Vol. 3, No. 1 | January - June 2020

A System Analysis Approach to Visual Object Tracking

Obaid Ahmed Khan[1], Ahmed Saeed[1]

Abstract:

In the modern-day study of system behavior and analysis, understanding the response of how the system will behave in a certain condition is considered vital. Thus, keeping object behavior as the primary objective and object detection as a core instrument this project intends to identify the object and track the motion of the desired object. This paper provides an overview of how a closed-loop control-based system was designed using a camera as visual input. The video signal is analyzed to detect distinct color and shape similarities to the desired object. Based on a high similarity index the Raspberry Pi controller generates a pulse width modulation signal to regulate the angular position of the rotary actuator to align the camera frame of view according to a desired object's position. Post system design experimentation results and potential realized system faults are expressed within the conclusion.

Keywords: *Computer Vision, Object tracking, closed-loop control system, System Analysis.*

1. Introduction:

Computer vision analysis of object characteristics and their behavior has always been a topic that has found a lot of success in the scientific community. The first studies date back all the way to the 1700s when electronic devices started getting more powerful. It was during this era the fundamental basics of how we can analyze objects were understood through applications in the field of medicine [1] [2].

While this was an amazing feat these techniques could not be applied towards the field of object detection due to the lack of computing power required for such a task. While the technology was not ready yet many techniques for detection of object characteristics were being studied [3]. Soon specialized computing devices were put into practice that allowed users to process images and apply pattern recognition techniques [4].

The implementation of these techniques meant that users can improve the previously existing techniques and algorithms to design better and more robust applications. One of these applications was the amalgamation of the control theory and computer vision. This led to many applications towards military, road control, and filmography [5] [6].

Application of this amalgamation between control theory and computer vision has been most prevalent in the field of automation. One such application of this technology in the automation is object tracking. Object tracking is the tracking of a specific object possessing desired characteristics using movable visual sensors. This concept of tracking objects through vision is not rather new with it being a crucial part of human development for years and with most infants having developed this ability soon after birth [7].

[1] DHA Suffa Univeristy Karachi, Pakistan
Corresponding Author: Obaid_10@live.com

2. State of the art

Research in the area of computer vision is quite dense with a lot of studies conducted towards efficient algorithm techniques. Some of these notable techniques can be consulted in [8]. This documentation analyzed the vast literature of computer vision and only the most well-established techniques were pondered on that have been physically realized. Based on an in-depth analysis of the literature it was found that there is a lack of quality of quantitative work towards the application of computer vision in the field of automation and control.

Some note-worthy efforts that have been made were either with a fixed viewpoint [9] or if the viewpoint was movable the system lacked an essential effort towards the incorporation of an efficient automation control strategy [10]. To overcome the issues plaguing object detection the system was designed with a control systems point of view.

3. Methodology

The ability to analyze objects visually and take actions accordingly are the features of smart controlling and when it is being developed for computing devices it needs to be developed in two steps. The first being object detection which involves the identification of the desired target and the second is object tracking, where the system reacts to the visual data.

A. Object Detection

The first step for object tracking is object detection or the ability to detect a specific object. In [11] the basic technique of object detection is considered using Color Threshold Method in which a basic array of RGB code is set, after which the code is compared with each individual pixel within the frame processing the image into binary setting white to the RGB code threshold and black to other code converting the image into a black and white image sending the required colors code to the processor.

This is further improved upon [12] by involving canny edge detection for the detection of edges by convolving a raw unprocessed image with a Gaussian filter resulting in a blurry image with no noticeable noise.

B. Object Tracking

To track the object, a basic control structure that would enable the controller to efficiently control the position of the system was designed. In this regard, an error reduction technique is used. Since the object tracking is based on error reduction so the error is the difference between the center of the screen frame and the center of the object. The setpoint i.e. object center was calculated using Hough transform [13] whereas, the process variable is the frame center which is calculated by dividing the screen resolution into half.

The error is calculated in terms of the pixels’ difference. So, in order to calculate the center of the screen measure the screen resolution, the screen used in this experiment is 720x1280p and divide the pixels into half to get the center of the screen which is (640,360).

