**CHAPTER 1**

**INTRODUCTION**

Moving article identification is a PC innovation identified with computer vision, image processing , neural network that manages identifying examples of semantic objects of a specific class (, for example, human, vehicles and so on) in computerized picture or video. All around inquired about areas incorporate vehicle identification, pedestrian recognition. Moving article location has numerous applications in the space of computer vision, including image recovery and video reconnaissance. Moving article recognition for certifiable applications is as yet a difficult issue. While late research informational collections builds the measure of preparing sets and testing guides to draw nearer to this present reality issues, the capacity of finders to process huge informational indexes in sensible time turns into another significant issue other than exactness.

Object tracking is the first step in surveillance systems, navigation systems and object recognition. There is a tremendous noteworthiness of object tracking in real time environment, as it enables several applications for example, to give better feeling that utilizes visual data, security and reconnaissance to perceive individuals, to break down shopping conduct of clients in retail space, video reflection to acquire programmed explanation of recordings, to create article based rundowns, traffic management for investigate stream, to recognize mishaps, video altering to dispose of unwieldy human administrator communication, to plan cutting edge video impacts.Video tracking is the way toward finding a moving item or numerous objects after some time utilizing a camera. It has an assortment of employments, some of which are human-Computer collaboration, and video correspondence and Compression, increased reality, traffic control, medical imaging and video altering. Video tracking can be a tedious procedure because of the measure of information that is contained in video. Adding further to the multifaceted nature is the conceivable need to utilize object recognition for following, a difficult issue in its own.

Simultaneous tracking of multiple objects is a condition of-workmanship issue in the field of computer vision. The issues of tracking single item and tracking various articles are not same. For tracking various objects, bunches of issue can emerge because of unexpected object movement, interaction among the different objects, drifting of objects and so on. Some ongoing work has proposed answer to limit these issues. The primary objective of this paper is breaking down the ongoing work in numerous articles with those difficulties.

The goal of video tracking is to partner target questions in back to back video outlines these outlines can be called as video frames. The affiliation can be particularly troublesome when the objects are moving quick with respect to the edge rate. Another circumstance that expands the unpredictability of the issue is the point at which the followed object changes direction after some time. For these circumstances video tracking frameworks generally utilize a movement model which portrays how the picture of the object may change for various potential movements of the object.

The multiplication of powerful computer and the expanding requirement for mechanized observation frameworks have created a lot of enthusiasm for article following calculations. A portion of the undertakings that utilization object tracking are:

* **Motion-based recognition**: human identification based on gait, automatic object detection, etc.,
* **Automated surveillance**: monitoring a scene to detect suspicious activities or unlikely events,
* **Video indexing**: automatic annotation and retrieval of videos for multimedia databases,
* **Human-computer interaction**: gesture recognition, eye gaze tracking for data input to computers,
* **Traffic monitoring**: real time gathering of traffic statistics to direct traffic flow, and
* **Vehicle navigation**: navigating vehicles with path planning and obstacle avoidance capabilities

Tracking can be defined as the problem of estimating the trajectory of an object as the object moves around a scene. Simply stated, want to know where the object is in the image at each instant in time. Tracking objects can be a complex problem due to

* Loss of information caused by projection of the 3D world on a 2D image
* Noise in images
* Complex object motion
* Partial and full object occlusions
* Complex object shapes
* Scene illumination changes and
* Real time processing requirements.

**PROBLEM STATEMENT**

Tracking-by-detection approach is used for tracking object recently. In this method CNN is trained with plenty of video data sets and used for detecting the object in the video frames. This CNN-based tracking by-detection approach achieves a breakthrough in tracking performance, but it suffers from an inefficient exhaustive search strategy that explores the region of interest and selects the best candidate by referring to the scores obtained by the network. Due to this tracking cannot be done in real time and tracking speed is very low. In this project we deal with problem and propose mechanism to increase the speed of tracking without GPU architecture.

The aim of visual object tracking is to find a bounding box tightly containing the target moving object in every frame of a video, which is one of the fundamental problems in the computer vision field. In recent years, there have been many advances in visual tracking algorithms, but there are still many challenging issues arising from diverse tracking obstacles, such as motion blur, occlusion, illumination change, and background clutter. Conventional tracking methods using low-level hand-crafted features encounter the above-mentioned tracking obstacles because of their insufficient feature representation. Convolutional neural networks (CNNs) have been proposed for robust tracking and vastly improved tracking performance with the help of rich feature representation. A CNN pertained on a classification data set is not sufficient to solve the problem of adaptation to object shape deformation and illumination changes in tracking, because the deep CNN is not appropriate for online adaptation. A tracking-by-detection approach using CNNs trained with plenty improves the ability to distinguish between the target and the background using sub networks that learn the discriminative features of the target and the background via online adaptation .This CNN-based tracking by-detection approach achieves a breakthrough in tracking performance, but it suffers from an inefficient exhaustive search strategy that explores the region of interest and selects the best candidate by referring to the scores obtained by the network. To overcome this exhaustive search problem with the tracking-by-detection methods, we introduce an action driven tracking mechanism that actively pursues the target movement considering the context change of the image within the bounding box.

There is a critical problem with constructing training data for a deep CNN-based tracker. Deep CNN-based trackers require a large amount of training data in order to learn convolutional features from scratch. Even though there are plenty of video sequences, it is extremely expensive to annotate the target position in every frame for the construction of training data. If author can train a deep CNN-based tracker using a partially labeled video sequence, a variety of videos can be utilized with less effort for the training.in contrast to the existing deep CNN-based trackers using SL, Author try to develop a reinforcement learning (RL) scheme to utilize the partially labeled video sequences effectively for training our action-driven deep tracker.

Tracking using convolutional neural networks (CNNs) have been proposed for robust tracking and vastly improved tracking performance with the help of rich feature representation by deep hidden layers. CNN is trained with objects and it can recognize the target object in a video frame.



**Fig 1 :Visual tracking controlled by sequential actions.**

The ADNet is designed to generate actions to find the location and the size of the target object in a new frame. The ADNet learns the policy that selects the optimal actions to track the target from the state of its current position. In the ADNet, the policy network is designed with a CNN,in which the input is an image patch cropped at the position of the previous state and the output is the probability distribution of actions, including translation and scale changes. This action-selecting process has searching steps than sliding window or candidate sampling approaches. The whole training framework is composed of an SL(Supervised Learning) stage and an RL stage.

