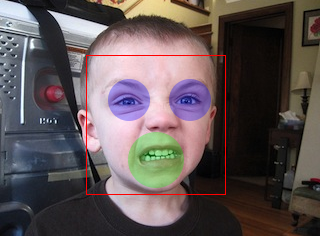
With the face detection API included within Core Image in the iOS 5 SDK facial recognition is now dead simple on devices running iOS 5, and it works extremely well.

Using this new API you can quickly detect the size of the face along with the locations of the mouth and nose.  As illustrated in this image:

[](http://maniacdev.com/wp-content/uploads/2011/11/Screen-shot-2011-11-02-at-2.15.29-PM.png)

No more need to roll your own code or use a framework such as OpenCV for most face recognition needs.

With adjustable levels of accuracy the face recognition API can be used in situations demanding high accuracy or high speed (such as when working with live video).

**1) Set Up The Project**

a) Create a Single View Application, I named mine FaceDetectionExample.

b) Include the **QuartzCore** and **CoreImage** frameworks within the project.

c) Drag the facedetectionpic.jpg file into the project.

**2) Import The Frameworks And Draw The Image**

a) Import the Quartz and Core Image frameworks into the AppDelegate.m file.

*#import <CoreImage/CoreImage.h>*  
*#import <QuartzCore/QuartzCore.h>*

b) Add the following method to draw the image onto the screen. I placed the faceDetector, and markFaces methods above the application:didFinishLaunching: method.

-(void)faceDetector  
{  
*// Load the picture for face detection*  
UIImageView\* image = [[UIImageView alloc] initWithImage:  
[UIImage imageNamed:*@*"facedetectionpic.jpg"]];  
  
*// Draw the face detection image*  
[self.window addSubview:image];  
  
*// Execute the method used to markFaces in background*  
[self markFaces:image];  
}

**3) Detect the faces**

a) Create a **CIImage** (Core Image image) using  the image in the**UIImageView** that we created in Step 2.

-(void)markFaces:(UIImageView \*)facePicture  
{  
*// draw a CI image with the previously loaded face detection picture*  
CIImage\* image = [CIImage imageWithCGImage:facePicture.image.CGImage];

b) Create the **CIDetector**.

Since we’re working with a still image here we will use a detector of high accuracy.  You can read about other **CIDetector** options available in Apple’s**CIDetector** documentation [here](http://developer.apple.com/library/ios/#documentation/CoreImage/Reference/CIDetector_Ref/Reference/Reference.html).

*// create a face detector - since speed is not an issue we'll use a high accuracy*  
*// detector*  
CIDetector\* detector = [CIDetector detectorOfType:CIDetectorTypeFace  
context:nil options:[[NSDictionary](http://developer.apple.com/documentation/Cocoa/Reference/Foundation/Classes/NSDictionary_Class/) dictionaryWithObject:CIDetectorAccuracyHigh forKey:CIDetectorAccuracy]];

c) Run the featuresInImage method in the **CIDetector** class on our **CIImage**to get an array containing the features of every face detected within the image.

*// create an array containing all the detected faces from the detector*  
[NSArray](http://developer.apple.com/documentation/Cocoa/Reference/Foundation/Classes/NSArray_Class/)\* features = [detector featuresInImage:image];

**4) Draw Shapes On The Found Faces**

The **CIFaceFeature** class provides us with the bounds for the face, the location of each eye, and mouth, and also BOOL’s indicating whether each eye or the mouth is found for each face.

You can read more on **CIFaceFeature** in Apple’s documentation [here](http://developer.apple.com/library/mac/#documentation/CoreImage/Reference/CIFaceFeature/Reference/Reference.html).

a) Iterate through the array of face features

*// we'll iterate through every detected face. CIFaceFeature provides us*  
*// with the width for the entire face, and the coordinates of each eye*  
*// and the mouth if detected. Also provided are BOOL's for the eye's and*  
*// mouth so we can check if they already exist.*  
for(CIFaceFeature\* faceFeature in features)  
{

b) Create a red border around each face found in the image using the feature bounds. We’ll also store the face width which we’ll be using for drawing on the other features of the face.

*// get the width of the face*  
CGFloat faceWidth = faceFeature.bounds.size.width;  
  
*// create a UIView using the bounds of the face*  
UIView\* faceView = [[UIView alloc] initWithFrame:faceFeature.bounds];  
  
*// add a border around the newly created UIView*  
faceView.layer.borderWidth = 1;  
faceView.layer.borderColor = [[UIColor redColor] CGColor];  
  
*// add the new view to create a box around the face*  
[self.window addSubview:faceView];

Now over the two eyes we’ll draw green circles.

 if(faceFeature.hasLeftEyePosition)  
{  
*// create a UIView with a size based on the width of the face*  
UIView\* leftEyeView = [[UIView alloc] initWithFrame:CGRectMake(faceFeature.leftEyePosition.x-faceWidth\*0.15, faceFeature.leftEyePosition.y-faceWidth\*0.15, faceWidth\*0.3, faceWidth\*0.3)];  
*// change the background color of the eye view*  
[leftEyeView setBackgroundColor:[[UIColor blueColor] colorWithAlphaComponent:0.3]];  
*// set the position of the leftEyeView based on the face*  
[leftEyeView setCenter:faceFeature.leftEyePosition];  
*// round the corners*  
leftEyeView.layer.cornerRadius = faceWidth\*0.15;  
*// add the view to the window*  
[self.window addSubview:leftEyeView];  
}  
  
