

Painting Security System using TensorFlow Based Object Detection Method

Muhammad Yoga Akbar Prasetya¹, M. Abdullah Anshori², Rieke Adriati Wijayanti³

^{1,2,3} Digital Telecommunication Network Study Program,
Electrical Engineering Department, State Polytechnic of Malang, Indonesia

¹1941160014@student.polinema.ac.id, ²anshori_ma@polinema.ac.id, ³riekeaw@polinema.ac.id

Abstract— Painting is an art form that should be preserved. Raden Saleh, Affandi, Hendra Gunawan, and Nyoman Gunarsa's paintings are among those of outstanding artistic worth. The existence of paintings may be jeopardized if the painting protection system is poor. Several well-known examples of art theft include the theft of Affandi's paintings in 1973 and Raden Saleh's paintings in 2007. Apart from famous paintings, personal paintings, which are equally expensive, are frequently stolen. This, of course, concerns the general public, particularly private art collectors with minimal security. Based on these problems, specific management is needed through the "SI LUKAS" Painting Security System. Using the TensorFlow-Based Object Detection Method, a system innovation that can continuously monitor the whereabouts of artworks and send notifications via Telegram, can be accessed from anywhere and at any time using TensorFlow. Based on evaluating the system's delay in reading items at a distance, it is discovered that the system can detect all objects, specifically hands and paintings, within a distance of 50cm to 550cm, however the latency varies. The average accuracy of object detection from various angles was found to be 89.6% for hand objects and 96.4% for painting objects, placing them in the very high category. In other words, the SI LUKAS system is implementable.

Keywords— *Object Detection, Painting, Security System, SI LUKAS, TensorFlow*.

I. INTRODUCTION

Painting is a two-dimensional work of art that exists in visual form and has aesthetic value [1]. The purpose of making paintings varies depending on the background of the painter in making paintings, such as a form of socio-cultural expression, documentation of historical events, and communication of a moral message. Painting is also an art heritage that needs to be maintained. There are various paintings that have such high artistic value as the paintings of Raden Saleh, Affandi, Hendra Gunawan, and Nyoman Gunarsa.

The existence of paintings can be threatened if the painting security system is inadequate. Several well-known cases of painting theft include the theft of Affandi's paintings in 1973 and Raden Saleh's paintings in 2007 [2]. The theft of Affandi's own painting was revealed only in 2014 when Affandi's grandson saw news of Affandi's painting auction in Hong Kong [3]. Meanwhile, the theft of Raden Saleh's painting from the National Archives Building in Jakarta in 2007 was made into the film "Mencuri Raden Saleh" in 2022.

Apart from famous paintings, personal paintings that are no less valuable are also frequently stolen. This is of course troubling the public, especially private art collectors who have a limited security system. Apart from the need for preventive measures, it is also necessary to have an effective and efficient painting security system. It's a shame that painting security systems are rarely found because the majority still use human surveillance or conventional CCTV cameras which cannot give a warning when a painting is stolen. This of course can increase the potential for painting theft and cause losses.

Based on these problems, special handling is needed through "SI LUKAS" Painting Security System Using the TensorFlow-Based Object Detection Method, which is a system innovation that is able to monitor the whereabouts of paintings continuously and send notifications via Telegram and can be accessed from anywhere and anytime based on TensorFlow. With this system, it is hoped that it can assist private art collectors and museums in maintaining the security of paintings. In addition, the automatic notification feature via Telegram also aims to let users know if there is a threat or potential for theft of a painting, such as when someone is holding a painting. This of course can improve the painting's security system and make it easier for art collectors and museums to monitor paintings.

This system has various features including monitoring the existence of paintings which is done by taking photos of paintings based on digital image processing. When a person or object holding or approaching a painting is detected, the system will automatically take a photo and provide an automatic notification via Telegram. This system uses the Raspberry pi microcontroller as the main controller to run the system. The input component used is a camera that functions to capture images of paintings when an object approaches or holds the painting with a buzzer output as a warning alarm. The data in the form of a photo of the painting is then sent to the user's Telegram so that the tool can be accessed without any space and time limitations.

Previous research is one of the references in conducting research so that it can increase the theory used in reviewing the

research carried out. Several studies were appointed as references in increasing the study material in this study. The following is previous research in the form of several journals related to the research conducted.

The first research was carried out by Hardisal et al entitled "Designing a Smart Security Camera with an Image Processing Model Using a Raspberry Pi". In this study designing a camera to be smarter by using image processing mode using a Raspberry pi device so that it can control a location or place and provide warning reports via the WhatsApp application to a smartphone that has been set. In this system, every activity recorded by the camera will be periodically reported. This research process uses this cascade file and sees whether there are people or not [4]. In this study it is used as a reference for the image processing method with Raspberry pi.

In research conducted by Hamzida et al regarding "Optimizing CCTV Performance in Detecting Potential Environmental Security Disturbances Using the Image Comparing Method" resulted in testing the ability to detect threats in the camera coverage area and sending notifications via e-mail. This system is designed through the GUI in the Matlab program [5]. However, this study still cannot detect human objects and is only able to compare objects based on images in the database.

Research on other image processing is Sani et al's research on "Smart Security Systems Using OpenCV-Based Cameras". This system is assisted by Machine Learning in the form of OpenCV which is programmed on a Raspberry pi4 device which is connected directly to a camera to detect people or movements in areas under CCTV monitoring so that it can immediately provide information to users sent via Telegram in video form [6]. The working system of this research is to detect human presence which will later be recorded and sent to Telegram users as a security system. However, this system only focuses on detecting the presence or absence of humans, so the data sent on Telegram also has no concrete threat parameters.

