Rijks*universiteit* Groningen

Faculty of Mathematics and Physics

Institute for Mathematics and Computer Sciences

Geometric Model Matching on License Plates

Final Thesis of Hans Timans

Supervisor: Dr. ir. J.A.G. Nijhuis

August 2003

Rijks*universiteit* Groningen

Faculty of Mathematics and Physics

Institute for Mathematics and Computer Sciences

Geometric Model Matching on License Plates

Final Thesis of Hans Timans

Supervisor: Dr. ir. J.A.G. Nijhuis

August 2003

Abstract

The recognition of license plates from input images often does not give a complete answer. There might be some characters that are not recognized by a neural network, or there might be more characters recognized then possible on a single plate. Using constraints in the syntax of a license plate it is possible to rule out a number of options, but certainly not all of them. To obtain the correct data from these license plates images an evaluation of the plate geometry is required. The goal in this thesis is to chart all the errors that occur in todays license plate recognition software, and use the results to create a license plate model for geometric matching. By using both geometry and syntax properties found in the dataset and the constraints put on license plate geometry and syntax by regulations an attempt is made to achieve a higher rate of recognition then before.

Table of Contents

Chapter 1: Introduction

The automatic license plate reading process can roughtly be divided into three steps. First there is the detection of vehicle and license plate in a video frame, photo image or other input source. The detection of vehicles and license plate regions is a well studied subject. Hough transformations are used since long ago [Ag88], [Ka95] and more recent studies like [Bar97], [Fu98], [Par99] reveal that the detection of a region of interest is a very popular subject. After undergoing this step the input source has been analysed and one or more regions of interest have been specified.

Character detection and recognition is the second step. Different methods to extract a license plate are in use. Here edge detection algorithms such as peak-to-valley [Lu95], [Bar97], or the Niblack algorithm [Co98] are used to localize characters from binarized images. Character segmentation on the binarized images will yield the dimensions of the characters on the plate and their relative positions [Lee02]. The character recognition is done by a neural network [Bru], [Ro02], [Nijh95-02] or for example by binary morphology [Ba98].

Combining the characters into a license plate is the final step. This part of the process will be looked into in this thesis. Conventional methods often use template matching [Mi99] to validate the license plates found. The approach proposed in this thesis will use geometric model matching [Ga99] to identify errors made in the character detection step.

Common usage of geometric model matching as proposed by Baird [Bair84] is the correction of errors in the perspective of an image. The use of parametrized 2D/3D models to geometrically match to real data has proved to be a profitable approach [Kim02], [Na99],[Jel01]. Generally the process uses a parametrized geometric shape, in 2D or 3D, on which a number of geometrical transformations is performed to acquire the best possible match on real data. After this the part of the image where the match is found will cut out and the inverted geometric transformation will result in the original model, and an image that has been transformed in the way desired.

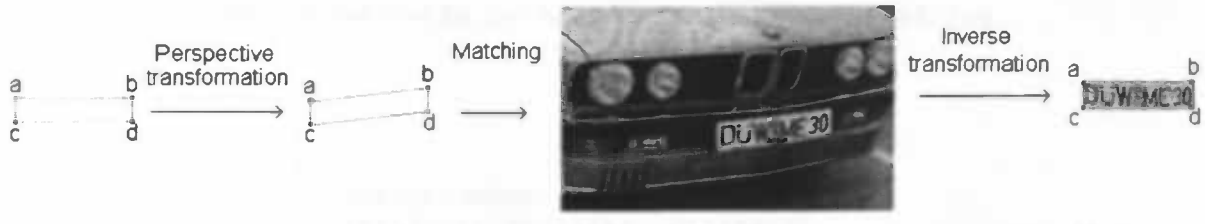


Figure i: Common use of geometric model matching for perspective correction. First a parametrized model is transformed to add perspective to it, second the model including perspective is matched to real data and last the inverted transformation will remove the perspective from the data.

Another use is detecting the location of which a certain image was taken. Using knowledge from the objects that are on an image certain features can be detected using geometric matching. These features include the position from where the image was taken, as well as certain information about the camera that was used, like the focal length of the lens.

In this thesis the use of geometric model matching is used to improve the licence plate recognition process. So far the only way to test a license plate detected in a photo image for correctness is by string comparison or using an finite state automaton to check the syntax symbols found [Hee01]. Validating the syntax of the license plate after the detection and recognition is done in several other works [Rod00-02] , but most disregard geometric attributes completely. Using a genetic algorithm to verify geometry has been done by [Pa96].

License plate recognition requires a large amount of image processing to take place. In this paper pre-processed data is used as input. The images created by the processing program are accompanied with some geometric data which can be used in the matching procedure. Some examples of processed images are shown in (Figure ii). These images are divided into three rows labelled a, b and c. Row a) is the license plate image after it has been localized by some algorithm from step one. The image has also been cleaned from perspective errors and atmospheric interferences such as bright sunlight and shadows. Row b) is a binarized picture obtained from the image in row a) using an edge detection algorithm. Characters have been segmented and bounding boxes are drawn around the regions that are identified as characters. In row c) is the result found after undergoing the character recognition step by a neural network. Characters that have been recognized are displayed in black with a white box around them.

Figure ii: Some examples from the input images used for this research. The three rows indicate three different stages of the recognition process. Original images in row a), binarized images after edge detection and segmentation in row b) and characters recognized by neural network in row c).

The bounding boxes that are drawn around the characters in row b) have a data structure attached to them in which some information regarding the size, confidence and location of the character is stored. For the leftmost picture in (Figure ii) the structure will look like this:

```
IMAGE_NAME: /home3/ALPR_DATA/00006/Images/000/00086.jpg
NR_AREAS: 5
AREA_ID: 0
NR_UPSCALES: 4
UPSCALE_ID: 0
NR_SYMBOLS: 6
SYMBOL: 2[c940i0l2424r3650b1700t200F146B240]
SYMBOL: 6[c941i1l4030r5310b1700t200F121B240]
SYMBOL: J[c956i2l6325r7225b1700t300F142B245]
SYMBOL: D[c844i3l7725r9025b1700t300F139B236]
SYMBOL: Z[c900i4l10024r11250b1600t100F150B237]
SYMBOL: X[c755i5l11275r12575b1500t100F140B234]
TIFF_DUMP: syntax_00140.tif
```

First there is some info about the image pre processing steps, followed by the characters (SYMBOL) that are found in the image. In this particular case six symbols are found, and for every character recognized there is an array of attributes included. The first character found is

the number "2" followed by the neural network confidence labeled "c" of 940, which resembles a 94% confidece rating. After that there is an index "i" for every character starting from 0 for the leftmost character. Then there is four values for the bounding box in which the character is found, labbeled "l", "r", "b" and "t" representing the left, right, bottom and top edges respectively. After these geometric properties there are also 2 colour properties "F" and "B" representing some information about the foreground and background grey levels.

Things that can be derived from this information are:

- Distance between two found locations of characters: does the spacing between two boxes match the spacing in the model?
- The shape of the box found. In some cases a small box is found, which can possibly exclude some characters. The letter "W" will take more space then an "I" for example.
- Determination of the average width of characters and digits.
- Finding characters that shouldn't have been recognized at all: when there are only 6 characters in the model and more are found, some will need to be excluded from the final result.

What needs to be done first is analyzing a large set of input pictures manually to determine what kind of errors are occurring most often. Based on these observations an analysis of the license plate constraints has to be made to determine which errors can be detected easily and which will be hard or impossible to detect. After making these comparisons the knowledge aquired can be used to create a license plate model to match the data to. The model has to be able to detect and/or correct the errors specified in the first research and it will need to reject the plates that fail the verification step.

Chapter 2: Error Type Survey

2.1 *Classification*

The left and the right border of the plate will sometimes be detected, as well as the area of the car surrounding the plate. These include headlights, car brand logos, bumpers, and commercial texts among others. Errors in which additional characters are recognized will from now on be referred to as **Additionals**.

Additionals can come in any amount, but very occasionally more then ten additionals are seen in one image. In (Figure iii) the Dutch model license plate is shown, having three pairs of characters, shown in green. The red squares resemble additionals found in the image. In (Figure iv) the images that match these two models are shown.

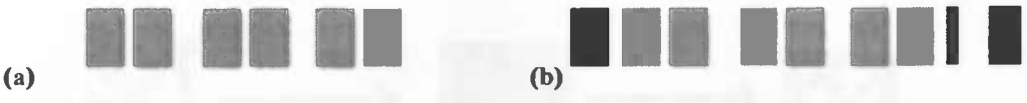


Figure iii: Correct Dutch plate model (a) and Dutch plate model with outside additionals (b) shown in red.

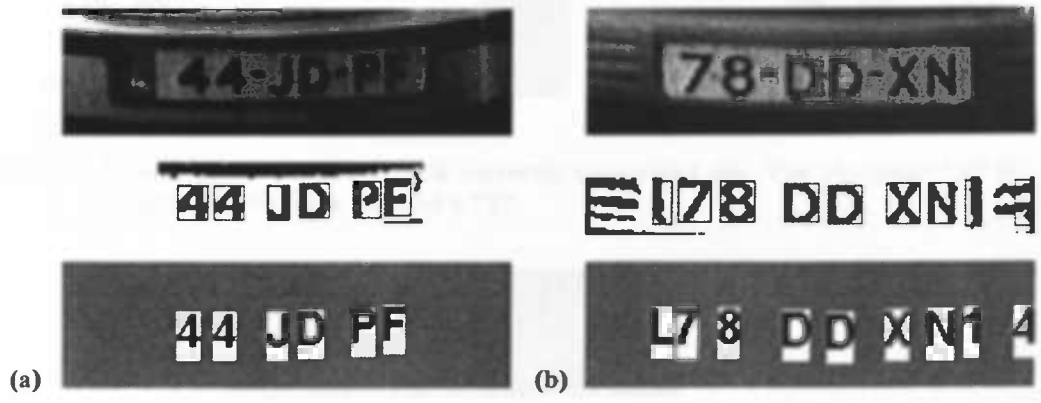


Figure iv: Correct Duch license plate (a) and Dutch license plate with outside additionals (b) matching the models in (Figure iii).

Additionals also occur between characters on the license plate. When an extra character is detected between the characters that are on the license plate, possibly caused by dirt smudges or stickers on the license plate. These two different errors will be referred to as **Inside Additionals** and **Outside Additionals.**

Figure v: Dutch license plate models with inside additionals shown in red.

Figure vi: Images corresponding to the models in (Figure v) with inside additionals caused by the separator dash on the plate.

Another error that occurs is the recognition of multiple characters on the same place on a license plate. Sometimes the detector finds two or more characters in the same region on the plate, and will put two separate bounding boxes around them. This error has some similarities with inside additionals, but there is a small difference: characters being recognized on top of each other both have the same index. When multiple characters are recognized when there should be only one, the character will be called a **Double character**.

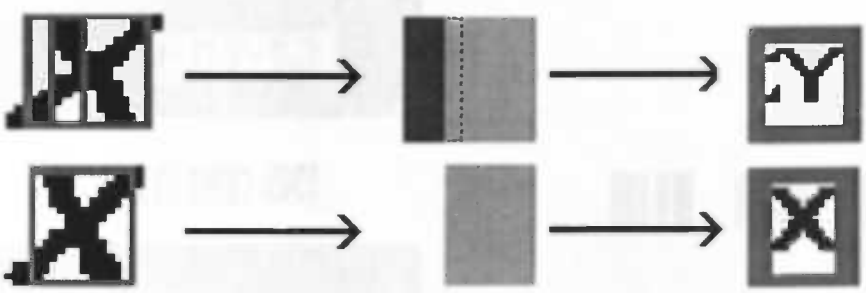


Figure vii: Double character (top) and a correctly recognized one. The character "X" is recognized twice; as a "2" and a "Y".

Figure viii: The Dutch license plate with the double character "X" as seen in (Figure vii).

In the above plate (Figure vii and viii), the character "X" is doubly recognized by the detector, first as a "2", but also as a "Y".

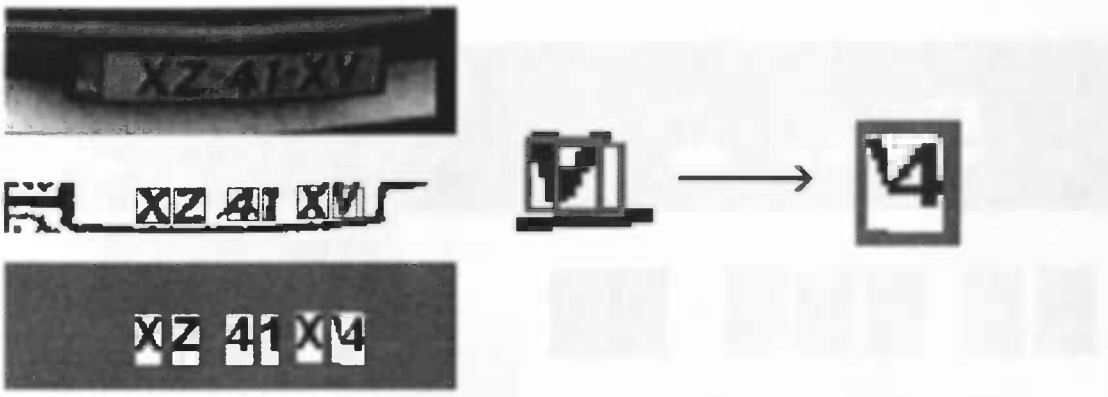
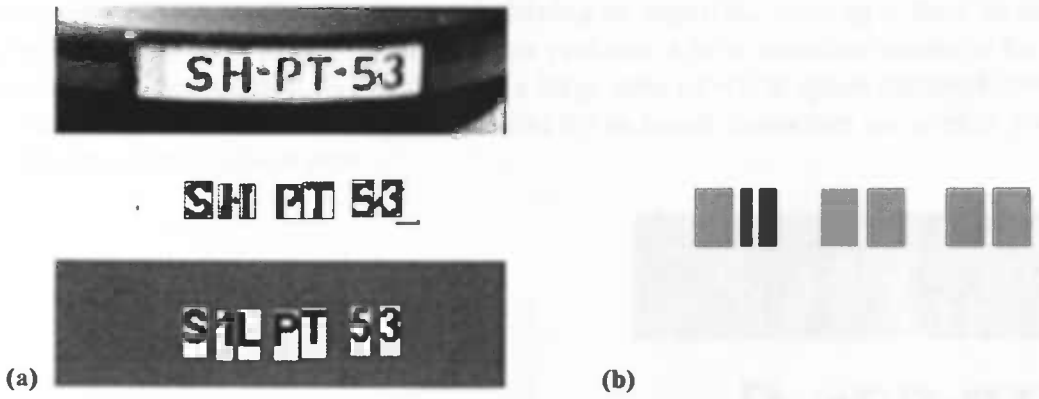


Figure ix: Another double character in a pre-processing output image, and the highlighted double. The character "V" is recognized twice; as a "V" and as a "4".

Split characters are characters that have been cut in two by the detection algorithm. In stead of a single character two are found in its place. A character split is the next type of error that can be pointed out, it has a lot of similarities with the double, but in a split character the two recognized symbols are not on top of each other which makes them a lot harder to detect. Since there is no overlap in the two recognized halves, they do not have the same index either which makes them hard to distinguish from an **Inside Additional** as well.



(a) (b)

Figure x: A Dutch license plate with a split character on it (a). The character "H" is split up into an "1" and "L". In (b) the corresponding plate model is shown.

The opposite situation also occurs. Sometimes two characters are recognized as a single one. Of course the bounding boxes are twice as wide as normal and should be easily detectable except for some special cases. Errors of this type will from now on be referred to as **Merged characters**.

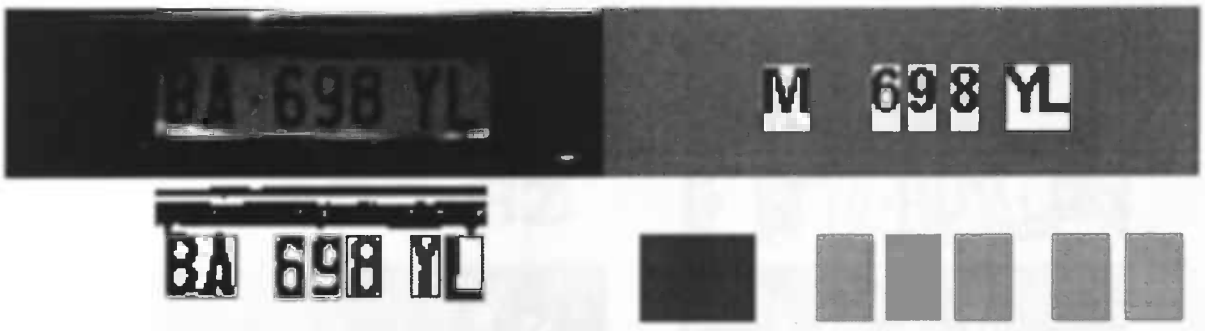
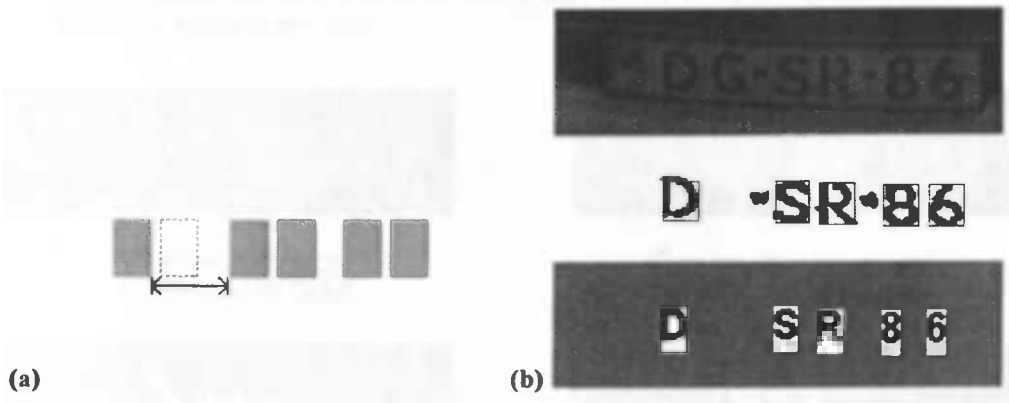


Figure xi: Italian license plate with a merged character. Characters "B" and "A" are merged into one and recognized as a "M". The last picture shows the coresponding model.

Character merging can be caused by numerous reasons. In (Figure xi) we see character merging caused by a shadow cast on the license plate. After pre-processing of the image the shadow border is also detected, and connects the character "B" to the "A" which causes the recognition module to regard it as one single character. These kind of distortions can obviously be caused by a smudge, a sticker or another factor on the plate as well.

Missing characters are another common found error. The original image might have been blurred a bit by weather, high speed driving or nightfall, making it hard to detect the characters on the license plate. To correct this problem a new character needs to be inserted to complete the license plate. When there is a large area of white space detected there might be a missing character, and the program should try to insert characters on several places and see if it fits to a model afterwards.



(a) (b)

Figure xii: Model Dutch license plate with a missing character shown in dashes (a), and an image that matches it (b).

In (Figure xii, b) a license plate with a very dark background colour, most likely being taken at dusk is shown. Due to this, the character "G" is not found by the detector causing a missing character on the plate.

There might be one or more missing characters in a license plate. The more there are the less likely it will become that the plate will be recognized. Missing characters also are found on plates that have been cut out wrong by the pre processing steps. Trying to fit the Dutch license plate model on the leftmost picture in (Figure xiii, a) will require two characters to be inserted into it, and the removal of the additional that is found on the left. In (Figure xiii, b) a plate which is cut out diagonally can be seen, which causes some characters to be missing as well since the plate isn't centered properly.

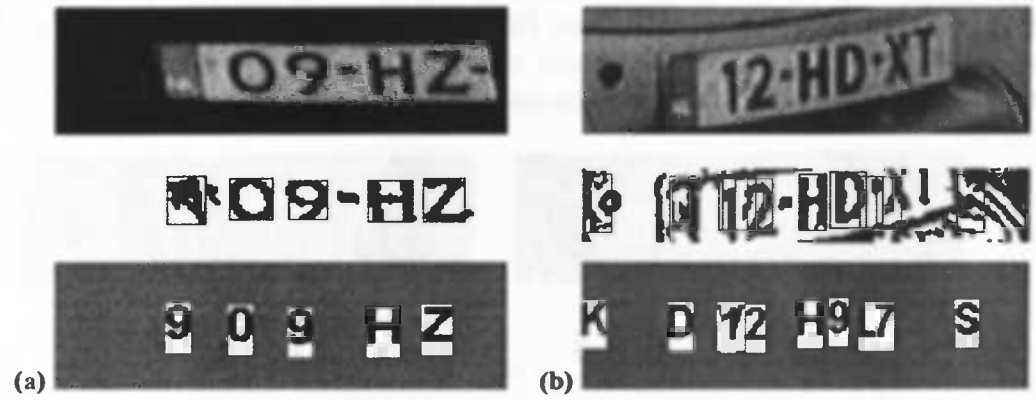


Figure xiii: Detection in badly cut out license plates. The plate in (a) is missing the last pair of characters, in (b) the plate isn't alligned properly.

Dimension errors are caused by dirt smudges and other small disturbances on the license plate. Mostly the error will cause the width of the detected character to be significantly larger then it should be. In theory this error could also affect the height of the character to increase but height variations have been removed in a previous pre-processing step. These errors will influence the average width ratio of a single character significantly, and will most certainly hamper geometric matching because the geometry is fraud.

Figure xiv: Dutch and Belgian license plate model with different types of dimension errors. Image (a) shows a character which is too wide, image (b) shows a character which is too thin.

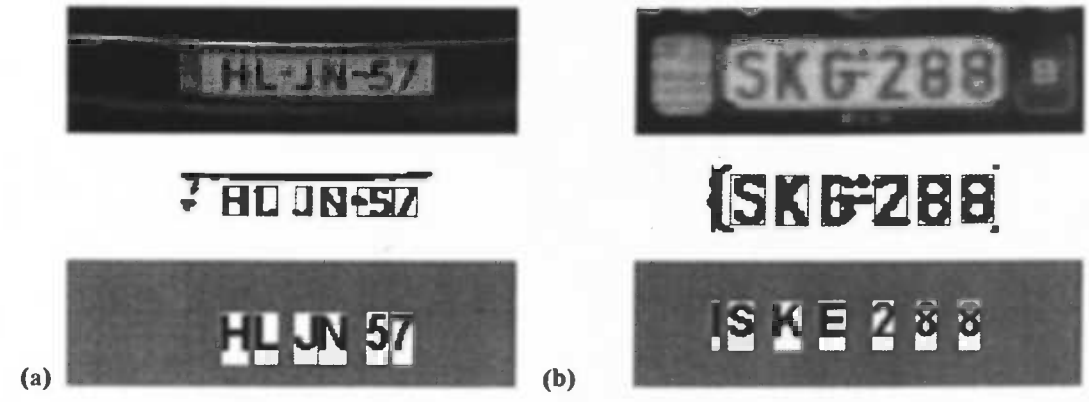


Figure xv: Plates having dimensional errors on them. The character "5" in (a) is cut out too wide, and the "G" in (b) is too thin. The plates are matching the models of (Figure xiv).

In (Figure xv) two different cases of dimensional errors are shown. The character "5" in the left image is about 1,5 times as wide as it should be, in this case it is caused by the separator on the license plate. In the right picture there is a sticker between the character "G" and "2" causing some problems. In this case the recognition module takes only half of the character "G" and recognizes it as an "E". Some other occurrences of dimension errors can be seen in (Figure viii) where the bounding boxes around characters "P" and "J" are somewhat smaller then they should be.

One of the biggest causes of errors is the angle in which the pictures are taken. Due to perspective in the picture the license plate will need to be adjusted in such a way that the characters are in line horizontally. Often this causes the character to undergo a series of stretching, tilting and turning with a resulting image like the ones in (Figure xvi) underneath.

Figure xvi: License plate images after undergoing a series of perspective corrections. Even after corrections the perspective sometimes still is not optimal.

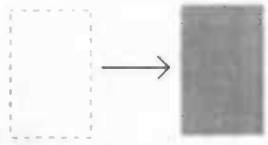
The characters on the resulting plates are often a bit tilted, as if they were in "italic" font. This sometimes causes the top of a character being above the bottom of another character, causing some overlap. Since the bounding boxes all have perpendicular edges the detection of these characters is often hampered because the neighbouring character has a few pixels that are inside the bounding box as well.

When the perspective is not correctly adjusted, another problem emerges. In this case the plate is not projected horizontally, but diagonally accross the cut out region in stead. In (Figure xiii) are two images of incorrectly cut out plates, that cannot be recognized since the plate edges have been cut off.

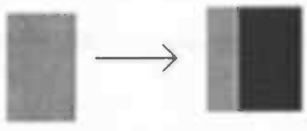
2.2 *Summary*

In short, we define the following types of errors:

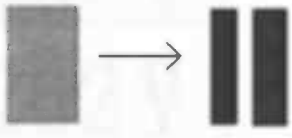
1: Additionals
-Inside
-Outside



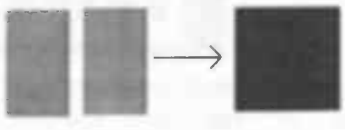
2: Double Characters (with overlap)



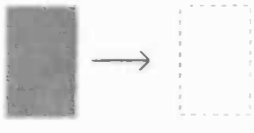
3: Split Characters



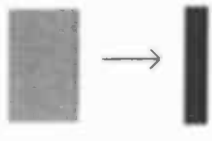
4: Merged Characters

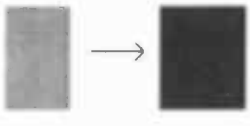


5: Missing Characters



6: Dimension Errors





To chart the frequency of these common error types, a large amount of data has to be analysed manually. In the table below can be seen that some error types are very country dependant. Inside additionals are very common on Austrian license plates as opposed to other countries. The reason for finding this large amount of inside additionals is the regional registration logo on the plate. As can be seen in (Figure xvii) there is a logo on the license

plate between the characters "M" and "2" that causes an additional character. The error that occurs the most is clearly the Additionals, and mostly the ones occurring outside the actual plate boundary.

	Total	Inside Additionals	Outside Additionals	Split Characters	Double Characters	Merged Characters	Missing Characters	Dimension Errors
NL	1000	11 (1.1%)	962 (96.2%)	20 (2,0%)	6 (0.6%)	8 (0.8%)	59 (5.9%)	12 (1.2%)
I	1000	8 (0.8%)	883 (88.3%)	0 (0,0%)	3 (0.3%)	39 (3.9%)	19 (1.9%)	11 (1.1%)
B	1000	3 (0.3%)	822 (88.2)	0 (0,0%)	0 (0.0%)	2 (0.2%)	37 (3.7%)	29 (2.9%)
A	1000	902 (90.2%)	760 (76%)	2 (0,2%)	1 (0.1%)	4 (0.4%)	13 (1.%)	9 (0.9%)

Table i: Error frequencies found on license plates from different countries (Netherlands, Italy, Belgium and Austria).

Figure xvii: On an Austrian license plate there is a high change to find Inside Additionals. These are caused by the regional logo found between characters "M" and "2" on this particular plate.

Chapter 3: Analysis of the Error Types

In order to get an idea of how create a good model for matching an image to the license plates defined in different countries we first need to analyse the different legal constraints. The many different types of license plates that are in use need to be analysed and charted to get an optimal model.

The program takes input files in the format described in the introduction. This input will be handled by two different routines, one for checking syntax and one for checking geometry.

