Master thesis 588:

Accuracy improvement of an optical paper position sensor

Author

Joost Rovers

Supervisors

Jozef Hooman	Radboud University Nijmegen	Computer science
Jan de Wit	Radboud University Nijmegen	Innovation Studies
Heico Sandee	Océ-Technologies	

Radboud Universiteit Nijmegen

Printing for Professionals

Abstract

In this thesis we analyse and improve the accuracy of paper displacement estimation of an optical paper position sensor developed by Océ Technologies. The goal of the research is to improve the sensor software, such that a higher accuracy is achieved in paper displacement estimation. Then it is possible for Océ to develop a cheaper sensor with less accurate components, which still gives the same accuracy as the old sensor. We demonstrate that with a few simple improvements a 65% increase in accuracy can be achieved.

For measuring the accuracy of paper displacement estimation, we developed a method which enabled us to measure and to compare different approaches under the same conditions. The accuracy of paper displacement estimation is influenced by many factors. The influence of illumination pattern, sensor's fingerprint, presence of noise, and the used algorithms on the accuracy is investigated and solutions to reduce these effects are found and tested. Some of these methods proved to be successful and enabled us to achieve a significant increase in accuracy.

Further we investigated the influence of this increase in accuracy on the total cost of ownership. Due to lack of available information about costs of possible substitute components of the sensor, we cannot give an exact overview of the influence on the total cost of ownership. However, we still managed to give an estimation of this influence and compared to the current sensor, it seems that less accurate components are not much cheaper, whereas more accurate ones do cost significant more.

Contents

1 Introduction

This report describes a research project about the accuracy improvement of an optical paper position sensor. The purpose of this sensor is to measure the displacement of paper within a printer more exact than is possible with traditional methods, such as counting the revolutions of the paper transport engine. At the start of this project, the accuracy of the displacement measurements with the existing sensor was not at an acceptable level and it was believed that improvements were possible. In our research we analyze causes of deterioration of the measurement's accuracy and we formulate solutions for that. We also discuss how we can improve the price/quality relationship of the sensor.

The result of this research project is the master thesis of Joost Rovers, studying at the Radboud University Nijmegen. Joost studies computing science, with the graduation theme `Embedded Systems' and the track `Management and Technology'. The project is an assignment of, and was done at, Océ-Technologies B.V. in Venlo.

For Océ, and other printer manufacturers, this research is interesting because the optical paper position sensor allows them to measure the paper displacement with more accuracy than with traditional methods. More exact measurement of the displacement of paper enables a better printing quality.

We start off with explaining the overview of this research in chapter 2. After that, in chapter 3, we provide an overview of the function of the sensor. We will discuss the functionality and it's built in algorithms. We also define what we mean with accuracy, explain how we measure the accuracy of the sensor and provide the results of these measurements. Next, chapter 4 provides an overview of causes which can have a negative influence on the accuracy of the sensor. We explain the improvements for the causes where we think most gain can be made. Chapter 5 gives a more in-depth analysis of the algorithms used in the sensor. In chapter 6 we present the proposed improvements of the sensor software. Then, in chapter 7 we explain the details of our measurements. Chapter 8 contains the evaluation of how much the accuracy is influenced by the proposed improvements. Next, in chapter 9 we discuss the influence of the improvements on the price/quality relationship. Finally, chapter 10 provides the conclusions of this research project.

2 Research description

In this chapter an overview is given of the research into accuracy improvement of an optical paper position sensor. We start with describing the objectives of this research. After that, in section 2.2, the research questions will be posed that need to be answered in order to achieve the objectives. In section 2.3 we will discuss why this research is relevant for Océ.

2.1 Objectives

In the embedded systems world it is common to use optical sensors for accurately measuring and calibrating devices. Océ is interested in how they can use this technology for accurately measuring the paper displacement within a printer to improve the printing quality. Of course, the cost of the sensor needs to be weighted against the added value. Océ is therefore interested to know how they can reduce the sensors costs with improved software, which is capable of working around the negative effects that can occur while working with less accurate but cheaper hardware or a cheaper production process. The negative effects that can occur are e.g. higher signal-to-noise ratios when capturing images, more visible illumination pattern, stronger sensors fingerprint, optical distortion, dust on the lens and bigger variances in depth of focus. Further we investigate better image processing algorithms which provide a higher accuracy when determining the paper displacement and the influence of measurements during paper displacements when the paper is moving. The goal is to improve the software in such a way that with cheaper hardware still sufficient accuracy can be achieved.

A balance needs to be found between costs and accuracy. Figure 1 shows the relation between costs and accuracy of the sensor. The numbers and functions are just indicative and do not represent the exact values of the cost-accuracy relationship. The solid line shows the estimated cost-accuracy relationship with the current hardware. We estimate that when using cheaper parts in the construction of the optical sensor the accuracy will be less (without modifying the software). The dashed line in Figure 1 shows the desired cost-accuracy relationship Océ would like to reach after the software improvements. By each software improvement the cost-accuracy relationship should move towards the desired one.

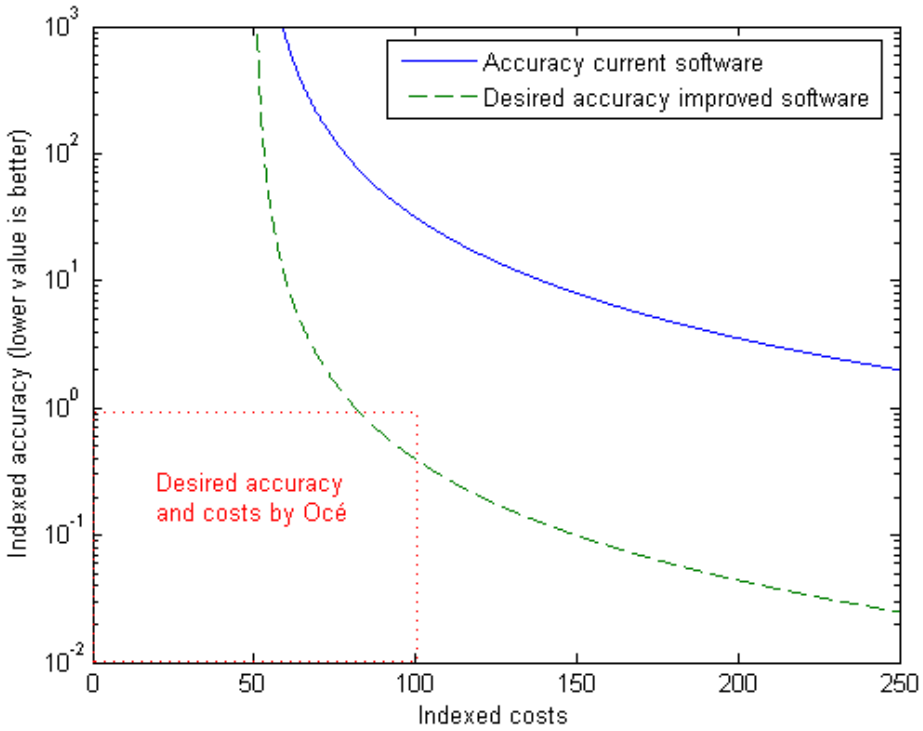


Figure 1 - Graph showing estimated relation between costs and accuracy with current software and the desired accuracy of the improved software

2.2 Accuracy

We first define what we mean by accuracy. Accuracy is the degree to which a given quantity is correct and free from error [16]. In the context of the optical paper position sensor it means the degree of correctness of the measured paper displacement. When doing repeated measurements in accuracy and no structural or systematic error is present, the results often form a normally distributed population. This population can be presented in a standard deviation diagram (Figure 2).

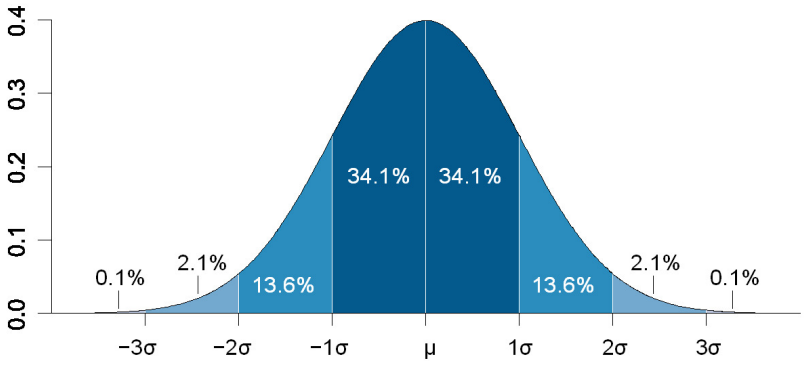


Figure 2 - Standard deviation diagram

In an approximately normally distributed population about 68% of the values lie within one standard deviation (σ) of the mean, about 95 % of the values are within two standard deviations and about 99.7 % lie within 3 standard deviations. This is known as the 68-95-99.7

rule, or the empirical rule. The standard deviation represents the amount of spreading of the values. In this research we are interested in the values that lie within 3 times the standard deviation. This is also called as the 3-σ value.

We will measure the accuracy of each different algorithm by doing many estimations of paper displacement and subtract the real paper displacements from these estimations. This will result in a big set of values which represent the deviations for all estimated paper displacements. The accuracy of the algorithm is then determined by the 3-σ value of this set.

The accuracy depends on the used algorithms and the used hardware. The accuracy that is achieved with the current hardware and software is not sufficient for Océ. They desire three times more accurate results from an optical paper position sensor. Moreover, we expect that cheaper and therefore less accurate hardware does influence the accuracy negatively. This needs to be compensated with 'smarter' software, to achieve the same accuracy as with the current sensor.

2.3 Research questions

From the research objectives we can derive the following hypothesis:

With 'smarter' software a cheaper version of the optical sensor can still provide the accuracy desired by Océ.

We can validate this hypothesis by answering the following questions:

1. What is the current status?
    a. What is the desired accuracy by Océ?
    b. How can we measure accuracy using the optical sensor?
    c. How accurate is the current optical sensor?

2. Which parts can we improve of the optical sensor?
    a. Which effects decrease the accuracy of cheaper optical sensors?
    b. How can we reduce these effects with smarter software?
    c. What is the quantitative contribution of these software solutions to the accuracy?

3. What are the consequences of different sensor configurations to the total cost of ownership of the printer?
    a. What are the advantages and disadvantages of the different configurations?
    b. Can we lower the total cost of ownership of the printer with other possible functionality of the optical sensor?

2.4 Relevance for Océ

Océ produces printers and copiers. A branch of printers in which Océ excels is wide-format printers. These printers can be used with different kinds of wide formats and have excellent throughput, high durability, high quality and ease of use. An example of this kind of printers is the TCS500 (see Figure 3).

Figure 3 - TCS500

This printer can print an A0 in black and white every 41 seconds and in full color in 63 seconds. It prints with a precision of 600x600 dpi and can print on different media like uncoated (56 - 90 gr/m2), coated (85 - 120 gr/m2), and transparent paper (90 - 112 gr/m2), polyester films (90 - 120 micron) and photo gloss (120 - 175 gr/m2) [10].
Customers of wide-format printers increasingly demand higher printing quality, higher productivity and higher user friendliness from these printers. To meet these demands, Océ investigates how to improve these features. User friendliness can be improved, for instance by allowing the printer to use more different types of paper at comparable printing quality, so that for example, paper from a competitor that a user has in stock could be used in the Océ printer. For each different type of paper the printers need to be calibrated to get the maximum quality, because each type of paper behaves slightly different within a printer. For Océ media, calibration can be done at the factory, but it is not possible to do this for all media types from all the competitors.

During the printing process the medium is transported through the printer and rows of dots are printed. One of the things that should be done to increase the quality of a print is to improve the accuracy of the paper displacement. When the paper displacement is made too big a thin white line between two printed rows can be noticed. When a paper step is made too small some parts will be double printed. In both situations a loss of print quality can be noticed. To minimize these problems, the paper positioning has to be as accurate as possible. Nowadays, a paper step accuracy of about 1 µm per mm paper transport is required to achieve acceptable print quality. [6]

For Océ it is therefore useful to know whether the accuracy of the optical paper position sensor can be improved and whether this sensor can be a mature replacement of or an addition to the current paper tracking methods.

3 Overview of the function of the sensor

This chapter gives an overview of the function of the sensor. We discuss the functionalities of the sensor, how we can operate with it and how the built-in algorithms calculate to the estimated paper displacement.

3.1 Functionality

In Figure 4 a schematic overview of the sensor's architecture is depicted. The paper moves in a certain direction and the sensor can capture an image of the paper surface. The embedded software triggers the sensor to capture an image. The captured images are retrieved by the embedded software. Then the embedded software can provide as output either the amount of paper displacement in µm between two captured images or the captured images themselves, so that later analysis can be done with the offline development software. We use the offline development software for testing different algorithms and techniques on the retrieved pictures and to compare the results.

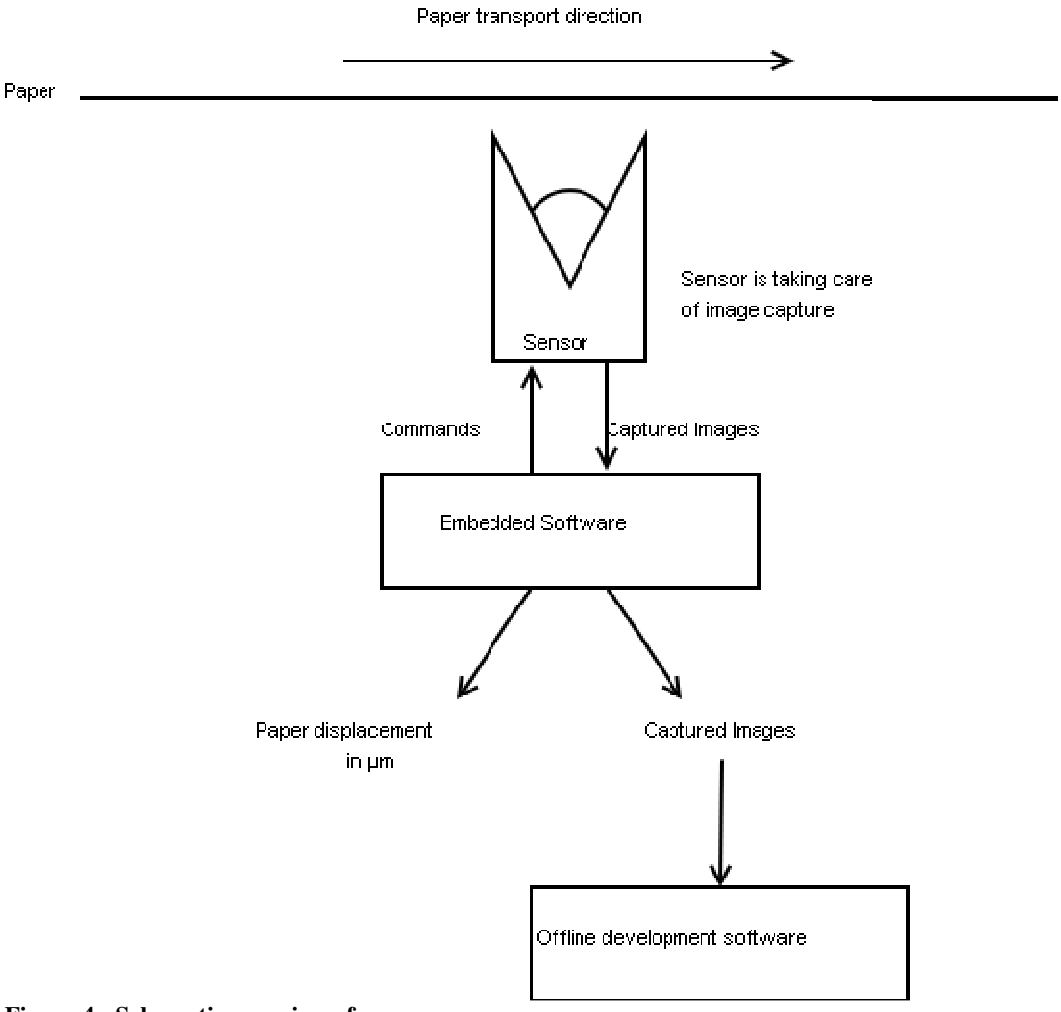


Figure 4 - Schematic overview of sensor

3.2 Algorithms

This section discusses the definition of accuracy as it is used in this research. Then we discuss the basis of the image alignment algorithm and two sub pixel estimation methods. Further we discuss some strategies for measuring distances larger than what can be seen in one captured image.

3.2.1 Basis of algorithm

As described earlier, the paper displacement is measured with an optical sensor. This sensor makes pictures of the paper surface at different positions. Paper usually has a distinctive pattern which is shown in Figure 5. We use this pattern to track the displacement of the paper, which is vital for measuring the paper displacement via an optical sensor.

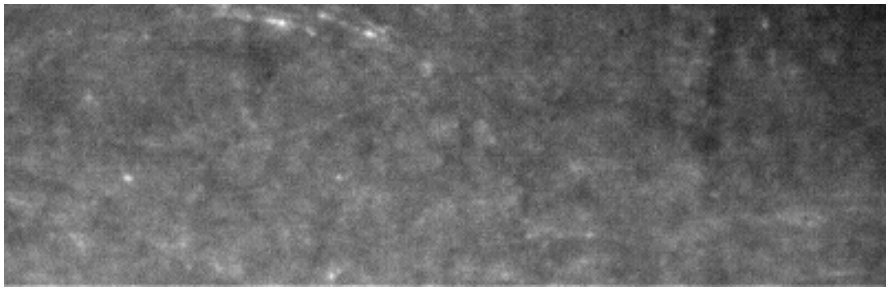


Figure 5 - Paper structure (magnification 50x)

Paper displacement can be measured with image alignment techniques. An image may be defined as a two-dimensional signal or function, *f(x,y)*, where x and y are *spatial* (plane) coordinates, and the amplitude of *f* at any pair of coordinates *(x,y)* is called the *intensity* or *gray level* of the image at that point. When *x*, *y* and the intensity values of *f* are all finite, discrete quantities, we call the image a digital image [5]. The correlation between two signals (cross correlation) is a standard approach to feature detection and this can be used for the alignment of two images [8]. Many image alignment techniques use this as basis of their algorithm. In Appendix A.1 an explanation can be found for cross correlation.

The displacement of paper can be seen in Figure 6. The rectangle points out the correlation window. This is a rectangle of a certain size, which is used as reference point. This part of the left picture is found in the right picture a few pixels higher.

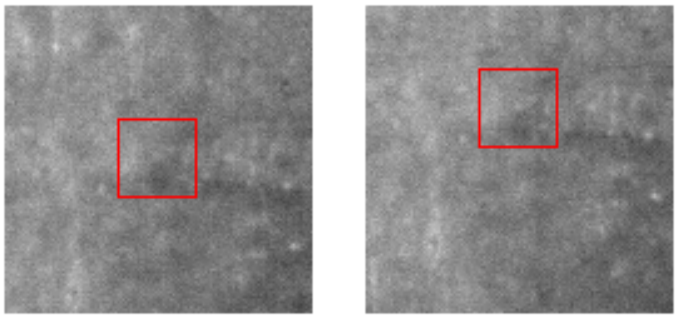


Figure 6 - Paper displacement of 20 pixels to the top.

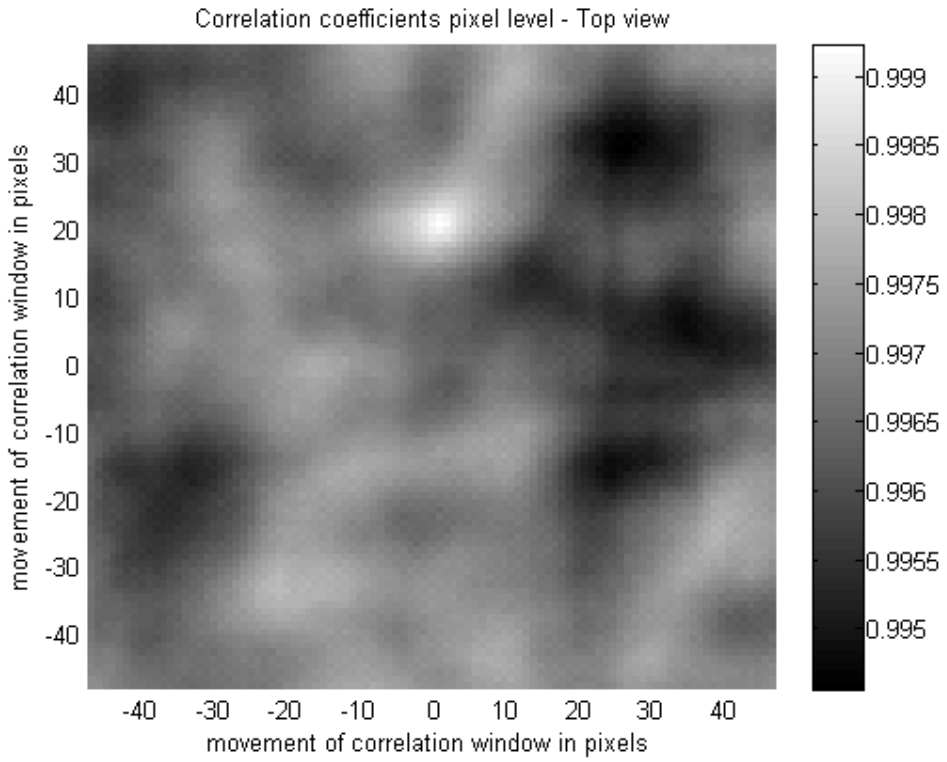


Figure 7 - Top view of matrix of correlation coefficients

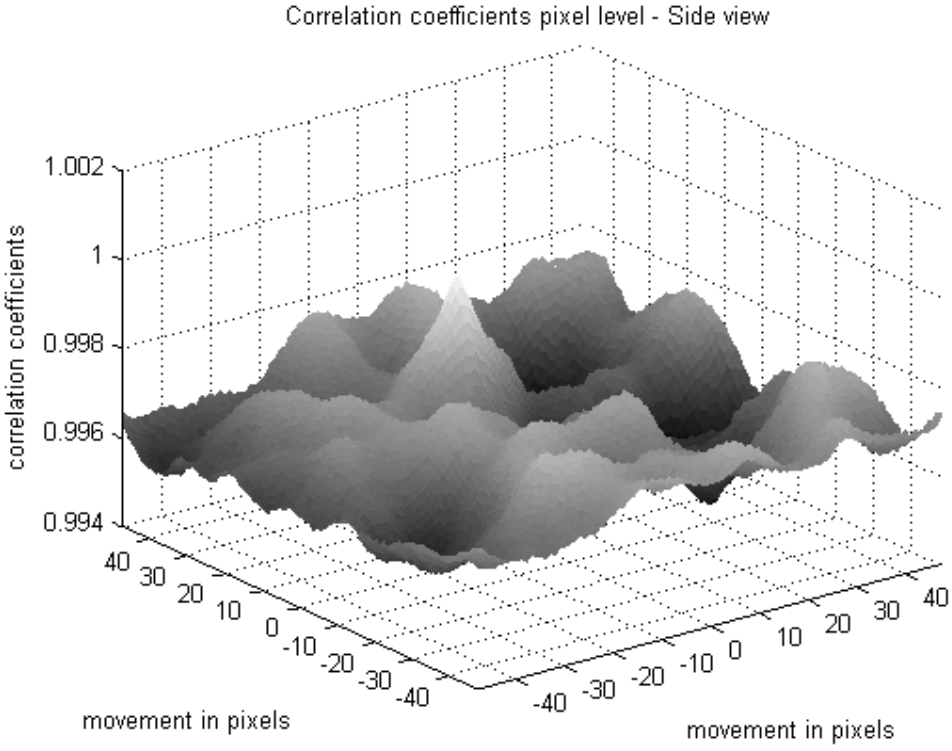


Figure 8 - Side view matrix of correlation coefficients

This displacement is calculated via the following algorithm. First, in the left picture the correlation window is determined. This can be done in various ways like choosing the centre of the image or choosing the area with most contrast. This part of the left picture is then searched for in the picture on the right by 'moving' the correlation window across it and calculate the cross correlation for each step. At the position where the cross correlation gives the maximum result, we find the best match of the correlation window. The displacement is then defined as the combination of the displacement in x direction and the displacement in the y direction. In Figure 7 and Figure 8 we can see correlation values and a high peak can be found twenty pixels above the centre. Thus the displacement is 20 pixels in y-direction and 0 pixels in the x-direction.