The next study entitled "Computer Visio-Based Student Attribute Completeness Detection Prototype Design" by Prayitna et al. In this research, the object attributes of SMA 5 Tangerang were identified using the Tensorflow framework with a dataset of 700 images. The results showed that using the CNN method, an accuracy rate of 50-100% was obtained for detecting school attributes for SMA 5 Tangerang students in an image and video frame [7]. The references from this study are used as references and materials related to the Tensorflow framework to produce high accuracy with hundreds of image datasets.

In a study entitled "Implementation of Deep Learning Object Detection K3 Signs on Video Using the Convolutional Neural Network (CNN) Method with Tensorflow" by Kartikasari resulted in the detection of K3 Signs objects for Evacuation Routes and Fire Extinguishers using the Tensorflow framework and the SSD (Single Shot Multibox Detector) model.). The dataset used in this study is 1500 image data. The results of the K3 object detection research on video using the CNN method obtained an accuracy rate ranging from 50% -

97% [8]. This research is used as a reference for using the Tensorflow framework in detecting objects.

In another study, there was a motion detection system prototype on CCTV cameras with Frame Difference and Frame Subtraction Algorithms. The results of this study are quite capable of detecting objects and their movements, based on the results of 25 tests from different videos, the percentage of success reaches 76.0%. The conclusion is, the method used is quite successful in detecting the movement of objects in the video based on the differences obtained from reducing the previous frame and the next frame [9].

This research will carry out development based on some of the research above, namely by creating a Painting Security System (SI LUKAS) with the TensorFlow-Based Object Detection Method, where the Raspberry pi acts as the main controller. The advantage of this system is that it uses digital image processing techniques so that it saves data storage and the user does not need to be in front of the monitor continuously to monitor paintings.

TensorFlow, Google's open-source machine learning and deep learning framework, is a convenient and flexible tool for developing popular deep learning models [10]. TensorFlow also supports conventional machine learning. TensorFlow was originally designed for huge numerical computations, not deep learning. It accepts data in the form of multidimensional arrays with greater dimensions known as tensors. Multidimensional arrays are extremely useful for handling massive volumes of data. TensorFlow consists of several nodes that are associated with mathematical processes, and the nodes communicate with each other via edges [11]. TensorFlow may be used in both local and cloud environments [12].

Raspberry Pi is a multi-purpose tiny computer that can be programmed to meet your specific requirements; it may also be used for gaming, video and audio playing [13]. By adding a camera module to the Raspberry Pi, it will be able to visualize objects, places, and so on. The Raspberry Pi camera can be configured to evaluate these things as needed. Raspberry Pi is the most often utilized because of its simple operation and ease of handling [14]. Compared to other devices, the Raspberry Pi is less expensive. Different Raspberry Pi boards or versions are available; depending on the requirements, the board or version can be chosen. Furthermore, the quantity of hardware and sensors required is reduced, and an additional Ethernet shield is not required in Raspberry Pi. The Raspberry Pi is ideal for software applications and can run on modest power sources like battery packs. However, battery packs cannot be used due to high current ratings. Raspberry Pi boards can be programmed using open-source languages like Python. There are numerous programming libraries available for image processing with the Raspberry Pi camera, which can be utilized for a variety of purposes [15].

II. METHOD

This type of research is a type of development research, where this research develops research that has existed before. This research will later discuss the research design in the form of design, system design and application, materials and tools

used and the determination of the LUKAS SI system procedure (Painting Security System).

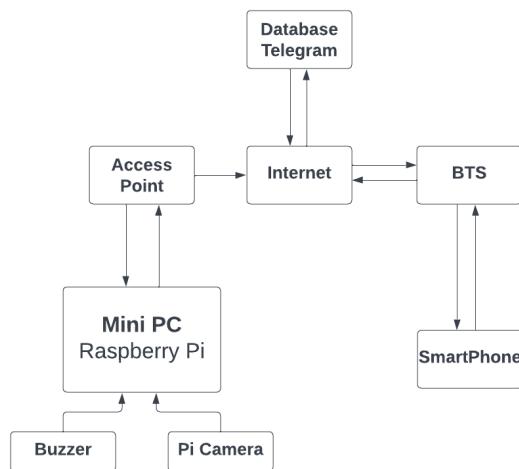


Figure 1. The system block diagram

Based on the system block diagram Fig. 1, it can be explained into several parts as follows:

1. Raspberry pi is used as the main controller which synchronizes all hardware components on the device and sends data to the Telegram ChatBot via the internet.
2. The camera used is the Pi Camera which functions to take pictures/images around the device.
3. The buzzer is used as an alarm when a painting is detected as being threatened (approached by a hand/object in a plot object).
4. The Telegram database functions to connect the tool with the Telegram ChatBot by storing data received from the Raspberry pi.
5. BTS is used for channeling signals to smartphones.
6. The smartphone is used as a data receiver from the Raspberry pi.

Mechanical design is made to describe the mechanical design that will be implemented in research. Fig. 2 is the mechanical design of the system along with the dimensions and descriptions that will be used in the study:

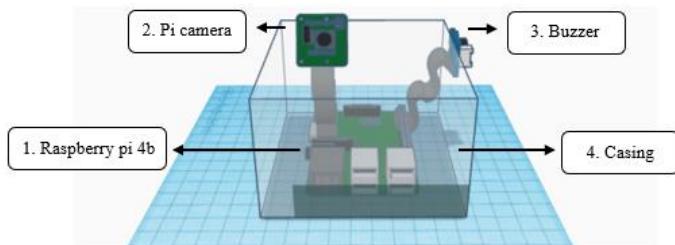


Figure 2. Mechanical design

Raspberry pi as the main controller is placed in a transparent box on the inside of the tool. The camera that functions to monitor the painting and hands is placed at the top.

