Introduction:

The problems posed are:

1. Carrying a hardware keyboard along with every device increases cost, space requirements and bulky.
2. Implementing new technology in education as per NEP 2020.
3. Modern laptops & tabs need to be made lighter, efficient and easy to use in every location.
4. Implementing AI in every domain to ease life.

Explanation:

1. We would need to carry a physical keyboard along with us when we travel –it costs us space in our luggage, increases weight of the device (a constraint especially in airports.) This problem requires tech enthusiasts to discover & implement AI & ML concepts in day-to-day life.
2. Using smart board to teach as per NEP requires smart keyboard to write on it as well. (Especially in coding classes). The virtual keyboard will reduce the space required there and max screen ratio can be used to display contents.
3. Modern laptops are being made lighter so to aid their ambition we can thing of applying AI to it and reduce the requirement of hardware keyboards. And just depend on virtual keyboard that doesn't eat up the screen space.
4. The need of the hour is to implement AI concepts that we learn in CSE to ease the human life and make our lives more comfortable.

Review of Literature:

Literature Survey on Vision Based Approach :

Vision based approach has the prospective to come up with natural and non-contact solutions, and is built on the way humans explicate and interpret information about their surroundings. It is in all probability the most tough approach to execute (H. Hasan & Abdul-Kareem, 2014).

A bare hand is used to extract data needed for recognition, and there is direct interaction between the user and with the system. For acquiring data needed for gesture analysis it uses some image characteristics like colour and texture.

Authors	Year	Description
P. Garg, N. Aggarwal, and S. Sofa(Garg, Aggarwal, & Sofat, 2009)	2009	This paper is a review about Vision based Hand Gesture Recognition techniques for human computer interaction, combining the various available approaches, listing out their general advantages and disadvantages
G. Murthy and R. Jadon,(Murthy & Jadon, 2009)	2009	The authors introduced the field of gesture recognition as a mechanism for interaction with computers.
M. K. Ahuja and A. Singh(Ahuja & Singh, 2015)	2015	The authors proposed a scheme using a database-driven hand gesture recognition based upon skin colour model approach and thresholding approach along with an effective template matching using PCA.

Camera Orientation and Distance :

It's necessary to be attentive about supervision of camera to allow easy alternative of background. Couple of good and more fruitful proposal is to direct camera towards ground or wall. The strength and power of light would be high and the effect of shadow will be low because camera was directed towards down.

The interspace of the camera from the hand should be such that it cover ups the whole motion mainly. No effect has been found on the accurateness of the structure if the picture is a focused one or not. Mainly the whole hand area should be covered.

Background selection :

Colour of background must be different from skin colour to maximize differentiation. The ground or floor colour used in the work was black. This colour was chosen because it showcased minimum amount of self-shadowing issue in comparison with other background colours.

Report on the present investigation:

Contours:

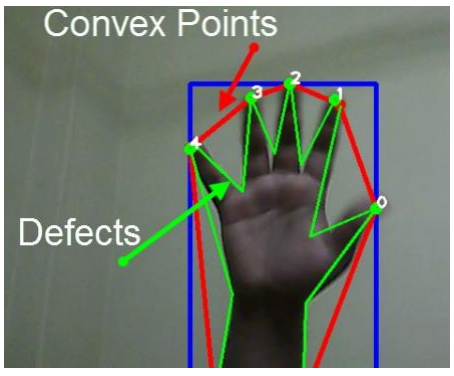
```
contours, hierarchy = cv.findContours(thresh,
cv.RETR_TREE,cv.CHAIN_APPROX_SIMPLE)
contours = max(contours, key=lambda x: cv.contourArea(x))
cv.drawContours(img, [contours], -1, (255,255,0), 2)
cv.imshow("contours", img)
```

Convex Hull:

```
hull = cv.convexHull(contours)
cv.drawContours(img, [hull], -1, (0, 255, 255), 2)
cv.imshow("hull", img)
```

Convexity Defects:

Any deviation of the object from this hull can be considered as convexity defect.




```
hull = cv.convexHull(contours, returnPoints=False)
defects = cv.convexityDefects(contours, hull)
```

Cosine Theorem:

In trigonometry, the law of cosines relates the lengths of the sides of a triangle to the cosine of one of its angles. Using notation as in Fig. 1, the law of cosines states where γ denotes the angle contained between sides of lengths a and b and opposite the side of length c.

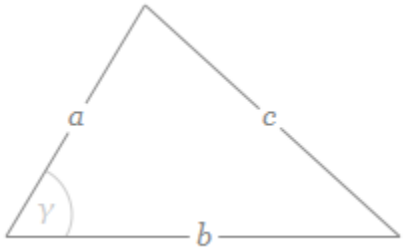


Fig. 1

Formula

$$c = \sqrt{a^2 + b^2 - 2ab\cos\gamma}$$

By seeing this formula now we understand that if we have; *a,b* and *gama* then we also find *c* as well as if we have; *a,b,c* then we also find *gamma (vice-versa)*

For finding gamma this formula is used:

$$\gamma = \cos^{-1}\left(\frac{a^2 + b^2 - c^2}{2ab}\right)$$

Using Cosine theorem to recognize fingers:

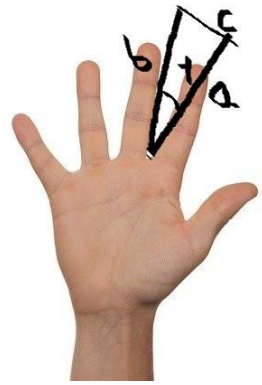


Fig. 2

In Fig. 2, I am draw a Side: *a,b,c* and angle: *gamma*. Now this gamma is always less than 90 degree, So we can say: If *gamma* is less than 90 degree or *pi/2* we consider it as a finger.

