Topic	Capstone class: Invisibility Cloak	
Class Description	**Students use the camera vision library to create an invisibility cloak.**	
Class	**C121**	
Class time	**45 mins**	
Goal	• Learn about the image processing techniques such as saturation and segmentation. • Write program to create an invisibility cloak	
Resources Required	• Teacher Resources ○ VS Code ○ Laptop with internet connectivity ○ Earphones with mic ○ Notebook and pen • Student Resources ○ VS Code ○ Laptop with internet connectivity ○ Earphones with mic ○ Notebook and pen	
Class structure	**Warm Up** **Teacher-led Activity** **Student-led Activity** **Wrap up**	**5 mins** **15 min** **15 min** **5 min**

CONTEXT

- **Introducing the concept of Image processing.**

Class Steps	Teacher Action	Student Action
Step 1: Warm Up (5 mins)	Hi <Student Name>! Welcome to the Capstone class.	**ESR:** - We explored the concept of naive bayes. - We learned about the

	How are you doing today? Can you quickly tell me what all we did in the previous class?	bayes theorem. -We wrote the Naive bayes algorithm . -We also compared it's output with the logistic regression.
	Perfect!! Do you like magic? Which magical power would you like to have?	**ESR:** Student talks about the magic he/she likes.
	Today we are going to do something magical and that is we are going to make an invisibility cloak which makes its wearer invisible. Are you excited for it?	**ESR:** Yes!!
	Let us get to the work and start creating an invisibility cloak.	
Teacher Initiates Screen Share		
CHALLENGE • **Use the different methods of CV2 library .** • **Use image processing to create a invisibility cloak**		
Step 2: Teacher-led Activity (15 min)	So you remember a few classes ago we used the camera of our system. Can you tell me what we did using the camera?	**ESR:** Using the camera we created a security system which took snaps after every 5 mins so that later on we can know who all used our system
	Yes. And can you tell me which library we used for that? Perfect. Today also we'll use the	**ESR:** We used the cv2 library to access the camera.

	same library to make an invisibility cloak. To create this magical experience using an image processing technique called Color detection and segmentation.	
	So our algorithm or the steps will be as follows:- 1. **Capture and store the background frame. [This will be done for some seconds]** 2. **Detect the red colored cloth using color detection and segmentation algorithm.** **3. Segment out the red colored cloth by generating a mask. [used in code]** **4. Generate the final augmented output to create a magical effect. [video.mp4]**	*The student observes and learns.*
	Let's start with our code then. *Teacher opens the VSCode editor and creates a new file named invisibilityCloak.py.*	-
	So we'll be using cv2, time and numpy libraries in our algorithm. So let's import them to our code. Note:- Make sure you have a cv2 library installed in your system. If not, install using the following code: - **sudo apt update** **- sudo apt install python3-opencv** # in order to check the cv2 version **- print(cv2.__version__)** *Teacher codes to import the libraries*	*The student observes and learns.*

	to the code. Code:- **import cv2** **import time** **import numpy as np**	
		
	Now let's do some preparation to save the video as the output. For that we'll use the **VideoWriter_fourcc**. FourCC is a 4-byte code used to specify the video codec. The list of available codes can be found in fourcc.org. It is platform dependent. **FourCC** code is passed as `cv.VideoWriter_fourcc('M','J','P','G')or cv.VideoWriter_fourcc(*'MJPG')` for MJPEG. Code:-	*The student observes and learns.*

	`` #To save the output in a file output.avi fourcc = cv2.VideoWriter_fourcc(*'XVID') output_file = cv2.VideoWriter('output.avi', fourcc, 20.0, (640, 480)) ```	

```
import cv2
import time
import numpy as np

#To save the output in a file output.avi
fourcc = cv2.VideoWriter_fourcc(*'XVID')
output_file = cv2.VideoWriter('output.avi', fourcc, 20.0, (640, 480))
```

	Now we'll start reading the video from the webcam of our system. We'll use **VideoCapture()** function to capture the video. Code:- ``` #Starting the webcam cap = cv2.VideoCapture(0) ``` And then we'll allow the camera to warm up. Code:- ``` #Allowing the webcam to start by making the code sleep for 2 seconds time.sleep(2) bg = 0 ```	*The student learns and asks questions.*

```
#To save the output in a file output.avi
fourcc = cv2.VideoWriter_fourcc(*'XVID')
output_file = cv2.VideoWriter('output.avi', fourcc, 20.0, (640, 480))

#Starting the webcam
cap = cv2.VideoCapture(0)

#Allowing the webcam to start by making the code sleep for 2 seconds
time.sleep(2)
bg = 0
```

