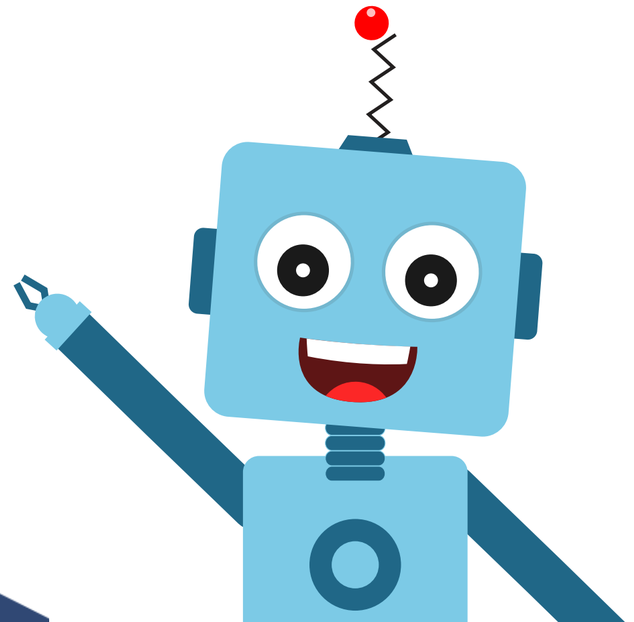
POWERED BY STEMpedia

Face Detection

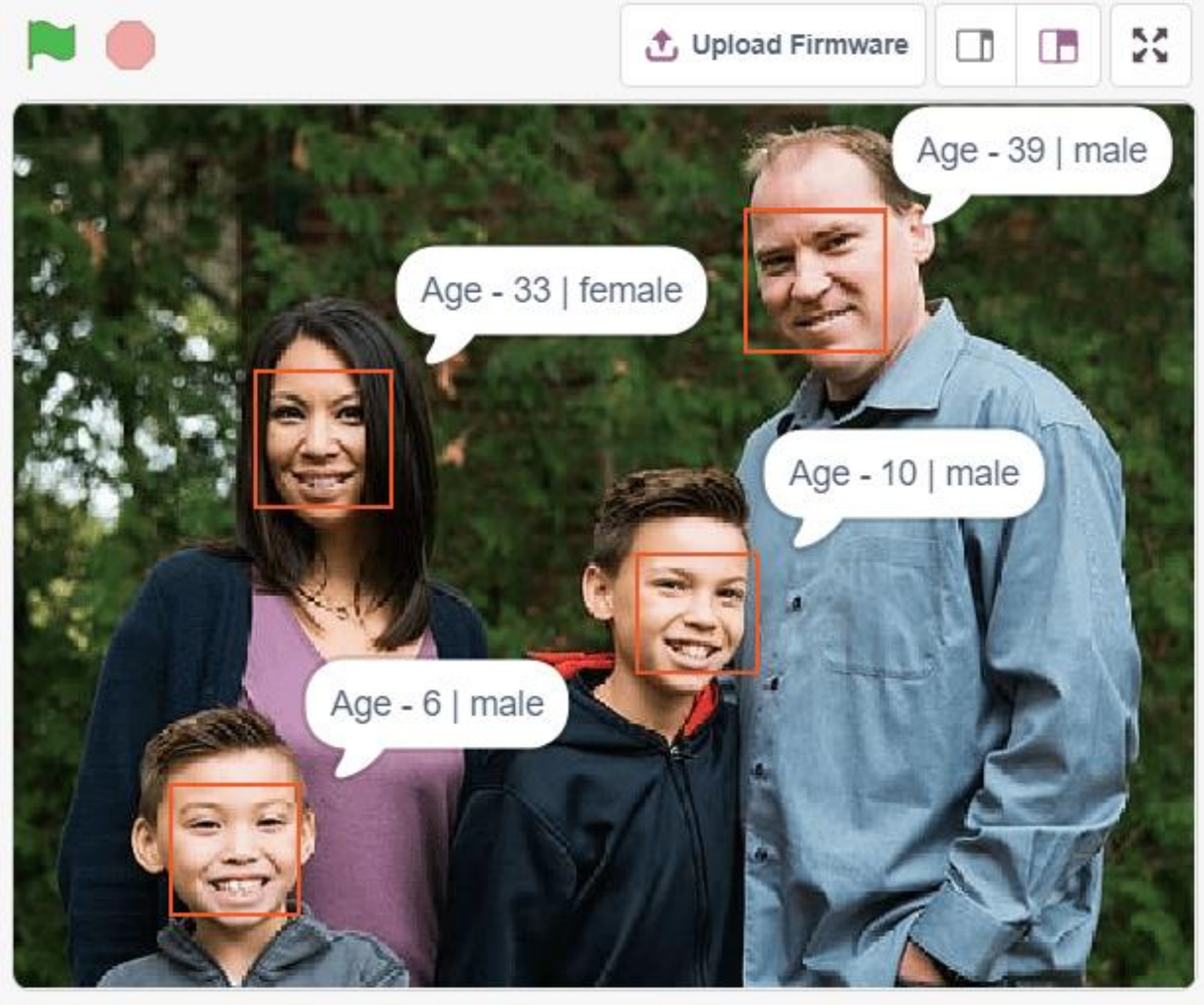
Session 14



What you will learn:

POWERED BY STEMpedia

- Face Detection in image (as shown in the image)
- Face detection functions in PictoBlox



Face Detection

POWERED BY STEMpedia

Definition- *Face detection is the action of locating human faces in an image and optionally, returning different kinds of face-related data.*

- **Face recognition** has become one of the most promising applications of **computer vision (CV)**. **Face detection** can be considered to be a substantial part of **face recognition.**

- **Why face detection in images is complicated?:** Human faces are widely different from one another! They can have different poses, expressions, position, and orientation, skin color, have glasses or facial hair or not, etc. Then, there are also differences in camera gain, lighting conditions, and image resolution.

How do we detect faces?

POWERED BY STEMpedia

There are a few methods that you can use to detect faces such as:

1) **Knowledge-Based** detection

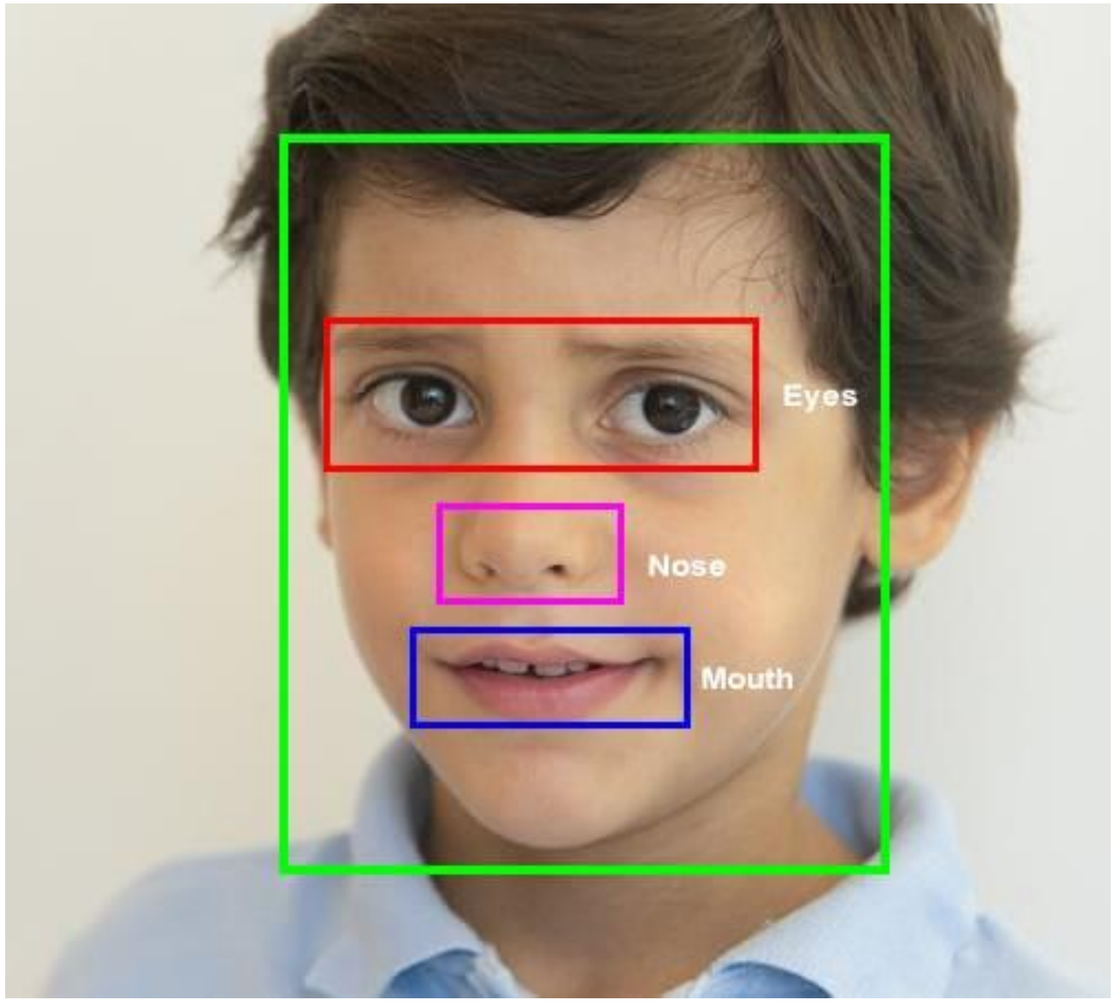
2) **Feature-Based** detection

Knowledge based Detection

POWERED BY STEMpedia

- The knowledge-based method **depends on a set of rules** and is based on **human knowledge** to detect the faces.

- E.g. A face often appears with two eyes that are symmetric to each other, a nose and a mouth.



Knowledge based Detection

POWERED BY STEMpedia

Pros

1) Easy to come up with simple rules.
2) Based on the coded rules. Facial features in an input image are extracted first, and face candidates are identified.

Cons

1) Difficult to translate human knowledge into rules precisely.
    i. Detailed rules fail to detect faces, like the rule that, ‘A person must have two eyes’: According to this rule, no face will be identified in this image.

Knowledge based Detection

ii. General rules may find many false positives: Like there is an eye in a face. According to this, the face will be identified in this tattoo of a similar-looking eye.

2) This approach alone is insufficient and unable to find many faces in multiple images.

Feature-Based Detection

POWERED BY STEMpedia

- Feature-based detection uses what are known as **face landmarks** to detect faces.

- **Face landmarks** are a set of easy-to-find points on a face, such as the pupils or the tip of the nose.

- By default, there are **68 predefined landmark points**. The following image shows all landmark points.

- The feature-based method detects faces by extracting these face landmarks.

Feature-Based Detection

POWERED BY STEMpedia

Pros

- Features are **invariant to pose and orientation change**. That is, face gets detected even if it is moved, rotated, or distorted in the image.
- This method has a **higher chance of face detection** (compared to other methods).

Cons

- Difficult to locate facial features due to noise in the images.