In the SL stage, author train the network to select actions to track the position of the target using samples extracted from training videos. In SL step, the network learns to track general objects without sequential information. In the RL stage, the trained network in the SL stage is used as an initial network. Sample for RL are obtained by training the performing tracking simulation on training sequences. The network is trained with deep RL based on a policy gradient , using the rewards obtained during the tracking simulation.

The prior work initially suggests the tracking mechanism by selecting sequential actions and the network architecture. In this paper ,the author supplement the related works by adding action-driven methods adopted in computer vision. The preliminary version evaluated the proposed tracker in general situations, the extended work includes evaluations with the various attributes of tracking scenes. Author investigate the effect of the movement size of the action on tracking performance and speed and analyze the action dynamics factors in the fc6(Fedora Core) layer to examine the impact of the actions. For self-evaluation ,two additional variants of the ADNet has to be conducted and discussed:

1. The Policy gradient method is replaced with a value function-based method
2. The reward function is replaced with a continuous one.

**Disadvantages of Existing System**

CNN based tracking takes lot of time to detect object and not suitable for real time performance.

**NEURAL NETWORK**

Neural networks are an approach to computing that involves developing mathematical structures with the ability to learn. Neural network have the remarkable ability to derive meaning from complicated or imprecise data and can be used to extract pattern and detect trends that are too complex to be noticed by either human or other computer techniques. A trained neural network can be thought as an expert in the category of information it has been given to analyze. Neural networks have broad applicability’s to real world business problem and have already been successfully applied in many industries. Neural networks use a set of processing elements analogous to neurons in the brain. These processing elements are interconnected in a network that can identify patterns in data. These distinguishes neural network from other computing programs that simply follow instructions in a fixed sequential order.

**EXISTING SYSTEM**

The main contributions of this paper are summarized as follows. The action-driven deep tracker is proposed for the first time to dynamically track the target object by pursuing actions instead of tracking-by-detection scheme. We cast the visual tracking problem as a Markov decision process (MDP) and design a deep network architecture to realize the decision process. Deep RL algorithm is developed to train the ADNet with partially labeled data in the SS setting. The proposed deep tracker can control the tradeoff between tracking performance and computational complexity by simply changing the meta parameter in tracking. The proposed tracker achieves a state-of-the-art performance with much more efficient searching complexity than the existing deep network-based trackers using a tracking-by-detection strategy. Also, the fast version of the proposed method operates in real time on graphics processing units (GPUs), outperforming the state-of-the-art real-time trackers.

**CHAPTER 2**

**LITERATURE SURVEY**

**[1] . H. Grabner, M. Grabner, and H. Bischof, “Real-time tracking via on-line boosting,” in Proc. Brit. Mach. Vis. Conf., 2006, vol. 1. no. 5, p.6.**

The efficient and robust tracking of objects in complex environments is important for a variety of applications including video surveillance , autonomous driving or human-computer interaction. Thus it is a great challenge to design robust visual tracking methods which can cope with the inevitable variations that can occur in natural scenes such as changes in the illumination, changes of the shape, changes in the viewpoint, reflectance of the object or partial occlusion of the target. Moreover tracking success or failure may also depend on how distinguishable an object is from its background. Stated differently, if the object is very distinctive, a simple tracker may already fulfill the requirements. However, having objects similar to the background requires more sophisticated features. As a result there is the need for trackers which can handle on the one hand all possible variations of appearance changes of the target object and on the other hand are able to reliably cope with background clutter

**[2] .H. Grabner, C. Leistner, and H. Bischof, “Semi-supervised on-line boosting for robust tracking,” in Proc. Eur. Conf. Comput. Vis., 2008,pp. 234–247.**

Designing robust tracking methods is still an open issue, especially considering various complicated variations that may occur in natural scenes, e.g., shape and appearance changes of the object, illumination variations, partial occlusions of the object, cluttered scenes, etc. Recently tracking has been formulated as a classification problem, i.e., the task of tracking is to optimally separate in each frame the object from the background (e.g., Avidan [1] used support vector machines). Also, feature based tracking methods are formulated as classification tasks, i.e., the work of Lepetit et.al. [2] uses randomized trees and ferns based on pixel pairs [3] to discriminate key points by classifiers. In these approaches, the object to be tracked is trained a priori. The main motivation for using classifiers in these approaches is the increased speed, i.e., the time is spent at the training stage and a fast classifier is available at the tracking stage. All these approaches use off-line training, which has two important limitations.

**[3] . B. Babenko, M.-H. Yang, and S. Belongie, “Robust object tracking with online multiple instance learning,” IEEE Trans. Pattern Anal. Mach.Intell., vol. 33, no. 8, pp. 1619–1632, Aug. 2011.**

Online tracking of an object in video is a difficult task because the objects appearance can change a lot across frames. For example, the object might undergo changes in illumination, rotation, or occlusion. A successful system must learn to adapt to these changes in order to continue tracking the object. One way to perform tracking is tracking by detection, which uses a detector and an appearance model to find the object in each frame. The detector keeps both positive samples of the object and negative samples of the background so that it can tell the difference between the object and its surroundings. Its important that positive and negative examples are chosen carefully, because error in the appearance model can cause significant drift in tracking.

Multi-object tracking poses many other problems. For example, objects may occlude each other, or the path of one object may intersect with the path of another. Berclaz et al propose a graph-based solution to this problem, where each node of the graph represents a specific point in time and space and is connected via directed edges to the nodes directly preceding or following it in time

**[4] . D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, “Visual object tracking using adaptive correlation filters,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2010, pp. 2544–2550.**

Visual tracking has many practical applications in video processing. When a target is located in one frame of a video, it is often useful to track that object in subsequent frames. Every frame in which the target is successfully tracked provides more information about the identity and the activity of the target. Because tracking is easier than detection, tracking algorithms can use fewer computational resources than running an object detector on every frame. Visual tracking has received much attention in recent. years. A number of robust tracking strategies have been proposed that tolerate changes in target appearance and track targets through complex motions

**[5] . M. Danelljan, G. Häger, F. Khan, and M. Felsberg, “Accurate scale estimation for robust visual tracking,” in Proc. Brit. Mach. Vis. Conf., Nottingham, U.K., Sep. 2014.**

