情報理工学特別研究報告書

題目

目的地までの経路を実写画像に正確に重畳表示する方法についての研究

学生証番号　853545

氏名　辻 勇太

提出日　令和 3 年 1 月 28 日

指導教員　蚊野　浩

京都産業大学
情報理工学部

要約

本論文では，Apple の Mapkit を利用して現在地から目的地までの経路を探索し，その経路情報を iPad で撮影する周囲のカメラ映像に重畳して表示する技術について研究した．そのためには，現在地から目的地までの経路を，端末を中心とする３次元座標系における３次元的なモデルとして生成し，それをカメラで撮影した周囲の映像と位置が重なるように表示する必要がある．

カメラで撮影した周囲の映像と,現在地と目的地の経路の位置が重なるように表示するために,現在地と目的地の緯度・経度・標高,現在地と目的地の間の経路情報,端末の姿勢を用いる.取得したこれらの値から,現在地と目的地の間の距離と仰俯角・方位角を求め,これらを用いて端末を中心とする 3 次元座標系に極座標形式の座標を求める.そして，極座標値を直交座標に変換した.また,端末の全ての回転に対応する必要があるが,この時点では yaw 角の回転に対応できていないため,直交座標に yaw 角の回転行列をかけることで補正を行った.その後,yaw 角の補正をした座標を使用して 3DCG で作成した経路をカメラ映像上に表示させた.

その結果,現在地から目的地の経路がカメラ映像の道路に重なっているように表示することができた.しかし,経路が正しい位置に表示されない場合があった.その原因として yaw 角の回転の補正が不十分であること,位置情報の精度があまり高くないことが挙げられる.yaw 角の回転の補正は,正しい yaw 角の回転角度を取得し,回転行列を正しく扱うことで解決することができると考える.位置情報については,取得する精度を上昇させることはできないため,ローパスフィルターなどのフィルターを加えることで問題を解決することができると考える.

目次

1章　序論

AR(Augmented Reality)は,「拡張現実」「強化現実」「拡張現実感」と訳される通り,現実世界をカメラで撮影した映像に,コンピュータで生成した情報を付加することで,現実を拡張・強化する技術である.カメラから取り入れた映像にGPSの位置情報データやCGの映像を重ねて表示することで,実際には存在しないものを実在するかのように見せることが可能である.本研究ではこの技術を使った3DCGの表示をAR表示と呼ぶ.

AR技術はエンターテイメント分野でゲームやイベントでの利用をはじめ,生活補助機能としての活用,産業やビジネスでの活用など多くの場所で活躍をしている.

ARを活用したゲームアプリ「Pokémon GO」を筆頭として,多くのアプリが登場している.Pokémon GOは端末のカメラで撮影されたリアルタイム映像に,3DCGのキャラクターの情報を重畳させることで,あたかも現実にキャラクターが存在しているように見えるアプリである.カメラ映像から得た情報を元に平面を検出することで,キャラクターが地面に着地しているように表示されたり,空を飛ぶキャラクターが空中に表示されたりと,現実に存在しているように表示させることが可能である(図1.1).

図1.1 Pokémon GOのAR表示の例

生活をサポートするARアプリとして,iOS12以上のiPhoneやiPadにインストールされている計測アプリ「ARMeasure」や,化粧品ブランドが提供しているバーチャルメイク,yahooやgoogleが提供しているARナビなどが挙げられる.いずれのアプリもカメラで撮影した映像からの情報や,デバイスに搭載されているセンサからの情報を組み合わせることで,ユーザにAR体験を提供している.

本論文では,Apple 社が提供している iOS 向けフレームワークである ARkit を使い,カメラ映像上に, 目的地までの経路を重ねて表示する AR ナビの技術の研究について説明する.現在地と目的地の３次元的な位置情報をカメラ映像上の座標に変換することで,映像上の道路部分に沿って経路情報を３次元 CG 表示することを目標とした.以下 2 章では AR と ARkit,類似の AR ナビアプリ,プログラム開発に使用した iOS の機能について述べる.3 章では ARkit を用いて経路表示を行う方法について説明する.4 章ではプログラムの検証結果と課題点を示し,5 章で結論を述べる.

2 章　AR ナビに関する従来技術と本研究で用いた技術

2.1　Augmented Reality と ARkit

AR(Augmented Reality)とは,仮想空間の情報やコンテンツを現実世界に重ね合わせて表示することなどにより,現実を拡張する技術や仕組みである.いくつかの企業は,スマートフォンなどで AR を実現するためにフレームワークを提供している.ここでフレームワークとは,関連する API やクラスライブラリをまとめたものの総称である.代表的なものとして,Apple 社が提供している iOS 向け AR フレームワークである ARkit, Google 社が提供している Android 向け AR フレームワークである ARCore, MagicLeap 社が提供している MagicLeap Toolkit などがある.今回の研究では iPad を用いた AR の研究を行うため,Apple 社が提供しているフレームワークである ARkit を用いた.

