**SPORTS BALL PROJECTION AND ITS PREDICTION**

*submitted by*

SHUVANKAR NASKAR

**21MA60R35**

*Master of Technology*

COMPTUTER SCIENCE AND DATA PROCESSING



DEPARTMENT OF MATHEMATICS

Indian Institute of Technology Kharagpur (IIT-KGP)

West Bengal 721302, INDIA

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to **Dr. P V S N Murthy**, Seminar coordinator for his constant support, invaluable guidance and encouragement during the entire period of this work.

I am also grateful for the support of **Dr. Somnath Bhattacharyya**, Head of the Department, Department of Mathematics, Indian Institute of Technology Kharagpur.

I would like to express my thanks to **Dr. Sourav Mukhopadhyay**, Faculty Advisor, Department of Mathematics, Indian Institute of Technology Kharagpur, for extending his support in the completion of this work.

SHUVANKAR NASKAR

12 February 2022

**TABLE OF CONTENTS**

ABSTRACT 4

1. Introductions 6

1.1) Projections in FOOTBALL 7

2.Implementation of projection in basketball using CV-ZONE in python. 8

2.1) Details of regression analysis 8

2.1.a) Implementation results and analysis 9

2.2) Polynomial Regression Models 10

2.3) Considerations in fitting polynomial in one variable 10

2.3.a) Order of the model. 10

2.3.b) Model building strategy. 11

2.3.c) Extrapolation of curve. 11

3. Libraries are used in my code. 12

3.1) Open cv 12

**3.2) Applications of OpenCV. 13**

**3.3) OpenCV Functionality**. 13

3.4) NUMPY 14

3.5) Python Math Module: 14

4.1 Explanations of implemented code in python. 15

CONCLUSION 19

REFERENCES 20

**ABSTRACT**

Modern day sport needs advancement in a review system, so that player always feel comfortable for correct decision. Advancement in sport like cricket, football and basketball is to make ball path using past data (or followed path) such as ball movement angle, speed, hight, curve formation by using this parameter we can predict future path and how much distance it can cover.

In this seminar I’m presenting basket-ball projection when ball start moving toward basket, after 10 frame we get initial path & followed path of ball and by comparing those data algorithm will generate similar path and how it is extrapolated towards basket for measuring whether ball is passing through basket or not. For making this type of projection I used quadratic equation or we can say as polynomial equation, here in this project my result is perfectly follows actual path using python open-cv implementation. After analysing predicted path, we can say whether ball enters into basket or not at early stage. This will help us to analysing in review and for future training purpose to enhance system performance.

1. **Introductions**

Making Projection of ball is necessary in modern day sports, because it give lot of information’s during live to get fast correct decision and detailed analysis to improve performance of player in future. In this seminar I’m presenting basket-ball projection when ball start moving toward basket, after 10 frame we get initial path & followed path of ball and by comparing those data will algorithm generate similar path and how it is extrapolated towards basket for measuring whether ball is passing through basket or not. For making this type of projection I used polynomial equation, here in this project my result is perfectly follows actual path using python open-cv implementation. After analysing predicted path, we can say whether ball enters into basket or not at early stage.

Like if we more focus on cricket, during live streaming within a second system get calculate distance cover by ball for each boundary whether ball hits stadium rooftop or land into pavilions. speed of bowler and swinging angle of ball and most important to get an analysis for review asked by player in LBW (leg before wicket).

Now, I am showing some projections analysis in terms of its advantages.