- Difficult to detect features in complex backgrounds.

Face Detection Functions in Python using PictoBlox

Now, we will use the variable ‘fd‘ the face detection object to build the Face Detection functions.

Python Functions for Face Detection

POWERED BY STEMpedia

In order to use the **Face Detection** functions for python in PictoBlox, we must first build the face detection object. For this follow the steps:

1) First, create a new project in **PictoBlox** and select **Python Coding** as your coding environment.
2) Now write the below code in the scripting area. This code, builds a face detection object and store it in the variable '**fd**'.

```
fd = FaceDetection()
```

1) Now, we will use the variable '**fd**' that contains the face detection object to build the **Face Detection** functions.

Facial Detection Function

POWERED BY STEMpedia

- To execute face detection, we can use two different functions, one **to detect face from the camera** and another **to detect face from the stage**.

1) To detect face from the camera, use:

```
fd.analysecamera()
```

2) To detect face from the stage, use:

```
fd.analysestage()
```

- The above two functions analyses the image (from camera or stage) and saves the face information locally. This face information can then be accessed using other python functions.

- You have to run these functions every time you want to analyze a new image from the camera or stage.

Controlling Camera Feed

POWERED BY STEMpedia

- You can control the camera feed on the stage using the following function:

```
fd.video("on", 0)
```

- This function has two parameters, which you can change as follows:

1) **Camera State**:
   i. **off** - The video feed on the stage will stop.
   ii. **on** – The video feed on the stage will start.
   iii. **on flipped** - The video feed on the stage will start with the video mirrored. So, your right hand will be shown as left hand and vice versa.

