Available online at www.coins-indocompt.org

COINS JOURNAL
Computing and Information System Journal

COmputing and INformation Systems Journal

IndoCompt PUBLISHER

Vol. 1, No. 1, April, 2025

Automatic Coin Detection and Counting Based on Image Processing with *OpenCV Python*

Fattachul Huda Aminuddin[*a], Ahmad Husna Ahadi[b], Novhirtamely Kahar[c]
Sukma Puspitorini[d], Elzas[e]

Nurdin Hamzah University, Jl. Kolonel Abunjani, Selamat Danau Teluk, Telanaipura, Jambi City, 36124, Indonesia

**fattachulhuda@unh.ac.id*

Abstract

Image processing is an important field in object detection and calculation, especially in the field of computer vision. A problem that often arises in automatic coin counting is the inability of machines to accurately recognize and calculate coins quickly when there are variations in conditions, such as differences in size, color, lighting, and orientation of coins in the imagery. This research develops OpenCV and Python-based methods to detect and calculate the number of coins in the image. The stages of this method include pre-processing, segmentation, feature extraction, and object detection. Pre-processing improves image quality and reduces noise, while segmentation separates coin objects from the background. The extraction feature identifies the characteristics of the coin such as edges and colors, aiding in the object detection process. This method was tested on a dataset of coin images with various variations in size, exposure, and orientation, and the results showed an average accuracy rate of 98.08%. This research contributes to the development of automated systems for object counting in various contexts and can be the basis for further research in the field of image processing and computer vision

Keywords: Coin Detection, Object Counting, Image Processing, Python, OpenCV

1. Introduction

In the era of information technology that continues to develop, image processing has become an increasingly important and relevant research field. The use of image processing technology is not only limited to the field of computer science, but has also spread to various sectors such as medicine, industry and security. With image processing, it is possible to change the initial image into a better quality image, so that it is easily recognized by both humans and computer machines [1] . One of the practical applications of this technology is the automatic detection and counting of objects in images, which has many benefits in everyday life and the industrial sector. Coin detection and counting are examples of real applications that have a wide range of applications, such as in vending machines, banking devices, and digital payment systems. A problem that often arises in automatic coin counting is the machine's inability to accurately recognize and count coins quickly when conditions vary, such as differences in size, color, lighting and orientation of coins in the image. In this research, the main focus is the detection and counting of the number of coins in images using OpenCV Python. OpenCV python is an open source library that can perform image processing functions that aim to process data visually by humans and computers.[2]

Object detection and counting. Based on the results of a study entitled Coin Sum Counter from an Image in tourist attractions and holy places in India by Tarang Kushwaha, Ujjwal Verma, Tanya Chaube in 2023, that the automatic coin detection system is very helpful for the process of separating coins and counting their amounts, this process will provide more accurate and efficient results, because it has a high level of accuracy and helps reduce time consumption. The research conducted uses an image classification model approach to classify INR coins from images and uses edge detection, gaussian blur, and other techniques to correctly identify the edges of coins as well as to determine coin fractions using Python and the Kaggle dataset [3]. The results of a similar study entitled Recognition of Counterfeit Currency Using OpenCV and Python in 2020 show that applications developed with the opencv python technique with the ORB model, gross force matching, and the KNN method have resulted in an efficient approach in detecting paper

currency accurately. With low costs, the resulting applications are very useful and efficient for everyone and this system will also be developed in various other countries [4], [5]

The method used in this study involves a series of stages, including pre-processing to improve image quality, segmentation to separate coin objects from the background, feature extraction to identify coin-specific features, and object detection using OpenCV Python. This research aims to contribute to the development of an automated system that can recognize and calculate the number of coins in an image quickly and efficiently. The use of OpenCV Python is expected to provide an effective solution in the implementation of object detection algorithms. By conducting this research, it is hoped that it can open up further development opportunities in the field of computer image processing and vision, as well as provide a solid foundation for practical applications in various industries [6].