	We need to have a video that has some seconds dedicated to the background frame so that it could easily save the background image. So we'll be capturing the background in the range of 60. Code:- `#Capturing background for 60 frames` `for i in range(60):` `  ret, bg = cap.read()` `#Flipping the background` `bg = np.flip(bg, axis=1)` Here we are flipping the background because the camera captures the image inverted.	*Student observes and asks questions.*

```
#Allowing the webcam to start by making the code sleep for 2 seconds
time.sleep(2)
bg = 0

#Capturing background for 60 frames
for i in range(60):
    ret, bg = cap.read()
#Flipping the background
bg = np.flip(bg, axis=1)
```

	Now that we have our background ready, we need to read every frame from the camera. Code: `#Reading the captured frame until the camera is open` `while (cap.isOpened()):` `  ret, img = cap.read()` `  if not ret:` `  break` `  #Flipping the image for consistency` `  img = np.flip(img, axis=1)` Here we are using **cap.isOpened** to check if the camera is opened or not. **ret** returns a boolean value of true or false. And using **np.flip** to flip the image.	*Student observes and asks questions.*

```
#Capturing background for 60 frames
for i in range(60):
    ret, bg = cap.read()
#Flipping the background
bg = np.flip(bg, axis=1)

#Reading the captured frame until the camera is open
while (cap.isOpened()):
    ret, img = cap.read()
    if not ret:
        break
    #Flipping the image for consistency
    img = np.flip(img, axis=1)
```

As we capture frames we are also capturing the colors in those frames. And we need to convert the images from BGR (Blue Green Red) to HSV (Hue, Saturation, Value).
We need to do this so that we can detect the red color more efficiently.
Code :-

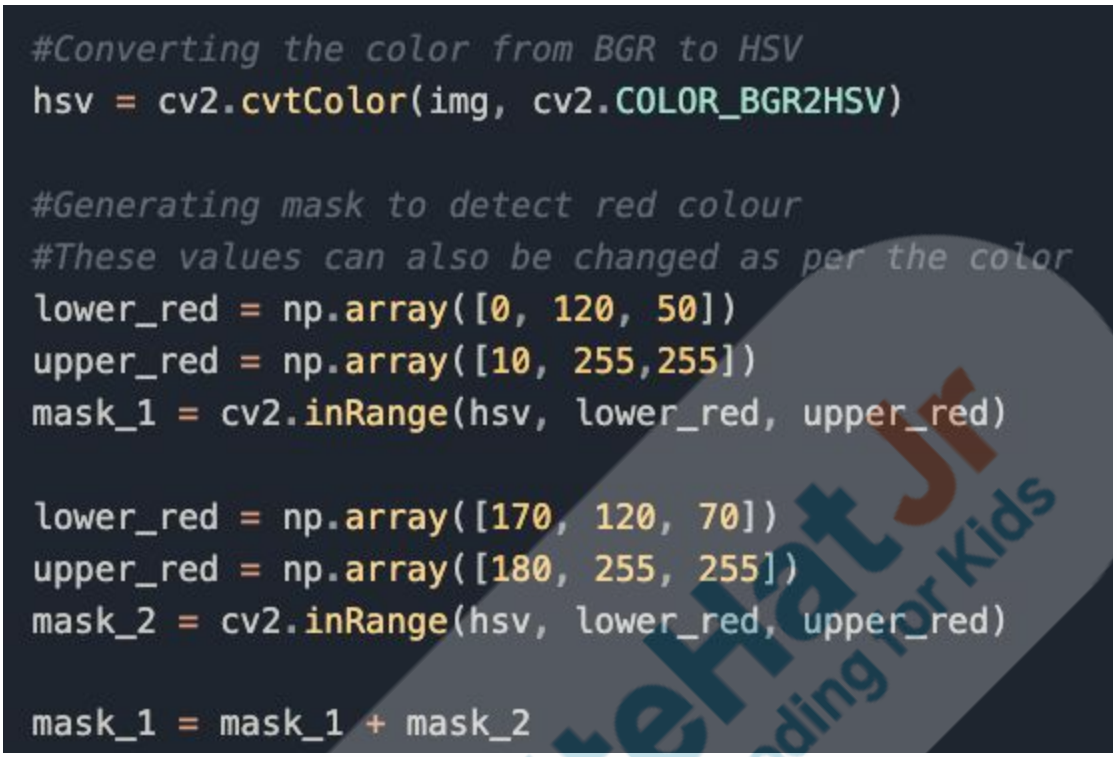
```
#Converting the color from BGR to HSV
hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
```

