applied sciences

MDPI

Article

# Particle Swarm Optimisation in Practice: Multiple Applications in a Digital Microscope System

**Louis Ryan [1], Stefan Kuhn [2,*,†], Simon Colreavy-Donnely [3] and Fabio Caraffini [4]**

[1] Ash Technologies Ltd., B5, Newhall, M7 Business Park, W91 P684 Co. Kildare, Ireland
[2] Institute of Computer Science, University of Tartu, 51009 Tartu, Estonia
[3] Department of Computer Science and Information Systems, University of Limerick, Castletroy, V94 T9PX Limerick, Ireland
[4] School of Computer Science and Informatics, De Montfort University, The Gateway, Leicester LE1 9BH, UK
* Correspondence: stefan.kuhn@ut.ee
† Previously at Cyber Technology Institute, School of Computer Science and Informatics, De Montfort University, The Gateway, Leicester LE1 9BH, UK.

**Abstract:** We demonstrate that particle swarm optimisation (PSO) can be used to solve a variety of problems arising during operation of a digital inspection microscope. This is a use case for the feasibility of heuristics in a real-world product. We show solutions to four measurement problems, all based on PSO. This allows for a compact software implementation solving different problems. We have found that PSO can solve a variety of problems with small software footprints and good results in a real-world embedded system. Notably, in the microscope application, this eliminates the need to return the device to the factory for calibration.

**Keywords:** digital microscope; inspection system; metrology; particle swarm optimisation; heuristics; image stitching

check for updates

**Citation:** Ryan, L.; Kuhn, S.; Colreavy-Donnely, S.; Caraffini, F. Particle Swarm Optimisation in Practice: Multiple Applications in a Digital Microscope System. *Appl. Sci.* **2022**, *12*, 7827. https://doi.org/10.3390/app12157827

Academic Editor: Yujin Lim

Received: 20 July 2022
Accepted: 2 August 2022
Published: 4 August 2022

## 1. Introduction

Computational intelligence has been a major field of research. This includes stochastic optimisation methods, such as those from the fields of evolutionary computing and swarm intelligence [1,2]. The latter optimisation paradigms feature several heuristics that can be used over black-box problems, or any other challenging optimisation problem for which using an exact method would not be viable or time-efficient. For this reason, such heuristics approaches do not only represent an interesting fundamental research topic, but have also become very popular among practitioners benefiting from their wide-scope applicability in several domains of sciences, engineering and industry [3–6]; however, in the vast majority of cases, the literature reporting on such application studies is written from an academic point of view, i.e., highlighting novelty and technical aspects, often overlooking the impact created by the proposed technology in industry in more practical terms. In this study, we instead focus on showing that the introduction of well-known heuristic optimisation in technological devices can improve some characteristics of the product, making it not only more efficient, but also more usable, thus improving upon customer satisfaction. For this reason, we selected an established and simple heuristic for optimisation, namely the particle swarm optimisation algorithm (see Section 2.2 for details), and used it to address feedback from customers on a digital microscope/inspection system for medical applications produced by Ash Ltd. Through customer feedback, four areas where improvements would help in increasing the quality and simultaneously reduce costs were identified. We aim to show that, if areas of improvements are well formulated in the fashion of the optimisation problem, even embedding only one simple general-purpose algorithm within the device (we present the details in Section 2.1) can make a major difference. Moreover, we present the production-ready solutions for these four problems, consisting of

- An object height measurement task (Section 3);
- An image stitching for high-resolution images task (Section 4);
- An accurate colour reproduction task (Section 5);
- An image distortion correction task (Section 6).

Note that using a single general-purpose optimiser for the four problems is preferable in this specific context. Advantages from this approach are multiple, including that only a single software implementation of an optimisation algorithm needs to be developed, that the software/firmware is easier to maintain, that porting the produced software to different platform is faster, but also that training engineers becomes simpler, faster and less costly. It must be also pointed out that the four addressed tasks are measurement problems. As the digital microscope is generally used in a wide variety of applications by the customers who decide to purchase it, these measurements processes might be very different; therefore, optimising algorithms to a particular dataset to gain marginal improvements is not the aim. In fact, this may cause deterioration of results on other datasets [7]. This further supports our approach, which is also consistent with the tenets of Industry 4.0, in providing bespoke solutions to practical problems, in an efficient and inexpensive manner, both resource wise and computationally.

## 2. Materials and Methods

### *2.1. Omni 3 Digital Microscope Inspection and Measurement System*

The Omni 3 is a system for inspection and measurement purposes. It is based on digital imaging technologies and includes a zoom lens with a magnification range between 2.5 and 672. The system has integrated software applications which eliminate the need for an external computer. Those applications include:

- Fast autofocus function;
- Image stitching;
- Image stacking;
- 2D and 3D measurement and graticules;
- Image comparator;
- Object counting;
- Colour analysis;
- 2D measurement functions including point to point, polyline, circle and rectangles.

The system achieves a 2D measurement accuracy of $+/-1\%$ and a Z-height accuracy of 100 µm. A full description of the system can be found at [8].

All software was developed using OpenCV [9] and the Qt C++ framework [10]. Other software and libraries used are outlined in the sections where they are used.

For an overview of the field of metrology and a literature review, we refer the reader to [11].

### *2.2. Particle Swarm Optimisation*

Particle swarm optimisation [12,13] is a swarm intelligence metaheuristic for optimising continuous nonlinear problems defined on an $n$-dimensional search space $\mathbf{D} \subset \mathbb{R}^n$ (characterised by $n$ lower and upper bounds, i.e., $l_i$ and $u_i$ with $i = 1, \ldots, n$, in this work). Its popular working mechanism is quite simple, and it is iterated until a stop condition is met—with the most used stop criteria being a fixed number of objective functional calls, usually expressed in terms of maximum number of iterations $I_{\max}$, failing at improving upon the objective value of the best ever found solution $\mathbf{x}^{GB}$ for a number $F_{\max}$ of iterations, or the achievement of a satisfactory level of accuracy for the solution (measured with a threshold $\tau$).

Each iteration consists of the perturbation, and interaction, of a number of particles $NP$ associated with their current position in the search space $\mathbf{x}_j \in \mathbf{D}$ (i.e., the candidate solution), their velocity $\mathbf{v} \in \mathbb{R}^n$, and the'personal best' ever explored position $\mathbf{x}_j^{PB}$ (with $j = 1, \ldots, NP$). Note that a solution is said to be better, or to outperform another one, if its objective function value is lower while facing a minimisation problem—the opposite for a maximisation one. The same implementation of a PSO can be used for both minimisation and maximisation processes by simply multiplying the objective function $f(*)$ by $-1$ as needed. A complete iteration requires calling $f(*)$ for $NP$ times as each solution $\mathbf{x}$ (which is initially randomly generated in $\mathbf{D}$) must be perturbed to attempt to explore a new position whose functional value outperforms the one of the global best solution $\mathbf{x}^{GB}$ to update it. This is achieved by applying the following operators to the the velocity and position vectors of the $NP$ particles:

$$\mathbf{v}_j^{(new)} = w\mathbf{v}_j + r_1\left(\mathbf{x}_j^{PB} - \mathbf{x}_j\right) + r_2\left(\mathbf{x}_j^{GB} - \mathbf{x}_j\right) \tag{1}$$

$$\mathbf{x}_j^{(new)} = \mathbf{x}_j + \mathbf{v}_j^{(new)} \tag{2}$$

where $w$ is the 'inertia' weight (controlling the exploratory step size [14]) while $r_1$ and $r_2$ are random weights uniformly drawn within $[0,\ c_1]$ and $[0,\ c_2]$, respectively. The acceleration constants $c_1$ and $c_2$ are user-defined to have more pressure towards local best solutions, the global best solution or to achieve a balanced search. More details can be obtained from [15]. Generally employed settings from the literature see $c_1 = c_2 = 2$ as good choices for several applications [12], while $w = 0.7$ is a quite established value, even though adaptive formulas for dynamically adjusting $w$, see, e.g., $w \in [0.4, 0.9]$ [16,17], as well as for adjusting the acceleration constants, see, e.g., [18–20], also exist. Note that the velocity vectors, which act more as a displacement vector for the current position, are usually initialised at random within $\mathbf{D}$ (small values are to be preferred [21]) can then assume any arbitrary value. This may lead to the so-called 'velocity explosion' undesired effect, which should be mitigated, e.g., as shown in [22], and by avoiding a too small swarm size [23]. Our implementation controls the velocity value via the 'clamping' method from [22]. Finally, we employ the 'absorbing walls' method in [24] to deal with infeasible solutions that might be generated by the PSO logic during the search for optima. This consists of saturating the infeasible positions of a generic $j$th particle (i.e., those components of $\mathbf{x}_j$ falling outside the boundaries) to the closest 'violated' boundary re-initialising the corresponding velocity vector $\mathbf{v}_j$ with zeros.

This simple working logic, together with good performances over benchmark and real-world problems, made PSO popular and widely used by practitioners, who use it over a variety of application fields such as engineering [25,26] and health-care, including environmental, industrial and commercial applications and many other general optimisation tasks [27–29]. PSO variants for discrete domains have also been proposed to tackle, e.g., planning and scheduling problems [30].

## 3. Accurate Objects' Height from Focus Measurement

The conventional approach to determine the height of a generic object under inspection with the digital microscope is based on the fixed single thin lens optical system model. This features a linear relation between the distances $d_o$ and $d_i$ (from the object and the image to the lens, respectively), as formulated in Equation (3) where $f$ is the focal length of the lens [31].

$$\frac{1}{f} = \frac{1}{d_o} + \frac{1}{d_i} \tag{3}$$

This simple equation makes it easy to determine the distance from the camera to the object by simply moving the camera towards and away from the object and finding the best in focus image [32]. Note that in cameras in practical use, things become more complex. This is because they require multiple lenses and also due to the fact that properties of the lenses are generally confidential, and that stepper motors used to move the internal focus lens do not have a linear relationship with the focus distance from the camera. Hence, the 0–25 mm admissible range of our system was initially discretised in 26 checkpoints, by moving the camera with a 1 mm step, to generate a lookup table matching corresponding heights of the object and peak focus positions of the camera. The calibration jig for performing this process is shown in Figure 1. This is used to move the camera in the vertical direction along a linear guide rail whose lead screw is turned by the stepper motor. Each movement of the camera, whose height is recorded through 10 µm resolution linear encoder, is equivalent to changing the height of the test sample. All communication from the PC software to the stepper motor occurs through RS232 ports.

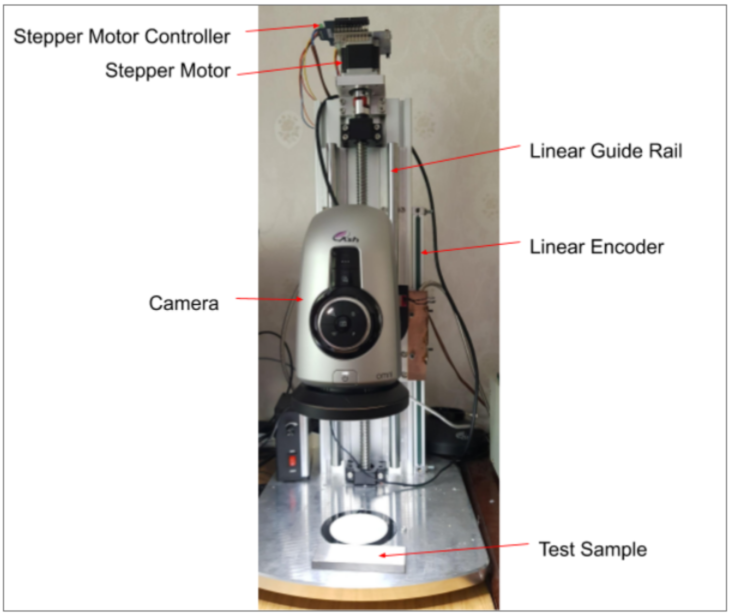


**Figure 1.** Camera calibration jig.

Using the lookup table to estimate the height of an object proved to be problematic for two main reasons. First, measuring focus is a noisy process, which often returns poor peak focus measurements. In turn, the latter leads to unreliable height values. Indeed, while moving the camera, a built-in function (set up in auto focus mode) is called to find the peak focus energy with the Laplacian operator method [33] to: determine the best focus position; read the focus motor position value; generate a table of 'focus value vs. object height' data; however, incorrectly returned position values negatively impact on the calibration process, thus introducing errors. We empirically confirmed this by noticing that the peak focus energy we measured clearly differed from the peak of the data when a polynomial is fitted to the data. Additionally, significant outliers in calibration data are likely to be present due to vibrations and changes in lighting conditions on the factory floor. Second, it requires re-calibrating the system accurately due to non-consistent optical/mechanical characteristics across cameras. This is achieved by returning the microscope back to the factory, where the previously stored 26 checkpoints are used to perform the camera re-calibration process. As the 26 points cannot always represent the ideal baseline, this approach is still prone to poor re-calibration outcomes. Obviously, having to ship the system for re-calibration adds costs and delays, and it is undesirable considering that these systems are expected to be in constant use. Hence, a built-in automatic calibration routine capable of mitigating the noisy component of the focus measurement would significantly and simultaneously improve performance, customer experience and value for money.

