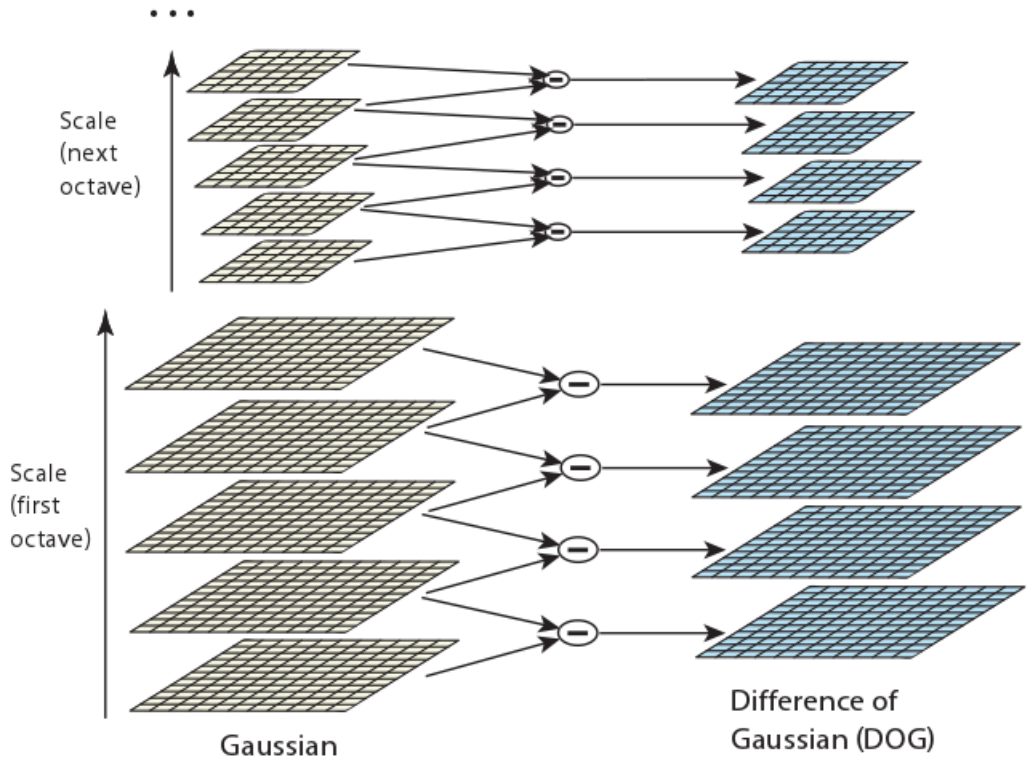


Figure 3.2: For each octave scale, the initial image is convolved with Gaussian to produce a set of scale space shown on the left. Adjacent Gausssians are subtracted to produce the Difference of Gaussians shown on right. Image source: [1].

an extrema or not. $\mathbf{X}$ is marked as a keypoint if it is greatest or smallest of all its 26 neighbors in the current scale, the scale above and below as shown in Fig. 3.3.

Locating sub-pixel extrema

The extrema points obtained previously are only approximate locations. The actual extrema points will generally be between pixel locations. Hence, mathematically the sub-pixel extrema points should be obtained.

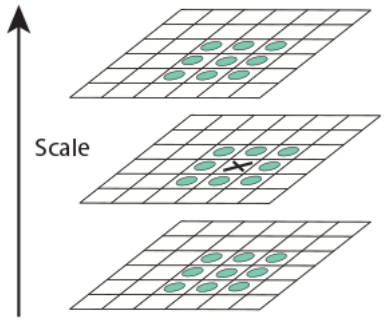


Figure 3.3: The points around the $\mathbf{X}$ mark in the current and adjacent 3×3 regions, indicates the 26 neighbors amongst which the extrema points are found.
Image source: [1].

The Taylor series expansion of the scale-space function, D(x,y,σ)^T, shifted so that the origin is at a sample pixel can be expanded as below:

$$D(\mathbf{x}) = D + \frac{\partial D^T}{\partial \mathbf{x}}\mathbf{x} + \frac{1}{2}\mathbf{x}^T\frac{\partial^2 D}{\partial \mathbf{x}^2}\mathbf{x} \tag{3.2}$$

where D and its derivatives are evaluated at the sample pixel and $\mathbf{x} = (x, y, \sigma)^T$ is the offset from this point. The location of the extremum $\hat{\mathbf{x}}$ is obtained by taking the derivative of the above equation with respect to $\mathbf{x}$ and equating it to zero, which gives

$$\hat{\mathbf{x}} = -\frac{\partial^2 D^{-1}}{\partial \mathbf{x}^2}\frac{\partial D}{\partial \mathbf{x}} \tag{3.3}$$

These sub-pixel locations increase the chances of matching and stability. The same procedure is followed over all octaves.

3.2.3 Removing edges and low contrast regions

The keypoints obtained from the previous step are too many. Some of them lie along the edge and some of them do not have enough contrast. These are are not very useful and hence must be eliminated.

Eliminating low contrast features

Low contrast feature is equivalent to low intensity pixel in a DoG image. If the magnitude of intensity at the extrema points is lower than a certain threshold, then they are discarded.

Removing edges

At every keypoint, two gradients are calculated, both being perpendicular to each other. If the keypoint is in a flat region, then both the gradients are small; else if it is in an edge region one of the gradients will be small and the other will be large. Only if the keypoint lies in the corner region will both the gradients be high; only such points are to be extracted [18]. This can be mathematically achieved by analyzing the Hessian matrix in a manner similar to Harris corner detection [19]. Here, ratio of the two eigenvalues are used instead of gradients to determine whether the point is a corner or not.

3.2.4 Assigning keypoint orientation

In this step, orientation is assigned to each keypoint. This orientation ensures rotation invariance. In order to assign the orientation, gradient magnitudes and direction are collected around each key point. The most prominent direction is

later assigned as the orientation for that keypoint. For each Gaussian smoothed image $\mathbf{L}$, gradient magnitudes and directions are calculated as,

$$m(x,y) = \sqrt{(\mathbf{L}(x+1,y) - \mathbf{L}(x-1,y))^2 + (\mathbf{L}(x,y+1) - \mathbf{L}(x,y-1))^2} \tag{3.4}$$

$$\theta(x,y) = \tan^{-1}\left[\frac{\mathbf{L}(x,y+1) - \mathbf{L}(x,y-1)}{\mathbf{L}(x+1,y) - \mathbf{L}(x-1,y)}\right] \tag{3.5}$$

where $m(x,y)$ is the gradient magnitude and $\theta(x,y)$ is the direction at any keypoint (x,y).

The orientation histogram is formed from the gradients of sample points around each keypoint. This histogram contains 30 bins covering 360 degrees range. Each sample added to the histogram is weighted according to its gradient magnitude and the Gaussian-weighted circular window with a σ that is 1.5 times that of the scale of the keypoint.

The highest peak in the histogram is detected and any other local peak with magnitude atleast 80% of the highest peak is used to create another keypoint with that orientation.

3.2.5 SIFT-Descriptor

Around every keypoint a 16×16 window is considered, each of which is divided into 16 windows of 4×4 each. Fig. 3.4 shows a case where 8×8 window is considered. Within each 4×4 window, gradient magnitudes and orientations are calculated. These orientations are put into 8-bin histogram. Unlike the previous histogram, the amount added here depends on the distance from the keypoint. This can be done using a Gaussian weighing function.

Thus, for every keypoint there will be 16 histograms, each with 8 bins, thus

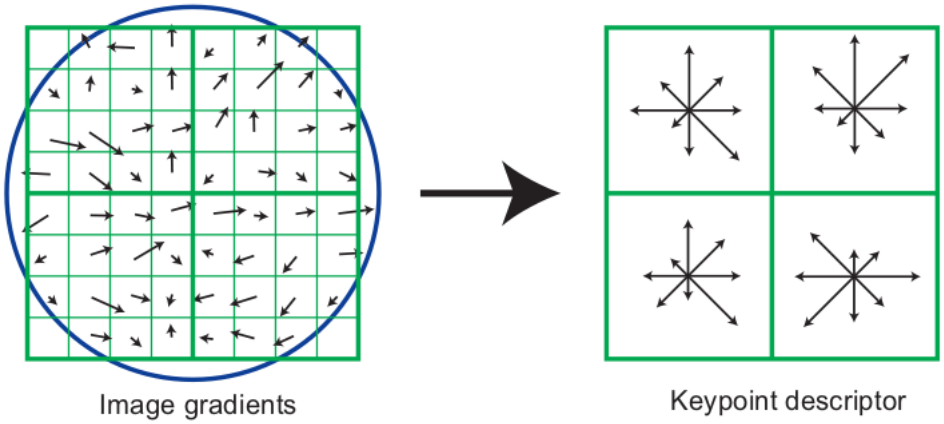


Figure 3.4: 2×2 Descriptor obtained from a 8×8 window around a keypoint. Image source: [1].

giving a 128-dimensional descriptor.

3.3 Matching Segments

In section 3.1, segmentation of an engineering drawing was explained. Given two such drawings, the sub-drawings in one has to be matched with the other. In order to achieve this, the matching points across the two drawings have to be determined. A SIFT-based matching algorithm [5] is adopted to do the same.

3.3.1 Finding matched keypoints

This is a very simple algorithm. First the feature vectors for both the engineering drawings are obtained. For every descriptor in the first image, its Euclidean distance from every other descriptor in the second image is calculated. A match is accepted only if its distance from the descriptor is less than certain fraction of

the distance to the second closest match. Thus a match is accepted when the ratio of its distance to the distance with the second closest match is less than a certain threshold. David Lowe suggested a threshold of 0.6, which gives a fair amount of matching keypoints. This is iterated for every descriptor of the first image.

For efficiency in Matlab, dot products between the descriptors is computed instead of finding the Euclidean distances. Ratio of the angle between the descriptors is close to the ratio of the Euclidean distances for smaller angles. The angle between the descriptors is calculated by taking the inverse cosine of the dot product between the descriptors.

3.3.2 SIFT for binary images

Since engineering drawings are all binary images, the gradient information in these images is not good enough for the SIFT algorithm to work. To get good descriptors for these drawings, they have to be converted into gray scale images so that there is considerable amount of gradient information. We apply Gaussian blur to these drawings and the intensities are scaled accordingly to get gray scale images. Experimentally, a Gaussian kernel with standard deviation 0.3 gives the best results.