The roof of the tool is designed with a hole which functions so that the flexibility of the Pi Camera can spread out.

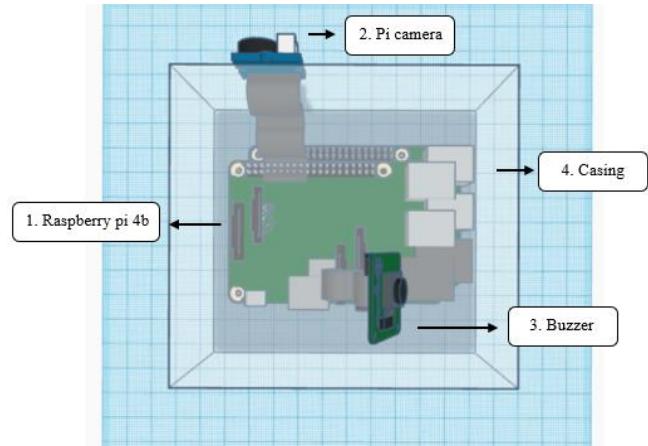


Figure 3. Mechanical design from above

The electrical design was made using the Fritzing application containing an arrangement of electrical components designed to become an electronic circuit system that is capable of running according to plan.

The Pi Camera is used as a camera to capture images/images around the device which will later be processed using digital image processing to detect the presence of cats. The Pi Camera used is the Raspberry pi Camera Board v1.3 with a 5MP resolution. The Pi Camera which has 15 pins is connected to the camera port on the Raspberry Pi using a flexible cable.

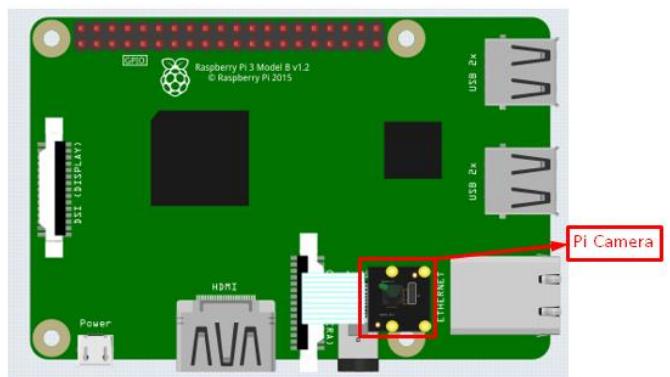


Figure 4. The circuit of pi cameras

Buzzer is used to sound the alarm. When the Pi Camera detects a hand object approaching the painting, the buzzer will sound. The buzzer used is Piezo Speaker. Piezo Speaker has 2 pins VCC (+) and GROUND (-) connected using jumper wires. For VCC (+) connected to the GPIO 27 leg then for GROUND (-) connected to the GROUND leg on the raspberry pi.

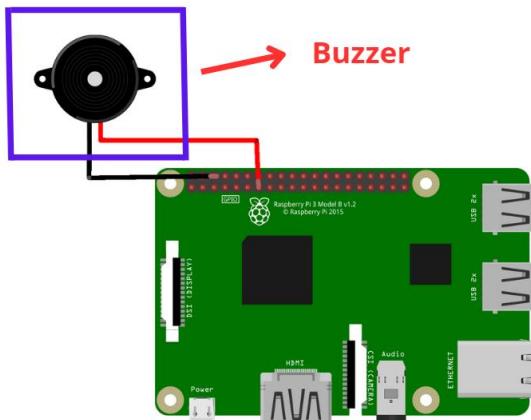


Figure 5. Buzzer circuit

After designing the individual component circuits, the next step is the overall electrical design which describes all the connections for each component.

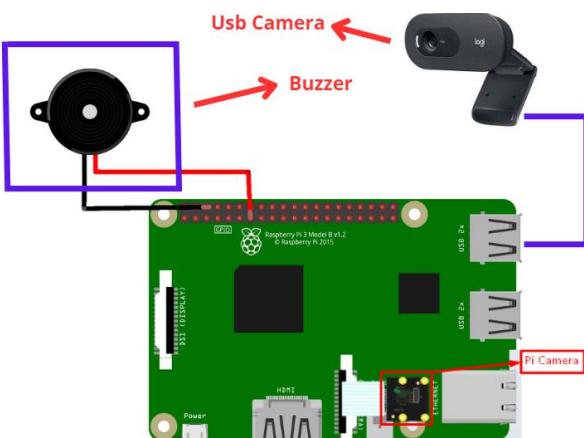


Figure 6. Overall electrical circuit

Based on the design above, there are 2 input components, namely the Pi Camera which is connected to the camera port with a flexible cable, the USB camera which is connected to the USB Port, then finally the Buzzer which is connected to GPIO 27 and the Raspberry pi ground.

Digital image processing in this study uses one of its applications, namely object detection. The programming language used is python 3 and uses the main libraries, namely opencv and tensorflow. OpenCV is an open source library developed by Intel which focuses on simplifying programming related to digital images, but in this study openCV is only used to call the Pi Camera to capture images. Tensorflow is a free and open source software library for machine learning, created by Google Brain which one of its functions is to train and run neural networks for image/object recognition purposes. In this study Tensorflow was used to process images captured by the Pi Camera and detect the presence of paintings and hands using the object detection method.