To overcome these problems, we present a self-calibrating approach where uncertainty is mitigated. Instead of using the 26 noisy measured peaks, we augmented the data-set to make it more robust and accurate, and we found the optimal coefficients for a polynomial fitting them, thus being able to generate the required lookup table with a smoother function and with any preferred number of points (this approach is known to reduce noise [34]).

In the following subsection, we first show the experimentation performed to increase the sample size from 26 to 260 samples (Section 3.1), to then show how we determined the order of the polynomial function to best fit that augmented data set (Section 3.1.2). Once found, such coefficients are to be stored onto each microscope before shipping it to customers and the firmware of the microscopes is also upgraded to exploit the polynomial function for performing automatic re-calibration and determine the height of the object accurately without the need of returning it to the factory.

### *3.1. Optimised Polynomial Fitting*

We make use of the PSO algorithm, as introduced in Section 2.2, to optimise the polynomial interpolator so that the mean squared error (MSE) between observed values $x_k$ and predicted values $\hat{x}_k$, as defined in Equation (4), is minimised.

$$\mathrm{MSE} = \sum_{k=1}^{SZ} (x_i - \hat{x}_k)^2 \tag{4}$$

#### 3.1.1. Experimental Set-Up

We performed a preliminary experiment to determine the most appropriate number of points, i.e., the sample size $SZ$, to use for evaluating the MSE objective function. For this, we used synthetic data obtained by generating a ground truth of height samples from a 2nd order polynomial setting of coefficients $a_0 = 3.4$, $a_1 = 0.022$ and $a_2 = 0.00003$ ($\pm 2$ steps of uniform noise in the range of 0 to 2 is added to the ground truth data).

To obtain the best polynomial function match, we ran the PSO algorithm (parameters are empirically tuned on the smallest $SZ$ returning the minimum error ) set with $NP = 50$; $c_1 = 1.50$; $c_2 = 0.30$; $w = 0.92$; velocity components clamping within $[-0.05, 0.05]$; $I_{\max} = 5000$; but with increasing height samples $SZ$, from 25 to 1000. The setup for changing $SZ$ is shown in Figure 1. For each height sample size, 10 independent runs of the PSO were performed. This process returns the polynomials functions required to calibrate the systems. As this is a measurement system, its accuracy is determined by its maximum error. Hence, for each run, we also record the corresponding max step error and compute their average. As evident from Table 1, it could be noted that this averaged max error (which is expected to be smaller when the sample size is higher) stopped decreasing at $SZ = 250$. Hence, we use and suggest this sample size (plus 10 extra points to allow for a margin of error) for fitting the polynomials with the PSO algorithm (i.e., we work with $SZ = 260$).

**Table 1.** Finding $SZ$ for best polynomial match. Selected values are shown in boldface.

| $SZ$ | 25 | 50 | 75 | 100 | 150 | 200 | **250** | 500 | 1000 |
|---|---|---|---|---|---|---|---|---|---|
| Max Step Error Average | 1.33 | 0.48 | 0.46 | 0.43 | 0.39 | 0.33 | **0.22** | 0.22 | 0.22 |

#### 3.1.2. Validation on Real Data

Our method was validated experimentally. We employed a 3rd, a 4th and a 5th order polynomial function to understand if higher order interpolators led to better results. Note that these require from 4 up to 6 coefficients $a_i \in [l_i,\ u_i]$, with $i = 0, 1, 2, 3, 4, 5$. The corresponding search spaces for the problems are indicated via the following lower bound vector **l** = [−400, 0, −3, −0.003, −0.00003, −0.000001] and upper bound vector **u** = [0, 50, 3, 0.003, 300,000, 1,000,000]. Before applying the PSO approach, a camera was randomly selected and calibrated five times using accurate data measured in the factory to produce the 'ground truth' described in Section 3.1.1. Results showing the height of the camera (in terms of steps) are reported in Table 2. These show errors varying from 3.38 steps to 7.65 steps. Subsequently, we fit the three polynomial functions with the PSO algorithm and used it to add calibration points as previously explained. The second approach led to better results, in particle when using the 4th order polynomial function, which led to a reduction in the error to the range of 0.24–0.53 steps.

**Table 2.** Validation results on five calibration processes—reported in steps.

| | | | | | |
|---|---|---|---|---|---|
| ground truth | 3.38 | 7.07 | 7.65 | 5.97 | 5.98 |
| 3rd order polynomial | 0.34 | 0.45 | 0.43 | 0.51 | 0.7 |
| 4th order polynomial | 0.24 | 0.31 | 0.39 | 0.32 | 0.53 |
| 5th order polynomial | 0.39 | 0.31 | 0.39 | 0.29 | 0.95 |

It must be remarked that the proposed approach, with the suggested settings for the PSO and the use of a 4th order polynomial for the regression problem at hand, has shown to be very robust also in presence of outliers. Indeed, we replicated the experimental phase by manually adding five outliers in the calibration data and were able to achieve the 0.24 steps error after the optimisation process.

As we make use of a heuristic method for fitting the polynomial function, we also compared the performances of the employed PSO to those obtained of another state-of-the-art stochastic optimisation algorithm, namely the CMAES [35]. The latter has a higher computational complexity and it has been applied to five data sets acquired from a single camera run through the calibration process and compared to the PSO when fitting the 4th order polynomial function to the data set; however, as shown in Table 3, only very marginal improvements were registered on the objective functions. Such small variations do not lead to better calibration processes, in favour of using PSO over CMAES due to its faster and simpler algorithmic structure.

**Table 3.** Logged MSE values for PSO and CMAES.

| | | | | | |
|---|---|---|---|---|---|
| PSO | 271.3 | 401.28 | 355.3 | 374.7 | 319.2 |
| CMAES | 270.8 | 398.7 | 354.3 | 373.4 | 318.7 |

## 4. Image Stitching

Image stitching is a method of blending two or more overlapped images into a higher resolution image. The requirement for image stitching can arise when an object is larger than the field of view of the microscope or when high-resolution images of an object are needed for quality assurance. Figure 2 shows an example of two images of the same object taken with overlapping regions. In order to blend these images together the exact region of overlap needs to be found. In microscopy applications where the sample is moved manually, translation and rotation are required. When using a manual *XY* table only translation is required.

The established methods for solving this problem largely fall into two categories, feature point matching and template matching:

### *4.1. Feature Point Matching*

Scale invariant feature transform (SIFT) [36] and speeded-up robust features (SURF) descriptor [37] are commonly used to detect common features in images and RANSAC [38] is used to match these features and calculate the homography. The images are then blended together to form a higher resolution image.

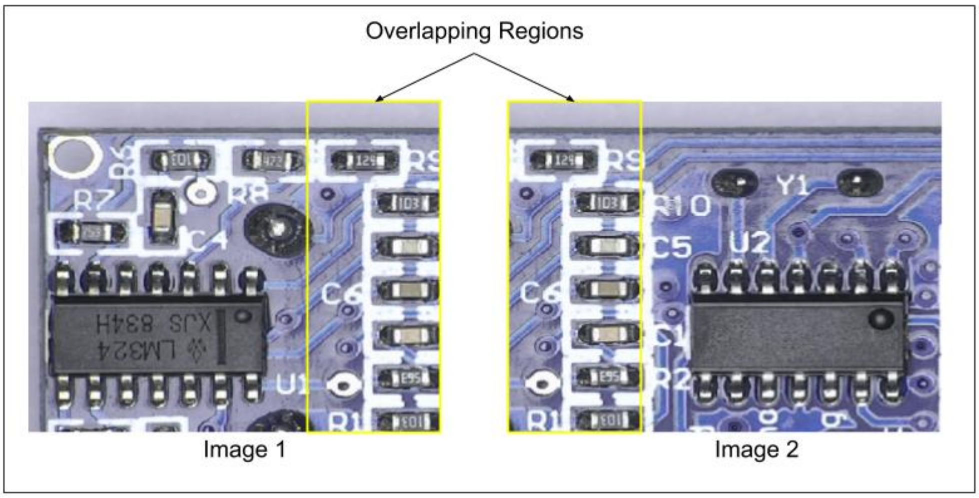


**Figure 2.** Example for the image stitching problem, where the two yellow areas represent the overlap to be found.

### *4.2. Template Matching*

There are many objects where registration by the detection of feature points is unsuitable. Figure 3 shows an example of such objects. If an interest point algorithm was used to find a match for the areas marked by the yellow rectangle it would not be robust due to the lack of sufficient features in these areas.

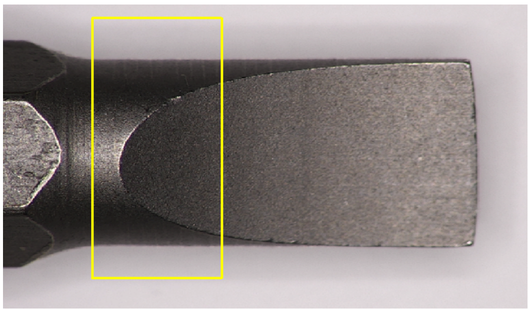


**Figure 3.** Example object not suitable for interest point matching.

Although interest points may not be suitable for these objects, a method called template matching can be used. The most basic form of template matching for translation is sliding the smaller template image over the larger image, and subtracting the pixel intensity of the template image from the larger image at each pixel position and obtaining the sum of the absolute or squared error of the pixel intensity differences. Where the difference is lowest, the best match is found. If multiple matches are required, then a threshold can be set and any match below this threshold is deemed a match.

A printed circuit assembly (PCA) is typically rich in colours with the printed circuit board (PCB) background, switches, connectors, resistors, capacitors and integrated circuits. Using a hue, saturation and value exploits the variety of colour.

An experiment was carried out to determine the difference between template matching using the components of the HSV colour space. A template was selected, slid over the master image and the loss function plotted as colour maps. The loss functions were the absolute difference in hues, saturations, values and a combination of all three. As per Figure 4, the hue matching results in a wider funnel at the best match position (global minimum) but have more local minimums. The saturation and value (Lightness or grayscale) result in a much narrower global minimum funnel but less local minimums. A combination of all three HSV components results in a narrow global minimum and the least local minimums.

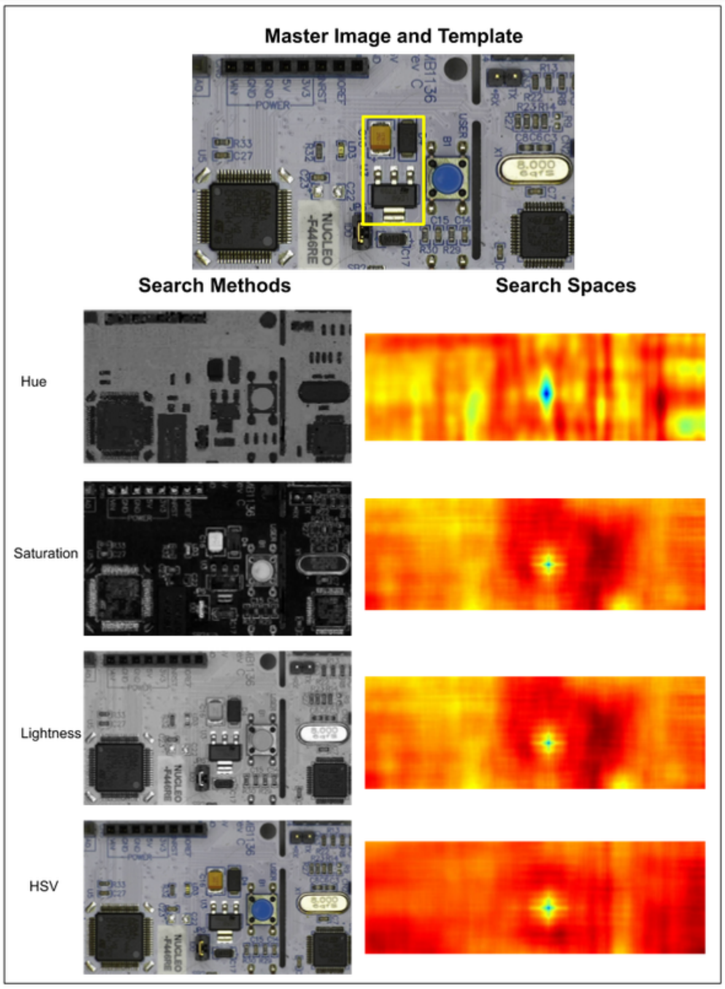


**Figure 4.** Loss in HSV for image template matching (right) over an example image (top) to a section of the image (yellow box). Blue indicates no loss and dark red indicates the maximum loss.