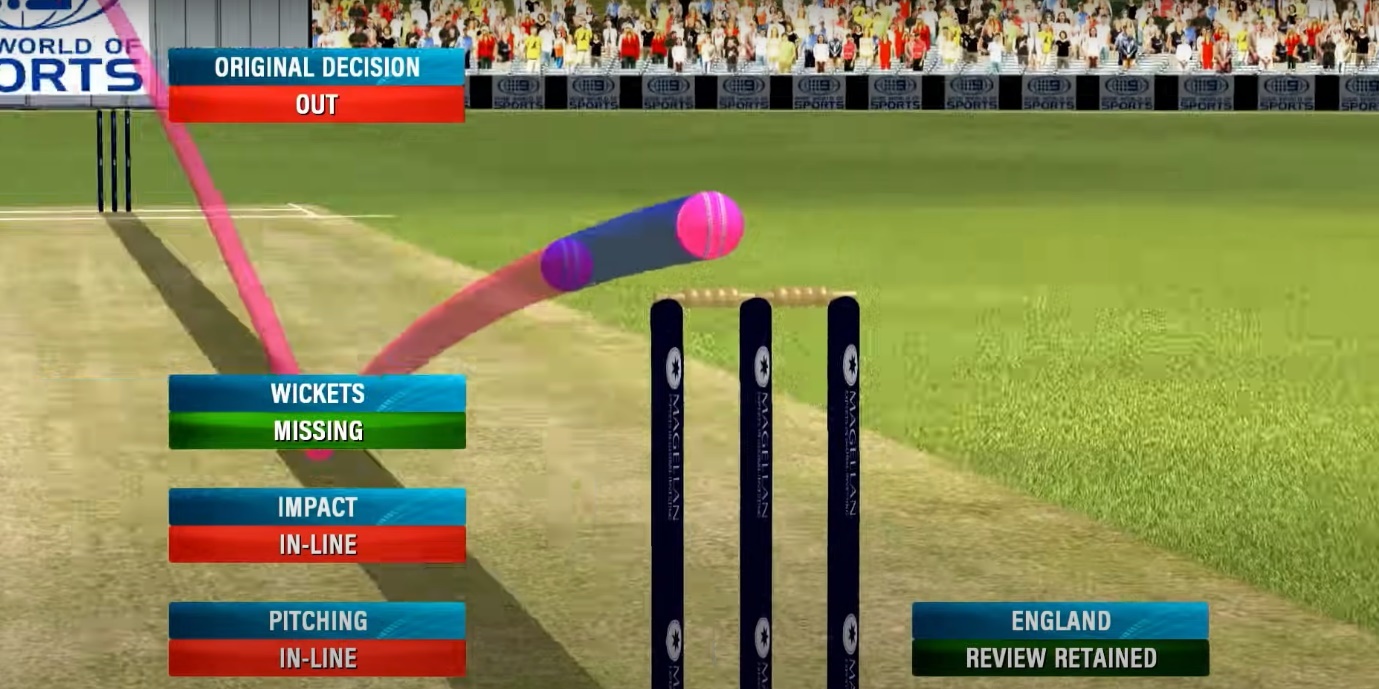


Fig 1.a(LBW review)

In fig 1.a the orange projection show physical movement of ball and blue line projection is predicted curve by using polynomial regression.  
here the original decision given by umpire was “out” but after extending ball projection we can see there is significant gap between stump and ball actual result is “not out” this is most widely used for getting correct decision.

The figure shown below is projection of boundaries in cricket. Here we are calculating a distance covered by ball in terms of plane horizontal surface trough equations (EQ1). And tracking a ball movement to project its path to get further analysis.