ARkit は,iPhone や iPad などのカメラからのライブビューに 2D や 3D のアイテムを追加して,そのアイテムがあたかも現実の世界に存在するかのように見せるためのフレームワークである.特殊なハードウェアを使わずに,iOS 端末のモーショントラッキングと,カメラ映像の取得・画像処理を元に空間認識を行い,AR 体験を実現している[1].図 2.1 は ARkit を使った AR アプリケーションの動作例である.現実世界には玩具の建物しか存在していないが,iPad の画面上では建物の周りに植物や人,動物などが違和感なく描画されていることがわかる.

図 2.1 ARkit の動作例
(https://www.apple.com/jp/newsroom/2018/06/apple-unveils-arkit-2/)

ARkit を利用することで,平面検出・顔認識・モーションキャプチャ・現実世界でのスケール検出・光源推定など様々な機能を実装することが可能である.これを実現させるために,ARkit では複数のモードが用意されている.今回の実験では,カメラ映像上で認識した物体の情報と,端末の位置と姿勢を組み合わせることで 3D オブジェクトを操

作することができる ARWorldTrackingConfiguration というモードを使用した.

2.2　既存の AR ナビアプリ

AR ナビアプリは,地図で取得した情報をもとに,現実世界のリアルタイム映像に経路情報を描画することで,目的地への経路をわかりやすく表示するアプリである.iOS 端末向けの AR ナビアプリに表 1 のものがあった.これらを使用して長所と短所をまとめたものを表 2 に示す.これらの使用経験から,経路を表示する際に経路の標高と，端末の向き・姿勢（端末の回転）について考慮して表示することが必要であることがわかった.

表 2.1　代表的な AR ナビアプリ

 @Balloon	 Yahoo!MAP
 GoogleMap	 @PinnAR

表 2.2 AR ナビアプリの長所と短所

	@Balloon	Yahoo! MAP	GoogleMap	PinnAR
長所	・標高が考慮されている. ・回転に対応している.	・道路に沿って経路表示されている. ・回転に対応している.	・標高が考慮されている. ・回転に対応している.	・道路に沿って経路表示されている.
短所	・目的地しか表示されずわかりずらい.	・標高が考慮されていない.	・曲がり角のみ表示される.	・回転に対応できていない. ・標高が考慮されていない.
問題例	・経路が表示されないため,わかりずらい.	・坂道では経路が正確に表示されない.	・経路が表示されないため,わかりずらい.	・端末の姿勢を変えると正確に表示されない. ・坂道では経路が正確に表示されない.

2.3　開発に用いた iOS の機能

iOS には,アプリケーションプログラム開発を支援するための API として多数のフレームワークが用意されている.本研究では，CoreLocation，Mapkit，CoreMotion，ARkit の，4 つのフレームワークを使用した.

CoreLocation は，GPS（衛星測位システム）や Wi-Fi，携帯電話通信網やデジタルコンパスなどを組み合わせることで,端末の位置情報を取得することができる.端末の位置情報として緯度・経度や標高・方角を取得可能である.

Mapkit は地図の表示を行い,地図にピンを立てることや,場所の検索,地図に情報を追加することなどが可能である.また,指定した 2 地点の間の経路を探索することができる.

CoreMotion は，端末のセンサを用いて，端末の姿勢・角速度・重力加速度・加速度・高度の値を取得することができる.iOS 端末には図 2.3 のように端末を中心とした 3 次元座標系が存在する.iOS 端末を縦向きで使用する場合,長手方向上向きが Y 軸,短手方向右向きが X 軸,ディスプレイの裏から表に垂直に貫く軸が Z 軸である.また,姿勢を表す回転として,X 軸,Y 軸,Z 軸を中心とした左回転の回転角がそれぞれ pitch 角,roll 角,yaw 角に対応する.

ARkit は iOS 端末のカメラやセンサを用いることで AR 体験を実現することが可能である.これら 4 つのフレームワークは,本研究を行うにあたり基盤となる機能である.

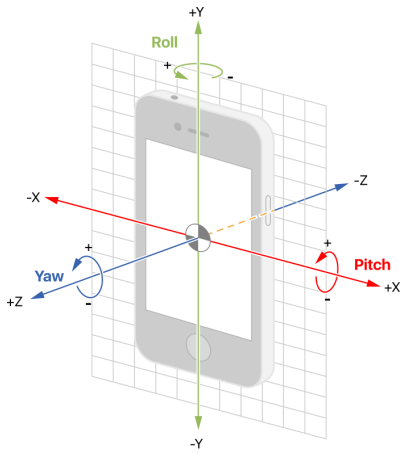


図 2.2　iOS 端末中心の 3 次元座標系と回転角の関係

3 章　ARkit を用いた経路表示

3.1 地図画面で経路を探索する方法

現在地から目的地までの経路を AR 表示するためには,2 地点間の経路情報を取得しなければならない.ここでは,端末位置の取得と,目的地を決定する方法,そして, ２地点間の経路情報の取得について説明する.

