Republic of Iraq
Ministry of Higher Education and Scientific Research
University of Babylon
College of Information Technology
Software Department

CLUSTERING MULTIPLE MOVING OBJECTS BASED ON THEIR TRAJECTORIES USING A GRAPH MINING ALGORITHM

A Dissertation

Submitted to the Council of the College of Information Technology, University of Babylon in Partial Fulfillment of the Requirements for the Doctor of Philosophy Degree in Information Technology-Software

By

Mustafa Asaad Hasan Alwan

Supervised by

Prof. Dr. Tawfiq Abdul Al-Khaleq Abbas Abdul Al-Reda

2022A.D. **1444 A.H.**

بسم الله الرحمن الرحيم

اقْرَأْ بِاسْمِ رَبِّكَ الَّذِي خَلَقَ * خَلَقَ الإِنْسَانَ مِنْ عَلَقٍ * اقْرَأْ وَرَبُّكَ الأَكْرَمُ * الَّذِي عَلَّمَ بِالْقَلَمِ * عَلَّمَ الإِنْسَانَ مَا لَمْ يَعْلَمْ

صدق الله العلي العظيم

سورة العلق: 1-5

Declaration

I hereby declare that this dissertation, submitted to the University of Babylon as fulfillment of requirements for the degree of Doctor of Philosophy in Information Technology\ Software has not been submitted as an exercise for a similar degree at any other university. I also certify that the work described here is entirely my own.

Signature:

Name: Mustafa Asaad Hasan Alwan

Date: \ \ 2022

Supervisor Certification

I certify that the dissertation entitled **(Clustering Multiple Moving Objects Based on their Trajectories Using a Graph Mining Algorithm)** was prepared under my supervision at the department of Software/ College of Information Technology/ University of Babylon as partial fulfillment of the requirements of the degree of Doctor of Philosophy in Information Technology-Software.

Signature:

Supervisor Name: Prof. Dr. Tawfiq Abdul Al-Khaleq Abbas Abdul Al-Reda

Date: / / 2022

Head of the Department Certification

In view of the available recommendations, I forward the thesis entitled **"Clustering Multiple Moving Objects Based on their Trajectories Using a Graph Mining Algorithm"** for debate by the examination committee.

Signature:

Prof. Dr. Ahmed Saleem Abbas

Head of Software Department

Date: / / 2022

Certification of the Examination Committee

We, the undersigned, certify that (Mustafa Asaad Hasan Alkhafaji) candidate for the degree of Doctor of Philosophy in Information Technology-Software, has presented his dissertation of the following title (**Clustering Multiple Moving Objects Based on their Trajectories Using a Graph Mining Algorithm**) as it appears on the title page and front cover of the dissertation that the said dissertation is acceptable in form and content and displays a satisfactory knowledge of the field of study as demonstrated by the candidate through an oral examination held on: (/   /2022).

Signature:
Name: Dr. Kadhim Mahdi Hashim
Title: Professor
Date: /   / 2022
(**Chairman**)

Signature:
Name: Dr.Hala Bahjat Abdul Wahab
Title: Professor
Date: /   / 2022
(**Member**)

Signature:
Name: Dr. Enas Hamood Al -Saadi
Title: Assist. Professor
Date: /   / 2022
(**Member**)

Signature:
Name: Dr. Hadab Khalid Obayes
Title: Assist. Professor
Date: /   / 2022
(**Member**)

Signature:
Name: Dr. Ahmed Habeeb Said Al-Azawei
Title: Assist. Professor
Date: /   / 2022
(**Member**)

Signature:
Name: Dr. Tawfiq Al-Assadi
Title: Professor
Date: /   / 2022
(**Member and Supervisor**)

Approved by the Dean of the College of Information Technology, University of Babylon.

Signature:
Name: Dr. Hussein Atiya Lafta
Title: Professor
Date: /   / 2022
(Dean of Collage of Information Technology)

Dedication

To my God, my master, creator and inspiration,

To my supportive mother who did not give birth to me,

To my mother and father who rest in peace,

To my wife, my pillar of support. Thanks for holding me when I fell and helping me rise above and beyond,

to my parents and kids,

To my brothers and sister, who supported me until the completion of this research,

And to my friends, colleagues, and relatives,

I dedicate this work.

Acknowledgements

First and foremost, I would like to thank my God, Allah Almighty, for giving me endless graces. My deep sense of gratitude goes to the beacon of science, to the master of creatures, to the greatest Prophet, Mohammed (Peace be upon Him and His Family).

I take this opportunity to express my sincere gratitude and greatest appreciation to my supervisor **Prof. Dr. Tawfiq Abdul Al-Khaleq Abbas Abdul Al-Reda** for his continuous support to accomplish my Ph.D. study, his patience, motivation, enthusiasm, and immense knowledge. Words are inadequate, and thanks are not enough to express my gratitude for his tremendous support and help. I feel motivated and encouraged every time I attend his meeting. His tireless guidance has helped me immensely in researching and writing this dissertation.

Also, I would like to show my gratitude to who weaved my happiness from strings woven from her heart to my dear Mother. My thanks and appreciations also go to my family, for their encouragement, support and patience.

I wish to express my love and gratitude to my beloved wife for her understanding and endless love through the duration of my study.

Finally, I would also like to express my thanks and gratitude to all those who contributed to making this dissertation possible, and foremost among them all the teaching and staff members at the college of Information Technology at the University of Babylon, headed by **Prof. Dr. Hussein Attieh Lafta.**

Abstract

Computer vision is one of the important scientific fields in the modern era because the video elements contain rich and important information. Hence, knowledge and data van be ontained to refer to a huge amount of useful information. The process of distinguishing and separating only the discovered information is one of the complex and well-known problems. The problem of classification and clustering of moving objects in video data is also a complex task that requires mechanisms, operations, as well as algorithms for the purpose of solving it and obtaining distinct results as possible.

In this dissertation, a system is proposed for the purpose of clustering moving objects based on their behavior using a graph mining algorithm. A new algorithm is proposed for the purpose of mining the large data that are represented using a graph. Moreover, another algorithm is proposed for the purpose of data reduction and extracting the important data only. Some of the algorithms used in the proposed system have also been adapted in order to increase their performance.

The proposed system firstly splits the video input into sequences frames. The second phase is to apply some preprocessing operations to enhance the quality of frame (still image). The third phase is to apply You Only Lock Once (YOLO) multiple objects detection and Simple Online and Real Time Tracking with a Deep Association Metric (Deep-SORT) tracking objects to discover and track objects with different classes. The fourth phase is to build trajectory for each object and apply a new proposed shape normalization algorithm. The fifth phase is to extract features for trajectories and construct graph for them. The graph data are stored in graph database. The sixth phase is to apply a new

suggested graph mining algorithm to mine the interested data. Finally, fuzzy c-means is applied to cluster data into a different number of groups.

The experimental results suggest that the proposed system is robust with high performance. Algorithms used for detection and tracking outperformed the findings of other detecting and tracking algorithms as they achieved a high accuracy. Moreover, the proposed normalization algorithm shows that about 50% of unrich points are discarded. Furthermore, the graph mining proposed algorithm showed high performance to extract interested data. In addition, the proposed algorithm for graph mining showed a high performance of more than 95% for extracting important data.

Declaration Associated with this Dissertation

Some of the works presented in this dissertation have been published as listed below.

1. M. A. H. Alkhafaji and T. A. Al-Assadi, “A Review of Tracking and Clustering Multiple Objects by Using Graph Mining Algorithms,” *Webology*, vol. Volume 18, no. Special Issue on Current Trends in Management and Information Technology, pp. 177–190, 2021, doi: 10.14704/WEB/V18SI05/WEB18222.
2. M. Asaad, H. Alkhafaji, and T. A. Al-Assadi, “Clustering Of Objects Trajectories By Using Graph Mining Algorithm,”pp. 634-644, 2021. [Online]. Available: http://www.webology.org
3. Graph Mining Algorithm of Multiple Objects Tracking and Data Clustering (The National University of Science and Technology International Conference for Pure and Applied sciences, Paper ID: NUSTPAS_P_3, Present Date: 2Jun 2022).

Table of Contents

List of Tables

List of Figures

List of Algorithms

List of Abbreviations

Abbreviation	Description
AGM	Apriori-like wise based Graph Mining
AMOTA	Meteor Observers in the Tokyo Area
BFS	Breadth First Search
CNN	Convolution Neural Network
CRF	Conditional Random Field
CRM	Customer Relationship Management
DAEGC	Deep Attentional Embedded Graph Clustering
DBSCAN	density-based spatial clustering of applications with noise
Deep-SORT	Simple Online and Realtime Tracking with a Deep Association Metric
DFS	Depth First Search
DNN	Deep Neural Networks
FAM	Fuzzy Adjacency Matrix
FCM	Fuzzy C-Means
fps	Frame Per Second
FSG	Frequent Structural Graph
GOTURN	Generic Object Tracking Using Regression Networks
GPS	Global Positioning System
gSpan	Graph-Based Substructure Pattern Mining
HOG	Histogram of Oriented Gradients
ICA	Independent Computing Architecture
IOU	Intersection Over Union

KCF	kernelized correlation filters
KDD	Knowledge Discovery in Databases
KITTI	Karlsruhe Institute of Technology and Toyota Technological Institute
MD-Net	Multi-Domain Net
MOT	Multiple Objects Tracking
PCA	Principle Component Analysis
RCNN	Recurrent Convolution Neural Network
ROLO	Recurrent YOLO
SIFT	Scale-Invariant Feature Transform
SORT	Simple Online and Realtime Tracking
SPP-Net	spatial pyramid pooling
SSD	Single Shot Multi-Box Detector
YOLO	You Only Lock Once

Chapter One: General Introduction

1.1 Background

Computers are used to extract knowledge from varied, enormous volumes of data that are stored in various types of databases or that may be produced by data structures with substantial amounts of image, text, audio, and video information. The difficulty in locating rich data is not appropriate with the size of input, which is growing linearly, according to search algorithm inventors, even though the amount of input data and the size of the database are both rising quickly. Indeed, the fastest computers and greatest algorithms are required [1].

Data mining is a method for obtaining useful information from a vast volume of data. It involves employing one or more pieces of software to look for patterns in huge amounts of data. Data mining is a multidisciplinary branch of computer science where its main goal is to extract information from datasets using intelligent techniques and organize the information into clear structures for subsequent use. Data mining is the "knowledge discovery in the database" process or KDD's analysis step [1]–[3]

Video mining is a method to figure out the interesting information or patterns from huge amount of video data. Video mining concept definition is a way not just to extract meaningful information, properties of moving objects, spatial or temporal of those attributes; however, it is figuring out video structure patterns [4], [5]. Video mining has three main tasks: pre-processing, features and semantic information extraction, video patterns and knowledge discovering and forming. Video mining has different applications and usages such as traffic video sequences, medicine, surveillance system and security programs [6], [7]. Video tracking is the process of utilizing a camera to determine the location of an item that changes its position over time. There are many applications for video tracking, including traffic control, augmented reality, security and surveillance, human-computer interaction, video communication, and compression [8]–[11].

Information is separated into related, homogeneous collections of items through the act of grouping. Some groups can describe the data, but they have made improvements. These groups typically lack detailed restriction specifics. A historical viewpoint based on mathematical analysis, statistical analysis, and statistics is used to organize data modeling. Clusters are connected to hidden modes from a machine learning perspective, finding clusters is unsupervised learning, and the post-framework symbolizes a data idea. Practically, clustering has a complicated function to play in data mining applications. Examples include the analysis of scientific data, text mining, information retrieval, applications for spatial databases, CRM, online analytics, computational biology, diagnostics for health care, object grouping, etc. [12]. In other words, cluster analysis is a data mining method for locating comparable data. This method aids highlighting data variances and similarities. The clustering process is like the classification process, except it groups block of data according to how related they are [13].

Data mining's clustering technique is crucial for identifying groups and structures that share characteristics. Data clustering has several applications, including data analysis, picture processing, pattern identification, and market research. There are many different types of clustering, including hierarchical, fuzzy, dense-based, and model-based techniques. Given data in n-dimensional space, the difficult task of clustering is to divide the data into k clusters that share comparable qualities. For mining graph-based data and doing insightful analysis on this data, numerous data mining techniques are employed. There are many graph mining methods. These methods rely on categorization, grouping, or decision-making. In the discipline of biology, clustering aids in the classification of genes, plants, and animals. This task involves unsupervised learning [14]–[17].

The relationship between things is represented in a graph, which is a block diagram made up of nodes which stand in for the objects and edges which represent the connections between nodes. The method and technology which is known as "graph mining" is used to examine the characteristics of real-world graphs, develop models that might produce realistic graphs, and forecast the qualities and structure of a given graph. Two types of graph mining exist: Apriori-style and pattern-growth methods [2], [18]–[20].

Multimedia data mining is the identification of intriguing patterns (structures) that are extracted from multimedia data, and it is used to store and manage enormous collections of multi-media objects, including image data, video data, audio data, as well as sequence data and hypertext data containing text, text labels, and linkages. An interdisciplinary field known as multimedia data mining combines image processing, computer vision, data mining, and pattern recognition. Multimedia data mining has many challenges such as content-based retrieval, similarity search, generalization, and multidimensional analysis [4], [5].

By tracking several objects in video footage and describing their locations as connected trajectories, graph mining methods can be used to cluster multiple objects. Each item in the movie is clustered using the trajectory as an adjacency matrix, or the entire video is clustered using a representation of all trajectories as a graph tree. A graph mining method can also be employed to organize these frequent subgraphs into homogenous and connected groups, each of which contained objects with somewhat similar behaviors and features [21].

In this dissertation, a graph mining algorithm is proposed to analyze and extract interesting data from huge of graph database. After video dataset is read, it will be converted into sequence frames. Then pre-processing will be applied on all frames to enhance the quality of recognized foreground moving objects. After that, the detecting and tracking algorithm is run to detect and track moving objects. Moreover, the trajectories of objects are constructed, and the features will be extracted. The normalization algorithm will be applied to eliminate the unnecessary points. Furthermore, graph will be constructed for each trajectory object. An adaptive graph mining algorithm is run to mine on interested data graph. Finally, clustering algorithm will be applied to cluster the objects into number of sets based on their behaviors.

1.2 Problem Statement

Video device capture becomes very popular and important to obtain better information because it takes huge number of frames (still image) where it records the views with high-speed and high accurate information. The video frames supply massive information about how the moving object changes during time. Clustering multiple moving objects of multi-classes is a complex task. This issue is complex because clustering needs to know the type of object and the nature of its moving behaviors, and that requires advance processing. Therefore, the traditional processes will show slow progress to solve these issues. Graph models can represent large database of video information in small and rich data platform. Therefore, graph mining can be used to mine and cluster multiple moving objects in video relied on their behaviors.

1.3 Research Objectives

There are four objectives of this dissertation which can be summarized as follows:

1- Building graph mining algorithm to seek and cluster patterns of multiple objects behaviors that contain the nature of relationship in video dataset.
2- Developing two types of detection and tracking multiple objects in video scene dataset.
3- Construction of a model to convert the trajectory object into graph structure. Each trajectory object represents by all frames from the frame of object appears (birth object) into last frame before the object disappeared (death object) during entire video and building graph database that represents a spatiotemporal data base.
4- Building a normalization shape framework of object trajectory to prune and extract only important points that affect the shape of the trajectory object.
5- Proposing a structure framework to cluster multiple moving objects using fuzzy c-means.

1.4 Research Contributions

The contributions of this dissertation can be highlighted as follows:

1- Proposing a new graph mining algorithm to mine the interesting data from huge graph data. This algorithm is depending on the features of object during entire moving inside the scene to know the behaviors of that object. This method helps to convert the huge and unknown data into rich and important data.

2- Suggesting a framework to detect and track multiple moving objects with high accuracy by using two robust and modern algorithms because these algorithms will assist the system to give its outcomes with best performance.
3- Proposing a structure to graph and graph database construction by using a new concept of converting the traditional adjacent matrix (0 or 1) into fuzzy adjacency matrix (FAM) (0-1) that represent the nature of relations between nodes of each graph.
4- Proposing a new adaptive normalization approach to norm the shape of each object trajectory by using just the necessary and rich points of each trajectory (path) of object.

1.5 Literature Review

Object tracking is very important research area in computer vision due to the large applications like video-surveillance, pedestrian protection systems, tracking complicated surfaces, medical image applications etc. Clustering is one that plays an important function in moving object trajectory mining. There are many research on clustering objects in video processing because clustering become very important research area in most studies

This dissertation emphasis three concepts: (1) objects detection and tracking, (2) graph mining, and (3) clustering of graph mining

1.5.1 Objects Detection and Tracking

- In [22], a solution to a more limited problem, namely bird's eye stationary videos without real-time condition is prpopsed. It applies three methods for detecting and tracking three different priority levels. The first level is background subtraction. It is a fast method to find which pixels have changed in the image (the foreground). The second level is multi-objects tracking methods such as kernelized correlation filters (KCF). This method takes the output of the detection and the next frame, and it gives the next positions of the objects. The third level is the deep learning-based detection and classification of the objects. Here, YOlO-v4 is used as a model structure where training is done separately.

- Tianwei et al. [23] proposed a center-based framework for simultaneous 3D object detection and tracking from the Lidar point-clouds. Their approach illustrates a standard 3D point-cloud encoder with a few convolutional layers in the head to produce a bird-eye-view heatmap and other dense regression outputs. Detection is a simple local peak extraction with refinement, and tracking is a closest-distance matching. CenterPoint is simple, near real-time, and achieves state-of-the-art performance on the Waymo and nuScenes benchmarks.

- In [24], YOLO detection technique and deep-sort tracking algorithm were proposed. It was found that the YOLO method is thinking worldwide when producing image predictions, and that the detection results for objects have high accuracy with high-speed recognition. They were also found that the deep-sort tracking algorithm, which dramatically treats long-period occlusion and reduces identity switches by roughly 45%, is a useful method to utilize in conjunction with the YOLO algorithm. This structure has a great learning capability and can detect numerous class kinds in the same image very quickly.

-Elhoseny [25], suggested a fresh MODT Multiple Objects Detection Tracking approach. The proposed method tracks moving objects in video frames using an ideal Kalman filtering technique. The expanding model was used to translate the video clips into morphological processes based on the number of frames. After differentiating the objects, the probability-based grasshopper method was used to apply Kalman filtering for parameter optimization. Using the best settings, a similarity metric tracked the chosen items in each frame. Finally, the suggested MODT framework was put into practice, and the outcomes were evaluated. The trials revealed that the highest detection and tracking accuracies of the MODT framework were 76.23% and 86.78%, respectively. The outcomes of earlier investigations are contrasted with the outcomes obtained with Kalman filtering in the MODT method.

1.5.2 Graph Mining

- Ngoc-Thao et al. [26], proposed propose a frequent subgraph algorithm on a weighted large graph, called Weighted Graph Mining (WeGraMi), which is based on two effective aspects of mining weighted subgraphs. Firstly, we apply a new strategy was applied to calculate the weight of all mined subgraphs, which is based on the weights of the nodes in that subgraph. Secondly, they apply a search space pruning strategy was performed based on the existing weights. If a frequent subgraph cannot satisfy the given weighting threshold, that subgraph will be pruned, which can reduce the processing time and storage space needed. With both directed and undirected graph datasets, their experimental results show that the runtime as well as the memory requirements of the algorithm are significantly better than those of GraMi.

- another research study [27] presented Kaleido, a single-machine, out- of-core graph mining system. Kaleido follows the subgraph-centric model and provides a user-friendly simple API that allows non-experts to build graph mining workloads easily. To efficiently store and process a huge amount of intermediate data, Kaleido builds a succinct intermediate data structure and adjusts the storage in memory or out of core smoothly according to the scale of intermediate data. Kaleido designs a lightweight and efficient graph checking algorithm for small graphs in which the number of vertices is less than 9. Experimental results demonstrate that Kaleido is more efficient than state-of- art graph mining systems in most cases. The isomorphism testing algorithm in Kaleido is more efficient and consumes less memory than the state-of-the-art graph library.

- In [28], a g-Span graph mining algorithm was used to find out patterns' sequences and relationships between objects in the movie, which then served as an illustration of object grouping. A graph data-friendly algorithm was created. The strategy uses several different techniques. The video must first be transformed into a few sequence photos. The second phase involves identifying the items in each frame and extracting their attributes using a segmentation technique. Making a database-driven row for each feature is the third stage. The researchers create a graph structure to represent each frame in the fourth stage. Finally, cluster the items and analyze their activity using the g-Span technique.

1.5.3 Clustering of Graph Mining

- Liu and Barahona [29] presented a graph-theoretical approach to data clustering, which merged the creation of a graph from the data with Markov Stability, a multiscale community detection framework. It was showed how the multiscale capabilities of the method allow the estimation of the number of clusters, as well as alleviating the sensitivity to the parameters in graph construction. Both synthetic and benchmark real datasets were used to compare and evaluate several graph construction methods and clustering algorithms and show that multiscale graph-based clustering achieves improved performance in comparison to popular clustering methods without the need to set externally the number of clusters.

- Huang et al. [30], proposed ClusterVO which is a general-purpose fast stereo visual odometry for simultaneous moving rigid body clustering and motion estimation. Comparable results to state-of-the-art solutions on both camera ego-motion and dynamic object pose estimation demonstrate the effectiveness of the proposed system. In the future, one direction would be to incorporate specific scene priors as pluggable components to improve ClusterVO performance on specialized applications such as autonomous driving; another direction is to fuse information from multiple sensors to further improve localization accuracy.

- Wang et al. [31] proposed an unsupervised deep attentional embedding algorithm, DAEGC, to jointly perform graph clustering and learn graph embedding in a unified framework. The learned graph embedding combines both the structure and content information and is specialized for clustering tasks. While the graph clustering task is unsupervised, a self-training clustering component was proposed which generates soft labels from "confident" assignments to supervise the embedding updating. The clustering loss and autoencoder reconstruction loss are jointly optimized to simultaneously obtain both graph embedding and graph clustering results. A comparison of the experimental results with various state-of-the-art algorithms validates DAEGC's graph clustering performance.

- In [32], the issue of offering detailed segmentation masks was considered to find objects in films. With the intention of grouping foreground pixels in films into clusters, the object-finding problem was formulated as foreground motion clustering different things. A brand-new pixel-trajectory recurrent neural network was presented which picks up feature embeddings of foreground pixel trajectories that are chronologically related. The method generates correspondences between foreground object masks over video frames by clustering the pixel trajectories using the learnt feature embeddings. They undertook trials on frequently used datasets for motion segmentation, where they reach state-of-the-art performance, to show the efficacy of their system for object detection.

1.6 Research Challenges

Many problems may face detection, tracking, and clustering multiple moving objects such as:

1. Noise produced by quickly moving objects and changes in appearance. Effect of lighting changes on correspondence between multiple viewpoints.

2. Pose change: When a moving object is projected into the image plane, such as during rotation or translation, it takes on a different appearance.

3. Occlusions: When an object is completely or partially hidden by other objects, it cannot be seen. Occlusions typically occur because:

a) A moving target behind a stationary obstacle, such as a wall or a cloud.

b) Other moving objects block the view of a target object.

The target detection algorithm or the target appearance model can handle partial occlusions that only affect a small portion of the target area, while whole occlusions, which constitute a high-complexity task, can be handled using higher-level reasoning or multi-hypothesis techniques.

4. Using two or more traditional graph mining methods that were used to detect recurring patterns, create a hybrid graph mining algorithm or suggest a new one to get good performance by mining important data from a massive graph database.

5. Find a relevant dataset, such as a collection of bacteria, a tank of fish swimming, or a virus under a microscope.

In addition, other difficulties might arise when it starts to implement this type of environment.

1.7 Dissertation Structure

The dissertation is sorted as follows:

Chapter two: this chapter illustrates the main concepts of video tracking, graph mining, and clustering.

Chapter three: this chapter copes with the suggested design stages of the proposed system. and algorithms where it illustrates the reading of the datasets and then clustering multiple moving objects with some experimental practical examples.

Chapter four: this chapter deals with the results that are produced from the proposed system. It then discusses and evaluates these results.

Chapter five: this chapter concludes this research and highlighting some possible future works.

Chapter Two: Theoretical Fundamentals

2.1 Overview

This chapter discusses the main concepts that are used in this dissertation. The proposed system is to clustering multiple objects using graph mining concepts and techniques. The main idea is to detect and track multiple moving objects in the video dataset, then to construct and extract trajectories and their geometric features. After that it normalizes the data to prune unnecessary and unrich data. It constructs a graph for each object trajectory based on its features. Proposed graph mining algorithm to mine the important and rich data for each object trajectory. Finally, it selects an appropriate clustering algorithm to cluster the trajectories and evaluate the results of each stage of the proposed system. The next section discusses the theoretical concept of each stage in this dissertation.

2.2 Preprocessing

Pre-processing stage is an important step before other processes begin because it will help to improve all other steps specifically in the field of computer vision applications. Before starting to detect and track an object in the video scene, it should clean the image (frame) from the noises and struggles that face when other processes will be started. In this dissertation, it will use some enhancement filters and operations to reduce and improve the frames that will increase the robustness and performance of the proposed system. These filters and operation will be applied automatically to each frame before going to the detection and tracking steps.

a. **Gaussian Blur (smoothing function)**: it is a popular method used in computer vision and graphic. It is used to enhance the image in various scales. The function is applied on each image pixel like a convolution function. It is called a low-pass filter, and it is reduced the high frequency components of the image. There are two types of gaussian functions which are one-dimension as shown in Equation 2.1 and two-dimension as presented in Equation 2.2:

$$G(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}} \ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.1)$$

$$G(x,y) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2+y^2}{2\sigma^2}} \ldots\ldots\ldots\ldots\ldots (2.2)$$

where x is the horizontal axis distance from the origin, y is the vertical axis distance from the origin, σ is the standard deviation [33], [34]. Figure 2.1 illustrates an example of applying the gaussian blur function on normal image [34].

(a) (b)

Figure 2.1: (a) Original image; (b) Gaussian blur image [34].

b. **Median filter:** it is a non-linear filter, and it reduces image noise. The method has been used in edge detection. It is a type of smoothing approach to enhance the image by removing noise. For instance, salt and pepper filters are a popular kind of the median filter technique. Figure 2.2 shows an example of median filter result after applying it to a noisy image [35].

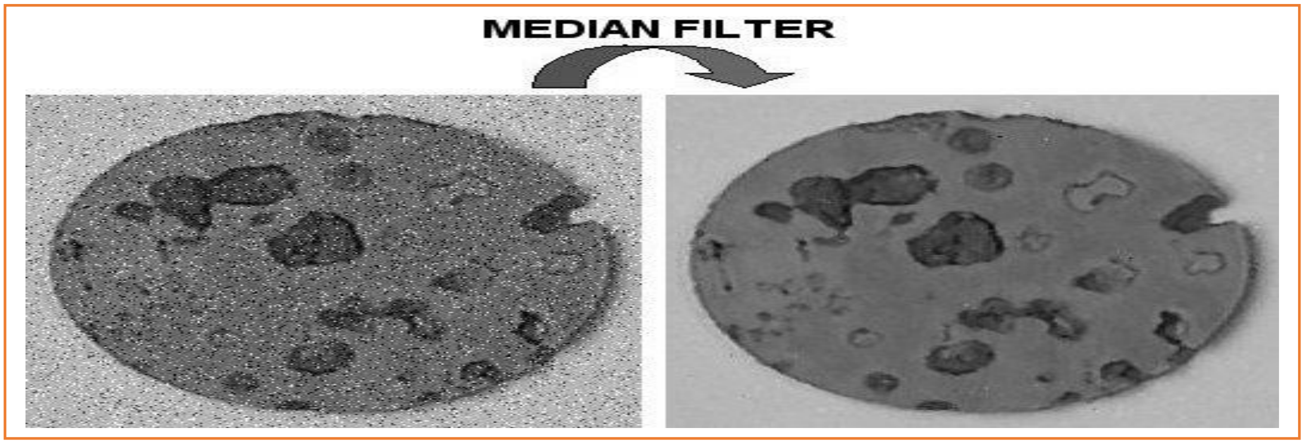


Figure 2.2: An example of applying the median filter on the image; the result is a smoothing image [35].

c. **Sharpening filters:** this method is an inverse of blurring filters and figures out the difference between the neighbors and complete by spatial differentiation. It is used to improve the edge of the objects, change the contrast, and shade properties. This approach is called a high-pass filter which reduces the lower frequencies and is strongly sensitive to remove noise. There are many types of sharpening filters Laplacian filter, high-pass filter, and unsharp masking [36]. Figure 2.3 shows the effect of the sharpening filter on the image and how the edges are sharper and brightness to recognize.

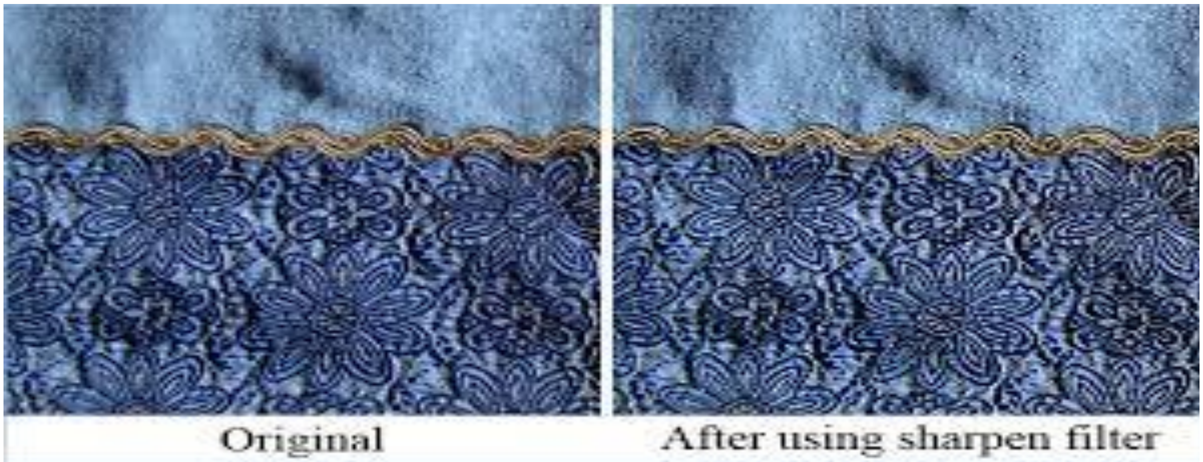


Figure 2.3: Applying sharpening filter on faded image [36].

d. **Morphology:** a set of non-linear operations that affect the morphology of features of shape in an image. These operations are applied to binary images to eliminate the imperfections that occur in binary images. It could be applied on grey scale images. The morphological operations depend on the relative sort of pixel values, not on numerical values that why they are used to be applied on binary images [37]. There are two main operations of morphology that are:

1. **Erosion:** is shrinking the foreground and enlarging the background of an objects image where erosion of image f and its structure element s is denoted by $f\theta S$ and the structure element is located with its origin center (x, y) and the new pixel value is calculated by using the rule as shown in Equation 2.3 [37].

$$g(x,y)=\begin{cases}1 \text{ if } s \text{ fits } f\\ 0 \text{ otherwise}\end{cases} \quad \ldots\ldots\ldots\ldots\ldots (2.3)$$

Figure 2.4 shows an example of the erosion morphology operation of different structure elements.

Figure 2.4: An example to apply erosion morphology operation with different structure elements [37].

2. **Dilation:** is enlarging the foreground and diminishing the background of an objects image where dilation of image f and its structure element s is denoted by $f \oplus S$ and the structure element is located with its origin center (x, y) and the new pixel value is calculated by using the rule as presented in Equation 2.4 [37].

$$g(x,y)=\begin{cases}1 \text{ if } s \text{ hits } f \\ 0 \text{ otherwise}\end{cases} \quad \text{..............} \quad (2.4)$$

Figure 2.5 illustrates an example of using dilation morphology operation of different structure elements.

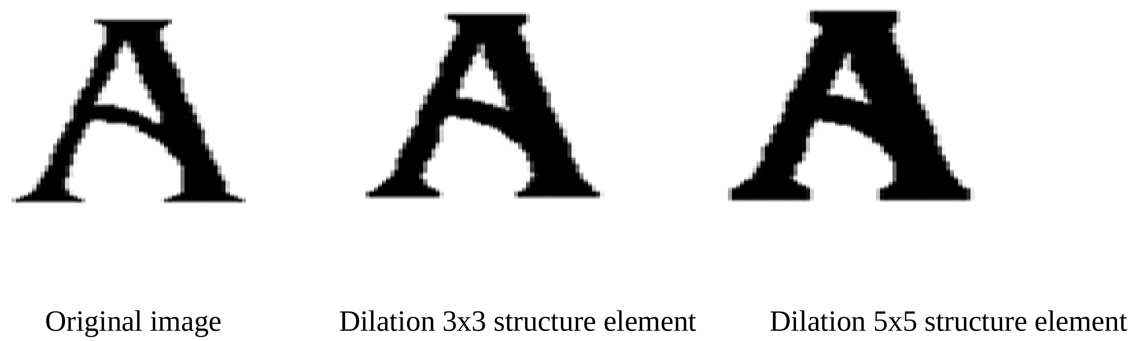


Figure 2.5: An example of image dilation by using different sizes of structure elements [37].