1. Hue: This channel encodes color information. Hue can be thought of as an angle where 0 degree corresponds to the red color, 120 degrees corresponds to the green color, and 240 degrees corresponds to the blue

Student observes and asks questions.

	color. 2. Saturation: This channel encodes the intensity/purity of color. For example, pink is less saturated than red. 3. Value: This channel encodes the brightness of color. Shading and gloss components of an image appear in this channel reading the videocapture video.	
 ``` #Reading the captured frame until the camera is open while (cap.isOpened()): ret, img = cap.read() if not ret: break #Flipping the image for consistency img = np.flip(img, axis=1) #Converting the color from BGR to HSV hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV) ```		
	Now that we have converted the color from BGR To HSV it will be easy for us to identify the colors. We have to create masks which will check for the colors in the specified range and then mask it with the background image. We'll be creating 2 different masks which will help us detect the colors in that given range.	*Student observes and asks questions.*

	We are creating masks to detect the red color. You can change the color value depending on the color you want. `#Generating mask to detect red colour` `#These values can also be changed as per the color` `lower_red = np.array([0, 120, 50])` `upper_red = np.array([10, 255,255])` `mask_1 = cv2.inRange(hsv, lower_red, upper_red)` `lower_red = np.array([170, 120, 70])` `upper_red = np.array([180, 255, 255])` `mask_2 = cv2.inRange(hsv, lower_red, upper_red)` `mask_1 = mask_1 + mask_2`	



```
#Converting the color from BGR to HSV
hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)

#Generating mask to detect red colour
#These values can also be changed as per the color
lower_red = np.array([0, 120, 50])
upper_red = np.array([10, 255,255])
mask_1 = cv2.inRange(hsv, lower_red, upper_red)

lower_red = np.array([170, 120, 70])
upper_red = np.array([180, 255, 255])
mask_2 = cv2.inRange(hsv, lower_red, upper_red)

mask_1 = mask_1 + mask_2
```

	Now we need to add effects on the colors that we have detected. What kind of effect do you think we'll be adding? We'll be adding the diluting effect to the image in the video. For that we'll be using the **morphologyEx()** method. Syntax of this method: morphologyEx(src, dst, op, kernel) This method accepts the following parameters: • src − An object representing the source (input) image. • dst − object representing the destination (output) image.	**ESR:** Varied.

- op − An integer representing the type of the Morphological operation.
- kernel − A kernel matrix.

morphologyEx() is the method of the class Img Processing which is used to perform operations on a given image.
Code:-

```
   #Open and expand the image where there is mask 1 (color)
   mask_1 = cv2.morphologyEx(mask_1, cv2.MORPH_OPEN, np.ones((3, 3), np.uint8))
   mask_1 = cv2.morphologyEx(mask_1, cv2.MORPH_DILATE, np.ones((3, 3), np.uint8))
```

MORPH_OPEN and MORPH_DILATE are the two types of effects.

```
#Generating mask to detect red colour
#These values can also be changed as per the color
lower_red = np.array([0, 120, 50])
upper_red = np.array([10, 255,255])
mask_1 = cv2.inRange(hsv, lower_red, upper_red)

lower_red = np.array([170, 120, 70])
upper_red = np.array([180, 255, 255])
mask_2 = cv2.inRange(hsv, lower_red, upper_red)

mask_1 = mask_1 + mask_2

#Open and expand the image where there is mask 1 (color)
mask_1 = cv2.morphologyEx(mask_1, cv2.MORPH_OPEN, np.ones((3, 3), np.uint8))
mask_1 = cv2.morphologyEx(mask_1, cv2.MORPH_DILATE, np.ones((3, 3), np.uint8))
```

<table>
<tr>
<td></td>
<td>Now we need to create a mask to segment out the red color from the frame.
To do so we'll be using the bitwise_not() method.
Code:-

<pre>
#Selecting only the part that does not have mask one and saving in mask 2
mask_2 = cv2.bitwise_not(mask_1)
</pre></td>
<td>Student observes and asks questions.</td>
</tr>
<tr>
<td colspan="3"><pre>
#Open and expand the image where there is mask 1 (color)
mask_1 = cv2.morphologyEx(mask_1, cv2.MORPH_OPEN, np.ones((3, 3), np.uint8))
mask_1 = cv2.morphologyEx(mask_1, cv2.MORPH_DILATE, np.ones((3, 3), np.uint8))

#Selecting only the part that does not have mask one and saving in mask 2
mask_2 = cv2.bitwise_not(mask_1)
</pre></td>
</tr>
<tr>
<td></td>
<td>Now, we need to create 2 resolutions. First one would be an image without color red (or any other color that you choose) and the second one would be the background from the background image we captured earlier just for the parts where red color was (mask 1).

Code:-

<pre>
#Keeping only the part of the images without the red color
#(or any other color you may choose)
res_1 = cv2.bitwise_and(img, img, mask=mask_2)
</pre></td>
<td>-</td>
</tr>
</table>