Given a master image size of width 640 and height of 360 and a template of width 105 pixels and height 180 pixels, the number of absolute difference operations is 96,300 and takes 3.54 s on an Intel i5 quad core processor running at 1.6 GHz with 8 GByte RAM. This could be sped up by using different scales e.g., scaling the template and master image down and finding the global minimum, repeating this process at different scales up to 1:1 but only searching in the area of the previous global minimum.

### *4.3. Accelerating Image Template Matching Using PSO*

In order to speed up the search compared to the exhaustive search described previously, a PSO was used with $NP = 50$; $c_1 = 0.9$; $c_2$ linearly adjusted from 0.1 to a maximum of 5; $w = 0.9$; velocity components clamping within $[-0.3, 0.3]$; $I_{\max} = 5000$. These values were empirically found.

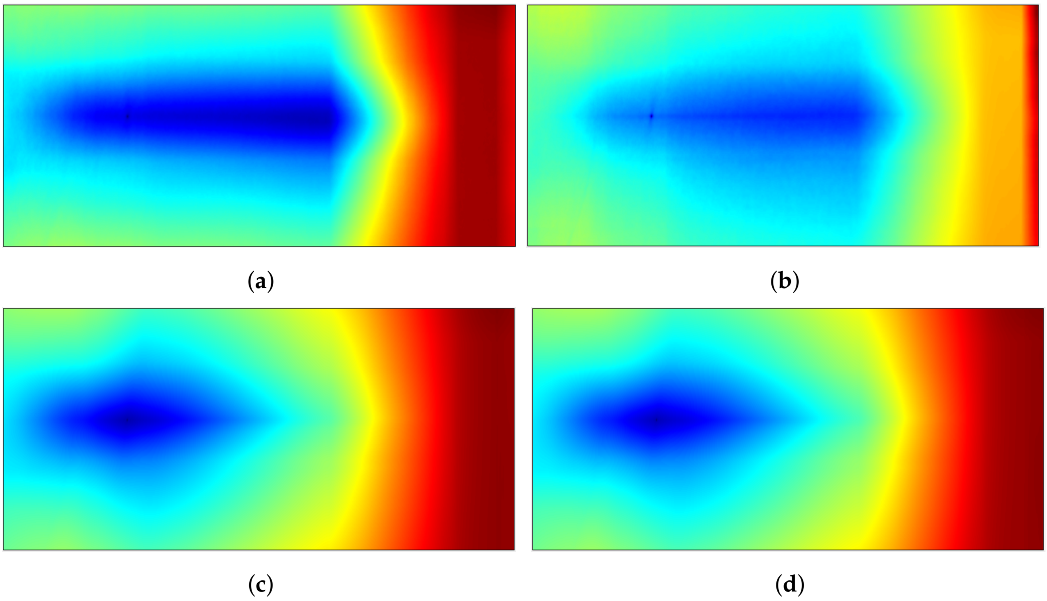
To achieve our goal with the PSO method, a loss function to be minimised is defined as the absolute difference of the error when subtracting the template image from the image predicted by each particle. The loss can be the hue, saturation, value (Lightness) or all three. Note that the lightness value was used in this investigation. Particles encode an $(x, y)$ point whose boundaries are the image (template) size.

A method was implemented to allow for more exploration at the start of the search and exploitation or convergence at the end of the test by varying the social coefficient $c_2$. When $c_2$ is selected, the search starts out with a setting of 0.1 until the search has searched half of the search cycles. After that $c_2$ is linearly increased from 0.1 to 5.0 for the second half of the search cycle. Results for the variable and fixed $c_2$ method are reported in Table 4.

We used the PSO-optimised image stitching approach for the test cases in the following paragraphs.

#### 4.3.1. Experiment 1

The object in Figure 3 was used to evaluate the PSO search algorithm for the application of image stitching. The loss over the search space is shown in Figure 5. Lightness and HSV have distinct minimums and a funnel-like structure around the minimum making them ideal for a search algorithm. As lightness required one third of the computation, it was chosen as the matching method.


(**a**) (**b**)

(**c**) (**d**)

**Figure 5.** Loss in HSV over the image used for experiment 1. Blue indicates no loss and dark red the maximum loss: (**a**) hue; (**b**) saturation; (**c**) lightness; (**d**) HSV.

The PSO was run 10 times.The average speed of the PSO search, 365 ms, is significantly faster than the previous method, which requires 2.85 s to achieve very similar performances. This means that the PSO approach does not deteriorate the performance with a 87% decrease in the search time—see Table 4.

**Table 4.** PSO results for experiment 1 obtained with both fixed value variable values for the $c_2$ parameter. The ground truth is (258, 112).

| Run | Fixed $c_2$ | Variable $c_2$ | Time (ms) |
|---|---|---|---|
| 1 | (257, 112) | (258, 112) | 350 |
| 2 | (258, 112) | (258, 112) | 374 |
| 3 | (258, 112) | (258, 112) | 356 |
| 4 | (258, 112) | (258, 112) | 355 |
| 5 | (258, 112) | (258, 112) | 364 |
| 6 | (258, 112) | (258, 112) | 365 |
| 7 | (258, 112) | (258, 112) | 365 |
| 8 | (258, 113) | (258, 112) | 365 |
| 9 | (258, 112) | (258, 112) | 365 |
| 10 | (258, 112) | (258, 112) | 365 |

#### 4.3.2. Experiment 2

The PCB selected in Figure 6 was used to evaluate the PSO search algorithm for the application of image stitching. The search space with the loss is shown in Figure 7. All search spaces have a very small point indicating the matching position.

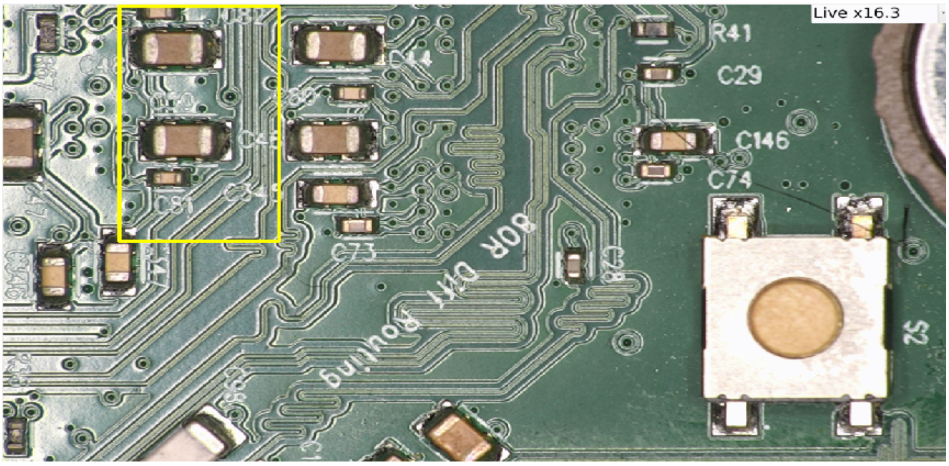
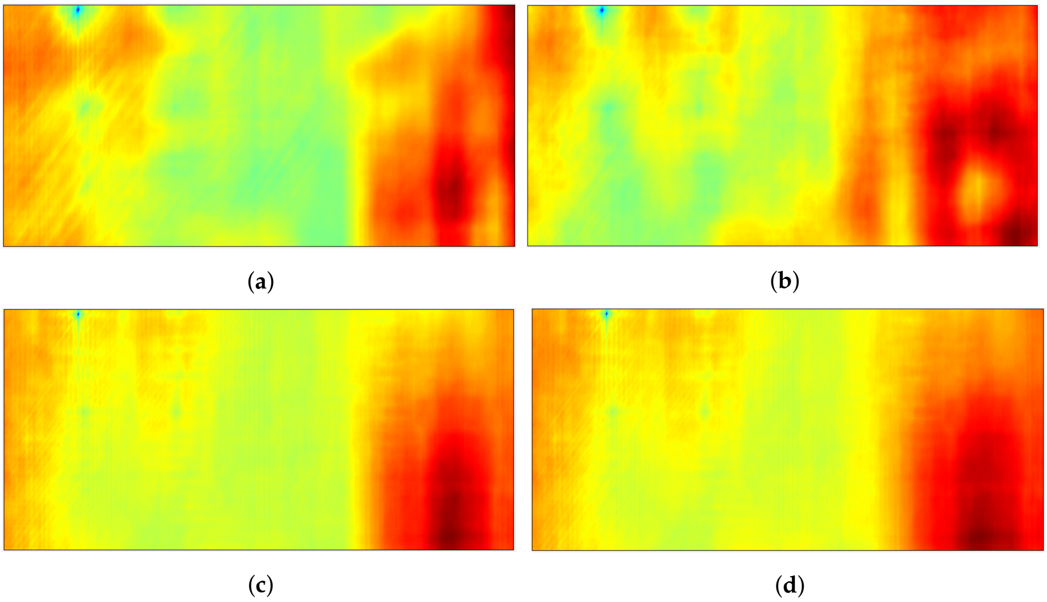


**Figure 6.** The yellow area is matched to the overall image in experiment 2.



(**a**) (**b**)

(**c**) (**d**)

**Figure 7.** Loss in HSV over the image used for experiment 2. Blue indicates no loss and dark red the maximum loss: (**a**) hue; (**b**) saturation; (**c**) lightness; (**d**) HSV.

#### 4.3.3. Results

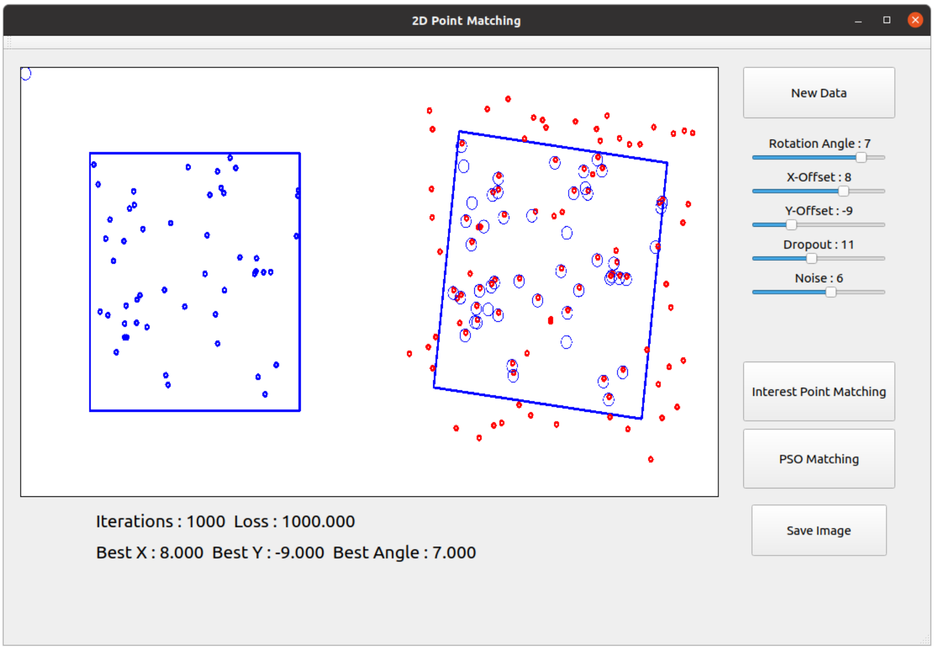
Full search template matching using hue only search time was 2.25 s. In order to obtain accurate matching using PSO search a large number of particles (500) was needed to ensure the area with the lowest loss was explored. Similarly to Experiment 1, minor errors occurred were C2 was fixed but none where C2 was varied during the search. The number of cycles was set to 15 and the search time 1.25 s resulting in a 45% speed increase in run time.