1.1. *Novelty of the Study*

This study presents several novelties compared to previous works. First, the coin detection method using the HoughCircles algorithm is enhanced with adaptive parameter tuning to address challenges in detecting overlapping coins and varying lighting conditions. Unlike prior studies that primarily relied on basic edge detection methods and limited datasets, this research employs a comprehensive preprocessing pipeline combined with adaptive HoughCircles parameters to improve accuracy under diverse lighting and overlapping scenarios. Additionally, the use of an authentic Indonesian rupiah coin dataset provides more practical and realistic validation compared to simulated datasets used by other studies. Experimental results demonstrate an accuracy improvement up to 98.08%, surpassing the average accuracy reported in earlier research. Furthermore, this study offers significant practical contributions to automated coin counting applications. Therefore, this work not only enhances detection speed and accuracy but also lays a foundation for future development of more complex image-based object detection systems.

2. Research Method

2.1. *Methodology*

The object of research in this study is in the form of a metal coin dataset that is used as a sample for researchers. The sample results will be detected using opencv python to calculate the number of detected images. This research uses several stages to develop a coin detection system using opencv python, namely: first, the data collection process in the form of a data set and images of coins to be used. The second is the image pre-processing stage which consists of 3 processes, the image conversion process to Grayscale, the image smoothing process, and edge detection. The third stage is the process of detecting coins in the form of circles with the houghtcircles algorithm and setting the minimum and maximum radius to filter the coin objects to be detected. The fourth process is the process of calculating the number of coins detected based on the circular objects detected by the system. The fifth process is the visualization process and the result validation process, where this process will display an image with a circle coin to be visualized along with evaluating the detection results and calculations to ensure the accuracy of the object. The flow diagram in developing a coin object detection system using opencv python is shown in Figure 1.

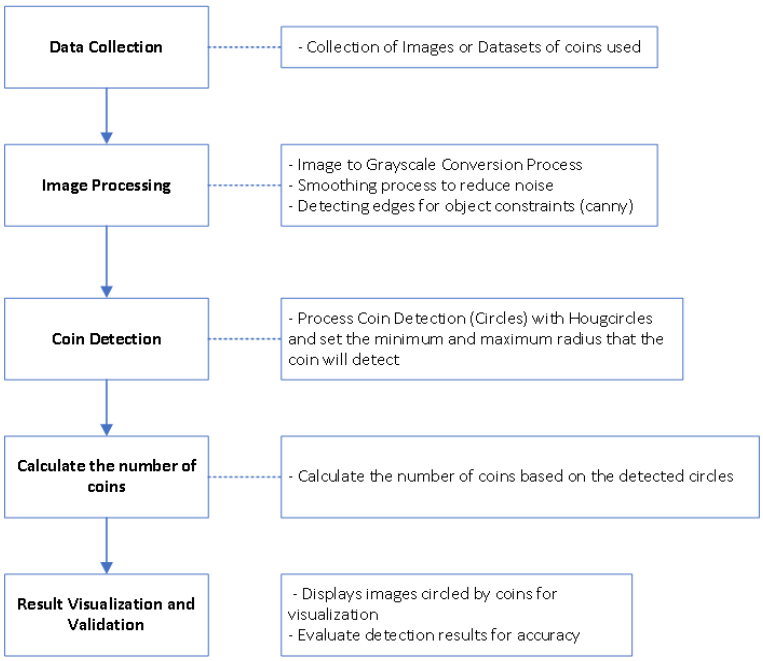

Figure 1. Research Method

2.2. *Data Collection*

At this stage, the first step is the process of collecting a dataset in the form of collections of coin images that will be detected using a camera or scanner in the form of full RGB format. In this process, the quality of the image is also very decisive in terms of lighting, shadows or reflections from the image results. This study uses a dataset in the form of rupiah coins in the form of metals consisting of Rp. 1, 5, 25, 50, 100, 200, 500, and 1000 notes and several samples of foreign courses. At this stage, the researcher tried to use a sample of the coin image dataset with a white background that is already available as shown in Figure 2.

Figure 2. Dataset

2.3. *Image Processing*

This process is a stage to improve image quality to make it easier to segment and detect objects. It involves 3 (three) processes, namely: conversion to Grayscale, which is a process to reduce the color dimension to a single intensity, as well as reduce the complexity of calculations. Grayscale is a color pixel that is at a value between the black and white gradations [7]. Furthermore, the noise removal process or smooting process which is useful for cleaning the image from the artifact using a Gaussian Blur filter or medium blur. Gasussian blur filter is a method used to remove noise very well from the results of the digital image capture process using a digital camera based on the natural conditions and sensitivity of the reflected light sensor [8]. This process is also used to enhance the detail of a specific area or often called “stretching gama”.