Visual object tracking is a popular problem in computer vision. The problem involves estimating the location of a visual target in each frame of an image sequence. Despite significant progress in recent years, the problem is still difficult due to factors such as partial occlusion, deformation, motion blur, fast motion, illumination variation, background clutter and scale variations. Most existing approaches provide inferior performance when encountered with large scale variations in complex image sequences. Tracking-by-detection methods have shown to provide excellent tracking performance. These approaches work by posing the task of target localization as a classification problem. The decision boundary is obtained by learning a discriminative classifier online using image patches from both the target and the background

**[6] . J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, “High-speed tracking with kernelized correlation filters,” IEEE Trans. Pattern Anal.Mach. Intell., vol. 37, no. 3, pp. 583–596, Mar. 2015.**

The task of tracking, a crucial component of many computer vision systems, can be naturally specified as an online learning problem . Given an initial image patch containing the target, the goal is to learn a classifier to discriminate between its appearance and that of the environment. This classifier can be evaluated exhaustively at many locations, in order to detect it in subsequent frames. Of course, each new detection provides a new image patch that can be used to update the model. It is tempting to focus on characterizing the object of interest – the positive samples for the classifier. However, a core tenet of discriminative methods is to give as much importance, or more, to the relevant environment – the negative samples. The most commonly used negative samples are image patches from different locations and scales, reflecting the prior knowledge that the classifier will be evaluated under those conditions.

**[7] . Y. Wu, J. Lim, and M. H. Yang, “Object tracking benchmark,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 37, no. 9, pp. 1834–1848, Sep. 2015.**

Object tracking has been one of the most important and active research areas in the field of computer vision. A large number of tracking algorithms have been proposed in recent years with demonstrated success. However, the set of sequences used for evaluation is often not sufficient or is sometimes biased for certain types of algorithms. Many datasets do not have common ground-truth object positions or extents, and this makes comparisons among the reported quantitative results difficult. In addition, the initial conditions or parameters of the evaluated tracking algorithms are not the same, and thus, the quantitative results reported in literature are incomparable or sometimes contradictory.

To address these issues, The author has carry out an extensive evaluation of the state-of-the-art online object-tracking algorithms with various evaluation criteria to understand how these methods perform within the same framework. The Author has first construct a large dataset with ground-truth object positions and extents for tracking and introduce the sequence attributes for the performance analysis. Second, Author has integrate most of the publicly available trackers into one code library with uniform input and output formats to facilitate large-scale performance evaluation. Third,extensively evaluate the performance of 31 algorithms on 100 sequences with different initialization settings. By analyzing the quantitative results, we identify effective approaches for robust tracking and provide potential future research directions in this field.

**[8] . M. Kristan, et al., “The visual object tracking VOT2014 challenge results,” in Proc. Eur. Conf. Comput. Vis. Workshops, Berlin, Germany, Mar. 2014, pp. 191–217.**

Recently, several attempts have been made towards benchmarking the class of trackers considered in this paper. Most notable are the online tracking benchmark (OTB) by Wu et al and the experimental survey based on Amsterdam Library of Ordinary Videos (ALOV) by Smeulders et al. Both benchmarks compare a number of recent trackers using the source code The OTB contains a dataset containing 50 sequences and annotates each sequence globally with eleven visual attributes. Sequences are not per-frame annotated. The ALOV benchmark provides an impressive dataset with 315 sequences annotated with thirteen visual attributes. A drawback of this dataset is that some sequences contain cuts and ambiguously defined targets such as fireworks.The ALOV benchmark provides an impressive dataset with 315 sequences annotated with thirteen visual attributes. A drawback of this dataset is that some sequences contain cuts and ambiguously defined targets such as fireworks.

**[9] . Who Let The Dogs Out? Modeling Dog Behavior From Visual Data, Kiana Ehsani1 , Hessam Bagherinezhad1 , Joseph Redmon1 Roozbeh Mottaghi2 , Ali Farhadi1,2 1 University of Washington, 2 Allen Institute for AI (AI2)**

Computer vision research typically focuses on a few well defined tasks including image classification, object recognition, object detection, image segmentation, etc. These tasks have organically emerged and evolved over time as proxies for the actual problem of visual intelligence. Visual intelligence spans a wide range of problems and is hard to formally define or evaluate. As a result, the proxy tasks have served the community as the main point of focus and indicators of progress.

**[10].** **Y. Wu, J. Lim, and M. H. Yang, “Object tracking benchmark,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 37, no. 9, pp. 1834–1848, Sep. 2015.**

Object tracking has been one of the most important and active research areas in the field of computer vision. A large number of tracking algorithms have been proposed in recent years with demonstrated success. However, the set of sequences used for evaluation is often not sufficient or is sometimes biased for certain types of algorithms. Many datasets do not have common ground-truth object positions or extents, and this makes comparisons among the reported quantitative results difficult. In addition, the initial conditions or parameters of the evaluated tracking algorithms are not the same, and thus, the quantitative results reported in literature are incomparable or sometimes contradictory. To address these issues, we carry out an extensive evaluation of the state-of-the-art online object-tracking algorithms with various evaluation criteria to understand how these methods perform within the same framework. In this work, we first construct a large dataset with ground-truth object positions and extents for tracking and introduce the sequence attributes for the performance analysis. Second, we integrate most of the publicly available trackers into one code library with uniform input and output formats to facilitate large-scale performance evaluation. Third, we extensively evaluate the performance of 31 algorithms on 100 sequences with different initialization settings. By analysing the quantitative results, we identify effective approaches for robust tracking and provide potential future research directions in this field.

# SYSTEM REQUIREMENT SPECIFICATION

In this section all functional and non functional requirements of the software are documented.

**3.1 Functional Requirement**

Functional Requirement defines a function of a software system and how the system must behave when presented with specific inputs or conditions. These may include calculations, data manipulation and processing and other specific functionality. In this system following are the functional requirements:-

1. Any video can be uploaded.
2. The object to track is selected
3. The object is tracked in further frames.
4. An output video of tracked result is generated.
5. Performance is measured and plotted.

**3.2 Non-functional Requirement**

Non functional requirements are the requirements which are not directly concerned with the specific function delivered by the system. They specify the criteria that can be used to judge the operation of a system rather than specific behaviors. They may relate to emergent system properties such as reliability, response time and store occupancy.