2.3 Object Detection Techniques

Object detection is an image processing and computer vision technology that deals with discovering semantic objects of a specific class in digital videos and images like cars, humans, animals, etc. Object detection is popularly used in computer vision applications such as face recognition, face detection, and activity recognition. Moreover, object detection is used in object tracking like tracking humans in video and tracking balls during a football match. Each object has its own features that assist to identify the class. For instance, when the method looks to detect squares, an object comes with four equal sides and corners are perpendicular, and the approach is also with face recognition where features like skin color and distance between eyes are used to distinguish face [38]. Figure 2.6 shows the object detecting and tracking mechanism model.

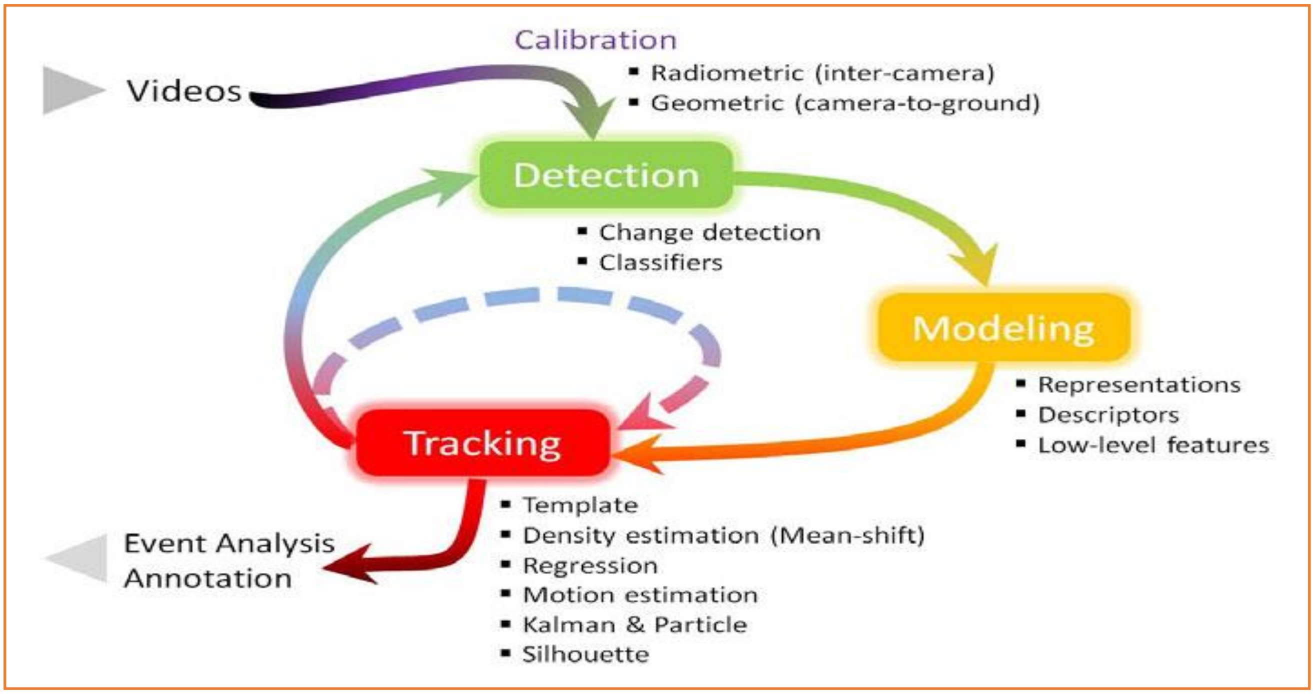


Figure 2.6: The objects detection and tracking model [39].

In general, methods of object detection are divided into non-neural approaches and network-based approaches. Each type has a special technique to detect the object and its class.

2.3.1 Non-neural Object Detection Approaches

In these techniques, features must be defined first, and then any classification method is used to separate the object class. It is also called traditional methods because the techniques used to detect the object are extracting simple features and processing steps with less accuracy. There are many types of non-neural object detection methods such as frames differencing, background subtraction, motion segmentation, scale-invariant feature transform (SIFT), Viola-Jones object detection framework based on Haar features, Histogram of oriented gradients features (HOG) and so many of them. These methods have many issues and challenges since they low to moderate accuracy and performance, and they are not appropriate for the detection process [40].

2.3.2 Neural Network-based Object Detection Approaches

It is traditionally used conventional neural networks (CNN). It is an end-to end object detection technique for defining any feature. CNN detection methods can be two-stage detectors or one-stage detectors. In a deep conventional neural network, detectors can learn strong and high-level representations of feature of an object image. Techniques of two-stages detection are RCNN (Recurrent Convolution Neural Network), Fast-RCNN, Faster-RCNN, SPPNet, and Feature Pyramid Networks. On the other hand, the one-stage detectors approaches are You Only Lock Once (YOLO), Single Shot MultiBox Detector (SSD), and RetinaNet. They are very fast and accurate detection methods in the current era [41], [42]. In this dissertation YOLO method is adapted.

- **You Only Lock Once (YOLO) Detection Algorithm**

In this dissertation, You Only Lock Once (YOLO) algorithm has been used to detect multiple objects with about more than 20 classes. YOLO is a new object detection method, and YOLO is a single (one-stage) neural network bounding box prediction and directly class probabilities from entire image. YOLO is extremely fast, and it works in real-time at 45 frames per seconds while the Fast YOLO version processes 155 frames per seconds. Since YOLO uses single pipeline neural network;

YOLO is optimized end-to-end detection performance directly. Three reasons lead to the use of the YOLO detection technique over other object detection approaches in this dissertation. First, YOLO is very fast and robust. Second, YOLO algorithm sees the full image during test and training time. Therefore, it encodes organized information about classes as best as their appearance. The third reason is that YOLO can learn generally and represents objects. When it is trained on natural images and tested on artwork, the YOLO exceeds top detection methods. It can be used with unknown inputs or new domains, and it is less likely to fail in a detection task [43], [44].

YOLO detection algorithm divides the input frame (image) into an S X S grid. If an object center in an image full in a grid cell, then that cell will be responsible for detecting an object. Each pixel grid can predict B bounding boxes and score confidence for the boxes. The confident scores represent how the object exists in the bounding box and how accurate the prediction of that box will be. It will be zero if no object exists in the grid cell; however, the score confidence is equal to the intersection over union (IOU) between the box prediction and any box of ground truth. Figure 2.7 expresses the detecting and tracking of multiple moving object mechanisms [44].

The bounding box has 5 predictions: x, y, w, h, and confidence score where the (x, y) coordinates are representing the center of the box compare with bounding in the grid cell. The w and h are the width and height which are predicted to the entire image relatively. Indeed, the confidence metric is the IOU (Intersection Over Inion). In addition, each cell also predicts C class probabilities that restrict with condition $Pr(Class_i \mid Object)$. It predicts just one set of class probabilities for each grid cell by ignoring the number of boxes B. The specific class confidence for each box can be calculated by Equation 2.5 [43]:

$$Pr(Class_i \mid Object) * Pr(Object) * IOU^{truth\text{-}pred} = Pr(Class_i) * IOU^{truth\text{-}pred} \quad \ldots\ldots (2.5)$$

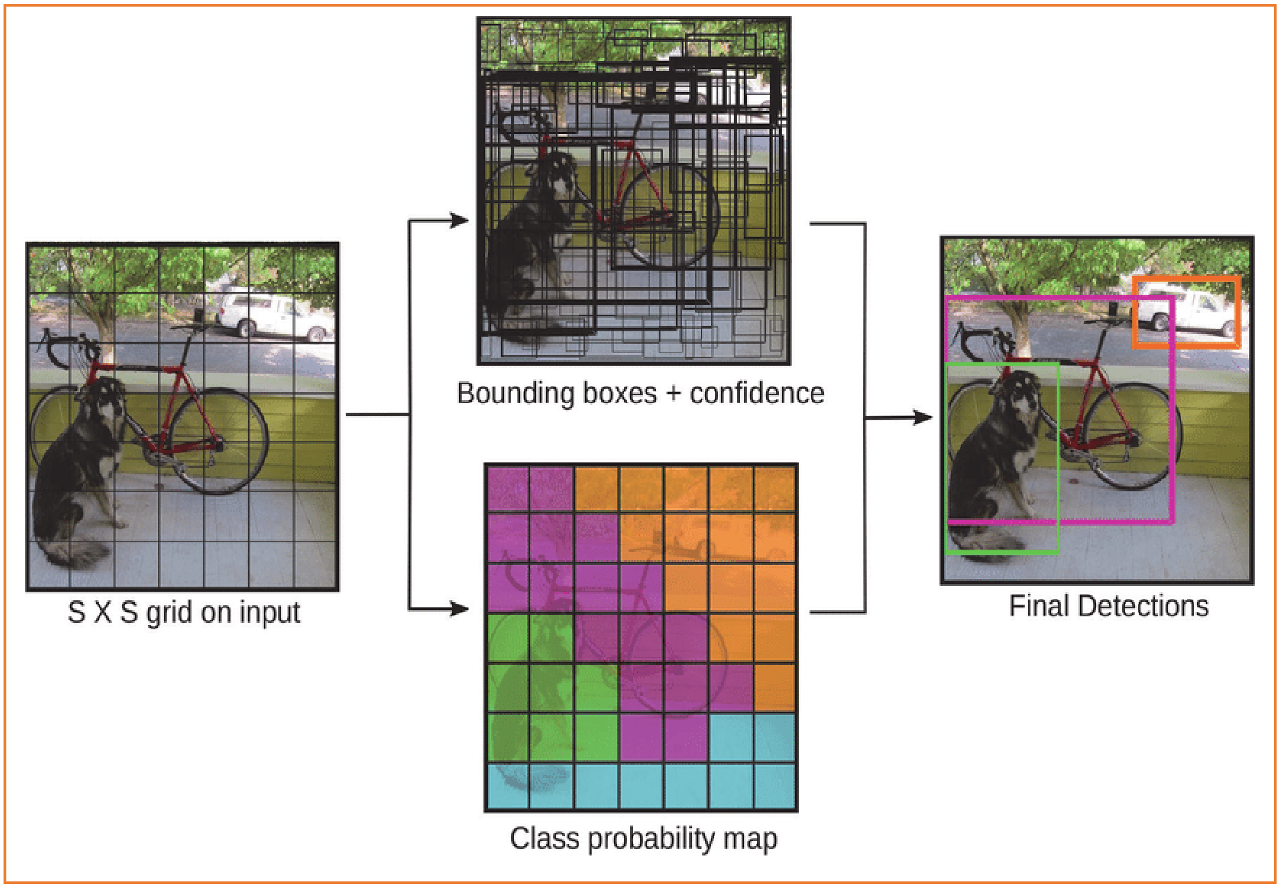


Figure 2.7: The YOLO detection algorithm Model as a Regression Problem [43].

The network that is implemented in this model is a conventional neural network. The first conventional layer is extracting attributes from the image, and the fully connected layers are predicting the coordinates and the output probabilities. Figure 2.8 shows the architecture network of YOLO detection algorithm.

The network has 24 conventional layers, and are followed by 2 fully connected layers. The developers simply use 1 x 1 reduction layers, and they are followed by 3 x 3 conventional layers. Moreover, they train YOLO of the fast version to enter the boundaries of fast object detection. The end output network is the 7 x 7 predictions tensor.

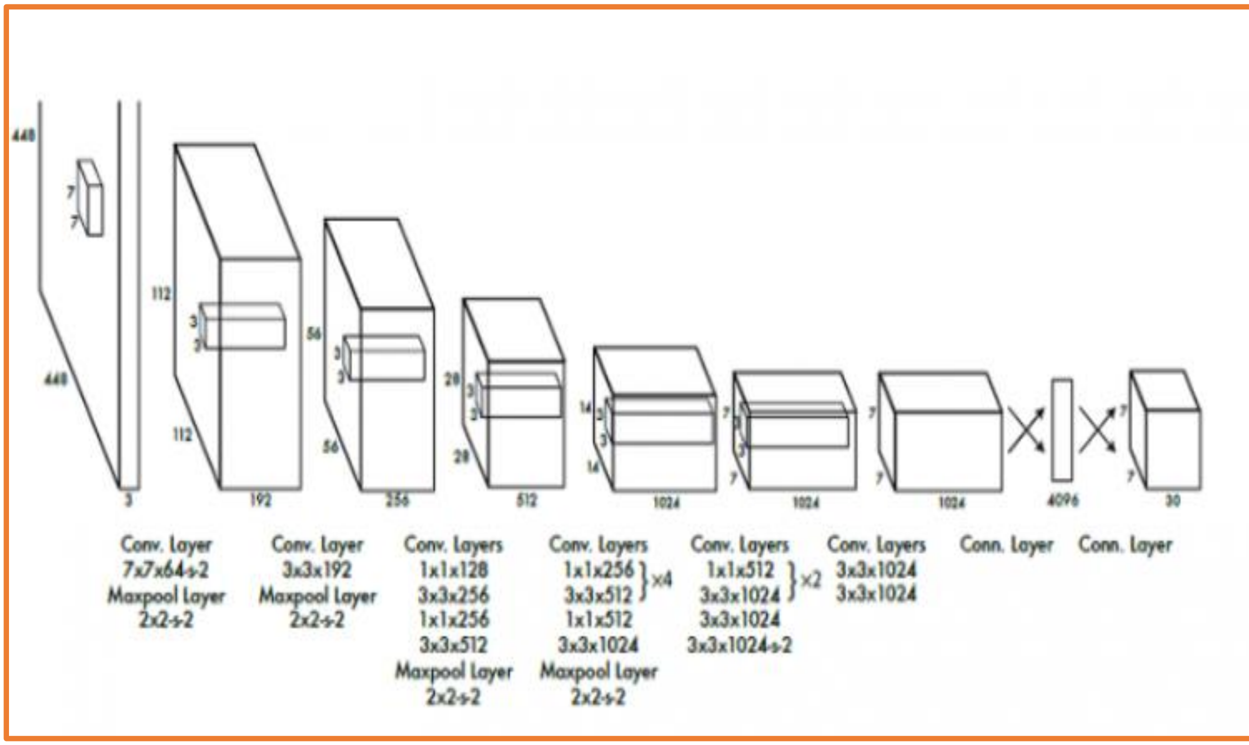


Figure 2.8: The Architecture of YOLO Detection Network Algorithm [44].

The researchers pretrained the conventional network layers on the ImageNet dataset of 1000-class. They used 20 conventional layers for pretraining, and it is average pooling and fully connected layer comes next. The developers added four conventional layers and two completely connected layers with stochastically initialized weight. They increased the input accuracy of the network where it was 224 x 224, and it became 448 x 448 so that can enhance the detection resolution of visual information. The class probabilities layer and the coordinates of the bounding box are the final layers predictions. It will normalize the bounding box by width and height by falling in the range of 0 to1. The coordinates x and y are also normalized to be bounded in between 0-1. Moreover, the last layer was activated using a linear activation function, while all other layers illustrate linear filtering rectified activation function as shown in Equation 2.6 [45]:

$$\phi(x) = \begin{cases} x & if\ x > 0 \\ 0.1x & otherwise \end{cases} \ldots\ldots\ldots\ldots\ (2.6)$$

Figure 2.9 shows the architecture of the YOLOv3.

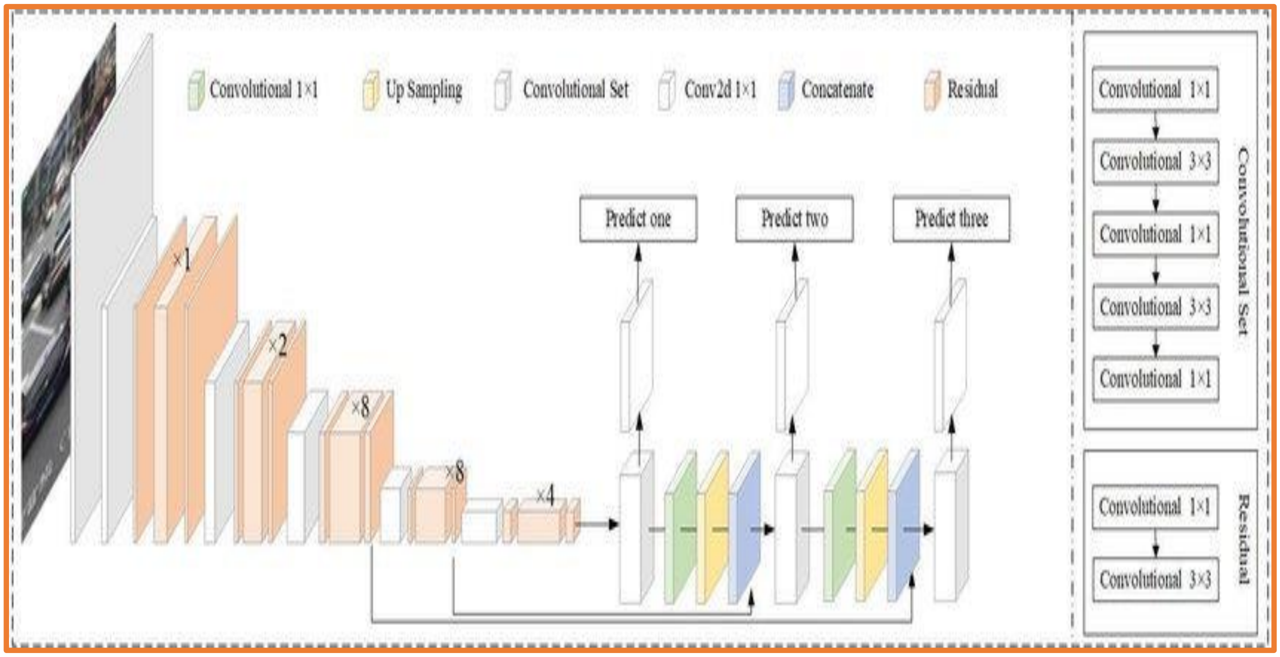


Figure 2.9: The architecture of YOLOv3 network object detection algorithm [45].

Table 2.1: YOLOv3 convolutional layers Darknet-53 [45].

Type	Filters	Size	Repeat	Output
Image				416x416
Conv	32	3x3	1	416x416
Conv	64	3x3/2	1	208x208
Conv Conv Residual	32 64	1x1 3x3	x 1	208x208 208x208 208x208
Conv	128	3x3/2	1	104x104
Conv Conv Residual	64 128	1x1 3x3	x 2	104x104 104x104 104x104
Conv	256	3x3/2	1	52x52
Conv Conv Residual	128 256	1x1 3x3	x 8	52x52 52x52 52x52
Conv	512	3x3/2	1	26x26
Conv Conv Residual	256 512	1x1 3x3	x 8	26x26 26x26 26x26
Conv	1024	3x3/2	1	13x13
Conv Conv Residual	512 1024	1x1 3x3	x 4	13x13 13x13 13x13
Avgppol		Global		
Connected SoftMax		1000		

There are many versions of YOLO, and each one has some characteristics and limitations. In this dissertation, YOLOv3 (You Only Lock Once version 3) is used to detect multiple objects and classes of objects. YOLOv3 is proposed by Redmon and Farhadi in 2018 [46]. YOLOv3 predicts bounding boxes at three different scales. Figure 2.9 shows the Architecture of the YOLOv3 network. Features are extracted from those metrics, and it uses the same concept as feature pyramid networks. YOLO v3 is performing 1 x 1 detection kernels on feature maps of three different sizes at three different scales in the network to finish the detection process. YOLOv3 utilizes binary cross-entropy for computing the classification loss for each label, while class predictions and object confidence are predicted through logistic regression [46]. The authors used a new hybrid network for feature extraction. It used the network between YOLOv2, Darknet-19, and modern residual network stuff. Figure 2.9 and table 2.1 illustrate the YOLOv3 convolutional network layers and scales (Darknet-53) in details [47].

2.4 Object Tracking Techniques

Video tracking is a way to estimate the location over time of one or more objects by using a camera. There are different usages of video tracking such as surveillance, video editing, augmented reality, traffic control, object recognition etc. The main goal of video tracking is to associate target objects in a sequences video frame. The fast improvement in resolution and quality, and the rapidly increased the computational power. There are many types of algorithms and applications which use video tracking [48], [49]. Figure 2.10 shows an example of multiple objects tracking in one frame.

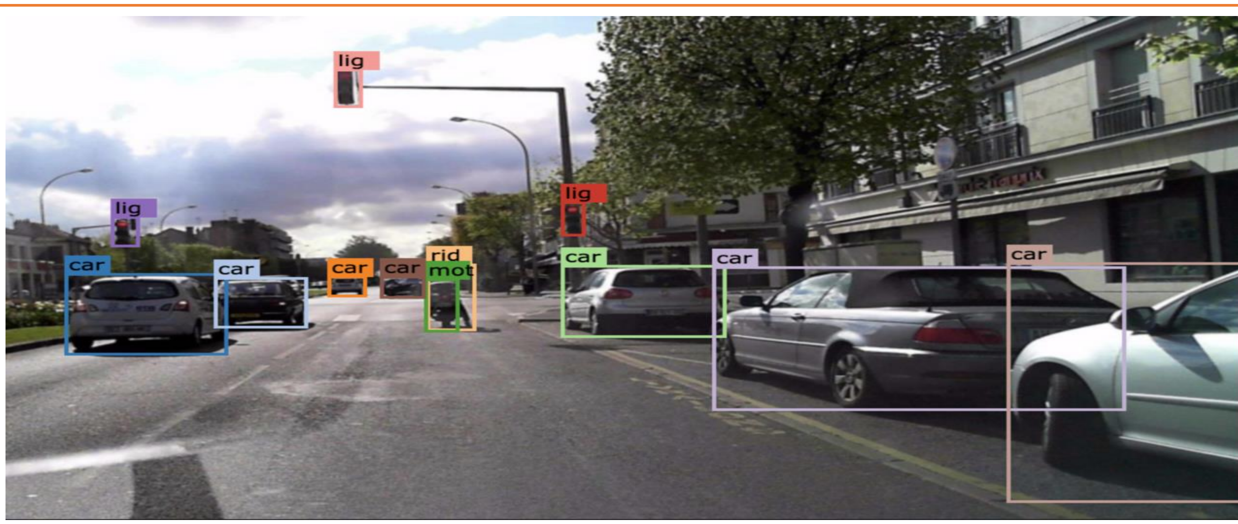


Figure 2.10: An example of multiple objects tracking with different classes [50].

2.4.1 Video Tracking Challenges

Tracking moving objects in video faces many challenges and issues that must be accounted before designing and operating tracking systems. Tracking challenges are divided into challenges that are related to the similarity between the object target and other objects in the video scene, and challenges that are related to the appearance changes of the object target.

Sometimes the appearance of the background and other objects in the scene is similar to the appearance of the target object. In such a case, figuring out frame features from non-target frame areas may be complex to recognize from the features that expect the target object to generate. This condition is referred to as clutter [8].

Video tracking faced another difficulty by changing the target object's appearance in a plane image because of one or more of the following factors:

- Changes in the pose: a moving target object differs from its appearance when it is projected on the image plane.
- Ambient illumination: the intensity, color, and direction of the ambient light affect the appearance of the target object. Moreover, varies in global illumination is always an issue in outdoor scenes.
- Noise: the image capture process could introduce a specific degree of noise, that replies to the quality of the camera sensor.
- Occlusions: object target could not be recognized when totally or partially occluded by other objects. Occlusions are often occurred because of either target object moving behind a fixed object like a wall, column, or desk or another moving object in the scene occluding the view area of a target object.

2.4.2 Object Tracking Approaches

Large techniques for object tracking have been suggested. Each technique is dependent on the way of its work where accurate tracking and robust are the main factors to evaluate each technique. Object tracking approaches are divided into two categories: classical and deep-learning approaches. The classical methods don't use any neural network to extract the features and tracking estimation states x, y, w, h where x and y represent the center of bounding box of the target object, and h is the height and w is the width of the predicted the bounding box. The classical techniques may include template matching, density estimation (Mean-Shift), regression, motion estimation, Kalman filtering, particle filtering, and silhouette tracking [51].

Deep-learning tracking methods work by the concept of tracking by detection which means they depend on the strength of the object detection algorithms. They used Deep Neural Networks (DNN) to extract the rich and complex features of target object from their input. Convolutional Neural Network (CNN) is currently used to extract spatial pattern data. Deep-learning tracking approaches are Multi-Domain Net (MDNet), GOTURN, ROLO (Recurrent YOLO), Deep-SORT, SiamMask etc. In this dissertation, the Deep-SORT tracking algorithm is used to track multiple objects with different classes. Deep-SORT is a modern approach with high accuracy and robustness [52].

2.4.3 Simple Online and Realtime Tracking with a Deep Association Metric (Deep-SORT):

Simple Online and Realtime Tracking (SORT) [53] is a realistic approach to tracking multiple objects with an emphasis on simple and effective algorithms. Deep-SORT is an updated version of the SORT approach where the researchers integrated the appearance information with the motion to evolve the performance of the SROT algorithm. Since this update, the Deep-SORT is able to track multiple objects during longer periods of occlusions, and it is effectively diminishing the number of identity switches. The heart of the new framework is to place much of the complicated computations into an offline pretraining phase where it learns a deep association measure on a huge re-identification dataset. Experimental illustrates that Deep-SORT reduces the number IDs switches by approximately 45%. Figure 2.11 shows an example of running the Deep-SORT algorithm to track multiple objects (persons)[54]

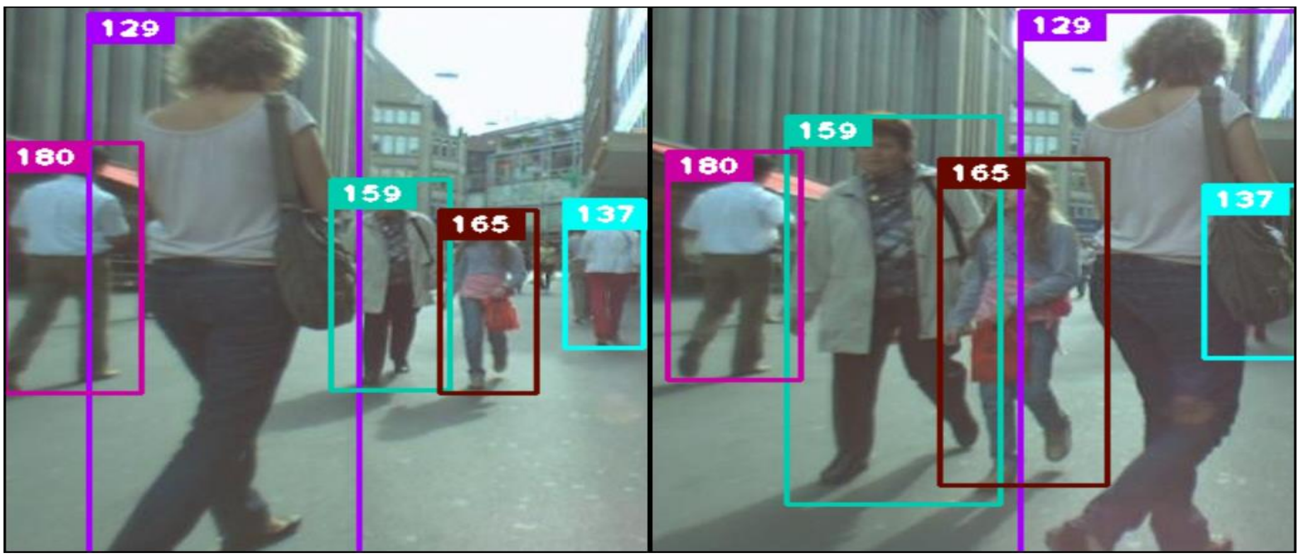


Figure 2.11: Output example for using Deep-SORT algorithm MOT [54].

2.4.3.1 Track Handling and State Estimation

The researchers adopt traditional single hypothesis tracking status with Kalman filtering estimation and frame-by-frame association data. The authors assume a general tracking way where it used uncalibrated cameras, and there is no motion information. The tracking scenario defined eight-dimensional states space (u, v, γ, h, x', y', γ', h') which has the bounding box center location (u, v), γ is an aspect ratio, h is height, and others are velocities in image coordinates. Then it is used a standard Kalman-filter with fix velocity motion and observation linear model, so it takes the bounding box coordinates (u, v, γ, h) observations of the object state directly [54].

For each k track, it counts the number of frames from the last successful association a_k. The counter is increased through Kalman filter prediction and becomes zero if the track is starting associated with measurement. If the tracks exceeded the predefined max-age, they will be considered to leave the scene, and they will be removed from the track set. A new hypothesis track is initialized for each detection process which is not able to associate with an existing track. These new tracks are grouped as unconfirmed through the first 3 frames. At this time, successful measurement of association could be seen at each time step. Indeed, tracks which aren't associated with measurement within the first 3 frames will be deleted [54].

2.4.3.2 Assignment Problem

The traditional how to address the Kalman filter correlation states predicted and new measurement is to construct an assignment issue which can be solved using the Hungarian approach. To formulate this issue, appearance information is integrated with motion during the combination of two suitable metrics. To perform motion information, the projected Kalman will illustrate distance squared by the Mahalanobis distance filter states space and newly came measurements as shown in Equation 2.7 [54]:

$$d^{(1)}(I, j) = (d_j - y_i)^T \, S_i^{-1} (d_j - y_i) \ldots\ldots\ldots\ldots (2.7)$$

where i-th track spread into measurement space (y_i, S_i), and the j-th is the detection of the bounding box by d_j. In addition, using this measure is possible to eliminate unsuccessfully associations with a 95% confidence score thresholding of the Mahalanobis distance that is computed from the inverse X^2 distribution. This decision indicator is shown in Equation 2.8 [54]:

$$b_{i,j}^{(1)} = 1\,[d^{(1)}\,(i, j) \leq t^{(1)}] \ldots\ldots\ldots\ldots\ldots\ldots (2.8)$$

If the association is true, the indicator will be equal to 1 between tracker i-th and detector j-th is acceptable. Where b_i and b_j represent the truth and predict detection, and t is the number of tracks.

The Mahalanobis distance is an appropriate metric when unreliable motion is low. An uncounted motion camera can perform speedy displacement in the plane image, which will make the Mahalanobis distance showing awareness measure for tracking through longer periods of occlusion. Therefore, the authors suggested combining another metric into the assignment issue. The smallest cosine distance between the track of i-th and j-th detection is the second metric measure to solve the problem that occurred when only Mahalanobis distance as illustrated in Equation 2.9 [54]:

$$d^{(2)}\,(i, j) = \min\,\{1 - r_j^T\, r_k^{(i)} \mid r_k^{(i)} \in R_i\} \ldots\ldots\ldots\ldots (2.9)$$

For each detection bounding box d_j, it will compute an appearance descriptor r_j where $\| r_j \| =1$. Moreover, it obtains the gallery $R_k = \{r_k^{(i)}\}_{k=1}^{Lk}$ of the last L_k=100 is associated with each track. Another way to perform a binary variable indicator if successful association is acceptable depending on this metric as shown in Equation 2.10 [54]:

$$b_{i,j}^{(2)} = 1\,[\,d^{(2)}\,(i, j) \;\leq t^{(2)}] \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.10)$$

Then the threshold figures out to this indicator on a different dataset. Practically, it applies pre-trained CNN to make a comparison of bounding box appearance descriptors. It uses the combination of Mahalanobis distance and the smallest cosine distance to integrate each other. Mahalanobis distance measure supplies information about object positions depending on the motion that is used for short-term predictions while the cosine metric performs appearance information that is used to retrieve identities after long-term occlusions when the motion is less descriptive. To construct the association problem by combining two metrics above using weighted sum as illustrated in Equation 2.11 [54]:

$$C_{i,j} = \lambda\, d^{(1)}(i, j) + (1 - \lambda)\, d^{(2)}(i, j) \quad \text{……….} \quad (2.11)$$

It is called an association acceptable if it is figured out from both metrics as shown in Equation 2.12 [54]:

$$b_{i,j} = \prod_{m=1}^{2} b_{i,j}^{(m)} \quad \text{……………………...} \quad (2.12)$$

In their experimental, the authors found that $\lambda = 0$ is a suitable choice if there is camera motion.

2.4.3.3 Matching Cascade

Instead of solving the issue of connections between measurement tracks in general assignment problems, a cascade is introduced which solves sequences of subproblems. When an object is obscured for longer periods, the next Kalman-filter predictions states will increase the unbelief associated with the object position. Therefore, the probability of the body seeps out into the state space and the observation become less than the maximum normal. If two tracks competes for the same detection, Mahalanobis distance will select the larger uncertainty track since it will be reduced the distance in standard deviations of detection through the period track mean. That means the matching cascade obtains high priority to objects that are frequently seen. In the end, it will apply intersection over union on the group of unmatched tracks that have age=1. This will assist to account for the abrupt changes suddenly such as partial occlusion with a static scene in addition to increasing the robustness and performance [54]. Algorithm 2.1 illustrates the matching cascade algorithm, in detail.