2.4. *Coin Detection*

This process involves 4 (four) image segmentation processes, the first is the Thresholding process. This process is used to separate objects (coins) from the background by thresholding methods such as otsu or adaptive thresholding. The second process is Edge Detection. This process uses the Canny Edge Detection algorithm to detect edge objects on the coin. The three Morhological operations processes are used to improve segmentation results, for example filling gaps in the edges of coins. The last process that detects the circle. This process uses the hough circle transform method to recognize and detect the shape of the coin's circle, as well as adjust the parameters of the object's circle (coin) such as the minimum and maximum radius according to the detected coin.

2.5. *Calculate the number of coins*

This process will calculate the number of coins detected from each coin circle. When calculating the circle of the number of coins, the radius of the circle can be analyzed to distinguish the value of coins from small coins and large coins. If the coin is not detected, then the radius of the circle needs to be readjusted. Each detected coin is assigned a value based on the classification results, and summing is done to calculate the total value. At this stage, it is also a process to identify coins based on their size, so that later the coins will be detected and calculated in number with a scalling ratio approach of relative size differences

2.6. *Result Visualization and Validation*

The purpose of this process is to assess the accuracy of the detection and calculation system. This process involves parameters in evaluating, first precision is the system's ability to detect the coin percentage correctly. The second is the

recall process, this ability to detect all existing coins. In the third stage, a combination of precision and recall to detect the overall evaluation. The advantage of the system using OpenCV python from this stage is that it is able to detect various types of coins with adjustable radius parameters

3. Result And Discussions

The results of image processing use opencv python to calculate the number of coins detected using the visual code editor application, as well as using the opencv library and the python programming language.

3.1. *Coin Images*

The coin image is the material that the coin will detect using opencv python. The images used are 7 images consisting of aluminum, silver and metal. The amount of money varies, namely: Rp., 5, Rp., 10, Rp., 25., Rp. 50, Rp. 100, Rp. 200, Rp. 500, Rp. 1000 and several foreign currency samples. The images obtained were taken from internet sources and some using smartphones.

3.2. *Image to Grayscale*

At this stage the image detected by opencv python is the original image in JPG format. This image will later be converted to Grayscale or gray color first using the threshold as in Figure 3.

Figure 3. Convert image to grayscale

3.3. *Gaussian Blur*

At this stage, the image that has been processed into Grayscale using a threshold will be processed and performed using the GaussianBlur filter. The purpose of this filter is to remove the noise present in the image. The following is a list of programs used to use the Gaussian filter listed in Figure 4.

```
# Using GaussianBlur to reduce noise
blurred = cv2.GaussianBlur(gray, (15, 15), 0)
```

Figure 4. Using gaussian blur listing program

3.4. *Coin Detection with Hougcircles*

At this stage, it is the process of detecting circles on the coin. The firmer the circle of the coin, the more accurate the coin will be detected. The process of using hougcircles is a process that involves parameters in the form of a minimum radius of a circle, a maximum radius of a circle, and a minimum distance between circles. This process also involves the Threshold canny method in detecting the edge of the circle. This process is also affected by the quality of the image, and the distance of the coins. For more details, you can see the program code in Figure 5.

```
# Detecting circles using HoughCircles
circles = cv2.HoughCircles(blurred, cv2.HOUGH_GRADIENT, dp=1, minDist=30, param1=50,
param2=30, minRadius=30, maxRadius=50)
```

Figure 5. Detection method with hougcircles

Dp 1 is the resolution of the accumulator, that is, the image that has the same resolution as the original image. minDist is the minimum distance between the centers of a circle, in order to prevent double detection of adjacent circles. Param1 is the method used to detect edges, this stage uses the canny edge detection method as the first coin threshold. Param2 is used for the accumulator threshold to decide whether the system's detected loops are valid or not. The higher the value, the fewer circles will be detected. minRadius and maxRadius are the radius of the detected coin circle

