2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) | 978-1-6654-4899-4/21/$31.00 ©2021 IEEE | DOI: 10.1109/CVPRW53098.2021.00269

Real-time analogue gauge transcription on mobile phone

Ben Howells
University of Cambridge
bmh39@cam.ac.uk

James Charles
University of Cambridge
jjc75@cam.ac.uk

Roberto Cipolla
University of Cambridge
rc10001@cam.ac.uk

Abstract

The objective of this paper is to automatically read any circular single pointer analogue gauge in real-time on mobile phone. We make the following contributions: (i) we show how to efficiently and accurately read gauges on mobile phones using a convolutional neural network (CNN) system which accepts both a high and low resolution gauge image; (ii) we introduce a large synthetic image dataset (far superior in size to prior works) with ground truth gauge readings, pointer layout and scale face homographies that is suitable for training a CNN for real world application; (iii) we also release a new real world analogue gauge dataset (larger meter variation than any previous) with annotation suitable for testing three different types of tasks and finally (iv) we beat state of the art performance for gauge reading on this dataset and an existing public dataset in multiple metrics by large margins, notably with pointer angle error less than 1 degree. Our method is fast and lightweight and runs up to 25fps on mobile devices.

Figure 1. Our system runs in real-time on mobile phone and is capable of transcribing unseen analogue gauges to a very high accuracy. The system is trained purely from synthetic data yet transfers very well to real world meters and is robust against the huge appearance variation in real world gauges due to bezel types, pointers and background scales.

1. Introduction

Despite the digital age, analogue gauges are still prevalent in both industrial and private sectors. Examples include the tracking of pressure, speed or temperature in industrial plant equipment or the monitoring of electric, gas and water supply in the typical home. Unfortunately, the push for automatic monitoring and analysis, is resulting in fully operational existing equipment becoming obsolete and abandoned at both huge monetary and environmental cost. Hence, enabling these analogue systems to interface with modern digital ones is of great benefit. To this end, we propose a computer vision based system for rapidly transcribing analogue gauges into a digital format using a mobile phone camera. Such a system would entail simply waving a phone camera over a gauge to securely transfer the reading onto the phone where the data can be further processed.

The majority of analogue (or dial) gauges are circular and employ a moving pointer that directly corresponds to a measurable parameter as indicated on a calibrated scale. Recovering this reading from images or video is difficult for a number of reasons but namely: (1) dim lighting and/or glare (typical in industrial environments), (2) parallax caused by off perpendicular camera view to the plane of the gauge face, and most notably (3) the huge variation of gauge appearances, due to differences in bevel, pointer and face designs (see Figure 1).

To the best of our knowledge we are the first to propose a mobile phone based system for analogue gauge transcription and make the following four contributions: (i) we show how a multitask CNN can be trained (solely with synthetic data) to run efficiently on mobile phone (using a low resolution and high resolution input image), (ii) we generate and release a large synthetic image dataset of 10,000 images, Synthetic-Gauges, that can be used for training a CNN model to transcribe real world gauges, (iii) we also release a new real world analogue gauge dataset for testing purposes on three tasks (gauge detection, perspective correction and reading), and finally (iv) we show our trained CNN beats

978-1-6654-4899-4/21/$31.00 ©2021 IEEE
DOI 10.1109/CVPRW53098.2021.00269

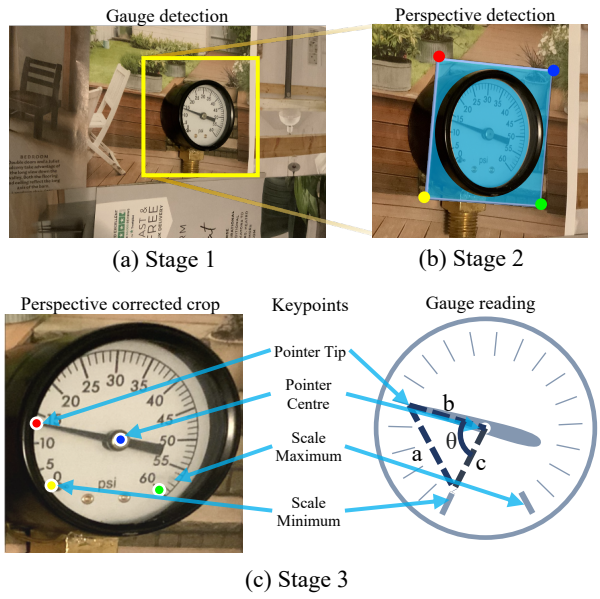


Figure 2. Method overview. Our system is split into three stages: (a) Stage 1 infers the bounding box of gauges in the image and recovers the corners of a virtual plane lying on the back face of the gauge. In (b) a blue plane and coloured dots illustrate the virtual plane and corner points respectively. Stage 2 rectifies the image to a head on view of the gauge face (c) and stage 3 detects scale minimum, maximum, pointer centre and tip as yellow, green, blue and red dots respectively. From these keypoints, the gauge is transcribed by computing the pointer angle relative to the scale minimum and maximum, see equation 1.

the state of the art in multiple metrics on our data as well as on the publicly available Kaggle dataset [13].

Related work. The task of analogue gauge transcription has been tackled numerous times. Recent methods fall largely into two groups: those which use traditional based computer vision [3, 20, 25, 8, 18, 23, 9, 26, 17] and those which utilise deep learning [14, 10, 11, 21, 1, 16, 5, 15].