3.1.1 位置情報の取得

CoreLocation を使用することで, 端末の位置情報として緯度・経度・標高・方角を取得することができる.これらは,取得する位置情報の種類を宣言し,ユーザが認証することで値を取得することがでる. 緯度と経度・方角の単位は「度」,標高の単位は「メートル」で取得している.CoreLocation で取得することのできる方角は,端末の pitch 回転に関わらず,端末の背面が向いている方向を確認することができる.方角の数値は右回りに 0～359 の値をとり,0 は北を示す.取得できる方角の基準には,磁北と真北の 2 種類が存在する.磁北は方位磁石が北を示す方角であり,真北は GPS の値を用いることで地図上での北を示す方角である.図 3.1 は真北を 0 になるように端末を回転させ,その状態のまま真北と磁北を確認した図であり,真北と磁北では多少の差があることが確認できる.

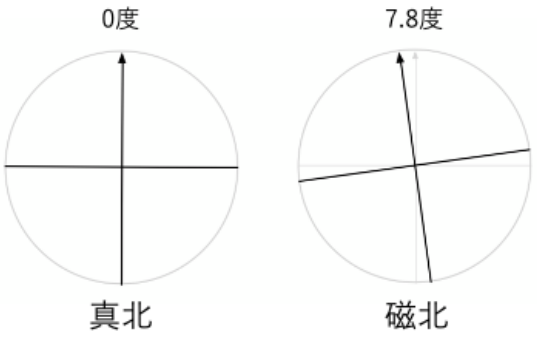


図 3.1 真北と磁北の差

CoreLocation にはアプリケーションが必要な精度に応じて,GPS の取得条件を変更することができる機能がある.精度の指定には,追加のセンサを使用した上での最高精度,追加のセンサを使用しないでの最高精度,10m・100m・1km・3km 以内の精度の 6 段階がある.これらの精度は,精度が上がる毎に真の現在地との誤差は少なくなるが,値を取得するための時間と消費電力が大きくなる.今回は,高い精度の値を取得したいため,追加のセンサを使用した上での最高精度を取得するように設定した.

3.1.2　位置情報の精度の調査

3.1.1 で設定した精度の誤差と,位置情報の取得にかかった時間を確認するための調査を次のような流れで行った.

① 端末位置の緯度・経度・標高を取得し,取得するためにかかった時間を記録する.

② ①を 100 回繰り返し,緯度,経度,標高,時間の平均と標準偏差を計算する.

③ 端末位置の緯度・経度・標高の真値を地図で確認する.

④ ②で計算した平均と③で確認した真値の差を計算する.

⑤ ②で求めた緯度と経度の標準偏差と,④で求めた緯度の差と経度の差の単位は度である.北緯 35 度では緯度 1 度あたり 110,940.58m,経度 1 度あたり 91,287.78(m)であるから,これを使って, 数値を m 単位に変換する[2].

本研究で作成したプログラムは基本的に屋外で使用するため,調査回数は屋外で 2 回,屋内で 1 回とした.

この手順で行った調査の結果を表 3.1 に示す.屋外と屋内の計 3 回の値について比較した結果,屋外 1 回目は値のばらつきが大きいが, 平均と真値の誤差は約 8.0m であり,位置情報を取得する時間の平均は 3 回の中で 1 番短かった.屋外 2 回目は値のばらつきが小さく,位置情報を取得する時間の平均も短いが,平均と真値の誤差は約 18.4m と大きかった.屋内は位置情報の取得時間は多少かかるが,値のばらつきが小さく,平均と真値の誤差は約 4.9m と 3 回の中で 1 番小さかった.

また,3 回全ての平均から,位置情報を取得するには約 14.8 秒必要とし,値のばらつきは緯度:2.12m,経度:8.07m,標高:0.47m 程度,誤差の平均は緯度:9.26m,経度:7.89m,標高:6.74m 程度ということが確認できた.

表 3.1 位置情報の精度に関する調査結果

	屋外1回目	屋外2回目	屋内	3回の平均
緯度の平均(度)	35.06951	35.06957	35.06920	/
経度の平均(度)	135.75694	135.75710	135.75559	/
標高の平均(m)	130.72	131.49	120.60	/
時間の平均(秒)	14.33	14.95	15.08	14.78
緯度の標準偏差(m)	4.695	0.927	0.742	2.121
経度の標準偏差(m)	21.275	1.604	1.315	8.065
標高の標準偏差(m)	1.364	0.049	0.002	0.472
時間の標準偏差(秒)	4.99	0.71	0.09	1.93
緯度の真値との差(m)	7.966	14.927	4.897	9.264
経度の真値との差(m)	3.633	18.428	1.617	7.893
標高の真値との差(m)	7.321	8.091	4.808	6.740

3.1.3 経路情報の取得

経路情報とは,2 地点間の経路について,使用する道路の情報,曲がり角の情報,時間や距離の情報をまとめたものである.経路情報の取得に必要なものは, 経路の始点(端末の現在地)の緯度・経度と,経路の終点(指定した目的地)の緯度・経度である.この 2 つの座標を指定し,Apple のサーバに送信することで経路情報を取得することができる.現在地の緯度・経度の取得方法は 3.1.1 節の通りである.また,Mapkit を用いて画面上に地図を表示し,目的地を指定することで目的地の緯度・経度を取得することができる.今回のプログラムでは,地図上を長押しする方法と,検索窓で目的地を検索する方法の 2 通りで目的地を指定し,端末の現在地の緯度・経度を組み合わせることで経路情報を取得できるようにした.取得した経路情報を用いて現在地から目的地までの経路を折れ線表示したものが図 3.2 である.