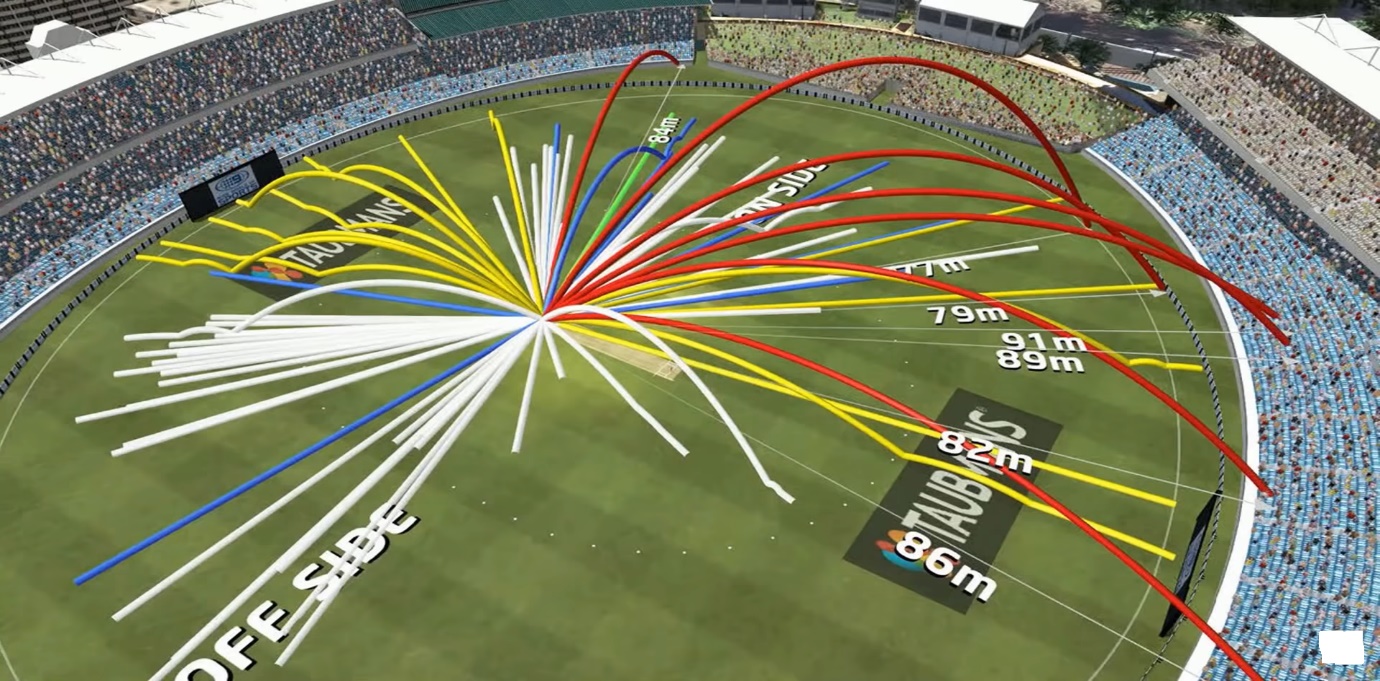


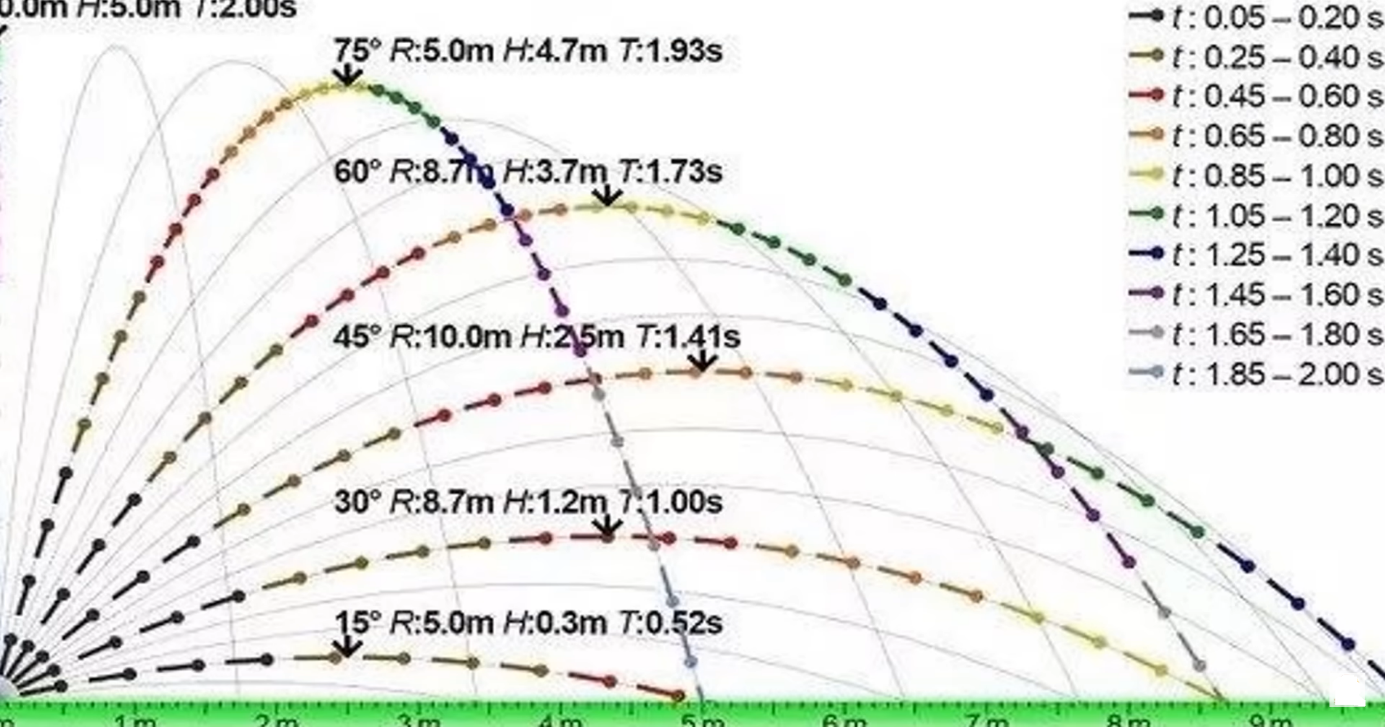
Fig 1.b

Distance = here, V = initial velocity of ball

g = gravitation constant (6.674 × 10-11 m3 kg-1 s-2)

when we analysis these equations it gives maximum distance covered by ball when initial angle is .

following below statistical graph shows different angle and distance covered by ball for same torque.



**1.1) Projections in FOOTBALL:**

In this figure 1.c based on simple analysis if we hit ball from RED arraow indicated then ball will follows red path and it lands into ground.

But Here, ball is not only follows a particuler path but also it has it’s rotations with respect to their axis. And ball is hollow inside with signicant physical dimensions so it add’s new parametes into equations which changes the directions of path which is known as swing in air.

Air swinging projections shown in figure by yellow path and it passed through goalcoast.

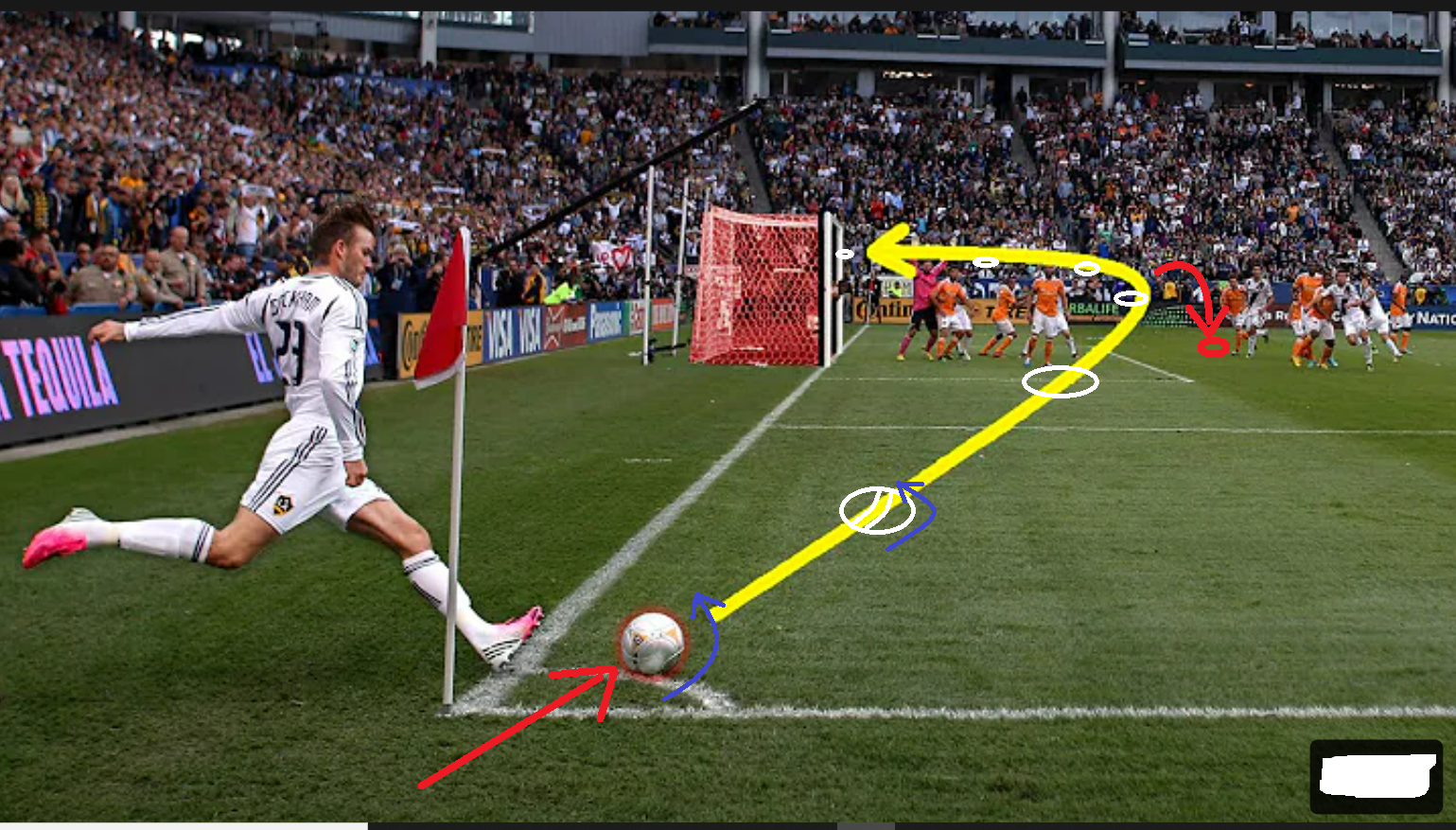
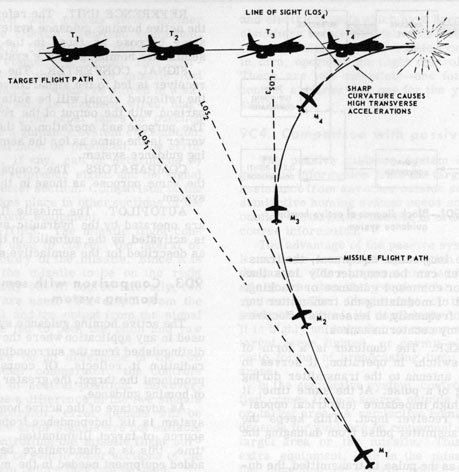