Methods using traditional based approaches are typically brittle to appearance variation in lighting, background clutter and highly constrained to particular types of gauges. Examples include methods using the Hough transform [12] for circular gauge detection and/or scale markings [10, 3, 17, 25, 26]; hand crafted heuristics based on curved shapes [20, 18]; or K-means and PCA for detecting and determining pointer angle [13]. Although these types of approaches are likely efficient to run on smartphones, they do not generalise well to in the wild conditions.

While many prior works are still using the traditional methods, there is a growing trend in applying deep learning to the task [10, 14, 15, 16]. However, these works often incorporate deep learning in just part of their transcription pipeline, probably due to the severe lack of suitable training data. Semi-synthetics have been used fairly successfully by Weidong *et. al* [5], but this was limited to a very small range of meters and single camera view. For transcribing digital meters, Charles *et. al* [6, 7] were successful in producing a mobile phone system. Using fully synthetic data for digit recognition, a single labelled image of a target meter and a CNN with modality converter, they obtained good in the wild performance. The drawback being a separate CNN needed to be trained per meter.

Notably [13] and [21] appear to be the only prior works which provide a dataset for analogue gauges with [13] being the most readily available, however annotations for pointer angle and readings are unavailable.

In section 2 an overview of our system is given, section 3 provides technical details, section 4 introduces our new dataset, section 5 describes our experiments, section 6 ablates and compares our method to the state of the art. Finally we conclude in section 7.

2. Method Overview

To digitise the reading displayed on a target gauge our system ingests an input image of the gauge along with the scale range. Operating in three stages: (1) the gauge is first detected on a low resolution image then (2) perspective distortion is corrected on a higher resolution crop of the gauge, and finally (3) the pointer position and angle relative to the scale is recovered. The process of using a low and high resolution image in this manner means detection in stage 1 is very fast (and low memory) and read precision in stage 3 is still accurate. These three stages are outlined in Figure 2 and described further below.

Stage 1: Gauge detection. A light weight detector is trained to carry out two roles: (i) locate the bounding box of the gauge and (ii) infer the homography between the back face of the gauge (where the scale resides) and the camera. Note this is not an easy task, the detector has to account for various types of bevel depth and glass reflection, see Figure 2(a) and (b).

Stage 2: Perspective correction. Perspective distortion is corrected by first predicting the four corners of the virtual gauge face plane using a keypoint detector, see Figure 2(b). Then secondly, recovering the homography between the gauge face plane and camera using these four points as image corner correspondences [2]. And finally, use the homography to warp the gauge face into an image as if taken by a head on camera, see Figure 2(c).

Stage 3: Gauge reading. A keypoint detector is trained to detect several keypoints on the rectified image of the gauge face. These are the scale minimum and maximum points, and the pointer center and tip, see Figure 2(c). The error in 2D positions due to parallax (caused by pointers not being

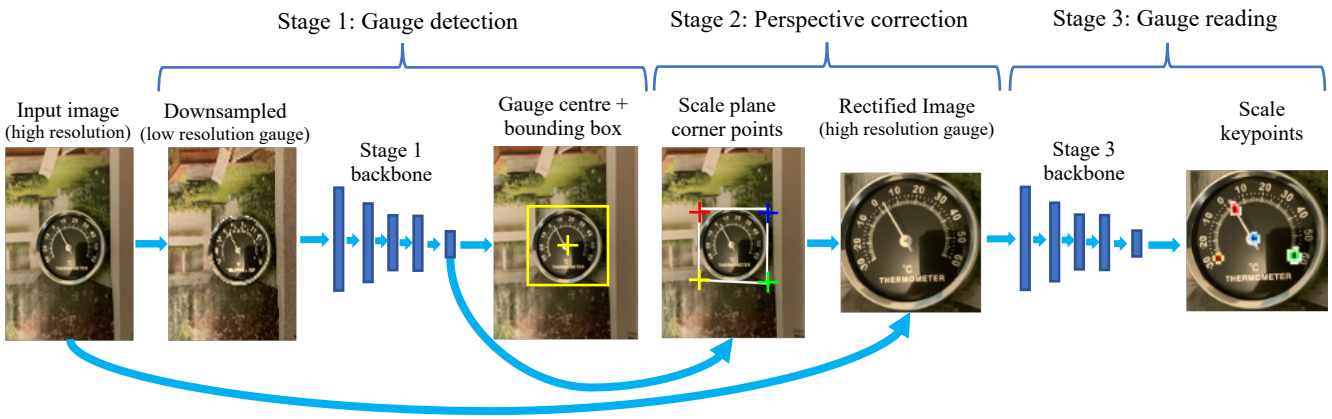


Figure 3. CNN architecture. Stage 1 and 3 models both use the same backbone architecture (with different weights). Separate fully convolutional heads provide outputs for the various tasks. Tasks are: gauge center detection and bounding box regression, virtual gauge face plane corner keypoint detection and finally keypoint detection on the rectified image. Heatmaps are used for keypoint regression, illustrated as a coloured blob overlay on the rectified image in stage 3, one colour for each keypoint.

in the same 3D plane as the scale) is accounted for and then the angle of the pointer with respect to the scale minimum is calculated, and with this a gauge reading is calculated.