To match the corresponding segments of one drawing with the other, only the graphics part of the drawing has to be considered. Hence the text and graphics are separated out from the engineering drawings shown in the Figs. 1.1 and 1.2. The graphics part of the drawings are blurred as mentioned previously. These drawings are generally very large in size, of the order of 8000×6000. It is computationally very expensive to find descriptors for such large images. Hence, the blurred drawings are scaled down by a factor of 4. The SIFT feature descriptors

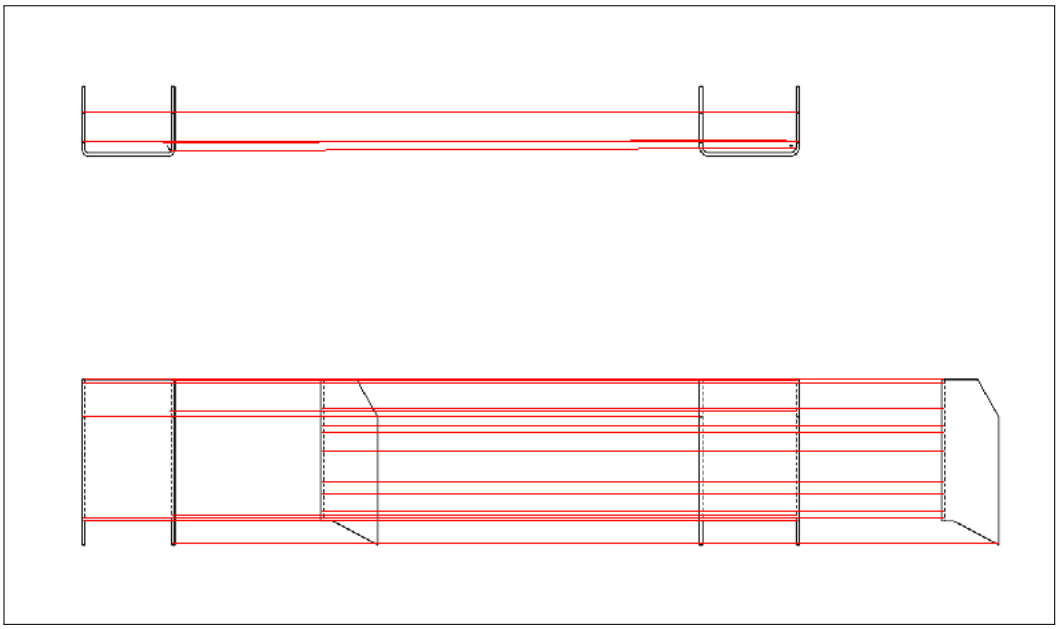


Figure 3.5: Result of SIFT on the graphics part of the drawings shown in Figs. 1.1 and 1.2.

are now obtained for these scaled and blurred drawings. Fig. 3.5 shows the result of finding the match between the graphics part of the drawings shown in Figs. 1.1 and 1.2.

3.3.3 Finding the match matrix

Once the SIFT-descriptors are available for an image pair, we need some measure across the segments of two drawings to find an appropriate match for each of the segments. Hence, we construct a match matrix which gives a measure of how much each segment matches with the other. For every segment of the first drawing, the number of matched keypoints with each of the other segment of the second drawing is calculated. Thus, every sub-drawing of the first drawing will have a vector of numbers indicating the number of matched keypoints with every

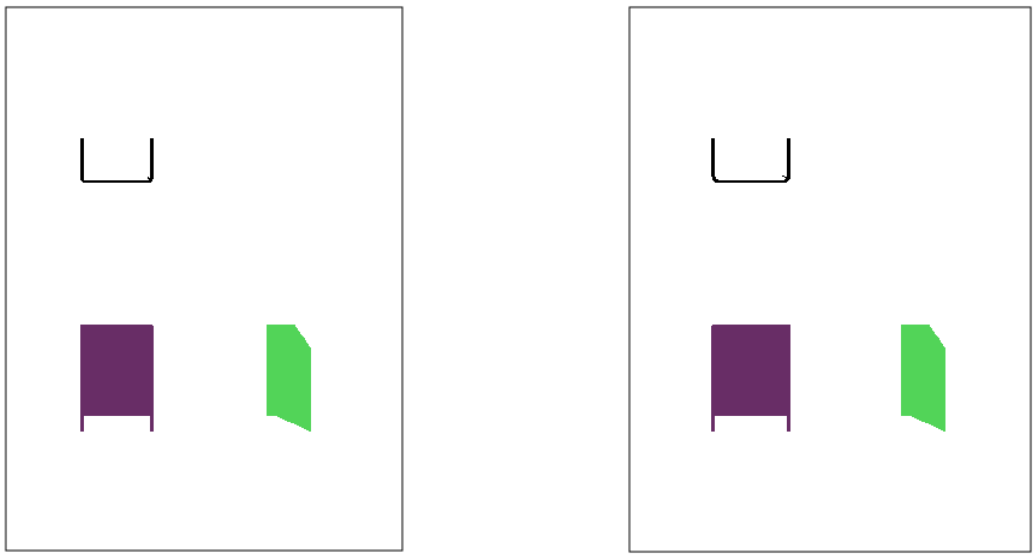


Figure 3.6: The regions in the two drawings corresponding to the same colour indicate the matched segments.

sub-drawing of the second drawing. All these vectors are stacked to get the Match matrix.

3.3.4 Finding the matched segment

If the number of segments in the first drawing is $\mathbf{N1}$ and the number of segments in the second is $\mathbf{N2}$, then the size of the Match matrix will be $\mathbf{N1} \times \mathbf{N2}$. From the maximum value in this matrix, we get the indices corresponding to the best matched segments. The row index corresponds to the segment in the first drawing and the column index corresponds to its matched segment in the second drawing. For every segment in a drawing there can be only one matched segment in the other. This implies that once the first matched pair is obtained, then those segments cannot be matched with other segments. Hence, the values in the row of the Match matrix corresponding to the row index and the column corresponding

to the column index are made zero and then the next maximum is found. This procedure is iterated till all the elements in the matrix become zero.

For the image pair shown in Figs. 1.1 and 1.2, the Match matrix was found to be $\begin{pmatrix} 8 & 0 & 0 \\ 0 & 10 & 2 \\ 0 & 0 & 9 \end{pmatrix}$. Since there are 3 segments in each of the drawing the match matrix is of the size 3×3. From this match matrix, the segments matched are shown in Fig. 3.6 at a scale of 0.45.

At the end of this procedure, the matched segment for every segment is known (if there are any). The rows and the columns which never got selected as matched indices correspond to the segments which do not have a match. This happens when a sub-drawing is either added or deleted from the drawing.

CHAPTER 4

Association of Labels with Segments

During segmentation only the graphics part of the engineering drawing was considered. Hence the segments obtained contain only the graphics part. The labels and the labeling lines have to be reassigned to their corresponding segments in order to obtain the complete sub-drawing.

4.1 Grouping Labeling Lines

Labeling lines are used to get the correspondence between the sub-drawing and its label. These lines can be either connected to the sub-drawing, or can just be close to the sub-drawing indicating that they belong to that sub drawing.

4.1.1 Obtaining segment masks

In the section 3.1, a method to obtain the segments was discussed. These segments always had a closed contour as their boundary. By using the morphological operation 'filling', a basic mask for the segments can be obtained. Since all the holes are filled during the process of filling, the region within the outermost contour also gets filled, thereby providing the mask corresponding to the segments. Fig. 4.1 shows the segment masks corresponding to the drawing shown in Fig. 1.1.

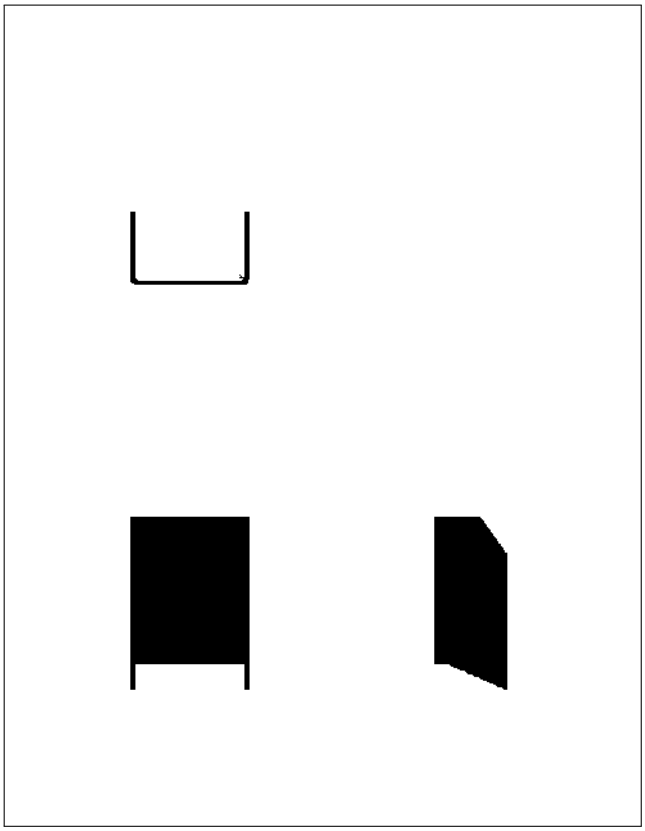


Figure 4.1: Segment mask corresponding to the drawing shown in Fig. 1.1.

4.1.2 Grouping labeling lines using dilation

One of the simplest methods to combine some of the labeling lines with the segment is by dilation. The mask obtained in the previous step can be dilated with a structuring element of size about 60×60 to get a dilated mask. When we remove both the text and the graphics part from the drawing, only the labeling lines remain. These can be combined with the dilated mask using the binary 'OR' operation to connect the labeling lines to the segments. As the masks were dilated, the labeling lines which were closer to the segments earlier will now get attached to the segments. Fig. 4.2 shows the dilated mask combined with the labeling lines from the drawing in Fig. 1.1.

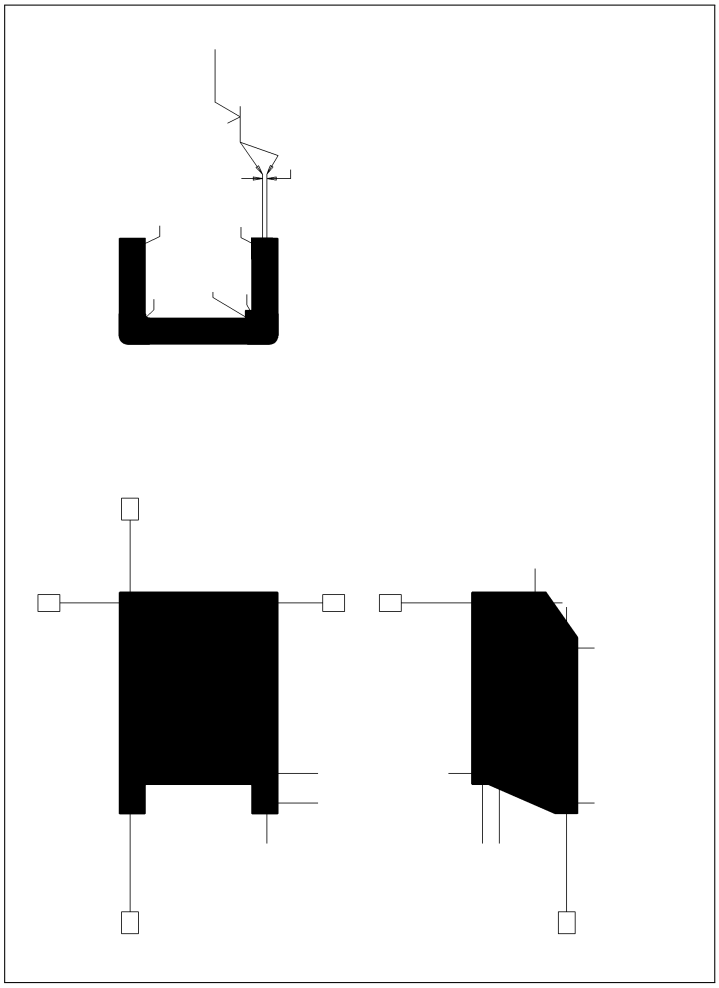


Figure 4.2: Dilated mask combined with labeling lines for the drawing shown in Fig. 1.1.