Object Detection is used to detect the presence of objects in the form of paintings and hands from the images captured by the Pi Camera. The models used are paintings and hands. This

model is a model that has been trained independently using Google Collabs so that it can run on Tensorflow Lite on Raspberry pi which has limited specifications well.

Object detection can be done specifically for certain objects, namely paintings and hands by filtering objects. So that the system will only respond when a predetermined object is detected.

The program on the Raspberry pi is a program that functions to run the system on the device. There are three programs, namely a program to detect the presence of hands and paintings, a program to sound a buzzer and a program to send images to Telegram. The whole program was made using the Visual Studio Code editor and can be seen in Appendix 1 and Appendix 2. The following is an explanation of each part of the program.

```

1 # Import packages
2 import os
3 import argparse
4 import cv2
5 import numpy as np
6 import sys
7 import time
8 import RPi.GPIO as GPIO
9 from time import sleep
10 GPIO.setwarnings(False)
11 GPIO.setmode(GPIO.BCM)
12 buzzer=27
13 GPIO.setup(buzzer,GPIO.OUT)
14 from datetime import datetime, timedelta
15 from threading import Thread
16 import importlib.util
17
18 import telegram
19

```

Figure 7. Import library

The image above shows the part of the program that functions to call all the libraries needed. The os library is a library that functions so that programs can interact with the operating system, cv2 or opencv functions to retrieve image data captured by the pi camera on /dev/video0/. Numpy (Numerical Python) is a Python library focused on scientific computing which has the ability to construct N-dimensional array objects, which are similar to lists in Python. Threading is a module that functions to separate parts of code and execute them in separate processes. tflite_runtime or Tensorflow Lite Runtime is a small part of the actual Tensorflow library where tflite only contains code to execute a python program, so it doesn't take up much CPU memory. The time and datetime libraries function to call the time variable.

```

88 # credentials bot telegram
89 bot_token = '6304916726:AAG-zqwU0zrxvrKoS5vP6TcWdtMG3m2Ce3Y'
90 chat_id = '740664873'
91
92

```

Figure 8. Telegram config

The Telegram Config variable contains data in json format which serves to configure the connection with the intended Telegram Chatbot.

```

24 class VideoStream:
25     """Camera object that controls video streaming from the Picamera"""
26     def __init__(self,resolution=(640,480),framerate=30):
27         #Initialize the PiCamera and the camera image stream
28         self.stream = cv2.VideoCapture(0)
29         ret = self.stream.set(3,resolution[0])
30         ret = self.stream.set(4,resolution[1])
31
32         # Read first frame from the stream
33         self.grabbed, self.frame = self.stream.read()
34
35         # Variable to control when the camera is stopped
36         self.stopped = False
37
38     def start(self):
39         # Start the thread that reads frames from the video stream
40         Thread(target=self.update,args=()).start()
41         return self
42
43     def update(self):
44         # Keep looping indefinitely until the thread is stopped
45         while True:
46             # If the camera is stopped, stop the thread
47             if self.stopped:
48                 # Close camera resources
49                 self.stream.release()
50
51

```

Figure 9. Videostream class

The VideoStream class above functions to stream video or take pictures. The image capture is carried out by the Pi Camera registered at /dev/Video0 which is listed in line 28. The videosStream process is carried out separately so that it can speed up image data processing using the command in line 41.

```

207 for i in range(len(scores)):
208     if ((scores[i] > min_conf_threshold) and (scores[i] <= 1.0)):
209
210         # Get bounding box coordinates and draw box
211         # Interpreter can return coordinates that are outside of image dimensions
212         ymin = int(max(1,(boxes[i][0] * imH)))
213         xmin = int(max(1,(boxes[i][1] * imW)))
214         ymax = int(min(imH,(boxes[i][2] * imH)))
215         xmax = int(min(imW,(boxes[i][3] * imW)))
216
217         object_name = labels[int(classes[i])] # Look up object name from "labels"
218         if object_name == "lukisan":
219             l_xmin, l_ymin, l_xmax, l_ymax = xmin, ymin, xmax, ymax
220         if object_name == "tangan":
221             t_xmin, t_xmax, t_ymin, t_ymax = xmin, ymin, xmax, ymax
222
223         # jika sudah melewati waktu interval capture / kirim gambar
224         if capture_time <= datetime.now():

```

Figure 10. Selection condition if object is detected

The program above shows the condition selection where if there is one of the many objects in the model with a score value greater than the minimum threshold specified at the beginning, the program will re-select the conditions where if the detected objects are "paintings" and "hands", the program will draws a bounding box around the object.

```

228 printFormat = "{0} : ({1}, {2}, {3}, {4})"
229 print(printFormat.format("lukisan", l_xmin, l_xmax, l_ymin, l_ymax))
230 print(printFormat.format("tangan", t_xmin, t_xmax, t_ymin, t_ymax))
231
232 # ambil dan simpan gambar ke local storage
233 timestamp = time.strftime("%Y-%m-%d %H:%M:%S")
234 photo_path = timestamp + '.jpg'
235
236 # ambil gambar
237 cv2.imwrite(photo_path, frame2)
238
239 #ambil gambar dengan kamera lain
240 capture.take_picture(photo_path)
241
242 # Kirim gambar ke telegram
243 caption = 'object collided on ' + timestamp
244 telegram.send_msg(bot_token, chat_id, photo_path, caption)
245 print("object collided")
246
247 # hapus gambar dari local storage
248 os.remove(photo_path)
249
250 # interval kirim capture / kirim gambar ke telegram
251 capture_time = datetime.now() + timedelta(seconds=second_delay_capture)
252
253