3. Implementation details

The detector for stages 1 and 3 is based on a single stage multi task architecture on top of a deep network backbone encoder. Design inspiration is taken from the anchor free network CenterNet [27]. Figure 3 details the full pipeline and shows how two types of image (one for each stage) are processed through the network.

Multi resolution inputs. The same square RGB image resolution is used for stage 1 and 3 (N x N x 3). For stage 1 (detection) the gauge is at a low resolution and unknown location, but for stage 3 (reading) the gauge is at higher resolution filling the bounds of the image and centered. If we did not use this approach we found a single input of (1024 x 1024) is necessary to achieve high enough resolution of the gauge for reading. However, using the above approach we found a (192 x 192) input for stage 1 and 3 gave good performance and resulted in an over five fold speed increase compared with using the single high resolution image.

CNN detectors. Two separate CNN detectors are trained for stage 1 and 3. The detectors are multitask networks with various fully convolutional network heads for different tasks. For the stage 1 network three heads are used, one detects the center of the gauge, the second infers width and height of the bounding box and the last outputs 2D keypoint projections of the gauge face plane corners (trained as in [27] for human pose estimation). For the stage 2 detector, a single head is used to predict gauge face keypoints (pointer tip, pointer centre and min and max scale markings). This head regresses to a heatmap representation of keypoints (one channel per keypoint type) and at inference a 2D coordinate for each keypoint type is selected based on the location of maximum value within the heatmap.

Network heads. Each head operates on a ($\frac{N}{32}$ x $\frac{N}{32}$ x K) tensor from the backbone (K is channel dimension) and consists of a 1x1 convolution for downsampling the channel dimensions followed by 3 sets of transposed convolutions for learnt upsampling to a resolution of ($\frac{N}{4}$ x $\frac{N}{4}$ x C_h), the channel size for head h being denoted by C_h.

Training. Our CNN detectors are *only* trained on synthetic data. For stage 3, rectified gauge images are fed to the network during training. Rectification is computed using ground truth gauge face plane 2D corner points (see Figure 2(b) for an illustration) from high resolution images used to train the stage 1 detector. This type of ground truth is available to us as we use synthetic data for training. Both CNN detectors are trained for an input resolution of 192x192px.

Gauge reading. As pointers do not lie on the same 3D plane as the back face of the gauge, side on camera views can cause read errors due to parallax. This error is accounted for by predicting the co-ordinates of the pointer tip and center locations projected onto the back face of the gauge, rather than their true co-ordinates. This projection is done in our ground truth annotations to train the model.

To produce a final value for gauge reading an estimate for pointer angle is first calculated. This angle is taken in reference to the scale minimum point and can be easily calculated with the cosine rule and some conditional logic. Assuming the scale is linear, a simple linear interpolation is used to infer the reading:

$$\theta = \cos^{-1}\left(\frac{b^2 + c^2 - a^2}{2bc}\right), \qquad (1)$$

please refer to Figure 2(c) for the definitions of symbols.

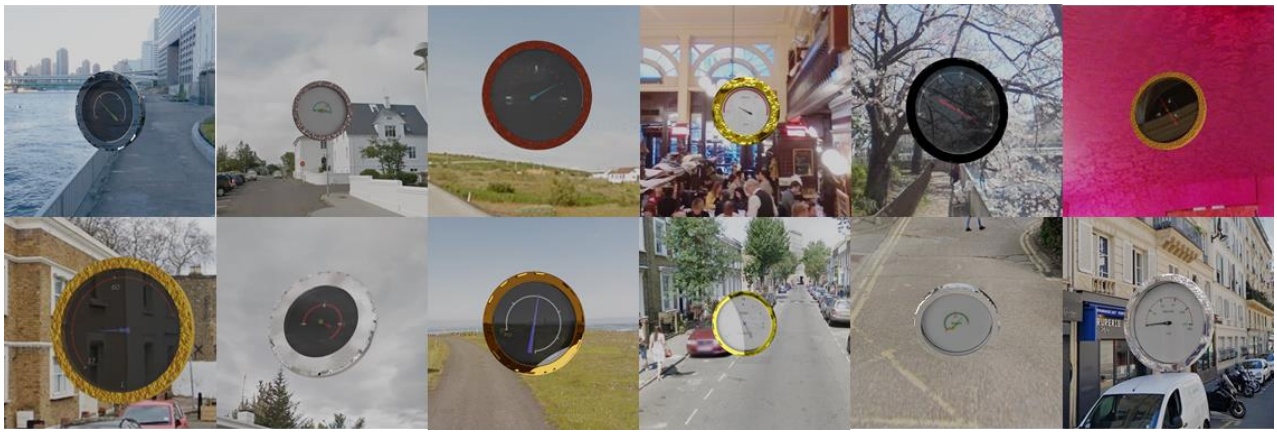


Figure 4. Synthetic-Gauges dataset. Examples of images from the Synthetic-Gauges dataset used for training our system.

Running on mobile phone. For our backbones we use the MobileNetV2 [22] architecture with an expansion factor or 1. This network is very efficient to run, with the heads of our model not adding much more computational burden. The models are trained using PyTorch [19], and for iOS mobile deployment, then converted to a 16bit CoreML model.