### *4.4. 2D Point Cloud Matching*

#### 4.4.1. Method

There are many objects in the medical device industry where high-resolution images are required for quality assurance. Some of these objects have little or no features that can be used as interest point. At the same time, they have areas that are identical, which eliminates the template matching method. So both the described methods are not applicable. For such objects interest points can be added by use of a random point laser.

For this, a 2D point cloud matching application (Figure 8) was developed using synthetic point data.



Rotation Angle : 7
X-Offset : 8
Y-Offset : -9
Dropout : 11
Noise : 6

Iterations : 1000 Loss : 1000.000
Best X : 8.000 Best Y : -9.000 Best Angle : 7.000

**Figure 8.** The 2D point cloud matching application.

In total, 100 points were generated with the X and Y coordinates uniformly randomly generated between $l_i = 0$ and $u_i = 300$ (Point Cloud 1). A second set of points were generated by mutating the first set with programmable random dropout where the point is replaced with a new random point and programmable point mutation where a random noise is added to the point position (Point Cloud 2). In addition to the mutations each point can be translated ±20 positions in X and Y and rotated $+/-20$ degrees. An additional 50 points are added to point cloud 2 outside the 0 to 300 area.

A PSO algorithm with the settings $NP = 50$, $c_1 = 2$, $c_2 = 0.3$; $w = 0.9$, velocity components clamping within 30% of the search space with $u_i - l_i = 300$ and $I_{\max} = 1000$ was developed. Each particle has a predicted $X, Y$ offset and rotation angle. The loss function computes the predicted dataset by offsetting each point in point cloud 1 by its $X, Y$ and rotating it by the rotation angle. For each point in the predicted dataset the distance to its nearest point in point cloud 2 is calculated. The sum of these distances squared is used as the loss function to be minimised.

### 4.4.2. Results

The PSO matching time for a 2D point cloud matching is approximately 5 s and the matching function was 100% accurate on every run with maximum dropout (25%) and noise (10%).

## 5. Colour Reproduction Accuracy

The colour reproduction accuracy is very important in digital microscopes as the colour sensitivity of the sensor seldom reproduces the colour space required by monitors [39]. In 1996, HP and Microsoft developed a standard colour space (sRGB) for use in monitors, printers and the internet [40]. In order to render the image on a display as accurately as possible these RGB pixel values need to be mapped to the device-independent colour space sRGB.

Figures 9 and 10 and demonstrates the essence of the challenge. A reference colour chart (Figure 9) is placed under the microscope and an image (Figure 10) taken with no colour correction applied. The measured colours are not an accurate representation of the reference colour chart and look washed out/lack vibrancy.

In the digital microscope, colour correction is part of the simplified image processing pipeline shown in Figure 11.

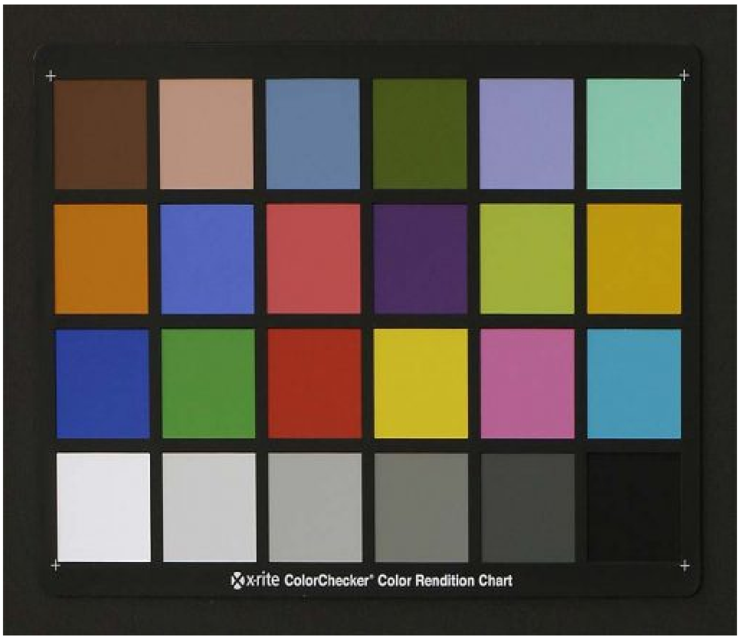


**Figure 9.** Macbeth chart.

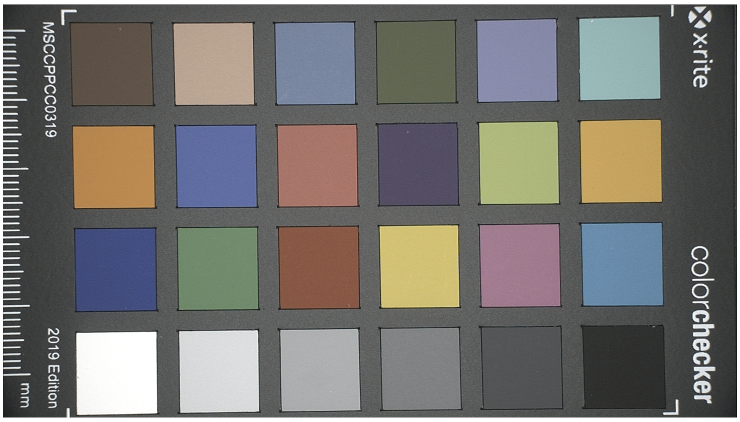


**Figure 10.** Measured colours.

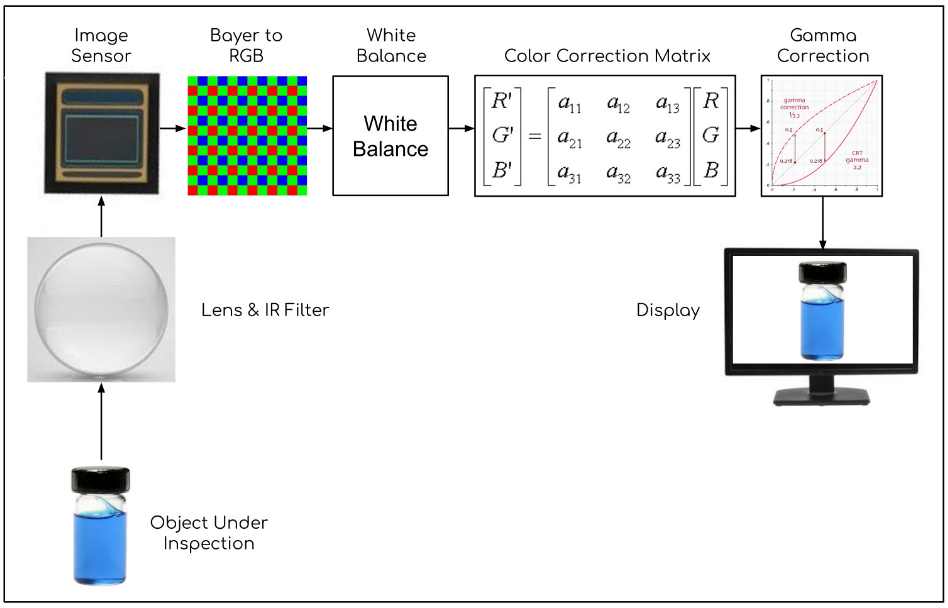


Image Sensor
Bayer to RGB
White Balance
Color Correction Matrix
Gamma Correction
White Balance
$\begin{bmatrix} R' \\ G' \\ B' \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$
gamma correction 1/2.2
CRT gamma 2.2
Lens & IR Filter
Display
Object Under Inspection

**Figure 11.** Microscope image processing pipeline.

The image processing pipeline works as follows:

- Light from the object under inspection is focused on the image sensor using a single or combination of lenses. Infrared light is blocked using an infrared filter.
- The image sensor converts the incoming light to a Bayer image.
- The Bayer image is converted into a red, green and blue (RGB) image.
- The RGB image is white balanced.
- The RGB colour pixel measurements that are dependent on the sensor, optics and lighting characteristics do not accurately represent the actual object RGB colours and are corrected by a colour correction matrix.
- Gamma correction is applied to match the gamma of the display.

There are many methods of mapping from device dependent RGB to the sRGB space, such as look up tables, linear and polynomial regression and neural networks [41]; however, a simple 3 × 3 matrix linearly transforming the device dependent RGB to the sRGB space “is not easily challenged” [41]. In addition, the application of this mapping is conducted in real time and performed in an FPGA, ruling out alternative methods due to speed and resource usage.

In order to find a 3 × 3 colour correction matrix, a number of known sRGB colours are measured by the sensor and a matrix is derived that maps these measured values to the known sRGB colours. A reference colour checker chart [42] was imaged by the camera. A region of interest (Figure 12) within each colour was selected and the mean RGB values of that region of interest calculated.

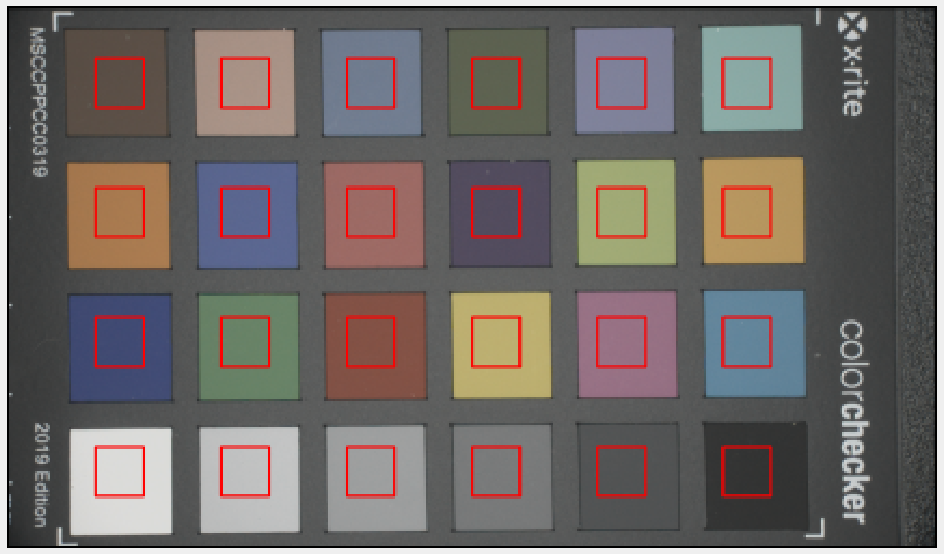


**Figure 12.** X-rite colour checker chart measured by the camera showing the region of interest areas.

A table of colour regions vs. reference sRGB values [43] provides the ground truth for each colour in the chart.

The reference sRGB values are non-linear, with an exponent Gamma of approximately 2.2 applied to their normalised values [40]. The measured RGB values also have a Gamma of 2.2 applied to their normalised values within the FPGA. To convert these into linear space $\text{RGB}_\text{L}$ where a linear 3 × 3 matrix can be used to transform the device-dependent RGB values to device-independent sRGB values, each RGB value is divided by 255 and a Gamma power of 2.2 is applied.

$$\text{RGB}_\text{L} = (\text{RGB}/255)^{2.2} \tag{5}$$

Table 5 shows the measured, reference sRGB values and the 3 × 3 colour correction matrix.