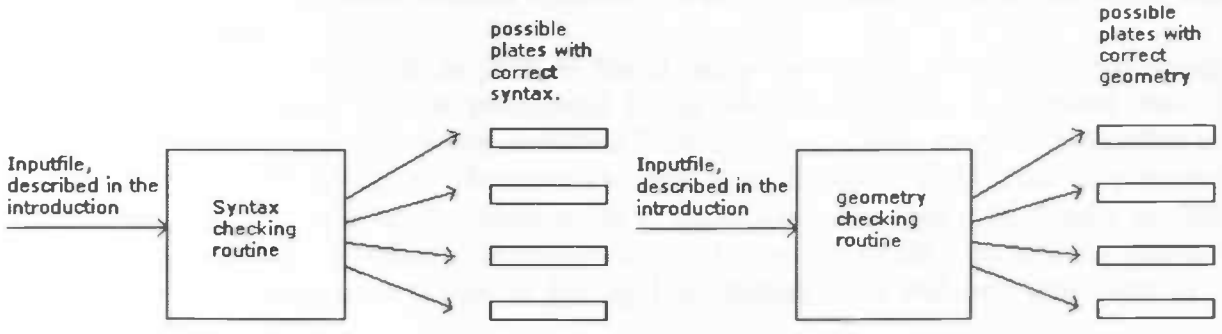


Figure xviii: Routines in the license plate recognition program.

The syntax checking module as mentioned briefly in [Nijh97], tests whether a candidate plate has the correct plate syntax. A license plate is bound to country-specific constraints, which enables the syntax checking module to immediately discard all the syntactically incorrect candidate plates. The syntax module will give an equal match on all the possible solutions it finds: either the syntax matches, or it does not.

The geometry checking module uses only the bounding boxes surrounding a character. Mapping the data from the input to a geometric model, which will be discussed in chapter five will result not only in a match, but also a match confidence will be added, stating the goodness of the match.

When the syntax routine is run first, a lot of possible results will appear, depending on the amount of countries that are included in the search. All possible plates found with correct syntax would be inserted to the geometry checking routine and after that all plates with correct geometry and syntax will remain. The second option is to put the geometry routine up first and start with checking for correct geometry. This way all geometrically correct plates will get a fast syntax check afterwards, resulting in the correct recognized plate.

The possible errors on the plates will cause a very big variance in the amount of plates having a correct syntax. Below will be a short resume for every errortype proposed in the introduction.

3.1 Additionals

A license plate image that has *a* additionals on it needs to delete *a* characters in order for a plate to have the correct amount (*n*) of characters. The more additionals on the plate the more possible combinations can be made to result in a correct plate. For example: when dealing with a Dutch license plate (which holds 6 normal characters) with 1 additional, all 7 characters are possibly an additional. Identifying which of them actually is the additional is the task of the geometry and syntax modules.

The total number of characters that is found on the input will be divided into candidate plates of *n* characters, and the other *a* will be considered additionals. Generally there are *p* candidates that need to be tested, where:

p = (n + a) above n

Detecting additionals from an input string using the geometry checking module can also be done. The difference here is that, unlike the syntax checking module, the geometry module has the means to check whether appointing a certain character to be an additional was a good choice or not.

Whenever a correct license plate is found using the syntax module, the remaining character(s) on the plate can be considered being additionals. Thus a character that is considered to be an additional on one candidate license plate, can very easily be a member of another plate, on which another character is considered to be an additional. The syntax module can be used to split up the input string of input characters into syntactically correct license plates, marking a number of characters as valid members of the license plate, and the rest as additionals. So in the example of having 1 additional, there will be a maximum of 5 possible plates.

3.2 Double Characters

Double characters are easily detectable by both the geometry and the syntax module. Since double characters have the same index assigned to them, for each double character having the same index a new candidate license plate is created. There can be any number of possible double characters having the same index, so if a license plate having *n* characters has 3 different options for index 2, and 2 options for index 4, there are 6 candidate license plates that need to be tested.

The total number of candidate license plates when double characters are present can be calculated by multiplication of the number of characters for every index.

*p = (#characters on index 0) * (#characters on index 1) * (#characters on index 2) * ...*

In short, detection of double characters by the syntax and geometry modules is unnecessary, since they are already known in advance. The double characters are detected by the neural network that is responsible for the character recognition phase of the process.

3.3 Missing Characters

Missing characters are perhaps the most difficult ones to handle for the syntax module. Since the syntax module is unable to detect a missing character, it has to be told one is present. When a character is unknown there are 26 possibilities for a missing character and 10 for a missing digit, resulting in 36 license plates that need to be checked. In general, when there are x characters missing on a single license plate, for the value of total candidate license plates p we can write:

$p = 36^x$

When the geometry module is testing a candidate license plate in which a character is missing the plate will most likely be rejected, since the module will conclude that the spacing between the remaining characters is too large.

When the position of a missing character and its dimension are known by the geometry checking module, the geometry module can rate all candidate plates to see which character fits best in the bounding box. In this case the syntax checking module will result in all syntactically correct license plates, which are all considered as equal matches.

3.4 Dimension Errors

When dealing with dimension errors the bounding boxes around the detected character are either too wide or too thin. Sometimes cause by the detection of a wrong character, but other causes are also common. In the case of an incorrectly detected character the syntax module will just use this as a valid one, creating a candidate license plate from it.

The geometry module can find the dimension error by checking the width of the found character on the plate to the width that this character is expected to have. When the geometry matches poorly, the plate will receive a low rating, or will be completely rejected in extreme cases.

Plates that have one or more dimension errors on it do not add any new candidate license plates to the process, so the extra strain on the system is minimal.

3.5 Split Double Characters

Whenever a character is split in two, there will be two recognized symbols on the plate which are both unlikely to have high confidence levels. The number of split characters s on a plate having n character on it will result in the same number of candidate license plates as if they were inside additionals. Hence the number of candidate license plates will be:

$p = (n + s)$ *above* n

The syntax checking module, which disregards the confidence level of any recognized character will result in a maximum of five candidate plates just like if it were additionals. Here the geometry checking module can make a difference, since the confidence levels of both halves of the split character are low it is also very unlikely that the geometry will match properly. Some very exceptional cases might pass the test though.

If by some means it will be possible to detect split character afteral, the best solution is to insert a missing character in place of both halves of the split character. The geometry of this missing character can be derived from both outer edges from both split characters.

3.6 Merged Characters

Merged characters on a license plate are usually shown as a single wide character. The merged character *m* is about *x* times as wide as a normal character, depending on the amount of characters merged to each other. A plate having *n* characters, with *m* occurrences of two characters merged into one will give:

$p = (n - m)$ *above* $(n - 2*m)$

candidate license plates (*p*). When more then two characters are merged to one another the total number of license plates will be lower.

The syntax checking module will use whatever character is recognized from the merged characters as a valid character, and will check the candidate license plates nevertheless. Using this incorrect data, it may still occur that a syntactically correct license plate can be created. In this case, the geometry check should find the error. The geometry checking module is able to notice the error in the width of the merged characters, and will in most cases reject the geometry of the candidate plate instantly. In the case of a geometrically correct symbol being created after a merge of two characters, which theoretically could happen, both the geometry and syntax checking modules will be unable to find the error.

Attempting to cut the merged characters up into separate ones is very unlikely to be succesful. The more characters that are attached to each other, the less likely the chances will be of separating them. When a separation is made however, all the resulting characters will have to be denoted as missing characters, which has shown to cause an exponential growth in candidate plates.

3.7 Reducing the number of candidate plates

The formulas described above will provide us with the worst case scenario, ie it will take *all* possible plates into consideration. However, the bigger part of the possibillities needs to be disregarded by the checking modules early on in the process, to keep the number of plates that need checking as low as possible. When for example a four character license plate is surrounded by four additionals, the formula states that 70 plates are possibly correct. Some of these plates will be rejected immediately since they do not meet the requirements for the syntax check, and others will match very poorly geometry-wise. When all the found objects have equal spacing between them, in the ideal case there would be only four plates that pass the geometry checks, and some of them may not have a correct syntax.

The first step in the geometry checking phase is doing measurements needed to make geometrical computations using the available information. Spacing between two characters should be within the allowable range, and so should character width. Plates that do not meet these requirements can be rejected instantly. The plates that do match all the checks will be matched on a model using different techniques.

3.8 Summary

Detection of the error types discussed in chapter 1 is not always possible. The two separate modules that check for geometry and syntax combined can however be used to detect a large percentage of incorrect license plates from input images. The number of candidate license plates that require testing can be calculated by using several formulae discussed in this chapter. In (Table ii) a short summary is given of which error type is or is not detectable by the syntax and geometry modules.

	Syntax Module	Geometry Module
Inside Additionals	Yes	Yes
Outside Additionals	Yes	Yes
Double Characters	Yes	Yes
Split Characters	No	No/Yes
Missing Characters	No	No/Yes
Merged Characters	No	No/Yes
Dimension Errors	No	No/Yes

Table ii: The possibility to detect the error types described in chapter 1 using the syntax and geometry modules.

The table shows a difference between both modules when detecting dimension errors, missing, split and merged characters. split, missing and merged characters can be handled accordingly by the geometry module, but not necessarily be identified. When a split character is present on a license plate, the geometry module notices two characters in stead of one and thus will use both halves of the split character as being separate characters. The split character now has not been identified as one, but the plate will have little chance of passing the geometry check nevertheless. Merged and missing characters have the same property, a missing character is noticed by the geometry module as a large empty region and the merged character is regarded as being a "large dimension error".

Chapter 4: The Official Models

In this chapter the constraints on license plates put on them by laws and regulations in different countries are discussed and related to the errors defined in the chapter 1. Knowledge from the license plate registration regulation as seen in [Ty99] can be used to help find ways to detect errors more easily. Since every country has its own set of prescriptions on what a license plate should look like, examples of plates from Germany, the Netherlands and Belgium are used below to illustrate the most important three variabilities.

4.1 Colours

To the eye the most significant difference between plates of different countries is, of course, the different **colours** used on the license plates. This feature however is less relevant in this study, since greyscale images are used as input in general. Some countries have different coloured license plate types in use simultaneously. A problem arrises here when the plates do not have the same foreground and backgrounds. When one of the plates in use has dark characters on a light background and the other has light characters on a dark background, one of the two types of plate will not be detected. When no plate is found the grayscale levels of the input image can be inverted, resulting in an image with the correct foreground and background. When running the program again on the new image the plate can be detected.

4.2 Fonts

The **font** used on the license plate is very important, since it has a great influence on the overall look of the license plate. Some countries use more then one font on their license plates, and of course there are differences between fonts on plates from different countries as well. For example in Germany there are two different fonts in use on license plates. Two types of character sets called *Mittelschrift* and *Engschrift* are used for the plates. Both have a similar character height of 75 mm, but the width of the *Engschrift* is a bit smaller.

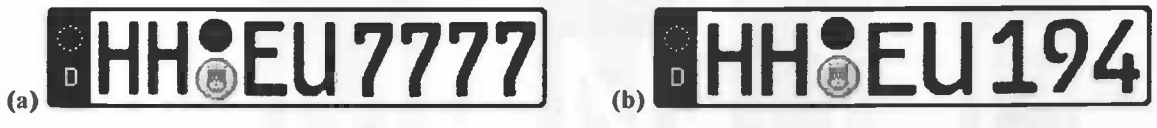


Figure xix: Two German license plates. The plate shown in (a) is in Engschrift, while (b) is in Mittelschrift.

In Germany, the license plate characters have fixed widths. The *Mittelschrift* character set uses 47,5mm for the letters and 44,5 mm for the numbers on the plate, while the

Engschrift uses 40,5 mm for the letters and 38,5 mm for the numbers. The *Mittelschrift* is the default character set used, but sometimes the plate length gets too long to fit on the maximum plate width of 520 mm, in this case the *Engschrift* is used.

4.3 Plate Length

The plate length, or the amount of characters that are on a license plate is not the same for every country. Sometimes even inside a single country license plates of various lenghts are used. For example in Germany there is a large variation in plate length. On the leftmost 1 to 3 positions on the German plates is a letter combination giving some information on where the vehicle carrying the plate has been registered. After this combination there is an open space of 63,5 mm – 67,5 mm where a sticker is found. After the sticker there is a combination of 1 or 2 letters followed by an open space of 24 – 30 mm, and finally there is a combination of 1 – 4 digits. Adding it all up we have a lot of different plates to take into consideration, at least 3 characters are on the plate, at most there are 9.

In German license plates we see **Variability in plate length,** which is a very important factor to take into consideration when creating a model license plate.

4.4 Character Width

Dutch license plates are made up out of three couples of characters, separated by dashes. The characters used are not all of the same width, but they are centered in the middle of the plate leaving 8 – 20mm space between the 2 characters in the couple.

Figure xx: Common Dutch license plate.

In (Figure xxi) is shown that the width of the five shown characters is not equal, the character "M" for example is about 1,5 times as wide as the character "L". For every character an unique geometry is defined, making some sort of box in which the character is in the middle. When creating a license plate 6 character-boxes and 2 dash-boxes are put in the correct order and no space is left between them. In Dutch license plates we have **Variable character size.** This is also an important factor that has to be taken into account when creating license plate models.

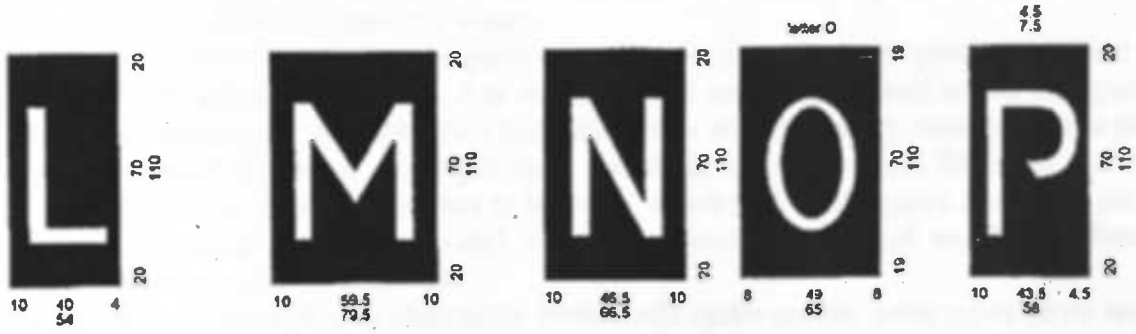


Figure xxi: Extract from the Dutch license plate definitions. Clearly can be seen that all characters have their own dimensions.

4.5 Plate distances

In Belgium yet another way of denoting the license plate is used. Here all characters on the plates have equal spacing between them, that is the distances between the "centers" of all characters are equal on the plate groups. The Belgian plates consist of two groups, the first having 3 letters, the last having 3 digits.

Figure xxi: Belgian license plates have equal space reserved for every character.

Using equal spacing leaves plates that have the digit "1" or the letter "I" on them with a lot of open space. In the figure below there is an example of such a license plate and the corresponding model.

Figure xxii: Although the number "1" on a Belgian license plate takes less space then other characters, an equal amount of space is reserved for it.

4.6 Summary

In short, legal constraints on the variability of license plate models can be divided up into 5 groups,

- **Variability in plate length**
- **Variability in character width**
- **Variability in plate distances**
- **Plate font.**
- **Plate colours.**

The two bottom features, font and colour are less relevant for geometric matching. Colour aspects are mostly lost since greyscale images are used, and font geometry is taken into account by the variability in character width.

The first factor, **variability in plate length,** will cause the most problems. Since the plate length is not known beforehand, it is unknown how many additionals are on the specific plate as well. Knowing from chapter two that additionals are the type of error that occur most frequent, this factor will most likely put the most strain on the system. For each different plate length a separate geometry test has to be done, which will need to give a result in such a way that plate ratings from one model can be compared to that of another without a transformation needing to be done first.

Having no **variability in character width** will make certain error types more easily detectable. When every character has equal width, dimension errors, missing characters and merged characters will be much easier to detect. dimension errors will even be detectable by

the geometry module without doing a syntax check beforehand, since all bounding boxes need to have the same dimensions, a small variation is easily detected. Merged characters will be recognizable by their size, which is some factor of a regular in width depending on how many characters are merged to each other. Missing characters are a little harder to detect, but fairly easy as well. Since there is no variation in character width, most likely there is no variation in distances between characters either which makes the detection of missing characters fairly easy.

When a plate has **fixed plate distances** it will be more simple to detect a missing character. In this case the location of all characters is know beforehand, so when a certain character is missing this will immediately be known.

Chapter 5: Models and Algorithms

Using the data gathered in the previous chapters it is now time to start making a suitable model to match the input with. The model license plate will be matched to the input and after the matching procedure a final result must be given. The resulting match success rating is dependant on a lot of factors, and the main goal of the model is to provide an accurate rating system. Below several possible methods to make an accurate model and rating system will be discussed.

5.1 Elastic Matching

Elastic matching [Bi97] is the first method that can be applied to the license plate matching problem. In this model the characters are connected by a imaginary piece of elastic, as well as there is a piece of elastic on the inside of each bounding box on the license plate. To match this model on a plate, the characters can be moved independently of each other to obtain the best match. This requires some of the characters to be moved to the left or right, creating some tension (= change in length of the elastic) on the connectors between them.

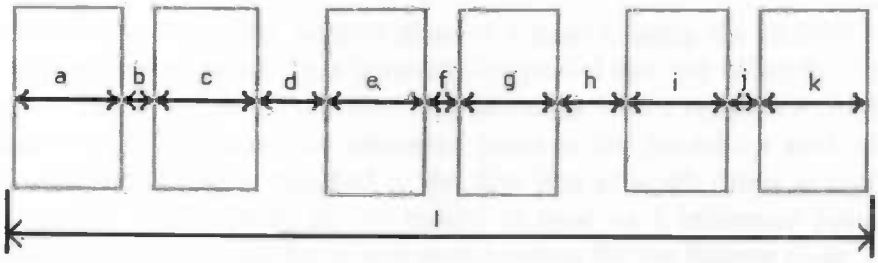


Figure xxiii: The elastic matching framework. The license plate model is connected by elastic material having a tension value when shifted to the left and right.

The amount of tension should obviously be as small as possible. The best match is when the tension on the model is zero (= exact match). In the model are twelve connecting elastics, labelled "*a*" to "*l*" from left to right. The tension on the elastic bands in the model is rated separately for each band. The rating is ontained by the use of a fitness function that results in a rating for each band, using the tension on the band as a parameter. The ratings for all the bands in the model will be combined and used to create a final plate rating. Fitness functions will be discussed further in the next chapter.

In (Figure xxiv) the elastic matching model is fitted twice on the same license plate. Since there is one outside additional on this plate there are seven possible ways to map the model on the plate, it shows the correct match and one of the six possible errors.

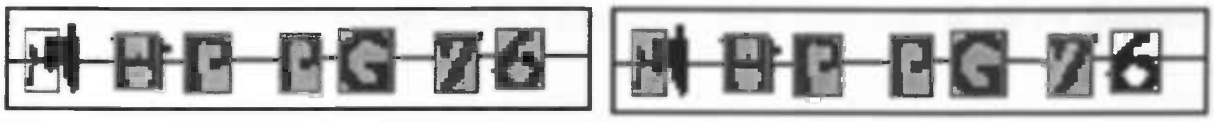


Figure xxiv: Two elastic matches of the model on the same plate with one outside additional.

In the left image we see that the boxes are almost unmoved from the original model, however in the right image there have been a lot of movements. The tension on for example the elastics "*b*", "*f*" and "*j*" are very high. These tension values should lead to a higher score for the left plate in the ranking system.

5.2 Jos Nijhuis Model

The Jos Nijhuis model [Internal communication] is the method that was in use before this research. The method is based on the elastic matching principle, but this model uses a boolean fitness functions in stead of the continuous one used in the previous paragraph. Whenever an elasit bond in the model is stretched beyond a certain degree, the rating for this point will become zero. All measuring points in the model receive a *true* or *false* rating, and the final plate rating is calculated by the logical OR of all separate ratings from the measuring points. Because of the boolean fitness function this method does not give any different plate rating then 1000 or 0; good or bad.

5.3 Centers Method

The Centers method matches the license plate to a model, using the centers of all characters on the plate as a reference point. This license plate model has one value for each character on the license plate. These values all represent the distance from a reference point, to the centers of the characters on it. There is one reference point in the model for each character on the license plate. And the model is matched to the data just as much times as there are reference points in the model. Every point in the model is used as a reference point once, and all separate matchings will make up for a final match rating for the license plate.

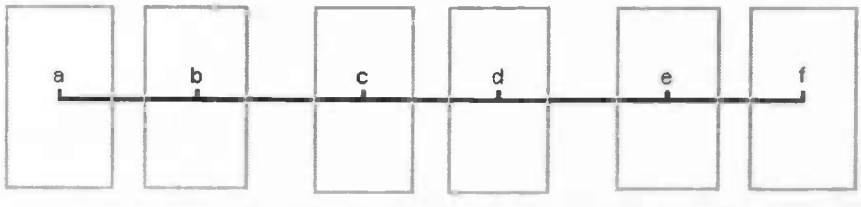


Figure xxv: Centers model, using 6 values and a plate center to match a license plate

If the character spacing is variable, an extra witch check for each character can be done to ensure that the width of all characters is correct. Like the elastic matching method, a fitness function is used to determine a match rating for all characters on the plate. The difference between the distance found in the model and the distance found in the data is used as a parameter for the fitness function here. In (Figure xxvi) the Centers method process is shown.

Six separate matches are done to obtain a final plate rating, using each of the six characters on the plate as a referrence point once.

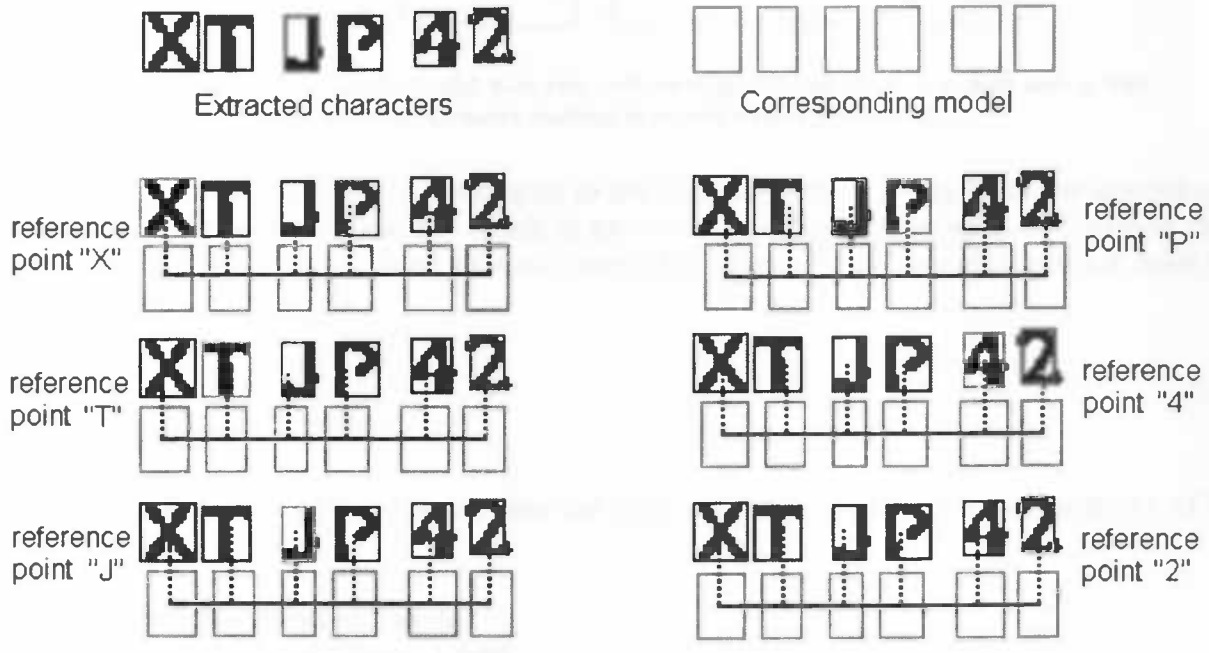


Figure xxvi : The matching process used by the Centers method. All points in the model are matched on the corresponding absolute position in the real data once for each point. All the separate matchings are combined to obtain a final plate rating afterwards.

When a character is in the wrong position, using it as a reference point will result in a very poor match, dropping the overall rating of the license plate tested. When a plate has one badly positioned character on it there will be 5 matching sequences that detect one character being misplaced, but the last test, using the badly positioned character as a referrence point will result in five misplaced characters. Therefore matching the Centers method to the real data only once is clearly not enough, since when the referrence point used in the match is on a character that is in the wrong position, the plate rating will drop significantly, as opposed to using a reference point that is in the correct position there will be only one measurement point that receives a bad rating.

5.4 Edges Method

Similar to the Centers method discussed above the Edges method uses fixed points on the license plate to map the license plate to. Instead of having only one point per character there are two, one for each side of the character. From here the distances to all other points will be determined, which means the plate has to be matched twice for each character in the model. The Edges method is better suitable for plates that have a variable character spacing and size, since it measures both sides of the character separately. This method also uses fitness functions to determine the impact of a small error on the total plate rating.

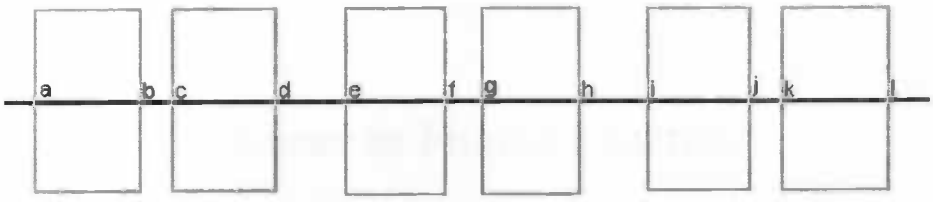


Figure xxvii: : The Edges model uses two reference points per character, and uses a way similar to the Centers method to obtain a final plate rating.

The Edges method is in many ways similar to the Centers method. Using twice the amount of measurement points will hopefully result in more accurate rating possibillities. However the amount of matchings is doubled as well using this approach, making this method have a longer run time.

5.5 Summary

From the previous chapter we know that the three aspects that influence the geometry of a license plate most are:

- **Variability in plate length**
- **Variability in character width**
- **Variability in plate distances**

The first aspect, variability in plate length can not be hadled by any of the algorithms proposed. The only way to solve this problem is to create a separate model for each plate length that can occur in the specific country. When checking for syntax first, the geometry checks are less numerous, since only the plates with correct syntax need to be checked.

Variability in character width can be handled by all the different methods. Using the Jos Nijhuis model characters and spaces that are too large or small will result in the plate being rejected instantly. The elastic matching method can handle the variability in character width by measuring the tension on the bands, and use the results from the fitness function to rate the entire plate. The Edges and Centers models can both detect errors in character width. When a character is too wide or too small, usually (but not always) the center shifts from its "perfect" position, which can be detected by the Centers method. The Edges method detects the left and right edges of the character being either too far apart or too close to eachother.

Variability in plate distances can be detected by all proposed methods as well. The Centers method however has a big advantage here; since the plate distances are usually measured from the center of the character. The other models are well suited to detect variability in plate distances as well, but they lack the property of using the centers of the characters to base their match rating on.

Chapter 6: Fitness Functions

To determine the goodness of a fit to the license plate model, a method is needed to distinguish between a bad, better good and best match. In [Yu99] a pixel based approach using templates with different ratings is used to match characters from the data to the template. The pixels in the template all add a value between 0 and 1 to the overall rating, obtaining a rating for each separate character. A similar approach will be used in this research to obtain a character rating. For each plate the differences between the model and the real data are stored, and for each character an error frequency histogram is created which can be used to defina a fitness function for each separate character.

