Laboratory of Image Processing

Face Detection & Tracking

Pier Luigi Mazzeo

pierluigi.mazzeo@cnr.it

Institute of Applied Sciences and Intelligent Systems

ScienceApp

National Research Council of Italy

Face Detection & Tracking

- This example shows how to automatically detect and track a face.

- Object detection and tracking are important in many computer vision applications including activity recognition, automotive safety, and surveillance. In this example, you will develop a simple face tracking system by dividing the tracking problem into three separate problems:
  - Detect a face to track
  - Identify facial features to track
  - Track the face

Step 1: Detect a Face to Track

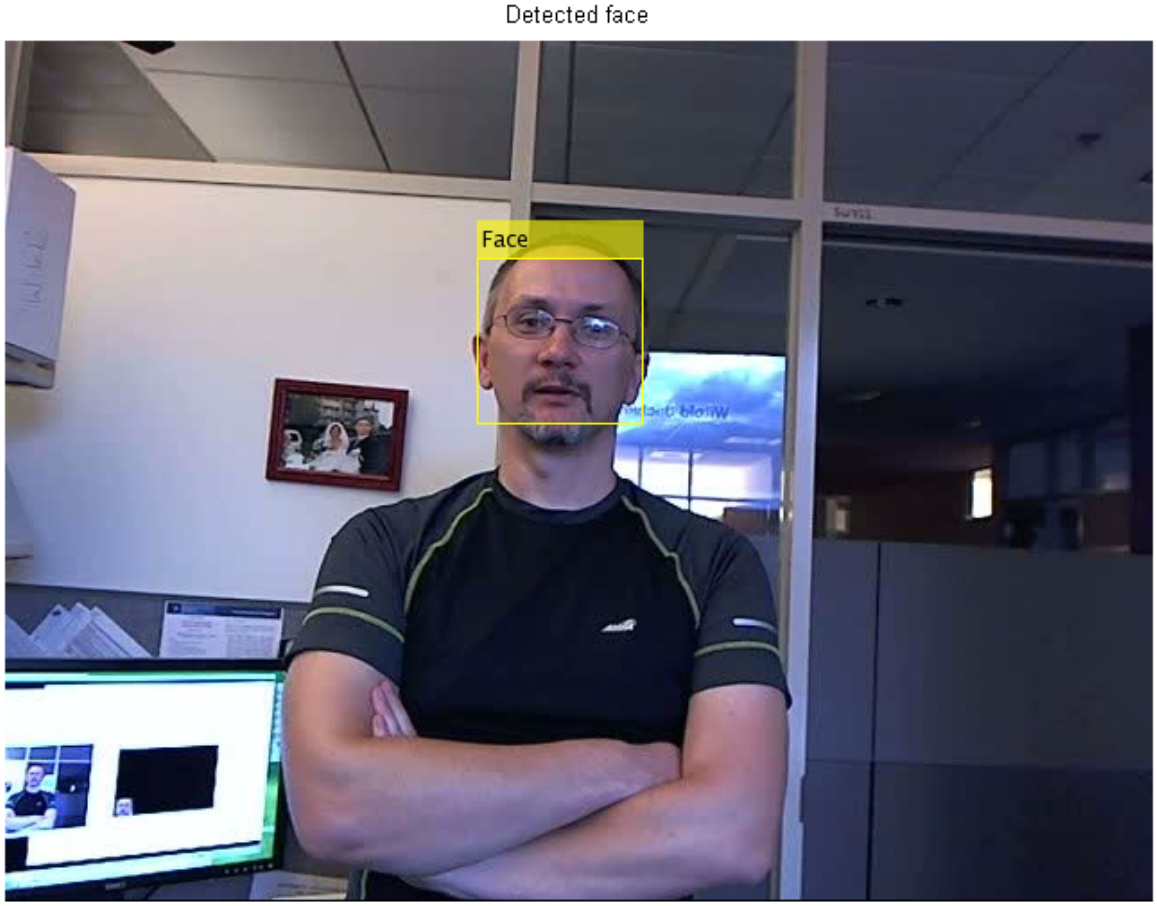
Before you begin tracking a face, you need to first detect it. Use the vision.CascadeObjectDetector to detect the location of a face in a video frame. The cascade object detector uses the Viola-Jones detection algorithm and a trained classification model for detection. By default, the detector is configured to detect faces, but it can be configured for other object types.

```
% Create a cascade detector object.
faceDetector = vision.CascadeObjectDetector();

% Read a video frame and run the detector.
videoFileReader =
vision.VideoFileReader('visionface.avi');
videoFrame      = step(videoFileReader);
bbox            = step(faceDetector, videoFrame);

% Draw the returned bounding box around the
detected face.
videoOut =
insertObjectAnnotation(videoFrame,'rectangle',bbo
x,'Face');
figure, imshow(videoOut), title('Detected face');
```

