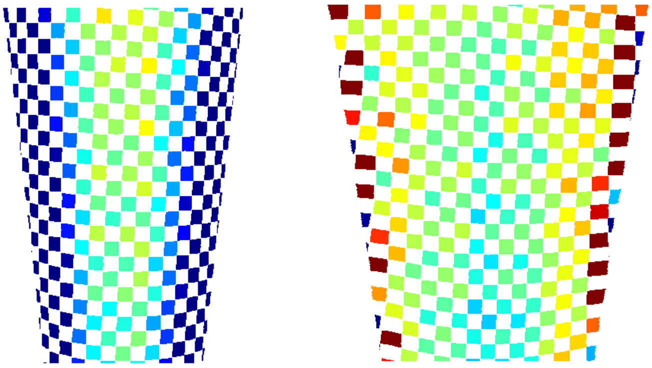
AUTOMATED SKIN PRICK TEST READING USING DIGITAL PHOTOGRAPHY

An optimized correction for the camera perspective and arm curvature for accurate wheal size measurement

MASTER THESIS

Anne-Marlein van den Berg

February 2024



Graduation committee:

Prof. dr. J.J. Fütterer	Chairman | University of Twente
Drs. D.M.W. Gorissen	Medical supervisor | Deventer Hospital
Dr. F.J. Siepel	Technical supervisor | University of Twente
B.J.C.C. Sweep MSc.	Process supervisor | University of Twente
T. Boers MSc.	External member| University of Twente

UNIVERSITY OF TWENTE.

University of Twente – Technical Medicine – Medical Imaging and Interventions

Deventer ziekenhuis

Deventer Hospital – Pediatrics department and Children Allergy Center

Abstract

BACKGROUND

A common test for diagnosing allergies is the skin prick test (SPT), which is widely used as it is minimally invasive, fast and low in costs. Reading the test results is performed by manually measuring wheal sizes, however, a major drawback is the inter- and intra-observer variability of the test, leading to inaccurate wheal measurements and lack of reproducibility, indicating there is need for a standardized and reliable SPT reading method. This study focuses on automated SPT reading using digital photography. A new method using a basic placemat with ArUco markers is suggested to optimize a correction for the camera perspective and arm curvature in the images, which is needed to obtain accurate wheal sizes that correspond to the real-world dimensions of the wheals.

METHODS

The perspective correction was performed using the ArUco markers for a projective image transformation. Its accuracy was tested on photographs taken from different camera angles and validated with a checkerboard as reference object. The curvature correction was performed by recovering the three-dimensional shape of the forearm to a two-dimensional flat surface via image transformation. The performance was evaluated with a checkerboard on six subjects and for different regions on the arm, and compared to the conventional barcode sticker method. For insight in clinical relevance, the location of wheals on the forearm regions was analyzed in 88 SPT photographs and related to the placemat performance.

RESULTS

The perspective correction showed a high accuracy for all tested camera angles; a maximum deviation of 1% was measured for 10×10 mm checkerboard squares. The curvature correction using the placemat showed for the inner region of the forearm an acceptable error below 10% of 25 mm^2 sized squares; the error increased towards the outer sides of the arm. It was found that most wheals in the SPT appear in the inner regions. The placemat method performed better than the barcode method.

CONCLUSIONS

This work showed that the correction for camera perspective and arm curvature in photographs of the SPT can be optimized using a basic placemat with ArUco markers. This provides more accurate wheal measurements and therefore contributes to obtaining reproducible and reliable SPT results. The performance of the curvature correction could be further improved and evaluation of the placemat on more subjects with varied ages and arms' shapes is needed.

KEYWORDS Skin prick test, digital photography, camera perspective, arm curvature, image transformation

Contents

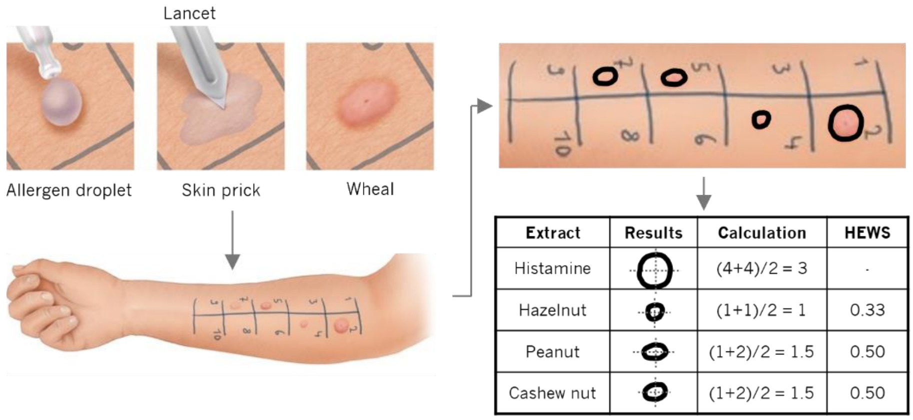
List of abbreviations

2D	Two-dimensional
3D	Three-dimensional
HEWS	Histamine equivalent wheal size
IgE	Immunoglobulin E
IQR	Interquartile range
mm	Millimeter
mm^2	Square millimeter
SPT	Skin prick test

CHAPTER 1. General introduction

In recent decades, allergic diseases have increased remarkably throughout the world and this trend is still on a rise, particularly in developed countries the prevalence rate is high.[1–4] Having an allergy can impact a person's daily life seriously, as it is known that individuals with an allergy may experience a lower quality of life and more mental health problems.[2,3,5–7] Besides the effects on physical and mental health, the medical costs of allergy health care impose a significant individual and public economic burden.[2,3,8] Common allergic diseases include allergic asthma, allergic rhinoconjunctivitis, and food allergies. Symptoms range from mild to severe and depend on the type of allergy, for example, inhalant allergies can cause sneezing, conjunctivitis, rhinorrhea, nasal obstruction or dyspnea. Manifestations of a food allergy are urticaria, angio-oedema, reactions of the intestinal and respiratory tract, or even anaphylaxis.[9–11] These are type I hypersensitivity reactions and known for their immediate immune response. If a person is sensitized for an allergen, specific immunoglobulin E (IgE) antibodies are produced, which are bound to high-affinity Fcε receptors on mast cells located in the skin and mucosal tissues.[12] When intruding the skin, airways or intestinal system, the allergen cross-links with the mast cell bound IgE after which degranulation of the mast cell occurs, resulting in the release of histamine and other mediators causing clinical symptoms.[9,11,13]

The diagnosis of allergic diseases is mainly based on the patient's medical history and can be confirmed with an additional sensitization test.[11,14–16] One of the most performed diagnostic tests for IgE-mediated allergies is the skin prick test (SPT).[17,18] Drops of allergen extracts and a positive control (histamine) as well as negative control drop (saline) are placed on the volar surface of the forearm or back. The droplets are pricked with a sterile lancet into the skin and results are read after 15 minutes. Sensitization is indicated by the appearance of a swollen and itchy area, called a *wheal*, surrounded by erythema. The outcome of the SPT is determined by measuring wheal sizes.[17] In current practice, the wheal contour is manually outlined on the skin with a marker after which adhesive tape is used to transfer the ink to a paper sheet. An average diameter can then be estimated by the sum of the largest wheal diameter and its perpendicular diameter divided by two. Additionally, the histamine equivalent wheal size (HEWS) is calculated which is the ratio of the average diameter of the allergen wheal and positive control wheal. The outcome is interpreted as positive when the average wheal diameter is > 3 mm or HEWS > 0.4.[17,19] The steps of the SPT procedure are depicted in Figure 1. The SPT is widely used since it is minimally invasive, fast, convenient, and low in costs.[18,20,21] However, a major drawback is the inter- and intra-observer variability of the test.[19,22,23] Precise manual marking of wheal contours is difficult and subjective, leading to a lack of repeatability.[24,25] The same applies for the manual measurement of wheal diameters. Also, calculating wheal sizes based on an ellipsoid shape is not an accurate outcome measure for the irregular shape of wheals.[19] Moreover, within the field of research, the different interpretations make comparison of study results difficult.[18] Inter-institutional comparison of a patient's test result is challenging as well. Because of these inaccuracies and lack of repeatability, there is need for a standardized and reliable SPT reading method.



Extract	Results	Calculation	HEWS
Histamine		(4+4)/2 = 3	·
Hazelnut		(1+1)/2 = 1	0.33
Peanut		(1+2)/2 = 1.5	0.50
Cashew nut		(1+2)/2 = 1.5	0.50

Figure 1 SPT procedure: the allergen droplets are placed on the forearm and pricked into the skin with a sterile lancet. In case of sensitization, a wheal appears, of which its contour is manually outlined with a pen and transferred to a paper sheet. The size of the wheal is then measured by calculating the mean diameter and HEWS.[27]

In literature, several methods are proposed that attempt to automate reading of the SPT. A modern technique that has mainly been researched in the last few years involves the use of three-dimensional (3D) imaging and reconstruction. Dos Santos et al. were the first who introduced 3D image technology in the field of allergy.[26] They captured the surface of the forearm with a commercial 3D acquisition system and detected the area and volume of wheals. Their study showed very little inter- and intra-operator variability for repeated measurements. They also reported that the accuracy obtained with 3D imaging was superior to conventional SPT measurements. A disadvantage, however, was that they were only able to analyze each wheal individually which extends the procedure time. Moreover, since wheals fade away quickly it is desired to capture all wheals simultaneously for consistent assessment.[22,23] In following studies, researchers have developed multiple 3D scanning systems and algorithms for automatic measurement.[22,24,28–30] Although these methods show great potential in improving the accuracy and reliability of SPT reading, there is one important limitation. The use of 3D imaging technology requires high-cost equipment which is impeding its implementation in clinical practice. Another technique that has gained attention during the last two decades and also shows considerable potential is digital photography.[21,23,25,31–33] In contrast to 3D imaging technologies, digital photography is a simple and low-cost technique, and therefore offers accessibility to any healthcare institution, making it appealing for implementation and standardization across institutions. Because of these advantages, the Deventer Hospital (Deventer, The Netherlands) is currently exploring and developing a SPT-reading method based on digital photography and automatic computation of wheal sizes using artificial intelligence. A photograph of the patient's forearm is taken with an Apple iPad Pro, which is accessible for every institution without the need for special equipment. After pre-processing, the image is analyzed by an in-house developed deep learning network that automatically segments and computes the area of each wheal.[34] The wheal area is initially obtained in pixels and then converted into mm^2. For a reliable clinical interpretation, it is important that all pixels correspond correctly to their real-world dimensions to obtain accurate and true wheal sizes. There are two components that affect the uniformity of pixel dimensions: the camera perspective and the curvature of the forearm. At the moment, a photograph is being corrected for both aspects but there are still some limitations.

First, regarding the camera perspective, it is necessary that the image plane is parallel to the forearm to create an orthogonal view on the wheals. In practice, however, the camera will often be held at a certain angle with respect to the forearm. As a consequence, wheals appear larger or smaller on the photograph resulting in an over- or underestimation of wheal sizes. To overcome this problem, the current solution is to place an ArUco marker right below the elbow cavity (Figure 2) which can be used to determine the camera perspective and apply an image transformation. Further explanation of these steps is provided in the introduction of Chapter 2. A limitation of this approach is that the marker is prone to incorrect placement; it is frequently positioned *in* the elbow cavity, especially when there is small space left on the patient's arm. Consequently, the marker is tilted and not being parallel to the forearm and wheals resulting in an inadequate correction for camera perspective. In addition, the image transformation is now based on a small marker area in the image which means that the transformation will fit best around the marker but will be less accurate for wheals located at an increasing distance from the marker. It is hypothesized that a larger marker area will lead to a more accurate computation of wheal sizes. Furthermore, it is desired to develop a method that is not prone to incorrect placement of markers.

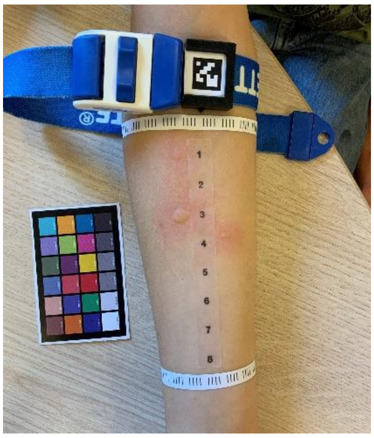


Figure 2 Setup of the current method for automated SPT reading, showing the use of barcode stickers and an ArUco marker.

Second, the cylindrical curvature of the forearm introduces a situation where the wheals are not located in the same plane when photographed from above. Especially when wheals appear more to the side of the arm their area is being underestimated. A possible way to correct for the arm curvature is to recover the 3D shape of the arm to a geometrically flat surface with image transformation.[35] This has previously been attempted with a script written by Geessinck, which is further elaborated on in the introduction of Chapter 3. It uses barcode stickers that are placed at the proximal and distal part of the forearm to detect the arm curvature (Figure 2). Unfortunately, a previous validation study of Krakers et al. has shown that the script is not working properly, probably due to ineffective barcode detection.[36] It is hypothesized that better detection of the arm contour will effectuate an image transformation that fits the shape of the arm more sufficiently, allowing for more accurate computation of wheal sizes.

In order to address the limitations mentioned above, the aim of this thesis is to further optimize and validate the automatic SPT reading method as developed by the Deventer Hospital. In Chapter 2, a novel method is presented to apply a correction for the camera perspective using a basic white placemat with printed ArUco markers, which is validated with a checkerboard as reference object. In Chapter 3, the same placemat is proposed as a new method to correct for the arm curvature, which is also validated with a checkerboard. A comparison between the performance of the arm curvature correction with the placemat method and the conventional barcode stickers is provided in Chapter 4. Finally, based on the studies in the previous chapters, a general discussion and future recommendations are outlined.

CHAPTER 2. Optimization and validation of a correction for the camera perspective

2.1 INTRODUCTION

The SPT is a widely used test to diagnose allergies, in which the sensitivity to an allergen is tested by manually measuring wheal sizes.[17,18] The test has numerous advantages, such as being inexpensive, minimally invasive and immediately providing results, but suffers from inter- and intra-observer variabilities, making reading of the test subjective.[18–21,37] Hence, it is desired to standardize the conventional SPT reading method, and automation through digital photography is a potential solution that is interesting because it is a low-cost and accessible technology.[23,31–33] When using photography for assessment of the SPT, it is important that the dimensions on the photograph match with the real-world dimensions for reliable and accurate wheal size measurement. As mentioned in Chapter 1, the camera perspective is one of the aspects that has an impact on the pixel dimensions. It is important that the image plane is parallel to the surface of the forearm, because when the camera is held at a certain angle with respect to the forearm this results in an over- or underestimation of the wheal area. This concept is illustrated in Figure 3; a perspective view is created when an object in the world plane is projected on the image plane at a certain angle whereas a parallel projection provides an orthogonal view.

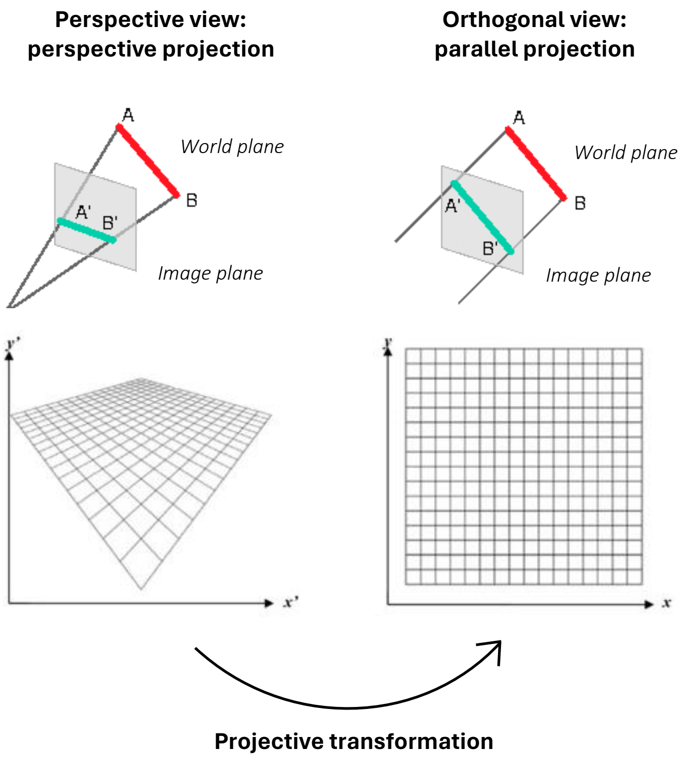


Figure 3 Illustration of the effect of camera perspective: projection of a real-world object on an image plane. The perspective view of an image can be transformed into an orthogonal view by performing a projective transformation. (Adjusted from [38,39])

In practice, however, the photograph for the SPT is often not taken from straight above the patient's forearm, so an appropriate way is to correct afterwards for the camera perspective through computer vision and apply a projective transformation.[39,40] This type of transformation enables a tilted image plane with lines converging to a vanishing point to become parallel lines again, thereby removing the effect of depth and altering the perspective view into an orthogonal view, as illustrated in Figure 3. In order to transform the image, control points are needed which are a set of points in the original image (moving points) that are correspondingly mapped to their desired positions in the transformed image (fixed points). Based on these paired control points, a 3×3 geometric transformation matrix is estimated that fits the moving and fixed points best and provides the parameters for optimal alignment of the original and reference image. The control points must be robust to detect and identify, which is why the current approach being investigated by the Deventer Hospital is to use an ArUco marker in the image for perspective correction. ArUco markers are square fiducial markers with a typical binary pattern that allow robust detection and identification and have a known geometry.[41] The four corners of the marker provide four control points that are located at a known distance of each other. In the current approach, the ArUco marker is placed onto a small band that is wrapped around the patient's forearm such that the marker is positioned right below the elbow cavity. As explained in Chapter 1, the first problem that arises is that the marker is sensitive to misplacement as it is frequently tilted *in* the elbow cavity, indicating that the marker is not in line with the surface of the forearm and wheals. As a consequence, since the image transformation is based on the perspective and tilt of the marker, the transformation matrix will fit the perspective of the forearm less properly, possibly leading to inaccurate computation of wheal sizes. Second, it is hypothesized that the use of four control points that are close to each other also might result in a suboptimal image transformation; a larger marker area is expected to generate a transformation matrix that better fits the entire image, and thus possibly leads to more accurate determination of wheal sizes.

The aim of this chapter was to optimize the correction for camera perspective in order to achieve uniform pixel dimensions that match with the real-world dimensions and make wheal size measurement more accurate. A novel method is proposed in which a basic placemat with four integrated ArUco markers in the corners is used to determine the perspective and perform a projective transformation. The approach is validated with a checkerboard of known shape and size to verify whether uniform pixel dimensions are obtained after transformation.

2.2 METHODS

2.2.1 Design of placemat

Based on a list of requirements (Table 1), a placemat was designed consisting of a sturdy white A4 paper sheet with four ArUco markers (Figure 4). The markers were aligned and printed on each corner of the placemat at a known distance from each other which was 237 mm in width and 150 mm in length. In a clinical setting, the purpose is to take a photograph of the patient's forearm placed on the placemat in between the markers, as illustrated in Figure 4.

2.2.2 Data acquisition for validation

In order to validate the correction for camera perspective using the ArUco markers, a checkerboard with squares of 10×10 mm was printed on the placemat. Checker patterns are widely used in computer

Table 1 Requirements for the placemat design

Requirement	Description
Size	The forearm must fully fit on the placemat.
Markers	To enable a correction for camera perspective using computer vision, fixed marker points must be integrated in the placemat. The markers should allow for automatic detection and have a known distance from each other. The area in between the markers must enclose the part of the image that captures the wheals.
Color	To enable a correction for arm curvature, the color of the placemat has to create a clear contrast between the forearm and the background to facilitate segmentation of the forearm (see Chapter 3).
Material	The placemat must not reflect light in order to provide good visibility of the markers, forearm and wheals. Furthermore, it must be hygienic, reusable and consist of firm material that is not prone to create folds.
Accessibility	The product should be low-cost, reproducible and available for other health care institutions.

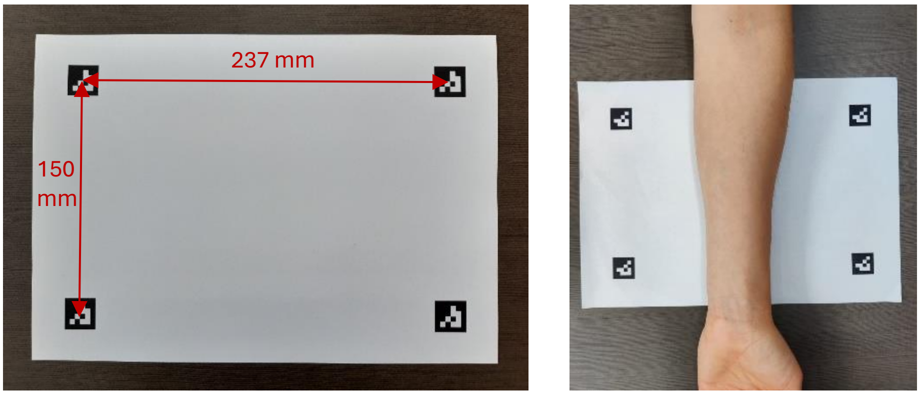


Figure 4 Design of the placemat: a sturdy white A4 paper sheet with four ArUco markers in the corners (left), and setup in clinical practice (right).

vision for camera calibration and geometry measurements because of their robust detection and high sub-pixel accuracy.[42–44] Four photographs of the placemat with checkerboard were taken from different camera angles to generate various perspective views. The axes of the camera are defined in Figure 5; the photographs were taken from a 15°, 30° and 45° rotation angle around the pitch axis, and one was taken at a 20° rotation angle around the pitch axis and 10° around the roll axis (Figure 6). The data was acquired with a smartphone camera (Samsung Galaxy A40, camera resolution: 16 megapixels) and the angles were measured with a mobile application (*Waterpas, Schietlood, Level*, NixGame, 2021).

2.2.3 Image processing

Several steps were performed to correct the photographs for camera perspective and validate the results, which is illustrated in a flowchart in Figure 7 and explained in more detail below. In summary, the perspective was corrected by first detecting the ArUco markers and use the information of their locations to fit and apply a projective transformation. The pixel sizes in the transformed image were then converted into mm after which the dimensions of the checkerboard squares could be measured. All data was processed in MATLAB (Release 2022b, The MathWorks, Inc). The full script can be found in Appendix A1.

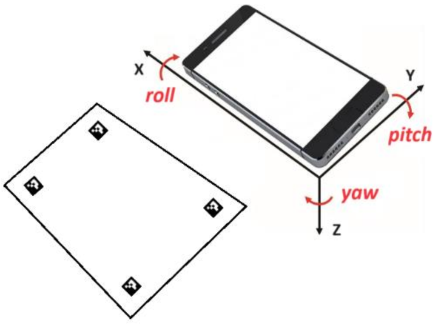


Figure 5 Definition of the camera axes: roll, pitch and yaw

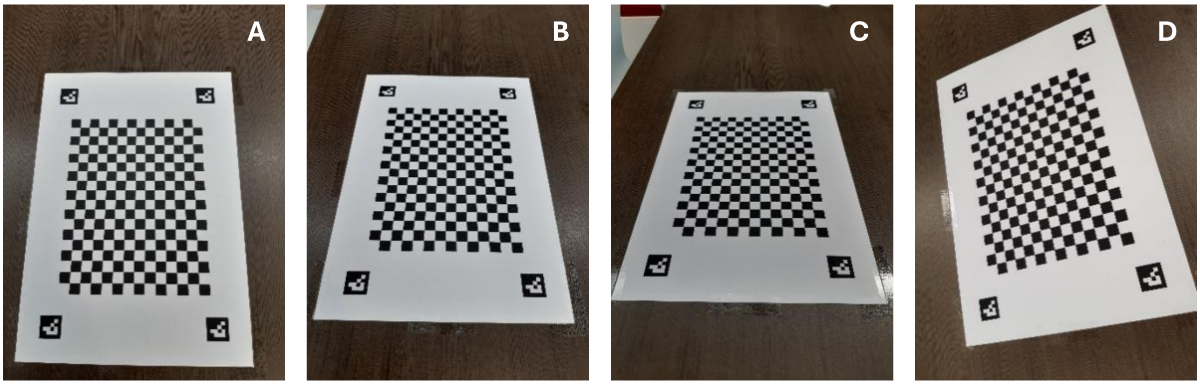


Figure 6 Photographs of the placemat with checkerboard for validation of the perspective correction viewed from different camera angles: 15° pitch (A); 30° pitch (B); 45° pitch (C); 20° pitch and 10° roll (D).

Detection of ArUco markers

Because MATLAB offers no specific function to detect ArUco markers, the detection of the markers was instead performed with histogram thresholding utilizing the stark contrast of the markers. Since the printed checkerboard and dark background of the table underneath the placemat also contained high contrast, this would interfere with isolated detection of the markers. Therefore, the checkerboard and background were first removed from the original image using Paint.NET (paint.net 5.0.7). This color image was then converted to a grayscale image on which thresholding (value 0.25) was applied, resulting in a binary image that leaved the ArUco markers. With the function *regionprops* four rectangular bounding boxes were created that enclosed each marker region. Subsequently, the centers of these boxes were determined giving the location of the ArUco markers in exact coordinates. The coordinates were sorted in order to identify which marker was located in the upper left (marker 1), upper right (marker 2), bottom left (marker 3) and bottom right (marker 4) corner of the placemat. The obtained coordinates of the centers of the markers formed the moving points for the image transformation.

Projective transformation

The fixed points for transformation were based on the known geometry and distances between the centers of the ArUco markers, for which the ratio of the length and width between the markers on the placemat was calculated. Marker 1 remained its original position in the image, and from there the fixed points of the other three markers were computed. Marker 2 was placed at the same height as marker 1 maintaining the distance between marker 1 and 2. Next, marker 3 and 4 were placed such that a rectangle was formed consisting of its known ratio. The function *fitgeotform2d* was then used to generate a 3x3 projective transformation matrix that fitted the moving and fixed points. This transformation matrix was then applied onto the original image using the function *imwarp*, resulting in an image corrected for camera perspective.

Pixel to mm conversion

In order to relate the measurements of the checkerboard squares in the image to real-world dimensions, the pixel sizes were converted to mm. The distance in pixels between the markers in the transformed image was determined and divided by their known distance in mm, resulting in a number of mm per pixel. To be as accurate as possible, the pixel-to-mm ratio was calculated for both the horizontal and vertical direction of the pixels, using the averaged pixel distances between marker 1&2 and 3&4, and between marker 1&3 and 2&4 respectively.

Measurement of checkerboard squares

The function *detectCheckerboardPoints* was used to automatically detect all corners of the checkerboard squares in the transformed image and return their coordinates. The algorithm behind this function relies on filtering the input image with small blocks of corner patterns and verifying corners by their gradient distribution.[44] The obtained corner coordinates were ordered by their positions: from left to right and then from top to bottom, and were used to calculate the x-length, y-length and area for each square. The outcome in pixels was converted to mm and mm^2 by multiplying the number of pixels by the previous determined pixel-to-mm ratios.