```

Figure 11. Command after successfully detected object

After the program has successfully detected the hand and painting, the next thing to do is take the image with the command on line 237 and name the file based on the current time with the command on line 233 and 234. The saved image will then be sent to Telegram storage with the command line 243 and so that image files do not accumulate on the Raspberry pi's internal storage, when the image is finished uploading, the image will be deleted with the command line 248.

```

1 import telepot
2
3 def send_msg(bot_token, chat_id, photo_path, caption):
4     bot = telepot.Bot(bot_token)
5
6     try:
7         with open(photo_path, 'rb') as photo:
8             bot.sendPhoto(chat_id=chat_id, photo=photo, caption=caption)
9             print('Photo sent successfully!')
10    except FileNotFoundError:
11        print('Photo file not found.')
12    except telepot.exception.TelegramError as e:
13        print('Error occurred while sending photo:', str(e))

```

Figure 12. OneSignal config

The program above is a program that works when Telegram successfully receives data from the raspberry pi and gives output or error notifications when the image fails to be sent.

```

253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278

```

Figure 13. Buzzer program

The program above is a program that functions to set the buzzer. In this program, if the system succeeds in detecting a hand approaching the painting, the buzzer will sound with a delay of 1.5 seconds. This function is found in lines 254-258. And the buzzer will turn off when it has passed the predetermined delay time.

III. RESULTS AND DISCUSSION

The results and discussion chapter is a chapter that contains the results and discussion of the implementation of the design planning that has been done in the research methods chapter. This chapter will explain the implementation of mechanical, electrical, software, and testing of several parameters. This test is needed to determine the specifications and capabilities of the system that has been made.

The following is an implementation of a mechanical design consisting of a front view, side view, top view, rear view and inside view.

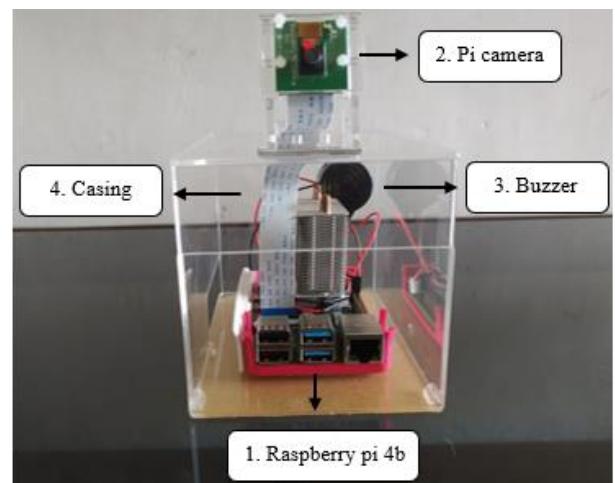


Figure 14. SI LUKAS prototype

The hinge on the back functions so that the top cover can be opened so that you can repair the tool if there are problems with the tool. The Pi Camera which is placed at a height of 12 cm with a slope of 30° has a coverage area of ± 3 meters in front of the device with a Horizontal field of view of 54° and a Vertical field of view of 41°. The roof is formed more elongated to the front with the aim of protecting the Raspberry Pi from dew and insects. The material used to make the mechanical system is acrylic which is resistant to rain and heat.

Below is the result of the implementation of the electrical design that has been made, namely a series of Pi Camera with Raspberry pi, Buzzer with Raspberry pi and USB Camera with Raspberry pi.



Figure 15. Raspberry pi series with pi camera



Figure 16. Buzzer circuit with raspberry pi



Figure 17. USB camera with raspberry pi



Figure 18. Overall system component series

Based on the figure above, it can be seen that there are 2 cameras. The first camera functions as an image processing manager, the second camera functions as a photographer from another angle and there is a DC IN Power Jack component for the 5V 3A adapter socket as a power source to run the Raspberry pi. The cable used to connect the Buzzer with the Raspberry pi is a jumper cable.

The results of the software implementation consist of the implementation of ChatBot Telegram, the process of learning/training datasets with Google Collabs and Object Detection using Tensorflow on painting and hands.

ChatBot Telegram is made using a Bot called Bot Father. Before using this bot must first be configured. For the Telegram Bot on the system created named SI LUKAS. Once configured the result is as follows.

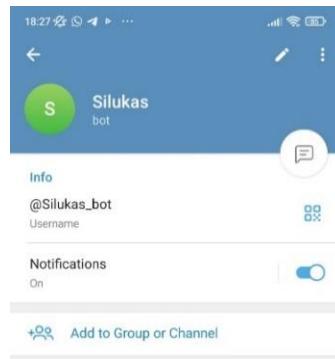


Figure 19. SI LUKAS profile page



Figure 20. Photo details page

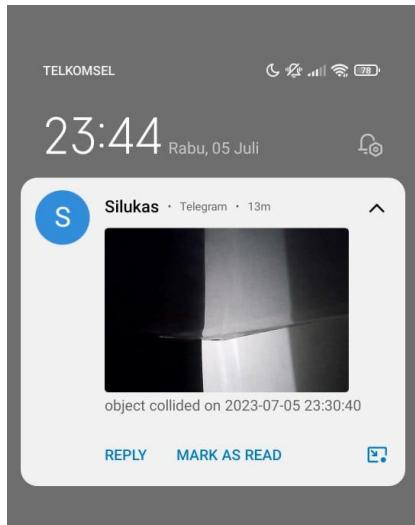


Figure 21. Notification



Figure 22. SI LUKAS gallery page

Based on the results of the software implementation, it can be seen that the SI LUKAS application has been synchronized with the Raspberry pi. Everything displayed in the application comes from the Raspberry pi in real time, so if there is a change in data, the application will automatically update the data to be displayed on the page. Because the application data is based on data on the Raspberry Pi, the use of the application can only be done when the smartphone is connected to the internet network.

