DOI: 10.15439/2022F8
ISSN 2300-5963 ACSIS, Vol. 32

Neural Network Enhanced Automatic Garment Measurement System

Paweł Kowaleczko*[1,2], Przemysław Rokita[1], Marcin Szczuka[3,4]
[1] Institute of Computer Science, Warsaw University of Technology
ul. Nowowiejska 15/19, 00-665 Warsaw, Poland
[2] QED Software sp. z o.o.
ul.Miedziana 3A/18, 00-814 Warsaw, Poland
[3] Institute of Informatics, University of Warsaw
Banacha 2, 02-097 Warsaw, Poland
[3] BAKERS sp. z o.o.
ul. Branickiego 11/154, 02-972 Warsaw, Poland
Email: pawel.kowaleczko.dokt@pw.edu.pl, pro@ii.pw.edu.pl, szczuka@mimuw.edu.pl

Abstract—**The measurement of garments is most often a very laborious task. Automatic garment measurement systems may be thus a great convenience in fashion e-commerce cataloguing issues. In this paper, we propose an automatic garment measurement system that uses classical computer vision algorithms, as well as an error correction neural network, which reduces the overall error. We make use of data collected by our partner, which contains photographs of garments with ArUco markers. Using such data, we estimate the coordinates of feature points, which are used to calculate a specific size of a garment. We apply the error correction neural network to this measured size to minimize the error. The conducted experiments show, that our method is a useful tool that meets the requirements of practicality and its results are comparable with the current state of the art methods. Additionally, our error correction neural network is a novelty in the field of automatic garment measurement and there is no need for the garments templates, which are used in the previous solutions.**

I. Introduction

IN RECENT years, a huge increase of interest in online shopping can be observed. Since 2017, the global e-commerce market almost doubled its value [1]. Because of its convenience, more and more people do their shopping online. However, shopping for garments may very often be problematic because of the inconsistency in sizes grading. The same piece of cloth may be tagged as size S by one producer, and as size M by another. This inconsistency is the cause of many returns of orders, which is one of the main problems and unnecessary expenses of running a fashion retail business online. To reduce the risk of return it is beneficial to additionally provide the specific dimensions, like sleeve length, waist width etc., in one of the common metrics - centimetres or inches.

The accurate measurement technique is particularly useful in the case of the online second-hand shops. The clothes sold there come from different suppliers and producers, whose size tables may be varied and inconsistent. In some cases, the clothes may have no indication of size whatsoever.

The solutions proposed in this paper are an answer to real-life challenges encountered during the commercial R&D project. The Kidihub (kidihub.com) online second-hand shop specialises in gathering, listing and selling used children's clothes. In Kidihub's business model the clothes are being sent to the company's warehouse by parents, and then assessed and qualified by the staff. The assessment process not only involves verification of quality and state (new, like new, small defect, ...) of the particular piece of clothing, but also leads to more extensive labelling of items. This labelling (tagging) is indeed the main focus of the R&D project. Once a piece of clothing is qualified as worthy listing, it is being ironed and arranged for picture taking. Then, the items need to be tagged with a number of feature values such as sex, colour, pattern, type, and – most crucially – size. The strive to increase the automation of measurement process and reduce the human involvement in the arrangement of clothes and picture-taking is what drives this research.

We propose a method for automatic garment measurement which uses as an input a photograph of a piece of cloth. This photograph includes ArUco marker [2] for the sake of the need to determine the pixel to centimetre ratio to calculate the dimensions in centimetres. However, any marker, even a piece of cardboard, can be used for this task if the detection algorithm of this marker is implemented.

