Dissertation on

Detection and Measurement of Two-Dimensions of an Object using ArUco Markers

Thesis submitted towards partial fulfilment
of the requirements for the degree of

Master in Multimedia Development

Submitted by

SOUMAVA DAS

EXAMINATION ROLL NO.: **M4MMD24001**

UNIVERSITY REGISTRATION NO.: **163782 of 2022-23**

Under the guidance of

MR. JOYDEEP MUKHERJEE

School of Education Technology

Jadavpur University

Course affiliated to

Faculty of Engineering and Technology

Jadavpur University

Kolkata-700032

India

2024

Master in Multimedia Development
Course affiliated to
Faculty of Engineering and Technology
Jadavpur University
Kolkata, India

CERTIFICATE OF RECOMMENDATION

This is to certify that the thesis entitled **"Detection and Measurement of Two-Dimensions of an Object using ArUco Markers"** is a bonafide work carried out by SOUMAVA DAS under our supervision and guidance for partial fulfilment of the requirements for the degree of Master in Multimedia Development in School of Education Technology, during the academic session 2023-2024.

SUPERVISOR
School of Education Technology
Jadavpur University,
Kolkata-700 032

DIRECTOR
School of Education Technology
Jadavpur University,
Kolkata-700 032

DEAN - FISLM
Jadavpur University,
Kolkata-700 032

Master in Multimedia Development
Course affiliated to
Faculty of Engineering and Technology
Jadavpur University
Kolkata, India

CERTIFICATE OF APPROVAL

This foregoing thesis is hereby approved as a credible study of an engineering subject carried out and presented in a manner satisfactory to warranty its acceptance as a prerequisite to the degree for which it has been submitted. It is understood that by this approval the undersigned do not endorse or approve any statement made or opinion expressed or conclusion drawn therein but approve the thesis only for purpose for which it has been submitted.

Committee of final examination for evaluation of Thesis

** Only in case the thesis is approved.

DECLARATION OF ORIGINALITY AND COMPLIANCE OF ACADEMIC ETHICS

I hereby declare that this thesis contains literature survey and original research work by the undersigned candidate, as part of his **Master in Multimedia Development** studies.

All information in this document has been obtained and presented in accordance with academic rules and ethical conduct.

I also declare that, as required by this rule and conduct, I have fully cited and referenced all materials and results that are not original to this work.

NAME : SOUMAVA DAS

EXAMINATION ROLL NUMBER : M4MMD24001

THESIS TITLE : Detection and Measurement of Two-Dimensions of an Object using ArUco Markers

SIGNATURE: DATE:

Acknowledgement

I am deeply honoured to present this thesis to the School of Education Technology, Jadavpur University, Kolkata, in partial fulfilment of the requirements for the degree of Master in Multimedia Development.

First and foremost, I extend my heartfelt gratitude to my guide, Mr. Joydeep Mukherjee, for his vigilant supervision and unwavering encouragement. His invaluable feedback, insightful comments, and constructive criticism were instrumental in shaping this research work. Without his enthusiasm, guidance, and unique support, this thesis would not have reached fruition. I am profoundly grateful to Prof. (Dr.) Matangini Chattopadhyay, Director of the School of Education Technology, for her continuous support, encouragement, and timely advice throughout this journey. My sincere thanks also go to Dr. Saswati Mukherjee for her steadfast support during the entire course of this research. Their guidance and encouragement were both inspirational and motivating. I would also like to take this opportunity to thank all of my classmates of Master in Multimedia Development course who motivated me to complete my research work successfully. I do wish to thank all of our departmental support staff and all of those who were associated with this research contributed in some form or the other. Last but not the least, I would like to acknowledge and thank my mother for the strong, unwavering support, inspiration and encouragement that she has provided me. Her strength and belief in me were the foundation upon which this work was built. Without her, this accomplishment would not have been possible.

DATE:

SOUMAVA DAS
Examination Roll no: **M4MMD24001**
University Registration Number:
163782 of 2022-23
Master in Multimedia Development
School of Education Technology
Jadavpur University, Kolkata - 700032

Contents

Topic Name **Page No.**

List of Figures

List of Tables

Executive Summary

This thesis explores the implementation of Aruco markers for two-dimensional object dimension detection, addressing challenges related to size variations due to proximity and perspective changes. The study adopts a simplified yet robust approach, avoiding complex camera calibration methods. By utilizing a standardized marker as a reference, the proposed method ensures reliability and practicality across various fields.

ArUco markers are easily accessible and versatile, providing consistent accuracy in diverse environments. The research focuses on identifying objects against a homogeneous background and accurately calculating their two-dimensional dimensions in both pixels and centimetres. This approach enhances applications in automation, precise dimension assessment, and spatial relationship analysis, and aids in the inspection of industrial components. Additionally, it supports architects and engineers in visualizing and analysing structures.

Key contributions of this work include:

- Simplified, robust two-dimensional dimension detection methodology.
- Elimination of the need for complex camera calibration.
- Reliable and practical application in diverse fields.
- Accurate measurement capabilities in both pixel and centimetre units.

The findings pave the way for future innovations in two-dimensional dimension detection, offering a solid foundation for further advancements and more powerful applications. This work is poised to influence a range of industries requiring precise two-dimensional measurements, facilitating enhanced automation, inspection, and structural analysis.

Chapter 1:

1. INTRODUCTION

In the realm of automation, accurately measuring the dimensions of objects plays a pivotal role. This ability finds applications across diverse industries, from robotics and manufacturing to logistics and quality control. The field of computer vision [1] offers a powerful tool for achieving this objective, which is ArUco markers [2][3]. ArUco markers are visual tags resembling small, 2D barcodes. Each marker contains a unique identity encoded within a grid of black and white squares. This unique code allows computer vision algorithms to not only detect the presence of the marker but also precisely determine its pose [4][5] (location and orientation) within 3D space. Computer vision algorithms can easily find these markers in images and videos, even when they're partially hidden or at an angle. The true power of ArUco markers lies in their ability to reveal not just their presence, but also their exact position and direction in 3D space. This precise information is the key to accurately measuring the size of objects.

1.1 Object Detection

Object detection [6] is still a hot topic in academia and industry. Object detection, a crucial branch of computer vision, empowers machines to identify and locate specific objects within images or videos. It's like teaching a computer to see and understand the world in a similar way humans do. This technology utilizes sophisticated algorithms, [7] often powered by neural networks, to analyze digital images and pinpoint the presence of objects like cars, people, animals, or any other pre-defined category. Beyond simply locating objects, object detection can also determine their size, position, and even orientation within the image, providing valuable information for various applications. From self-driving cars navigating streets to robots performing tasks in factories, object

detection has become an essential tool for enabling machines to interact with the physical world in a meaningful way.

The importance of reliable and robust object detection will increase further. It has applications in many fields, e.g., medical sciences, maintenance identification of aircraft parts etc. Hence reliable and robust object detection is of high importance.

Furthermore, in remote sensing applications reliability and robustness are increasingly critical as deploying object detection on-board satellites is becoming a necessity. Having an updated mosaic of the Earth's surface with a frequency of once per hour and ground sample distance (GSD) of 5m would require over 1000 pet bytes of data per day [8].

Thus, it is required to have only metadata updates of object detection, pattern recognition and anomalies instead of full-scale raw data transmission. Sensor development has led the way to ground structures detection [8] and analysis from remote sensing satellite imagery data.

1.1.1 Mathematical Model for Object Detection

The mathematical model for object detection in computer vision typically involves several key components and equations. Here's an overview of the mathematical model often used in object detection algorithms like YOLO (You Only Look Once) [9]:

1. **Input Image Representation:**
    - Let II represent the input image.
    - I (x, y, c) I (x, y, c) represents the pixel value at coordinates (x, y) (x, y) in channel cc of the image.

2. **Convolutional Neural Network (CNN):**

   - The CNN processes the input image II through multiple layers, including convolutional layers, activation functions (e.g., ReLU), pooling layers, and fully connected layers.
   - The output of the CNN is a feature map.

3. **Bounding Box Prediction:**

   - For each grid cell in the feature map FF, the model predicts bounding boxes.
   - Let BB represent the number of bounding boxes predicted per grid cell.
   - For each bounding box bb, the model predicts:
     - Coordinates (x, y) (x, y) of the box's centre relative to the grid cell.
     - Width w and height h of the bounding box.
     - Confidence score Confidence(b)Confidence(b) indicating the likelihood of containing an object.
   - The confidence score is typically computed using logistic regression: Confidence(b)=σ (F(x′, y′,0)) Confidence(b)=σ(F(x′,y′,0)), where σ is the sigmoid function.

