

O'Reilly Where 2.0 Online Conference – 12/3/2009

# COMPUTER VISION AND THE IPHONE CAMERA + Q&A



Jeffrey Powers, Co-Founder, Occipital

- ◆ **Overview:** Computer Vision
- ◆ **Getting started** with iPhone Camera Development
  - ◆ Basic picture taking
- ◆ **Advanced** Development with the camera
  - ◆ Manipulating the live feed
  - ◆ Digging into the pixels
- ◆ **Future** of iPhone Computer Vision
- ◆ Q&A

Part 1/4

# Overview Computer Vision on iPhone

# Computer Vision on iPhone



[www.weeklyreader.com](http://www.weeklyreader.com)

Paramount Pictures

» Overview

# Iron Man Vision

## Not yet available on iPhone ☺



Paramount Pictures

» Overview

# Iron Man Vision

Not yet available on iPhone ☺



Paramount Pictures

» Overview

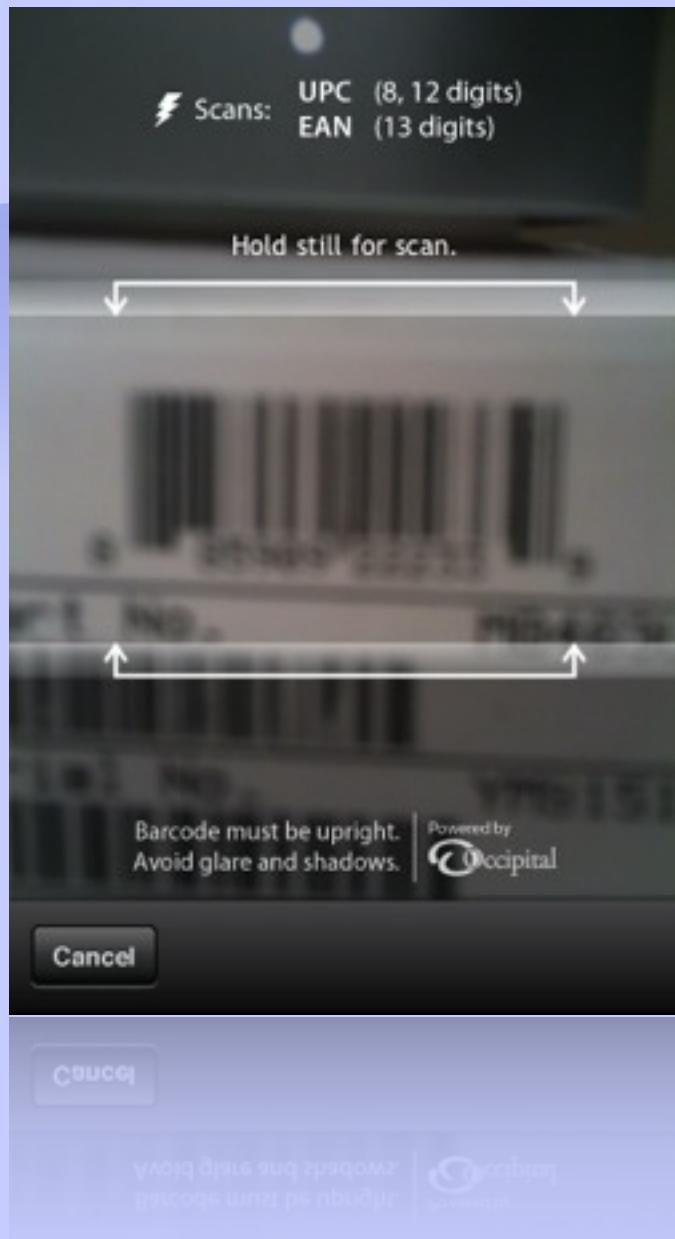
# Our stuff on iPhone Today

## RedLaser

A barcode scanner based on computer vision.



