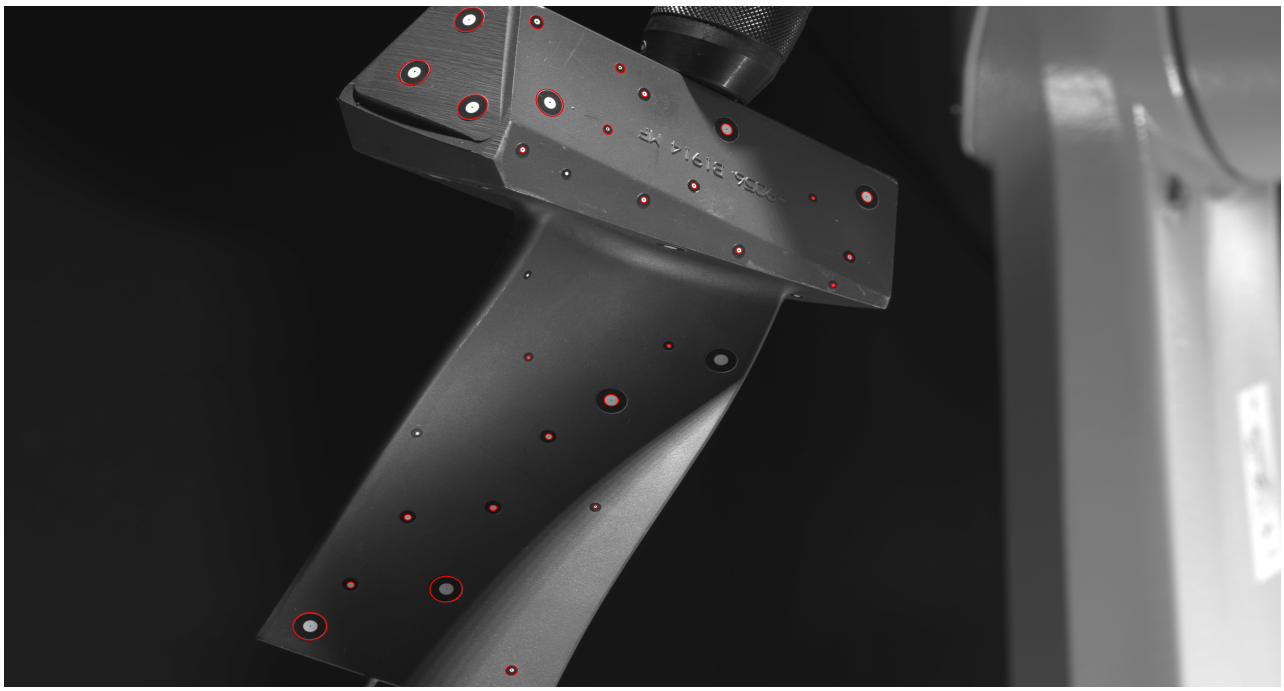
Increasing the Robustness and Accuracy of CCC-Marker Detection under Challenging Conditions

Scientific Report on the Computer Science Project
as part of the Master's Program in Computational Engineering

Friedrich-Alexander-Universität Erlangen-Nürnberg
Institute for Factory Automation and Production Systems
Prof. Dr.-Ing. Jörg Franke



Author: Vipul Patil, 23218152

Supervisor(s): Prof. Dr.-Ing. Jörg Franke
M.Sc. Oguz Kedilioglu

Submission date: 01.07.2024

Processing time: 5 months

Declaration

I hereby declare that I have written this thesis without outside help and without using any sources other than those indicated. This thesis has not been submitted in the same or a similar version to any other examination office and been accepted as part of an examination performance. All statements that have been adopted literally or paraphrased are marked as such.

Erlangen, 01.07.2024

Vipul Patil

Abstract

This scientific report introduces a novel methodology for robustly and accurately detecting Concentric Contrasting Circles (CCC) markers in challenging environments. These markers are crucial reference points in various industrial tasks like object localization, alignment, and navigation. Additionally, the methodology presented in this report extends its potential to high-accuracy pose estimation, providing a reliable solution for precise object positioning and orientation determination in dynamic settings. Previously proposed approaches often struggled with robustness, facing challenges such as varying brightness, background noise, and imperfections in the markers themselves, limiting their effectiveness in real-world industrial scenarios. Notably, these prior methods were solely based on computer vision techniques, rendering them highly dependent on external factors. To tackle these challenges, the proposed methodology integrates advanced image processing techniques with CNN classification. By combining computer vision with deep learning, this approach proves effective in robustly detecting CCC markers under diverse conditions, resulting in significant improvements in feature detection accuracy. Furthermore, the method takes into account low-level factors that affect the localization of the center of detected markers, thereby enhancing the precision of the task. Ultimately, the project aims to improve the efficiency and dependability of CCC markers specifically in high-precision pose estimation, while also expanding their utility across various aspects of industrial automation.

Contents

List of Figures

List of Tables

1 Introduction

The accurate detection and localization of ellipses, particularly Concentric Contrasting Cir-cles (CCC) markers on workpieces, are critical tasks in various industrial applications. However, due to the limited availability of real-world training data, synthetic data genera-tion becomes essential for training robust models. In this paper, a comprehensive methodol-ogy for ellipse detection and localization is proposed, addressing the challenges posed by limited training data and diverse environmental conditions.

The methodology encompasses several key stages, beginning with synthetic data generation. Synthetic images containing ellipses with varying sizes, orientations, and positions are gen-erated, simulating realistic scene compositions. Techniques such as random transformations and noise injection are employed to augment the synthetic data, enhancing variability and robustness, crucial for training effective detection models.

Following synthetic data generation, the input images undergo preprocessing and feature ex-traction. This includes reading the images in grayscale format, resizing them to a stand-ard size, and detecting edges using the Canny edge detection algorithm. Hierarchical con-tour analysis is then performed to identify parent-child relationships between contours, fa-cilitating accurate ellipse detection.

For ellipse detection, the MobileNet V2 architecture is employed, chosen for its lightweight and efficient design, making it suitable for deployment on resource-constrained devices. The model is fine-tuned on the synthetic data using transfer learning techniques, optimizing its performance for ellipse detection tasks. Evaluation metrics such as mean average preci-sion (mAP) and accuracy are employed to assess the model's performance.

Finally, the methodology includes a procedure for fitting ellipses to detected contours and calculating their centroids. Inspired by the Region-based Convolutional Neural Network (R-CNN) framework, this procedure accurately detects, filters, fits ellipses, and calculates the centers of CCC markers in the image.

Through this comprehensive methodology, a robust and precise approach to CCC marker detection and analysis is developed, addressing the challenges posed by real-world scenari-os with limited training data. This methodology holds promise for various industrial appli-cations, facilitating automation and quality control processes.

2 Related Work

In the field of computer vision, detecting and localizing fiducial markers and objects in complex scenes is a well-established yet challenging task. Traditional methods for fiducial marker detection often rely on geometric properties and edge detection techniques. For example, Gonzalez and Woods introduced foundational concepts in digital image processing that underpin many modern computer vision methods, including edge detection and object recognition [1]. Their work laid the groundwork for more advanced techniques such as the Canny edge detector, which remains a standard tool for identifying object edges in images [2].