Non functional requirements arise through the user needs, because of budget constraints, organizational policies, the need for interoperability with other software and hardware systems or because of external factors such as:-

1. Product Requirements

2. Organizational Requirements

3. User Requirements

4. Basic Operational Requirements

**3.2.1 Product Requirements**

* **Scalability:** Project should work for any number of frames in the video
* **Debugging Ability:** Any errors in tracking must be shown in a neat graph.
* **Modularity:** The project design should be very modular so that any new tracking algorithms can be added later on.

**3.2.2 Organizational Requirements**

**Process Standards:** IEEE standards are used to develop the application which uses most of the standard software for development for all over the world.

**Design Methods:** Design is one of the important stages in the software engineering process. This stage is the first step in moving from problem to the solution domain. In other words, starting with what is needed design takes us to work how to satisfy the needs.

The design of the system is perhaps the most critical factor affecting the quality of the software and has a major impact on the later phases, particularly testing and maintenance. We have to design the product with the standards which has been understood by the developers of the team.

**3.2.3 User Requirements**

## The user must be able to visualize Graphical User Interface Window.

## The user must be able to configure all the parameters with neat GUI.

**3.2.4 Basic Operational Requirements**

The users are those that perform the eight primary functions of systems engineering, with special emphasis on the operator as the key customer. Operational requirements will define the basic need and, at a minimum, will be related to these following points:-

**Mission profile or scenario:** The mission of the project is to design robust object tracking system

**Performance and related parameters:** The accuracy of tracking must be higher.

**Utilization environments:** The system will be very useful security and surveillance.

.

**CHAPTER 3**

**DESIGN**

System analysis is the method, by which we get some answers concerning the present issues, portrays things and necessities and evaluates the courses of action. It is the perspective about the affiliation and the issue it incorporates, a plan of advances those assistants in dealing with these issues. Feasibility study expects a key part in structure examination which gives the goal for layout and headway.

**4.1 Feasibility Study**

The datasets used for project testing is available in internet and the tools used as matured and easy to use. So there is risk in implementing the project

**4.1.1 Economical feasibility**

Since all the tools and dataset are open source and easily downloaded without any cost, there is no economic risk in the project.

**4.1.2 Technical Feasibility**

The project coding is in Matalb and we estimate around 2000 lines of code for the project.

**4.1.3 SOCIAL FEASIBILITY**

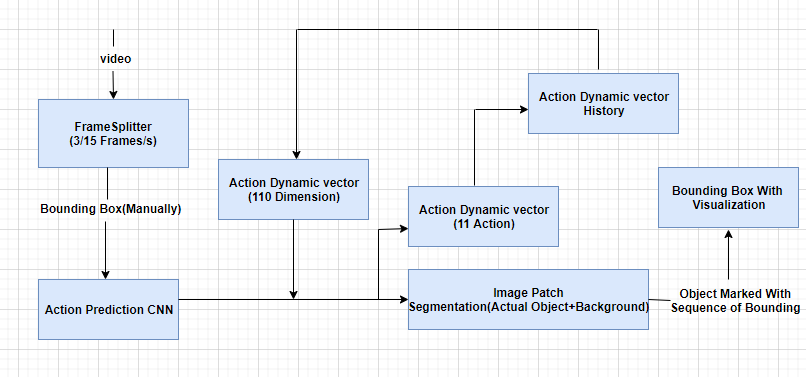
The project will be very useful at security establishments for tracking of people.

Design is a creative process, a good design is the key to effective system. The system design is defined as “The process of applying various techniques and principles for the purpose of defining a process or a system in sufficient detail to permit its physical realization”. Various design features are followed to develop the system. The design specification describes the features of the system, the components or elements of the system and their appearance to end-users.

**SYSTEM ARCHITECTURE**

System architecture is the conceptual design that defines the [structure](http://en.wikipedia.org/wiki/Structure) and [behavior](http://en.wikipedia.org/wiki/Behavior) of a [system](http://en.wikipedia.org/wiki/System). An architecture description is a formal description of a system, organized in a way that supports reasoning about the structural properties of the system. It defines the [system](http://en.wikipedia.org/wiki/System) components or building blocks and provides a plan from which products can be procured, and systems developed, that will work together to implement the overall system.

The System architecture is shown below.



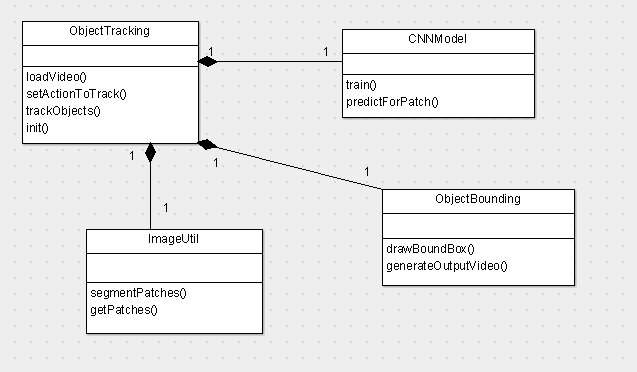
**Fig :Block Diagram of the System**

Video is an electronic medium for the recording, copying, playback, broadcasting, and display of moving visual media. The inputted is splitter in to number of frames at a time 3 to 5 frames are sent. First Frame is sent by manually drawing the bounding box and send over the action prediction convolution neural network, the CNN is just like the black box. Action Dynamic vector is maintained with 110 dimension we call it as ‘Dt’ this vector will be concatenated with the outcome of the CNN. That further produce two vectors, one vector contain 11 action and other vector which has 2 dimension which specifies foreground and background .Finally one action dynamic history is created and updated at each iteration of object tracking. Finally the object is tracked with the bounding box and the final output video will be displayed

**Classes Designed for the system**

A class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes , their attributes, and the relationships between the classes.

The class diagram is shown below.

**Fig 3.2 :Class Diagram of the object Tracking**

The class diagram has the following classes

The classes are

**Object Tracking :** This class is the user interface class. User can load a video , set action to track and track the object in the video.

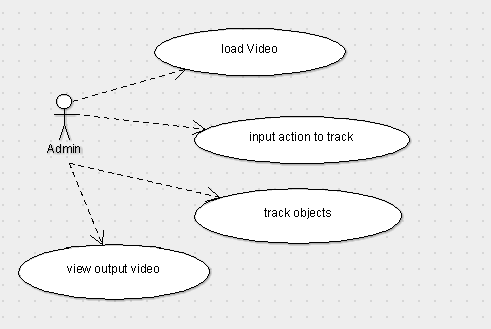
**CNN Model :** This model is trained with patches and the action. Based on the training, it can predict action for a image patch.