AR 表示を行った際に道路に沿って経路を描画するためには,図中の円で囲まれた経路の湾曲点(図 3.2 右)の緯度・経度・標高が必要である.経路情報の中には,経路の湾曲点の緯度・経度の値が含まれているが標高の値は含まれていない.また,Mapkit を含めた swift の機能に,経路上の標高を取得する機能や,緯度・経度から標高を求める機能は存在していなかった.そのため, 国土地理院が公開している「場所情報コード閲覧システム」の API 使用することで,指定座標における標高を取得した[3].場所情報コー

ド閲覧システムでは，ジオコーディングや逆ジオコーディング，緯度・経度を指定することで標高を取得することなどが可能である．

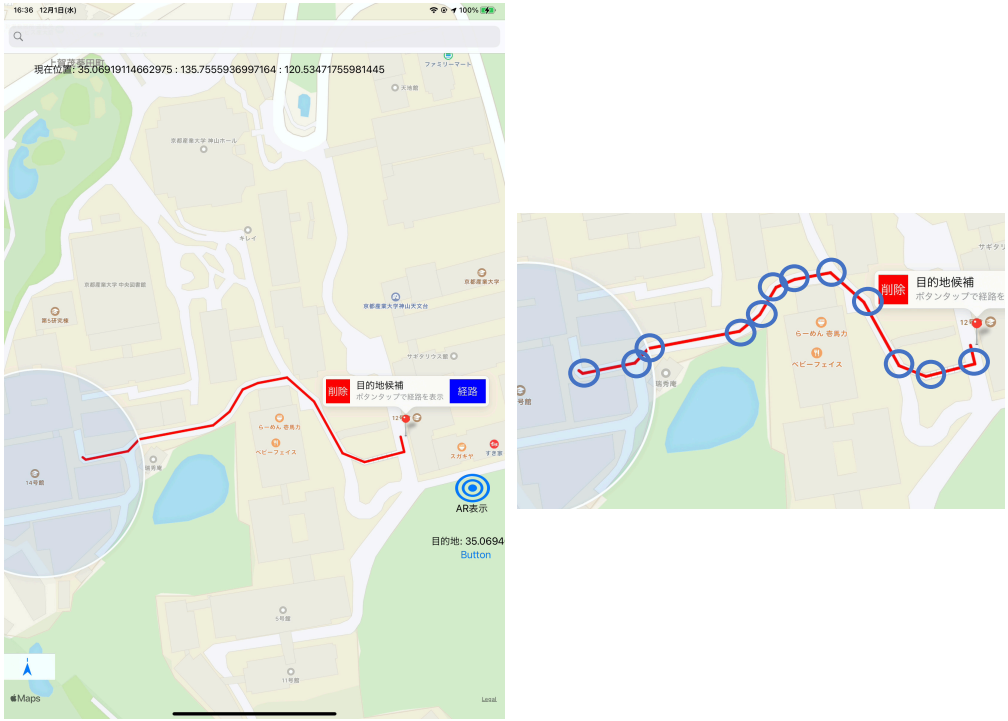


図 3.2 地図上の現在地から目的地までの経路表示例

3.2 現実世界の座標から端末中心の 3D 座標への変換

3.2 で述べた方法で，現在地と目的地，経路上の湾曲点におけるそれぞれの緯度・経度・標高を取得することができた．これらの値に加えて後述する端末の姿勢の値を使用することで現実世界の緯度・経度・標高の座標から AR 画面上の 3 次元座標に変換した．

端末の現在地から見た目的地の座標を計算する方法は，端末の姿勢を用いて 2 つの座標軸の姿勢を合わせる方法と，現在地と目的地の距離と方向を用いて極座標系で目的地の座標を表現する方法の 2 通りの方法がある．3.2 節ではこの 2 通りの方法について説明する．現実世界での座標を(緯度，経度，標高)とし，現実世界での現在地と目的地の座標をそれぞれ(cLa, cLo, cEl)，(tLa, tLo, tEl)で表す．また，現在地と目的地の差をそれぞれ，$\Delta La = cLa - tLa,\quad \Delta Lo = cLo - tLo,\quad \Delta Ele = cEl - tEl$ とし，ΔLa と ΔLo についてはメートル換算したものを用いる．

今回の研究では，2 つの方法について，AR 画面上で指定位置に 3DCG を表示するプログラムを作成し，比較を行った．その結果，「現在地と目的地の距離と方向を用いて極座標