4. **Class Probability Prediction:**

   - For each bounding box bb, the model predicts class probabilities for different object categories.
   - Let CC represent the number of object classes.
   - The class probabilities for each class CC are computed using SoftMax activation.

5. **Non-Max Suppression (Post-processing):**

   - After predictions are made for all bounding boxes, non-max suppression is applied to remove redundant boxes with low confidence scores.
   - The algorithm selects the most confident bounding boxes for each object class based on their confidence scores and overlap with other boxes.

6. **Loss Function for Training:**

   - The model is trained using a combination of localization loss (e.g., bounding box regression loss) and classification loss (e.g., cross-entropy loss).
   - The total loss LL is typically defined as a weighted sum of the localization loss and classification loss.

This mathematical model forms the backbone of object detection algorithms like YOLO and serves as the basis for training and inference processes. It involves various computations such as convolution, activation functions, regression, SoftMax, and loss calculations to accurately detect and classify objects in images.

1.1.2 YOLO

YOLO [9], which stands for "You Only Look Once," is a powerful object detection system known for its speed and accuracy. Unlike earlier methods that involved multiple processing stages, YOLO achieves object detection in a single pass through a neural network. This streamlined approach makes it incredibly fast, processing images in real-time. YOLO also boasts impressive accuracy, making it a valuable tool for various applications. Its ability to detect multiple objects within an image and its adaptability to custom object classes further enhance its

versatility. With its speed and accuracy, YOLO has become a leading force in real-time object detection tasks across various fields.

At the core of YOLO's architecture is a convolution neural network (CNN) that divides the input image into a grid and predicts bounding boxes and class probabilities for objects within each grid cell simultaneously. By using anchor boxes and multi-scale detection, YOLO can accurately localize objects of varying sizes and aspect ratios within the image, enhancing its versatility and applicability across diverse scenarios.

YOLO has undergone several iterations, each bringing enhancements in speed, accuracy, and functionality. Versions like YOLOv2, YOLOv3, and YOLOv4 have introduced improvements,[10] such as feature pyramid networks, advanced tracking capabilities, and optimizations to handle challenging object detection scenarios.

The impact of YOLO extends across numerous domains, including surveillance, autonomous vehicles, industrial automation, and medical imaging. Its ability to deliver fast and accurate object detection has made it the go to choice for developers and researchers tackling complex computer vision tasks.

While YOLO has achieved remarkable success, ongoing research aims to address challenges like occlusions, small object detection, and crowded scenes. Future developments in YOLO and related algorithms are poised to further enhance real-time object detection capabilities and pave the way for more sophisticated applications in the realm of artificial intelligence and automation.

YOLO, or You Only Look Once, is a state-of-the-art real-time object detection algorithm that has gained significant popularity in the computer vision and machine learning communities.

1.2 ArUco Marker

The ArUco marker is one of the most popular squared fiducial markers [11] used for precise location acquisition.

ArUco is a widely-used library for augmented reality applications [12-14], particularly in the field of computer vision. Developed as a part of the OpenCV (Open-Source Computer Vision Library) project, ArUco specializes in marker-based tracking systems. These markers are essentially visual tags that can be printed and placed in the physical environment, allowing cameras or other imaging devices to detect and track their position and orientation [15]. ArUco markers are characterized by their unique patterns which facilitate fast and accurate identification. This library provides functionalities for both marker detection and pose estimation, making it a valuable tool in various industries ranging from robotics [16] and gaming to marketing and industrial automation [10]. Its simplicity, efficiency, and open-source nature have contributed to its widespread adoption and integration into numerous projects and applications worldwide.

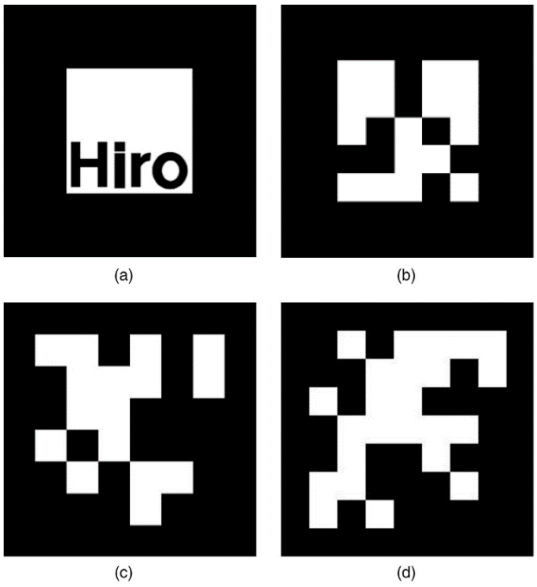

Fig. 1.1 Different types of Fiducial Markers [17]

Specialized computer vision tools can quickly identify ArUco markers due to their distinct patterns. Libraries like OpenCV offer readily available functions to make this process fast and easy. Alongside finding the ArUco marker, computer vision techniques can be used to recognize the object attached to it. Here a homogeneous background detector is used to detect objects with homogeneous background. Once the positions of both the object and the ArUco marker are determined, leveraging the known size of the ArUco marker (such as a 5x5 cm marker), the dimensions of the object can be calculated in real-world units with high precision. Few Fiducial markers are: (a) AR Toolkit, (b) ARTag, (c) April Tag, (d) ArUco, which are shown in Fig. 1.1.

1.2.1 Major challenges of ArUco marker

Working with ArUco markers presents several challenges that developers often encounter. One significant challenge is marker detection accuracy, especially in scenarios with poor lighting conditions, occlusions, or perspective distortions [17]. Achieving robust detection and precise pose estimation requires careful calibration of the camera and meticulous parameter tuning. Another challenge lies in marker occlusion and overlapping, which can confuse the detection algorithm and lead to erroneous results. Additionally, ensuring real-time performance [18], especially in applications requiring high frame rates, poses a computational challenge, demanding efficient algorithms and optimization techniques. Moreover, designing applications that can handle dynamic environments where markers may move or change positions adds complexity to marker tracking systems. Finally, integrating ArUco markers into larger augmented reality systems or applications may require addressing synchronization issues, user interface design considerations, and compatibility with other technologies, which can present further challenges. Overall, while

ArUco markers offer powerful capabilities for augmented reality, addressing these challenges is essential for achieving reliable and seamless marker-based tracking experiences.

This method of measuring object dimensions using ArUco markers and object detection brings significant benefits:

- **Accuracy**: ArUco markers provide highly precise location and dimension information, leading to very accurate calculations of object size.
- **Versatility:** This method can be applied to a wide variety of objects, regardless of their shape or size.
- **Efficiency:** Libraries like OpenCV simplify the process of finding and analyzing ArUco markers, making it easy to implement.

Using this technique offers benefits such as compatibility with diverse objects and mitigating issues arising from different camera angles and object distances.

1.3 Problem Statement

This study aims to develop a robust method for accurately measuring the two dimensions of objects in images. It seeks to overcome setup constraints and scalability issues in industrial applications, making the method adaptable to various conditions without extensive setup.

1.4 Assumptions

- **Object Characteristics**: It is assumed that the objects being detected and measured have distinguishable features and dimensions that can be captured accurately by the sensors or imaging devices used.
- **Object Background**: It is assumed that the background for the objects being detected are white or homogeneous, i.e., without having any major aberrations.
- **Environmental Conditions**: The proposed method assumes that the environment where the object detection and dimension measurement are performed is controlled to some extent, such as having consistent lighting and minimal obstructions.

1.5 Motivation of the Work

Recent literature reveals significant advancements in object detection and tracking through image and video analysis. Despite these advancements, many potential areas remain unexplored, underscoring the need for further research in this field. Accurate measurement of detected objects is particularly critical, yet current methods often impose stringent conditions that hinder practical applications. This gap motivates the focus of this work on improving object dimension measurement techniques to enhance practical usability.

1.6 Organization of the Thesis

The thesis is organized as follows:

- **Chapter 1: Introduction** – Presents the overview of the thesis.
- **Chapter 2: Literature Survey** - Summarizes the contributions of relevant papers, discussing recent works, their findings, and potential future research directions.
- **Chapter 3: Proposed Approach** - Introduces the proposed model, detailing its structure and methodology with illustrations.
- **Chapter 4: Experimentation and Results** - Presents the experimental procedures and results.
- **Chapter 5: Comparative Analysis** - Compares the proposed model with existing approaches.
- **Chapter 6: Conclusions and Future Scopes** - Summarizes the key findings and implications of the research, and discusses potential avenues for future research.

The thesis concludes with a comprehensive list of references.