We extract the contours of the garment and find the key feature points essential to determine the specific dimensions, such as points where sleeves end, hips edge points of pants etc. We calculate the Euclidean distance between those points and correct the error using a neural network. The main task of this network is to reduce the calibration error - the photos were taken in different camera to plane distances setups and only a single ArUco marker was provided, so the calibration was not possible. This neural network is also the main novelty of our work, one that makes it possible to reduce the error

This research was co-funded by Smart Growth Operational Programme 2014-2020, financed by European Regional Development Fund, in frame of project POIR.01.01.01-00-0420/20, operated by National Centre for Research and Development in Poland.
*Corresponding author

by roughly 30%. This paper is organized as follows - in Section 2. we shortly describe all related works available in the public domain. In the next section, we describe the dataset collected by our partner and the methods of filtering that we applied, as well as preprocessing methods and implementation details. In Section 4. we explain implemented measurement algorithms in detail for each of the garment types separately. In the next section, we introduce the main novelty of our work, error correction neural network - we explain the purpose of this network, input data preparation methods and architecture. In Section 6. we present the performance of our system, especially laying emphasis on the improvement that the error correction neural network introduces. In the last section, we summarize our work and suggest potential ways of future development and improvement of our system and, in general, the whole area of automatic garment measurement.

II. Related Work

Even though many automatic garment measurement systems exist, most of them are based on determining the physical features of the clothes, where special devices need to be dressed and often photographs from multiple views have to be taken. The most comprehensive overview at all of the methods is presented in [3].

There are only a few works [4], [5], [6] of automatic measurement systems based on the computer vision algorithms. All of them (as in the case of our system) focus on determining the key points based on the corner detection algorithm. Those works then focus on matching the contour of the photographed garment with the predefined template of the garment type contour. Those key points are then matched with the points that are marked on the template contour. We decided to use a different approach, where we do not use any template.

Our automatic measurement module will be a part of a bigger system of automatic garment tagging, part of which will be a garment type classifier. Assuming that we have the garment type information, we defined four measurement algorithms, each for one type of garment. Those types include pants, skirts, sleeveless dresses and top garments with sleeves (t-shirts, jackets, sweaters etc.). To our surprise, none of the previously developed methods make use of deep neural networks, which currently outperform classic algorithms in most computer vision tasks. The main limitation in applying deep neural networks to the automatic garment measurement may be the fact, that they need large amounts of tagged data to train.

III. Dataset and Preprocessing

A. Dataset

We use a dataset collected by Bakers company (the owner of Kidihub store). This dataset consists of 900 images of near HD resolution with various height to width ratios (see Fig. 1). There are nine different types of garments (coats, dresses, jackets, jerseys, shirts, shorts, skirts, tops, trousers). These nine types are later grouped into four groups - tops (coats, jackets, jerseys, shirts, tops), skirts, pants (shorts, trousers) and dresses. To increase the contrast, photos of dark garments were taken on a white background. For the same purpose, bright garments were photographed on black background.

Unfortunately, some parts of the dataset did not meet our contrast requirements so those photos had to be removed. The other cause of rejection was the improper orientation of the photos - some pictures were flipped by 90 degrees. We were unable to rotate them back automatically using the ArUco marker orientation, because in many cases the marker itself was also rotated by some angle. Those requirements resulted in a rejection of 76 pictures, leaving 824 pictures for further processing. All ground truth labels for each picture, such as garment type, color, material and dimensions were annotated by the human operator during the collection process.

B. Preprocessing

Extracting the contour of the garment is essential for the feature points extraction task. We reduce the size of each image so the width of the image is 500 pixels and the height is properly adjusted to the width so the original proportions of images are retained. We apply gamma correction with $\gamma = 0.4$ and increase the contrast of the image multiplying all pixel values by two (adjusted to the maximum 8-bit pixel value). Next, the Canny filter is applied to detect the edges and we dilate the image twice to avoid the situations where there are some gaps in the contour which makes it impossible to detect.

Using such preprocessing methods, the maximal outer contour (which, as we assume, belongs to the garment) is found. Additionally, we detect the ArUco marker (using the function from OpenCV library) and calculate the pixel to cm ratio (real, measured by human operator, ArUco marker size is 10cm) which we then use to adjust the value of the distance between two feature points to be given in centimetres. The general flowchart of our algorithm is outlined in Fig 2.

C. Implementation Details

We decided to use the Python programming language for implementation purposes. Python is known for its syntax simplicity which is the cause that it is currently one of the most popular programming languages. Because Python uses an interpreter, not a compiler, it is considerably slower than other programming languages, like C, C++ or Java. However, in our task the inference time (if it is in reasonable range) is not that important.