Then we can estimate the real distance of the paper displacement in micrometers by multiplying the amount of pixels with the size per pixel. The pixel pitch in our sensor is 2.5 by 2.5 µm. This means that each pixel in the picture made by our sensor reflects 2.5 x 2.5 µm of the paper surface. Thus the displacement of the paper is in our example about 20 x 2.5 = 50µm in y-direction.

The actual displacement of the paper in this example is 50.05µm, so in this particular case the paper displacement is very accurately determined by using this algorithm with only 0.05µm deviation. However using only the pixel level for the displacement calculation, limits us to a resolution of 2.5µm and this will more often result in a lower accuracy. A displacement of 51.20µm will be recognized as a displacement of 50.00µm and a displacement of 51.30µm as 52.50µm. The deviation in these cases is 1.20µm, which is not accurate enough. We can achieve a higher resolution and therefore higher accuracy by using sub pixel displacement algorithms.

3.2.2 Sub pixel level displacement estimation

There are different sub pixel displacement algorithms and different ways to use them. In the sensor's embedded software there are two different methods present and we will use these to determine the current accuracy of the sensor. Later we will introduce other techniques and analyze if they improve the accuracy of the sensor.

First method - polynomial fitting through 3 points

This method uses the assumption that real displacement can be calculated by making a polynomial fit through the values of the correlation coefficients matrix (Figure 7).

Assume that we determined the displacement of the paper at the pixel level by finding the maximum within the matrix of correlation coefficients at point *(x,y)*. Then we will interpolate the real maximum in the x-direction by making a polynomial fit through three points by using *(x-1,y)*, *(x,y)* and *(x+1,y)*. We will do the same for the y-direction with points *(x,y-1)*, *(x,y)* and *(x,y+1)*.

For x-direction we will use the following formula for determining the optimum.

$$x\_optimum = x\_offset + \frac{\frac{1}{2}\times\left(cor[(x+1,y)] - cor[(x-1,y)]\right)}{2\times cor[(x,y)] - cor[(x+1,y)] - cor[(x-1,y)]}$$

where cor[(x,y)] gives the correlation value in position (x,y)

For the y-direction we use the formula which looks quite the same, but we use the correlation values within the y-direction:

$$y\_optimum = y\_offset + \frac{\frac{1}{2}\times\left(cor[(x, y+1)] - cor[(x, y-1)]\right)}{2\times cor[(x, y)] - cor[(x, y+1)] - cor[(x, y-1)]}$$

where cor[(x,y)] gives the correlation value in position (x,y)

See appendix A.2 for the derivation of these formulas.

We demonstrate the usage of this algorithm with the values in Figure 7. Only the values around the maximum correlation coefficient are used. The maximum correlation coefficient is 0.999215 and can be found at position (0, 20) in Figure 7. Note that the centre is defined as (0,0), so it means that the two images have a displacement of 20 pixels on the y-axis. Table 1 shows the maximum value for the correlation coefficient and its surrounding values.

Table 1 – Maximum correlation coefficient and the surrounding values within the matrix

	0.999133	
0.999065	0.999215	0.999091
	0.999041	

When we fill out both equations, we get the estimated displacement on sub pixel level.

$$x\_optimum = 0 + \frac{\frac{1}{2}\times(0.999091 - 0.999065)}{2\times 0.999215 - 0.999091 - 0.999065} = 0.0474$$

$$y\_optimum = 20 + \frac{\frac{1}{2}\times(0.999133 - 0.999041)}{2\times 0.999215 - 0.999133 - 0.999041} = 20.1797$$

Thus the estimated paper displacement on subpixel level is 0.0474 pixels on horizontal axis and 20.1797 pixels in the vertical axis. Converted to an estimation of the real displacement, so multiplying it with the pixel size of 2.5 µm, this gives a displacement 0.12 µm in horizontal axis and 50.45 µm in vertical axis. The real displacement is 50.05µm and we can calculate a deviation of 0.40µm.

In this example the sub pixel displacement estimation has a lower accuracy than in the previous example without sub pixel displacement algorithm. One measurement however doesn't say anything about the overall accuracy of an algorithm, so no conclusions can be made at this point. In chapter 7 and 8 we discuss how we measure the overall accuracy of the different algorithms.

Second method - bilinear interpolation

We showed in section 3.2.1 that the displacement of paper can be found by moving a correlation window across a second image to find a best match by doing a cross correlation at each step and searching for the maximum correlation coefficient. The accuracy is limited to size of the pixels. The second method 'reduces' the pixel pitch (size of the pixels) by creating new pixels between the original pixels via bilinear interpolation (See appendix A.3). Then we are able to move the correlation window with 'smaller' steps across the second image and estimate more accurately the displacement of paper.

Increasing the amount of pixels via interpolation increases the amount of values used in the cross correlation quadratic, e.g. estimating the displacement at a five times higher resolution, results in $5^2 = 25$ times more values used in the cross correlation. This increases the usage of memory and the amount of needed computations. In the bilinear interpolation algorithm only the best match window is interpolated and for each calculation only a subset of the

interpolated values are used. This decreases the amount of calculations and used memory and keeps the algorithm fast and usable for embedded systems.

We will demonstrate the algorithm with an example. Figure 9 shows an 8 by 8 correlation window on the left and the found best match window on the right. We use the same images as in the example on pixel level displacement estimation, where the best match window was found 20 pixels above the position of the correlation window. These windows are cut from the used images and enlarged for better visibility. Observe the resemblance between the correlation window and the best match window.

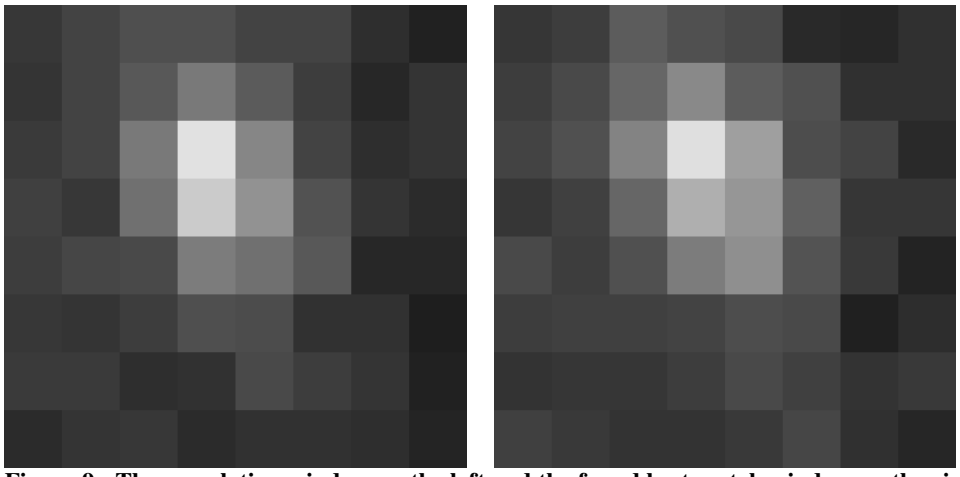


Figure 9 - The correlation window on the left and the found best match window on the right. These windows are cut from the captured images and enlarged for better visibility.

Interpolating the best match window five times with the bilinear interpolation algorithm results in an image with 40 by 40 pixels, as shown in Figure 10.

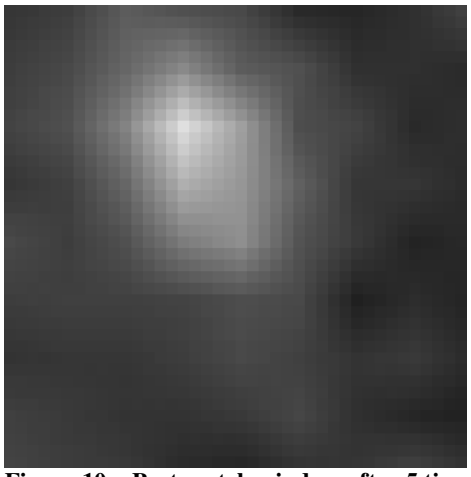
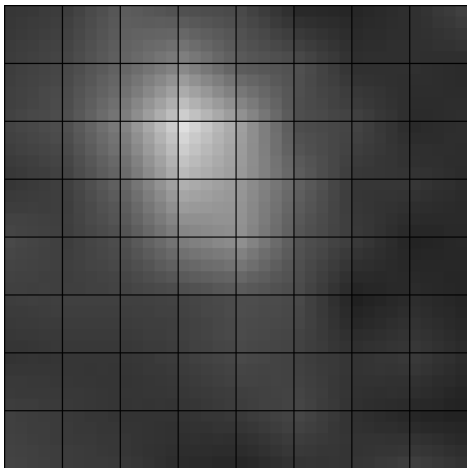


Figure 10 – Best match window after 5 times bilinear interpolation

The interpolated values are used to determine the subpixel level paper displacement. Since we already determined the displacement on pixel level, we can expect that the real displacement must be somewhere within one pixel distance. Thus, we are only interested in calculating the correlation coefficients for the subpixel steps within one pixel distance. In our example we used 5 times interpolation, thus 1 pixel consists of 5 subpixel steps. Therefore, the correlation window moves maximally 4 steps up, 4 steps right, 4 steps down and 4 steps left from the maximum found on pixel level. This gives us a matrix of 81 correlation coefficients.

However, the correlation window in Figure 9 consists of 64 values and the interpolated best match window of Figure 10 consists of 1600 values. Since calculating a correlation coefficient requires an equal amount of values, we can only use 64 of the 1600 interpolated values for each cross correlation.

To get a correlation coefficient matrix that still makes sense only specific subsets of the interpolated values are used. We divide the values of the bilinear interpolated best match window in 64 blocks of 5 by 5 values, as is shown in Figure 11. We indicate each value of a block with a separate number.



1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

Figure 11 – Left: Interpolated best match window divided in blocks of 5 by 5 pixels. Right: each pixel of a block is indicated with a number.

Pixel 1 of each block has the same value as in the non-interpolated best match window. When these pixels are used as a subset, the same correlation coefficient is calculated as on the pixel level displacement estimation. Calculating the correlation coefficients of each subpixel step is done by using its respective value in each block. E.g. the correlation coefficient of one subpixel step to the right is calculated using pixel 2 of each block. The correlation coefficient of one subpixel step down is calculated with pixel 6 of each block. Calculating the correlation coefficients for the subpixel steps up and to the left use the values in the blocks on the left and above. Thus, all the correlation coefficients are calculated with only one value of each block. This gives us the correlation coefficient matrix as visualized in Figure 12 and Figure 13.

The maximum value is found at point (0.4, 0.4) in the correlation coefficient matrix. Now we calculate the new optimum values.

$$x\_optimum = 0 + 0.4 = 0.4$$
$$y\_optimum = 20 + 0.4 = 20.4$$

Multiplying with the pixel size gives us a 1µm displacement on horizontal axis and 51µm displacement on vertical axis. The deviation using this algorithm in this example is 0.95µm.

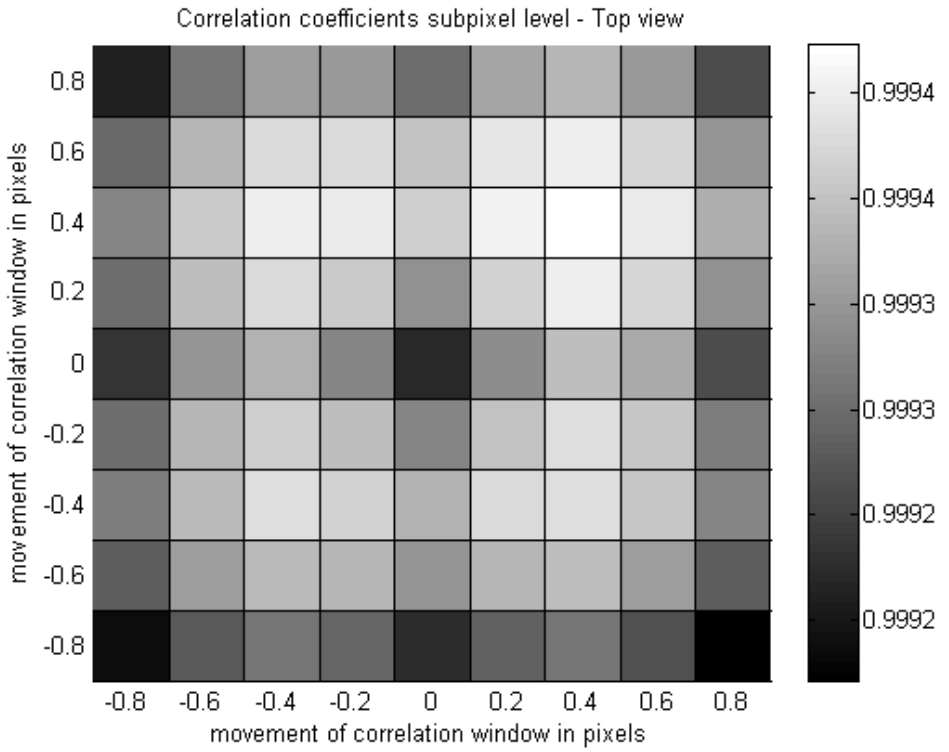


Figure 12 - Top view of matrix of correlation coefficients on subpixel level

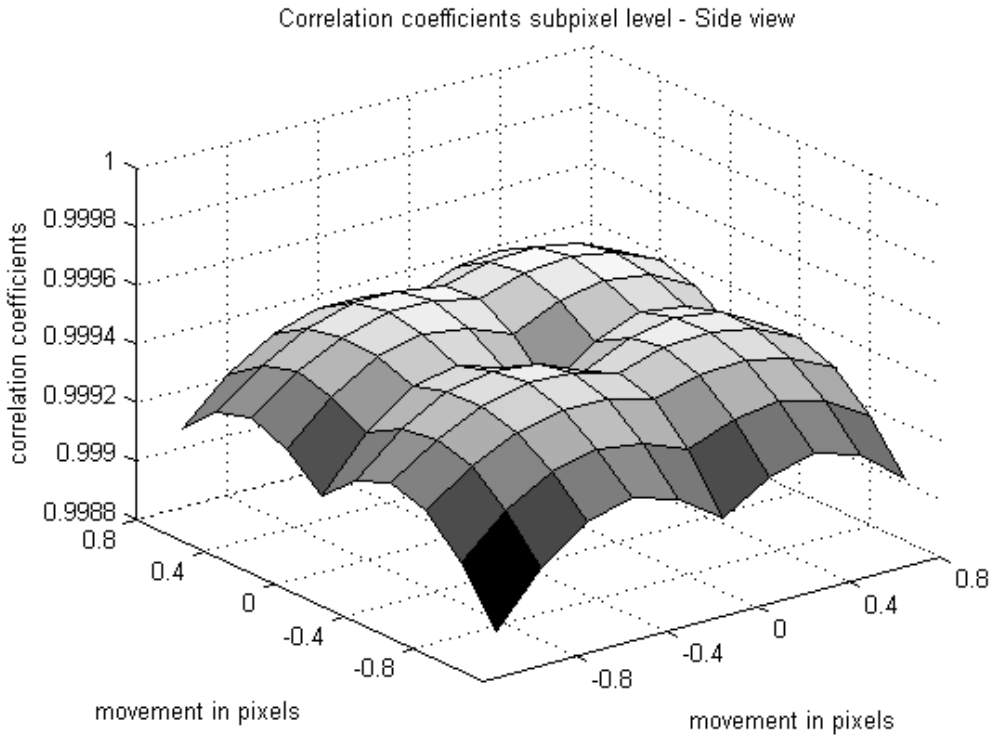


Figure 13 - Side view of matrix of correlation coefficients on subpixel

3.3 Defining the margin for maximum displacement

In section 3.2.1 we stated that a correlation window is determined by choosing the centre of the image or choosing the area with most contrast. This part of the first picture is then searched for in the second picture. The algorithms we use choose the area with most contrast for determining the correlation window, since this will give more accurate results when estimating the displacement. Choosing the correlation window this way needs special care, because e.g. if we would choose in Figure 6 the correlation window close to the top edge of the picture. Then the best match window will not be found correctly, since that piece of the paper surface is not even captured in the second image. Then the best match will be found somewhere else in the picture, since the algorithm just looks for the maximum correlation coefficient. This will result obviously in a very wrong estimation of displacement.

Therefore we define a margin for maximum displacement in x and y directions. For that we need to estimate in advance what this displacement could become. This information is needed anyway, because when the displacement is longer than what can be captured in the view of a camera, other techniques are needed. These are discussed in section 3.4.

Defining margins has two consequences for the correlation calculations. First, the margins limit the area in which the correlation window needs to be determined. E.g. when the expected displacement in x-direction is maximally 10 pixels, the correlation window is determined at least 10 pixels from the side edges. This makes sure that the best match window still can be found in the second picture. Second, since we know that the displacement is maximally 10 pixels in x-direction, we only need to find the best match window within this margin. This reduces the amount of needed calculations, because less correlation coefficients need to be calculated.

3.4 Strategies for measuring longer distances

So far, the paper displacements were very small and fitted easily within the view of one camera. Within a printer the paper usually moves relatively fast and over larger distances. These paper steps are usually between 5 mm and 50 mm. We will discuss three possible strategies.

One option is to continuously keep making pictures with a certain time interval while the paper is moving. This time interval needs to be chosen such that the correlation window still can be found in the second captured image. The faster the paper moves the shorter the interval needs to be chosen. Longer distances are measured by adding up all the estimations. However since each estimation has a certain deviation, the deviations add up and longer measured distances have therefore a higher deviation. Capturing images of moving objects gives a blurred effect. Experiments have shown that motion blurring also gives an extra deviation to the displacement estimation.

The second option is to insert two cameras in the sensor. These cameras are positioned at a certain distance which is exactly measured. This makes it possible to capture an image with the first camera, move the paper with a distance which is about equal to the distance between the cameras and capture the second image. The distance between the cameras plus the measured deviation give then the estimated displacement of the paper. Compared to the first option, this method needs just one displacement estimation. The paper transport within a printer is usually not continuous but stepwise. The paper transport is on hold after a certain distance, while one row of dots can be printed. If the distance between the cameras would be the same distance, then it gives the advantage to capture the paper surface when the paper is not moving and this will give a higher accuracy.

A third option is to combine the first and the second option. Two cameras are used to measure a big paper displacement while the paper is moving. Each time that the paper is displaced over the distance between the two cameras a measurement is done. Due to motion blurring there will be a bigger deviation in the displacement estimation, but since only a few measurements are needed, the displacement estimation can still be accurate enough.

We can conclude that the second option is capable of providing higher accuracy for paper displacement estimation, but the sensor then needs at least two cameras. A disadvantage of the second option is that only the distance between the two cameras can be measured. When other distances need to be measured, we need to use the first or third option.

4 Negative influences from hardware on accuracy

Image quality has a big influence on the accuracy of the paper displacement calculations, therefore it is important to improve the image quality. This chapter discusses the various negative influences on image quality.

4.1 Sensor's finger print and illumination pattern

The optical sensor uses a concentrated light source for illuminating the paper and a light sensitive chip for capturing the paper's surface. Both parts influence the quality of the captured image. The concentrated light source is not uniform and gives a particular pattern on captured image. In Figure 14 it is shown that in our sensor the top left and the top right corner are less illuminated than the centre part of the picture. Furthermore, each pixel on the light sensitive chip has a different pixel gain and a different offset. The pattern of pixel offsets is shown in Figure 15. Especially this is noticeable for the different columns. Difference in pixel gain is noticed by pixels that are lighter or darker than their surroundings.

Figure 14 - Pattern of concentrated light source and pixel gain (high contrast)

The pattern in Figure 14 is reconstructed by averaging 1000 captured images which were made with different paper surface. By averaging, the common characteristics get emphasised and in this case the sensor's pixel gain and illumination pattern are reconstructed.

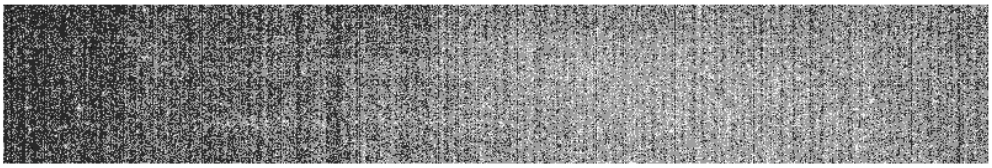


Figure 15 - Pattern of pixel offset (high contrast)

The algorithms described in the previous chapter are based on the correlation coefficient of groups of pixels. It is easy to imagine that if the pixel values are influenced by the illumination pattern, this will negatively influence the results of the displacement calculations.

4.2 Noise

Electronic noise is an unwanted signal characteristic for all electronic circuits. Depending on the circuit, the noise output from electronic devices has a big variance. This noise comes from many different electronic effects. Also in our sensor the effect of noise is noticeable. We captured 80 frames from the same piece of paper surface and averaged them to average out the noise. Then this noise free image is subtracted from one of the captured frames to calculate the noise pattern present on this captured frame. In Figure 16 a high contrast version of the present noise is shown.