4.1.3 Grouping labels

The labels in the engineering drawings are a group of characters. In section, 2.1 the method to extract all the characters from the drawing was discussed. These extracted characters have to grouped logically to obtain the labels.

String extraction

The spacing between the characters of a label are considerably small. Hence, these can be grouped using the dilation method.

In order to combine the characters, all the characters are extracted from the drawing. Then a bounding box is formed for each of these characters. The region within these bounding boxes are filled using the binary morphological operation 'filling' to obtain the text mask. The text mask can be dilated with a small structuring element, so that the bounding boxes of the neighboring characters merge. A new set of connected components are extracted from the resulting image. All the characters in each of the connected components are grouped as one. This provides a group of characters which are close to each other. Now, a common bounding box is inserted into these group of characters and the boxes are filled to get a new mask which has all the closer characters grouped.

The grouped characters need not necessarily be a part of the same label. The previous procedure only results in grouping of strings which are close to each other. There can be cases where the two labels are in very close vicinity. In such cases, the characters of both the labels are grouped into one. Hence, the next step would be to split these groups so that only characters belonging to the same label are grouped.

In general, the height of the characters is more than its width. The orientation of the grouped characters are decided based on this. All the characters within a label will have same orientation and will belong to the same line. Hence, a group of characters in each of the connected component can be analyzed to split them accordingly into labels. The mean height and width of all the characters in a group is calculated. If the mean height is higher than the mean width, then they are hori-

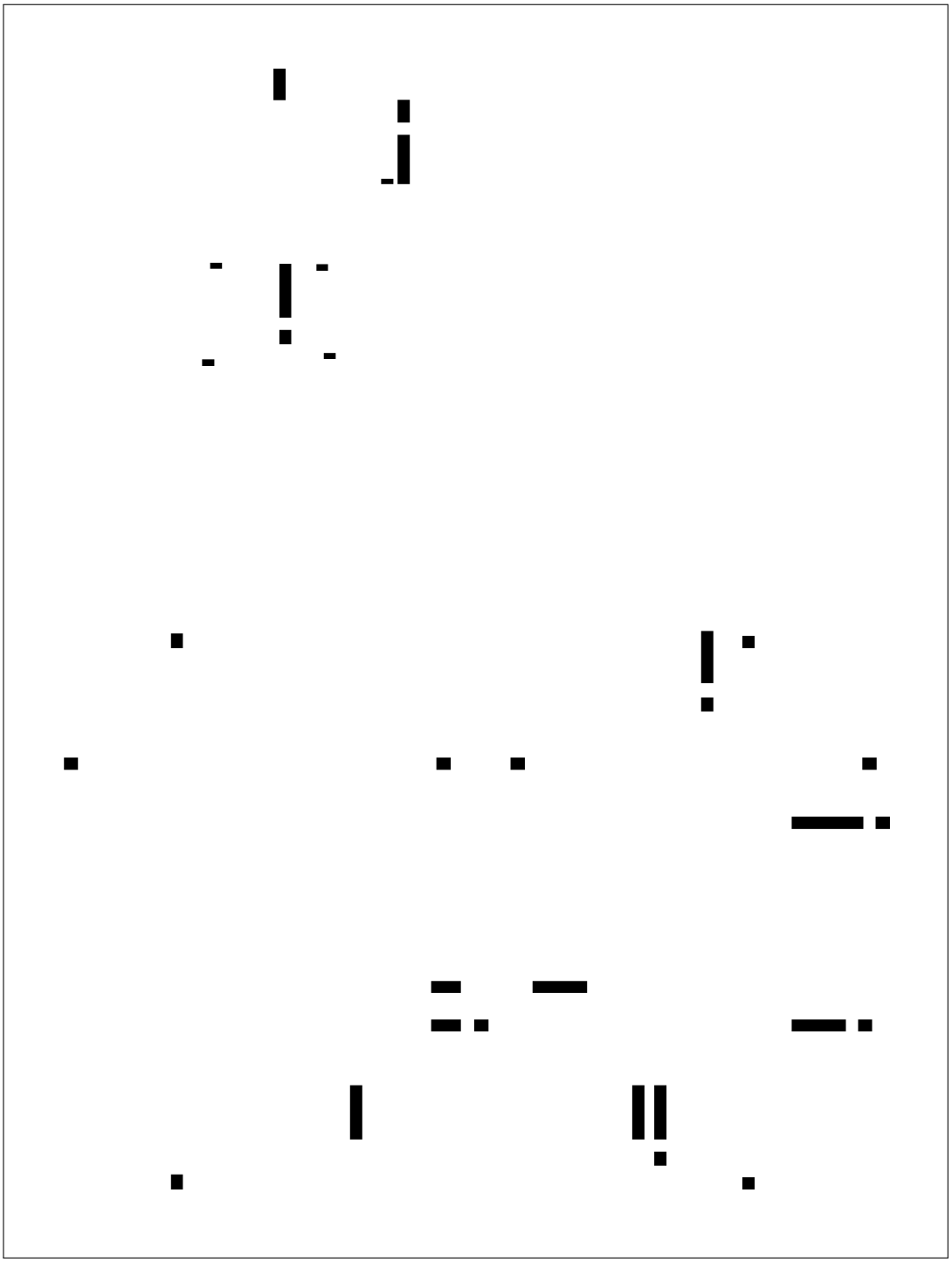


Figure 4.3: Label mask for the drawing shown in Fig. 1.1.

zontally oriented, else they are vertically oriented. Once the orientation is known, a similar grouping procedure is followed as mentioned in the previous step. The bounding boxes of the characters are dilated along the direction of the orientation. If the strings are horizontally oriented, bounding boxes are dilated horizontally, so that only the characters in that row are combined. Similarly if the characters are vertically oriented, bounding boxes are dilated vertically, so that only the characters in the same column are combined. Thus, the previously grouped characters are split according to their orientation and a new set of bounding boxes is inserted to the split set of groups. From these bounding boxes, the mask for the labels can be obtained. The label mask for the drawing shown in Fig. 1.1 is shown in Fig. 4.3.

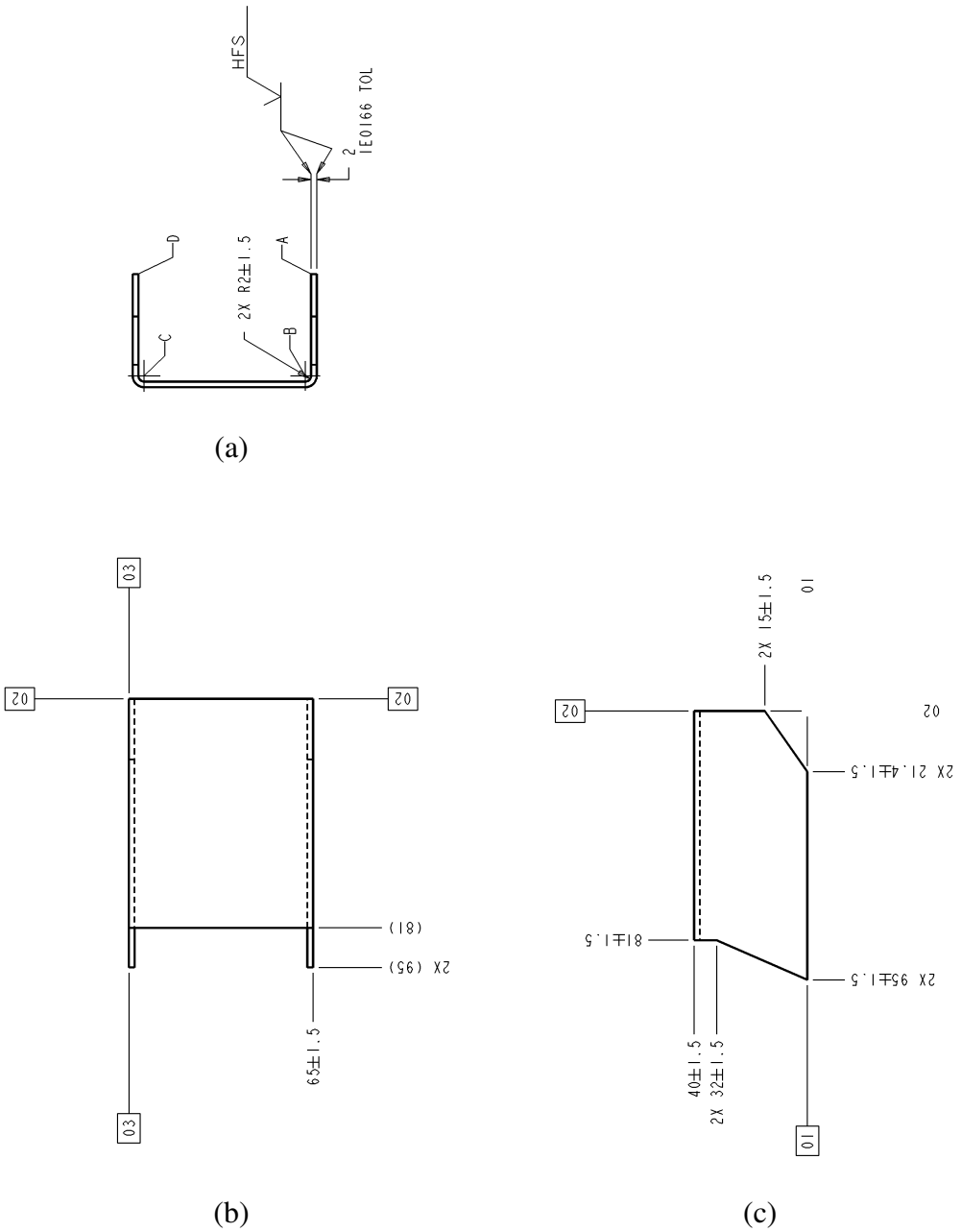
Assigning Labels to Segments

From the previous step, all the characters are grouped to form labels. These labels now have to be assigned to the corresponding segments. We use distance transform [20] to achieve the same.

Distance Transform

This operation transforms a binary image into a gray scale image. It gives a measure of distance from the foreground to the background. Farther the pixels are from boundaries, higher will be the value of their distance measure. Across the boundary, the value will be one and the value increases as we move away from the boundary. The standard distance measure used is the Euclidean distance.