Chapter 2:

2. LITERATURE SURVEY

Today, real-time object detection and dimensioning have emerged as critical challenges across various industries, driving significant advancements in computer vision. The landscape of object detection, tracking, and dimensioning has witnessed substantial evolution, with a plethora of research papers contributing to its growth. Initially considered a formidable task, object detection has undergone remarkable progress in recent literature. However, few of such papers which are relevant to this work are considered here and briefly discussed below.

Madhavi et. al. in their paper [19] have used computer vision for object detection. In their proposed work they have proposed the process for detection of object and measurement of dimension of the object also.

In this paper, they have proposed an approach that uses the video or image of the surroundings captured by the computer's webcam or external camera as an input to identify items, measure their dimensions, and identify them. With a stand, they kept the camera at a specific distance while also adjusting the camera's height, width, and depth. With the use of this system, they were able to identify many objects at once. They have found dimensional measurements of the objects and obtained other information such as the object's area of occupancy. It is shown in the paper that the accuracy for object detection and dimension measurement are of significant level. The accuracy of the dimension measurement is reported about 95% or more, most of the times. These dimensions are used in calculation of the area occupied by the object. Various applications of their work are reported as, e-commerce applications, Industrial use and AI Automobiles.

Mustafa M. Faisal et. al. in their paper [20] proposed an application-based module which may attempt to detect vehicles and persons on a road, to warn a driver. The main aim of their proposed model is to increase the safety for the autonomous car using two neural networks. The first stage is the enhancement of the dark road (night) and then, the network performs object detection. YOLO v3 network is proposed to predict the bounding box and the class of each object in the frame. YOLO v3 has the advantages of being fast and operates with high accuracy that can be utilized in the real-time applications. Second network is used to detect the distance values. This network will be used to predict the distance of the object from the autonomous car using the bounding box of YOLO and focal length of the camera. The final step is to draw the warning area, that depends on the steering angle. This is achieved using an input from a web camera mounted on dashboard of the car.

As suggested by the authors, the safety system may be improved by adding more cameras on each side of the car, so that the vision all around the car may be provided.

In some works, fiducial markers are predesigned patterns containing sufficient location and identification information that can be detected and recognized by preset algorithms. Each fiducial marker dictionary contains a certain number of fiducial markers, which perform camera pose estimation using specific feature points. The identity of each marker needs to be coded as an organized pattern and be decoded by effective and stable algorithms. ARToolkit6 offers a small

number of classical squared markers composed of external black borders and internal artificial patterns. As an early fiducial marker dictionary, it has many shortcomings such as restricted size of marker dictionary owing to the use of temporary internal patterns.

ArUco9 is a relatively complete fiducial marker system, including advantageous marker dictionary generation method, which can generate marker dictionaries of any encoding grid size, and a productive and robust algorithm for marker location and recognition.

In 2021, **Oguz Kedilioglu** et al, in their paper [21] presents a novel fiducial marker type called ArUcoE. It is obtained from a standard ArUco marker by enhancing it with a chessboard-like pattern. In their approach the pose estimation accuracy of any ArUco marker is claimed to be increased. In this paper further methods to increase the accuracy are analysed. By applying a sub pixel algorithm to the corner regions, they have located the corner points within a pixel and overcome the restriction of pixel-level accuracy. A deep- learning-based super-resolution method is used to artificially increase the pixel density in the same regions. Additionally, for this work, the effect of using a single and a stereo camera setup is required.

They have evaluated the methods in a simulation environment created with Blender and with real hardware. In this ArUco marker enhancement, the sub pixel and super-resolution approaches are used with a stereo setup instead of a mono setup. For estimation of accuracy, Camera specifications and distance from the marker are to be specified. Also, they have investigated sub pixel and super-resolution methods and the positive effect of using a stereo setup. According to them slightly better accuracy and much lower computational cost of the sub pixel algorithm suggests its superiority over the super-resolution method. As reported by them accurate detection of smaller marker is very important.

Martin Hirzer in their paper [12] presented a fast and robust marker detection front end inspired by the ARTag system. By using an edge-based approach is

presented where changing lighting conditions and occlusions are considered. Also, this algorithm is compared to ARToolKitPlus, a threshold-based marker detection system. The experiments revealed the advantages of the algorithm over threshold-based systems in cases of changing illumination and marker occlusion. As reported further improvements are needed and there is still room for future research. For example, if a marker is occluded by a very dark object this object might form a black-white edge with the bright background surrounding the marker. Thus, the algorithm cannot distinguish the object's edge from the marker's edges, and so the detection process, if it was not restricted, could find more than four corners for a single marker. But just restricting the maximum allowed number of corners per marker to four does not ensure that the four right corners are chosen

Their field of application ranges from industrial systems, where markers are designed to label parts or carry certain information, e.g. shipping data, to systems where markers are used for localization, e.g. Augmented Reality and robot navigation systems. Examples for the first case are Maxi-code, used by the US Postal Service, and Data Matrix and QR (Quick Response), used in industrial settings for the purpose of part labelling. Examples for the second case are ARToolKit, ARTag and ARSTudio, three systems for Augmented Reality. To reduce the sensitivity to lightning conditions and camera settings planar marker systems typically use bitonal markers. So, there is no need to identify shades of grey, and the decision made per pixel is reduced to a threshold decision. Furthermore, many marker systems use some of the marker's data bits to convey redundant information, which allows for error detection and correction. The design of the markers mainly depends on the application. Data Matrix, MaxiCode and QR are applicable for encoding information under controlled environments, e.g. conveyor belts, but are not very suitable for systems that use markers for

localization. The markers of these three systems are not designed for large fields of view and the perspective distortions involved. Furthermore, they require a large area in the image, so that the range at which these markers can be used is very limited. And finally, they do not provide enough points in the image to enable three-dimensional pose calculation.

Boxuan Li et al. in their work [17] presented a novel method to detect, recognize, and extract the location points of single ArUco marker based on convolutional neural networks (CNN). YOLOv3 and YOLOv4 networks are applied for end-to-end detection and recognition of ArUco markers under occlusion.

The proposed method predicts the locations of four corners by modifying the regression mechanism. Comparison experiments between the original method with four bounding box parameters are done and performance indicators were set. Experimental results show that the proposed innovative approach performed well in detecting and recognizing ArUco markers with high mAP and F1 scores over 0.9, both in integral and separate classes. It is reported in the paper that the prediction on corner position is also accurate with <3% average distance error for each corner and over 0.9 polygonal IoU between prediction and ground truth. Also, in this work, random corner coverage is done and validation of the robustness under the occlusion condition of the proposed method are completed. It is reported that, a point prediction error of <9% and IoU over 0.8 between the prediction and the ground truth are achieved.

Yutao Wang et al. in their paper [5], proposed an improved ArUco marker method to improve the robustness of distance measurement between the ArUco marker and the measured object in high noise and mechanical vibration environment.

As reported in the paper, ArUco markers can be used to measure the three-dimensional distance between the lens of a monocular camera and the target markers. The corner points of the marker, used for distance measurement by pose estimation are the intersection of the edges obtained by line fitting. The accuracy of straight-line fitting has a great influence on the accuracy of corner coordinates, which directly affects the stability of ranging. In the environment of high noise and high mechanical vibration, there is a problem of corner jitter. In this paper, an improved method of adding four feature circles on the diagonal of ArUco marker is proposed and the centres of the circles are fitted into a straight line by the least square method to modify the corners of ArUco marker. Experiments are performed by measuring the distance between the marker and the camera in depth and lateral directions respectively. When using the improved marker for ranging, the relative error is less than 1%, and the standard deviation is less than 0.3cm, which shows that the method improves the robustness and stability of measurement.

Four feature circles are added to the diagonal of ArUco marker, and the centre of the feature circles in ROI are obtained by calculating the moment. The corner points of ArUco marker are modified. The experimental results show that the relative error is less than 1%, the standard deviation is less than 0.3cm, and the measurement stability is obviously improved. Therefore, the improved marker method proposed in this paper can improve the ranging accuracy. It should be noted that the radius, position and number of characteristic circles in the experiments are given directly. The performance of measurement is affected by the measurement distance and the location of the marker in the image. Moreover, the measurement results may also be influenced by the proportion of markers in the whole image and the quality of image edge distortion correction. These factors should be considered in future research, and the

relationship between these parameters and the detection quality needs to be further explored.

Object tracking is one of the most important and fundamental disciplines of Computer Vision. Many Computer Vision applications require specific object tracking capabilities, including autonomous and smart vehicles, video surveillance, medical treatments, and many others. The OpenCV as one of the most popular libraries for Computer Vision, includes several hundred Computer Vision algorithms. Object tracking tasks in the library can be roughly clustered in single and multiple object trackers. The library is widely used for real-time applications, but there are a lot of unanswered questions such as when to use a specific tracker, how to evaluate its performance, and for what kind of objects will the tracker yield the best results.