Figure 16 - Noise on captured image (high contrast)

Although Figure 16 shows only the noise in one captured image there is still some pattern noticeable in the form of horizontal lines. Future research is needed to give an exact explanation, but we assume that somehow the pixels on the CMOS are not totally independent from each other.

Greyscale images are usually stored with a maximum of 256 intensity levels, where 0 represents black and 255 represents white. Images with noise can be written as the function:

$$v(i) = u(i) + n(i)$$

where *v(i)* is the observed value, *u(i)* would be the "true" value at pixel *i*, and *n(i)* is the noise perturbation. Measuring the amount of noise by its standard deviation, σ(n), one can define the signal noise ratio (SNR) as described by Buades et al.[1]:

$$SNR = \frac{\sigma(u)}{\sigma(n)}$$

where σ(u) denotes the empirical standard deviation of u(i),

$$\sigma(u) = \left(\frac{1}{|I|} \sum_{i \in I} \left(u(i) - \bar{u} \right)^2 \right)^{\frac{1}{2}}$$

and $\bar{u} = \frac{1}{|I|} \sum_{i \in I} u(i)$ is the average grey level value.

In our captured images we have an SNR of value 6.4. This means that the signal of the pattern is 6.4 times stronger than the present noise level. This amount of noise does influence the accuracy of the paper displacement estimations, as discussed section 5.2.

4.3 Dust

Image artefacts that result from sensor dust are a common problem when working with optical systems. These image artefacts can have a big influence on the accuracy of the paper displacement calculation.

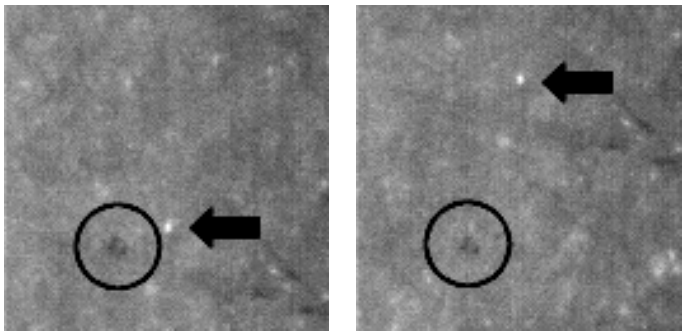


Figure 17 - Dust example

In Figure 17 an example of dust on the sensor can be seen. The arrow points out an area with enough contrast which can be used for determining the paper displacement. The encircled area indicates a dust spot. This dust spot is very small and experiments have shown that this size doesn't influence the overall accuracy of the optical paper position sensor, but they can have big influence when the dust spot is bigger. The reason for this is that the algorithm chooses an area with the highest contrast for finding the correlation window. When a dust spot is relatively small it doesn't give a high contrast difference and the algorithm doesn't use this part of the image as correlation window. When the dust spot is bigger, then it can become the area with highest contrast and can become a problem for calculating the displacement.

Therefore we searched for techniques to overcome this problem. Three different approaches are possible. The first one is to reconstruct the pattern behind the dust spot as best as possible, and use the newly created information in the calculation of the paper displacement. C. Zhou and S. Lin have pointed out that this is possible [18]. They presented a method for reconstructing the pattern behind a dust pixel. The second approach is to detect the dust particles and don't use these regions for calculation. The third option is to neutralize the dust spot in such a way that it doesn't influence the accuracy calculations anymore.

The first approach is mainly focussed on the aesthetic quality of the pictures. We think that this approach could improve the displacement estimation, but is too complex for our needs. The second approach works about the same as the third approach. They both avoid using the actual pixel values in the dusty region. The second approach doesn't use them at all, while the third approach uses corrected values. We noticed in our experiments that via flat-field correction, visible dust spots are neutralized. Therefore we have chosen to go for the third approach. Flat-field correction is discussed in section 6.1.

4.4 Captured images of moving paper

An interesting topic for investigation is the accuracy of the optical paper position sensor while the paper is moving along the sensor with a continuous motion. The effect that will occur is motion blurring. The details visible on the paper surface will get spread out over a certain distance, depending on the shutter time and the speed of the paper displacement. Since the paper within a printer also moves with a certain speed it is important to check the accuracy while images are captured during paper movement. Figure 18 shows motion blurring, as details from the paper surface are smeared out from left to right.

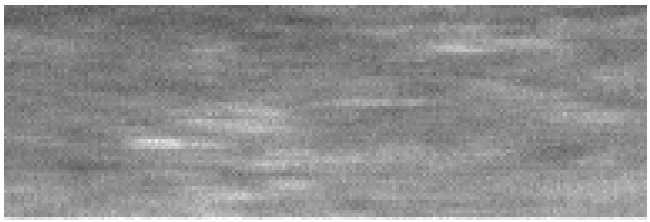


Figure 18 - Motion blurred captured image

Experiments with motion blurred captured images have shown that the accuracy decreases as motion blurring gets more intense. However, we were not able to simulate the motion blurring as it occurs when using the sensor exactly, and hence future research is needed to specify the influence accurately.

The paper transport within a printer is accelerating and decelerating all the time. It can occur that the sensor needs to find the displacement between a motion-blurred image and a non-motion-blurred image. This is prone to difficulties since the images do not have the same pattern. When the degree of motion blurring is known, this difference between the images can be corrected. This can be done by either motion-blurring the non-motion-blurred image or undo the motion-blurring from the motion-blurred image. In our research we did not focus on these techniques, so future research is needed to conclude whether they work sufficiently.

4.5 Depth of field

Depth of field is defined as the range of object distances within which objects are imaged with acceptable sharpness [11]. Although a lens can precisely focus at only one distance, the decrease in sharpness is gradual on either side of the focused distance, so that within the depth of field the influence of the lack of sharpness on the displacement estimation algorithm is imperceptible.

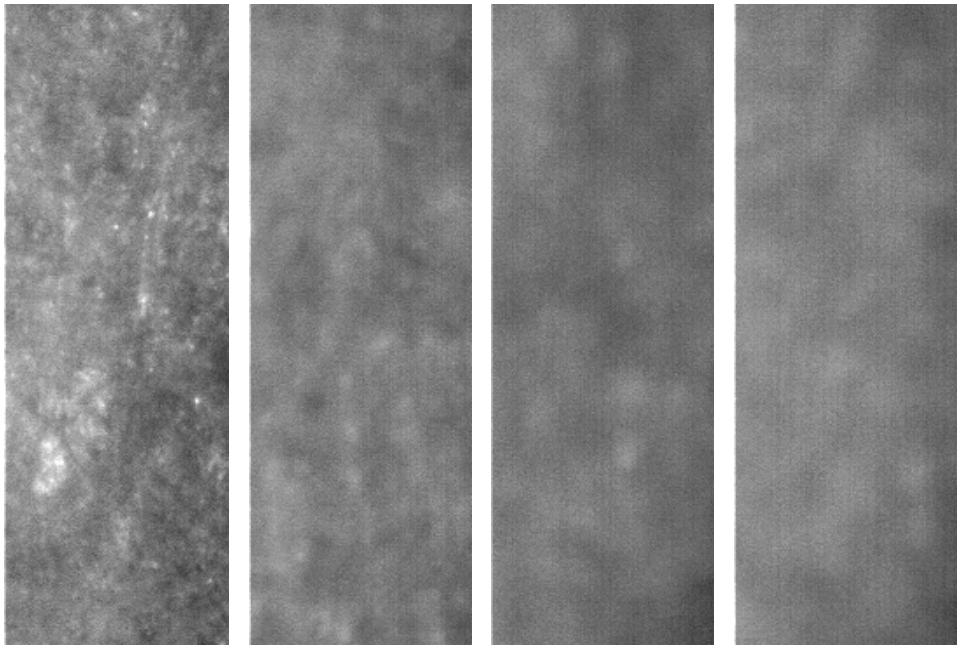


Figure 19 – Sharpness at distance of 0µm, 50µm, 100µm and 250µm from the sensor.

Note that the sharpness of the captured images may not be constant. Paper within a printer is constantly transported and it does happen that the paper loses the contact with the sensor and that the paper surface moves in the direction perpendicular to the paper surface.

As is shown in Figure 19, the sharpest picture is retrieved when there is no distance between the paper and the sensor. The sharpness decreases quite quickly when the paper surface moves away from the sensor. Though still a distinct pattern can be seen, the accuracy of the displacement measurement decreases when too little amount of pattern can be seen. Experiments indicate that still a reasonable accuracy is achieved in displacement estimation up till a distance of 250µm from the sensor, but when the paper surface moves further away not enough detail is captured and the displacement algorithm fails. Since we did not focus our research on this point, future research is needed to determine the exact influence.

4.6 Optical distortions

Lens distortion is one of the main factors affecting the quality of retrieved images when using optical systems. Two examples of lens distortion can be seen in Figure 20.

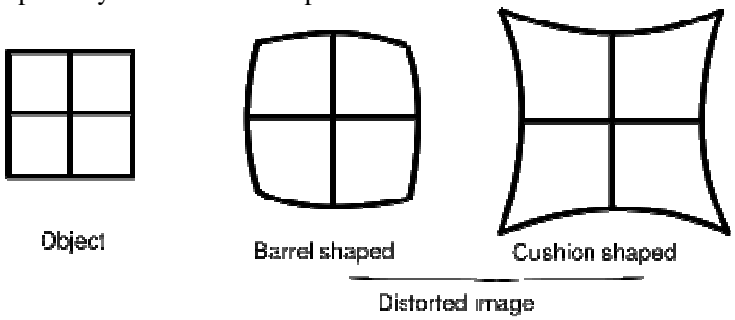


Figure 20 - Examples of lens distortion

The problem with lens distortion is that the captured images do not represent the real structure and shapes, but deformed ones. The paper displacement measurement algorithm is designed to find the best match of a certain area of one captured image within the other captured image. When these patterns are differently shaped due to optical distortions, the best match doesn't necessarily need to be an accurate match. Thus the accuracy is heavily influenced by optical distortions.

Optical distortions are fairly easy to compensate for with different techniques [15]. These techniques have been tested thoroughly and generate good results. A quick experiment showed us that these optical distortions are not noticeably present in our current sensor, but future research is needed to quantify this exactly. We have decided not to test these algorithms, and their influence on the accuracy.

5 Analysis of sensor's algorithms

This chapter provides an in-depth analysis of the subpixel level algorithms implemented in the sensor's software. We first discuss the polynomial fitting through 3 points algorithm and then we analyse the bilinear interpolation algorithm.

5.1 Polynomial fitting through 3 points

The polynomial fitting through three points algorithm calculates the subpixel displacement in both axis separately, using only the correlation coefficient next to the optimal correlation coefficient. This is not a very refined approach, but can be improved when more correlation coefficients are included in the calculation to find the optimum. An algorithm that does this is the N-Cubed displacement estimation algorithm which is introduced by Dr. Raymond Beausoleil from HP Laboratories [4]. This algorithm uses the maximum correlation coefficient and all eight surrounding correlation coefficients within the matrix to define sub-pixel-level displacement on the x- and in y-axis. We will discuss this algorithm in section 6.3.1.

5.2 Bilinear interpolation

We noticed that the implemented bilinear interpolation method, as explained in section 3.2.3, does not work well. The presence of noise and the way the estimation is calculated result in structural deviation in the subpixel-level displacement. To analyse this method, we have modelled it in Matlab[13].

We created two sinusoids which represent the two signals where between we want to determine the subpixel level displacement with the bilinear interpolation algorithm. These are shown in Figure 21. We have to sample and quantize the signals in order to simulate the process of what happens within the sensor. There, the projected image on the CMOS/CCD is also converted to a digital signal via sampling and quantizing. Before sampling and quantizing the two sinusoids we shift one sinusoid with respect to the other with a small shift. This shift is smaller than the sample width and is indicated as the 'shift in pixel pitch'.

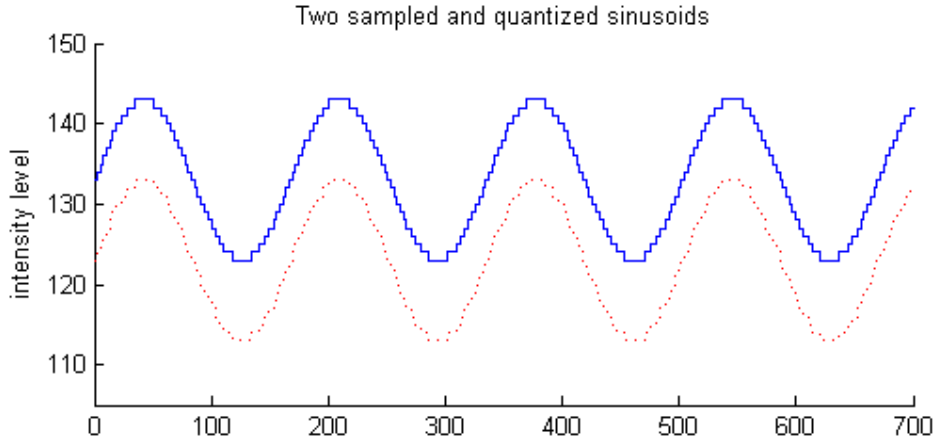


Figure 21 - Two sampled and quantized sinus signals. The sampling and quantizing is noticeable by the sharp corners in the maxima and minima of the signals.

During the sampling and quantizing operation within the sensor, noise is affecting the retrieved values. The noise level within the sensor has a signal-to-noise ratio (SNR) of 6.4.

We simulate this noise by adding Gaussian noise with the same SNR to the two signals. The result is shown in Figure 22.

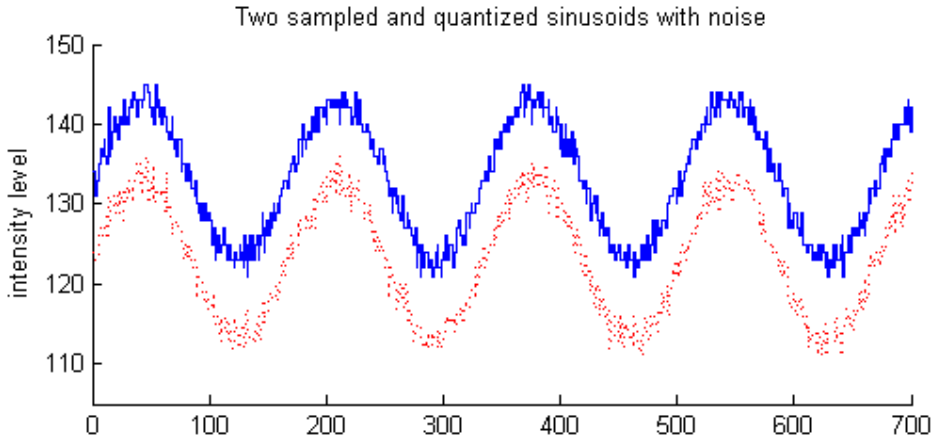
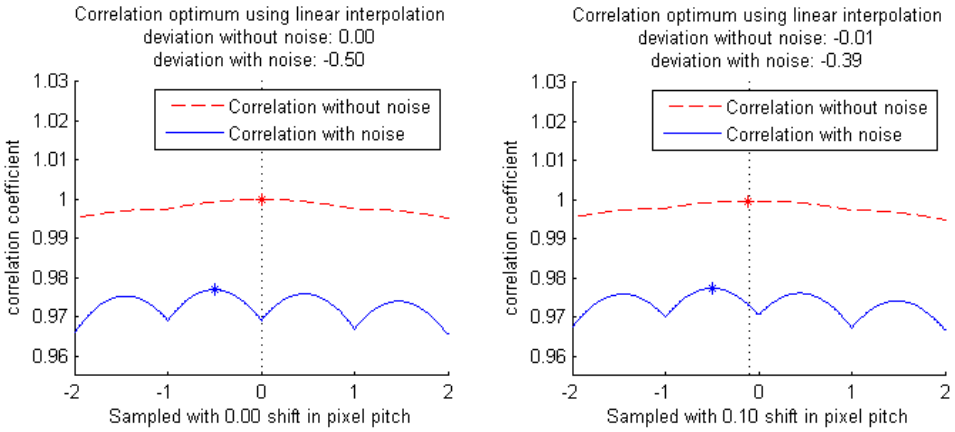


Figure 22 - Two quantized sinus signal with noise

We implemented bilinear interpolation algorithm in Matlab, such that it can be used for one-dimensional signals. Now we are able to determine the subpixel displacement between these two sinusoids.

In Figure 23 we show the calculated correlation coefficients. The sampled values of the second sinusoid act as a correlation window. We 'move' this correlation window over the sampled values of the first sinusoid. After each step we calculate the correlation coefficient of the two signals. This is the same method as the pixel level approach discussed in section 3.2.2. Since we know that the displacement of the signals is less than one sample width we only calculated the cross correlation over a range of 5 samples. In Figure 23 it is visible that the correlation coefficients are only calculated between -2 and +2. The correlation coefficients between the discrete sample points are calculated with the bilinear interpolation algorithm as described in section 3.2.3.

Within the algorithm, the place where the maximum correlation coefficient is found is determined as the estimated displacement. In Figure 23 is shown that the maximum correlation coefficient is often found with a structural deviation compared to the real displacement, when the displacement is estimated between the two noisy sinusoids. Note that these graphs actually consist of calculated correlation coefficients, but for better visibility we connected these values to form a smooth function.



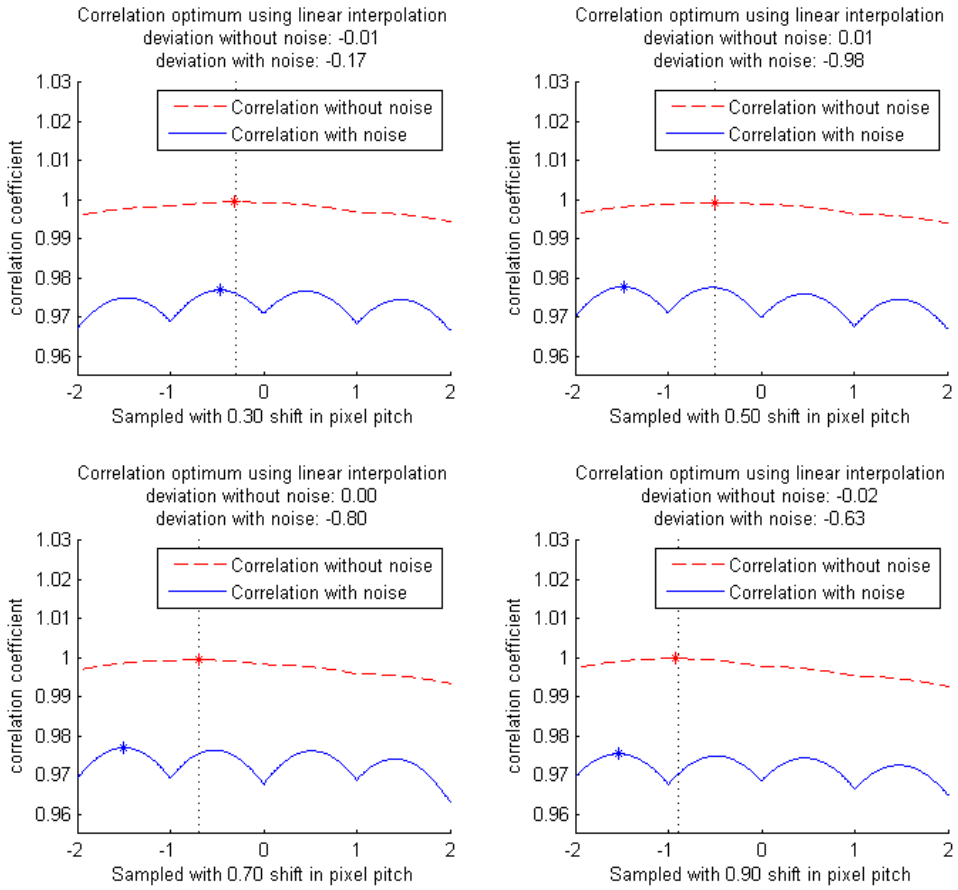


Figure 23 - Graphs of correlation coefficients for different shifts in pixel pitch

Another remarkable issue is that the correlation coefficients found at the pixel level are in these graphs seen as the local minima. Between these points there is a smooth parabola formed by the correlation coefficients calculated with the bilinear interpolation approach. This pattern can be also noticed in the bilinear interpolation example in Figure 12 and Figure 13 in section 3.2.

This effect can be explained in the following way. To provide a better overview, we present two lemmas.

1. The noise we have in our captured images is distributed normally, with a mean value of zero. Applying a mean filter on a dataset containing noise, reduces therefore the noise.

2. Noise is random and influences cross correlations negatively. The more noise is present the worse it is for the cross correlation. Correlating two noiseless signals will provide a high correlation coefficient. Correlating one noisy signal with a noiseless signal will provide a lower correlation coefficient. Correlating two noisy signals will provide an even lower coefficient. This effect is depicted in Figure 24.

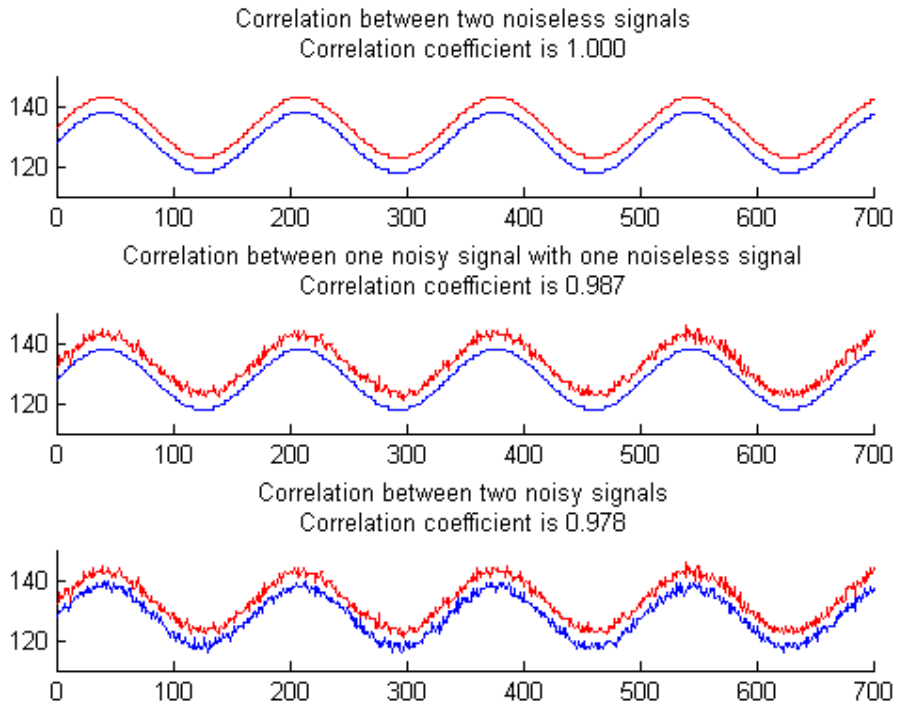


Figure 24 - Demonstration that correlation coefficient decreases when the total amount of noise increases

The bilinear interpolation algorithm only takes a subset of the interpolated values as is described in section 3.2.3. The interpolated values which are positioned exactly between the original pixel values have the mean value of the surrounding pixels. Using the first lemma we can conclude that these interpolated values have a lower noise level.