Label masks are combined with the dilated masks using the binary 'OR' operation. We then take the distance transform of the resulting image. Based on the



(a)

(b)

(c)

Figure 4.4: (a), (b) and (c) shows the segments after the assignment of label for the segments shown in Fig. 3.1.

values of the distance transform in each of the label mask, the labels are assigned to the segments. For every label, the minimum value of distance transform is taken and the segment which is at that distance from the label is the one to which it belongs. The same procedure is iterated for all the labels. The results of assigning label to the segments in Fig. 3.1 are shown in Fig. 4.4.

Thus, using the methodology described above, dimensioning information related to a segment is grouped and all such segments are separated out and analyzed independently in the further course of the work.

CHAPTER 5

Image Comparison

When a change is made in any drawing, the change is reflected in its corresponding dimensions. These changes can hence be tracked by tracking the labels. Since we now have the segments and the corresponding labels, instead of comparing the drawings directly, we can compare the labels corresponding to the segments to get the difference between the two drawings. The changes in the drawings can also include addition or deletion of segments. These changes have to be detected before we compare the matched segments using the labels.

Finding addition/deletion of Segments

In section 3.3, we discussed a method to match the segments of two engineering drawings. The output of this algorithm is a vector V. The number corresponding to the index of vector V represents the matching segment corresponding to the index. If the number corresponding to the certain index is zero, then it implies that the segment does not have any matching segment in the second drawing. Thus any segment which has been deleted in the second drawing can be detected. Along similar lines, the indicies which are not present in the vector V are the segments in the second drawing which do not have any matching segment in the first. From this, any new segments which got added into the second drawing can be detected.

5.1 Comparison of Labels

The changes in the labels reflect the changes in the drawing. Thus comparison of labels helps in understanding the changes in the drawing. Since label to label association is not known, every label in each of the sub-drawing is compared with every other label in the corresponding sub-drawing of the other drawing. The comparison is done using Hausdorff distance [7]. The search space for comparison using the labels is reduced based on various parameters which are discussed further in this chapter. When a label does not get any matches, it is reported as a change.

Length

One of the simplest ways of finding the labels which do not have any match is by using the number of characters in the label. By using the method of connected components, the number of characters present in a label can be calculated. For every label of a sub drawing, all those labels of the corresponding sub-drawing which have the same length is made note of. This array of indices corresponds to the labels which match in length.

Euler number

Label is nothing but a series of characters. The properties of these characters can be made use of as another parameter to find the differences between the labels. Euler number [6] is one such property. Euler number is defined as the difference between the number of connected components and the total number of holes. Since we already have the list of labels which have already been matched accord-

ing to their lengths, we can also distinguish based on the number of holes in the label. Finding the number of holes in a character is very simple. Firstly, all the holes in the character can be filled using the binary morphological operation called 'filling'. From this, the original character can be subtracted to get the image which has only those regions where the holes were present. The number of connected components in the resultant image gives the number of holes in the character. For every label, only those labels which have the same number of holes in the corresponding characters are retained for further comparison, thus reducing the search space.

Dimensions of the characters

Generally all the labels in the drawing are written using the same font style and size. Hence, two labels can be same only if the corresponding characters in both the labels are same. Thus for every label, the search space is further reduced by considering only those labels whose characters do not vary much in their dimensions.

Hausdorff Distance

For two point sets A and B, the Hausdorff distance between them is defined as

$$H(A, B) = \max(h((A, B), h(B, A)) \tag{5.1}$$

where,

$$h(A, B) = \max_{a \in A} \min_{b \in B} \parallel a - b \parallel \tag{5.2}$$

and $\|.\|$ denotes a norm on the points of A and B [7]. The function $h(A, B)$ is

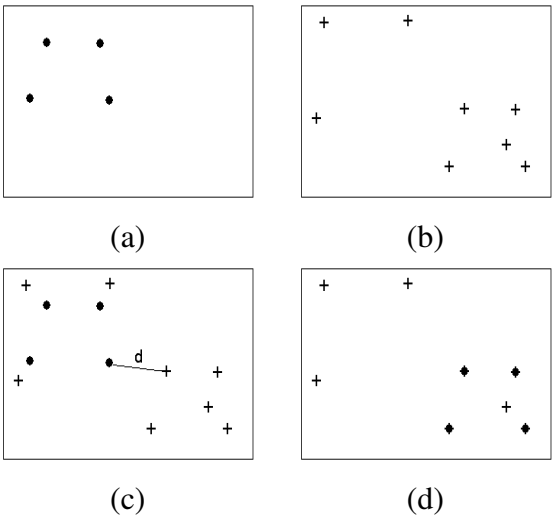


Figure 5.1: Point sets (a) A and (b) B. (c) Directed Hausdorff distance d. (d) Translation when distance is minimized.

called the directed Hausdorff distance from point set A to set B. It identifies the point $a \in A$ that is farthest from any point of B and measures the distance from a to its nearest neighbor in B. If $h(A, B) = d$, then each point of A will be within distance d of some point of B. In Figs. 5.1 (a) and (b) we see point sets denoting set A and B, respectively. The directed Hausdorff distance from set A to set B denoted by d is shown in Fig. 5.1 (c). When we translate the points of set A, as shown in Fig. 5.1 (d), the distance $h\,(A, B)$ is minimized. At this translation, all the points of set A coincide with some points of set B thereby minimizing the value of $h\,(A, B)$.

Matching labels

For the two labels to be same, all the characters in the label should be same. The Hausdorff distance between the corresponding characters is calculated. Two labels are said to be matched only if the all the Hausdorff distances between the

characters are less than a threshold. A threshold of 4 gives good results for the case of characters in engineering drawings. At the end of all comparisons, we know which labels are matched. Sometimes, a single label can have multiple labels which are matched to it. In such cases, the match is decided based on the location of the label. The centroids of all the matched labels are found and the label which is at a minimum distance from the centroid of the label is chosen as the match.

Finally, all those labels which do not have any matches are the ones which have changed. These labels are highlighted to indicate the difference between the two drawings. Figs. 5.2 and 5.3 show the final result of comparison of the drawings in Figs. 1.1 and 1.2.

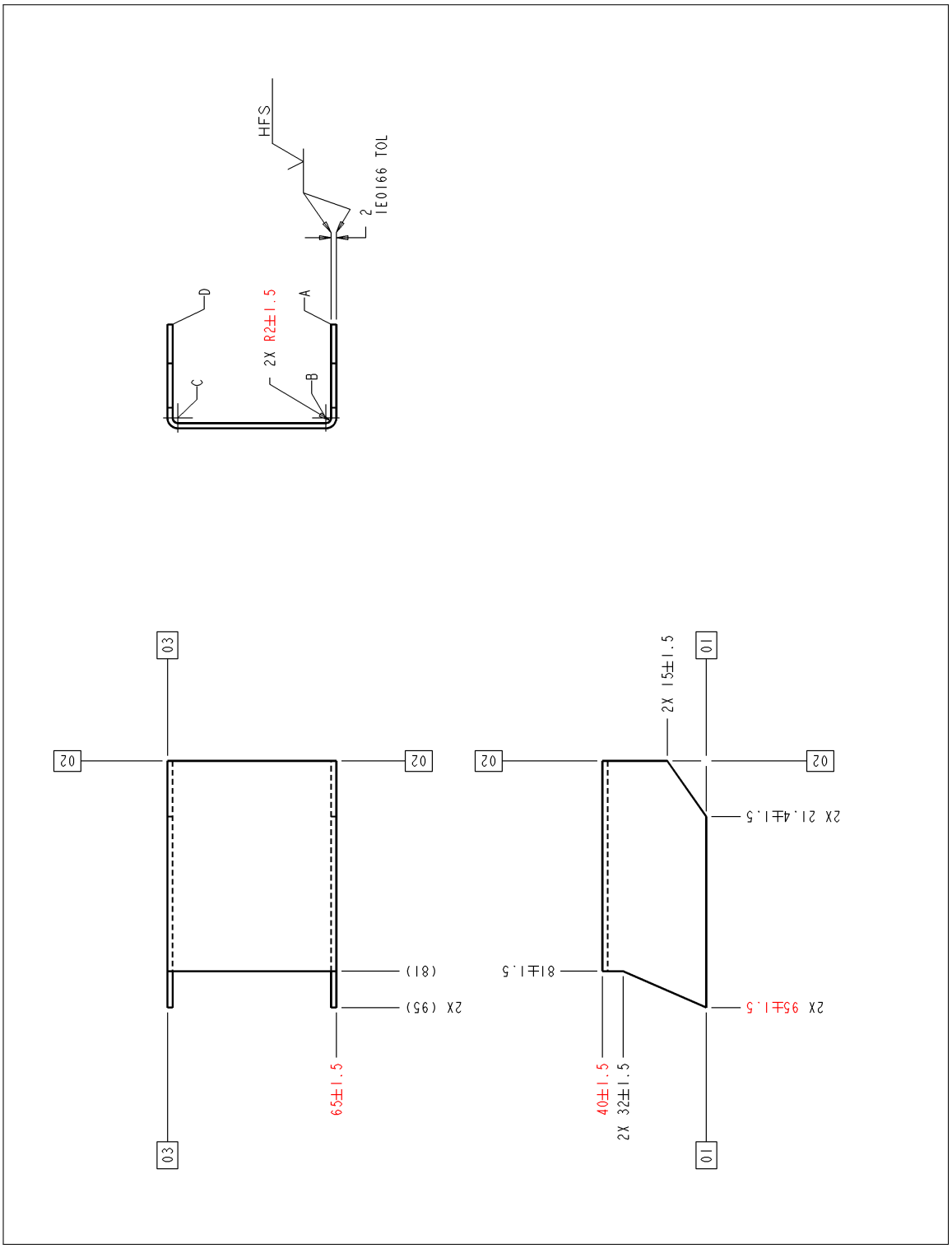


Figure 5.2: Result of comparison of the drawings in Figs. 1.1 and 1.2.