We make use of computer vision algorithms implemented in the OpenCV library [7]. For training deep neural networks task we use PyTorch library [8] with its wrapper called PyTorch Lightning [9]. If not stated otherwise we use all functions from those libraries with default parameters values.

IV. Baseline Measurement Model

A. Corner points detection

To find the specific dimensions of a garment, we have to determine the coordinates of feature points, which are usually the corner points of the contour. We focus on analysing the variability of the coordinates of points that are part of the

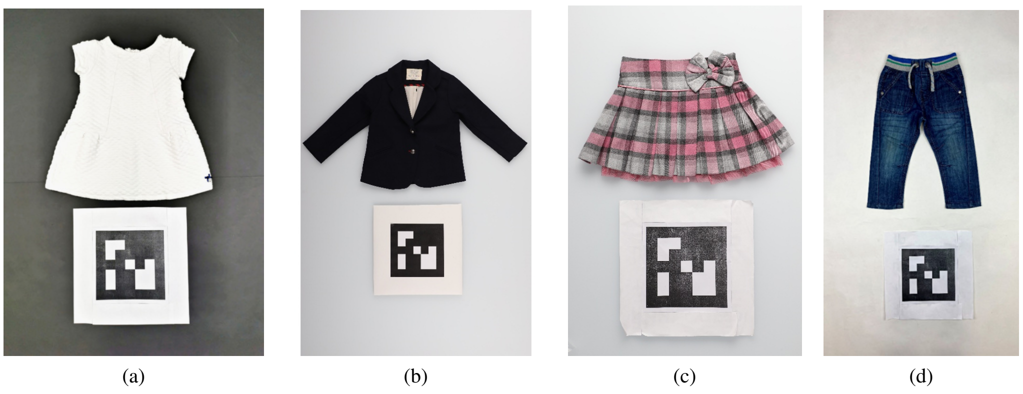


Fig. 1: Example pictures from the dataset: dress (a), jacket (b), skirt (c) and trousers (d)

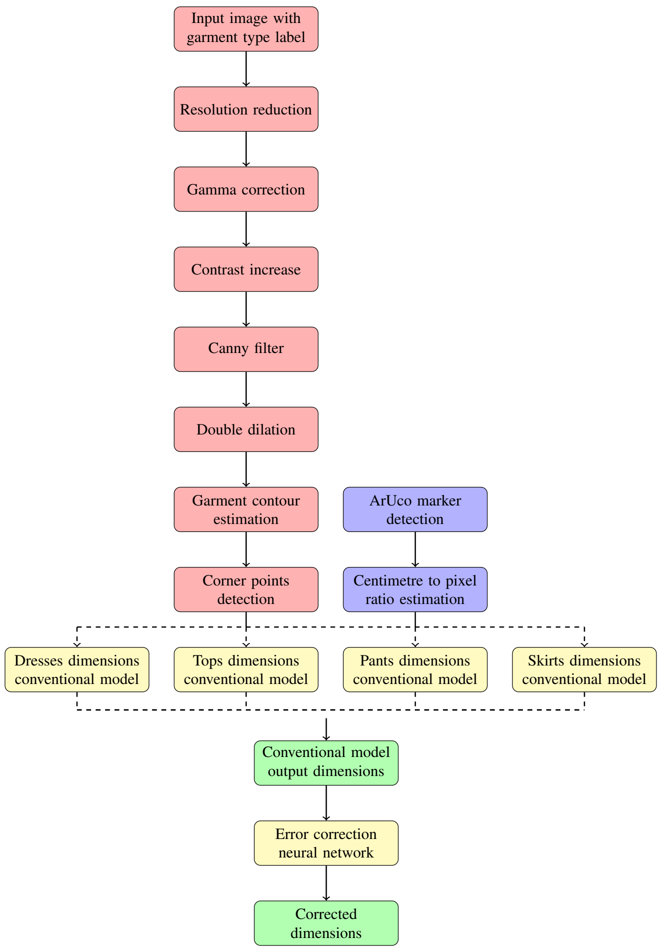

Input image with garment type label
Resolution reduction
Gamma correction
Contrast increase
Canny filter
Double dilation
Garment contour estimation
ArUco marker detection
Corner points detection
Centimetre to pixel ratio estimation
Dresses dimensions conventional model
Tops dimensions conventional model
Pants dimensions conventional model
Skirts dimensions conventional model
Conventional model output dimensions
Error correction neural network
Corrected dimensions