Counting Finger:

Convexity Defects returns an array where each row contains these values:

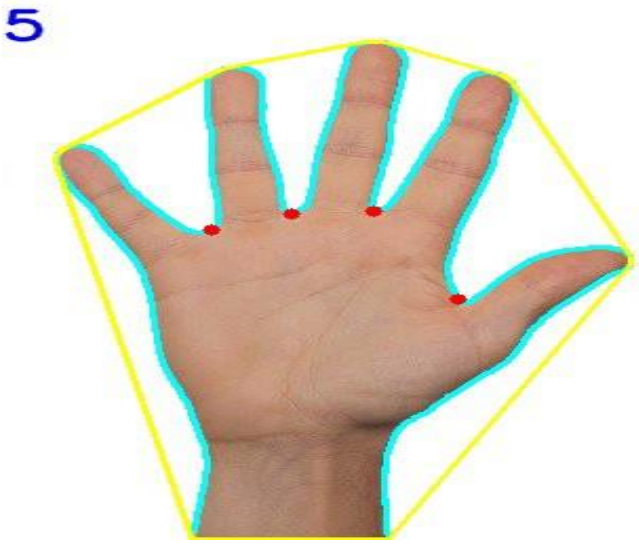
- start point
- end point
- farthest point
- approximate distance to farthest point

By, this point we can easily derive Sides: *a,b,c* (see CODE) and from cosine theorem we can also derive *gamma* or *angle between two finger.* As you read earlier, if gamma is less than 90 degrees, we treated it as a finger. After knowing gamma, we just draw circle with radius 4 in approximate distance to farthest point**.** And after we just simply put text in images, we represent finger counts (cnt).

```
if defects is not None:
 cnt = 0
for i in range(defects.shape[0]): # calculate the angle
 s, e, f, d = defects[i][0]
 start = tuple(contours[s][0])
 end = tuple(contours[e][0])
 far = tuple(contours[f][0])
 a = np.sqrt((end[0] - start[0]) ** 2 + (end[1] - start[1]) ** 2)
 b = np.sqrt((far[0] - start[0]) ** 2 + (far[1] - start[1]) ** 2)
 c = np.sqrt((end[0] - far[0]) ** 2 + (end[1] - far[1]) ** 2)
 angle = np.arccos((b ** 2 + c ** 2 - a ** 2) / (2 * b * c)) #      cosine theorem
 if angle <= np.pi / 2: # angle less than 90 degree, treat as fingers
   cnt += 1
   cv.circle(img, far, 4, [0, 0, 255], -1)
if cnt > 0:
 cnt = cnt+1
cv.putText(img, str(cnt), (0, 50), cv.FONT_HERSHEY_SIMPLEX,1, (255, 0, 0) , 2,
cv.LINE_AA)
```

Let's see our final result:

```
cv.imshow('final_result',img)
```



We can also do it for Videos, just by calling "cv.VideoCapture()".

Results and Discussions:

The result of the project is that we have had an immense brainstorming and knowledge sharing during the period of research and the implementation of the project. We are very enthusiastic that our project is an implantation of new trends of CSE into the core subjects like Mathematics (Linear Algebra).

Also, about the fact that this feature developed by us is capable of helping millions in the near future when the companies start manufacturing smart devices that don't require hardware keyboards but this smart, Sleek, Appealing Virtual Keyboard is going to be the next generation of Keyboards.

Summary/Conclusion and Future Scope:

For a prolonged hour, an issue of differentiating movement was vital in computer perception due to the opposition of removing the targeted object, like the hand from a framework which was making mess in actual time.

In actual fact, a person while gazing to a certain picture can effortlessly identify what is in it whereas, the same things is much tough for the computer if it looks at the same picture due to its functionality of dealing with a picture as a three-dimensional matrix. In future we would like to improve the accuracy further and add more gestures to implement more functions

Bibliography:

The sources of our project content and implementation are as follows:

Idea:

YouTube channel : Murtaza's Workshop - Robotics and AI

Link: https://youtu.be/jzXZVFqEE2I

Reference materials (websites for in-depth knowledge):

https://medium.com/analytics-vidhya/hand-detection-and-finger-counting-using-opencv-python-5b594704eb08

https://stackoverflow.com/questions/32694007/opencv-python-how-to-change-image-pixels-values-using-a-formula

https://docs.opencv.org/3.4/d8/dbc/tutorial_histogram_calculation.html

https://www.pythonpool.com/opencv-moments/

Research Papers:

1. Online Hand Gesture Recognition by Using OpenCV 1Manasa Srinivasa H S, 2Laxmi Tyapi,3 Suresha H S 1 PG Student, 2 PG Student, 3Associate Professor 1ECE Department, 1DBIT, Bangalore,India
2. Hand Gesture Recognition using OpenCV and Python Surya Narayan Sharma, Dr. A Rengarajan Department of Master of Computer Applications, Jain Deemed to be University, Bengaluru, Kamataka, India