Zhaleh Sadreddini et al. in their paper [22], a method based on a single-camera is proposed where vision techniques is used to measure the distance of objects by assuming camera is stationary. The aim is to implement a way to find the distance of an object in front of the imaging system in indoor environment. In this method, the first step is to extract the appropriate vertical floor line from the snapshots sequence. Moreover, the line passing through the base side of the object is found simultaneously with the floor line. Then, the part of the floor line starting from this point is measured in pixels, by taking the intersection point of these lines. Then it is converted to distance between the camera and the object.

This paper presents a successful implementation of a distance measurement method based on vision algorithms. This idea can be implemented with simple image processing techniques. The proposed method is designed for a single-camera condition. The distance is calculated in pixels. Experimental results are carried out to observe the relationship between real and measured distances.

Results of experiments prove that the proposed method has a good performance and it can be applied to real applications. The proposed method can be employed especially for flat indoor buildings with regular lines on the floor. This type of floors can be seen in many buildings such as schools, shopping centres, museums, hospitals and airports that have regular and striated surfaces.

Nada Dardagan et al. in their paper [23], implemented in OpenCV against the MOT20 dataset. The results are shown based on Multiple Object Tracking Accuracy (MOTA) and Multiple Object Tracking Precision (MOTP) metrics.

In this paper, they evaluated multiple object trackers available in the OpenCV library based on their duration, accuracy, and precision. They have Considered the threshold of 50 objects tracked at every frame of the four videos in the MOT20 dataset. The evaluation is based on the class of the object, the label of the object, and conducted based on the well-annotated ground truth bounding boxes provided in the dataset. It is reported that, MOSSE and Median flow have the potential to be used in real-time scenarios, where the number of tracked objects on a scene can exceed 100. According to the paper, other algorithms do not have the potential for real-time applications if the number of tracked objects on a scene exceeds 10. Also, their model provides significantly higher accuracy and success rates.

In the paper [24], published by **Ayushi Sharma** et al., important topics are discussed, such as, how artificial intelligence and machine learning algorithms help in object detection and how OpenCV becomes a useful tool. Also, in this paper, they have discussed how real time object identification and tracking can be done. It also shows the flexibility of a tracking system to a moving camera, ideal for automotive safety applications. Image identification makes use of techniques like detection of an object, its recognition, and segmentation. The

use of artificial intelligence and machine learning enhances the rate of processing the data and maintaining the standard of the outcome. It is fact that effects of artificial intelligence for future use in industry and humans are important as a whole. Hence, AI acts as the main operator of emerging technologies such as big data, robotics and IoT. Using machine learning and computer vision for detection and classification of different activities is obtained like, observing which direction the driver is looking when he operates the vehicle, how fast he is driving, which direction he is driving, locating the people that surround him, etc. Therefore, main goal is to prevent accidents by increasing efficiency.

Object detection is passing through many changes in algorithms, to improve performance both on speed and accuracy. By the continuous effort of so many researchers, deep learning algorithms are growing rapidly with an improved object detection performance. Various popular applications like pedestrian detection, medical imaging, robotics, self-driving cars, face detection, etc. reduces the efforts of humans in many areas. Due to the vast field and various state-of-the-art algorithms, it is a tedious task to cover all at once. **Pranav Adarsh** et al. in their paper [9] proposed a fundamental overview of object detection method by including two classes of object detectors. In two stage detector covered algorithms are RCNN, Fast RCNN, and Faster RCNN, whereas in one stage detector YOLO v1, v2, v3, and SSD are covered. Two stage detectors focus more on accuracy, whereas the primary concern of one stage detectors is speed. YOLO version called YOLO v3-Tiny, and then its comparison with previous methods for detection and recognition of object is described graphically in the paper.

Some of the popular object detection algorithms are Region-based Convolutional Neural Networks (RCNN), Faster-RCNN, Single Shot Detector (SSD) and You Only Look Once (YOLO). Amongst these, Faster-RCNN and SSD have better accuracy, while YOLO performs better when speed is given preference over accuracy. This algorithm presented in the paper [25] published by **Chandan G**, et al. is claimed to be efficient object detector without compromising on the performance.

It is shown in the work that, SSD gives results with considerable confidence level. Main Objective of SSD algorithm is to detect various objects in real time video sequence and track them in real time. This model showed excellent detection and tracking results on the object trained and can further utilized in specific scenarios to detect, track and respond to the particular targeted objects in the video surveillance. This real time analysis of the ecosystem can yield great results by enabling security, order and utility for any enterprise. Further extending the work to detect ammunition and, may be used in order to trigger alarm in case of terrorist attacks.

As per the report, the model can be deployed in CCTVs, drones and other surveillance devices to detect attacks on many places like schools, government offices and hospitals where arms are completely restricted.

Nashwan Adnan OTHMAN, et al. in their paper [26], suggested an object measurement technique for real-time video by utilizing OpenCV libraries. In this work, the suggested technique comprises of four stages: (1) identifying an object to be measured by using canny edge detection algorithm, (2) using morphological operators includes dilation and erosion algorithm to close gaps between edges, (3) find and sort contours, (4) measuring the dimensions of objects. In the implementation of the proposed technique, they designed a

system that used OpenCV software library, Raspberry Pi 3 and Raspberry Camera. It is reported that the proposed technique was nearly achieved 98% success in determining the size of the objects at some situations.

In this study, they have proposed a system to measure objects in a real time video and pictures. After the object has been detected by using canny edge detector, the size is obtained for each object by using OpenCV functions. They enhanced the canny edge detector algorithm through utilizing Morphological operations. This procedure benefits to eliminate extra noises. Furthermore, whereas eliminating the extra noises it likewise smoothens the shape and keeps the outline and size of each object. Thus, the outlines of the different objects in the scene were kept. The proposed technique works fast and five frames can be processed pending one second. Raspberry Pi 3 used to implement the systems since it has very great features and low-cost embedded equipment platform.

Amidst the extensive review of existing literature and methodologies, it becomes increasingly apparent that the field of object detection and dimensioning harbors numerous untapped avenues for exploration. This underscores the critical imperative for continuous research and innovation in this dynamic domain. Furthermore, the outcomes gleaned from current implementations unequivocally underscore the vast potential for future advancements.

In light of these findings, the selection of object detection and dimension measurement for research pursuit has been made. In this work, existing challenges are addressed with the aim of pushing the boundaries currently achievable.

Chapter 3:

3. PROPOSED APPROACH

In today's era, object detection and dimension measurement are of paramount importance. The proposed approach in this study demonstrates a high level of accuracy and operates independently of setup constraints. This method can be effectively utilized to address real-life problems and industrial applications. Additionally, it offers several advantages over existing methods. The approach is elaborated step-by-step in the following sections.

3.1 Image Acquisition:

3.1.1 Acquiring the Marker:

The selected marker, in this case 5x5 of size 50 mm, as shown in figure 3.1, is first printed on a piece of A4 size paper. This marker is used for all the experiments.

Fig. 3.1. A 5x5 ArUco Marker
(self-generated)

3.1.2 Acquiring the Image:

The images are acquired using a cell phone camera. On the A4 size paper, the objects are placed in the vicinity of the marker. Then the images are clicked while keeping both the marker and the objects in its frame.

3.2 Detection of the Marker:

3.2.1 Installing OpenCV in the environment:

The first step is installing and importing OpenCV in the working directory. The OpenCV module contains the sub-module '**aruco**' which contains various classes and functions needed for the detection of the marker.

3.2.2 Loading the Marker Detector:

ArUco markers come in a predefined dictionary in OpenCV. Just importing the cv2 module enables the use of 'aruco', a sub-module within OpenCV that provides functionalities related to ArUco markers. The 'aruco' module includes a '**DetectorParamete**r' class that helps in detecting the markers in images. It holds various parameters that can be tuned to adjust the behaviour of the ArUco marker detection algorithm. A function is contained within the 'aruco' module, which is used to obtain a predefined ArUco marker dictionary with a specified size and number of markers. The function takes arguments such as the dictionary type (**DICT_5X5_50** in this case), which determines the size and complexity of the markers in the dictionary. The dictionary contains unique marker patterns that the detector uses for identification and localization during marker detection tasks. Finally, an ArucoDetector object is created using the ArucoDetector class from 'aruco' module. This detector object is initialized with the previously obtained ArUco dictionary and the detector parameters. This detector will be used to detect ArUco markers in images or video streams using the specified dictionary and parameters.