4. Datasets

For the task of analogue gauge reading, there is a severe shortfall in large publicly available datasets (also acknowledged by [1, 5, 11, 13]). We tackle this issue by (i) using advances in 3D rendering to produce a very large set of realistic synthetic gauge images with annotation (which we call *Synthetic-Gauges*) and (ii) collecting our own real world gauge dataset (called *Real-Gauges*) with comparatively larger meter variation than current public datasets.

4.1. Synthetic-Gauges

Our synthetic dataset consists of 10,000 training and 1,000 validation images of high quality renderings of gauges, at resolution 1024x1024px. This set of images contains large variations in colour, gauge shape, lighting conditions, scale style and background appearance. It was produced using the open source 3D creation software, Blender [4], utilising a high fidelity rendering engine to produce realistic environmental lighting and photo realistic images (important for good domain adaptation [24]).

A 3D scene was designed to contain a single 3D model of an analogue gauge with one pointer. The scene and gauge model are parameterised into 50 key variables describing the gauge shape and texture and environmental appearance. The gauge has various discrete components which can be randomised *e.g.* pointer shape and angle, gauge bevel, gauge material and scale texture. For the environment we can randomly alter camera angle, lighting conditions and background content. The gauge is glass fronted to capture the real world physics of reflection and specularity. For the scale texture, scale ranges, tick markings (based on a linear scale) , colours and positions are all variable. Each image is produced with a random parameter assignment from a constrained set to ensure a realistic looking image is generated. Backgrounds were randomly generated by sampling 360 degree background images from Google Street View. Using such a large set of parameters produced images with massive appearance variation, as shown in Figure 4 (and supplementary material).

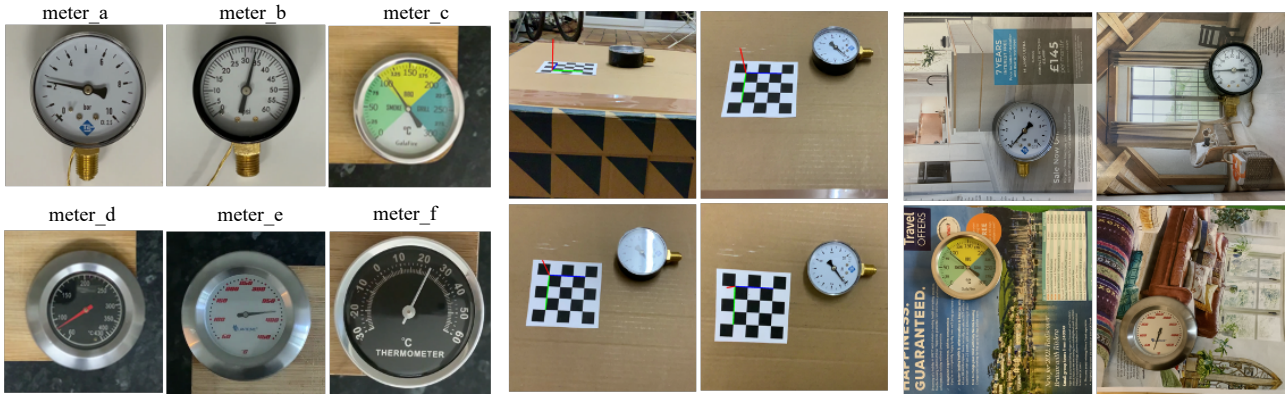
Annotation includes 2D gauge bounding boxes; 2D corner points of gauge face plane; 2D keypoint locations for pointer tip, center and scale min/max tick marks as well as the 3D camera angle with respect to the gauge.

4.2. Real-Gauges

Expanding upon publicly available real data, we collect and annotate a large quantity of real gauge images and videos. The dataset is split into three sets to evaluate three different tasks: (1) gauge detection, (2) perspective recovery and (3) gauge reading. Six meters with varying style and function were used during the capturing process, as shown in Figure 5(a). Details for each task follows.

Task 1: Gauge detection. The gauge detection dataset consists of each gauge photographed on 36 different backgrounds. Backgrounds are taken from an interior design magazine to give a high variation in colour and geometry whilst also simulating environments in which gauges will be typically situated. Centres of each gauge are labelled on each background, see Figure 5(c) for examples.

Task 2: Perspective recovery. The perspective recovery dataset consists of images of gauges from a variety of camera angles. For each image intrinsic and extrinsic camera



(a) Real-Gauges (b) Perspective recovery (c) Gauge detection

Figure 5. The 6 meters used for the Real-Gauges dataset are shown in (a), example data collected for the perspective recover task in (b) and examples of images used for gauge detection in (c).

Figure 6. Example meter crops from the 7 test videos of the Kaggle-Dataset.

parameters are estimated using a checkerboard camera calibration pattern. Normal vectors to the gauge face (situated on the calibration pattern) can then be computed, see Figure 5(b). The purpose of this dataset is to evaluate the read accuracy of a system with respect to perspective distortion.

Task 3: Gauge reading. The gauge reading set consists of 3 videos of each gauge which are 5 seconds long at 30fps. Each frame is labelled with the pointer angle and the gauge reading, a total of 2700 labelled frames. Each video is taken from a camera view face on to the gauge, so one can measure performance of gauge reading without interference from perspective effects.