Fig 1.c



Here is another image is shown left side in fig 1.d of projection in guided missile system which follows pre-defined path to hit enemy object.

Fig 1.d

1. **Implementation of projection in basketball using CV-ZONE in python.**

here ball thrown by player to pass into basket shown in (figure 2.a). we have to guess what is possible result in term of basket or not basket.



Fig 2.a

To guess the result, we have plot graph using past projections and make a equation of projection for prediction of future projection using polynomial regression.

**2.1) Details of regression analysis**:

To get result whether ball will enter into basket or not, can be calculated using polynomial regression. As graph plotted in below figure 2.b. this projected graph must be pass through inside of basket ring which is explained at code implemented page.

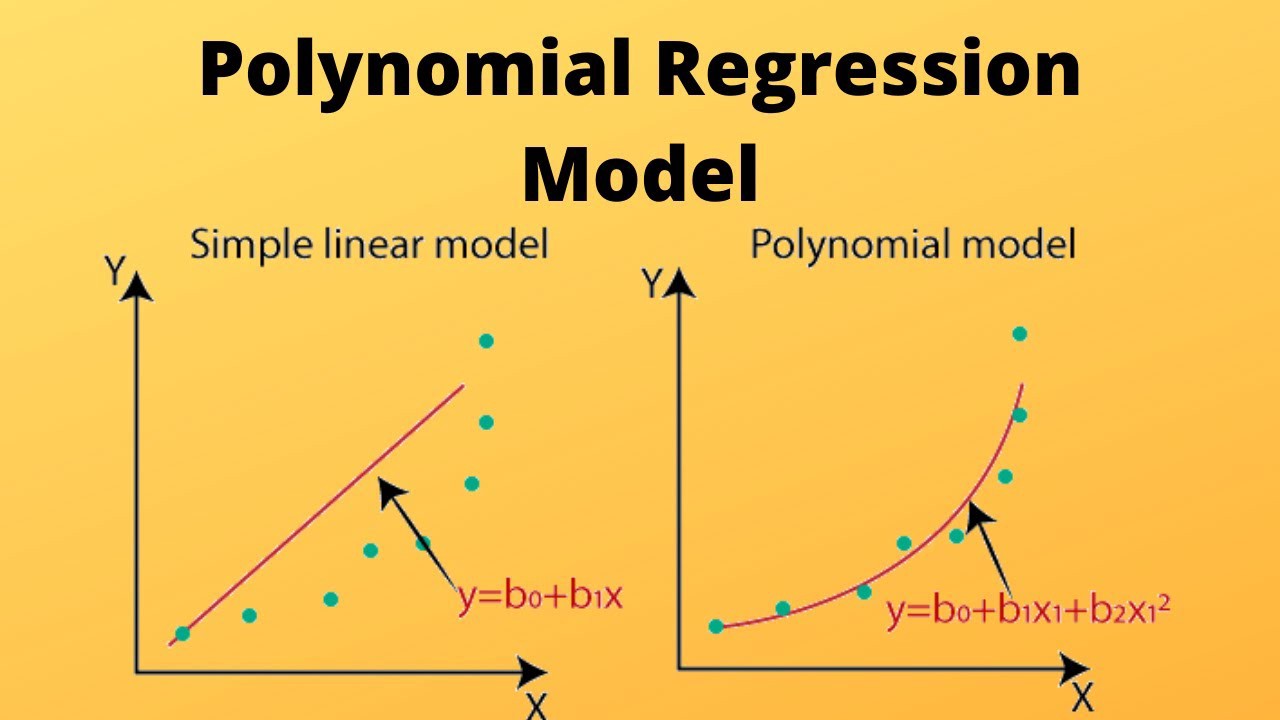


Fig2.b

By considering the 2nd equation to plot best fit curve and to get projection of ball movement as shown in fig (2.c and 2.d) using python open computer vision (CV), numpy library and mathematical model.