Recent advancements in fiducial marker systems have focused on increasing robustness to challenging conditions such as occlusion, varying distances, and extreme angles. For instance, Calvet et al. developed a fiducial marker system based on concentric rings, which utilizes projective properties to localize markers even under severe conditions like motion blur accurately [3]. Their method leverages the geometric structure of concentric circles to detect markers and determine the center of the fiducial robustly. This approach demonstrates the importance of geometric features in improving marker detection accuracy under difficult circumstances.

In the realm of object tracking and pose estimation, several studies have addressed high-speed and high-frequency tracking challenges. Liang et al. presented a high-speed vision system for tracking rotating objects, achieving high-frequency pose estimation even under occlusion and high-speed conditions [4]. Their system illustrates the potential of advanced vision systems for real-time applications but also highlights the need for further optimization to achieve even higher performance and robustness.

The integration of deep learning techniques with traditional computer vision methods has shown considerable promise in overcoming some of the limitations of conventional approaches. The work by Sandler et al. on MobileNetV2 introduced efficient convolutional neural network architectures that are suitable for real-time object detection and classification tasks [5]. Building on these ideas, Dong et al. introduced Ellipse R-CNN, a novel approach for detecting elliptical objects in cluttered scenes by extending the Mask R-CNN framework with ellipse regression techniques [6]. This method excels in scenarios with occlusion and clustering, demonstrating that deep learning models can significantly improve object detection and localization tasks.

Combining R-CNN-based methods with conventional computer vision techniques, such as edge detection and geometric fitting, offers a promising approach for enhancing marker detection and localization capabilities. The strengths of R-CNN models, such as those demonstrated by Mask R-CNN for instance segmentation and the Ellipse R-CNN for ellipse detection [6], can be effectively integrated with traditional techniques for preprocessing and feature extraction. For example, using the Canny edge detector to preprocess images for R-CNN-based models can enhance feature extraction and improve the overall accuracy of object detection systems [2]. This hybrid approach leverages the strengths of both deep learning and classical methods to address complex challenges in marker detection and object localization.

In summary, while traditional computer vision techniques provide a strong foundation for image processing tasks, integrating R-CNN-based methods with conventional preprocessing techniques presents a promising avenue for advancing fiducial marker detection and object localization. The combination of robust geometric feature extraction with advanced deep learning models can lead to more accurate and reliable solutions for complex vision tasks.

3 Methodology

This chapter explains the methodology used to train a model for detecting CCC-markers. The methodology consists of several key stages, including data generation, image preprocessing, feature extraction, model training, and ellipse detection. *Figure 1* provides a comprehensive overview of the entire methodology.

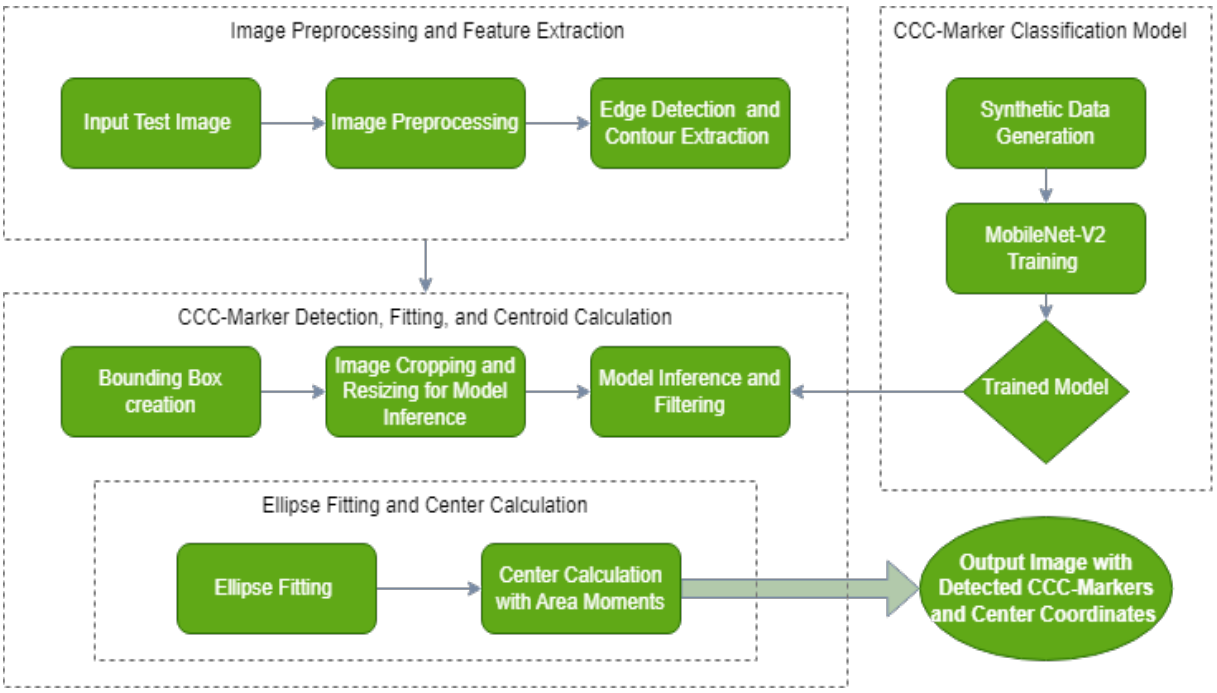


Figure 1: Overview of Methodology.

3.1 Synthetic Data Generation

The objective of generating synthetic data is to train a model for detecting Concentric Contrasting Circles (CCC) markers on workpieces, especially due to the unavailability of sufficient real-world training images. Synthetic data is preferred as it allows for the creation of a diverse dataset tailored to the specific needs of the model. By using only edge contour images, the training task is simplified, focusing the model's attention on key features. Varied ellipse images with concentric child ellipses are generated to simulate the desired markers on workpieces. Introducing variability in the synthetic data is crucial for ensuring the model's robustness in detecting markers of different sizes, orientations, and noise levels, thereby preventing overfitting and improving the model's generalization ability. Labels for each image are calculated to indicate the presence of the main ellipse and provide information about its parameters.

3.1.1 Data Generation Procedure for Positive Class (Ellipse)

- **Ellipse Parameters Generation**
  - Random parameters for the main ellipse are generated

- Center: Random coordinates within a specified range (50 to 64).
- Axes: Random semi-major and semi-minor axes within a specified range (20 to 64).
- Ensure the major axis is always greater than the minor axis by swapping if necessary.
- Adjust parameters based on image dimensions to ensure the ellipse fits within the image.
- Clip the center coordinates to avoid exceeding image boundaries.
- Limit axes lengths to ensure the ellipse remains within the image bounds.