**Object Bounding :** This class draw bounding box around the tracked object and generate an output video.

**ImageUtil:** This class has functions for segmenting the image to patches

**Use case Diagram of the system**

A use case diagram is a type of behavioral diagram created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.



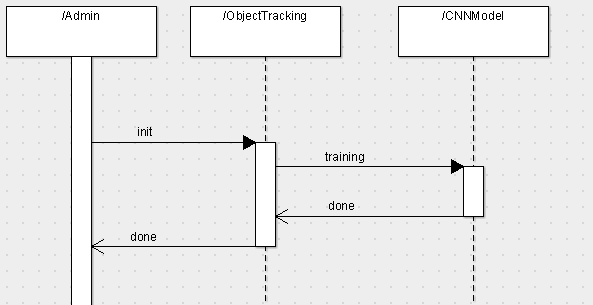
Admin is the user of the system. Admin can load the video, input the action to track ( left, right, up , down ) and track the objects. He can view the output object tracked video.

**Sequence diagram of system operation**

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart.

The sequence diagrams shows below.

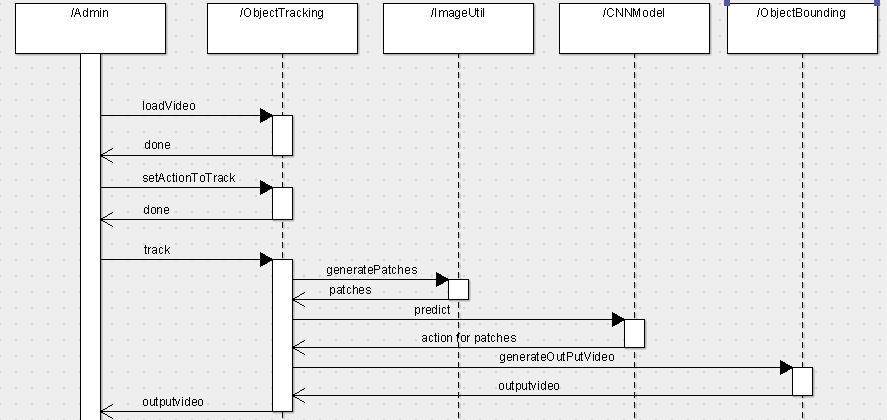
**Sequence diagram for initialization**



**Fig :Sequence Diagram for Basic Initialization**

Admin initialize the Object tracking where the CNN model is trained.

**Sequence diagram for object tracking**



**Fig:Sequence Diagram of the object Tracking**

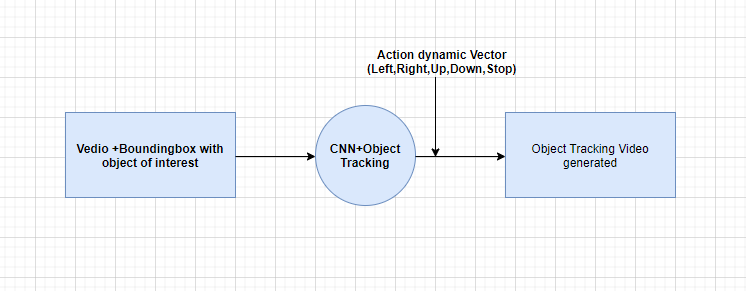
Admin loads the video and action to track and tracks the objects. The object put tracked output video is generated as output.

**Data Flow Diagram of the system**

A data-flow diagram (DFD) is a graphical representation of the "flow" of data through an information system. DFDs can also be used for the visualization of data processing (structured design). On a DFD, data items flow from an external data source or an internal data store to an internal data store or an external data sink, via an internal process.

**Level 0 Data flow diagram**

A context-level or level 0 data flow diagram shows the interaction between the system and external agents which act as data sources and data sinks. On the context diagram (also known as the Level 0 DFD) the system's interactions with the outside world are modeled purely in terms of data flows across the system boundary. The context diagram shows the entire system as a single process, and gives no clues as to its internal organization

  
**Fig: Level-0 Data Flow Diagram**

Object tracking is the process, it takes the video + action to track as input and generates the output video with object tracked using bounding box.

**Level 1 Data flow diagram**

The Level 1 DFD shows how the system is divided into sub-systems (processes), each of which deals with one or more of the data flows to or from an external agent, and which together provide all of the functionality of the system as a whole. It also identifies internal data stores that must be present in order for the system to do its job, and shows the flow of data between the various parts of the system.

The object tracking process is split to sub process as below



**Fig :Level-1 Data Flow Diagram**

**CHAPTER 4**

**IMPLEMENTATION**

Implementation is the process of creating an executable version of the software. Implementation may involve developing programs in high- level or low-level programming languages or making and adapting generic, off-the-shelf systems to meet the specific requirements. The execution stage of every project is the process of translating the detailed design into an executable code. The main goal of this stage is to translate the system design to possible working code by means of suitable programming language. It gives the complete information about programming language and building platform used. It also gives the detailed outline about top level and sub level modules.

The implementation stage needs to follow the few subsequent tasks as follows.

* Attentive planning.
* Examination of limitations and system.
* Design of technique to obtain changeover.
* Assessment of the replacement method.
* Preferential selection of platform
* Choose excellent language to build application.

**HARDWARE REQUIREMENTS:**

* Pentium or higher processor.
* 16 MB or more RAM.
* A standard keyboard and Microsoft compatible mouse.
* Video Graphics Array(VGA) monitor.
* If the user wants to save the created files a secondary storage medium can be used.
* Standard graphics Hardware/Software(GPU)

**SOFTWARE REQUIREMENTS:**

* MATLab Coding
* MATLab R2013 b or MATLab R2017a
* Ubuntu 14.04 or Higher Version
* Operating System(OS)- windows 7,8,8.1,10

**MATLAB**

MATLAB is a multi-paradigm, numerical computing environment and proprietary programming language developed by Math Works. MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including C, C++,.Net, Java, Fortran and Python. Programming paradigms are a way to classify programming languages based on their features. Languages can be classified into multiple paradigms. Numerical analysis is the study of algorithms that use numerical approximation (as opposed to symbolic manipulations) for the problems of mathematical analysis (as distinguished from discrete mathematics). Numerical analysis naturally finds application in all fields of engineering and the physical sciences, but in the 21st century also the life sciences, social sciences, medicine, business and even the arts have adopted elements of scientific computations. Although MATLAB is intended primarily for numerical computing, an optional toolbox uses the MiPads symbolic engine, allowing access to symbolic computing abilities. An additional package, Simulink, adds graphical multi-domain simulation and model-based design for dynamic and embedded systems. The MATLAB application is built around the MATLAB scripting language. Common usage of the MATLAB application involves using the Command Window as an interactive mathematical shell or executing text files containing MATLAB code.