**2.1.a) Implementation results and analysis:**

In fig 2.c left side shows ball movement captured after 10 frames, which indicated in green dots and using those dots we have generated purple colour projection to predict final result, before ball reaches to basket.

As we can see purple projection shows path is not inside of basket, so the prediction is NO BASKET, now after continuing this video ball is hitting to outer ring of basket and jumps other than basket area so, the final result is same as predicted and projection is also follows very much similar to actual path. And result is NO BASKET.



Fig 2.c

Now Considering 2nd case in fig 2.d left side shows ball movement after 10 frames in green doted path by using this data, equation generate purple colour path which show ball will certainly hits basket. After continuing this frame(video) the ball is enters into basket and final result is also same as predicted BASKET as earlier.

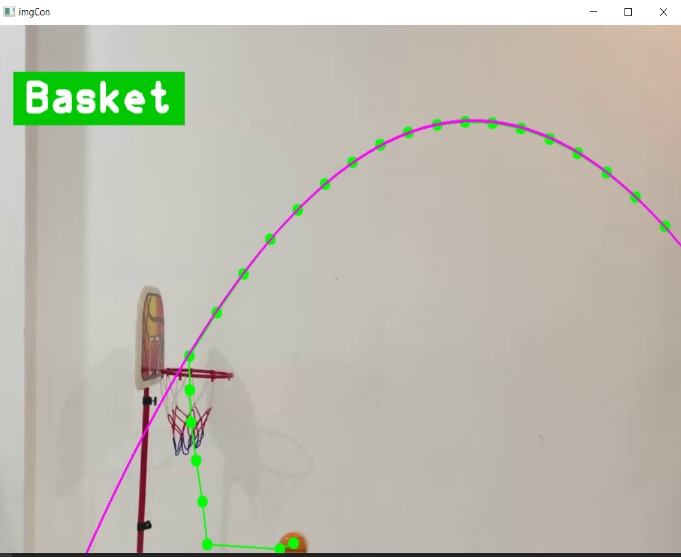


Fig 2.d

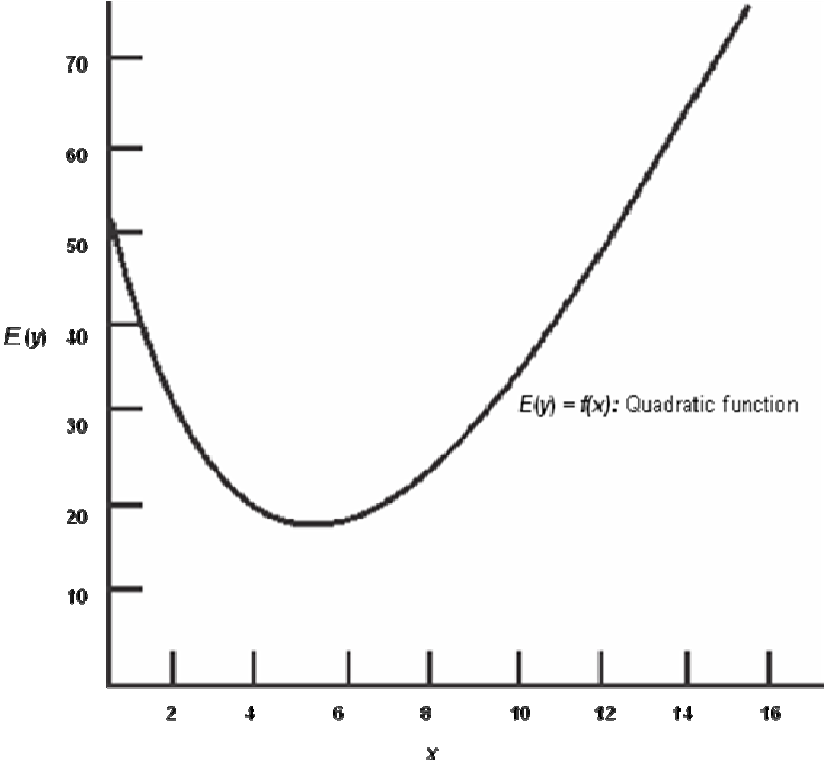
**2.2) Polynomial Regression Models: -**

The polynomial models can be used in those situations where the relationship between study and explanatory variables is curvilinear. Sometimes a nonlinear relationship in a small range of explanatory variable can also be modelled by polynomials.

|  |
| --- |
| For example: - Y = a + bX + cX^2 |

It is a polynomial regression model in one variable and is also called a **second-order model** or **quadratic model.** The coefficients b and c are called the **linear effect parameter** and **quadratic effect parameter,** respectively

The interpretation of parameter a is initial value of E(y), when X = 0 and it can be included in the model provided the range of data includes X=0. If X=0 is not included, then a has no Interpretation. An example of the quadratic model is like as shown below.



The polynomial models can be used to approximate a complex nonlinear relationship. The polynomial models are just the Taylor series expansion of the unknown nonlinear function in such a case.