In the SI LUKAS application there is a notification feature, namely when the hand is close to the painting. The notification will always appear even if the smartphone is not in use.

Next section will explain the results of implementing dataset learning/training using Google Collabs.

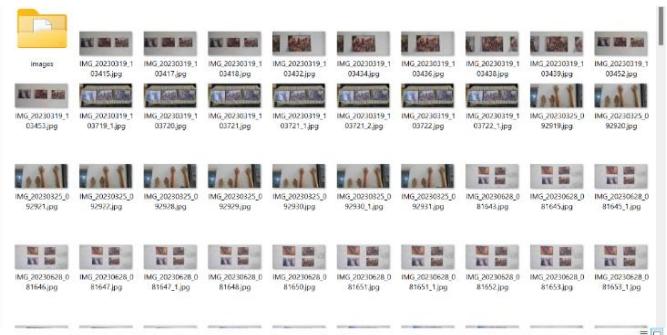


Figure 23. Dataset retrieval process

Before training the dataset, it is necessary to retrieve the dataset. In this experiment the system uses 230 images to produce a high level of dataset accuracy. The image format used is JPG.

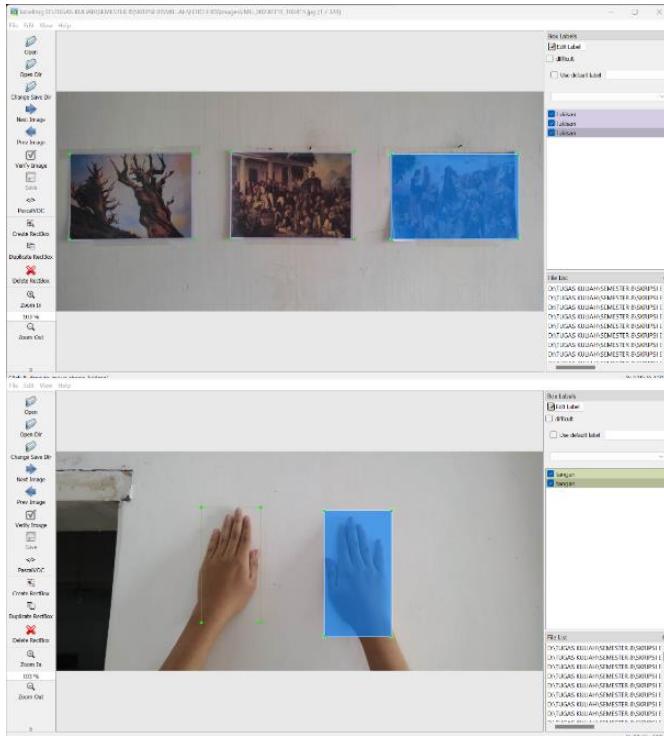


Figure 24. Dataset labeling process

After taking the picture, the next step is to label the image or dataset. The function of this labeling is to identify or distinguish between hand objects and painting objects.

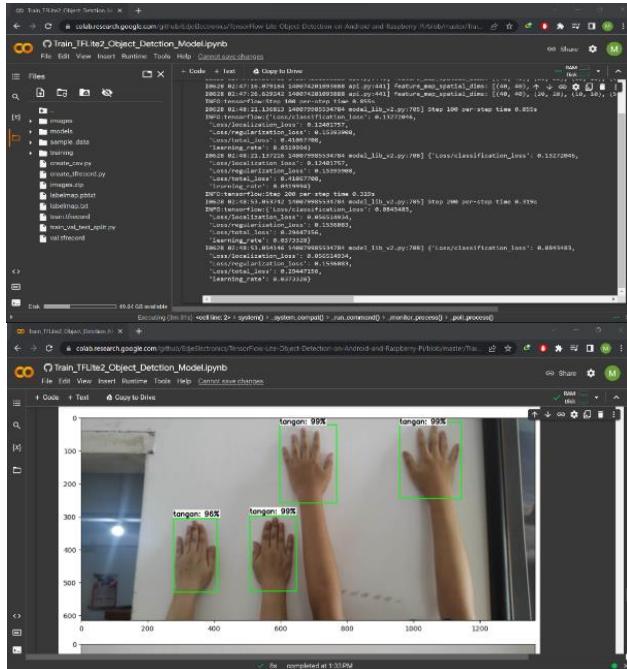


Figure 25. Dataset training process

After labeling the images, the next step is to train the images or datasets using Google Collabs. This training process takes around 2 – 4 hours depending on how many images or datasets are taken, the more images the longer the training process.

After the dataset training process, the Preview Dataset that was trained will appear as shown above. The picture above shows that the level of object detection accuracy is high, both in paintings and in hands.