3.3 Detection of the Object:

3.3.1 Converting Image to Grayscale:

The detector used here is a homogeneous background detector, that detects objects in plain background. The image is first converted BGR (Blue, Green, Red) colour space to grayscale. For certain operations like edge detection or grayscale analysis, it's beneficial to convert images to grayscale. Grayscale images have a single channel representing intensity, whereas BGR images have three channels for each colour component.

3.3.2 Image Segmentation for Edge Detection:

Following this conversion, an adaptive thresholding technique is applied, where the threshold value for binarization is dynamically determined based on local image characteristics. This adaptive thresholding operation utilizes parameters such as mean calculation for thresholding, inversion of the binary image for clarity, neighbourhood area size for threshold adaptiveness, and a constant for fine-tuning. The output is a binary mask where pixels above the determined threshold are set to white (255), while those below are set to black (0). Subsequently, the binary mask undergoes contour detection, which identifies continuous curves outlining objects or regions in the image. Specifically, the detector uses **cv2.RETR_EXTERNAL** to retrieve only external contours, excluding internal contours within others, and **cv2.CHAIN_APPROX_SIMPLE** for efficient approximation of contour shapes. Detected contours are stored in a list.

3.3.3 Loading the Object Detector:

For every contour, the program calculates its area in terms of pixels using the cv2.contourArea function. If a contour's area exceeds a specified threshold value, in this case, 2000 pixels, it is deemed a potential object of interest. This approach

allows the program to filter out contours that represent smaller or less significant elements in the image, focusing instead on larger and more relevant objects. The contours identified as potential objects are then stored for further analysis or processing.

3.4 Calculation of Object Dimensions:

3.4.1 Creating Bounding Box for Marker:

It first draws a polygon around the detected ArUco marker using the corners obtained from detection. Then it calculates the perimeter of the detected ArUco marker by using **cv2.arcLength** function

3.4.2 Calculating Pixel to CM Ratio:

The pixel to centimetre ratio is calculated, based on the perimeter of the ArUco marker. The actual size of the ArUco marker is known here, (e.g., 20 cm), and thus the ratio is calculated accordingly.

3.4.3 Finding Dimensions of Detected Objects:

After detecting objects in the image using a custom object detector, it iterates through each contour, drawing a polygon around each object and calculating its width and height in centimetres. These dimensions are annotated on the image, along with a bounding box around each object. The annotated image is displayed and saved for further analysis or visualization.

The flowchart for the above explained model is given in Fig. 3.2:

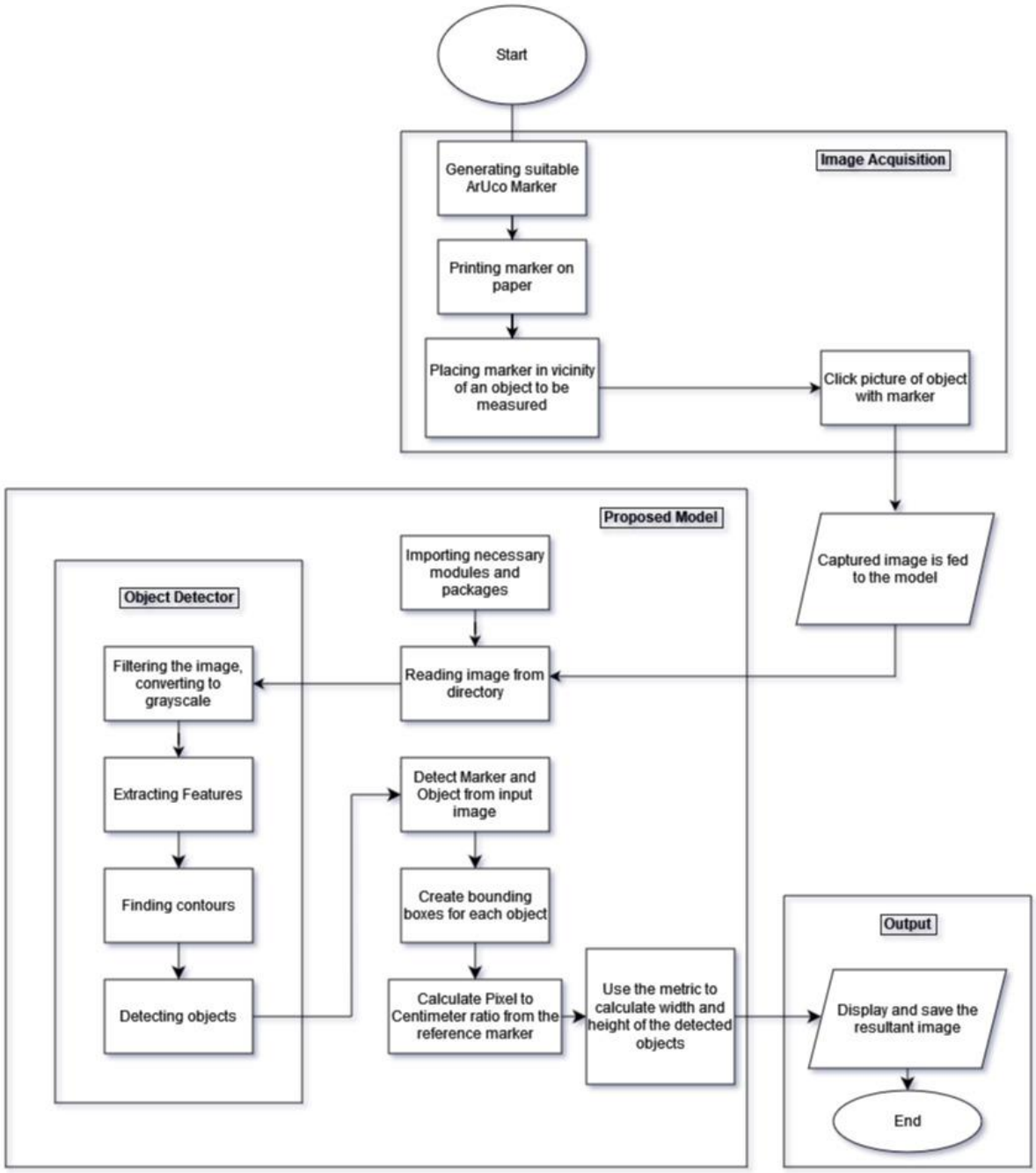

Fig. 3.2. Flowchart showing the flow of processes of the model

(self-made)

Chapter 4:

4. EXPERIMENTAL RESULTS

The proposed model was run with some self-made images, which are shown in the figures below.

4.1 Findings:

Input images and the corresponding output images for each object are provided below.

Fig 4.1. Input Image of a TV remote (self-made)

Fig 4.2. Output image showing dimensions (self-made)

Fig 4.3. Input Image of a TV remote (Inverted) (self-made)

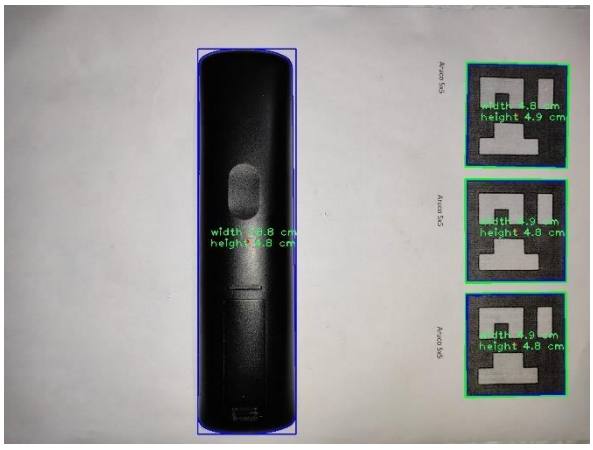

Fig 4.4. Output image showing dimensions (self-made)

Fig. 4.5. Input image of a duster

(self-made)

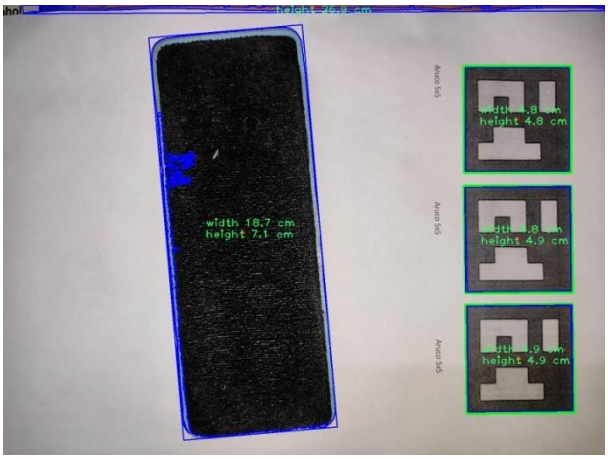

Fig 4.6. Output image showing dimensions

(self-made)