4.3. Kaggle-Dataset

Jakob S. Lauridsen *et. al* [13] provides a dataset of 10 videos of varying length, comprised of 3 train videos and 7 test videos. Each video is from a static camera showing a single gauge in all but 1 video (where an off center background gauge is present), 6 gauges are used in total. We use the 7 test videos (illustrated in Figure 6), and use the same video naming convention as Jakob S. Lauridsen *et. al* [13]. Unfortunately, Jakob S. Lauridsen *et. al* [13] do not publicly release annotation for this dataset. Therefore for each video we partially labelled each video by annotating pointer angle and reading for approximately every 6^{th} frame. This annotation (although sparse) allows us to form a comparison to the reading method of Jakob S. Lauridsen *et. al* using an approximate metric.

5. Experimental setup

Here we thoroughly evaluate our method on three tasks: gauge detection, gauge reading and perspective recovery. Each task, the dataset and evaluation metrics used is described below.

Gauge detection. Evaluation is done on the Real-Gauges detection task, we measure the distance of predicted bounding box centre against ground truth (GT) in pixels. The mean norm of this distance is defined as μ_{error}, with standard deviation σ_{error}. Intersection over union (typically used for detection scores) is not used here as the exact bounds of the meter are not necessarily important for transcription and not all prior methods output bounding boxes making comparisons difficult. *Detection Accuracy (Det. Acc.)* is defined as the proportion of images in which a gauge is correctly detected, for the Real-Gauges dataset a gauge is present in every image.

Gauge reading. The Real-Gauges and the Kaggle dataset are both used to evaluate read accuracy. Several evaluation metrics are defined. The *read return rate* $R_{\%}$ is used to measure the proportion of frames where the system re-

turns a reading (regardless of value). *Absolute read error* is measured as the mean μ_V and the standard deviation σ_V of the difference between the ground truth gauge reading and the system prediction. *Absolute relative read error* is measured as the mean μ_R and the standard deviation σ_R, and represents absolute read error as a proportion of the gauges reading range, to demonstrate how significant the error is in terms of each gauge. *Absolute pointer angular error* is measured as the mean, μ_θ, and standard deviation, σ_θ, which measure the difference between ground truth angle of the pointer and the system prediction. These metrics are only computed over frames where a reading is returned.

Perspective recovery. The 1,000 validation images from Synthetic-Gauges dataset are used to evaluate perspective recovery by measuring the distance between the ground truth virtual gauge face plane corners and predicted corner keypoints. The mean μ_{error} and standard deviation σ_{error} of error are recorded in units of pixels. These are only computed over frames where a gauge is detected. As an indirect measure of perspective recovery we also use the Real-Gauges perspective recovery dataset and measure read accuracy across different camera angles.

Training details. Our detector was trained solely using the Synthetic-Gauges dataset using a MobileNetV2 backbone pretrained on ImageNet. Image based augmentations during training, such as cropping, rotation and flipping were applied. We used a batch size of 64, learning rate $1.25\text{x}10^{-3}$, with steps down to $1.25\text{x}10^{-4}$ and $1.25\text{x}10^{-5}$ at epochs 750 and 1500 respectively, and ADAM optimiser. We trained for 2000 epochs, stopping training when no further drop in train loss was observed.

Baseline method. We directly compare our method against the recent state of the art approach of Jakob S. Lauridsen *et. al* [13].

6. Results and discussion

Gauge Detection. Our performance is detailed in Table 2. There is a low μ_{error} across all gauges. meter_a was the most difficult to detect, its colour and geometry closely matched objects in the background scene, resulting in false detections. This highlighted that even further variation in the synthetic set would help detection.

Gauge Reading. Table 4 compares our performance to the baseline. Only a direct comparison to pointer angle prediction is made as Jakob S. Lauridsen *et. al* [13] was not capable of outputting gauge readings, it also measured angles in reference to the horizontal, and so the angle from the horizontal to the scale minimum was provided to compare to the ground truth. Jakob S. Lauridsen *et. al* [13] measures there performance with non absolute values, this can artificially cause the mean error to become 0, and so we recalculate performance metrics for this system. Our system vastly outperforms the baseline in prediction error and reading return. The poor performance of Jakob S. Lauridsen *et. al* [13] is explained by Figure 7. For a high number of frames it predicts an angle 0 degrees to the horizontal, meanwhile our system closely tracks the ground truth at all times. Our system is much more capable of generalising to unseen gauges.