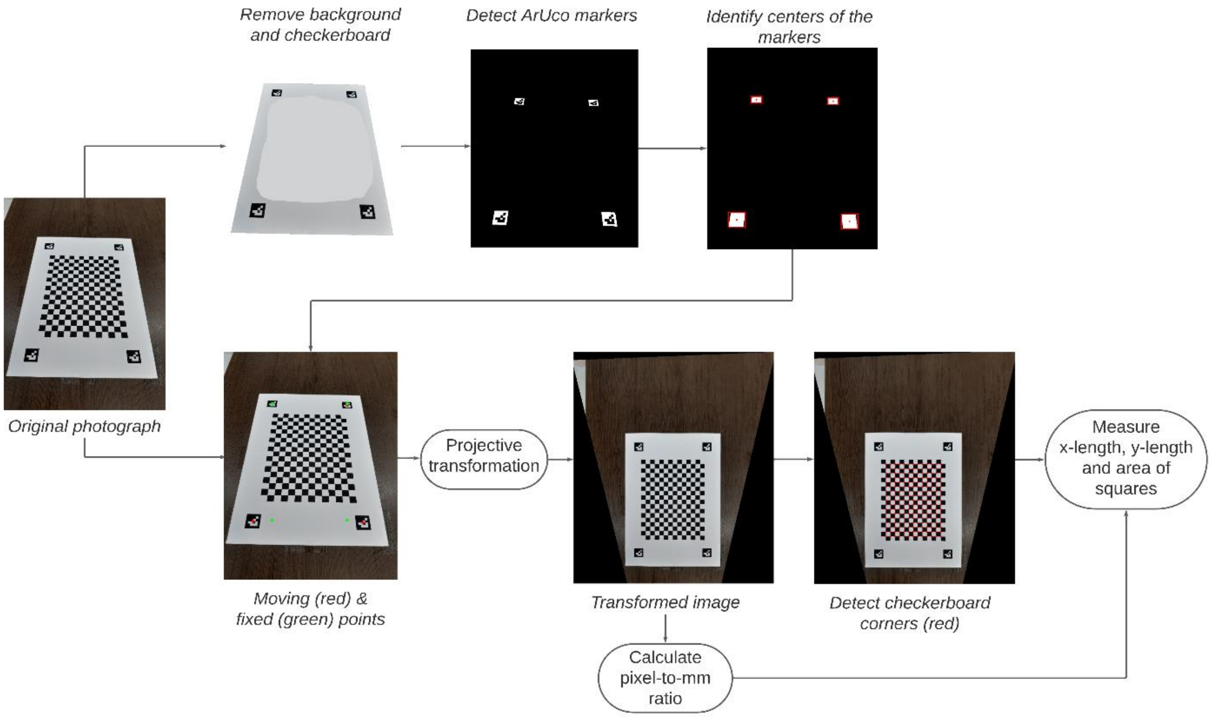

Remove background and checkerboard
Detect ArUco markers
Identify centers of the markers
Original photograph
Projective transformation
Moving (red) & fixed (green) points
Transformed image
Detect checkerboard corners (red)
Measure x-length, y-length and area of squares
Calculate pixel-to-mm ratio

Figure 7 Flowchart of the correction for camera perspective with the placemat method, including validation

2.2.4 Data analysis

The x-length, y-length and area of all detected checkerboard squares in the transformed image were analyzed qualitatively as well as quantitatively. For qualitative analysis, the shape of the squares was visually inspected to assess whether they were equally sized and returned their original squared shapes. For quantitative analysis, the mean and standard deviation of the lengths and area of the checkerboard squares were calculated for each photograph and compared to the original size of 10×10 mm.

2.3 RESULTS

Figure 8 shows a close-up of the checkerboards from the four photographs after applying the projective transformation. The entire transformed images can be found in Appendix B. When assessed qualitatively, all transformed images return similar results: the placemat consists of a rectangle where the checkerboard squares appear to be square in shape and of equal size.

Table 2 gives the mean and standard deviation of the x-length, y-length and area of the checkerboard squares given in mm and mm^2 for each transformed image. All results are close to the original size of the 10×10 mm squares including small standard deviations.

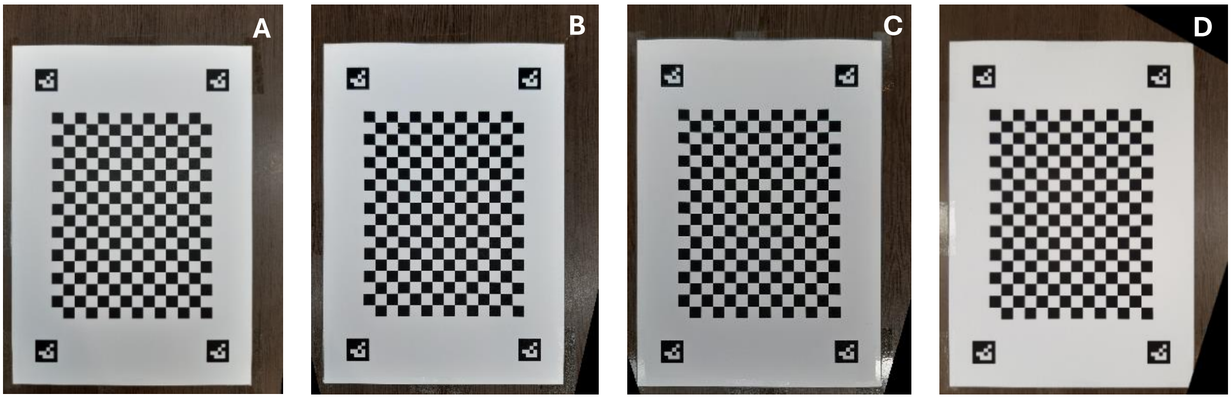


Figure 8 After projective transformation: photographs of the placemat with checkerboard (close-up) corrected for different camera perspectives: 15° pitch (A); 30° pitch (B); 45° pitch (C); 20° pitch and 10° roll (D).

Table 2 Mean and standard deviation (SD) of the x-length, y-length and area of the checkerboard squares measured after projective transformation.

	Mean ± SD		
Photograph	**X-length (mm)**	**Y-length (mm)**	**Area (mm^2)**
15° pitch	10 ± 0.1	10 ± 0.1	99.6 ± 1.5
30° pitch	9.9 ± 0.1	10 ± 0.1	99.0 ± 1.0
45° pitch	9.9 ± 0.1	10 ± 0.1	98.8 ± 1.0
20° pitch and 10° roll	10 ± 0.1	10 ± 0.1	99.6 ± 1.1

2.4 DISCUSSION

This work showed that the correction for camera perspective in photographs of the SPT can be optimized using a basic placemat with ArUco markers for performing a projective transformation of the image. The performance of the placemat was demonstrated to be satisfactory, showing a high accuracy. For all four camera angles tested, the distorted checkerboard squares in the photograph returned very close or exact to their original sizes after the perspective correction. The mean x-length, y-length and area of the squares deviated maximal 1% from the original size meaning only a few tenths of a millimeter, thereby showing very small standard deviations. These results are difficult to compare to related work. Five previous studies have been found that attempt to standardize the SPT with a photograph but none of them has taken into account a perspective correction.[25,31–33,45] The importance of a correct perspective, however, is emphasized by Zoltie et al. who state that within medical photography the camera must be perpendicular to the subject because an incorrect perspective can lead to misinterpretation of size, in this case the wheal size.[46] In general, projective transformations are widely used to correct for a perspective view in images and is also proven to be an accurate technique in the field of medical image registration, which is in line with the high accuracy found in the present study.[47]

The minimal deviations observed are probably not the result of the projective transformation but are more likely to be caused by the image quality. Because wheal sizes in the SPT are measured in the order of magnitude of millimeters, the deviations found are negligible and not clinically relevant. Moreover, from the fact that the results are similar for all different camera perspectives it can be stated that the proposed method is robust and works well even for the extreme angles (45° pitch; 20° pitch and 10° roll). It also is a positive indication that the standard deviations are very small since this indicates that the squares are equally sized throughout the image, which was one of the main goals of this research, to create uniform pixel dimensions. Unfortunately, this result cannot be directly translated to the measurement of wheals in the SPT because this research has a limitation with clinical impact: the setup consisted of a checkerboard that was printed onto the placemat and therefore is in the exact same plane as the markers. Objects in this plane are shown to be accurately corrected for the camera perspective. For the SPT, however, the forearm will be placed onto the placemat and, when following the anatomy, the arm has a larger diameter near to the elbow which becomes smaller towards the wrist, resulting in a varying height along the arm with respect to the placemat. The forearm thus has a certain inclination and will not be completely parallel to the surface of the placemat. Therefore, it will still have a certain perspective projection in the transformed image. This means that after correction the upper part of the forearm would be closer to the camera than the lower part, which consequently would lead to still a certain over- and underestimation of the wheals respectively. This is an important issue for which further research is needed to determine the effect of the varying height of the forearm on the perspective correction and whether it is clinically relevant. If it does have a significant impact, then it may be possible that the error is predictable. The correction with the placemat namely results in consistent images that are parallel to the placemat so that a baseline is created in which the height of the arm with respect to the placemat is the only variable factor left. The difference in height along the arm may then, for example, be predicted by the diameter of the arm. This opens options to yet enable reliable measurement of wheals even though initially the arm is not completely parallel to the image plane. These options should be looked into as a part of future research. The small sample size in this study can be considered as a limitation as well. However, the four images that are validated include a wide range of camera angles; even the more extreme perspective views are tested for which it is unlikely that a photograph of the SPT will be taken from such angles. As the results are consistent and accurate for all perspective views, there is no reason to suggest that the proposed method would perform worse for other camera angles within this range.

Whether the placemat outperforms the conventional ArUco marker band cannot be concluded based on this study as there was no comparison made between both methods. Although it can be stated that the correction with the placemat shows a high accuracy and has several practical advantages over the ArUco marker band, including that the placemat is not sensitive to misplacement and leaves more space on the forearm to place the allergen drops. Considering these arguments, it is assumed that the placemat improves the conventional method for the perspective correction, even though it cannot be proved within this research.

Besides the before mentioned advantages, the placemat is flexible and convenient in use, low-cost and generally applicable for health-institutions. Another benefit, regarding its design, is that the distance between the markers is not limited to one specific ratio; it can be any distance as long as the markers form a rectangle of which the proportions are known so that the geometry of the transformed image can be calculated. This allows for adjustment of the size of the placemat, which might be desired for younger patients who have a smaller arm. A side note is that the detection of the markers was now performed in MATLAB with histogram thresholding that is based on binarizing the image. This approach has a limitation since other high-contrast objects than the markers are also detected such as a darker background on the photograph, for which in this study these parts of the image were removed. This step is not ideal and can simply be overcome by using a specific function that can detect ArUco markers directly. This function is not available in MATLAB but does exist in OpenCV for which Python programming language is required.[41] Therefore, it is recommended to translate the algorithm to Python to provide an optimal and more robust marker detection.

2.5 CONCLUSION

This is the first study that suggested an approach to correct for the camera perspective in photographs of the SPT. ArUco markers on a basic designed placemat were used to perform a projective transformation of the image to obtain an orthogonal view on the placemat, which showed promising results with a high accuracy. As, in this study, the approach was tested on a checker pattern that was situated in the same plane of the placemat, the most important future step is to further assess the performance of the perspective correction for the situation of a forearm with wheals that is not exactly parallel to the placemat and has a certain inclination. Using the placemat for correction of the photographs' perspective has a high potential in obtaining more reliable wheal size measurement as it could provide uniform pixel dimensions in the image, meaning that the wheals measured in the image match the real wheal dimensions, which is a step forward to automated and standardized SPT reading with digital photography.

CHAPTER 3. Optimization and validation of a correction for the arm curvature

3.1 INTRODUCTION

The SPT is a widely used tool for diagnosing allergies since it is a fast, simple and minimally non-invasive test.[18,20,37] However, the manual measurement of wheals makes assessment of the test prone to inter- and intra-observer variability, indicating there is need for standardizing the results.[19,22–24] To realize a standardized SPT readout, automation through digital photography may offer a solution.[23,31–33] When capturing the wheals on a photograph, it is important that the dimensions in the image match with the real-world dimensions for reliable and accurate wheal size measurement. As mentioned in Chapter 1, the curvature of the forearm is one of the aspects that affects the pixel dimensions; wheals located more towards the sides of the arm will appear smaller in the image leading to an underestimation of their area. This is because, due to the cylindrical shape of the arm, the wheals are not located in the same plane and are therefore not uniformly projected onto the image. In medical photography, it is advised to account for curved surfaces as they cause inaccurate measurement of object sizes in the image.[48] Additionally, Justo et al. emphasized that it is crucial to remove the significant curvature of the forearm for determination of wheal areas.[24] For two-dimensional (2D) images, an appropriate option to correct for the curvature is to apply computer vision and transform the image such that the 3D curved surface of the forearm is recovered to a 2D flat surface where all wheals are situated in the same plane, which is illustrated in Figure 9.

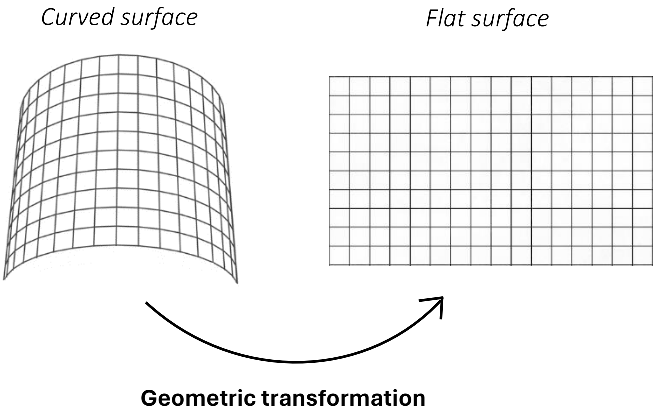


Figure 9 Illustration of a curved surface (i.e. forearm curvature) that is not uniformly projected onto an image, which can be corrected for with a geometric transformation that recovers the curved surface into a flat surface with equal dimensions.

This approach is currently being investigated by the Deventer Hospital, where Geessinck et al. developed an algorithm that makes use of two small barcode stickers placed near the elbow cavity and wrist. The stickers are used to define the control points for the geometric transformation, which are a set of corresponding points in the original image (source points) and their desired destination in the transformed image (fixed points). Detection of the barcode stripes gives information about the shape of the curved arm in the original image from which the source points can be estimated. Because the number of detected stripes and the distance between them is known, their desired positions for a flat geometry can then be calculated as fixed points. Further explanation of this method using barcode stickers is provided in Chapter 4. The source and fixed maps form the input for generating a transformation matrix that fits the paired control points best and finds an optimal transformation. The type of transformation performed in the algorithm is a B-spline (basis spline) transformation, which is non-linear and performs local deformations in the image. B-spline transforms gained considerable attention in medical image registration because they are known for its good performance in localized deformation and registration.[49]

Although an image transformation seems a feasible solution to correct for the arm curvature, a validation study of Krakers et al. showed that the algorithm with the barcode stickers does not work properly.[36] The problem seems to be ineffective detection of the barcode: the outer barcode stripes at the left and right edge are not always detected sufficiently. As a consequence, a suboptimal source and fixed map are obtained that do not follow the contour of the arm closely. In these cases the sides of the forearm are missing in the calculated geometry and this region is thus not corrected for in the transformation to a flat surface. The barcodes are detected by thresholding on grey values of the image and subsequent morphological operations that filter out the specific shape of the stripes. Small improvement can be achieved by adjusting these parameters, but still the full barcode is not always detected. Moreover, the parameters must be optimized manually for each individual photograph which is not a robust method. It is hypothesized that better detection of the arm contour will yield a source and fixed map that encompasses the shape of the forearm more completely and therefore may lead to a transformed image that matches the real-world dimensions more accurately. Also, it is expected that the performance of the transformation may differ for different locations on the forearm; the curvature is stronger towards the sides of the arm and therefore this region requires a larger deformation which is more challenging to be correct. It is therefore interesting to gain insight in the areas on the forearm where the antigen droplets of the SPT are typically placed and where the wheals generally appear, as this is where accurate correction for the arm curvature is most crucial.

This chapter is divided into two studies. In the first study, the typical location of wheals on the forearm is explored by analyzing a number of photographs of patients who had an SPT. The second study aimed to optimize the current method for arm curvature correction and to achieve uniform pixel dimensions in the photograph, so that assessment of the SPT will be more reliable and accurate. The same placemat as introduced in Chapter 2 is suggested to replace the barcode stickers. With the placemat it is attempted to improve the detection of the arm contour with the ultimate goal to recover a flat surface that fits the complete forearm. To test its performance a validation method is developed that uses a checkerboard in the image with squares of known shape and size that can be measured to verify whether the original and real-world dimensions are returned after curvature correction. The results are compared to the uncorrected image. Moreover, the performance is related to the outcome of the first study by assessing the results of the curvature correction at specific regions on the arm and relating this to the general appearance of wheals at these locations.

3.2 METHODS

All data processing has been executed in MATLAB (Release 2022b, The MathWorks, Inc). The scripts are provided in Appendix A2-A4. Section 3.2.1 describes the first study concerning the localization of the wheals on the forearm. The methods of the second study are detailed in Section 3.2.2, outlining the steps for the arm curvature correction using the placemat, and in Section 3.2.3, which specifies the validation process.

3.2.1 Location of wheals on the arm

Data acquisition

A pre-existing dataset was used consisting of photographs of SPTs performed on patients from the Children Allergy Center of the Deventer Hospital. From this dataset, the wheals in 88 photographs of children <18 years of age were analyzed.

Localizing the wheals

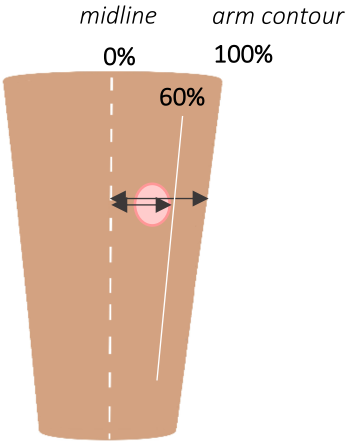


Figure 10 Example of a wheal position at 60%. The location of a wheal is defined as the relative distance of the wheal with respect to the midline of the arm.

The location of a wheal on the forearm was defined as the relative distance with respect to the midline of the arm and ranges from 0-100%. For example, a wheal position of 60% means that the most lateral side of the wheal was located at 60% away from the midline as is clarified in Figure 10. No difference was made between the distance from the midline to the either left or right arm contour. The first step for localization was to approximate the left and right arm contour by two straight lines of which the begin and end points were manually clicked in the photograph. Second, the position of the midline was determined by drawing a line from calculated average points between the two begin and two end points of the arm contour. The most lateral protrusion of a wheal circumference was then manually clicked in the photograph. Next, the image was rotated such that the midline was rendered as a straight vertical line, which was combined with the coordinates of the wheal points and lines of the arm contour in a plot. From here, for each wheal separately, the y-coordinate of the wheal was used to determine the intersection with the arm contour at that height. Now that the positions of the midline, the wheal and its associating intersection with the arm contour were known, their x-coordinates could be used to compute the distance between the midline and both the wheal and arm contour. These distances were divided by each other to obtain a relative distance of the wheal with respect to the midline. An overview of the previous steps is shown in Figure 11.

Data analysis

The relative distances of all wheals with respect to the midline of the arm were analyzed using a histogram giving the variation and frequency of the different wheal positions.

3.2.2 Arm curvature correction with placemat

The proposed method to correct for the forearm curvature was based on the use of the placemat introduced in Chapter 2, for which the requirements in Table 1 were taken into account. It was chosen to design a white placemat instead of a specific color since this met the requirement of an accessible and reproducible product, while it is proven to offer sufficient contrast with a light skin color of the arm

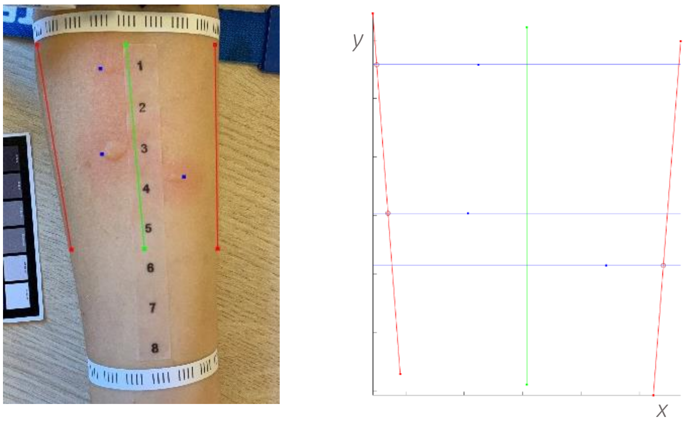


Figure 11 Illustration of localizing the wheals on an arm: the approximated left and right arm contour (red lines), midline of the arm (green line), and most lateral point of the wheals (blue dots) were first determined in the photograph and then visualized in a plot to calculate the position of the wheals with respect to the midline.

and therefore also expected to supply for darker skin tones.[50] The processing steps for correction are explained below and depicted in Figure 12. In short, the contrast between the arm and the placemat enabled detection of the arm contour which formed the input for the source points of the image transformation. The fixed points were then calculated with an estimation of the arm circumference using the formula of an ellipse.

Detection of the arm contour

First, to select the region in the photograph containing the placemat, the image was cropped around the four ArUco markers. For this step, the markers were detected by thresholding on grey values which is explained in Chapter 2, Section 2.2.3. After cropping, segmentation of the arm was performed. As the presence of shadows in the image posed a challenge for proper segmentation, this effect was overcome by converting the standard RGB (red-green-blue) color channels to a Lab color space. Lab contains a distinct channel for luminance (L) and color channels for the red-green chrominance (a) and the blue-yellow chrominance (b).[51] By omitting the L component, the impact of shadows was suppressed. Subsequently, k-means clustering was applied on the remaining a and b channels, where the image colors were divided into two clusters, thereby segmenting the arm from the background.[52] The contour from the segmented arm was then obtained with Sobel edge detection.

Source map

Next, four source key points were defined: an upper and lower point on either left and right arm contour. To obtain these points, two horizontal lines were created between the upper and lower two ArUco markers which intersected with the arm contour and gave four intersection points. When the forearm is not positioned vertically straight on the placemat, the lines will cause an oblique cross-section of the arm; however, a perpendicular cross-section is needed as the arm’s width will be used to estimate the fixed points later in the process. To overcome this problem, the image was rotated such that the midline of the arm was aligned vertically straight. Here, the midline was created as follows: an upper and lower midpoint in between the two upper and lower intersection points were determined, after which a line was drawn connecting these midpoints. Subsequently, the angle formed between the midline and the horizontal line of the ArUco markers was calculated and subtracted from 90°, resulting in the angle needed for rotation. The arm contour was then rotated while the horizontal lines of the

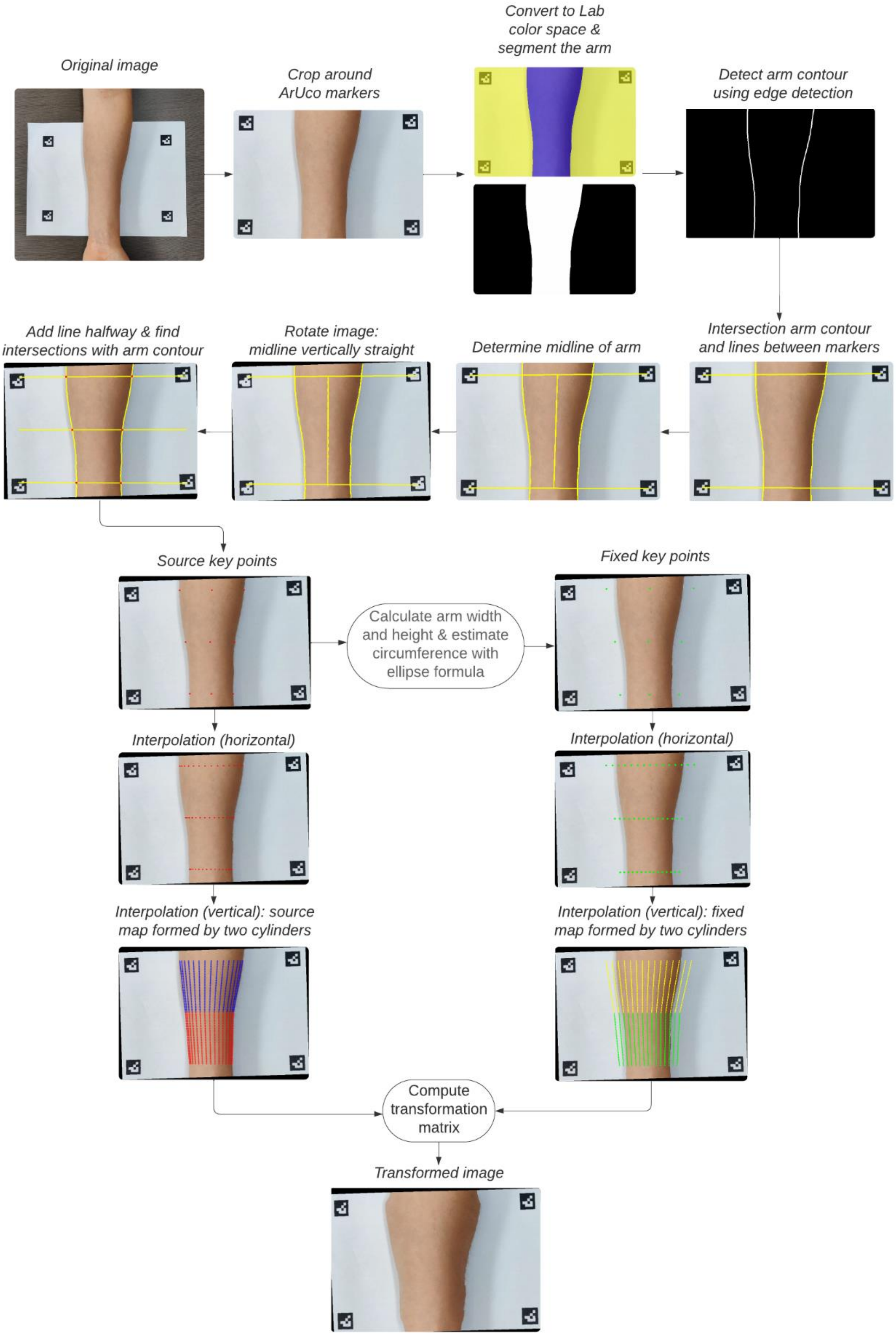

Original image
Crop around ArUco markers
Convert to Lab color space & segment the arm
Detect arm contour using edge detection
Intersection arm contour and lines between markers
Determine midline of arm
Rotate image: midline vertically straight
Add line halfway & find intersections with arm contour
Source key points
Calculate arm width and height & estimate circumference with ellipse formula
Fixed key points
Interpolation (horizontal)
Interpolation (horizontal)
Interpolation (vertical): source map formed by two cylinders
Interpolation (vertical): fixed map formed by two cylinders
Compute transformation matrix
Transformed image

Figure 12 Flowchart of the arm curvature correction using the placemat

markers were remained in their position. This way, four new intersection points were obtained which now corresponded to a perpendicular cross-section of the arm. To further improve the accuracy of the estimated arm curvature, two extra source points were added halfway the arm to follow the arm contour more closely along its length. These points were obtained by the intersection of the arm contour and an added horizontal line halfway between the markers. Last, three key points at the midline of the arm were determined at the level of the upper, middle and lower intersection points. In total, nine source key points were defined.