系で目的地の座標を表現する方法」の方が正確な位置に 3DCG を配置することができたため，この方法で AR 画面上に経路表示を描画することにした．

3.2.1 2 つの座標系を用いた座標計算

iOS 端末の 3 次元座標と pitch，yaw，roll 回転は図 2.3 のように定義されている．この 3 次元座標を端末 3D 座標系とし，軸を表す文字を小文字(x，y，z)とした．また，現実世界の端末位置を原点とする座標系を定義する．この座標系は図 3.3 に示すように，現実世界での西から東方向を X 軸，真上方向を Y 軸，北から南方向を Z 軸とした．この 3 次元座標を地理 3D 座標とし，軸を表す文字を大文字(X，Y，Z)とした．

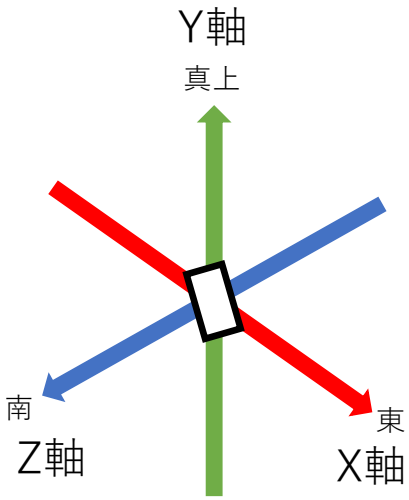


図 3.3　端末位置を原点とする地理 3D 座標系

端末の AR 画面に 3DCG を描画する場合，端末 3D 座標系で記述した座標の値を使用しなければならない．一方，目的地の座標については地理 3D 座標の値として得るため，地理 3D 座標から端末 3D 座標に変換する必要がある．これは 2 つの座標系の姿勢を合わせることで変換が可能となり，変換には端末の pitch，yaw，roll の 3 つの角度を用いる．

2 つの座標の姿勢は，次のような流れで合わせることができる．

① 端末に pitch 回転を適用し，端末の画面を地面と垂直に立てる．これにより，端末は地面と垂直になるが，端末の長手方向は傾いている可能性がある．

② 端末に yaw 回転を適用し，端末の長手方向を真上に向けるこれにより，y 軸と Y 軸が重なる．

③ 端末に roll 回転を適用し，端末の背面を北に向ける．これで x 軸と X 軸が重なり，z 軸と Z 軸が重なる．

pitch 回転による端末 3D 座標系の回転行列を R_{pitch}，yaw 回転による端末 3D 座標系の回転行列を R_{yaw}, roll 回転による端末 3D 座標系の回転行列を R_{roll} とする. ①～③の合成変換は $R = R_{pitch} \times R_{yaw} \times R_{roll}$ である. よって, 地理 3D 座標系に回転行列をかけて修正した(X',Y',Z') = (X,Y,Z)R = 端末 3D 座標系(x,y,z)となり,座標を変換することができる.

以上のように端末 3D 座標系と地理 3D 座標系を合わせることで, 地理 3D 座標系の座標を端末 3D 座標系の座標に記述すると, AR 画面上で正しい位置に 3DCG が表示される.

3.2.2 極座標系を用いた座標計算

3 次元座標系の極座標は原点からの距離と, 2 つの角度からなる. ここで現在地を原点として, 現在地と目的地の距離を L, 2 つの角度を θ と ϕ とした. 図 3.4 は端末の現在地から見た目的地の座標を極座標を用いて表現した図である. 後で説明するように，極座標を用いることで, 地理 3D 座標と端末 3D 座標を関係づけることができる.

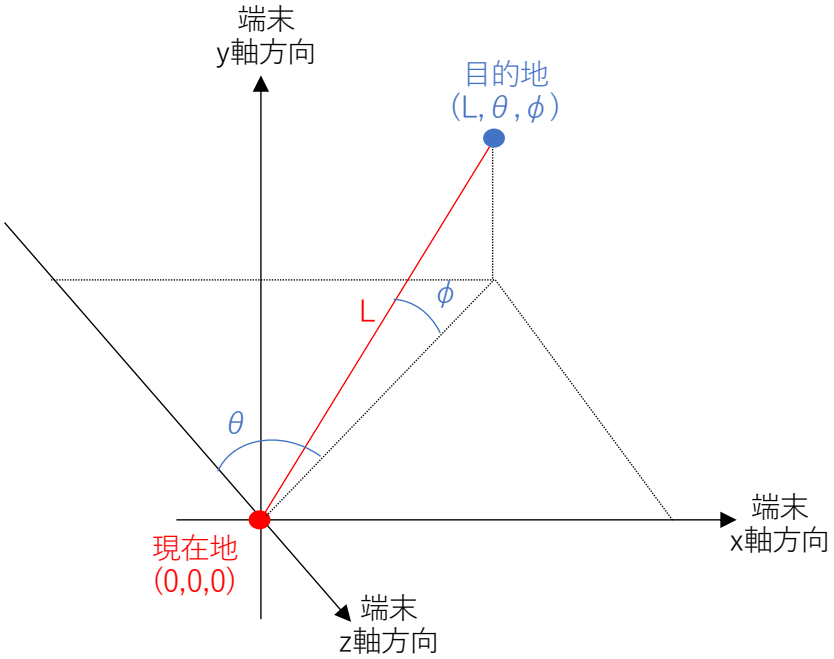


図 3.4 現在地から見た目的地を極座標系で表現した様子

現在地から目的地までの, 標高を考慮しない距離を l とすると，$l = \sqrt{\Delta La^2 + \Delta Lo^2}$ である. そのため, 標高を考慮した距離 l' は, $l' = \sqrt{\Delta El^2 + l^2}$ になる(図 3.5). 今回の研究では標高を考慮した座標に 3DCG を配置するため, L の値は l' とした.