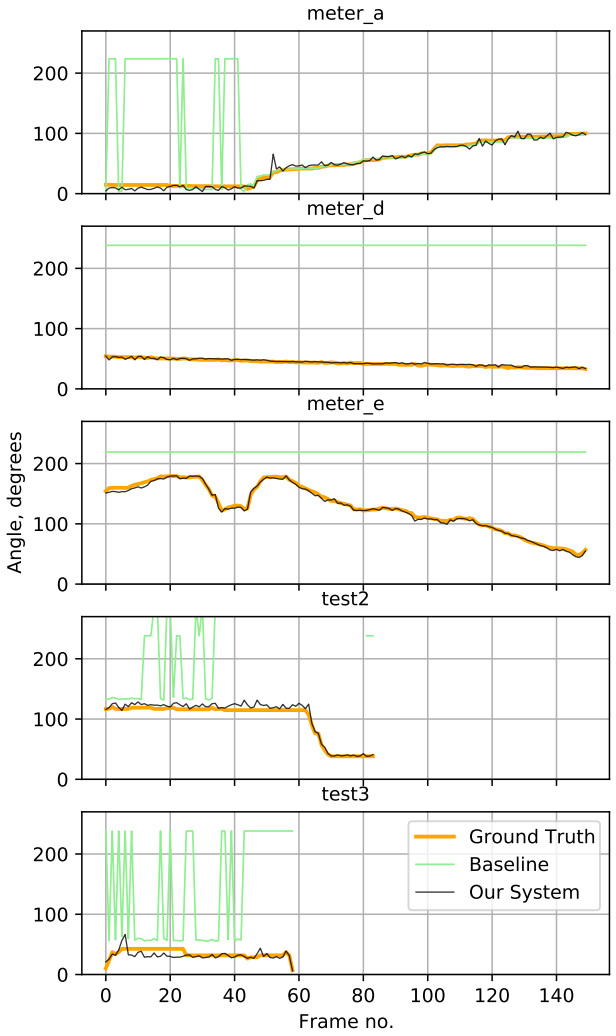


Figure 7. Real Gauge Prediction Performance. Ground truth (orange), Our system (black), Jakob S. Lauridsen *et. al* [13] (green). In every gauge type and motion our system demonstrates superior tracking.

Table 1 shows performance of our system at predicting the actual value of the gauge. In most cases the relative error is very small, showing that our system can make accurate predictions for a wide variety of gauges.

Gauge	μ_R	σ_R	μ_V	σ_V	$R_{\%}$	No. Frames	Reading Range
meter_a	0.053	0.079	0.53	0.79	100	450	0 – 10
meter_b	0.070	0.115	4.22	6.89	100	450	0 – 60
meter_c	0.009	0.008	2.74	2.53	100	450	0 – 300
meter_d	0.043	0.056	15.84	20.59	43	450	60 – 430
meter_e	0.012	0.012	4.61	4.81	100	450	50 – 450
meter_f	0.023	0.019	2.04	1.70	100	450	-30 – 60
test1	0.003	0.002	0.02	0.01	100	268	0 – 6
test2	0.01	0.01	0.04	0.04	100	84	-1 – 3
test3	0.036	0.128	0.09	0.32	100	59	0 – 2.5
test4	0.008	0.012	0.02	0.03	100	42	-1 – 1.5
test5	-	-	-	-	0	25	-1 – 1.5
test6	0.045	0.015	0.18	0.06	100	42	-1 – 3
test7	0.042	0.01	0.25	0.06	100	30	0 – 6
Mean	0.030	0.039	2.55	3.15	88	-	-

Table 1. Gauge Reading performance. All results for testing on the 6 gauges from our dataset and the 7 test videos from Jakob S. Lauridsen *et. al* [13]. Prediction performance is very strong in most cases and detection accuracy is very high.

Gauge	μ_{error}	σ_{error}	Detection Accuracy, %
meter_a	71.6	224.4	97.2
meter_b	14.5	10.3	100
meter_c	27.3	17.3	97.2
meter_d	47.7	140.2	100
meter_e	32.3	18.7	97.2
meter_f	19.1	9.5	100
Mean	35.4	70.1	98.6

Table 2. Gauge detection results. Performance predicting centre of each gauge on various backgrounds. Units are in pixels, the input image size is 4032x3024 pixels, the average diameter of gauges in images is 1026 pixels.

Key Point	μ_{error}	σ_{error}	Det. Acc., %
Top Left	2.34	1.86	100
Bottom Left	2.43	2.42	100
Bottom Right	2.32	1.87	100
Top Right	2.44	2.41	100

Table 3. Perspective Recovery Performance. Measurable performance for rectification is difficult with real images, so metrics are computed with synthetic data. The mean error norm for each key point for 1000 synthetic images is shown. Each input image has size 1080x1080 pixels.

We found that the primary cause of errors in our predictions to be noise in predicting the scale minimum point. There is a large amount of visual noise in the real gauges in the form of company branding, screws, etc. Improving the fidelity of our synthetic dataset to simulate these effects would likely help. We also noticed strong shadows could sometimes be problematic, especially for the pointer, where pointer shadow could be mistaken for the pointer itself. Given the locality of the shadow to its source, this does not result in too large of an error, but is still an area that is worth addressing in the future.

Perspective Recovery. Table 3 shows our system's performance for perspective recovery. Errors for each virtual plane keypoint corner are shown. Keypoint error is approximately 2 pixels from ground truth, which compared to the input image size of 1080x1080 pixels, is almost negligible. A heatmap illustrating how pointer angle prediction error varies with camera angle on synthetic data is shown in Figure 8. Perspective recovery is shown to be harder as the camera approaches side on views of the gauge. However, notice that even at the most severe angles we are able to recover perspective relatively well. Perspective recovery on the Real-Gauges dataset is analysed qualitatively in Figure 10. Note we obtain good rectification under a range of

	Our System		Jakob S. L. et al. [13]	
Gauge	μ_θ	$R_{\%}$	μ_θ	$R_{\%}$
meter_a	9.24	100	36.51	100
meter_b	26.06	100	103.63	100
meter_c	1.94	100	127.68	32.5
meter_d	11.70	43	207.07	81.1
meter_e	3.70	100	116.14	99.6
meter_f	4.31	100	-	0
test1	0.83	100	0.99	100
test2	2.95	100	63.89	46
test3	9.90	100	95.56	100
test4	3.39	100	34.36	100
test5	-	0	96.63	80
test6	1.43	100	15.96	93
test7	11.38	100	14.22	100
Mean	7.24	87.9	76.05	79.4

Table 4. Pointer angle prediction performance. Performance is shown for our system and also the baseline. Only absolute angle error is compared as the baseline is not designed to output an actual reading. The baseline assumes a perfectly horizontal line passing through the scale minimum and maximum, therefore we have to manually adjust the baseline angles to account for this. Our system makes no such assumptions and demonstrates superior performance in all metrics.