Fixed map

To obtain the corresponding fixed points, their desired position in the transformed image was determined by approximating the circumference of the arm with the formula of an ellipse[53]:

$$C = \pi(a+b)\frac{135168 - 85760h - 5568h^2 + 3867h^3}{135168 - 119552h + 22208h^2 - 345h^3} ,$$

where h is given by the following equation:

$$h = (\frac{a-b}{a+b})^2 .$$

Here, a and b are the radii of the ellipse, which are in this case the width and height of the forearm respectively. These were determined at the level of the upper, middle and lower source points on the arm. The width was computed by measuring the distance between the source points on the left and right arm contour. The height of the arm was estimated to be approximately two-thirds of the width:

$$b = \frac{2}{3}a .$$

For all three sets of upper, middle and lower source key points, the fixed points were positioned such that the distance between them was half of the calculated arm circumference, while maintaining their midpoints on the midline of the arm.

Image transformation

From here, the same steps were performed as in the existing script for the curvature correction with the barcode stickers. The source and fixed key points were expanded by interpolation to form a complete source and fixed map with a grid of points estimating the overall shape of the arm. In the horizontal direction fifteen points were interpolated to connect the left and right arm contour for all three upper, middle and lower determined key points. For the source map, these points are not equally spaced but become closer together towards the sides of the arm since the curvature is increasing here leading to higher spatial information that must be captured. For the fixed map, the fifteen points are interpolated equally because the dimensions should be uniform across the forearm after transformation. In addition, interpolation in vertical direction was performed connecting the rows of fifteen points, including hundred interpolated points per line. This was done similarly for both the source and fixed map. Finally, a B-spline transformation matrix was computed that fitted the constructed source and fixed map as optimal as possible. By applying the transformation matrix, the original image was then transformed into a curvature corrected image.

3.2.3 Validation of the arm curvature correction

Data acquisition

To evaluate the performance of the curvature correction using the placemat approach, data of six healthy adult volunteers were collected. Per volunteer one photograph was taken of the forearm positioned on the placemat. A paper checkerboard with squares of 25 mm^2 was wrapped around the arm, thereby following the shape of the arm as close as possible. Additionally, barcode stickers were added which were not purposed for this study but for the analysis of another study described in Chapter 4 where the same data was used. The photographs were taken with an Apple iPad Pro (second generation; camera resolution: 12 MP). An example of the setup is shown in Figure 13.

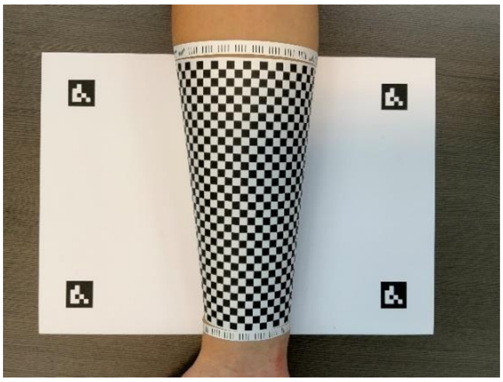


Figure 13 Setup for validation of the arm curvature correction

Pre-processing

Each photograph was first corrected for the camera perspective as described in the methods of Chapter 2. This eliminated the effect of a perspective view on the dimensions of the checkerboard and provided exclusive examination on the effect of the curvature correction.

Curvature correction

After pre-processing, the images were corrected for the arm curvature by performing the placemat method as detailed in Section 3.2.2, except for one step concerning the segmentation of the arm that gives the arm contour. This part is based on the contrast between the skin color and placemat but was hindered in the validation images by the checkerboard that covered the skin. To overcome this problem, the area of the checkerboard in the image was overpainted with skin color in Paint.NET (paint.net 5.0.7). The photoshopped image was only used to obtain the transformation matrix, from there the transformation matrix was applied to the original checkerboard image for validation.

Validation method

A semi-automated validation method was developed which was based on segmentation of the squares. The steps described below have been performed for both the uncorrected and corrected images.

Masks

To select the region of the checkerboard in the image, a binary mask was created that included the part of the checkerboard that fell within the area of the ArUco markers. Masks were created for the uncorrected and corrected image separately, for the entire forearm and the different regions. The mask for the uncorrected image followed the arm contour and was obtained from the segmented arm. This initial mask covered the complete checkerboard and therefore also included the half squares at the edges of the arm of which their area was only partially visible in the image. As these squares were not representative as full-sized squares, they were excluded from analysis by creating a second mask that covered 96% of the width of the forearm. The width of this mask was determined by calculating the distance between the arm contours and the midline of the arm, and calculating the new position of the arm contour when taking 96% of the distance. The same method was applied to create masks for different regions on the forearm, by shifting the arm contour inwards to a defined percentage. The associating masks for the corrected images were derived from the uncorrected masks by transforming the corrected masks with the transformation matrix that was previously computed for the curvature correction.

Segmentation of checkerboard squares

For segmentation of the checkerboard squares, the color image was first converted to a grayscale image of which its contrast was enhanced and then converted to a binary image. A complement of the binary image was obtained where all black squares were segmented, after which the previously created masks were applied onto the image. At this point, the checkerboard squares were not recognized as separate objects because the corners of the squares were connected to each other causing large fused areas. To disconnect the squares the image was first eroded with a square shaped structuring element, after which the separated squares were labelled with the function *bwlabel.* The image was then dilated using the same structuring element to reobtain the original sizes while maintaining the distinct label for each square.

Measuring and visualizing the area of the squares

For measuring the size of the checkerboard squares, the area of each square was calculated by counting the number of pixels. This was done with the function *regionprops* with the specified property name "*area"*. The absolute areas were scaled to relative areas by dividing the number of pixels of each square by the gold standard area, giving a percentage deviation. Here, the gold standard was defined as the mean area in pixels of the squares situated on the midline of the arm in the uncorrected image, assuming that these squares had a negligible influence of the arm curvature and were therefore considered to have its original dimensions. The gold standard area was given value 1 and, for example, a relative area of 0.8 or 1.2 meant that the square was respectively 20% smaller or larger than its original size. The value of the relative area was assigned to all pixels of that square. This way, the areas in the image could be visualized using a colormap where the colors represented the pixel values and thus the size of the checkerboard squares.

Data analysis

To evaluate the uniformity of pixel dimensions after arm curvature correction, both quantitative and qualitative analyses were conducted. For both uncorrected and corrected images, boxplots of the relative areas of the checkerboard squares were made for different regions of the forearm. Additionally, the results were visualized by displaying the checkerboard squares using a colormap that represented the values of the relative areas. For qualitative analysis, the shape of the squares after correction was visually inspected.

3.3 RESULTS

3.3.1 Location of wheals on the arm

A total of 236 wheals were analyzed and the distribution of the wheal locations on the forearm is rendered in a histogram in Figure 14. To offer more insight, Table 3 shows the exact number of wheals for different regions of the forearm, including wheals located at a relative distance of <70%, 70-80%, 80-90% and >90% from the midline of the arm. Most wheals appear halfway between the midline of the forearm and the arm contour. Towards the sides of the arm the appearance of wheals decreases to a few percent.

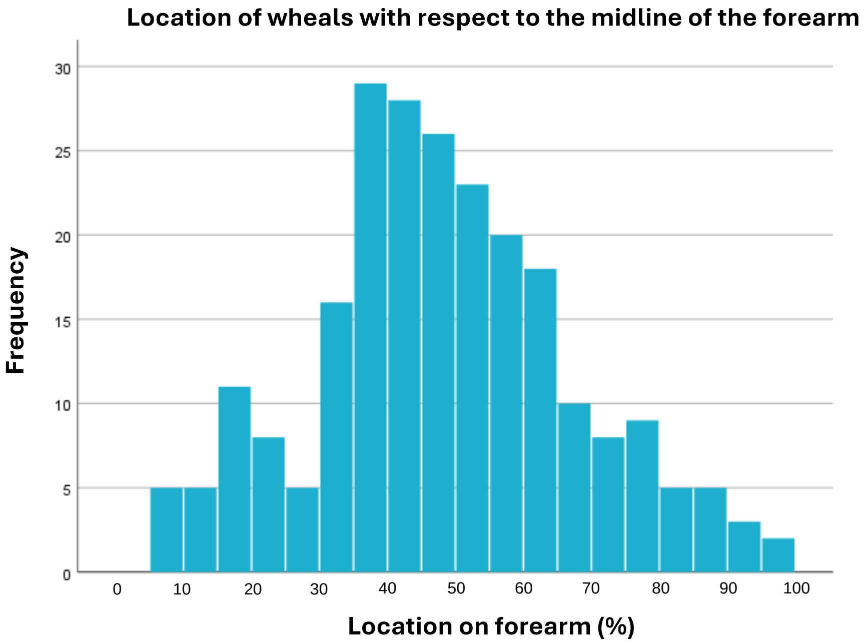


Figure 14 Histogram showing the typical location of wheals (n=236) on the forearm during SPT. The location is defined as the relative distance with respect to the midline of the arm, where 0% indicates the midline and 100% either the left or right arm contour.

Table 3 Distribution of the typical location of wheals for several regions on the forearm

Location of wheals on forearm	Number of wheals
< 70%	204 (86.4%)
70-80%	17 (7.2%)
80-90%	10 (4.2%)
> 90%	5 (2.1%)

3.3.2 Validation of the arm curvature correction

As an example, Figure 15 shows for subject 1 the image before and after arm curvature correction with the placemat. The uncorrected and corrected images for all individual subjects are given in Appendix C1.

The relative areas of the checkerboard squares are measured and visualized for the six subjects in Figure 16, where the colors represent the size of the squares as given in the colorbar. Per subject the results are given for the images before arm curvature correction and after correction with the placemat. The relative areas are quantitatively rendered in the boxplots of Figure 17 and show the distribution of the

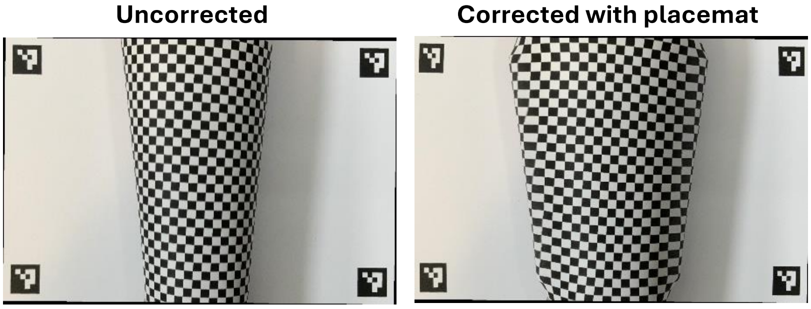


Figure 15 Example (subject 1) of the validation images before arm curvature correction (left) and after correction using the placemat (right).

size of the squares for four distinct regions on the forearm: <70%, 70-80%, 80-90% and >90% as illustrated in Figure 18. The boxplots contain the data of all subjects together as there were no major differences among the individual boxplots. Since the data is not normally distributed, the median and interquartile range (IQR) of the areas are calculated and given in Table 4.

From the visualizations and boxplots it can be seen that the checkerboard squares become increasingly smaller towards the sides of the arm when no correction for arm curvature is applied, thereby clearly demonstrating the effect of the forearm curvature. After correction with the placemat, the squares are returned closer to their original size indicated by the median relative area that approaches value 1. The performance differs for the distinct regions on the arm as the error becomes larger towards the sides of the arm, which corresponds with the orange to (dark) red colored squares in the visualizations of all subjects. Additionally, the IQR shows that the variability increases towards the sides of the arm, especially for the squares located at >90% away from the midline. Furthermore, in the uncorrected as well as corrected images a gradient can be seen in the vertical direction; from proximal to distal the squares are decreasing in size, which is most clearly visible for subject 3. The strength of the gradient differs per subject and ranges from a 5-20% difference in size based on the colors. Qualitative evaluation of the shape of the checkerboard squares reveals that the squares towards the sides of the arm become horizontally stretched after correction, which correlates with a larger size as displayed by their color. The middle squares are overall square shaped and no vertical distortion is observed for all squares.

Table 4 Median and interquartile range (IQR) of the relative areas of the checkerboard squares before arm curvature correction and after correction with the placemat.

	Median (IQR)	
Region on forearm	**No correction**	**Correction with placemat**
< 70%	0.94 (0.86 – 1.00)	1.01 (0.97 – 1.04)
70-80%	0.71 (0.66 – 0.77)	1.04 (1.01 – 1.09)
80-90%	0.57 (0.50 – 0.64)	1.07 (1.01 – 1.13)
> 90%	0.39 (0.33 – 0.45)	1.11 (0.92 – 1.26)

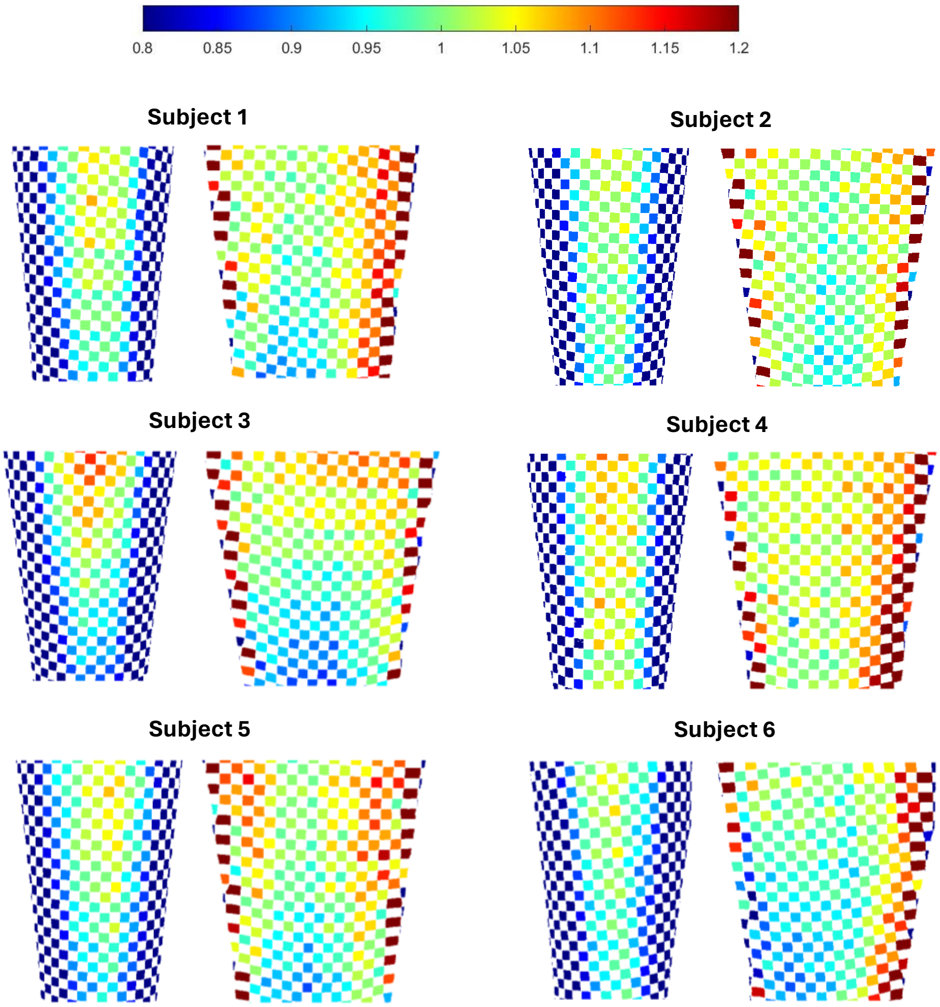


Figure 16 Visualization of the relative areas of the checkerboard squares obtained before arm curvature correction (left) and after correction using the placemat (right). The colors represent the value of the relative areas as shown in the colorbar, where value 1 is considered to be the gold standard area. The regions presented in the images include 96% of the forearm.

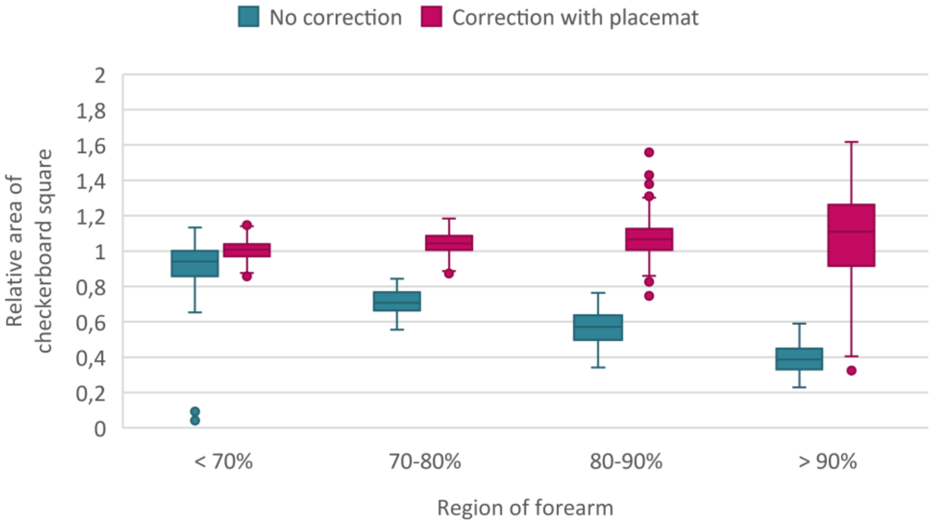


Figure 17 Distribution of the relative areas of the checkerboard squares in different regions of the forearm before arm curvature correction and after correction with the placemat, where value 1 is considered to be the gold standard. The plots present the data of all six subjects.

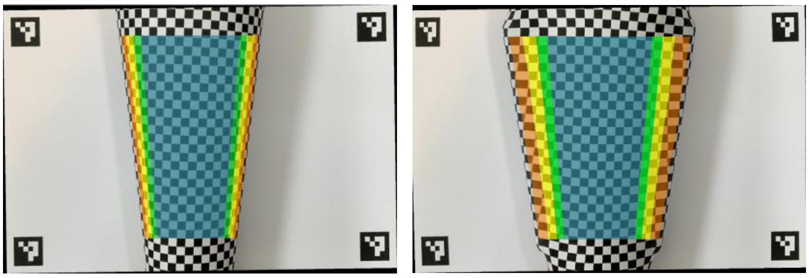


Figure 18 Illustration of the four defined regions on the forearm: <70% (blue), 70-80% (green), 80-90% (yellow) and >90% (orange). Example of the uncorrected image (left) and the corresponding regions on the arm after curvature correction with the placemat (right).

3.4 DISCUSSION

The present study showed that the correction of the forearm curvature in photographs of the SPT can be optimized using a basic placemat with ArUco markers.

3.4.1 Performance of placemat method

After curvature correction with the placemat the median size of the checkerboard squares became closer to its true size when compared to no correction, indicating that wheal size measurement becomes more accurate after applying the curvature correction. The performance depended on the region of the forearm: the accuracy decreased towards the sides of the arm, where a larger error was seen including a larger variability in size. The median relative areas of the 25 mm^2 squares were 1.01 (IQR: 0.97 – 1.04), 1.04 (IQR: 1.01 – 1.09), 1.07 (IQR: 1.01 – 1.13) and 1.11 (IQR: 0.92 – 1.26) for the regions <70%, 70-80%, 80-90% and >90% of the forearm respectively. Whether this accuracy is in agreement with or different from other studies that used digital photography for wheal measurement is difficult to say. Among the five studies found, only Svelto et al. corrected the image for the arm curvature.[25] They used the same technique as described in the present study: recovering the curved forearm to a theoretically flat surface. However, they have not described their approach in detail and also did not evaluate the accuracy of the curvature correction. Some of the other studies did measure the accuracy of their method, but these outcomes were primarily influenced by other factors, such as their approach of wheal detection, and therefore cannot be directly compared to the results in the present study.[31–33,45] When looking at the accuracy of other techniques than digital photography, it was demonstrated that the application of 3D imaging technologies resulted in errors <1 mm^2 for wheal area and <1.5 mm for wheal diameter.[22,24,26,29] Although these techniques seem to achieve higher accuracies than the performance of the placemat method, they are not necessarily recommended as they all have their own limitations of which the common drawback is that they all require more complicated setups or expensive equipment, which make them less attractive for clinical implementation.

To assess whether the suggested method for the curvature correction leads to reliable SPT examination, it is interesting what minimal accuracy is required for reliable wheal measurement. This is difficult to say since no statements are made about a generally accepted accuracy. However, the following reasoning could give an indication: a minimal difference of 1 mm should be measurable in order to assess the SPT as positive (mean wheal diameter >3 mm) or negative (mean wheal diameter 2 mm).[17] This corresponds with an approximated wheal area of 4.7 mm^2 and 2.1 mm^2 for a positive and negative test respectively[19], which means that a difference of 2.6 mm^2 must be measurable. This indicates that in the present study a maximum deviation of circa 10% of the actual checkerboard squares of 25 mm^2 would be acceptable. Based on this assumption and the IQRs, the performance of the curvature correction would be acceptable for up to 80% of the forearm. According to the results of the typical location of wheals analyzed in this study, this region includes most of the wheals in the SPT. The remaining 20% of the forearm occupies circa 6% of the wheals in patients, suggesting that the lower accuracy towards the sides of the arm would have a minimal clinical significance. Based on these findings, it can be considered to advise the clinicians to place the allergen drops more towards the midline of the arm to assure that all wheals will be located in the accurate region. For younger children with smaller arms, however, this is more challenging as there is less space on the arm for placing the droplets.

The fact that the error of the curvature correction becomes larger for the outer regions where the arm curvature increases confirmed the hypothesis. The larger squares imply an overestimation of the curvature. An explanation for this could be that the estimated arm circumference for the fixed map is not sufficiently accurate. The assumption that the height of the forearm is two-thirds of the measured width may not be a correct approximation, and moreover may differ per subject and age. To improve the estimation, it could help to generate a database of true measured widths and heights of the forearm of patients, which for example can be extracted from an existing set of CT scans. Another possibility is that the source map is not estimated accurately (see Appendix C2 for an overview of the created source maps). From the positions of the source key points it can be concluded that the arm contour is detected successfully and that the key points are determined correctly onto the arm contour, which was one of the goals of this study. However, it was found that a discrepancy arises when interpolating between the left and right key points, which is illustrated in Appendix D. The interpolated points are placed closer to each other towards the sides of the arm and their distance is calculated with several mathematical formulas that are based on the curvature of the cylinder in relation to their projection onto the 2D image. With this calculation the key points are moved a small distance inwards along with the other interpolated points, for which no reason could be found. The final source map therefore becomes slightly smaller than the originally detected arm contour and thus fits the shape of the arm less optimal, resulting in an overestimated curvature. In addition, this could on the other hand explain the appearance of some of the outermost squares that were underestimated, which might be squares that were not included in the transformation due to the smaller source map. Better matching of the interpolation points with the source key points could improve the source map and accuracy of the placemat approach. Finally, the deviation in size of the squares could also be explained by the fact that the forearm in general cannot be adequately fitted by a cylindrical shape. A limitation of this study is the small sample size of six subjects. Therefore, the results show a first indication of the performance of the placemat and need to be further evaluated on a larger number of subjects with different ages and arms' shapes.

The vertical gradient that was observed along the arm can be attributed to the effect of a perspective view. As a preprocessing step, the images were corrected for the camera perspective using the approach that is described in Chapter 2. However, as discussed in that chapter, a limitation is that after correction the image plane is parallel to the placemat but not fully parallel to the arm due to the varying height of the forearm. Although the degree of the effect differs per subject, it is a notable issue that must be addressed apart from the curvature correction.

3.4.2 Wheal locations

This is the first study that mapped the location of where the wheals on the forearm during SPT typically appear, which puts light on the clinical importance of the performance of the arm curvature correction for the different regions on the arm. The large sample size in this study is a notable strength. It is, however, worth noting that the analyses are performed on data of children of <18 years old for which the specific age characteristics were not available. The age has an influence on the width of the forearm; for younger children there is smaller space left on the arm to place the allergen drops which will probably lead to more wheals appearing towards the sides of the forearm. On the other hand, for older children and adult patients, the wheals may appear closer to the midline because of a larger width of the forearm. Taking in mind that the accuracy of the arm curvature correction decreases towards the sides of the arm, as discussed in the previous paragraph, this indicates that the number of wheals affected by a lower accuracy would be larger for the younger patients and smaller for older children and adults. Another note is that the wheal localization in this study is based on pinpointing the outermost part of the wheal that has the largest distance from the midline. It was chosen to define the wheal position by this single point because of its feasibility within the time limits of the research. A more complex but more accurate method would be to define the location of the wheal by its area on the forearm, which gives more information about the location of the entire wheal. Wheal areas may be spread over multiple regions, meaning that a larger percentage of the wheals may fall in the inner regions of the forearm than classified in this study, which would be beneficial for the accuracy of wheal measurement as the curvature correction performs better towards the midline.

3.4.3 Validation method

With the developed validation method of this study it was made possible to evaluate the arm curvature correction through segmentation and measurement of the checkerboard squares. The segmentation and measurement focused exclusively on the black squares, however, this effectively represents evaluation of the entire forearm, as these black squares are evenly distributed across the arm because of the alternating black and white checker pattern. The approach overcomes the problem of automatic detection with the existing MATLAB function *detectCheckerboardPoints,* which was initially tried. The algorithm behind this function searches for corners within in a checker pattern with a constant number of rows and columns[44], which was not adequate for the images in this study due to the tapered shape of the checkerboards as they were wrapped around the forearm. The newly developed method was not limited by this issue. A disadvantage, however, is that the new approach is less robust and requires manual input. One of the pre-processing steps is to manually overpaint the area of the checkerboard with skin color, which is necessary to create the mask that selects the region of the checkerboard. Also, the parameters for segmentation of the squares, including the threshold value for binarization and the structuring elements for erosion and dilation, need to be adjusted and optimized for each subject individually. For running larger amount of data, this can be a drawback. Another limitation is that there comes a certain inaccuracy with the segmentation of the squares during the step where the image is binarized. Due to a lower image quality, the hard edges of the black and white checkerboard squares become smoother in the image causing blurring of intensities. As a result, part of the segmented squares include more or less pixels than its actual area which could have slightly influenced the measured performance of the curvature correction. This could be improved by using a higher camera resolution, which increases image quality and reduces blurring. Moreover, notice that the mentioned inaccuracy concerns the validation method, and that it is unrelated to the accuracy of measuring wheals in a photograph of the SPT where a different segmentation method is used for wheal detection.

3.4.4 Future recommendations

Several recommendations for future research and further improvement of the placemat approach have already been briefly discussed above. Most importantly, it is needed to evaluate the performance of the placemat on a larger cohort and in subjects with different ages to test its accuracy for a variety in shapes of forearms. In addition, a suggestion is made for an even more adequate approximation of the shape of the arm: information about the complete arm contour was not available with the conventional barcode stickers, but can now be obtained with the placemat through segmentation of the arm. Instead of taking source key points at only three levels of the arm and further expanding them by straight vertical interpolation, it is possible to follow the actual arm contour by sampling many points on the detected contour. For each level of cross-section of the arm, the corresponding fixed points can be estimated with the diameter of the arm at that level and the ellipse formula. This way, vertical interpolation can be omitted and only interpolation in the horizontal direction would be needed. Furthermore, it is recommended to take a closer look at the segmentation process of the arm. During the development of the placemat approach it was seen that the Lab color space deals successfully with lighter shadows but performs less effectively in the presence of darker shadows, which is illustrated in Appendix E. This problem may be solved by controlling the light conditions when taking the photograph or using an alternative color space such as YCbCr, which also has a specific channel for illumination that can be discarded.[33] Moreover, the skin color may have an impact on the quality of the segmentation. As in this study the segmentation method was only tested on light skin color, it must be explored whether the method also works well for darker skin tones.