6.1 Estimated Norm Width

A very important thing to determine is the **plate scaling ratio**. Images are rarely taken from an angle that keeps the original height/width ratios intact. Usually these are being distorted by the angle in which the camera takes pictures on the road. The characters that are on the plate however have a common height/width ratio, which can be determined for every plate. Character widths may vary, as seen in chapter 3, but the character widths can all be found in the lawbook of a certain country. So we can estimate the **normal character width** (estimated norm width) for every plate.

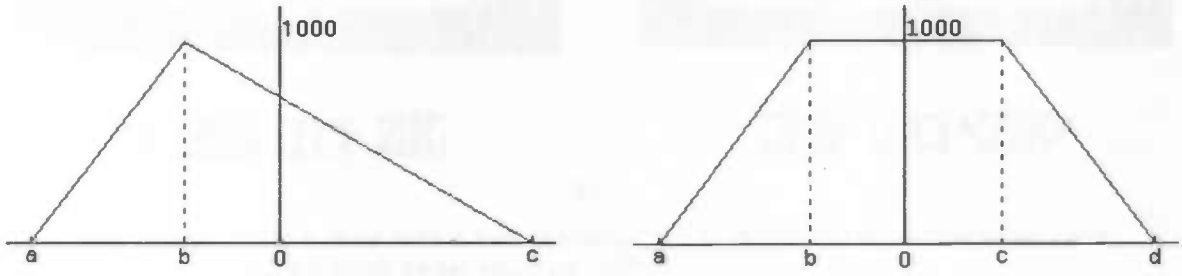
The estimated norm width is in fact nothing more then a number in which we express the distances measured in an image. For each license plate made by specific constraints, the normal character width is the same. Estimating the normal character width can be done by applying a label to each character, expressing the particular character width in estimated-norm-widths. These numbers can be used to estimate the normal character width by comparing the actual width found on the license plate to the width that the character should have. For example if a character "1" is found, its width should be smaller then the width of a character "W" on the same plate.

Displacement and perspective caused by the angle in which the picture is taken cause distortions that make it impossible to know the absolute width of the license plate on an image, nor will the exact widths of a single character be known. Using the estimated norm width computed for the plate in progress we can determine relative distances on the license plates, and represent them as factors of the estimated norm width. For example the total plate length is 12,5 estimated-norm-widths, character "B" is 0,92 estimated-norm-widths wide, etc.

6.2 Error Frequency Histograms

Results from a test, using ~1500 different Dutch license plate images using the 3 different models from chapter 5 gave the following result for the fitness function per plate character [appendix A,B,C]. The histograms correspond to the characters that are commonly on a

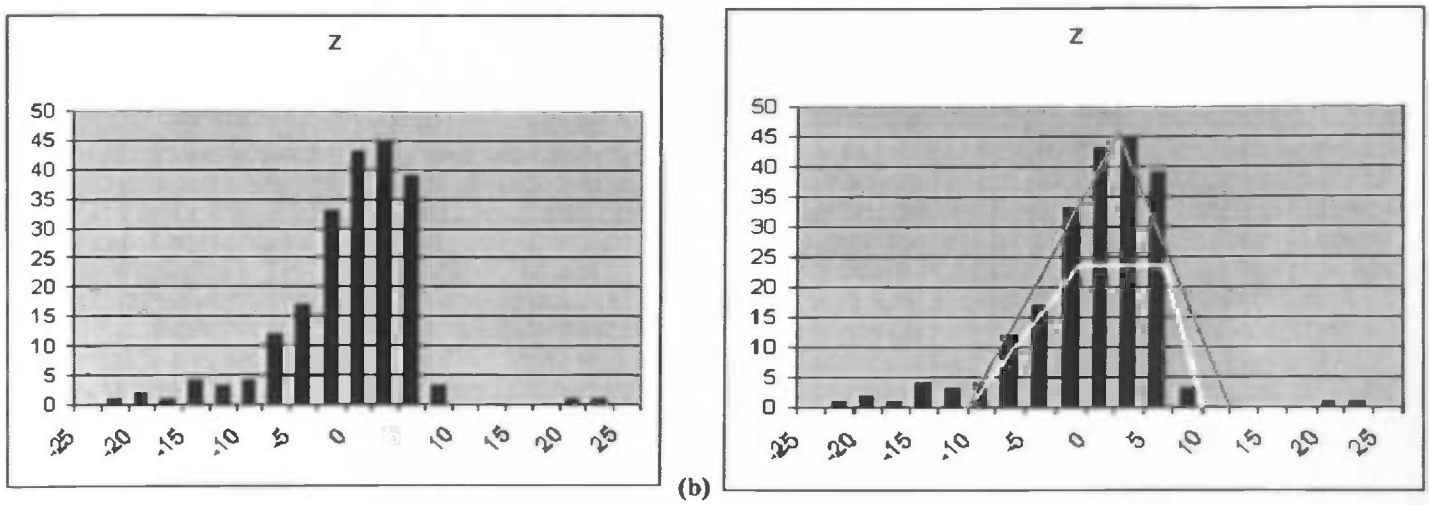
Dutch license plate, and only characters recognized with a confidence level above 50% are included in the histograms to give a more accurate picture. With help from these histograms a fitness function per character can be determined. There are many different ways to create a fitness function, in (Figure xxviii) 2 possibilities are shown. For the remainder of this thesis, (Figure xxviii, a) will be referred to as fitness function 1 (FF1) and (Figure xxviii, b) will be referred to as fitness function 2 (FF2). Both generic fitness functions have a few parameters (the values *a, b, c* and *d*) that have to be deducted separately for each license plate character. Some characters have a wider range of error then others have, thus a more tight region of acceptance can be used for the characters with smaller varration in the error.



(a) (b)

Figure xxviii: Generic fitness functions FF1 (a) and FF2 (b). The values a, b, c and d have to be deducted manually for each character.

Taken from appendix A is the following histogram shown in (Figure xxix, a). On the x-axis are the error margins from a character "Z" to the model license plate, expressed in normal character widths. The x-axis has been divided in to 20 different groups, each corresponding to a 2,5% error range. On the y-axis are the occurrences of all 20 different error ranges. In this particular case the error range from 2,5 to 5% has the most occurrences, closely followed by the 0 to 2,5% range. Clearly the most errors are found in the range between –2,5% and 7,5%. Using this data FF1 and FF2 for this character can be derived manually, as shown in (Figure xxix, b).

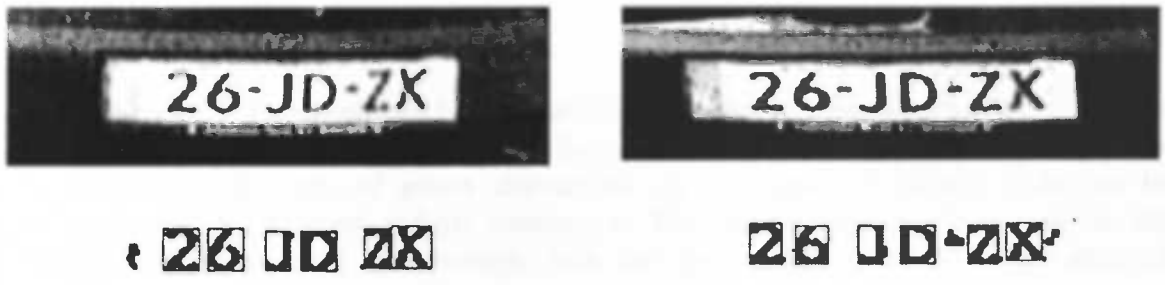


(a) (b)

Figure xxix: Histogram for character "Z" (a) and corresponding fitness functions (b). FF1 is shown in green and FF2 in yellow.

To relate these histograms to the original data, for giving some visual feedback below two pictures are included. Using different kinds of image processing methods a license plate

picture had two different outcomes shown in (Figure xviii). Matching the two plates gave two different results for the character "Z" on both license plates, in (Figure xxx, a) the error measured: 21.7%, a very large error, which is justified by the picture. The character "Z" on (Figure xxx, a) is clearly less wide then the others. In (Figure xxx, b), the character "Z" had a much lower error: 1.2%, a small distance from where it should have been according to the model.



(a) (b)

Figure xxx: In (a) a plate with a bad character "Z" is shown, (b) shows the same plate with a much better character "Z".

Chapter 7: Results

Testing of the proposed methods has been done in two ways. First, by using knowledge of the character dimensions and fitness functions, a license plate that has the maximum rating was created. Effects of all kinds of errors introduced on this "perfect" license plate can be observed and analyzed to draw a final conclusion. The second approach is testing on real data. One has to make sure that enough data has been tested to come to an accurate conclusion.

First the "perfect" test license plate is created to analyze the effect of small variances in the data. Variables that are usefull to test are for example the effects of a change in:

- **Scaling ratio**; after increasing all values on a license plate by a certain factor the end result should be equal. This test is to ensure that the estimated norm width was calculated correctly.
- **Character movement**; Additional characters that do have correct dimensions but are located in a wrong position on the license plate need to be detected. In this test the effect of moving a character away from its perfect position and the effect of these additionals on the total plate rating can be observed.
- **Character dimension changes**; when character dimensions are modified from the perfect size, the plate rating should decrease. This test includes the detection of merged and split characters, and the effects of various dimension errors can be measured.
- **Missing character**; when looking for an unknown character on a certain position, the expected character should be rated highest.

The "perfect" license plate used in the following test is XX-XX-44 which resembles the input string:

```
X[i0f0c1000l101r1758b0t200]X[i1f0c1000l2225r3883b0t200]X[i2f0c100015300r695
8b0t200]X[i3f0c1000l7425r9083b0t200]4[i4f0c1000l10500r12073b0t200]4[i5f0c10
00l12540r14113b0t200]
```

A license plate having a maximum score can be calculated by using the defined widths for every character. The fitness function for every character also contributes a lot to the final plate rating. In this test characters "X" and "4" have been used since the fitness functions for these characters have their maximum value at an error of 0%, making it much easier to calculate the estimated norm withd manually.

7.1 Plate Scaling Ratio

In this test the Dutch format is tested. First of all, the "perfect" license plate is rated, to show that it is actually "perfect":

```
...answer  : found in total 1 possible plate(s)
             PLATE[0] = XXXX44 (NL XX-XX-44) confidence: 1000
```

Using all four different methods from chapter 4, the license plate was rated 1000 points, the maximum score.When all values are multiplied by 100, the plate rating drops a little to this is most likely caused by some rounding errors.

```
Elastic matching:
...answer  : found in total 1 possible plate(s)
             PLATE[0] = XXXX44 (NL XX-XX-44) confidence: 998

Centers method:
...answer  : found in total 1 possible plate(s)
             PLATE[0] = XXXX44 (NL XX-XX-44) confidence: 998

Edges method:
...answer  : found in total 1 possible plate(s)
             PLATE[0] = XXXX44 (NL XX-XX-44) confidence: 999
```

When all values are divided by 10 or 100, some loss of data will occur, so the total rating is expected to drop a little. The results when the plate was scaled down 10 times for each method:

```
Elastic matching:
...answer  : found in total 1 possible plate(s)
             PLATE[0] = XXXX44 (NL XX-XX-44) confidence: 963

Centers method:
...answer  : found in total 1 possible plate(s)
             PLATE[0] = XXXX44 (NL XX-XX-44) confidence: 788

Edges method:
...answer  : found in total 1 possible plate(s)
             PLATE[0] = XXXX44 (NL XX-XX-44) confidence: 787
```

And scaled down 100 times:

```
Elastic matching:
...answer  : found in total 1 possible plate(s)
             PLATE[0] = XXXX44 (NL XX-XX-44) confidence: 717

Centers method:
...answer  : found in total 1 possible plate(s)
             PLATE[0] = XXXX44 (NL XX-XX-44) confidence: 456

Edges method:
...answer  : found in total 1 possible plate(s)
             PLATE[0] = XXXX44 (NL XX-XX-44) confidence: 461
```

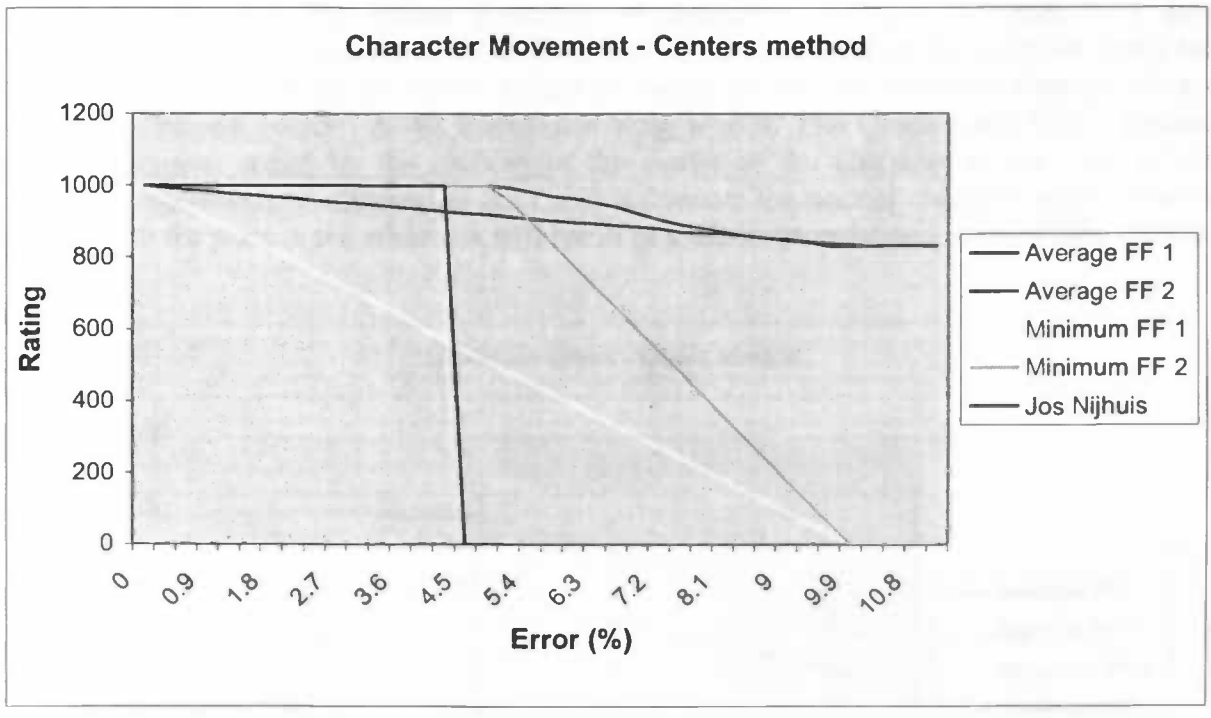
The division by 10 did not cause any loss of data that was significant enough to change the final plate rating a lot when using elastic matching, but the rating on the other two methods dropped significantly. However when scaled downwards 100 times the confidence dropped

on all three methods. Scaling a plate upwards does not cause as much data loss as downwards, since there is no loss of input data. The Jos Nijhuis approach accepted the upscaling and downscaling by a factor of 10, but scaling upwards and downwards by a facter of 100 resulted in the plate being rejected.

7.2 Character Movement

Character movement is also expected to be of some influence on the final plate rating. When a character is moved sideways from the perfect license plate, the rating is expected to drop. The effect that the character movement has on the final plate rating is dependant on the way the final plate rating is formed. Logically, when using the average from all measurement points as plate rating the end result will be higher then when using the minimum of all ratings.

To check the effect of character movement, a series of experiments have been done. All the proposed methods have been tested separately, using the two different fitness functions from chapter 5, as well as using average and minimum to obtain a final plate rating from the separate charater ratings.



Graph i: Effects of character movement on plate rating using the Centers method.

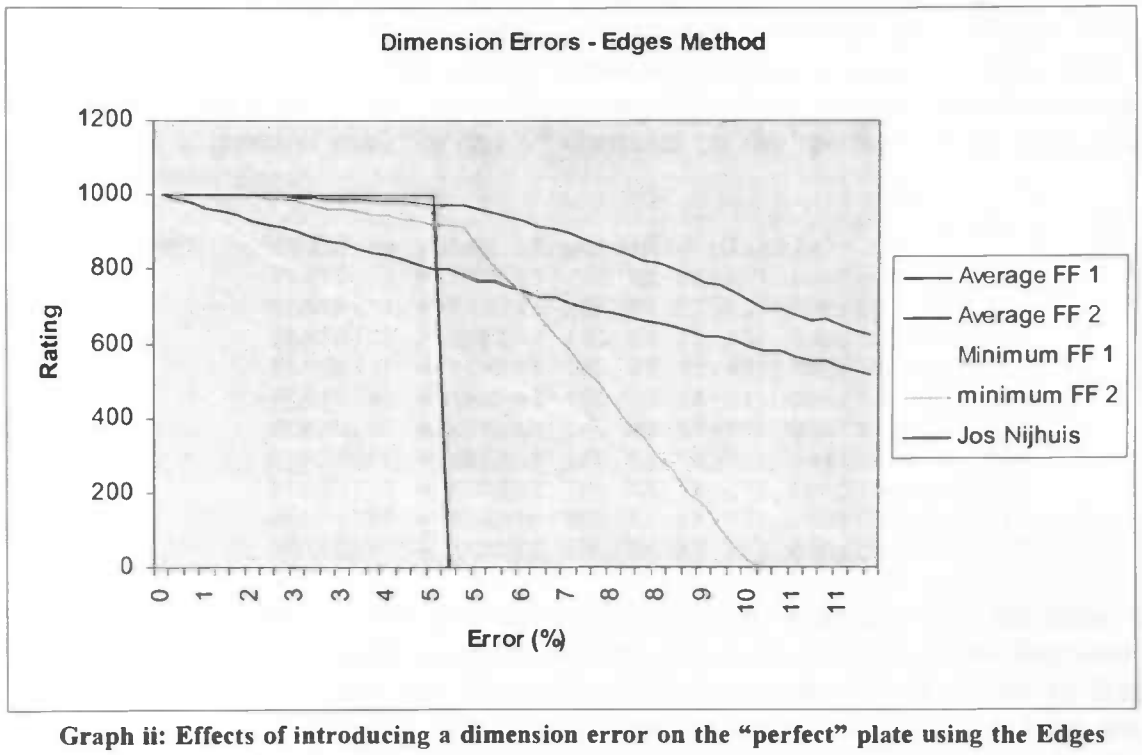
In (Graph i) the effect of moving one single character on the perfect plate influences the plate rating using the centers method. In yellow and light blue the minimum of all character ratings has been used to obtain a final plate rating, and for the dark blue and light red lines the average was used. Clearly, when using the minimum of all characters the fitness function of the character moved will appear, in (Graph i) the error margin on the character moved was 10% before the character would be considered to be entirely wrong. When using the average of all measurement points, all but one point in the model is rated "perfect" and the remaining one drops from 1000 (max) to 0 (min). This particular test was done on a Dutch license plate having six characters on it, thus using the Centers method 5 measurements rated 1000, so the

final plate rating cannot drop below 5000/6 = 833 points. The Edges method has a similar result since 2 out of 12 measurement points are rated 0, and 10 are rated 1000. When using the elastic matching approach, a similar graph appears. However since there are 12 measurement points in the elastic matching method, and only one of them is wrong, the average rating always stays above 11000/12 = 917 here. This is even higher then the Centers and Edges methods. Clearly, this is undesirable, since an additional character should lower the final plate rating much more. Therefore the use of the minimum of all character ratings is a much better approach here, rejecting the plate when the character is moved away beyond a certain margin.

The Jos Nijhuis model was set to tolerate a 5% error margin, after which the plate rating would drop to 0. All other characters became irrelevant after only one character crossed this margin. In Appendix E the same graph can be found for the elastic matching and the Edges method.

7.3 Dimension Changes

Dimension changes are the ones most dependant on the character fitness function. A character being to wide can have a larger impact on the plate rating then a character being too thin, and vice versa. The fitness functions on characters "X" and "4" both have their maximum value when the error is 0, however the "X" is more tolerant for character being too wide then to those that are too small. Like a character movement dimension changes always affect two "elastic bonds" in the elastic matching model. The Centers and Edges method detect dimension errors by the shifting of the center of the character or the edge of the character respectively. A dimension error also influences the normal character width, when a character on the plate is too wide this will result in a too large estimated norm width, and vice versa.



Graph ii: Effects of introducing a dimension error on the "perfect" plate using the Edges method.

Testing for dimension errors can be done in the same way as testing for character movement, with the difference of moving only one edge of the character sideways in stead of both. In this test the left edge of a character on the plate is moved sideways slowly, and the effect of the movement is charted in (Graph ii).

Again, the yellow and light blue lines represent the plate rating when using the minimum of all rating points, and the dark blue and light red represent the average. When using the average the rating does not stay on a certain level like when testing for character movement, since the estimated norm width changes along with the dimension error. This results in a gradually dropping graph for the averaging functions. When using the Minimum the rating does not stay at 1000 (max) for long either, again caused by the change in estimated norm width caused by the dimension error. The angle in the light blue line caused by the fitness function can still be seen, but it is not exactly equal to the fitness function used like the light blue line in (Graph i). In Appendix F the same graph can be found for the elastic matching and the Centers method.

Dimension errors clearly have a bigger impact on the overall plate rating then character movement does. When a single characters dimension has a 10% error in it, the plate rating will be around 600 when using the average of all measuring points, and below 200 when using the minimum. Interpolating the light red and dark blue lines in (Graph ii) one can see that the rating will drop further when the dimension error is increased. When detecting dimension errors it is more beneficial to use the average then to use the minimum, because in most cases there are very small dimension errors on all characters on the plate, and adding those up will result in a accurate plate rating when using the average of all characters.

7.4 Missing Characters

Missing characters are tested by inserting a question mark into the input string in the place of a character. When a question mark is found, the program will insert characters itself, and if the syntax of the plate including the inserted character is correct the plate will be rated for geometry as well. All syntactycally correct plates will receive a separate rating. This will result in 26 different possibilities when a character is reported missing, and 10 for a digit. When inserting a question mark for the 5th character on the "perfect" plate, using elastic matching the results are:

```
...answer  : found in total 10 possible plate(s)
             PLATE[0] = XXXX40 (NL XX-XX-40) confidence: 983
             PLATE[1] = XXXX41 (NL XX-XX-41) confidence: 30
             PLATE[2] = XXXX42 (NL XX-XX-42) confidence: 634
             PLATE[3] = XXXX43 (NL XX-XX-43) confidence: 586
             PLATE[4] = XXXX44 (NL XX-XX-44) confidence: 1000
             PLATE[5] = XXXX45 (NL XX-XX-45) confidence: 634
             PLATE[6] = XXXX46 (NL XX-XX-46) confidence: 739
             PLATE[7] = XXXX47 (NL XX-XX-47) confidence: 693
             PLATE[8] = XXXX48 (NL XX-XX-48) confidence: 811
             PLATE[9] = XXXX49 (NL XX-XX-49) confidence: 857
```

Our "perfect" plate is the one that gets the maximum rating, allthough there are others that are very close to it. This is mainly caused by the fact that some character fitness functions are more tolerant then others. Also the normwidth of the character "0" is closest to that of character "4". Characters "6", "8" and "9" are equally close to "4". The next ones are, in decreasing order are "7", "5", "2", "3" and finally "1" is the farthest away. This order can be seen in the found ratings as well.

Testing the same thing, now removing a character in stead of a digit has the following result:

```
...answer  : found in total 26 possible plate(s)
             PLATE[0] = AXXX44 (NL AX-XX-44) confidence: 695
             PLATE[1] = BXXX44 (NL BX-XX-44) confidence: 562
             PLATE[2] = CXXX44 (NL CX-XX-44) confidence: 536
             PLATE[3] = DXXX44 (NL DX-XX-44) confidence: 849
             PLATE[4] = EXXX44 (NL EX-XX-44) confidence: 395
             PLATE[5] = FXXX44 (NL FX-XX-44) confidence: 395
             PLATE[6] = GXXX44 (NL GX-XX-44) confidence: 774
             PLATE[7] = HXXX44 (NL HX-XX-44) confidence: 625
             PLATE[8] = IXXX44 (NL IX-XX-44) confidence: 27
             PLATE[9] = JXXX44 (NL JX-XX-44) confidence: 238
             PLATE[10] = KXXX44 (NL KX-XX-44) confidence: 804
             PLATE[11] = LXXX44 (NL LX-XX-44) confidence: 420
             PLATE[12] = MXXX44 (NL MX-XX-44) confidence: 501
             PLATE[13] = NXXX44 (NL NX-XX-44) confidence: 785
             PLATE[14] = OXXX44 (NL OX-XX-44) confidence: 942
             PLATE[15] = PXXX44 (NL PX-XX-44) confidence: 536
             PLATE[16] = QXXX44 (NL QX-XX-44) confidence: 536
             PLATE[17] = RXXX44 (NL RX-XX-44) confidence: 852
             PLATE[18] = SXXX44 (NL SX-XX-44) confidence: 562
             PLATE[19] = TXXX44 (NL TX-XX-44) confidence: 687
             PLATE[20] = UXXX44 (NL UX-XX-44) confidence: 719
             PLATE[21] = VXXX44 (NL VX-XX-44) confidence: 911
             PLATE[22] = WXXX44 (NL WX-XX-44) confidence: 304
             PLATE[23] = XXXX44 (NL XX-XX-44) confidence: 1000
             PLATE[24] = YXXX44 (NL YX-XX-44) confidence: 649
             PLATE[25] = ZXXX44 (NL ZX-XX-44) confidence: 555
```

Again showing the same result, and clearly the normwidths of all characters can be deducted from the results of this test. The results of this test performed with the Centers and Edges methods are included in Appendix D. The use of these methods did not result in any unexpected results.

7.5 Testing on Real Data(1): Manually Rated Plates

To test on real data arbitrarily three different license plate classes have been created. In (Figure xxxi) a plate corresponding to each class is shown. Class 1 contains plates that are correct in all aspects; no notewothy dimension errors and near optimal character positioning. In Class 2 are plates that have a small error in the character positioning or character dimension. In short, all plates that are correct except for one small error that catches the eye of someone with a little experience in the subject have been put in Class 2. The third class

(a) (b) (c)

Figure xxxi: Examples from images from classes 1, 2 and 3. Class 1 in (a), class 2 in (b) and class 3 in (c).

contains plates with a rather large error on them, which can immediately be determined by anyone. Plates with dimension errors caused by an dirt smudge or other factor that was assigned to be part of the character in the edge detection step as seen in (Figure xxxi,c), as well as characters that are cut out too thin account for the most of the plates in Class 3.

After declaring these three classes in the data, the same thing has to be done for the plate rating program. The program results in ratings from 0 to 1000 points, so having the boundaries of the classes at ratings of 333 and 666 respectively intuitively. However, knowing from experience that Class 2 will most likely contain the plates that are the hardest to classify, boundaries on ratings 300 and 700 have been arbitrarily chosen to make Class 2 slightly bigger then the other two.

Having around 50 images in each class the images were tested using the methods proposed in chapter 4, using the two different fitness functions from chapter 5. The first class resembles a plate rating above 700, in class 2 are plates rated between 300 and 700 and all plates rated below 300 are assigned to class 3. To obtain a final plate rating the minimum and the average of the measuring points from the model have been taken and compared. Using 163 different input images the results can be seen in (Table iii).