- **Ellipse Drawing**
  - Using OpenCV's `cv2.ellipse` function, the script draws the main ellipse on a black background image.
  - A random thickness 't' is chosen for drawing the ellipse edges.
- **Child Ellipse Calculation and Drawing**
  - Parameters for the concentric child ellipse are calculated based on a random ratio between child and parent ellipse axes.
  - The child ellipse has the same center and orientation as the parent ellipse but with adjusted axes.
  - Using `cv2.ellipse`, the child ellipse is drawn on the same image.
- **Optional Salt Noise Addition**
  - With a probability of 30%, salt noise is added to the image to introduce variability.
  - Salt noise is added by randomly setting pixels to white (255) within the image.
- **Labeling and Saving**
  - Label information for the presence of the ellipse and the outer bounding box is calculated: The presence label indicates the presence of the ellipse and includes parameters like center coordinates, angle, and axes lengths. The outer bounding box label includes the coordinates of the bounding box surrounding the outer ellipse.
  - Images along with their corresponding labels are saved in appropriate directories.
- **Data Integrity Check**

  Images are saved only if all coordinates of the outer bounding box are non-negative, ensuring that the bounding box lies within the image boundaries.
- **Overview of Synthetic Data Generation Procedure**

  The synthetic data generation procedure for training the model includes various cases to enhance the robustness and simulate real-world scenarios. Each case introduces specific types of noise or imperfections into the generated images.

  Occlusion Noise: Occlusion noise mimics real-world scenarios where objects partially obscure or occlude parts of the image. The following types of occlusions are included:
  - Black Rectangles and Circles: Simulate objects blocking parts of the image.
  - Random Lines: Represent interference or obstructions in the image.
  - Optional Salt Noise: Additional variation for realism.

  Random Noise: Random noise simulates imperfections or distortions during image acquisition or processing. This includes:
  - Various Shapes: Rectangles, circles, and triangles contribute to randomness and complexity.
  - Gaussian Noise: Represents natural variation in pixel intensity.
  - Optional Salt Noise: Adds further diversification to noise characteristics.

  Imperfect Ellipse: This case focuses on introducing imperfections into elliptical shapes, similar to edge detection noise encountered in real-world images. The process involves:
  - Main Ellipse with Imperfections: Deviates from a perfect elliptical shape.

- Gaussian Noise: Introduces subtle variations in pixel intensity.
- Optional Salt Noise: Adds randomness and realism.

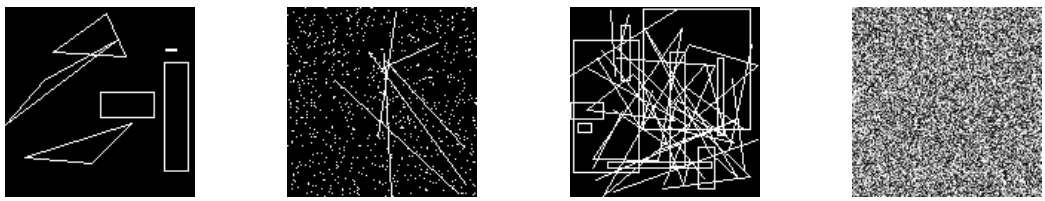
a) Without Noise *b) Salt Noise* *c) Random Noise* *d) All Noises*

Figure 2: Samples of synthetic training images generated for the positive class.

3.1.2 Data Generation Procedure for Negative Class (Non-Ellipse)

The functionality of this process is to generate synthetic images that represent scenarios where ellipses are absent. The approach involves utilizing various methods to create binary images with different patterns and shapes.

- **Random Noise Images:** Creates images filled with random noise, simulating background clutter. Optionally adds salt noise for additional variation.
- **Geometric Shapes:** Draws single closed-loop shapes such as rectangles and triangles. Shapes are generated with random positions and sizes.
- **Irregular Closed-Loop Figures:** Produces irregular closed-loop shapes with random curvature. Shapes vary in complexity and size, mimicking natural irregular patterns.
- **Combination of Elements:** Randomly combines geometric shapes, lines, and curves to form complex patterns. Introduces randomness and diversity into the generated images.
- **Optional Salt Noise:** Provides an option to add salt noise to the images for increased realism.
- **Labeling and Saving:** Assigns absence labels (0) to indicate the absence of ellipses in the images. Saves the generated images along with corresponding absence label files.



a) Closed Shapes *b) Random Lines* *c) Random Shapes* *d) Salt Paper Noise*

Figure 3: Samples of synthetic training images generated for the negative class.(Inclusion of closed shapes other than ellipse makes the model learn elliptical shape specifically)

3.2 Image Preprocessing and Feature Extraction

This step involves preparing the input image and extracting key features to facilitate the detection of concentric contrasting circles (CCC) markers. It includes reading and resizing the

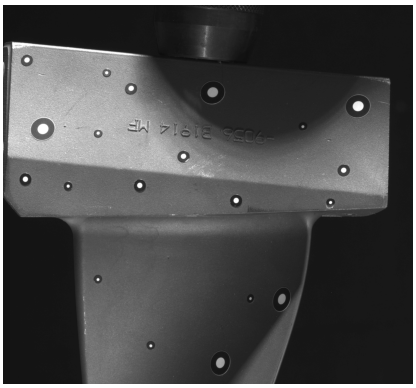
image, detecting edges, and extracting contours, which together provide the necessary input for the model to identify markers accurately.

3.2.1 Image Preprocessing

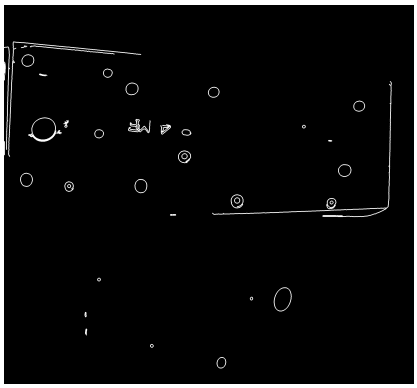
- **Image Reading:** Read the input image in gray-scale format. Gray-scale conversion simplifies the image by reducing it to intensity values, which is essential for edge detection and contour extraction.
- **Image Resizing:** Resize the image to a standard size for consistency in processing using the `cv2.resize` function, which employs pixel interpolation methods for resizing the image. This step ensures that the subsequent operations are applied uniformly across different image sizes. Standardizing image size is crucial for ensuring that the features extracted are comparable across all images."

3.2.2 Edge Detection

- **Canny Edge Detection:** Apply the Canny edge detection algorithm (`cv2.Canny`) to detect edges in the pre-processed image. This step identifies regions of high gradient intensity, often corresponding to object boundaries or contours. The Canny edge detector is known for its ability to detect a wide range of edges in images [2].



a) Input Image



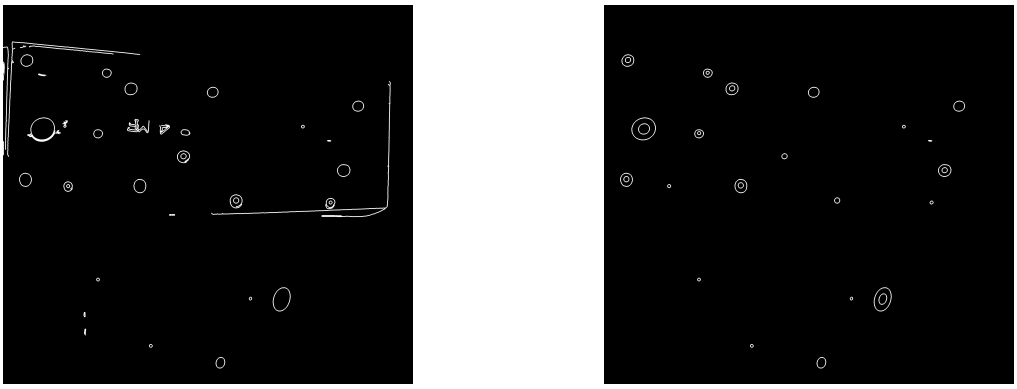
b) Edge Image

Figure 4: Input image conversion to edge image.