3.5 CONCLUSION

In this study a new placemat approach is proposed for correction of the arm curvature in photographs of the SPT, showing promising results. The performance of the correction differed per region on the forearm and decreased towards the sides of the arm. As it was found that most wheals in the SPT appear towards the midline of the forearm, the majority of the wheals are located in the region where the accuracy was considered to be acceptable. This was the first study that looked into the typical location of wheals on the forearm. Given the small sample size, this work provides a first indication of the performance of the placemat method. The performance may be further improved by optimizing the source and fixed map for a better estimation of the shape of the forearm, possibly leading to higher accuracies. Further research is needed to evaluate the placemat on more data, including subjects with different ages and arms' shapes. Overall, the placemat approach shows potential in obtaining uniform pixel dimensions in the image, which would overcome the problem of underestimation of wheal sizes due to the arm curvature. This way, it could contribute to more reliable wheal size measurement and is therefore a promising step towards standardizing and automating the conventional SPT reading method.

CHAPTER 4. Comparison of the barcode and placemat approach for arm curvature correction

4.1 INTRODUCTION

In the previous chapter, the performance of a white placemat with ArUco markers was studied to correct for the forearm curvature in photographs of the SPT. The arm curvature causes wheals that are located at the sides of the arm to be underestimated in size in the image. With a correction it is aimed to create equal dimensions in the image that match with the real-world dimensions so that wheals in the SPT can be measured accurately with digital photography. The proposed placemat was evaluated in six adult volunteers with a paper checkerboard wrapped around the forearm as a reference object of which the squares were measured before and after curvature correction to assess whether they returned to their original size after correction. Overall the squares became closer to their original size, however, it was demonstrated that the performance differed for different regions on the forearm and that the error increased towards the sides of the arm. The placemat was designed to improve the current method for arm curvature correction which is based on two small barcode stickers placed near the elbow cavity and wrist as introduced in Chapter 1. The purpose of the stickers is to estimate the shape and circumference of the forearm needed for the image transformation. With the barcode stickers several problems arise: first, the barcode lacks sufficient detection; second, because of its intermittent pattern with spaces in between the barcode stripes, the stripes do not fully run to the outer sides of the arm, meaning that the arm contour cannot be exactly detected; and third, the stickers are not always appropriately applied on the arm, for example, when the sticker is placed oblique or is not tightly soft-glued to the skin, thereby not adequately following the circumference of the forearm. As a consequence of these factors, the curvature correction is being inadequate.[36] Following on from Chapter 3, the aim of this chapter is to investigate whether the proposed placemat method actually results in a more accurate correction of the arm curvature and thus in more reliable assessment of wheal sizes. Therefore, this chapter compares the performance of the arm curvature correction with the barcode stickers and the placemat. It is hypothesized that the placemat approach will show a higher accuracy because of a more adequate source map of the forearm.

4.2 METHODS

4.2.1 Data acquisition

The same data was used as in the previous chapter, see Section 3.2.3, consisting of photographs of six healthy volunteers with a paper checkerboard of 25 mm^2 sized squares wrapped around the forearm. This time, the placemat as well as the barcode stickers in the photographs were used in order to compare both approaches. Each photograph was transformed with the placemat and with the barcode method separately.

4.2.2. Arm curvature correction

For the correction of the arm curvature using the placemat, the images were processed and transformed as described in the methods of Chapter 3. For the curvature correction with the barcode method, the existing MATLAB script written by Geessinck was used. The image processing steps are described below and illustrated in the flowchart of Figure 19.

Barcode detection and source map

First, the colored image was converted to grayscale after which the contrast was enhanced. The barcode was then detected with edge detection using the Canny method. Because this also resulted in detection of other features in the image, solely the two barcodes were selected through a series of sequential morphological operations, including closing, erosion and dilation. This resulted in separate segmented areas for all barcode stripes, which were then labeled and counted. Next, the centroid of each stripe was determined. For both barcodes, the first, middle and last centroid were used, which corresponded to the detected left and right points and a top point in between on the barcode stickers. These points formed the six source key points.

Fixed map

For the corresponding fixed map, it was known that the distance between the two beginnings of a barcode group was 10.15 mm in real life, where a barcode group was defined as each four stripes followed by the white space. The captured number of barcode groups was obtained by dividing the segmented and counted number of barcode stripes by four. By multiplying the number of barcode groups with 10.15 mm, the desired total length of the barcode could be calculated in millimeters, which was then converted to a distance in pixels. For this, a pixel-to-mm conversion factor was calculated as follows: manually two points were drawn that selected the start and end of a white space at the center of the sticker where the arm curvature was nihil; the distance between these points was calculated in pixels and because its real-life distance was known to be 4.4 mm, it could be calculated how many pixels corresponded with 1 mm. Now that the desired length of the barcode in pixels was known, a reference point was determined that was exact in between the left and right source key point. From this midpoint, half the desired length of the barcode was added in both left and right direction, resulting in the fixed key points. This was done for both barcode stickers.

Image transformation

The source and fixed key points were both interpolated to construct a complete source and fixed map consisting of a grid points that estimated the entire shape of the arm before and after the image transformation respectively. The left, middle and right key points were first interpolated with fifteen points that were not equally spread but became closer together towards the sides of the barcode. This way two ellipses were formed for the source map and two reconstructed straight lines for the fixed map. Next, further interpolation along the arm was applied but now in the vertical direction between the upper and lower fifteen points, including a hundred of points per interpolated line. The final step involved the computation of a B-spline transformation matrix with parameters that fitted the source and fixed map best. This matrix was applied onto the original image resulting in a transformed image that was corrected for the arm curvature.

Differences between placemat and barcode method

The basis of both the placemat and barcode method is similar; both corrections yield a transformation of the 3D shaped forearm to a 2D flat surface where the entire surface of the forearm is displayed in

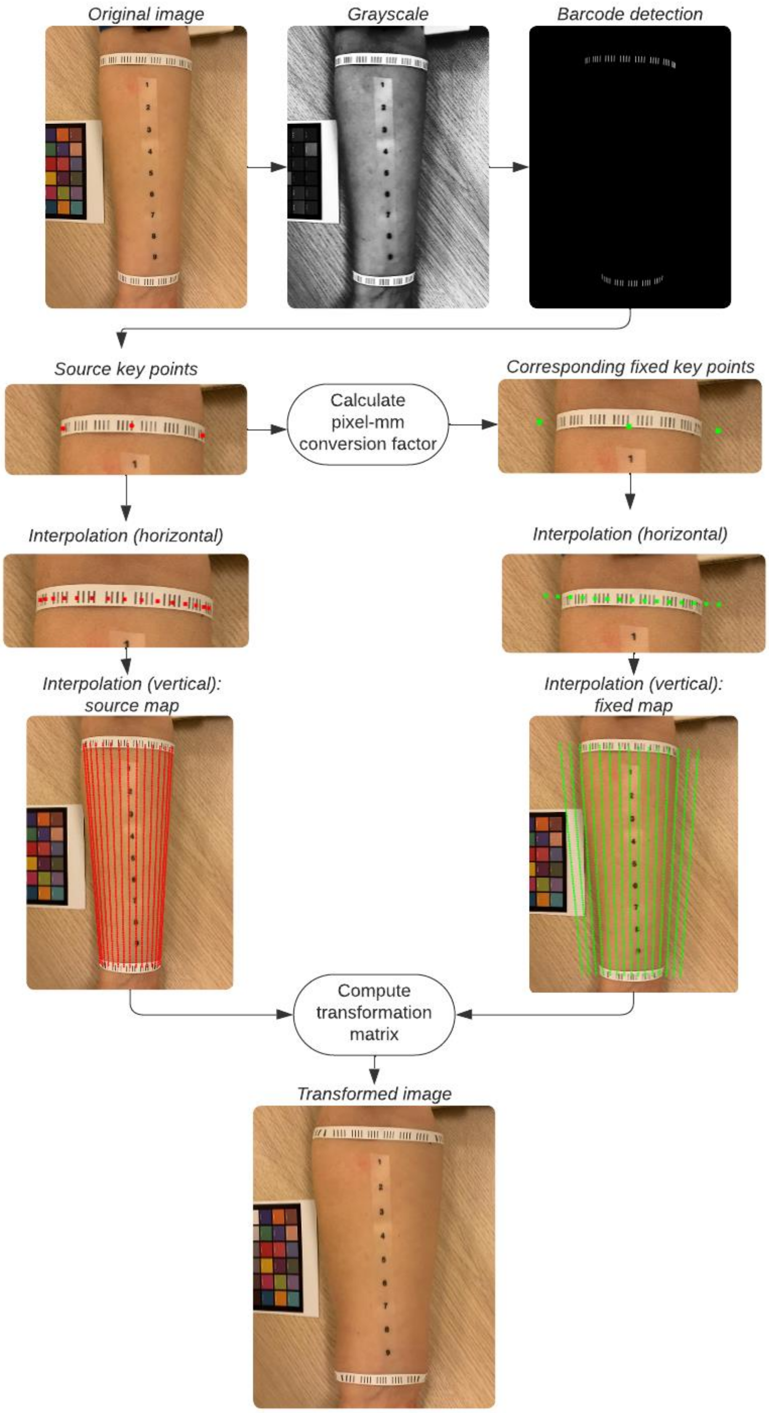

Original image
Grayscale
Barcode detection
Source key points
Calculate pixel-mm conversion factor
Corresponding fixed key points
Interpolation (horizontal)
Interpolation (horizontal)
Interpolation (vertical): source map
Interpolation (vertical): fixed map
Compute transformation matrix
Transformed image

Figure 19 Flowchart of the arm curvature correction using the barcode stickers

the same plane, as illustrated in Figure 9 of Chapter 3. How the source and fixed maps are created for the image transformation differs in four aspects. First, the left and right source key points are defined with the barcode by detection of the outermost stripes on the stickers, whereas the placemat approach uses segmentation of the forearm and determines its source points by intersection with a horizontal line between the ArUco markers. Second, the top point in between the left and right key points can with the barcode method be detected by the middle stripe on the sticker of which its position follows the arch of the curved sticker; when using the placemat this point is replaced by a calculated midpoint that is halfway and at the same height of the left and right key points. Third, the source map of the barcode method estimates the shape of the arm with a single cylinder, whereas for the placemat it consists of two cylinders that fits the upper and lower part of the forearm separately. Last, concerning the fixed key points, the desired position of these points is defined by a different approach for estimating the circumference of the arm. Whereas the barcode method relies on counting the number of detected stripes and knowing the distance between them, the arm circumference for the placemat is calculated with the formula of an ellipse using the diameter of the segmented arm. Except for these differences, the other processing steps are similar, including the interpolation of the source and fixed points and the final image transformation.

4.2.3. Validation and data analysis

The performance of the barcode method was validated in the same way as was done for the placemat in the previous chapter. In the corrected images, the checkerboard squares were segmented based on binarization of the image. The relative areas of the squares were measured, giving a percentage deviation from their original size, where value 1 corresponded to the original size. This was done for different regions of the forearm for which masks were created that selected the regions. A detailed description of the validation steps is provided in Section 3.2.3 of the previous chapter.

The data was analyzed creating a boxplot and colored visualizations of the measured relative areas of the checkerboard squares for the different regions of the arm. For comparison with the placemat method and uncorrected image, the boxplots and visualizations from the results in Chapter 3 were used.

4.3 RESULTS

Figure 20 shows an example of the three images in which the checkerboard squares are measured for evaluation of the arm curvature correction: the uncorrected image, the transformed image after curvature correction with the placemat, and the transformed image after correction using the barcode stickers. Appendix C1 provides both the uncorrected and corrected images for each subject individually.

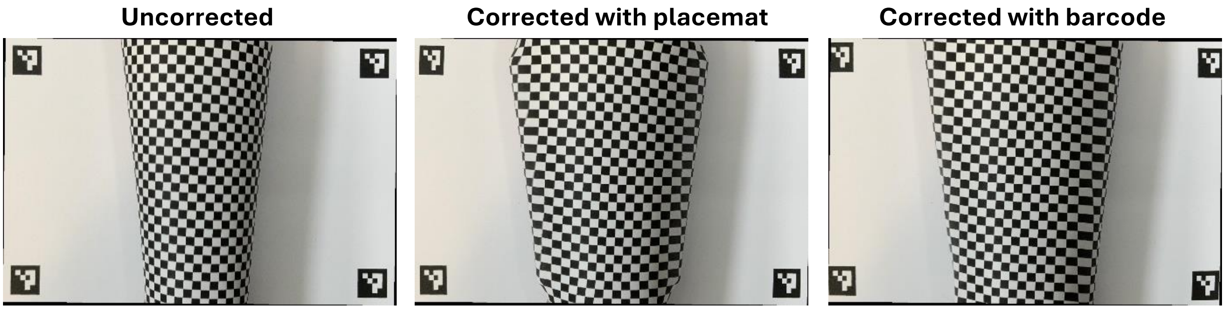


Figure 20 Example (subject 1) of the validation images before arm curvature correction (left) and after correction using the placemat (middle) and barcode (right).

Figure 22 visualizes for each subject the relative areas of the measured squares before and after correction. Their size is represented by the colors as given in the colorbar. Boxplots of the relative areas of all subjects are rendered in Figure 23 for the different regions on the forearm: <70%, 70-80%, 80-90% and >90%. These regions are clarified in Figure 24. Since the data is not normally distributed, the median and IQR are calculated and given in Table 5.

For the uncorrected images, the visualizations and boxplots show that the checkerboard squares become increasingly smaller towards the sides of the arm which demonstrates the effect of the forearm curvature. The correction with the placemat and barcode method both result in median areas that are closer to the original size of the squares. For the region <70% the results are similar. Towards the sides of the arm the deviation increases for both methods but is larger for the barcode method, including a larger error of the median areas as well as a larger variability. The deviation includes an overestimation as well as underestimation of the squares, of which the latter is seen in the outermost region.

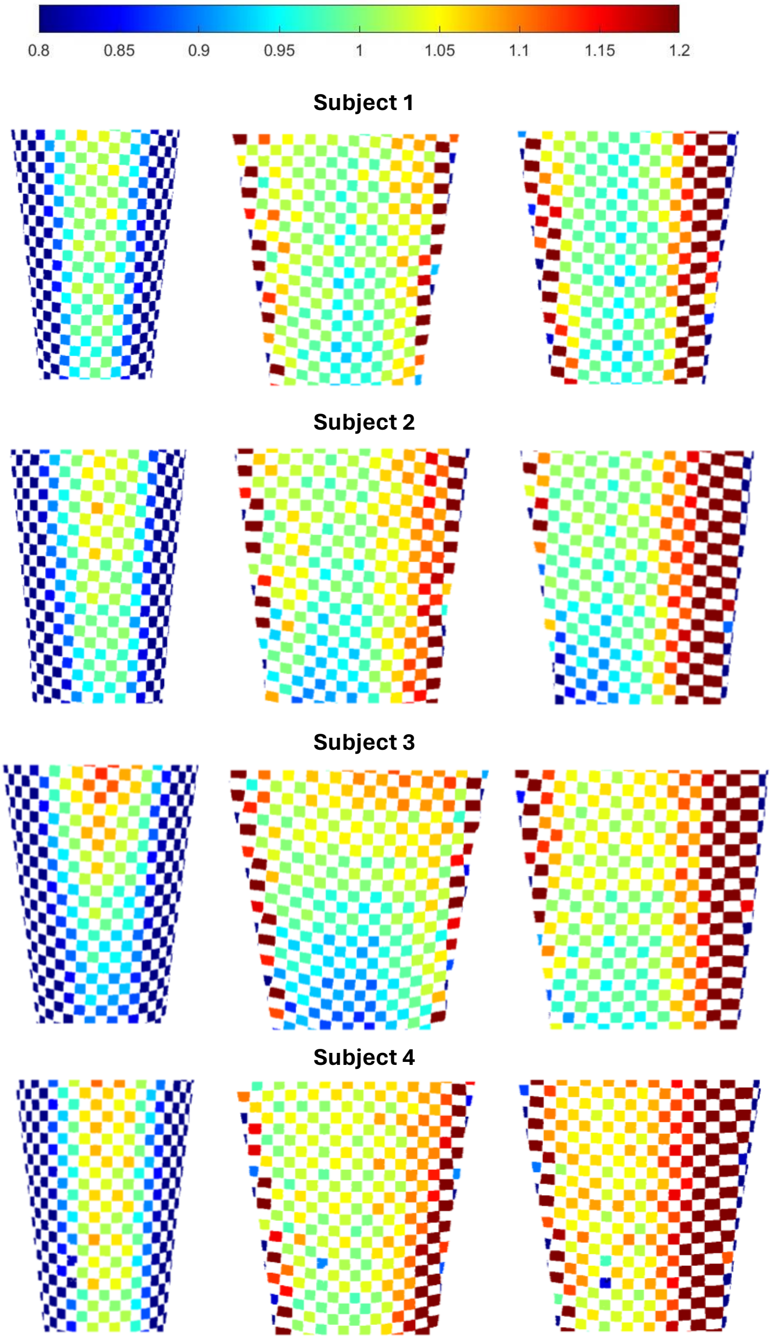
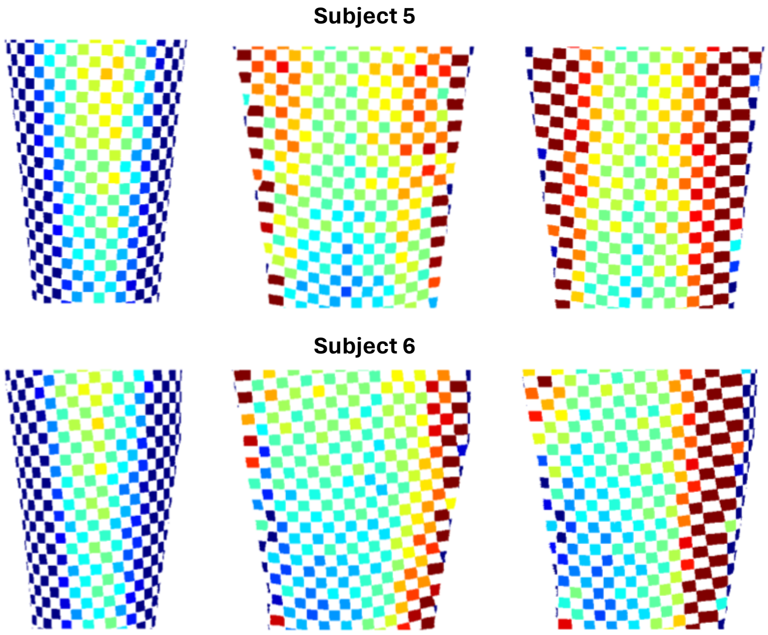


Figure 21 (Continued) Visualization of the relative areas of the checkerboard squares obtained before arm curvature correction (left) and after correction using the placemat (middle) and barcode (right). The colors represent the value of the relative areas as shown in the colorbar, where value 1 is considered to be the gold standard area. The regions presented in the images include 96% of the forearm.


Subject 5

Subject 6

Figure 22 (Continued).

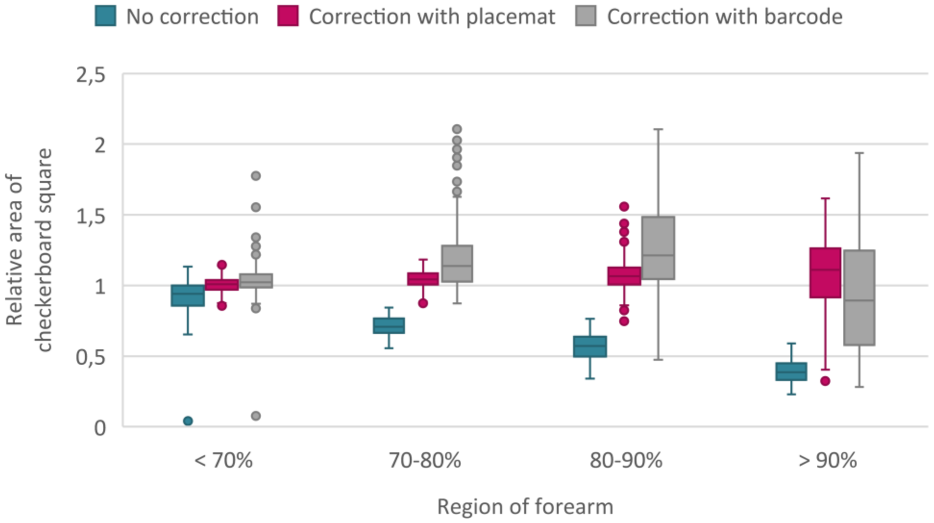


Figure 23 Distribution of the relative areas of the checkerboard squares on different regions of the forearm before arm curvature correction and after correction with the placemat and barcode, where value 1 is considered to be the gold standard. The plots present the data of all six subjects.

Table 5 Median and interquartile range (IQR) of the relative areas of the checkerboard squares before arm curvature correction and after correction with the placemat and the barcode.

	Median (IQR)		
Region on forearm	No correction	Correction with placemat	Correction with barcode
< 70%	0.94 (0.86 – 1.00)	1.01 (0.97 – 1.04)	1.02 (0.99 – 1.08)
70-80%	0.71 (0.66 – 0.77)	1.04 (1.01 – 1.09)	1.14 (1.03 – 1.28)
80-90%	0.57 (0.50 – 0.64)	1.07 (1.01 – 1.13)	1.21 (1.04 – 1.48)
> 90%	0.39 (0.33 – 0.45)	1.11 (0.92 – 1.26)	0.89 (0.58 – 1.25)

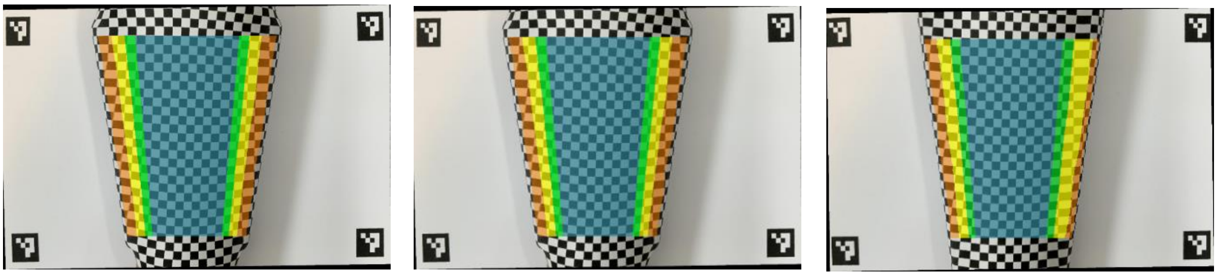


Figure 24 Illustration of the four defined regions on the forearm: <70% (blue), 70-80% (green), 80-90% (yellow) and >90% (orange). Example of the uncorrected image (left) and the corresponding regions on the arm after curvature correction with the placemat (middle) and barcode (right) for subject 2.

4.4 DISCUSSION

This study, focusing on the comparison between the arm curvature correction with the conventional method using barcode stickers and the new placemat method, showed that both methods performed similar for the region <70% of the forearm, and that the placemat showed a higher accuracy for the remaining regions towards the sides of the arm.

The better performance of the placemat can be explained by a more adequate construction of the source map, which leads to an improved transformation of the image and confirms the stated hypothesis. In Appendix C2 and C3 an overview of the source maps is given, from which it can be seen that the arm contour is followed more closely when using the placemat approach due to two factors. First, with the barcode method the outer barcode stripes were not detected sufficiently, thereby positioning the key points more inwards and misinterpreting the width of the arm. In contrast, with the placemat method the key points were correctly determined on the arm contour through adequate segmentation of the arm. Second, the fact that the placemat approach includes additional key points halfway the arm and estimates the shape of the arm with two cylinders instead of one resulted in a source map that more accurately conformed to the arm when compared to the barcode method. These factors explain why the corrections based on the barcode stickers show an increased number of larger squares, caused by a stronger overestimation of the curvature. Moreover, it explains the presence of more smaller squares in the outer region of the arm since the barcode method led to a significant number of squares that were missed by the source map and thus not included in the transformation.

Considering the clinical impact on wheal size measurement in the SPT and based on the present study, it is suggested that the barcode method yields sufficiently accurate measurements for the regions <70% of the forearm whereas this is up to 80% with the placemat approach. This is based on the assumption that a 10% error is acceptable for the 25 mm^2 checkerboard squares in this study, as reasoned in the discussion of Chapter 3. When relating these accuracies to the regions on the forearm where the wheals typically appear, as described in Chapter 3, it can be inferred that with the barcode method circa 14% of the wheals would not be measured accurately which is doubled compared to the circa 6% with the placemat method. This provides support for the use of the placemat over the barcode stickers. Moreover, the placemat is more robust since it is not prone to misplacement; it overcomes the problem of oblique or not tightly glued barcode stickers. Another advantage is that it leaves more space on the arm for placing the allergen droplets of the test, which is especially beneficial in small children's arms.

This study had the advantage of testing both the conventional and the placemat method for curvature correction in the exact same photograph so that the results were not influenced by other variables and direct comparison was possible. A limitation, however, is that the measured checkerboard squares may slightly differ from their actual size in the image due to a certain inaccuracy coming with the segmentation of the squares, which has been clarified in the discussion of Chapter 3. Another limitation is the small sample size that was used. In future research, a larger number of subjects with different ages and arms' shapes should be included for further testing the performance of the placemat to conclude whether the placemat actually outperforms the conventional barcode method.

4.5 CONCLUSION

This study compared the conventional barcode method to the new placemat method for correction of the arm curvature in photographs of the SPT. The placemat showed a better performance, especially for the regions towards the sides of the arm a higher accuracy is achieved, showing more uniformity of the pixel dimensions in the image. With this improved performance, a substantially larger number of wheals on the forearm could be measured more accurately compared to when the barcode method is applied. Moreover, the placemat has several practical advantages over the conventional used barcode stickers. Given the limited sample size in this study, more data of subjects with varied ages and arms' shapes must be acquired to draw firm conclusions on whether the placemat outperforms the barcode method. Based on the present study, the placemat shows potential in optimized SPT reading with digital photography, ultimately contributing to more reliable wheal size measurement and standardization of the manual SPT.

CHAPTER 5. General discussion and conclusion

The research presented in this thesis focused on standardizing SPT reading in allergy diagnostics using digital photography. Specifically, it was aimed to optimize a correction for the camera perspective and arm curvature to obtain uniform pixel dimensions in the image, so that wheal size measurement in the photograph matches with their real-life dimensions to assure accurate assessment. A basic placemat with four ArUco markers was suggested incorporating a method for both the perspective and curvature correction based on image transformations. The performances were validated with a checker pattern of known sizes as reference object. For the perspective correction, a high accuracy was obtained when taking the photograph from smaller as well as larger camera angles. From the arm curvature correction using the placemat it could be observed that its accuracy decreased towards the sides of the arm; an acceptable error was measured for up to 80% of the forearm which covered most of the wheals in a SPT. This accuracy was found to be higher when compared to the curvature correction performed with the conventional barcode stickers.