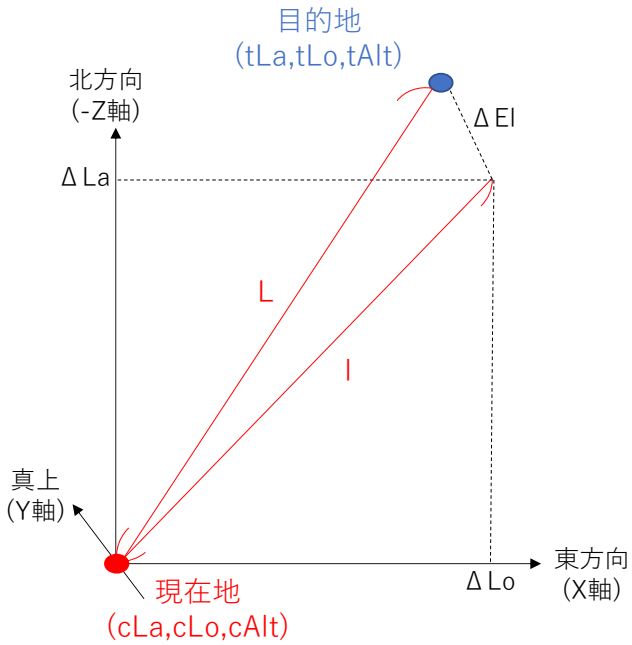


図 3.5 目的地の地理 3D 座標

2 つの角度のうち, θ は現在地から目的地までの水平方向の角度であり, ϕ は垂直方向の角度である. θ は図 3.6 右のように, 北と目的地の角度(方位角)から端末の roll 角の値を引くことで計算することができる. 北と目的地の角度(方位角)は次のように求める.

$$x = \sin(\Delta Lo)$$
$$y = \cos(cLa) \times \tan(tLo) - \sin(cLa) \times \cos(\Delta Lo)$$
$$\text{方位角} = atan2(x, y)$$

ϕ は図 3.6 左のように, 端末の pitch 角と, 目的地と現在地の上下方向の角度(仰俯角)を用いて計算することができる. 仰俯角は, 仰俯角 $= \arctan(\Delta \text{El} - l)$ で求めることができ, ϕ は, $\phi =$ 仰俯角 $+ \frac{\pi}{2} - pitch$ で求めることができる.

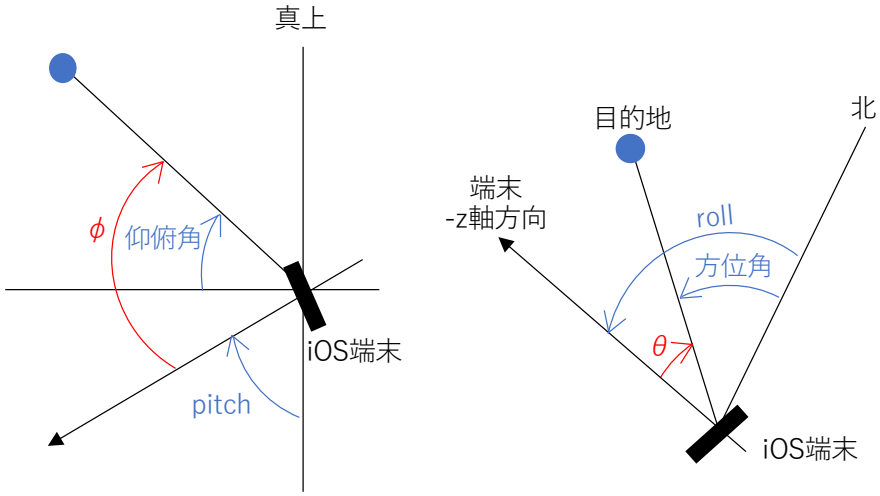


図 3.6 目的地の位置を極座標で表現した場合の θ，ϕ

以上より，現在地が原点の極座標系で表した目的地の座標(L，θ，ϕ)を求めることができた．これを端末 3D 座標系の直交座標に変換することで，AR 画面上に表示する 3DCG の座標を求めることができる．極座標から直交座標への計算は次の通りである．

$z = -(L \times sin\theta \times cos\phi)$

$x = L \times sin\theta \times sin\phi$

$y = -(L \times cos\theta)$

この極座標を用いて座標を計算する方法には，yaw 回転の補正が含まれていない．そのため，極座標から変換した直交座標に yaw 回転の回転行列をかけることで補正を行った．