3.2.3 Contour Extraction

- **Contour Detection:** Extract contours from the edge-detected image using the `cv2.findContours` function in OpenCV. Edge detection identifies points in an image where the brightness changes sharply, representing object boundaries. Contour detection, on the other hand, involves retrieving the continuous curves that trace the boundaries of these detected edges. This process helps in identifying the shape and structure of objects or regions within the image.

- **Hierarchy Analysis:** Analyse contour hierarchies to identify parent-child relationships between contours. This analysis helps in distinguishing between individual objects and their components within the image. Without hierarchy analysis, we would only obtain the outer ellipse (parent) contour. To detect child contours within the parent contour, we fuse the contour detection step with hierarchical analysis. By leveraging the hierarchical data, we can identify and extract the child contour nested within the parent contour, crucial for accurately detecting CCC markers. Hierarchical analysis is important for understanding complex structures within an image [5].



a) Edge Image *b) Contour Extraction*

Figure 5: Contour extraction from edge image.

3.3 Image Classification Model - MobileNet-v2

In this section, the process of selecting, training, and evaluating a MobileNet V2 model for detecting the presence of ellipses in synthetic images is detailed. The MobileNet-v2 architecture is chosen for its lightweight and efficient design, making it suitable for deployment on resource-constrained devices or platforms.

3.3.1 Model Selection

MobileNet -v2 is an efficient deep learning architecture designed for resource-constrained devices [5]. It utilizes depth-wise separable convolutions, reducing the model size and computational requirements without compromising performance [7]. This efficiency makes it ideal for our task of ellipse detection, where the model needs to run on potentially limited hardware.

3.3.2 Transfer Learning

- **Initial Fine-Tuning Strategy:** A pre-trained MobileNet-v2 model, originally trained on the ImageNet dataset, was fine-tuned using a synthetic dataset of ellipse images. This transfer learning approach was chosen to speed up the training process and improve performance by leveraging the model's broad visual knowledge [8].

- **Overfitting Issue:** Initially, all layers in the pre-trained MobileNet-v2 were frozen. For fine-tuning, the top layers of the base model were gradually unfrozen to enable more specialized feature learning. However, this approach led to overfitting, with performance deteriorating on the validation set due to redundant feature learning.
- **Final Configuration:** To mitigate overfitting, the initial configuration was reverted to, where all layers of the original MobileNet-v2 model were kept frozen. Only the newly added top layers were fine-tuned, which improved the model's generalization and performance on the validation set.

3.3.3 Training Procedure

The training procedure involves several key steps:

- **Data Preprocessing:** Images are loaded in RGB format and resized to 128x128 pixels. Normalization is applied to scale pixel values between 0 and 1. Labels indicating the presence or absence of ellipses are also loaded.
- **Model Configuration:** The MobileNet-v2 model, pre-trained on ImageNet, is modified by adding a *'global average pooling'* layer and *'dense'* layers for binary classification. The final dense layer uses a *'sigmoid activation function'* to predict the presence of an ellipse.
- **Loss Function and Optimizer:** The model is compiled using *'binary cross-entropy'* loss, which is suitable for binary classification tasks, and the *'Adam'* optimizer, known for its efficiency and adaptive learning rate capabilities [9].
- **Hyperparameter Tuning:** Hyperparameters such as textitlearning rate and *batch size* are tuned to optimize model performance. A learning rate of 0.001 and a batch size of 32 are selected based on preliminary experiments.

3.3.4 Model Architecture and Parameters

Below is a summary of the model architecture and the parameters involved:

Table 1: Bottleneck residual block transforming from k to k' channels, with stride s, and expansion factor t [5].

Input	Operator	Output
$h \times w \times k$	1×1 conv2d, ReLU6	$h \times w \times (tk)$
$h \times w \times tk$	3×3 dwise, $s = s$, ReLU6	$\frac{h}{s} \times \frac{w}{s} \times (tk)$
$\frac{h}{s} \times \frac{w}{s} \times (tk)$	linear 1×1 conv2d	$\frac{h}{s} \times \frac{w}{s} \times k'$

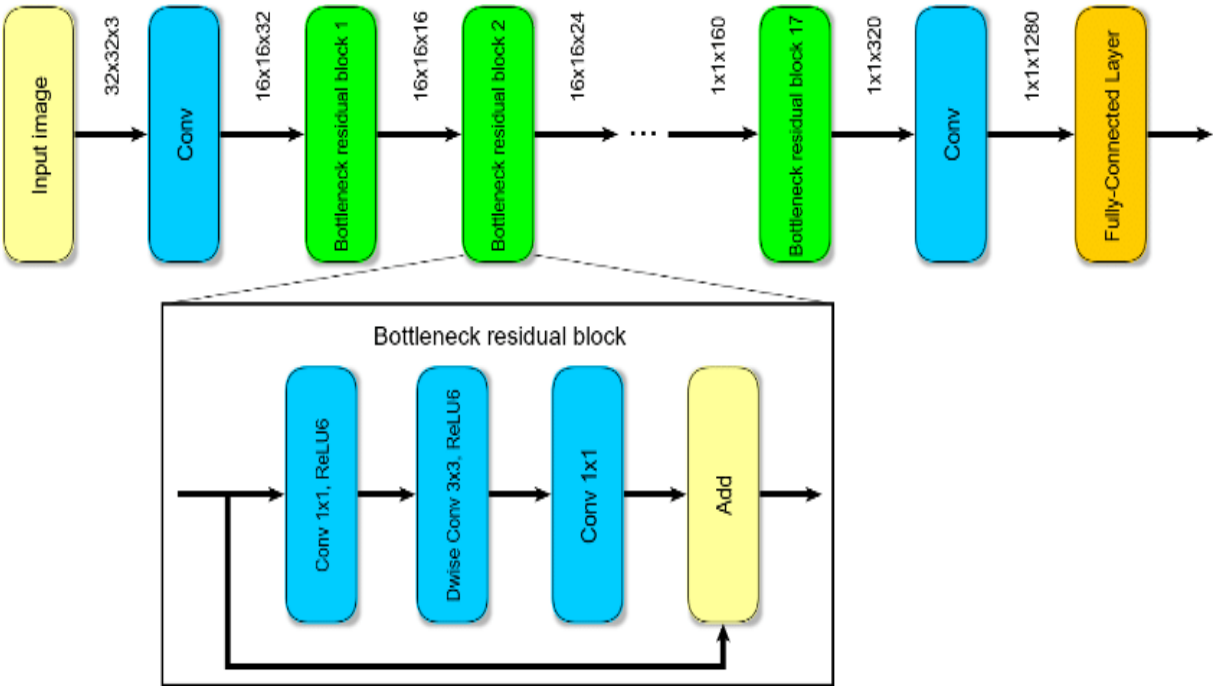


Figure 6: MobileNet-V2 Architecture. All layers in the same sequence have the same number c of output channels. The first layer of each sequence has a stride s and all others use stride 1. All spatial convolutions use 3×3 kernels. The expansion factor t is always applied to the input size as described in Table 1 [5][10].