From the second lemma we can conclude that if one of the two signals has lower amount of present noise, it will give a higher correlation coefficient. The subsets containing values between the original pixel values will therefore give a higher correlation coefficient when cross correlated with the values from the correlation window. This explains why correlation coefficient graphs of noisy signals in Figure 23 have such a typical pattern.

We now showed that the current implementation of this algorithm has a serious defect and therefore not suitable for estimating accurately the displacement of two signals containing noise. The presence of noise always causes a deviation in the accuracy, but it should not be as bad as in the example presented here.

In section 6.3.2 we present a corrected bilinear interpolation algorithm, which does not have this serious defect.

6 Proposed improvements

This chapter contains the proposed improvements for the software. These improvements tackle some of the presented problems in chapter 4. The problems we want to tackle are the sensor's finger print and illumination pattern (section 4.1), presence of noise (section 4.2), dust robustness (section 4.3) and we present two algorithmic improvements on the sensor's displacement estimation algorithms (section 5.1 and 5.2). Further we propose the use of Fourier transformations to speed up the displacement estimation algorithms. We discuss these in section 6.4.

6.1 Flat-field correction

The sensor's finger print and the illumination pattern can be compensated by "Flat Fielding" or "Shading Correction" [9].

Flat fielding requires the acquisition of two calibration frames. First, a bias frame or a dark frame (IB) should be taken. Bias clears the camera of any accumulated charge and reads out the cleared CMOS/CCD. The resulting image is a low signal value image. In this image, all of the pixels have approximately the same value, which consists of the electronic offset of the system of the inherent structure of the CCD. The bias frame should be taken with same exposure time as is used with capturing the image. The second calibration frame, the flat field frame, measures the response of each pixel in the CCD array to illumination and is used to correct for any variation in illumination over the field of the array. As we have seen the optical system introduces some variation in the illumination pattern over the field of the array. The flat fielding process corrects for uneven illumination, if that illumination is a stable characteristic of each object exposure. We created the flat field frame by averaging thousand taken images of different places of the paper surface. Averaging makes the common characteristics between these images stronger. The only common feature of these thousand images is the light distribution over the CCD. The acquired pattern via averaging reflects therefore the illumination pattern and thus we can use this as flat field frame.

Flat fielding can be represented by the following equation:

$$IC(x,y) = \frac{(IR(x,y) - IB(x,y)) \times M}{IF((x,y) - IB(x,y)}$$

where IC is the flat-field corrected image; IR is the non-corrected image; IB is the bias or dark frame; M is the average pixel value of the flat field frame; and IF is the flat field frame.

In Figure 25 the different images involved in the flat-field correction are shown. Observe that in the flat-field frame the lower left corner is more illuminated than the upper right corner. This is also noticeable in the non-corrected image, but corrected in the corrected image.

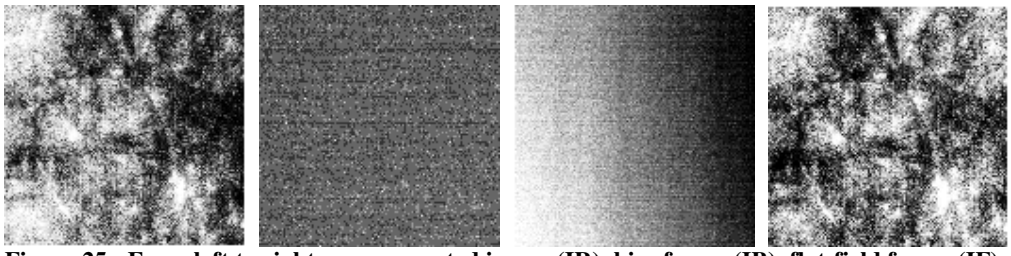


Figure 25 - From left to right: non-corrected image (IR), bias frame (IB), flat-field frame (IF) and the corrected image (IC) – all pictures are displayed in high contrast to emphasize the effects of flat-field correction.

Flat-field correction also provides a good solution for the removal of image artifacts due to sensor dust. A dust artifact is a darker or black spot which is constantly present in all the captured images and can be compared to a non-illuminated area. We are not directly interested in reconstructing this area, but in eliminating the dark contrast, such that it doesn't influence the displacement estimation. Flat fielding offers this solution since the dust spot will also be corrected and removed. Figure 26 shows the result through of a dust dot after flat-field correction. Besides that the dot is removed, also the light intensity is more evenly spread across the image. Note that the dust does not influence the bias frame and it is therefore not depicted again.

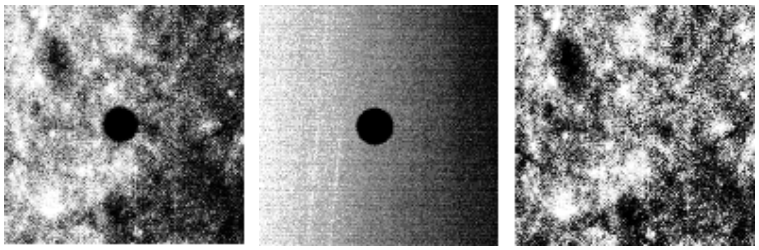


Figure 26 - Demonstration of flat-field correction with dust. From left to right: non-corrected image (IR), flat-field frame (IF) and the corrected image (IC)

6.2 Noise reduction

We have demonstrated in chapter 5 that noise has a negative impact on the accuracy of the displacement estimations. There are several methods of improving the quality of images containing noise. Many of these methods are made for increasing visual appearance and are not specifically meant for improving image alignment operations. There is little documentation about the effects of these filters on the accuracy of image displacement estimation. Therefore it is an interesting topic for our research to see if these filters can have an added value to displacement accuracy.

We will use and test multi-sampling and various spatial filters as methods for improvement.

6.2.1 Multi-sampling

The best way of retrieving the highest quality image is by averaging many captured images of the same paper surface. Averaging the result of more measurements is a common method for retrieving samples with higher accuracy. Taking ten samples and averaging those, results in an image with only $\frac{1}{\sqrt{10}}$ times of the amount of noise.

This method can only be used when the paper is not moving, because averaging images of different spots of paper will result only in motion blurred images, of which not accurately the displacement can be estimated.

6.2.2 Spatial filters

We have selected several simple spatial filters to reduce the noise in images. We used the arithmetic mean filter, geometric mean filter, harmonic mean filter, median filter and an adaptive filter in our research. These filters are discussed in detail by Gonzalez et al. [5].

Spatial filters work in the spatial domain and do their calculations directly on the intensity values of the pixels. All the filters we considered use a neighbourhood S_{xy} of $m \times n$ pixels, centred at point (x, y), to calculate the new intensity value of this pixel. The simplest example is a mean filter which calculates the arithmetic mean of the pixels in the region defined by S_{xy} for pixel (x, y). See appendix B for the descriptions of all the considered filters.

Since there is hardly any documentation how these filters can help in the estimation of paper displacement, it is hard to predict how these filters will perform. Based on measurements we will conclude in chapter 8 whether these filters are an added value or not.

6.3 Algorithmic accuracy improvement

In this section, we discuss two possible subpixel displacement algorithms, which can achieve a higher accuracy. The first algorithm is the Standard N-Cubed displacement estimation algorithm developed by HP laboratories. The second algorithm is an corrected version of the bilinear interpolation algorithm. This corrected algorithm does not give the particular pattern as is shown in Figure 23 when calculating the correlation coefficients.

6.3.1 Standard N-Cubed displacement estimation algorithm

The N-Cubed displacement estimation algorithm is developed by HP Laboratories [4] and has been proven in many applications, including HP's handheld scanner and HP's high-accuracy OMAS (Optical Media Advance Sensor). The basis of this algorithm is that the correlation surface is modelled as the general second-order two-dimensional Taylor-series expansion.

$$f(x, y) = a_{00} + a_{10}x + a_{01}y + a_{20}x^2 + a_{11}xy + a_{02}y^2$$

There are six coefficients in the second-order fitting function that need to be calculated. This is done using the 9 correlation values visible in Figure 27. $C_{0,0}$ represents the maximum correlation value that is found within the correlation matrix.

$$\begin{bmatrix} C_{-1,1} & C_{0,1} & C_{1,1} \\ C_{-1,0} & C_{0,0} & C_{1,0} \\ C_{-1,-1} & C_{0,-1} & C_{1,-1} \end{bmatrix}$$

Figure 27 - Correlation values used for N-cubed algorithm

The method of least squares fitting is used to calculate the six coefficients of the paraboloid. The least squares method is a regression analysis method, thus instead of calculating an exact fit through all the correlation coefficients, a best fit is found. Gao et al. [4] derived the following 6 functions for determining the coefficients.

$$\begin{cases} a_{00} = \frac{1}{9}\left[5C_{0,0} + 2\left(C_{-1,0} + C_{0,1} + C_{1,0} + C_{0,-1}\right) - \left(C_{-1,1} + C_{1,1} + C_{1,-1} + C_{-1,-1}\right)\right] \\ a_{10} = \frac{1}{6}\left[\left(C_{1,1} + C_{1,0} + C_{1,-1}\right) - \left(C_{-1,1} + C_{-1,0} + C_{-1,-1}\right)\right] \\ a_{01} = \frac{1}{6}\left[\left(C_{-1,1} + C_{0,1} + C_{1,1}\right) - \left(C_{-1,-1} + C_{0,-1} + C_{1,-1}\right)\right] \\ a_{11} = \frac{1}{4}\left[\left(C_{1,1} + C_{-1,-1}\right) - \left(C_{-1,1} + C_{1,-1}\right)\right] \\ a_{20} = \frac{1}{6}\left[\left(C_{-1,-1} + C_{-1,0} + C_{-1,1} + C_{1,-1} + C_{1,0} + C_{1,1}\right) - 2\left(C_{0,-1} + C_{0,0} + C_{0,1}\right)\right] \\ a_{02} = \frac{1}{6}\left[\left(C_{-1,1} + C_{0,1} + C_{1,1} + C_{-1,-1} + C_{0,-1} + C_{1,-1}\right) - 2\left(C_{-1,0} + C_{0,0} + C_{1,0}\right)\right] \end{cases}$$

The second-order two-dimensional Taylor-series expansion using these coefficients form an elliptic paraboloid (Figure 28) embedded in a three-dimensional space from which the maximum is determined.

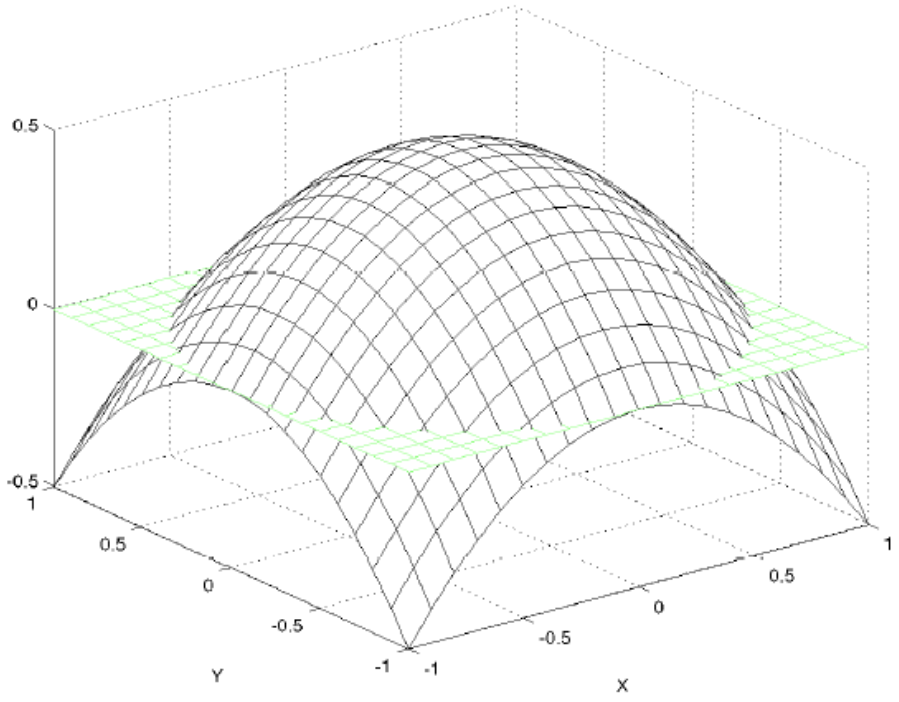


Figure 28 - Example of paraboloid

The maximum is determined by solving the following two equations

$$\begin{cases} x_0 = \dfrac{a_{01}a_{11} - 2a_{10}a_{02}}{4a_{20}a_{02} - a_{11}^2} \\ y_0 = \dfrac{a_{10}a_{11} - 2a_{01}a_{20}}{4a_{20}a_{02} - a_{11}^2} \end{cases}$$

The coordinates $\{x_0, y_0\}$ form the maximum likelihood displacement estimation.

We will demonstrate this algorithm on the example values we presented in section 3.2.3. For the polynomial fitting through 3-point algorithm we only needed the maximum correlation coefficient and four of its surrounding correlation coefficients. For the N-cubed algorithm we need all eight surrounding correlation coefficients. These are presented in Table 2.

Table 2 - Maximum correlation coefficient and the surrounding values within the matrix

0.998890	0.999133	0.998956
0.999065	0.999215	0.999091
0.998868	0.999041	0.998824

Using these correlation coefficients we can calculate the six coefficients for the second-order two-dimensional Taylor-series expansion. Filling out the formulas for calculating the coefficients gives us the following values.

$$\begin{cases} a_{00} = 0.9992 \\ a_{10} = 8.000 \times 10^{-6} \\ a_{01} = 4.1000 \times 10^{-5} \\ a_{11} = 2.7500 \times 10^{-5} \\ a_{20} = \text{-}1.8067 \times 10^{-4} \\ a_{02} = \text{-}1.7167 \times 10^{-4} \end{cases}$$

And the position of the maximum of the paraboloid is then calculated as

$$\begin{cases} x_0 = 0.0314 \\ y_0 = 0.1219 \end{cases}$$

In section 3.2 we already calculated a pixel level displacement along the x-axis of 0 pixels and along the y-axis of 20 pixels. When we add the calculated subpixel level displacement we get an estimation of 0.03 pixels along the x-axis and 20.12 along the y-axis. Multiplying with the pixel size gives us a 0.08 µm displacement on horizontal axis and 50.30µm displacement on vertical axis. The deviation using this algorithm in this example is only 0.25µm, which is smaller than the deviation we had with the 3-points polynomial fit algorithm.

6.3.2 Corrected bilinear interpolation estimation algorithm

In section 5.2 we showed that the current implementation of the bilinear interpolation algorithm gives a structural deviation. We will present here a corrected algorithm that does not lead to such a structural deviation.

In the current implementation only the best match window is interpolated and only a subset of the interpolated data is used for the cross correlations. We explained in section 5.2 that when noise is present, usage of only a subset of data leads to a structural deviation, since higher correlation coefficients are calculated when the noise level becomes lower due to bilinear interpolation.

To remove the structural deviation we must make sure that the noise level of the used values does not alter when doing a cross correlation. This can be done by bilinear interpolating both the correlation window and the best match window and using all the values for calculating the correlation coefficients while doing the subpixel steps. The interpolated values between the original pixel values will have still a lower amount of noise than the values on pixel level, but since both interpolated images have this, the overall amount of noise is the same when

calculating the correlation coefficient. We present this method as the corrected bilinear interpolation estimation algorithm.

When using the corrected bilinear interpolation algorithm on the same two sinusoids from section 5.2 we get the graphs in Figure 29 of correlation coefficients for different shifts in pixel pitch.

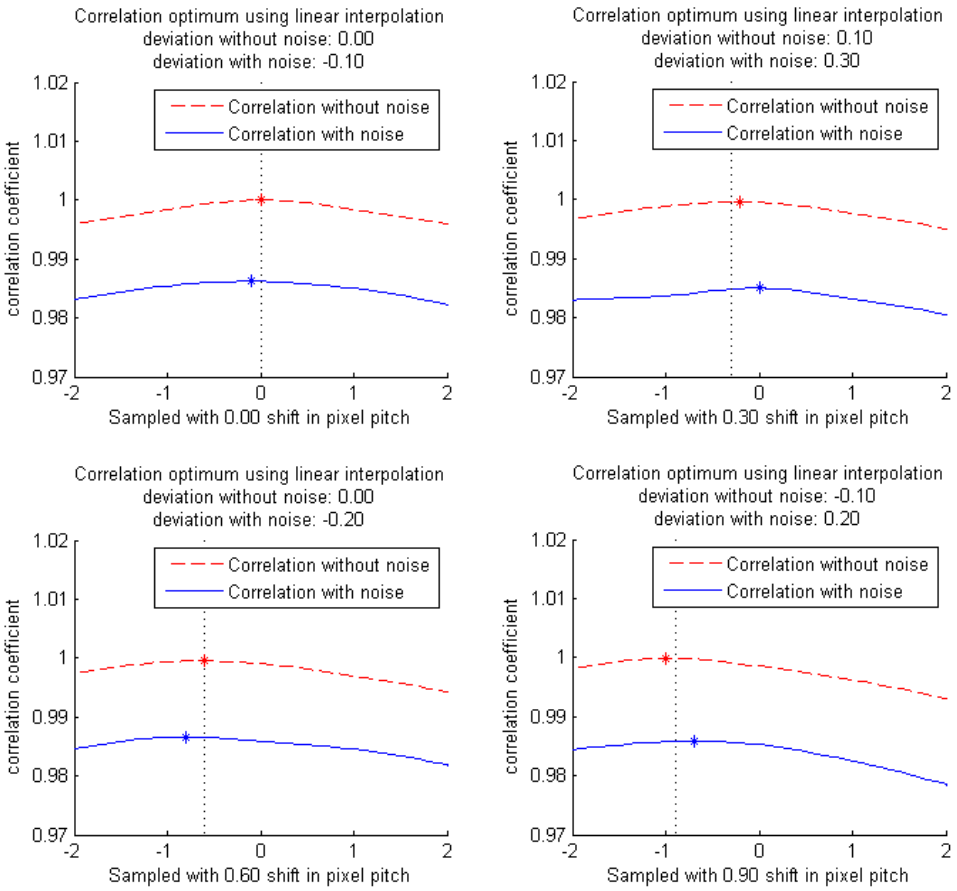


Figure 29 - Graphs of correlation coefficients for different shifts in pixel pitch using the corrected bilinear interpolation algorithm

We used now all interpolated values of both signals. From Figure 29 we can conclude that although there is still a deviation in noticeable, we now have a smooth function without the different parabola as can be seen in Figure 23.

We will demonstrate this algorithm on the example values we presented in section 3.2.3. There we already determined that the displacement is 20 pixels and we determined the correlation window and best match window. Since the displacement of paper is not exact a multitude of the pixel size, we expect to find the real displacement somewhere within one pixel distance from the estimated pixel level displacement.

We calculate the interpolated values with bilinear interpolation for both the correlation window and the best match window. Figure 30 shows the 8 by 8 correlation window and best match window. These windows are cut from the original images and enlarged for better visibility. Figure 31 shows the five times bilinear interpolated correlation window and best match window. These are 40 by 40 pixels.

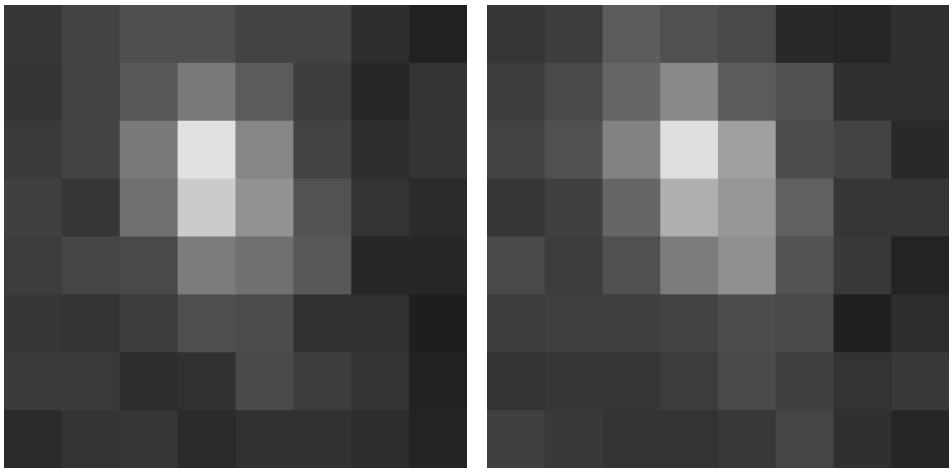


Figure 30 - Correlation and best match window

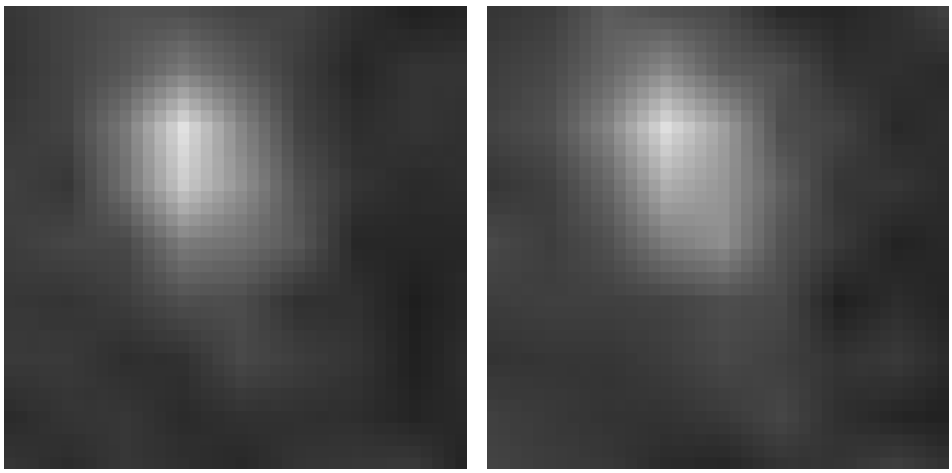


Figure 31 - Bilinear interpolated correlation and best match window

In the corrected algorithm, we use all the interpolated values to determine the subpixel level paper displacement. As in section 3.2.2, the correlation window is moved 4 subpixels in each direction, starting from the pixel where the maximum was found on pixel level. This gives us a matrix of 81 correlation coefficients which is visualized in Figure 32 and Figure 33.