When considering the accuracies found, it is important to keep in mind the clinical relevance and the feasibility of the approach. A popular modern technique to standardize SPT reading includes 3D imaging technologies.[22–24,29,30] Although some of these technologies seem to achieve higher accuracies on wheal size measurement compared to digital photography, they come with a significant limitation. This limitation involves the need for more complex measurement setups or expensive equipment, which impedes their implementation in the clinic. Therefore, it should be questioned whether these more advanced technologies are necessary for a reliable SPT or that sufficient accuracy can be achieved with photography, which seems to be promising based on the research in this thesis. It can, however, be debated what accuracy is required since a generally accepted error for SPT measurement is not given.[54] Moreover, the accuracy of the conventional SPT is not known due to the lack of a gold standard because the real wheal area is not known, which makes comparison against the manual SPT difficult. It can, however, be stated that the proposed placemat method provides an objective and automatic approach for assessment of the SPT, thereby eliminating the factors that cause the inaccuracies in the manual SPT such as the intra- and interobserver variability. It must be realized that accurate and reliable SPT reading is determined by other factors than the performance of the correction for camera perspective and arm curvature as well. This includes successfully detecting the wheal in the image. In the Deventer Hospital, an artificial intelligence model has been developed that automatically segments and computes wheal areas in a photograph. The performance of the model, which at the moment is being trained on a larger dataset for further improvement and in the future will be tested on different skin tones, greatly contributes to the outcome of the wheal measurement. At the end, obtaining reliable SPT results does not only depend on the test reading but is also influenced by variable factors that are involved in carrying out the test. For example, the reaction to the allergen depends on the location of the pricks; the back is more reactive than the forearm and the area near the elbow cavity is more reactive than the wrist.[55,56] Additionally, it has been demonstrated that the injection depth of the prick has an influence on the wheal size.[27] Also, the type of SPT technique and testing device introduces a variability in outcome.[57,58] For reliable SPT outcomes, it is crucial not only to standardize the reading part but also to standardize the performance of the test.

The aim to optimize the correction for camera perspective and arm curvature is achieved. The placemat overcomes the limitations of the currently used ArUco marker and barcode stickers and shows a better performance. Moreover, it has several advantages regarding its implementation in the clinic, including that it is a low-cost and accessible product and convenient in use. When implemented, the placemat needs to be produced in a material that meets the requirements of hygiene and durability; for example, instead of sturdy paper the placemat can be laminated with a matte coating that does not reflect light. A minor drawback is that the placemat approach is purposed for situations where the SPT is performed on the forearm. In practice, however, the SPT in some cases is performed on the skin of the back. Particularly in younger children the back can be preferred as it prevents the child from scratching.[17,59] In comparison with the forearm, the presence and influence of curvature is less pronounced for the back. However, camera perspective will still remain an issue to be accounted for. If digital photography with use of the placemat will be implemented in the clinic for standardization of the SPT, it should be considered to test solely on the forearm of patients. All other methods suggested in literature are also only applicable for wheal measurements on the forearm.

For future research, it is needed to further evaluate the performance of the placemat on a larger study population with varying ages and arms' shapes. Furthermore, it is recommended to improve the performance of the correction for the arm curvature. Higher accuracies could be achieved by an improved source and fixed map that better estimate the shape of the forearm, which could be attempted with the suggestions discussed in Chapter 3. In addition, it is important to focus on obtaining uniform pixel dimensions in the direction along the arm and account for the fact that the upper part of the forearm is closer to the camera than the lower part because of the varying height of the arm. Another effect of this issue is that it complicates the conversion of pixels to millimeters for wheal measurement. Calculating the conversion factor could be done by using a reference object of known size in the image, such as the distance between the ArUco markers of the placemat. For this concept, however, it is essential that the wheals are situated in the same plane as the placemat, which is not the case due to the inclination of the arm. To ensure correct wheal areas in mm^2, this aspect must be further looked into. When measuring the wheal size in HEWS, a pixel-to-mm conversion factor is not needed as this outcome measure consists of a ratio which is dimensionless and could be calculated by dividing the wheal area of the allergen and histamine in pixels, on the condition that the pixel dimensions are uniform along the arm.

Concluding, the study presented in this thesis suggested the use of a basic placemat with ArUco markers to correct for the camera perspective as well as the arm curvature in photographs of the SPT. These corrections are important for accurately measuring wheal sizes in the image that correspond to the real-world dimensions of the wheals. The perspective correction using the placemat showed a high accuracy. The curvature correction using the placemat showed a better performance than the conventional used barcode stickers, including an acceptable accuracy for the inner part of the forearm, where most of the wheals in the SPT appear, and an increasing error towards the outer sides of the arm. Based on the present studies, the placemat method yields promising results in achieving accurate wheal size measurements using digital photography. This way, it addresses the inaccuracies that come with the manual SPT reading and shows considerable potential in obtaining reproducible and reliable SPT results. As the placemat combined with digital photography is an accessible and low-cost technique, clinical implementation offers a feasible way to standardize SPT reading. Before implementing, future research should focus on further evaluation of the placemat in a larger dataset consisting of varied ages and arms' shapes. Also, it is recommended to further optimize the performance of the arm curvature correction and to account for the inclination of the arm with respect to the placemat.

References

1. Custovic, A. *Middleton's Allergy Essentials*. (Elsevier, 2017). doi:10.1016/C2013-0-18596-0.
2. Warren, C. M., Jiang, J. & Gupta, R. S. Epidemiology and Burden of Food Allergy. *Curr Allergy Asthma Rep* **20**, 6 (2020).
3. Tang, M. L. K. & Mullins, R. J. Food allergy: is prevalence increasing? *Intern Med J* **47**, 256–261 (2017).
4. Thomsen, S. F. Epidemiology and natural history of atopic diseases. *Eur Clin Respir J* **2**, 24642 (2015).
5. Hammer-Helmich, L. *et al.* Mental health associations with eczema, asthma and hay fever in children: a cross-sectional survey. *BMJ Open* **6**, e012637 (2016).
6. Cummings, A. J., Knibb, R. C., King, R. M. & Lucas, J. S. The psychosocial impact of food allergy and food hypersensitivity in children, adolescents and their families: a review. *Allergy* **65**, 933–945 (2010).
7. Oh, H., Koyanagi, A., DeVylder, J. & Stickley, A. Seasonal Allergies and Psychiatric Disorders in the United States. *Int J Environ Res Public Health* **15**, 1965 (2018).
8. Fox, M. *et al.* Health sector costs of self-reported food allergy in Europe: a patient-based cost of illness study. *Eur J Public Health* **23**, 757–762 (2013).
9. Abbas, M., Moussa, M. & Akel, H. Type I Hypersensitivity Reaction. *StatPearls* https://www.ncbi.nlm.nih.gov/books/NBK560561/ (2023).
10. Brand, P. *Werkboek Kinderallergologie. Anamnese En Lichamelijk Onderzoek*. (VU University Press, 2014).
11. Kumar, P. & Clark, M. *Kumar and Clark's Clinical Medicine*. (Saunders Elsevier, 2012).
12. Amin, K. The role of mast cells in allergic inflammation. *Respir Med* **106**, 9–14 (2012).
13. Rijkers, G., Dorst, D. & de Haan, N. *Werkboek Kinderallergologie. Allergie En Het Immuunsysteem*. (VU University Press, 2014).
14. Wheatley, L. M. & Togias, A. Allergic Rhinitis. *New England Journal of Medicine* **372**, 456–463 (2015).
15. Muraro, A. *et al.* EAACI Food Allergy and Anaphylaxis Guidelines: diagnosis and management of food allergy. *Allergy* **69**, 1008–1025 (2014).
16. Muthupalaniappen, L. & Jamil, A. Prick, patch or blood test? A simple guide to allergy testing. *Malaysian Family Physician* **16**, 19–26 (2021).
17. de Jong, N. *Werkboek Kinderallergologie. Huidtests.* (VU University Press, 2014).
18. Heinzerling, L. *et al.* The skin prick test – European standards. *Clin Transl Allergy* **3**, (2013).
19. van der Valk, J. P. M. *et al.* Measurement and interpretation of skin prick test results. *Clin Transl Allergy* **6**, 8 (2015).
20. Frati, F. *et al.* The skin prick test. *J Biol Regul Homeost Agents* **32**, 19–24 (2018).
21. Gorris, S. *et al.* Reduced intra-subject variability of an automated skin prick test device compared to a manual test. *Allergy* **78**, 1366–1368 (2023).
22. Pineda, J. *et al.* Robust automated reading of the skin prick test via 3D imaging and parametric surface fitting. *PLoS One* **14**, (2019).
23. Justo, X., Díaz, I., Gil, J. J. & Gastaminza, G. Prick test: evolution towards automated reading. *Allergy: European Journal of Allergy and Clinical Immunology* vol. 71 1095–1102 Preprint at https://doi.org/10.1111/all.12921 (2016).
24. Justo, X., Diaz, I., Gil, J. J. & Gastaminza, G. Medical device for automated prick test reading. *IEEE J Biomed Health Inform* **22**, 895–903 (2018).
25. Svelto, C., Matteucci, M., Pniov, A. & Pedotti, L. Skin prick test digital imaging system with manual, semiautomatic, and automatic wheal edge detection and area measurement. *Multimed Tools Appl* **77**, 9779–9797 (2018).

26. dos Santos, R. V., Mlynek, A., Lima, H. C., Martus, P. & Maurer, M. Beyond flat weals: validation of a three-dimensional imaging technology that will improve skin allergy research. *Clin Exp Dermatol* **33**, 772–775 (2008).
27. De Jong, J. The effect of the injection depth and injection volume on the wheal size in the skin prick test. (University of Twente & Deventer Ziekenhuis, 2023).
28. Pena, J. C., Pacheco, J. A. & Marrugo, A. G. Skin prick test wheal detection in 3D images via convolutional neural networks. in *2021 IEEE 2nd International Congress of Biomedical Engineering and Bioengineering, CI-IB and BI 2021* (Institute of Electrical and Electronics Engineers Inc., 2021). doi:10.1109/CI-IBBI54220.2021.9626125.
29. Marrugo, A. G. *et al.* Toward an automatic 3D measurement of skin wheals from skin prick tests. in *Dimensional Optical Metrology and Inspection for Practical Applications VIII* (eds. Zhang, S. & Harding, K. G.) 3 (SPIE, 2019). doi:10.1117/12.2519034.
30. den Blanken, M. D. *et al.* Quantification of cutaneous allergic reactions using 3D optical imaging: A feasibility study. *Skin Research and Technology* **26**, 67–75 (2020).
31. Vieira dos Santos, R., Titus, R. G. & Cavalcante Lima, H. Objective evaluation of skin prick test reactions using digital photography. *Skin Research and Technology* **13**, 148–153 (2007).
32. Almeida, A. L. M. *et al.* Objective evaluation of immediate reading skin prick test applying image planimetric and reaction thermometry analyses. *J Immunol Methods* **487**, 112870 (2020).
33. Bulan, O. Improved wheal detection from skin prick test images. in (eds. Niel, K. S. & Bingham, P. R.) 90240J (2014). doi:10.1117/12.2038442.
34. Geessinck, M. S. M. An automatic segmentation method and prediction model for skin prick test results. MSc Assignment. (University of Twente & Deventer Ziekenhuis, 2021).
35. Zankevich, A. How to unwrap wine labels programmatically. (2018).
36. Krakers, S. Validation of the forearm curvature correction and further automatization of the skin prick test examination using digital photography. (University of Twente & Deventer Ziekenhuis, 2022).
37. Jakka, S. Skin prick testing in children. *Karnataka Pediatric Journal* **35**, 67 (2021).
38. Taligent, Inc. Cameras: Projection from 3D to 2D. https://root.cern/TaligentDocs/TaligentOnline/DocumentRoot/1.0/Docs/books/GS/GS_261.html (1995).
39. Soycan, A. & Soycan, M. Perspective correction of building facade images for architectural applications. *Engineering Science and Technology, an International Journal* **22**, 697–705 (2019).
40. The MathWorks, Inc. Matrix representation of geometric transformations: 2D projective transformations. https://nl.mathworks.com/help/images/matrix-representation-of-geometric-transformations.html (2023).
41. OpenCV. Detection of ArUco markers. https://docs.opencv.org/4.x/d5/dae/tutorial_aruco_detection.html (2023).
42. Narayanan, A. & Bijlani, K. A practical solution to checker pattern detection based on contour validation. in *International Conference on Image Processing, Computer Vision, and Pattern Recognition* (Las Vegas, USA, 2012).
43. Dao, V. N. & Sugimoto, M. A Robust Recognition Technique for Dense Checkerboard Patterns. in *2010 20th International Conference on Pattern Recognition* 3081–3084 (IEEE, 2010). doi:10.1109/ICPR.2010.755.
44. Geiger, A., Moosmann, F., Car, O. & Schuster, B. Automatic camera and range sensor calibration using a single shot. in *2012 IEEE International Conference on Robotics and Automation* 3936–3943 (IEEE, 2012). doi:10.1109/ICRA.2012.6224570.
45. Suchecki, R., Lipiec, A., Wierzejski, W., Zaworski, W. & Grzanka, A. Computer planimetry in allergology skin prick tests. in 320 (2005). doi:10.1117/12.610753.
46. Zoltie, T., Blome-Eberwein, S., Forbes, S., Theaker, M. & Hussain, W. Medical photography using mobile devices. *BMJ* e067663 (2022) doi:10.1136/bmj-2021-067663.

47. Alam, F., Rahman, S. U., Khusro, S., Ullah, S. & Khalil, A. Evaluation of Medical Image Registration Techniques Based on Nature and Domain of the Transformation. *J Med Imaging Radiat Sci* **47**, 178–193 (2016).
48. Bhedi, A., Saxena, A. K., Gadani, R. & Patel, R. Digital Photography and Transparency-Based Methods for Measuring Wound Surface Area. *Indian Journal of Surgery* **75**, 111–114 (2013).
49. Wu, Z., Lan, T., Wang, J., Ding, Y. & Qin, Z. Medical Image Registration Using B-Spline Transform. *International Journal of Simulation: Systems, Science & Technology* (2016) doi:10.5013/IJSSST.a.17.48.01.
50. van Uum, E. Improving wheal area determination in skin prick tests: adjustment of the arm curvature for improved accuracy. (University of Twente & Deventer Ziekenhuis, 2023).
51. Roullot, E., Autegarden, J.-E., Devriendt, P. & Leynadier, F. Segmentation of Erythema from Skin Photographs for Assisted Diagnosis in Allergology. in 754–763 (2005). doi:10.1007/11552499_83.
52. Zheng, X., Lei, Q., Yao, R., Gong, Y. & Yin, Q. Image segmentation based on adaptive K-means algorithm. *EURASIP J Image Video Process* **2018**, 68 (2018).
53. Rackauckas, C. So you want to know the circumference of an ellipse?
54. Nevis, I. F., Binkley, K. & Kabali, C. Diagnostic accuracy of skin-prick testing for allergic rhinitis: a systematic review and meta-analysis. *Allergy, Asthma & Clinical Immunology* **12**, 20 (2016).
55. NELSON, H., KNOETZER, J. & BUCHER, B. Effect of distance between sites and region of the body on results of skin prick tests. *Journal of Allergy and Clinical Immunology* **97**, 596–601 (1996).
56. Voorhorst, R. Perfection of Skin Testing Technique. *Allergy* **35**, 247–250 (1980).
57. Antunes, J., Borrego, L., Romeira, A. & Pinto, P. Skin prick tests and allergy diagnosis. *Allergol Immunopathol (Madr)* **37**, 155–164 (2009).
58. Buyuktiryaki, B. *et al.* Optimizing the Use of a Skin Prick Test Device on Children. *Int Arch Allergy Immunol* **162**, 65–70 (2013).
59. British Society for Allergy and Clinical Immunology. Paediatric Allergy Skin Prick Testing. Preprint at (2019).

Appendices

A. MATLAB scripts

A1. Correction and validation of the camera perspective

```
%% Perspective transformation and validation
% This script corrects the photograph for camera perspective using a projective transformation. The last sections provide the
code for validating the performance of the perspective correction through measurement of checkerboard size.
clear all
close all
clc

image = "...";
im_original = imread('filename.jpg');    % non-transformed image
im_fotoshop = imread('filename.jpg'); % with removed background (and checkerboard)

%% Find markers and calculate centers
% In Python, this section can be replaced by detecting the markers based on the function 'aruco.detectMarkers'

% Binarize image and find marker areas
im_gray = rgb2gray(im_fotoshop);                 % convert to grayscale
im_bi2 = ~im2bw(im_gray,0.40);                   % create binary image with thresholding to find markers
im_squares = imfill(im_bi2, "holes");            % imfill creates filled squares
im_label = bwlabel(im_squares);                  % label the selected areas/objects
im_label = bwareaopen(im_label,100);             % remove small objects fewer than 100 pixels. Markers (> 100 pixels) will remain.
stat = regionprops(im_label,'boundingbox'); % bounding box enclosing the markers

% Calculate centers of ArUco markers
centerpoints_markers = [];
info_marker = [];
for marker_number = 1 : 4
   info_marker(marker_number,:) = stat(marker_number).BoundingBox;
   rectangle('position',info_marker(marker_number, :),'edgecolor','r','linewidth',2);
   centerpoint = [info_marker(marker_number,1)+info_marker(marker_number,3)/2,
info_marker(marker_number,2)+info_marker(marker_number,4)/2];
   centerpoints_markers(marker_number, :) = centerpoint;
end

%% Source points
% Sort markers
[centerpoint1,i] = sort(centerpoints_markers(:,2));
centerpoint2 = centerpoints_markers(:,1);
centerpoint2 = centerpoint2(i);
[centerpointtop, i1] = sort(centerpoint2(1:2,1));
[centerpointbottom, i2] = sort(centerpoint2(3:4,1));
centerpoint2 = [centerpointtop;centerpointbottom];
i = [i1;i2+2];
centerpoint1 = centerpoint1(i);
centerpoints_markers = [centerpoint2,centerpoint1];

% Ordered
left_up_corner = [centerpoints_markers(1,1), centerpoints_markers(1,2)];    % marker 1
right_up_corner = [centerpoints_markers(2,1), centerpoints_markers(2,2)];   % marker 2
left_down_corner = [centerpoints_markers(3,1), centerpoints_markers(3,2)];  % marker 3
right_down_corner = [centerpoints_markers(4,1), centerpoints_markers(4,2)]; % marker 4
coordinates_box = [left_up_corner;right_up_corner;left_down_corner;right_down_corner]; % Source/moving points

% Show found centers
figure, imshow(im_original); hold on;
scatter(coordinates_box(:,1),coordinates_box(:,2),'r.'); hold off
```

```
%% Fixed points
% Calculate ratio of rectangle formed by the markers (depends on orientation of the placemat in image)
long = 237;        % known length between centers in mm
short = 150;       % known width in mm
a = norm(centerpoints_markers(1,:) - centerpoints_markers(2,:));  % width in pixels
b = norm(centerpoints_markers(1,:) - centerpoints_markers(3,:));  % height in pixels
if a > b
   f = short/long;   % ratio of rectangle
else
   f = long/short;   % ratio
end

% Calculate position of fixed points
l = norm(coordinates_box(1,:) - coordinates_box(2,:)); % distance between marker 1&2
ny = coordinates_box(1,2) + l.*f;        % fixed y-coordinate of marker 3&4
fxd1 = coordinates_box(1,:);             % marker 1: remains its position
fxd2 = [coordinates_box(1,1)+l coordinates_box(1,2)]; % marker 2
fxd3 = [coordinates_box(1,1) ny];      % marker 3
fxd4 = [coordinates_box(2,1) ny];      % marker 4
fixed_points = [fxd1;fxd2;fxd3;fxd4]; % Fixed points

% Show fixed points
figure; hold on;
scatter(coordinates_box(:,1),coordinates_box(:,2),'g.'); hold off

%% Projective transformation
tform = fitgeotform2d(coordinates_box,fixed_points,'projective'); % find optimal transformation matrix
im_T = imwarp(im_original,tform); % apply transformation
figure, imshow(im_T)

%% Validation - Convert pixels to mm
centerpoints_T = transformPointsForward(tform,coordinates_box); % position of fixed points in transformed image

% Pixels to mm
if a > b
   length_x = 237; % mm
   length_y = 150; % mm
else
   length_x = 150; % mm
   length_y = 237; % mm
end
Pixels_x1 = norm(centerpoints_T(2,:) - centerpoints_T(1,:)); % pixels between two centers in x-direction
Pixels_x2 = norm(centerpoints_T(4,:) - centerpoints_T(3,:));
pixels_x = (Pixels_x2+Pixels_x1)./2;
Pixels_y1 = norm(centerpoints_T(3,:) - centerpoints_T(1,:)); % pixels between two centers in y-direction
Pixels_y2 = norm(centerpoints_T(4,:) - centerpoints_T(2,:));
pixels_y = (Pixels_y2+Pixels_y1)./2;
pixel_to_mm_x = length_x/pixels_x;    % mm per pixel in x direction
pixel_to_mm_y = length_y/pixels_y;    % mm per pixel in y direction
oppPixel_mm = pixel_to_mm_y*pixel_to_mm_x;  % mm^2 per pixel

%% Validation - Detect checkerboardpoints
% Detect corners of checker pattern
[imagePoints_c,boardSize_c] = detectCheckerboardPoints(im_T);

figure, imshow(im_T); hold on;
scatter(imagePoints_c(:,1), imagePoints_c(:,2),'y.'); hold off

%% Validation - Calculate x-length, y-length and area of each checkerboard square
Name = "filename"; % filename of Excel file that will be created

% Order all detected corners, from left to right and then from top to bottom
```

```
[imagePoints_c1,i] = sort(imagePoints_c(:,2));
imagePoints_c2 = imagePoints_c(:,1);
imagePoints_c2 = imagePoints_c2(i);
imagePoints_c2_ordered = [];
i2 = [];
for a = 0:(boardSize_c(2)-2)
    b = 1+(boardSize_c(1)-1)*a;
    c = (boardSize_c(1)-1)*(a+1);
    [imagepoints, i1] = sort(imagePoints_c2(b:c));
    imagePoints_c2_ordered = [imagePoints_c2_ordered; imagepoints];
    i2 = [i2;(i1+(((boardSize_c(1)-1))*a))];
end
imagePoints_c1 = imagePoints_c1(i2);
imagePoints_c = [imagePoints_c2_ordered, imagePoints_c1];

% Pre-allocate arrays
x_length_top = [];
x_length_bottom = [];
y_length_left = [];
y_length_right = [];
areas = [];

% Run through all corners and calculate the associating lengths and area of each square
rownumber = 0;
for k = 0:(boardSize_c(1)-1):(((boardSize_c(1)-1)*(boardSize_c(2)-2))-1)
    rownumber = rownumber+1;
    colnumber = 0;
    for i = 0:(boardSize_c(1)-3)
        colnumber = colnumber+1;
        tl = (k+1)+i;                           % top left corner of square
        tr = (k+2)+i;                           % top right
        bl = (k+(boardSize_c(1)-1)+1)+i;  % bottom left
        br = (k+(boardSize_c(1)-1)+2)+i;  % bottom right
        str = sprintf('Row %d and col %d', rownumber, colnumber); % create string to correctly name each variable

        x = [imagePoints_c(tl,1); imagePoints_c(tr,1); imagePoints_c(br,1); imagePoints_c(bl,1)]; % x-coordinates of vertices of square
        y = [imagePoints_c(tl,2); imagePoints_c(tr,2); imagePoints_c(br,2); imagePoints_c(bl,2)]; % y-coordinates
        [~,A] = boundary(x,y); % store areas in table

        % x-length
        Xtop = norm(imagePoints_c(tl,:) - imagePoints_c(tr,:));
        x_length_top = [x_length_top; table(Xtop*pixel_to_mm_x,'rowname',{str})];
        filenamexlength = append('xlength_', Name, '.xlsx');
        writetable(x_length_top, filenamexlength,'sheet',1,'Range','A1','WriteRowNames',true);
        Xbottom = norm(imagePoints_c(bl,:) - imagePoints_c(br,:));
        x_length_bottom = [x_length_bottom; table(Xbottom*pixel_to_mm_x,'rowname',{str})];
        writetable(x_length_bottom, filenamexlength,'sheet',1,'Range','D1','WriteRowNames',true);

        % y-length
        Yleft = norm(imagePoints_c(tl,:) - imagePoints_c(bl,:));
        y_length_left = [y_length_left; table(Yleft*pixel_to_mm_y,'rowname',{str})];
        filenameylength = append('ylength_', Name, '.xlsx');
        writetable(y_length_left, filenameylength,'sheet',1,'Range','A1','WriteRowNames',true);
        Yright = norm(imagePoints_c(tr,:) - imagePoints_c(br,:));
        y_length_right = [y_length_right; table(Yright*pixel_to_mm_y,'rowname',{str})];
        writetable(y_length_right, filenameylength,'sheet',1,'Range','D1','WriteRowNames',true);

        % areas
        areas = [areas; table(A*oppPixel_mm,'rowname',{str})];
        filenameareas = append('areas_', Name, '.xlsx');
        writetable(areas,filenameareas,'Sheet',1,'Range','A1','WriteRowNames',true)
    end
```

```
end
```