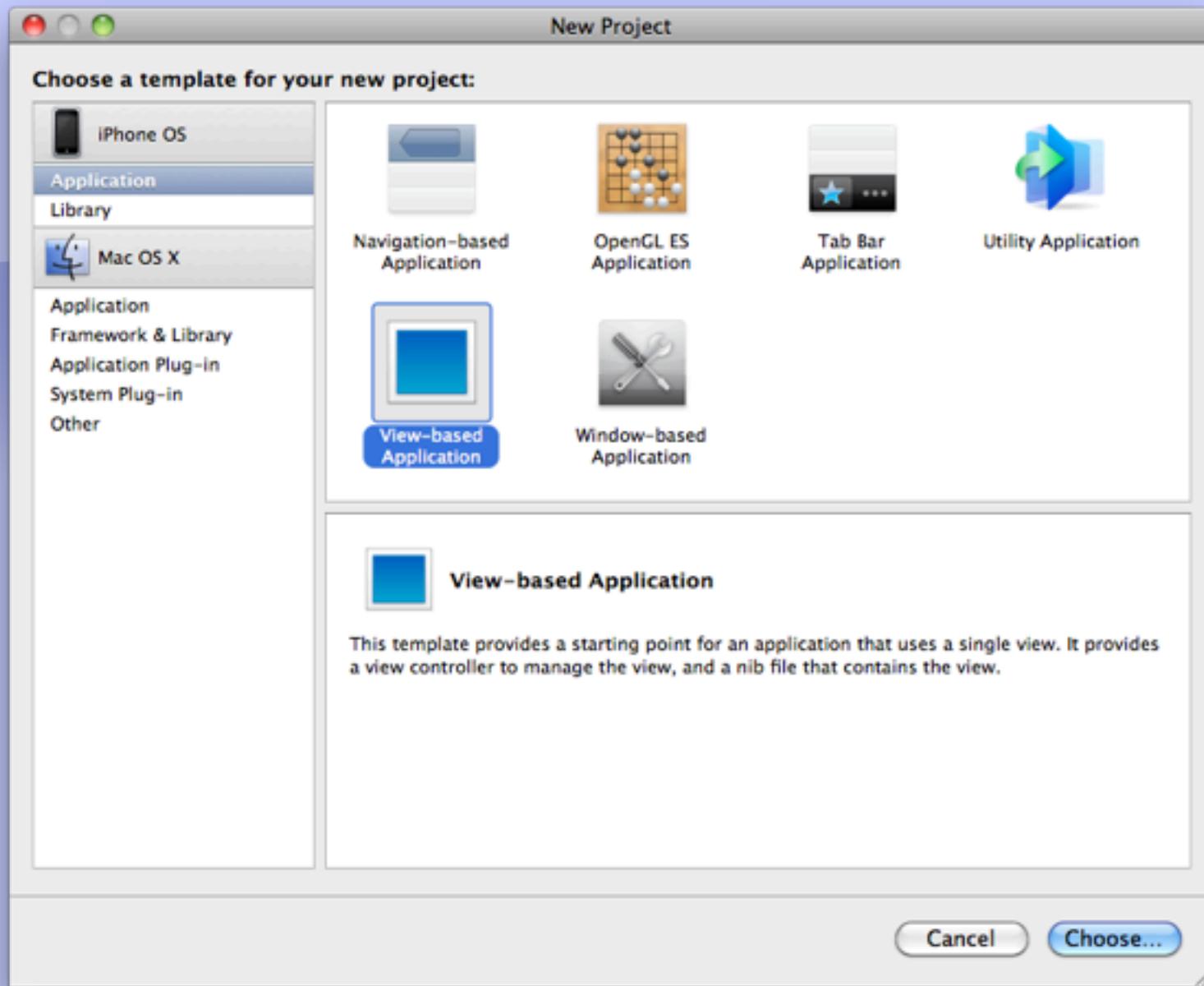
» Overview



» Overview

Part 2/4

# Getting Started with iPhone Camera Development



The screenshot shows the Xcode IDE interface with the following details:

- Title Bar:** CameraTestAppDelegate.m - CameraTest
- Toolbar:** Device - 3.1.2 | Debug, Action, Breakpoints, Build and Run, Tasks, Info, String Matching, Search.
- Editor Area:** Displays the code for `applicationDidFinishLaunching:`. The code initializes a window, creates a UIImagePickerController, checks for camera availability, sets the delegate, and presents the picker controller modally.

```
- (void)applicationDidFinishLaunching:(UIApplication *)application {
    // Override point for customization after app launch
    [window addSubview:viewController.view];
    [window makeKeyAndVisible];

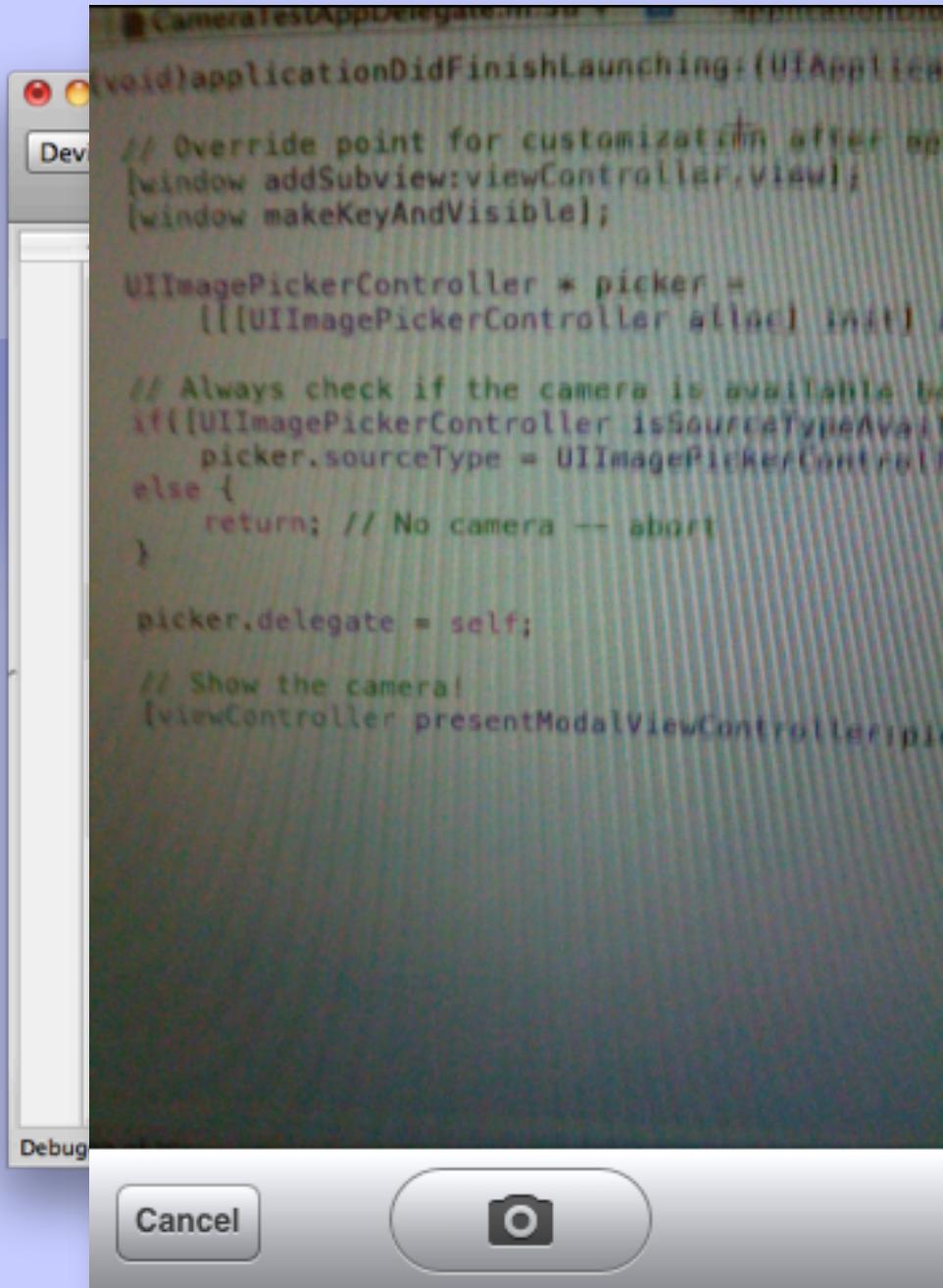
    UIImagePickerController * picker =
        [[[UIImagePickerController alloc] init] autorelease];

    // Always check if the camera is available before using it
    if([UIImagePickerController isSourceTypeAvailable:UIImagePickerControllerSourceTypeCamera])
        picker.sourceType = UIImagePickerControllerSourceTypeCamera;
    else {
        return; // No camera -- abort
    }

    picker.delegate = self;

    // Show the camera!
    [viewController presentModalViewController:picker animated:NO];
}
```