```
    #Keeping only the part of the
images with the red color
    #(or any other color you may
choose)
    res_2 = cv2.bitwise_and(bg, bg,
mask=mask_1)
```

```
#Keeping only the part of the images without the red color
#(or any other color you may choose)
res_1 = cv2.bitwise_and(img, img, mask=mask_2)

#Keeping only the part of the images with the red color
#(or any other color you may choose)
res_2 = cv2.bitwise_and(bg, bg, mask=mask_1)
```

All that is left is to generate the final video with the invisible effect and also, to display the result to the user.

-

```
    #Generating the final output by
merging res_1 and res_2
    final_output =
cv2.addWeighted(res_1, 1, res_2, 1,
0)
    output_file.write(final_output)
    #Displaying the output to the user
    cv2.imshow("magic", final_output)
    cv2.waitKey(1)
```

```
#Keeping only the part of the images with the red color
#(or any other color you may choose)
res_2 = cv2.bitwise_and(bg, bg, mask=mask_1)

#Generating the final output by merging res_1 and res_2
final_output = cv2.addWeighted(res_1, 1, res_2, 1, 0)
output_file.write(final_output)
#Displaying the output to the user
cv2.imshow("magic", final_output)
cv2.waitKey(1)
```

	And now we'll just close all the windows that have opened up. Code:- `cap.release()` `out.release()` `cv2.destroyAllWindows()` *Teacher runs the code.* *Waits for some seconds till the camera gets warmed up.* *Then brings an object of red color in front of the camera to see the magic.*	*The student observes and learns.*

```
    #Generating the final output by merging res_1 and res_2
    final_output = cv2.addWeighted(res_1, 1, res_2, 1, 0)
    output_file.write(final_output)
    #Displaying the output to the user
    cv2.imshow("magic", final_output)
    cv2.waitKey(1)


cap.release()
out.release()
cv2.destroyAllWindows()
```

	Alright then. Would you like to try this by yourself? Let's get started then.	**ESR:** Yes!
Teacher Stops Screen Share		
	Now it's your turn. Please share your screen with me.	

- **Ask Student to press ESC key to come back to panel**
- **Guide Student to start Screen Share**
- **Teacher gets into Fullscreen**

ACTIVITY

- **Use the different methods of CV2 library .**
- **Use image processing to create a invisibility cloak**

<table>
<tr><td>Step 3:
Student-Led
Activity
(15 min)</td><td>Teacher helps the student to install the Cv2 library on his/her system.

Using the commands
sudo apt update
-sudo apt install python3-opencv</td><td>Student opens the VSCode and creates a new file called invisibility cloak.py.
Student also installs the open -cv library to his/her system if it's not installed.</td></tr>
<tr><td></td><td>Teacher helps the student to import the libraries.</td><td>Student imports the numpy, time and cv library to the file.</td></tr>
<tr><td colspan="3"> 
</td></tr>
<tr><td></td><td>Teacher helps the student to code to start reading the video from the system and allow the camera to warm up for a while.</td><td>Student codes to start reading the video from the system and allow the camera to warm up for a while.</td></tr>
</table>

```
#To save the output in a file output.avi
fourcc = cv2.VideoWriter_fourcc(*'XVID')
output_file = cv2.VideoWriter('output.avi', fourcc, 20.0, (640, 480))

#Starting the webcam
cap = cv2.VideoCapture(0)

#Allowing the webcam to start by making the code sleep for 2 seconds
time.sleep(2)
bg = 0
```

	Teacher helps the student with the code.	*Student codes to capture the background for 60 frames.*

```
#Capturing background for 60 frames
for i in range(60):
    ret, bg = cap.read()
#Flipping the background
bg = np.flip(bg, axis=1)
```

	Teacher helps the student to code to read frames from the webcam, until the camera is open.	*Student codes read every frame from the webcam, until the camera is open.*

```
#Reading the captured frame until the camera is open
while (cap.isOpened()):
    ret, img = cap.read()
    if not ret:
        break
    #Flipping the image for consistency
    img = np.flip(img, axis=1)
```

	Teacher helps the student to convert the color space from BGR to HSV.	*Student codes to Convert the color space from BGR to HSV.*