Algorithm 2.1: Matching Cascade algorithm [54].

Algorithm Name: Matching Cascade Algorithm

Input: Track indices T = {1, …, N}, Detection Indices D = {1, …, M}, Max age A_{max}

Output: Matched and Unmatched Tracks and Detections

Begin

Step1: Compute the cost matrix of C = $[c_{i,j}]$ …Eq. (2.11)

Step2: Compute the cost matrix of B = $[b_{i,j}]$ …Eq. (2.12)

Step3: Create set of matches tracks M=0

Step4: Create set of unmatched detection U=D

Step5: For n ∈ {1, …, Amax} do

Step6: Choose tracks by their age $T_n = \{i \in T \mid a_i = n\}$

Step7: $[x_{i,j}]$ = Minimum_Cost_Matching (C, T_n, U)

Step8: $M = M \cup \{(i, j) \mid b_{i,j} . x_{i,j} > 0\}$

Step9: $U = U \setminus \{j \mid \sum_i b_{i,j} . x_{i,j} > 0\}$

Step10: end for

Step11: Return (Matched, Unmatched)

End

2.4.3.4 Deep Appearance Descriptor

Because of using simple neighbour nearest queries and no additional learning measure, effective application of the researcher's approach needs a good descriptive embedding feature to be trained offline before the tracking online application. To solve this issue, Deep-SORT released a CNN that trained onto a large-scale re-identification dataset of more than 1 million images and 1,261 pedestrians; it will be suitable for deep learning metrics. The convolutional neural network is shown in Table 2.2, and it released a wide residual network and two layers followed by 6 residual blocks. The dimensionality 128 of global map feature is computed with 10 dense layers. The final batch and $\mathscr{L}_2$ normalization perform features in hypersphere to be appropriate with cosine appearance distance metric. Indeed, the network has 2,800,864 factors and 32 bounding box takes about 30 milliseconds [54].

Table 2.2: Overview of the CNN Architecture (Deep-SORT tracking algorithm) [54].

Name	Patch Size	Output Size
Conv 1	3 x 3/1	32 x 128 x 64
Conv 2	3 x 3/1	32 x 128 x 64
Max Pool 3	3 x 3/2	32 x 64 x 32
Residual 4	3 x 3/1	32 x 64 x 32
Residual 5	3 x 3/1	32 x 64 x 32
Residual 6	3 x 3/2	64 x 32 x 16
Residual 7	3 x 3/1	64 x 32 x 16
Residual 8	3 x 3/2	128 x 16 x 8
Residual 9	3 x 3/1	128 x 16 x 8
Dense 10	------	128
Batch & $\mathscr{L}_2$ Normalization	------	128

2.5 Moving Objects Trajectory Construction

An object moves from one frame (still image) to another sequence frame to imagine how it moves smoothly in a way that is felt by the eye as if it was an integrated video. Geometrically, an object takes a specific location into the image plane, and its location is updated in each consecutive frame based on its moving distance. The line that is constructed by concatenating multiple sequence points is called a trajectory. A trajectory is the path of a moving object that follows across the space with the time function. It is denoted by $\{(x_1, y_1, t_1), (x_2, y_2, t_2), \ldots, (x_N, y_N, t_N)\}$ where x_i, y_i represent geographic locations of an object during the t_i time and N is the total number of frames that object moves through them. The path that is created by connecting several points together to indicate an object's changing location from the beginning to the end throughout that position is called a trajectory. For each item in each frame, the X- and Y-centers must be determined in order to determine the trajectory that is associated with the time stamp. Figure 2.12 shows how the trajectory is generated from the digital video camera frame by frame with time for each position frame as illustrated in Equation 2.13 [55].

$$\text{Trajectory} = T\,(\text{Xcenter}, \text{Ycenter}, t) \;\ldots\ldots\ldots\; (2.13)$$

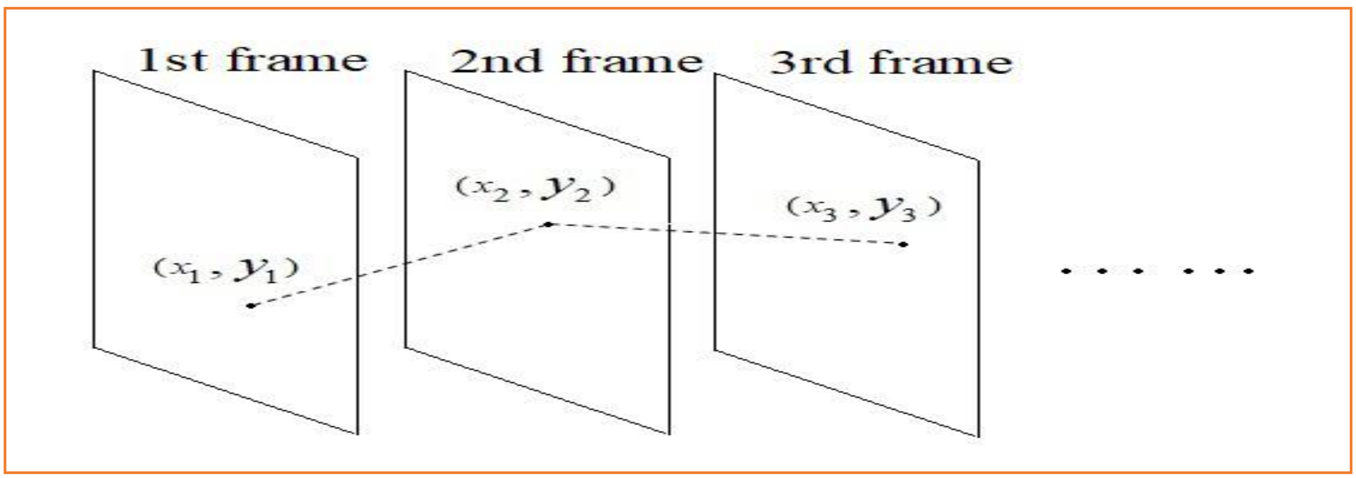


Figure 2.12: Trajectory Constructed by Video Camera Device [55].

There are two main types of trajectories either geographic or semantic. The geographic (Spatial) type is based on their locations with time association where the location represents the center position of an object in a specific frame with a time stamp (frame per second). On the other hand, the semantic type is to extract the interesting point for the trajectory depending on the semantic information on a specific location that obtains by GPS (Global Positioning System) or by using computer vision open-source applications [56].

Trajectories have different lengths, directions, and shapes where each trajectory has special attributes that distinguish it from others. However, many properties are considered to be similarity factors to compare the trajectories. Figure 2.13 illustrates trajectories with varying lengths [57].

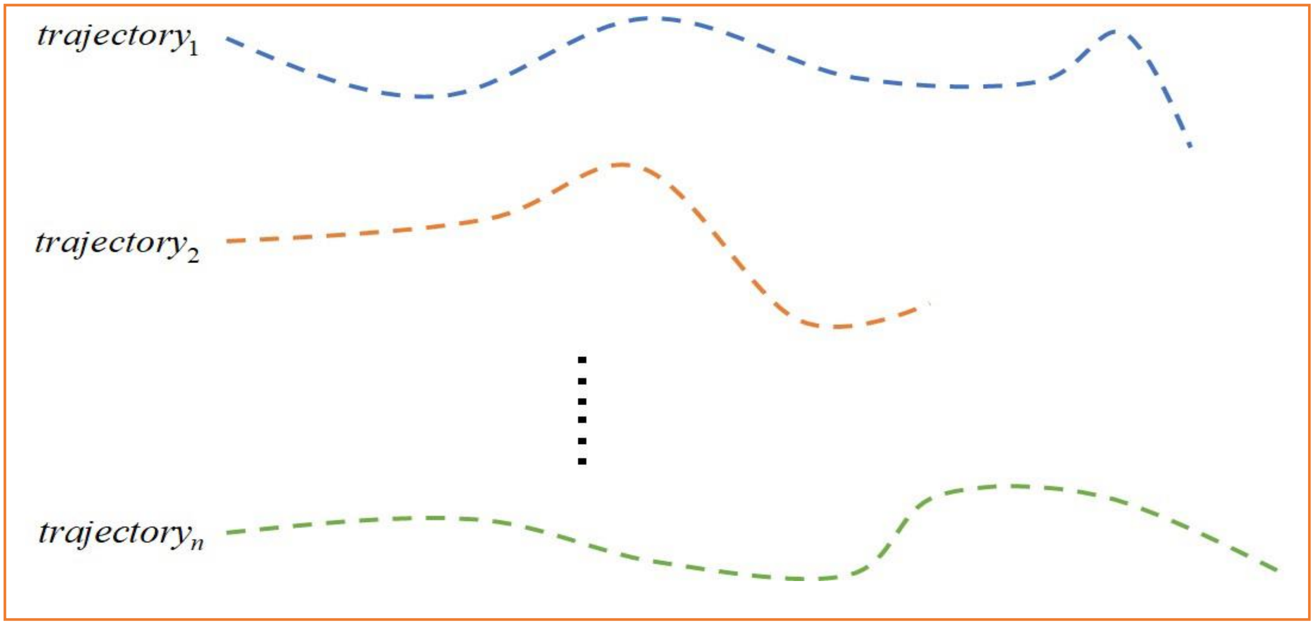


Figure 2.13: Trajectories with different lengths [57].

There are many similarity measures to ensure that the object trajectory is closed to others in the scene. The similarity may be measured by whole trajectory and sometime by sub-trajectory that depends on the method that is used to compute the similarity and the type of the dataset. Figure 2.14 shows different measures of trajectories with many attributes [58].

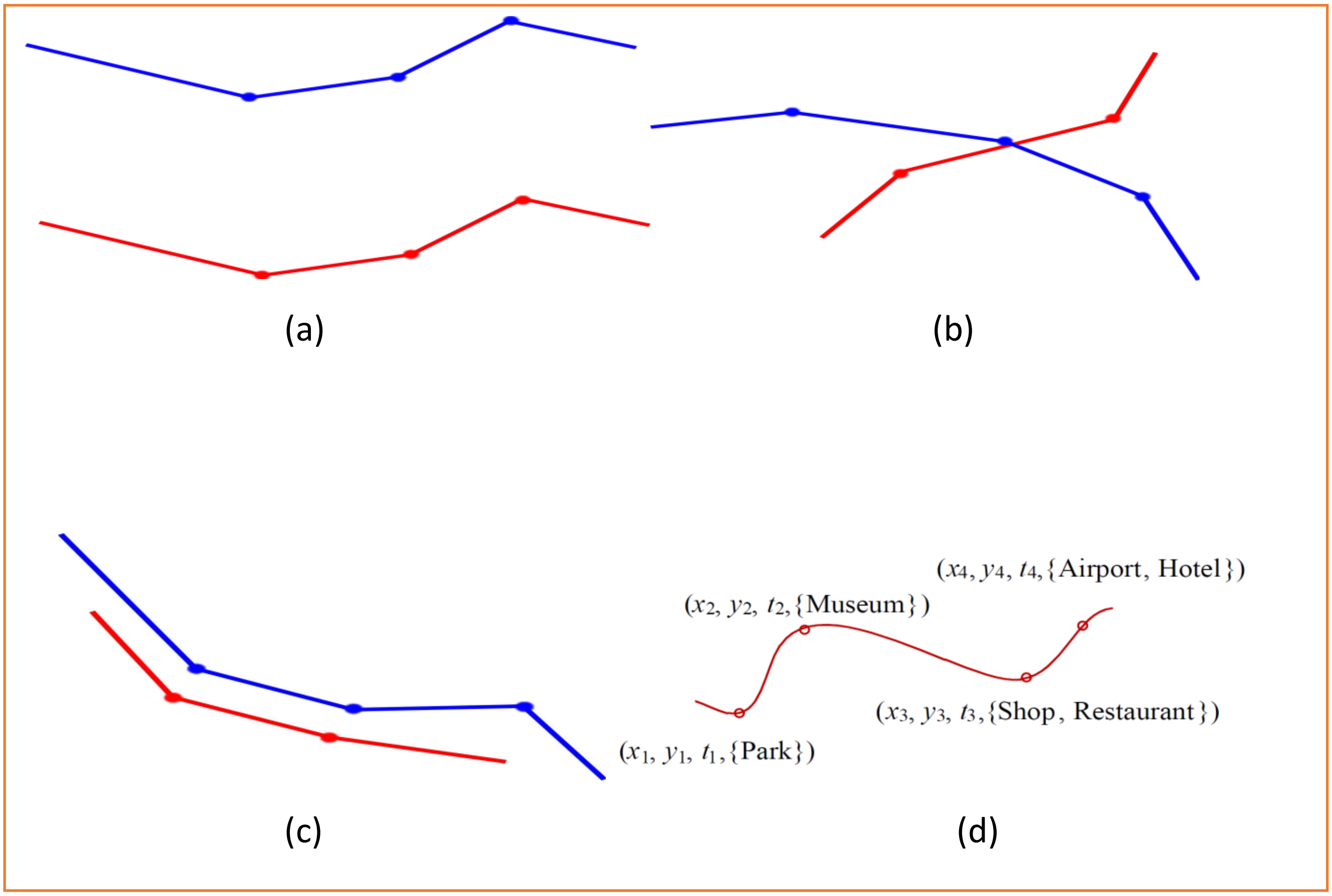


Figure 2.14: (a) Two trajectories of similar orientation and shape. (b) Two overlap trajectories with different orientations. (c) Two trajectories of different lengths but the same orientation. (d) Semantic trajectory [58].

There are two main types of similarity measures: spatial similarity which emphases spatial geometric shape without temporal, and the second type measure is a spatio-temporal similarity which considers the spatial and temporal aspects of the moving data object. Figure 2.15 expresses the classification of similarity methods between the trajectories in different measures [59], [60].

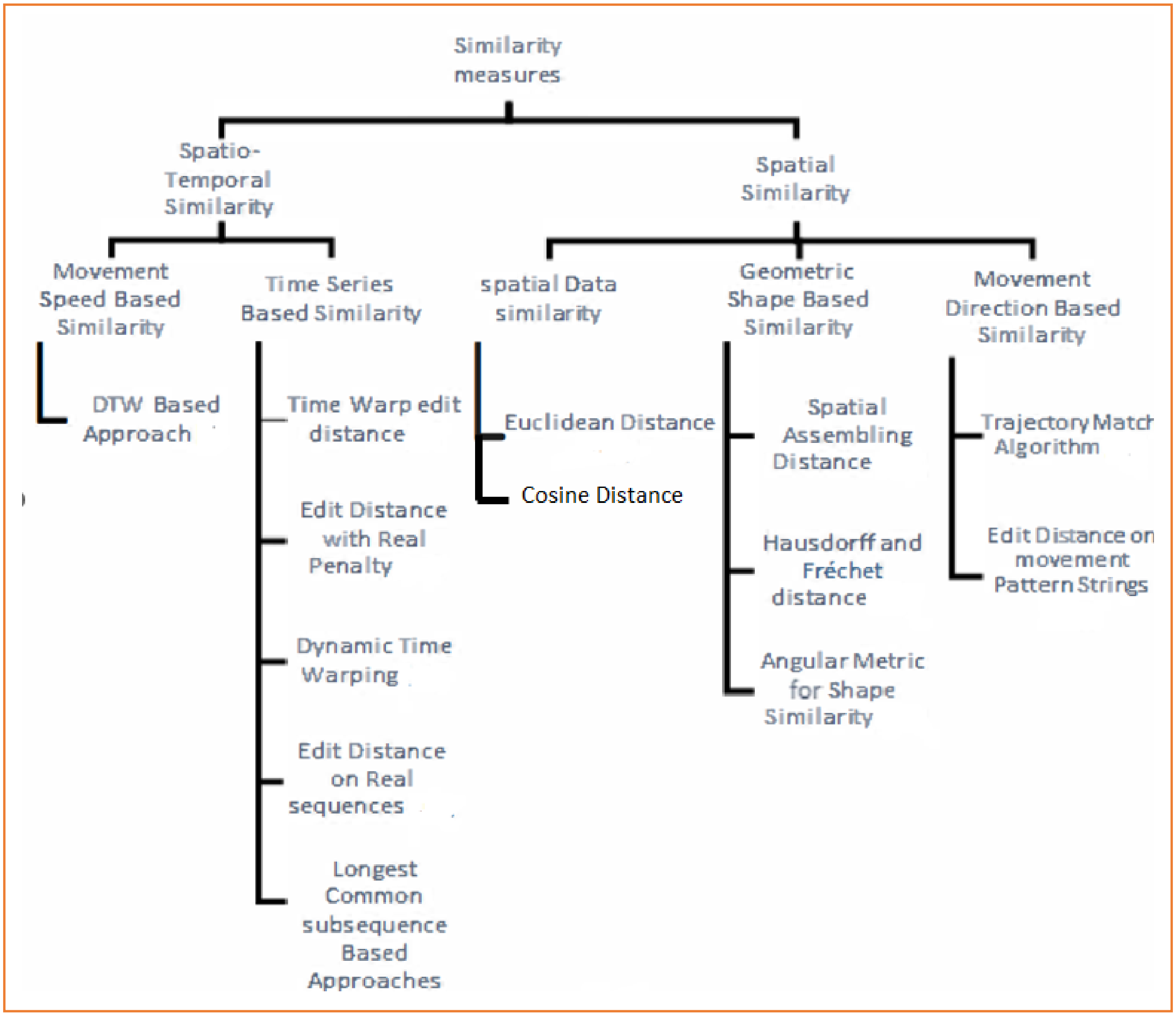


Figure 2.15: Classification of Similarity Measures of Trajectories [59].

2.6 Shape Normalization and Reduction

Automatic object recognition is one of computer vision's most crucial objectives. Based on a variety of sensory information, including shape, color, and texture, the human visual system accomplishes this task. The shape is one of the most helpful cues among these inputs because color and texture can quickly change within the same class of items, such as cars or pedestrians, but the shape is largely constant. However, there is some variation in the forms of objects even within the same class. For instance, the object can be nonrigid or visible from several angles. As a result, normalizing shapes becomes a crucial stage in creating shape models.

Shape normalizations are made invariant under a specific class of transformations by the process of shape-normalization. As a result, there are two ways to categorize existing shape normalizing methods: (i) shape representation and (ii) transformations. Different methods of representation, such as landmarks like the Procrustes analysis, parametric curves or surfaces like splines, or implicit functions like the distance transformations, have all been proposed. In terms of transformations, most people think of translation, isotropic scaling, and rotation as similarity transformations. Therefore, finding the shape's center, scale factor, and orientation will enable shape normalization to achieve its special purpose of making the shape model invariant concerning the three transformations. Figure 2.16 illustrates an example of a shape normalization mechanism [61].

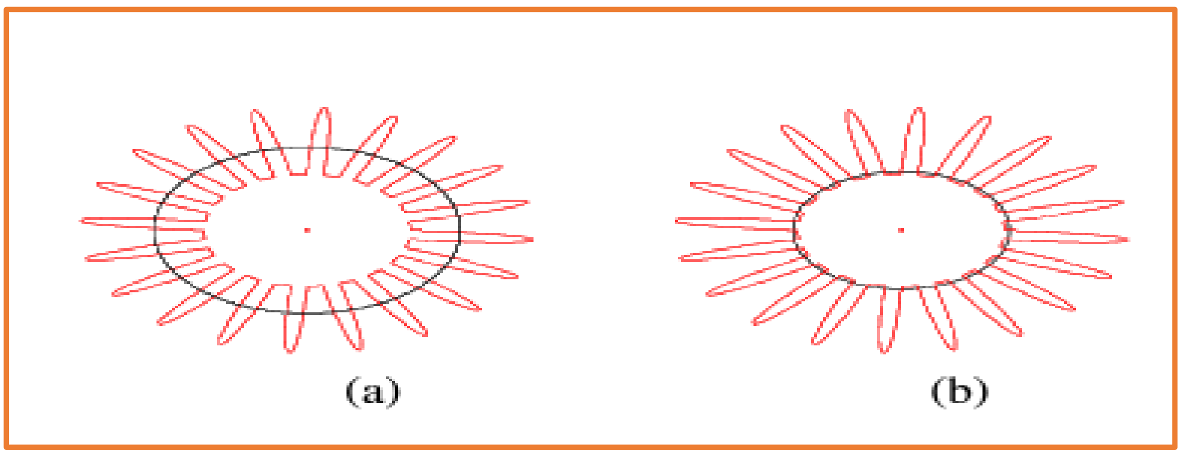


Figure 2.16: a) Object before shape normalize; b) Object after shape normalize [61].

2.6.1 Implicit Shape Representations

Implicit Shape Representation is defined as a signed distance function by saying the following given a closed planar curve C that defines a shape S as shown in Equation 2.14 [61]:

$$\phi\,(x, y) = d\left((x, y), C\right) \quad inside\ C$$

$$-d\,((x, y), C) \quad if\,(x, y)\ outside\ C$$

$$0 \quad if\,(x, y) \in C \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(2.14)$$

where the minimal Euclidean distance (d((x,y),C)) exists between the point (x,y) and the curve (C). As a result, (x,y) embeds C as its zero-level set and has a gradient according to the unit norm.

2.6.2 Robust Shape Normalization

Robust Shape Normalization is a novel shape-normalizing technique based on implicit representations to lessen the sensitivity to shape deformations. Designing a density function $\rho(x, y)$for a form based on its implicit representation φ as illustrated in Equation 2.15 [61]:

$$\rho(x,y) = \varphi\lambda(x,y)\ (\lambda = 1,2,...)\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.15)$$

and after that, change the shape's center, scale factor, and orientation are redefined. The level of robustness is controlled by the parameter λ. The shape normalization is less susceptible to shape deformations as grows larger λ.

2.6.3 Ramer–Douglas–Peucker Algorithm

It is also known as the Douglas–Peucker algorithm and iterative end-point fit algorithm. It is a mathematical algorithm that reduces a curve made up of line segments to a curve with fewer points. It was one of the first methods for cartographic generalization to be effective. The algorithm's goal is to identify a similar curve with fewer points given a curve made up of line segments (also known as a polyline in some settings). Based on the greatest distance between the original curve and the simplified curve, the algorithm determines what is "dissimilar" (i.e., the Hausdorff distance between the curves). A portion of the points that made up the original curve is presented in the simplified curve. Figure 2.17 shows an example of the Douglas–Peucker algorithm [62].

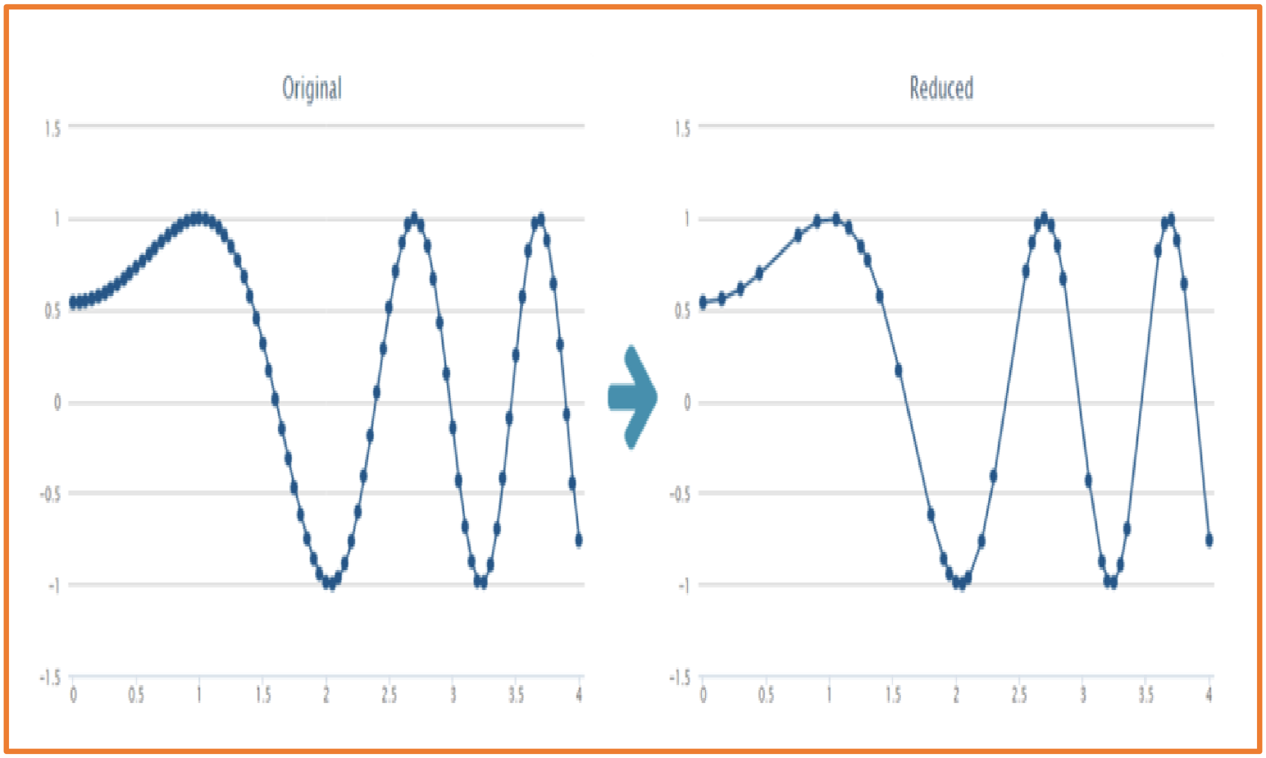


Figure 2.17: An example of the applied Douglas–Peucker algorithm [63].

The distance dimension $\varepsilon > 0$ and an ordered collection of points or lines make up the initial curve. The line is divided repeatedly by the algorithm. All of the points between the first and last points are initially awarded to it. The first and last points are automatically marked as being kept. The point that is farthest on the curve from the approximate line segment between the end points is found next. This point is obviously farthest from the line segment with the first and last points as end points. Any points not already designated to be kept can be deleted if the point is closer than ε to the line segment without worsening the simplified curve by more than. The farthest point from the line segment must be retained if its distance from the approximation is more than. The procedure, which includes the farthest point being marked as kept, recursively calls itself with the initial point and the farthest point, then with the farthest point and the last point. When the recursion is finished, a new output curve made up entirely of the points that were designated as maintained can be created [62] [64].

2.7 Graph Mining: Concepts and Techniques

Data mining is automatic processing that seeks meaningful information among huge data repositories, or it is discovering patterns in massive data sets. Graph mining is a branch of the data mining concept. Graph mining is the tools and techniques which are used to analyze the attributes of real-world graphs; to anticipate how the graph structure and its properties may influence some applications and to develop models capable of producing accurate graphs like interest patterns figured out from real-world graphs. A graph is used to represent large data with high accuracy of the description of their items which is an important reason to use a graph. Since there are many types of graph patterns, frequent pattern mining is the basic pattern which could be discovered in a group of graphs [65].

2.7.1 Basics and Definition

a. Graph is a set of nodes that are related by edges. A graph is defined as G= (V, E), where V is a set of nodes and E is a set of edges that clarify the relations between nodes. A graph can be directed or undirected, also it could be weighted on an unweighted graph based on the problem representation. Figure 2.18 shows two mix types of graphs [66].

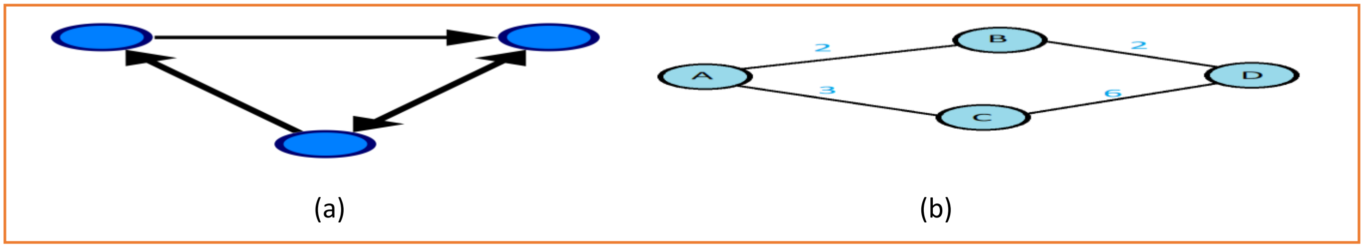


Figure 2.18 :(a) Unweighted Directed Graph of three nodes. (b)Weighted Directed Graph of four nodes [66].

b. Graph Isomorphism: the graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are **isomorphism** if the vertices of two graphs are the same while the edges of each of them is resorted. In other words, graph isomorphism can be defined as a one-to-one mapping between the nodes of V_1 and V_2 which obtains adjacency. Figure (2.19) illustrates an example of graph isomorphism aspect. Graph isomorphism is a bijection between two graphs and denoted by [67]: $f: V(G_1) \longrightarrow V(G_2)$

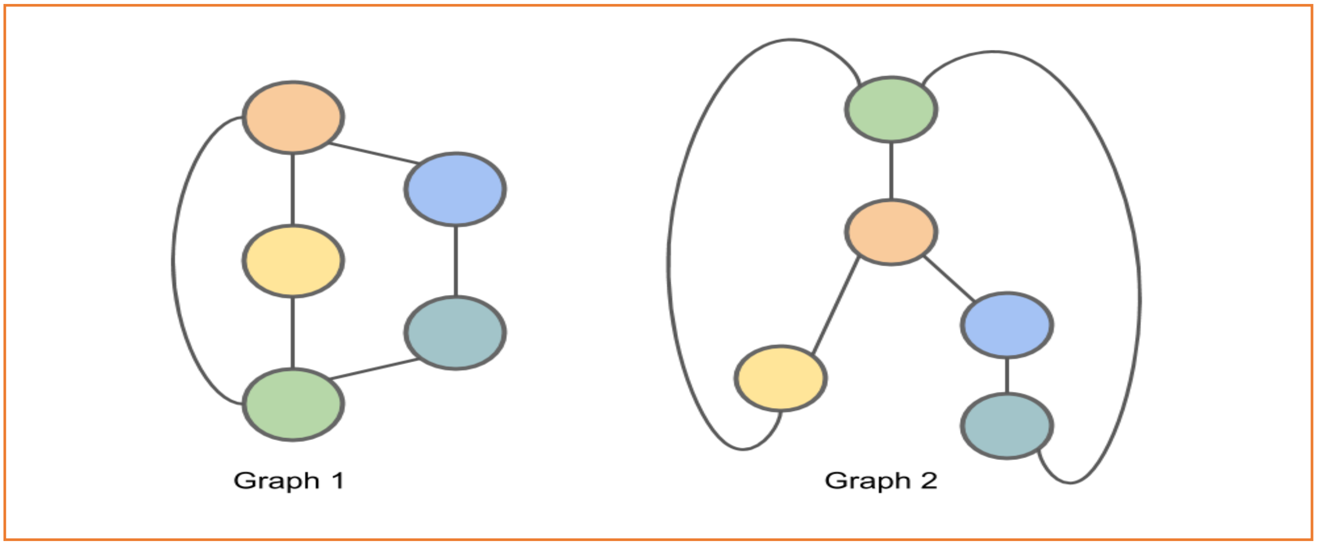


Figure (2.19): An example of two graphs (Graph 1, Graph 2) isomorphism [67].

c. Adjacency List: is a group of unsorted lists that used to represent a graph. It is an array of size that is equal to the number of vertices. The advantages to using adjacency list are time complexity O (|V| + |E|), and it is easier to add a new node. However, queries about which edge from node u to node v aren't efficient O (V). In general, an adjacency list is used with graphs that are sparse [66]. Figure 2.20 shows an example of an adjacency list by representing the directed graph.

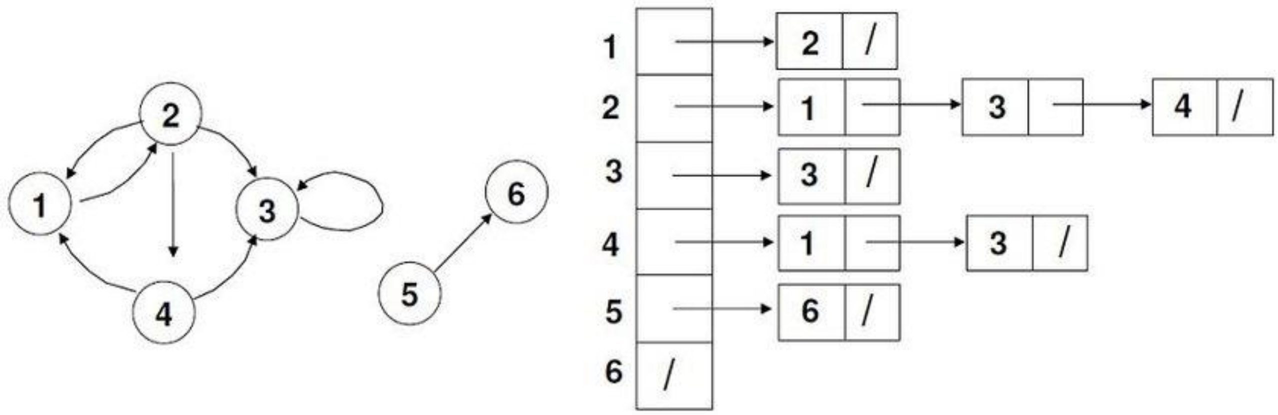
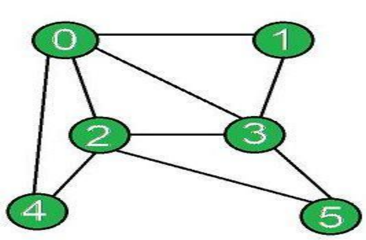


Figure 2.20: An example of represented of a directed graph with its adjacency list [66].

d. Adjacency Matrix: another important representation of a graph. It is a square array that is used to represent a limited graph. The values of the array are (0, 1) with 0 for the main diagonal. The adjacency matrix is symmetric because the upper triangle of the array is equal to the lower triangle pf the array. The advantages of the adjacency matrix are easier to eliminate an edge where it takes O(1) time, representation data is simple, queries about ab edge from u to v are efficient, and it consumes only O(1). On the other hand, the drawbacks of the adjacency matrix that it consumes more space, the same space is also consumed even if the graph is sparser [66]. Figure 2.21 shows an example of the adjacency matrix representation of a graph.