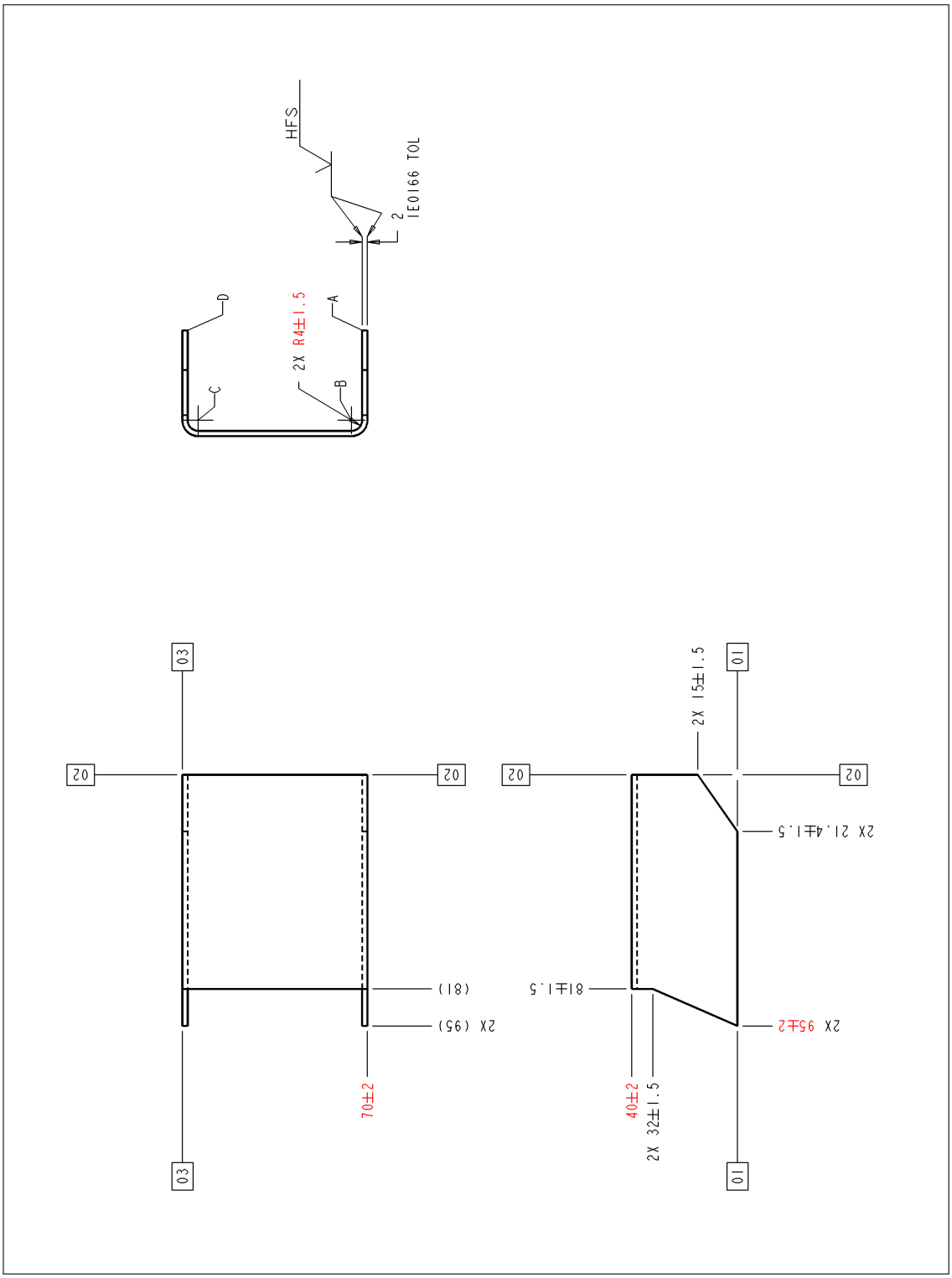


Figure 5.3: Result of comparison of the drawings in Figs. 1.1 and 1.2.

CHAPTER 6

Experimental Results

In addition to the example which was used in previous chapters to explain the method to compare engineering drawings, we considered 9 pairs of engineering drawings with different amounts of complexities and sizes provided by Caterpillar India Private Ltd to evaluate our method. Out of the 9 pairs considered, we illustrate segmentation, matching and comparison results for 2 pairs of example drawings shown in Figs. 6.1, 6.2, 6.3 and 6.4. We can see that example 2 (Figs. 6.1 and 6.2) is a simple engineering drawing pair whereas example 3(Figs. 6.3 and 6.4) shows a more complicated one. Also, the number of changes in example 3 are much more than the changes in example 2.

The result of segmentation and matching for these example drawings are shown in Figs. 6.5 and 6.6, wherein we observe that our method has successfully segmented and matched all the sub-drawings in both the examples. Final results of comparison for example 2 are shown in Figs. 6.7 and 6.8 and Figs. 6.9 and 6.10 show the results for example 3. The changes detected are marked in red. As the size of the labels are too small, a red dot also has been marked wherever there was a change. In example 2, only modifications of dimensions are present whereas in example 3, we can see that the changes include both modification of certain dimensions as well as deletion of certain regions. The results indicate that both kind of changes were detected by our method.

Through example 3, a possible case where a change is not detected is illustrated. Here, the dimension of the line which was 24 (marked with in green color)

in Fig. 6.3 was changed to 26 in Fig. 6.4 and the dimension of the arc which was 22 in Fig. 6.3 was changed to 24 (marked in green color) in Fig. 6.4. Our method has reported the changes about the dimensions 26 and 22 whereas dimension 24 was not recognized as a change because there was match with the same dimension in the modified drawing, though it belonged to a different part. This can be avoided by associating the labels with each other based on what specific parts they correspond to, before they are compared with each other.

We also illustrate the case of false detection through example 3. The labels marked in blue in Figs. 6.9 and 6.10 are reported as changes though they are exactly the same. This is because, the labels are grouped differently as the thickness of the characters are different in the two drawings. This can be avoided by making the thickness of all the characters uniform before grouping them to form labels.

The consolidated results for all the 9 pairs are drawings are tabulated below:

Drawing	1	2	3	4	5	6	7	8	9
No. of labels in the original drawing	25	28	119	112	28	21	15	82	101
No. of labels in the modified drawing	25	27	118	101	27	23	15	82	88
No.of changes Additions / deletions / modifications	0/0/4	2/0/2	1/0/1	0/10/14	1/0/7	0/13/2	0/0/4	1/2/4	0/16/3
No. of changes detected	4	4	2	23	8	15	4	5	18
No. of false detections	0	0	0	2	1	1	0	0	1

Table 6.1: Performance on 9 pairs of drawings from Caterpillar dataset.

From the results, it is clear that our method detected 95.4% of the changes with a false detection of 0.4%.

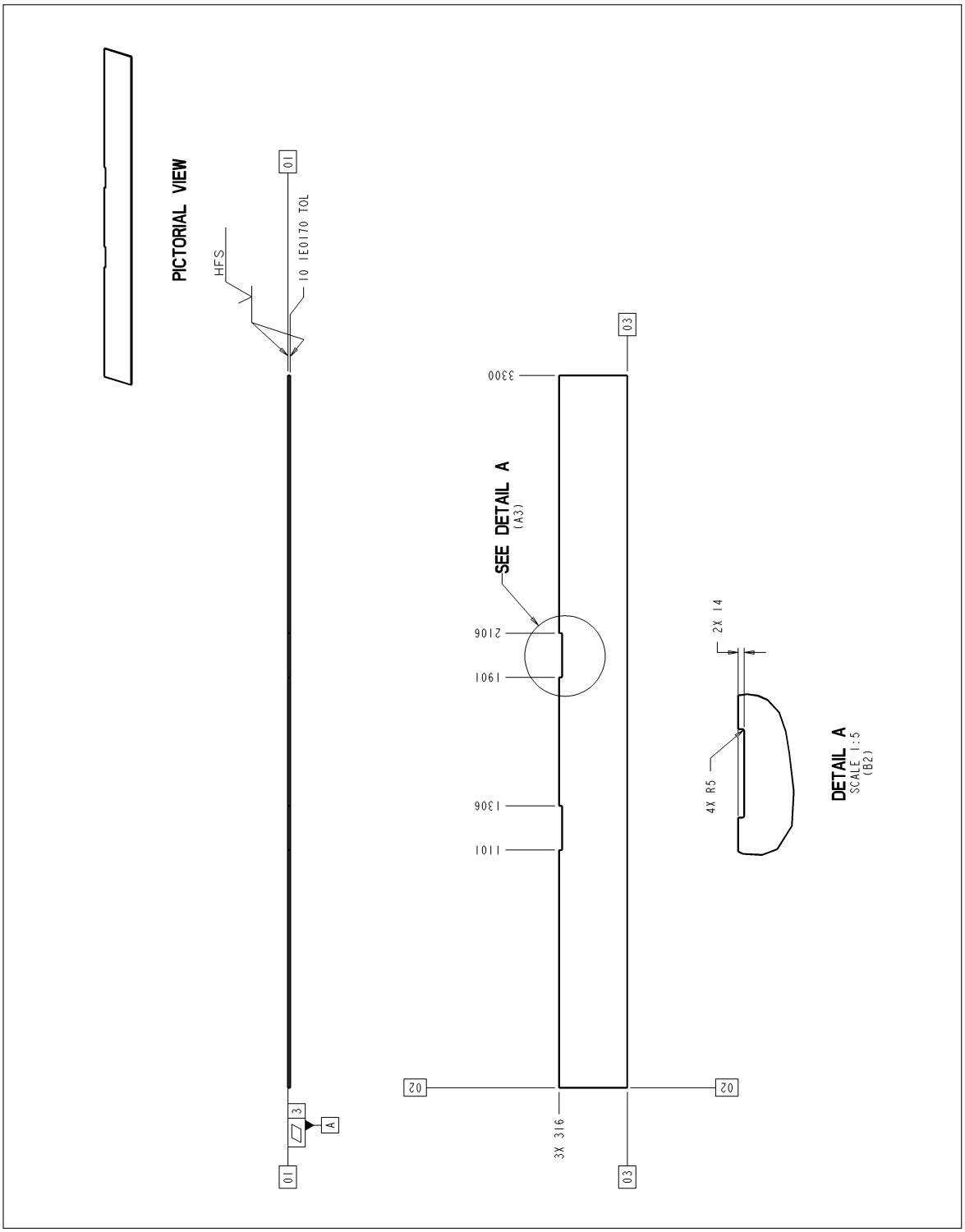


Figure 6.1: Original drawing of Example 2.

