```
// Create the Map View

 _mapView = [[MKMapView alloc] initWithFrame:self.view.bounds];
 _mapView.showsUserLocation = YES;
 _mapView.delegate = self;
 [self.view addSubview:_mapView];

// Once you have the current location, define the map region you want to be
    visible:

 MKCoordinateRegion viewRegion =  MKCoordinateRegionMakeWithDistance(self.
     location.coordinate, REGION_SIZE, REGION_SIZE);
 MKCoordinateRegion adjustedRegion = [_mapView regionThatFits:viewRegion];
 [_mapView setRegion:adjustedRegion animated:NO];

// Also request Google Directions API to retrieve the route:

 AFHTTPClient *_httpClient = [AFHTTPClient clientWithBaseURL:[NSURL
     URLWithString:@"http://maps.googleapis.com/"]];
 [_httpClient registerHTTPOperationClass: [AFJSONRequestOperation class]];

 NSMutableDictionary *parameters = [[NSMutableDictionary alloc] init];
 [parameters setObject:[NSString stringWithFormat:@"%f,%f",
      location.coordinate.latitude, location.coordinate.longitude]
             forKey:@"origin"];
 [parameters setObject:[NSString stringWithFormat:@"%f,%f",
      endLocation.coordinate.latitude, endLocation.coordinate.longitude]
             forKey:@"destination"];
 [parameters setObject:@"true" forKey:@"sensor"];

 NSMutableURLRequest *request = [_httpClient requestWithMethod:@"GET" path:
     @"maps/api/directions/json" parameters:parameters];
 request.cachePolicy = NSURLRequestReloadIgnoringLocalCacheData;

 AFHTTPRequestOperation *operation = [_httpClient
     HTTPRequestOperationWithRequest:request
      success:^(AFHTTPRequestOperation *operation, id response) {
        NSInteger statusCode = operation.response.statusCode;
            if (statusCode == 200) {
                [self parseResponse:response];
            } else {
            }
        } failure:^(AFHTTPRequestOperation *operation, NSError *error) { }];

[_httpClient enqueueHTTPRequestOperation:operation];

// Get what you need:

 - (void)parseResponse:(NSDictionary *)response {
    NSArray *routes = [response objectForKey:@"routes"];
    NSDictionary *route = [routes lastObject];
    if (route) {
        NSString *overviewPolyline = [[route objectForKey:
            @"overview_polyline"] objectForKey:@"points"];

        _path = [self decodePolyLine:overviewPolyline];
```

```
    }
}

// And use the code provided by Ankit Srivastava:

-(NSMutableArray *)decodePolyLine:(NSString *)encodedStr {
    NSMutableString *encoded = [[NSMutableString alloc]
           initWithCapacity:[encodedStr length]];
    [encoded appendString:encodedStr];
    [encoded replaceOccurrencesOfString:@"\\\\" withString:@"\\"
                                options:NSLiteralSearch
                                  range:NSMakeRange(0,
                                        [encoded length])];
    NSInteger len = [encoded length];
    NSInteger index = 0;
    NSMutableArray *array = [[NSMutableArray alloc] init];
    NSInteger lat=0;
    NSInteger lng=0;
    while (index < len) {
        NSInteger b;
        NSInteger shift = 0;
        NSInteger result = 0;
        do {
            b = [encoded characterAtIndex:index++] - 63;
            result |= (b & 0x1f) << shift;
            shift += 5;
        } while (b >= 0x20);
        NSInteger dlat = ((result & 1) ? ~(result >> 1)
              : (result >> 1));
        lat += dlat;
        shift = 0;
        result = 0;
        do {
            b = [encoded characterAtIndex:index++] - 63;
            result |= (b & 0x1f) << shift;
            shift += 5;
        } while (b >= 0x20);
        NSInteger dlng = ((result & 1) ? ~(result >> 1)
               : (result >> 1));
        lng += dlng;
        NSNumber *latitude = [[NSNumber alloc] initWithFloat:lat * 1e-5];
        NSNumber *longitude = [[NSNumber alloc] initWithFloat:lng * 1e-5];

        CLLocation *location = [[CLLocation alloc] initWithLatitude:
             [latitude floatValue] longitude:[longitude floatValue]];
        [array addObject:location];
    }

    return array;
}

// Create the MKPolyline annotation:

NSInteger numberOfSteps = _path.count;
CLLocationCoordinate2D coordinates[numberOfSteps];
for (NSInteger index = 0; index < numberOfSteps; index++) {
```

```
    CLLocation *location = [_path objectAtIndex:index];
    CLLocationCoordinate2D coordinate = location.coordinate;

    coordinates[index] = coordinate;
}

MKPolyline *polyLine = [MKPolyline polylineWithCoordinates:coordinates
    count:numberOfSteps];
[_mapView addOverlay:polyLine];

// And make it visible on the map view

 - (MKOverlayView *)mapView:(MKMapView *)mapView viewForOverlay:(id)overlay
     {
    MKPolylineView *polylineView = [[MKPolylineView alloc] initWithPolyline:
        overlay];
    polylineView.strokeColor = [UIColor redColor];
    polylineView.lineWidth = 1.0;

    return polylineView;
}
```