

PRETRAINED MODELS



What is our GOAL for this MODULE?

We learned how to add a clown nose filter to our webcam live feed.

What did we ACHIEVE in the class TODAY?

- Drew a red circle on the nose.
- Drew the clown nose image on the nose.
- Learned how to host an image on a server.

Which CONCEPTS/ CODING did we cover today?

- Used the circle() function to draw a red circle on the nose.
- Loaded a clown nose image in p5.js.

How did we DO the activities?

1. First we defined 2 variables for holding the x and y coordinate values of the nose, and initialize it with a value of 0.

```
noseX=0;  
noseY=0;  
  
function preload() {  
}
```

2. Stored the x and y coordinates of nose in the new variables

```
function gotPoses(results)
{
  if(results.length > 0)
  {
    console.log(results);
    noseX = results[0].pose.nose.x;
    noseY = results[0].pose.nose.y;
    console.log("nose x = " + noseX);
    console.log("nose y = " + noseY);
  }
}
```

- Output on console screen:

PoseNet Is Initialized	main.js:19
▶ [{}]	main.js:26
nose x = 117.61359750178822	main.js:29
nose y = 166.35042714793778	main.js:30

3. Then added code for drawing a circle using the **circle()** function. And in the **circle()** function passed **noseX** in place of **x-coordinate** and **noseY** in place of **y-coordinate**.

Code:

```
function draw() {
  image(video, 0, 0, 300, 300);
  circle(noseX, noseY, 20);
}
```

- 20 is the radius.

Also added the **color** and **border color** to the circle.

```
function draw() {  
  image(video, 0, 0, 300, 300);  
  fill(255,0,0);  
  stroke(255,0,0);  
  circle(noseX, noseY, 20);  
}
```

- **fill** is used to give **color**, inside fill we need to pass the rgb value for the color we want.
- **stroke** is used to give **border-color**, inside fill we need to pass the rgb value for the color we want.

4. Then we added code for loading the clown nose image inside the **preload()** function

```
noseX=0;  
noseY=0;  
  
function preload() {  
  clown_nose = loadImage('https://i.postimg.cc/7ZBcjDqp/clownnose.png');  
}
```

- **clown_nose** - this is the variable in which we stored the loaded image.
 - **loadImage** - it is a predefined p5.js function used to load the image. In this we need to pass the image URL.
5. Then inside the **draw()** function added code for drawing the clown nose image on the x and y coordinates of the nose.

```
function draw() {  
  image(video, 0, 0, 300, 300);  
  fill(255,0,0);  
  stroke(255,0,0);  
  circle(noseX, noseY, 20);  
  image(clown_nose, noseX, noseY, 30, 30);  
}
```

Explaining the above code:

- **image** - It is a predefined function of p5.js for loading an image on the canvas. Use this function for loading the clown nose image on the canvas.
- **clown_nose** - It holds the URL of the clown nose image.
- **noseX,** - It has the x coordinate of the nose, so at this x coordinate we want our clown nose image to be drawn.
- **noseY** - It has the y coordinate of the nose, so at this y coordinate we want our clown nose image to be drawn.
- **30, 30);** - The first 30 is for the width of the clown nose image. And the second 30 is for the height of the clown nose image. Both values are in pixels. You can give the width and height as you wish.