A2. Localizing wheals in photographs of SPT

```
%% Location of wheals on arm
% This script is written to determine the position of wheals on the forearm. The input is the original photo of the wheals.
% The output returns the relative distance of the wheal with respect to the midline of the arm.
clear all
close all
clc

%% Prepare files
% Load all photos
path = 'C:\...';    % Folder containing the data
photos = dir(path);
photos(1:2) = [];

% Put photos upright
for k = 1:length(photos)
filename = photos(k).name;
photo = imread(filename);
s = size(photo);
if   s(1,1) < s(1,2)
    photo = imrotate(photo,-90);  % Rotate photo if needed
else photo = photo;
end

%% Draw points for arm contour and define midline
% Define arm contour by four manually clicked points
imshow(photo); set(gcf, 'Position', get(0, 'Screensize'));  % Open image on full screen
title('Click 4 corners of armcontour')
pt1 = drawpoint;  % Click four points: 2 right below the elbow cavity and 2 right above the wrist
pt2 = drawpoint;
pt3 = drawpoint;
pt4 = drawpoint;
corners = [pt1.Position;pt2.Position;pt3.Position;pt4.Position];
corners_reshaped = reshape(corners,[2,4]);  % Four points

% Define midline of arm by two midpoints
mid_coord = mean(corners_reshaped);
mid_upper = [mid_coord(1,1),mid_coord(1,3)];
mid_lower = [mid_coord(1,2),mid_coord(1,4)];
mid_points = [mid_upper;mid_lower];
mid_target = [mid_points(1,1),mid_points(2,2)];  % Target point for a vertical straight midline: [X_upper,Y_lower]

%% Align midline of the arm vertically straight
% Calculate angle of rotation
overstaand = abs(mid_lower(1,1)-mid_target(1,1));
aanliggend = abs(mid_upper(1,2)-mid_target(1,2));
angle = atand(overstaand/aanliggend);

if mid_target(1,1) < mid_lower(1,1)
  R = [cosd(angle) -sind(angle); sind(angle) cosd(angle)];
else
  R = [cosd(-angle) -sind(-angle); sind(-angle) cosd(-angle)];
end
r_mid_points = R*mid_points';  % Rotate the midpoints on midline of arm
r_corners = R*corners';              % Rotate the four points of arm contour

%% Define arm contour with linear formula
% Left arm contour: formula of line
a1 = (r_corners(2,3)-r_corners(2,1)) / (r_corners(1,3)-r_corners(1,1));  % Slope: delta y divided by delta x
```

```
b1 = r_corners(2,1) - a1*r_corners(1,1);                              % b = y-ax

% Right arm contour
a2 = (r_corners(2,4)-r_corners(2,2)) / (r_corners(1,4)-r_corners(1,2));
b2 = r_corners(2,2) - a2*r_corners(1,2);

%% Draw wheal points and calculate relative distance to midline
% Rotate points and lines
f = figure;
imshow(photo); set(gcf, 'Position', get(0, 'Screensize')); title('Click on wheals'); hold on
plot(mid_points(:,1),mid_points(:,2),'g'); hold off

wheals = [];  % Clear wheal positions of previous photo
r_wheal = [];
x = [];
for i = 1:25
   wheal = drawpoint;  % Click wheal point
   wheals(:,i) = wheal.Position;
   r_wheal(:,i)= R*wheal.Position';  % Rotate wheal point

% Intersection with closest arm contour
if r_wheal(1,i) < r_mid_points(1,1)  % If wheal is located to the left of the midline
   x(i) = (r_wheal(2,i)-b1)/a1;          % Intersection with left arm contour: y=ax+b  -->  x=(y-b)/a
else                                      % If wheal is located to the right of the midline
   x(i) = (r_wheal(2,i)-b2)/a2;          % Intersection with right arm contour: y=ax+b  -->  x=(y-b)/a
end

% Relative distance
d1 = abs(r_wheal(1,i)-r_mid_points(1,1));  % Distance wheal to midline
d2 = abs(x(i)-r_mid_points(1,1));           % Distance arm contour to midline
ratio(k,i) = d1/d2;

pause                             % Press Enter (or any key other than s or d) to click the next wheal
 if f.CurrentCharacter == 's'   % Press s to stop clicking more wheals
    break
 elseif f.CurrentCharacter == 'd' % Press d to delete the last wheal (if you accidentally clicked another point)
    wheals(:,i) = NaN;
    r_wheal(:,i) = NaN;
    ratio(k,i) = NaN;
    break
 end
end

%% Check
% Show original image and clicked points
figure,
imshow(photo); hold on;
scatter(corners(:,1),corners(:,2),'r.'); hold on;
scatter(mid_points(:,1),mid_points(:,2),'g.'); hold on;
plot(mid_points(:,1),mid_points(:,2),'g'); hold on;
plot([corners(1,1) corners(3,1)],[corners(1,2) corners(3,2)],'r'); hold on;
plot([corners(2,1) corners(4,1)],[corners(2,2) corners(4,2)],'r'); hold on;
scatter(wheals(1,:),wheals(2,:),30,'b.'); hold off

% Plot arm contour, midline and wheal points
figure,
scatter(r_corners(1,:),r_corners(2,:),'r.'); hold on;
scatter(r_mid_points(1,:),r_mid_points(2,:),'g.'); hold on;
scatter(r_wheal(1,:),r_wheal(2,:),'b.'); hold on
plot(r_mid_points(1,:),r_mid_points(2,:),'g'); hold on;
plot([r_corners(1,1) r_corners(1,3)],[r_corners(2,1) r_corners(2,3)],'r'); hold on
plot([r_corners(1,2) r_corners(1,4)],[r_corners(2,2) r_corners(2,4)],'r'); hold on
y = yline(r_wheal(2,:),'b'); hold on;   % Draw y-line at wheal height
```

```
scatter(x(:),r_wheal(2,:),20)
set(gca, 'YDir','reverse')   % Flip y-axis
axis image
hold off
end

%% Save wheal locations
folder = 'C:\...\';
save(folder + "filename.mat", "ratio")
writematrix(ratio, folder + "filename.xlsx")
```

A3. Arm curvature correction using the placemat method

```
%% Arm curvature correction with placemat
% This script performs a correction for the arm curvature through an image
% transformation for which a source and fixed map are created.
clear all
close all
clc

% Load input images
im_checker = imread("....jpg"); % image with checkerboard for validation
im = imread("....jpg");        % image with removed checkerboard
figure, imshow(im_checker)
figure, imshow(im)

%% Detect ArUco markers
% Detect markers
im_gray = rgb2gray(im); % convert to grayscale image
im_bi = ~im2bw(im_gray,0.30); % binary image with threshold to find the markers
im_squares = imfill(im_bi, "holes"); % imfill creates filled areas of the markers
im_label = bwlabel(im_squares); % label the areas
im_label = bwareaopen(im_label,10000); % remove labeled areas with <1000 pixels. Markers (> 1000 pixels) will remain.
stat = regionprops(im_label,'boundingbox'); % provides the position of left-top corner of the markers and x- and y-width
centerpoints_markers = [];
info_marker = [];
for marker_number = 1 : 4 % determine center of each ArUco marker
   info_marker(marker_number,:) = stat(marker_number).BoundingBox;
   centerpoint = [info_marker(marker_number,1)+info_marker(marker_number,3)/2,
info_marker(marker_number,2)+info_marker(marker_number,4)/2];
   centerpoints_markers(marker_number, :) = centerpoint;
end

% Sort the centers by ordering the markers: left top, right top, left bottom, right bottom
[centerpoint1,i] = sort(centerpoints_markers(:,2)); % order based on y
centerpoint2 = centerpoints_markers(:,1);
centerpoint2 = centerpoint2(i);
[centerpointtop, i1] = sort(centerpoint2(1:2,1)); % order first two y's based on the x
[centerpointbottom, i2] = sort(centerpoint2(3:4,1)); % order last two y's based on the x
centerpoint2 = [centerpointtop;centerpointbottom];
i = [i1;i2+2];
centerpoint1 = centerpoint1(i);
centerpoints_markers = [centerpoint2,centerpoint1]; % centers

%% Crop the image around the markers
% Initial corner points for cropping:
% Positioned next to the markers with an expanding distance of 1/4 length of the markers
max_diff_x = max(info_marker(:,3)).*3/4; % max x-width of markers
max_diff_y = max(info_marker(:,4)).*3/4; % max y-width of markers
left_up_corner = [centerpoints_markers(1,1)-max_diff_x, centerpoints_markers(1,2)-max_diff_y];
right_up_corner = [centerpoints_markers(2,1)+max_diff_x, centerpoints_markers(2,2)-max_diff_y];
left_down_corner = [centerpoints_markers(3,1)-max_diff_x, centerpoints_markers(3,2)+max_diff_y];
```

```
right_down_corner = [centerpoints_markers(4,1)+max_diff_x, centerpoints_markers(4,2)+max_diff_y];
coordinates_box = [left_up_corner;right_up_corner;left_down_corner;right_down_corner];

% Final corner points for cropping (needed if the perspective of the image is not orthogonal to placemat):
% Find minimum and maximum x- and y-coordinates of the initial crop corners
x1 = min(coordinates_box(:,1));
x2 = max(coordinates_box(:,1));
y1 = min(coordinates_box(:,2));
y2 = max(coordinates_box(:,2));
width_x = x2-x1; % width of crop
height_y = y2-y1; % height of crop
im = imcrop(im, [x1, y1, width_x, height_y]); % crop image
im_checker = imcrop(im_checker, [x1, y1, width_x, height_y]); % crop image

% Put image upright
a = norm(centerpoints_markers(1,:) - centerpoints_markers(2,:));  % width
b = norm(centerpoints_markers(1,:) - centerpoints_markers(3,:));  % height
if a < b
  im = imrotate(im, -90); % rotate image if needed
  im_checker = imrotate(im_checker, -90);
end
figure, imshow(im)

%% Segmentation of arm
lab = rgb2lab(im);  % convert RGB to LAB color space
ab = lab(:,:,2:3);  % discard L (luminance) channel
ab = im2single(ab);
numColors = 2;
L2 = imsegkmeans(ab,numColors); % create two areas based on k-means: separate arm and background
stats = [regionprops(L2, 'Area').Area]; % size of areas
armarea = find(stats == min(stats)); % select smallest area, based on the fact that the arm should be smaller than the background
arm = ismember(L2, armarea);
se = strel('disk', 10);
arm = imdilate(arm,se);
arm = imerode(arm,se);
figure, imshow(arm);

%% Detect edge of arm
Edge = edge(arm,'sobel','vertical'); % detect vertical edges based on sobel edge detection
[linex,liney] = find(Edge==1);       % create an edge line of 1 pixel wide
for n = 1:size(linex)
   Edge(linex(n),(liney(n)+1)) = 0;
end
figure, imshow(Edge);

%% Detect ArUco markers in cropped image
% Detect markers
im_gray = rgb2gray(im);
im_bi2 = ~im2bw(im_gray,0.30);
im_squares = imfill(im_bi2, "holes");
im_label = bwlabel(im_squares);
im_label = bwareaopen(im_label,1000);
stat = regionprops(im_label,'boundingbox');
centerpoints_markers = [];
for marker_number = 1 : 4 % determine center of each arucomarker
   info_marker(marker_number,:) = stat(marker_number).BoundingBox;
   rectangle('position',info_marker(marker_number,:),'edgecolor','r','linewidth',2); % provides image to check if its the right box
   centerpoint = [info_marker(marker_number,1)+info_marker(marker_number,3)/2, info_marker(marker_number,2)+info_marker(marker_number,4)/2];
   centerpoints_markers(marker_number, :) = centerpoint;
end
```

```
% Sort the centers by ordering the markers: left top, right top, left bottom, right bottom
[centerpoint1,i] = sort(centerpoints_markers(:,2));
centerpoint2 = centerpoints_markers(:,1);
centerpoint2 = centerpoint2(i);
[centerpointtop, i1] = sort(centerpoint2(1:2,1));
[centerpointbottom, i2] = sort(centerpoint2(3:4,1));
centerpoint2 = [centerpointtop;centerpointbottom];
i = [i1;i2+2];
centerpoint1 = centerpoint1(i);
centerpoints_markers = [centerpoint2,centerpoint1];

% Check
figure, imshow(im), hold on;
scatter(centerpoints_markers(:,1), centerpoints_markers(:,2)); hold off

%% Initial intersections with arm contour (oblique cross-section)
% Create lines between markers
n = width(im_bi2);
m = height(im_bi2);
mask = false(m,n); % create black mask with size of image
mask =
createline(centerpoints_markers(1,1),centerpoints_markers(2,1),centerpoints_markers(1,2),centerpoints_markers(2,2),
mask); % create lines between centers of markers 1&2
mask =
createline(centerpoints_markers(3,1),centerpoints_markers(4,1),centerpoints_markers(3,2),centerpoints_markers(4,2),
mask); % between marker 3&4

% Find intersections with arm contour
inters = bitand(mask,Edge);     % create mask with only 1-values on overlapping area/intersections
[rows, cols] = find(inters==1); % find the coordinates of the pixels with value 1: intersection points

% Order the four intersection points: upper left, upper right, lower left, lower right
[rows,i] = sort(rows);  % order the rows
cols = cols(i);         % order the columns based on the order of the rows
[colstop, i1] = sort(cols(1:2,1));    % order the first two based on the columns
[colsbottom, i2] = sort(cols(3:4,1)); % order the last two based on the columns
cols = [colstop;colsbottom];
i = [i1;i2+2];
rows = rows(i); % combine the last orders in the order of the rows
coordinates = [rows,cols];

%% Align midline of the arm vertically straight
% Create midline
midpoint1 = (coordinates(1,:) + coordinates(2,:)).'/2; % find midpoint between the two intersection points
midpoint2 = (coordinates(3,:) + coordinates(4,:)).'/2;
mask2 = false(m,n); % create new mask
mask2 = createline(midpoint1(2,1),midpoint2(2,1),midpoint1(1,1),midpoint2(1,1), mask2); % create midline of the arm
arm_lines = imoverlay(im, (mask2+mask+Edge)); % image with arm contour, midline, and lines between markers

% Calculate angle of rotation
v_1 = [midpoint1(1,1),midpoint1(2,1),0] - [midpoint2(1,1),midpoint2(2,1),0]; % create vector of the midline
v_2 = [centerpoints_markers(4,2),centerpoints_markers(4,1),0] - [midpoint2(1,1),midpoint2(2,1),0]; % create vector of
baseline
Theta = atan2(norm(cross(v_1, v_2)), dot(v_1, v_2)); % calculate angle between the midline and baseline
Theta = rad2deg(Theta); % in degrees
Theta = (90-Theta); % calculate the angle needed for correction

% Rotate
Rotated_arm_lines = imrotate(arm_lines, Theta); % rotate lines with calculated angle
Rotated_arm = imrotate(im, Theta); % rotate image
Rotated_arm_checker = imrotate(im_checker, Theta); % used for validation
```

```
%% Source key points - Final intersections with arm contour (perpendicular cross-section)
% Create line halfway between markers for source points halfway the arm
middlepoint_left_line = [(centerpoints_markers(1,1)+centerpoints_markers(3,1))./2,
(centerpoints_markers(1,2)+centerpoints_markers(3,2))./2]; % midpoint in between marker 1&3
middlepoint_right_line = [(centerpoints_markers(2,1)+centerpoints_markers(4,1))./2,
(centerpoints_markers(2,2)+centerpoints_markers(4,2))./2]; % midpoint in between marker 2&4
mask = createline(middlepoint_left_line(1),middlepoint_right_line(1),middlepoint_left_line(2), middlepoint_right_line(2),
mask); % create line between the midpoints

% Create new lines for intersection
Rotated_edge = imrotate(Edge, Theta); % rotate the arm contour
[linex,liney] = find(Rotated_edge==1);
for i = 1:(size(linex)-1)
    Rotated_edge(linex(i),(liney(i)+1)) = 0;
end
n1 = width(Rotated_arm_lines);
m1 = height(Rotated_arm_lines);
mask3 = false(m1,n1); % create new mask, based on rotated image size
x = (n1-n)/2;
y = (m1-m)/2;
s = size(mask);
mask3(x:x+s(1)-1, y:y+s(2)-1) = mask; % add original mask to new sized mask

% Find intersections with arm contour
inters2 = bitand(mask3,Rotated_edge); % create mask with only 1-values on overlapping area
[rows1, cols1] = find(inters2==1); % find the coordinates with the value 1

% Order the intersection points: upper left, upper right, middle left, middle right, lower left, lower right
[rows1,i] = sort(rows1);
cols1 = cols1(i);
[colstop1, i1] = sort(cols1(1:2,1));
[colsmiddle1, i2] = sort(cols1(3:4,1));
[colsbottom1, i3] = sort(cols1(5:6,1));
cols1 = [colstop1;colsmiddle1;colsbottom1];
i = [i1;i2+2;i3+4];
rows1 = rows1(i);
coordinates1 = [rows1,cols1];

% Source key points
pt1 = coordinates1(1,:); % upper left point
pt2 = (coordinates1(1,:) + coordinates1(2,:)).'/2; % point in between upper two intersections
pt3 = coordinates1(2,:); % upper right
pt4 = coordinates1(3,:); % middle left
pt5 = (coordinates1(3,:) + coordinates1(4,:)).'/2; % point in between middle two intersections
pt6 = coordinates1(4,:); % middle right
pt7 = coordinates1(5,:); % lower left
pt8 = (coordinates1(5,:) + coordinates1(6,:)).'/2; % point in between lower two intersections
pt9 = coordinates1(6,:); % lower right
float_points = [pt1;pt2';pt3;pt4;pt5';pt6;pt7;pt8';pt9];

% Check source points
figure, imshow(Rotated_arm); hold on; scatter(float_points(:,2),float_points(:,1),'r.'); hold off

%% Determine width, height and circumference of arm
% For upper key points:
r1_top = norm(coordinates1(1,:)-coordinates1(2,:))./2; % width of arm (first radius of ellipse)
r2_top = 2/3.*r1_top; % height of the arm (second radius of ellipse) is assumed to be two-thirds of the width
h_top = ((r1_top-r2_top)/(r1_top+r2_top)).^2; % parameter needed for ellipse formula
length_top = (pi*(r1_top+r2_top)*((135168-(85760*h_top)-(5568*(h_top.^2))+(3867*(h_top.^3)))/(135168-
(119552*h_top)+(22208*(h_top.^2))-(345*(h_top.^3))))))./2; % arm circumference is approximated with formula of ellipse

% For middle key points:
r1_center = norm(coordinates1(3,:)-coordinates1(4,:))./2;
```

```
r2_center = 2/3.*r1_center;
h_center = ((r1_center-r2_center)/(r1_center+r2_center)).^2;
length_center = (pi*(r1_center+r2_center)*((135168-85760*h_center-5568*(h_center.^2)+3867*(h_center.^3))/(135168-
119552*h_center+22208*(h_center.^2)-345*(h_center.^3))))./2;

% For lower key points:
r1_bottom = norm(coordinates1(5,:)-coordinates1(6, :))./2;
r2_bottom = 2/3.*r1_bottom;
h_bottom = ((r1_bottom-r2_bottom)/(r1_bottom+r2_bottom)).^2;
length_bottom = (pi*(r1_bottom+r2_bottom)*((135168-85760*h_bottom-
5568*(h_bottom.^2)+3867*(h_bottom.^3))/(135168-119552*h_bottom+22208*(h_bottom.^2)-345*(h_bottom.^3))))./2;

%% Fixed key points
% Determine fixed points for upper key points
ref_point_top = pt2; % midpoint
r_top_source = norm(float_points(3,:)-ref_point_top'); % source distance between left and midpoint
r_top_fixed = 0.5*length_top; % fixed distance

% Interpolation in horizontal direction
x_src = (float_points(3,2)-ref_point_top(2));
x_fxd = (x_src*r_top_fixed)/r_top_source;
y_src = (float_points(3,1)-ref_point_top(1));
y_fxd = (y_src*r_top_fixed)/r_top_source;
y1 = ref_point_top(1)+y_fxd;x1=ref_point_top(2)+x_fxd;
y2 = ref_point_top(1)-y_fxd;x2=ref_point_top(2)-x_fxd;
y_top_interpolated = linspace(y1,y2,15); % y coordinates of interpolated points
x_top_interpolated = linspace(x1,x2,15); % x coordinates of interpolated points

% Repeat steps above for the middle key points
ref_point_center = pt5;
r_center_source = norm(float_points(6,:)-ref_point_center');
r_center_fixed = 0.5*length_center;

x_src = (float_points(6,2)-ref_point_center(2));
x_fxd = (x_src*r_center_fixed)/r_center_source;
y_src = (float_points(4,1)-ref_point_center(1));
y_fxd = (y_src*x_fxd)/x_src;
y1 = ref_point_center(1)+y_fxd;x1=ref_point_center(2)+x_fxd;
y2 = ref_point_center(1)-y_fxd;x2=ref_point_center(2)-x_fxd;
y_center_interpolated = linspace(y1,y2,15);
x_center_interpolated = linspace(x1,x2,15);

% Repeat steps above for the lower key points
ref_point_bottom = pt8;
r_bottom_source = norm(float_points(9,:)-ref_point_bottom');
r_bottom_fixed = 0.5*length_bottom;

x_src = (float_points(9,2)-ref_point_bottom(2));
x_fxd = (x_src*r_bottom_fixed)/r_bottom_source;
y_src = (float_points(7,1)-ref_point_bottom(1));
y_fxd = (y_src*x_fxd)/x_src;
y1 = ref_point_bottom(1)+y_fxd;x1=ref_point_bottom(2)+x_fxd;
y2 = ref_point_bottom(1)-y_fxd;x2=ref_point_bottom(2)-x_fxd;
y_bottom_interpolated = linspace(y1,y2,15);
x_bottom_interpolated = linspace(x1,x2,15);

% Check fixed key points
figure, imshow(Rotated_arm_checker); hold on;
scatter(x_top_interpolated,y_top_interpolated,'*'); hold on;
scatter(x_center_interpolated,y_center_interpolated,'*'); hold on;
scatter(x_bottom_interpolated,y_bottom_interpolated, '*'); hold off

%% Source map
```

```
col_count = 15;
row_count = 50;
points = float_points;

% Interpolate horizontally: the 15 source points are not equally spread
% For upper row of points:
left = points(1,:);right=points(3,:);top=points(2,:);
center = (left + right)/2;
a = norm(left + right, 'fro')/2;
b = norm(3*(center-top), 'fro');
var = top-center;
if var(2)>0
   delta = pi/(col_count+1);
else
   delta = -pi/(col_count+1);
end
var2 = center-left;
cos_rot = var2(1)/a;
sin_rot = var2(2)/a;
top_points = [];

for i = 1:col_count
   phi = i* delta;
   dx= a*cos(phi);
   dy= b*sin(phi);
   y = center(1) + (dx * cos_rot) -(dy*sin_rot);
   x = center(2) +(dx* sin_rot)+ (dy * cos_rot);
   top_points(i,:) = [x,y];
end

% For middle row of points
left = points(4,:);right=points(6,:);top=points(5,:);
center = (left + right)/2;
a = norm(left + right, 'fro')/2;
b = norm(8*(center-top), 'fro');
var = top-center;
if var(2)>0
   delta = pi/(col_count+1);
else
   delta = -pi/(col_count+1);
end
var2 = right-center;
cos_rot = var2(1)/a;
sin_rot = var2(2)/a;

center_points = [];
for i = 1:col_count
   phi = i* delta;
   dx = a*cos(phi);
   dy = b*sin(phi);
   y = center(1) + dx * cos_rot -dy*sin_rot;
   x = center(2) +dx* sin_rot+ dy * cos_rot;
   center_points(i,:) = [x,y];
end

% For lower row of points
left = points(7,:);right=points(9,:);top=points(8,:);
center = (left + right)/2;
a = norm(left + right, 'fro')/2;
b = norm(8*(center-top), 'fro');
var = top-center;
if var(2)>0
   delta = pi/(col_count+1);
```

```
else
    delta = -pi/(col_count+1);
end
var2 = right-center;
cos_rot = var2(1)/a;
sin_rot = var2(2)/a;

bottom_points = [];
for i = 1:col_count
    phi = i* delta;
    dx = a*cos(phi);
    dy = b*sin(phi);
    y = center(1) + dx * cos_rot -dy*sin_rot;
    x = center(2) +dx* sin_rot+ dy * cos_rot;
    bottom_points(i,:) = [x,y];
end

% Construct the entire source map - interpolate vertically
% Upper part of arm (first cylinder)
rows = [];
for i = 1:row_count
    row = [];
    for j = 1:col_count
        top_point = top_points(j,:);
        center_point = center_points(j,:);

        delta = (top_point- center_point) / (row_count-1);
        point = top_point - delta * (i-1);
        row(j,:) = point;
    end
    rows(i,:,:) = row;
end
source_map_topcenter = rows;

figure, imshow(Rotated_arm_checker); hold on;
for i = 1:row_count
    scatter(source_map_topcenter(i,:,1),source_map_topcenter(i,:,2))
end
source_map_topcenter_reshaped =
reshape(source_map_topcenter,(size(source_map_topcenter,1)*size(source_map_topcenter,2)),2);

% Lower part of arm (second cylinder)
rows = [];
for i = 1:row_count
    row = [];
    for j = 1:col_count
        center_point = center_points(j,:);
        bottom_point = bottom_points(j,:);

        delta = (center_point- bottom_point) / (row_count-1);
        point = center_point - delta * (i-1);
        row(j,:) = point;
    end
    rows(i,:,:) = row;
end
source_map_centerbottom = rows;

for i = 1:row_count
    scatter(source_map_centerbottom(i,:,1),source_map_centerbottom(i,:,2))
end
source_map_centerbottom_reshaped =
reshape(source_map_centerbottom,(size(source_map_centerbottom,1)*size(source_map_centerbottom,2)),2);
hold off
```

```
%% Fixed map
% Translate the previously created rows of fixed points, basically align them with the source points:
% Find middle location in row of source points
midpoint_src_top = source_map_topcenter(1,ceil(end/2),:);
midpoint_src_top = reshape(midpoint_src_top,1,2);
midpoint_src_center = source_map_topcenter(end,ceil(end/2),:);
midpoint_src_center = reshape(midpoint_src_center,1,2);
midpoint_src_bottom = source_map_centerbottom(end,ceil(end/2),:);
midpoint_src_bottom = reshape(midpoint_src_bottom,1,2);

% Find middle location in row of fixed points
midpoint_fxd_top = [x_top_interpolated(ceil(end/2)),y_top_interpolated(ceil(end/2))];
midpoint_fxd_center = [x_center_interpolated(ceil(end/2)),y_center_interpolated(ceil(end/2))];
midpoint_fxd_bottom = [x_bottom_interpolated(ceil(end/2)),y_bottom_interpolated(ceil(end/2))];

% Translate the rows of fixed points
diff1 = midpoint_fxd_top-midpoint_src_top; % determine required translation for upper fixed points
diff2 = midpoint_fxd_center-midpoint_src_center; % determine required replacement for middle fixed points
diff3 = midpoint_fxd_bottom-midpoint_src_bottom; % determine required replacement for lower fixed points
x_top_interpolated = x_top_interpolated-diff1(1); % shift
y_top_interpolated = y_top_interpolated-diff1(2); % shift
x_center_interpolated = x_center_interpolated-diff2(1); % shift
y_center_interpolated = y_center_interpolated-diff2(2); % shift
x_bottom_interpolated = x_bottom_interpolated-diff3(1); % shift
y_bottom_interpolated = y_bottom_interpolated-diff3(2); % shift

figure, imshow(Rotated_arm); hold on;
scatter(x_top_interpolated,y_top_interpolated,'*'); hold on;
scatter(x_center_interpolated,y_center_interpolated,'*'); hold on;
scatter(x_bottom_interpolated,y_bottom_interpolated,'*'); hold off

% Construct the entire fixed map
% Determine the ratio between length of the source en fixed rows
top_source = reshape(source_map_topcenter(1,:,:),size(source_map_topcenter,2),2); % first row of source map
length_top_source = norm(top_source(1,:)-top_source(end,:)); % length between first point and last point of this row
length_top_fixed = norm([x_top_interpolated(1),y_top_interpolated(1)]-
[x_top_interpolated(end),y_top_interpolated(end)]); % length of first point and last point of fixed row
factor_top = length_top_fixed/length_top_source; % factor between the length of source and fixed row

% Repeat for middle row
center_source = reshape(source_map_topcenter(end,:,:),size(source_map_topcenter,2),2);
length_center_source = norm(center_source(1,:)-center_source(end,:));
length_center_fixed = norm([x_center_interpolated(1),y_center_interpolated(1)]-
[x_center_interpolated(end),y_center_interpolated(end)]);
factor_center = length_center_fixed/length_center_source;

% Repeat for lower row
bottom_source = reshape(source_map_centerbottom(end,:,:),size(source_map_centerbottom,2),2);
length_bottom_source = norm(bottom_source(1,:)-bottom_source(end,:));
length_bottom_fixed = norm([x_bottom_interpolated(1),y_bottom_interpolated(1)]-
[x_bottom_interpolated(end),y_bottom_interpolated(end)]);
factor_bottom = length_bottom_fixed/length_bottom_source;

% Create array with all the midpoints of each row of the source map along the arm
midpoints_source_topcenter = [];
midpoints_source_centerbottom = [];

for i = 1:size(source_map_topcenter,1)
    row = reshape(source_map_topcenter(i,:,:),size(source_map_topcenter,2),2);
    mid_point = row(ceil(end/2),:);
    midpoints_source_topcenter(i,:) = mid_point;
end
```

```
for i = 1:size(source_map_centerbottom,1)
    row = reshape(source_map_centerbottom(i,:,:),size(source_map_centerbottom,2),2);
    mid_point = row(ceil(end/2),:);
    midpoints_source_centerbottom(i,:) = mid_point;
end

% Calculate the gradient of the rows of fixed points
gradient_top = (y_top_interpolated(1)-y_top_interpolated(end))/(x_top_interpolated(1)-x_top_interpolated(end));
gradient_center = (y_center_interpolated(1)-y_center_interpolated(end))/(x_center_interpolated(1)-x_center_interpolated(end));
gradient_bottom = (y_bottom_interpolated(1)-y_bottom_interpolated(end))/(x_bottom_interpolated(1)-x_bottom_interpolated(end));
% Difference in gradient between the upper-middle and middle-lower rows
diff_gradient_topcenter = gradient_center-gradient_top;
diff_gradient_centerbottom = gradient_bottom-gradient_center;
% Difference in gradient between between the length of the source and fixed row
diff_factor_topcenter = factor_center-factor_top;
diff_factor_centerbottom = factor_bottom-factor_center;

% Interpolation of fixed points, for upper part of arm (first cylinder)
figure, imshow(image); hold on;
required_length = [];
fxd_left = [];
fxd_right = [];
for j = 2:(size(source_map_topcenter,1)-1)
    row = reshape(source_map_topcenter(j,:,:),size(source_map_topcenter,2),2); % select a row
    length_row_source = norm(row(1,:)-row(end,:)); % determine length of the row
    diff_length_sources = length_row_source-length_top_source; % determine delta length row
    factor = factor_top+((diff_length_sources*diff_factor_topcenter)/(length_center_source-length_top_source)); % calculate multiplying factor needed to obtain required length. Based on geometry
    gradient = gradient_top+((diff_length_sources*diff_gradient_topcenter)/(length_center_source-length_top_source)); % calculate the gradient to determine the direction of the row of fixed points. Based on geometry
    required_length(j-1) = length_row_source*factor; % required length of the fixed row

    deltax = sqrt((0.5*required_length(j-1))^2/(1+gradient^2)); % apply pythagoras to get difference in x position
    new_point = [midpoints_source_topcenter(j,1)+deltax,midpoints_source_topcenter(j,2)+(gradient*deltax)]; % end of row with fixed points towards right
    deltax2 = -deltax;
    new_point2 = [midpoints_source_topcenter(j,1)+deltax2,midpoints_source_topcenter(j,2)+(gradient*deltax2)]; % end of row with fixed points towards left
    fxd_left(j-1,:) = new_point2; % store all fixed points at left side
    fxd_right(j-1,:) = new_point; % store all fixed points at right side
end
x_between_interpolated = [];
y_between_interpolated = [];
for i = 1:size(fxd_left, 1)
    x_between_interpolated(i,:) = linspace(fxd_right(i,1),fxd_left(i,1),col_count); % interpolate between left and right
    y_between_interpolated(i,:) = linspace(fxd_right(i,2),fxd_left(i,2),col_count); % interpolate between left and right
end

% Repeat steps for lower part of arm (second cylinder)
fxd_left = [];
fxd_right = [];
for j = 2:(size(source_map_centerbottom,1)-1)
    row = reshape(source_map_centerbottom(j,:,:),size(source_map_centerbottom,2),2);
    length_row_source = norm(row(1,:)-row(end,:));
    diff_length_sources = length_row_source-length_top_source;
    factor = factor_center+((diff_length_sources*diff_factor_centerbottom)/(length_bottom_source-length_center_source));
    gradient = gradient_center+((diff_length_sources*diff_gradient_centerbottom)/(length_bottom_source-length_center_source));
    required_length(j-1) = length_row_source*factor;
    deltax = sqrt((0.5*required_length(j-1))^2/(1+gradient^2));
    new_point = [midpoints_source_centerbottom(j,1)+deltax,midpoints_source_centerbottom(j,2)+(gradient*deltax)];
    deltax2 = -deltax;
    new_point2 = [midpoints_source_centerbottom(j,1)+deltax2,midpoints_source_centerbottom(j,2)+(gradient*deltax2)];
    fxd_left(j-1,:) = new_point2;
    fxd_right(j-1,:) = new_point;
end
x_between_interpolated_centerbottom = [];
y_between_interpolated_centerbottom = [];
for i = 1:size(fxd_left, 1)
    x_between_interpolated_centerbottom(i,:) = linspace(fxd_right(i,1),fxd_left(i,1),col_count);
    y_between_interpolated_centerbottom(i,:) = linspace(fxd_right(i,2),fxd_left(i,2),col_count);
end

fxd_x = cat(1,x_top_interpolated,x_between_interpolated,x_center_interpolated);
fxd_y = cat(1,y_top_interpolated,y_between_interpolated,y_center_interpolated);
fxd_x1 = cat(1,x_center_interpolated,x_between_interpolated_centerbottom,x_bottom_interpolated);
fxd_y1 = cat(1,y_center_interpolated,y_between_interpolated_centerbottom,y_bottom_interpolated);

% Combine the constructed fixed maps of the first and second cylinder
fxd_x2 = [fxd_x,fxd_x1];
fxd_y2 = [fxd_y,fxd_y1];
source_map_combined_reshaped = [source_map_topcenter_reshaped; source_map_centerbottom_reshaped];
y2_2 = fxd_x2(:);
x2_2 = fxd_y2(:);
y1_2 = source_map_combined_reshaped(:,1);
x1_2 = source_map_combined_reshaped(:,2);

%% Check source and fixed map
% Show source map
figure, imshow(Rotated_arm_checker); hold on;
scatter(float_points(:,2),float_points(:,1), 'r.'); hold on; % intersection points
scatter(source_map_topcenter_reshaped(:,1),source_map_topcenter_reshaped(:,2),'b.');
scatter(source_map_centerbottom_reshaped(:,1),source_map_centerbottom_reshaped(:,2),'r.'); hold off

% Show fixed map
figure, imshow(Rotated_arm_checker); hold on;
scatter(fxd_x(:),fxd_y(:),'y.'); scatter(fxd_x1(:),fxd_y1(:),'g.'); hold off

%% Apply B-spline transformation for curvature correction
% For this section, open source functions written by D.J. Kroon are used.
options.Verbose = true;
[O_trans,Spacing, Xreg] = point_registration(size(Rotated_arm_checker),[x2_2(:) y2_2(:)],[x1_2(:) y1_2(:)],options);
[Ireg] = bspline_transform(O_trans,Rotated_arm_checker,Spacing,1); % transform the image

% Show the corrected image
figure,
subplot(1,2,1),imshow(Rotated_arm_checker); title('Original Image');
hold on; for i=1:length(x1), plot([y1_2(i) y2_2(i)],[x1_2(i) x2_2(i)],'b'); end
subplot(1,2,2),imshow(Ireg); title('Registered Image')
hold on; for i=1:length(x1), plot([y1_2(i) y2_2(i)],[x1_2(i) x2_2(i)],'b'); end
hold off
figure, imshow(Rotated_arm_checker) % original image
figure, imshow(Ireg) % transformed image
```