Fig. 2: The flowchart of our proposed measurement method. Preprocessing steps are outlined as red blocks, blue blocks are associated with AruCo marker operations, yellow blocks correspond to dimensioning models' blocks and green block are estimated dimensions blocks. Dashed lines are used in cases where only one processing path is possible.

found contour. Those contour points are ordered counterclockwise in a Python array. If x's or y's derivative with respect to point index in array ($\frac{dx}{d(idx)}$, $\frac{dy}{d(idx)}$) changes its value rapidly (it is discontinuous) we can suspect, that this point is a corner point. We decided to use the index step (step used in calculating the derivatives) equal to five (based on empirical testing) to make the derivative smoother. After calculating the derivatives we calculate the absolute differences between the following derivatives (also using step value equal to five), which should result in a plot of peaks visible in Figure 3. We sum up those differences calculated for the x and y coordinates and find the most distinctive peaks. The whole process is presented in Figure 3. We use those corner points as candidates for the extreme points selection, which are described in the following section, as well as candidates for some of the feature points.

B. *Pants Measurement*

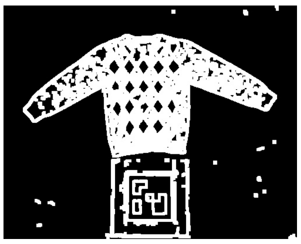
In the pants measurement task, we need to find the length of pant legs and waist width. We assume that the waist section of the photographed pants is not heavily curved, because we calculate the distance only between two extreme waist points. In this case, we call the feature points that we need to find the extreme points - the outer ends of pant legs and waist points are the four points that are the furthest points of contour with respect to top left, top right, bottom left and bottom right corners of the image. These points are selected from the set of detected corner points. To find the waist width, we calculate the euclidean distance between the top left and top right points of the contour. Pantlegs length is the average of distances between the top left - bottom left and top right - bottom right points.

C. *Skirts Measurement*

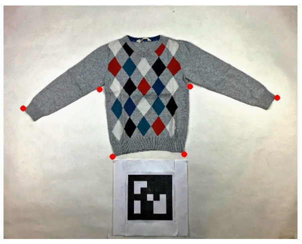
In this task, we need to determine the waist width and total length of the garment. Waist width is found in the same way as it was in the case of pants. To find total length we calculate an approximate middle vertical axis value (the mean value of the x coordinates of the top left and top right points of the contour). Then we find two intersection points of this axis with the contour points - one in the top part of the contour and one in the bottom.

D. *Sleeveless Dresses Measurement*

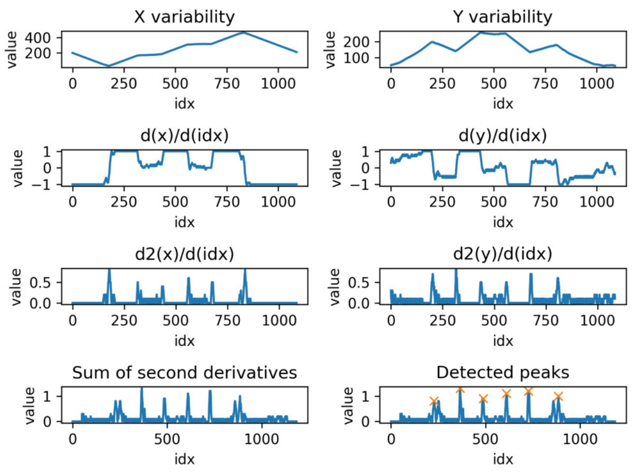
In the sleeveless dresses measurement task we need to determine the total length (which is done the same ways as in the case of skirts) and width in the chest. To find chest width we first define the chest area which is the range of y values included in the subjectively selected range (0.3h, 0.85h) where h is the vertical span of the contour of the garment. We once again define the middle vertical axis the same way, as it was defined in the measurement of the skirts. Then for each y from range (0.3h, 0.85h) and constant value of x vertical axis (point (x_v, y_n)) we find the points from the left and right part of the contour (contour is divided by vertical axis), which distances to those (x_v, y_n) points are the smallest. This way we obtain pairs of points from the left and right parts of the contour, and chest width is the pair which distance is the smallest.