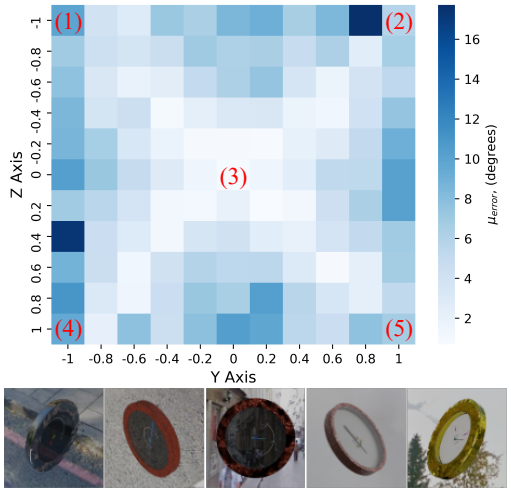


Figure 8. Prediction performance with perspective recovery, synthetic. Perspective recovery was tested on synthetic data by analysing images at a variety of camera positions and recording prediction error. These images were sorted into bins of similar camera position and the mean pointer angle prediction error, μ_{error} is shown. Example images at various positions are shown.

camera angles. We found that the system performed worse when other objects were interacting with the gauge and occluding it. We think this can be be addressed by introducing occluding objects in our synthetic training data.

Read error sensitivity. The read sensitivity of our system due to camera angle is plotted in Figure 9. Using the Real-Gauges perspective recovery dataset, average absolute pointer angle error and reading return rate at a range of viewing angles is computed. Averages are taken over sets of images split into bins based on camera angles relative to the gauge plane normal: [-75,-50,-25,0,25,50,75] degrees. We observe our system performs best at angles of $\pm 20^o$, suitable for application where a human would find it difficult to obtain perfect face on camera alignment with the gauge.

Computation Performance. On an iPhone 11 using the Neural Engine, our system runs at 25fps with ≈300MB memory usage. Using only GPU on iPhone 7 is very good at ≈20fps. Comparatively the method of Jakob S. Lauridsen *et. al* [13] runs at a much slower 0.1 fps on a Intel i7 CPU, given its slow performance on a PC it can be assumed unsuitable for a real-time mobile application.

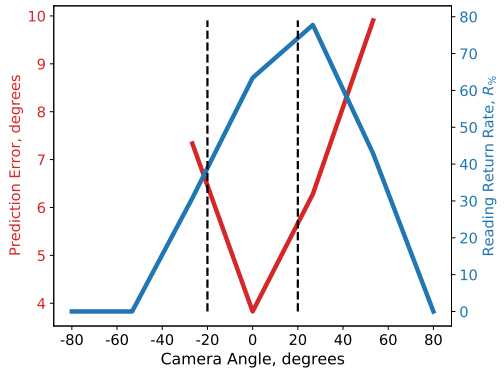


Figure 9. Prediction performance with perspective recovery, real. Perspective recovery was tested on real data by analysing images at a variety of camera positions and recording prediction error. These images were sorted into bins of similar camera position and the mean pointer angle prediction error (with angle taken from horizontal), μ_{error} and detection accuracy are shown. Suitable usage limits for camera angle are shown to be around $\pm 20^o$

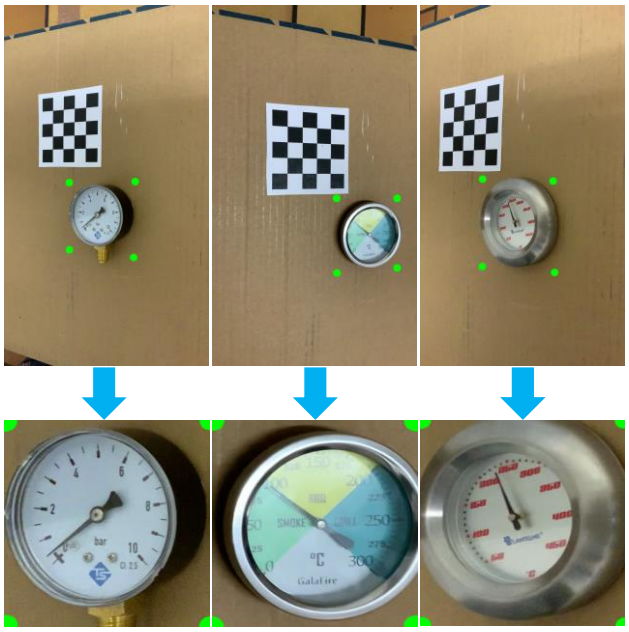


Figure 10. Perspective Recovery on real images. Qualitative demonstration of rectification on real images. Our system can predict virtual plane points on real images as well as synthetic and use these points to produce rectified images.