	0	1	2	3	4	5
0	0	1	1	1	1	0
1	1	0	0	1	0	0
2	1	0	0	1	1	1
3	1	1	1	0	0	1
4	1	0	1	0	0	0
5	0	0	1	1	0	0

Figure 2.21: An example of an undirected graph and its adjacency matrix representation.

2.7.2 Graph Mining Approaches

There are two main types of graph mining algorithms:

a. **A priori-based approach**: In the graph data structure, all of the frequently recurring sub-graph patterns can be successfully extracted using the priority-based graph mining (AGM) algorithm. The breadth-first search (BFS) idea, an algorithm for searching graph data structures, serves as the foundation for this approach. Before going to the following level, it investigates all of the nearby vertices of the current level, beginning from the root of the tree (or some other random node on the graph). For instance, let's say we have two sets of size-3 frequent elements (abc) and (bcd). They produce simple (abcd) frequent element set candidates of size, and because the two substructures are related, the candidate size-4 generation problem in the frequent substructure mining is more challenging than the candidate generation problem in frequent element set mining. Recently, substructure extraction from priority-based algorithms such as AGM, FSG, and path-binding approaches has become rather popular [68], [69]. Figure 2.22 illustrates the mechanism of Apriori likewise approach to search of subgraphs pattern.

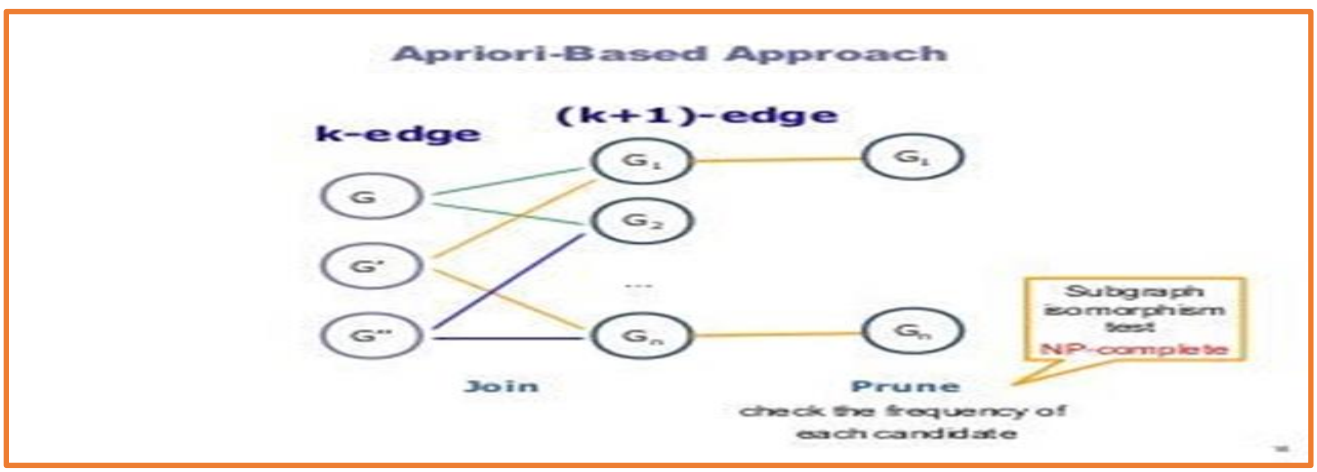


Figure 2.22: A priori-based approach of graph miming search [68].

b. **Pattern-Growth Approach**: The strategy (BFS), which is a step-by-step candidate build, must be used by the priority-based method. Check all of the k-size subgraphs that correspond to the (k + 1) size chart to determine its upper bound of frequency before deciding if it is common. Therefore, the A priori-based technique must typically finish the extraction of subgraphs of size k before extracting subgraphs of any size (k + 1). The pattern growth technique, however, employs deep search (DFS), and its search technology is more adaptable. The DFS search method begins at the root node or any other randomly chosen node and scans as much of each branch as it can before moving on to deeper levels of scanning. The frequent pattern growth approach is a technique for identifying common patterns; it does not produce new candidate patterns. Instead of using the A priori build and test technique, construct an FP tree. The FP growth algorithm's main goal is to distinguish between the trajectories of different pieces and extract recurring patterns. Pattern growth techniques like gSpan, ADIMine, and DPMin are examples [68], [69]. Figure 2.23 expresses the mechanism of pattern-growth (tree extension) search technique.

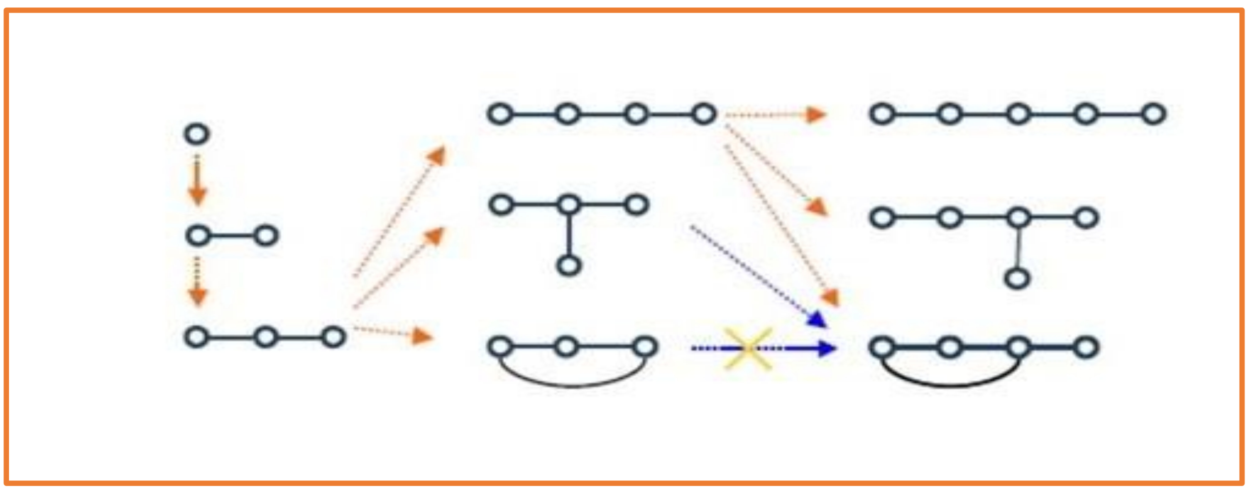


Figure 2.23: Pattern-Growth search approach mechanism [68].

2.8 Clustering

Clustering is the process of grouping set of objects that have approximately similar attributes, and they are homogeneous, connectivity, meaningful, and useful. Clustering is a branch of data miming concept because it focuses on the aspect of mining the interesting data based on the algorithms that are followed. There are many types of clustering hierarchical and partitional. Partitional clustering is dividing the set of data objects into non-overlapped clusters (subsets) where each object must belong to a specific subset. Hierarchal clustering is allowing clusters to contain subclusters, where a set of overlapped clusters are sorted as a tree. In addition, each type of clustering approach has two subtypes where hierarchical clustering has divisive and agglomerative approaches while partitional clustering has hard and soft clustering approaches. Figure 2.24 shows the difference between hierarchical and partitional clustering in mechanism [70], [71].

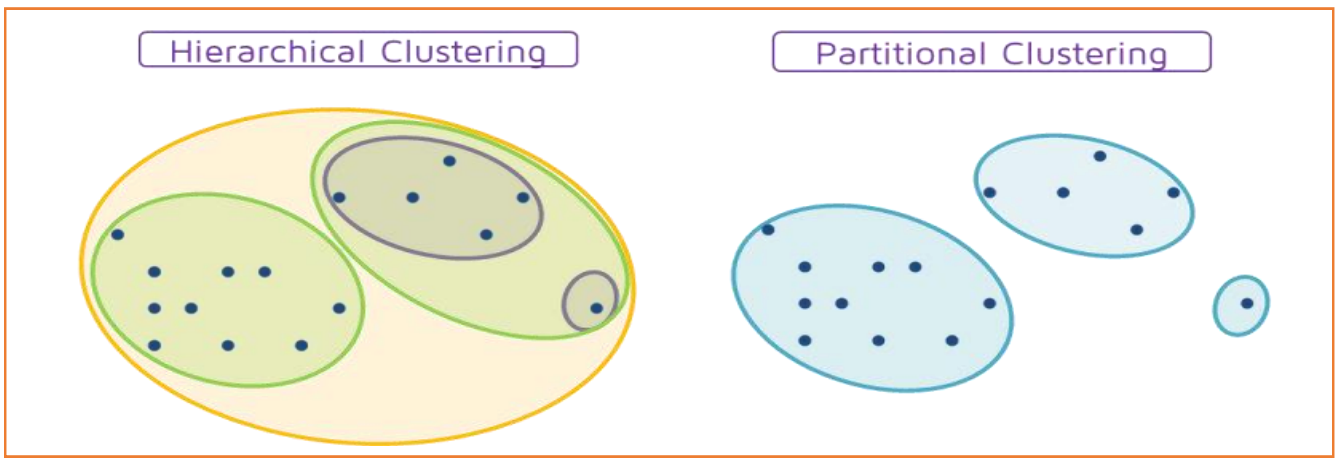


Figure 2.24: Hierarchical and partitional clustering [70].

2.8.1 Clustering Models

The concept of "cluster" can be found in many algorithms, and it is a variety important in their attributes. Therefore, there are different models and algorithms, and each of these models can be given a different algorithm [71]:

- Connectivity Models: constructs models depended on distance connectivity such as hierarchal clustering.
- Centroid Models: acts each cluster as a single average vector such as the K-means algorithm.
- Distribution Models: a set of clusters are modeled by statistical distribution like an expectation-maximization algorithm.
- Density Models: declares the clusters as a region densely connected in space of data such as DBSCAN.
- Group Models: sometimes there is no derived model for some results, so it just supplied the grouping information.
- Graph-Based Models: it happens in subsets of nodes from a graph where every two nodes are connected by an edge that can be conducted form of a cluster (Clique).
- Neural Models: it is called an "unsupervised neural network". It can be similar to the above model just in subspace models if a neural network is implemented aspect of PCA and ICA.
- Subspace Models: it is called bi-clustering or two-mode -cluster, and it is modeled with both cluster items and pertinent properties.

2.8.2 Hard and Soft Clustering

In general, clustering can be divided into two types of clustering methods: hard and soft clustering. Hard clustering is defined as assigning object data to be in just one of the clusters. For example, if we want to read all Facebook posts then we want to separate the posts into a negative or positive post. K-Means is a popular hard clustering algorithm to cluster where the data objects are clustered into K clusters, and each object will belong to only one cluster [72].

In some cases, it doesn’t need a binary separation since the data nature types are different, and it needs to cluster the data into clusters where each object could belong to multiple clusters based on a certain degree. Fuzzy C-means (FCM) is an important algorithm for soft clustering where a membership vector is generated through FCM progress that indicates the probability of the membership. the ranging values of the membership vector are in between 0 to 1 to refer to how the object belongs to one cluster and with others [73]. Figure 2.25 shows the difference between hard and soft clustering approaches.

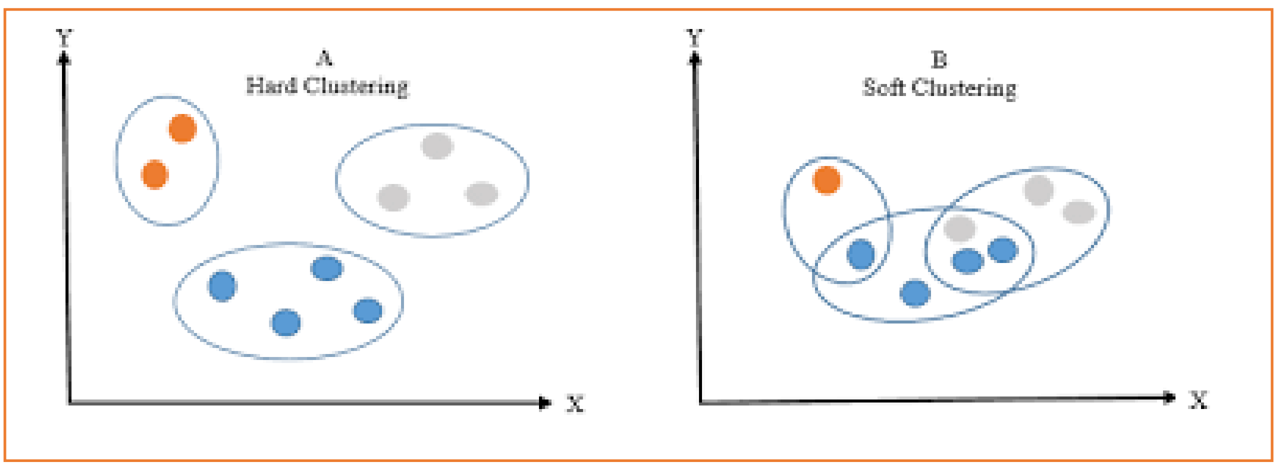


Figure 2.25: (A) Hard clustering method. (B) Soft clustering method [74].

2.8.3 Fuzzy C-Means Clustering Algorithm

FCM clustering algorithm was evolved by J.C Dunn in 1973 and enhanced by J.C. Bezdek in 1981 [74]. It is very similar to the K-Means algorithm where:

a- Select a number of clusters.
b- Randomly assign coefficients to a data item to be clustered.
c- The algorithm will be repeated until it is converged (when the coefficients change between two iterations is less or equal to £ sensitivity threshold.
    1- Calculate the centroid of each cluster.
    2- Calculate the coefficients of the clusters for each data point.

1) Centroid

For each point x, it has a group of coefficients that obtain the degree of being in the kth cluster of $w_k(x)$. therefore, the fuzzy c-means centroid of a cluster is representing the average of all points and is weighted to the cluster belonging mathematically as shown in Equation 2.17 [74]:

$$c_k = \frac{\Sigma_x w_k(x)^m x}{\Sigma_x w_k(x)^m} \;\ldots\ldots\ldots\ldots\; (2.17)$$

Where m is the hyperparameter that manages the form of the fuzzy cluster where the higher m, the fuzzier of the cluster will be at the end.

2) Algorithm

The FCM algorithm tries to divide a finite set of n elements where X = {x1,…, xn} into a set of c clusters (fuzzy) with some given condition. The algorithm will return a list of c centers of cluster C = {c1, … , cc} by releasing a finite group of data, and partition array W = wi,j ∈ [0,1], i=1,...,n, j=1,...c, each element wi,j called the degree of element x_i belongs to cluster $c_{j.}$ The FCM reaches to minimize an objective function as illustrated in Equation 2.18 [74]:

$$\text{Arg Min (c)} \sum_{i=1}^{n} \sum_{j=1}^{C} w_{ij}^{m} \left\| x_i - C_j \right\|^2 \quad \text{……….} \ (2.18)$$

where:

$$w_{ij} = \frac{1}{\sum_{k=1}^{c} \left(\frac{\left\| x_i - c_j \right\|}{\left\| x_i - c_k \right\|} \right)^{\frac{2}{m-1}}} \quad \text{…………………..} \ (2.19)$$

Algorithm 2.2 illustrates the fuzzy c-means algorithm.

Algorithm 2.2: The Fuzzy C-Means Algorithm [74].

Algorithm Name: The Fuzzy C-Means Algorithm

Input: Number of Flatten Vectors that Represents Objects Trajectory Weighted Graphs After Data Reduction

Output: Number of Clusters

Begin

Step 1: Read csv file that contains number of vectors each one represents objects trajectory weighted graph after applying cosine similarity measure

Step 2: Determine Number of Clusters that should be gotten and investigation good outcomes

Step 3: Applying Fitting Function on Flatten Vectors and Store the Values

Step 4: Specifying the Centers of All Clusters

Step 5: Predict the output values by Applying Predicting c-means Function

Step 6: Go to step 1 if there is new file has been ready to input

End

2.9 Results Evaluation Measures

The results must be evaluated in any system to assess it, and sometimes makes changes and updates to improve the outcomes. There are many types of metrics to evaluate the stages of any system, and it must select the important and effective metrics to assess each stage.

2.9.1 Object Detection and Tracking Evaluation Metrics

All metrics that could be used to evaluate and compute the strenghth and the quality of the algorithm rely on the confusion matrix of the actual and prediction results [75].The confusion matrix of object detection and tracking assessment is illustrated in table 2.3 [76].

Table 2.3: Confusion matrix of object detection and tracking.

Total Population = P +N		Prediction Condition	
Actual Condition	Positive	Positive	Negative
		True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

True Positive (TP): when detecting and tracking the correct target object, and the intersection over union (IOU) between actual and prediction (bounding boxes) is more than or equal to 0.5 thresholds.

False Negative (FN): when there is no detecting and of tracking of any target objects, regardless the IOU between actual and prediction of bounding boxes.

False Positive (FP): it sets when detecting another target object but takes a different ID switch. Also, ignore the IOU between prediction and actual.

True Negative (TN): it sets when object detection is correct, but IOU is less than 0.5 threshold. For tracking, it sets when tracking a correct target object while it takes the wrong ID switch.

Then, any important and general metrics can be computed:

Accuracy: is a measure that describes the model applied across all object's classes. The formula for accuracy measure is shown in Equation 2.20 [77]:

$$\textbf{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \quad \ldots\ldots\ldots \quad (2.20)$$

Precision: is computed as a ratio between the number of correctly positive samples that is figured out the total number of correct and incorrect positive samples. It is measuring the accuracy model in the positive sample for actual and prediction as illustrated in Equation 2.21 [77]:

$$\textbf{Precision} = \frac{TP}{TP+FP} \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (2.21)$$

Recall: is computed as the ratio between the number of correctly positive samples which is figured out the positive samples to the total number of positives. It can be able to discover the positive samples where the higher value of the recall means more positive samples are discovered. The formula for recall measure is shown in Equation 2.22 [77]:

$$\textbf{Recall} = \frac{TP}{TP+FN} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (2.22)$$

F-Score: is computed from precision and recall. It is the harmonic average of the precision and recall. The formula of the F-score measure as shown in Equation 2.23 [77]:

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} \ldots (2.23)$$

2.9.2 Evaluation of Clustering Results

The main goal of clustering is to obtain high intra-cluster and low inter-cluster which is the underlying criterion for cluster quality. For clustering measures quality, there are internal and external measures of quality of clustering. A review of the important and effective measures for clustering data is highlighted [78]:

a. **Purity:** is an external clustering evaluation measure, and it is a metric to measure the quality of a cluster which contains only a single class on an object's data. The formula of purity cluster is illustrated by Equation 2.24 [79]:

$$Purity(\Omega, c) = \frac{1}{N} \sum_{k} \max_{J} \left| w_k \cap c_j \right| \ldots\ldots\ldots (2.24)$$

where:

$\Omega = \{w_1, w_2, \ldots, w_k\}$is the group of clusters, and c = $\{c_1, c_2, \ldots, c_j\}$ is a group of classes type.

b. **Rand Index**: is a metric to measure the similarity between two data clustering, it will be given learning of the ground truth class functions and clustering algorithm functions of the same samples. The formula of the Rand Index is shown in Equation 2.25 [80]:

$$RI = \frac{TP+TN}{TP+TN+FP+FN} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \text{(2.25)}$$

c. **Homogeneity:** is a clustering metric evaluation to measure if the cluster has only one single class of its members. The formula to calculate the homogeneity clustering measure is defined in Equation 2.26 [81]:

$$h = 1 - \frac{H(Ytrue|Ypred)}{H(true)} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots \text{(2.26)}$$

d. **Completeness**: is another metric clustering evaluation measure to compute all class members of a given class are grouped at the same cluster. The equation formula to compute the completeness measure is illustrated in Equation 2.26 [81]:

$$c = 1 - \frac{H(Ypred|Ytrue)}{H(pred)} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \text{(2.26)}$$

e. **V-measure (Normalized Mutual Information)**: is obtaining the goodness of clustering algorithm, and it is the harmonic average between homogeneity and completeness. The formula to calculate the V-measure is defined by Equation 2.27 [81]:

$$v = \frac{\left((1-\beta)*homogeneity*completeness\right)}{\beta*homogeneity+completeness} \quad \ldots\ldots\ldots\ldots\ldots\ldots \text{(2.27)}$$

where the default β is 1, and it is ranged between 0-1.

Chapter Three: The Proposed System

3.1 Overview

The main objective of this dissertation is to cluster multiple objects tracking by using modified YOLO and Deep-Sort algorithms with many classes and extract trajectories and features for them. Then a graph mining algorithm is suggested to be a graph clustering algorithm to group the objects of the same class based on their behaviors. In this chapter, the proposed system is described in detail. The first stage is how to prepare and read the dataset. The second stage is to split a video into frames. The third stage is applying pre-processing on each frame before the detection stage begins when it needs. The fourth proposed stage is to detect multiple objects in each frame; after that tracking multiple objects algorithm is run. The fifth stage is to extract the trajectory of each object and some features of each object trajectory. The sixth stage is to store the graph information for each trajectory object in large graph database. The dissertation then proposes a new adaptive graph mining to figure out the interesting information for each trajectory object stored in huge database. After that, it applies a graph clustering algorithm to divide the data into groups of similar behaviors.

3.2 The Proposed System

The proposed system has many stages to cluster moving objects in video. The input system is the benchmark videos dataset. The dataset is VB100 Video Bird Dataset, and it contains 22 classes and about 14 video clips for each class [82], [83]. The next stage is to split video data into frames (still images) and then prepare them to be input into the preprocessing phase. Furthermore, preprocessing step is set of operations that are applied on frame (image) to refine it. Moreover, many operations can be applied to the frame before starting the clustering model such as morphology, filtering, etc.

In order to complete the preprocessing phase, each frame is inserted into the detection step to detect multiple objects with different classes. Here, modify YOLO (You Only Lock Once) algorithm is used for object detection because this algorithm is more accurate than others and robust. When the model starts detecting the objects, the Deep-Sort tracking algorithm will begin to track the objects from frame to frame and so on. Then, it will extract the trajectory for an object after making normalization for trajectory points. After that, it will figure out some features for every two points of the trajectory to prepare it for the next step. Next, by using the features between points, it can construct a graph for each trajectory. Therefore, converting each trajectory into a graph by using its features by representing it as a Fuzzy **A**djacency

Matrix (FAM) to act the graph. Moreover, a graph dataset is built by converting trajectories into a graph. In addition, an adaptive graph mining algorithm is proposed to mine and cluster these objects trajectories based on behaviors of the same object type. The outcomes are the number of clusters, and each cluster must have purity, and homogeneity, completeness for their elements. Figure 3.1 illustrates the block diagram of the proposed system.

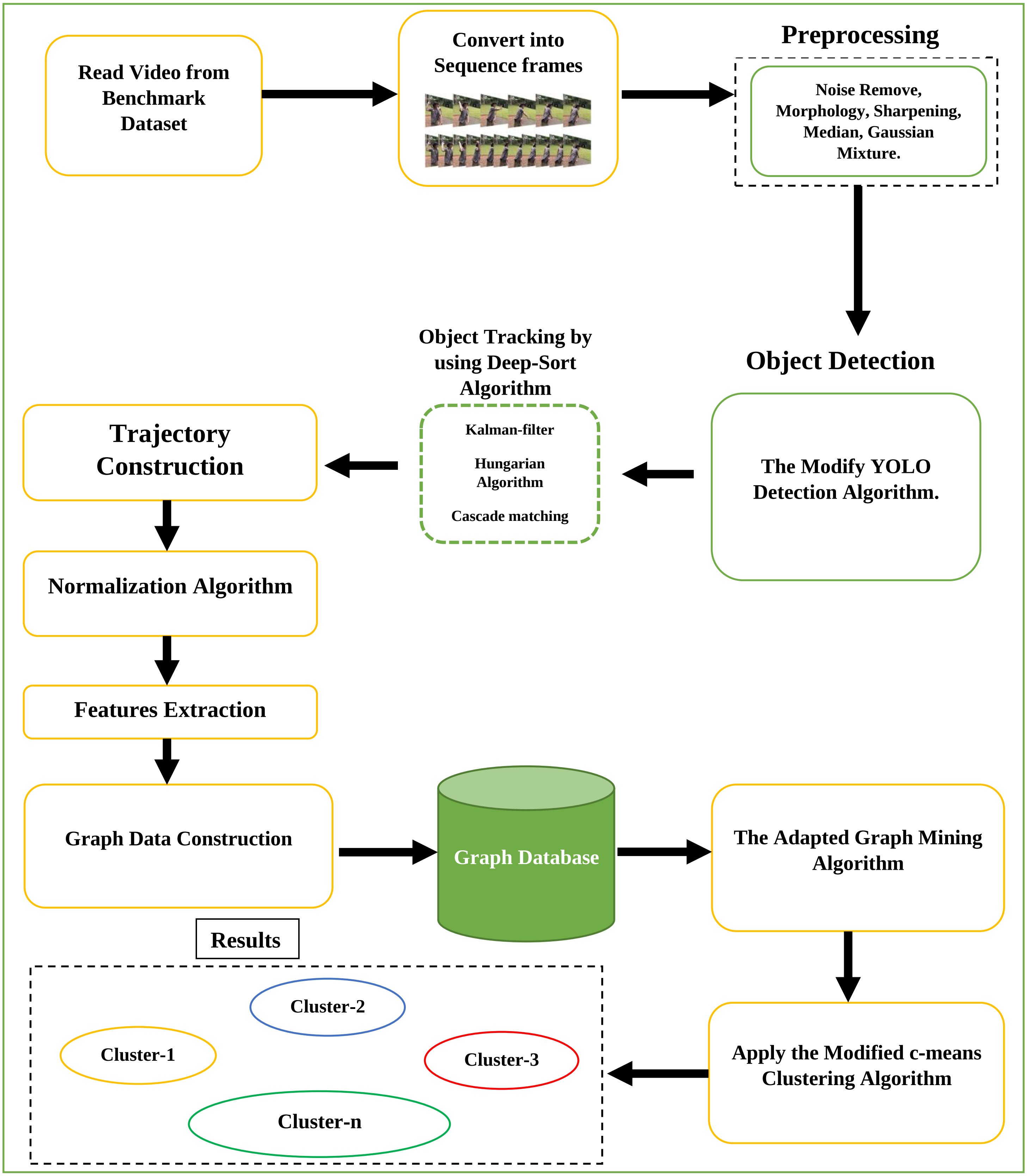


Figure 3.1: The Block Diagram of the Prposed System

3.2.1 Reading and Preparing Video Benchmark Dataset Stage

The first stage in any model is the data input. The dataset must meet some data requirements, and it must also be tested and published. The dataset used in this model is video bird dataset. It is appropriate with the concept of clustering model; it has various classes of the same bird animal kind. The dataset is called VB100 Video Bird Dataset. The proposed system first read video datasets from video benchmarks and from scenes that were recorded in different environments.

3.2.2 The Splitting Video into Frames Stage

To begin the processing stage on video data, first, we must divide the dataset video into several frames (still images) to deal with them as a static image since it is simple to cope with the type of data. Splitting video into frames must be appropriate with the clustering model of this dissertation. It must be considered with splitting frame rate (number of frames per second) where it is representing the motion speed. Therefore, it must balance the frame rate per second (fps) with the mechanism of the proposed system.

3.2.3 The Preprocessing Stage

Preprocessing stage is starting after splitting the video into frames where sometimes the image needs some operation to be ready for detection and tracking, so some frames have noise or blurring and there are many effects the image should be polluted. There are many causes for noise where sometimes the received image is degraded by many degraded methods. One of the important noisy images causes is lens optical in a digital camera that captured visual information where if the camera does not exactly focus on the resulting image, it will be a blurred image. Another issue that may be faced is outdoor changes such as foggy weather and ambient illumination called variation appearance. This needs techniques to enhance and filter the images. The outcomes frames should be free from noises and aberrations. A list of some filtering and image enhancement operations are used in the proposed system are listed below:

1- Gaussian blur filter is applying the gaussian function on the blurred image, and it is typically used to reduce detail and noise. The image produced from gaussian blur is smooth. It is also used with images that capture affected by lens out focuses or images have shadows too. The gaussian function is reduces the high-frequency components image, so gaussian blur is a low-pass filter.

2- Morphology operation applies on each frame. Mathematically, morphology is an operation to eliminate imperfections in the region of interest of the image, and it provides rich information patterns and image structure. In the proposed system, two morphological operations are used which are erosion and dilation methods to improve the detection and tracking processes as well.

3- Median and sharpening filters should be applied when they are needed. The median filter is used for noise removal, whereas the sharpening filter is utilized for enhancing the edges of the objects and for contrasting the shade properties. The median filter is one of the most important nonlinear filters, and it is robust with impulse noise like “salt and pepper”. On the other hand, sharpening filters are sensitive to noise, and they are used for noise reduction. Sharpening filters are making bright pixels more bright compared with their neighbors which will help the model to improve object detection in each frame (image).

All these filters and noise removal processes are applied on each frame before transferring it into the detection stage because they improve the quality of object recognition will assist the detection algorithm to increase the accuracy and speed of object detection. All formulas and equations with brief details of each process are discussed in Chapter 2.

3.2.4 The Object Detection Stage

Object detection is a computer vision task to detect instances of semantic objects of a specific class such as humans, cars, or birds in videos and digital images. It is commonly used in many tasks of computer vision like image annotation, vehicle counting, face detection and recognition, and activity recognition. Each object class has its own certainty features that assist to classify the class. Object detection is necessary to initialize the tracking process, and it is applied for each frame. Moving object detection is one of the important tasks of object detection. A widely used approach to moving object detection is to use temporal information fetched from consecutive frames.

In this proposed system, the multiple objects detection process is done by using YOLOv3 algorithm. The modified YOLO algorithm is a detection method that frames object detection as a regression issue to separated spatial bounding boxes and class probability associated. YOLO has a single neural network for predicting bounding boxes and class probability directly from the entire image (frame) in one step of evaluation. It is a speed and robust architecture, and it processes 45 frames per second in real-time system while taking 155 frames per second with a small version of the network. YOLO detection algorithm reduces false positive prediction on the background.

In this proposed system, Yolov3 (an Increment Improvement) is used because it is faster than other approaches, where it is at 320x320 Yolov3 executes in 22 ms, and it is more accurate and three times faster than SSD (Single Shot Multi-Box Detector). Yolov3 is a very fast detection algorithm, and it reasons generally for the image when prediction making since it sees the entire image in training and testing time. Yolo detection learns generalizable acts of objects. Figure 3.2 illustrates an example of object detection using YOLO algorithm. Algorithm 3.1 illustrates the modified YOLO detection algorithm.