3.5. *Image Result and Coin Calculation*

After the detection stage uses the hougcircles and canny method, then the next step is to count the detected coins. As for counting coins, use a code like the following Figure 6.

```
    # Calculating the number of coins
    coin_count = len(circles)
    cv2.putText(frame, f"Jumlah Koin: {coin_count}", (10, 30), cv2.
    FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2)

else:
    coin_count = 0
    cv2.putText(frame, "Tidak ada koin terdeteksi", (10, 30), cv2.
    FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255), 2)

return frame, coin_count
```

Figure 6. Detection method with hougcircles

In the experiment conducted on the coin sample 1, it showed that the results were detected well and accurately according to the number of coins and the edges of the circle, which was 4 coins. The results of the trial are shown in Figure 7.

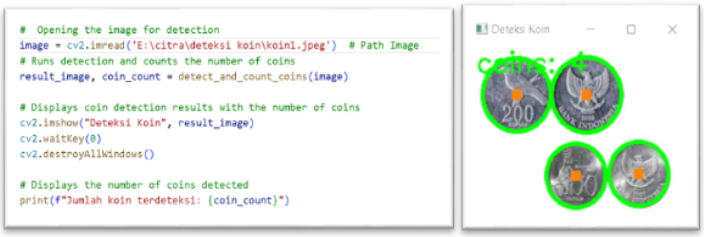

Figure 7. Calculate the number coins code

The next result uses another image with a different number of coins. The results of the trial are shown in Figure 8.

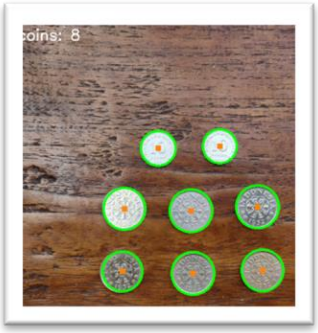

Figure 8. Different coin detection result

Figure 8 shows the results of a test of the coin detection system using a new image with a different number of coins from the previous sample. Based on the results of the image processing, the system managed to detect as many as eight coins marked with green circles as a result of the HoughCircles algorithm, as well as an orange dot in the center of each coin that represents the coordinates of the center of the object. The accuracy of the detection in this image shows the success of the method in recognizing circular objects despite the difference in position and surface lighting. Background imagery with a wood texture also does not have a significant influence on the segmentation process, which shows the system's resistance to visual interference from the background. The ability of the HoughCircles algorithm to consistently detect circular objects has been proven through various previous studies, which have shown that this method is effective against variations in orientation and scale of objects and resistant to visual noise [9]. In addition, the use of OpenCV as an object detection platform simplifies the implementation and validation process through a responsive graphical interface and directly visualized detection results [10].

Based on the two test samples shown, the OpenCV Python-based coin detection system shows accurate and stable performance in detecting circular objects. In Figure 8, the system managed to detect as many as eight coins with high precision, even though the background has a wooden texture that has the potential to cause interference. Meanwhile, in another image, Figure 7, the system processes an image containing four different coins and displays the detection results visually through a graphical interface window labeled "Coin Detection", as well as printing the number of detected coins to the terminal output. These two results show that the integration between pre-processing, segmentation, and detection processes using the HoughCircles algorithm is able to work consistently against images with variations in orientation, coin count, and background contrast. Direct visualization of the detection results through the OpenCV GUI also helps the system validation process practically. Therefore, it can be concluded that this system is effective as an early prototype of automatic coin detection and counter applications, and has the potential to be further developed, such as the classification of coin values based on size or surface imagery. The overall results of the test on the coin sample data are presented in Table 1 below.

Table 1. Coin Calculation Detection Result

File name	Original Image	Detection Result	Number of Coins
Koin1			Counted: 4
Koin2			Counted: 9
Koin3			Counted: 8
Koin4			Counted: 14
Koin5			Counted: 10
Koin6			Counted: 6
Koin7			Counted: 6

3.6. *Result of Analysis*

Based on the results of the analysis of the coin image dataset that has been tested based on Table 1 above, some of the results of the analysis by level are presented in Table 2:

Table 2. Result of accuracy analysis

Dataset	Number of original coins	Number of coins detected	True detection	Accuracy Level
coin1	4	4	4	100%
coin2	10	9	9	93,33%
coin3	8	8	8	100%
coin4	14	14	14	100%
coin5	10	10	10	100%
coin6	6	6	6	100%
coin7	7	6	6	93,33%
Average				98.09 %