7. Conclusions

Presented is a method for transcribing analogue gauges using a mobile phone. The system can reliably detect gauges in images, account for perspective distortion and accurately recover pointer position. Unlike previous attempts [5, 14, 15, 21], our system is far more general and is capable of transcribing any circular gauge with one pointer. Our method can be used in industrial environments (on mobile phone or other embedded systems) to aid other systems such as robotic machine control, enable legacy industrial equipment to be IOT compatible or for public use as an app to easily record values from analogue meters. Our method is fast, robust and can run in real-time on mobile phone.

References

[1] Alexey Alexeev, Georgy Kukharev, Yuri Matveev, and Anton Matveev. A highly efficient neural network solution for automated detection of pointer meters with different analog scales operating in different conditions. *Mathematics*, 2020. 2, 4

[2] Alex M Andrew. Multiple view geometry in computer vision. *Kybernetes*, 2001. 2

[3] Haojing Bao, Qingchang Tan, Siyuan Liu, and Jianwei Miao. Computer vision measurement of pointer meter readings based on inverse perspective mapping. *Applied Sciences*, 2019. 2

[4] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Blender Institute, Amsterdam. 4

[5] Weidong Cai, Bo Ma, Liu Zhang, and Yongming Han. A pointer meter recognition method based on virtual sample generation technology. *Measurement*, 2020. 2, 4, 8

[6] J. Charles, S. Bucciarelli, and R. Cipolla. Real-time screen reading: reducing domain shift for one-shot learning. In *Proc. of British Machine Vision Conference*, 2020. 2

[7] J. Charles, S. Bucciarelli, and R. Cipolla. Scaling digital screen reading with one-shot learning and re-identification. In *Proc. of Workshop on Applications of Computer Vision*, 2021. 2

[8] Yung-Sheng Chen and Jeng-Yau Wang. Computer vision-based approach for reading analog multimeter. *Applied Sciences*, 2018. 2

[9] Jiannan Chi, Lei Liu, Jiwei Liu, Zhaoxuan Jiang, and Guosheng Zhang. Machine vision based automatic detection method of indicating values of a pointer gauge. *Mathematical Problems in Engineering*, 2015. 2

[10] S. Dumberger, R. Edlinger, and R. Froschauer. Autonomous real-time gauge reading in an industrial environment. In *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2020. 2

[11] Jian Huang, Junzhe Wang, Yihua Tan, Dongrui Wu, and Yu Cao. An automatic analog instrument reading system using computer vision and inspection robot. *IEEE Transactions on Instrumentation and Measurement*, 2020. 2, 4

[12] J Kittler and J Illingworth. The adaptive hough transform. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1987. 2

[13] Jakob S. Lauridsen, Julius A. G. Graasmé, Malte Pedersen, David Getreuer Jensen, Søren Holm Andersen, and Thomas B. Moeslund. Reading circular analogue gauges using digital image processing. In *Proceedings of the 14th International Conference on Computer Vision Theory and Applications*, 2019. 2, 4, 5, 6, 7, 8

[14] Zhu Li, Yisha Zhou, Qinghua Sheng, Kunjian Chen, and Jian Huang. A high-robust automatic reading algorithm of pointer meters based on text detection. *Sensors*. 2, 8

[15] Yue Lin, Qinghua Zhong, and Hailing Sun. A pointer type instrument intelligent reading system design based on convolutional neural networks. *Front. Phys. 8: 618917*, 2020. 2, 8

[16] Yang Liu, Jun Liu, and Yichen Ke. A detection and recognition system of pointer meters in substations based on computer vision. *Measurement*, 2020. 2

[17] Yifan Ma and Qi Jiang. A robust and high-precision automatic reading algorithm of pointer meters based on machine vision. *Measurement Science and Technology*, 2018. 2

[18] Xiaoming Mai, Wensheng Li, Yan Huang, and Yingyi Yang. An automatic meter reading method based on one-dimensional measuring curve mapping. In *IRCE*, 2018. 2

[19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*. 4

[20] R. Sablatnig and W. G. Kropatsch. Automatic reading of analog display instruments. In *Proceedings of 12th International Conference on Pattern Recognition*, 1994. 2

[21] Gabriel Salomon, Rayson Laroca, and David Menotti. Deep learning for image-based automatic dial meter reading: Dataset and baselines. In *IJCNN*, 2020. 2, 8

[22] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks, 2019. 4

[23] T Selvathai, Swarna Ramesh, KK Radhakrishnan, et al. Automatic interpretation of analog dials in driver's instrumentation panel. In *AEEICB*. 2

[24] Apostolia Tsirikoglou, Joel Kronander, Magnus Wrenninge, and Jonas Unger. Procedural modeling and physically based rendering for synthetic data generation in automotive applications, 2017. 4

[25] Junzhe Wang, Jian Huang, and Rong Cheng. Automatic reading system for analog instruments based on computer vision and inspection robot for power plant. In *ICMIC*, 2018. 2

[26] Chao Zheng, Shaorong Wang, Yihan Zhang, Pengxiang Zhang, and Yong Zhao. A robust and automatic recognition system of analog instruments in power system by using computer vision. *Measurement*, 2016. 2

[27] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points, 2019. 3