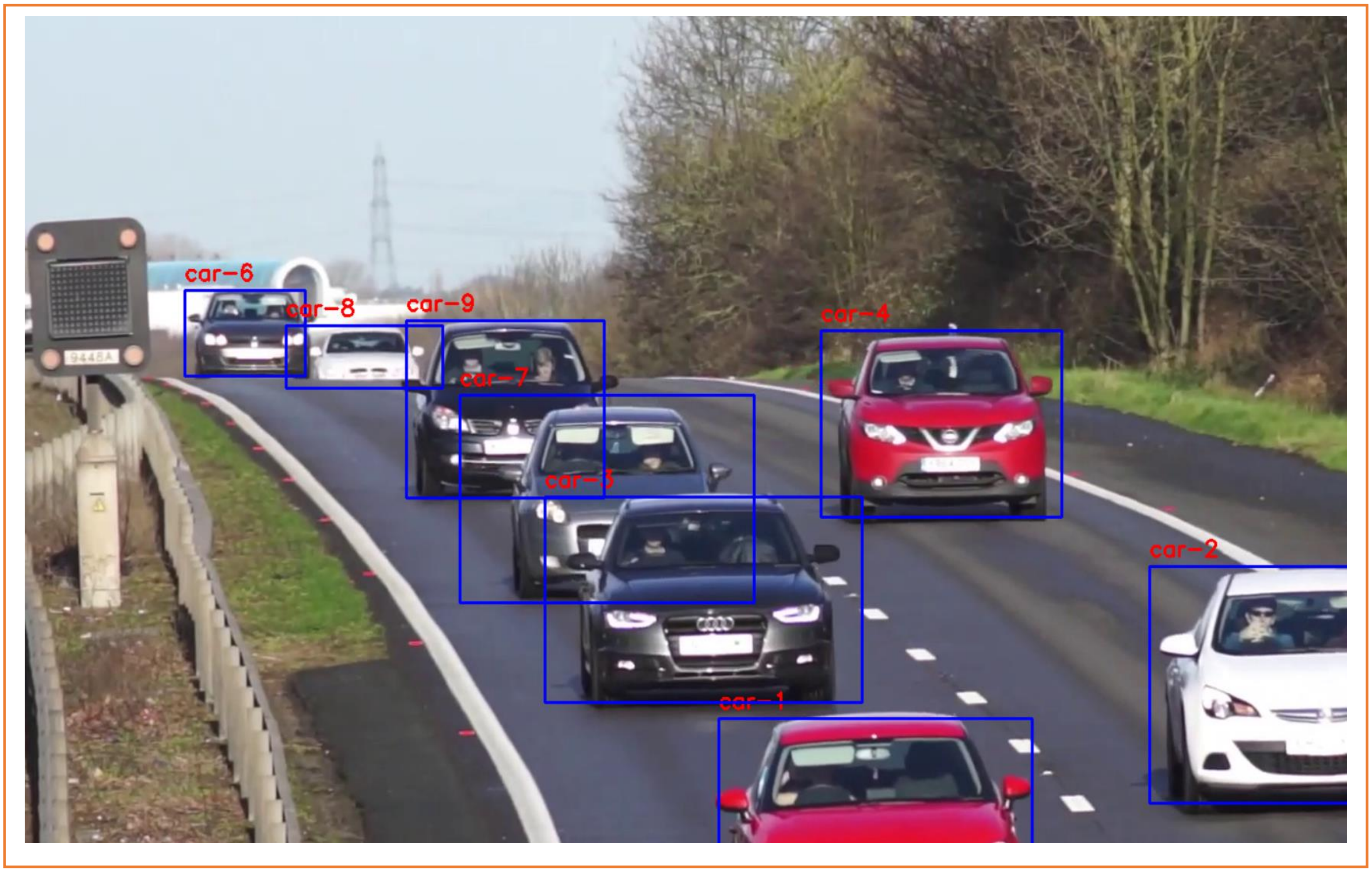


Figure 3.2: An example of Object Detection Using the YOLO Detection Algorithm

Algorithm 3.1: The adapted YOLO Objects Detection algorithm.

Algorithm Name: The Adapted YOLO Algorithm

Input: Raw_Video_Data

Output: Bounding_Boxes_Along to Recognize Class (animal, person, etc.)

Begin

Step 1: Divide video into number of frames (30 frame per second)

Step 2: Resize each frame into 416*416*3.

Step 3: Convert each frame from BGR to RGB

Step 4: Call_Preprocessing functions stage

Step 5: Determine and Detect Region of Interest (ROI) in each Frame by Using CNN (add more layers to enhance the accuracy) pretrained model:

- ***Splitting each frame to regular grids and the origin coordinates assigned to all the grid cells***
- ***Check all pixels to Select which Pixel has object then assign origin point of object***
- ***For each grid cell:***
  - ***it predicts B boundary boxes, and each box has confidence score***
  - ***it detects one object only regardless of the number of boxes***
  - ***it predicts C conditional class probabilities (one per class for the likeliness of the object class)***

Step 6: Set Single Bounding Box for each Object by Using Intersection Over Union and Non_Max_Suppression:

- ***Calculate area overlapping and area union between predicted bounding boxes with actual bounding box.***
- ***Calculate IOU by divided intersection area on union area for all bounding boxes.***
- ***Order bounding boxes with its probabilities (IOU).***
- ***Discard all the boxes having probabilities less than or equal to a pre-defined threshold (0.8 in this case)***
- ***Pick the box with the highest probability and take it as the output prediction.***
- ***Discard any other boxes which have IOU greater than the threshold and less value from the output prediction box from the previous step***
- ***Repeat step 2 until all the boxes are either taken as the output prediction or discarded***

End

YOLO version-3 is used in the proposed system because the salient features of YOLOv3 are making detection with three various scales, so it is better to detect small objects.

Moreover, YOLOv3 selected 9 anchor boxes, three for each scale. In addition, version 3 of the YOLO detection algorithm has more bounding boxes for each image (frame) rather than versions 1 and 2 which will enhance the detection accuracy process. For instance, Yolov3 predicts 10x number of bounding boxes than Yolov2.

In this proposed system, YOLO is modified in its layers such as adding several layers to get highly accurate performance, but it is time-consuming. It depends on the dataset that works on it, and many challenges faced by any video data like illumination, similarity in appearance (Clutter), noise, etc. Also, it modified the number of kernels for each layer based on the nature and content of the dataset, and it can train the CNN of YOLO on a new class such as fish. Moreover, it modified the number of frames per second that depends on how the dataset is captured. In VB100 Video Bird Dataset, the fps is modified to 30 fps since it is captured with 30 fps. YOLO is explained in detail in chapter 2. Algorithm 3.1 shows the YOLO objects detection algorithm.

3.2.5 The Multiple Objects Tracking (MOT) Stage

Object tracking is an operation that predicts the position of a moving object over time. It is a technique to study the visual system of multiple moving objects. MOT is to track many objects' locations at the same time. Tracking multiple objects needs three requirements: detection, prediction, and data association. In other words, MOT is the issue of automatically identifying multiple objects in video data and forming them as a group of trajectories with good accuracy. A multiple object tracking is divided into two aspects: detection-free tracking and tracking-by-detection, and there many approaches to tracking for each group. Detection-free tracking is a manual initialization for a constant number of objects at the first frame. Then, the object is localized in consecutive frames. On the other hand, detection-based tracking is a process to track multiple objects at each frame, where the detection step is applied at each frame. Then, the objects link into trajectories. In the proposed model, the method that is used to track objects is Tracking-by-detection, where the tracking task is based on the detection step.

- **The Deep-SORT Tracking Objects Algorithm**

The Simple Online and Realtime Tracking (SORT) algorithm is an algorithm to track multiple objects with emphasis on a simple and effective approach. However, it is integrated appearance information and motion to enhance the performance of the SORT algorithm. Deep-Sort or Simple Online and Realtime Tracking with a Deep Association Metric is an improved version of SORT to be able to track objects through a long time of occlusion. It is robust and has a high-accuracy algorithm which it is used in the tracking stage of the proposed system. This tracking approach reduces the number of switches identities by 45% and reaches a high performance and frame rate.

The Deep-Sort tracking algorithm works in four phases. The first phase is to define eight-dimensional state space (u, v, γ, h, x', y', γ', h') and confidence score that have the bounding boxes center location (u, v), aspect ratio γ, height h, and the velocities x' and y' of image coordinates. Then, it uses the Kalman filter with fixed velocity motion and model of linear observation, where it takes the bounding box coordinates (u, v, γ, h) as the object state directly. In addition, for each track t, it will count the number of frames for the last successful measure association a_t (it is increasing during Kalman filter prediction). The track that will be greater than the predefined max age of the tracker, will leave the scene and will be removed from the track group. Tracks that are not associated with the measurement at the first three frames will be deleted.

The second important phase of the Deep-Sort tracking algorithm is to apply the Hungarian algorithm, which integrates the appearance information and motion by using two metric which are Mahalanobis distance and cosine distance measures. Mahalanobis distance measure is an effective distance measure that is used to calculate the distance between a point and its distribution. The cosine similarity measure counts the cosine similarity between two vectors (angles). Hungarian approach used Mahalanobis distance to predict Kalman states space and new measurements that arrived, while it utilized cosine distance measure to make retrieve identities after a long period of occlusion specifically when motion with less discriminative.

The third phase of the Deep-Sort algorithm is the matching cascade, where it runs intersection over union association for unexpected appearance changes. For example, partial occlusion causes and improves robustness versus initialization error. The final phase in this tracking algorithm is a deep appearance descriptor. CNN architecture used 10 dense layers to compute global map features of 128 dimensions. Indeed, it used the Deep-Sort tracking algorithm in this proposed system because it is strong through a long time of occlusion with high accuracy.

Algorithm 3.2 illustrates the modified Deep-SORT multiple objects tracking that is used in the proposed system.

Algorithm 3.2: The Modify Deep-SORT Algorithm.

Algorithm Name: The Modify Deep-SORT Algorithm

Input: Bounding Boxes and IDs

Output: Trackers to Track Multiple Objects

Begin

Step 1: State Estimation and Track Controlling:

- ***defined on the eight-dimensional state space (u, v, γ, h, x', y', γ', h') that contains the bounding box center location (u, v), aspect ratio γ, height h, and their velocities in image coordinates***
- ***use Kalman filter with fixed velocity motion and linear observation model***
- ***For each track t we count the number of frames since the last successful measurement association a_t***
- ***if Tracks exceed a predefined maximum age (A_{max}), then object C leave the scene and remove from the track group***

Step 2: Call Hungarian Algorithm: used two metrics to incorporate motion with information appearance:

- ***Use the Mahalanobis distance measure between estimated Kalman states and newly arrived measurements***
- ***Use cosine similarity metric to retrieve identities after long-period of occlusion when motion is less discriminative***

Step 3: Matching Cascade:

- ***Apply intersection over union association to compute for abrupt appearance changes, such as partial occlusion with static scene geometry, and to enhance strongest versus initialization errors***

Step 4: Deep Appearance Descriptor:

- ***using CNN architecture to release a large residual network, with two convolutional layers and six residual blocks***
- ***The general feature map of 128 dimensional is calculated in 10 dense layers***
- ***A final batch and the project of normalization feature onto the unit hypersphere to be appropriate with cosine appearance metric***

End

It is possible to modify the Deep-Sort tracking algorithm by adding more layers for its convolutional neural network to obtain a robust tracking system and reduce the switched identification of objects, and it treats longer periods of occlusion as well. These modifications are added to the Deep-Sort algorithm depending on the dataset and its contents.

3.2.6 The Trajectories Construction Stage

Trajectory means a path of moving object that consists of a set of points that represents the position ($X_{center,}$ Y_{center}) of object at time moment. To extract the trajectory for an object that means it can recognize and analyze the behavior of that object. It is a very important and complex step in the proposed model because it needs to identify the same object in each frame by propagating an object over the time frame by frame. After obtaining the last frame, the whole trajectory of an object will be constructed. In the case of that object at i frame is disappeared, the entire trajectory for it will be built after i appear frame. Figure 3.3 shows an example of trajectory representation constructed by using the OpenCV python library.

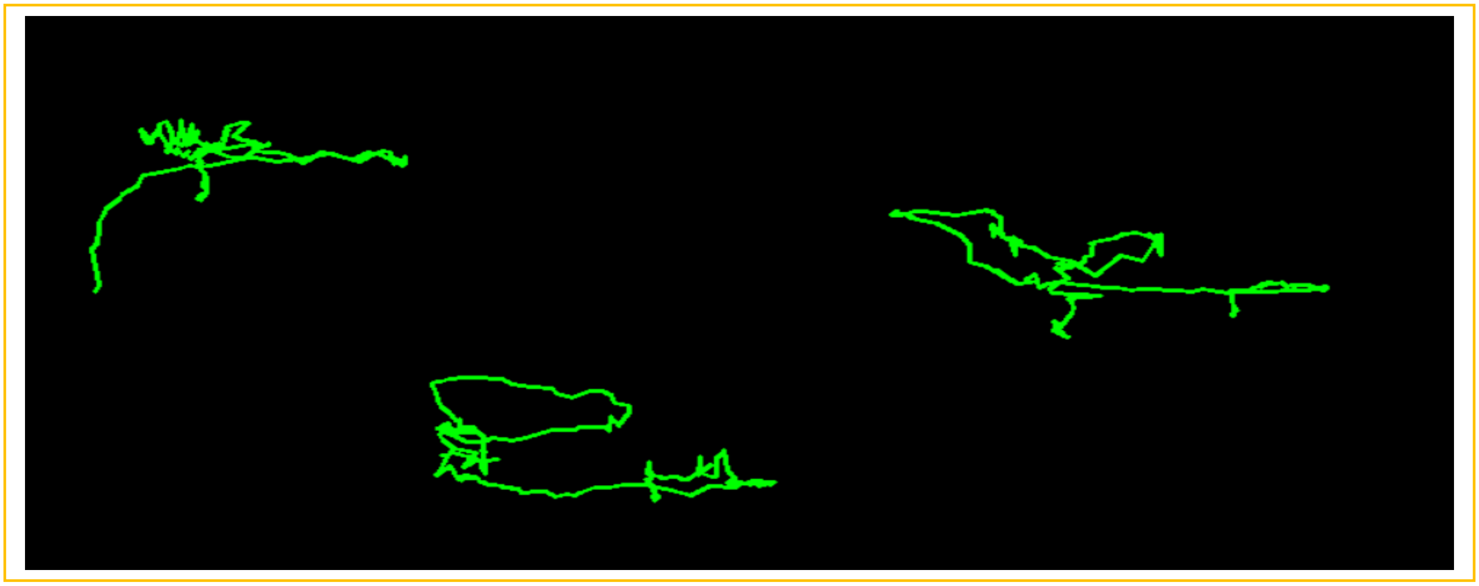


Figure 3.3: An example of Trajectories Representation in python (open cv).

In Figure 3.3 each trajectory is built by propagating the same object at each frame where in each frame the object represents centroid points of that object at this frame. Therefore, the trajectory is a collection of consecutive points that are grouped together to imply a path that represents the entire moving object (positions).

3.2.7 The Normalization Algorithm for Data Reduction Stage

In this important step of the proposed system, trajectory shape needs to be normalized since many of them are not distinct and rich. The normalization approach is a way to prune weak points that build a trajectory with the alignment of other points. Statistical normalization of points has many techniques such as normal score, ratio distribution, standard score, and feature scale. In this dissertation, the feature scale is used to normalize trajectory points.

In this stage, a new normalization shape method is suggested. It depends on dynamic changes for direction between every two points in a trajectory, where the input of this algorithm is all trajectory points extracted from bounding boxes for each frame object location. After that, the current point for each trajectory in the current frame is computed, and the next point is also calculated. Furthermore, it computes angle#1 between the current and the next point. The angle is the direction between two points, and it is calculated by finding the slop $(Y_2 – Y_1) / (X_2 - X_1)$ where (X_1, Y_1) and (X_2, Y_2) are current and next points respectively. Then, the algorithm takes the tangent inverse to figure the direction between two points, where the slops is tan theta for angle.

Moreover, it swaps the current point by the next point and creates a new point by assigning reading a new point. In addition, angle-2 will be computed between the new point and the current point by the same angle-1 obtained above. It creates a new variable called Dynamic Threshold to take the value of the absolute subtracting operation between angle-1 and angle-2 at every frame. This dynamic threshold value is updated in each frame with new point extraction. Algorithm 3.3 illustrates the normalization and the data reduction algorithm.

Then, test the dynamic threshold value if it isn’t equal to zero and the current point also isn’t equal to the next point which means this point will be considered to be taken in the proposed model; therefore, this point will be deleted.

Algorithm 3.3: The Normalization and The Data Reduction Algorithm

Algorithm Name: The Normalization and The Data Reduction Algorithm

Input: All Trajectories (points of each of trajectory)

Output: Distinct Points for each trajectory

Begin

Step 1: Read_Current _ Point = ($X_{\text{center of bounding box}}$, $Y_{\text{center of bounding box}}$) (current frame)

Step 2:Read _ Next _ Point = ($X_{\text{center of bounding box}}$, $Y_{\text{center of bounding box}}$) (next frame)

Step 3: Computing Angle-1 = Tan^{-1} (current _ point, Next _ point)

Step 4: New _ point =Read _ New_ Point

Step 5: Assign Next _ point into Current _ point

Step 6: ComputingAngle-2 = Tan^{-1} (Current _ point, New _ Point)

Step 7: if (Angle-1 is not equal to Angle-2 and Current _ Point is not equal to New _ point)

then take Current _ Point as a normalize point to be considered in the proposed system

Else

Point removed

Step 8: While (there is a frame in system) do (step 1)

End

This progress will be continued until the last point for each trajectory.

This normalization point approach reduces the number of points for each trajectory in about 50% with remaining the important and distinct points are needed in the system.

An example of normalization data points is taken for an object from the dataset. The example that is taken for three objects of birds that moved in the video without occlusion among them. The normalization algorithm will reduce the data positions and orientations by approximately 50% or more. The example will be taken before applying the normalization data algorithm and after making the normalization method to see the difference between them and the performance of the suggested approach in this proposed system. Since pruning weak and redundant points of trajectory, it can obtain trajectory with distinct points that are distinguished as different in behaviors of objects. Figure 3.4 expresses the normalization and data reduction algorithm.

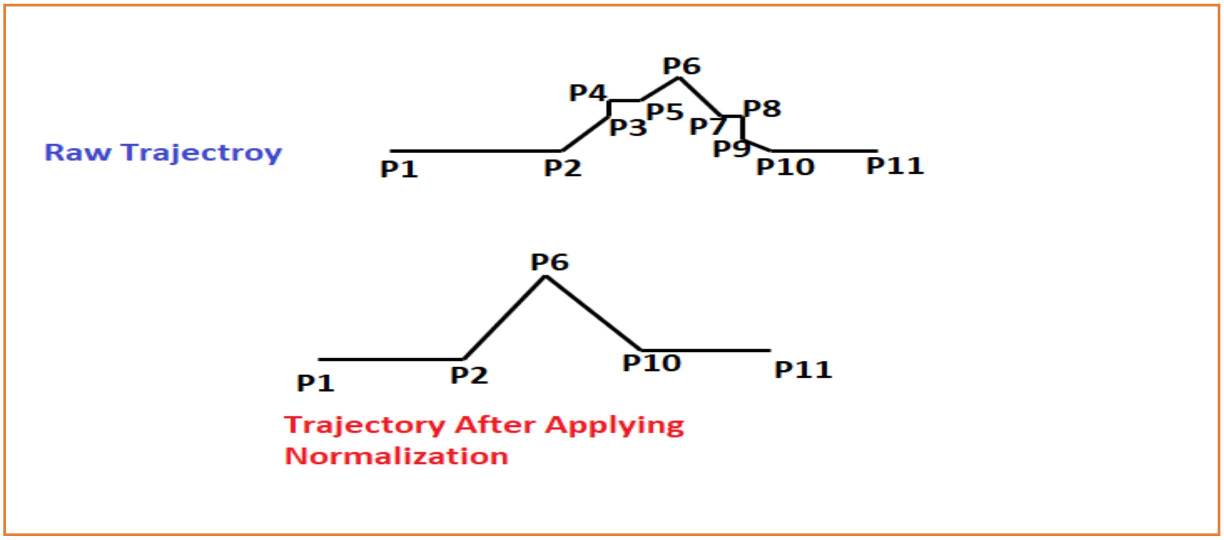


Figure 3.4: An example to show the normalization algorithm mechanism

Table 3.1: An example data before applying normalization

id	X-center	Y-center	Orientation
1	**[0, 0]**	**[0, 0]**	**0**
1	**[680, 240]**	**[680, 240]**	**45**
1	**[680, 240]**	**[681, 240]**	**63**
1	**[681, 240]**	**[680, 240]**	**0**
1	**[680, 240]**	**[681, 240]**	**45**
1	**[681, 240]**	**[681, 239]**	**34**
1	**[681, 239]**	**[680, 237]**	**14**
1	**[680, 237]**	**[680, 238]**	**0**
1	**[680, 238]**	**[680, 238]**	**63**
1	**[680, 238]**	**[680, 238]**	**45**
1	**[680, 238]**	**[680, 237]**	**45**
1	**[680, 237]**	**[681, 229]**	**45**

As seen in Table 3.1 and Figure 3.4 the data before applying the normalization algorithm are redundant in positions and directions, and they do not take any distinct attributes to recognize the behavior of an object. It takes the positions of the X-axis for testing, so it can apply that on Y-axis also.

Table 3.2: Data locations after applying normalization algorithm for 12-frames.

id	X-center	Y-center	Orientation
1	[681, 239]	[680, 237]	14
1	[680, 237]	[681, 229]	45
1	[681, 229]	[680, 236]	45
1	[680, 236]	[679, 237]	63
1	[679, 241]	[680, 240]	45
1	[680, 240]	[679, 241]	45
1	[679, 241]	[678, 240]	45
1	[679, 240]	[680, 238]	63
1	[680, 238]	[679, 239]	45
1	[678, 242]	[679, 243]	63
1	[679, 244]	[678, 241]	63
1	[678, 241]	[679, 240]	14

As seen in Table 3.2, after applying the normalization algorithm to raw data, the data are produced is without repetition, and this can indicate that data are representing an interesting properties of object behavior. Figure 3.4 expresses an example of how the normalization and data reduction algorithm works in the proposed system.

3.2.8 The Trajectory Features Extraction Stage

In image processing, machine learning and pattern recognition, feature extraction is a way to extract distinct information that is unique to recognize objects. It should be not redundant, informative, unique, and meaningful. It is a branch of dimensionality reduction since it focusses on rich areas of object study. In image processing and video processing concepts, feature extraction is a technique to detect and separate different destination shapes and portions of digital image or video frames. Many types of features could be extracted which are high-level features, low-level features, and features which are extracted by using a deep neural network. It is dependent on what problem that needs to be solved. Object in a video scene has various attributes to be figured out such as object area, perimeter, location etc., while an image has other types of properties like color and texture.

After extracting trajectories and applying normalization, the system is extracted trajectory features that assist to construct a graph from the trajectory in this chapter later. Features are calculated between two points for all trajectory points. For trajectory, the features that are extracted are spatial because they are based on the trajectory points' location. The features that are extracted:

1- Number of Steps: this is computed by Euclidean Distance between two points, where it represents the length of the line segment of two points. It can be obtained from cartesian coordinates. In this dissertation, the distance feature means the distance length that an object travels from one frame to another frame. It is an important feature, and it looks like a step for the object from the current frame and the next frame. It is calculated by the formula as shown in Equation 3.1:

$$\text{Distance (no. Steps)} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \quad \text{……... Eq (3.1)}$$

where (x_1, y_1) represents the current point, and (x_2, y_2) acts as the next point of the line trajectory segment.

2- Orientation (direction): is the angle between two points in the line space of the trajectory segment. It is computed between start point and end point of each line segment of trajectory. Direction is used to measure an angle of rotation, and it is measured as the tangent inverse of the difference of vertical distance divided by the difference of horizontal distance of (x_1, y_1), (x_2, y_2), where (x_1, y_1) is presenting the starting point, and (x_2, y_2) is representing the ending point as proposed in Equation 3.2:

$$\text{Orientation} = \tan^{-1}\left(\frac{(y_2-y_1)}{(x_2-x_1)}\right) \quad \ldots\ldots\ \text{Eq (3.2)}$$

3- Velocity: object velocity represents the rate of change of object location as a time function. Velocity is acting as the motion of an object between two frames; it is the direction and the speed of the moving object. Velocity is calculated by dividing the Euclidean distance by the time difference between frame per second start time and frame per second end time as illustrated in Equations 3.3 and 3.4:

$$\text{Time Difference} = (\text{fps\_end\_time} - \text{fps\_start\_time}) \quad \ldots\ldots\ \text{Eq (3.3)}$$

Velocity = no. of steps / Time Difference …... Eq (3.4)

3.2.9 The Graph Construction and Graph Mining Stage

Graph mining is a collection set of approaches and functions used to (a) properties analysis of real-world graphs, (b) guess how the properties and structure of a given graph might influence some applications, and (c) evolve models that can create realistic graphs that match the patterns found in real-world graphs.

In the proposed system each trajectory is converted into a graph because graph is better in representing huge data. Moreover, the graph reflects patterns frequently for each object. A graph is constructed for each object trajectory by using cosine similarity between every two nodes. Each trajectory represents the locations of an object in each frame after making normalization. Each node acts a point of trajectory object in a specific frame. Cosine similarity is applied between every two nodes (points) for all trajectory nodes. The value of cosine similarity reflects the strongest relationship between nodes. The values act as weights that represent the weighted similarity between each two nodes (points). The similarity function compares the three above-mentioned features for every node (Direction, Distance, Velocity), where the comparison will be applied for the feature to feature such direction to direction for each two nodes. Indeed, a graph will be built, and it will represent the behavior and connections among the nodes of the trajectory object in video frames. For example, it is converted 5 trajectory points for a bird object and then applied cosine similarity between every two nodes of it to construct graph data for that trajectory object. Table 3.3 above is represented 5 frames for the bird object dataset.

Table 3.3: An example of Features Extraction of only one object trajectory (object-2 in bird dataset)

id	X-center	Y-center	Orientation	Distance	Velocity
2	360	374	0	4	4
2	361	376	27	3	4.2
2	363	378	37	1	1.33
2	362	379	0	2	3
2	361	379	76	2	3.14
2	360	379	85	3	4.88

These trajectory points and features are gotten after making normalization to prune weak and unwanted points and remaining rich and distinct points because they have important information about the object that is detected and tracked. Table 3.4 is obtained after applying the cosine similarity function between every two nodes for all trajectory nodes.

Table 3.4: Fuzzy Adjacency Matrix after applying Similarity Cosine between Nodes

Nodes/frames	0	1	2	3	4	5
0	0	0.185208	0.044484	0.980581	0.047765	0.065405
1	0.185208	0	0.989656	0.187665	0.990212	0.992587
2	0.044484	0.989656	0	0.044855	0.999985	0.999737
3	0.980581	0.187665	0.044855	0	0.048916	0.067195
4	0.047765	0.990212	0.999985	0.048916	0	0.999831
5	0.065405	0.992587	0.999737	0.067195	0.999831	0

A fuzzy adjacency matrix (FAM) is gotten after the cosine similarity measure is applied. From the fuzzy adjacency matrix that got, it can construct an undirected weighted graph that forms object trajectory pattern behavior. This FAM has symmetric data which means it must apply data reduction to take either the upper triangle or the lower triangle of it. This graph will be stored in the graph database to prepare it to be input for graph mining algorithm to mine the frequent pattern. Graphs can represent huge data with important information that reflects the conduct of the object. Figure 3.5 shows an example of converting each trajectory into a weighted undirected graph based on nodes' features.

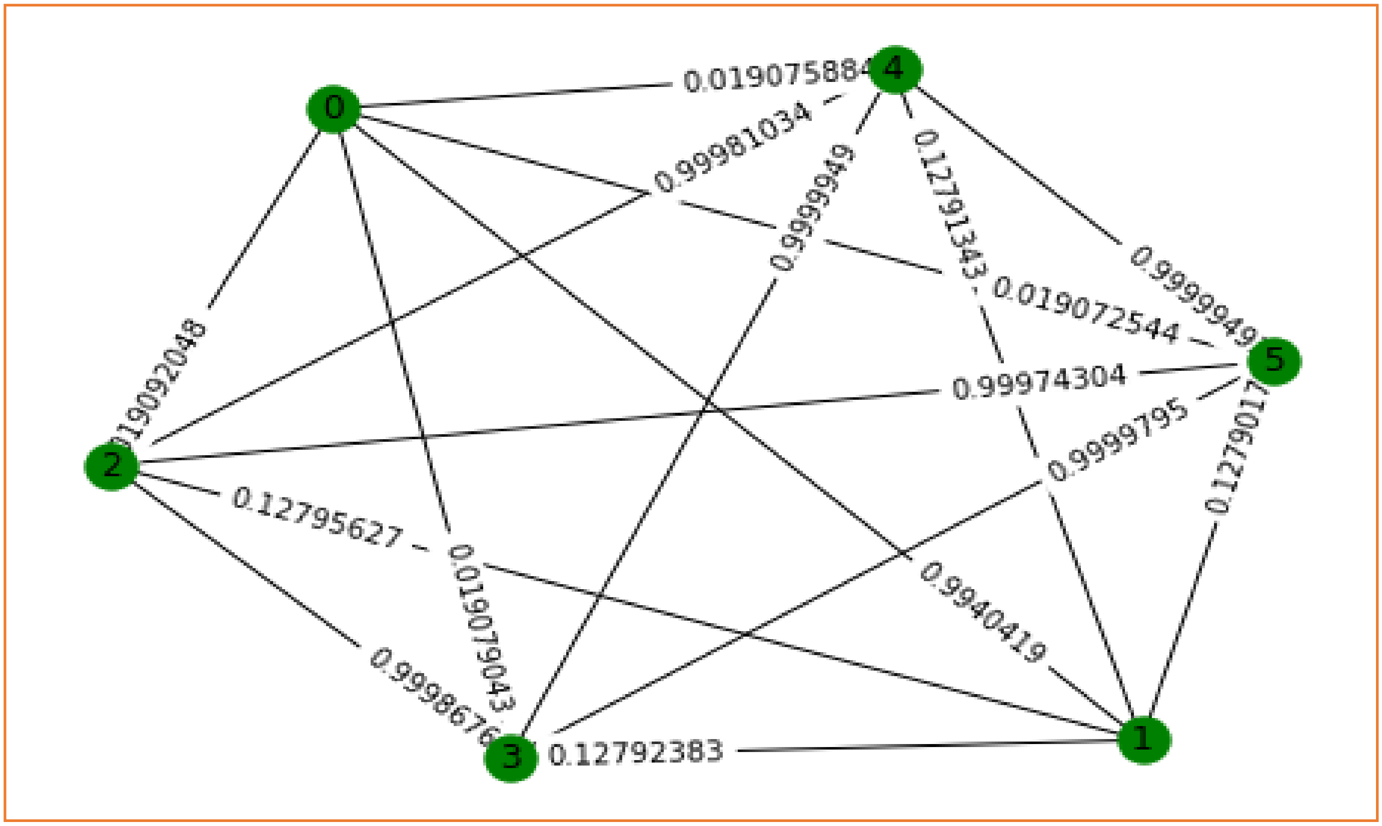


Figure 3.5: An example of a weighted graph generated from the Fuzzy Adjacency Matrix.

In addition, a graph database is constructed to store all graphs that represent the trajectories of objects that are tracked in the scene. Data reduction will be applied before graph data is stored in the graph database because the fuzzy adjacency matrix (FAM) has symmetric data where the upper triangle data is like lower triangle data in the table; therefore, it must select either upper or lower triangle data to be stored in the database.

Table 3.5: Fuzzy Adjacency Matrix after applying Data Reduction

Nodes/frames	0	1	2	3	4
0					
1	0.185208				
2	0.044484	0.989656			
3	0.980581	0.187665	0.044855		
4	0.047765	0.990212	0.999985	0.048916	
5	0.065405	0.992587	0.999737	0.067195	0.999831

For instance, Table 3.5 takes just lower data from the fuzzy adjacency matrix to represent the graph in the best representation in the graph database. If it notices to values in the table, it can know the strongest weight between every two nodes (frames) where, for example, node 3 is strongly connected to node 0 (0.980581) while node 3 is weakly related with node 1 and node 2 (0.187665 and 0.044855) respectively. Strong connection means that the node features are approximately similar to another node that is connected to and vice versa.

Here, it will mine the important data to prepare them for clustering progress. The remaining data values in the lower or upper fuzzy adjacency matrix are the core and significant data of the next important step in the proposed system since the data are taken after many steps of pruning and removing unwanted and not distinct. The data are ranged between 0-1 floating values, so they called fuzzy adjacency matrices (FAM) in their representation.

The remaining data values in the fuzzy adjacency matrix will be converted into flattened numeric vector of fuzzy values for each object where each object will be represented by one input flatten vector (one dimension). In summary, the suggested graph mining algorithm is converting each object trajectory into a graph based on the features between two nodes (frames) by using cosine similarity measure and data reduction.

Table 3.6: 5-Features vectors of 5-objects as Input for Clustering Algorithm

Objects/Feattures					
0	0.185208	0.044484	0.980581	0.047765	0.065405
1	0.988527	0.981487	0.943193	0.98278	0.213806
2	0.999701	0.052958	0.99995	0.99975	0.999948
3	0.99995	0.040373	0.999955	0.997521	0.999953

Table 3.6 shows the final features values for each object that will be an input of the clustering algorithm of graph mining data values that exist in the fuzzy adjacency matrix in Table 3.4 mentioned before. Where each row represents features of that object which are extracted from the graph data representation. Algorithm 3.4 shows the proposed adaptive graph mining algorithm.