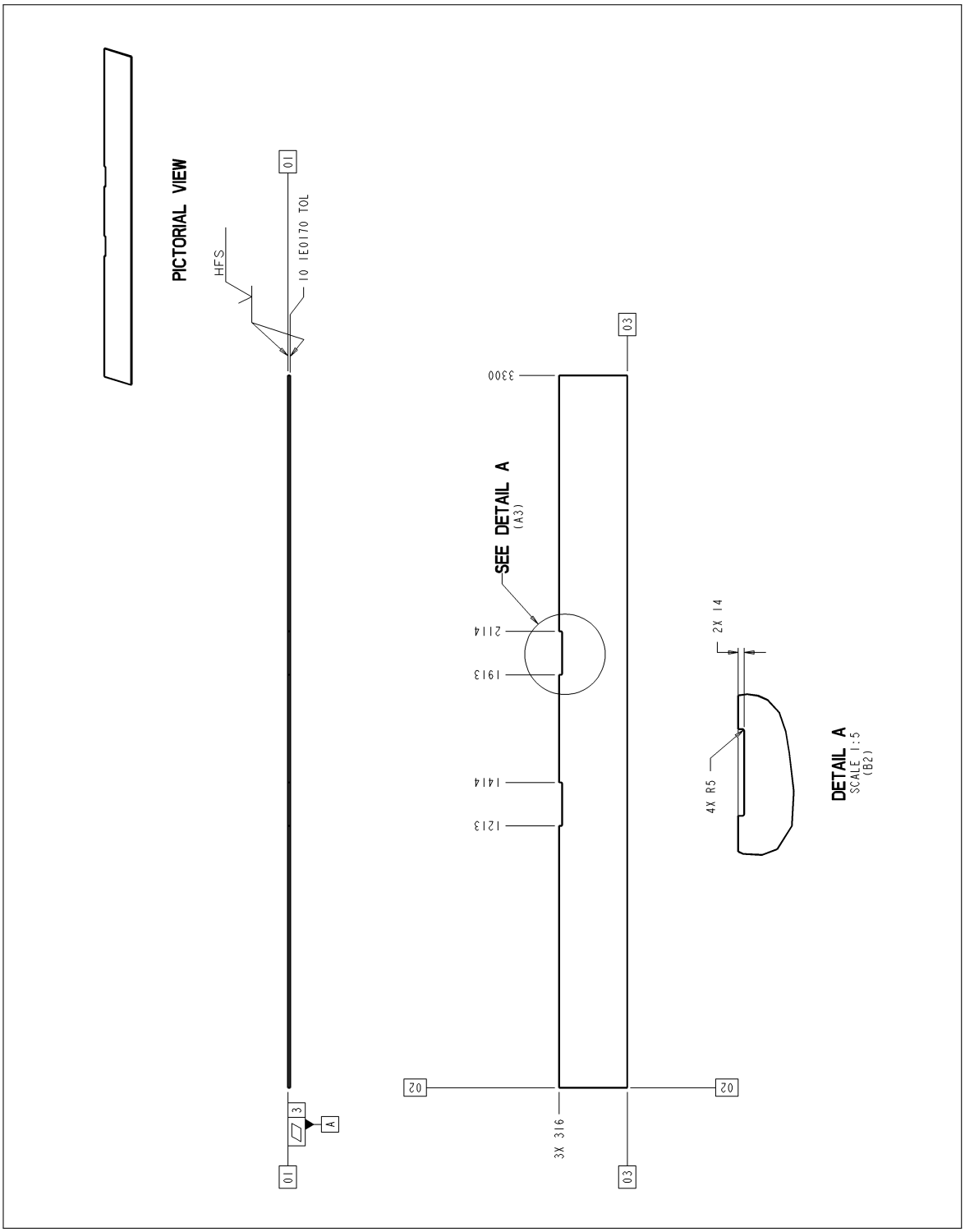


Figure 6.2: Modified drawing of Example 2.

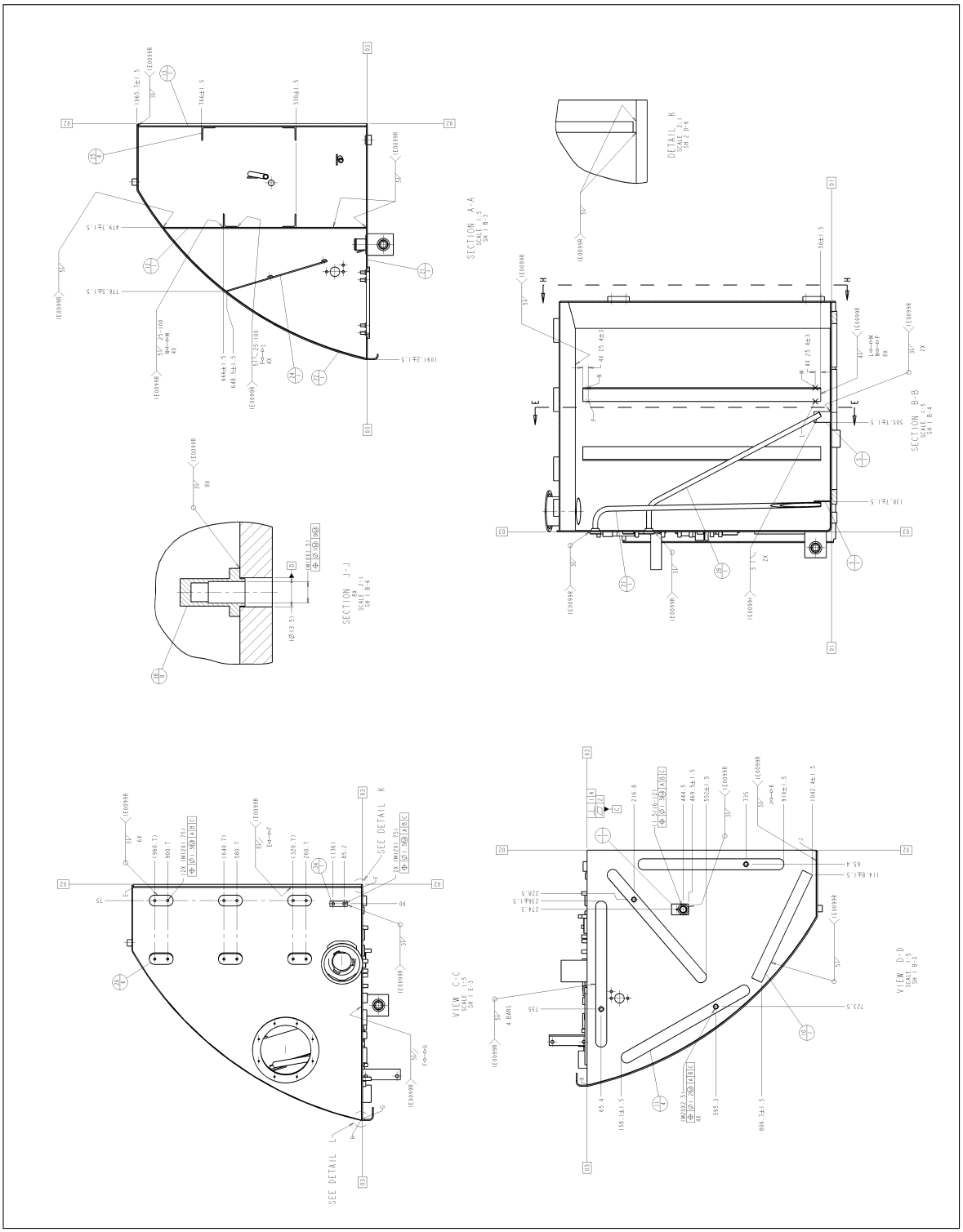


Figure 6.3: Original drawing of Example 3.

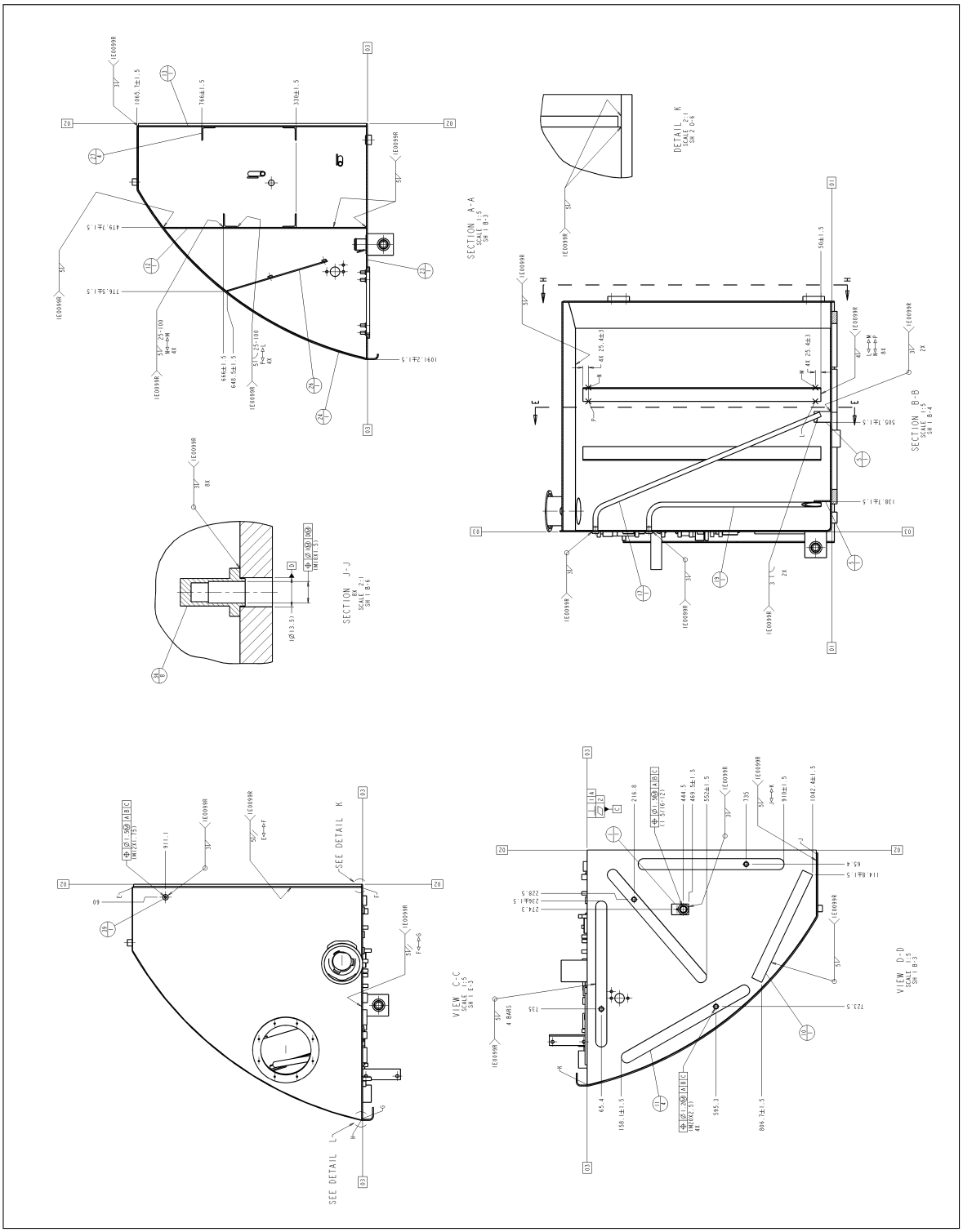


Figure 6.4: Modified drawing of Example 3.

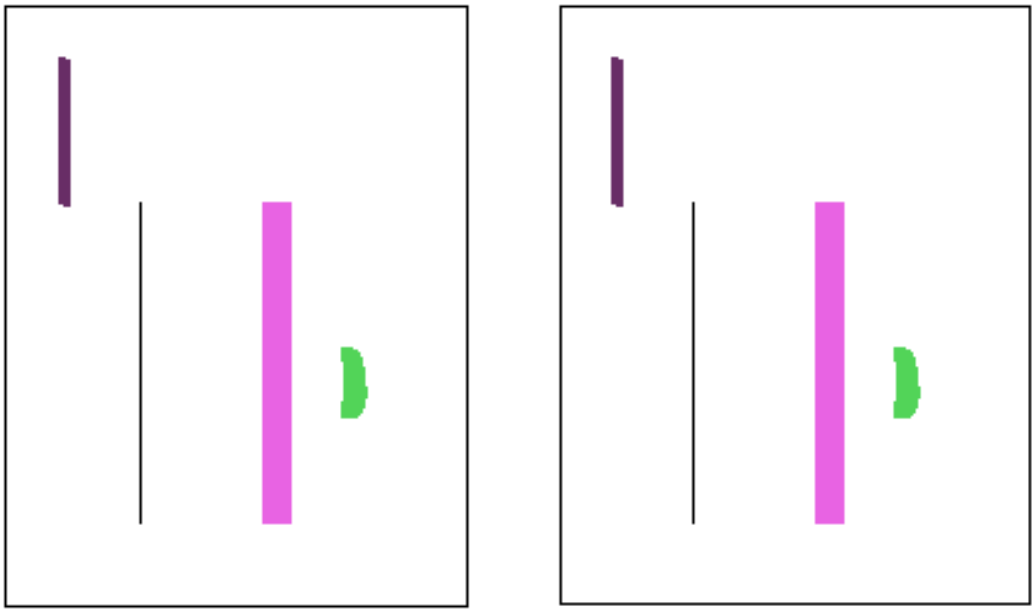


Figure 6.5: Result of segmentation and matching for Example 2. The segments shown in same color are matched.