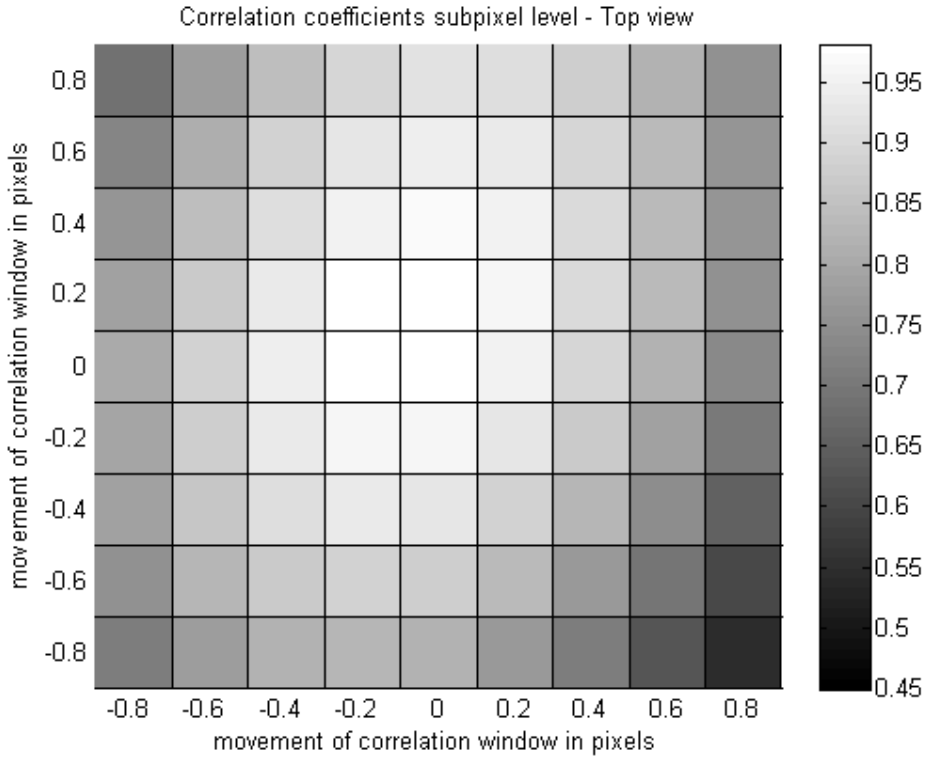


Figure 32 - Top view of matrix of correlation coefficients on subpixel level

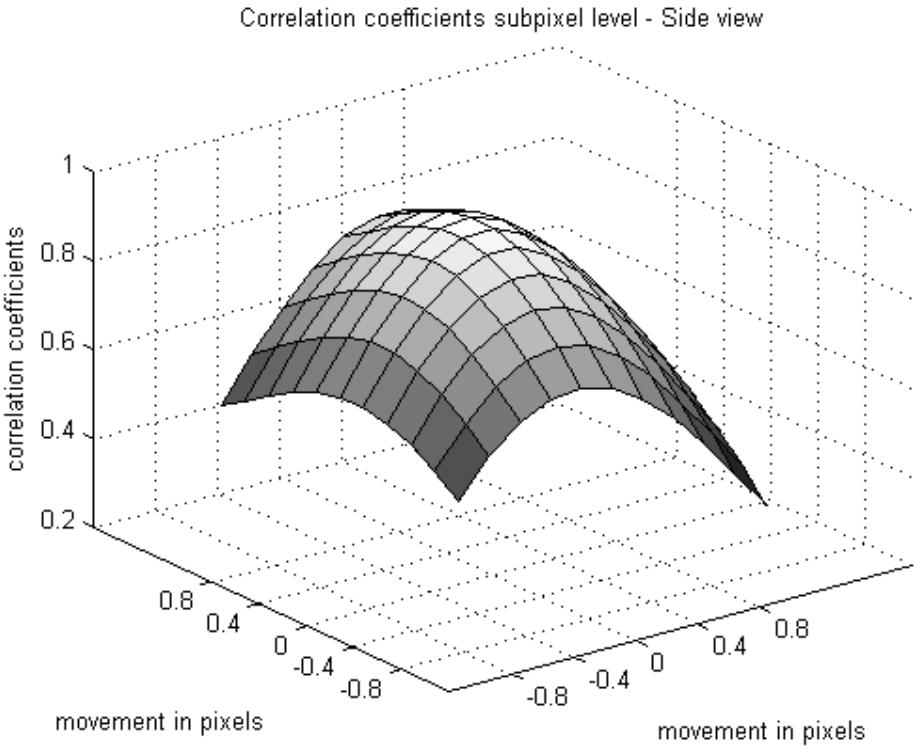


Figure 33 - Side view of matrix of correlation coefficients on subpixel level

The maximum value is found at point (0, 0.2) in the correlation coefficient matrix. Now we calculate the new optimum values.

$$x\_optimum = 0$$
$$y\_optimum = 20 + 0.2 = 20.2$$

Multiplying with the pixel size gives us a 0µm displacement on horizontal axis and 50.5µm displacement on vertical axis. The deviation using this algorithm in this example is 0.45µm.

In Figure 33 we can notice that the correlation coefficients form a normal paraboloid with only one optimum. This was not the case in Figure 13 where four optima can be noticed.

Hence, in this example we have showed that the corrected bilinear interpolation algorithm works better than the version implemented in the sensor where the estimation had a structural deviation.

6.4 Algorithmic speed improvement

Image alignment algorithms generally fall into two groups: (1) algorithms based on statistical analyses of images, such as correlations of the reference (non-displaced) frame with respect to the comparison (displaced) frame; (2) algorithms based on Fourier transforms of the individual frames. Algorithms of the latter type can be divided into two groups, phase correlation method and phase delay detection. They both take advantage of the translation property of Fourier transforms [4]. The sensor's built-in algorithms of the sensor described in section 3.2 are based on statistical analyses, but we think that a significant speed increase can be achieved by using Fourier transforms.

With the statistical analysis methods, we calculated the cross correlation coefficients by moving a correlation window along the target window, while calculating the correlation coefficient for each step. W. van Belle [14] describes a complexity analysis of the two possible ways of calculating the correlation coefficient matrix. A Fourier transform takes around $s_x \cdot s_y \log_2(s_x \cdot s_y)$ operations, where s_x and s_y are respectively the width and the height of the correlation window in pixels. Combined with the conjugation, inproduct and finding of the maximum value, this results in a total calculation time of around $3s_x \cdot s_y \log_2(s_x \cdot s_y) + 3s_x \cdot s_y$ operations. The spatial cross correlation, as explained in section 3.2.2, needs $(s_x \cdot s_y)^2$ operations.

The analysis of W. van Belle is based on the assumption that the correlation window has the same size as the window where the best match window must be found. We introduced in section 3.3 margins which define the region in which the best match window is found. This gives us $(s_x \cdot s_y) \times 2 \cdot (m_x \cdot m_y)$ operations where where s_x and s_y are respectively the width and the height of the correlation window and m_x and m_y are the width and height of the margins.

In the spatial cross correlation, the correlation window is smaller than the region in the second window where it is searched for. When using FFT not only the correlation window but the correlation window including the margins is correlated with the second window. Since we use a margin of 10 pixels in x-direction and 300 pixels in y-direction, FFT does cost a lot more computations for the smaller correlation windows, but an advantage is gained in the bigger correlation windows. Table 3 gives an overview of the difference in calculations between FFT and spatial cross correlation.

Table 3 - Needed computations for doing cross correlations with margin of 10 pixels in x-direction and 300 pixels in y-direction

	Amount of needed computations with margin of 10 pixels in x-direction and 300 pixels in y-direction	
Correlation window size	FFT	Spatial cross correlation
4x4	644639	192000
8x8	768903	768000
16x16	1026974	3072000
24x24	1296870	6912000
32x32	1577888	12288000
48x48	2171559	27648000
64x64	2805648	49152000

We can conclude that calculating the correlation coefficient matrix is done faster via Fast Fourier transforms than via spatial cross correlation for correlation windows bigger than 8x8 pixels when using a margin of 10 pixels in x-direction and 300 pixels in y-direction.

The method of calculating the correlation coefficients via Fast Fourier Transforms (FFT) is based on the following three lemmas.

1. The convolution theorem states that the Fourier transform of a convolution is the point wise product of Fourier transforms [17].

2. A convolution of two functions involves flipping (rotating by 180° degrees) one function about its origin and sliding it past the other. At each displacement in the sliding process, we perform a computation. This is comparable to the correlation of two images what we have shown in section 3.2.2., but there we didn't flip one of the signals [5].

3. In the frequency domain, rotating a matrix around its origin can be done by conjugating each complex number in the matrix [17].

From these lemmas we can derive the following function which calculates the correlation coefficient matrix of matrix A and B.

$$C(\delta_x, \delta_y) = F^{-1}\left(F(A) \times F(B)^*\right)$$

where F is the Fourier transform function and $C(\delta_x, \delta_y)$ is the correlation coefficient matrix and δ_x and δ_y denote the shift of matrix B over matrix A.

The Fourier transforms of both A and B are calculated. The standard behaviour of a multiplication in the frequency domain is convolution according to lemma 1. Since we want to have the effect of a correlation and not a convolution we rotate the matrix B by conjugating each complex number in that matrix. Then we do a point-wise multiplication between the Fourier transforms of A and B. On the result of this multiplication we use the Inverse Fourier Transform function to convert the correlation coefficient matrix to the spatial domain. This operation gives a matrix of correlation coefficients. The displacement between the two images is then determined by the position of the maximum correlation coefficient.

7 Measuring the sensor's accuracy

This chapter contains the description of our working methods. We first define which datasets are needed and how we generate them. Then, we discuss how and which measurements are done to realize the results.

7.1 Datasets

We created different datasets on which we can test different algorithms and options. The datasets are constructed in such way that they represent what happens during ordinary usage of the sensor.

7.1.1 Creating the datasets

We have constructed a testing environment in which we can measure the accuracy of the sensor. We are capable of displacing paper along one axis along the sensor with accuracy up to ±0.2µm. The position can be read with a resolution of 50nm. Henceforth, we refer to this position as the 'exact position' although it only has accuracy up to ±100nm. In Figure 34 a schematic overview of creating the datasets can be seen.

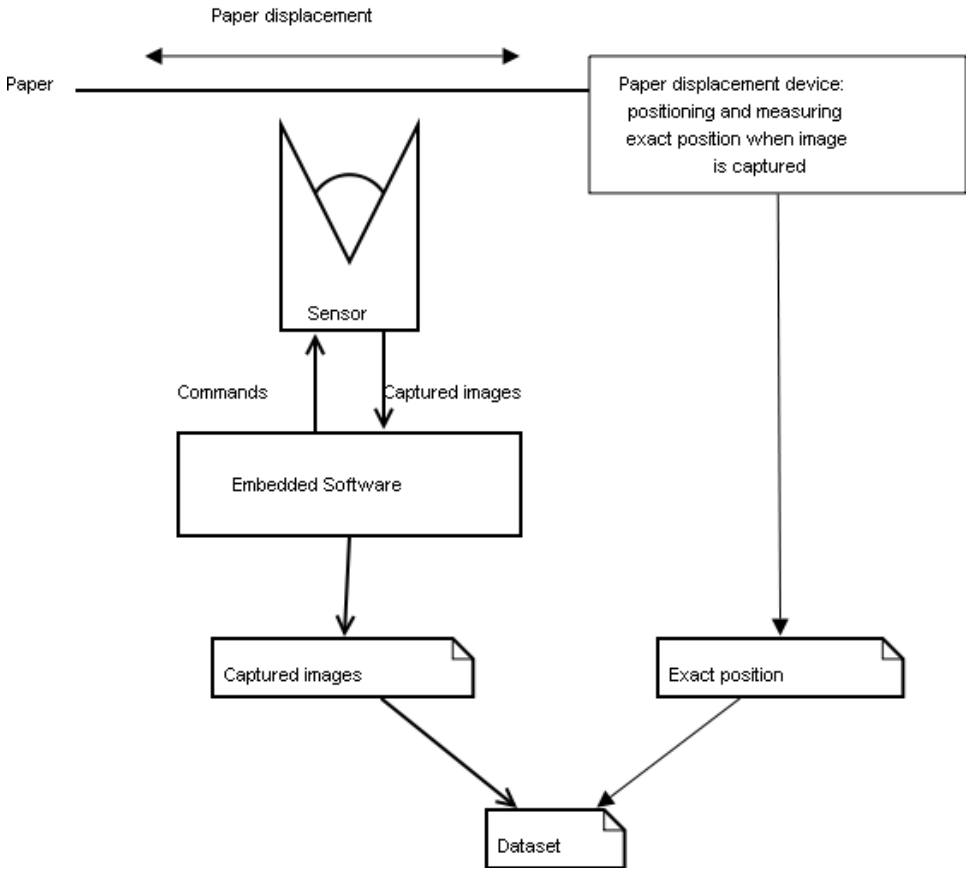


Figure 34 - Schematic overview of creation dataset.

The paper is displaced to various positions. At each position the sensor is triggered and captures the paper surface into a digital image. At the same time the exact position is read out from the paper displacement device. Each dataset consists of a set of captured images of the

paper surface combined with their exact position. The displacement of each two subsequent taken images is not more than 200µm which is equivalent to 80 pixels in the digital image.

The captured images have a resolution of 128 by 800 pixels and are made with 300µs shutter time. This has a consequence that only 2mm of the paper surface is visible in the y-axis and 0.32mm in the x-axis. We chose the 300µs shutter time, because the captured images have then good contrast and the fibres within the paper surface are also visible for the human eye.

7.1.2 Created datasets for the measurements

With the setup as described in section 7.1.1 we captured 1310 images at 131 different positions. From these captured images we created 8 datasets which simulate some of the different aspects we want to test.

Normal and multi-sampled dataset

First we group together the captured images according to the position where it was captured. To create the first dataset we choose randomly one image from each group and combined with its position as a dataset. The second dataset is created by averaging the 10 images per group and combined with its position as the multi-sampled dataset. This dataset is used for measuring the effect of multi-sampling.

Creating dusty datasets

We assume that a dust spot stays in the same place on the captured image. The dust spot is usually not placed in the depth of field and therefore out of focus. This results in that dust does not have sharp edges. To simulate dust we created a white frame with a single big black dot on it with soft edges. To add this dust dot to the images we multiply this white frame with the image frame. Multiplication is used to simulate the effect of the soft edges of a dust particle. The structure of the paper surface behind the soft edges is still there, but less illuminated. The result is shown in Figure 35.

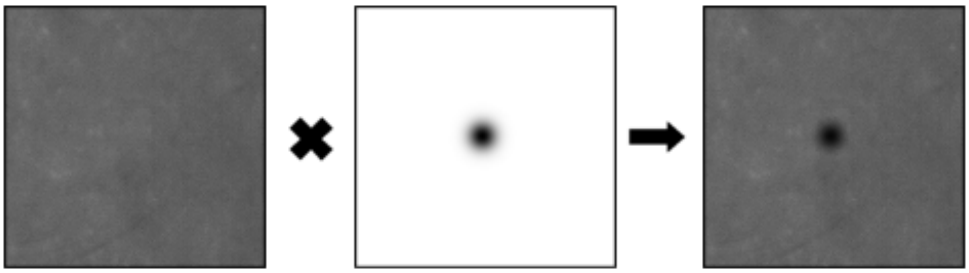


Figure 35 - Applying an artificial dust dot

To create the dusty datasets we multiplied each image in the normal and multi-sampled dataset with the white frame containing the dust dot to generate two new datasets. The dust spot is positioned exactly in the centre of the image.

Flat field corrected datasets

To the four created datasets we also apply flat-field correction as is demonstrated in Section 6.1. This method can also be applied in real time, just before doing the displacement estimation, but we chose to apply this correction already beforehand, because this has a positive influence on the speed of the calculations. This gives us in total 8 datasets of 131 pictures and their exact distances.

7.2 Measuring method

We created a program to measure the accuracy of the different algorithms. It loads two pictures into the memory and performs the displacement estimation. The real displacement is calculated by subtracting the two stored positions for the images. The real displacement is then subtracted from the estimated displacement to calculate the deviation of each estimated displacement. This algorithm is repeated for 250 pre-selected combinations within the datasets. We use the same combinations for measuring each algorithm to ensure that we can compare the results. Then we calculate the 3-σ value of all these deviations to indicate the accuracy of each method.

First we measure the accuracy for all the options as described in 7.3.1 for the pixel level estimation and then we will do the same for the sub pixel level estimation for the options described in 7.3.2.

In the measurements we used margins (as discussed in section 3.3) of 10 pixels in the x-direction and 300 pixels in the y-direction. We chose these margins, because we know that the displacement in x-direction is minimal, since we only moved the paper in y-direction. The maximal displacement in y-direction is 200µm. It would have been sufficient to have a margin of 80 pixels, but we chose for 300 pixels such that the correlation window will be selected close to the centre of the picture. In the centre, the illumination is more uniform and gives better results. These results are more representative for the accuracy of the sensor.

These margins have influence on the amount of data used in the FFT algorithm. As discussed in section 6.4, FFT uses the region of correlation window including the margins for its calculations. The margins used in our tests are 10 pixels in the x-direction and 300 pixels in the y-direction. The margins stretch to both sides equally, so when a correlation window of 32 by 32 pixels is used, the region which is used for the FFT is then (32+2*10) = 52 pixels wide by (32+2*300) = 632 pixels high.

All the measurements are done on a normal PC with enough processing power and enough internal memory. Therefore we do not have the constraints of working with a DSP where the processing power and memory are very limited.

7.3 Parameters to measure

To find out which improvement or combination of improvements works the best, we have to identify over which parameters we want to compare measurements in accuracy. Besides establishing the accuracy of the methods present in the sensor's software, we also want to measure the influence of the possible solution, as discussed in chapter 6.

First we present measurements on the pixel level estimations in section 7.3.1. Next, we describe measurements on subpixel level estimations in section 7.3.2.

7.3.1 Pixel level displacement estimation

As described in section 7.2 we used 250 combinations of two images to determine the 3-σ value. For each combination we are interested on pixel level displacement estimation in the influence of:

- Spatial cross correlation with correlation windows of 16x16, 24x24, 32x32, 48x48, 64x64 pixels
- FFT method of section 6.4, with correlation window sizes that include margins of 36x616, 44x624, 52x632, 68x648, 84x664 pixels.
- Add a dust particle in the centre of both images or not. The dust dot has a diameter of 20 pixels with a soft edge as is shown in Figure 35.
- Apply flat-field correction or not

This gives us 5 x 2 x 2 x 2 = 40 options that need to be tested. We decided not to include the influence of the various noise reducing methods, because experiments have shown that on pixel level these methods only have a marginal influence on the displacement estimation.

7.3.2 Subpixel level displacement estimation

On subpixel level displacement estimation we are interested in the influence of:

- Subpixel level displacement estimation algorithm. This can be polynomial fitting through 3 points, N-Cubed displacement algorithm, original bilinear interpolation algorithm and the corrected bilinear interpolation algorithm.
- Correlation window sizes of 16x16, 24x24, 32x32, 48x48 and 64x64 pixels.
- Apply flat-field correction or not
- Reduce the noise or not. Noise can be reduced either by multi-sampling or by one of the five spatial filters we discussed in section 6.2.2. Spatial filters can have the size of 3x3, 5x5, 7x7 and 9x9 pixels.

This gives us 4 x 5 x 2 x (2 + 5 x 4) = 880 options that need to be tested. We do not test the influence of dust for the subpixel level estimations, because, as will be said in chapter 8, we notice in the results of the pixel level estimations that the dust dot strongly influences the estimations, but that the accuracy results of the flat-field corrected dust dot are about the same as those for the flat-field corrected images without a dust dot. Experiments have shown that this is also the case for subpixel level estimations. Including the option for dust does therefore not give more information.

In the next chapter we will discuss the results of all these measurements.

8 Results of improvements

In this chapter we discuss the results of the measurements described in section 7.3. First, we discuss the results of the measurements done on pixel level accuracy. After that we do this for the subpixel level accuracy. The accuracy results are the 3-σ values of the deviations. This means that the lower the value, the better the accuracy.

8.1 Pixel level accuracy

The results of the pixel level accuracy measurement can be found in Table 4. We discuss in this section first the difference in accuracy between cross correlation and FFT. Then we discuss the influence of flat-field correction and after that the effect of dust on the accuracy. Further, we discuss the influence of the correlation window size.

8.1.1 Cross correlation versus FFT

In Table 4 the accuracy measurement results of the spatial cross correlation algorithm are seen in the left column and the ones from FFT in the right column. It is quickly visible that the cross correlation results are more accurate than the results of FFT. The average of all correlation window sizes when no dust is present and no flat-fielding is applied is 3.76 µm for FFT and only 3.12µm for spatial cross correlation. A reason that FFT is less accurate than spatial correlation could be that FFT uses the same amount of values in both images when calculating the correlation coefficient matrix. This has as a consequence that due to paper displacement parts of the values in one image that are not present in the second image (and vice versa) are still used for calculating the correlation coefficients. When the displacement is big enough this can lead to deviations. Future research is needed to quantify this occurance.

Table 4 - Accuracy results on pixel level

Dust	Flat-field Correction	Correlation window size	Algorithm	
			Cross Correlation	FFT
No	No FF	16x16	2.853	3.642
		24x24	3.059	3.58
		32x32	2.923	3.729
		48x48	3.403	3.851
		64x64	3.389	4.001
	FF	16x16	2.737	3.358
		24x24	2.732	3.452
		32x32	2.745	3.501
		48x48	2.829	3.552
		64x64	2.565	3.616
Yes	No FF	16x16	202.835	202.765
		24x24	202.835	202.733
		32x32	202.835	202.699
		48x48	202.835	202.361
		64x64	202.835	202.164
	FF	16x16	2.571	3.358
		24x24	2.772	3.429
		32x32	2.707	3.499
		48x48	2.781	3.552
		64x64	2.582	3.616

8.1.2 Influence of flat-field correction

When we compare the results of the measurements done without and with flat-field correction, we notice that the accuracy improves for both cross correlation and FFT. The average of the results is 2.72 for cross correlation and 3.49 for FFT. Compared to the values 3.12 and 3.76 this is a considerable improvement. If the illumination pattern would be even less uniform, then we would have seen worse results for the non-flat-field corrected measurements, but roughly the same for the flat-fielded ones.