Table 2: Custom layers on top of MobileNetV2. The table shows the layer type, output shape, and number of parameters for each layer.

Layer (type)	Output Shape	Parameters
mobilenetv2_1.00_128 (Func)	(None, 4, 4, 1280)	2,257,984
global_average_pooling2d (G)	(None, 1280)	0
dense (Dense)	(None, 128)	163,968
dense_1 (Dense)	(None, 1)	129
Total parameters	2,422,081 (9.24 MB)	-
Trainable parameters	164,097 (641.00 KB)	-
Non-trainable parameters	2,257,984 (8.61 MB)	-

3.3.5 Evaluation Metrics

To assess the performance of the MobileNet-v2 model, several evaluation metrics are used:

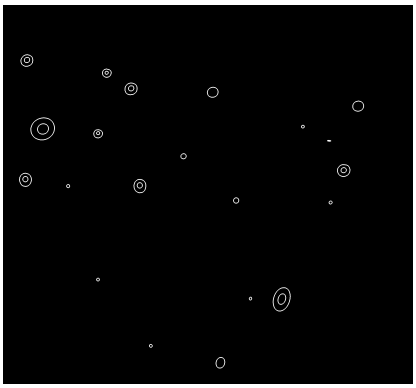
- **Mean Average Precision (mAP):** Measures the precision-recall trade-off across different thresholds, providing a comprehensive assessment of the model's accuracy.
- **Accuracy:** Indicates the proportion of correctly classified images out of the total number of images.
- **F1-Score:** Combines precision and recall to provide a single metric that balances both aspects, particularly useful for imbalanced datasets.

3.4 CCC Marker Detection, Fitting, and Centroid Calculation

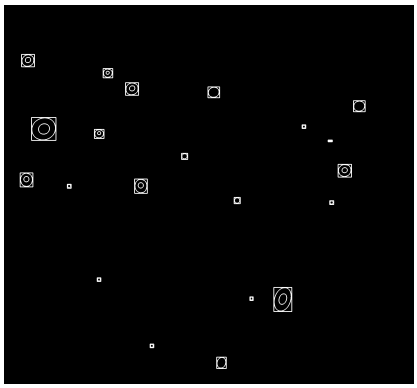
The following steps describe the procedure for fitting ellipses to detected contours and calculating their centroids. This approach was inspired by the Region-based Convolutional Neural Network (R-CNN) framework, which combines region proposal and convolutional neural network methods for object detection [11][12].

3.4.1 Bounding Box Creation

- **Contour Detection:** The pre-processed edge image is used to detect contours. Contours are curves that join all the continuous points along a boundary with the same color or intensity. The cv2.findContours function is used to detect contours in the binary edge image.
- **Bounding Box Calculation:** For each detected contour, a bounding box is calculated using cv2.boundingRect. This function computes the bounding box coordinates (x, y, w, h) where x and y are the top-left corner coordinates, and w and h are the width and height of the bounding box. These bounding boxes are stored in a list for further processing.



a) Contour Image



b) Bounding Box Fitting

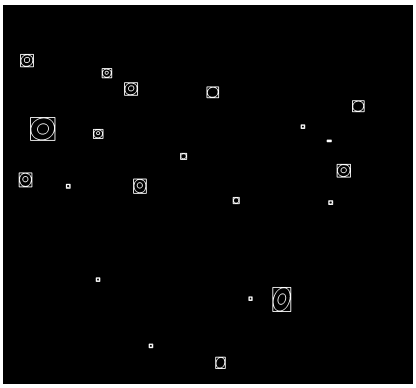
Figure 7: Fitting bounding boxes to contours.

3.4.2 Image Cropping and Resizing for Model Inference

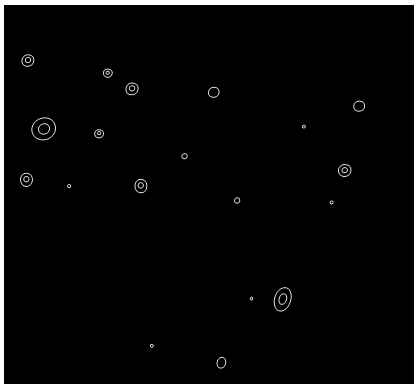
- **Image Cropping:** The regions of interest (ROIs) within the bounding boxes are cropped from the original image. The cropped images are then resized to a fixed dimension of 128x128 pixels. If the cropped region is smaller than 128x128, it is placed on a black background of the same size.
- **Normalization and Batch Preparation:** The resized images are normalized to a range of [0, 1]. The normalized images are then stacked into a batch array to be fed into a pre-trained model for ellipse detection.

3.4.3 Model Inference and Filtering

- **Model Prediction:** The batch of images is sent to a pre-trained model, which predicts whether each image contains an ellipse. Predictions are made using a threshold value (e.g., 0.5) to filter bounding boxes that likely contain ellipses.
- **Filtered Bounding Boxes** Based on the model's predictions, bounding boxes are filtered to retain only those that contain ellipses.



a) Bounding Box Image



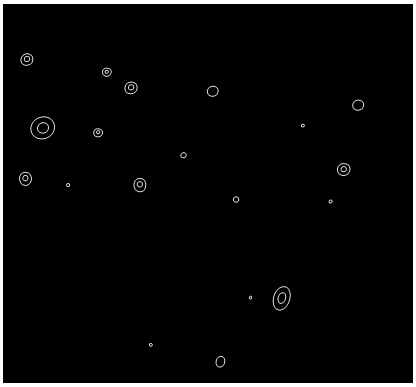
b) Contour Image with Only CCC Markers

Figure 8: Bounding boxes from image a) are cropped, resized to model input shape, stacked into batches of 32, and fed into a trained model for inference. We get an output image b), containing only ellipses.

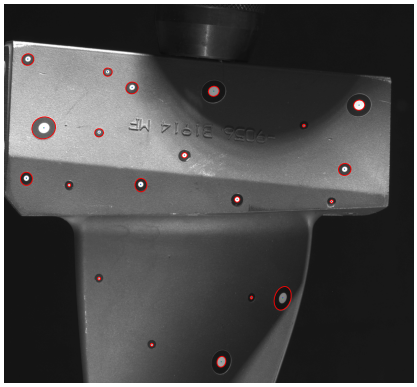
3.4.4 CCC Marker Detection, Fitting, and Center Calculation

- **Fit Ellipses to Contours:** For each contour within the filtered bounding boxes, an ellipse is fitted using the cv2.fitEllipse function, provided the contour has more than five points. The fitted ellipses are drawn on a copy of the original image.
- **Inner Ellipse Area Moments for Center Calculation:** A mask is created for the bounding box around each contour. Otsu's thresholding is applied within the bounding box to segment the inner ellipse area. The moments of the region inside each bounding box are calculated using cv2.moments. The center of each bounding box is computed using the moments, which gives the coordinates (center_x, center_y).