**CNN (Convolution Neural Network)**

A convolutional neural network consists of an input and an output layer, as well as multiple hidden layers. The hidden layers of a CNN typically consist of convolutional layers, RELU layer i.e. activation function, pooling layers, fully connected layers and normalization layers. Description of the process as a convolution in neural networks is by convention. Mathematically it is a cross-correlation rather than a convolution (although cross-correlation is a related operation). This only has significance for the indices in the matrix, and thus which weights are placed at which index.

When programming a convolutional layer, each convolutional layer within a neural network should have the following attributes:

Input is a tensor with shape (number of images) x (image width) x (image height) x (image depth).Number of convolutional kernels. Width and height of kernels are hyper-parameters. Depth of kernels must be equal to the image depth. Convolutional layers apply a convolution operation to the input, passing the result to the next layer. The convolution emulates the response of an individual neuron to visual stimuli.

Each convolutional neuron processes data only for its receptive field. Although fully connected feed forward neural networks can be used to learn features as well as classify data, it is not practical to apply this architecture to images. A very high number of neurons would be necessary, even in a shallow (opposite of deep) architecture, due to the very large input sizes associated with images, where each pixel is a relevant variable. For instance, a fully connected layer for a (small) image of size 100 x 100 has 10,000 weights for each neuron in the second layer. The convolution operation brings a solution to this problem as it reduces the number of free parameters, allowing the network to be deeper with fewer parameters. For instance, regardless of image size, tiling regions of size 5 x 5, each with the same shared weights, requires only 25 learnable parameters. In this way, it resolves the vanishing or exploding gradients problem in training traditional multi-layer neural networks with many layers by using back propagation.

Object tracking has been one of the most important and active research areas in the field of computer vision. A large number of tracking algorithms have been proposed in recent years with demonstrated success. However, the set of sequences used for evaluation is often not sufficient or is sometimes biased for certain types of algorithms. Many datasets do not have common ground-truth object positions or extents, and this makes comparisons among the reported quantitative results difficult. In addition, the initial conditions or parameters of the evaluated tracking algorithms are not the same, and thus, the quantitative results reported in literature are incomparable or sometimes contradictory. To address these issues, we carry out an extensive evaluation of the state-of-the-art online object-tracking algorithms with various evaluation criteria to understand how these methods perform within the same framework. In this work, we first construct a large dataset with ground-truth object positions and extents for tracking and introduce the sequence attributes for the performance analysis. Second, we integrate most of the publicly available trackers into one code library with uniform input and output formats to facilitate large-scale performance evaluation. Third, we extensively evaluate the performance of 31 algorithms on 100 sequences with different initialization settings. By analyzing the quantitative results, we identify effective approaches for robust tracking and provide potential future research directions in this field.

**CHAPTER 5**

**TESTING**

Framework testing is really a progression of various tests whose basic role is to completely practice the PC based framework. Albeit every test has an alternate reason, all work to confirm that all the framework components have been appropriately incorporated and perform dispensed capacities .The testing procedure is really completed to ensure that the item precisely does likewise what should do. Testing is the last check and acceptance action inside the association itself.

In the testing stage taking after objectives are attempted to accomplish:-

* To attest the nature of the undertaking.
* To find and wipe out any leftover blunders from past stages.
* To accept the product as an answer for the first issue.
* To give operational dependability of the framework.

**7.1 Unit Testing**

Here every module that contains the general framework is tried exclusively. Unit testing centers confirmation endeavors even in the littlest unit of programming configuration in every module. This is otherwise called "Module Testing". The accompanying unit testing table demonstrates the capacities that were tried at the season of programming. The primary section records every one of the capacities which were tried and the second segment gives the portrayal of the tests done.

**7.2 Integration**

After successful completion of unit testing or module testing, singular capacities are incorporated into classes. Again joining of various classes assumes into position lastly mix of front-end with back-end happens.

**Integration of capacities into classes**

Toward the begin of coding stage just the capacities required in various parts of the system are created. Each of the capacities is coded and tried autonomously. After check of rightness of the distinctive capacities, they are coordinated into their separate classes.

**Integration of various classes**

Here the distinctive classes are tried autonomously for their usefulness. After confirmation of accuracy of yields in the wake of testing every class, they are incorporated together and tried once more.

**Integration of front-end with back-end**

The front-end of the undertaking is created in MATLAB environment. The client interface is intended to encourage the client to enter different summons to the framework and perspective the framework's typical and flawed conduct and its yields. The back-end code is then coordinated with the GUI and tried.

**7.3 Integration Testing**

Information can be lost crosswise over interface. One module can adverse affect another. Sub capacities when joined, ought not lessen the sought real capacity. Mix testing is a precise procedure for developing the system structure. It addresses the issues connected with the double issues of confirmation and project development.

**7.3.1 Top down Integration**

This method is an incremental approach to the construction of program structure. Modules are integrated by moving downward, beginning with the main program module. Modules that subordinates to the main program module are incorporated into the structure in either a depth first or breadth first manner.

**7.3.2. Bottom-up Integration**

This method begins the construction and testing with the modules at the lowest level in the program structure. Since the modules are integrated from bottom to up, processing required for modules subordinate to a given level is always available. Therefore in this case the need for stubs is eliminated.