8.1.3 Effect of Dust

The dust dot we applied to the dataset had a disastrous effect on the displacement estimation. The algorithm was not able to determine the displacement correctly anymore and thus estimated that not any or hardly any displacement had taken place. In fact, only the dust dot stayed firmly in place. The results of the measurements did not form a normal distribution anymore, so actually taking the standard deviation of this set of result, does not make sense, but with an average accuracy of 202.83µm and 202.54µm for respectively cross correlation and FFT, we can conclude that when the presence of dust is severe and totally disrupts the correct working of the sensor.

However when we apply flat-field correction on measurements that were influenced by dust, we get results comparable to flat-field corrected measurements without the dust. Further, it is remarkable that the average of the flat-field corrected dust measurements is slightly more accurate than the flat-field corrected images without dust. Future research is needed to find the reason for this, but it might be an accidental occurrence within the used datasets.

We did not investigate the effect on the accuracy when only one of the two images has a dust particle. This situation typically occurs when two cameras are used and the images of both cameras are compared. It is very unlikely that a dust particle on the lens of one camera is also present on the second camera. Future research is needed to find out what the influence on the accuracy is in this situation.

8.1.4 Correlation window size

In the measurements without dust and without flat-fielding it is noticeable that the accuracy is decreasing as the window size is growing. This is happening for both algorithms. When we compare this to the results of the flat-field corrected measurements, we see that these are not changing significantly. This can be explained that before flat-fielding the illumination is not uniform and thus when the correlation window is bigger, the more the variance of the illumination pattern is in the captured image. In the flat-field corrected illumination pattern the non-uniform illumination is corrected and therefore the size of the correlation window does not influence anymore the accuracy of displacement estimation on pixel level.

8.2 Subpixel level accuracy

In this section we discuss the results of the measurements done on subpixel level accuracy. These can be found in Table 12 in Appendix C. As we noticed in the analysis of the pixel level accuracy results, flat-field correction has a big influence on the accuracy. Therefore we subdivide each of the results between non-flat-field corrected and flat-field corrected measurements. First we will compare the four subpixel level estimation algorithms. Then we discuss the influence of the different filters and their filter size. After that we discuss the influence of the correlation window size.

8.2.1 Comparing the algorithms

We compared four subpixel level displacement estimation algorithms: polynomial fit through 3 points, N-Cubed, the original bilinear interpolation method and the corrected bilinear interpolation method. To analyse these methods we take the minimum, the mean and the maximum value of all the 3-σ values of the measurements we have done, split up between non-flat-field corrected (FF-No) and flat-field corrected (FF-Yes) measurements.

Table 5 and Figure 36 show that the minimum of the 3-point algorithm is comparable with the N-Cubed algorithm, but that the mean and the maximum value are higher for the 3-point algorithm.

Table 5 - Summary of accuracy subpixel level algorithms

		3-point	N-cubed	Bilinear	Corrected Bilinear
FF-No	Minimum	0.579	0.581	1.056	1.079
	Mean	1.316	1.030	1.768	1.729
	Maximum	4.810	2.805	3.421	2.899
FF-Yes	Minimum	0.488	0.463	1.024	0.792
	Mean	1.267	0.971	1.739	1.516
	Maximum	4.871	3.152	3.781	2.381

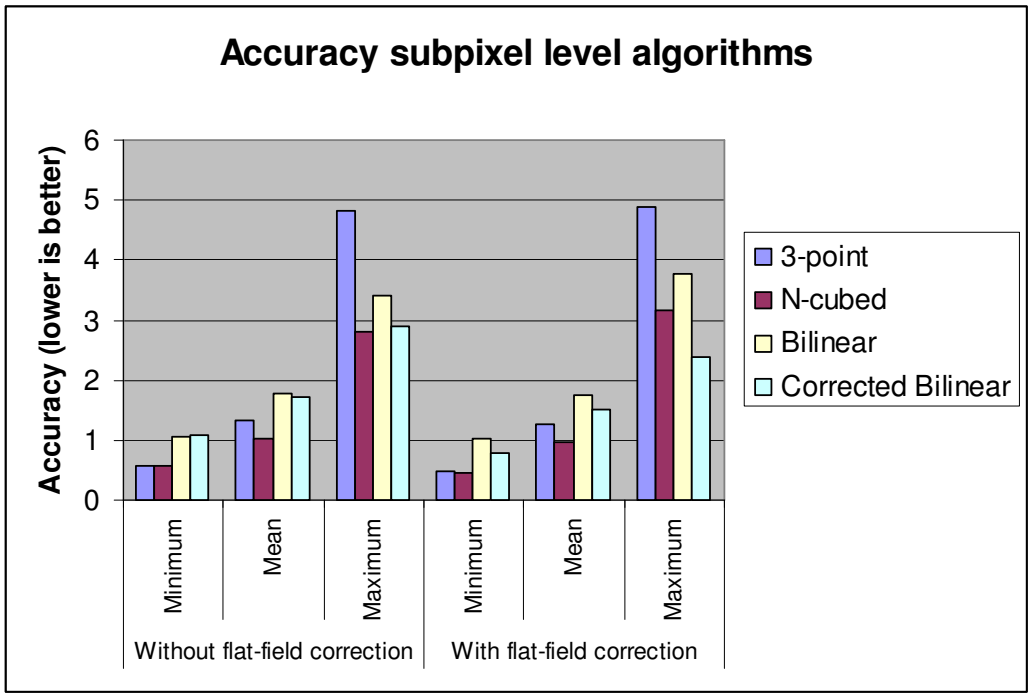


Figure 36 - Accuracy subpixel level algorithms

Observe that here again the flat-field correction does improve the results of all algorithms. Further, we can see that the bilinear interpolation based algorithms perform less than the other two algorithms. We already pointed out in section 5.2 that the original bilinear interpolation algorithm has a structural error, and we indeed observe that this algorithm has a bigger deviation. The corrected bilinear interpolation algorithm is performing only slightly better than the original bilinear interpolation algorithm and it is less accurate than the first two algorithms.

8.2.2 Influence of noise reduction methods

We presented 6 different solutions for reducing noise: five different spatial filters focused on noise reduction and multi-sampling. For each of the filters we tested different sizes. First we narrow down which filter size gives the best result. Then we will compare the results of the filters of that size with the multi-sampling and the non-noise reduced results.

We can easily show which filter size is the best choice by calculating the mean value of all results grouped by filter size. Again we make the subdivision between non-flat-field corrected and flat-field corrected measurements. The result is shown in Table 6.

Table 6 - Mean values of filters grouped by filter size

	Filter size	3-point	N-Cubed	Bilinear	Corrected Bilinear
FF-No	3x3	0.75784	0.75848	1.20528	1.65136
	5x5	1.0622	0.90348	1.63064	1.685
	7x7	1.476	1.09428	1.98384	1.718
	9x9	1.94776	1.3096	2.30252	1.7578
FF-Yes	3x3	0.7224	0.72512	1.16048	1.47596
	5x5	1.03568	0.84028	1.6026	1.49568
	7x7	1.45504	1.01484	2.00536	1.52584
	9x9	1.98272	1.2186	2.30312	1.5562

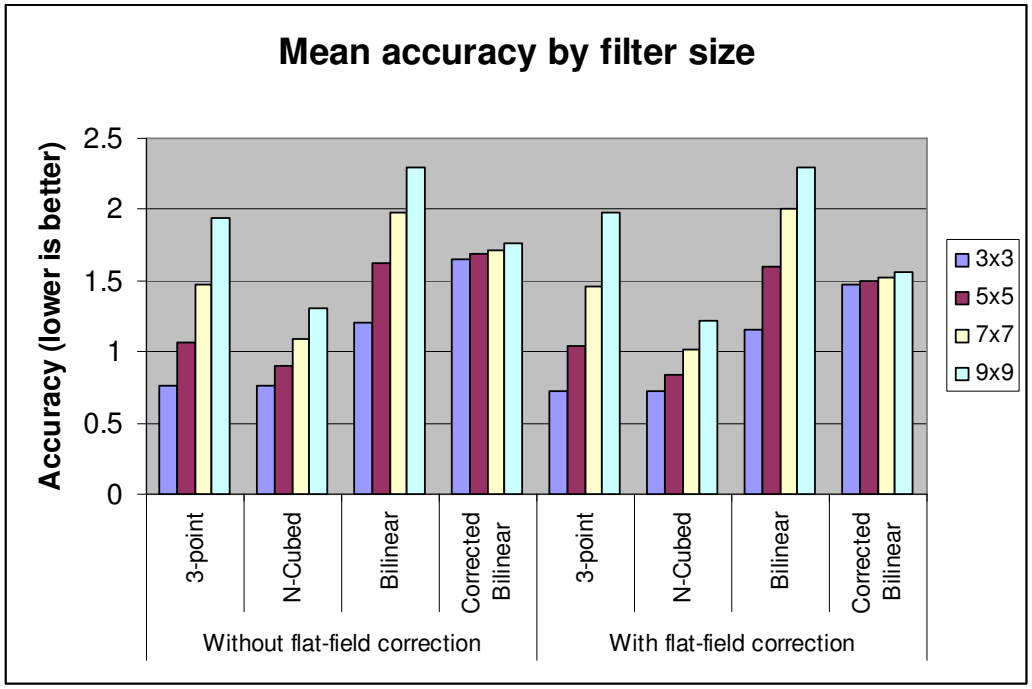


Figure 37 - Mean accuracy by spatial filter size

From Table 6 and Figure 37 we can clearly see that there is almost a direct linear relation between filter size and accuracy. For both flat-field corrected and non-flat-field corrected we can see that the best accuracy is achieved with the smallest filter. This indicates that the present noise has a high frequency, which we filter out with a relatively small (3x3) spatial filter. Bigger filters reduce also the presence of lower frequency signals which are part of the image surface. These filter throw away important information has therefore a bad influence on the accuracy.

We determined that the 3x3 filter performs the best on average for the different spatial filters. We compare the different filters only for this filter size. As mentioned in section 7.1 we tested 6 methods for reducing noise and compared these with the results when no attempt is done for removing the noise. In
Table 7 and Figure 38 the mean values of the noise reducing methods are shown.

Table 7 - Mean values grouped by noise reducing method. Only the data of the filter size of 3x3 is used.

	Filter	3-point	N-cubed	Bilinear	Corrected Bilinear
FF-No	No	1.4784	1.4772	1.8106	2.0556
	Multi-sampling	1.258	0.8512	1.4756	1.9304
	Arithmetic	0.7414	0.7422	1.1352	1.6422
	Geometric	0.75	0.7454	1.1484	1.6414
	Harmonic	0.7542	0.7466	1.1608	1.643
	Median	0.796	0.8282	1.2608	1.6782
	Adaptive	0.7476	0.73	1.3212	1.652
FF-Yes	No	1.1674	1.641	1.7546	1.696
	Multi-sampling	0.9024	0.8298	1.2844	1.545
	Arithmetic	0.7104	0.7084	1.1378	1.463
	Geometric	0.7268	0.709	1.1608	1.4724
	Harmonic	0.7328	0.71	1.1504	1.4726
	Median	0.762	0.7934	1.159	1.4994
	Adaptive	0.68	0.7048	1.1944	1.4724

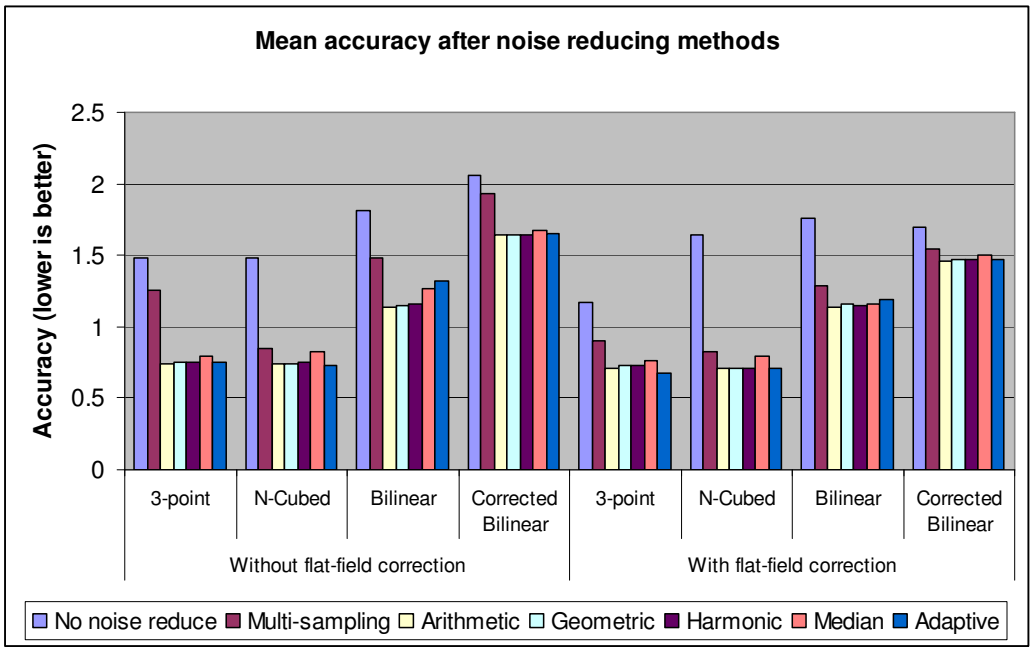


Figure 38 - Mean accuracy after noise reducing methods

We can conclude that the usage of filters drastically improves the accuracy, compared to the options when no noise is reduced and the multi-sampling method. Between the filters there is not that much difference as they perform all about equal. The median filter is performing the worst of the filters, while the adaptive filter comes as best out of the test.

An interesting fact that can be noticed here is that the effect of flat-fielding is much less visible when using the filters. From this we can assume that apparently the filters also compensate the non-uniform illumination pattern partly. Future research is needed to prove this hypothesis.

8.2.3 Correlation window size

Also on subpixel level estimations the size of the correlation window has a high influence on the accuracy of all the methods. As shown in section 8.1.4, the accuracy on pixel level is getting worse as the size of correlation window increases. After flat-field correction, the window size is less relevant for the accuracy. In Table 12 in appendix C with the measurements on subpixel level algorithms we observe the same trend for the multi-sampling method, except that when flat-field correction is applied, the accuracy is improving as the correlation window is growing.

When we apply any of the filters to reduce the noise, we get better results as the correlation window is growing, independent from flat-field correction. In section 8.2.2 we already noticed that the spatial filters sometimes behave as flat-field correction, and when we only consider their influence on accuracy, the effect of the correlation window size is the same.

8.3 Conclusion

We mention the five most important conclusions from the measurements. First, we can conclude that FFT does give less accurate results than spatial cross correlation for pixel level estimation. But these results are not that inaccurate that they become useless, because FFT estimates the displacement sometimes only 1 pixel next to the correct estimation. With FFT, the speed increase is significant when the correlation window and margins are large enough. Since this is often the case, a combination with the spatial cross correlation could be useful. First estimate on pixel level roughly the displacement with FFT. Then calculate with spatial cross correlation the correlation coefficients around the previous found displacement. Then, the maximum found correlation coefficient gives a better estimation of the displacement. Since only a few correlation coefficients need to be calculated the total amount of computations is very low. In this way the best of both methods are combined: the speed of FFT and the accuracy of spatial cross correlation.

Second, we can conclude that, in general, flat-field correction has a positive influence on the accuracy and it is capable of removing dust particles. In our datasets we had only a relatively small distance (max 80 pixels) over which we displaced the paper. Also we used only the region around the centre for our displacement estimations, because in this region the illumination pattern is more uniform than on the edges. If the displacements would be bigger and therefore a bigger area of the images is needed, the illumination pattern is less uniform and then flat-field correction will give even more advantage.

Third, we can say that the usage of spatial filters has a very positive impact on the accuracy in our tests. Little was known about the effect of spatial filters on displacement estimations. From our tests, the 3x3 adaptive filter turned out to be the best performing spatial filter for reducing noise and by this improving the accuracy of the displacement estimation.

Fourth, the N-cubed algorithm is performing the best in our tests. The polynomial fit through 3-points algorithm is at some measurements performing slightly better than the N-cubed algorithm, but in most measurements, it was a good second. The bilinear interpolation based algorithms did not perform well. We already pointed that out for the original algorithm, but we did not expect this for the corrected version. Future research is needed to clarify this.

Fifth, we can conclude that when applying image enhancement techniques, like flat-fielding and spatial filters, a bigger correlation window gives more accurate results when using the subpixel level displacement estimations. However, when these image enhancement techniques are not applied, we get opposite results.

When we combine the options flat-field correction, 3x3 adaptive filter and the N-cubed subpixel displacement algorithm we get the highest accuracy of 0.463 as 3σ-value. The algorithms which were present in the sensor's software have without all improvements a best accuracy of 1.336 for the polynomial fit through 3-points algorithm and 1.782 for the bilinear interpolation algorithm. From this we can conclude that our research improved the accuracy of the optical paper position sensor with $\frac{1.336 - 0.463}{1.336} x100\% = 65\%$.

An important note is that these results are achieved from these datasets and under specific conditions, e.g. where the displacement of paper was relatively small and where only a very small part of the images was used to estimate these displacements. Real life usage of the sensor always will give unpredicted situations, thus the accuracy probably becomes less in those situations.

9 Construction options

This chapter is about possible improvements of the price/quality ratio of the sensor. First we give an overview of the different parts within the sensor including indexed cost price. Second, we identify which parts of the sensor have currently the most influence on the accuracy. Further we did a field search for components of the sensor and conclude how much influence each component has on the whole. Also we give a short overview of possible added functionality which increases the value of the sensor.

9.1 Components and costs

An optical sensor is typically constructed from the following components:

- Casing
- A digital signal processor
- One or more identical camera blocks
- Electronics

Casing

The casing is the aluminium or plastic component in which all the other components are placed. It serves for both protection and fixation of the different parts.

Digital Signal Processor

The digital signal processor is taking care of the paper displacement calculations. It has a certain speed and certain memory size which limit the possibilities for paper displacement algorithms.

Camera block

Each camera block consists of at least the following necessary components:

- Lens or lens system
- CMOS or CCD

Lens or lens system

A lens or combination of lenses is needed to project the paper surface onto the light sensitive chip. Specifying the different properties of lenses is very complex and out of the scope of this research.

CMOS or CCD

The CMOS or CCD is a light sensitive chip which converts the analogue projected image into a digital image. This light sensitive chip is controlled via electronics.

Electronics

Various small electronic parts are needed to connect all the components together, and communicate with the printer.

We indexed the costs of the sensor containing one camera block as 100. For the various parts the indexed costs can be seen in Table 8.

Table 8 - Indexed costs per component

Part	Amount	Costs	Total Costs
Casing	1	20	20
DSP	1	28	28
Electronics	1	17	17
Subtotal			65
Lens	1	28	28
CMOS/CCD	1	7	7
Subtotal			35
Total indexed costs for 1 sensor			100

Using the same type of components the costs of a sensor with two camera blocks becomes 135 and with three blocks 170. This is a simplified model of the real situation, because the casing also needs to become bigger and a little bit more electronics is needed giving around 1 or 2 extra costs. These extra costs are very small and therefore are neglected in this model.

9.2 Limitations in accuracy in current construction design

One of the goals of the research was to improve the accuracy via software improvement. We gained a significant improvement, but we noticed that a higher accuracy is prevented due to several limitations. We can conclude from our research that the accuracy is limited by the following factors:

- Noise level in the captured images
- Size of the area encompassed by one pixel in the captured image
- Number of cameras
- Algorithm

For each of these options we discuss which component of the sensor is involved and how much space there is for improving the cost price.

Noise

As described earlier, noise has a big influence on the accuracy of the paper displacement calculations. Noise is a property that is present on all electric devices and in our case it is the noise present on the CMOS for capturing the images. We assume that a better quality CMOS has better ways to suppress noise in the captured images, which would result in better accuracy of the paper displacement calculations. On the other hand, a higher quality CMOS would also be more expensive. Since it is very difficult to find the noise levels of different CMOS, we have not investigated how this influences the price/quality ratio. Reducing noise is probably also possible by shortening the exposure time of the CMOS. A result of this method is that the captured images get darker and this must be compensated with stronger light source to illuminate the object.

Pixel size

We demonstrated before that the accuracy of the optical paper position sensor is closely related to the size of the area encompassed by one pixel in the captured image. This image

pixel size is dependent on magnification factor of the lens and the pixel size on the CMOS. Suppose in our sensor the magnification factor of the lens is *m* times and the size per pixel *x* µm. Then each pixel of the captured image represents *x/m* by *x/m* µm on the object size. Hence, in the current sensor *x/m = 2.5*, i.e., *m = x/2.5*. If the sensor would be using a lens with a bigger magnification factor, e.g., *m = x*, then each pixel in the captured image would be smaller, e.g., 1 by 1µm. In this example, the accuracy would be then improved by a factor *x/m* compared to the current construction design. However the size of the total captured image would also be reduced, since the amount of pixels stays the same. Of course, reducing the pixel size on the CMOS by a factor *x/m* will lead to the same accuracy improvement. Note, however, that reducing the pixel size cannot be done without checking if the lens is capable of providing that resolution, because then a higher quality lens is needed. Pixel sizes < 3 µm are not properly addressed by "off the shelf" lenses [7].

Number of cameras

The paper within a printer is not moving continuously and is at hold when a row of dots is printed. When two or more cameras are connected together in one sensor, they can be positioned at such a distance that the paper surface is captured only at the times when the paper is not moving. During experiments we noticed that the accuracy is decreasing when capturing images while the paper is moving, thus it is an advantage when the paper is on hold while capturing images. Another advantage of capturing images when the paper is not moving is that it is then possible to capture more than one image of the paper surface. Averaging the captured images improves the quality by suppressing the noise. A disadvantage is that it has to be defined in advance on which distances the cameras must be placed and then only at these distances the sensor will be able to measure exactly. For measuring other distances another sensor with different construction will be needed. An option could be to develop a sensor in which the second camera can be positioned at more distances from the first camera. This gives the advantage that the sensor can be tuned to work at any predefined distance. However we do not think it gives such an advantage, because printers usually have a predefined paper step size, which is not subject to radical changes. Moreover, this design can only address one printing mode at the same time. Another reason is that sensor needs to be specifically designed for each different type of printer, since the space constraints and influences on other parts differ for each different type.

As discussed in section 3.4, using only one camera has the disadvantage that many measurements have to be made per millimetre paper transport. Each measurement has a certain error and when making more measurements after each other the accuracy of the total paper displacement estimation is decreasing, because the errors are added up. Using two cameras in the sensor reduces the amount of measurements that need to be taken, since these cameras can cooperate together. First, one camera captures the paper surface. The paper is transported further and then the second camera captures the same piece of paper when it passes by. Since the distance between the cameras is known, we are able to calculate with one measurement a big paper displacement very accurate.