(a)



(b)



(c)

Fig. 3: Corner detection algorithm. (a) image with Canny filter applied, (b) image with marked corner points, (c) plots of variabilities of x and y coordinates of the contour with potential corner points marked with orange crosses on peaks plot.

E. *Tops Measurement*

Tops measurement was the most challenging task. We need to find bottom width, chest width, sleeves lengths and total length.

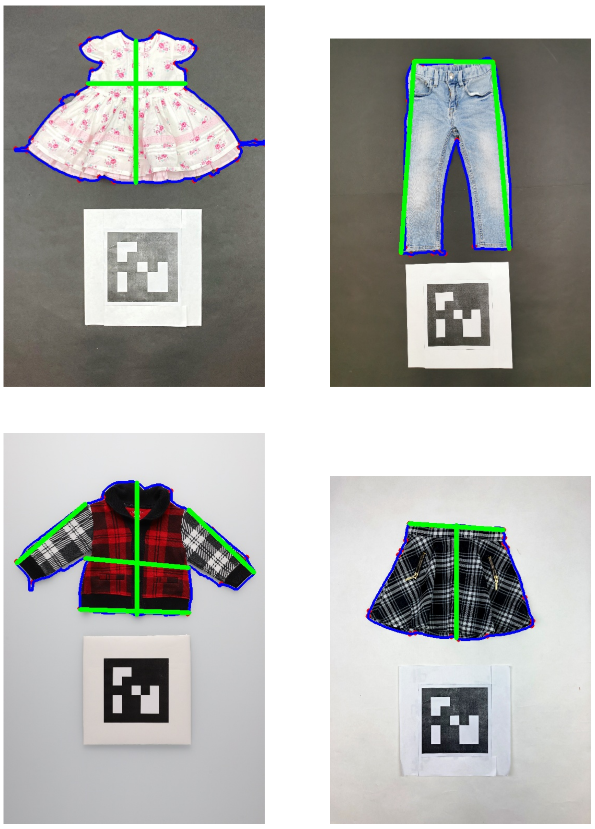


Fig. 4: Output from our automatic measurement system. Measurement dimensions are marked as green lines. Garments are cased in blue contours.

To find the bottom (waist) width we once again use the vertical axis to split the contour into two parts and specify the minimal y value over which (counting from the top of the image, where $y = 0$) the bottom line should be. We arbitrarily choose this value to be in the 0.85 of the vertical span of the garment (difference between top and bottom point of the contour). We divide the contour using the vertical axis and minimal y into the bottom left and bottom right parts. Those parts may include not only the bottom part of the contour, which is our area of interest but also parts of the sleeves. However, those sleeves parts of the contour are separated from our area of interest - the bottom part of the contour. To filter out those sleeve parts we investigate the variability of the values of the x coordinates of these parts - if the distance from the vertical axis increases or decreases rapidly (there is a big gap betwen the x coordinates of the contour points) we reject those points. Then we find the maximal distance between the vertical axis and points of contour left after applying the filtering procedure. The maximal distance for the left and right parts of the contour is equal to the sought bottom width.

We select chest points out of all corner points of the contour. To find chest feature points we firstly restrict the area where

TABLE I: Experimental results of our measurement module on the test dataset. Mean percentage errors and standard deviations of errors are presented for the results from the baseline model and error correction neural network model. In the last column the shares of corrected dimensions in relation to all dimensions are presented.