**2.3) Considerations in fitting polynomial in one variable**

Some of the considerations in the fitting polynomial model are as follows:

**2.3.a) Order of the model.**

The order of the polynomial model is kept as low as possible. Some transformations can be used to keep the model to be of the first order equation. If this is not satisfactory, then the second-order polynomial is modelled. The arbitrary fitting of higher-order polynomials can be a serious abuse of regression analysis.

A model which is consistent with the knowledge of given data and its environment should be taken into account. It is always possible for a polynomial of order (*n* -1) to pass through *n* points so that a polynomial of sufficiently high degree can always be found that provides a “good” fit to the data or path. Such models neither enhance the understanding of the unknown function nor be a good predictor.

**2.3.b) Model building strategy: -**

A good strategy should be used to choose for the order of an approximate polynomial.

Here, we have One possible approach is to successively fit the models in sorted order and test the significance of regression coefficients at each step of model fitting. Keep the order increasing until *t* -test for the highest order term is nonsignificant. it is called a **forward selection procedure.**

Another approach is to fit the appropriate highest order model and then delete the terms one at a time starting with the highest order. This is continued until the highest order remaining term has a significant *t* - statistic. This is called a **backward elimination** procedure.

The forward selection and backward elimination procedures do not necessarily lead to the same model. Generally, the first and second-order polynomials are mostly used in practice.

**2.3.c) Extrapolation of curve:**

We have to be very cautioned in extrapolation with polynomial models. The curvatures in the region of data and the region of extrapolation can be significant different. For example, in the following figure 2.3.a, the trend of data in the region of original data is increasing, but it is decreasing as rectangular parabolic manner in the region of extrapolation. So predicted response would not be based on the true behaviour of the initial data.

In general, polynomial models may have unanticipated turns in inappropriate directions. This may provide incorrect inferences in interpolation as well as extrapolation.



Fig 2.3.a

A basic assumption in linear regression analysis is that rank of *X* - matrix is full column rank. In polynomial regression models, as the order increases, the *X* ' *X* matrix becomes ill conditioned. As a result, the (X' *X*)-1 may not be accurate, and parameters will be estimated with a considerable error.

If values of *x* lie in a narrow range, then the degree of ill-conditioning increases and multicollinearity in the columns of *X* matrix enters. For example, if *x* varies between 2 and 3, then *x*2 varies between 4 and 9. This introduces strong multicollinearity between *x* and *x*2.

**3. Libraries are used in my code: -**

**3.1) Open cv**

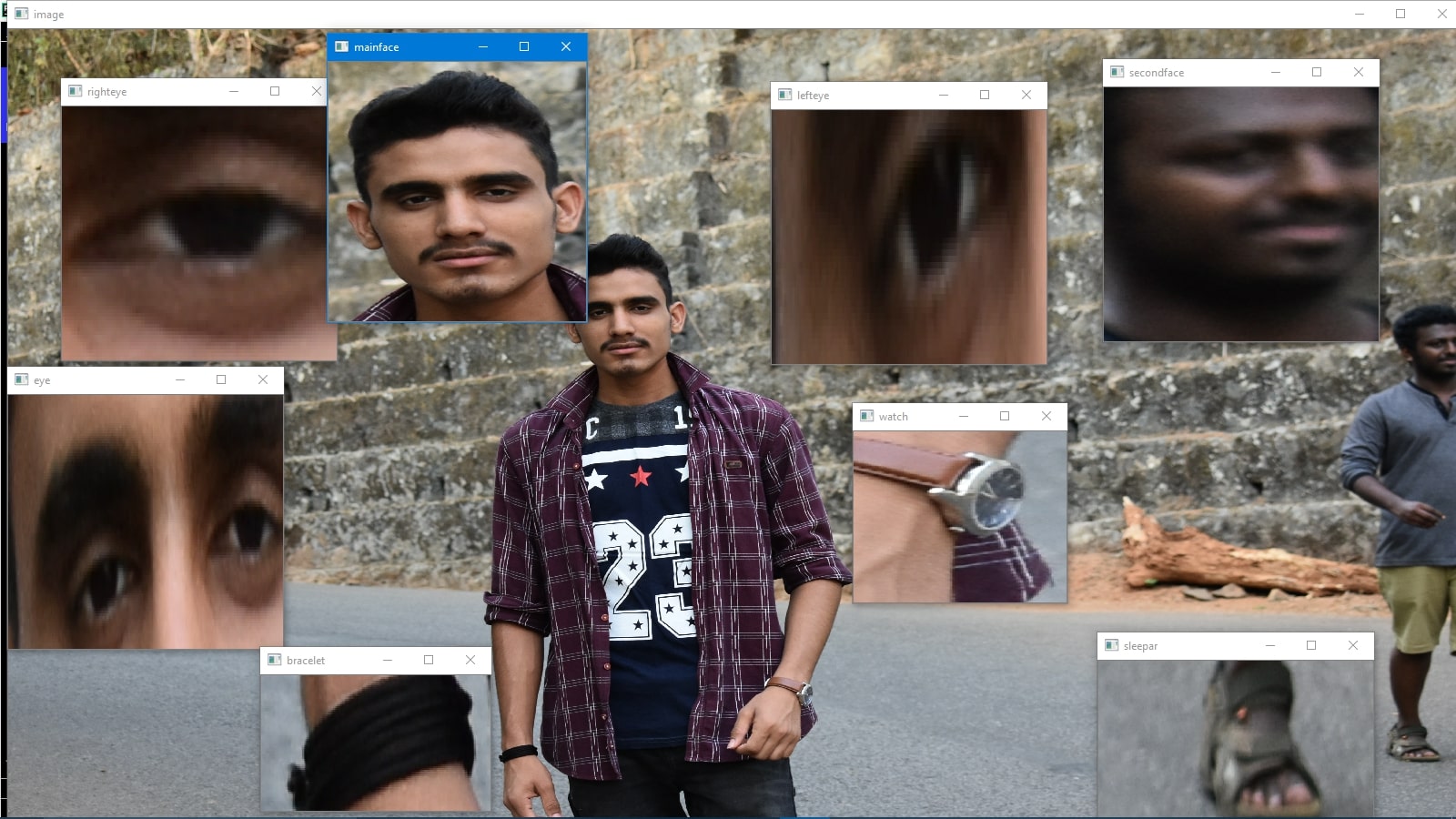
**Computer vision (CV)** is a process or tool by which we can understand the images and its sequence how they are stored and how we can modify to retrieve data from them.