- Status Bar:** Debugging of "CameraTest" ended normally. Succeeded



Camera appears, but now what?

» Getting Started

CameraTestAppDelegate.m - CameraTest

Device - 3.1.2 | Debug    Action    Breakpoints    Build and Run    Tasks    Info    String Matching

Overview    Search

```
- (void)applicationDidFinishLaunching:(UIApplication *)application {
    // Override point for customization after app launch
    [window addSubview:viewController.view];
    [window makeKeyAndVisible];

    UIImagePickerController * picker =
        [[[UIImagePickerController alloc] init] autorelease];

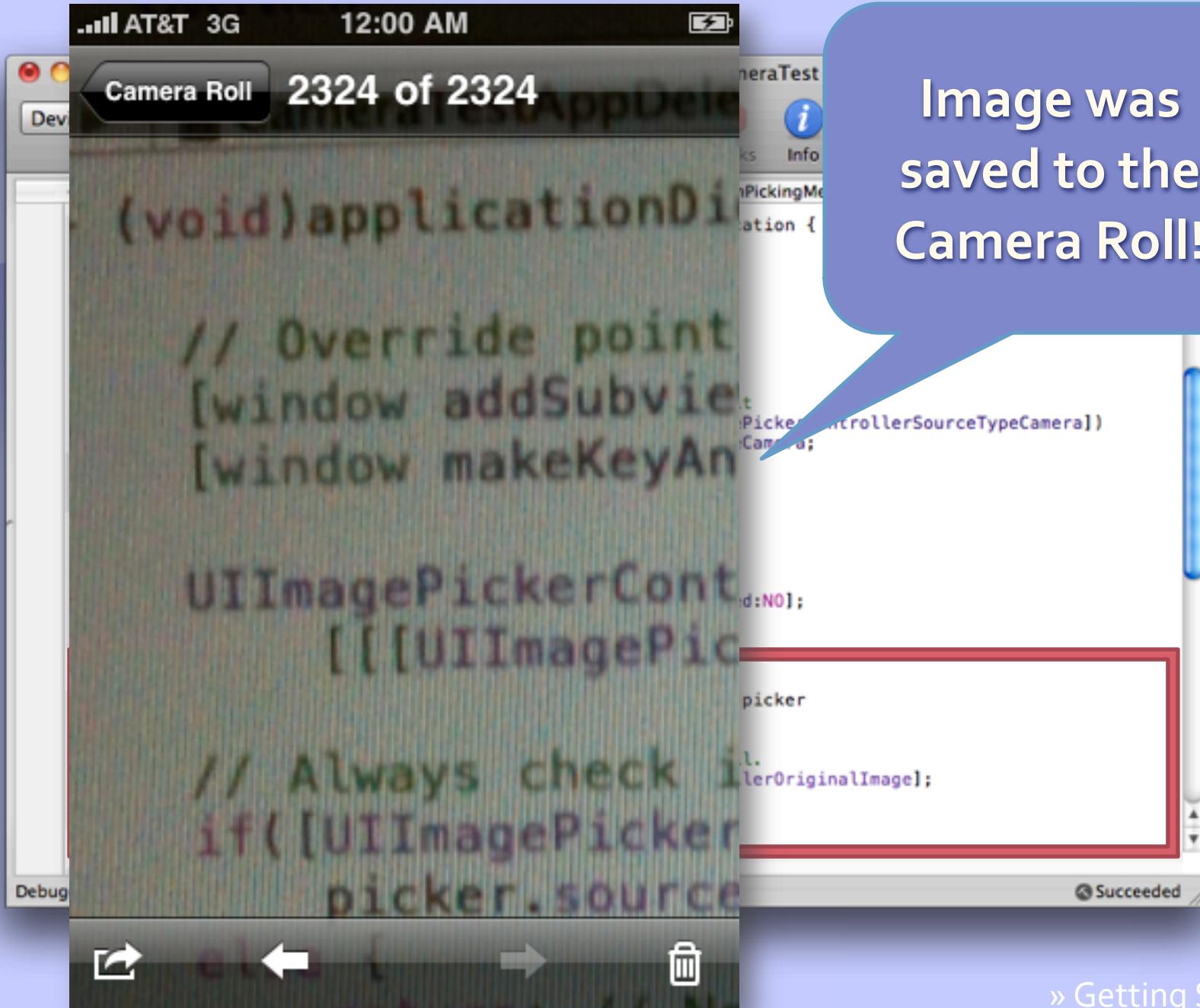
    // Always check if the camera is available before using it
    if([UIImagePickerController isSourceTypeAvailable:UIImagePickerControllerSourceTypeCamera])
        picker.sourceType = UIImagePickerControllerSourceTypeCamera;
    else {
        return; // No camera -- abort
    }

    picker.delegate = self;

    // Show the camera!
    [viewController presentModalViewController:picker animated:NO];
}

// Called when image has been captured.
- (void)    imagePickerController:(UIImagePickerController *)picker
    didFinishPickingMediaWithInfo:(NSDictionary *)info
{
    // Retrieve the captured image and save to the camera roll.
    UIImage * image = [info objectForKey:UIImagePickerControllerOriginalImage];
    UIImageWriteToSavedPhotosAlbum(image, nil, nil, nil);
}
```

Debugging of "CameraTest" ended normally.    Succeeded



» Getting Started

Part 3/4

# Advanced Development with the iPhone Camera

# Zoom in on the clock!

A multi-touch interface built using new APIs in iPhone OS 3.1.