Fig. 4.7. Image of round objects

(self-made)

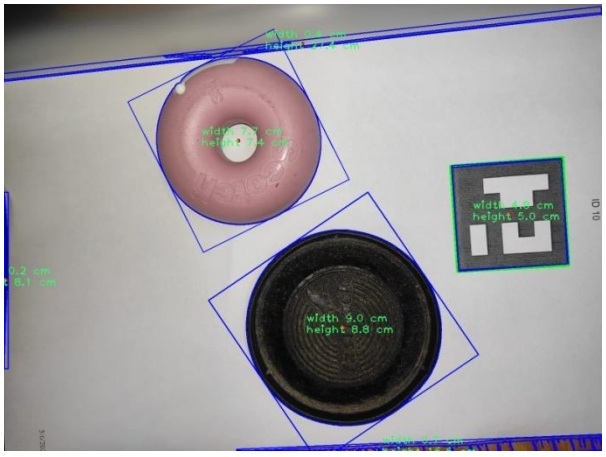

Fig. 4.8. Output image with dimensions

(self-made)

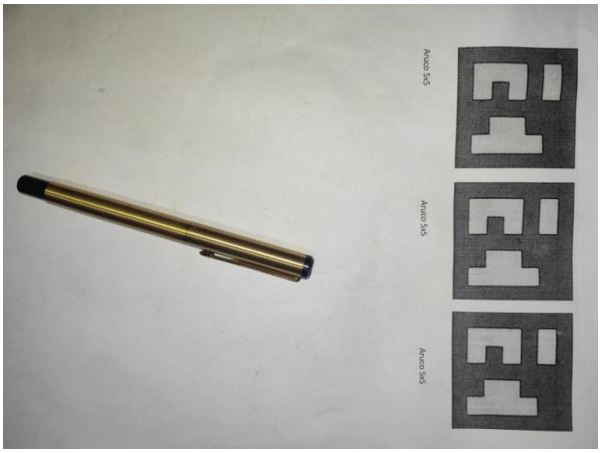

Fig. 4.9. Input Image of a pen

(self-made)

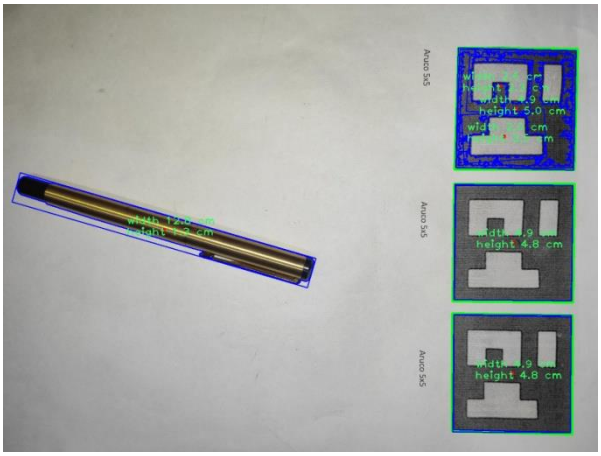

Fig. 4.10. Output showing dimension of pen

(self-made)

4.2 Accuracy:

The model gives a very accurate prediction of the object dimensions which has been analysed in the table below.

Table 4.1. Results obtained from testing the proposed model

Objects		**Real Dimensions (cm)**		**Predicted Dimensions (cm)**		**Accuracy (%)**
		Width	**Height**	**Width**	**Height**	
TV Remote (Up facing)		17.7	4.5	19.5	5.1	89.50
TV Remote (Down facing)		17.7	4.5	18.8	4.8	93.95
Duster		16.6	6.3	18.7	7.1	88.75
Round Objects (diameter)	1	7.5	7.5	7.7	7.4	98.03
	2	8.8	8.8	8.8	9.0	**98.89**
Pen		13.1	1.3	12.8	1.3	98.85

The results show that the model gives an average accuracy of ~95% on dimension detection.

The model showed accurate object measurements for all the objects. However, it has been noticed that the accuracy is better for objects having a flat surface rather than those with a rounded or uneven surface, as seen with the object in Fig. 4.1 and Fig. 4.3, when lying on its flat surface, the accuracy was 94%, while it dropped to 89.5% when set on its rounded surface. It is due to difference in lighting and shadows.

Chapter 5:

5. COMPARATIVE ANALYSIS

The work presented in this thesis offers several advantages compared to previous approaches on this topic. Firstly, this method eliminates constraints related to image orientation, object orientation, or camera position, significantly enhancing its practicality and usability. In addition to its increased flexibility, this approach achieves a high accuracy rate of over 94%, demonstrating that it not only removes constraints but also provides results on par with, if not superior to, previous methods.

In this study, two previous works on dimension detection of objects were analysed and compared side by side. Each method has its own strengths and weaknesses, which are summarized in the following sections and detailed in Table 5.1 and Table 5.2.

5.1 Model 1: Using a reference object

In [26] the proposed model uses a reference object for determining dimensions of other objects in the image. The reference object is always the rightmost object in the image, and its dimensions (width and height) is known beforehand. The results of this method are given in Table 5.1.

Table 5.1. Results as given in the paper for Model 1

Name of objects	AM-H (cm)	PM-H (cm)	AM-W (cm)	PM-W (cm)	Accuracy (%)
White glasses	10.0	10.5	6.8	7.1	95.45
Orange cup	8.5	9.0	7.5	7.4	96.56
Bottle	15.2	14.9	7.4	7.3	**98.23**
Potatoes	4.8	5.0	7.4	7.6	96.68

Using this technique the model achieved a highest accuracy of 98.23%. The features of the model are as follows:

- The reference object is always the left most object.
- The same reference object needs to be in all the pictures, or the dimension of the reference object have to be manually updated every time the reference object is changed.
- The background must be black, and using canny edge detection, the edges are detected.
- The measurement prediction has an average accuracy of 96.73%.

5.2 Model 2: Using a fixed camera

In the work done in [19], the model requires input from the user to predict the width and height of objects. The camera is fixed at a certain distance from the surface on which the objects are placed, using a stand. Using that information the model has some predefined parameters or metrics which allow the user to attain measurement of detected objects in the frame. The results of this method are given below in Table 5.2.

Table 5.2. Results as given in the paper for Model 2

Object	Actual Length (mm)	Actual Breadth (mm)	Actual Area (mm*mm)	Observed Length (mm)	Observed Breadth (mm)	Observed Area (mm*mm)	Accuracy (%)
Mobile	155	72	11160	153.29	70.29	10774.47	96.54
Pen	120	13	1560	119.10	13.09	1559.19	**99.95**
Book	275	164	45100	276.01	163.5	45127.63	99.94
Box	302	305	92110	294.50	310.1	91324.45	99.14
Wallet	91	73	6643	92.74	74.69	6926.75	95.72
Pencil Box	140	45	6300	141.1	45.0	6349.5	99.21
Adapter	34	47	1598	34.99	48.46	1695.61	93.89

The features of the model are discussed below:

- The width and height of the frame in view is first measured and given as input parameters to the model.
- White background is used in this work.
- The image undergoes Gaussian blurring and thresholding to find contours that help determine object outlines.
- Uses math libraries to determine dimensions and area.
- The model has an average accuracy of 97.77%.

5.3 Comparison of the Models:

The features of the 3 models under comparison is summarised below in Table 5.3.

Table 5.3. Summary of the comparison between the three models

Features	Model 1	Model 2	Model 3 (Proposed Model)
Background	Black	White	White
Reference Object	Left most object in the image	No reference object	ArUco marker
Camera Position	No restrictions	Restricted: fixed	No restrictions
Prerequisites for setup	▪ Leftmost object dimension must be known	▪ Frame width and height should be known in real world units	▪ Only marker size needs to be known
	▪ Same object must be present in all the images.	▪ All objects must come inside the frame defined by the camera.	▪ The marker needs to be printed only once and can be included in any image.
	▪ In case the reference object is changed in any image, the parameters need to be updated.	▪ If camera position is changed, the width and height must be updated accordingly.	▪ Gives same result irrespective of camera position. No need for changing reference marker.
Flexibility of usage	Highly restrictive as the reference object needs to be measured, and the same object needs to be present in all the images.	Highly restrictive in nature as the camera needs to be fixed and frame dimension needs to be specified. Slightest change in the setup can lead to erroneous measurement.	Highly flexible in nature as there is no restriction on the distance from camera, and the model can be used by anyone by just printing the same marker, without needing any calibration or changes in code.