A4. Validation of the arm curvature correction

```
%% PART 1: Mask for uncorrected image (for full width of arm)
% This mask is derived from the binary image of the segmented arm from the curvature correction script.

% Mask of non-transformed arm (100% of forearm)
arm_seg = imrotate(arm, Theta);     % segmented arm, binary image, rotated
```

```
intersections = [coordinates1(:,2) coordinates1(:,1)];   % x in first column, y in second column
H_top = (1:intersections(1,2));
H_bottom = (intersections(5,2):size(arm_seg,1));
L = (1:size(arm_seg,2));
arm_seg(H_top,L) =0;
arm_seg(H_bottom,L) =0;
figure, imshow(arm_seg)

% Mask for corrected image with placemat (100% of forearm)
[mask_Tp] = bspline_transform(O_trans,arm_seg,Spacing,1);   % transform the uncorrected mask
figure, imshow(mask_Tp)


%% PART 2: Masks for different regions of interest (ROI) on arm
% This script creates masks for different regions on the arm.
% Input needed: binary image of segmented arm obtained from curvature correction script.
clear all
close all
clc

name = "...";  % filename
folder = "C:\...\" + name + "\";
percentage = 0.96; % part of forearm included in ROI

%% Determine arm contour/edge (100% of arm)
% load the initial non-transformed mask
image_nonTransformed = im2double(imread(folder + name + "_nonTransformed.jpg"));
mask_nonTransformed = imread(folder + name + "_mask_nonTransformed.png"); % 100% mask without ROI

% edge of mask
Edge = edge(mask_nonTransformed,'sobel','vertical'); % find vertical edges based on sobel edge detection
[linex,liney] = find(Edge==1);      % create an edge line of 1 pixel wide
for n = 1:size(linex)
  Edge(linex(n),(liney(n)+1)) = 0;
end

% separate left and right contour
[edgeY, edgeX] = find(Edge);
edgeXY = [edgeX edgeY];             % x and y coordinates of the edge
mid = min(edgeX) + (max(edgeX)-min(edgeX))/2;   % calculate x value in between the left and right contour
[edgeLeftY] = find(edgeX<mid);      % y coord of left arm contour
[edgeRightY] = find(edgeX>mid);     % y coord of right arm contour
edgeLeft = edgeXY(1:length(edgeLeftY),:);   % x,y of left contour
edgeRight = edgeXY(edgeRightY(1):max(edgeRightY),:);    % x,y of right contour

% smooth the contours by removing staggered points on the edges:
% Left contour
for i = 1:length(edgeLeft)
  if mask_nonTransformed(edgeLeft(i,2),edgeLeft(i,1)) == 0
    edgeLeft(i,:) = NaN;
  end
end
edgeLeft = rmmissing(edgeLeft);

% Right contour
for i = 1:length(edgeRight)
  if mask_nonTransformed(edgeRight(i,2),edgeRight(i,1)) == 1
    edgeRight(i,:) = NaN;
  end
end
edgeRight = rmmissing(edgeRight);

% order the coordinates:
```

```
% Left contour
edgeLeft_ordered = edgeLeft;
[a,b] = sort(edgeLeft_ordered(:,2));
edgeLeft_ordered(:,1) = edgeLeft_ordered(b);
edgeLeft_ordered(:,2) = a;

% Right contour
edgeRight_ordered = edgeRight;
[a,b] = sort(edgeRight_ordered(:,2));
edgeRight_ordered(:,1) = edgeRight_ordered(b);
edgeRight_ordered(:,2) = a;

% show contours
figure, imshow(mask_nonTransformed), hold on;
scatter(edgeLeft_ordered(:,1),edgeLeft_ordered(:,2),'b*');
scatter(edgeRight_ordered(:,1),edgeRight_ordered(:,2),'m*');
hold off

%% Midline of arm
% get x-coordinate of midline
midline = load(folder + name + "_midline.mat");   % midline defined on non-transformed image
midline = midline.Rotated_midline;
[~,Xb] = find(midline);
X_midline = Xb(1);

%% Adjust ROI
% Narrowed left and right contours:
edgeLeft_ROI = edgeLeft_ordered;
for i = 1:length(edgeLeft_ordered)
   dist_midLeft = abs(X_midline - edgeLeft_ordered(i,1)); % full width of arm 100%
   new_dist = (1-percentage)*dist_midLeft;
   edgeLeft_ROI(i,1) = edgeLeft_ordered(i,1) + round(new_dist);    % new position of edge
end

edgeRight_ROI = edgeRight_ordered;
for i = 1:length(edgeRight_ordered)
   dist_midRight = abs(X_midline - edgeRight_ordered(i,1)); % full width of arm 100%
   new_dist = (1-percentage)*dist_midRight;
   edgeRight_ROI(i,1) = edgeRight_ordered(i,1) - round(new_dist);
end

% Create mask for non-transformed image
vertices_ROI = [edgeLeft_ROI; flipud(edgeRight_ROI)];
mask_ROI_nonTransformed = roipoly(image_nonTransformed,vertices_ROI(:,1),vertices_ROI(:,2));
overlay1 = imfuse(image_nonTransformed,mask_ROI_nonTransformed);
figure, imshow(overlay1) % check

%% MaskPlacemat
% transform mask with transformation matrix obtained from curvature correction
mask_ROI_Tp = MaskPlacemat(mask_ROI_nonTransformed,folder,name);

% check
image_Tp = im2double(imread(folder + name + "_Tp.jpg"));
overlay2 = imfuse(image_Tp,mask_ROI_Tp);
figure, imshow(overlay2)

%% From here: create MaskBarcode
% To create the masks for the barcode corrected images, the segmented arm in the cropped image (obtained from placemat
method) must now be obtained in the full-sized original image as the transformation matrix of the barcode method is only
applicable on the full-sized image.
mask_nonTransformedFull = imread(folder + name + "_mask_nonTransformedFull.png");  % segmented arm in full-sized
image
```

```
% create midline
verticesBarcode = load(folder + name + "_verticesBarcode.mat");
verticesBarcode = verticesBarcode.float_points;
midpoint1 = verticesBarcode(2,:);
midpoint2 = verticesBarcode(8,:);
m = size(mask_nonTransformedFull,1);
n = size(mask_nonTransformedFull,2);
midlineBarcode = false(m,n);
midlineBarcode = createline(midpoint1(1,1),midpoint2(1,1),midpoint1(1,2),midpoint2(1,2), midlineBarcode); % create the midline of the arm

% rotate arm and midline to a vertically straight position
% (needed to determine distance between arm contour and midline to adjust the ROI)
load(folder + name + "_Theta.mat");
mask_R = imrotate(mask_nonTransformedFull,Theta);
midlineBarcode_R = imrotate(midlineBarcode,Theta);
rotated = imfuse(midlineBarcode_R,mask_R);

%% Determine arm contour/edge (100%)
% edge of mask
Edge = edge(mask_R,'sobel','vertical'); %find vertical edges based on sobel edge detection
[linex,liney] = find(Edge==1);      %create an edge line of 1 pixel wide
for n = 1:size(linex)
   Edge(linex(n),(liney(n)+1)) = 0;
end

% separate left and right contour
[edgeY, edgeX] = find(Edge);
edgeXY = [edgeX edgeY];            % x and y coordinates of the edge
mid = min(edgeX) + (max(edgeX)-min(edgeX))/2;   % calculate x value in between the left and right contour
[edgeLeftY] = find(edgeX<mid);      % y coord of left arm contour
[edgeRightY] = find(edgeX>mid);     % y coord of right arm contour
edgeLeft = edgeXY(1:length(edgeLeftY),:);   % x,y of left contour
edgeRight = edgeXY(edgeRightY(1):max(edgeRightY),:);    % x,y of right contour

% order the coordinates:
% Left contour
edgeLeft_ordered = edgeLeft;
[a,b] = sort(edgeLeft_ordered(:,2));
edgeLeft_ordered(:,1) = edgeLeft_ordered(b);
edgeLeft_ordered(:,2) = a;
edgeLeft_ordered(1:100,:) = [];
edgeLeft_ordered((length(edgeLeft_ordered)-100):length(edgeLeft_ordered),:) = [];

% Right contour
edgeRight_ordered = edgeRight;
[a,b] = sort(edgeRight_ordered(:,2));
edgeRight_ordered(:,1) = edgeRight_ordered(b);
edgeRight_ordered(:,2) = a;
edgeRight_ordered(1:100,:) = [];
edgeRight_ordered((length(edgeRight_ordered)-100):length(edgeRight_ordered),:) = [];

% show contours
figure, imshow(mask_R), hold on;
scatter(edgeLeft_ordered(:,1),edgeLeft_ordered(:,2),'b*');
scatter(edgeRight_ordered(:,1),edgeRight_ordered(:,2),'m*');
hold off

%% Adjust ROI
% get x-coordinate of midline
[~,Xb] = find(midlineBarcode_R);
X_midlineBarcode = Xb(1);
```

```
% Narrowed left and right contours:
edgeLeft_ROI = edgeLeft_ordered;
for i = 1:length(edgeLeft_ordered)
   dist_midLeft = abs(X_midlineBarcode - edgeLeft_ordered(i,1)); % full width of arm 100%
   new_dist = (1-percentage)*dist_midLeft;
   edgeLeft_ROI(i,1) = edgeLeft_ordered(i,1) + round(new_dist);    % new position of edge
end

edgeRight_ROI = edgeRight_ordered;
for i = 1:length(edgeRight_ordered)
   dist_midRight = abs(X_midlineBarcode - edgeRight_ordered(i,1)); % full width of arm 100%
   new_dist = (1-percentage)*dist_midRight;
   edgeRight_ROI(i,1) = edgeRight_ordered(i,1) - round(new_dist);
end

% Create mask for non-transformed image
vertices_ROI = [edgeLeft_ROI; flipud(edgeRight_ROI)];
mask_ROI_R = roipoly(mask_R,vertices_ROI(:,1),vertices_ROI(:,2));
overlay3 = imfuse(mask_R, mask_ROI_R);
figure, imshow(overlay3)

%% Back to original image size (so you can transform from there with the barcode transformation matrix)
% Inverse rotation
mask_ROI = imrotate(mask_ROI_R,-Theta);

% Resize to initial image size
xmin = (size(mask_ROI,2)-size(mask_nonTransformedFull,2))/2;
ymin = (size(mask_ROI,1)-size(mask_nonTransformedFull,1))/2;
width = size(mask_nonTransformedFull,2) -1;
height = size(mask_nonTransformedFull,1) -1;
mask_ROI = imcrop(mask_ROI,[xmin ymin width height]);

% check
image_nonTransformedFull = im2double(imread(folder + name + "_P.jpg"));
overlay4 = imfuse(image_nonTransformedFull,mask_ROI);
figure, imshow(overlay4)

%% Transform to barcode mask
mask_ROI_Tb = MaskBarcode(mask_ROI,folder,name);

% cut mask at arucomarker level
verticesPlacemat = load(folder + name + "_verticesPlacemat.mat");
verticesPlacemat = verticesPlacemat.float_points;
start_row = verticesPlacemat(1,2);
end_row = verticesPlacemat(9,2);

H_top = (1:start_row);
H_bottom = (end_row:size(mask_ROI_Tb,1));
L = (1:size(mask_ROI_Tb,2));
mask_ROI_Tb(H_top,L) =0;
mask_ROI_Tb(H_bottom,L) =0;

%% Adjust height of masks
% cut at lower height to exclude half squares at the top and bottom of the mask
shift_height = 60;
verticesPlacemat = load(folder + name + "_verticesPlacemat.mat");
verticesPlacemat = verticesPlacemat.float_points;
start_row = verticesPlacemat(1,2) + shift_height;
end_row = verticesPlacemat(9,2) - shift_height;
H_top = (1:start_row);  % start of mask
H_bottom = (end_row:size(mask_nonTransformed,1)); % end of mask
L = (1:size(mask_nonTransformed,2)); % total length of mask image
```

```
% Non-transformed
mask2_ROI_nonTransformed = mask_ROI_nonTransformed;
mask2_ROI_nonTransformed(H_top,L) =0;
mask2_ROI_nonTransformed(H_bottom,L) =0;

% Transformed with placemat
mask2_ROI_Tp = mask_ROI_Tp;
mask2_ROI_Tp(H_top,L) =0;
mask2_ROI_Tp(H_bottom,L) =0;

% Transformed with barcode
mask2_ROI_Tb = mask_ROI_Tb;
mask2_ROI_Tb(H_top,L) =0;
mask2_ROI_Tb(H_bottom,L) =0;

% check
image_nonTransformed = im2double(imread(folder + name + "_nonTransformed.jpg"));
image_Tp = im2double(imread(folder + name + "_Tp.jpg"));
image_Tb = im2double(imread(folder + name + "_Tb.jpg"));
overlay6 = imfuse(image_nonTransformed,mask2_ROI_nonTransformed);
overlay7 = imfuse(image_Tp,mask2_ROI_Tp);
overlay8 = imfuse(image_Tb,mask2_ROI_Tb);
figure, imshow(overlay6)
figure, imshow(overlay7)
figure, imshow(overlay8)

%% Save mask
imwrite(mask2_ROI_nonTransformed, folder + name + "_mask_ROI_" + percentage + "_nonTransformed.png")
imwrite(mask2_ROI_Tp, folder + name + "_mask_ROI_" + percentage + "_Tp.png")
imwrite(mask2_ROI_Tb, folder + name + "_mask_ROI_" + percentage + "_Tb.png")

imwrite(overlay1, folder + name + "_mask_ROI_" + percentage + "_overlay_nonTransformed.png")
imwrite(overlay2, folder + name + "_mask_ROI_" + percentage + "_overlay_Tp.png")
imwrite(overlay5, folder + name + "_mask_ROI_" + percentage + "_overlay_Tb.png")



%% PART 3: Masks for region of interest (ROI) on arm
% This script creates the final masks for the particular regions of the arm.
% Input needed: previously created masks of different regions of arm.
clear all
close all
clc

name = "...";
folder = "C:\...\" + name + "\";
image_nonTransformed = im2double(imread(folder + name + "_nonTransformed.jpg"));
image_Tp = im2double(imread(folder + name + "_Tp.jpg"));
image_Tb = im2double(imread(folder + name + "_Tb.jpg"));

%% Part of arm: 90-96%
% load 90% mask
mask_nonTransformed_ROI90 = imread(folder + name + "_mask_ROI_0.9_nonTransformed.png");
mask_Tp_ROI90 = imread(folder + name + "_mask_ROI_0.9_Tp.png");
mask_Tb_ROI90 = imread(folder + name + "_mask_ROI_0.9_Tb.png");

% load 96% mask
mask_nonTransformed_ROI96 = imread(folder + name + "_mask_ROI_0.96_nonTransformed.png");
mask_Tp_ROI96 = imread(folder + name + "_mask_ROI_0.96_Tp.png");
mask_Tb_ROI96 = imread(folder + name + "_mask_ROI_0.96_Tb.png");

% subtract masks
mask_nonTransformed_ROI90_96 = mask_nonTransformed_ROI96 - mask_nonTransformed_ROI90;
mask_Tp_ROI90_96 = mask_Tp_ROI96 - mask_Tp_ROI90;
```

```
mask_Tb_ROI90_96 = mask_Tb_ROI96 - mask_Tb_ROI90;

%% Part of arm: 80-90%
% load 80% mask
mask_nonTransformed_ROI80 = imread(folder + name + "_mask_ROI_0.8_nonTransformed.png");
mask_Tp_ROI80 = imread(folder + name + "_mask_ROI_0.8_Tp.png");
mask_Tb_ROI80 = imread(folder + name + "_mask_ROI_0.8_Tb.png");

% subtract masks
mask_nonTransformed_ROI80_90 = mask_nonTransformed_ROI90 - mask_nonTransformed_ROI80;
mask_Tp_ROI80_90 = mask_Tp_ROI90 - mask_Tp_ROI80;
mask_Tb_ROI80_90 = mask_Tb_ROI90 - mask_Tb_ROI80;

%% Part of arm: 70-80%
% load 70% mask
mask_nonTransformed_ROI70 = imread(folder + name + "_mask_ROI_0.7_nonTransformed.png");
mask_Tp_ROI70 = imread(folder + name + "_mask_ROI_0.7_Tp.png");
mask_Tb_ROI70 = imread(folder + name + "_mask_ROI_0.7_Tb.png");

% subtract masks
mask_nonTransformed_ROI70_80 = mask_nonTransformed_ROI80 - mask_nonTransformed_ROI70;
mask_Tp_ROI70_80 = mask_Tp_ROI80 - mask_Tp_ROI70;
mask_Tb_ROI70_80 = mask_Tb_ROI80 - mask_Tb_ROI70;

%% Part of arm: 0-70%
% This ROI is covered by the 70% mask (does not have to be subtracted from
% other ROI mask). See previous section.


%% PART 4: Validation method based on segmentation of checkerboard squares
% This script validates the arm curvature correction by segmenting checkerboard squares in the corrected image and
% calculating their areas. The results are visualized with a colormap.
% Input needed: the masks created in another script for selecting regions of interest on the arm.
clear all
close all
clc

%% Load images and masks
name = "...";  % filename
folder = "C:\...\" + name + "\";

% load images to be validated
image_nonTransformed = im2double(imread(folder + name + "_nonTransformed.jpg"));   % non-transformed (cropped and
rotated with placemat-script)
image_Tp = im2double(imread(folder + name + "_Tp.jpg"));   % transformed with placemat method
image_Tb = im2double(imread(folder + name + "_Tb.jpg"));   % transformed with barcode method

% enhance contrast
contrast_nonTransformed = imadjust(rgb2gray(image_nonTransformed));
contrast_Tp = imadjust(rgb2gray(image_Tp));
contrast_Tb = imadjust(rgb2gray(image_Tb));

% convert to binary image
threshold = 0.4;
bi_nonTransformed = imcomplement(im2bw(contrast_nonTransformed,threshold));
bi_Tp = imcomplement(im2bw(contrast_Tp,threshold));
bi_Tb = imcomplement(im2bw(contrast_Tb,threshold));

% load masks
mask_nonTransformed = im2double(imread(folder + name + "_mask_nonTransformed.jpg"));
mask_Tp = im2double(imread(folder + name + "_mask_Tp.jpg"));   % transformed with placemat method
mask_Tb = im2double(imread(folder + name + "_mask_Tb.jpg"));   % transformed with barcode method
```

```
% apply masks
nonTransformed = image_nonTransformed.*mask_nonTransformed; % on rgb image
Tp = image_Tp.*mask_Tp;
Tb = image_Tb.*mask_Tb;

bi_nonTransformed = bi_nonTransformed.*mask_nonTransformed; % on binary image
bi_Tp = bi_Tp.*mask_Tp;
bi_Tb = bi_Tb.*mask_Tb;

%% Label the segmented squares
SE = strel('square',10);

% erosion
erode_nonTransformed = imerode(bi_nonTransformed,SE);
erode_Tp = imerode(bi_Tp,SE);
erode_Tb = imerode(bi_Tb,SE);

% fill objects
fill_nonTransformed = imfill(erode_nonTransformed,"holes");
fill_Tp = imfill(erode_Tp,"holes");
fill_Tb = imfill(erode_Tb,"holes");

% labeling of objects
Label_nonTransformed = bwlabel(erode_nonTransformed);
Label_Tp = bwlabel(erode_Tp);
Label_Tb = bwlabel(erode_Tb);

% dilation while remaining the separate labeled objects
label_nonTransformed = imdilate(Label_nonTransformed, SE);
label_Tp = imdilate(Label_Tp, SE);
label_Tb = imdilate(Label_Tb, SE);

%%  Calculate areas in pixels
% Pre-allocate arrays
Areas_nonTransformed = zeros(max(label_nonTransformed(:)),1);
Areas_Tp = zeros(max(label_Tp(:)),1);
Areas_Tb = zeros(max(label_Tb(:)),1);

% Areas in non-transformed image
for object = 1:max(label_nonTransformed(:))  % for all objects
   area = regionprops(label_nonTransformed == object,'Area');
   Areas_nonTransformed(object) = area.Area;
end

% Areas in transformed image using placemat
for object = 1:max(label_Tp(:))
   area = regionprops(label_Tp == object,'Area');
   Areas_Tp(object) = area.Area;
end

% Areas in transformed image using barcode
for object = 1:max(label_Tb(:))
   area = regionprops(label_Tb == object,'Area');
   Areas_Tb(object) = area.Area;
end

%% Adjust pixel values of each object
% If the area of a checkerboard square contains 3340 pixels, then you give all pixels in that area the value 3340.

% Non-transformed image
label2_nonTransformed = label_nonTransformed;
for i = 1:max(label_nonTransformed(:))    % number of objects
   label2_nonTransformed(label_nonTransformed==i) = length(find(label_nonTransformed==i)); % replace pixel value
```

```
end

% Transformed with placemat
label2_Tp = label_Tp;
for i = 1:max(label_Tp(:))
   label2_Tp(label_Tp==i) = length(find(label_Tp==i));
end

% Transformed with barcode
label2_Tb = label_Tb;
for i = 1:max(label_Tb(:))
   label2_Tb(label_Tb==i) = length(find(label_Tb==i));
end

%% Gold standard for area of square
% Based on the squares situated on the midline of the arm (where the curvature is negligible).
% Gold standard is determined in the non-transformed image.

% get x-coordinate of midline
midline = load(folder + name + "_midline.mat");   % midline defined on non-transformed image
midline = midline.Rotated_midline;
[~,X] = find(midline);
X_midline = X(1);

% calculate mean area along midline
areas_midline = label2_nonTransformed(:,X_midline);
areas_midline(areas_midline==0)= NaN;      % remove zeros along the midline
areaPixels_goldStandard = mean(areas_midline,"omitnan");     % gold standard area in pixels

% check
Overlay = imfuse(midline,nonTransformed); figure, imshow(Overlay)

%% Relative areas (gold standard is value 1)
% Calculate relative deviation with respect to gold standard:
% Non-transformed image
for i = 1:length(Areas_nonTransformed)
   Areas_nonTransformed(i,2) = Areas_nonTransformed(i,1)/areaPixels_goldStandard;
end

% Transformed with placemat
for i = 1:length(Areas_Tp)
   Areas_Tp(i,2) = Areas_Tp(i,1)/areaPixels_goldStandard;
end

% Transformed with barcode
for i = 1:length(Areas_Tb)
   Areas_Tb(i,2) = Areas_Tb(i,1)/areaPixels_goldStandard;
end

%% Adjust to relative/scaled/normalized pixel values
% Non-transformed image
labelscaled_nonTransformed = label_nonTransformed;
for i = 1:max(label_nonTransformed(:))  % number of object
   labelscaled_nonTransformed(label_nonTransformed==i) = Areas_nonTransformed(i,2);  % replace pixel value
end

% Transformation with placemat
labelscaled_Tp = label_Tp;
for i = 1:max(label_Tp(:))
   labelscaled_Tp(label_Tp==i) = Areas_Tp(i,2);
end

% Transformation with barcode
```

```
labelscaled_Tb = label_Tb;
for i = 1:max(label_Tb(:))
    labelscaled_Tb(label_Tb==i) = Areas_Tb(i,2);
end

%%  Calculate areas in mm2
pixeltomm = 25/areaPixels_goldStandard; % the checkerboard squares are 5x5 mm2

% convert areas in pixels to mm2:
% Non-transformed image
for object = 1:length(Areas_nonTransformed)
    Areas_nonTransformed(object,3) = Areas_nonTransformed(object,1) * pixeltomm;
end

% Transformed with placemat
for object = 1:length(Areas_Tp)
    Areas_Tp(object,3) = Areas_Tp(object,1) * pixeltomm;
end

% Transformed with barcode
for object = 1:length(Areas_Tb)
    Areas_Tb(object,3) = Areas_Tb(object,1) * pixeltomm;
end

%% From here: apply mask for selected region of interest (ROI) on the arm
% load masks for ROI
percentage = "0.96";
mask_nonTransformed_ROI = im2double(imread(folder + name + "_mask_ROI_" + percentage + "_nonTransformed.png"));
mask_Tp_ROI = im2double(imread(folder + name + "_mask_ROI_"+ percentage + "_Tp.png"));
mask_Tb_ROI = im2double(imread(folder + name + "_mask_ROI_"+ percentage + "_Tb.png"));

% convert to binary but remain class double
mask_nonTransformed_ROI(mask_nonTransformed_ROI>0) = 1;
mask_Tp_ROI(mask_Tp_ROI>0) = 1;
mask_Tb_ROI(mask_Tb_ROI>0) = 1;

% apply masks on labeled areas
label1_nonTransformed = label_nonTransformed.*mask_nonTransformed_ROI;
label1_Tp = label_Tp.*mask_Tp_ROI;
label1_Tb = label_Tb.*mask_Tb_ROI;

% binarize the labeled image with adjusted height
label1_bi_nonTransformed = im2double(label1_nonTransformed>0);
label1_bi_Tp = im2double(label1_Tp>0);
label1_bi_Tb = im2double(label1_Tb>0);

% erosion
label1_erode_nonTransformed = imerode(label1_bi_nonTransformed,SE);
label1_erode_Tp = imerode(label1_bi_Tp,SE);
label1_erode_Tb = imerode(label1_bi_Tb,SE);

% number the objects again
label1_numbered_nonTransformed = bwlabel(label1_erode_nonTransformed);
label1_numbered_Tp = bwlabel(label1_erode_Tp);
label1_numbered_Tb = bwlabel(label1_erode_Tb);

%% Calculate areas in pixels (for ROI)
% pre-allocate arrays
Areas1_nonTransformed = zeros(max(label1_numbered_nonTransformed(:)),1);
Areas1_Tp = zeros(max(label1_numbered_Tp(:)),1);
Areas1_Tb = zeros(max(label1_numbered_Tb(:)),1);

centroids_nonTransformed = zeros(max(label1_numbered_nonTransformed(:)),2);
```

```
centroids_Tp = zeros(max(label1_numbered_Tp(:)),2);
centroids_Tb = zeros(max(label1_numbered_Tb(:)),2);

% Non-transformed
for object = 1:max(label1_numbered_nonTransformed(:))    % number of squares in the masked image
   centroid = regionprops(label1_numbered_nonTransformed == object,'Centroid');  % centroid of a square
   centroids_nonTransformed(object,:) = [round(centroid.Centroid(2)),round(centroid.Centroid(1))]; % centroids of squares
   Areas1_nonTransformed(object) =
label2_nonTransformed(centroids_nonTransformed(object,1),centroids_nonTransformed(object,2)); % pixel value of square
on the location of centroid coordinates
end

% Transformed with placemat
for object = 1:max(label1_numbered_Tp(:))
   centroid = regionprops(label1_numbered_Tp == object,'Centroid');
   centroids_Tp(object,:) = [round(centroid.Centroid(2)),round(centroid.Centroid(1))];
   Areas1_Tp(object) = label2_Tp(centroids_Tp(object,1),centroids_Tp(object,2));
end

% Transformed with barcode
for object = 1:max(label1_numbered_Tb(:))
   centroid = regionprops(label1_numbered_Tb == object,'Centroid');
   centroids_Tb(object,:) = [round(centroid.Centroid(2)),round(centroid.Centroid(1))];
   Areas1_Tb(object) = label2_Tb(centroids_Tb(object,1),centroids_Tb(object,2));
end

%% Convert areas in pixels to relative values and to mm2
pixeltomm = 25/areaPixels_goldStandard; % the checkerboard squares are 5x5 mm2

% Non-transformed image
for i = 1:length(Areas1_nonTransformed)
   Areas1_nonTransformed(i,2) = Areas1_nonTransformed(i)/areaPixels_goldStandard;  % Calculate relative area (gold
standard is value 1)
   Areas1_nonTransformed(i,3) = Areas1_nonTransformed(i) * pixeltomm;            % Pixels to mm2
end

% Transformed with placemat
for i = 1:length(Areas1_Tp)
   Areas1_Tp(i,2) = Areas1_Tp(i)/areaPixels_goldStandard;
   Areas1_Tp(i,3) = Areas1_Tp(i) * pixeltomm;
end

% Transformed with barcode
for i = 1:length(Areas1_Tb)
   Areas1_Tb(i,2) = Areas1_Tb(i)/areaPixels_goldStandard;
   Areas1_Tb(i,3) = Areas1_Tb(i) * pixeltomm;
end

%% Visualization with colormap
% apply masks on relative areas
labelscaled1_nonTransformed = labelscaled_nonTransformed.*mask_nonTransformed_ROI;
labelscaled1_Tp = labelscaled_Tp.*mask_Tp_ROI;
labelscaled1_Tb = labelscaled_Tb.*mask_Tb_ROI;

% Set all black pixels to NaN (for visualization)
labelscaled1_nonTransformed(labelscaled1_nonTransformed==0) = NaN;
labelscaled1_Tp(labelscaled1_Tp==0) = NaN;
labelscaled1_Tb(labelscaled1_Tb==0) = NaN;

% Non-transformed
figure,
h = imshow(labelscaled1_nonTransformed,[lowerLim,upperLim],colormap=cmap);   % imshow with colormap
set(h, 'alphadata',~isnan(labelscaled1_nonTransformed))
set(gca,'color','white')
colorbar;
title("Non-transformed "+percentage)

% Transformed with placemat
figure,
h = imshow(labelscaled1_Tp,[lowerLim,upperLim],colormap=cmap);  % imshow with colormap
set(h, 'alphadata',~isnan(labelscaled1_Tp))
set(gca,'color','white')
colorbar;
title("Transformed with placemat "+percentage)

% Transformed with barcode
figure,
h = imshow(labelscaled1_Tb,[lowerLim,upperLim],colormap=cmap);  % imshow with colormap
set(h, 'alphadata',~isnan(labelscaled1_Tb))
set(gca,'color','white')
colorbar;
title("Transformed with barcode "+percentage)
```