Inverse-pinch to zoom in.



Zoom is applied to the live feed.

» Advanced: Manipulating the live feed (3.1+)

# Zoom in on the clock!

A multi-touch interface built using new APIs in iPhone OS 3.1.

Inverse-pinch to zoom in.

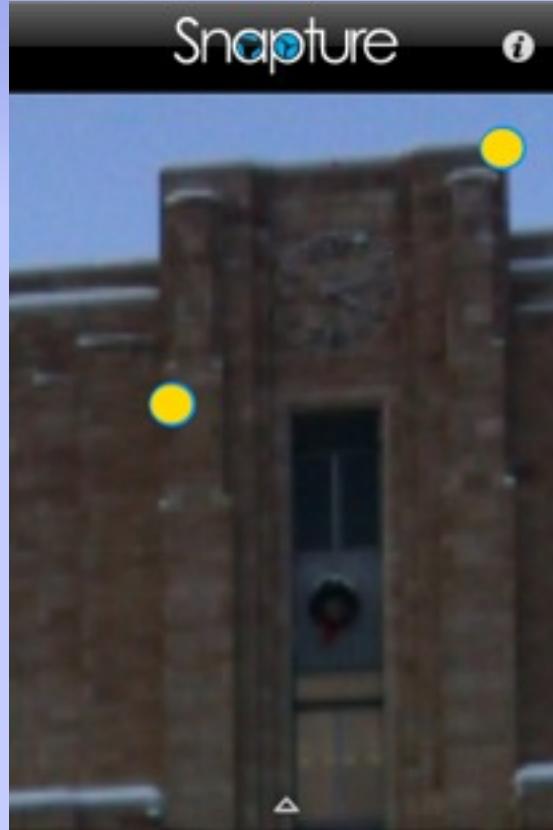


Zoom is applied to the live feed.

» Advanced: Manipulating the live feed (3.1+)

# Zoom in on the clock!

A multi-touch interface built using new APIs in iPhone OS 3.1.  
Let's see how it works...



Zoom is applied to the live feed.

» Advanced: Manipulating the live feed (3.1+)

TouchView.m – Snapture

Device - 3.1 | Release ▾ Action Build and Go Tasks Info String Matching

Overview Search

TouchView.m:430 -applyMultitouchTransform

```
// Calculate live view transform based on current multi-touch pinch action
- (CGAffineTransform) transform
{
    curScale = scale0 * scale;
    // Bound Zoom
    curScale = curScale < 1.0 ? 1.0 : curScale > MAX_ZOOM ? MAX_ZOOM : curScale;
    // Produce more natural motion after scale bound is exceeded (zoomed in too far)
    float effectiveScale = curScale / scale0;
    curTranslate.x = effectiveScale * translate0.x + translate.x;
    curTranslate.y = effectiveScale * translate0.y + translate.y;
    // Create affine transform
    return CGAffineTransformScale(
        CGAffineTransformMakeTranslation(curTranslate.x, curTranslate.y), curScale, curScale);
}

-(void)applyMultitouchTransform
{
    UIImagePickerController * picker = [SnaptureAppDelegate sharedInstance].mainPicker;
    picker.cameraViewTransform = [self transform]; // Set the camera transform (Requires 3.1+)
}
```

Found "Multitouch" - 1 occurrence

» Advanced: Manipulating the live feed (3.1+)

# Reading blurry barcodes.

So blurry that people can't even read them.



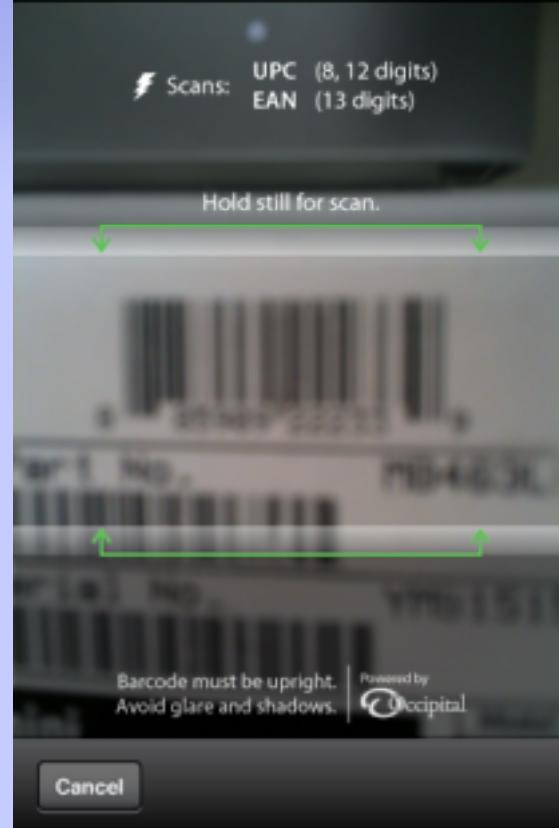
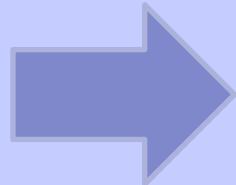
**How it works** (we'll get into these next)

- ◆ Custom overlay.
- ◆ Rapid screenshots.
- ◆ Image processing (computer vision).

» Advanced: Custom overlays and image processing.