From the Table 5.3, it evident that the proposed model is much more flexible and practical compared the existing models, for use in the real world.

The model maintains a level of accuracy that meets or exceeds the performance of more rigid and constrained models. This combination of high accuracy and operational flexibility positions the proposed model as a superior solution for object dimension detection in real-world applications.

Chapter 6:

6. CONCLUSION AND FUTURE SCOPES

This thesis explores the detection of two-dimensional measurements of objects from images using ArUco markers as reference points. This method offers significant advantages over previous approaches by eliminating many constraints related to image and object orientation and camera position while maintaining high accuracy (greater than 94%). The practicality of this method extends its applicability across various industries, including augmented and virtual reality, industrial component inspection, and architectural visualization. By incorporating markers into design models or physical structures, professionals can assess dimensions, proportions, and spatial relationships more effectively. Retailers can use these markers to showcase product dimensions online, aiding customers in making informed purchasing decisions based on size and scale. By enabling precise dimension measurements, this approach enhances the functionality and usability of image-based measurement systems, demonstrating its potential for widespread implementation in real-world applications.

FUTURE SCOPES

This research lays the groundwork for future advancements in object dimension detection. Several avenues for improvement and expansion exist:

1. **Enhanced Accuracy**: The accuracy of measurements can be improved through calibration of the camera and optimization of the environmental conditions. Fine-tuning these parameters can yield more precise results.
2. **Thickness Measurement**: Currently, the model measures only the maximum width or height of objects. Future work could involve using multiple cameras to capture different perspectives, allowing for the determination of object thickness and providing a more comprehensive three-dimensional measurement.
3. **Non-Geometrical Shapes**: The current model is limited to detecting the maximum dimensions of objects. Future improvements could enable the detection and measurement of irregular, non-geometrical shapes, expanding the model's applicability.
4. **Robust Object Detection**: Enhancing the object detection capabilities to perform accurately in noisy or cluttered backgrounds would make the model more versatile and reliable in real-world scenarios.
5. **Integration with Emerging Technologies**: As computer vision continues to evolve, incorporating new technologies and algorithms could further improve the accuracy and ease of object dimension detection. Advances in deep learning, improved image processing techniques, and more sophisticated hardware could all contribute to the refinement of this model.

By addressing these areas, further research work can expand on the foundation established in this thesis, leading to the development of more robust, accurate, and versatile object dimension detection systems.

REFERENCES

[1] Athanasios Voulodimos, Nikolaos Doulamis, Anastasios Doulamis, and Eftychios Protopapadakis: Deep Learning for Computer Vision: A Brief Review, Computational Intelligence and Neuroscience Volume 2018, DOI: https://doi.org/10.1155/2018/7068349

[2] Garrido-Jurado, S., Muñoz-Salinas, R., Madrid-Cuevas, F.J., Marín-Jiménez, M.J.: Auto- matic generation and detection of highly reliable fiducial markers under occlusion. Pattern Recognition. 47, 2280–2292 (2014). https://doi.org/10.1016/j.patcog.2014.01.005.

[3] Bacik, J., Durovsky, F., Fedor, P., Perdukova, D.: Autonomous flying with quadrocopter using fuzzy control and ArUco markers. Intel Serv Robotics. 10, 185–194 (2017).https://doi.org/10.1007/s11370-017-0219-8

[4] Anton Poroykov, Pavel Kalugin Sergey Shitov and Irina Lapitskaya: Modeling ArUco Markers Images for Accuracy Analysis of Their 3D Pose Estimation, CEUR-WS.org/vol-2744/short 14.pdf

[5] Yutao Wang, Zongpeng Zheng1, Zhiqi Su, Gang Yang1, Zhong Wang, Yu Luo: An Improved ArUco Marker for Monocular Vision Ranging, 978-1-7281-5855-6/20/$31.00 ©c 2020 IEEE

[6] Anelia Angelova, Shenghuo Zhu: Efficient object detection and segmentation for fine-grained recognition, CVPR 2013, Computer Vision Foundation, IEEE Xplore.

[7] Rob G.J. Wijnhoven, Peter H.N. de With, Fast Training of Object Detection using Stochastic Gradient Descent, 2010 International Conference on Pattern Recognition, 1051-4651/10 $26.00 © 2010 IEEE, DOI 10.1109/ICPR.2010.112

[8] Simon Wenkel, Khaled Alhazmi, Tanel Liiv, Saud Alrshoud and Martin Simon: Confidence Score: The Forgotten Dimension of Object Detection Performance Evaluation. Sensors 2021, 21, 4350. https://doi.org/10.3390/s21134350

[9] Pranav Adarsh, Pratibha Rathi, Manoj Kumar: YOLO v3-Tiny: Object Detection and Recognition using one stage improved model, 2020 6th International Conference

on Advanced Computing & Communication Systems (ICACCS), 978-1-7281-5197-7/20/$31.00 ©2020 IEEE

[10] Peiyuan Jiang, Daji Ergu*, Fangyao Liu, Ying Cai, Bo Ma: A Review of Yolo Algorithm Developments, The 8th International Conference on Information Technology and Quantitative Management (ITQM 2020 & 2021)

[11] Francisco J. Romero-Ramirez, Rafael Muñoz-Salinasa, Rafael Medina-Carnicer: Speeded up detection of squared fiducial markers, Image and Vision Computing 76 (2018) 38–47, 0262-8856/© 2018 Elsevier B.V. All rights reserved, https://doi.org/10.1016/j.imavis.2018.05.004

[12] Martin Hirzer: Marker Detection for Augmented Reality Applications, Technical Report ICG–TR–08/05, Graz, October 27, 2008.

[13] Xiang Zhang, Fronz S., Navab N.: Visual marker detection and decoding in AR systems: a comparative study. In: Proceedings. International Symposium on Mixed and Augmented Reality. pp. 97–106 (2002). https://doi.org/10.1109/ISMAR.2002.1115078.

[14] Malbezin P., Piekarski W., Thomas B.H.: Measuring ARTootKit accuracy in long distance tracking experiments. In: The First IEEE International Workshop Agumented Reality Toolkit, p. 2 pp.- (2002). https://doi.org/10.1109/ART.2002.1107000.

[15] A. Ujkani, E., Dybedal, J., Aalerud,A., Kaldestad, K.B., Hovland, G.: Visual Marker Guided Point Cloud Registration in a Large Multi- Sensor Industrial Robot Cell. In: 2018 14th IEEE/ASME International Conferenceon Mechatronicand Embedded Systems and Appli- cations (MESA). pp. 1–6 (2018). https://doi.org/10.1109/MESA.2018.8449195.

[16] Shabalina K., Sagitov A., Svinin, M., Magid, E.: Comparing Fiducial Markers Performance for a Task of a Humanoid Robot Self-calibration of Manipulators: A Pilot Experimental Study. In: Ronzhin, A., Rigoll, G., and Meshcheryakov, R. (eds.) Interactive Collaborative Robotics. pp. 249–258. Springer International Publishing, Cham (2018). https://doi.org/10.1007/978-3-319-99582-3_26

[17] Boxuan Li, Benfei Wang, Xiaojun Tan , Jiezhang Wu and Liangliang Weia: Corner location and recognition of single ArUco marker under occlusion based on YOLO algorithm, published online May 25, 2021; corrected Jun. 2, 2021. © 2021 SPIE and IS&T [DOI: 10.1117/1.JEI.30.3.033012]

[18] Abawi D.F., Bienwald J., Dorner R.:Accuracy in optical tracking with fiducial markers: an accuracy function for ARToolKit. In: Third IEEE and ACM International Symposium on Mixed and Augmented Reality. pp. 260–261 (2004). https://doi.org/10.1109/ISMAR.2004.