Algorithm 3.4: The Adapted Graph Mining Algorithm

Algorithm Name: The Adapted Graph Mining Algorithm

Input: Set of graph trajectories

Output: Flatten Vectors of Fuzzy Values (interested features in range between 0 to 1) for each Object Trajectory

Begin

Step 1: Read All Trajectories as Positions and Features

Step 2: Apply Cosine Similarity Function between each Two Nodes for all Trajectories to Create Fuzzy Adjacency Matrixes

Step 3: For Each Object Trajectory, It Will Generate and Draw Weighted Undirected Graph from FAM to show the strongest relationship among nodes of the same object trajectory

Step 4: Apply Data Reduction on Fuzzy Adjacency Matrix by Taking Either Lower or Upper triangle of FAM

Step 5: Resize the Remaining Lower or Upper FAM Depend on the Minimum length of one Objects Fuzzy Adjacency Matrices in graph database

Step 6: Convert Each FAM into Flatten Vector of the Same Size of FAM after reduction to be as Input for Clustering Algorithm

End

3.2.10 Clustering Stage

Clustering is grouping a set of objects that have similar behavior and features to some extent. Objects inside each cluster are connected, homogeneous, compact, and should have high purity. The main goal of the clustering concept is to explore data analysis, and the main usages of clustering tasks are pattern recognition, information retrieval, machine learning, computer graphics etc. Clustering has many types of models are connectivity-based clustering (hierarchal clustering), centroid-based clustering, distribution-based clustering, density-based clustering, grid-based clustering, and model-based clustering.

Clustering can be recognized as hard clustering and soft clustering (fuzzy clustering) where hard clustering is each object must belong to a cluster or not whereas soft clustering is each object belongs to a cluster in a certain degree. In the proposed system, soft clustering is used distinguish because the data are mined from graph mining algorithm are ranged between [0-1]. Fuzzy clustering is an improved generation of hard clustering where sometime data can’t be considered to belong in one group; objects could belong to more than one group set but by a certain percentage.

Fuzzy c-means FCM clustering algorithm is used to cluster the graphs of trajectory objects based on their behaviors. FCM clustering algorithm is extremely better than the k-means algorithm, and it gives good results and system data values are in between 0-1. Fuzzy clustering is flexible to making decision for which object belongs to more than one cluster (soft clustering) and that well to know the complete behavior of that object and not limitations and specifications. It is an improvement generation on the k-means cluster technique, it is a centroid-based clustering type.

The modified FCM clustering technique works by reading a dataset from a csv file that was figured out from the results of the graph mining algorithm as vectors of values cosine similarity function. After that, it will select randomly initialize cluster. It must determine the number of clusters that should be expected. Then applying the fit function to get the results of vector values and determine the centers of the FCM clustering algorithm. Moreover, it will find the predicted values of the FCM method of graphs of the trajectory objects vectors. Finally, it will plot clusters in scatter by different colors to recognize them.

At the end of this process of fuzzy clustering, it will cluster the data into several groups, and each group has several graphs that represents objects trajectory. Graphs of objects in one cluster behave similarly to a homogeneous group set. For instance, if it takes dataset video that contains 8-birds moving in some way and begins tracking and detection steps, it extracts features for them and applies the cosine similarity function to create the graph for each trajectory object. Indeed, fuzzy c-means clustering algorithm must be illustrated to cluster them. The modification made in the fuzzy c-means clustering algorithm is to manage the input to flatten vectors of all object's trajectories from the fuzzy adjacency matrix (graph data).

Figure 3.6 illustrates an instance of the modified c-means fuzzy clustering algorithm. As seen in Figure 3.6, it has 8 objects clustered. After applying modifying the c-means cluster algorithm on video, it figured out 3-clusters with good performance. As seen in Figure, objects-1, object- 6, and object-8 represent cluster-1, so they behave and have similar paths and features. Furthermore, object-2, object-

3, object-5, and object-7 are representing cluster-2 because they are close in behaviors and attributes. Finally, only object-4 belongs to cluster-3 since there is not another object that is similar to it in conducts and properties.

Figure 3.6: An example of clustering 8-objects (birds) depending on their behaviors by using python.

After the complete clustering stage and division of the data into a number of groups (clusters), the clustering progress must be assessed by using some good and dependent clustering measures to evaluate all clustering outputs that will be discussed in the next chapter of results and output performance.

Chapter Four: Results and Discussion

4.1 Overview

This Chapter displays the results of each step of the proposed system that has been discussed in Chapter three. After the proposed system is constructed, this work evaluates the whole system step by step to show its quality performance by using different experimental measures. Two datasets are used to test the suggested system in this dissertation. The main dataset that is used is VB100 Video Bird Dataset [82], [83]. VB100 is a struggling computer vision data that contains about 1416 video clips of 22 classes of birds. It is captured by a bird expert watcher. This benchmark dataset is used in deep-learning and fine-grained categorization experiments. Bird species is recorded at a good distance, and it is taking in consideration many challenges are considered such as changing pose, scale variation, camera movement (blurring), and background. The data benchmark contains three types: video data, audio data for birds, and classification and geographical localization. Furthermore, each class has about 14 video records with an average length of 32 seconds for each video. One of the most important descriptions and attributes of the dataset is taken under different frame rates where about 69% of clips are captured at 30 fps, 30% at 25 frames per second and the remaining videos are in the range of 60-100 fps. Moreover, the camera must move to track the bird. Therefore, this type of camera is presented in 798 video clips. Then, the 618 remaining videos are captured using hugely static cameras.

4.2 System Requirements and Execution Time

All machine learning and deep learning algorithms that work with computer vision applications need high speed and performance to run these algorithms with good accuracy and results. In this dissertation project the hardware of computer properties used to run the proposed system was:

- Operating System: Windows 10 Pro 64-bit (10.0, Build 19044)
- Processor: Intel(R) Core (TM) i7-6920HQ CPU @ 2.90GHz (8 CPUs), ~2.9GHz
- Memory: 32768MB RAM
- Page File: 8913MB used, 28605MB available
- Card name: Intel(R) HD Graphics 530
- Manufacturer: Intel Corporation
- Chip type: Intel(R) HD Graphics Family
- Display Memory: 16455 MB
- Dedicated Memory: 128 MB
- Shared Memory: 16327 MB
- Current Mode: 1920 x 1080 (32 bit) (60Hz)

The programming language that is used to program all algorithms and results is python 3.8.

The time complexity for the birds (ducks) dataset when it was applied on CPU was 0:01:02.636868 and the max memory usage was 1465 MB, while when the dataset was applied on GPU, the time complexity was 0:00:46.475929 and them max memory used was 812 MB.

The time complexity for the cow's raw video when it was applied on CPU was 0:00:34.652290 and the max memory used is 1503, while when the video data was applied on GPU the time complexity was 0:00:24.480973 and the max memory used was 970 MB

4.3 Cases Study

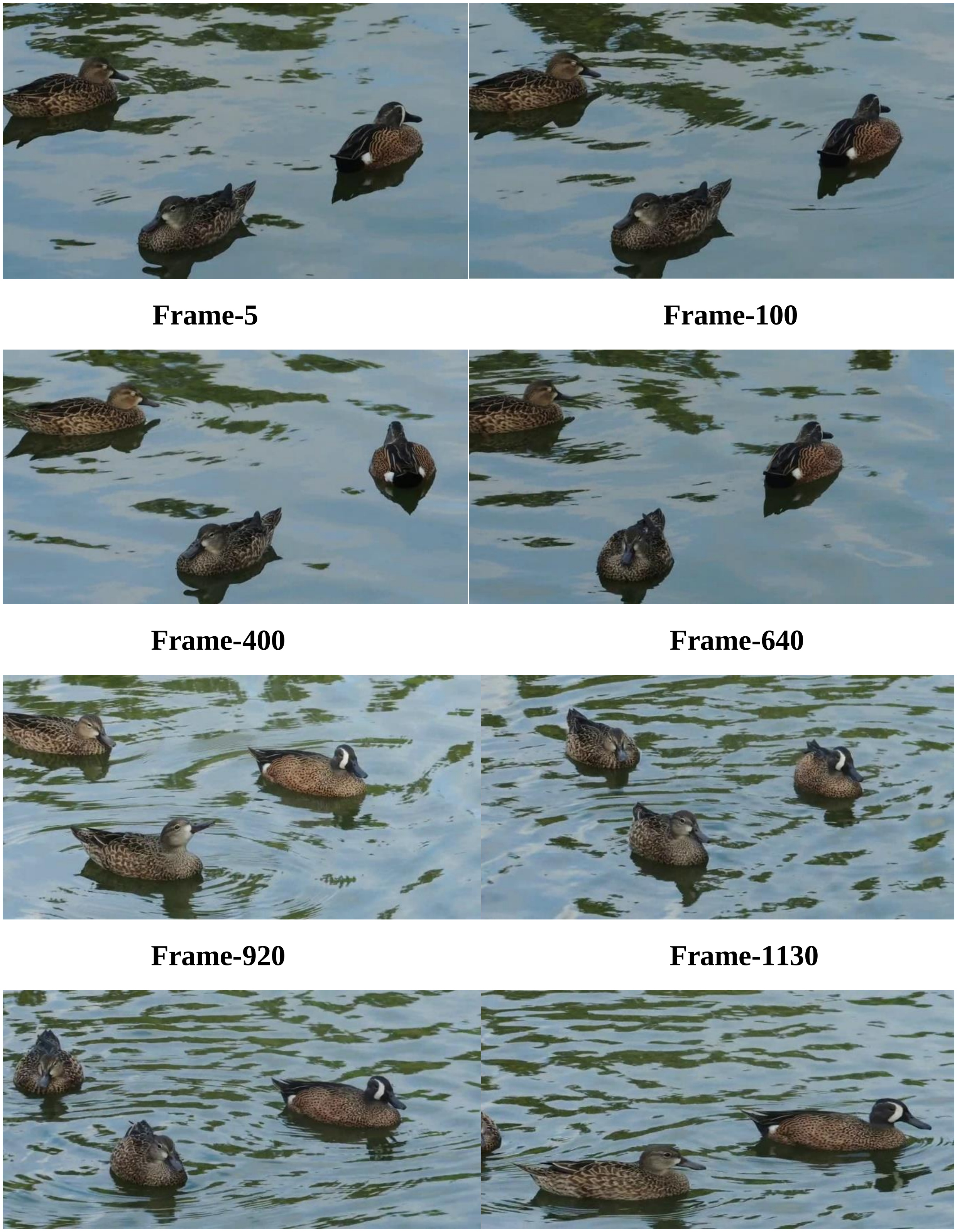
In this dissertation, two cases are studied to test the proposed system and evaluate the results of each step. The first step that is applied to the video is the preprocessing stage to improve the quality of visual objects in the scene and prepare the video for the next stage. The next stage is applied to preprocessing video frames is detection all moving objects either one class or multiple classes. The next important stage in the proposed system is to track multiple objects in video data (frames) that come from the detection stage. After that, the trajectory of each tracked object is constructed. Then, some features are extracted for each object. The normalization of feature data is constructed before going to the next step because many unwanted and poor data should be pruned. The next stage is building a graph for each object based on its features. Then, the graph database is constructed to store all graph trajectories. It proposes a new graph mining data to mine the interesting and rich data from huge graph data for each object graph. Finally, the fuzzy c-means clustering algorithm is applied to cluster the data into a known number of clusters based on the object's behaviors.

4.3.1 Case Study 1

In this case, the bird video dataset is chosen from the benchmark VB100 Video Bird Dataset, the video has three birds of similar class and type. The video is processed as input to the proposed system.

4.3.1.1 Converting Video into Frames and Preprocessing

Figure 4.1 shows samples of frames that are acquired from two processing steps in the proposed system: splitting the video into sequences frames (still images) and preprocessing stages.



Frame-5 **Frame-100**

Frame-400 **Frame-640**

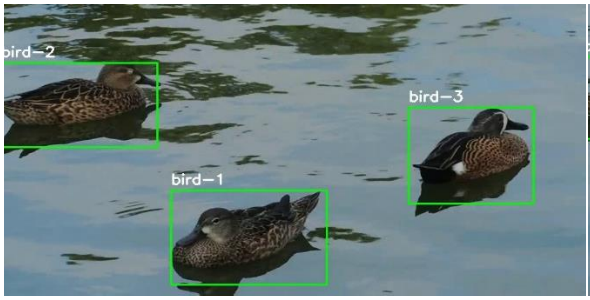
Frame-920 **Frame-1130**

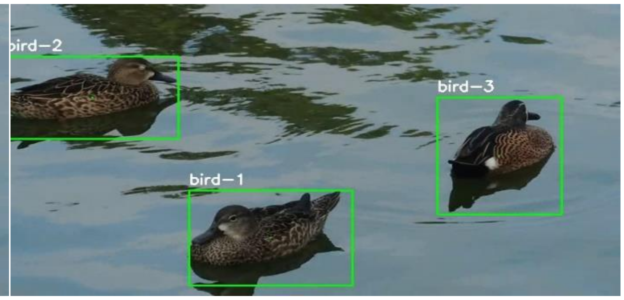
Frame-1300 **Frame-1464**

Figure (4.1): Samples of Sequences frames after removing noise preprocessing stages.

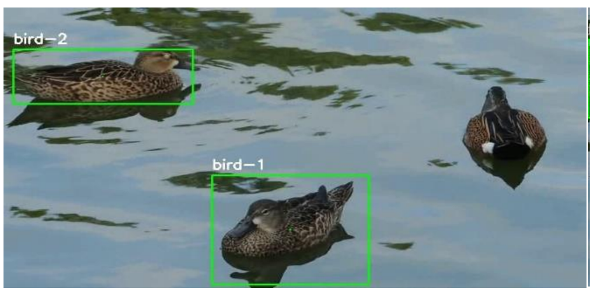
4.3.1.2 Detection and tracking

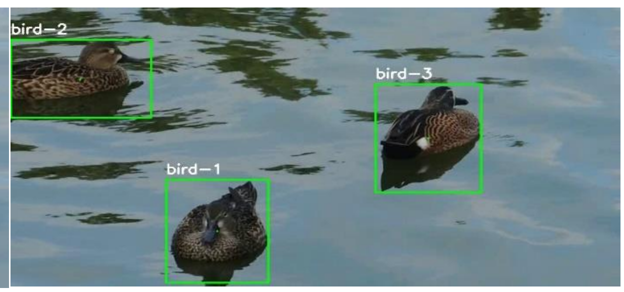
Figure 4.2 illustrates the results of detecting and tracking multiple objects by using YOLO detection and Deep-SORT tracking algorithms.



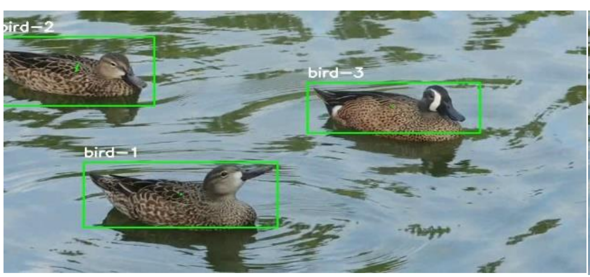


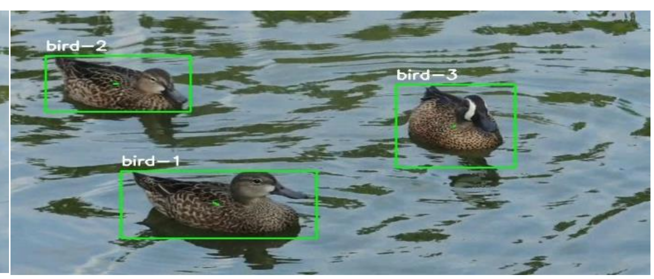
Frame-5 **Frame-100**



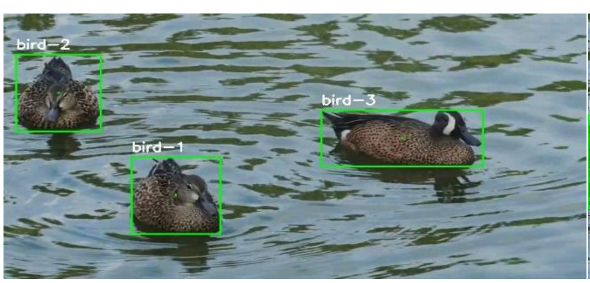


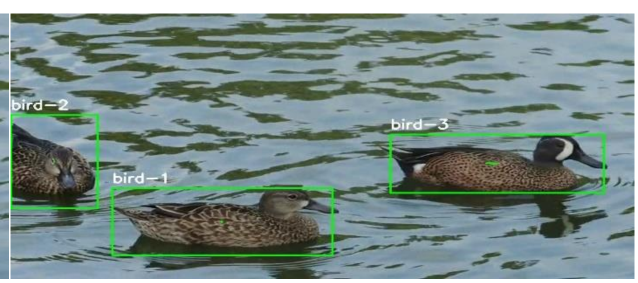
Frame-400 **Frame-640**





Frame-920 **Frame-1130**



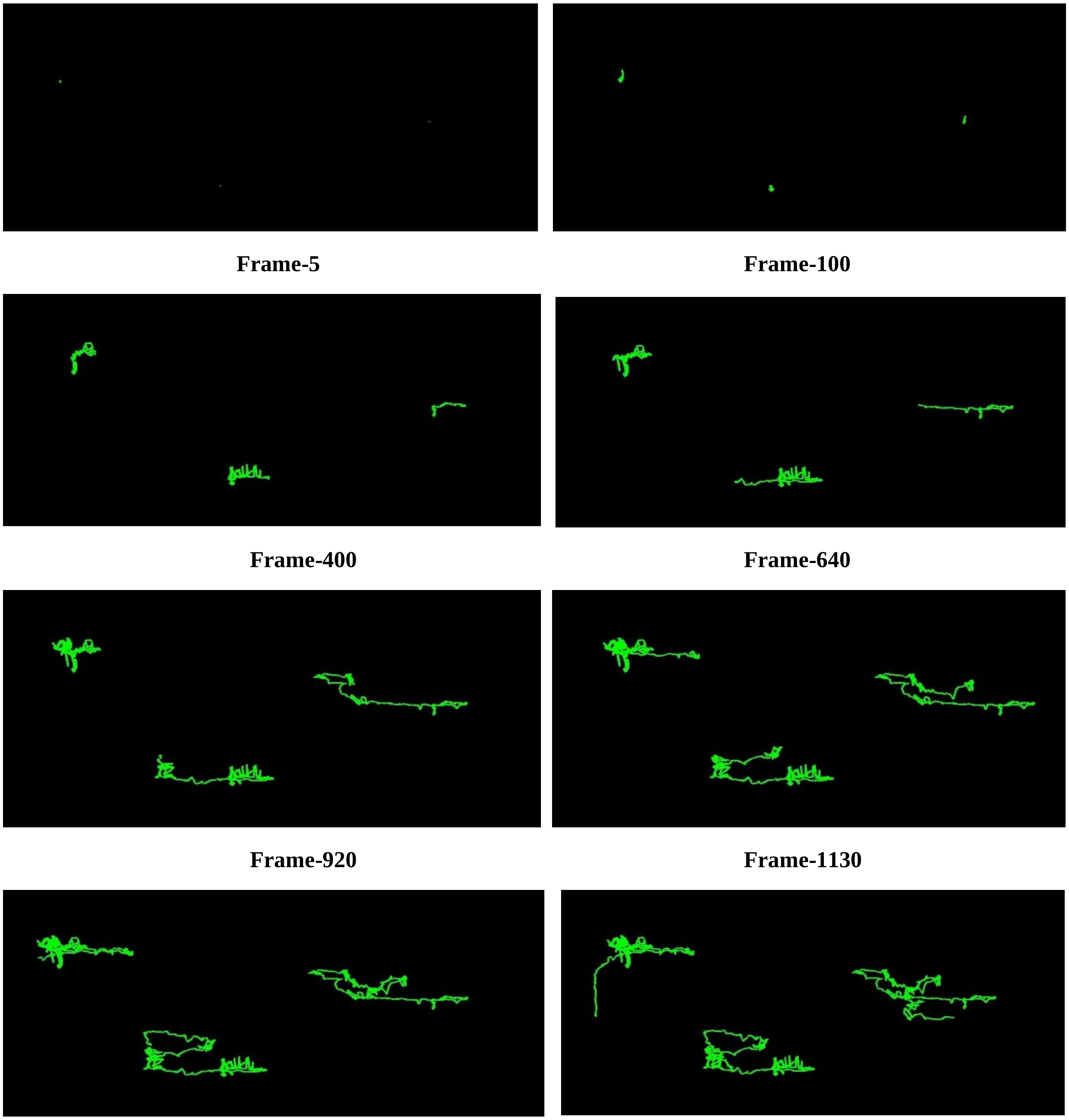


Frame-1300 **Frame-1464**

Figure 4.2: Samples of frames after applied detection and tracking steps

4.3.1.3 Trajectory Construction

Figure 4.3 expresses the sample frames of trajectory construction that detected and tracked objects in the input video.



Frame-5 **Frame-100**

Frame-400 **Frame-640**

Frame-920 **Frame-1130**

Frame-1300 **Frame-1464**

Figure 4.3: Samples frames to represent the trajectories of objects detected and tracked.

4.3.1.4 Feature Extraction

Table 4.1 illustrates an example of features that are extracted from each trajectory object.

Table 4.1: An Example of trajectories features extraction

id	X-center	Y-center	Orientation	Distance	Velocity
1	[681, 239]	[680, 237]	14	2	3.33
1	[680, 237]	[681, 229]	45	8	13.33
1	[681, 229]	[680, 236]	45	7	12.44
1	[680, 236]	[679, 237]	63	1	1.7
1	[679, 243]	[678, 242]	0	1	1.88
1	[678, 242]	[679, 243]	63	1	1.82
1	[679, 244]	[678, 241]	63	3	5.5
1	[678, 241]	[679, 240]	14	1	1.89
1	[679, 240]	[678, 241]	72	1	1.9
1	[680, 236]	[679, 229]	56	7	12.19
2	[358, 371]	[360, 373]	68	3	3.75
2	[360, 376]	[357, 378]	45	4	6
2	[357, 378]	[361, 379]	76	4	6.67
2	[360, 376]	[361, 377]	27	1	1.75
2	[361, 377]	[362, 376]	63	1	1.67
2	[362, 376]	[361, 375]	72	1	1.78
2	[361, 378]	[363, 379]	45	2	3.45
2	[363, 379]	[362, 380]	63	1	1.82
2	[362, 380]	[361, 379]	0	1	1.75
2	[361, 380]	[360, 379]	0	1	1.77
2	[360, 380]	[361, 381]	72	1	1.8
2	[360, 378]	[359, 376]	76	2	3.65
2	[359, 376]	[361, 377]	63	2	3.76
3	[111, 157]	[113, 152]	0	5	6.25
3	[113, 150]	[115, 148]	63	3	4.5
3	[115, 148]	[114, 152]	45	4	6.67
3	[112, 154]	[111, 152]	45	2	3.25
3	[111, 152]	[113, 153]	83	2	3.5
3	[113, 153]	[112, 155]	82	2	3.33
3	[112, 158]	[110, 155]	63	4	6.8
3	[109, 156]	[108, 154]	45	2	3.64
3	[113, 154]	[115, 153]	45	2	3.67
3	[110, 156]	[113, 158]	45	4	7.5
3	[113, 158]	[114, 154]	45	4	7.29
3	[114, 154]	[113, 156]	72	2	3.76

4.3.1.5 Graph Construction

Figure 4.4 shows an example of graph construction of object 1 trajectory based on its features by using an adjacency matrix to represent the similarity between nodes.

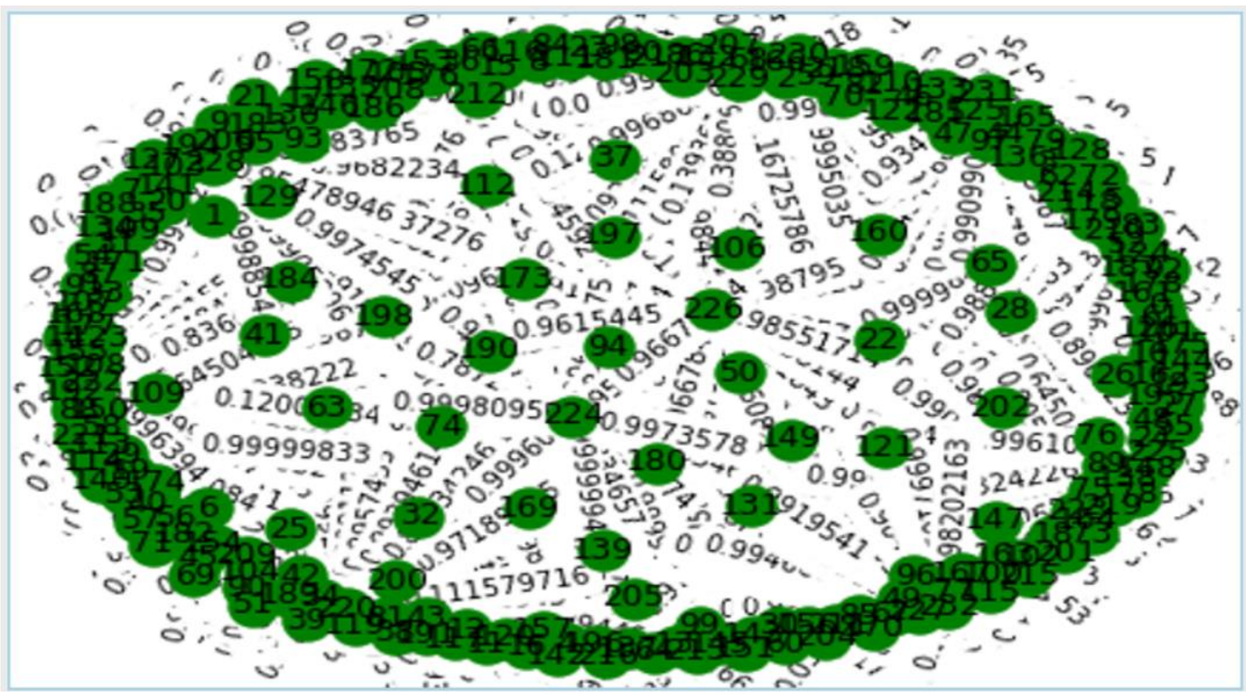


Figure 4.4: An Example of graph construction based on its features (object 1).

4.3.1.6 Graph Data by Using Suggested Graph Mining Algorithm

Table 4.2 represents an example of interesting graph data obtained from proposed graph mining algorithm.

Table 4.2: An example of mining interesting graph data using the proposed graph mining algorithm.

0	0.998513	0.946921	0.353278	0.947727	0.968275	0.980916	0.946423	0.997088	0.989499	0.35288	0.353014	0.352908
1	0.996867	0.994331	0.989178	0.988367	0.996309	0.98942	0.179987	0.179977	0.988912	0.992705	0.993926	0.996805
2	0.997298	0.99999	0.998789	0.998943	0.999306	0.99999	0.99999	0.995138	0.994603	0.999357	0.097363	0.097383

4.3.1.7 Clustering Multiple Objects

Figure 4.5 shows the clustering result of multiple objects using graph mining and fuzzy c-means algorithms based on their behaviors.

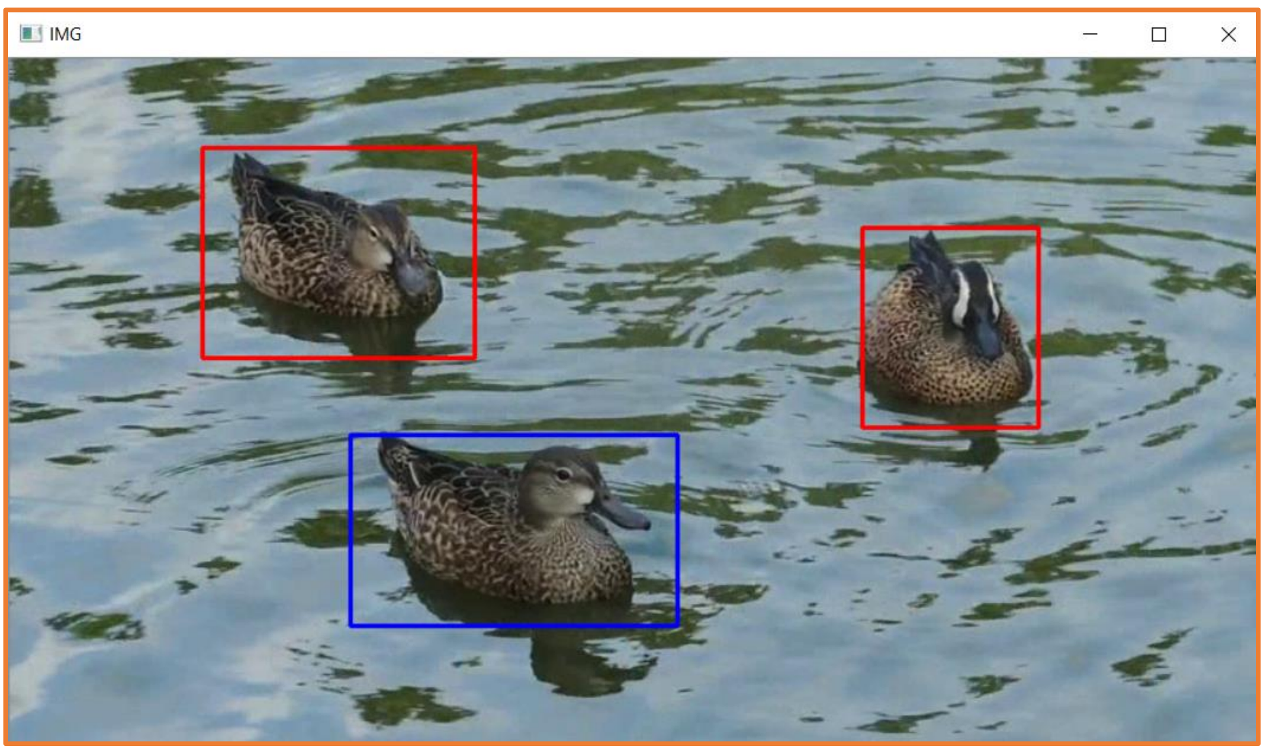


Figure 4.5: Clustering result of objects using graph mining and fuzzy c-means algorithms.

4.3.1.8 Evaluation Measurements

a. Detection evaluation

Figure 4.6 shows the results of detection stage performance with some metrics values using the confusion matrix as the main measure applied on a short length of video dataset (about 5 seconds).

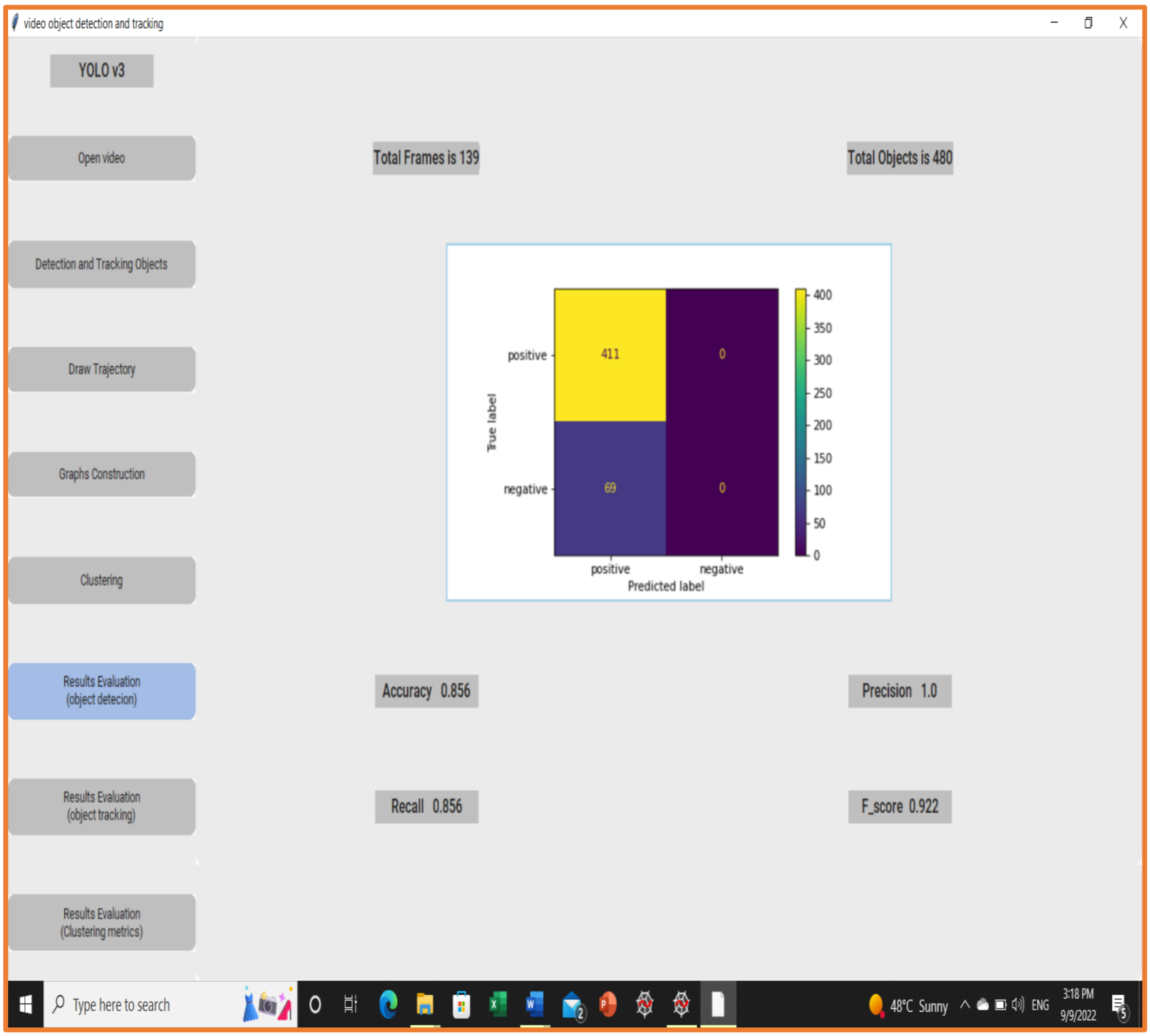


Figure 4.6: Detection stage evaluation results with some metrics values.

b. Tracking Evaluation

Figure 4.7 illustrates the chart that represent the performance of tracking stage.