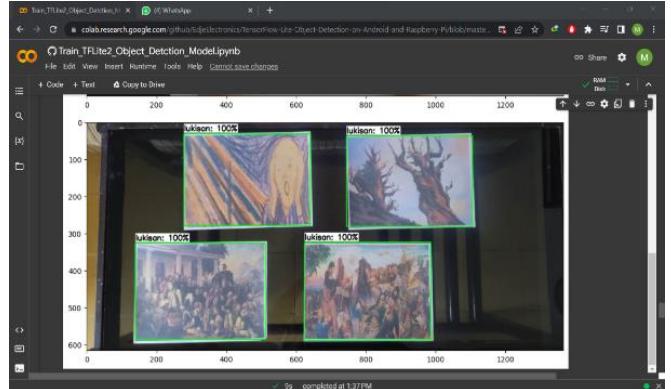


Figure 26. Dataset training results

TABLE I
CONVERSION POTENTIAL OF SOME RADIONUCLIDES

Distance (cm)	Success Rate (%)	Average Delay (s)
50	99	0.72
100	99	1,2
150	99	2,14
200	99	2,54
250	99	4,02
300	99	5,62
350	99	5,89
400	99	5,98
450	99	6,12
500	99	6,32
550	99	6,8
600	0	-

The table above is data obtained from testing the object readout delay at a distance using the Pi Camera Rev 1.3 camera and processed by Raspberry pi 4B. The test results show that the system can read objects with a success rate of 99% starting from a distance of 50 cm to 550 cm, while at a distance of more than 600 cm the success rate of reading is 0% because the Pi Camera's coverage area is 550 cm or 5.5 meters.

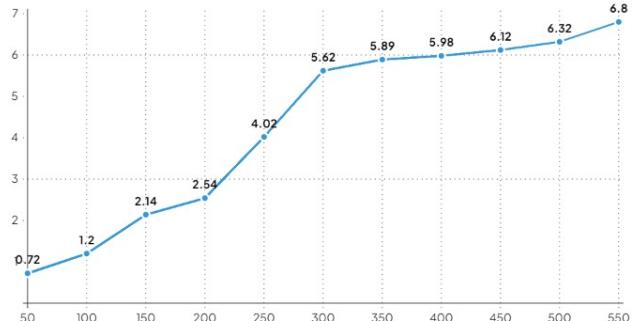


Figure 27. Graph of object reading delay testing against distance

Based on the data above, even though the system can detect all objects, namely hands and paintings from a distance of 50 cm to 550 cm, there is a difference in terms of delay, where at a longer distance the system takes longer to read. This is because objects that are at a greater distance have a smaller number of pixels, so the system takes longer to process the digital image data.

Object Detection testing based on camera angles aims to determine the quality or level of accuracy of the system in reading an object. This test was conducted to determine the accuracy of the system to see objects based on the camera's point of view. In this test there are several testing angles ranging from 30 degrees, 60 degrees, 90 degrees, 120 degrees and 150 degrees.

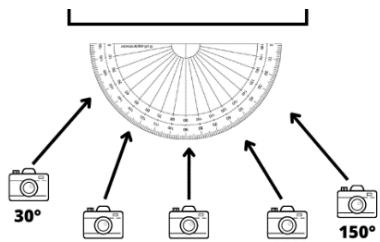


Figure 28. Illustration of camera viewpoint

The following are the steps for conducting the test.

1. Make sure that the Raspberry pi is connected to wifi internet.
2. Open the VNC viewer application to display the Raspberry Pi display.
3. After that place the camera at predetermined angles. In testing this time starting from an angle of 30 degrees to 150 degrees.
4. When finished, screenshot all the experimental results.
5. The results of the capture are then analyzed to determine the parameters being tested.

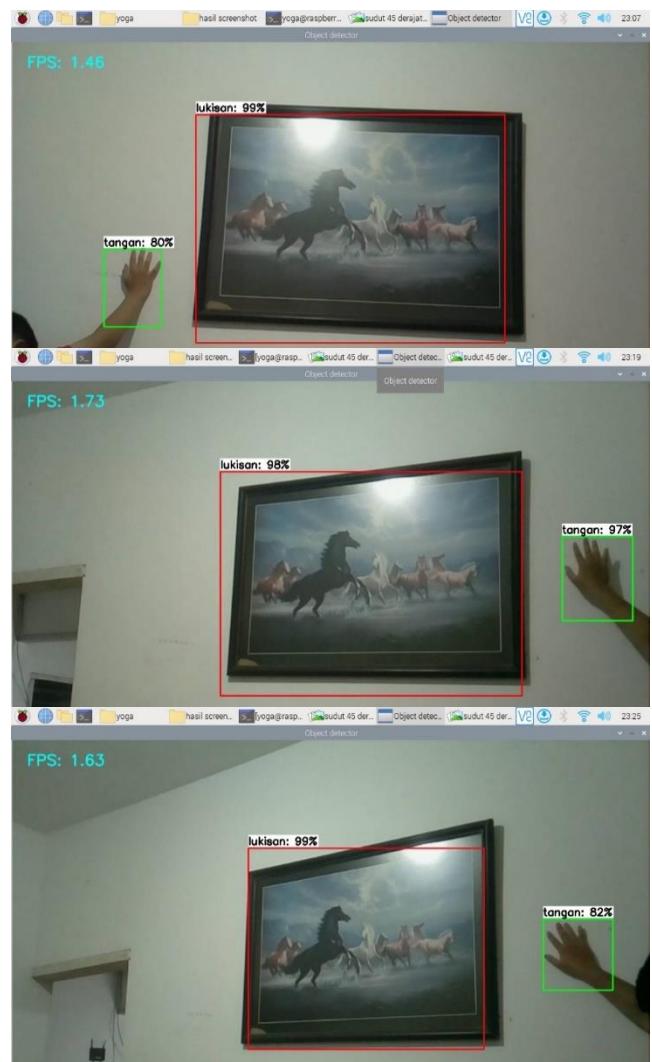
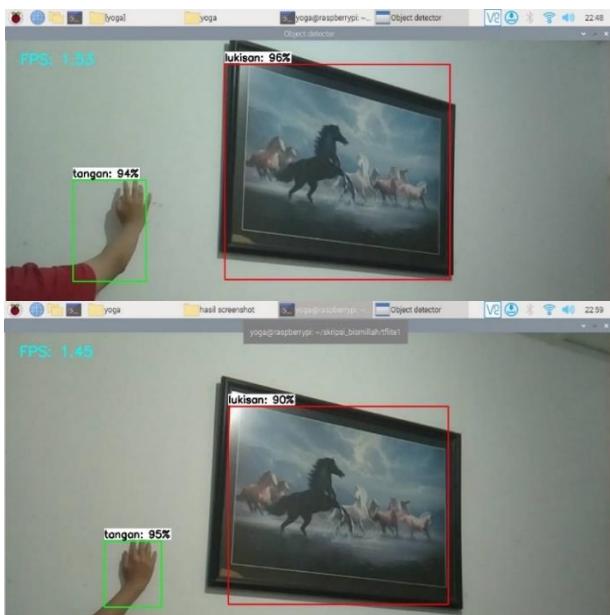