- **Direct Center of Fitted Ellipse:** The center of the fitted ellipse is extracted directly from the cv2.fitEllipse function. This center is less accurate for distorted ellipses but provides a quick estimate of the center.



a) Image with Only CCC Marker Contours



b) Ellipse Fitting and Center Localization)

Figure 9: The image containing only CCC marker contours is then processed further to fit ellipses to the contours and calculate center positions.

4 Performance Evaluation

4.1 Robustness Analysis

To evaluate the performance of the CCC marker detection and center calculation methodology, a robustness analysis was conducted using a set of 10 images from the original test setup. Each image may contain multiple CCC markers, and the detection accuracy was assessed by comparing the detected markers against the ground truth.

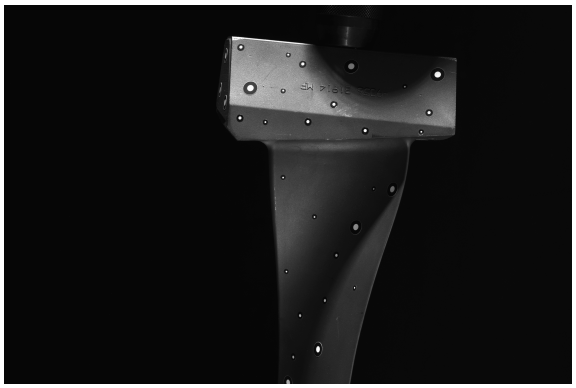
Detection Accuracy: This metric measures the proportion of correctly detected CCC markers out of the total number of markers present in the test images. It is calculated using the formula:

$$\text{Detection Accuracy (DA)} = \frac{\text{Number of correctly detected markers}}{\text{Total number of markers}}$$

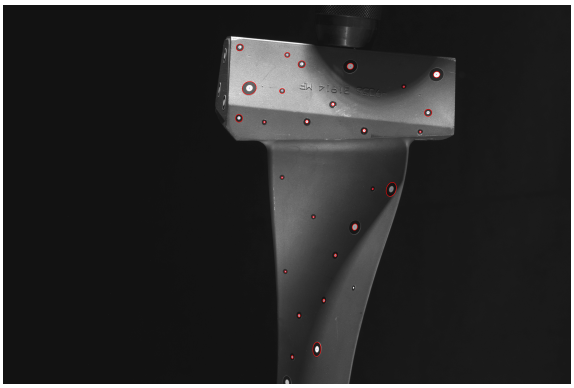
The robustness analysis was performed on 10 test images, each containing multiple CCC markers. The detection accuracy was calculated for each image, and the results are summarized in the table below:

Table 3: CCC-Marker Detection Accuracy

Image	1	2	3	4	5	6	7	8	9	10
DA(%)	87.5	87.5	100	100	100	100	100	96.66	87.5	100



a) Input Image with CCC Markers



b) Output Image with Detected Markers

Figure 10: Comparison of the input image with CCC markers (a) and the output image showing the detected ellipses (b) under controlled lighting in the setup.

The robustness analysis of the CCC marker detection algorithm reveals that it performs with a high degree of accuracy. The average detection accuracy across the test set is approximately 96%, with individual accuracy values ranging from 87.5% to 100%. This high average indicates that the algorithm is reliable and consistent in detecting CCC markers under various conditions.

One of the notable strengths of the algorithm is its robustness to varying brightness conditions. In real-world scenarios.

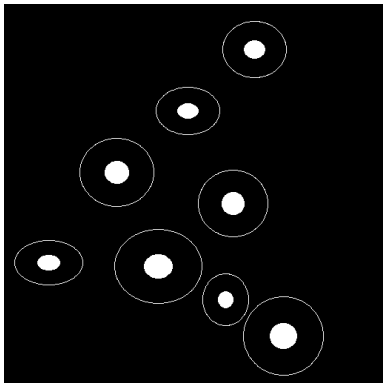
a) Input Image with CCC Markers

b) Output Image with Detected Markers

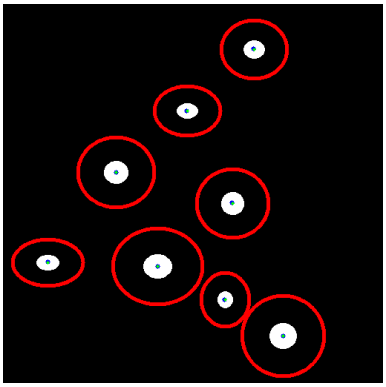
Figure 11: Comparison of the input image with CCC markers (a) and the output image showing the detected ellipses (b) under natural lighting.

4.2 Accuracy Analysis

The accuracy analysis assesses the performance of the ellipse detection and center calculation methodology. This analysis is conducted through visual inspection, subjective assessment, and a set of metrics including Mean Squared Error (MSE), Mean Absolute Error (MAE), and Mean Euclidean Distance (MED) for both the X and Y coordinates of the ellipse centers.



a) Input Image with CCC Markers



b) Output Image with Detected Markers

Figure 12: Comparison of the input image with CCC markers (a) and the output image showing the detected ellipses and computed centers (b).

To conduct this analysis, 10 synthetic images have been used to evaluate the performance, focusing on detection quality, localization precision, and robustness to variations in image content and conditions. For each image, the following metrics have been computed:

- Root Mean Squared Error in X (RMSE X): Measures the square root of the average squared difference between the detected and ground truth X-coordinates of the ellipse centers.
- Root Mean Squared Error in Y (RMSE Y): Measures the square root of the average squared difference between the detected and ground truth Y-coordinates of the ellipse centers.
- Root Mean Squared Error (RMSE): The average of RMSE X and RMSE Y.

- Mean Absolute Error in X (MAE X): Measures the average absolute difference between the detected and ground truth X-coordinates of the ellipse centers.
- Mean Absolute Error in Y (MAE Y): Measures the average absolute difference between the detected and ground truth Y-coordinates of the ellipse centers.
- Mean Absolute Error (MAE): The average of MAE X and MAE Y.
- Mean Euclidean Distance (MED): Measures the average Euclidean distance between the detected centers and the ground truth centers.