Summarized, the usage of more than one camera gives possibilities to improve the accuracy, but this gives more costs. In the next section we will discuss which effect it has on the price/quality ratio.

Algorithm

In embedded hardware, constraints of speed and memory are always present. Therefore the algorithm needs to be designed in such a way that the hardware is capable of running it. We demonstrated in chapter 8 that usage of a bigger correlation window resulted often in a better result. However, using a bigger correlation window requires more memory and it takes more processing time for doing the paper displacement calculation. The current digital signal

processor is limited in the amount of computations per second, thus if the computations need to be finished within a certain amount of time, then a small enough correlation window needs to be chosen. As discussed in section 6.4 Fast Fourier Transforms can provide a faster alternative for estimating paper displacement, but it provides a slightly less accuracy.

9.3 Influence on the price/quality ratio

In previous paragraphs we pointed out that the lens, the CMOS, and the DSP are the components which have the biggest influence on the accuracy. The costs of these components typically follow the normal Price Quality relationship which can be seen in Figure 39. Besides these components also the amount of cameras has influence on the accuracy.

Figure 39 - Price quality relationship

For the CMOS/CCDs and DSP it is very difficult to construct a real price/quality curve. As can be seen for the DSPs on the Texas Instruments website [12] the prices of the different DSPs are not so much influenced by processor speed or memory size, but rather under influence of the additional functionality.

For the CMOS/CCDs it is difficult to find the right prices in catalogues on the internet, because the prices are dependent on the amount of components that are ordered and they are quickly outdated. Usually, only one or two batches are made when the manufacturers switch to producing a newer version of the chip. There is also a high-end market for e.g. military and satellite purposes. These chips are available for a longer time, but are relatively very expensive. Fairchild Imaging [3] is a company that addresses this market. The lack of prices for the low budget market makes it rather difficult to establish a price quality. However from the information in the catalogue on Digikey.com [2] we can conclude that the prices for CMOS/CCDs hardly decrease for pixel sizes bigger than we have in our current sensor. For CMOS/CCDs with smaller pixel sizes no price data could be found, but since we know that the lenses also need increasing quality, the total price of using CMOS/CCDs with a smaller pixel size will have increased costs.

There are roughly two types of lenses which can be used within the construction of a sensor, namely off-the-shelf lenses and custom designed lenses. Off-the-shelf lenses are relatively cheap and are widely available; while custom designed lenses are very expensive and need about 6 to 18 months development time [7]. Since there are many different kinds of lenses available on the market it is very important to define on beforehand which exact specifications the lens must have. Then it is possible to search for an off-the-shelf solution. The specifications for the lens depend very much on the other factors within the construction of the sensor, such as available space within the casing, the needed resolution and aperture for correctly projecting the paper surface on the CMOS/CCD. Such an extensive research falls out of the scope of this research and needs to be investigated as a separate project.

Despite that we can hardly find exact price information we can still estimate the following price/quality curve, which is shown in Figure 40.

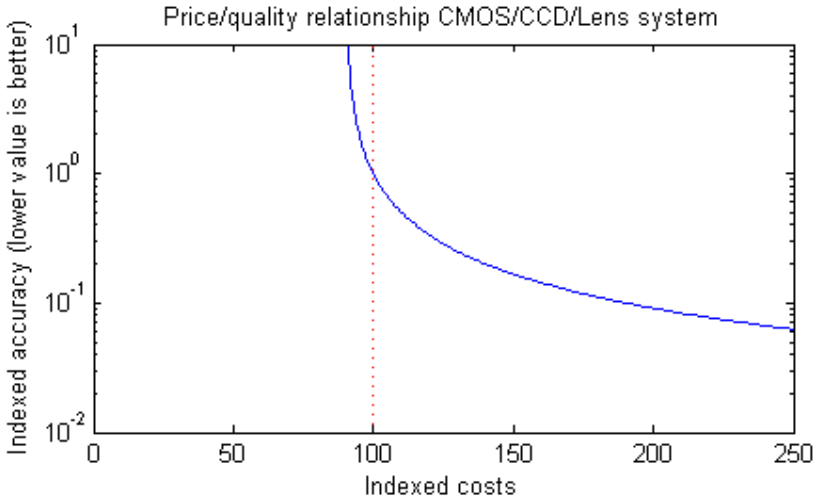


Figure 40 - Price/quality relationship CMOS/CCD/Lens system

As explained in previous paragraph, the amount of cameras has an influence on the accuracy of the sensor. The extra costs for each sensor are 35 but for each pair of cameras the paper displacement can be measured very accurately when the paper stops moving after a paper displacement of this distance. Within each printer there are one or more modes which influence the distance between printed rows. For each mode for which accurate paper displacement measurement is needed, two cameras need to be coupled. A printer cannot print in two modes at the same time, thus the same camera can be used in combination with more cameras to support the different modes. Future research is needed to measure the difference in accuracy for capturing moving paper and non-moving, but logical reasoning provides us enough understanding that there is an advantage when using more than one camera. Besides this there is the option of averaging out the noise by using the multi-sampling technique. This is possible because the paper is not moving then. As we pointed out in chapter 8 reducing noise can give a higher accuracy. We can conclude that using more cameras is a considerable improvement to the accuracy.

In Figure 41 the price/quality relationship of number of cameras can be seen. This is not a smooth curve and a very simplistic model. Measuring longer distances is best done with two or more cameras, but adding more cameras does not increase the accuracy further. Only more distances can be measured accurately. All the distances that are not equal to the distances between the cameras can be measured only with a lower accuracy. Still it is important to point out that for 35% of extra costs a higher accuracy can be achieved if a longer distance needs to be measured.

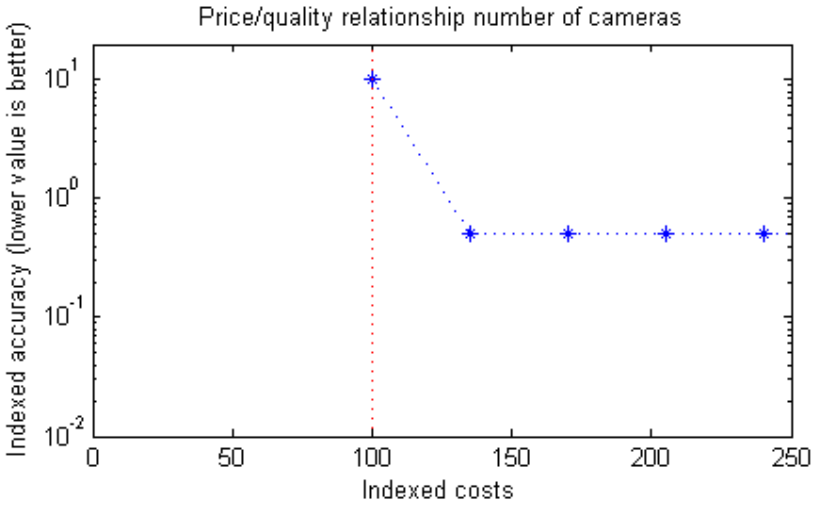


Figure 41 - Price/quality relationship number of cameras

9.4 Other functionality for the sensor

The sensor is developed for measuring the paper displacement within a printer. Instead of trying to make the sensor cheaper it is also possible to increase the value of it by thinking of additional functionalities which can be implemented in the future and thus improving the total costs of ownership of the printer.

We mention three examples of added functionality that might be added to the sensor to make it more valuable:

- Avoiding paper touch by print head

Currently sometimes the paper curls up underneath the print head in printers. The print head is the part of the printer that shoots out the ink onto the paper. This is a very complex and expensive component. When paper starts to curl up, the print head touches the paper and will be damaged. If this happens too many times the print head must be replaced. By detecting in time that the paper curls up, it is possible to prevent that this will happen. When the paper curls up it will get out the focus area of the sensor and we think it must be possible to detect this phenomenon automatically.

- Media detection

Increasingly, customers have higher demands and want to print on more different media than ever before. Each medium requires different settings of the printer and therefore it would be very user friendly when the sensor would be able to detect the type of medium by just ‘looking’ at it and put the right settings within the printer automatically. We consider this possible because different media types have different type of paper structure and thickness of paper could be measured by different amount of transparency when a bright light source is illuminating the paper surface from the other side.

- Detection of the paper front edge

To make sure that the print head only prints on paper, the paper front edge is now detected by a separate sensor. This sensor can be replaced when the optical paper position sensor takes over this task.

9.5 Conclusion

We have seen that it is not easy to determine how design changes affect the cost price of the sensor. We can conclude that even if we would manage to increase accuracy of the sensor significantly, replacing components of the sensor by less precise ones does not have a big impact on the cost price of the sensor. Improving the quality of the components does have an impact on the cost price. These extra costs are caused mostly by the lens or lens system. We also pointed out that extra cameras within the construction design can improve the accuracy considerably for approximately 35% of extra costs. Further we showed that added functionality can increase the value of the sensor, which can make it more attractive for usage in a printer.

10 Conclusion

In this research project we analysed and improved the accuracy of the optical paper position sensor developed by Océ. We also gave a quick analysis of the influence of the different components on the price/quality relationship. In this chapter we discuss how much of the research objectives are met. In section 2.3 we stated the following hypothesis:

With 'smarter' software a cheaper version of the optical sensor can still provide the accuracy desired by Océ.

We validate this hypothesis by answering the research questions of this project. We had three main research questions.

- What is the current status of the optical paper position sensor within Océ?
- What can we improve?
- What is the influence on the total cost of ownership?

10.1 What is the current status?

Before being able to determine how accurate the current sensor is and what the results of the suggested improvements are, we had to develop a method to measure the accuracy. In chapter 7 we described how we achieved this. We created datasets of images, which represent the real life working of the sensor. We also developed a program to easily measure the accuracy by estimating the paper displacements in the dataset and compare them to the real life displacement. In this way we measured that the current accuracy of the sensor is defined as a 3-σ value of 1.336µm deviation when using the 16x16 pixel correlation window and the polynomial fit through 3-points algorithm. An important observation is that we limited the displacement within the dataset to 200µm and we only used the region in the images where the most uniform illumination was present. Measuring bigger paper displacements and/or using other regions as well for calculations will most likely give a worse accuracy.

10.2 What can we improve?

We identified several causes which influence the accuracy negatively. These are discussed in chapter 4. We did not implement and test solutions for all the causes due to time constraints. We focused on techniques for reducing the influence of illumination pattern and sensor's fingerprint, noise reduction and dust robustness. Flat-field correction provided not only the solution for eliminating the influence of the illumination pattern and the sensor's finger print, but it also proved to be a good solution for increasing dust robustness. To reduce the influence of noise we implemented and tested several methods. All of them improved the accuracy, but with different success. From the test results we can conclude that the adaptive spatial filter performs the best, and we suggest implementing this filter in the embedded software to increase the accuracy significantly. Further we analysed the algorithms present in the sensor's software and suggested improvements. We noticed that the bilinear interpolation algorithm has a serious error, which prevents very accurate displacement estimations. The corrected bilinear interpolation algorithm didn't have this flaw, but it proved to be not that accurate either. The best results were gained with the N-Cubed algorithm developed by HP laboratories, but these were only marginally better than the polynomial fit through 3-points algorithm, which was already implemented in the software. Further we showed in section 6.4 that Fast Fourier Transforms (FFT) do have a positive impact on the speed of the algorithm when the correlation window and the margins are big enough. In our situation it gave less accurate results, but we believe that in combination with the spatial cross correlation method,

as described in chapter 3, we can profit from the speed increase while still maintaining enough accuracy.

The accuracy before the improvements was indicated with a 3-σ value of 1.336µm. After testing all the possible combinations of improvements the results showed that the best results are achieved when flat-field correction, the 3x3 adaptive filter, and the N-cubed algorithm is used. Then we obtain a 3-σ value of 0.463µm. This is an increase of accuracy of 65%, which is a considerable improvement. It is important to keep in mind that the tests were done in stable laboratory conditions. We expect that in real life usage of the sensor a worse accuracy is achieved, but nevertheless this improvement will also be noticeable there.

10.3 What is the influence on the total cost of ownership

Due to the fact that very limited up-to-date information was available about the actual prices of possible substitutes of the different parts of the sensor, we were not able to give an exact indication on the influence on the total cost of ownership. However, we did notice from the small amount of information we got, that less accurate components are not that much cheaper than the currently used ones, though more accurate components are much more expensive. We also discussed the option of having more than one camera in the sensor and let them cooperate together. We concluded that this certainly can improve the accuracy when measuring bigger paper displacements. Further we discussed that the total cost of ownership can also be lowered by adding more functionality to the sensor. We shortly discussed three options: avoiding paper touch by the print head, media detection and detection of paper front edge. Future research is needed to determine if it is possible to implement these options and if these are worth the extra costs.

10.4 Conclusion and future work

Although we were not able to give a better insight in the implications for the total cost of ownership, we did achieve a 65% increase in accuracy by using ‘smarter’ software. Therefore, we can conclude that we validated the hypothesis partly, due to lack of information about cheaper hardware.

We showed in our research that with better software, we can achieve higher accuracy in paper displacement estimation, but we only focussed on a limited amount of causes. We believe that a higher accuracy can be achieved if also the other causes are addressed, such as optical distortions. We also noticed that the presence of noise had a big impact on the accuracy. To improve the accuracy further it is important that the noise level is reduced by hardware improvements. We showed that much can be gained with software, but we believe that the best results can be achieved with hardware improvements.

11 References

[1] Buades, A., Coll, B., & Morel, J.M. (2004). On image denoising methods, Technical Report 2004-15, *CMLA 2004-15*

[2] Digi-Key Corporation. (2008). Retrieved June 20, 2008, from: http://www.digikey.com

[3] Fairchild Imaging. (2008). Retrieved June 20, 2008, from: http://www.fairchildimaging.de

[4] Gao, J., Picciotto, C., & Jackson, W. (2005). Displacement sensing and estimation theory and applications. *Applied physics A: Materials Science & Processing, 80*, 1265-1278.

[5] Gonzales, R.C. & Woods, R.E. (2008). *Digital image processing (third edition),* New Jersey: Pearson Prentice Hall.

[6] *HP Designjet Z6100-series printers: optical media advance sensor*. Retrieved January 20, 2008, from Hewlett-Packard Development Company Web site: http://h30267.www3.hp.com/Data/en/Z6100_optical.pdf

[7] Kellett P. (2007, November). *MV sensors, cameras and optics: inter-relationships in product development*. Machine Vision Online. Retrieved on June 20, 2008, from http://www.machinevisiononline.org/public/articles/details.cfm?id=3397

[8] Lewis, J.P. (2008). *Fast normalized cross-correlatio*n. Industrial Light & Magic. Retrieved June 20, 2008, from http://www.idiom.com/~zilla/Work/nvisionInterface/nip.html

[9] Roper Scientific GmbH. (2008). *Flat Field Correction*. Retrieved June 20, 2008, from: http://roperscientific.de/tflatfield.html

[10] *Specifications Océ TCS500 print system*. (2006, January). Retrieved January 20, 2008, from Océ Technologies Web site: http://dl.oce.com/downloads/En/Pdf/products/tcs500printspec.pdf

[11] Stroebel, L. (2000). *Basic photographic materials and processes, 2nd ed*. Focal Press.

[12] Texas Instruments website. (2008). Retrieved June 20, 2008, from: http://www.ti.com

[13] The MathWorks. (2008). Retrieved June 20, 2008, from: http://www.mathworks.com/

[14] Van Belle, W. (2007), An adaptive filter for the correct localization of subimages: FFT based subimage localization requires image normalization to work properly. Retrieved June 20, 2008 from: http://werner.yellowcouch.org/Papers/subimg/index.html

[15] Wang, J., Shi, F. Zhang, J., & Liu, Y. (2007), A new calibration model of camera lens distortion. *Pattern Recognition 41*, 607 - 615.

[16] Weisstein, E.W. (2008). *Accuracy*. MathWorld -- A Wolfram Web Resource. Retrieved June 20, 2008 from: http://mathworld.wolfram.com/Accuracy.html

[17] Weisstein, E.W. (2008) *Convolution Theorem*. MathWorld -- A Wolfram Web Resource. Retrieved June 20, 2008 from: http://mathworld.wolfram.com/ConvolutionTheorem.html

[18] Zhou, C., Lin, S. (2007). *Removal of image artifacts due to sensor dust*, IEEE Conf on Computer Vision and Pattern Recognition (CVPR '07), 1-8

A. Mathematical background

In this appendix a couple of the used mathematical functions are explained for better understanding.

A.1 Cross Correlation

A common practise in digital signal processing is to use cross-correlation to calculate a shift between two signals of the same shape.

Assume we have two digitized signals, signal A and signal B, as shown in Table 9. To visualize the signals better we plotted the values as a graph in Figure 42.

Table 9 - Sampled values of two signals

signal A	0	0	0	2	4	8	2	4	2	0	0	0	0	0	0	0
signal B	0	0	0	0	0	0	2	4	8	2	4	2	0	0	0	0

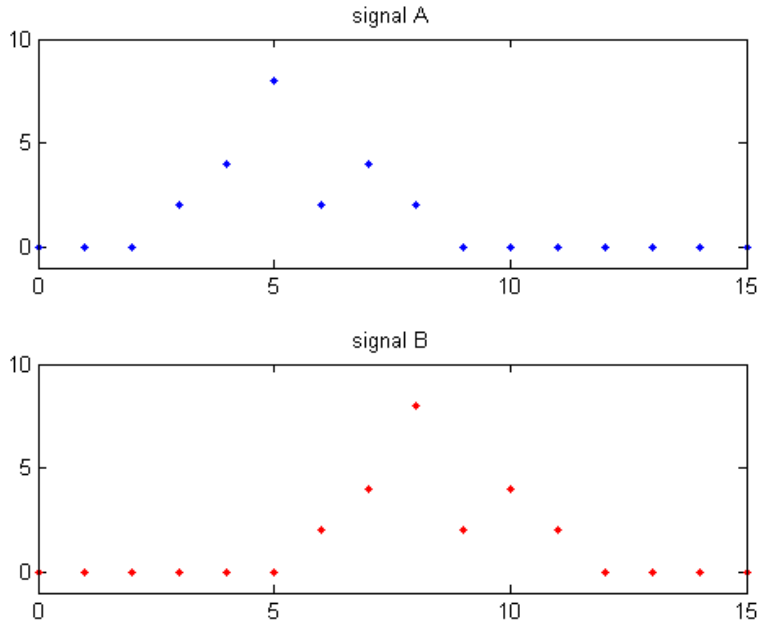


Figure 42 - Plotted graphs of two signals

Now we calculate how much the signal A has shifted compared to signal B. We do this by shifting signal A in steps and at each shift we calculate the correlation coefficient. The correlation coefficient can be calculated with the following formula:

$$Correlation\_coefficient = \frac{\text{Covariance}(SignalA, SignalB)}{\sqrt{(\text{Variance}(SignalA) \times \text{Variance}(SignalB))}}$$

Let δ be the shift between two signals, i.e., δ is 0 when there is no shift, it is 1 when all samples of signal A are shifted one step to the right, etc. The relation between δ and the correlation coefficient is shown in Table 10. This is visualized in Figure 43.

Table 10 - Relation between δ and the correlation coefficient

δ	0	1	2	3	4	5	6	7
Correlation coefficient	0.074	0.383	0.537	1	0.537	0.383	0.074	-0.183
δ	8	9	10	11	12	13	14	15
Correlation coefficient	-0.338	-0.389	-0.389	-0.389	-0.389	-0.389	-0.338	-0.183

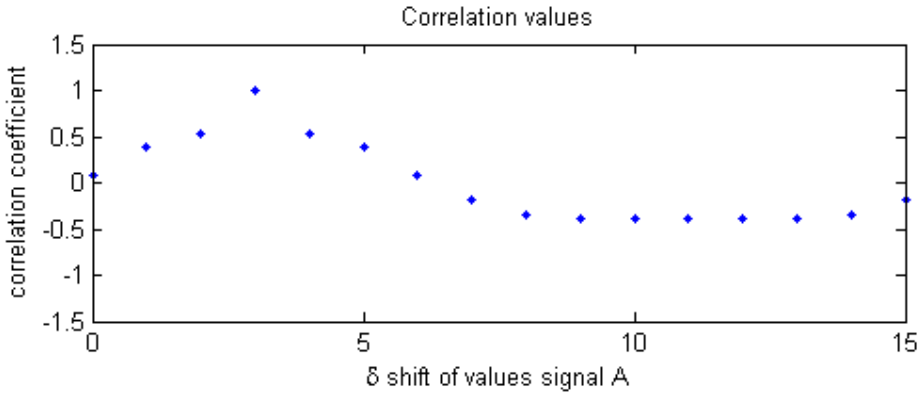


Figure 43 - Correlation coefficients of signal A shifted over signal B

We can clearly see that the correlation coefficient is maximal when δ has a value of three and we can conclude that signal B is shifted 3 sample distances compared to signal A.

This technique can also be applied to higher dimensional signals like images to calculate the shift between two images.

A.2 Finding optimum of 2nd order polynomial

We derive the validity of the algorithm used in section 3.2.3 for determining the optimum of a second order polynomial. By symmetry, we consider the x-direction only:

$$x\_optimum = x\_offset + \frac{\frac{1}{2}\times\left(cor[(x+1,y)] - cor[(x-1,y)]\right)}{2\times cor[(x,y)] - cor[(x+1,y)] - cor[(x-1,y)]}$$

where cor[(x,y)] gives the correlation value in position (x,y)

We will proof the validity of this algorithm by taking 3 points (x_1, y_1), (x_2, y_2) and (x_3, y_3). Between these points we can draw a second order polynomial. The values y_1, y_2 and y_3 can have any value, but the distance between x_1-x_2 and the distance between x_2-x_3 are equal and $x_1 \neq x_2 \neq x_3$. This is a valid assumption as *cor[(x-1,y)], cor[(x,y)]* and *cor[(x+1,y)]* are the equidistant sample points in the correlation coefficient matrix.