Figure 4.8 shows the IOU applied on specific frames.

Table 4.3 illustrates the compound confusion matrix of detecting and tracking algorithms.

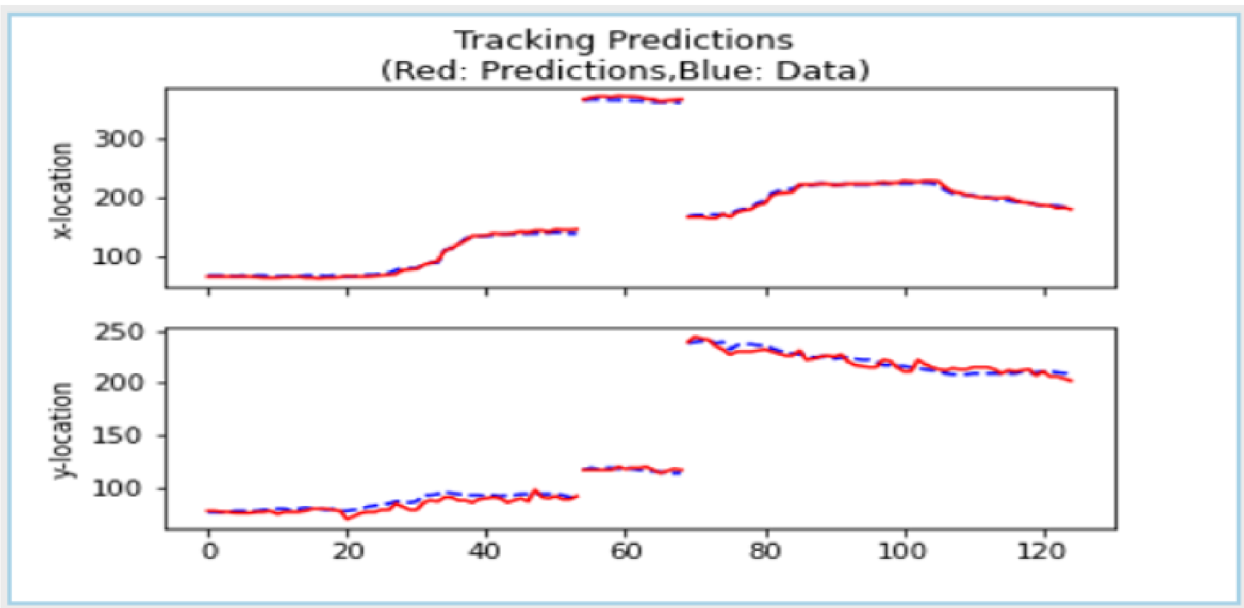


Figure 4.7: Performance chart of tracking stage.

Figure 4.8: IOU of tracking stage that applied of the specific frame (frame-399).

Table 4.3: Compound confusion matrix of tracking algorithm performance.

Total frames=1467 / Frame No.	ID	Truth Bounding Boxes (x, y, w, h)	Predicted Bounding Boxes (x, y, w, h)	Intersection over union threshold IOU > 0.5	True Detection TP	False Detection FP	Not Detection FN	Background Detection TN
398	1	(318,280,512,412)	(302,279,529,459)	0.88	1	0	0	0
	2	(3,62,290,156)	(15,71,275,159)	0.89	1	0	0	0
	3	(667,137,795,247)	(666,142,787,302)	0.89	1	0	0	0
399	1	(319,284,514,416)	(302,278,530,458)	0.63	1	0	0	0
	2	(10,64,290,155)	(13,71,274,160)	0.82	1	0	0	0
	3	(666,138,793,246)	(667,141,786,300)	0.62	1	0	0	0
400	1	(321,289,510,412)	(303,279,530,457)	0.48	0	1	0	0
	2	(8,61,289,155)	(15,70,275,160)	0.8	1	0	0	0
	3	--------------	----------------	---	0	0	1	0
401	1	(318,288,511,415)	(303,278,529,459)	0.6	1	0	0	0
	2	(8,63,293,154)	(15,72,272,158)	0.78	1	0	0	0
	3	--------------	---------------	---	0	0	1	0
402	1	(318,286,510,414)	(303,278,530,457)	0.6	1	0	0	0
	2	(8,64,289,156)	(14,70,274,159)	0.84	1	0	0	0
	3	--------------	-------------------	---	0	0	1	0

- **Precision = TP/(TP+FP) =1390/ (1390+ 6) = 0.99%.**
- **Recall = TP/(TP+FN) =1390/ (1390+71) = 0.95%.**
- **F_Score =2*(precision*recall/(precision + recall)) = 2*(0.99*.95/0.99+.95) =0.97%.**
- **Accuracy = (TP+TN)/(TP+TN+FP+FN) = 1390/1467=0.94%.**

c. Clustering Measures

Figure 4.9 expresses the chart results of clustering based on two features.

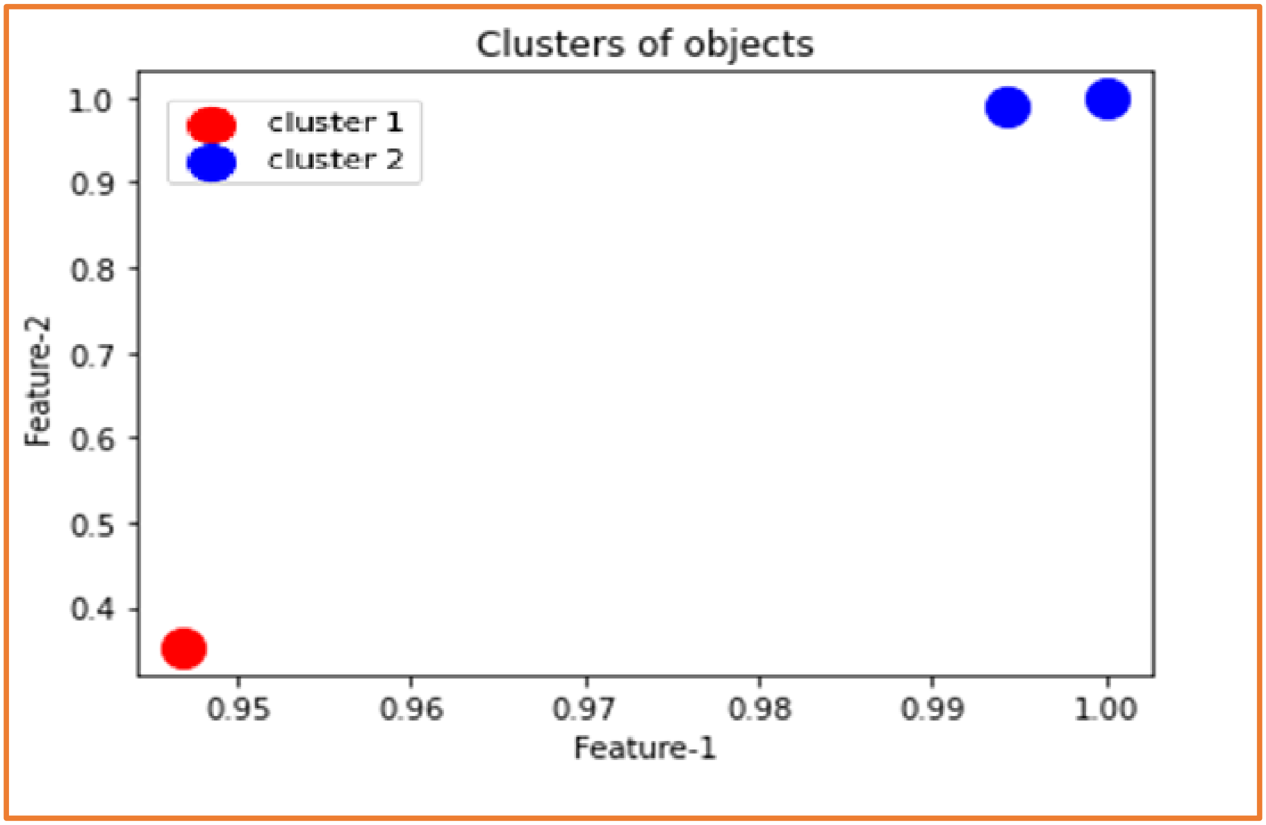


Figure 4.9: Clustering chart based on two features.

- **Clustering Purity Measure = 0.85%.**
- **Rand index = 0.95%.**
- **Homogeneity = 0.89%.**
- **Completeness = 0.99%.**
- **V-measure = 0.94%.**

4.3.2 Case Study 2

In this case, raw videos of the cows [84] are selected to be input to the proposed system. The video is preprocessed before transferring to the detection stage. The preprocessing progress consists of noise removal, edge and borders elimination, and removing any logo or watermarks from all the frames.

4.3.2.1 Converting Video into Frames and Preprocessing

Figure 4.10 expresses samples of frames obtained from two processing steps in the proposed system: converting video data into frames (still images) and preprocessing stages.

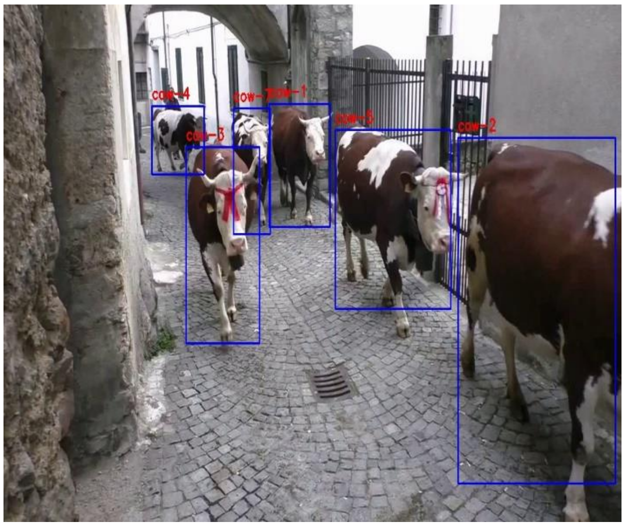
Frame-5 **Frame-50**

Frame-90 **Frame-126**

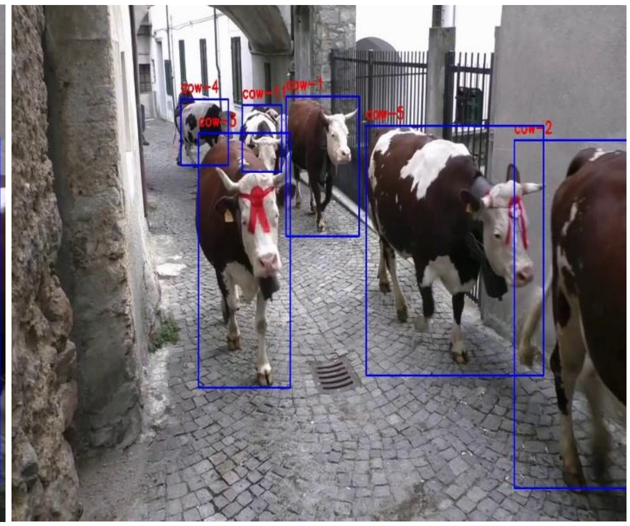
Figure 4.10: Samples of sequences frames after converting and preprocessing stages

4.3.2.2 Detection and tracking

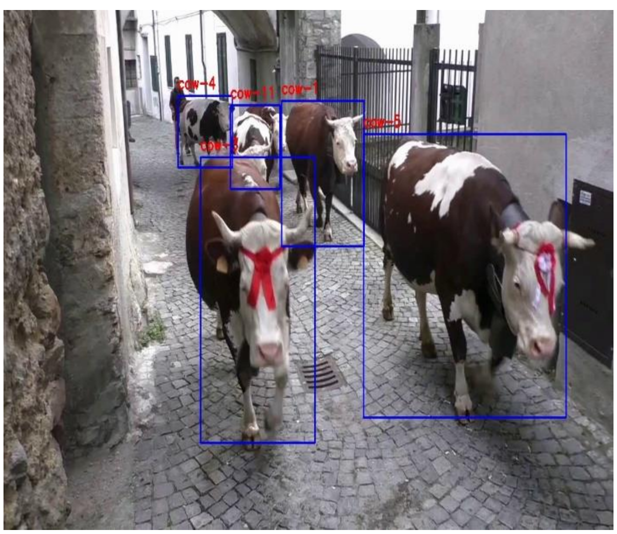
Figure 4.11 presents the outcomes of detecting and tracking multiple objects by using YOLO detection and Deep-SORT tracking algorithms.



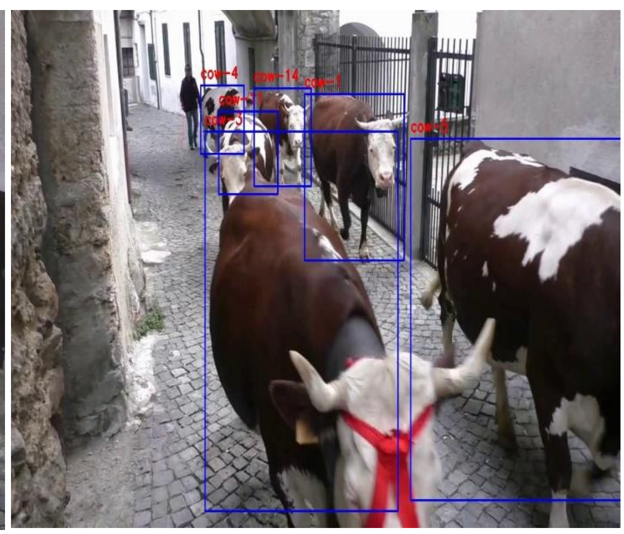
Frame-5



Frame-50



Frame-90

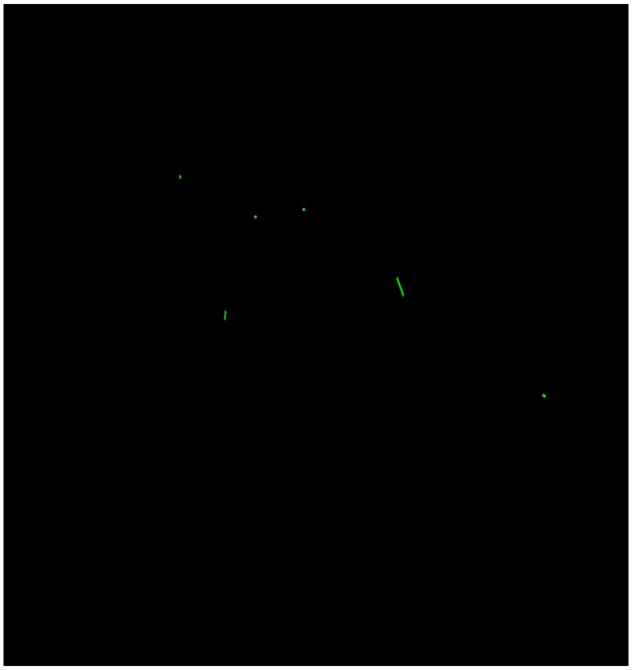


Frame-126

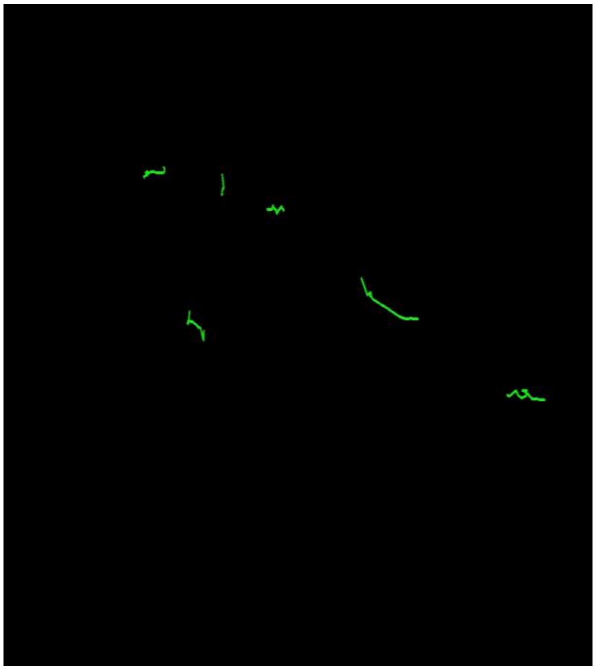
Figure 4.11: Samples outcome after applying detecting and tracking algorithms.

4.3.2.3 Trajectory Construction

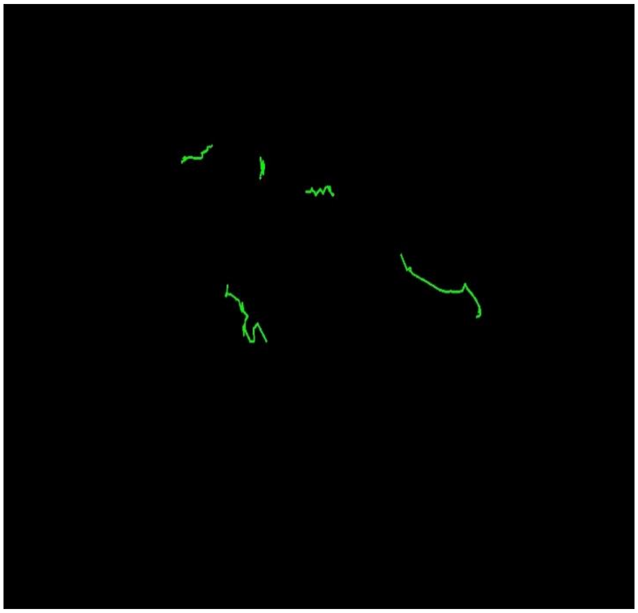
Figure 4.12 illustrates the sample frames of trajectory construction that detected and tracked multiple objects in the video input.



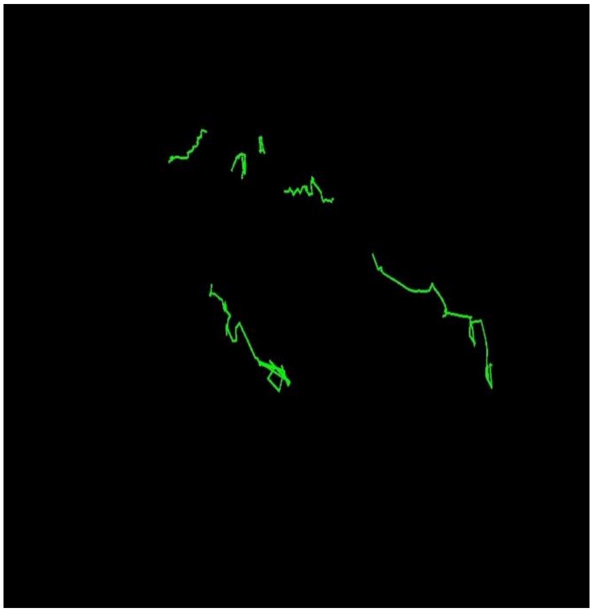
Frame-5



Frame-50



Frame-90



Frame-126

Figure 4.12: Samples of seqence frames which represent trajectores construction.

4.3.2.4 Feature Extraction

Table 4.4 shows an example of attributes that were figured out from each trajectory object.

Table 4.4: Values of trajectories features extraction of objects 1,2,3 and 4.

id	X-center	Y-center	Orientation	Distance	Velocity
1	[516, 231]	[517, 230]	56	1	1.6
1	[517, 230]	[516, 229]	56	1	1.5
1	[522, 229]	[523, 228]	63	1	1.62
1	[843, 325]	[847, 327]	45	4	6.22
1	[847, 327]	[852, 328]	45	5	8.33
1	[852, 328]	[885, 340]	45	35	56
2	[622, 223]	[623, 222]	63	1	1.4
2	[623, 222]	[625, 219]	21	4	6.4
2	[625, 219]	[627, 222]	14	4	6
2	[627, 222]	[629, 223]	45	2	3.33
2	[630, 223]	[631, 224]	8	1	1.5
2	[631, 224]	[632, 226]	0	2	3.25
3	[1104, 425]	[1109, 426]	83	5	6.67
3	[1109, 426]	[1122, 420]	56	14	17.5
3	[1125, 423]	[1126, 425]	0	2	2.8
3	[1126, 425]	[1134, 428]	45	9	14.4
3	[1134, 428]	[1142, 426]	0	8	12
3	[1142, 426]	[1143, 425]	72	1	1.67
4	[454, 334]	[453, 342]	45	8	10.67
4	[453, 342]	[451, 345]	0	4	5
4	[452, 347]	[453, 344]	0	3	5
4	[453, 344]	[457, 345]	18	4	6.29
4	[460, 345]	[465, 347]	18	5	8.12
4	[465, 347]	[469, 349]	11	4	6.22

4.3.2.5 Graph Construction

Figure 4.13 illustrates an example of graph construction of object-4 trajectory depending on its properties by using an adjacency matrix to act the similarity between nodes.

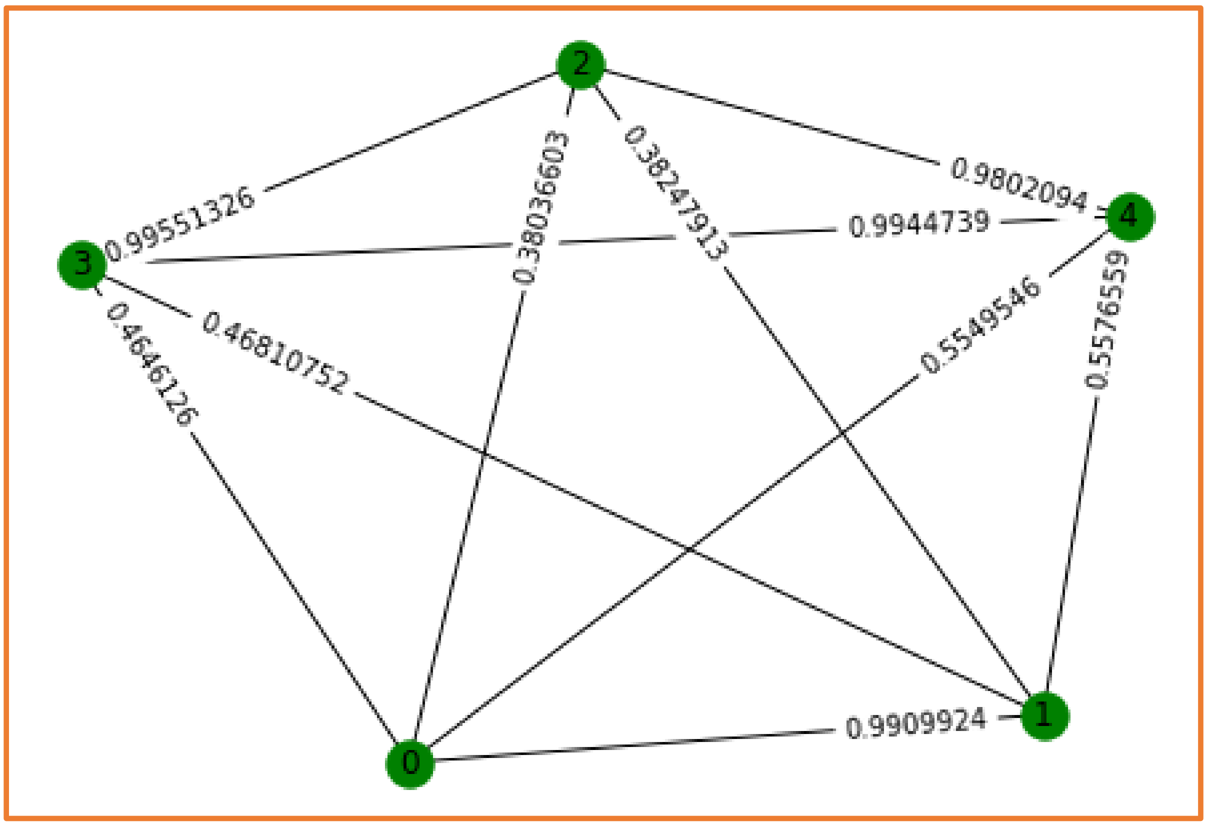


Figure 4.13: An example of graph construction of object 4.

4.3.2.6 Graph Data by Using the Proposed Graph Mining Algorithm

Table 4.5 depicts an example of interesting graph data that was reached by the proposed graph mining algorithm.

Table 4.5: Values of interesting graph data vectors of all object's trajectories.

0	0.999997	0.99147	0.983755	0.589402	0.991211	0.983406	0.58782	0.998746	0.689659	0.724801
1	0.991414	0.966666	0.992367	0.338203	0.925044	0.967917	0.457603	0.990808	0.085996	0.219689
2	0.370995	0.99892	0.37015	0.938024	0.352387	0.99948	0.026937	0.352905	0.944799	0.026993
3	0.990992	0.380366	0.464613	0.554955	0.382479	0.468108	0.557656	0.995513	0.980209	0.994474
4	0.976373	0.980554	0.988062	0.973818	0.999781	0.997995	0.999908	0.999041	0.999421	0.997214
5	0.996241	0.076129	0.164291	0.996241	0.076203	0.165334	1	0.995942	0.076203	0.165334
6	0.986397	0.986928	0.969126	0.971822	0.999993	0.996474	0.997345	0.996192	0.997107	0.999932
7	0.99965	0.050001	0.955239	0.998726	0.023576	0.947084	0.997042	0.343214	0.100338	0.968951

4.3.2.7 Clustering Multiple Objects

Figure 4.14 expresses clustering outcomes of multiple objects using graph mining and fuzzy c-means approaches relying on their conducts.

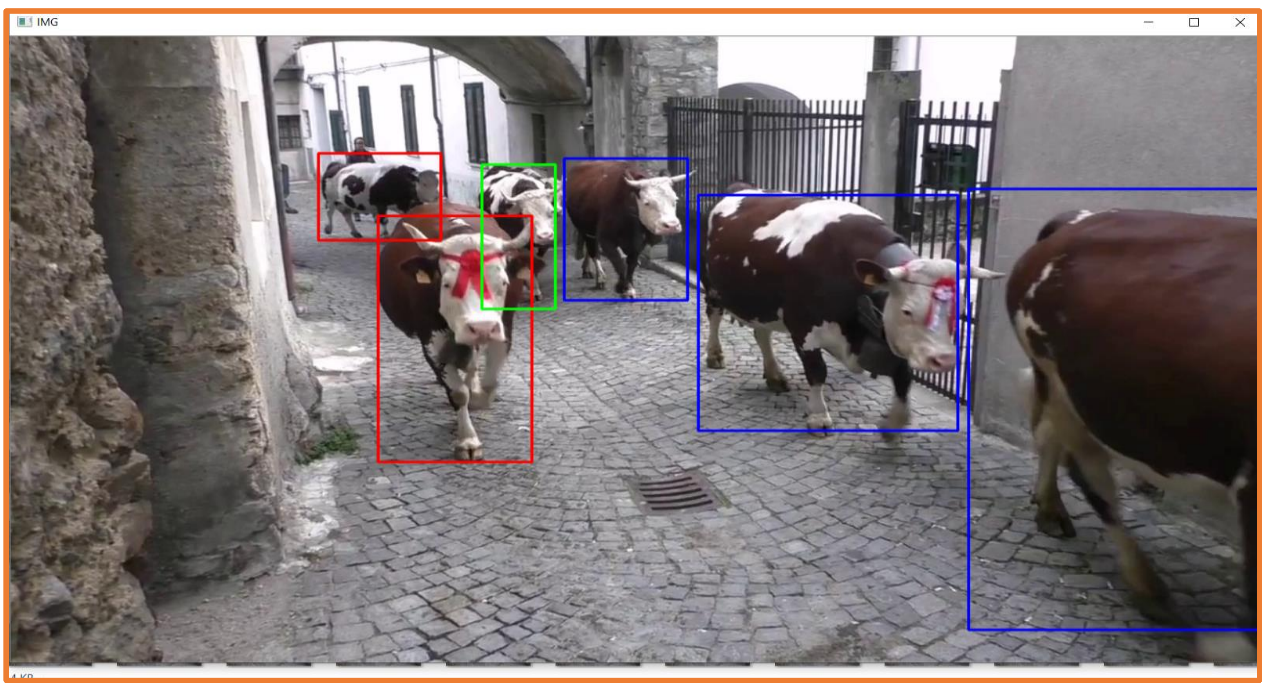


Figure 4.14: Clustering results using graph mining and fuzzy c-means algorithm (frame-20).

4.3.2.8 Evaluation Measurements

a. Detection evaluation

Figure 4.15 shows the results of detection stage performance with some metrics values by using the confusion matrix as the main measure.

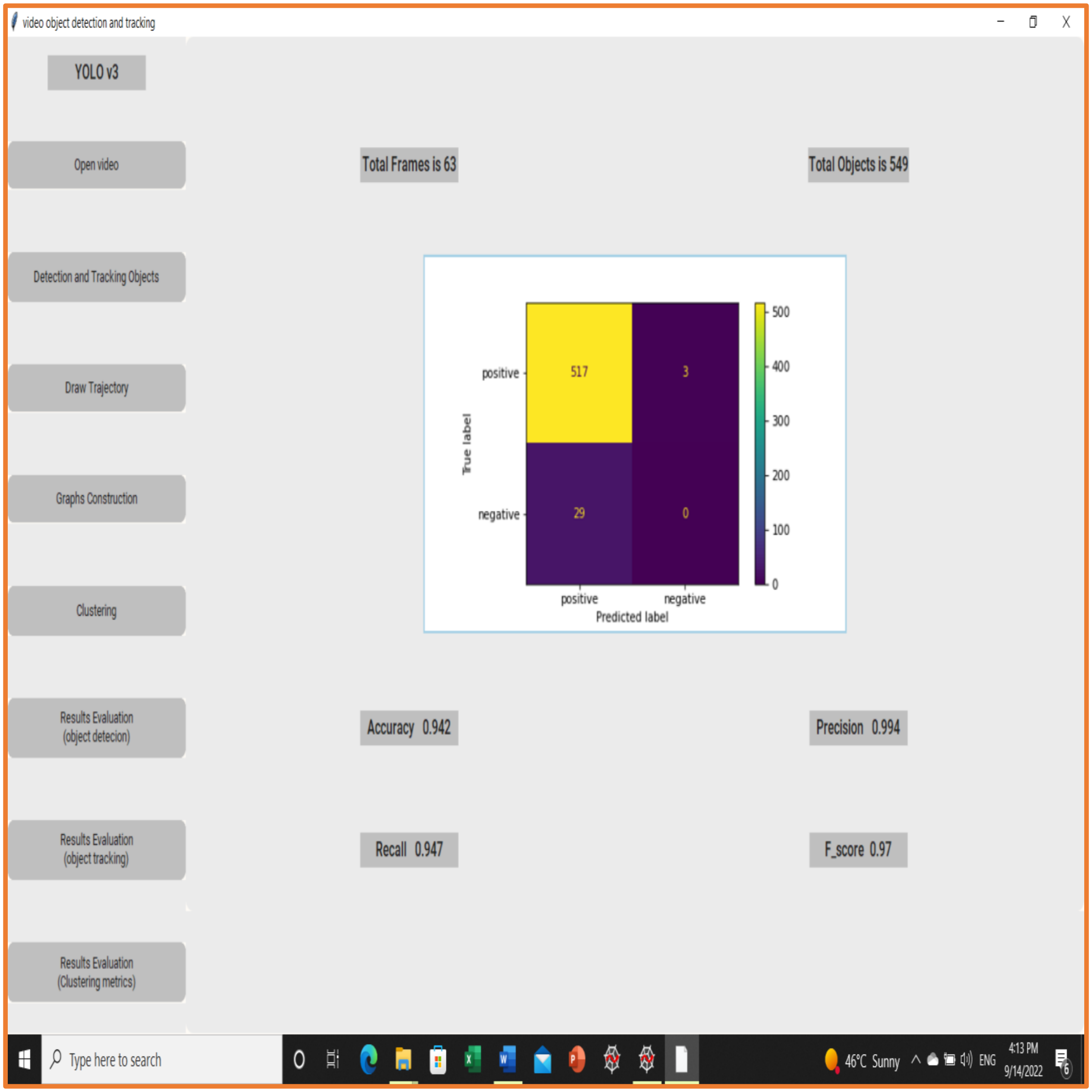


Figure 4.15: The values of the confusion matrix of detection stage performance.

b.Tracking Evaluation

Figure 4.16 shows the performance evaluation of the tracking stage between actual tracking and prediction tracks.