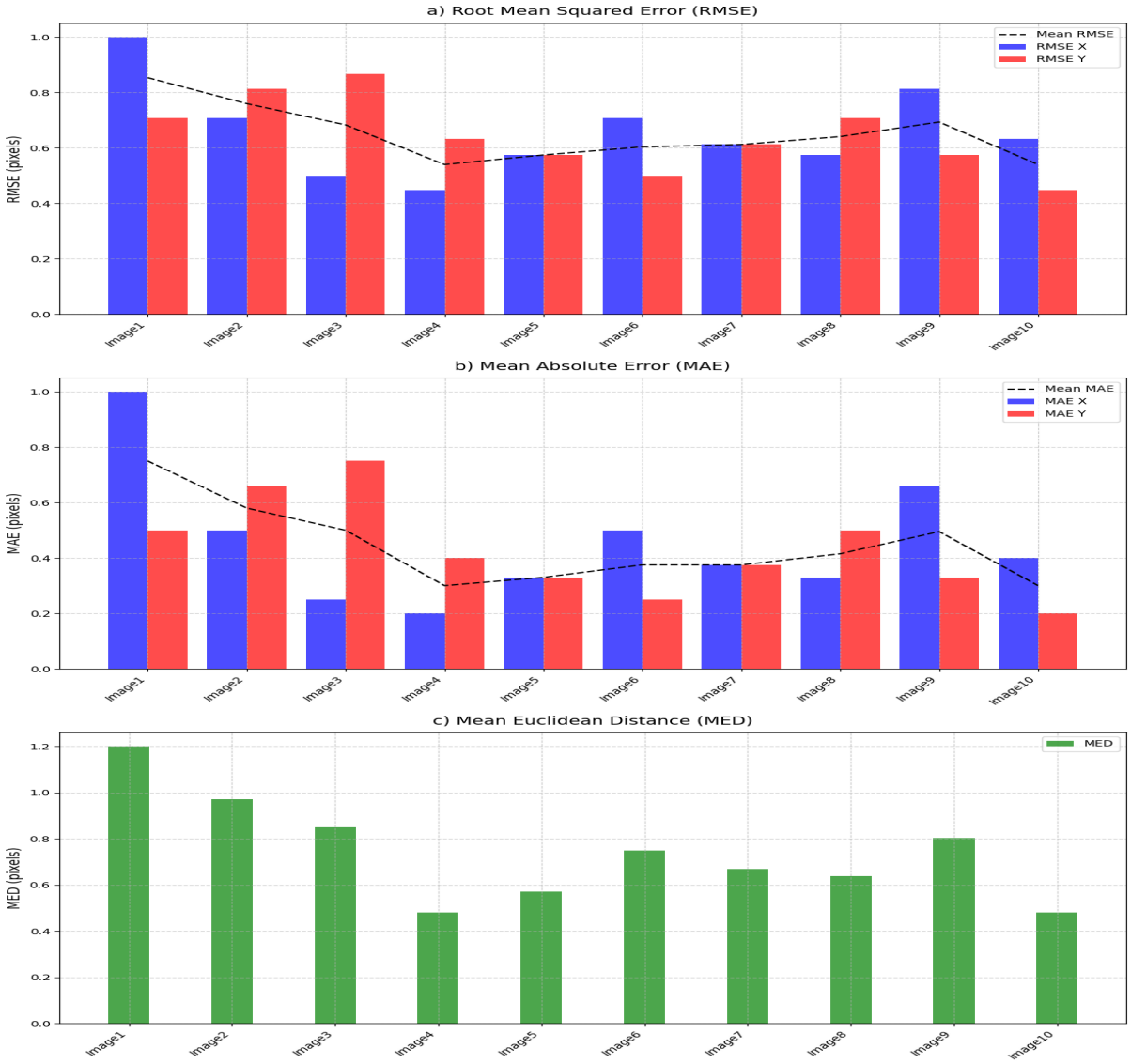


Figure 13: Accuracy Analysis Metrics a) RMSE b) MAE C)MED.

As the original test data does not contain information about the center locations of the CCC markers, synthetic test images with known center locations have been created to perform the assessment. This approach ensures the evaluation is conducted with precise ground truth data.

The accuracy evaluation metrics highlight the algorithm's ability to accurately detect and localize the centers of CCC markers. The average RMSE over the test images is **0.6501** pixels, the

average MAE is **0.4426** pixels, and the average MED is **0.7400** pixels. These metrics indicate that the methodology performs well in detecting and localizing the markers with minimal error.

The accuracy analysis demonstrates the high precision and reliability of the ellipse detection and center calculation methodology. The low error metrics and visual inspections confirm that the algorithm can accurately detect and localize CCC markers, making it suitable for practical applications requiring precise marker tracking. The use of synthetic test images with known center locations further ensures the authenticity of the evaluation process.

4.3 Performance Comparison with Previous Model

In this section, we compare the performance of our proposed method against a previous model presented in the thesis *Marker-based 6DoF Pose Estimation with Deep Learning* by Yihui Ren [13]. The comparison focuses on key error metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Mean Euclidean Distance (MED). Our method demonstrates significant improvements in accuracy, as shown in the results.

For clarity, the error metrics of the previous model are provided in millimeters (mm), whereas our method's metrics are in pixels. Considering the actual conversion factor is around 20 pixels per millimeter, but for synthetic test images and to accommodate real test factors, we set the conversion factor to 10 pixels per millimeter. We will present both sets of results in the same unit (millimeters) to facilitate a direct comparison.

Table 4 presents the performance metrics of both methods, highlighting the advancements achieved by our approach.

Table 4: Comparison of Performance Metrics Between Proposed Method and Previous Model

Metric	Previous Model (mm)	Proposed Method (mm)
MAE (X-direction)	0.84	0.0442
MAE (Y-direction)	0.90	0.0442
RMSE (X-direction)	0.88	0.0650
RMSE (Y-direction)	0.94	0.0650
MED	N/A	0.0741

The previous model, as detailed in Yihui Ren's thesis [13], reported an MAE of 0.84 mm in the X-direction and 0.90 mm in the Y-direction. The RMSE values were 0.88 mm and 0.94 mm in the X and Y directions, respectively.

In contrast, our proposed method achieves significantly lower MAE and RMSE values in both directions. Specifically, the MAE is reduced from 0.84 mm and 0.90 mm to 0.0442 mm in both the X and Y directions. Similarly, the RMSE is reduced from 0.88 mm and 0.94 mm to

0.0650 mm. Additionally, our method reports an Average MED of 0.0741 mm, demonstrating its superior accuracy and robustness.

In summary, the proposed method provides substantial improvements over the previous model, reducing errors by an order of magnitude and offering a more precise and reliable performance.

In addition to quantitative metrics, we also compared the visual performance of our method against the previous model under varying brightness conditions. The previous model struggled with marker detection when the brightness was too low, resulting in a detection failure rate of 30-40%. In contrast, our method can detect almost 90% of the markers even under low brightness conditions, though it may produce some noise and false positives, which can be addressed with post-processing techniques.