Computer-Vision is playing a major role in self-driving cars, robotics as well as in photo correction apps and mostly Computer Vision is used for Artificial Intelligence.

**OpenCV** is the huge open-source library for the computer vision, machine learning, and image processing and in modern days it plays a major role in real-time operation which is very important in today’s daily life and for batter society.

By using open-cv, one can process images and videos to identify objects, faces, or even handwriting of a human. When it integrated with various libraries, such as NumPy, python is capable of processing the OpenCV array structure for analysis. To Identify image pattern and its various features we use vector space and perform mathematical operations on these features.

The first OpenCV version was 1.0. OpenCV is released under a BSD license and hence it’s free for both **academic** and **commercial** use. It has C++, C, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android. When OpenCV was designed the main focus was real-time applications for computational efficiency. All things are written in optimized C/C++ to take advantage of multi-core processing.   
Look at the image shown below. 



from the above original (backside) image, a piece of information that are present in the original image can be obtained. Like in the above image there are two faces available and the middle person in the images wearing a bracelet, watch, cloth so by the help of OpenCV we can get all these types of information from the original image.

**3.2) Applications of OpenCV:**There are lots of applications which are solved using OpenCV, some of them are listed below: -

* face recognition
* Automated inspection and surveillance
* number of people – count (foot traffic in a mall, etc)
* Vehicle counting on highways along with their speeds
* Interactive art installations
* Ana moly (defect) detection in the manufacturing process (the odd defective products)
* Street view image stitching
* Video/image search and retrieval
* Robot and driver-less car navigation and control
* object recognition
* Medical image analysis
* Movies – 3D structure from motion
* TV Channels advertisement recognition

**3.3) OpenCV Functionality**: -

* Image/video I/O, processing, display (core, imgproc, highgui)
* Object/feature detection (objdetect, features2d, nonfree)
* Geometry-based monocular or stereo computer vision (calib3d, stitching, videostab)
* Computational photography (photo, video, superres)
* Machine learning & clustering (ml, flann)
* CUDA acceleration (gpu)

## **Image-Processing**

Image processing is a method to perform certain operations on an image, in order to get an enhanced image to extract some useful information from it.   
the basic definition of image processing is **“Image processing is the analysis and manipulation of a digitized image, especially in order to improve its quality”.** 

**Digital-Image:**  
An image may be defined as a two-dimensional function f(x, y), where x and y are spatial(plane) coordinates, and the amplitude of fat any pair of coordinates (x, y) is called the intensity or grey level of the image at that point.   
or we can say an image is a two-dimensional matrix (3-D in case of coloured images) which is defined by the mathematical function f(x, y) at any point is giving the pixel value at that point of an image, the pixel value describes how bright that pixel is, and what colour it should be.   
Image processing is basically signal processing in which input is an image and output is image or characteristics according to requirement associated with that image. 

**How Does A Computer Read An Image?**   
The computer reads any image as a range of values between 0 and 255. For any colour image, there are 3 primary channels -red, green and blue. And rest colours are derived from RGB combination.

* 1. **NUMPY**: - pip install numpy to import for making computations.

NumPy is the elementary package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including, logical mathematical, I/O, shape manipulation, sorting, selecting, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and many more.

At core of the NumPy package, is the ndarray object. This encapsulates n-dimensional arrays of homogeneous data types, with many operations being performed in compiled code for performance. There are several important differences between NumPy arrays and the standard Python sequences:

• NumPy arrays have a fixed size at creation, unlike Python lists (which can grow dynamically). Changing the size of a ndarray will create a new array and delete the original.

• The elements in a NumPy array are all required to be of the same data type, and thus will be the same size in memory. The exception: one can have arrays of (Python, including NumPy) objects, thereby allowing for arrays of different sized elements.

• NumPy arrays facilitate advanced mathematical and other types of operations on large numbers of data. Typically, such operations are executed more efficiently and with less code than is possible using Python’s built-in sequences.

• A growing plethora of scientific and mathematical Python-based packages are using NumPy arrays; though these typically support Python-sequence input, they convert such input to NumPy arrays prior to processing, and they often output NumPy arrays. In other words, in order to efficiently use much (perhaps even most) of today’s scientific/mathematical Python-based software, just knowing how to use Python’s built-in sequence types is insufficient - one also needs to know how to use NumPy arrays.

* 1. Python Math Module: -

Sometimes when we working with some kind of financial or scientific projects it becomes necessary to implement mathematical calculations in the project. Python provides the **math module**to deal with such calculations. Math module provides functions to deal with both basic operations such as addition(+), subtraction(-), multiplication(\*), division(/) and advance operations like trigonometric, logarithmic, exponential functions.