# Reading blurry barcodes.

First ingredient: A custom camera overlay



**Default UI:**  
not suitable in this case.

**Helpful custom UI.**

» Advanced: (Custom overlays) and image processing.

```
Device - 3.1.2 | Debug Overview Action Breakpoints Build and Run Tasks Info String Matching Search
CameraTestAppDelegate.m - CameraTest
- (void)applicationDidFinishLaunching:(UIApplication *)application {
    // Override point for customization after app launch
    [window addSubview:viewController.view];
    [window makeKeyAndVisible];

    UIImagePickerController * picker =
        [[[UIImagePickerController alloc] init] autorelease];

    // Always check if the camera is available before using it
    if([UIImagePickerController isSourceTypeAvailable:UIImagePickerControllerSourceTypeCamera])
        picker.sourceType = UIImagePickerControllerSourceTypeCamera;
    else {
        return; // No camera -- abort
    }

    picker.delegate = self;

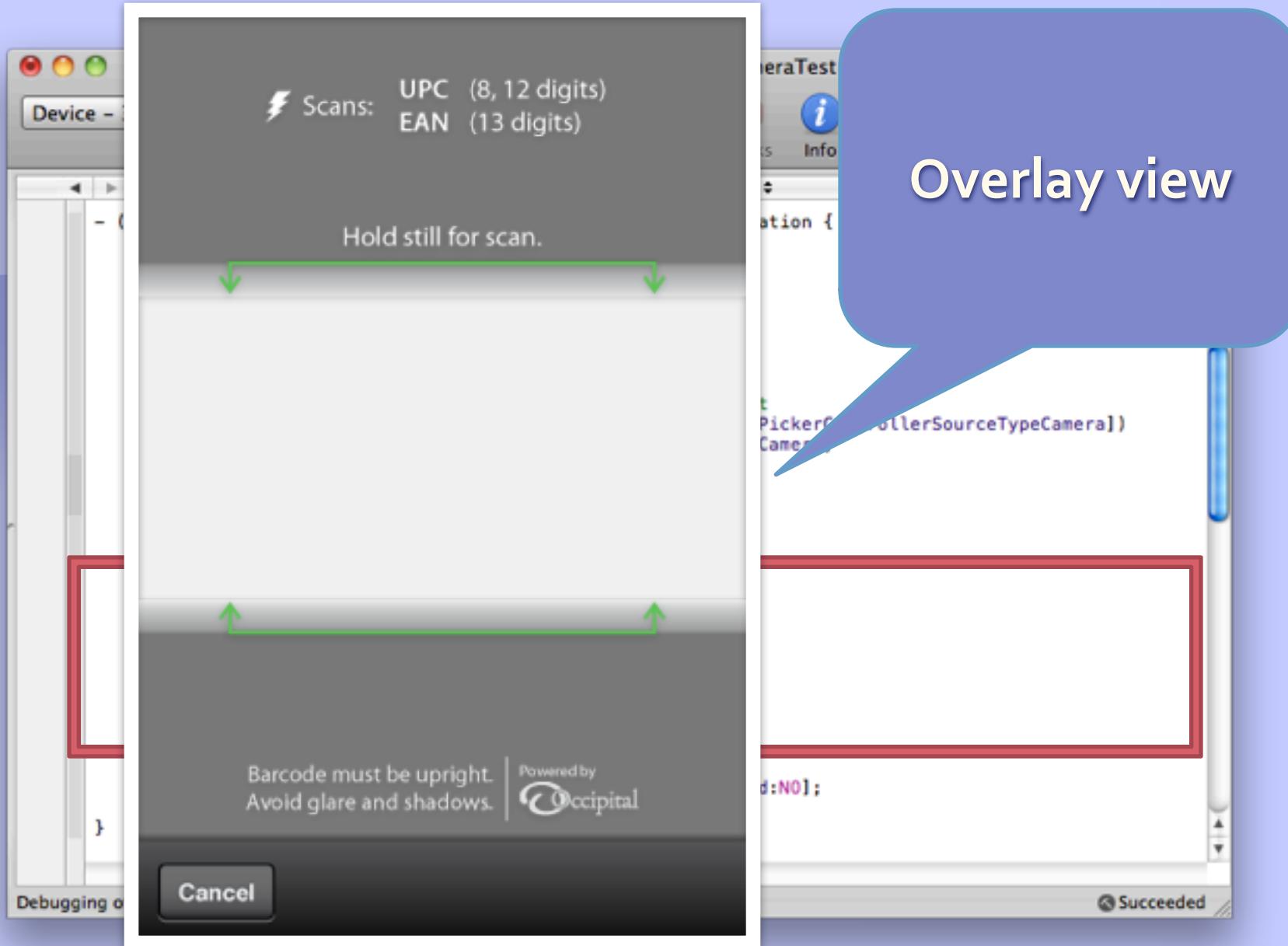
    // Allocate an overlay -- this can be ANY UIView
    UIImageView * overlayUI =
        [[[UIImageView alloc] initWithImage:
            [UIImage imageNamed:@"OverlayImage.png"]]
        autorelease];

    // Tell the image picker to use this overlay (3.1+)
    picker.cameraOverlayView = overlayUI;

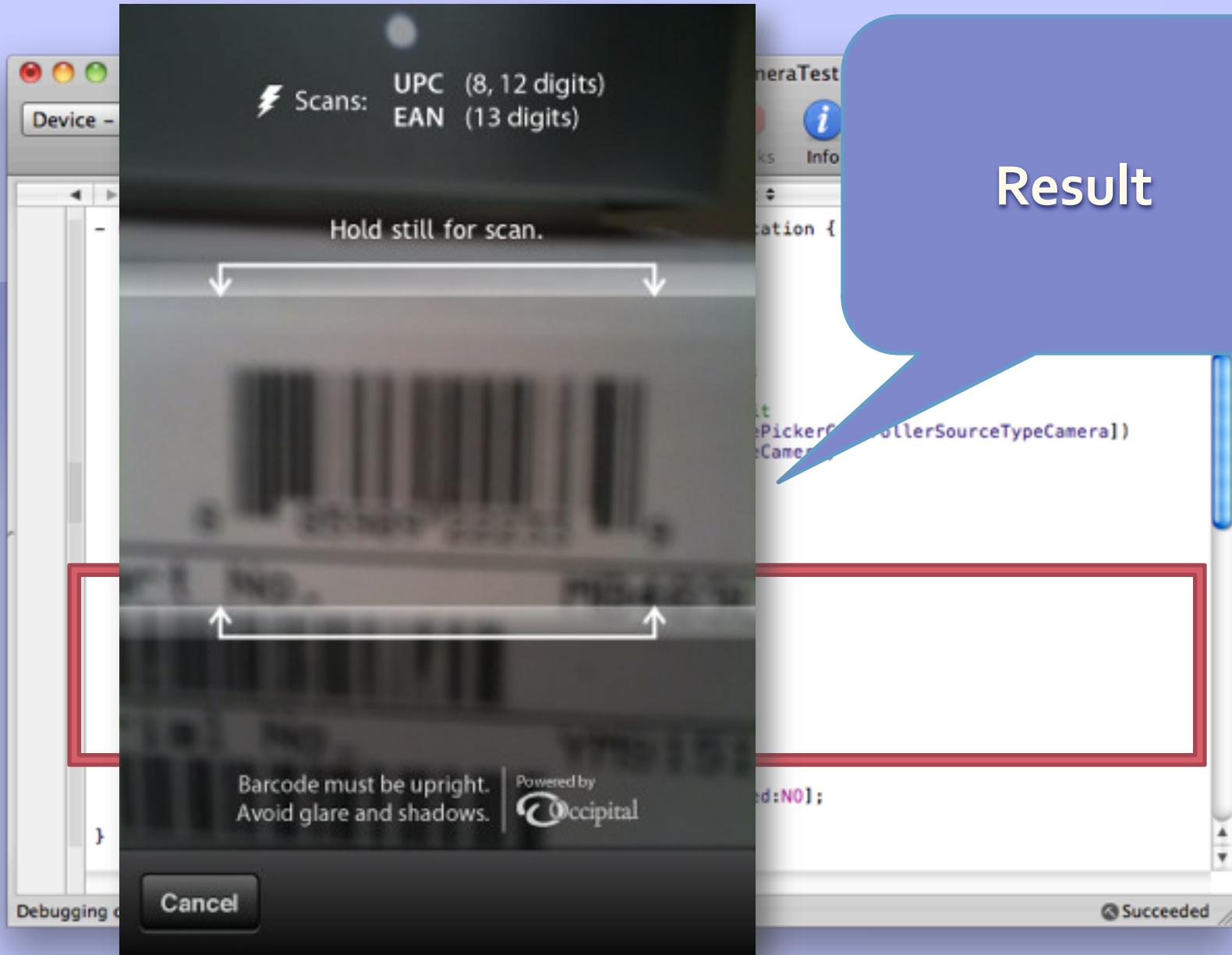
    // Show the camera!
    [viewController presentModalViewController:picker animated:NO];
}
```