Figure 14 shows a comparison of the results from both methods under low brightness conditions:

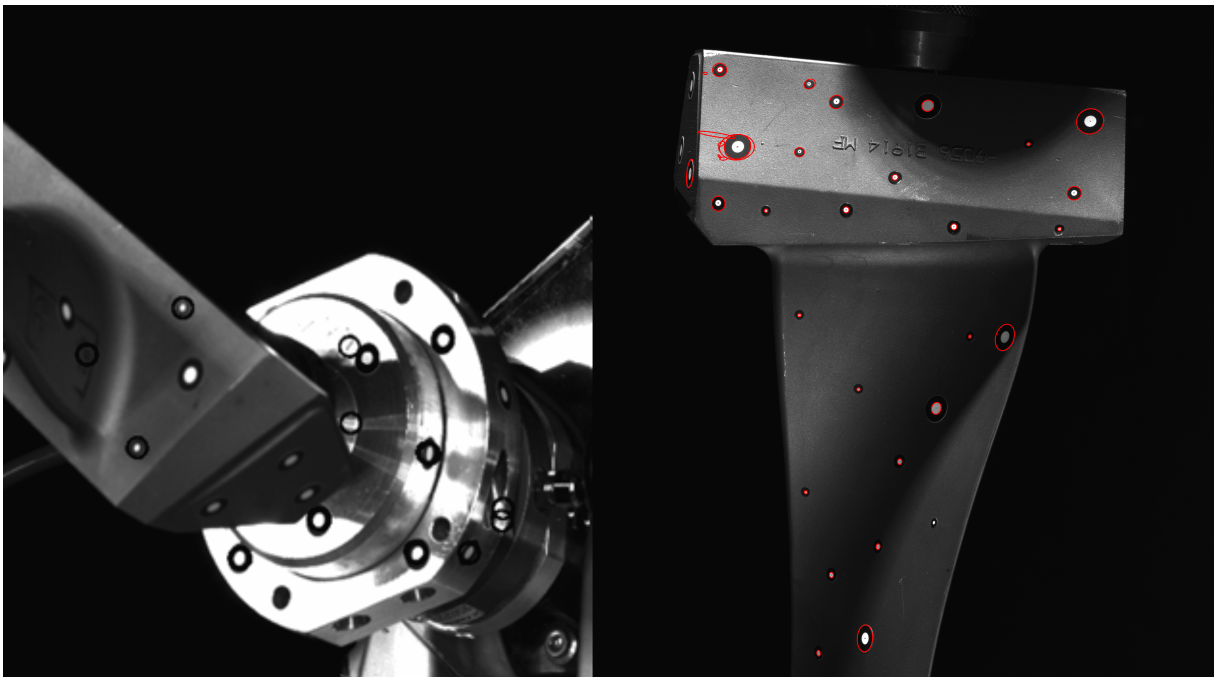


Figure 14: Comparison of Marker Detection Results Under Low Brightness Conditions. The left image shows results from the previous model, which fails to detect many markers due to low brightness. The right image shows results from the proposed method, which successfully detects the majority of markers even under challenging conditions[13].

As illustrated in Figure 14, the previous model is unable to detect a significant portion of the markers in low brightness conditions. In contrast, our method maintains high detection accuracy, demonstrating its robustness to varying lighting conditions. While our method does produce some noise and false positives, these can be managed with post-processing techniques to further enhance the results.

In conclusion, our proposed method not only outperforms the previous model in terms of quantitative accuracy metrics but also shows superior performance in practical scenarios with challenging conditions, such as low-brightness environments.

5 Conclusion and Future Work

This project has successfully tackled the computational challenges associated with detecting and localizing CCC markers in images, demonstrating both high performance and robustness in the context of various image conditions. The methodology encompassed a comprehensive approach involving image preprocessing, edge detection, contour extraction, and deep learning-based object detection, which together facilitated accurate marker detection and localization. The preprocessing steps included contrast enhancement, noise reduction, and lighting normalization to optimize image quality, followed by the application of Laplace and Sobel edge detectors to identify marker edges. Adaptive thresholding and the Canny edge detector were used for effective contour detection, while a pre-trained deep learning model enabled precise object classification and localization. The framework also utilized ellipse fitting and centroid calculation techniques to refine the detection results and visualize the detected markers.

The quantitative analysis revealed a high average detection accuracy of 96%, with results ranging from 87.5% to 100%. This indicates that the algorithm is robust and effective at tracking CCC markers under various lighting conditions. The qualitative analysis, conducted using synthetic test images due to the unavailability of ground truth center information for the original test images, validated the reliability of the ellipse detection and center calculation methods. Metrics such as Mean Squared Error (MSE), Mean Absolute Error (MAE), and Mean Euclidean Distance (MED) were employed to assess the accuracy of the detected marker centers, demonstrating a strong performance in terms of both detection quality and localization precision.

Despite these successes, there are several areas for future work and potential improvements. One significant aspect for future development is optimizing the model's inference time. Currently, the model's inference time ranges from 57ms to 110ms depending on the image size, with some TensorFlow warnings affecting performance. Addressing these warnings and exploring techniques for model optimization, such as model compression or alternative architectures, could help achieve a more consistent and reduced inference time, bringing the model closer to real-time processing capabilities.

In addition to optimizing performance, future work should focus on expanding the test dataset to include a broader variety of scenarios. This includes varying lighting conditions, marker occlusions, and complex backgrounds, which would enable a more comprehensive evaluation of the system's robustness and adaptability. Furthermore, obtaining accurate ground truth data for original test images would provide a foundation for more rigorous qualitative analysis and algorithm refinement.

Moreover, future efforts could involve integrating the CCC marker detection system with other computer vision technologies for multi-modal analysis. Exploring applications in diverse fields, such as medical imaging, industrial inspection, or advanced robotics, could uncover new opportunities and challenges, further pushing the boundaries of computer vision technology.

In conclusion, this project has established a solid foundation for accurate and robust CCC marker detection and localization. Moving forward, the focus should be on enhancing model efficiency, expanding test scenarios, and exploring real-world applications, which will contribute to the advancement of more sophisticated and reliable computer vision systems.

Bibliography

[1] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*. Upper Saddle River, NJ: Prentice Hall, 2002.

[2] J. Canny, “A computational approach to edge detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679–698, 1986.

[3] L. Calvet, P. Gurdjos, C. Griwodz, and S. Gasparini, “Detection and accurate localization of circular fiducials under highly challenging conditions,” 06 2016.

[4] X. Liang, M. Hirano, and Y. Yamakawa, “Real-time marker-based tracking and pose estimation for a rotating object using high-speed vision,” *Journal of Robotics and Mechatronics*, vol. 34, no. 5, pp. 1063–1072, 2022.

[5] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “Mobilenetv2: Inverted residuals and linear bottlenecks,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4510–4520, 2018.

[6] W. Dong, P. Roy, C. Peng, and V. Isler, “Ellipse r-cnn: Learning to infer elliptical object from clustering and occlusion,” *IEEE Transactions on Image Processing*, vol. 30, p. 2193–2206, 2021.

[7] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications,” 4 2017.

[8] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, “A survey on deep transfer learning,” 2018.

[9] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2017.

[10] U. Seidaliyeva, D. Akhmetov, L. Ilipbayeva, and E. Matson, “Real-time and accurate drone detection in a video with a static background,” *Sensors*, vol. 20, p. 3856, 07 2020.

[11] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2017.

[12] R. Girshick, “Fast r-cnn,” in *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1440–1448, 2015.

[13] Y. Ren, “Marker-based 6dof pose estimation with deep learning,” 2022. Lehrstuhl für Fertigungsautomatisierung und Produktionssystematik, Prof. Dr.-Ing. J. Franke, Betreuer: M.Sc. O. Kedilioglu.