**Table 5.** Measured and reference sRGB values.

| **Measured Values** | | | | **CCM** | | | | **Reference Values** | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **R** | **G** | **B** | | | | | | **R** | **G** | **B** |
| 0.141 | 0.098 | 0.078 | | ? | ? | ? | | 0.173 | 0.082 | 0.055 |
| 0.482 | 0.349 | 0.294 | × | ? | ? | ? | = | 0.548 | 0.311 | 0.227 |
| 0.188 | 0.231 | 0.326 | | ? | ? | ? | | 0.122 | 0.198 | 0.344 |
| 0.122 | 0.137 | 0.090 | | | | | | 0.094 | 0.151 | 0.053 |
| 0.247 | 0.255 | 0.384 | | | | | | 0.239 | 0.220 | 0.448 |
| 0.322 | 0.494 | 0.478 | | | | | | 0.136 | 0.517 | 0.410 |
| 0.502 | 0.243 | 0.098 | | | | | | 0.680 | 0.212 | 0.021 |
| 0.118 | 0.153 | 0.329 | | | | | | 0.078 | 0.104 | 0.389 |
| 0.376 | 0.157 | 0.141 | | | | | | 0.542 | 0.101 | 0.125 |
| 0.090 | 0.071 | 0.122 | | | | | | 0.111 | 0.041 | 0.151 |
| 0.400 | 0.455 | 0.208 | | | | | | 0.344 | 0.511 | 0.048 |
| 0.580 | 0.369 | 0.137 | | | | | | 0.752 | 0.374 | 0.023 |
| 0.063 | 0.086 | 0.227 | | | | | | 0.036 | 0.043 | 0.311 |
| 0.161 | 0.259 | 0.153 | | | | | | 0.058 | 0.302 | 0.064 |
| 0.255 | 0.086 | 0.063 | | | | | | 0.437 | 0.033 | 0.041 |
| 0.565 | 0.467 | 0.180 | | | | | | 0.805 | 0.580 | 0.010 |
| 0.361 | 0.184 | 0.271 | | | | | | 0.505 | 0.092 | 0.307 |
| 0.149 | 0.275 | 0.396 | | | | | | 0.000 | 0.239 | 0.364 |
| 0.984 | 0.973 | 0.957 | | | | | | 0.899 | 0.899 | 0.891 |
| 0.643 | 0.647 | 0.651 | | | | | | 0.586 | 0.586 | 0.586 |
| 0.388 | 0.392 | 0.396 | | | | | | 0.359 | 0.359 | 0.359 |
| 0.220 | 0.220 | 0.224 | | | | | | 0.198 | 0.198 | 0.194 |
| 0.094 | 0.098 | 0.102 | | | | | | 0.089 | 0.089 | 0.089 |
| 0.039 | 0.039 | 0.039 | | | | | | 0.030 | 0.030 | 0.030 |

A 3 × 3 matrix needs to be found that transforms the measured colours to the reference colours as accurately as possible.

The most common method of correcting colour is the linear method using a 3 × 3 matrix to transform the measured colour to the reference colour [44]. Let R and M denote 3 × N matrices of reference colours and measured colours. N is the number of RGB samples. A 3 × 3 matrix C needs to be found that minimises:

$$\min_{\mathrm{C}} = ||\mathrm{MC} - \mathrm{R}|| \tag{6}$$

An inverse matrix can be used to solve MC = R. Add the inverse of M to each side of the equation

$$\mathrm{M}^{-1}\mathrm{MC} = \mathrm{M}^{-1}\mathrm{R} \tag{7}$$

As any matrix multiplied by its inverse matrix is the identity matrix I.

$$\mathrm{IC} = \mathrm{M}^{-1}\mathrm{R} \tag{8}$$

The matrix C is not changed when multiplied by an identity matrix.

$$\mathrm{C} = \mathrm{M}^{-1}\mathrm{R} \tag{9}$$

However, as the matrix M and R are not square a Moore–Penrose pseudo-inverse matrix method [44,45] is commonly used to find the matrix C. Using the openCV [9] function “inv(DECOMP_SVD)” to find such a matrix (called a colour correction matrix (CCM) hereafter) resulted in the values shown in Table 6.

**Table 6.** CCM calculated using pseudo-inverse.

| | | |
|---|---|---|
| 1.855 | −0.750 | −0.174 |
| −0.252 | 1.567 | −0.382 |
| 0.079 | −0.649 | 1.471 |

A standard PSO search method was used with the exception of 'Outlier Rejection'. Outlier rejection was implemented for each particle by sorting the losses for each colour and eliminating the top losses. The number of outliers can be selected from 0 to 5. The PSO settings used are shown in Table 7.

**Table 7.** PSO settings for the distortion correction task.

| Parameter | Value |
|---|---|
| $NP$ | 50 |
| Velocity Limits | −0.3 to 0.3 |
| Particles Initial Values | Random number ( −2.5, 2.5 ) |
| $c_1$ | 0.90 |
| $c_2$ | 0.50 |
| W | 0.9 |
| Outliers | 0/1 |
| Stopping Criteria | Stop if the loss is not changing for 100 cycles |

Loss function: The normalised and gamma corrected measured sRGB values M were multiplied by particles colour correction matrix C. The resulting corrected colours RGB values were subtracted from the normalised and gamma corrected reference values R. The sum of the square of the absolute differences were used as the loss calculation (Equation (10)).

$$\text{Loss} = ||\text{MC} - \text{R}||^2 \tag{10}$$

The PSO algorithm found a colour correction matrix (Table 8) virtually identical to the pseudo-inverse matrix method when no outliers were allowed.

**Table 8.** CCM found using PSO.

| | | |
|---|---|---|
| 1.856 | −0.751 | −0.174 |
| −0.252 | 1.567 | −0.382 |
| 0.078 | −0.639 | 1.470 |

The total colour reproduction error was calculated which is made up of the sum of the absolute errors of each of the sRGB corrected values compared to the reference sRGB values. Pseudo-inverse matrix produced a consistent error of 310. The PSO algorithm produced errors from 308 to 310. Allowing a single outlier in the PSO algorithm reduced the error to 274, an 11.3% decrease in the colour reproduction error. Figure 13 shows colour examples (using the pseudo-inverse matrix) in groups of three. The top of each group is the uncorrected colour, the middle is the required colour and the bottom is the corrected colour. Three examples are highlighted by a yellow border where the colour matching have noticeably improved with the PSO algorithm.

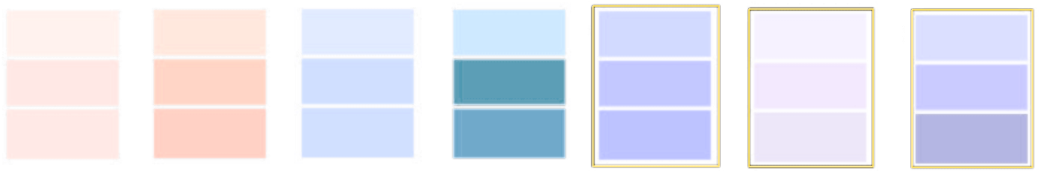


**Figure 13.** Examples of colour comparisons using pseudo-inverse matrix.

The PSO algorithm found a colour correction that gave an equal or better result than the pseudo-inverse matrix 100 times out of 100 times it was run. The number of cycles to perform a search varied between 320 and 680.

## 6. Image Distortion Correction

In the digital microscope used a simplified image distortion correction pipeline is shown in Figure 14.

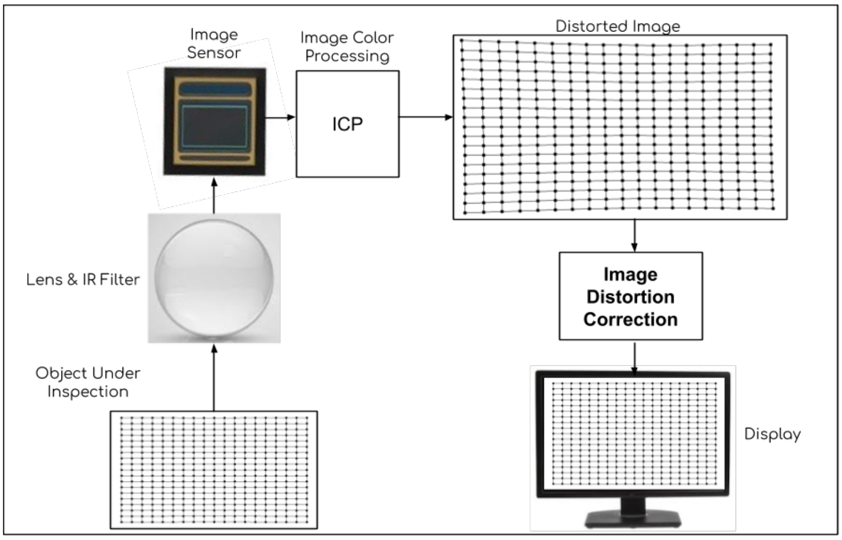


**Figure 14.** Microscope image distortion processing pipeline.

Digital microscopes are commonly used to measure objects being inspected. Typically, the system is calibrated by using a reference ruler, which is placed under the microscope and a reference measurement is made. From that reference measurement, a pixel to mm ratio is determined and this is used to make further measurements.

This method relies on accurate spatial reproduction of the image; however, optics can introduce lens distortion errors due to non-linear magnification of lenses and the alignment of the sensor with the optics. In addition, the optics with the object being inspected can introduce linear affine and non-linear perspective errors. Figure 15 shows an example of a rectangular grid image that has perspective and lens distortions applied. As the two blue lines are the exact same length, it is evident that significant measurement errors occur if a fixed pixel to mm ratio is used to make measurements.

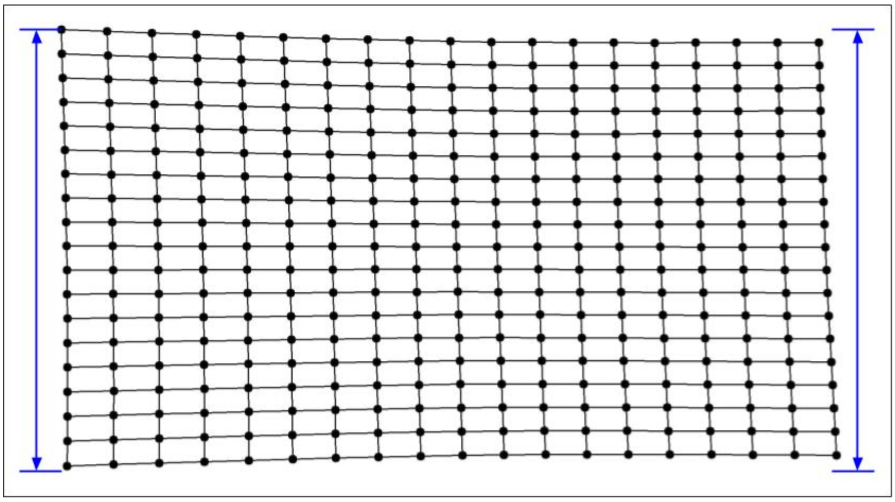


**Figure 15.** Image distortion errors.

In theory, a pinhole camera produces a geometric perfect image of an object; however, in real optical systems lens manufactures have to balance many aberrations resulting in radial image distortion [46]. Radial distortions appear as barrel (Figure 16) or pincushion (Figure 17) distortions.

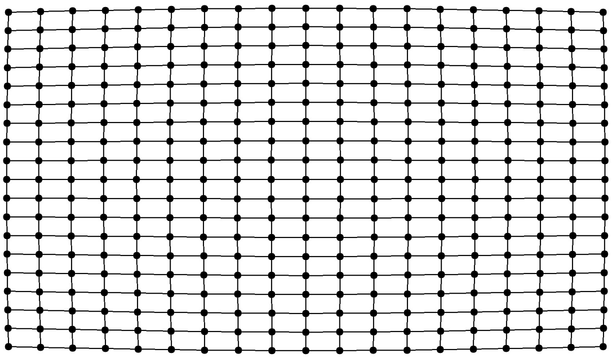


**Figure 16.** Barrel distortion.

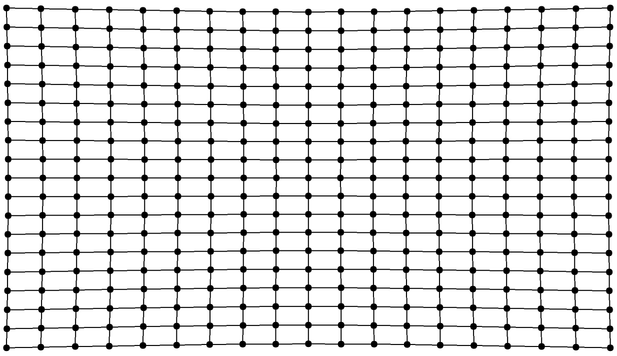


**Figure 17.** Pincushion distortion.

Radial distortion can be modelled using a polynomial equation. For each pixel (x,y) in the distorted image, the correct position of where that pixel should be is given in Equations (17) and (18). $R$ is the radius and $r$ is the normalised distance of a pixel from the optical centre for a 1920 × 1080 image and $k_1$ to $k_n$ are the polynomial coefficients.