3.3 プログラムの実装

3.3.1 iPad の姿勢の取得

プログラムに実装するにあたり，改めて iPad の姿勢について説明する．iOS 端末の姿勢の値には，pitch 角，roll 角，yaw 角の回転があり，それぞれ x 軸，y 軸，z 軸を中心に回転する．これら 3 つの値を取得するには，CoreMotion フレームワークの CMDeviceMotion クラスを用いると，pitch 角の角度は attitude.pitch，roll 角の角度は attitude.roll，yaw 角の角度は attitude.yaw の値で取得できる．回転行列とクォータニオンについても attitude.rotationMatrix と attitude.quaternion で取得可能である．CMDeviceMotion で取得した pitch，yaw，roll について表 3.2 にまとめる．

表 3.2 iOS 端末の姿勢の値について

	取得方法	基準となる軸	値の範囲
pitch	attitude.pitch	端末3D座標系のy軸と， 地理3D座標系のZ軸	$-\pi$ ～ π
yaw	attitude.yaw	端末3D座標系のy軸と， 地理3D座標系のY軸	0～2π
roll	attitude.roll	端末3D座標のz軸と， 地理3D座標系のX軸	0～2π

CMDeviceMotion で取得した回転角の値は，端末のモーションセンサの raw データである．このデータの yaw 角の値はプログラムを起動した際の姿勢を初期位置としている．そのため，初期姿勢において yaw 角の値が 0 でない限り，他の回転角に影響を与えてしまう．これを改善するために，pitch 角と roll 角の値は，yaw 角の回転の影響を無くすために，クォータニオンから計算を行った．また，yaw 角の値については初期姿勢の条件を無くすため， CMDeviceMotion で取得できる重力ベクトルの値を用いて計算を行った．値の取得のために行った計算を図 3.7 にまとめた．

クォータニオンのパラメータを(qx,qy,qz,qw)とする．

・クォータニオンからpitch回転を算出する

$$pitchX = 2\times(qw\times qx + qy\times qz)$$
$$pitchY = 1 - 2\times(qx\times qx + qy\times qy)$$
$$pitch = atan2(pitchX, pitchY)$$

・クォータニオンからroll回転を算出する

$$rollX = 2\times(qy\times qw - qx\times qz)$$
$$rollY = 1 - 2\times(qy\times qy + qz\times qz)$$
$$roll = atan2(rollX, rollY)$$

・重力ベクトルのy成分(gravityY)からyaw回転を算出する

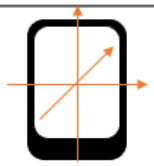
$$\begin{cases} yaw = \mathrm{acos}(-gravityY) & (\text{重力ベクトル}x\text{成分} > 0) \\ yaw = -\mathrm{acos}(-gravityY) & (\text{重力ベクトル}x\text{成分} < 0) \end{cases}$$

図 3.7 端末の姿勢の計算

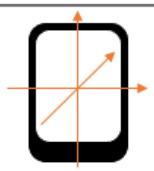
この変更を行った結果が表 3.3 である．表 3.3 より，回転角それぞれがほとんど影響を受けずに値を取得できていることがわかる．

表 3.3 値の取得方法を変更した結果

変更前

pitch	87.970	50.009	33.652
yaw	4.020	-7.049	8.412
roll	-177.834	-93.103	-17.585
姿勢			

変更後

pitch	87.972	83.835	35.535
yaw	-0.001	-39.990	-56.347
roll	-177.836	-93.106	-17.586
姿勢			

3.3.2 AR 画面への経路情報の描画

3.3.1 節までで,経路情報における座標の取得と,目的地の座標について地理 3D 座標系から端末 3D 座標系に変換する方法について説明した.これらを組み合わせて,プログラムを作成することで,AR 画面上の正しい位置に経路表示を描画することができる.経路をカメラ映像の道に沿って表示するために,直方体を組み合わせた折れ線で経路を表現した.一つ一つの直方体の長さは, 図 3.2 の円から次の円までの長さを 3.2.2 節の L と同じ計算方法で計算した. 直方体の他の 2 辺の長さは 1m とした.計算した座標をそのまま表示するとカメラの目の前から経路が表示され, 不自然な感じになった.そのため,端末 3D 座標系に変換した値から 5.0m を引くことでその問題を解決した.

4 章　検証結果

4.1　プログラムの動作検証

3 章の説明を元に作成したプログラムを動作させた. プログラムの流れは以下の通りである.

① 端末の現在地を取得するとともに, 地図を画面上に表示する.

② 目的地を検索または, 画面長押しで決定し, 経路を探索する.

③ 経路情報を保持して, AR 画面に切り替える.

④ 経路の座標を計算し, 道に沿った形で AR 画面上に経路を描画する.