Jos Nijhuis Model	Class 1	Class 2	Class 3
	78%	0%	48%

Elastic Matching	Class 1	Class 2	Class 3
Fitness function 1, Average	68%	77%	80%
Fitness function 2, Average	100%	89%	97%
Fitness function 1, Minimum	4%	41%	28%
Fitness function 2, Minimum	36%	39%	29%

Centers Method	Class 1	Class 2	Class 3
Fitness function 1, Average	12%	32%	58%
Fitness function 2, Average	61%	55%	74%
Fitness function 1, Minimum	2%	8%	18%
Fitness function 2, Minimum	6%	11%	23%

Edges Method	Class 1	Class 2	Class 3
Fitness function 1, Average	29%	39%	55%
Fitness function 2, Average	73%	66%	86%
Fitness function 1, Minimum	3%	13%	21%
Fitness function 2, Minimum	5%	20%	20%

Table iii: For all approaches in chapter 5 using different setups, the percentage of plates assigned to the correct class is shown.

In (Table iii) for each method and fitness functions can bee seen what percentage of the images that *should* be in a certain class, were actually assigned a rating that would put them into that class by the algorithms.

This test clearly favours the elastic matching algorithm above the Centers and Edges methods. When using fitness function 2, and averaging the separate measurement points this algorithm points the highest percentage of license plates to the user assigned classes. Also it can be seen that when testing on real data the use of the minimum of all separate character ratings is clearly undesirable. The percentage of correctly assigned plates rarely exceeds 40%, which is much too low.

The results from testing on real data are a bit dependant on human decisions. Since all the test images have been rated manually, the classification can be somewhat different when some other person divides the images into classes. One may also choose to create more or less then three different classes. Therefore the use of a perfect license plate on which errors are introduced, is a more favourable way to test the performance of the algorithm.

7.6 Testing on Real Data(2): Relative Testing

A second approach using real data to validate the effectivenes of the algorithms is taking a set of input images and compare their ratings relative to eachother. This way the use of an absolute rating value for each plate can be omitted and relations between separate cases can be expressed in factors. In (Figure xxxii) eight license plates are shown with their relative ratings using the Centers Method in the middle. The values in the center of the figure

Figure xxxiii: Relative plate ratios of eight randomly chosen plates using the Centers Method, fitness function 2 and averaging.

represent the rating as a factor of the plate with the maximum rating, in this case the bottom right one. Hence, to calculate the factor between two plate ratings one only needs to divide them by eachother to obtain the factor.

The figure shows that the top left and bottom right plates are considered best by this method, and the biggest difference is that between the bottom left and right images which are

considered to be a factor five appart. For this relative comparison of the plates the average of all separate measurement point ratings to was used obtain a final result. Similar to the tests done in the previous section the use of the minimum function proved to be much less suited to handle real data, so these results have not been included.

The use of the other methods and fitness functions in these test can be seen in (Figure xxxiv) where the relative values of all methods are shown, taken from the input images from (Figure xxxiii). As can be seen there is a difference between the Edges and Centers Method on the one hand, and the Elastic Matching approach on the other. The maximum factor

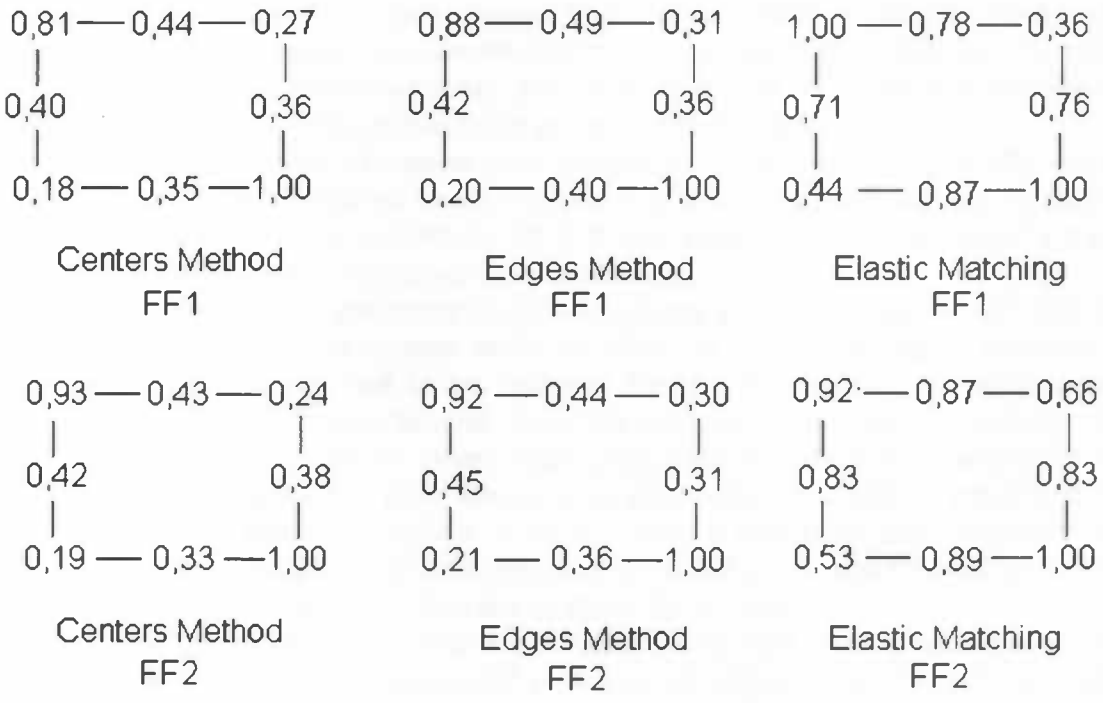


Figure xxxiv: Relative ratings corresponding to the images in Figure xxxiii using different methods and fitness functions.

between good and bad is much lower using Elastic Matching; rarely a factor bigger then 2 is found between a perfect plate and a very poorly recognized one. Both Edges and Centers Methods go up to a factor 5 between the same images; the bottom right image in (Figure xxxiii) is rated 5 times as good as the bottom left one.

Results from relative testing are slightly more reliable since no human interference is neccesary to obtain the restults, in contrast to manually rated plates where the ratings are dependant on the person that rates the license plates.

Chapter 8: Conclusion

The current license plate recognition program uses the Jos Nijhuis model discussed in chapter 5 of this thesis. During the project several new questions were formed that have not yet been solved. As time passed it became more and more important to create a "plate rating" for every input license plate. This proved to be easier said then done.

First, a large number of license plate images had to be analyzed manually and all the variations of errors that occurred had to be put in a few separate categories so they could easily be distinguished from eachother. All different error types were assigned a name, so that they could be referred to troughout the entire thesis.

Since the syntax and geometry checking routines are running separate of each other it is very important to find the correct order in which to handle the input. The error types defined in the first chapter had to be analysed further. By denoting the total number of possibilities in a formula it soon became clear that for some error types it was more efficient to run a syntax check first, and for others running the geometry check first gave better results. In the summary of chapter 3, a table showing the possibility to detect the error types by the syntax and geometry checking module is shown. Since it was know from chapter 2 that the additionals error type is by far the most common, it proved to be most efficient to do a syntax check first and using the data from this test to check the geometry.

Like the error types were charted, the same thing had to be done for the laws and regulations that determine the syntax and geometry of license plates in different countries. Variations in syntax had already been worked out, so only the geometry was of concern. The layout of a license plate has several properties that determine its look . Besides colour, which wasn't an issue, properties like font, plate length, character width and character spacing are the most relevant properties that determined the overall look of the plate. These features were looked into closely, and the relation between the error types in the previous chapter was made in the summary of chapter 4.

The next step was designing a way to test the geometry of the found data. For this purpose a couple of models and algorithms partially based on other research have been applied for geometry matching on license plates. These models all needed to be analysed and tested to find the model best suited for this research. The Jos Nijhuis model that was in use before this research was started was unable to find differences in the goodness of a fit, either the plate matched the model, or it didn't. To incorporate a rating system to determine which match was best and finding the best model to use for this purpose was to be the most important part of this research. Three different approaches were proposed which all have a certain number of measurement points, and the way these methods were expected to handle the variabilities in the license plate regulations from chapter 4.

After a while it became clear that there was a large variation in the errors that occurred on different license plates. To increase performance it was decided to look into these errors, and chart them. The program was run with a large test set of over a thousand license plates, and for every character with a recognition confidence higher then 50% the error was used to plot an error-frequency-histogram for every character. For all three approaches the fitness functions were made and included in the matching program. Having

created a program that can rate the fitness of the measurement points in the license plate model to a satisfying degree, obtaining a final plate rating from the separate ratings was the next point that caused some difficulties. For this a large number of different functions can be used, and depending on which result is more satisfying to the user one or another function can be used for this part of the program. The most interesting ones were the average and minumum functions, which were tested further.

After testing the algorithms proposed in chapter 5, the Edges and Centers methods performed better then elastic matching when characters are moved from their original positions. The effect of a single additional on the license plate rating was much too small when using the average of all measuring points, so in this case taking the minimum of all points is the best way to get an accurate plate rating.

When testing for dimension errors the elastic matching did much better then the other methods. Contrary to character movement here taking the average of all measuring points gave a much better result then taking the minimum. And since small variations in the character dimensions are always present the use of the average seemed the best way to go.

The final test included a number of real data images which were divided into three different classes. The program was run on all of these input images, and the percentage of plates that were assigned to the correct classes wereput in a table in chapter 7. On real data is showed that the elastic matching approach performed best, working with the average of all measuring points and fitness function 2 (FF2).

Another real data test included a comparison between relative plate ratings. This test resulted in a big rating variance when using the Edges en Centers Methods of up to a factor 5, where Elastic Matching only had a maximum of factor 2 difference between a good and a bad recognized plate. Here, the Elastic Matching method shows a small dissadvantage towards the other two methods, since a larger range -or a bigger variance in ratings- makes the process of designing a reliable rating system much easier.

Since in the real data there are dimension errors of some kind on every plate character, the elastic matching approach did the best job in this field. Using fitness functions improved the performance of the recognition significantly, and the best results were achieved when using the elastic matching algorithm using the average of all measurement points to obtain a final plate rating, combined with FF2 type from (Figure xxviii, b).

Chapter 9: References

[Ag88] T. Agui, H. Jin Choiand M. Nakajima, *Method of Extracting Car Number Plates by Image Processing.* Systems and Computers in Japan, Vol. 19, No. 3, 1988. (Translated from Japanese)

[Ba98] D. Báez-López, A.V. González and J.M Ramírez, *Pattern Recognition in Automotive Plates.* Proceedings of the 1998 Midwest Symposium on Systems and curcuits, pages 314-317.

[Bar97] J.Barroso, A. Rafael, E.L. Dagless and J. Bulas-Cruz, *Number Plate Reading using Computer Vision.* IEEE International Symposium on Industrial Electronics ISIE '97, Universidade do Minho, Guimãres, Portugal.

[Bair84] H.S. Baird, *Model-Based Image Matching Using Location.* Ph. D. Thesis, Princeton University, Published by the MIT Press, Cambridge MA, 1984.

[Bi97] A. Del Bimbo and P. Pala, *Visual Image Retrieval by Elastic Matching of User Sketches*, IEEE Transactions on Pattern Recognition and Machine Intelligence 1997. Vol 19, No. 2, pages 121-132.

[Bru] M.H. ter Brugge, J.A.G. Nijhuis, L. Spaanenburg and J.H. Stevens, *License plate recognition.* Department of Computing Science, University of Groningen, The Netherlands.

[Co98] C. Coetzee, C. Botha and D. Webber, *PC Based Plate Recognition System,* IEEE International Symposium on Industrial Electronics, Proceedings ISIE '98, Vol. 2, pages 605-610.

[Fu98] H. Fujiyoshi, T. Umezaki ,T. Imamura and T. Kanade, *Area Extraction of License Plates Using an Artificial Neural Network.* Systems and Computers in Japan, Vol. 29, No. 11.

[Ga99] M. Gavrilov, P. Indyk, R. Motwani, S. Venkatasubramanian, *Geometric Pattern Matching: A Performance Study.* Department of Computer Science, Stanford University, Stanford, California, USA.

[Ge97] S. Gendy, C.L. Smith and S. Lachowicz, *Automatic Car Registration Plate Reading Using Fast Hough Transform.* School of Engineering, Edith Cowan University, Perth, Australia, 1997.

[Hee01] R.P. van Heerden and E.C. Botha, *Optimization of vehicle plate segmentation and symbol recognition.* Proceedings of the 12th annual symposium of the pattern recognition associastion of South Africa, November 29th 2001, pages 83-88.

[Jel01] D. Jelinek and C.J. Taylor, *Reconstruction of Linearly Parametrized Models from Single Images with a Camera of Unknown Focal length.* IEEE Transactions on Pattern analysis and Machine Intelligence, Vol. 23, No. 7, July 2001, Pages 767-773.

[Ka95] V. Kamat and S. Ganesan, *An Efficient Implementation of the Hough transform for Detecting Vehicle License Plates Using DSP's.* Proceedings of the real-time Technology and Applications Symposium (RTAS'95).

[Kim02] D. Kim, I. Choi and S. Chen, *Correction and Extraction of Perspectivel Distorted License Plates Using Scal Line Based Generalized Symmetry Transform.* IEICE Transactions, Information and System Technology, Vol. E85-D, No. 11, pages 1776-1783.

[Lee02] S. Lee, Y. Seok and E. Lee, *Multi-National Car License Plate Recognition System Using Geometrical Features and Hybris Pattern Vectors.* International Technical conference on circuits/systems, compilers and communications, Phuket, Thailand, 2002.

[Lu95] Y. Lu, *Machine Printed Character Segmentation.* Pattern recognition, Elsevier Science Vol. 28, No. 1, pages 67-80.

[Mi99] K. Miyamoto, M. Tamagawa, I. Fujita, Y. Hayama and S. Eiho, *Extraction of Character String Region by a Correlation Method.* Systems and Computers in Japan, Vol. 30, No. 14, pages 43-52.

[Na99] T. Naito, T. Tsukada, K. Yamada, K. Kozuka and S. Yamamoto, *Lisence plate recognition method for license plates outdoors.* Proceedings of the 1999 International Conference on Information Intelligence and Systems, pages 304-312.

[Nijh95] J.A.G Nijhuis, M.H. ter Brugge, K.A. Helmholt, J.P.W. Pluim, L. Spaanenburg, R.S Venema and M.A. Westenberg, *Car License Plate Recognition with Neural Networks and Fuzzy Logic.* IEEE International conference on Neural Networks, 1995, Vol. 5, pages 2232-2236.

[Nijh97] J.A.G Nijhuis, M.H. ter Brugge, K. Kok and L. Spaanenburg, *Handling Unseen Data in a Neural License Plate Recognition System.* EUFIT '97 - 5th European Congress on Intelligent Techniques and Soft Computing, Vol. 1, pages 380-383.

[Nijh02] J.A.G Nijhuis, A. Broersma and L. Spaanenburg, *A modular neural network classifier for the recognition of occluded characters in automatic license plate reading.* Proceedings of the 5th International FLINS conference, Gent, Belgium, September 2002, pages 363-372

[Pa96] J.R. Parker and P. Federl, *An approach to licenseplate recognition..* Laboratories of Computer vision and Computer graphics, University of Calgary, Canada.

[Par99] S.H. Park, K.I. Kim, K Jung and H.J. Kim, *Locating Car License Plates using Neural Networks.* Electronics Letters, Vol. 35, No. 17, 1999, pages 1475-1477.

[Ro02] A. da Rocha Gesualdi, J.M. de Seixas, Marcelo Portes de Albuquerque and Márcio Portes de Albuquerque, *Character Recognition in Car License Plates Based on Principal Components and Neural Processing.* Proceedings of the Brazilian Symposium on Neural Networks (SBRN'02), pages 206-211.

[Rod00] F.M. Rodriguez, X.F. Hermida, *New Advances in Automatic Reading of Vehicle License Plates.* Proceedings of the 2000 Signal Processing and Communications Conference, Marbella, Spain.

[Rod02] F.M. Rodriguez, M. Garcia and J.L. Alba, *New Methods for Automatic Reading of Vehicle License Plates.*Proceedings of the SPPRA-2002, Heraklion, Greece.

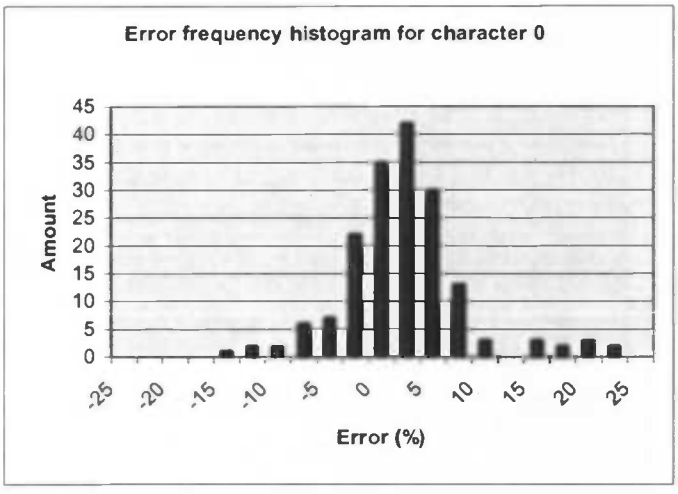
[Si98] T. Sirithinaphong and K. Chamongthai, *Extracting of Car License Plate Using Motor Vehicle Regulation and Character Pattern Recognition.* Proceedings of the Asia Pacific Conference on Circuits and Systems, 1998 (APCCAS'98), pages 559-562.

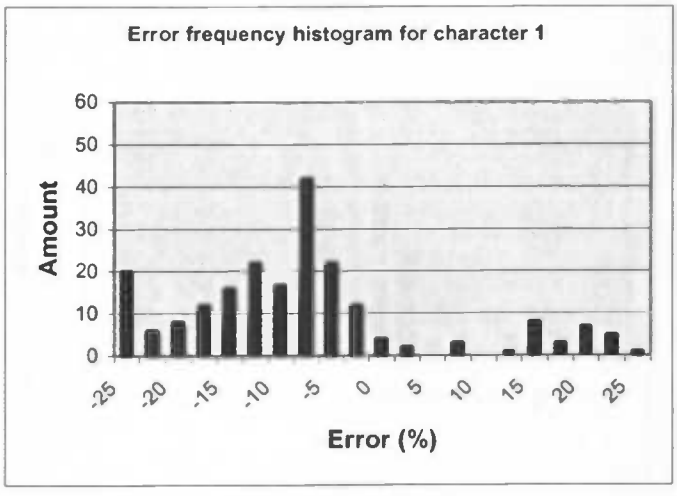
[Ty99] J.W. Tyan, C.Neubauer and Lj.Goganovic, *A Character Segmentation Algorithm for Recognition of Vehicle License Plates.* SPIE Conference on Mobile Robots XIV, Boston MA, 1999.

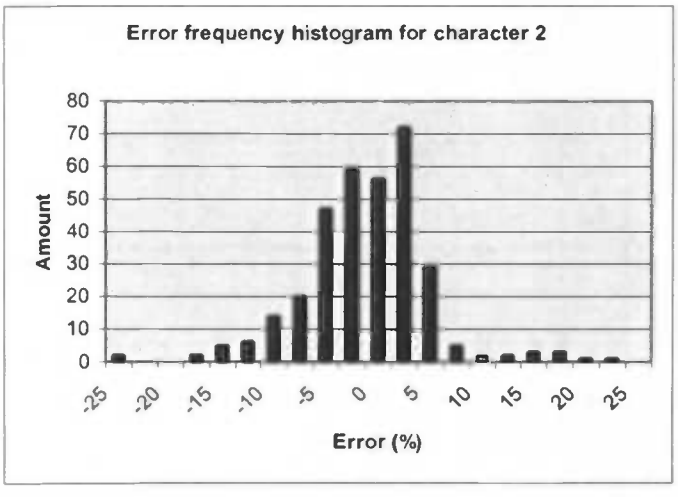
[Yu99] N.H.C. Yung, K.H.Au and A.H.S. Lai, *Recognition of Vehicle Mark on Moving Vehicles in an Outdoor Environment.* IEEE International conference on Intelligent Transportation Systems, Tokio, Japan, 1999.

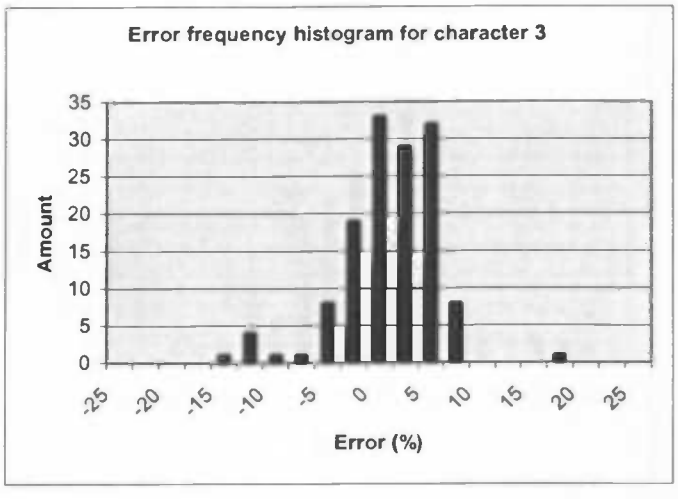
Appendices

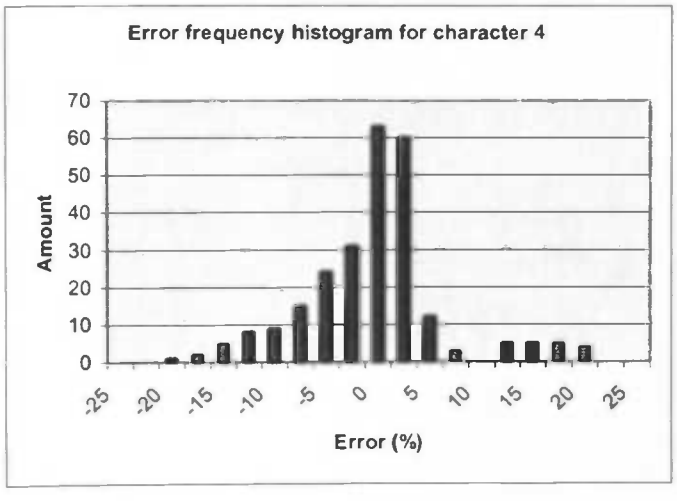
Appendix A : Error Frequency Histograms [Elastic Matching]

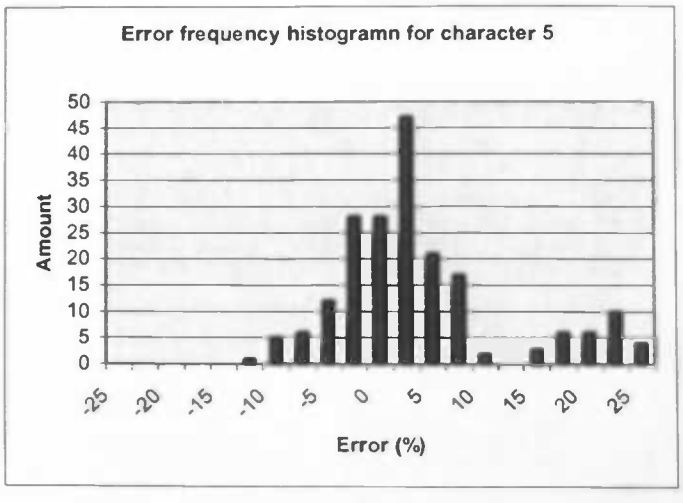


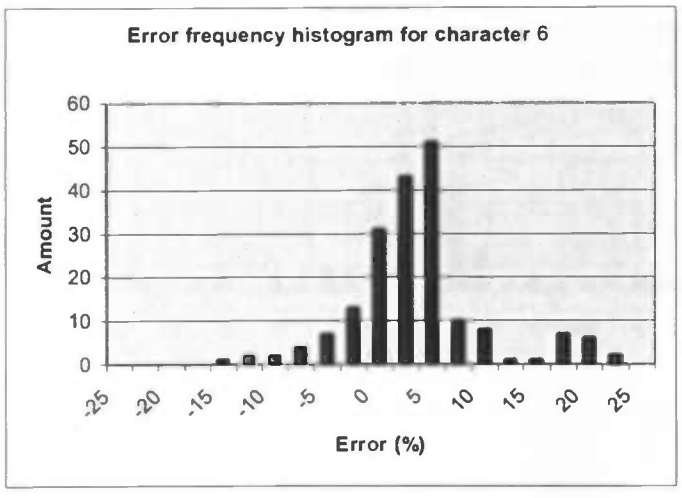


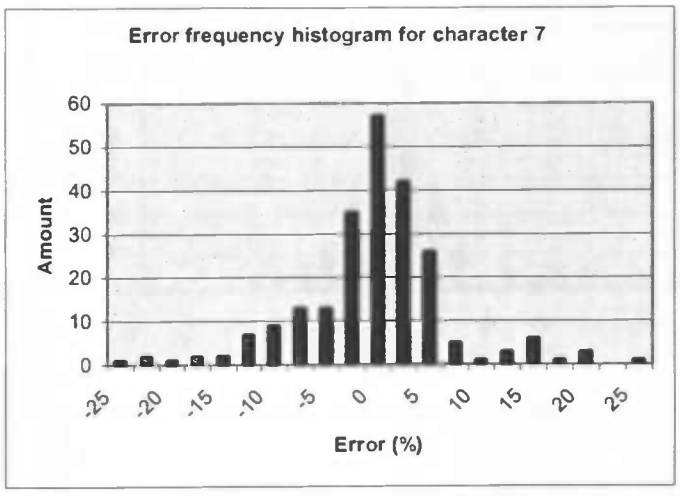


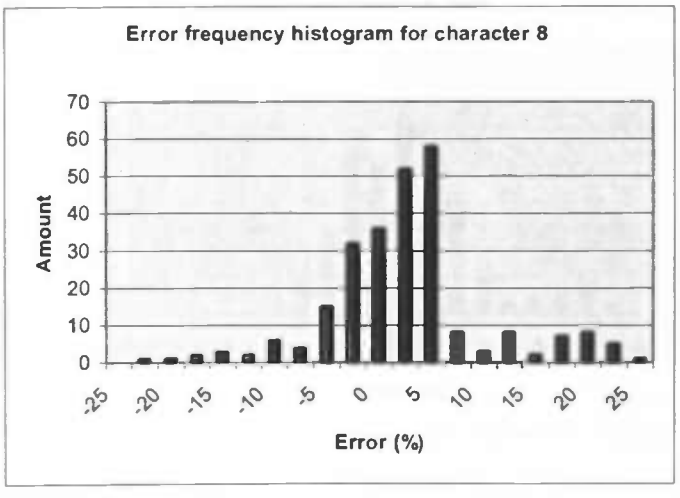


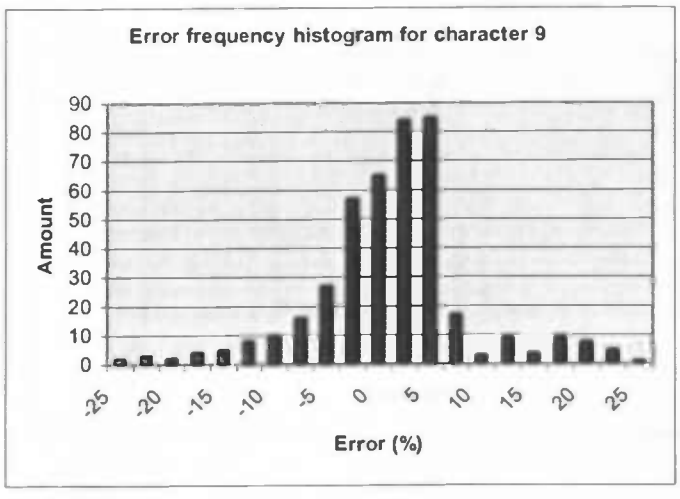


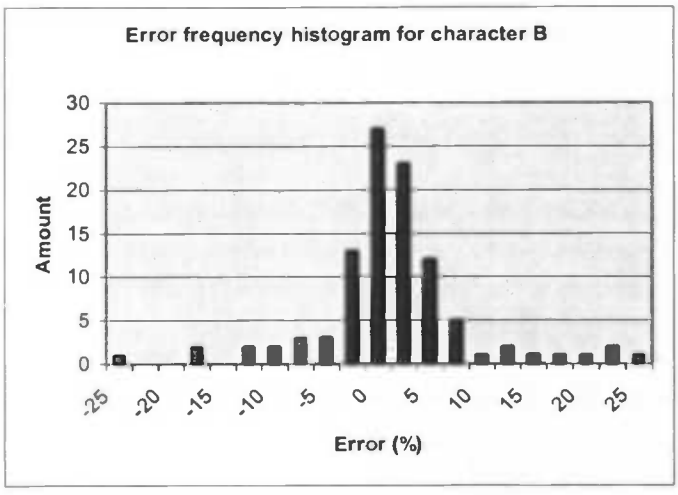


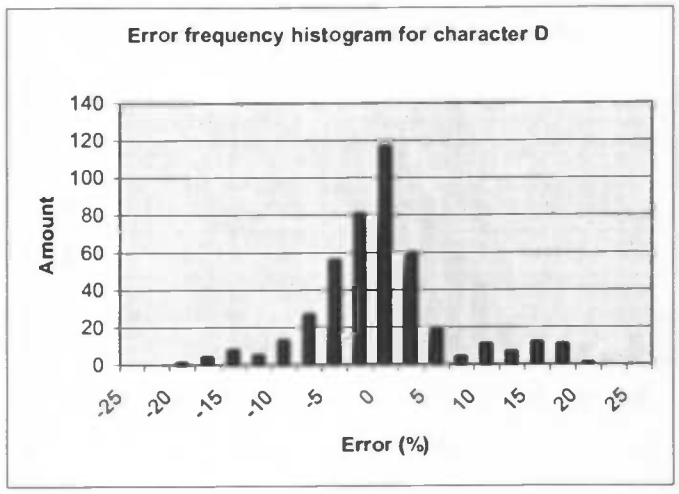

Error frequency histogram for character 6
Amount
60
50
40
30
20
10
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