if(faceFeature.hasRightEyePosition)  
{  
*// create a UIView with a size based on the width of the face*  
UIView\* leftEye = [[UIView alloc] initWithFrame:CGRectMake(faceFeature.rightEyePosition.x-faceWidth\*0.15, faceFeature.rightEyePosition.y-faceWidth\*0.15, faceWidth\*0.3, faceWidth\*0.3)];  
*// change the background color of the eye view*  
[leftEye setBackgroundColor:[[UIColor blueColor] colorWithAlphaComponent:0.3]];  
*// set the position of the rightEyeView based on the face*  
[leftEye setCenter:faceFeature.rightEyePosition];  
*// round the corners*  
leftEye.layer.cornerRadius = faceWidth\*0.15;  
*// add the new view to the window*  
[self.window addSubview:leftEye];  
}

c) Finally we’ll draw a circle over the mouth.

 if(faceFeature.hasMouthPosition)  
{  
*// create a UIView with a size based on the width of the face*  
UIView\* mouth = [[UIView alloc] initWithFrame:CGRectMake(faceFeature.mouthPosition.x-faceWidth\*0.2, faceFeature.mouthPosition.y-faceWidth\*0.2, faceWidth\*0.4, faceWidth\*0.4)];  
*// change the background color for the mouth to green*  
[mouth setBackgroundColor:[[UIColor greenColor] colorWithAlphaComponent:0.3]];  
*// set the position of the mouthView based on the face*  
[mouth setCenter:faceFeature.mouthPosition];  
*// round the corners*  
mouth.layer.cornerRadius = faceWidth\*0.2;  
*// add the new view to the window*  
[self.window addSubview:mouth];  
}  
}  
}

**5) Adjust For The Coordinate System**

If you were to run the app now you might notice that the y-locations of the circles drawn over the eyes and mouth are off, this is because of the different coordinate system used by Core Image (and the default on Mac OS X).

Flip the image, and then flip the entire window containing our newly created circles to make everything right side up. Doing things this way only requires a couple of lines of code which we’ll add into the **facedetector** method.

*// flip image on y-axis to match coordinate system used by core image*  
[image setTransform:CGAffineTransformMakeScale(1, -1)];  
  
*// flip the entire window to make everything right side up*  
[self.window setTransform:CGAffineTransformMakeScale(1, -1)];

**Conclusion**

Finally add the from the following code application: didFinishLaunchingWIthOptions: method before the return statement to run the face detector.

[self faceDetector];

That’s all there is to it!  Thanks to Tom of b2cloud who’s [tutorial](http://b2cloud.com.au/how-to-guides/face-detection-in-ios-5) on face detection I found after starting this one who’s code I used to simply this example. Also thanks to Tobyotter on Flickr for the monster face image.

**One more thing…**

Face detection can take awhile, especially on older devices so you may want to run your face detection method on the background.  You can simply change:

[self markFaces:image];

to

[self performSelectorInBackground:@selector(markFaces:) withObject:image];

and the face detection and drawing will run in a separate thread and the app will start up faster (some advice I picked up in the extensive Core Image section of the iOS 5 by Tutorials [book](http://maniacdev.com/ios-5-by-tutorials-book) (aff)).  Even on a newer device I can see the difference.

That’s all there is to it!  Please post any issues in the comments below.

**More iOS 5 SDK Programming Tutorials**

For more on iOS 5 programming check out the [iOS 5 tutorial](http://maniacdev.com/ios-5-sdk-tutorial-and-guide/" \t "_blank) page.

[inShare](javascript:void(0);)8

**Related posts:**

1. [Example: Using Core Image Face Detection On A Live Video Feed](http://maniacdev.com/2012/05/example-using-core-image-face-detection-on-a-live-video-feed" \o "Example: Using Core Image Face Detection On A Live Video Feed)
2. [Example Source Code: Using Face Detection To Create The Illusion Of Real 3D On iOS Devices](http://maniacdev.com/2012/07/example-source-code-using-face-detection-to-create-the-illusion-of-real-3d-on-ios-devices" \o "Example Source Code: Using Face Detection To Create The Illusion Of Real 3D On iOS Devices)
3. [Example Source Code: Face Detection And Augmented Reality](http://maniacdev.com/2011/02/example-source-code-face-detection-and-augmented-reality" \o "Example Source Code: Face Detection And Augmented Reality)
4. [Open Source: Library For Adding Easy Face Recognition To Your iOS App With The Free Face.Com API](http://maniacdev.com/2011/12/open-source-library-for-adding-easy-face-to-your-ios-app-with-the-free-face-com-api" \o "Open Source: Library For Adding Easy Face Recognition To Your iOS App With The Free Face.Com API)

**Submit A Resource**

Have you created a useful tutorial, library or tool for iOS development that you would like to get in front of our 300,000+ monthly page views from iOS developers?

You can [submit the url here](http://maniacdev.com/submit/).

The resources we feel will appeal to our readers the most will be posted on the front page.

**Comments**

« [Open Source: Library For Stacking Customized iPad UIViewControllers](http://maniacdev.com/2011/11/open-source-library-for-stacking-customized-ipad-uiviewcontrollers" \o "Open Source: Library For Stacking Customized iPad UIViewControllers)

[Solving Issues When Converting A Project To Use ARC (Automatic Reference…](http://maniacdev.com/2011/11/solving-issues-when-converting-a-project-to-use-arc-automatic-reference-counting" \o "Solving Issues When Converting A Project To Use ARC (Automatic Reference Counting)) »