Based on Table 2 above, the coin detection algorithm has an excellent overall accuracy rate. Of the 7 datasets tested, 5 datasets (coin1, coin3, coin4, coin5, coin6) achieved 100% accuracy, indicating that the algorithm was able to detect all coins in the image correctly. Meanwhile, on coin2 and coin7, the algorithm failed to detect all coins. On coin2, 1 coin was undetected out of 10 coins, and the resulting accuracy was 93.33%. Similarly, on coin7, only 6 were detected and 1 was undetected, and the same accuracy was 93.33% for coin2. The 2 datasets that were not detected perfectly were affected by several things, namely: poor image quality, as well as overlapping coin positions, making it difficult for the system to detect. An average of 98.09% accuracy was obtained. This proves that the accuracy level of the python opencv system is very good and accurate.

4. Acknowledgements

The author would like to thank the lecturers involved in the research, as well as the academic community of Nurdin Hamzah University for the support and facilities that have been provided during the implementation of this research.

5. Funding

This research does not obtain external funding from any institution or funding institution, whether public, commercial, or non-profit.

6. Declaration of Competing Interest

The authors declare that they have no known competing financial or non-financial interests that could have influenced the outcomes of this research.

7. References

[1] Rewina, A. E., Sulistyowati, S., Kurniawan, M., N, M. D., & Yunanda, S. F. (2024). Application of CNN (Convolutional Neural Network) Method in Classifying Banknotes and Coins. TIN: Terapan Informatika Nusantara, 4(12), 778–785. https://doi.org/10.47065/tin.v4i12.5128

[2] Ulfah, J., & Nurdin, N. (2023). IMPLEMENTATION OF THE CANNY EDGE DETECTION METHOD TO CALCULATE THE NUMBER OF COINS IN THE IMAGE USING OPENCV. Jurnal Informatika Dan Teknik Elektro Terapan, 11(3). https://doi.org/10.23960/jitet.v11i3.3147.

[3] Kushwaha, T., Verma, U., Chaubey, T., Tibrewal, T., & Khan, N. U. (2023). Coin Sum Counter from an Image. *International Journal for Research in Applied Science and Engineering Technology*, *11*(1), 687–691. https://doi.org/10.22214/ijraset.2023.48655.

[4] A. Akshatha, A. P. B. P. C. G. (n.d.). *Recognition of Counterfeit Currency Using OpenCV and Python*. Kushwaha, T., Verma, U., Chaubey, T., Tibrewal, T., & Khan, N. U. (2023). Coin Sum Counter from an Image. *International Journal for Research in Applied Science and Engineering Technology*, *11*(1), 687–691. https://doi.org/10.22214/ijraset.2023.48655.

[5] Akshatha, A., Punuganti, A., Panduranga, B., & Girish, C. (2020). Recognition of counterfeit currency using OpenCV and Python. International Journal of Research in Engineering, Science and Management, 3(7), 408–412.

[6] Baygin, M., Karakose, M., Sarimaden, A., & Akin, E. (2018). An image processing based object counting approach for machine vision application. arXiv preprint arXiv:1802.05911.

[7] Nurdin, N., Hamdhana, D., & Setiawan, M. J. (2017). A System for Detecting the Pattern of The Words of Allah and Muhammad in The Image of the Qur'an Using the Peirce Method. *TechsI - Jurnal Teknik Informatika*, *9*(2), 78–90. https://doi.org/10.29103/techsi.v9i2.215.

[8] Yuwono, B. (2015). Image Smoothing Uses Mean Filtering, Median Filtering, Filtering Mode and Gaussian Filtering. *Telematika*, *7*(1). https://doi.org/10.31315/telematika.v7i1.416

[9] Y. Zheng, L. Wu, and D. Zhang, "Robust circle detection algorithm based on Hough Transform," *Journal of Visual Communication and Image Representation*, vol. 67, p. 102724, 2020, doi: 10.1016/j.jvcir.2019.102724

[10] S. Gupta and R. Patel, "Real-time coin detection using OpenCV and Python," *International Journal of Computer Applications*, vol. 180, no. 47, pp. 10–13, 2018.