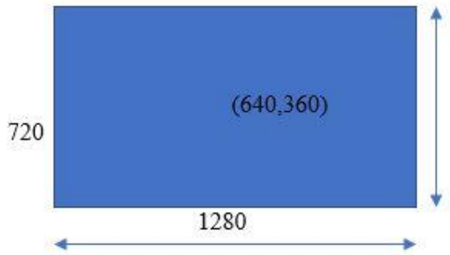

Fig 1: Frame of Reference

$$DutyCycle\% = \left((x * On\,Time) + bias\right) * Time\ interval$$

Where,
On-Time = Total pulse duration for servo movement between 0-180 degrees in ms.

bias = The minimum pulse duration required for the servo to become active in ms.

Time interval = Time in ms.

x = the calculated ratio between time intervals and pulse width for achieving the desired angle.

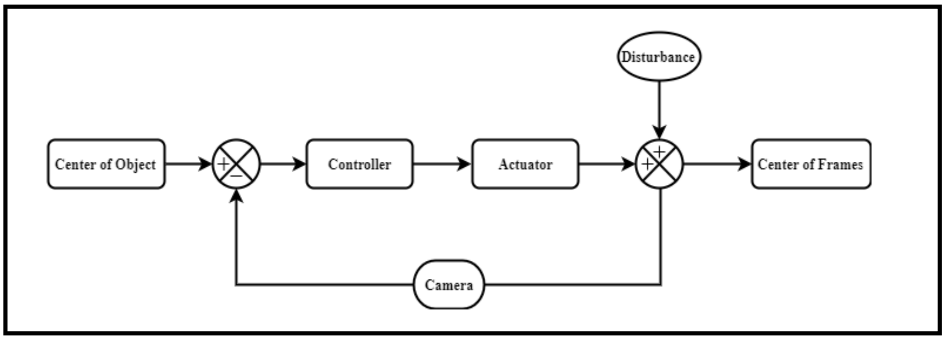

Fig 2: Block Diagram of Control System

4. System modeling

A. Control System

The basic control structure could not be directly employed without the system model. To understand the system, model the hardware specifications were initially explored.

Based on hardware specifications (Given in table 1) it was speculated that the most appropriate control structure for the system would be a proportional controller design.

Table 1: Servo motor specifications

Weight (g)	9
Torque (KG)	1.8
Speed (Sec/60deg)	0.1
Dead bandwidth	1 us
Temperature range	0 C – 55 C
Operating voltage	4.8 V
Stall torque	1.8kg/cm
Dimension	23 x 12.2 x 29mm
Gear type	POM gear set

To test the hypothesis of the desired control, structure a system response analysis was conducted:

- *System Response Analysis*

To record the input-output data, an excitation signal is sent through the system. Using this data, a plant model is built, and based on plant model accuracy the system model is tuned to achieve a quick and stable response. After analyzing system response data, it was realized that a second-order model was the most appropriate for the system. Exploring the previous hypothesis merits of each controller design were compared which led to the use of P controller, compared to PD and PID controllers the system response using a proportional controller is much faster which is a major requirement of the system design [14]

In P controller the P is proportional to the value generated of the error between setpoint and process variable.

P controller is used in the first-order system by optimizing the transient behavior by reducing the steady-state error of the system. The gain K is inversely proportional to the steady-state error, as one of them increases the other decreases and vice versa. This controller is only applicable when the steady-state error is in the applicable range.