Controlling Camera Feed

POWERED BY STEMpedia

2) **Transparency**: This parameter makes the video translucent. You can give any value from 0 to 100.
   i. At value ‘0’, the camera feed will be shown as it is on the stage.
   ii. At value ‘50’, the camera feed will be semi-transparent.
   iii. At value ‘100’ the camera feed will fade out completely from the stage.

Example: Analyzing Image on the Stage

POWERED BY STEMpedia

1) Download this image from the link: https://ai.thestempedia.com/wp-content/uploads/2022/02/Boy-and-Girl.jpg

2) In PictoBlox, turn off the camera by running the function **fd.video()**, and choosing the first parameter '**off**' as shown.

3) Next upload the downloaded image as backdrop, by clicking the Upload Backdrop button.

Example: Analyzing Image on the Stage

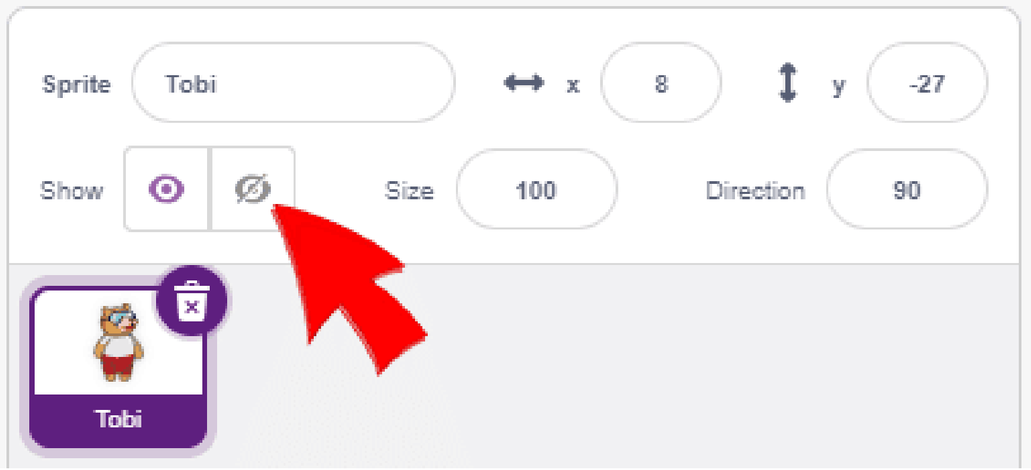
POWERED BY STEMpedia

4) Select **Tobi** sprite and **hide** it.

5) Now, to analyse image from the stage area, we use the function **analysestage().**

```
fd.analysestage()
```

6) Now we will see what things we get when we analyze images in face detection.



Get number (#) of faces

POWERED BY STEMpedia

- Once you have analyzed the images, you can use the function **count()** to get the number of faces recognized from the image.

```
fd.count()
```

- If we want to output the number of faces detected in the terminal of PictoBlox, we need to first convert the output of **fd.count()** to string.

- We do this by using the inbuilt function **str()** of python. Finally, we use **print()** function on it to output it in the terminal, as shown.

```
print(str(fd.count()))
```

Get expression of face

POWERED BY STEMpedia

- In order to get the expression of face from the camera, we can use **expression()** function.

```
fd.expression()
```

- This function **reports the recognized expression** of the selected face. If no face is detected, then it reports **NULL**.

- We can make the sprite say the emotion that is detected from the camera using the sprite function **say()**.

```
sprite.say()
```