Through these 3 points we can draw a second order polynomial of the form:
$f(x) = ax^2 + bx + c$ with the derivative $f(x)' = 2ax + b$. We can find the x-value of the maximum of function *f(x)* by equating the derivative *f'(x)* to zero.

$$2ax + b = 0$$
$$2ax = -b$$
$$x = \frac{-b}{2a}$$

With Lagrange polynomial we can construct the function *f(x)* from the 3 points we have.

$$f(x) = y_1 \cdot \frac{x - x_2}{x_1 - x_2} \cdot \frac{x - x_3}{x_1 - x_3} + y_2 \cdot \frac{x - x_1}{x_2 - x_1} \cdot \frac{x - x_3}{x_2 - x_3} + y_3 \cdot \frac{x - x_1}{x_3 - x_1} \cdot \frac{x - x_2}{x_3 - x_2}$$

Since we know that the distances between x_1-x_2 and x_2-x_3 are equal we can assign values to the x-coordinates and we are allowed to use $x_1 = -1$, $x_2 = 0$ and $x_3 = 1$. The formula *f(x)* will then become:

$$f(x) = y_1 \cdot \frac{x - 0}{-1 - 0} \cdot \frac{x - 1}{-1 - 1} + y_2 \cdot \frac{x - -1}{0 - -1} \cdot \frac{x - 1}{0 - 1} + y_3 \cdot \frac{x - -1}{1 - -1} \cdot \frac{x - 0}{1 - 0}$$
$$f(x) = y_1 \cdot \frac{x}{-1} \cdot \frac{x - 1}{-2} + y_2 \cdot \frac{x + 1}{1} \cdot \frac{x - 1}{-1} + y_3 \cdot \frac{x + 1}{2} \cdot \frac{x}{1}$$
$$f(x) = y_1 \cdot \frac{x^2 - x}{2} + y_2 \cdot \frac{x^2 - 1}{-1} + y_3 \cdot \frac{x^2 + x}{2}$$
$$f(x) = \left(\tfrac{1}{2} y_1 - y_2 + \tfrac{1}{2} y_3\right)x^2 + \left(\tfrac{1}{2} y_3 - \tfrac{1}{2} y_1\right)x + y_2$$
$$a = \tfrac{1}{2} y_1 - y_2 + \tfrac{1}{2} y_3$$
$$b = \tfrac{1}{2} y_3 - \tfrac{1}{2} y_1$$
$$c = y_2$$

Now we can derive the formula for determining the maximum by filling in the values for a, b and c.

$$x\_optimum = \frac{-b}{2a}$$
$$x\_optimum = \frac{-\left(\tfrac{1}{2} y_3 - \tfrac{1}{2} y_1\right)}{2\left(\tfrac{1}{2} y_1 - y_2 + \tfrac{1}{2} y_3\right)}$$
$$x\_optimum = \frac{-\left(\tfrac{1}{2} y_3 - \tfrac{1}{2} y_1\right)}{\left(-2y_2 + y_3 + y_1\right)}$$
$$x\_optimum = \frac{\tfrac{1}{2} y_3 - \tfrac{1}{2} y_1}{2y_2 - y_3 - y_1}$$

This formula gives the x-value of the maximum of the second order polynomial relative to x_2, which was chosen as 0. We can use this formula for calculating the x value of the optimum of the polynomial between three correlation coefficients by replacing y_1, y_2 and y_3 with respectively *cor[(x-1,y)]*, *cor[(x,y)]* and *cor[(x+1,y)]*.

$$x\_optimum = \frac{\frac{1}{2} \times (cor[(x+1, y)] - cor[(x-1, y)])}{2 \times cor[(x, y)] - cor[(x+1, y)] - cor[(x-1, y)]}$$

Now the formula gives the x-value of the maximum of the polynomial relative to *cor[(x,y)]*, which is the maximum value in the correlation coefficient matrix. To calculate the correct displacement estimation the estimated displacement on pixel level needs to be added. This value is indicated as *x_offset* and when we add it to the formula we get the same formula as in the beginning of this section. Thus we have proved that this formula can be used for calculating the x-value of the maximum correlation coefficient in a second order polynomial.

A.3 Bilinear Interpolation

In mathematics, bilinear interpolation is an extension of linear interpolation for interpolating functions of two variables on a regular grid. The key idea is to perform linear interpolation first in one direction, and then in the other direction.

Given the two points at the end of the lines, the line is the linear interpolant between the points, and the value *y* at *x* may be found by linear interpolation (Figure 44).

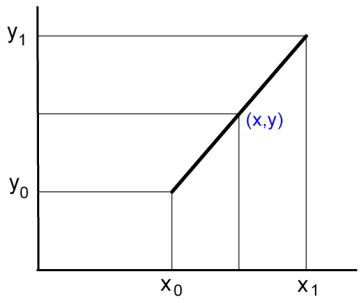


Figure 44 - Linear interpolation

Suppose that we want to find the value of the unknown function f at the point P = (x, y). It is assumed that we know the value of f at the four points $Q_{11} = (x_1, y_1)$, $Q_{12} = (x_1, y_2)$, $Q_{21} = (x_2, y_1)$, and $Q_{22} = (x_2, y_2)$.

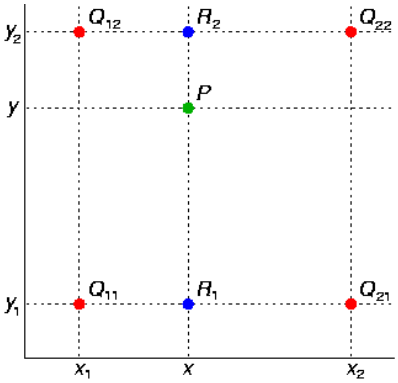


Figure 45 - Bilinear Interpolation

We first do linear interpolation in the x-direction. The value of R_1 is calculated by interpolating Q_{11} and Q_{21} and the value of R_2 is calculated by interpolating Q_{12} and Q_{22}. Then we calculate the value of P by doing linear interpolation between R_1 and R_2 (Figure 45).

B. Noise reducing filters

This chapter discusses the noise reducing filters we used for our research. These filters are mainly focused on improving the visual aspect of images and therefore do not necessarily need to have a good impact on the accuracy. These filters are presented and discussed by Gonzalez [5].

When the only degradation present in an image is noise, the following equation can be used

$$g(x, y) = f(x, y) + \eta(x, y)$$

Where $f(x, y)$ denotes the pixel intensity value in point (x, y) in the noiseless image, $\eta(x, y)$ the noise intensity value and $g(x, y)$ the pixel intensity value as is in the noise degraded image.

All filters we discuss here use a neighbourhood S_{xy} of $m \times n$ pixels, centered at point (x, y), to calculate the new intensity value of this pixel.

Arithmetic mean filter

The simplest example is a mean filter which calculates the arithmetic mean of the pixels in the region defined by S_{xy} for pixel $g(x, y)$. In other words

$$f(x, y) = \frac{1}{mn} \sum_{(s,t) \in S_{xy}} g(s,t)$$

This operation is implemented as a spatial filter of size $m \times n$ in which all coefficients have value $1/mn$. A mean filter smoothes local variations in an image, and noise is reduced as a result of blurring.

Geometric mean filter

An image restored using a geometric mean filter is giving by the expression

$$f(x, y) = \left[\prod_{(s,t) \in S} g(s,t) \right]^{\frac{1}{mn}}$$

Here, each restored pixel is given by the product of the pixels in the sub image window, raised to the power $1/mn$. A geometric mean filter achieves smoothing comparable to the arithmetic mean filter, but it tends to lose less image detail in the process.

Harmonic mean filter

The harmonic mean filtering operation is given by the expression

$$f(x,y)=\frac{mn}{\sum_{(s,t)\in S_{xy}}\frac{1}{g(s,t)}}$$

The harmonic mean filter is member of a set of nonlinear mean filters which are better at removing Gaussian type noise and preserving edge features than the arithmetic mean filter.

Median filter

Order-statistic filters are spatial filters whose response is based on ordering (ranking) the values of the pixels contained in the image area encompassed by the filter. The ranking result determines the response of the filter. The best-known order-statistic filter is the median filter, which replaces the value of a pixel by the median of the intensity levels in the neighborhood of that pixel:

$$f(x,y)=\underset{(s,t)\in S_{xy}}{\text{median}}\{g(s,t)\}$$

Median filters provide excellent noise-reduction capabilities, with considerably less blurring than linear smoothing filters.

Adaptive, local noise reduction filter

The behavior of adaptive filters changes based on statistical characteristics of the image inside the filter region defined by the $m\times n$ rectangular window S_{xy}.

The simplest statistical measures of a random variable are its mean and variance. These are reasonable parameters on which to base an adaptive filter because they are quantities closely related to the appearance of an image. The mean gives a measure of average intensity and the variance gives a measure of contrast in the region over which they compute.

The response of the filter at any point (x,y) on which the region S_{xy} is centred, is based on four quantities: (a) $g(x,y)$ the value of the noisy image at (x,y);(b) σ_μ^2, the variance of the noise corrupting $f(x,y)$ to form $g(x,y)$; (c) m_L, the local mean of the pixels in S_{xy}; and (d) σ_L^2, the local variance of the pixels in S_{xy}. This adaptive filter can be written as

$$f(x,y)=g(x,y)-\frac{\sigma_\mu^2}{\sigma_L^2}[g(x,y)-m_L]$$

The only quantity that needs to be known or estimated is the variance of the overall noise. For our captured images the variance of the present noise is 5.4. It can happen that in a certain region the σ_L^2 becomes lower than σ_μ^2. To prevent that nonsensical results influence the outcome, we set the ratio to 1 when $\sigma_\mu^2>\sigma_L^2$.

C. Measurement results

This appendix contains the accuracy results of all the combination of methods. First the accuracy results on the pixel level measurements are displayed. Then the results for the subpixel level estimation methods are shown. These are the 3-σ value of the deviations, therefore the lower the value, the better the accuracy.

Table 11 - Accuracy results on pixel level

Dust	Flat-field Correction	Correlation window size	Algorithm	
			Cross Correlation	FFT
No	No FF	16x16	2.853	3.642
		24x24	3.059	3.580
		32x32	2.923	3.729
		48x48	3.403	3.851
		64x64	3.389	4.001
	FF	16x16	2.737	3.358
		24x24	2.732	3.452
		32x32	2.745	3.501
		48x48	2.829	3.552
		64x64	2.565	3.616
Yes	No FF	16x16	202.835	202.765
		24x24	202.835	202.733
		32x32	202.835	202.699
		48x48	202.835	202.361
		64x64	202.835	202.164
	FF	16x16	2.571	3.358
		24x24	2.772	3.429
		32x32	2.707	3.499
		48x48	2.781	3.552
		64x64	2.582	3.616

Table 12 - Accuracy results on subpixel level

Flat-field correction	Noise reduction	Correlation window size	Algorithm			
			3-point	N-cubed	Bilinear	Corrected Bilinear
No FF	No	16x16	1,336	1,406	1,782	2,268
		24x24	1,451	1,866	1,910	1,774
		32x32	1,375	0,905	1,868	2,022
		48x48	1,589	1,648	1,747	2,899
		64x64	1,641	1,561	1,746	1,315
	Multi-sampling	16x16	0,973	0,609	1,265	2,079
		24x24	1,143	0,713	1,380	1,543
		32x32	1,163	0,642	1,455	1,894
		48x48	1,485	1,073	1,651	2,870
		64x64	1,526	1,219	1,627	1,266
	Arithmetic mean 3x3 pixels	16x16	0,963	0,925	1,215	1,856
		24x24	0,815	0,873	1,228	1,512
		32x32	0,730	0,707	1,056	1,630
		48x48	0,619	0,614	1,104	2,125

	64x64	0,580	0,592	1,073	1,088
Arithmetic mean 5x5 pixels	16x16	1,646	1,091	1,822	1,937
	24x24	1,109	1,075	1,722	1,528
	32x32	0,977	0,862	1,551	1,667
	48x48	0,725	0,721	1,488	2,143
	64x64	0,677	0,693	1,688	1,098
Arithmetic mean 7x7 pixels	16x16	2,519	1,283	2,400	1,980
	24x24	1,617	1,313	1,947	1,514
	32x32	1,379	1,120	1,937	1,663
	48x48	0,882	0,835	1,892	2,123
	64x64	0,890	0,817	1,880	1,099
Arithmetic mean 9x9 pixels	16x16	2,493	1,473	2,636	1,999
	24x24	2,620	1,516	2,700	1,557
	32x32	2,119	1,321	2,538	1,686
	48x48	1,076	0,935	2,050	2,160
	64x64	1,134	0,912	1,989	1,086
Geometric mean 3x3 pixels	16x16	1,005	0,930	1,273	1,852
	24x24	0,815	0,878	1,247	1,512
	32x32	0,733	0,711	1,061	1,630
	48x48	0,616	0,617	1,094	2,125
	64x64	0,581	0,591	1,067	1,088
Geometric mean 5x5 pixels	16x16	1,681	1,106	1,867	1,935
	24x24	1,168	1,081	1,845	1,528
	32x32	1,038	0,869	1,592	1,662
	48x48	0,738	0,725	1,540	2,146
	64x64	0,684	0,695	1,656	1,098
Geometric mean 7x7 pixels	16x16	2,596	1,306	2,570	1,988
	24x24	1,659	1,331	1,988	1,512
	32x32	1,443	1,145	1,923	1,669
	48x48	0,891	0,843	1,809	2,137
	64x64	0,915	0,821	1,827	1,099
Geometric mean 9x9 pixels	16x16	2,723	1,514	2,749	2,042
	24x24	2,488	1,542	2,644	1,563
	32x32	2,248	1,362	2,621	1,683
	48x48	1,088	0,946	2,049	2,159
	64x64	1,150	0,917	2,039	1,096
Harmonic mean 3x3 pixels	16x16	1,023	0,939	1,278	1,856
	24x24	0,815	0,879	1,287	1,516
	32x32	0,740	0,708	1,062	1,630
	48x48	0,614	0,616	1,092	2,125
	64x64	0,579	0,591	1,085	1,088
Harmonic mean 5x5 pixels	16x16	1,768	1,123	1,869	1,936
	24x24	1,200	1,092	1,841	1,511
	32x32	1,075	0,878	1,577	1,661
	48x48	0,738	0,731	1,576	2,146
	64x64	0,701	0,698	1,665	1,098
Harmonic mean 7x7 pixels	16x16	2,641	1,338	2,581	1,976
	24x24	1,694	1,351	2,098	1,521
	32x32	1,477	1,169	1,956	1,663
	48x48	0,921	0,849	1,828	2,159
	64x64	0,926	0,826	1,802	1,099
Harmonic mean	16x16	2,599	1,568	2,627	2,129

	9x9 pixels	24x24	2,529	1,566	2,529	1,563
		32x32	2,348	1,401	2,774	1,683
		48x48	1,064	0,955	2,105	2,195
		64x64	1,147	0,923	1,993	1,096
	Median 3x3 pixels	16x16	1,070	1,088	1,294	1,970
		24x24	0,891	0,979	1,374	1,526
		32x32	0,770	0,772	1,232	1,659
		48x48	0,642	0,668	1,205	2,157
		64x64	0,607	0,634	1,199	1,079
	Median 5x5 pixels	16x16	1,912	1,421	2,049	1,984
		24x24	1,374	1,220	1,731	1,521
		32x32	1,310	0,981	1,708	1,695
		48x48	0,821	0,834	1,299	2,150
		64x64	0,772	0,777	1,240	1,101
	Median 7x7 pixels	16x16	3,605	1,826	3,421	2,248
		24x24	2,189	1,637	2,359	1,555
		32x32	1,818	1,431	2,235	1,700
		48x48	0,933	0,984	1,639	2,191
		64x64	1,003	0,931	1,696	1,107
	Median 9x9 pixels	16x16	4,810	2,805	3,413	2,413
		24x24	4,315	2,329	3,240	1,578
		32x32	2,605	1,964	2,651	1,763
		48x48	1,216	1,134	2,075	2,212
		64x64	1,255	1,068	1,906	1,100
	Adaptive 3x3 pixels	16x16	0,947	0,901	1,269	1,829
		24x24	0,787	0,840	1,400	1,537
		32x32	0,742	0,717	1,323	1,634
		48x48	0,643	0,611	1,261	2,163
		64x64	0,619	0,581	1,353	1,097
	Adaptive 5x5 pixels	16x16	1,046	0,944	1,443	1,889
		24x24	0,922	0,916	1,589	1,568
		32x32	0,867	0,765	1,506	1,778
		48x48	0,804	0,661	1,407	2,210
		64x64	0,802	0,628	1,495	1,135
	Adaptive 7x7 pixels	16x16	1,095	0,979	1,510	1,913
		24x24	1,007	1,008	1,639	1,593
		32x32	0,967	0,832	1,641	1,847
		48x48	0,908	0,713	1,454	2,395
		64x64	0,925	0,669	1,564	1,199
	Adaptive 9x9 pixels	16x16	1,184	1,084	1,585	2,018
		24x24	1,158	1,173	1,680	1,632
		32x32	1,111	0,845	1,702	1,897
		48x48	1,081	0,763	1,588	2,428
		64x64	1,133	0,724	1,680	1,207
FF	No	16x16	1,233	1,781	1,764	2,189
		24x24	1,245	3,152	1,883	1,602
		32x32	1,193	1,115	1,803	1,901
		48x48	1,139	1,297	1,651	1,913
		64x64	1,027	0,860	1,672	0,875
	Multi-sampling	16x16	0,737	0,777	1,072	1,945
		24x24	0,895	0,874	1,267	1,396
		32x32	0,923	0,788	1,378	1,721

	48x48	1,046	0,961	1,396	1,871
	64x64	0,911	0,749	1,309	0,792
Arithmetic mean 3x3 pixels	16x16	0,979	0,942	1,219	1,815
	24x24	0,799	0,848	1,249	1,466
	32x32	0,730	0,712	1,112	1,557
	48x48	0,549	0,544	1,024	1,668
	64x64	0,495	0,496	1,085	0,809
Arithmetic mean 5x5 pixels	16x16	1,735	1,092	1,972	1,945
	24x24	1,084	0,997	1,672	1,472
	32x32	0,946	0,846	1,610	1,583
	48x48	0,753	0,613	1,515	1,654
	64x64	0,670	0,585	1,588	0,809
Arithmetic mean 7x7 pixels	16x16	2,393	1,268	2,431	1,969
	24x24	1,416	1,191	1,994	1,478
	32x32	1,503	1,107	2,026	1,632
	48x48	0,885	0,712	1,727	1,661
	64x64	0,783	0,690	1,771	0,829
Arithmetic mean 9x9 pixels	16x16	2,440	1,429	2,623	2,035
	24x24	2,314	1,376	2,398	1,503
	32x32	2,847	1,285	2,925	1,613
	48x48	1,230	0,813	1,978	1,672
	64x64	0,965	0,785	2,008	0,830
Geometric mean 3x3 pixels	16x16	1,014	0,943	1,305	1,858
	24x24	0,820	0,853	1,280	1,470
	32x32	0,746	0,711	1,092	1,557
	48x48	0,558	0,542	1,026	1,668
	64x64	0,496	0,496	1,101	0,809
Geometric mean 5x5 pixels	16x16	1,780	1,104	1,924	1,945
	24x24	1,055	1,004	1,652	1,467
	32x32	0,969	0,849	1,599	1,578
	48x48	0,763	0,617	1,547	1,656
	64x64	0,686	0,584	1,574	0,809
Geometric mean 7x7 pixels	16x16	2,574	1,290	2,631	1,954
	24x24	1,556	1,212	1,971	1,506
	32x32	1,619	1,127	2,235	1,626
	48x48	0,901	0,724	1,792	1,664
	64x64	0,798	0,696	1,842	0,829
Geometric mean 9x9 pixels	16x16	2,462	1,463	2,588	2,031
	24x24	2,280	1,402	2,453	1,503
	32x32	2,772	1,322	2,771	1,613
	48x48	1,296	0,829	1,998	1,673
	64x64	0,970	0,799	1,996	0,837
Harmonic mean 3x3 pixels	16x16	1,016	0,945	1,276	1,859
	24x24	0,826	0,853	1,259	1,470
	32x32	0,758	0,714	1,111	1,557
	48x48	0,563	0,542	1,035	1,668
	64x64	0,501	0,496	1,071	0,809
Harmonic mean 5x5 pixels	16x16	1,824	1,114	1,963	1,944
	24x24	1,064	1,008	1,653	1,467
	32x32	1,007	0,856	1,634	1,583
	48x48	0,766	0,620	1,501	1,656
	64x64	0,692	0,588	1,584	0,809

	Harmonic mean 7x7 pixels	16x16	2,766	1,315	2,777	1,965
		24x24	1,545	1,226	1,960	1,507
		32x32	1,808	1,150	2,287	1,629
		48x48	0,923	0,731	1,791	1,663
		64x64	0,815	0,706	1,801	0,830
	Harmonic mean 9x9 pixels	16x16	2,769	1,504	2,665	2,056
		24x24	2,312	1,425	2,456	1,501
		32x32	2,849	1,358	2,973	1,613
		48x48	1,381	0,845	2,091	1,672
		64x64	0,981	0,808	1,935	0,833
	Median 3x3 pixels	16x16	1,040	1,100	1,188	1,924
		24x24	0,877	0,961	1,257	1,448
		32x32	0,779	0,774	1,180	1,625
		48x48	0,587	0,601	1,092	1,679
		64x64	0,527	0,531	1,078	0,821
	Median 5x5 pixels	16x16	2,115	1,359	2,161	2,044
		24x24	1,380	1,188	1,847	1,465
		32x32	1,246	0,940	1,640	1,651
		48x48	0,800	0,686	1,228	1,680
		64x64	0,754	0,602	1,274	0,798
	Median 7x7 pixels	16x16	4,016	1,792	3,669	2,145
		24x24	2,083	1,492	2,290	1,503
		32x32	1,847	1,387	2,269	1,634
		48x48	1,118	0,858	1,790	1,696
		64x64	0,921	0,753	1,641	0,819
	Median 9x9 pixels	16x16	4,871	2,659	3,781	2,381
		24x24	4,795	2,192	3,486	1,514
		32x32	2,818	1,987	2,630	1,666
		48x48	1,808	0,989	2,290	1,674
		64x64	0,972	0,893	1,851	0,802
	Adaptive 3x3 pixels	16x16	0,869	0,942	1,215	1,781
		24x24	0,757	0,834	1,307	1,457
		32x32	0,722	0,739	1,188	1,609
		48x48	0,564	0,546	1,110	1,691
		64x64	0,488	0,463	1,152	0,824
	Adaptive 5x5 pixels	16x16	0,974	0,994	1,425	1,789
		24x24	0,817	0,896	1,497	1,458
		32x32	0,818	0,794	1,370	1,599
		48x48	0,640	0,585	1,292	1,696
		64x64	0,554	0,486	1,343	0,835
	Adaptive 7x7 pixels	16x16	1,020	1,046	1,535	1,891
		24x24	0,865	0,935	1,558	1,474
		32x32	0,915	0,840	1,568	1,671
		48x48	0,688	0,620	1,337	1,729
		64x64	0,618	0,503	1,441	0,842
	Adaptive 9x9 pixels	16x16	1,041	1,133	1,482	1,964
		24x24	0,937	1,123	1,556	1,538
		32x32	1,004	0,870	1,680	1,774
		48x48	0,766	0,652	1,450	1,752
		64x64	0,688	0,524	1,514	0,855