The optical centre of the camera may not be the centre of the image due to sensor and lens misalignment. In Figure 18 the red grid is an undistorted grid. The blue grid has barrel distortion and decentring in the x-plane. This is because decentring P1 and P2 are not equidistant from the image centre. If the image centre is used to calculate $r$ in Equations (11)–(14), significant errors would be introduced; therefore, the optical centre position needs to be found.

$$dx = x_{point} - x_{optical-centre} \tag{11}$$

$$dy = y_{point} - y_{optical-centre} \tag{12}$$

$$R = \sqrt{dx^2 + dy^2} \tag{13}$$

$$r = R/1100.1 \tag{14}$$

$$x_{scale} = 1 + k_1 r^2 + k_2 r^4 + k_3 r^6 ... \tag{15}$$

$$y_{scale} = 1 + k_1 r^2 + k_2 r^4 + k_3 r^6 ... \tag{16}$$

$$x_{\text{corrected}} = x_{\text{optical-centre}} + (x - x_{\text{optical-centre}}) \cdot \text{scale} \tag{17}$$

$$y_{\text{corrected}} = y_{\text{optical-centre}} + (y - y_{\text{optical-centre}}) \cdot \text{scale} \tag{18}$$

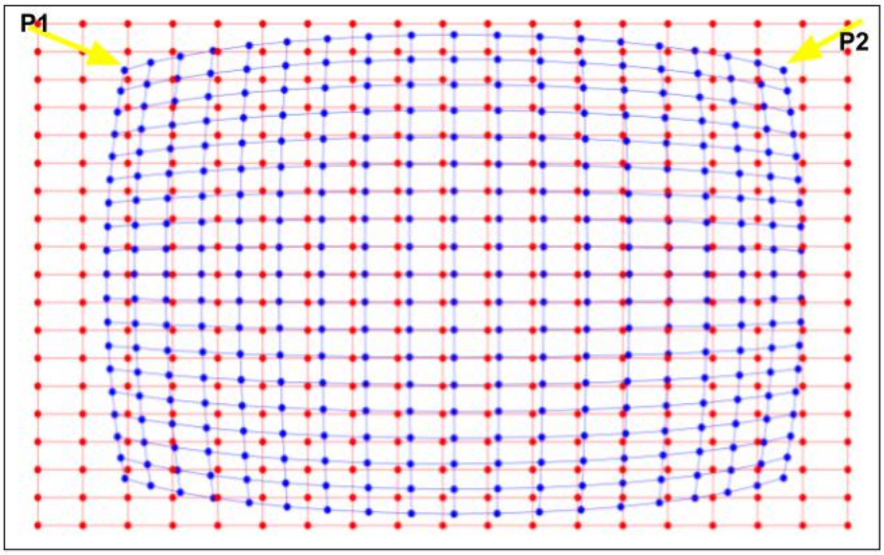
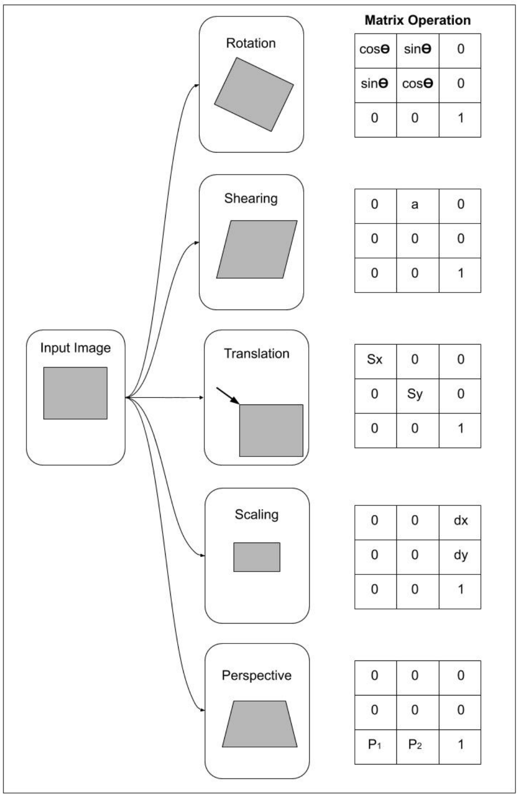


**Figure 18.** Optical decentring.

### *6.1. Affine and Perspective Distortion*

Affine (linear) and perspective (non-linear) distortions are shown in Figure 19, including a 3 × 3 matrix that can be used to carry out that transformation. The new location (x′, y′) of each pixel is attained by multiplying the matrix by the current location of each pixel. Equation (19) depicts a pixel rotation operation. A 1 is added to the (x, y) and (x′, y′) pixel locations to allow matrix multiplication.

$$\begin{bmatrix} x^{'} \\ y^{'} \\ 1 \end{bmatrix} = \begin{bmatrix} cos\theta & sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{19}$$



Matrix Operation

| | | |
|---|---|---|
| cosΘ | sinΘ | 0 |
| sinΘ | cosΘ | 0 |
| 0 | 0 | 1 |

| | | |
|---|---|---|
| 0 | a | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 1 |

| | | |
|---|---|---|
| Sx | 0 | 0 |
| 0 | Sy | 0 |
| 0 | 0 | 1 |

| | | |
|---|---|---|
| 0 | 0 | dx |
| 0 | 0 | dy |
| 0 | 0 | 1 |

| | | |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| $P_1$ | $P_2$ | 1 |

**Figure 19.** Affine and perspective distortions.

Homography Matrix

All of these matrix operations can be combined into a single 3 × 3 matrix known as a homography matrix Equation (20). This single matrix with 8 degrees of freedom can fix all the following distortions:

- Rotation.
- Shearing.
- Translation.
- Scaling.
- Perspective.

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} * \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{20}$$

### *6.2. Problem Definition*

Figure 20 demonstrates the essence of the challenge. A reference grid chart is placed under the microscope and an image taken with no image distortion correction applied.

A method of correcting Figure 20 (left) to produce a corrected image (Figure 20 (right)) is required.

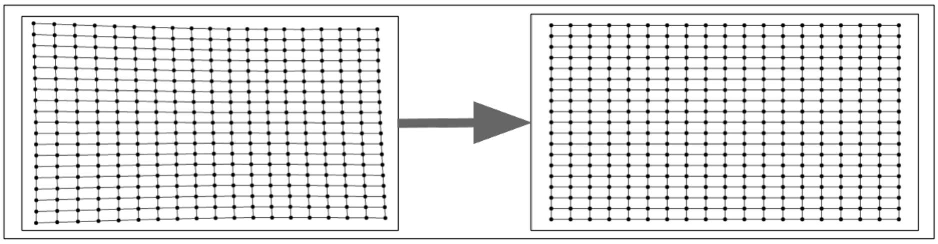


**Figure 20.** Distortion correction.

### *6.3. The Method*

A known reference grid is syntactically generated or imaged by the microscope and the distorted image points are detected and their position saved.

A combination of radial lens distortion correction with optical centre compensation and a homography matrix that map the detected points to the known point positions is searched for. A flowchart of the search is depicted in Figure 21.

A standard PSO search method was used with the settings reported in Table 9.

**Table 9.** PSO settings for the distortion correction task. Ranges per component of this search space **D** are reported in Table 10.

| Parameter | Value |
|---|---|
| Particles | 50 |
| Velocity Limits | clamped at 20% of **D** |
| C1 | 2.00 |
| C2 | 0.2 |
| W | 0.9 |
| Stopping Criteria | Stop after 25,000 cycles |

Loss function: The square of the differences of the corrected detected points and the reference points were used as the loss calculation Equation (21):

$$Loss = \sum_{i=1}^{n} \Big( (x_i, y_i)_{predicted} - (x_i, y_i)_{real} \Big)^2 \tag{21}$$

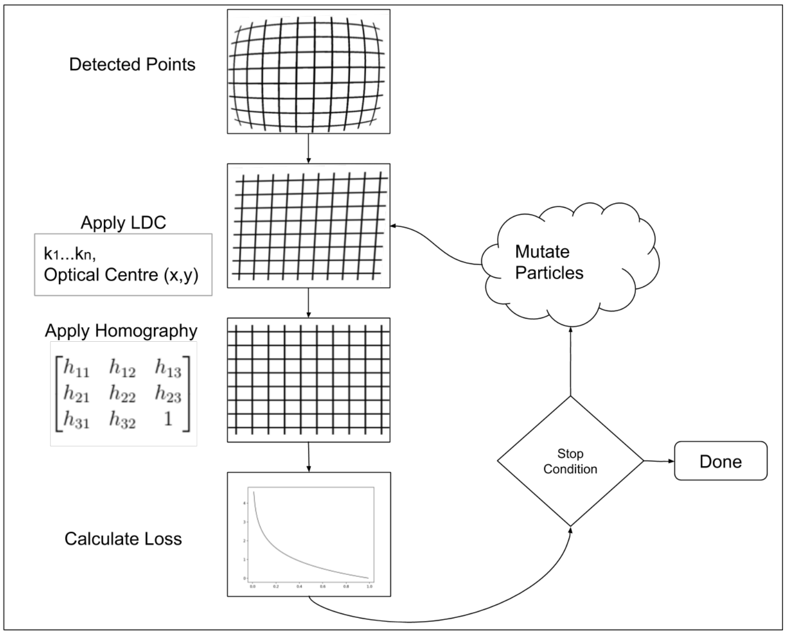


**Figure 21.** Distortion correction flowchart.

### 6.3.1. Particle Initial Values

The range of initialisation values for the homography matrix ($h_{row,col}$), polynomial constants ($k_1$ to $k_4$) and the optical centre ($O_x, O_y$) are given in Table 10.

**Table 10.** Search space for the distortion correction task.

| Component | Lower Bound | Upper Bound |
|---|---|---|
| $h_{11}$ | 0.9 | 1.1 |
| $h_{12}$ | −0.3 | 0.3 |
| $h_{13}$ | −20 | +20 |
| $h_{21}$ | −0.3 | 0.3 |
| $h_{22}$ | 0.9 | 1.1 |
| $h_{23}$ | −20 | +20 |
| $h_{31}$ | −0.000005 | +0.000005 |
| $h_{32}$ | −0.000005 | +0.000005 |
| $h_{33}$ | 0.95 | 1.05 |
| $K_1$ | −0.05 | +0.05 |
| $K_2$ | −0.0005 | +0.0005 |
| $K_3$ | −0.00005 | +0.00005 |
| $K_4$ | −0.000005 | +0.000005 |
| $O_x$ | −5 | +5 |
| $O_y$ | −5 | +5 |

### 6.3.2. Velocity Limits

The velocity limits are clamped at 20% of the ranges given in Table 10.

### 6.3.3. Bounds Checking

No bounds checking of the position of the particles is performed as any solution that provides a good transform between distorted points and corrected points can be implemented.

## *6.4. Results*

### 6.4.1. Synthetic Data Results

A synthetic chart, shown in Figure 22, with 19 × 19 dots, is generated using random perspective, user variable lens distortion 0.3 and optical centre (4, −5).

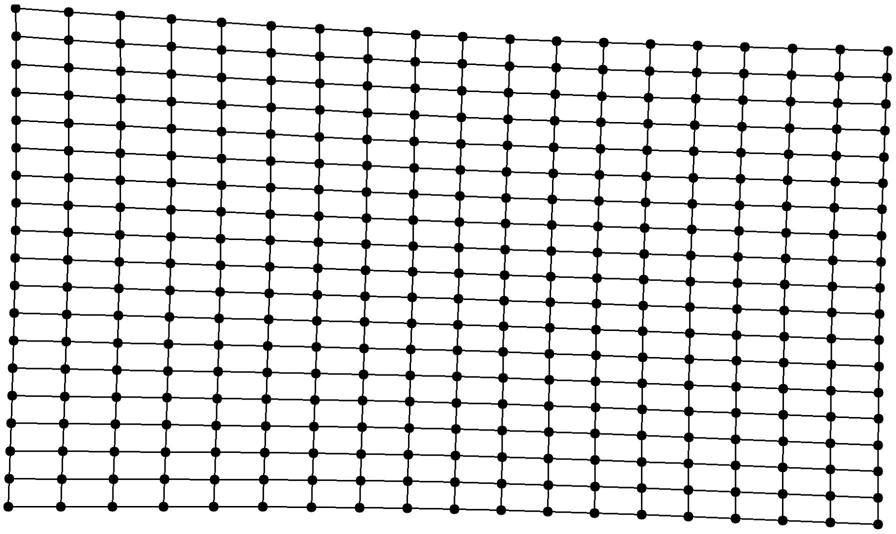


**Figure 22.** Generated synthetic image example.

The PSO algorithm and CMA-ES [35] algorithm were used to find solutions to map the distorted points to undistorted points.