```
#Converting the color from BGR to HSV
hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
```

	Teacher helps the student to generate the masks of red color.	*Student codes to generate the masks of red color.*

```
#Generating mask to detect red colour
#These values can also be changed as per the color
lower_red = np.array([0, 120, 50])
upper_red = np.array([10, 255,255])
mask_1 = cv2.inRange(hsv, lower_red, upper_red)

lower_red = np.array([170, 120, 70])
upper_red = np.array([180, 255, 255])
mask_2 = cv2.inRange(hsv, lower_red, upper_red)

mask_1 = mask_1 + mask_2
```

	Teacher helps the student to: Open and Dilate the mask image.	*Student codes to: Open and Dilate the mask image.*

```
#Open and expand the image where there is mask 1 (color)
mask_1 = cv2.morphologyEx(mask_1, cv2.MORPH_OPEN, np.ones((3, 3), np.uint8))
mask_1 = cv2.morphologyEx(mask_1, cv2.MORPH_DILATE, np.ones((3, 3), np.uint8))
```

	Teacher helps the student to: Create an inverted mask to segment out the red color from the frame.	*Student codes to: Create an inverted mask to segment out the red color from the frame.*

```
#Selecting only the part that does not have mask one and saving in mask 2
mask_2 = cv2.bitwise_not(mask_1)
```

	Teacher helps the student to: Segment the red color part out of the frame using bitwise and with the inverted mask.	*Student codes to: Segment the red color part out of the frame using bitwise and with the inverted mask.*
``#Keeping only the part of the images without the red color #(or any other color you may choose) res_1 = cv2.bitwise_and(img, img, mask=mask_2)```		
	Teacher helps the student to: Create image showing static background frame pixels only for the masked region.	*Student codes to: Create image showing static background frame pixels only for the masked region.*
``#Keeping only the part of the images with the red color #(or any other color you may choose) res_2 = cv2.bitwise_and(bg, bg, mask=mask_1)```		
	Teacher helps the student to code to generate the final output and show the output and then close all the windows.	*Student codes to generate the final output and show the output and then close all the windows.*

```
    #Keeping only the part of the images with the red color
    #(or any other color you may choose)
    res_2 = cv2.bitwise_and(bg, bg, mask=mask_1)

    #Generating the final output by merging res_1 and res_2
    final_output = cv2.addWeighted(res_1, 1, res_2, 1, 0)
    output_file.write(final_output)
    #Displaying the output to the user
    cv2.imshow("magic", final_output)
    cv2.waitKey(1)


cap.release()
out.release()
cv2.destroyAllWindows()
```

		Student runs the code and checks for output.

Teacher Guides Student to Stop Screen Share

FEEDBACK

- **Appreciate the student for their efforts**
- **Identify 2 strengths and 1 area of progress for the student**

Step 4: Wrap-Up (5 min)	So, today we created an invisibility cloak. How was your experience?	**ESR:** varied

	Amazing. So today we had some fun. So these are some cool things that we can do using tech. You can also show this to your friends and amaze them. We'll be exploring more such things in our upcoming classes. Are you excited for that?	**ESR:** Yes
	See you in the next class then.	
	Congratulations! You have accomplished a milestone. This Capstone project will be an extension of our current class. You have to use your understanding and add value to the Invisibility cloak.	
Project Overview	**Background Matters** **Goal of the Project:** In this project, you will have to practice and apply what you have learned in the class. You will be modifying the background of the user with any other background.	

	Story: Sohail and his family had plans to visit Bangkok this summer vacation. Unfortunately, they had to cancel the trip due to COVID situation in his country. However, Sohail decided to play a prank on his friends. Using his programming skills, he decided to create pictures of herself containing the background of the famous places in Bangkok City - The Grand Palace, Temple of Emerald Buddha etc. He would then send these pictures to his friends as if he is already on the planned trip. Sohail has several pictures of himself in different postures with black backgrounds. Can you help Sohail mask this black background with pictures of famous places in Bangkok. I am very excited to see your project solution and I know you will do really well. Bye Bye!	
Teacher Clicks ✖ End Class		
Additional Activities	*Encourage the student to write reflection notes in their reflection*	*The student uses the markdown editor to write*

	journal using markdown. Use these as guiding questions: ● What happened today? - Describe what happened - Code I wrote ● How did I feel after the class? ● What have I learned about programming and developing games? ● What aspects of the class helped me? What did I find difficult?	*her/his reflection in a reflection journal.*

Activity	Activity Name	Links
Teacher Activity 1	Teacher reference code	https://github.com/whitehatjr/invisibility_cloak/tree/master/C121