B. System identification

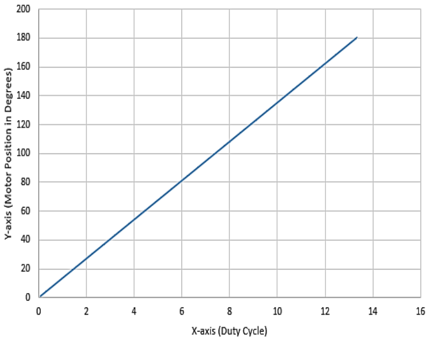

Fig 3: graph plotted between Duty cycle and Position

The slope of the graph measured is 13.53.

$$G(s) = \frac{\Delta y}{\Delta x}$$
$$\frac{d\theta}{dx} = 13.53$$

Where $y(x) = (position)$; $x = \%$ Duty cycle;
The Laplace transform equation of the system was modeled as,

$$\mathcal{L}[y(x)](s) = \frac{13.53\text{x(s)}}{s}$$

Assuming $y(0) = 0$ initially

$$y(s) = \frac{13.53x(s)}{s^2}$$
$$y(x) = c1 + 13.53x$$

5. Algorithm

The algorithm initiates the camera draws a new frame and scans the surrounding for the desired object. Depending on the yaw motor movement the algorithm decides whether to turn right or left. The program goes into tracking state when the desired object is the detected and after feature extraction proceeds with tracking of the object. The frame is updated in real-time tracking objects' current position.

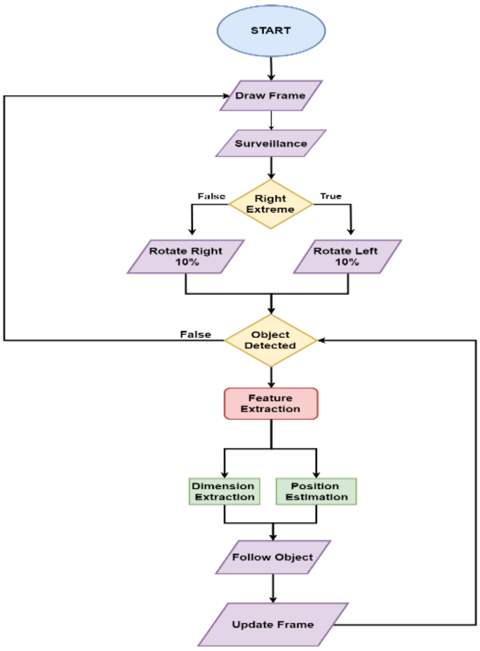

Fig 4: Algorithm flow chart

A. *Experimentation*

The experiment included the physical model implantation of the above mention algorithm and constructed a system consisting of a raspberry pi 3 model B and two high fidelity servo motors with a specified control structure. The first motor acted as the yaw rotational motor and the other acted as the pitch rotational motor. Then a webcam was installed on the top of the pitch rotational servo motor and a series of trials were conducted using an object.

B. *Calibration*

The object was placed first directly in front of the system to identify if any bias errors are present so that these can be calibrated out of the final working model. The system showed no presence of any bias error and would remain constantly stuck with the center of the view frame locked at the calculated cylinder center but with small bias.

C. *Sensitivity Analysis*

The object was then moved to identify whether the algorithm adapts to the input change, at small error values the system would quickly adapt to the change and reposition the view frame to the correct value with negligible error. The problem arose when the system was presented very high.

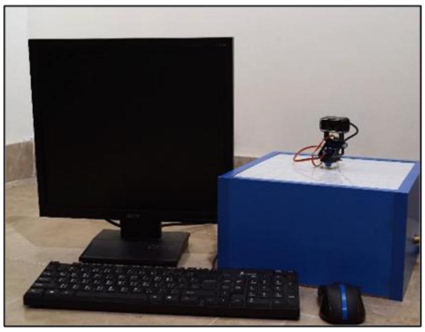

Fig 5: Hardware

6. Results

The object detection method proposed in [11] was further improved upon by using color thresholding in combination with canny edge detection to find a similarity between the camera image and the desired object. This technique led to the object being detected and tracked very accurately. Table 2 shows the optimal timing of the system:

Table 2: Optimal timing of the system

Timing		Time is taken
T 0	The system turns on and Scans the environment	0 - ∞
T1	Object detected by the system	20 milli-second
T2	Object Tracked with 5% threshold error	4.5 seconds
T3	Object tracked with high accuracy	1 second
T4	An object within steady-state	800 Milli-seconds
T5	System readjusts due to proportional control	1.5 seconds

*Note: Time between T0 and T1 could an infinite amount of time and depends on when the object is introduced in the system. Please see Figure 8 for an example of the object detected at 0.3 Seconds.