The following integration testing table shows the functions that were combined into different classes and the class as a whole tested for its functionality.

|  |  |  |  |
| --- | --- | --- | --- |
| **Classes integrated** | **Functions integrated in each class** | **Tests done** | **Remarks** |
| Class: ObjectTracking | loadVideo  setActionToTrack  trackObject  init | Class tested to check whether all internal classes are called | Success |
| Class: CNNModel | Train  Predict | Class tested to check whether CNN model is trained and able to predict | Success |
| Class: DecisonTree | trainID3tree | Class tested to check whether ID3 decision tree is trained | Success |
| Class: Objectbounding | drawBoundBox  generateOutputVideo | Class tested to check whether object tracked in show in box and a output video is generated for tracked object | Success |

**Table 7.2:- Integration testing table**

**7.4 Validation Testing**

At the culmination of integration testing, software is completed and assembled as a package. Interfacing errors are uncovered and corrected. Validation testing can be defined in many ways. Here the testing validates the software function in a manner that is reasonably expected by the customer.

|  |  |  |  |
| --- | --- | --- | --- |
| **Functionality to be tested** | **Input** | **Tests done** | **Remarks** |
| Working of Front-End | User interaction with help of a mouse and keyboard | Appropriate forms open when buttons are clicked | Success |
| Working of object selection | User browse and choose the video | First frame is generated and displayed with marker | Success |
| Working of tracking | User selects the object to track in video and press track | Frames are displayed with object to track and output video generated | Success |

**Table 7.3:- Validation testing table**

**7.5 Output Testing**

After performing the validation testing, the next step is output testing of the proposed system, since no system could be useful if it does not produce the required output in the specified format. Therefore the output testing involves first of all asking the users about the format required by them and then to test the output generated or displayed by the system under consideration.

The output format is considered in 2 ways: –

1. On screen
2. Printed format

Printed format is used for validating the sites crawled , the relevance score of sites, the ranking score of sites.

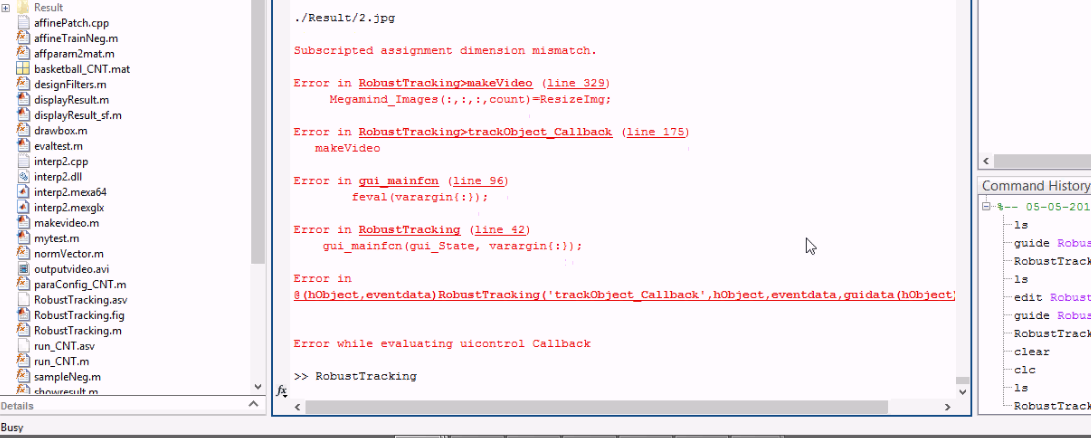
**CHAPTER 6**

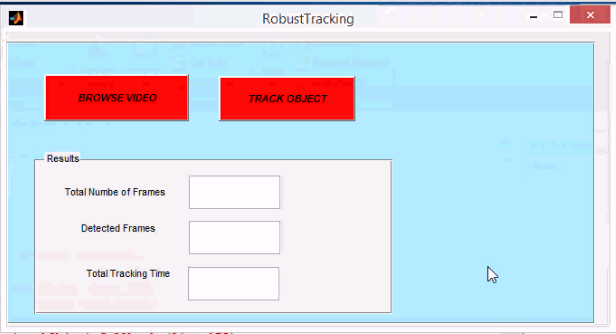
**RESULTS AND ANALYSIS**

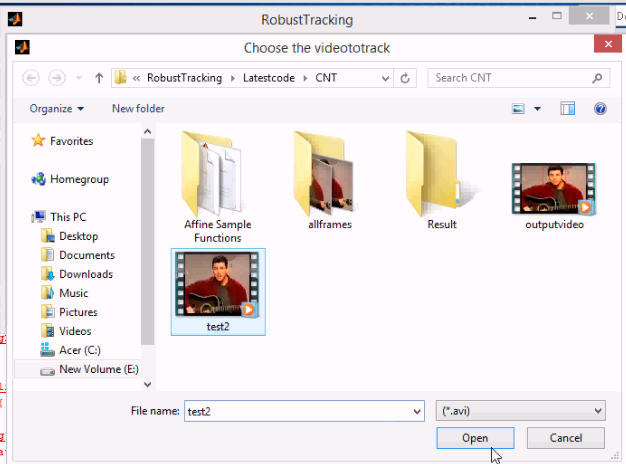
The following snapshots define the results or outputs that we will get after step by step execution of all the modules of the system.

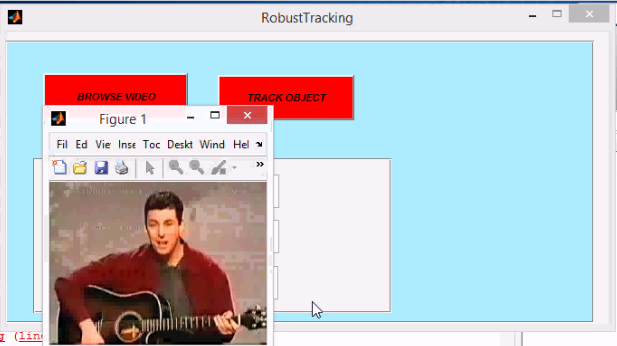
**Interpretation:**

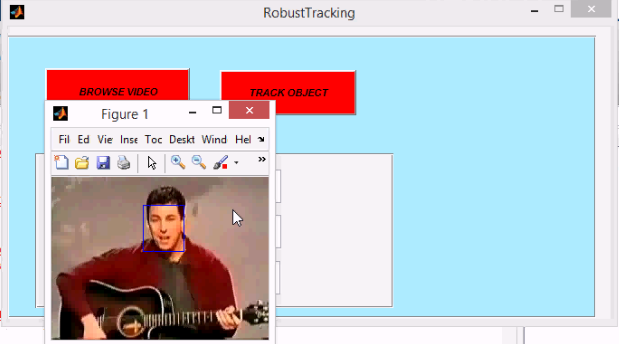
Execute the program in Matlab command line