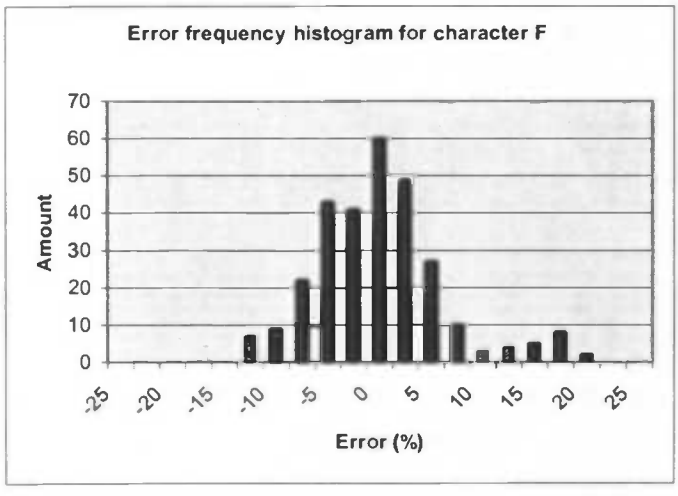

Error frequency histogram for character 7
Amount
60
50
40
30
20
10
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

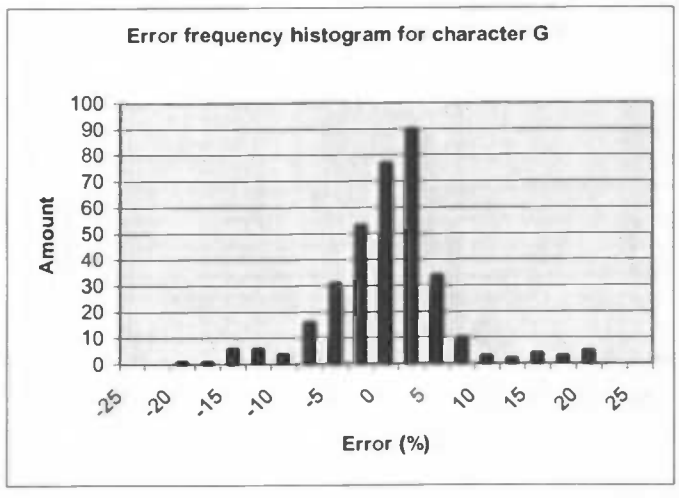

Error frequency histogram for character 8
Amount
70
60
50
40
30
20
10
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

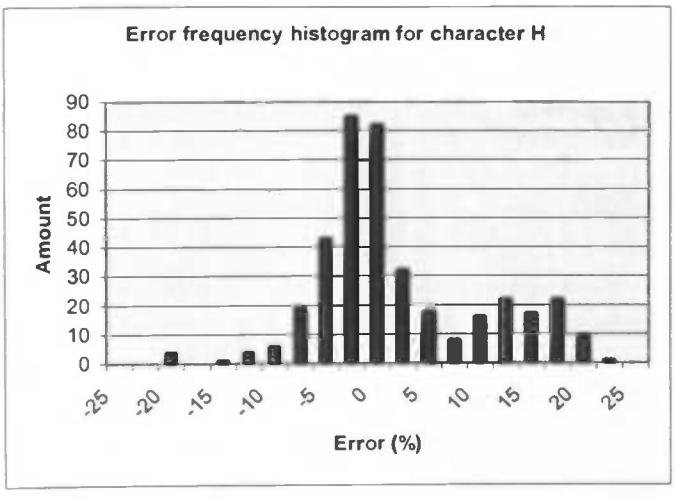

Error frequency histogram for character 9
Amount
90
80
70
60
50
40
30
20
10
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

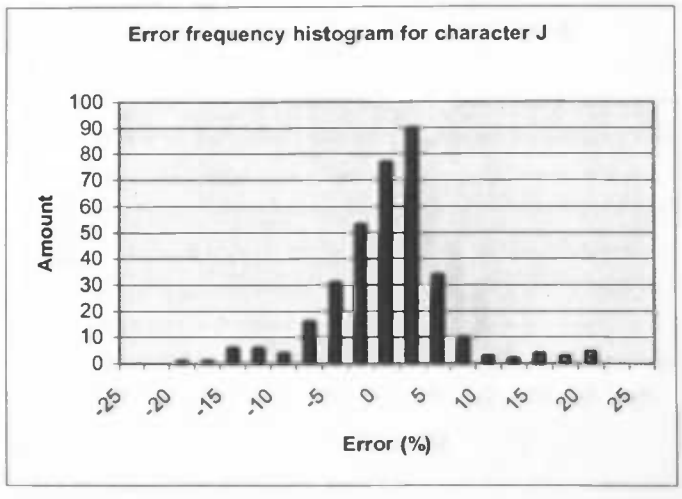

Error frequency histogram for character B
Amount
30
25
20
15
10
5
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

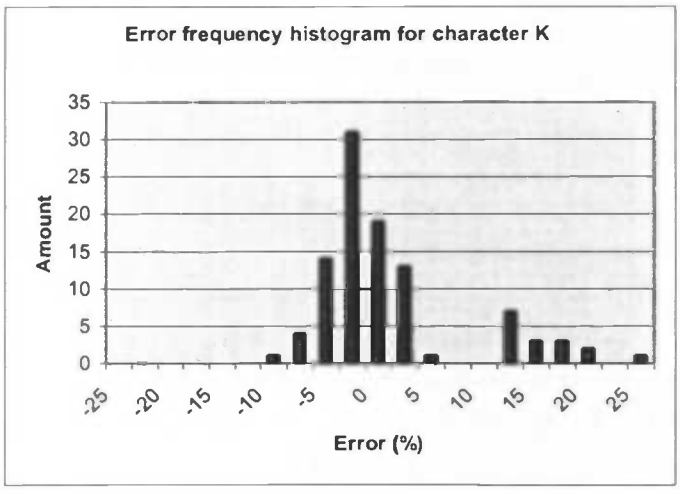

Error frequency histogram for character D
Amount
140
120
100
80
60
40
20
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

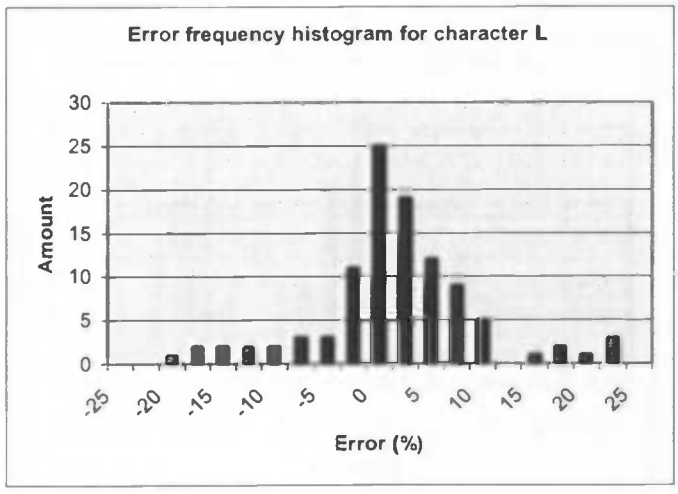

Error frequency histogram for character F
Amount
70
60
50
40
30
20
10
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

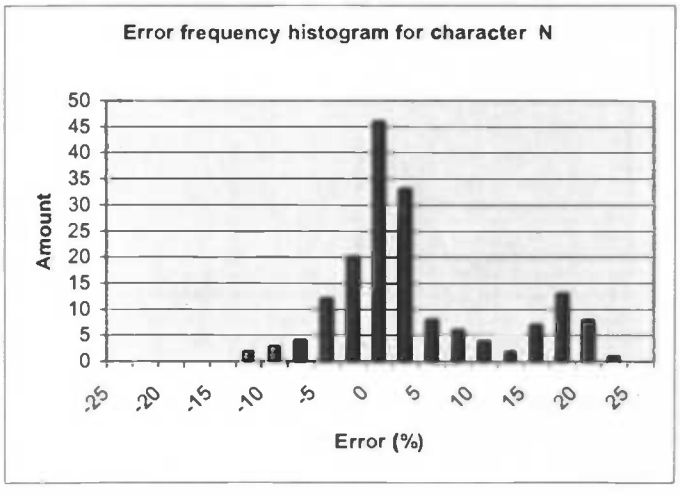

Error frequency histogram for character G
Amount
100
90
80
70
60
50
40
30
20
10
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

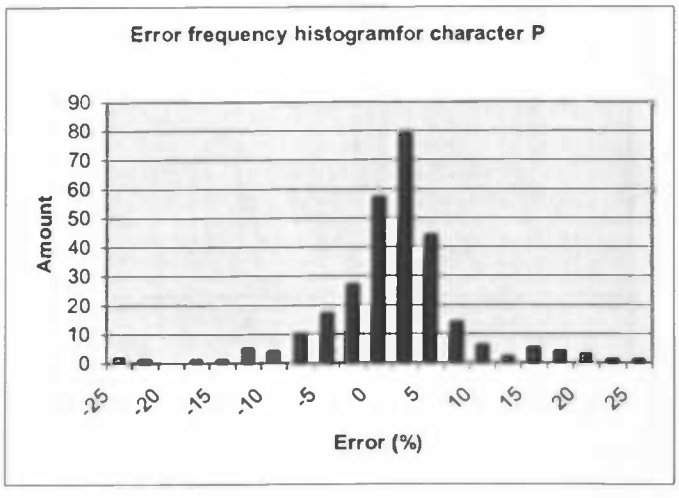

Error frequency histogram for character H
Amount
90
80
70
60
50
40
30
20
10
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

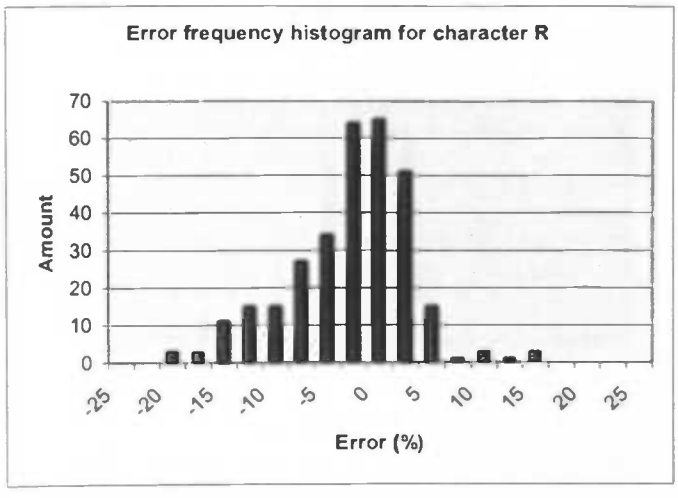

Error frequency histogram for character J
Amount
100
90
80
70
60
50
40
30
20
10
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

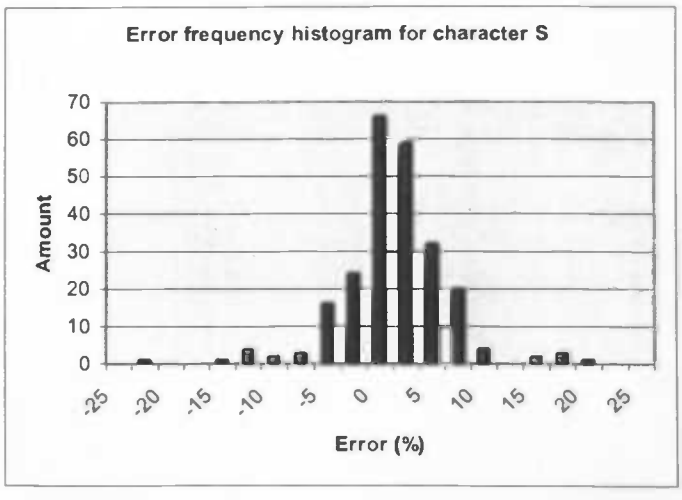

Error frequency histogram for character K
Amount
Error (%)

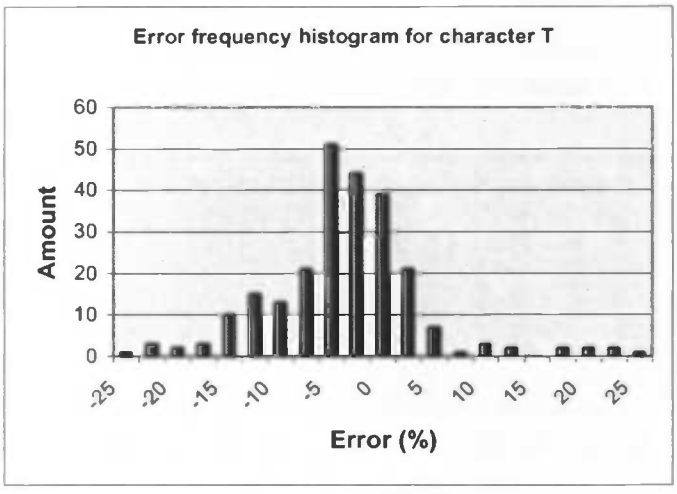

Error frequency histogram for character L
Amount
Error (%)

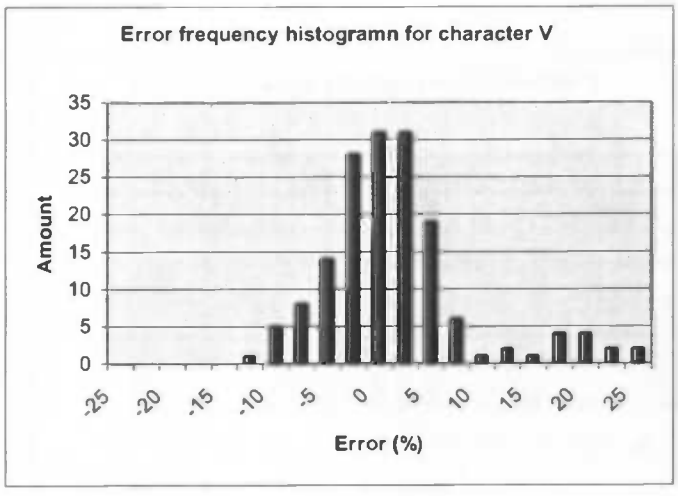

Error frequency histogram for character N
Amount
Error (%)

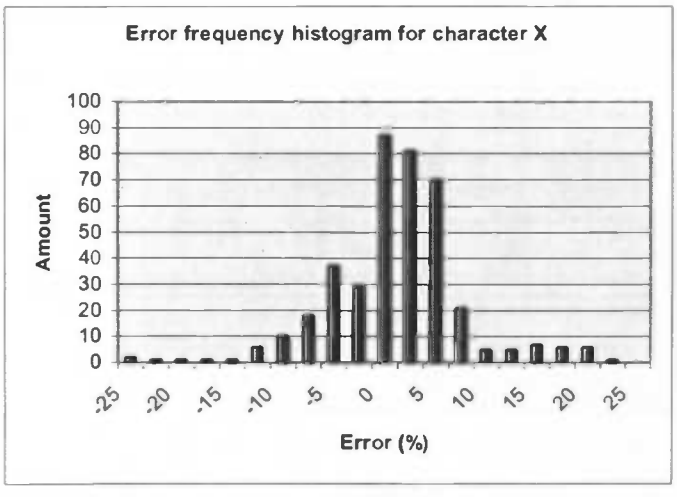

Error frequency histogramfor character P
Amount
Error (%)

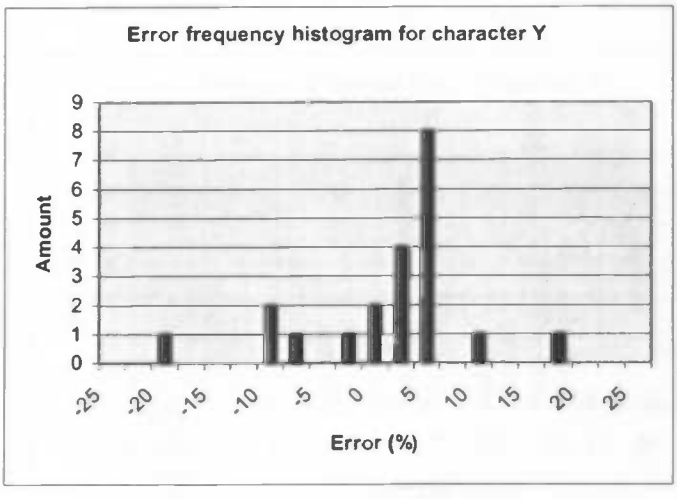

Error frequency histogram for character R
Amount
Error (%)

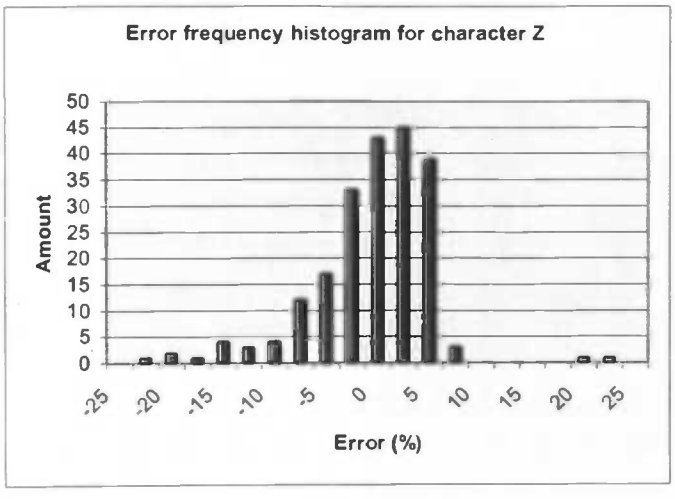

Error frequency histogram for character S
Amount
Error (%)


Error frequency histogram for character T
Amount
60
50
40
30
20
10
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

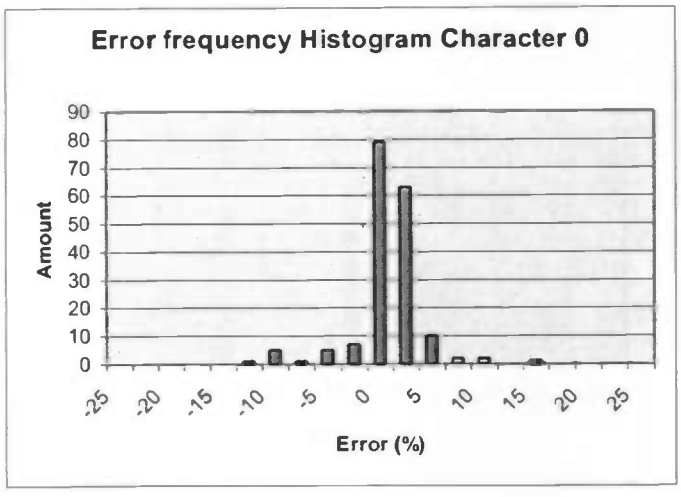

Error frequency histogramn for character V
Amount
35
30
25
20
15
10
5
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

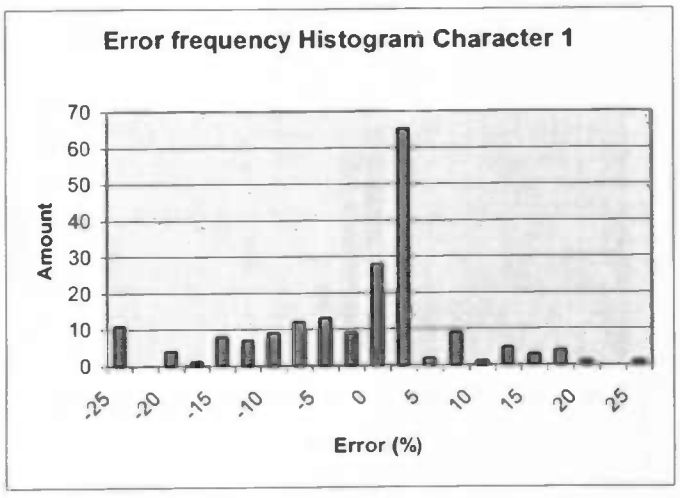

Error frequency histogram for character X
Amount
100
90
80
70
60
50
40
30
20
10
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

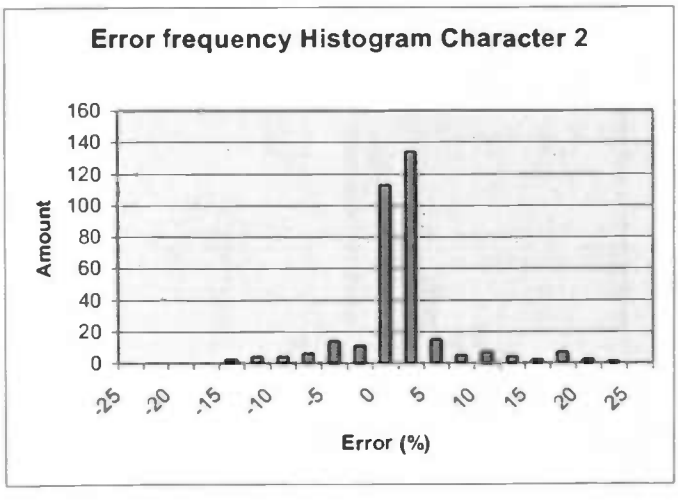

Error frequency histogram for character Y
Amount
9
8
7
6
5
4
3
2
1
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

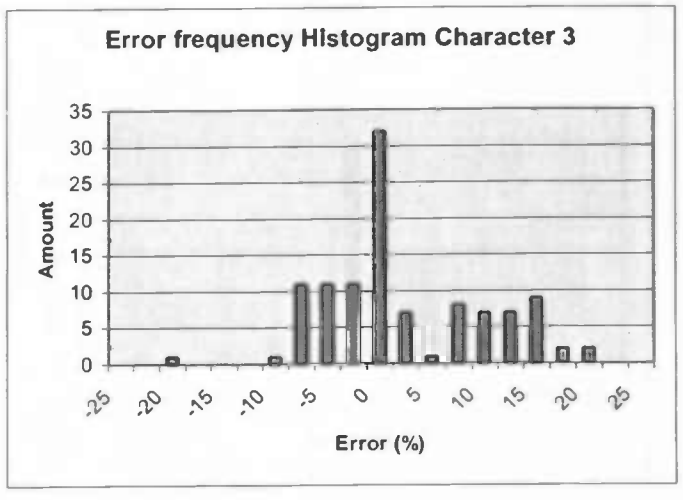

Error frequency histogram for character Z
Amount
50
45
40
35
30
25
20
15
10
5
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

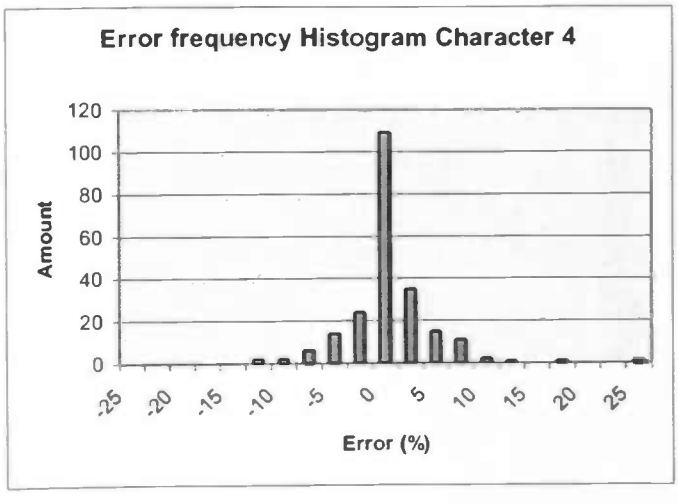
Appendix B : Error Frequency Histograms [Centers Method]