Figure 29. Pi camera viewpoint test results

Based on the Pi Camera angle test, the average accuracy of hand objects is 89.6% and the accuracy of painting objects is 96.4%, which is a very high category. In other words, the SI LUKAS system is feasible to run and implement.

IV. CONCLUSION

Based on the results of the research that has been done, the following conclusions can be drawn:

SI LUKAS: A Painting Security System with a TensorFlow Based Object Detection Method can be created using a Raspberry pi as the main controller, TensorFlow as a digital image processing library for Object Detection and a Telegram application for IoT monitoring. Based on testing the system delay in reading objects at long distances, it was found that the system could detect all objects, namely hands and paintings within a distance of 50 cm to 550 cm, but there was a difference in delay, where at a longer distance the system required more time. This is because objects that are further away have fewer pixels so the system takes longer to process the digital image data. Testing the average accuracy of detecting objects from

various angles, the average accuracy of hand objects was 89.6% and the accuracy of painting objects was 96.4% which is included in the very high category. In other words, the SI LUKAS system is feasible to run and implement.

REFERENCES

- [1] R. Aprianti, S. Sadono and C. R. Yuningsih, "Analysis of Aesthetic Values in Arya Sudrajat's Paintings in the NGINDEUW Exhibition," in *eProceedings of Art & Design*, vol. 8, no. 5, 2021.
- [2] A. Hamzah, "Identification of other creativity by Raden Saleh and Basoeki Abdullah in the paintings of modern artists," *Ars: Journal of Fine Arts and Design*, vol. 22, no. 3, pp. 125-138, 2019.
- [3] A. L. Ibrahim, and R. Dirkareshza, "Eradicating the Transnational Crime of Smuggling Cultural Heritage Through National Law and International Cooperation," *Justitia et Pax*, vol. 36, no. 1, 2020.
- [4] H. Hardisal, R. A. Candra, D. N. Ilham, and E. Sipahutar, "Design ff Smart Security Camera with Image Processing Model using Raspberry PI," *METHOMIKA: Journal of Management Informatics & Accounting Computerization*, vol. 3, no. 2, pp. 105-111, 2019.
- [5] N. K. Hamzida, and M. M. Parenreng, "Optimization of CCTV performance in detecting potential environmental security disturbances using the Image Comparing Method," *Journal of Electrical Technology*, vol. 17, no. 1, pp. 2656-0143, 2020.
- [6] A. Sani, R. Julianto, H. M. Maulidiah, and J. W. Wicaksana, "Smart Security System using OpenCV-Based Cameras," *Journal of Applied Electrical Engineering*, vol. 7, no. 1, pp. 42-47, 2023.
- [7] D. H. Prayitna, and A. Djajadi, "Design of Computer Vision-Based Student Completeness Detection Prototype," *J. Inov. Inform*, vol. 7, no. 1, pp. 57-69, 2022.
- [8] M.D. Kartikasari, "Implementation of Deep Learning Object Detection K3 Signs on Video using the Convolutional Neural Network (CNN) Method with Tensorflow (Case Study: Occupational Health and Safety (K3) Signs for Evacuation Routes and Fire Extinguishers in FMIPA UII Building)," UII, Indonesia, 2020.
- [9] R. A. Yuha, and M. Harahap, "Motion Detection on CCTV Cameras with Frame Difference and Frame Subtraction Algorithms," In *APTIKOM NATIONAL SEMINAR (SEMNSTIK) 20 R. A.*, 19 (pp. 503-511), 2019.
- [10] L. Yu, B. Li, and B. Jiao, "Research and Implementation of CNN Based on TensorFlow," *IOP Conf. Series: Materials Science and Engineering*, vol. 490, 2022.
- [11] S. Satyavolu and A. Bagubali, "Implementation of TensorFlow and Caffe Frameworks: in View of Application," *2019 International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN)*, Vellore, India, 2019, pp. 1-4.
- [12] P. Singh, and A. Manure, *Learn TensorFlow 2.0: Implement Machine Learning and Deep Learning Models with Python*, India: Apress, 2020.
- [13] I. G. B. Wirawan, I. G. P. W. Wedashwara, and A. Z. Mardiansyah, "Storefront Health Protocol IoT System Using Raspberry Pi Camera and Haar Cascade Classifier," *Journal of Computer Science and Informatics Engineering (J-Cosine)*, vol. 6, no. 1, pp. 30-38, 2022.
- [14] M. D. Mudaliar and N. Sivakumar, "IoT based real time energy monitoring system using Raspberry Pi," *Elsevier: Internet of Things*, 2020.
- [15] R. Kamath, M. Balachandra and S. Prabhu, "Raspberry Pi as Visual Sensor Nodes in Precision Agriculture: A Study," in *IEEE Access*, vol. 7, pp. 45110-45122, 2019.