Step 1: Detect a Face to Track



You can use the cascade object detector to track a face across successive video frames. However, when the face tilts or the person turns their head, you may lose tracking. This limitation is due to the type of trained classification model used for detection. To avoid this issue, and because performing face detection for every video frame is computationally intensive, this example uses a simple facial feature for tracking.

Step 2: Identify Facial Features to Track

- Once the face is located in the video, the next step is to identify a feature that will help you track the face. For example, you can use the shape, texture, or color. Choose a feature that is unique to the object and remains invariant even when the object moves.
- In this example, you use skin tone as the feature to track. The skin tone provides a good deal of contrast between the face and the background and does not change as the face rotates or moves.

```
% Get the skin tone information by extracting the Hue
from the video frame
% converted to the HSV color space.
[hueChannel,~,~] = rgb2hsv(videoFrame);

% Display the Hue Channel data and draw the bounding
box around the face.
figure, imshow(hueChannel), title('Hue channel
data');
rectangle('Position',bbox(1,:),'LineWidth',
2,'EdgeColor',[1 1 0])
```

Step 2: Identify Facial Features to Track

Step 3: Track the Face

With the skin tone selected as the feature to track, you can now use the `vision.HistogramBasedTracker` for tracking. The histogram based tracker uses the CAMShift algorithm, which provides the capability to track an object using a histogram of pixel values. In this example, the Hue channel pixels are extracted from the nose region of the detected face. These pixels are used to initialize the histogram for the tracker. The example tracks the object over successive video frames using this histogram.

```
% Detect the nose within the face region. The nose provides a
more accurate
% measure of the skin tone because it does not contain any
background  pixels.
noseDetector = vision.CascadeObjectDetector('Nose');
faceImage    = imcrop(videoFrame,bbox(1,:));
noseBBox     = step(noseDetector,faceImage);
```

Step 3: Track the Face

```
% The nose bounding box is defined relative to the cropped
face image.
% Adjust the nose bounding box so that it is relative to the
original video
% frame.
noseBBox(1,1:2) = noseBBox(1,1:2) + bbox(1,1:2);

% Create a tracker object.
tracker = vision.HistogramBasedTracker;

% Initialize the tracker histogram using the Hue channel
pixels from the
% nose.
initializeObject(tracker, hueChannel, noseBBox(1,:));

% Create a video player object for displaying video frames.
videoInfo    = info(videoFileReader);
videoPlayer  = vision.VideoPlayer('Position',[300 300
videoInfo.VideoSize+30]);
```

Step 3: Track the Face

```
% Track the face over successive video frames until the video is
finished.
while ~isDone(videoFileReader)
    % Extract the next video frame
    videoFrame = step(videoFileReader);
    % RGB -> HSV
    [hueChannel,~,~] = rgb2hsv(videoFrame);
    % Track using the Hue channel data
    bbox = step(tracker, hueChannel);

    % Insert a bounding box around the object being tracked
    videoOut =
insertObjectAnnotation(videoFrame,'rectangle',bbox,'Face');

    % Display the annotated video frame using the video player
object
    step(videoPlayer, videoOut);

end
```

Step 3: Track the Face

```
% Release resources
release(videoFileReader);
release(videoPlayer);
```