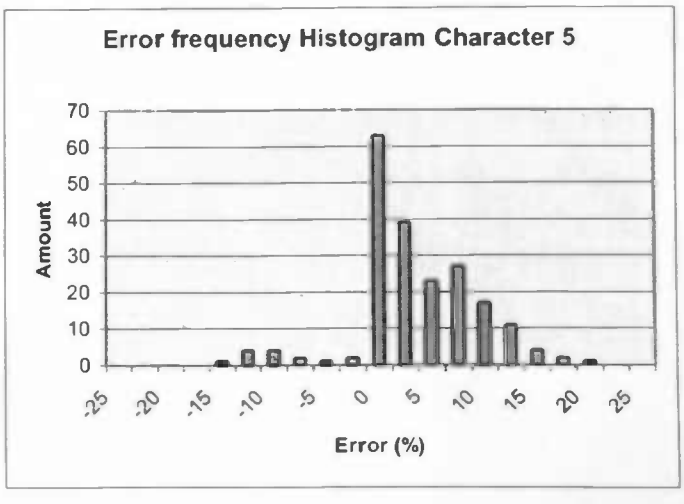


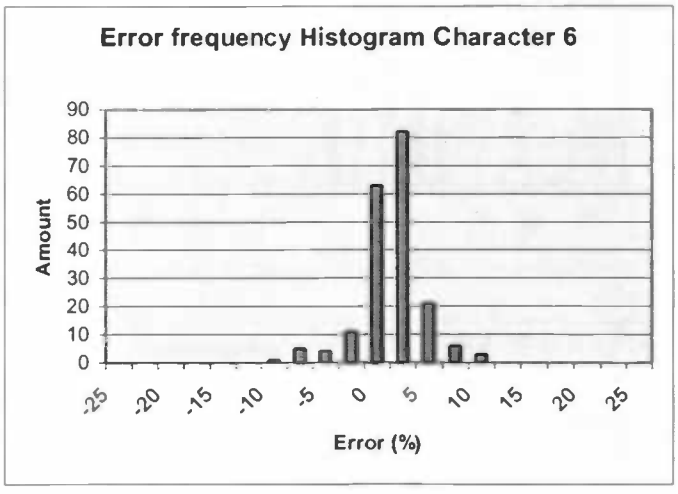


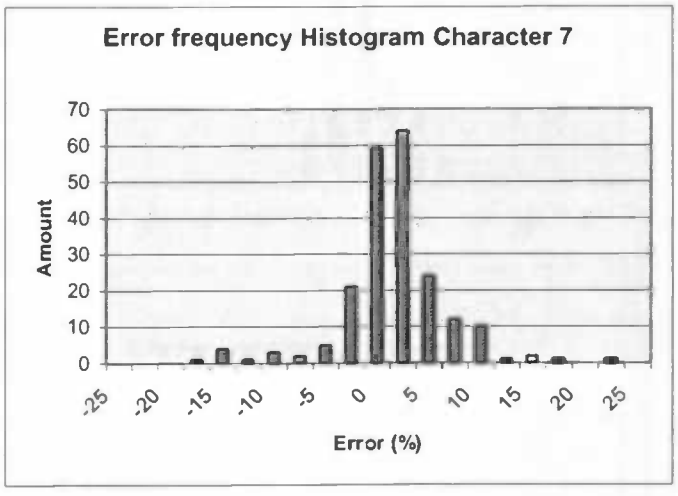


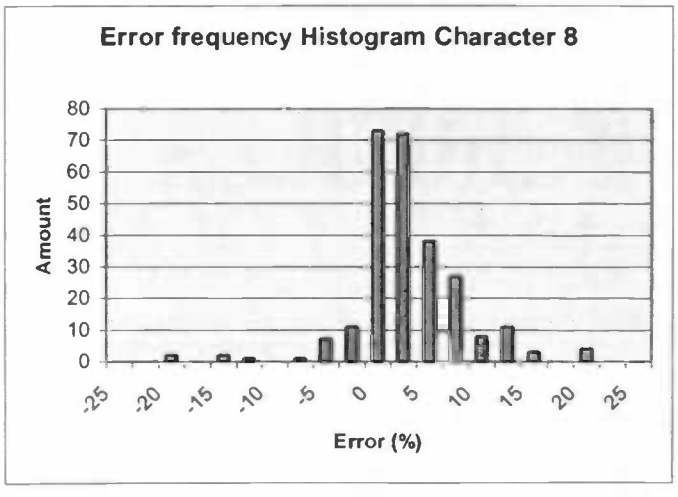


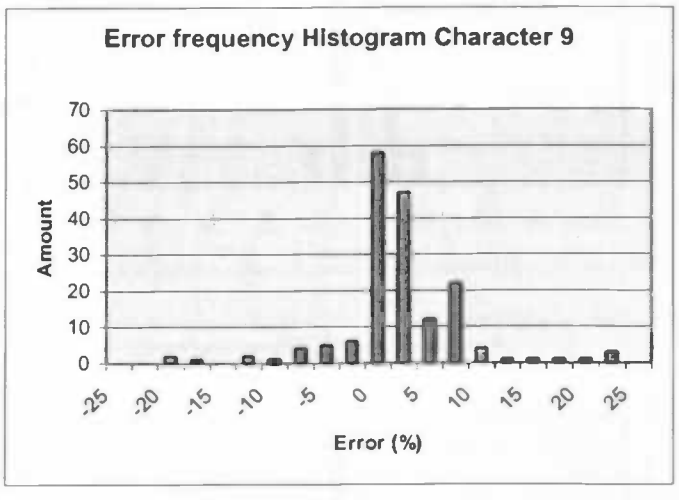


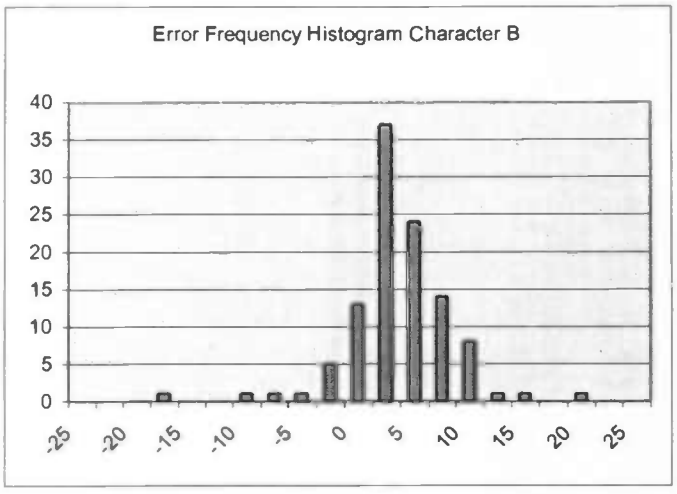


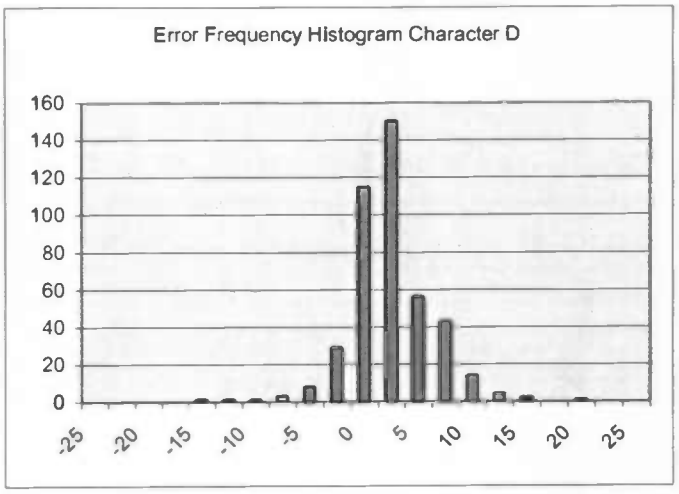

Error frequency Histogram Character 6
Amount
90
80
70
60
50
40
30
20
10
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

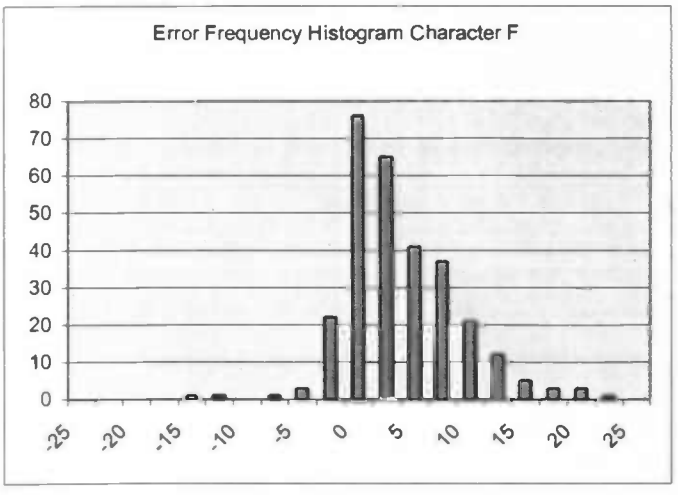

Error frequency Histogram Character 7
Amount
70
60
50
40
30
20
10
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

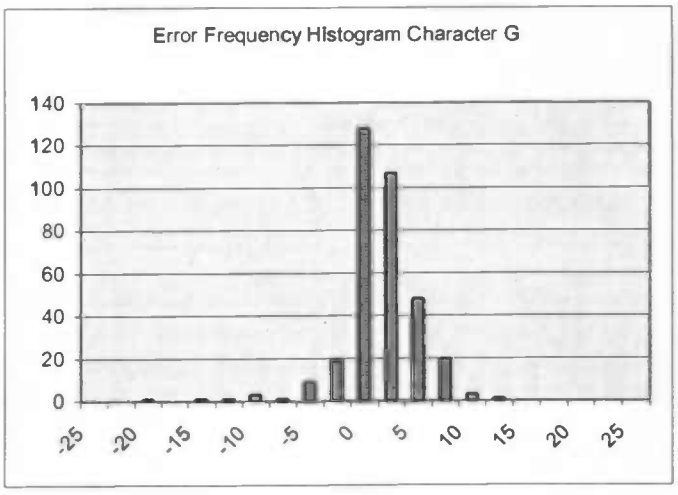

Error frequency Histogram Character 8
Amount
80
70
60
50
40
30
20
10
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

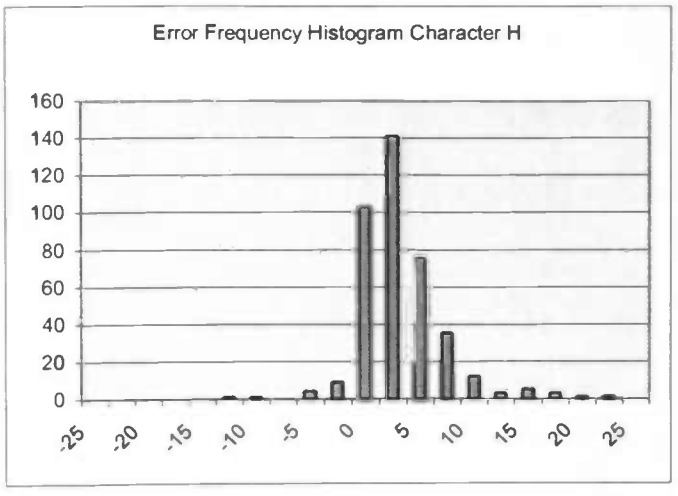

Error frequency Histogram Character 9
Amount
70
60
50
40
30
20
10
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

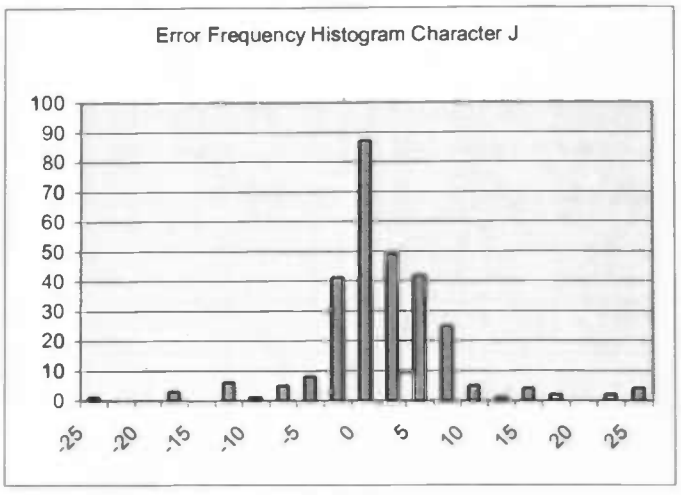

Error Frequency Histogram Character B
40
35
30
25
20
15
10
5
0
-25 -20 -15 -10 -5 0 5 10 15 20 25

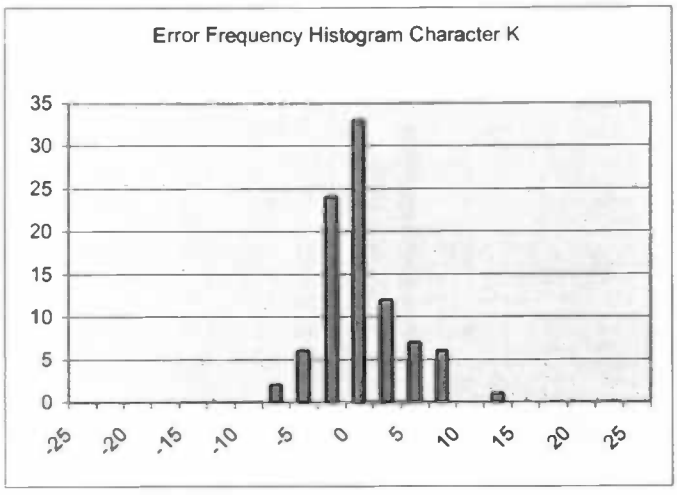

Error Frequency Histogram Character D
160
140
120
100
80
60
40
20
0
-25 -20 -15 -10 -5 0 5 10 15 20 25

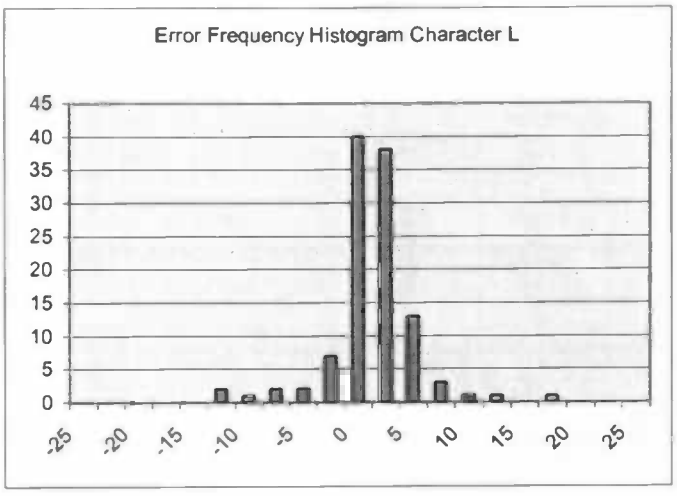

Error Frequency Histogram Character F
80
70
60
50
40
30
20
10
0
-25 -20 -15 -10 -5 0 5 10 15 20 25

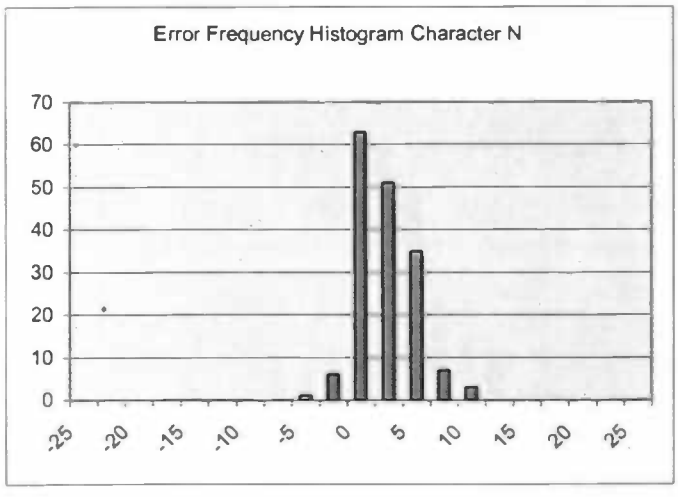

Error Frequency Histogram Character G
140
120
100
80
60
40
20
0
-25 -20 -15 -10 -5 0 5 10 15 20 25

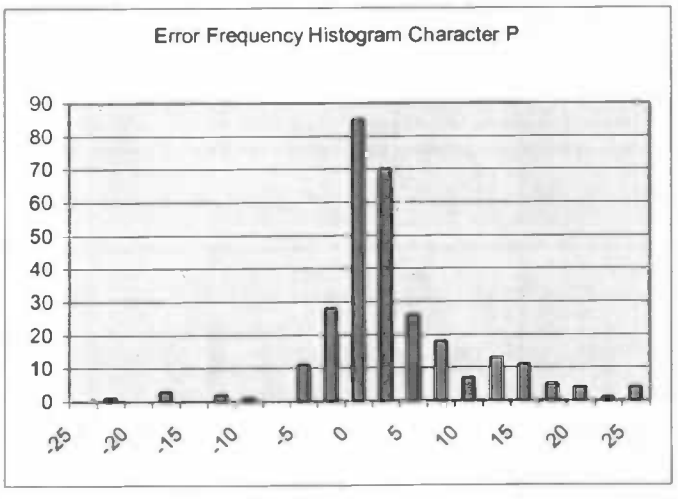

Error Frequency Histogram Character H
160
140
120
100
80
60
40
20
0
-25 -20 -15 -10 -5 0 5 10 15 20 25

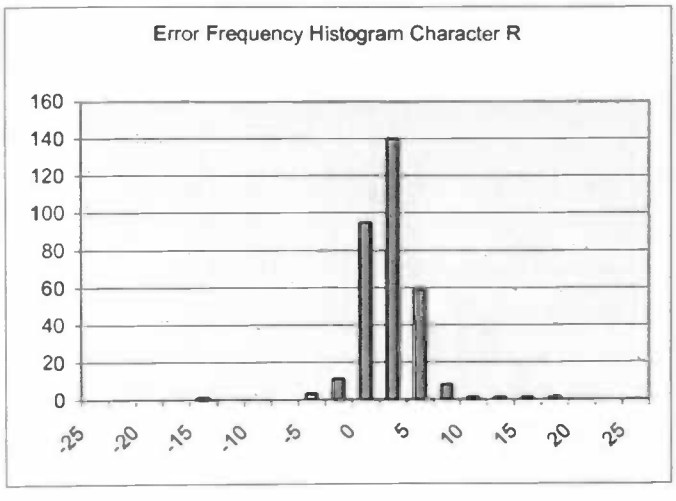

Error Frequency Histogram Character J
100
90
80
70
60
50
40
30
20
10
0
-25 -20 -15 -10 -5 0 5 10 15 20 25

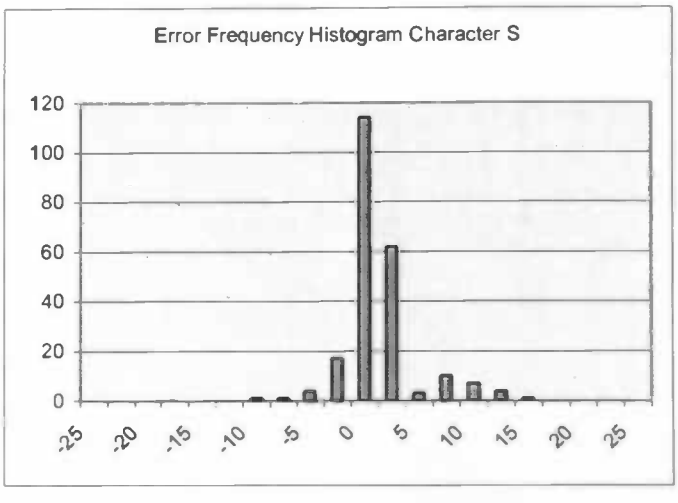

Error Frequency Histogram Character K
35
30
25
20
15
10
5
0
-25 -20 -15 -10 -5 0 5 10 15 20 25

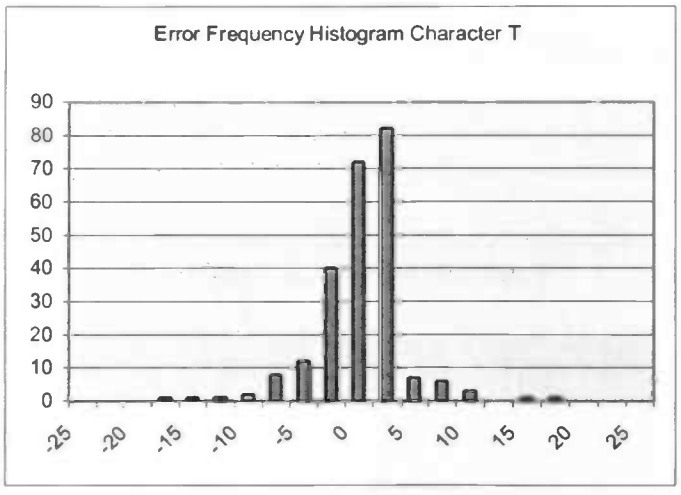

Error Frequency Histogram Character L
45
40
35
30
25
20
15
10
5
0
-25 -20 -15 -10 -5 0 5 10 15 20 25

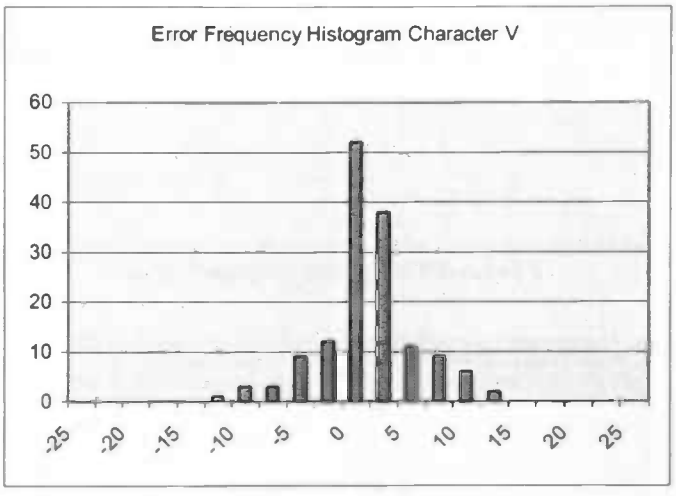

Error Frequency Histogram Character N
70
60
50
40
30
20
10
0
-25 -20 -15 -10 -5 0 5 10 15 20 25

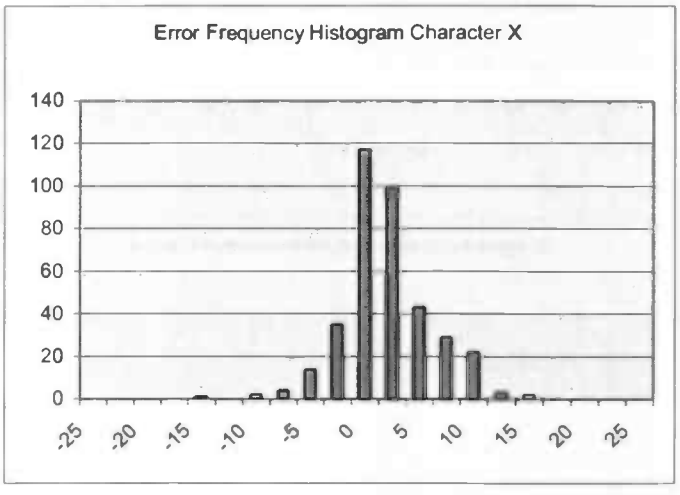

Error Frequency Histogram Character P
90
80
70
60
50
40
30
20
10
0
-25 -20 -15 -10 -5 0 5 10 15 20 25

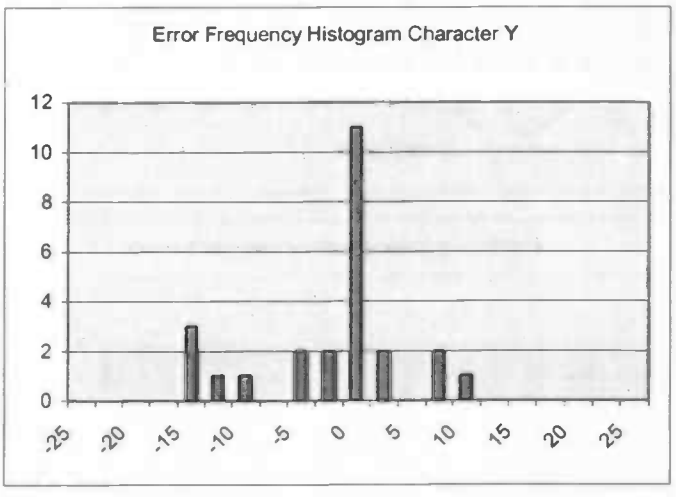

Error Frequency Histogram Character R
160
140
120
100
80
60
40
20
0
-25 -20 -15 -10 -5 0 5 10 15 20 25

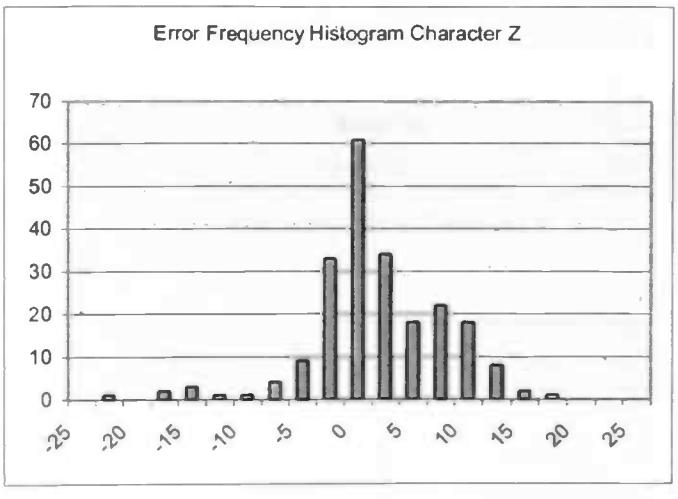

Error Frequency Histogram Character S
120
100
80
60
40
20
0
-25 -20 -15 -10 -5 0 5 10 15 20 25


Error Frequency Histogram Character T
90
80
70
60
50
40
30
20
10
0
-25 -20 -15 -10 -5 0 5 10 15 20 25

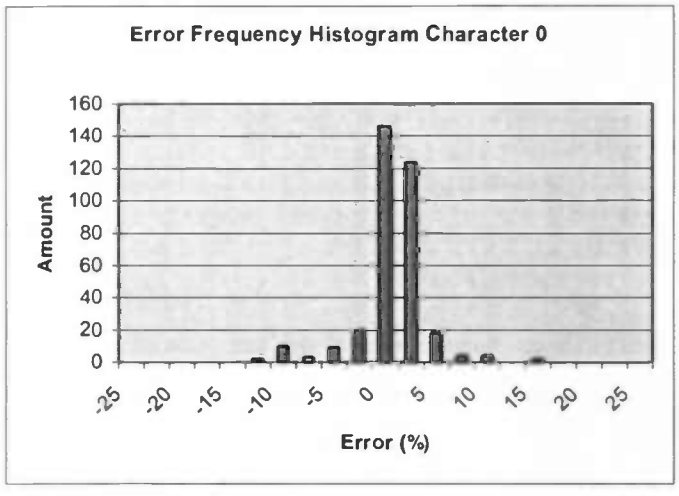

Error Frequency Histogram Character V
60
50
40
30
20
10
0
-25 -20 -15 -10 -5 0 5 10 15 20 25

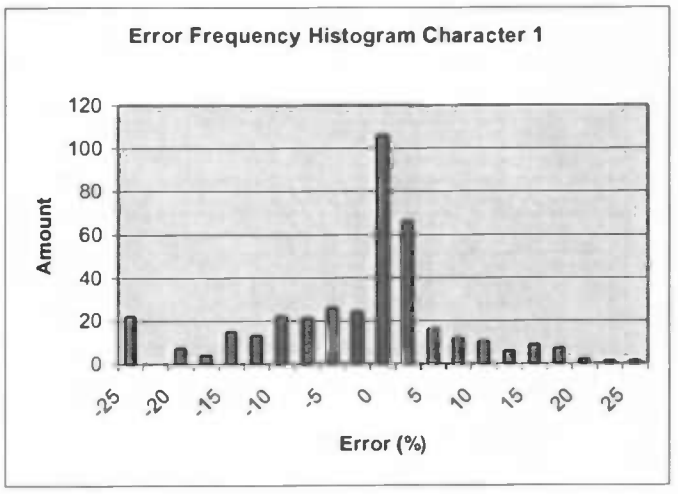

Error Frequency Histogram Character X
140
120
100
80
60
40
20
0
-25 -20 -15 -10 -5 0 5 10 15 20 25

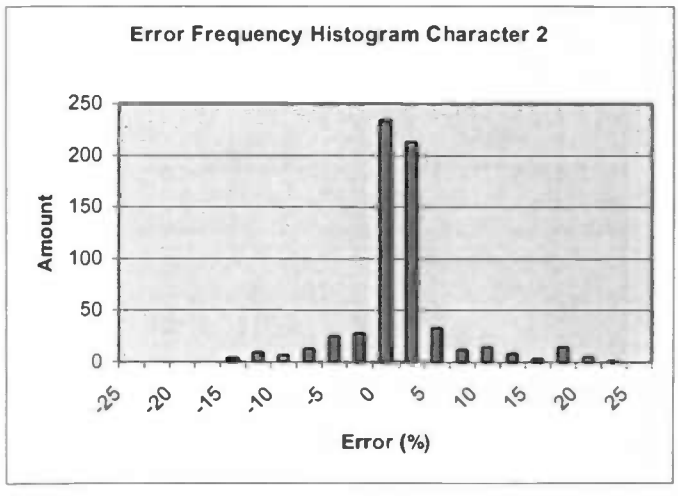

Error Frequency Histogram Character Y
12
10
8
6
4
2
0
-25 -20 -15 -10 -5 0 5 10 15 20 25