Debugging of "CameraTest" ended normally. Succeeded

» Advanced: ([Custom overlays](#)) and image processing.



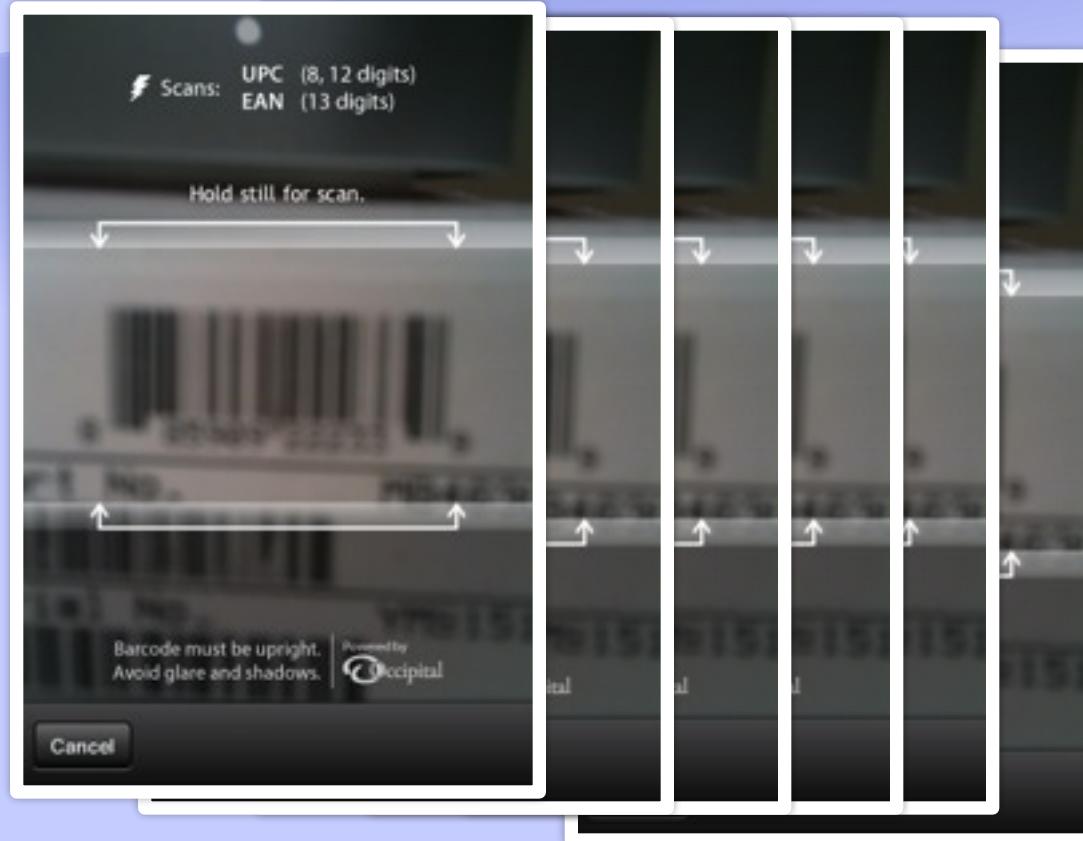
» Advanced: (Custom overlays) and image processing.