Both CMA-ES and PSO algorithms give similar results of max pixel error from 0.22 to 0.28 and average pixel error from 0.09 to 0.08.

### 6.4.2. Real Data Results

A calibration chart, shown in Figure 23, with 19 × 19 dots was manually placed under the camera and positioned in the approximate centre of the field of view. The dots were detected with a blob detector and their centre recorded as per Figure 24.

The PSO algorithm and CMA-ES [35] algorithm were used to find solutions to map the distorted points to undistorted points. A library was used to implement the CMA-ES function [47]. The results are shown in Tables 11 and 12, respectively.

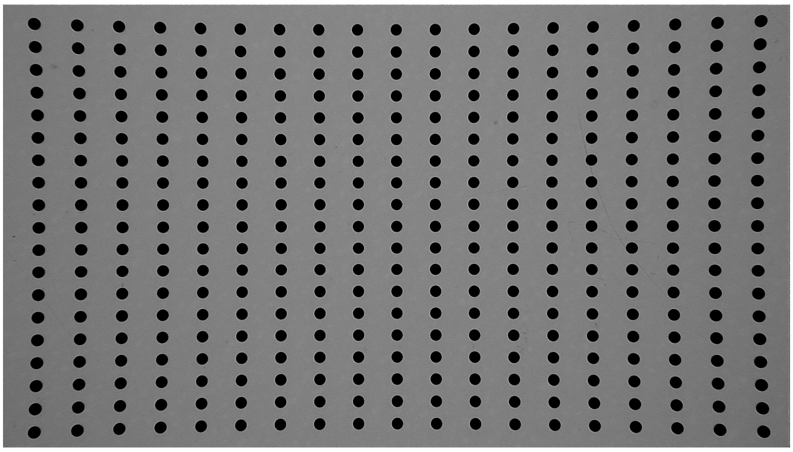


**Figure 23.** Cal chart.

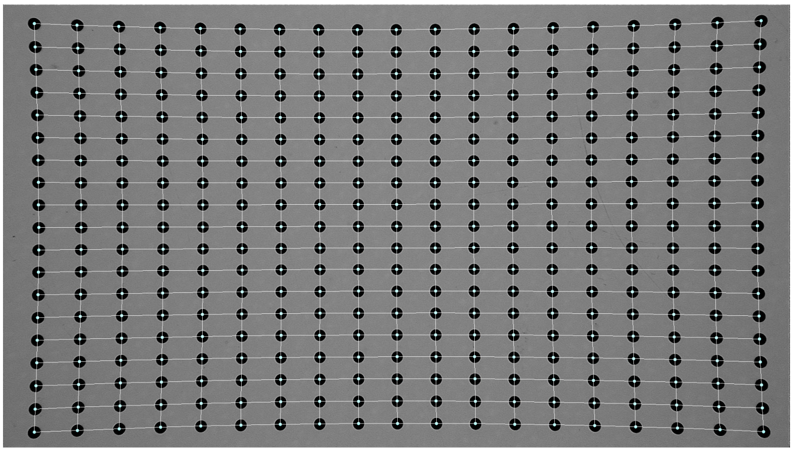


**Figure 24.** Detected points.

**Table 11.** PSO real results.

| Run | Loss | Max Pix Error | Avg Pix Error |
|---|---|---|---|
| 1 | 186.7 | 1.55 | 0.62 |
| 2 | 187.6 | 1.67 | 0.62 |
| 3 | 187.2 | 1.64 | 0.62 |
| 4 | 209.9 | 1.87 | 0.66 |
| 5 | 212.9 | 1.88 | 0.67 |
| 6 | 250.8 | 2.04 | 0.72 |
| 7 | 209.4 | 1.83 | 0.66 |
| 8 | 229.1 | 2.02 | 0.69 |
| 9 | 187.9 | 1.70 | 0.62 |
| 10 | 224.6 | 1.91 | 0.68 |

**Table 12.** CMA-ES real results.

| Run | Loss | Max Pix Error | Avg Pix Error |
|---|---|---|---|
| 1 | 168.2 | 1.67 | 0.59 |
| 2 | 168.3 | 1.67 | 0.59 |
| 3 | 167.8 | 1.67 | 0.59 |
| 4 | 167.8 | 1.67 | 0.59 |
| 5 | 168.2 | 1.67 | 0.59 |
| 6 | 168.3 | 1.67 | 0.59 |
| 7 | 168.0 | 1.67 | 0.59 |
| 8 | 168.8 | 1.67 | 0.59 |
| 9 | 167.8 | 1.67 | 0.59 |
| 10 | 167.8 | 1.67 | 0.59 |

In total, 12 images were taken with a reference ruler in different positions and points 10 mm apart were selected manually. Figure 25 shows these images and the reference 10 mm measurements.

Referring to Table 13, the points at the start and end of each 10 mm measurement are shown, with $P1_x$, $P1_y$ being one point and $P2_x$, $P2_y$ being the second point. The real number of pixels between the points is shown in the $R_{pix}$ column.

Each point is transformed using the solutions found and the predicted number of pixels between the transformed points is given in the $P_{pix}$ column. The average of the predicted number of pixels is divided by 12 to give a pixel to mm ratio. This ratio is multiplied by the predicted number of pixels to give a predicted length shown in column $P_{len}$. The real length, column $R_{len}$ is calculated the same way.

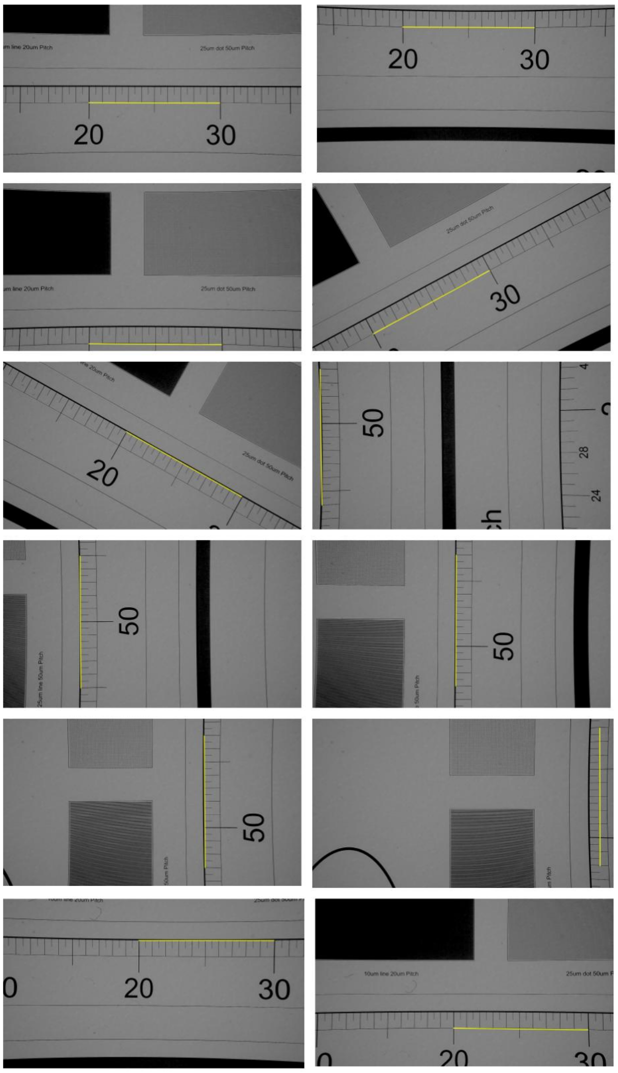


**Figure 25.** Twelve sample images, including a reference ruler, used for evaluating the image distortion algorithms.

**Table 13.** *Resultsonrealim*age using CMA-ES and PSO. The first column (#) reports the measurement number (see Figure 25) while raw data are reported in the second column as reference.

| # | Raw Data | | | | | CMA-ES | | | PSO | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | **P1$_x$** | **P1$_y$** | **P2$_x$** | **P2$_y$** | **R$_{\text{pix}}$** | **P$_{\text{pix}}$** | **R$_{\text{len}}$** | **P$_{\text{len}}$** | **P$_{\text{pix}}$** | **R$_{\text{len}}$** | **P$_{\text{len}}$** |
| 1 | 550 | 631 | 1395 | 633 | 845.0 | 857.5 | 9.850 | 10.008 | 856.3 | 9.850 | 9.994 |
| 2 | 552 | 144 | 1402 | 148 | 850.0 | 857.3 | 9.909 | 10.006 | 857.0 | 9.909 | 10.002 |
| 3 | 551 | 1034 | 1407 | 1039 | 856.0 | 856.1 | 9.979 | 9.992 | 856.3 | 9.979 | 9.995 |
| 4 | 390 | 971 | 1141 | 562 | 855.2 | 856.6 | 9.969 | 9.998 | 857.5 | 9.969 | 10.009 |
| 5 | 792 | 451 | 1537 | 870 | 854.7 | 856.7 | 9.964 | 9.999 | 857.4 | 9.964 | 10.007 |
| 6 | 61 | 925 | 49 | 48 | 877.1 | 857.3 | 10.224 | 10.006 | 877.1 | 10.224 | 10.007 |
| 7 | 500 | 954 | 495 | 102 | 852.0 | 856.9 | 9.932 | 10.002 | 857.2 | 9.932 | 10.005 |
| 8 | 922 | 940 | 926 | 96 | 844.0 | 857.3 | 9.839 | 10.007 | 856.4 | 9.839 | 9.996 |
| 9 | 1300 | 961 | 1294 | 111 | 850.0 | 857.1 | 9.909 | 10.004 | 856.8 | 9.909 | 10.001 |
| 10 | 1834 | 939 | 1832 | 60 | 879.0 | 856.2 | 10.247 | 9.993 | 856.2 | 10.247 | 9.993 |
| 11 | 863 | 266 | 1726 | 266 | 863.0 | 856.4 | 10.060 | 9.995 | 856.5 | 10.060 | 9.997 |
| 12 | 888 | 833 | 1756 | 844 | 868.1 | 856.2 | 10.119 | 9.993 | 856.2 | 10.119 | 9.994 |

The resolution in the horizontal direction is 10 mm ÷ 1920 = 5.2 um and in the vertical direction is 10 mm ÷ 1080 = 9.26 um. When no image distortion correction is used, a maximum measurement error of 247 um (Table 13) is found. Using the best CMA-ES solution, the maximum error is 8 um.

The PSO Search was run 10 times and the maximum error logged for each run. Results are shown in Table 14. This gives a standard deviation $\sigma$ of 2.24, a mean of 8.4 and a variance of 5.04.

**Table 14.** PSO real results over 10 searches.

| Run | Max Pix Error (um) |
|---|---|
| 1 | 7 |
| 2 | 7 |
| 3 | 8 |
| 4 | 9 |
| 5 | 6 |
| 6 | 12 |
| 7 | 6 |
| 8 | 13 |
| 9 | 8 |
| 10 | 8 |

## 7. Conclusions

We have shown some software components of a digital inspection microscope for testing medical devices. The underlying theme is the use of PSO as an algorithm in many different tasks. We have shown that such a heuristic algorithm provides good results for all of them.

The first task sees the use of the PSO algorithm to improve upon the original calibration system for the height from focus. Originally this was performed with a poor measurement method, resulting in noisy data and required shipping the microscope back to factory. In the new approach, the PSO is used to optimally fit a polynomial to these noisy data automatically, thus creating a smoother calibrating model. This method has been shown to significantly increase the accuracy of the system. It is important to take sufficient samples for this method to work. When trying to replace PSO with more complex algorithms, i.e., the CMA-ES, we noted only a marginal difference in the results, which is not significant enough to use this computationally more expensive algorithm. The method used shows the viability of calibrating a system for this and other measurements prior to having optimised measurement methods, and releasing those systems to customers and correcting the noisy calibration data once the measurements are optimised.

The second important task is the so-called matching of images for image stitching. Here, we improve a conventional template matching algorithm using PSO. Results show that the newly proposed approach is most accurate when the object has a low variance in features as this results in an ideal search space. When there are many features, the search space results in a very small area. If no particle explores this area, then the matching fails; however, when there are many features, standard interest point matching can be used. The PSO matching time takes approximately 365 ms and the matching function was 100% accurate on every run of experiments 1 and 2 in Section 4.3 with maximum dropout (25%) and noise (10%). With PSO-supported 2D point cloud matching, we introduce a method to deal with cases where neither feature point matching nor template matching can be applied. This works by adding laser points to the scene and shows a lot of promise. Using PSO provides a robust solution and creating interest points using distance to other points results in the potential for real-time matching.