	Mean percentage error [%]	Error SD [%]	Reduced mean percentage error [%]	Reduced error SD [%]	Percentage corrected [%]
Tops	6.36	5.24	4.10	3.30	66.4
Skirts	4.56	3.30	4.20	3.44	69.2
Pants	7.13	5.29	4.01	2.91	76.7
Dresses	6.08	4.82	3.92	2.66	63.6
All	6.45	5.14	4.07	3.16	69.2

the corner points may be in the $x_b \pm 0.15 \cdot I_{width}$ area, where x_b is the x coordinate of one of the bottom points and I_{width} is the width of the image. We also reject all the feature points for which y coordinates are greater than $y_b - 0.1 \cdot I_{height}$. This way we propose candidates for the potential left and right chest points out of which we select the pair which distance is the smallest.

We determine shoulders points (which are used to find the sleeve lengths and are not corner points, because the curvature of the contour is too small) by choosing the points from the top part of the contour which intersects with two vertical axes. Those axes are x coordinates of the chest points. To get total sleeve lengths we apply contour corner detection algorithms starting from the left shoulder point and going counterclockwise on the contour for the left sleeve and starting from the right shoulder point going clockwise for the right sleeve. The first corner point found is identified as the sleeves feature point. We make sure that the monotonicity changed permanently to reject "local" corner points, which may be a result of some distortions in the contour.

After obtaining all feature points in the way described above we calculate Euclidean distances between them. Sleeve length is the average of right and left sleeve lengths, similarly to the pantlegs lengths.

V. Error Correction Neural Network

After obtaining the dimensions of a measured garment and recounting it into centimetres (using the ArUco marker of known size), there are still possible sources of errors. The main source of errors may be the lack of camera calibration which is essential to reduce the impact of lenses distortions errors. There may be also some imperfections in determining the coordinates of the feature points and we do not take into account the curvature of the contours.

To overcome those issues (especially to simulate the camera calibration) we propose the error correction neural network. This network takes as an input a 14 element vector which includes x and y coordinates of all four ArUco marker corners and two feature points used in the specific measurement process. The two remaining elements of the input are image width and height. The task of this network is to predict the normalized residual, which is the difference between target dimension value (measured by human operator and taken from the dataset) and baseline model prediction (measured by our model) in relation to baseline model prediction (relative percentage error). Such prediction may reduce the total error of our baseline model.

The model itself consists of four fully connected layers with 64, 128, 64, 32 nodes respectively and returns single value - relative percentage error - which may be both positive and negative number. Each layer is followed by batch normalization layer, leaky ReLU activation and dropout (with the probability of dropout of 0.3) layers. We use Adam optimizer and MSE loss function. We did not focus our efforts on finding the most optimal values of network parameters or architecture. This model is trained on the dataset of all independent dimensions evaluated by our base model where the predicted dimensions error is no greater than the 20% of the target dimension. We apply this constraint to reject the predictions which were obvious errors, however, it is not guaranteed that all dataset elements left are correct. Such a dataset have 2369 elements. 90% of the dataset is then used for training and 10% for testing. The final measurement of our system $Dim_{corrected}$ is obtained by applying the simple equation:

$$Dim_{corrected} = (1 + ECNN_{out}) \cdot Dim \qquad (1)$$

where $ECNN_{out}$ is the output of the error correction neural network, and Dim is the output dimension from the conventional model.

VI. Experimental Results

As it was stated in the previous section, we reject those measurements for which the error is greater than 20%. We assume that such errors will be easy to notice on the output image (Figure 4) and those images will be rejected by a human operator after just taking a look at the result of our baseline model measurement. All ground truth values are measured by the employees of Kidihub during the process of photographing. In Table I, we present the experimental results per each category of our baseline automatic measurement system tested on test dataset. Each of our measurements is a Euclidean distance between two specific key points (which are described in detail in the previous section). We do not consider the curvature of the cloth, hence our requirement that the garment should be tiled on a photograph. The mean percentage relative error (relative error of measured size in centimetres in relation to ground truth size) varies from 4.56% for skirts to 7.13% for pants and is 6.45% on average for all garments (row All). The standard deviation of error for all categories is also quite high - 5.14%. After applying the error correction neural network the mean percentage error is significantly reduced for each category, and on average by almost 30%. The average error for all categories is reduced to 4.07%. For each category, the majority of measurement results errors were reduced (on average for 69.2% of data). Target data resolution, which is 1cm, also have a significant impact on the final error value. The value of the average error of which the source is the measurement resolution is 3.16%. One should also have in mind that the process of measuring the garments may also produce some kind of human-related error - the measured piece of cloth may be partially folded or the human annotator may read the result of measurement incorrectly. Because of this quite significant level of measurement resolution error, as well as potential human annotator errors, it is hard to tell if our method outperforms state of the art solutions, of which errors are about 2-3%.