B. Results of the correction for camera perspective: images after projective transformation

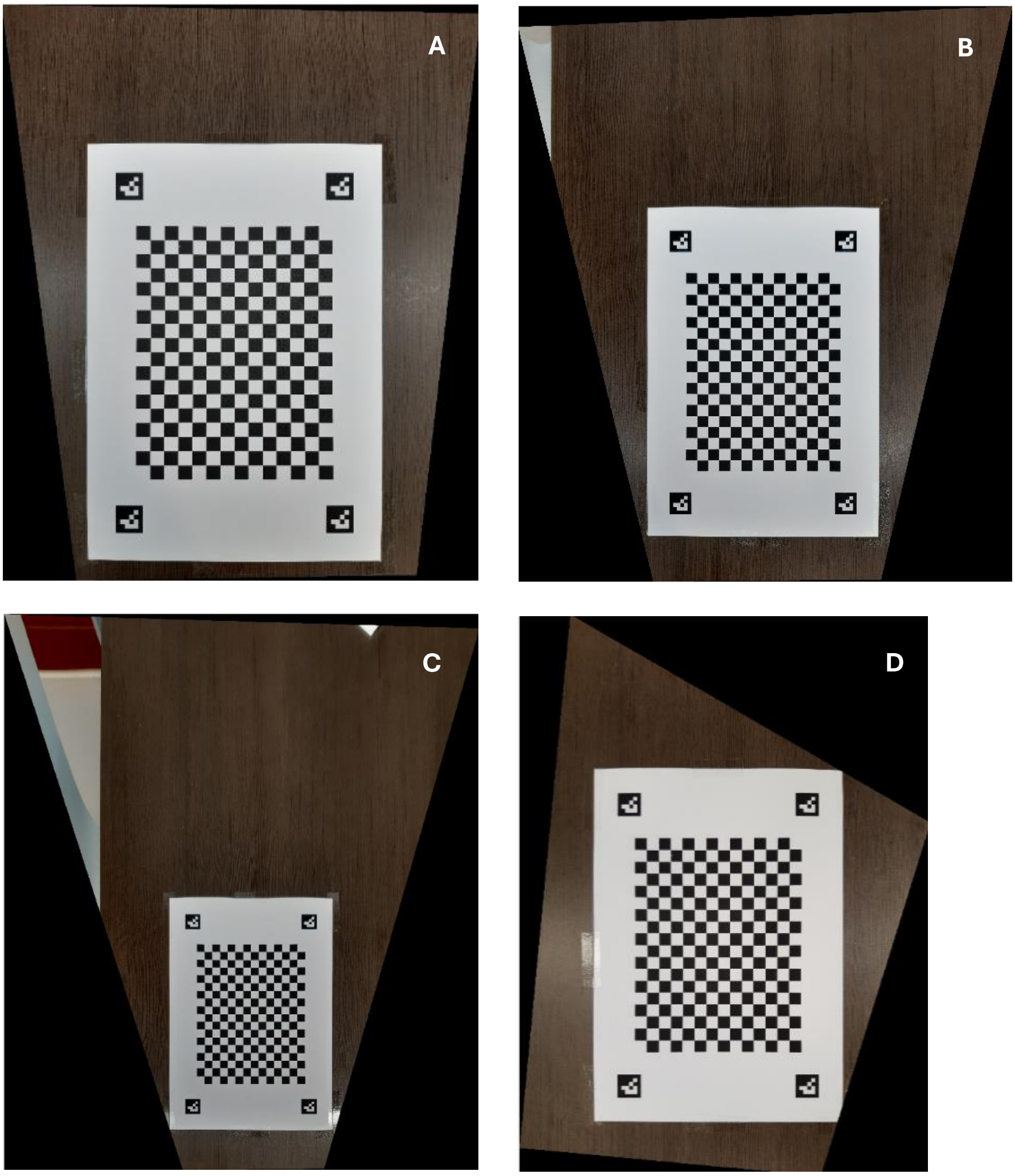


Figure 25 After projective transformation: photographs of the placemat with checkerboard corrected for different camera perspectives: 15° pitch (A); 30° pitch (B); 45° pitch (C); 20° pitch and 10° roll (D). The black areas in the images are regions that are filled up to return a rectangular image after the transformation resulted in irregular shaped and stretched images.

C. Results of the arm curvature correction

C1. Images before and after curvature correction with placemat and barcode

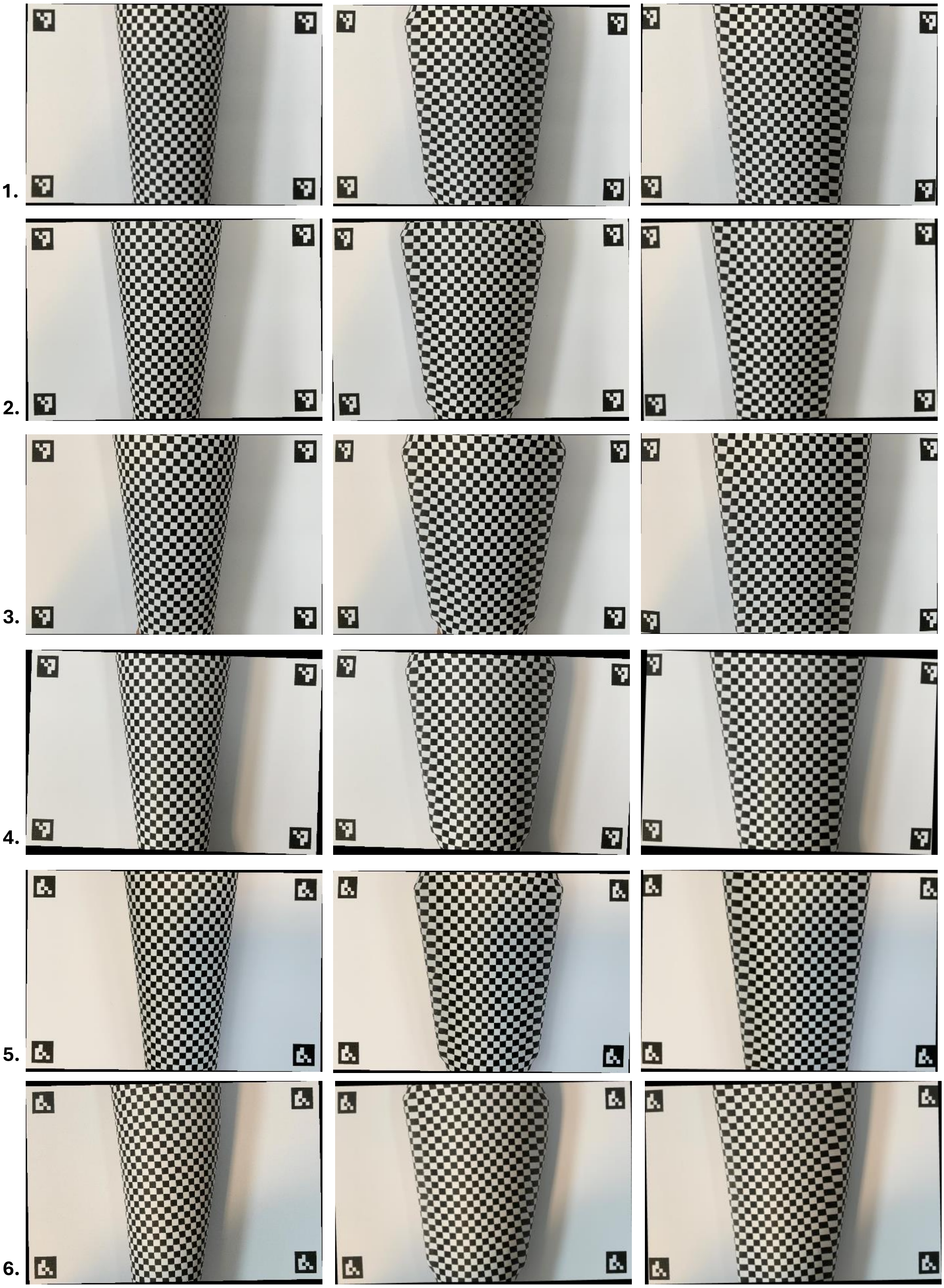


Figure 26 Images for validation of the arm curvature correction of each individual subject: before correction (left) and after correction using the placemat (middle) and the barcode stickers (right).

C2. Source maps of placemat method

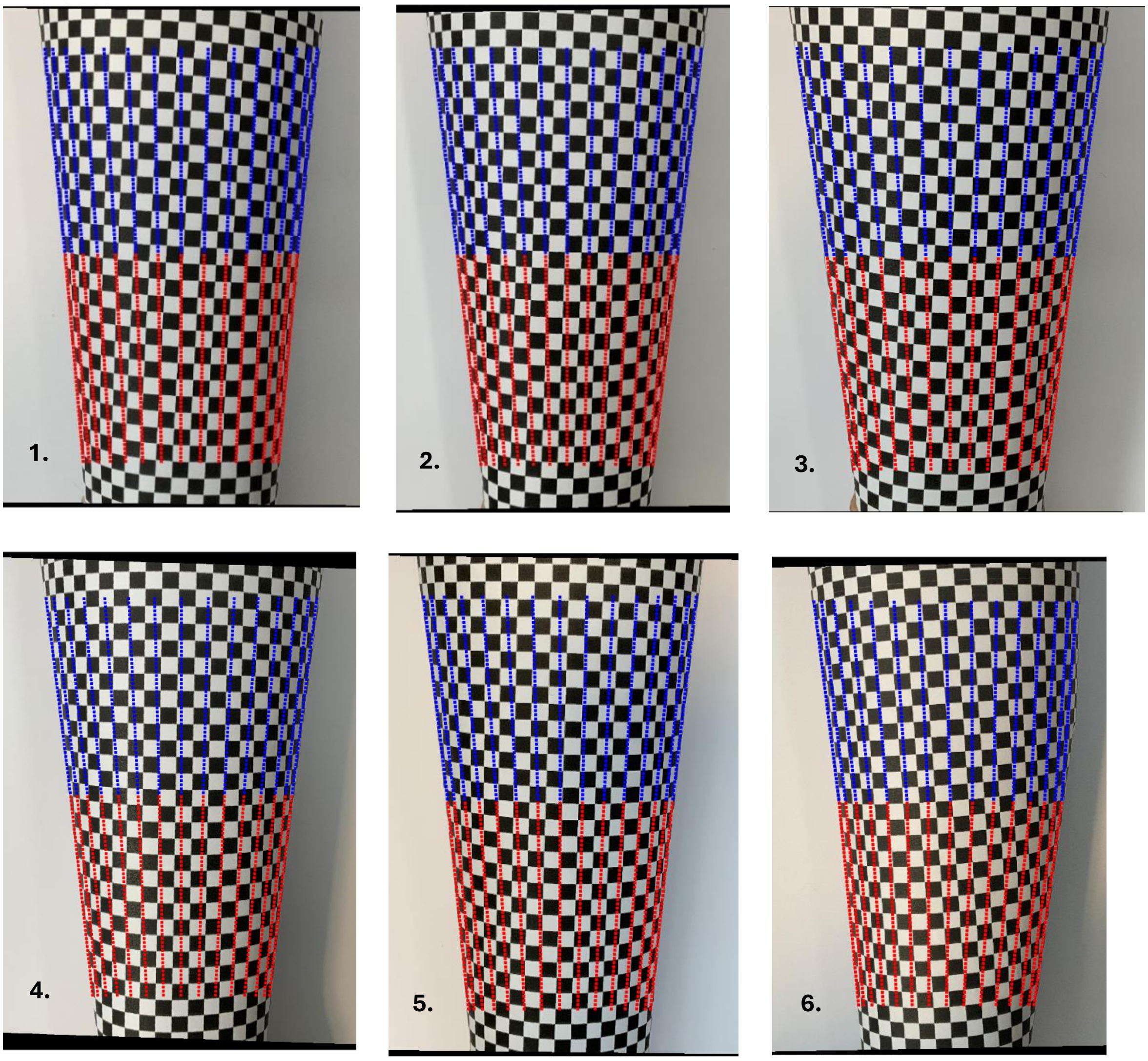


Figure 27 Source maps obtained from the placemat method, for each subject. The blue and red points indicate the two separately created cylinders in this approach, formed by the two extra source points that are added halfway between the ArUco markers.

C3. Source maps of barcode method

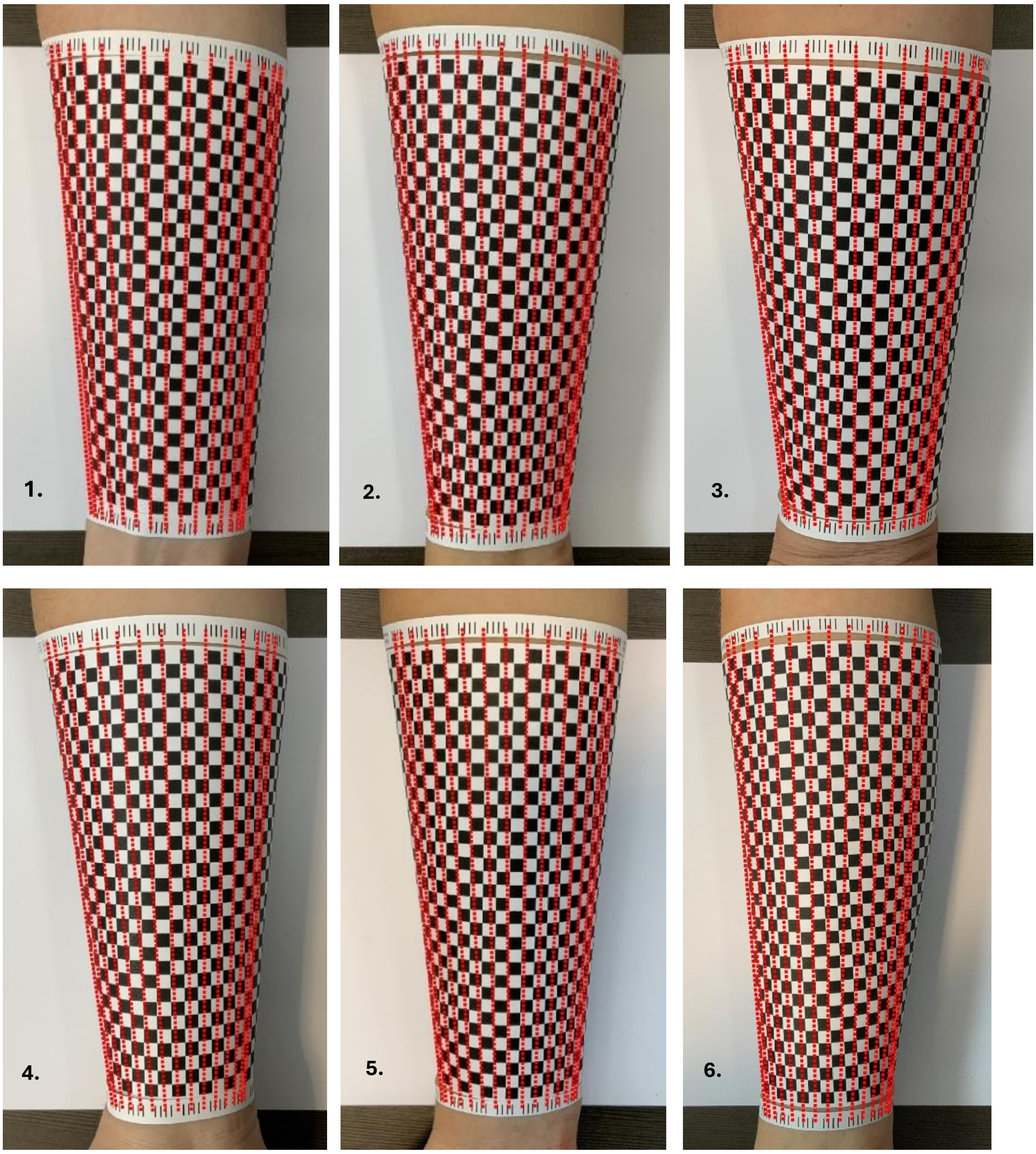


Figure 28 Source maps obtained from the barcode method, for each subject.

D. Discrepancy between source key points and interpolated points

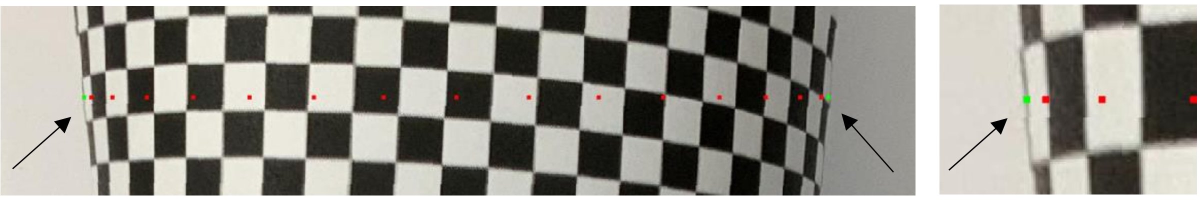


Figure 29 Illustration of the mismatch between the left and right source key points and the interpolated points for the arm curvature correction with the placemat (right image: zoomed in).

E. Result of using the Lab color space for arm segmentation

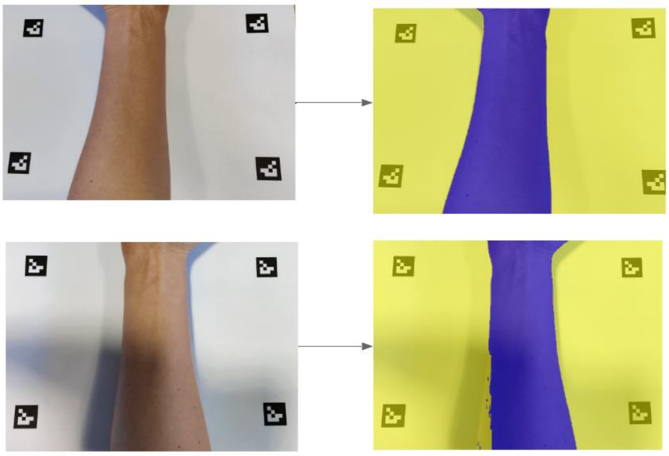


Figure 30 Example of a successful (upper) and failed (lower) segmentation of the forearm using the Lab method in the presence of lighter and darker shadows.