The methodology described in [11] does lend well to the rapid detection of the object. However, this approach lacks a fundamental feature in distinguishing between different objects with the same color code which results in false object detection. Additionally, the design methodology [12] based on canny edge detection focuses on the outer boundary of the object but it can only provide the approximation between similar size objects and fails to extract different features. Whereas, the present paper proposes the solution for object detection using a colour threshold in combination with the Canny edge detection methodology to bridge the flaws inherited with the two techniques. On the other hand, object tracking using the error reduction method is efficient and consumes low processing power. Although there are some minor fluctuations in tracking between error calculation and actual execution, it can be easily removed by introducing deadtime once the error is less than the desired accuracy.

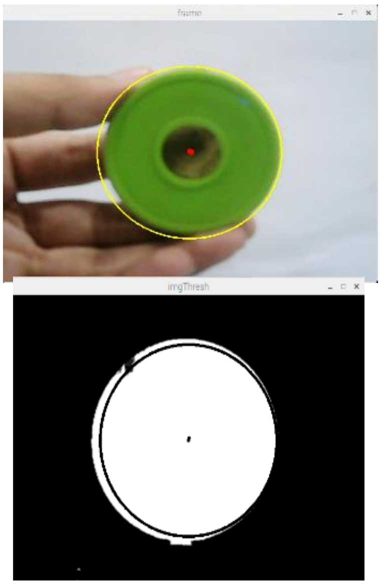

Fig 6: Object Detection &Tracking

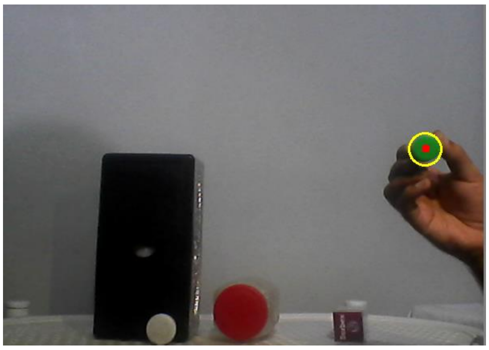

Fig 7: Testing with different objects

Figure 8 shows error plotting with respect to time which shows that error reduces as the camera approaches towards the desired set point.

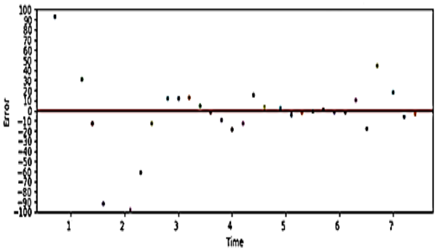

Fig 8: Error plotting

7. Conclusion & Future Work

The algorithm was then rewritten to include the change and based on further evaluation it was found that most of the sensitivity errors were resolved and the bias was also seemingly removed. The system now with the object even at the farthest end of the view frame could centralize the frame of view of the camera to the object's position in under 11 milliseconds and could track it smoothly.

A problem that was realized during the system implementation was that when the object was static the system would slightly move and reposition itself due to a refresh of the system cache.

This issue was fixed by inserting an additional condition in the algorithm when switched off the control actuation whenever the error accumulation of the last 5 values is lower than 0.5.

Some areas for further research are:

- Instead of involving the condition of switching off the control actuation, the error could be minimized further but without a severe increase in tracking time. This would require an effort towards optimization of the system by techniques such as branch and bound method.

- The control structure was based on a P controller with a 3-point model but in order to achieve the best results a higher polynomial data analysis is required. This would be investigated to further improve the system's performance.

References

[1] M. Forouzanfar, N. Forghani, and M. Teshnehlab, "Parameter optimization of improved fuzzy c-means clustering algorithm for brain MR image segmentation", Engineering Applications of Artificial Intelligence, vol. 23, no. 2, pp. 160-168, 2010. Available: 10.1016/j.engappai.2009.10.002..