VII. Conclusions

We proposed a set of fast automatic measuring algorithms for four different categories of garments. We designed those algorithms with the quality and characteristics of the dataset from our partner, as well as their requirements in mind. The pictures must contain tiled garments on a contrasting background with the specific ArUco marker. We also must know the category of the garment, however, this step will be automated in our system by applying garment classification neural network in the future. We introduced the error correction neural network, which reduces the error on average by 2.38%. This neural network does not need any additional information as an input, it uses only information extracted from the images by classic computer vision algorithms. Application of this neural network reduces the mean average error to about 4%, however very significant is the fact, that the resolution of the ground truth dimensions that we compare to is rather low (1cm) which results in large measurement resolution error of 3.16% on average. Taking this into account our method is very close to the state of the art methods (about 2-3% of relative error) and do not need a special shooting setup, camera calibration and garment category template as the previous methods did.

There are still many areas of our system, that can be improved. We especially have in mind our baseline algorithm in this regard. We pin our hope on involving more neural network solutions. In our opinion, if a dataset of images and all feature points would be provided, the neural network trained on such dataset would easily outperform state of the art methods. However, collecting such a dataset is a laborious task. Our solution also would surely benefit from filtering the output data from our baseline model on which we train the error correction neural network. We also think that it would be beneficial to check how our solution works if the calibration pattern was present on the images. The reason for this is that we are not entirely sure what types of errors and to what extent the error correction neural network reduces. If those errors were orthogonal to the calibration error, such a system would probably outperform current state of the art methods without the additional constraints.

References

[1] S. Chevalier, Retail e-commerce sales worldwide from 2014 to 2024. URL https://www.statista.com/statistics/379046/worldwide-retail-e-commerce-sales/

[2] A. Rosebrock, Detecting ArUco markers with OpenCV and Python - an online tutorial. URL https://www.pyimagesearch.com/2020/12/21/detecting-aruco-markers-with-opencv-and-python/

[3] J. Juvonen, Study of the automatic garment measurement, report prepared for ROBOCOAST EU network by Aarila Dots Oy. (2019). URL https://new.robocoast.eu/wp-content/uploads/2020/09/Feasibility-study-Automatic-garment-measurement_Aarila-Dots.pdf

[4] L. Cao, Y. Jiang, M. Jiang, Automatic measurement of garment dimensions using machine vision, in: 2010 International Conference on Computer Application and System Modeling (ICCASM 2010), Vol. 9, 2010, pp. V9–30–V9–33. doi:10.1109/ICCASM.2010.5623093.

[5] K. Chen, Image analysis technology in the automatic measurement of garment dimensions., Asian Journal of Information Technology 4 (9) (2005) 832–834. URL https://medwelljournals.com/abstract/?doi=ajit.2005.832.834

[6] C. Li, Y. Xu, Y. Xiao, H. Liu, M. Feng, D. Zhang, Automatic measurement of garment sizes using image recognition, in: Proceedings of the International Conference on Graphics and Signal Processing, ACM, 2017, pp. 30–34. doi:10.1145/3121360.3121382.

[7] G. Bradski, A. Kaehler, The OpenCV Library, Dr. Dobb's Journal of Software Tools 3 (2000).

[8] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, Pytorch: An imperative style, high-performance deep learning library, in: H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, R. Garnett (Eds.), Advances in Neural Information Processing Systems (NeurIPS 2019), Vol. 32, Curran Associates, Inc., 2019. doi:10.48550/arXiv.1912.01703. URL https://arxiv.org/abs/1912.01703

[9] W. Falcon, et al., PyTorch Lightning, gitHub repository. (2019). URL https://github.com/PyTorchLightning/pytorch-lightning