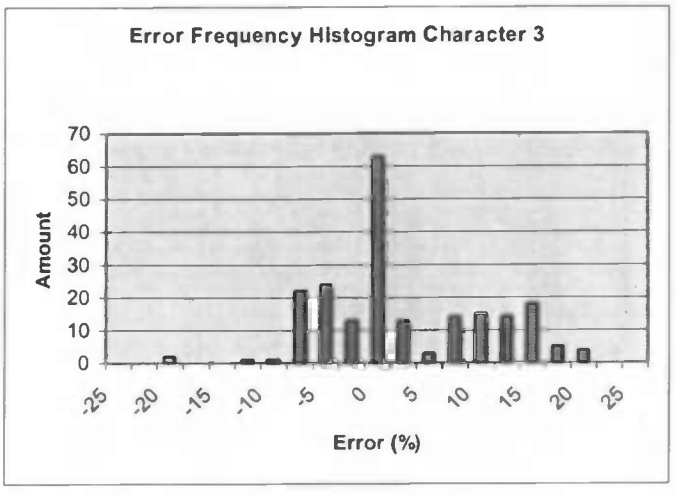

Error Frequency Histogram Character Z
70
60
50
40
30
20
10
0
-25 -20 -15 -10 -5 0 5 10 15 20 25

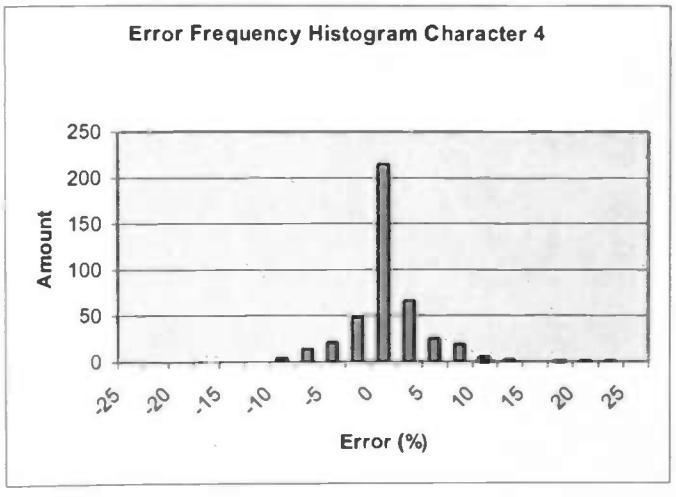
Appendix C : Error Frequency Histograms [Edges Method]

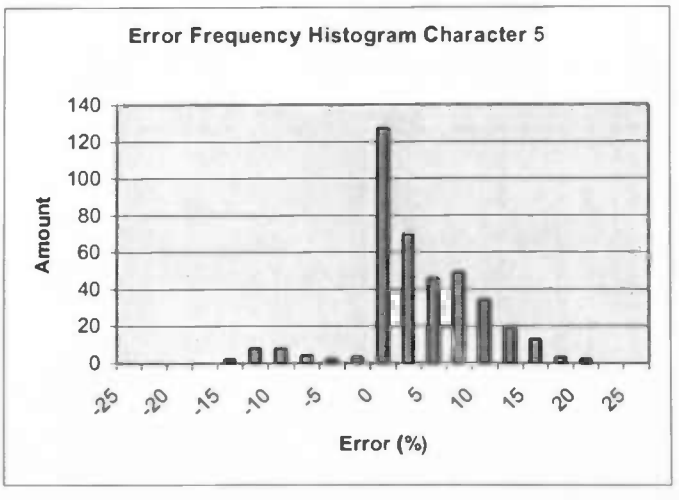


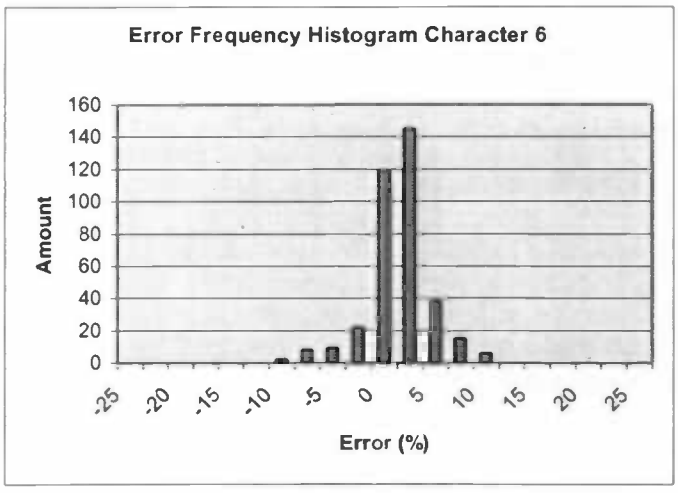


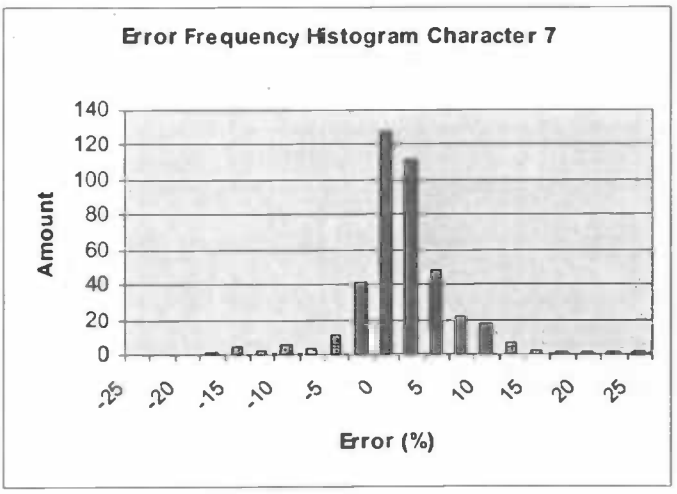


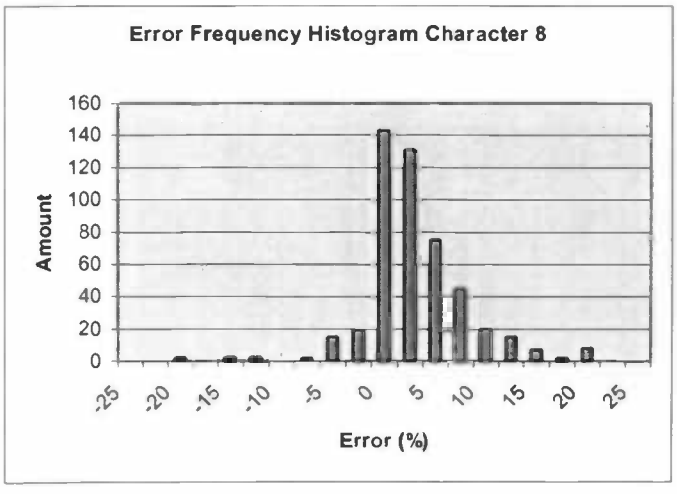


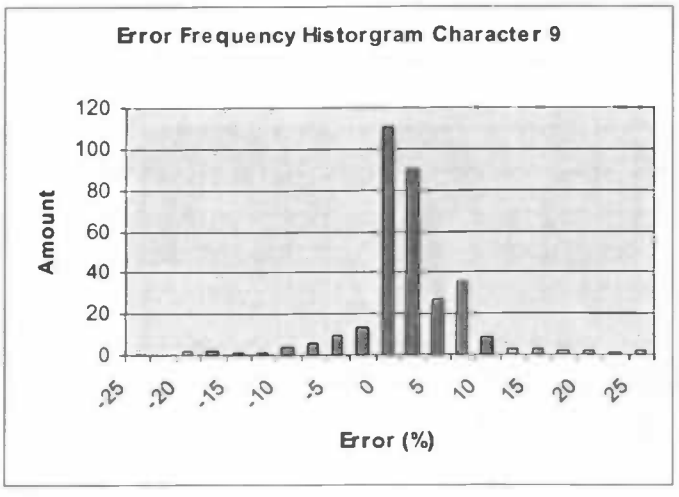


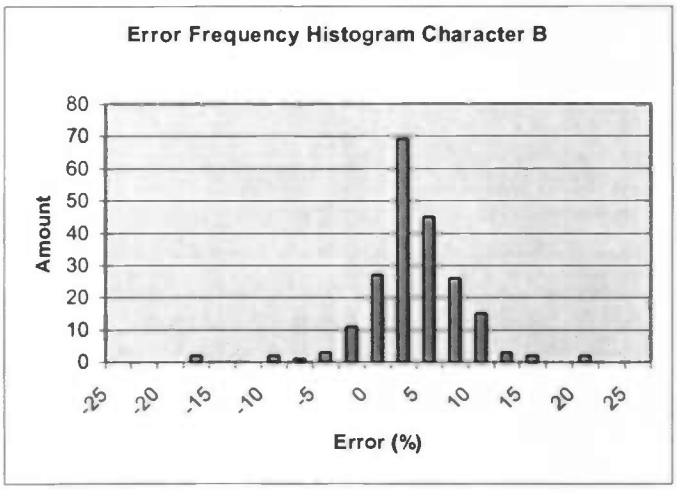


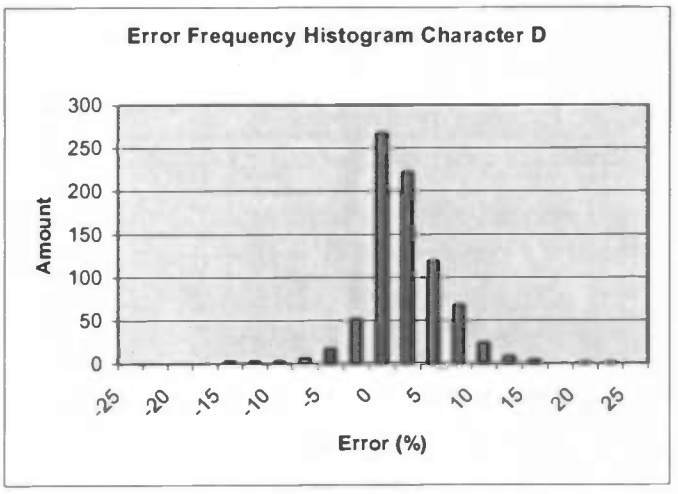

Error Frequency Histogram Character 6
Amount
160
140
120
100
80
60
40
20
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

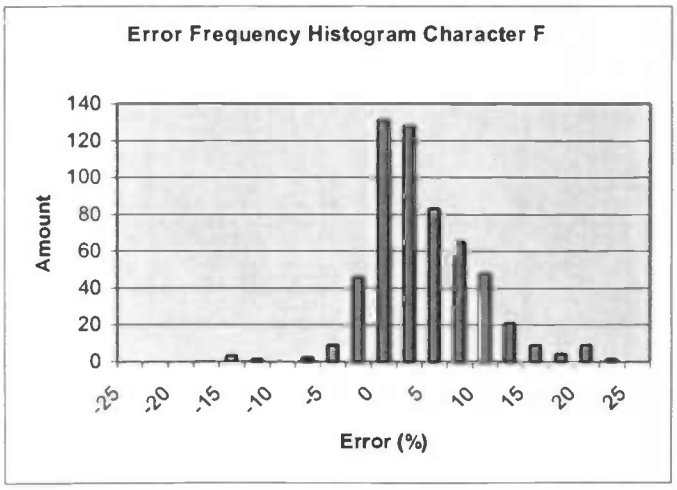

Error Frequency Histogram Character 7
Amount
140
120
100
80
60
40
20
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

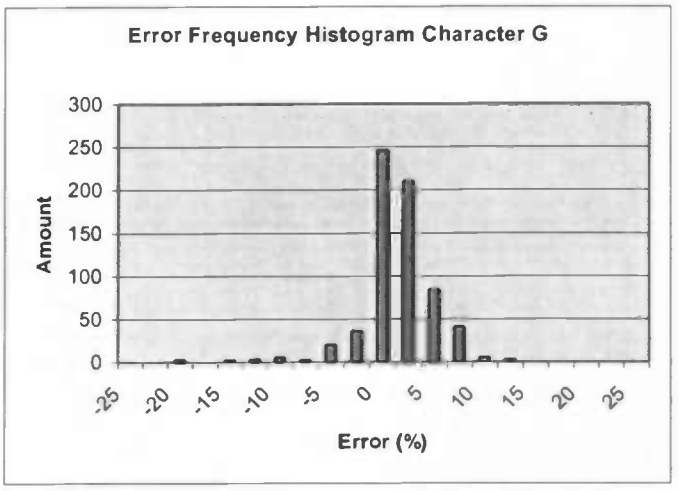

Error Frequency Histogram Character 8
Amount
160
140
120
100
80
60
40
20
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

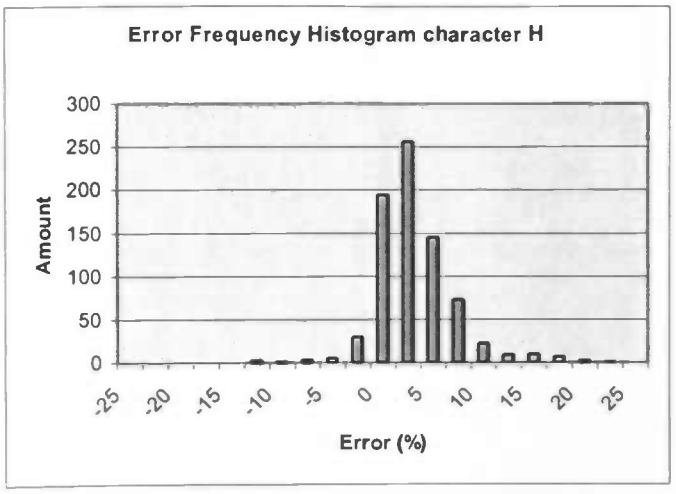

Error Frequency Historgram Character 9
Amount
120
100
80
60
40
20
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

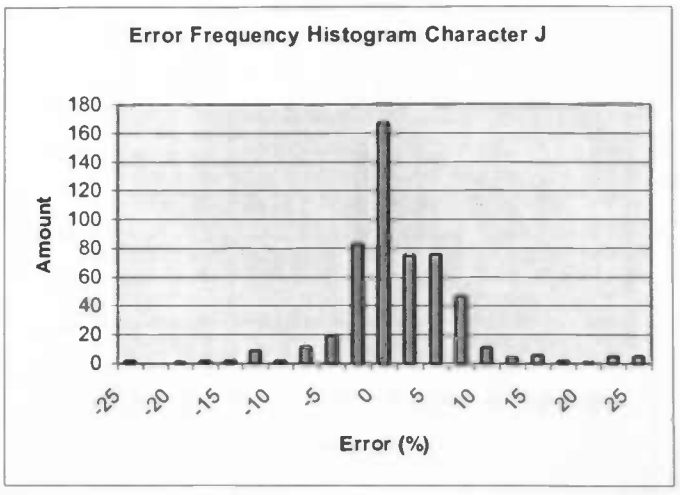

Error Frequency Histogram Character B
Amount
80
70
60
50
40
30
20
10
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

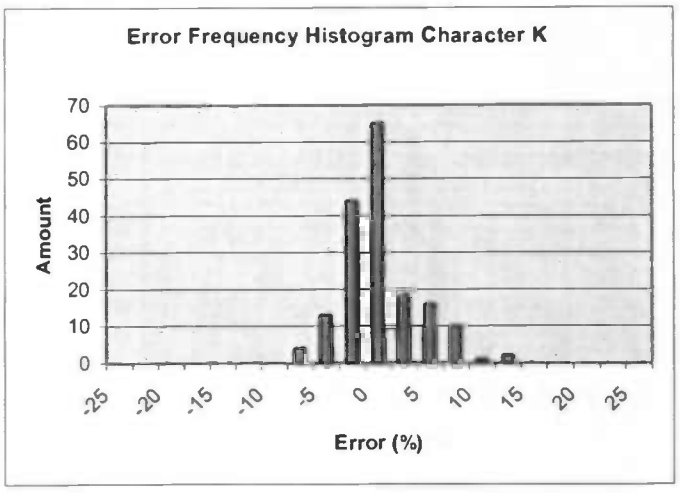

Error Frequency Histogram Character D
Amount
300
250
200
150
100
50
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

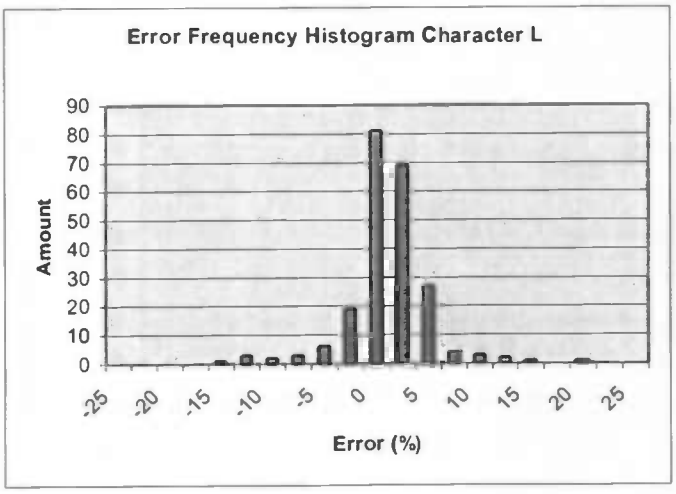

Error Frequency Histogram Character F
Amount
140
120
100
80
60
40
20
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

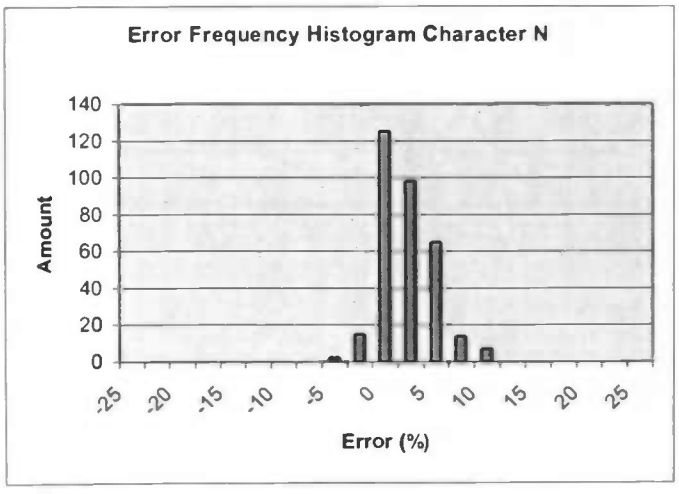

Error Frequency Histogram Character G
Amount
300
250
200
150
100
50
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

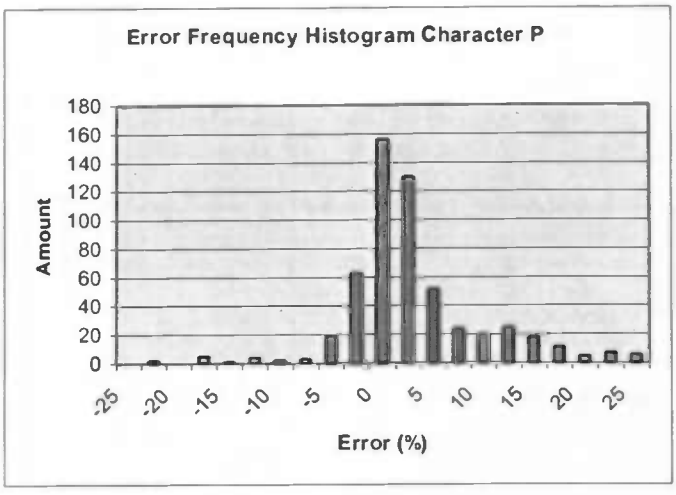

Error Frequency Histogram character H
Amount
300
250
200
150
100
50
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

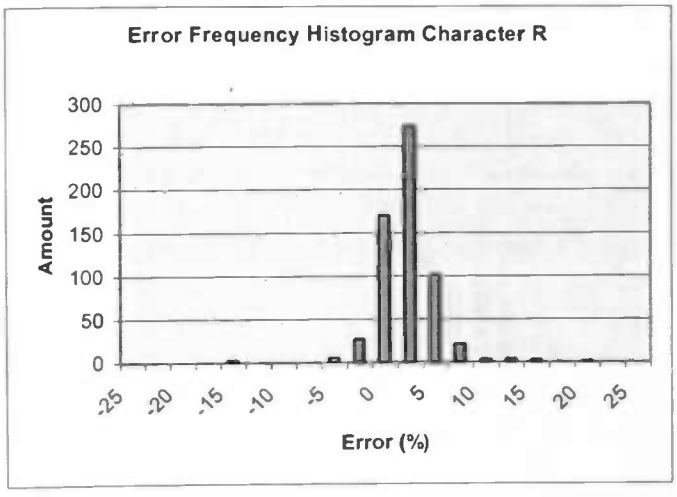

Error Frequency Histogram Character J
Amount
180
160
140
120
100
80
60
40
20
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

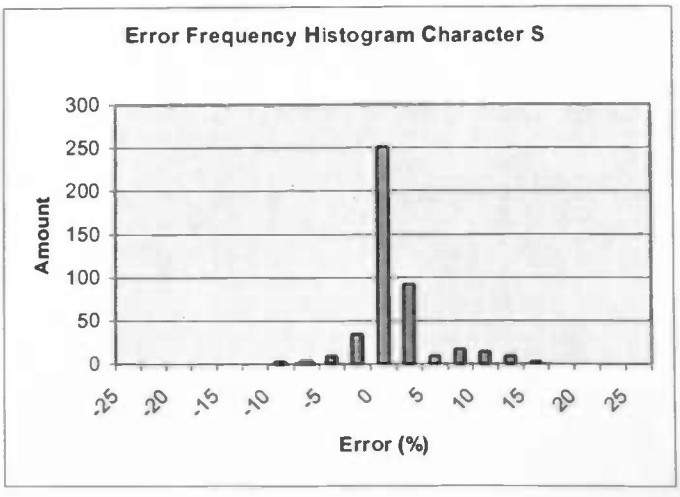

Error Frequency Histogram Character K
Amount
70
60
50
40
30
20
10
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

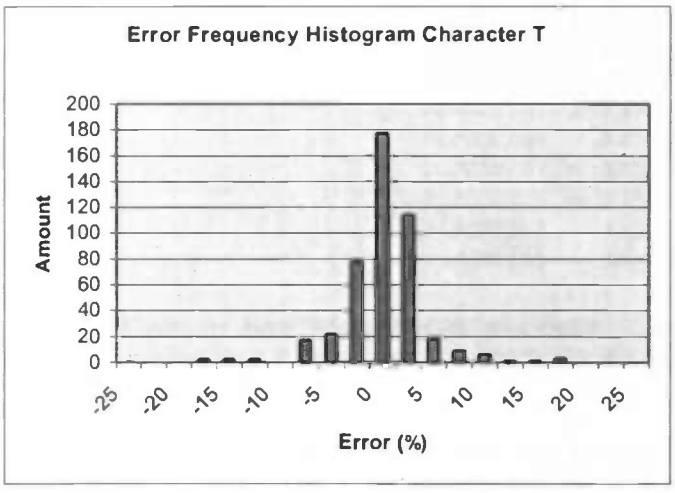

Error Frequency Histogram Character L
Amount
90
80
70
60
50
40
30
20
10
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

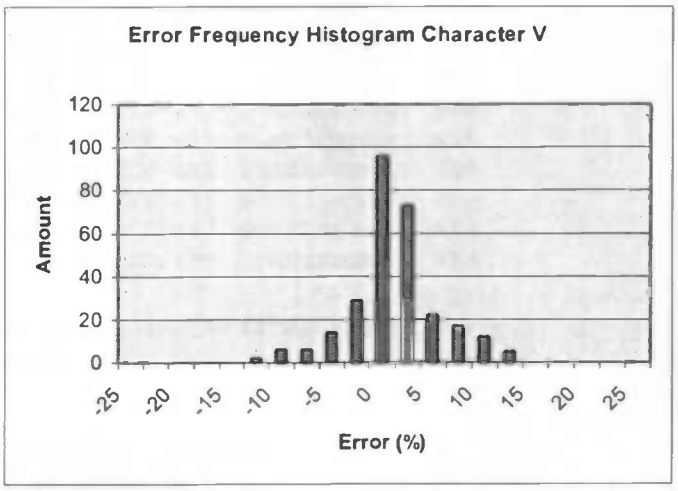

Error Frequency Histogram Character N
Amount
140
120
100
80
60
40
20
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

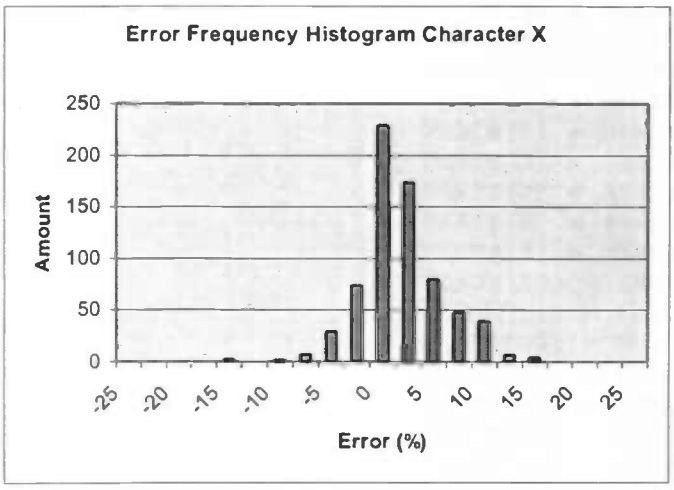

Error Frequency Histogram Character P
Amount
180
160
140
120
100
80
60
40
20
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

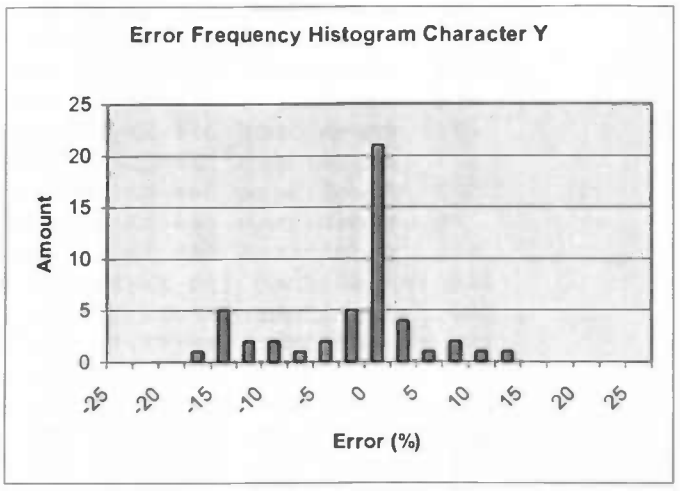

Error Frequency Histogram Character R
Amount
300
250
200
150
100
50
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