» Advanced: (Custom overlays) and image processing.

# Reading blurry barcodes.

Second ingredient: Rapid screenshots.



Currently requires a function that's not published.

» Advanced: Custom overlays and image processing.

# Reading blurry barcodes.

Last ingredient: Image processing (Computer vision)



» Advanced: Custom overlays and (image processing).

# Reading blurry barcodes.

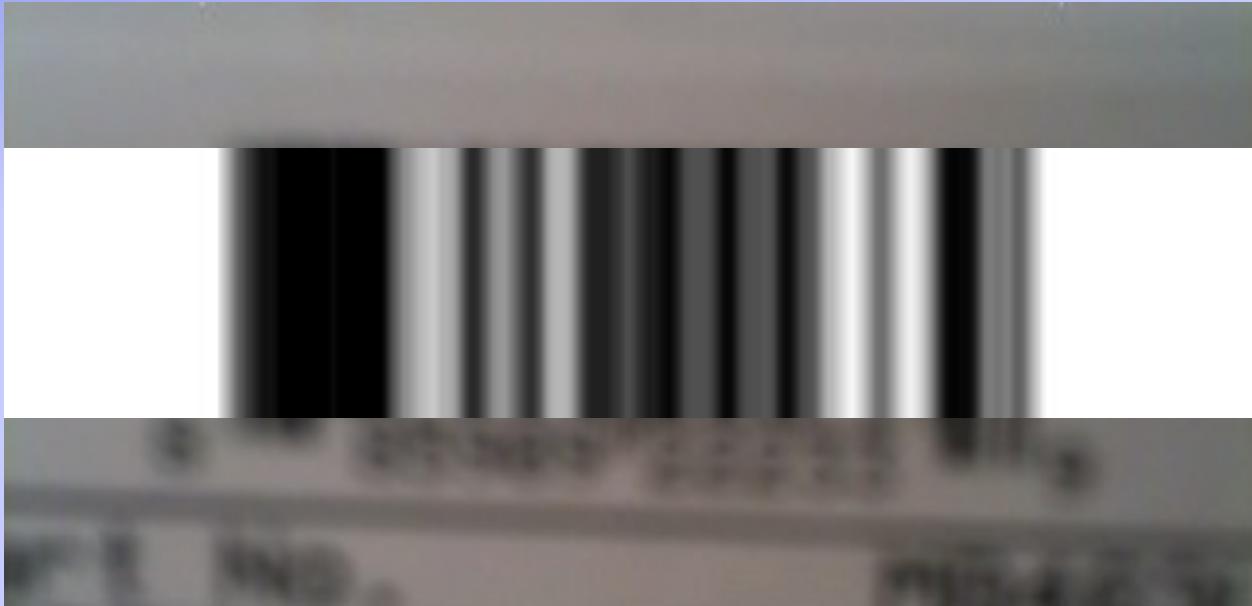
Last ingredient: Image processing (Computer vision)



» Advanced: Custom overlays and (image processing).

# Reading blurry barcodes.

Last ingredient: Image processing (Computer vision)



» Advanced: Custom overlays and (image processing).

# Reading blurry barcodes.

Last ingredient: Image processing (Computer vision)



» Advanced: Custom overlays and (image processing).

# Reading blurry barcodes.

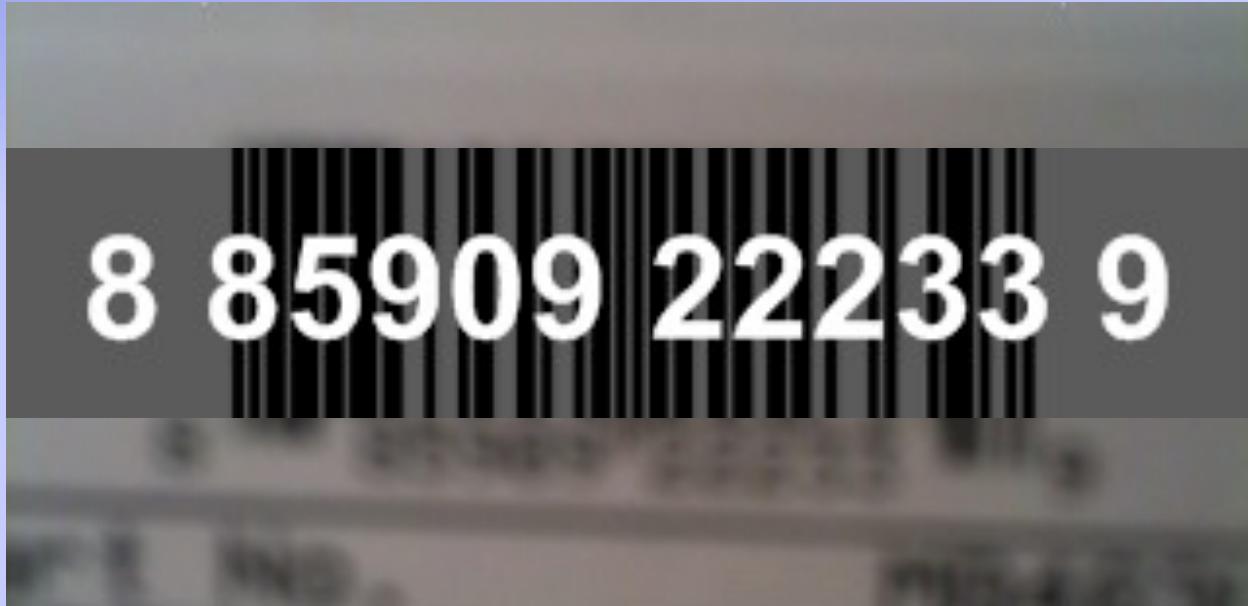
Last ingredient: Image processing (Computer vision)



» Advanced: Custom overlays and (image processing).