Thirdly, the problem of colour correction is tackled by using a PSO algorithm with no outlier rejection. This produces a colour correction matrix as good as, or slightly better than, the standard pseudo-inverse matrix method. Adding outlier rejection decreased the error by 11.3%, visibly improving the colour reproduction. Hence, the PSO algorithm is reliable in this application and beats conventional algorithms.

Fourthly, image distortions are corrected through PSO and CMA-ES. Here, the CMA-ES search seems to be more consistent across multiple runs than the PSO search, which also gave satisfactory results. Indeed, the PSO search for a solution to image distortion correction has been shown to increase measurement accuracy from 247 µm to average accuracy of 8.4 µm. As this calibration is performed once per system, a number of runs could be made and the lowest error homography matrix and polynomial constants picked, reaching the quality of CMA-ES.

Overall, there are many potential uses of PSO in this application. We have demonstrated four of them. In all cases, the PSO-based algorithm can either reach the quality of the current best approach or beat it. In some cases, it also offers a speed advantage. In this way, we have demonstrated the applicability of PSO in an industrial setting.

**Author Contributions:** Conceptualization, L.R., F.C., S.K. and S.C.-D.; methodology, L.R.; software, L.R.; validation, L.R.; formal analysis, L.R.; investigation, L.R.; resources, L.R.; data curation, L.R.; writing—original draft preparation, L.R., F.C. and S.K.; writing—review and editing, L.R., F.C., S.K. and S.C.-D.; visualization, L.R.; supervision, F.C. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Not applicable.

**Acknowledgments:** The authors would like to thank Ash Technologies Ltd. for providing the inspection system and relevant related information.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Eiben, A.E.; Smith, J.E. *Introduction to Evolutionary Computation*; Springer: Berlin, Germany, 2003. [CrossRef]
2. Caraffini, F.; Santucci, V.; Milani, A. (Eds.) *Evolutionary Computation & Swarm Intelligence*; MDPI: Basel, Switzerland, 2020.
3. Miettinen, K. (Ed.) *Evolutionary Algorithms in Engineering and Computer Science: Recent Advances in Genetic Algorithms, Evolution Strategies, Evolutionary Programming, GE*; John Wiley & Sons, Inc.: New York, NY, USA, 1999.
4. Oduguwa, V.; Tiwari, A.; Roy, R. Evolutionary computing in manufacturing industry: An overview of recent applications. *Appl. Soft Comput.* **2005**, *5*, 281–299. [CrossRef]
5. Kumar, K.; Zindani, D.; Davim, J.P. *Optimizing Engineering Problems through Heuristic Techniques*; CRC Press: Boca Raton, FL, USA, 2019.
6. Caraffini, F.; Chiclana, F.; Moodley, R.; Gongora, M.; Gupta, D.; Kose, U.; Castillo, O.; Al-Turjman, F. Applications of computational intelligence-based systems for societal enhancement. *Int. J. Intell. Syst.* **2022**, *37*, 2679–2682. [CrossRef]
7. Wolpert, D.H.; Macready, W.G. No Free Lunch Theorems for Optimization. *Trans. Evol. Comp* **1997**, *1*, 67–82. [CrossRef]
8. Ash Technologies. Omni-3, Digital Microscope and Measurement System. Available online: https://www.ashvision.com/omni-3/ (accessed on 1 July 2022).
9. Bradski, G. The OpenCV Library. *Dr. Dobb's J. Softw. Tools Prof. Program.* **2000**, *25*, 120–125.
10. The Qt Company. Qt, Cross-Platform Software Development for Embedded & Desktop. Available online: https://www.qt.io/ (accessed on 6 August 2020).
11. Catalucci, S.; Thompson, A.; Piano, S.; Branson, D.T.; Leach, R. Optical metrology for digital manufacturing: A review. *Int. J. Adv. Manuf. Technol.* **2022**, *120*, 4271–4290. [CrossRef]
12. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the IEEE International Conference on Neural Networks, Perth, Australia, 27 November–1 December 1995; Volume 4, pp. 1942–1948. [CrossRef]
13. Okwu, M.O.; Tartibu, L.K. Particle Swarm Optimisation. In *Metaheuristic Optimization: Nature-Inspired Algorithms Swarm and Computational Intelligence, Theory and Applications*; Springer International Publishing: Cham, Switzerland, 2021; pp. 5–13. [CrossRef]
14. Shi, Y.; Eberhart, R. A modified particle swarm optimizer. In Proceedings of the 1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence, Anchorage, AK, USA, 4–9 May 1998; pp. 69–73. [CrossRef]
15. Poli, R.; Kennedy, J.; Blackwell, T. Particle swarm optimization. *Swarm Intell.* **2007**, *1*, 33–57. [CrossRef]

16. Zheng, Y.l.; Ma, L.h.; Zhang, L.y.; Qian, J.x. Empirical study of particle swarm optimizer with an increasing inertia weight. In Proceedings of the 2003 Congress on Evolutionary Computation, 2003. CEC '03, Canberra, Australia, 8–12 December 2003; Volume 1, pp. 221–226.
17. Bansal, J.C.; Singh, P.K.; Saraswat, M.; Verma, A.; Jadon, S.S.; Abraham, A. Inertia Weight strategies in Particle Swarm Optimization. In Proceedings of the 2011 Third World Congress on Nature and Biologically Inspired Computing, Salamanca, Spain, 19–21 October 2011; pp. 633–640. [CrossRef]
18. Ratnaweera, A.; Halgamuge, S.; Watson, H. Particle swarm optimiser with time varying acceleration coefficients. In Proceedings of the International Conference on Soft Computing and Intelligent Systems, Tsukuba, Japan, 21–25 October 2002; pp. 240–255.
19. Ratnaweera, A.; Halgamuge, S.K.; Watson, H.C. Self-Organizing Hierarchical Particle Swarm Optimizer With Time-Varying Acceleration Coefficients. *IEEE Trans. Evol. Comput.* **2004**, *8*, 240–255. [CrossRef]
20. Tripathi, P.K.; Bandyopadhyay, S.; Pal, S.K. Multi-objective particle swarm optimization with time variant inertia and acceleration coefficients. *Inf. Sci.* **2007**, *177*, 5033–5049. [CrossRef]
21. Engelbrecht, A. Particle swarm optimization: Velocity initialization. In Proceedings of the 2012 IEEE Congress on Evolutionary Computation, Brisbane, Australia, 10–15 June 2012; pp. 1–8. [CrossRef]
22. Shahzad, F.; Masood, S.; Khan, N. Probabilistic opposition-based particle swarm optimization with velocity clamping. *Knowl. Inf. Syst.* **2014**, *39*, 703–737. [CrossRef]
23. Clerc, M.; Kennedy, J. The particle swarm-explosion, stability, and convergence in a multidimensional complex space. *IEEE Trans. Evol. Comput.* **2002**, *6*, 58–73. [CrossRef]
24. Xu, S.; Rahmat-Samii, Y. Boundary Conditions in Particle Swarm Optimization Revisited. *IEEE Trans. Antennas Propag.* **2007**, *55*, 760–765. [CrossRef]
25. Kulkarni, M.N.K.; Patekar, M.S.; Bhoskar, M.T.; Kulkarni, M.O.; Kakandikar, G.; Nandedkar, V. Particle Swarm Optimization Applications to Mechanical Engineering—A Review. *Mater. Today Proc.* **2015**, *2*, 2631–2639. [CrossRef]
26. Li, G.; Li, Y.; Chen, H.; Deng, W. Fractional-Order Controller for Course-Keeping of Underactuated Surface Vessels Based on Frequency Domain Specification and Improved Particle Swarm Optimization Algorithm. *Appl. Sci.* **2022**, *12*, 3139. [CrossRef]
27. Eberhart; Shi, Y. Particle swarm optimization: Developments, applications and resources. In Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No.01TH8546), Seoul, Korea, 27–30 May 2001; Volume 1, pp. 81–86. [CrossRef]
28. Gad, A.G. Particle Swarm Optimization Algorithm and Its Applications: A Systematic Review. *Arch. Comput. Methods Eng.* **2022**, *29*, 2531–2561. [CrossRef]
29. Ma, K.; Ren, C.; Zhang, Y.; Chen, Y.; Chen, Y.; Zhou, P. A New Vibration-Absorbing Wheel Structure with Time-Delay Feedback Control for Reducing Vehicle Vibration. *Appl. Sci.* **2022**, *12*, 3157. [CrossRef]
30. Guo, Y.; Li, W.; Mileham, A.; Owen, G. Applications of particle swarm optimisation in integrated process planning and scheduling. *Robot. Comput.-Integr. Manuf.* **2009**, *25*, 280–288. [CrossRef]
31. Freeman, M.; Freeman, M.; Hull, C.; Charman, W. *Optics*; Butterworth-Heinemann: Oxford, UK, 2003; p. 92.
32. Murawski, K. Method of Measuring the Distance to an Object Based on One Shot Obtained from a Motionless Camera with a Fixed-Focus Lens. *Acta Phys. Pol. A* **2015**, *127*, 1591–1596. [CrossRef]
33. Zhang, X.; Liu, Z.; Jiang, M.; Chang, M. Fast and accurate auto-focusing algorithm based on the combination of depth from focus and improved depth from defocus. *Opt. Express* **2014**, *22*, 31237. [CrossRef]
34. Laakso, T.I.; Tarczynski, A.; Paul Murphy, N.; Välimäki, V. Polynomial filtering approach to reconstruction and noise reduction of nonuniformly sampled signals. *Signal Process.* **2000**, *80*, 567–575. [CrossRef]
35. Hansen, N.; Ostermeier, A. Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In Proceedings of the IEEE International Conference on Evolutionary Computation, Nagoya, Japan, 20–22 May 1996; pp. 312–317.
36. Lowe, D.G. Object recognition from local scale-invariant features. In Proceedings of the Seventh IEEE International Conference on Computer Vision, Kerkyra, Greece, 20–27 September 1999; Volume 2, pp. 1150–1157. [CrossRef]
37. Bay, H.; Tuytelaars, T.; Van Gool, L. SURF: Speeded Up Robust Features. In Proceedings of the Computer Vision—ECCV 2006, Graz, Austria, 7–13 May 2006; pp. 404–417.
38. Fischler, M.A.; Bolles, R.C. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Commun. ACM* **1981**, *24*, 381–395. [CrossRef]
39. Bianco, S.; Bruna, A.R.; Naccari, F.; Schettini, R. Color correction pipeline optimization for digital cameras. *J. Electron. Imaging* **2013**, *22*, 023014. [CrossRef]
40. Anderson, M.; Motta, R.; Chandrasekar, S.; Stokes, M. Proposal for a standard default color space for the internet-srgb. In Proceedings of the 4th Color Imaging Conference. Society for Imaging Science and Technology, Scottsdale, AZ, USA, 19–22 November 1996; pp. 238–245.
41. Finlayson, G.D.; Mackiewicz, M.; Hurlbert, A. Color correction using root-polynomial regression. *IEEE Trans. Image Process.* **2015**, *24*, 1460–1470. [CrossRef] [PubMed]
42. X Rite. ColorChecker Classic: X-Rite Photo & Video. Available online: https://xritephoto.com/colorchecker-classic (accessed on 1 October 2020).
43. X Rite. Color Management Products, Tools, Solutions: X-Rite Photo & Video. Available online: https://xritephoto.com/ph_product_overview.aspx?ID=820 (accessed on 1 April 2019).

44. Fang, F.; Gong, H.; Mackiewicz, M.; Finlayson, G. Colour Correction Toolbox. Available online: https://ueaeprints.uea.ac.uk/id/eprint/65098/4/Colour_Correction_Toolbox.pdf (accessed on 15 June 2022).
45. Dresden, A. The fourteenth western meeting of the American Mathematical Society. *Bull. Amer. Math. Soc.* **1920**, *26*, 385–396. [CrossRef]
46. Mallon, J.; Whelan, P. Precise radial un-distortion of images. In Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004, Cambridge, UK, 26 August 2004; Volume 1, pp. 18–21. [CrossRef]
47. Fabisch, A. CMA-ESpp. Available online: https://github.com/AlexanderFabisch/CMA-ESpp (accessed on 6 August 2020).