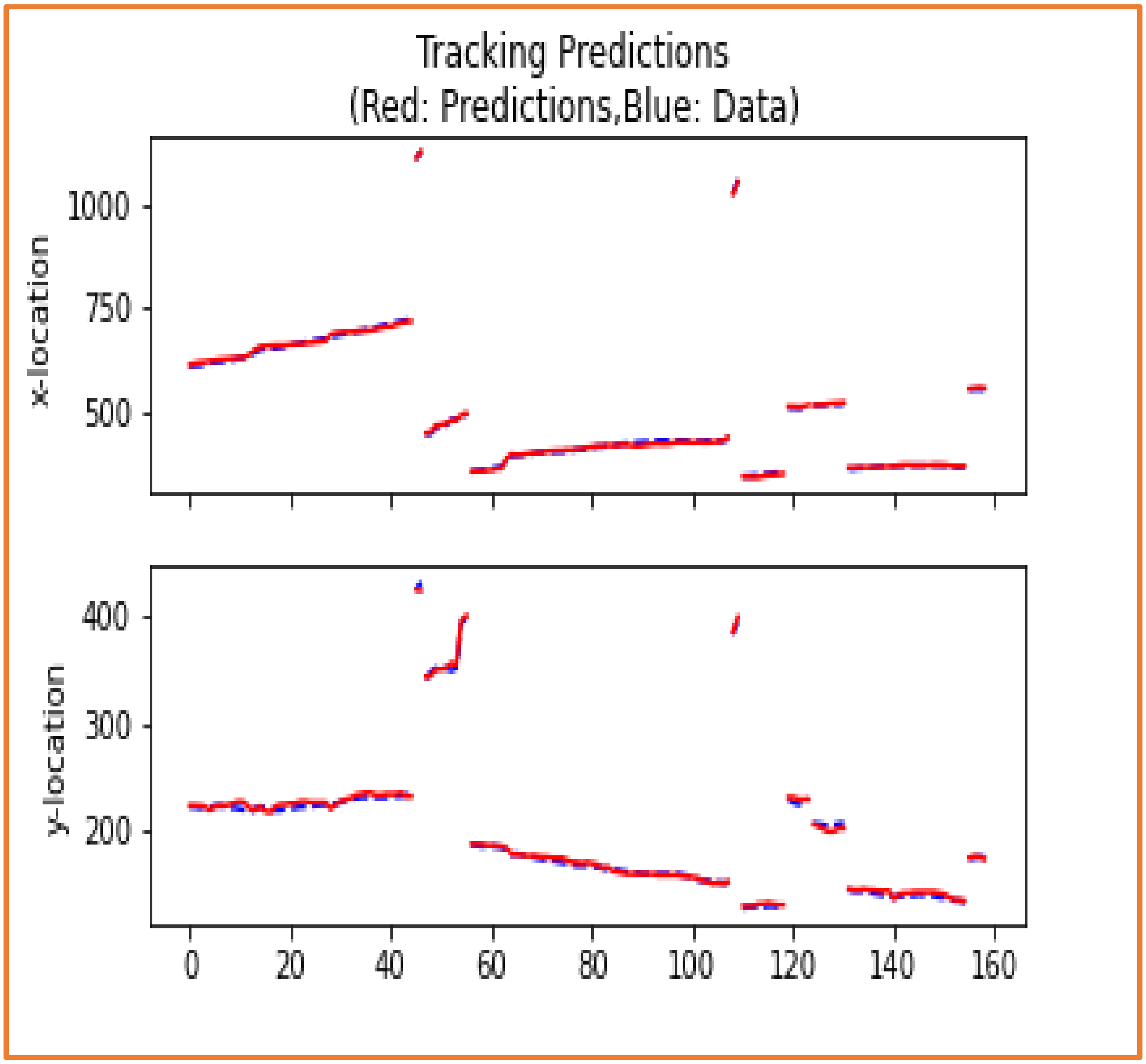


Figure 4.16: Tracking performance evaulation of truth and prediction.

To calculate the evaluation matrices of detecting and tracking, IOU is applied and measures the compound confusion matrix as computed in case 1.

- **Precision = TP/(TP+FP) =517/ (517+3) = 0.99%.**
- **Recall = TP/(TP+FN) =517/ (517+29) = 94%.**
- **F_Score= 2*(precision*recall/(precision+ recall)) = 2*(0.99* .94/0.99+ 0.94) =0.97%.**
- **Accuracy = (TP+TN)/(TP+TN+FP+FN) = 517/549=0.94%.**

c. Clustering Measures

Figure 4.17 illustrates the chart of clustering progress between two features.

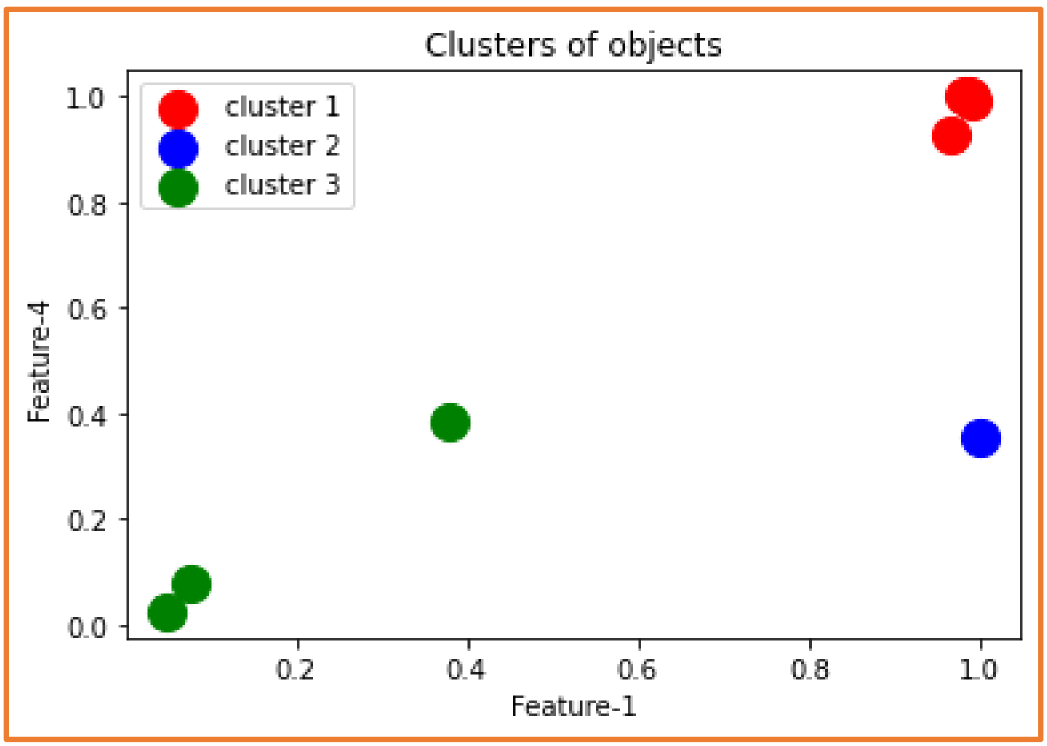


Figure 4.17: Chart of clustering objects process between two features

- **Clustering Purity Measure = 0.87%.**
- **Rand index = 0.96%.**
- **Homogeneity = 0.91%.**
- **Completeness = 0.99%.**
- **V-measure = 0.95%**

Chapter Five: Conclusion and Future Work

5.1 Conclusions

This dissertation proposed a graph mining algorithm to construct and mine huge graph data from multiple moving objects in a video dataset. This technique clusters object trajectories based on their features. The outcomes of applying this approach and other algorithms which are developed in the proposed system to obtain good clustering results. Additionally, all the processes in the proposed system are calculated with less cost and high accuracy metrics values.

The proposed graph mining algorithm copes with mining massive data from graph data representation depending on three important spatial features. The graph mining algorithm and fuzzy c-means clustering algorithm seek objects which have the same behavior to a certain degree.

After designing and implementing the proposed system and the output results, it can be concluded that are follows:

1- The suggested model (proposed system) studies the analysis of multiple objects' behaviors in the raw video without any previous information.
2- The proposed system deals with geometrical relations between every two nodes that construct the trajectory of an object.
3- The selected multiple objects detection algorithm and tracking algorithm that are applied in the proposed system show good quality performance, where they deal with a long period of occlusions and reduce the identity switches. They give excellent results to the next stages in the proposed system.
4- The approach to constructing graph data by using (FAM) is very effective to build a weighted graph that gives the nature of the relationship between nodes in all frames that the object tracked.

5- The geometrical features that are extracted (distance, velocity, and orientation) are very effective to measure the relation between nodes.
6- The proposed shape normalization proposed method is robust by dropping out the unnecessary data points for approximately 50%.
7- The proposed graph mining algorithm is good to mine interesting data and convert them into flattened vectors to be as input to the clustering algorithm.
8- The modified fuzzy c-means clustering algorithm gives very good results to separate the object into groups by highlighting each of them.

5.2 Future Works

There are many future works which can propose:

1- YOLOv4 or YOLOv5 can be used to detect multiple objects with different classes where they are a new and fast version of the YOLO algorithm. It will help to obtain high accuracy of object detection and increase the speed of the detecting process.
2- It can extract new geometrical features such as acceleration, aspect ratio, etc., to improve the graph construction progress.
3- It can construct a graph not for all frames but for a specific number of frames (pattern) to study the behaviors of the object in a specific time.
4- It can be used by any other soft clustering algorithms and compare the results with the clustering outcomes of this dissertation.
5- It can be used by any other graph mining algorithm in this model and then test the results of it.

References

[1] J. Bian, D. Tian, Y. Tang, and D. Tao, “A survey on trajectory clustering analysis,” pp. 1–40, Feb. 2018, [Online]. Available: http://arxiv.org/abs/1802.06971

[2] C. C. Aggarwal, “Data Mining,” 2015, doi: 10.1007/978-3-319-14142-8.

[3] J. Leskovec, A. Rajaraman, and J. D. Ullman, “Mining of Massive Datasets,” 2015.

[4] C. A. Bhatt and M. S. Kankanhalli, “Multimedia data mining: state of the art and challenges,” *Multimedia Tools and Applications 2010 51:1*, vol. 51, no. 1, pp. 35–76, Nov. 2010, doi: 10.1007/S11042-010-0645-5.

[5] S. Vijayarani and A. Sakila, “Multimedia Mining Research – An Overview,” *International Journal of Computer Graphics & Animation*, vol. 5, no. 1, pp. 69–77, Jan. 2015, doi: 10.5121/ijcga.2015.5105.

[6] D. Kexue, Z. Jun, and L. Guohui, “Video mining: Concepts, approaches and applications,” *MMM2006: 12th International Multi-Media Modelling Conference - Proceedings*, vol. 2006, pp. 477–480, 2006, doi: 10.1109/MMMC.2006.1651376.

[7] P. Shivakumara, A. Alaei, and U. Pal, “Mining text from natural scene and video images: A survey,” *Wiley Interdiscip Rev Data Min Knowl Discov*, vol. 11, no. 6, p. e1428, Nov. 2021, doi: 10.1002/WIDM.1428.

[8] Andrea. Cavallaro and Emilio. Maggio, “Video tracking : theory and practice,” 2010, Accessed: Dec. 07, 2022. [Online]. Available: https://www.wiley.com/en-sg/Video+Tracking%3A+Theory+and+Practice-p-9780470749647

[9] S. H. Shaikh, K. Saeed, and N. Chaki, “Moving Object Detection Using Background Subtraction,” *SpringerBriefs in Computer Science*, vol. 0, no. 9783319073859, pp. 15–23, 2014, doi: 10.1007/978-3-319-07386-6_3.

[10] S. Gollapudi, “Learn Computer Vision Using OpenCV,” *Learn Computer Vision Using OpenCV*, 2019, doi: 10.1007/978-1-4842-4261-2.

[11] M. Hassaballah and A. I. Awad, “Deep Learning in Computer Vision,” *Deep Learning in Computer Vision*, Mar. 2020, doi: 10.1201/9781351003827.

[12] T. S. Madhulatha, “An Overview on Clustering Methods,” *IOSR Journal of Engineering*, vol. 02, no. 04, pp. 719–725, May 2012, doi: 10.48550/arxiv.1205.1117.

[13] D. Xu and Y. Tian, “A Comprehensive Survey of Clustering Algorithms,” *Annals of Data Science 2015 2:2*, vol. 2, no. 2, pp. 165–193, Aug. 2015, doi: 10.1007/S40745-015-0040-1.

[14] K. M. Archana Patel and P. Thakral, “The best clustering algorithms in data mining,” *undefined*, pp. 2042–2046, Nov. 2016, doi: 10.1109/ICCSP.2016.7754534.

[15] J. Arora, K. Khatter, and M. Tushir, “Fuzzy C-means clustering strategies: A review of distance measures,” *Advances in Intelligent Systems and Computing*, vol. 731, pp. 153–162, 2019, doi: 10.1007/978-981-10-8848-3_15.

[16] H. S. Khaing and T. Thein, “An Efficient Clustering Algorithm for Moving Object Trajectories,” *undefined*, 2014.

[17] N. Pelekis, I. Kopanakis, E. E. Kotsifakos, E. Frentzos, and Y. Theodoridis, “Clustering trajectories of moving objects in an uncertain world,” *Proceedings - IEEE International Conference on Data Mining, ICDM*, pp. 417–427, 2009, doi: 10.1109/ICDM.2009.57.

[18] S. U. Rehman, A. U. Khan, and S. Fong, “Graph mining: A survey of graph mining techniques,” *undefined*, pp. 88–92, 2012, doi: 10.1109/ICDIM.2012.6360146.

[19] F. Yang, H. Fan, P. Chu, E. Blasch, and H. Ling, “Clustered Object Detection in Aerial Images.” pp. 8311–8320, 2019.

[20] Z. Liu and M. Barahona, “Graph-based data clustering via multiscale community detection,” *Appl Netw Sci*, vol. 5, no. 1, pp. 1–20, Dec. 2020, doi: 10.1007/S41109-019-0248-7/TABLES/4.

[21] F. Diot, E. Fromont, B. Jeudy, E. Marilly, and O. Martinot, “Graph mining for object tracking in videos,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 7523 LNAI, no. PART 1, pp. 394–409, 2012, doi: 10.1007/978-3-642-33460-3_31.

[22] Y. M. Yousif, A. Mukbil, and J. P. Müller, “OfflineMOT: A Python package for multiple objects detection and tracking from bird view stationary drone videos,” *J Open Source Softw*, vol. 7, no. 74, p. 4099, Jun. 2022, doi: 10.21105/JOSS.04099.

[23] T. Yin, X. Zhou, and P. Krahenbuhl, “Center-Based 3D Object Detection and Tracking.” pp. 11784–11793, 2021. Accessed: Dec. 07, 2022. [Online]. Available: https://github.com/tianweiy/CenterPoint.

[24] T. L. Dang, G. T. Nguyen, and T. Cao, “Object tracking using improved deep sort yolov3 architecture,” *ICIC Express Letters*, vol. 14, no. 10, pp. 961–969, Oct. 2020, doi: 10.24507/ICICEL.14.10.961.

[25] M. Elhoseny, “Multi-object Detection and Tracking (MODT) Machine Learning Model for Real-Time Video Surveillance Systems,” *Circuits, Systems, and Signal Processing 2019 39:2*, vol. 39, no. 2, pp. 611–630, Aug. 2019, doi: 10.1007/S00034-019-01234-7.

[26] N. T. Le, B. Vo, L. B. Q. Nguyen, H. Fujita, and B. Le, “Mining weighted subgraphs in a single large graph,” *Inf Sci (N Y)*, vol. 514, pp. 149–165, Apr. 2020, doi: 10.1016/J.INS.2019.12.010.

[27] C. Zhao, Z. Zhang, P. Xu, T. Zheng, and J. Guo, “Kaleido: An Efficient Out-of-core Graph Mining System on A Single Machine,” *undefined*, vol. 2020-April, pp. 673–684, Apr. 2020, doi: 10.1109/ICDE48307.2020.00064.

[28] Israa Hadi, “Objects Clustering of Movie Using Graph Mining Technique.” University of Babylon Repository, 2014. Accessed: Dec. 07, 2022. [Online]. Available: http://repository.uobabylon.edu.iq/papers/publication.aspx?Pubid=6092

[29] Z. Liu and M. Barahona, “Graph-based data clustering via multiscale community detection,” *Appl Netw Sci*, vol. 5, no. 1, pp. 1–20, Dec. 2020, doi: 10.1007/S41109-019-0248-7/TABLES/4.

[30] J. Huang, S. Yang, T. J. Mu, and S. M. Hu, “ClusterVO: Clustering Moving Instances and Estimating Visual Odometry for Self and Surroundings,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2165–2174, Mar. 2020, doi: 10.48550/arxiv.2003.12980.

[31] C. Wang, S. Pan, R. Hu, G. Long, J. Jiang, and C. Zhang, “Attributed Graph Clustering: A Deep Attentional Embedding Approach,” *IJCAI International Joint Conference on Artificial Intelligence*, vol. 2019-August, pp. 3670–3676, Jun. 2019, doi: 10.48550/arxiv.1906.06532.

[32] C. Xie, Y. Xiang, Z. Harchaoui, and D. Fox, “Object Discovery in Videos as Foreground Motion Clustering.” pp. 9994–10003, 2019.

[33] G. R. C. Wood and R. E. Woods, “Chapter 5: Image restoration,” *Digital Image Processing*, 2017.

[34] B. Desai, U. Kushwaha, and S. Jha, “Image Filtering-Techniques, Algorithm and Applications,” 2020. doi: 20.18001.GSJ.2020.V7I11.20.36036.

[35] U. Erkan, L. Gökrem, and S. Enginoğlu, “Different applied median filter in salt and pepper noise,” *COMPUTERS & ELECTRICAL ENGINEERING*, vol. 70, pp. 789–798, Aug. 2018, doi: 10.1016/J.COMPELECENG.2018.01.019.

[36] J. A. Lian, “Two adaptive schemes for image sharpening,” *2019 IEEE 2nd International Conference on Information and Computer Technologies, ICICT 2019*, pp. 122–125, May 2019, doi: 10.1109/INFOCT.2019.8711269.

[37] E. Dougherty, “Mathematical Morphology in Image Processing,” *Mathematical Morphology in Image Processing*, Oct. 2018, doi: 10.1201/9781482277234/MATHEMATICAL-MORPHOLOGY-IMAGE-PROCESSING-BRIAN-THOMPSON-EDWARD-DOUGHERTY.

[38] Z. Zou, Z. Shi, Y. Guo, J. Ye, and S. Member, “Object Detection in 20 Years: A Survey,” pp. 1–39, May 2019, doi: 10.48550/arxiv.1905.05055.

[39] I. E. Olatunji and C.-H. Cheng, “Video Analytics for Visual Surveillance and Applications: An Overview and Survey,” pp. 475–515, 2019, doi: 10.1007/978-3-030-15628-2_15.

[40] A. A. Alsanabani, M. A. Ahmed, and A. M. al Smadi, “Vehicle Counting Using Detecting-Tracking Combinations: A Comparative Analysis,” *ACM International Conference Proceeding Series*, vol. PartF168342, pp. 48–54, Dec. 2020, doi: 10.1145/3447450.3447458.

[41] Z. Q. Zhao, P. Zheng, S. T. Xu, and X. Wu, “Object Detection With Deep Learning: A Review,” *IEEE Trans Neural Netw Learn Syst*, vol. 30, no. 11, pp. 3212–3232, Nov. 2019, doi: 10.1109/TNNLS.2018.2876865.

[42] A. Dhillon and G. K. Verma, “Convolutional neural network: a review of models, methodologies and applications to object detection,” *Progress in Artificial Intelligence 2019 9:2*, vol. 9, no. 2, pp. 85–112, Dec. 2019, doi: 10.1007/S13748-019-00203-0.

[43] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-December, pp. 779–788, Dec. 2016, doi: 10.1109/CVPR.2016.91.

[44] A. C. Biju, A. K. George, and V. K. H., “Object Detection Using YOLO Algorithm,” *Kristu Jayanti Journal of Computational Sciences (KJCS)*, vol. 2, no. 2022, pp. 25–37, Jun. 2022, Accessed: Dec. 08, 2022. [Online]. Available: http://www.kristujayantijournal.com/index.php/ijcs/article/view/2219

[45] J. Redmon and A. Farhadi, “YOLOv3: An Incremental Improvement,” Apr. 2018, doi: 10.48550/arxiv.1804.02767.

[46] L. Zhao and S. Li, “Object Detection Algorithm Based on Improved YOLOv3,” *Electronics 2020, Vol. 9, Page 537*, vol. 9, no. 3, p. 537, Mar. 2020, doi: 10.3390/ELECTRONICS9030537.

[47] E. Dong, Y. Zhu, Y. Ji, and S. Du, “An improved convolution neural network for object detection using Yolov2,” *Proceedings of 2018 IEEE International Conference on Mechatronics and Automation, ICMA 2018*, pp. 1184–1188, Oct. 2018, doi: 10.1109/ICMA.2018.8484733.

[48] S. Challa, M. R. Morelande, D. Mušicki, and R. J. Evans, “Fundamentals of Object Tracking,” *Fundamentals of Object Tracking*, vol. 9780521876285, pp. 1–375, Jan. 2011, doi: 10.1017/CBO9780511975837.

[49] W. Luo, J. Xing, A. Milan, X. Zhang, W. Liu, and T. K. Kim, “Multiple object tracking: A literature review,” *Artif Intell*, vol. 293, p. 103448, Apr. 2021, doi: 10.1016/J.ARTINT.2020.103448.

[50] K. Bernardin and R. Stiefelhagen, “Evaluating multiple object tracking performance: The CLEAR MOT metrics,” *EURASIP J Image Video Process*, vol. 2008, 2008, doi: 10.1155/2008/246309.

[51] S. R. Balaji and S. Karthikeyan, “A survey on moving object tracking using image processing,” *Proceedings of 2017 11th International Conference on Intelligent Systems and Control, ISCO 2017*, pp. 469–474, Feb. 2017, doi: 10.1109/ISCO.2017.7856037.

[52] S. Krebs, B. Duraisamy, and F. Flohr, “A survey on leveraging deep neural networks for object tracking,” *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, vol. 2018-March, pp. 411–418, Mar. 2018, doi: 10.1109/ITSC.2017.8317904.

[53] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, “Simple online and realtime tracking,” *Proceedings - International Conference on Image Processing, ICIP*, vol. 2016-August, pp. 3464–3468, Aug. 2016, doi: 10.1109/ICIP.2016.7533003.

[54] N. Wojke, A. Bewley, and D. Paulus, “Simple online and realtime tracking with a deep association metric,” *Proceedings - International Conference on Image Processing, ICIP*, vol. 2017-September, pp. 3645–3649, Feb. 2018, doi: 10.1109/ICIP.2017.8296962.

[55] C. Parent *et al.*, “Semantic trajectories modeling and analysis,” *ACM Computing Surveys (CSUR)*, vol. 45, no. 4, Aug. 2013, doi: 10.1145/2501654.2501656.

[56] A. Bera, S. Kim, and D. Manocha, “Efficient Trajectory Extraction and Parameter Learning for Data-Driven Crowd Simulation,” pp. 3–5, 2015, doi: 10.5555/2788890.2788903.

[57] M. Plaue, M. Chen, G. Bärwolff, and H. Schwandt, “Trajectory extraction and density analysis of intersecting pedestrian flows from video recordings,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6952 LNCS, pp. 285–296, 2011, doi: 10.1007/978-3-642-24393-6_24/COVER.

[58] C. Parent *et al.*, “Semantic trajectories modeling and analysis,” *ACM Computing Surveys (CSUR)*, vol. 45, no. 4, Aug. 2013, doi: 10.1145/2501654.2501656.

[59] N. Magdy, M. A. Sakr, T. Mostafa, and K. El-Bahnasy, “Review on trajectory similarity measures,” *undefined*, pp. 613–619, Feb. 2015, doi: 10.1109/INTELCIS.2015.7397286.

[60] M. Riyadh, N. Mustapha, and D. Riyadh, “Review of Trajectories Similarity Measures in Mining Algorithms,” *NTCCIT 2018 - Al Mansour International Conference on New Trends in Computing, Communication, and Information Technology*, pp. 36–40, Apr. 2019, doi: 10.1109/NTCCIT.2018.8681186.

[61] T. Jiang and C. Tomasi, “Robust shape normalization based on implicit representations,” *Proceedings - International Conference on Pattern Recognition*, 2008, doi: 10.1109/ICPR.2008.4761202.

[62] L. Zhao and G. Shi, “A method for simplifying ship trajectory based on improved Douglas–Peucker algorithm,” *Ocean Engineering*, vol. 166, pp. 37–46, Oct. 2018, doi: 10.1016/j.oceaneng.2018.08.005.

[63] “Ramer–Douglas–Peucker Algorithm.” https://ayanbag.com/rdp-algo (accessed Dec. 08, 2022).

[64] G. Reyes Zambrano, “GPS trajectory compression algorithm,” *Communications in Computer and Information Science*, vol. 959, pp. 57–69, 2019, doi: 10.1007/978-3-030-12018-4_5/COVER.

[65] JIAWEI. HAN, “DATA MINING : concepts and techniques.,” 2022.

[66] TERESA. HAYNES, “DOMINATION IN GRAPHS : core concepts.,” 2022.

[67] N. Huang and S. Villar, “A Short Tutorial on The Weisfeiler-Lehman Test And Its Variants,” *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2021-June, pp. 8533–8537, Jan. 2022, doi: 10.1109/icassp39728.2021.9413523.

[68] Z. Shaul and S. Naaz, “cgSpan: Closed Graph-Based Substructure Pattern Mining,” *Proceedings - 2021 IEEE International Conference on Big Data, Big Data 2021*, pp. 4989–4998, Dec. 2021, doi: 10.48550/arxiv.2112.09573.

[69] N. Asrafi, “Comparing performances of graph mining algorithms to detect malware,” *ACMSE 2019 - Proceedings of the 2019 ACM Southeast Conference*, pp. 268–269, Apr. 2019, doi: 10.1145/3299815.3314485.

[70] A. Ghosal, A. Nandy, A. K. Das, S. Goswami, and M. Panday, “A Short Review on Different Clustering Techniques and Their Applications,” *undefined*, vol. 937, pp. 69–83, 2019, doi: 10.1007/978-981-13-7403-6_9.

[71] K. Kameshwaran and K. Malarvizhi, “Survey on Clustering Techniques in Data Mining,” vol. 5, no. 2, pp. 2272–2276, 2014.

[72] E. H. Ruspini, J. C. Bezdek, and J. M. Keller, “Fuzzy clustering: A historical perspective,” *IEEE Comput Intell Mag*, vol. 14, no. 1, pp. 45–55, Feb. 2019, doi: 10.1109/MCI.2018.2881643.

[73] M. Obiedat, A. Al-Yousef, A. Khasawneh, N. Hamadneh, and A. Aljammal, “Using fuzzy c-means for weighting different fuzzy cognitive maps,” *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 5, pp. 545–551, 2020, doi: 10.14569/IJACSA.2020.0110569.

[74] J. C. Bezdek, R. Ehrlich, and W. Full, “FCM: The fuzzy c-means clustering algorithm,” *Comput Geosci*, vol. 10, no. 2–3, pp. 191–203, Jan. 1984, doi: 10.1016/0098-3004(84)90020-7.

[75] B. Karasulu, “Review and Evaluation of Well-Known Methods for Moving Object Detection and Tracking in Videos,” 2010.

[76] B. Karasulu and S. Korukoglu, “Performance Evaluation Software,” 2013, doi: 10.1007/978-1-4614-6534-8.

[77] D. M. W. Powers and Ailab, “Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation,” Oct. 2020, doi: 10.48550/arxiv.2010.16061.

[78] A. Saxena *et al.*, “A review of clustering techniques and developments,” *Neurocomputing*, vol. 267, pp. 664–681, Dec. 2017, doi: 10.1016/J.NEUCOM.2017.06.053.

[79] C. Zhang, L. Chen, Y. P. Zhao, Y. Wang, and C. L. P. Chen, “Graph Enhanced Fuzzy Clustering for Categorical Data Using a Bayesian Dissimilarity Measure,” *IEEE Transactions on Fuzzy Systems*, 2022, doi: 10.1109/TFUZZ.2022.3189831.

[80] I. Pauletić, L. Načinović Prskalo, and M. Brkić Bakarić, “An Overview of Clustering Models with an Application to Document Clustering,” *Proceedings of the 42nd International Convention MIPRO 2019, Computers in Education*, p. 1928, 2019, Accessed: Dec. 08, 2022. [Online]. Available: https://www.bib.irb.hr/1003000

[81] M. Al-Mhairat, M. Almhairat1, R. Alabbadi2, R. Shaban3, and A. Alqudah4, “Performance Evaluation of clustering Algorthims,” 2019. [Online]. Available: https://www.researchgate.net/publication/334971445

[82] “VB100 Video Bird Dataset.” https://arma.sourceforge.net/vb100/ (accessed Dec. 08, 2022).

[83] Z. Ge *et al.*, “Exploiting Temporal Information for DCNN-based Fine-Grained Object Classification,” *2016 International Conference on Digital Image Computing: Techniques and Applications, DICTA 2016*, Aug. 2016, doi: 10.1109/DICTA.2016.7797039.

[84] “(412) Transhumance in Italy and cow bells - YouTube.” https://www.youtube.com/watch?v=kninnU5X6mg&ab_channel=TerritorioeGoverno (accessed Dec. 08, 2022).

الخلاصة

تعتبر الرؤية الحاسوبية من المجالات العلمية المهمة في العصر الحديث لأن عناصر الفيديو تحتوي على معلومات غنية ومهمة يمكن من خلالها الحصول على المعرفة والبيانات للإشارة إلى كمية هائلة من المعلومات المفيدة. تعتبر عملية تمييز وفصل المعلومات المكتشفة فقط من المشاكل المعقدة والمعروفة. تعتبر مشكلة تصنيف وتجميع الأجسام المتحركة في بيانات الفيديو أيضًا مهمة معقدة تتطلب آليات وعمليات وخوارزميات لغرض حلها والحصول على نتائج متميزة قدر الإمكان.

في هذه الرسالة، تم اقتراح نظام لغرض تجميع الأجسام المتحركة بناءً على سلوكها باستخدام خوارزمية تعدين الرسم البياني. تم اقتراح خوارزمية جديدة لغرض استخراج البيانات الكبيرة التي يتم تمثيلها باستخدام الرسم البياني. علاوة على ذلك، تم اقتراح خوارزمية لغرض تقليل البيانات واستخراج البيانات المهمة فقط. تم أيضًا تكييف بعض الخوارزميات المستخدمة في النظام المقترح من أجل زيادة أدائها.

يقوم النظام المقترح أولاً بتقسيم إدخال الفيديو إلى إطارات متسلسلة. المرحلة الثانية هي تطبيق بعض عمليات المعالجة المسبقة لتحسين جودة الإطار (الصورة الثابتة). المرحلة الثالثة هي تطبيق YOLO للكشف عن كائنات متعددة وكائنات تتبع Deep-SORT لاكتشاف وتتبع الكائنات ذات الفئات المختلفة. المرحلة الرابعة هي بناء مسار لكل كائن وتطبيق خوارزمية تطبيع الشكل الجديدة المقترحة. المرحلة الخامسة هي استخراج ميزات المسارات وإنشاء رسم بياني لها، يتم تخزين بيانات الرسم البياني في قاعدة بيانات الرسم البياني. تتمثل المرحلة السادسة في تطبيق خوارزمية جديدة مقترحة لتعدين الرسم البياني لاستخراج البيانات المهتمة. أخيرًا، يتم تطبيق وسائل c الضبابية على البيانات العنقودية في عدد مختلف من المجموعات.

تشير النتائج التجريبية إلى أن النظام المقترح قوي وذو أداء عالٍ. تعد الخوارزميات المستخدمة في الكشف والتتبع هي الأفضل من خلال مقارنتها بخوارزميات الكشف والتتبع الأخرى، ولديها دقة عالية. علاوة على ذلك، تُظهر خوارزمية التطبيع المقترحة أنه يتم تجاهل حوالي 50 ٪ من النقاط غير الغنية. علاوة على ذلك، أظهرت الخوارزمية المقترحة لتعدين الرسم البياني أداءً عاليًا لاستخراج البيانات المهتمة. بالإضافة إلى ذلك، أظهرت الخوارزمية المقترحة لتعدين الرسم البياني أداءً عاليًا بأكثر من 95٪ لاستخراج البيانات المهمة.

جمهورية العراق
وزارة التعليم العالي والبحث العلمي
جامعة بابل- كلية تكنولوجيا المعلومات
قسم البرمجيات

تجميع الكائنات المتحركة المتعددة استنادًا إلى مساراتها باستخدام خوارزمية تنقيب الرسومات

أطروحة مقدمة

الى مجلس كلية تكنولوجيا المعلومات في جامعة بابل وهي جزء من متطلبات نيل درجة الدكتوراه فلسفة في تكنولوجيا المعلومات / برمجيات

من قبل

مصطفى أسعد حسن علوان

إشراف

أ. د. توفيق عبد الخالق عباس عبد الرضا

2022 م

1444 هـ