Get expression of face

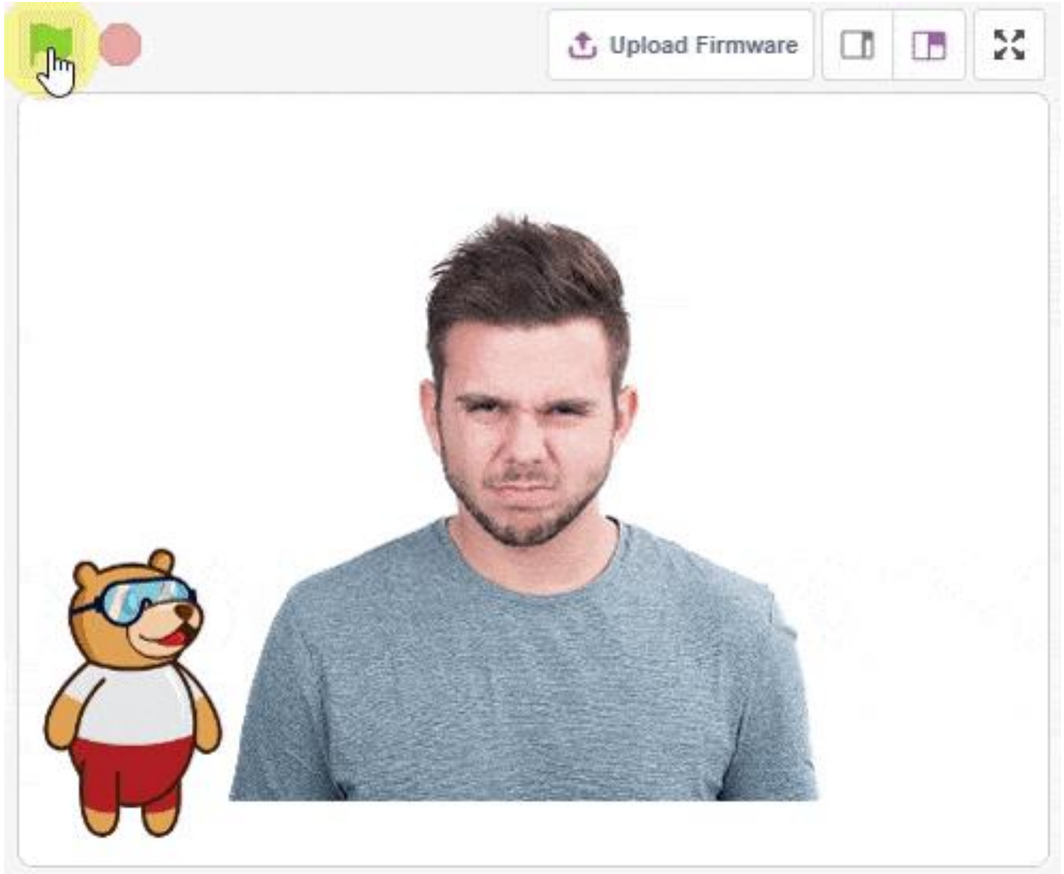
POWERED BY STEMpedia

- The entire code must include the following code as shown:

```
sprite = Sprite("Stage")

fd = FaceDetection()

sprite.say(fd.expression())
```



isexpression(face_number, emotion)

POWERED BY STEMpedia

- In order to check whether a **particular face** has a **particular emotion** or not, we can use the function **isexpression()**.
- If the **emotion matches**, then the block returns the value **true**, otherwise it returns the value **false**.
- This function has two parameters. The **first parameter** denotes the **face number** that you want to analyze and the **second parameter** denotes the emotion that you want to check on the face.
- **Example:** If we want to check whether for face 1 in the image, the emotion is ‘happy’ or not, we code:

```
fd.isexpression(1, ‘happy’)
```

Similarly, for checking ‘angry’ emotion on face 2, we code:

```
fd.isexpression(2, ‘angry’)
```

Getting positions of the face

POWERED BY STEMpedia

- In order to get various attributes of the face that is detected, like its **x-coordinate**, **y-coordinate**, **width** and **height** we use a bunch of face detection functions:

1) X position:
```
fd.x(1)
```
2) Y position:
```
fd.y(1)
```
3) Width:
```
fd.width(1)
```
4) Height:
```
fd.height(1)
```

- All the 4 functions above, have a single parameter that denotes the face number.

Note: These functions report an accurate position, when the image is analyzed from stage or camera feed.

Get Position of Landmarks of Face

POWERED BY STEMpedia

- Each face that is detected in PictoBlox using the face detection module has **landmarks** (from **1 to 68**), as shown in the image below.

- We can get the x and y position of any of these landmarks for any face using the below two functions:

  1) For getting the x-position:

```
fd.landmarkx(face_num, landmark_num)
```

  2) For getting the y-position:

```
fd.landmarky(face_num, landmark_num)
```

Get Position of Landmarks of Face

POWERED BY STEMpedia

Example:

- To get the x-position of the 1st image, 20th landmark, we code:

```
fd.landmarkx(1, 20)
```

- To get the y-position of the 2nd image, 1st landmark, we code:

```
fd.landmarky(2, 1)
```

POWERED BY
STEMpedia

www.ai.thestempedia.com