[2] "A New Perspective of Wavelet-Based Image Denoising Using Different Wavelet Thresholding", International Journal of Science and Research (IJSR), vol. 5, no. 8, pp. 1504-1509, 2016. Available: 10.21275/art20161212.

[3] R. Jarvis, "Special feature: An interactive minicomputer laboratory for graphics, image processing, and pattern recognition", Computer, vol. 7, no. 10, pp. 49-60, 1974. Available: 10.1109/mc.1974.6323334.

[4] F.Schlag, A. C.Sanderson, C. P. Neuman, and F. C. Wimberly, "Implementation of Automatic Focusing Algorithms for a Computer Vision System with Camera Control", Robotics Institute, 1983. [Accessed 25 July 2019].

[5] E. Dickmanns and A. Zapp, "A Curvature-based Scheme for Improving Road Vehicle Guidance by Computer Vision", Mobile Robots I, 1987. Available: 10.1117/12.937795 [Accessed 25 July 2019].

[6] W. Miller, "Real-time application of neural networks for sensor-based control of robots with vision", IEEE Transactions on Systems, Man, and Cybernetics, vol. 19, no. 4, pp. 825-831, 1989. Available: 10.1109/21.35345 [Accessed 25 July 2019].

[7] M. Moore, R. Borton and B. Darby, "Visual tracking in young infants: Evidence for object identity or object permanence?", Journal of Experimental Child Psychology, vol. 25, no. 2, pp. 183-198, 1978. Available: 10.1016/0022-0965(78)90076-0 [Accessed 25 July 2019]

[8] R. Jain, R. Kasturi and B. Schunck, Machine vision. New York: McGraw-Hill, 1995.

[9] B. Shao and H. Xin, "A real-time computer vision assessment and control of thermal comfort for group-housed pigs", Computers and Electronics in Agriculture, vol. 62, no. 1, pp. 15-21, 2008. Available:

10.1016/j.compag.2007.09.006 [Accessed 24 July 2019].

[10] D. Lee and Y. Park, "Vision-based remote control system by motion detection and open finger counting," in IEEE Transactions on Consumer Electronics, vol. 55, no. 4, pp. 2308-2313, November 2009.

[11] S. Jayati Singh, "Object Detection by Color Threshold Method," *International Journal of Advanced Research in Computer and Communication Engineering,* vol. 4, no. 10, pp. 354-356, 2015

[12] J. Canny, "A Computational Approach to Edge Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vols. PAMI-8, no. 6, pp. 679 - 698, 1986

[13] M. Rizon and et al., "Object detection using circular Hough transform," *American Journal of Applied Sciences,* vol. 2, pp. 1606-1609, 2005.

[14] K Smriti Rao and Ravi Mishra, “Comparative study of P, PI and PID controller for speed control of VSI-fed induction motor” International Journal of Engineering Development and Research, vol. no 2, pp 2740-2744, IJEDR1402230, 2014.

[15] Onkar R. Kirpan and et al., “Object Detection on Raspberry Pi,” International Journal of Engineering Science and Computing, vol. 7, no.3, pp. 5360-5362, 2017

[16] Vijayalaxmi, et al, “OBJECT DETECTION AND TRACKING USING IMAGE PROCESSING,” Global Journal of Advanced Engineering Technologies, Special issue (CTCNSF), pp. 113-115, 2014.

[17] B. Dinesh, "Understanding and Design of an Arduino-based PID Controller,” Research Master Thesis, Physics and Applied Physics, *Virginia Commonwealth_Univ.,* VA., 2016

[18] Katsuhiko Ogata, “PID Controllers and Modified PID Controllers,” in Modern Control Engineering, Prentice Hall, fifth edition, pp. 567-647, 2010.

[19] D.Chris, “Github,” Github,Inc., 10 October 2015. [Online]. Available: https://github.com/MicrocontrollersAndMore/Raspberry_pi_2_and_OpenCV_3_Tutorial_Part_1. [Accessed 13 September 2017].

[20] R.Adrian, “pyimagesearch,” 14 September 2015. [Online].Available: https://ww.pyimagesearch.com/2015/09/14/ball-tracking-with-opencv/. [Accessed 17 October 2017].