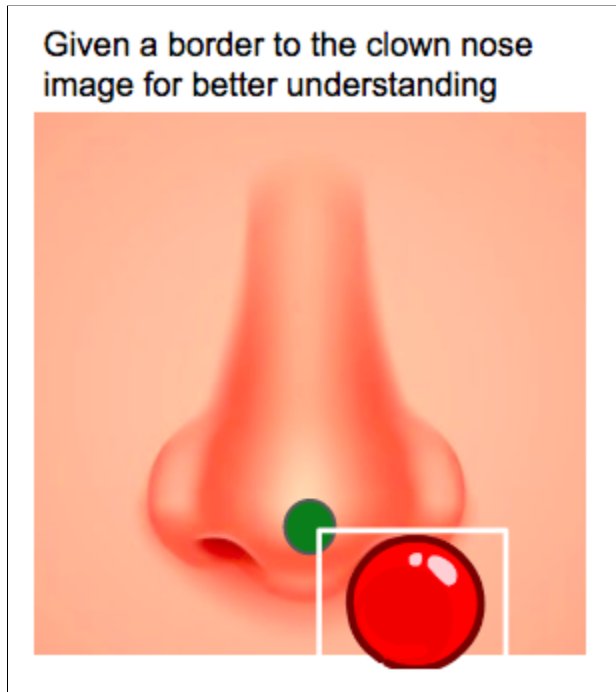
Also removed the code of drawing a circle.

```
function draw() {  
  image(video, 0, 0, 300, 300);  
  image(clown_nose, noseX, noseY, 30, 30);  
}
```

After coding this, we did a tested and observed that the clown nose image was going a little away from the nose.

Hence it means the clown image was not aligned with the nose properly, this happens

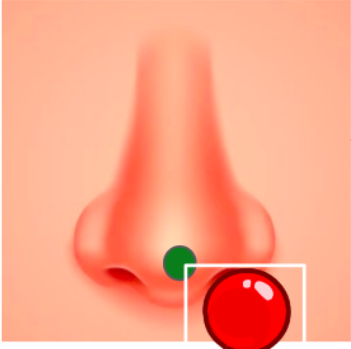
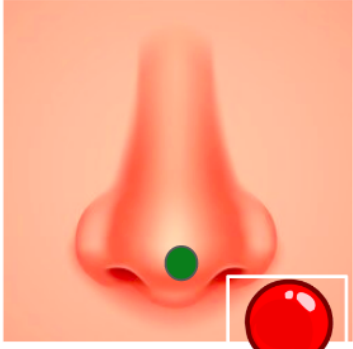
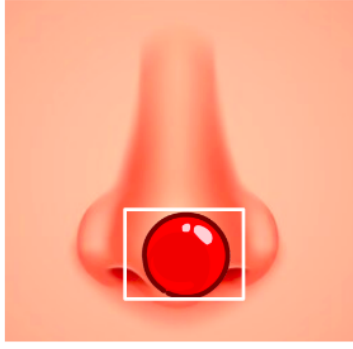
because:



Consider that the **green dot** is the location of our nose which is coming from posenet and is in the form of x and y coordinates of the nose.

So when we draw any image at any coordinates, the drawing of the image starts from the **top left corner of the image and not from the center of the image**, this leads to the alignment problem.

So we had fixed this by playing with the values of variable noseX and noseY like this:

<p><--- x coordinates --></p>  <p>x- coordinates increases left to right</p>	<p>If we increase x and y coordinates of the nose then clown nose image will move away from the nose. As the image given below</p> 	<p>If we decrease x and y coordinates of the nose then clown nose image will move closer to the nose. As the image given below</p>  <p>So as a conclusion we need to decrease x and y coordinates of the nose, that will result in moving the clown nose image closer to the nose. As the image given above</p>
---	--	---

6. And we decreased x and y coordinates by 5 pixels, like this:

```
function gotPoses(results)
{
  if(results.length > 0)
  {
    console.log(results);
    noseX = results[0].pose.nose.x - 5;
    noseY = results[0].pose.nose.y - 5;
  }
}
```

- And did a live test, and if still clown nose was not aligned then again decreased x and y coordinates by 5 more pixels:

```
function gotPoses(results)
{
  if(results.length > 0)
  {
    console.log(results);
    noseX = results[0].pose.nose.x - 10;
    noseY = results[0].pose.nose.y - 10;
  }
}
```

- And did a live test, and if still clown nose was not aligned then again decreased x and y coordinates by 5 more pixels:

```
function gotPoses(results)
{
  if(results.length > 0)
  {
    console.log(results);
    noseX = results[0].pose.nose.x - 15;
    noseY = results[0].pose.nose.y - 15;
  }
}
```

What's NEXT?

In the next class we will start working on a new neural network model which is able to identity the sketches