# Reading blurry barcodes.

Last ingredient: Image processing (Computer vision)



**Let's take a look at how to access raw pixels.**

» Advanced: Custom overlays and (image processing).

The screenshot shows the Xcode IDE interface. The title bar reads "CameraTestAppDelegate.m - CameraTest". The top menu bar includes "File", "Edit", "Project", "Product", "Run", "Stop", "Breakpoints", "Build and Run", "Tasks", "Info", "Search", and "Help". A toolbar below the menu bar contains icons for Device selection ("Device - 3.1.2 | Debug"), Action, Breakpoints, Build and Run, Tasks, Info, and String Matching.

The main editor area displays the following code:

```
// Called when image has been captured.  
- (void) imagePickerController:(UIImagePickerController *)picker  
    didFinishPickingMediaWithInfo:(NSDictionary *)info  
{  
    // Retrieve the captured image and save to the camera roll.  
    UIImage * image = [info objectForKey:UIImagePickerControllerOriginalImage];  
    UIImageWriteToSavedPhotosAlbum(image, nil, nil, nil);  
}
```

The status bar at the bottom left says "Debugging terminated." and the status bar at the bottom right says "Succeeded".

» Advanced: Custom overlays and (image processing).

The screenshot shows the Xcode IDE interface. The title bar reads "CameraTestAppDelegate.m - CameraTest". The top menu bar includes "File", "Edit", "Project", "Product", "Run", "Stop", "Breakpoints", "Build and Run", "Tasks", "Info", "Search", and "Help". A toolbar below the menu bar contains icons for Device (3.1.2), Debug, Action, Breakpoints, Build and Run, Tasks, Info, and String Matching. The main editor area displays the following code:

```
// Called when image has been captured.  
- (void) imagePickerController:(UIImagePickerController *)picker  
    didFinishPickingMediaWithInfo:(NSDictionary *)info  
{  
    // Retrieve the captured image and save to the camera roll.  
    UIImage * image = [info objectForKey:UIImagePickerControllerOriginalImage];  
    //UIImageWriteToSavedPhotosAlbum(image, nil, nil, nil);  
}
```

The status bar at the bottom left says "Debugging terminated." and the bottom right says "Succeeded".

» Advanced: Custom overlays and (image processing).

The screenshot shows the Xcode IDE interface with the following details:

- Title Bar:** CameraTestAppDelegate.m - CameraTest
- Toolbar:** Device - 3.1.2 | Debug, Action, Breakpoints, Build and Run, Tasks, Info, String Matching.
- Code Editor:** The code is for `CameraTestAppDelegate.m`. The current line is `info objectForKey:UIImagePickerControllerOriginalImage];`. The code handles image capture and processing:

```
// Called when image has been captured.  
- (void) imagePickerController:(UIImagePickerController *)picker  
    didFinishPickingMediaWithInfo:(NSDictionary *)info  
{  
    // Retrieve the captured image and save to the camera roll.  
    UIImage * image = [info objectForKey:UIImagePickerControllerOriginalImage];  
    //UIImageWriteToSavedPhotosAlbum(image, nil, nil, nil);  
  
    // Use the CoreGraphics Data Provider to tap into image pixels  
    CGDataProviderRef dataProvider = CGImageGetDataProvider([image CGImage]);  
    CFDataRef data = CGDataProviderCopyData(dataProvider);  
    const UInt8 * bytes = CFDataGetBytePtr(data);  
  
    // Now we can access bytes directly to read pixel values.  
    // TODO: Run some computer vision algorithms here.  
  
    NSLog(@"The first image pixel is %x %x %x %x",  
          bytes[0],  
          bytes[1],  
          bytes[2],  
          bytes[3]);  
  
    // Clean up  
    CFRelease(data);  
    CGDataProviderRelease(dataProvider);  
}
```

**Status Bar:** Debugging terminated. Succeeded

» Advanced: Custom overlays and (image processing).

CameraTestAppDelegate.m - CameraTest

Device - 3.1.2 | Debug    Action    Breakpoints    Build and Run    Tasks    Info    String Matching

Overview    Search

CameraTestAppDelegate.m:59 -imagePickerController:didFinishPickingMediaWithInfo: +

```
// Called when image has been captured.
```

{

    Device - 3.1.2 | Debug    Overview    Breakpoints    Build and Run    Tasks    Restart    Pause    Clear Log

    ttys009  
    Loading program into debugger...  
    Program loaded.  
    target remote-mobile /tmp/.XcodeGDBRemote-66645-94  
    Switching to remote-macosx protocol  
    mem 0x1000 0x3fffffff cache  
    mem 0x40000000 0xffffffff none  
    mem 0x00000000 0x0fff none  
    run  
    Running...  
    [Switching to thread 11779]  
    [Switching to thread 11779]  
    sharedlibrary apply load rules all  
    (gdb) continue  
    2009-12-02 23:52:11.397 CameraTest[1353:207] The first image pixel is 6 d a ff

GDB: Running...    Succeeded

    CGDataProviderRelease(dataProvider);

}

Debugging terminated.    Succeeded

» Advanced: Custom overlays and (image processing).

Part 4/4

# Future of iPhone Computer Vision

# *What's not possible today?*

- ◆ Vision-based augmented reality
  - ◆ Marker-based
  - ◆ Marker-less



[morethantechnical.com](http://morethantechnical.com)

» Future

# *What's not possible today?*



Paramount Pictures

» Future

# What's *not* possible today?



Orion Pictures

Terminator Vision: 1984

» Future

Mobile computer vision will eventually become a part of most applications we already use today.

# Q&A

P.S. Check us out in Newsweek this week! (page 17)



Jeffrey Powers, Co-Founder, Occipital