4.1 **Explanations of implemented code in python :**

|  |
| --- |
| import cv2  import cvzone  from cvzone.ColorModule import ColorFinder  import numpy as np  import math |

Import above mentioned library and install it.

cap = cv2.VideoCapture('Videos/vid (4).mp4') # read video from videos/vid path

myColorFinder = ColorFinder(False)

#perticuler object movement to get result and its matrix value

hsvVals = {'hmin': 8, 'smin': 124, 'vmin': 13, 'hmax': 24, 'smax': 255, 'vmax': 255}

posListX = []

posListY = []

listX = [item for item in range(0, 1300)]

start = True

prediction = False

while True:

if start:

if len(posListX) == 10: start = False

success, img = cap.read()

# img = cv2.imread('Ball.png') #image reading

img = img[0:900, :] #to crop flour portion

imgPrediction = img.copy()

imgResult = img.copy()

imgBall, mask = myColorFinder.update(img, hsvVals)

imgCon, contours = cvzone.findContours(img, mask, 200)

if contours:

posListX.append(contours[0]['center'][0])

posListY.append(contours[0]['center'][1])

if posListX:

if len(posListX) < 18:

coff = np.polyfit(posListX, posListY, 2)

for i, (posX, posY) in enumerate(zip(posListX, posListY)):

pos = (posX, posY)

cv2.circle(imgCon, pos, 10, (0, 255, 0), cv2.FILLED)

cv2.circle(imgResult, pos, 10, (0, 255, 0), cv2.FILLED)

if i == 0:

cv2.line(imgCon, pos, pos, (0, 255, 0), 2)

cv2.line(imgResult, pos, pos, (0, 255, 0), 2)

else:

cv2.line(imgCon, (posListX[i - 1], posListY[i - 1]), pos, (0, 255, 0), 2)

cv2.line(imgResult, (posListX[i - 1], posListY[i - 1]), pos, (0, 255, 0), 2)

for x in listX: # equations to make quadratic equations

y = int(coff[0] \* x \*\* 2 + coff[1] \* x + coff[2])

cv2.circle(imgPrediction, (x, y), 2, (255, 0, 255), cv2.FILLED)

cv2.circle(imgResult, (x, y), 2, (255, 0, 255), cv2.FILLED)

# Predict

if len(posListX) < 10:

# y = int(coff[0] \* x \*\* 2 + coff[1] \* x + coff[2])

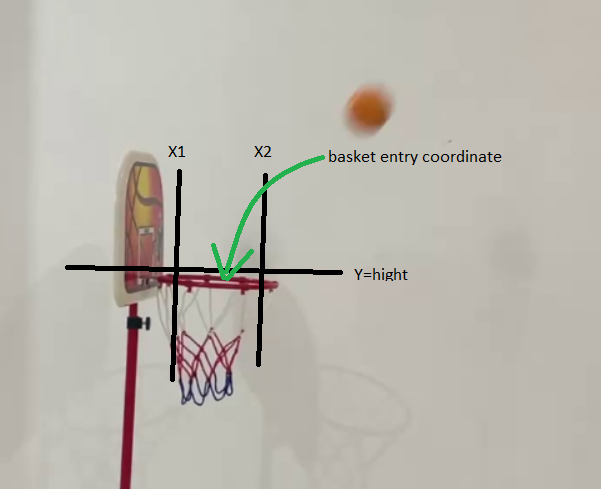
a, b, c = coff

c = c - 593

x = int((-b - math.sqrt(b \*\* 2 - (4 \* a \* c))) / (2 \* a))

prediction = 300 < x < 430 #basket outer ring size

X1=300 and X2=430



if prediction:

cvzone.putTextRect(imgResult, "yesss, Basket", (50, 150), colorR=(0, 200, 0),

scale=5, thickness=10, offset=20)

else:

cvzone.putTextRect(imgResult, "No Basket", (50, 150), colorR=(0, 0, 200),

scale=5, thickness=10, offset=20)

cv2.line(imgCon, (330, 593), (430, 593), (255, 0, 255), 10)

imgResult = cv2.resize(imgResult, (0, 0), None, 0.7, 0.7)

# imgStacked = cvzone.stackImages([img,imgCon,imgPrediction,imgResult],2,0.35)

cv2.imshow("imgCon", imgResult)

key = cv2.waitKey(100) #video streaming speed

if key == ord("s"): #press s to continue

start = True

**worst case time complexity: -**

**worst case space complexity: -**

**CONCLUSION**

In basket-ball shot prediction, implemented on 2-D coordinate picture. In that I calculated constant coefficient value and continuous changing value of X is traced and by using this parameter Y coordinate were found. Based on this data ball is continuously traced for 10 frames after that a predicted projection in drawn which show whether ball will hit basket or not, tracing of ball Is recognised using open-cv module and X & Y is calculated using polynomial equation which is available in math library, numpy library is used to make projection using Xlist and Ylist value.

In future or advancement of prediction on 3-D coordinate still very much difficult but by using some of the advance module we can generate that also.

**LIST OF REFERENCES**

* 1. Murtaza's Workshop - Robotics and AI
  2. Python library
  3. Geeks for geeks
  4. Chapter 12 Polynomial Regression Models,*Shalabh, IIT Kanpur*
  5. [youtube.com](https://www.youtube.com)/sportshala
  6. [analyticsvidhya.com](http://www.analyticsvidhya.com)
  7. google images