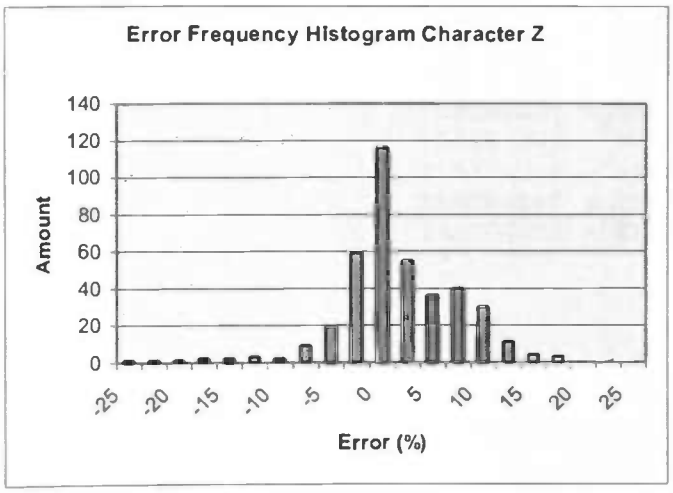

Error Frequency Histogram Character S
Amount
300
250
200
150
100
50
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)


Error Frequency Histogram Character T
Amount
200
180
160
140
120
100
80
60
40
20
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)


Error Frequency Histogram Character V
Amount
120
100
80
60
40
20
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)


Error Frequency Histogram Character X
Amount
250
200
150
100
50
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)


Error Frequency Histogram Character Y
Amount
25
20
15
10
5
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)


Error Frequency Histogram Character Z
Amount
140
120
100
80
60
40
20
0
-25 -20 -15 -10 -5 0 5 10 15 20 25
Error (%)

Appendix D: Missing Characters

Elastic Matching using perfect license plate XX-XX-44 as input, reporting the sixth character as missing:

```
...answer  : found in total 10 possible plate(s)
             PLATE[0] = XXXX40 (NL XX-XX-40) confidence: 983
             PLATE[1] = XXXX41 (NL XX-XX-41) confidence: 30
             PLATE[2] = XXXX42 (NL XX-XX-42) confidence: 634
             PLATE[3] = XXXX43 (NL XX-XX-43) confidence: 586
             PLATE[4] = XXXX44 (NL XX-XX-44) confidence: 1000
             PLATE[5] = XXXX45 (NL XX-XX-45) confidence: 634
             PLATE[6] = XXXX46 (NL XX-XX-46) confidence: 739
             PLATE[7] = XXXX47 (NL XX-XX-47) confidence: 693
             PLATE[8] = XXXX48 (NL XX-XX-48) confidence: 811
             PLATE[9] = XXXX49 (NL XX-XX-49) confidence: 857
```

Elastic Matching using perfect license plate XX-XX-44 as input, reporting the first character as missing:

```
...answer  : found in total 26 possible plate(s)
             PLATE[0] = AXXX44 (NL AX-XX-44) confidence: 695
             PLATE[1] = BXXX44 (NL BX-XX-44) confidence: 562
             PLATE[2] = CXXX44 (NL CX-XX-44) confidence: 536
             PLATE[3] = DXXX44 (NL DX-XX-44) confidence: 849
             PLATE[4] = EXXX44 (NL EX-XX-44) confidence: 395
             PLATE[5] = FXXX44 (NL FX-XX-44) confidence: 395
             PLATE[6] = GXXX44 (NL GX-XX-44) confidence: 774
             PLATE[7] = HXXX44 (NL HX-XX-44) confidence: 625
             PLATE[8] = IXXX44 (NL IX-XX-44) confidence: 27
             PLATE[9] = JXXX44 (NL JX-XX-44) confidence: 238
             PLATE[10] = KXXX44 (NL KX-XX-44) confidence: 804
             PLATE[11] = LXXX44 (NL LX-XX-44) confidence: 420
             PLATE[12] = MXXX44 (NL MX-XX-44) confidence: 501
             PLATE[13] = NXXX44 (NL NX-XX-44) confidence: 785
             PLATE[14] = OXXX44 (NL OX-XX-44) confidence: 942
             PLATE[15] = PXXX44 (NL PX-XX-44) confidence: 536
             PLATE[16] = QXXX44 (NL QX-XX-44) confidence: 536
             PLATE[17] = RXXX44 (NL RX-XX-44) confidence: 852
             PLATE[18] = SXXX44 (NL SX-XX-44) confidence: 562
             PLATE[19] = TXXX44 (NL TX-XX-44) confidence: 687
             PLATE[20] = UXXX44 (NL UX-XX-44) confidence: 719
             PLATE[21] = VXXX44 (NL VX-XX-44) confidence: 911
             PLATE[22] = WXXX44 (NL WX-XX-44) confidence: 304
             PLATE[23] = XXXX44 (NL XX-XX-44) confidence: 1000
             PLATE[24] = YXXX44 (NL YX-XX-44) confidence: 649
             PLATE[25] = ZXXX44 (NL ZX-XX-44) confidence: 555
```

Edges method using perfect license plate XX-XX-44 as input, reporting the sixth character as missing:

```
...answer  : found in total 10 possible plate(s)
             PLATE[0] = XXXX40 (NL XX-XX-40) confidence: 831
             PLATE[1] = XXXX41 (NL XX-XX-41) confidence: 130
             PLATE[2] = XXXX42 (NL XX-XX-42) confidence: 570
             PLATE[3] = XXXX43 (NL XX-XX-43) confidence: 503
             PLATE[4] = XXXX44 (NL XX-XX-44) confidence: 1000
             PLATE[5] = XXXX45 (NL XX-XX-45) confidence: 582
             PLATE[6] = XXXX46 (NL XX-XX-46) confidence: 783
             PLATE[7] = XXXX47 (NL XX-XX-47) confidence: 616
             PLATE[8] = XXXX48 (NL XX-XX-48) confidence: 803
             PLATE[9] = XXXX49 (NL XX-XX-49) confidence: 815
```

Edges method using perfect license plate XX-XX-44 as input, reporting the first character as missing:

```
...answer  : found in total 26 possible plate(s)
             PLATE[0] = AXXX44 (NL AX-XX-44) confidence: 566
             PLATE[1] = BXXX44 (NL BX-XX-44) confidence: 461
             PLATE[2] = CXXX44 (NL CX-XX-44) confidence: 430
             PLATE[3] = DXXX44 (NL DX-XX-44) confidence: 723
             PLATE[4] = EXXX44 (NL EX-XX-44) confidence: 294
             PLATE[5] = FXXX44 (NL FX-XX-44) confidence: 296
             PLATE[6] = GXXX44 (NL GX-XX-44) confidence: 697
             PLATE[7] = HXXX44 (NL HX-XX-44) confidence: 528
             PLATE[8] = IXXX44 (NL IX-XX-44) confidence: 112
             PLATE[9] = JXXX44 (NL JX-XX-44) confidence: 216
             PLATE[10] = KXXX44 (NL KX-XX-44) confidence: 637
             PLATE[11] = LXXX44 (NL LX-XX-44) confidence: 311
             PLATE[12] = MXXX44 (NL MX-XX-44) confidence: 385
             PLATE[13] = NXXX44 (NL NX-XX-44) confidence: 626
             PLATE[14] = OXXX44 (NL OX-XX-44) confidence: 849
             PLATE[15] = PXXX44 (NL PX-XX-44) confidence: 430
             PLATE[16] = QXXX44 (NL QX-XX-44) confidence: 430
             PLATE[17] = RXXX44 (NL RX-XX-44) confidence: 652
             PLATE[18] = SXXX44 (NL SX-XX-44) confidence: 472
             PLATE[19] = TXXX44 (NL TX-XX-44) confidence: 499
             PLATE[20] = UXXX44 (NL UX-XX-44) confidence: 621
             PLATE[21] = VXXX44 (NL VX-XX-44) confidence: 775
             PLATE[22] = WXXX44 (NL WX-XX-44) confidence: 253
             PLATE[23] = XXXX44 (NL XX-XX-44) confidence: 1000
             PLATE[24] = YXXX44 (NL YX-XX-44) confidence: 584
             PLATE[25] = ZXXX44 (NL ZX-XX-44) confidence: 450
```

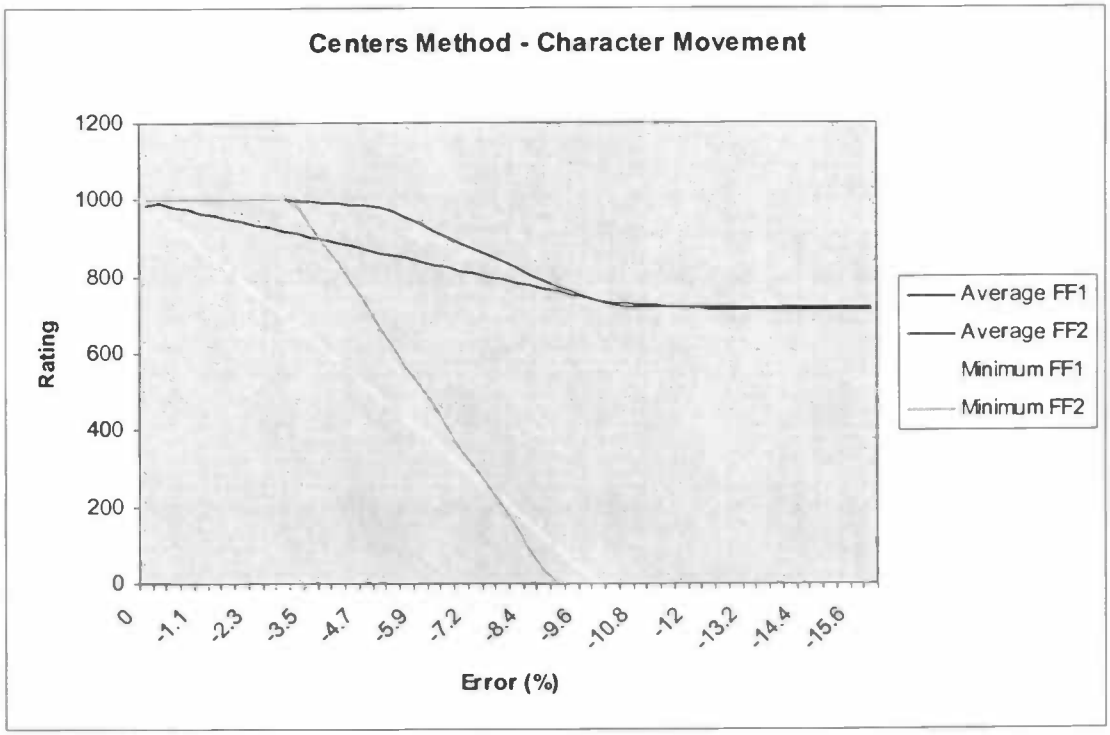
Centers method using perfect license plate XX-XX-44 as input, reporting the sixth character as missing:

```
...answer  : found in total 10 possible plate(s)
             PLATE[0] = XXXX40 (NL XX-XX-40) confidence: 828
             PLATE[1] = XXXX41 (NL XX-XX-41) confidence: 95
             PLATE[2] = XXXX42 (NL XX-XX-42) confidence: 568
             PLATE[3] = XXXX43 (NL XX-XX-43) confidence: 489
             PLATE[4] = XXXX44 (NL XX-XX-44) confidence: 1000
             PLATE[5] = XXXX45 (NL XX-XX-45) confidence: 583
             PLATE[6] = XXXX46 (NL XX-XX-46) confidence: 776
             PLATE[7] = XXXX47 (NL XX-XX-47) confidence: 611
             PLATE[8] = XXXX48 (NL XX-XX-48) confidence: 795
             PLATE[9] = XXXX49 (NL XX-XX-49) confidence: 808
```

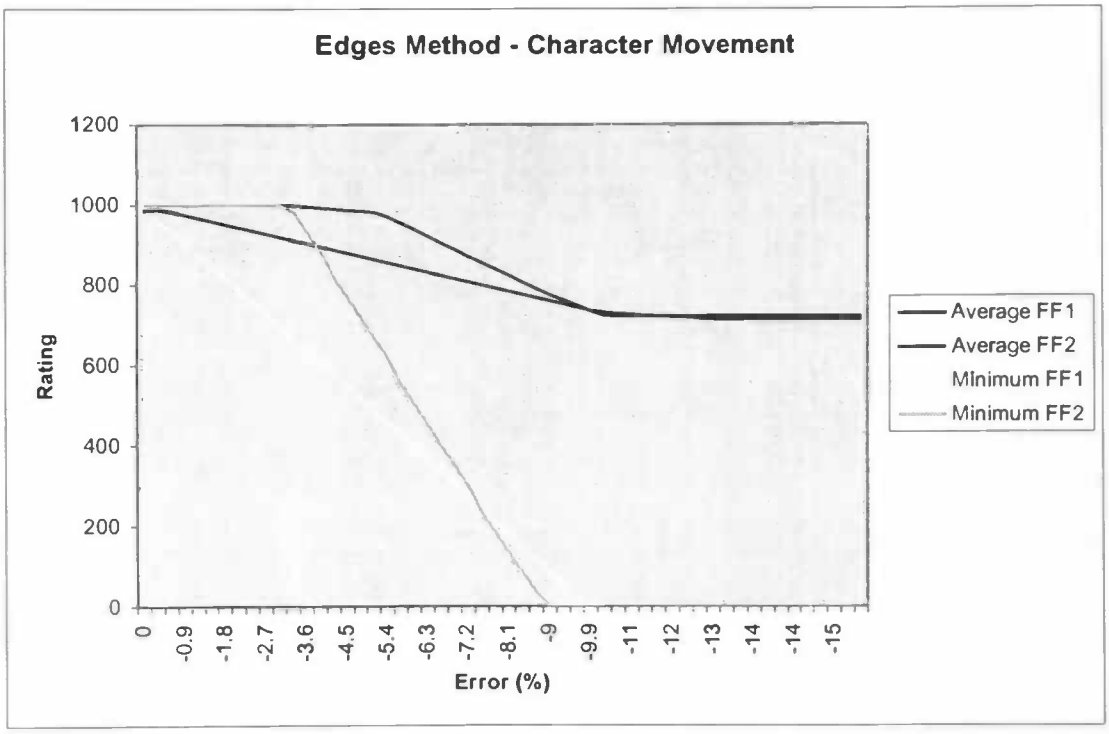
Centers method using perfect license plate XX-XX-44 as input, reporting the first character as missing:

```
...answer  : found in total 26 possible plate(s)
             PLATE[0] = AXXX44 (NL AX-XX-44) confidence: 604
             PLATE[1] = BXXX44 (NL BX-XX-44) confidence: 484
             PLATE[2] = CXXX44 (NL CX-XX-44) confidence: 454
             PLATE[3] = DXXX44 (NL DX-XX-44) confidence: 752
             PLATE[4] = EXXX44 (NL EX-XX-44) confidence: 292
             PLATE[5] = FXXX44 (NL FX-XX-44) confidence: 297
             PLATE[6] = GXXX44 (NL GX-XX-44) confidence: 734
             PLATE[7] = HXXX44 (NL HX-XX-44) confidence: 554
             PLATE[8] = IXXX44 (NL IX-XX-44) confidence: 58
             PLATE[9] = JXXX44 (NL JX-XX-44) confidence: 210
             PLATE[10] = KXXX44 (NL KX-XX-44) confidence: 667
             PLATE[11] = LXXX44 (NL LX-XX-44) confidence: 315
             PLATE[12] = MXXX44 (NL MX-XX-44) confidence: 412
             PLATE[13] = NXXX44 (NL NX-XX-44) confidence: 661
             PLATE[14] = OXXX44 (NL OX-XX-44) confidence: 865
             PLATE[15] = PXXX44 (NL PX-XX-44) confidence: 458
             PLATE[16] = QXXX44 (NL QX-XX-44) confidence: 454
             PLATE[17] = RXXX44 (NL RX-XX-44) confidence: 683
             PLATE[18] = SXXX44 (NL SX-XX-44) confidence: 508
             PLATE[19] = TXXX44 (NL TX-XX-44) confidence: 532
             PLATE[20] = UXXX44 (NL UX-XX-44) confidence: 659
             PLATE[21] = VXXX44 (NL VX-XX-44) confidence: 797
             PLATE[22] = WXXX44 (NL WX-XX-44) confidence: 252
             PLATE[23] = XXXX44 (NL XX-XX-44) confidence: 1000
             PLATE[24] = YXXX44 (NL YX-XX-44) confidence: 642
             PLATE[25] = ZXXX44 (NL ZX-XX-44) confidence: 481
```

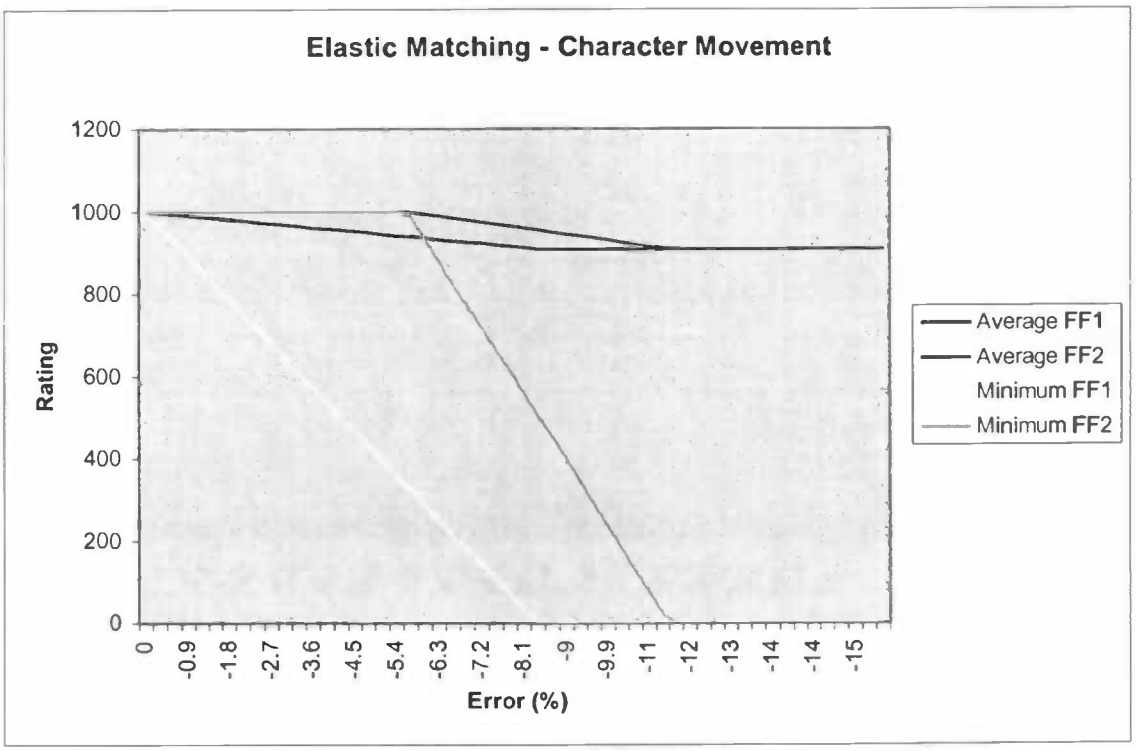
Appendix E : Character Movement



Graph i: Effect of moving one character on total plate rating in a perfect license plate using the Centers method.

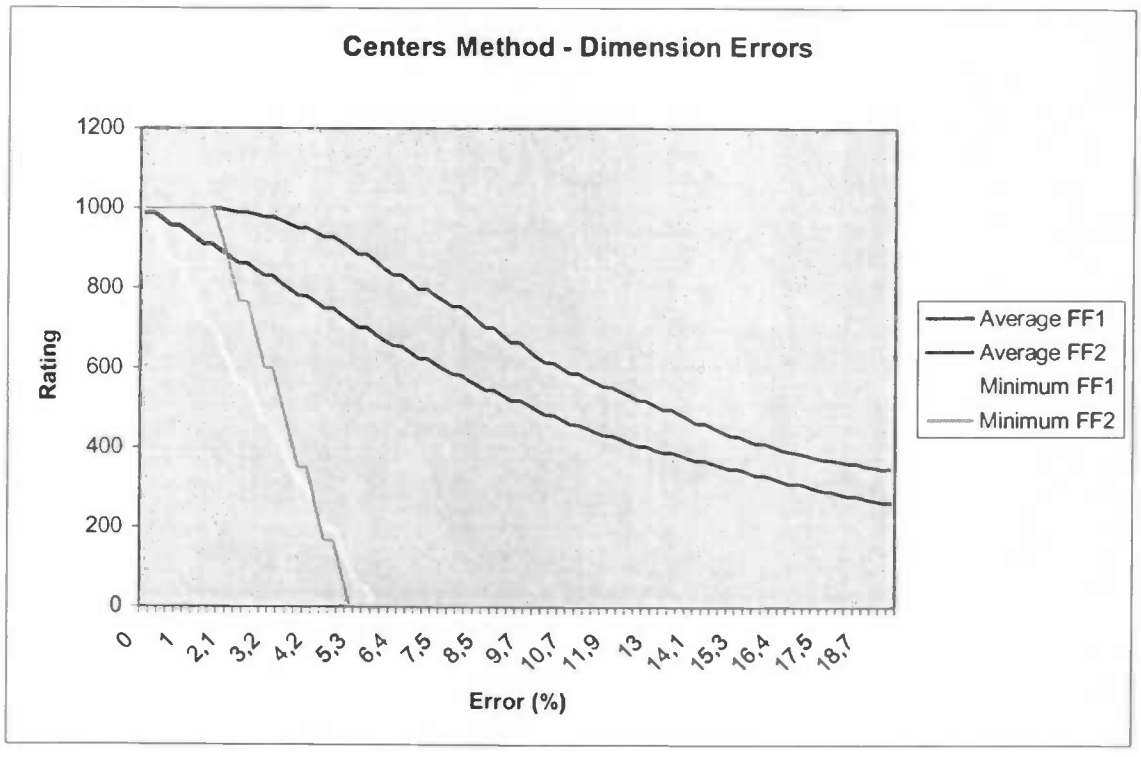


Graph ii: Effect of moving one character on total plate rating in a perfect license plate using the Edges method.

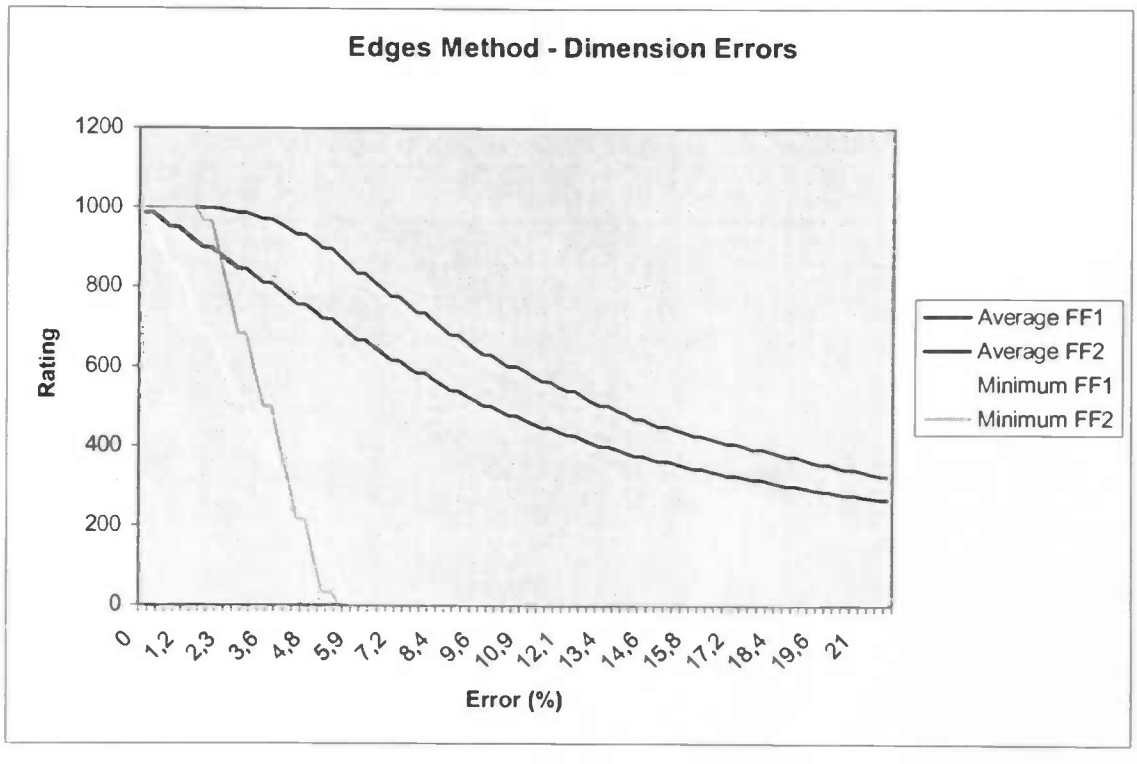


Graph ii: Effect of moving one character on total plate rating in a perfect license plate using Elastic Matching.

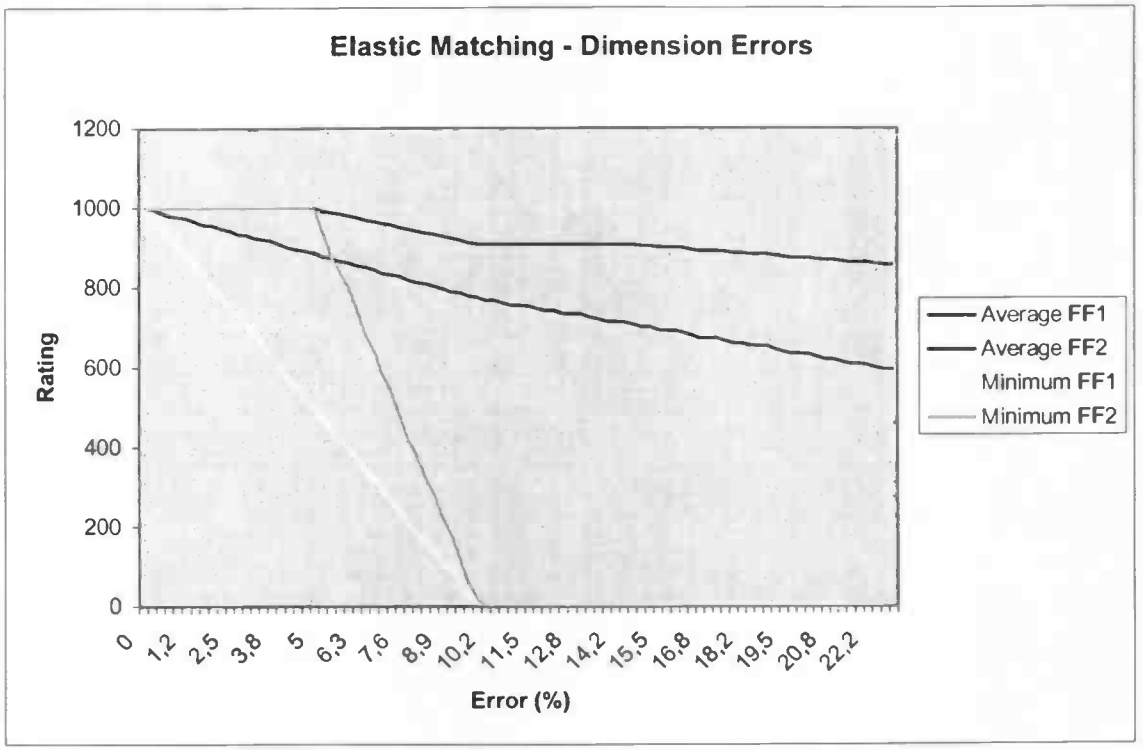
Appendix F : Dimension Errors



Graph i: Effect of one dimension error on total plate rating in a perfect license plate using the Centers method.



Graph ii: Effect of one dimension error on total plate rating in a perfect license plate using the Edges method.



Graph iii: Effect of one dimension error on total plate rating in a perfect license plate using Elastic matching.