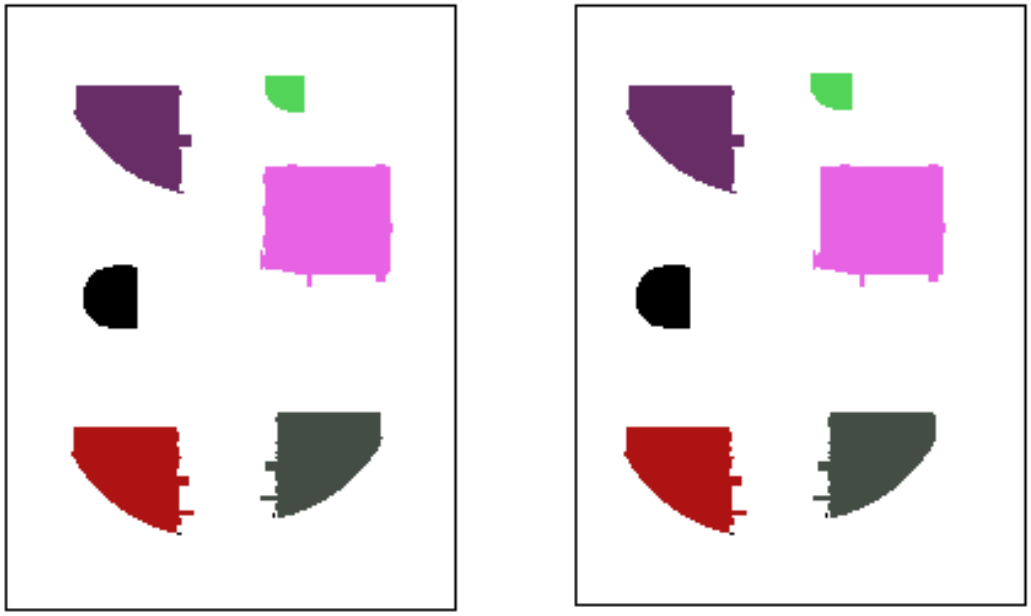


Figure 6.6: Result of segmentation and matching for Example 3. The segments shown in same color are matched.

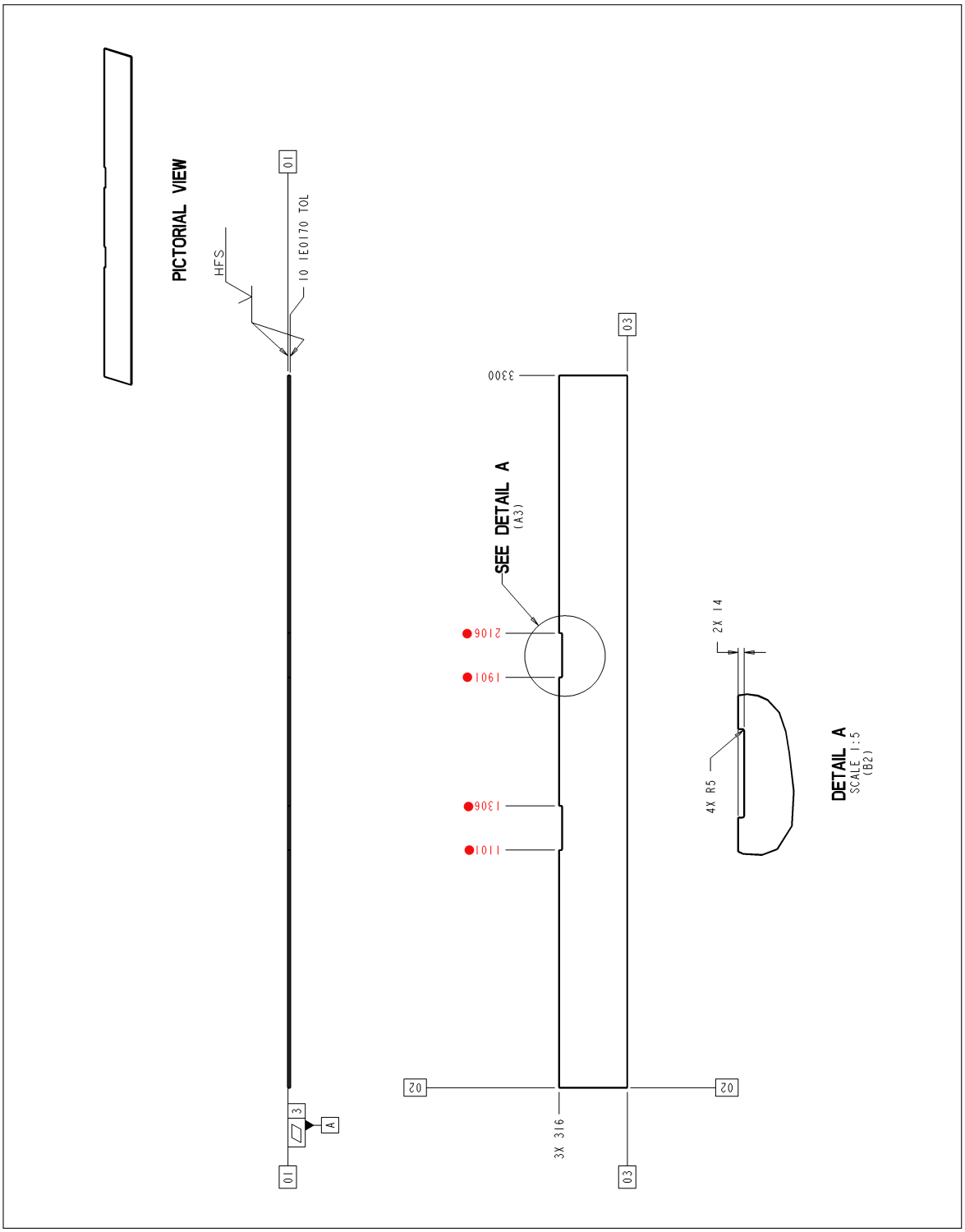


Figure 6.7: Result of image comparison for Example 2.

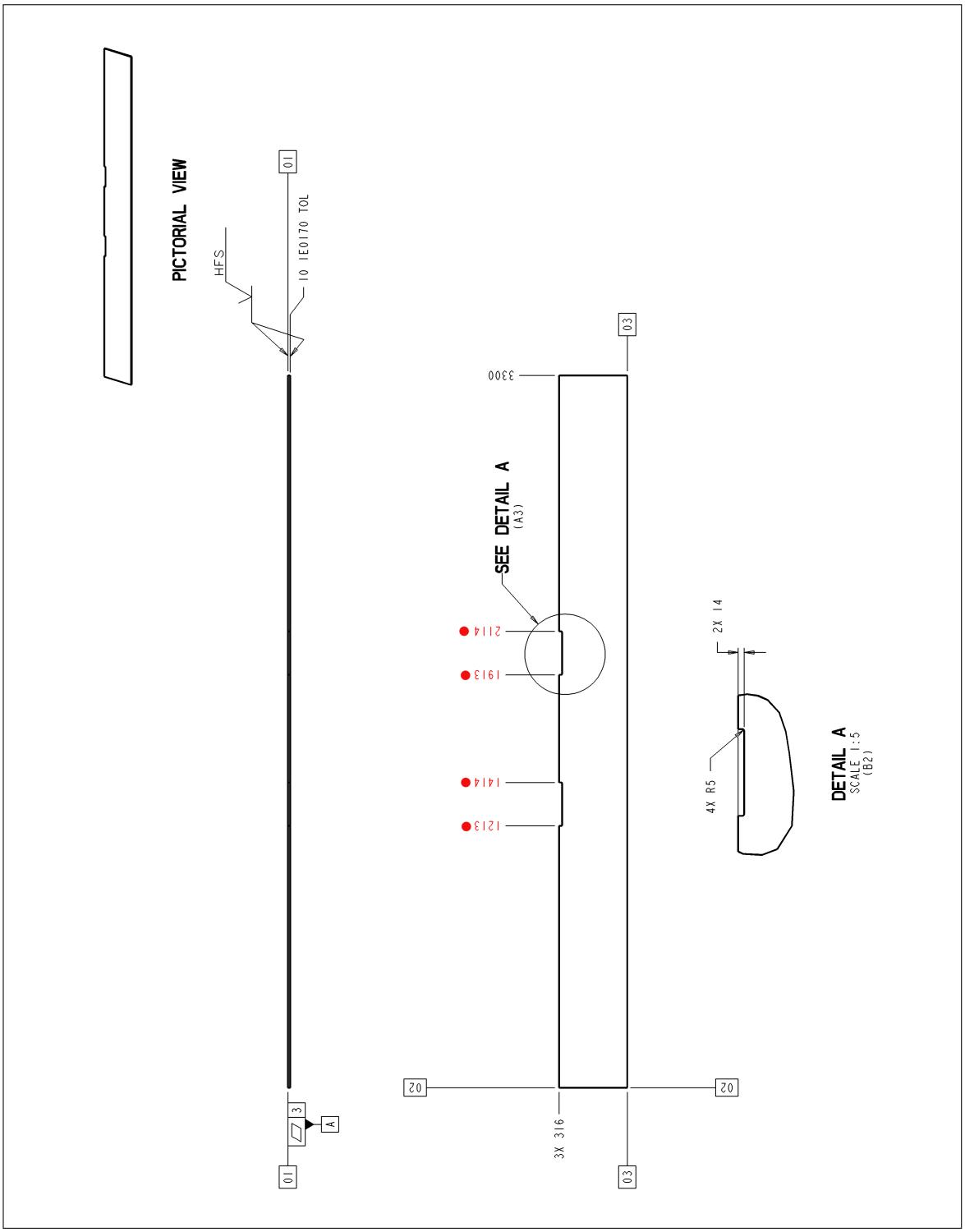


Figure 6.8: Result of image comparison for Example 2.