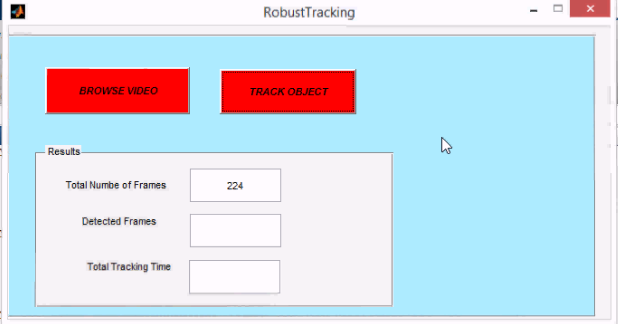




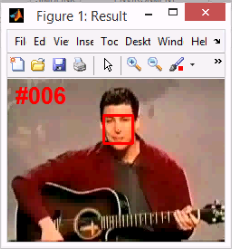


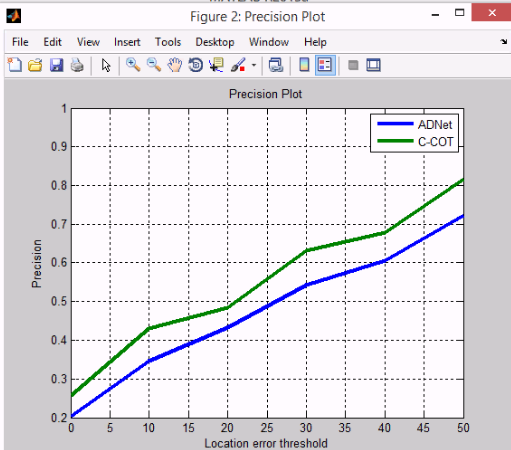


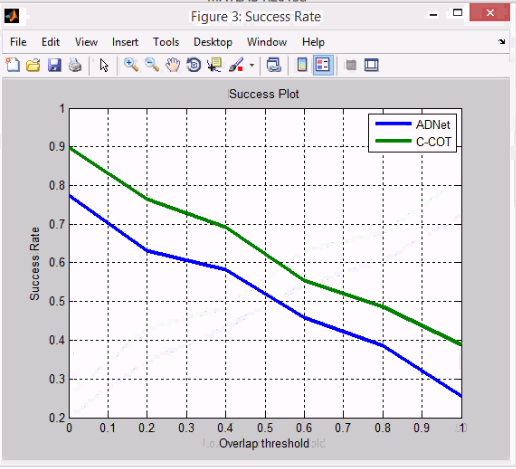


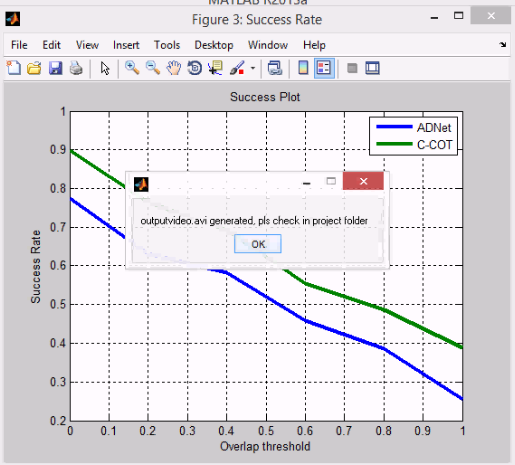


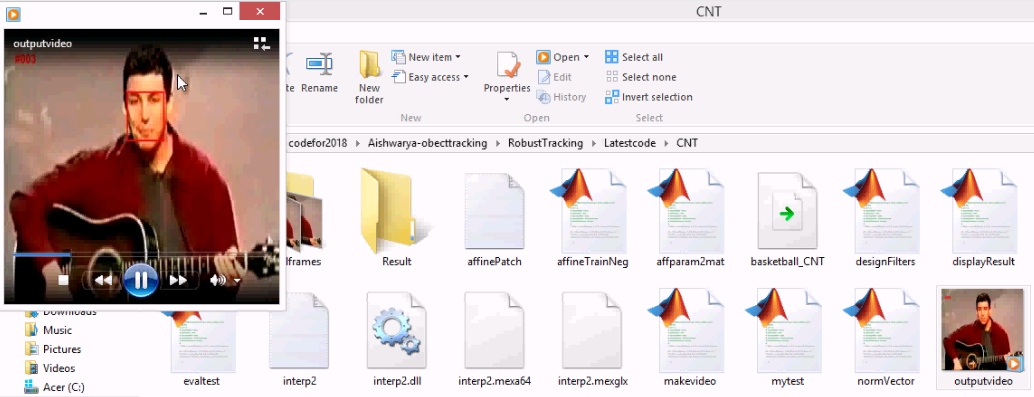












**CONCLUSION**

Author has developed a CNN based object action tracking system which can track the selected object in video for any of actions. The system is able to track with more than 98% accuracy. The proposed tracker is controlled by an ADNet, which pursues the target object by sequential actions iteratively. The action-driven tracking strategy makes a signiﬁcant contribution to the reduction of computation complexity in tracking. In addition, RL makes it possible to use partially labeled data, which could greatly contribute to the building of training data with a little effort

**REFERENCES**

[1] H. Grabner, M. Grabner, and H. Bischof, “Real-time tracking via on-line boosting,” in Proc. Brit. Mach. Vis. Conf., 2006, vol. 1. no. 5, p. 6.

[2] H. Grabner, C. Leistner, and H. Bischof, “Semi-supervised on-line boosting for robust tracking,” in Proc. Eur. Conf. Comput. Vis., 2008, pp. 234–247.

[3] B. Babenko, M.-H. Yang, and S. Belongie, “Robust object tracking with online multiple instance learning,” IEEE Trans. Pattern Anal. Mach.

Intell., vol. 33, no. 8, pp. 1619–1632, Aug. 2011.

[4] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, “Visual object tracking using adaptive correlation filters,” in Proc. IEEE Conf.

Comput. Vis. Pattern Recognit. (CVPR), Jun. 2010, pp. 2544–2550.

[5] M. Danelljan, G. Häger, F. Khan, and M. Felsberg, “Accurate scale estimation for robust visual tracking,” in Proc. Brit. Mach. Vis. Conf.,

Nottingham, U.K., Sep. 2014.