[19] Madhavi Karanam, Varun Kumar Kamani1, Vikas Kuchana, Gopal Krishna Reddy, Koppula, and Gautham Gongada: Object and it's dimension detection in real time, E3S Web of Conferences 391, 01016 (2023), ICMED-ICMPC 2023, https://doi.org/10.1051/e3sconf/202339101016

[20] Mustafa M. Faisal, Mohammad S. Mohammed, Ali M. Abduljabar: Object Detection and Distance Measurement using AI, 2021 14th International Conference on Developments in eSystems Engineering (DeSE), 978-1-6654-0888-2/21/$31.00 ©2021 IEEE

[21] Oguz Kedilioglu1, Tomas Marcelo Bocco, Martin Landesberger, Alessandro Rizzo and Jorg, Franke1: ArUcoE: Enhanced ArUco Marker, 2021 The 21st International Conference on Control, Automation and Systems (ICCAS 2021) Ramada Plaza Hotel, Jeju, Korea, Oct. 12～15, 2021

[22] Zhaleh Sadreddini, Turul Çavdar, and Hossein Barghi Jond: A Distance Measurement Method Using Single Camera for Indoor Environments, 978-1-5090-1288-6/16/$31.00 ©2016 IEEE

[23] Nada Dardagan, Adnan Brdanin, Džemil Džigal, Amila Akagic: Multiple Object Trackers in OpenCV: A Benchmark, 978-1-7281-9023-5/21/$31.00 © 2021 IEEE

[24] Ayushi Sharma, Jyotsna Pathak, Muskan Prakash: Object Detection using OpenCV and Python, 2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN), ISBN: 978-1-6654-3811-7/21/$31.00 ©2021 IEEE

[25] Chandan G, Ayush Jain, Harsh Jain, Mohana, Real Time Object Detection and Tracking Using Deep Learning and OpenCV, Proceedings of the International Conference on Inventive Research in Computing Applications (ICIRCA 2018) IEEE Xplore Compliant Part Number: CFP18N67-ART; ISBN:978-1-5386-2456-2

[26] Nashwan Adnan OTHMAN, Mehmet Umut SALUR, Mehmet KARAKOSE and Ilhan AYDIN: An Embedded Real-Time Object Detection and Measurement of its Size, 978-1-5386-6878-8/18/$31.00 ©2018 IEEE

APPENDIX

Environment used: **Jupyter Notebook**
Python version: **3.12.1**
Libraries used:

- **OpenCV**
- **OS**
- **Numpy**
- **Imutils**

Sample code: (Python 3)

Leftmost Reference Object Detector method:

```
# import the necessary packages
from scipy.spatial import distance as dist
from imutils import perspective
from imutils import contours
import numpy as np
import argparse
import imutils
import cv2
def midpoint(ptA, ptB):
	return ((ptA[0] + ptB[0]) * 0.5, (ptA[1] + ptB[1]) * 0.5)
# construct the argument parse and parse the arguments
ap = argparse.ArgumentParser()
ap.add_argument("-i", "--image", required=True,
	help="path to the input image")
ap.add_argument("-w", "--width", type=float, required=True,
	help="width of the left-most object in the image (in inches)")
args = vars(ap.parse_args())
# load the image, convert it to grayscale, and blur it slightly
image = cv2.imread(args["image"])
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
gray = cv2.GaussianBlur(gray, (7, 7), 0)
# perform edge detection, then perform a dilation + erosion to
# close gaps in between object edges
```

```
edged = cv2.Canny(gray, 50, 100)
edged = cv2.dilate(edged, None, iterations=1)
edged = cv2.erode(edged, None, iterations=1)
# find contours in the edge map
cnts = cv2.findContours(edged.copy(), cv2.RETR_EXTERNAL,
	cv2.CHAIN_APPROX_SIMPLE)
cnts = imutils.grab_contours(cnts)
# sort the contours from left-to-right and initialize the
# 'pixels per metric' calibration variable
(cnts, _) = contours.sort_contours(cnts)
pixelsPerMetric = None
# loop over the contours individually
for c in cnts:
	# if the contour is not sufficiently large, ignore it
	if cv2.contourArea(c) < 100:
		continue
	# compute the rotated bounding box of the contour
	orig = image.copy()
	box = cv2.minAreaRect(c)
	box = cv2.cv.BoxPoints(box) if imutils.is_cv2() else cv2.boxPoints(box)
	box = np.array(box, dtype="int")
	# order the points in the contour such that they appear
	# in top-left, top-right, bottom-right, and bottom-left
	# order, then draw the outline of the rotated bounding
	# box
	box = perspective.order_points(box)
	cv2.drawContours(orig, [box.astype("int")], -1, (0, 255, 0), 2)
	# loop over the original points and draw them
	for (x, y) in box:
		cv2.circle(orig, (int(x), int(y)), 5, (0, 0, 255), -1)
			# unpack the ordered bounding box, then compute the
midpoint
	# between the top-left and top-right coordinates, followed by
	# the midpoint between bottom-left and bottom-right coordinates
	(tl, tr, br, bl) = box
	(tltrX, tltrY) = midpoint(tl, tr)
	(blbrX, blbrY) = midpoint(bl, br)
	# compute the midpoint between the top-left and top-right points,
	# followed by the midpoint between the top-righ and bottom-right
	(tlblX, tlblY) = midpoint(tl, bl)
```

```
(trbrX, trbrY) = midpoint(tr, br)
# draw the midpoints on the image
cv2.circle(orig, (int(tltrX), int(tltrY)), 5, (255, 0, 0), -1)
cv2.circle(orig, (int(blbrX), int(blbrY)), 5, (255, 0, 0), -1)
cv2.circle(orig, (int(tlblX), int(tlblY)), 5, (255, 0, 0), -1)
cv2.circle(orig, (int(trbrX), int(trbrY)), 5, (255, 0, 0), -1)
# draw lines between the midpoints
cv2.line(orig, (int(tltrX), int(tltrY)), (int(blbrX), int(blbrY)),
        (255, 0, 255), 2)
cv2.line(orig, (int(tlblX), int(tlblY)), (int(trbrX), int(trbrY)),
        (255, 0, 255), 2)

        # compute the Euclidean distance between the midpoints
dA = dist.euclidean((tltrX, tltrY), (blbrX, blbrY))
dB = dist.euclidean((tlblX, tlblY), (trbrX, trbrY))
# if the pixels per metric has not been initialized, then
# compute it as the ratio of pixels to supplied metric

if pixelsPerMetric is None:
        pixelsPerMetric = dB / args["width"]
# compute the size of the object
dimA = dA / pixelsPerMetric
dimB = dB / pixelsPerMetric
# draw the object sizes on the image
cv2.putText(orig, "{:.1f}in".format(dimA),
        (int(tltrX - 15), int(tltrY - 10)), cv2.FONT_HERSHEY_SIMPLEX,
        0.65, (255, 255, 255), 2)
cv2.putText(orig, "{:.1f}in".format(dimB),
        (int(trbrX + 10), int(trbrY)), cv2.FONT_HERSHEY_SIMPLEX,
        0.65, (255, 255, 255), 2)
# show the output image
cv2.imshow("Image", orig)
cv2.waitKey(0)
```

Proposed Method using Homogeneous Background detector:

Object_detector.py

```
import cv2
class HomogeneousBgDetector():
    def __init__(self):
        pass

    def detect_objects(self, frame):
        # Convert Image to grayscale
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

        # Create a Mask with adaptive threshold
        mask = cv2.adaptiveThreshold(gray, 255, cv2.ADAPTIVE_THRESH_MEAN_C,
cv2.THRESH_BINARY_INV, 19, 5)

        # Find contours
        contours, _ = cv2.findContours(mask, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)

        #cv2.imshow("mask", mask)
        objects_contours = []

        for cnt in contours:
            area = cv2.contourArea(cnt)
            if area > 2000:
                #cnt = cv2.approxPolyDP(cnt, 0.03*cv2.arcLength(cnt, True), True)
                objects_contours.append(cnt)

        return objects_contours
```

The above object detection method is used to detect objects in the proposed method.

The object detection process begins with identifying contours, followed by determining the pixel-to-centimetre conversion metric.

Dimension_detection.py

```
#draw object boundaries
for cnt in contours:
   #draw polygon
   cv2.polylines(img, [cnt], True, (255, 0, 0), 2)

   #get rectangle
   rect = cv2.minAreaRect(cnt)
   (x, y), (w, h), angle = rect

   #Get width and weight of the objects by applying ratio pixel cm
   ob_width= w / pixel_cm_ratio
   ob_height= h / pixel_cm_ratio



   box = cv2.boxPoints(rect)
   box = np.int0(box)



   cv2.circle(img, (int(x), int(y)), 5, (0, 0, 255), -1)
   cv2.polylines(img, [box], True, (255, 0, 0), 2)
   cv2.putText(img, "width {} cm".format(round(ob_width, 1)), (int(x - 100), int(y
- 15)), cv2.FONT_HERSHEY_PLAIN, 2, (100, 250, 50), 2)
   cv2.putText(img, "height {} cm".format(round(ob_height, 1)), (int(x - 100),
int(y + 15)), cv2.FONT_HERSHEY_PLAIN, 2, (100, 250, 50), 2)

#    print(x, y)
#    print(w, h)
#    print(angle)
#    print(box)

cv2.imshow("Image", img)
```