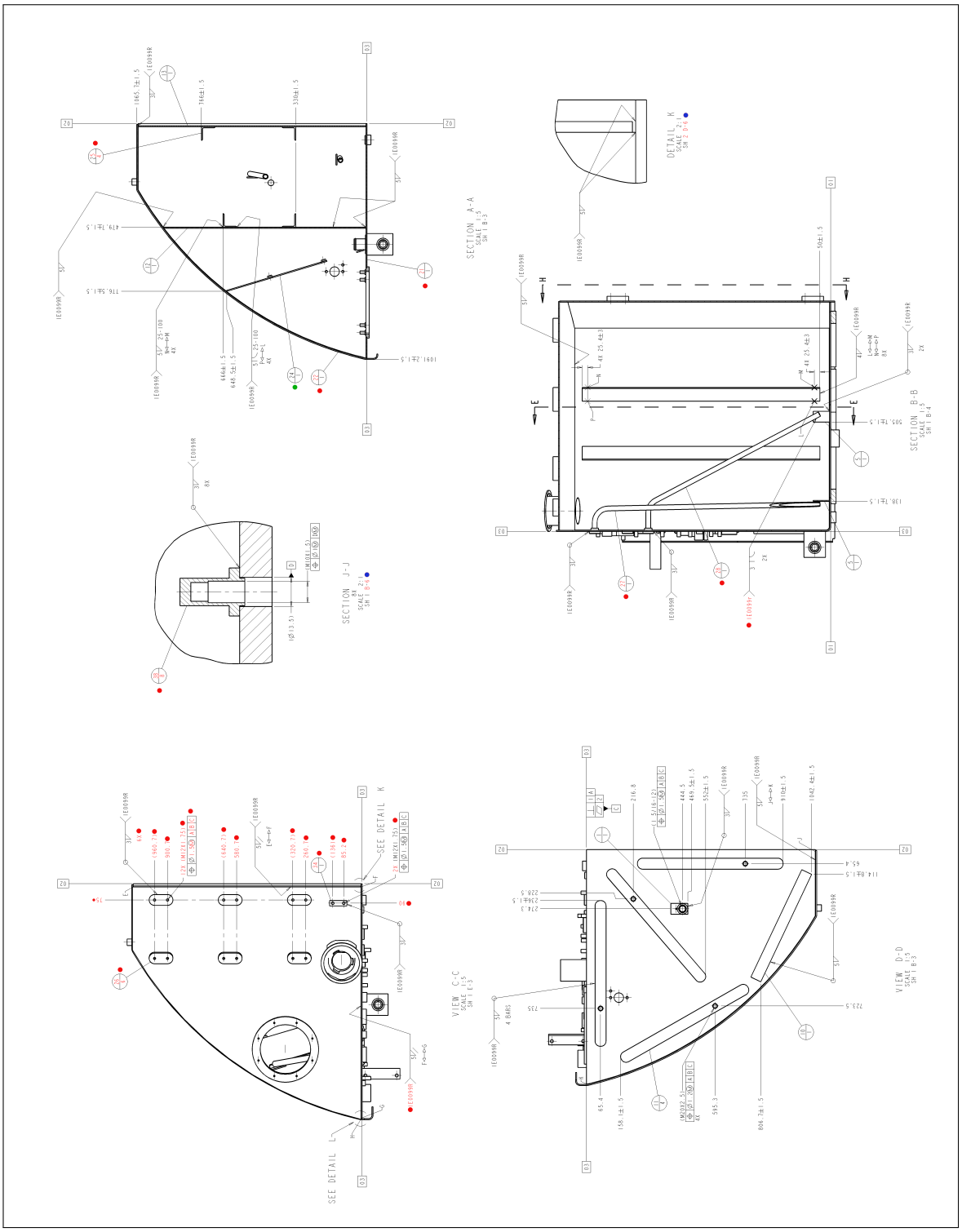


Figure 6.9: Result of image comparison for Example 3.

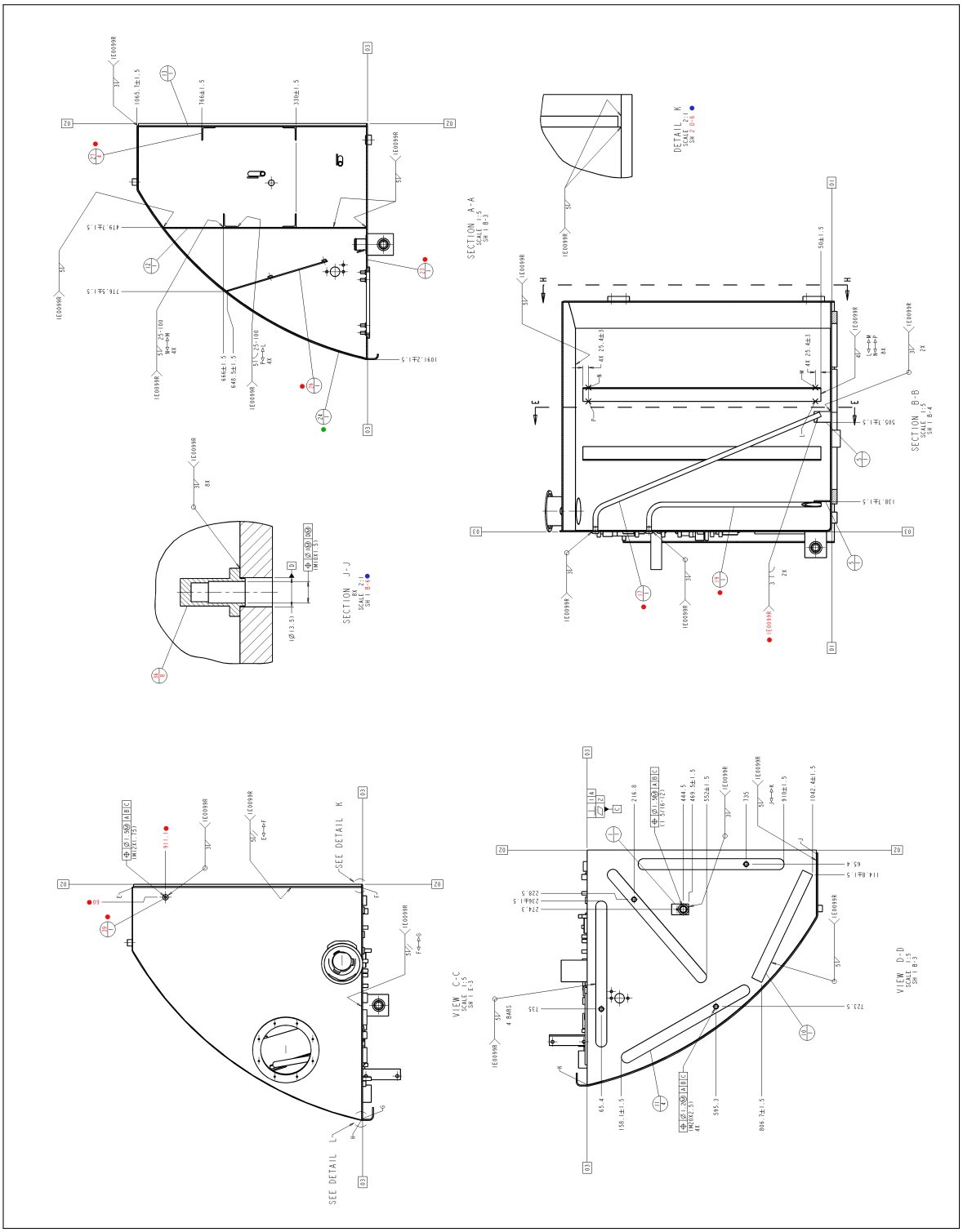


Figure 6.10: Result of image comparison for Example 3.

CHAPTER 7

Conclusions

In this thesis, we have proposed a method to separate the text and the graphics part in engineering drawings. We analyze the connected components based on the characteristics of text and graphics in these drawings to extract the text. An algorithm to segment into sub-drawings has also been proposed.

A SIFT-based algorithm was discussed to perform the matching of segments across the drawings using match matrix. This algorithm is robust to any misalignments in the two drawings and works without the need for registration.

We have presented a method to compare any two engineering drawings by means of finding matching labels. Various properties of the labels such as length, Euler number and dimensions were exploited in order to eliminate the labels which do not match. The labels were then matched using a Hausdorff distance based measure and the location of the labels. All those labels which do not have a match are highlighted to indicate the changes.

The entire implementation was done in MATLAB. The proposed method was tested on drawings with different amounts of complexities and sizes.

REFERENCES

[1] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[2] L. A. Fletcher and R. Kasturi, "A robust algorithm for text string separation from mixed text/graphics images," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 10, no. 6, pp. 910–918, 1988.

[3] C. P. Lai and R. Kasturi, "Detection of dimension sets in engineering drawings," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 16, no. 8, pp. 848–855, 1994.

[4] Z. Lu, "Detection of text regions from digital engineering drawings," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 20, no. 4, pp. 431–439, 1998.

[5] D. G. Lowe, "Object recognition from local scale-invariant features," in *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, vol. 2, pp. 1150–1157, Ieee, 1999.

[6] X. Lin, J. Ji, and Y. Gu, "The euler number study of image and its application," in *Industrial Electronics and Applications, 2007. ICIEA 2007. 2nd IEEE Conference on*, pp. 910–912, IEEE, 2007.

[7] D. P. Huttenlocher, G. A. Klanderman, and W. J. Rucklidge, "Comparing images using the hausdorff distance," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 15, no. 9, pp. 850–863, 1993.

[8] R. Fabbri, L. D. F. Costa, J. C. Torelli, and O. M. Bruno, "2d euclidean distance transform algorithms: A comparative survey," *ACM Computing Surveys (CSUR)*, vol. 40, no. 1, p. 2, 2008.

[9] R. M. Haralick, S. R. Sternberg, and X. Zhuang, "Image analysis using mathematical morphology," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, no. 4, pp. 532–550, 1987.

[10] L. Lam, S.-W. Lee, and C. Y. Suen, "Thinning methodologies-a comprehensive survey," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 14, no. 9, pp. 869–885, 1992.

[11] T. Pavlidis, "Filling algorithms for raster graphics," *Computer graphics and image processing*, vol. 10, no. 2, pp. 126–141, 1979.

[12] A. Bosch, A. Zisserman, and X. Munoz, "Scene classification via plsa," in *Computer Vision–ECCV 2006*, pp. 517–530, Springer, 2006.

[13] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1, pp. 886–893, IEEE, 2005.

[14] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 27, no. 10, pp. 1615–1630, 2005.

[15] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *Computer Vision–ECCV 2006*, pp. 404–417, Springer, 2006.

[16] P. Burt and E. Adelson, "The laplacian pyramid as a compact image code," *Communications, IEEE Transactions on*, vol. 31, no. 4, pp. 532–540, 1983.

[17] J. L. Crowley and R. M. Stern, “Fast computation of the difference of low-pass transform,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, no. 2, pp. 212–222, 1984.

[18] J. Shi and C. Tomasi, “Good features to track,” in *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR’94., 1994 IEEE Computer Society Conference on*, pp. 593–600, IEEE, 1994.

[19] C. Harris and M. Stephens, “A combined corner and edge detector,” in *Alvey vision conference*, vol. 15, p. 50, Manchester, UK, 1988.

[20] G. Borgefors, “Distance transformations in digital images,” *Computer vision, graphics, and image processing*, vol. 34, no. 3, pp. 344–371, 1986.