4 章の検証では, プログラムの起動を京都産業大学 14 号館前で行い, その場で目的地を京都産業大学並楽館前のエスカレーター乗り場に設定した. その後, その場で AR 画面に遷移して, カメラ映像上に経路を表示させ, 移動を開始した. 図 4.1 は地図画面で目的地を設定し経路探索を行った結果を示している. 図 4.2 の左図は地図画面から AR 画面に遷移した時の経路表示, 図 4.2 の中図は一定距離を移動した際の経路表示, 図 4.2 の右図は目的地周辺に移動した際の経路表示である.

図 4.1 地図画面で経路探索を行った結果

初期位置　　移動中　　目的地付近

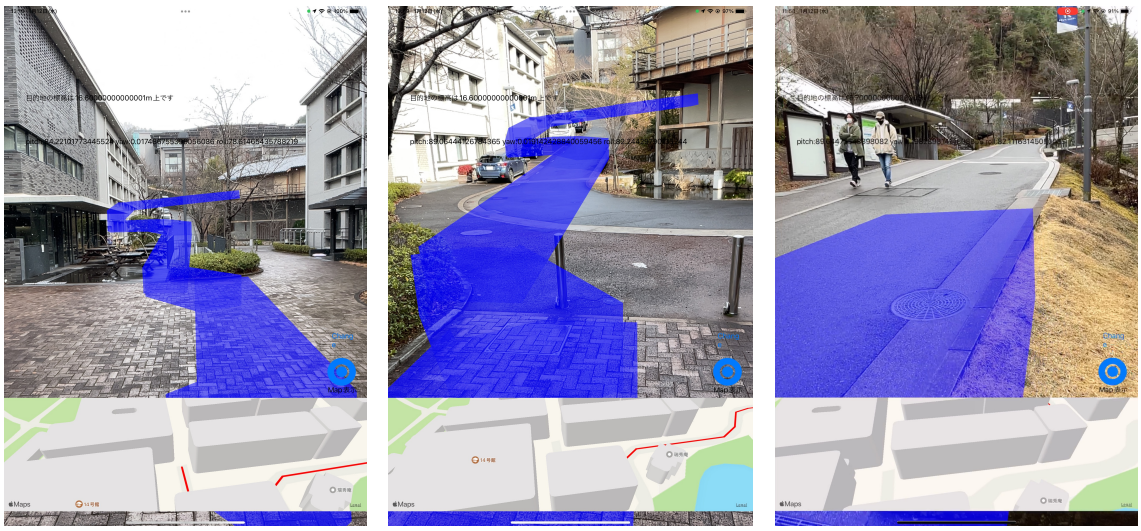


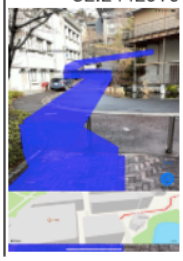
図 4.2 AR 画面での経路表示

図 4.2 では,経路の道路に沿った形状の 3DCG が描画されており,経路の表示している位置についても概ね正しいことが確認できる.また, 国土地理院の情報から初期位置と目的地付近の標高は,それぞれ 118.2m と 134.9m であり,16.7m の標高差が生じている.それにも関わらず,初期位置と目的地付近ではどちらも道路に沿った高さに表示されていることが確認できる.このことからこのプログラムの経路表示では標高が考慮されていることが確認できた.

4.2　課題

4.1 節で行った検証を 10 回行った結果が表 5 である.この 10 回の検証ではプログラムを起動する位置と目的地を 4.1 節と同じ位置に設定し,pitch 角の角度を 90 度付近,yaw 角の角度を 0 度付近,roll 角は表示する経路情報が画面に収まる角度になるように端末を持ってプログラムを動作させた.

表 4.1 AR 画面上に経路を描画した結果の一覧

	1	2	3	4	5
pitch角の値	89.0644413	90.3113431	90.0608023	89.2277589	90.3680247
yaw角の値	0.01914243	-0.3818903	-1.297695	0.00028994	0.03883884
roll角の値	82.2442979	76.3010684	68.9377818	70.5834483	70.6846818
経路表示					

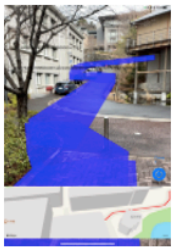
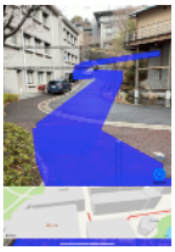
	6	7	8	9	10
pitch角の値	71.9658955	75.5837191	80.9805123	94.974132	83.2700529
yaw角の値	0.0448193	0.0009894	0.00021788	-0.0334102	0.0025973
roll角の値	71.3078545	79.1964062	74.9824682	77.9857329	96.8952066
経路表示					

表 5 を確認すると,3 回目のように yaw 角の値が 1 を超えると大きな誤差を生じていることがわかる. また,4 回目と 8 回目のように yaw 角の値がほとんど変わらない場合でも表示の座標には差があることがわかった.

検証の結果として,2 つの課題点が見つかった.1 つ目は, 経路を移動していると, 段々と経路表示の座標の精度が悪くなり,結果的に図 4.3 に示す通り大幅にずれが生じることである. この問題について,屋外で大きく移動した場合,ワールドトラッキング機能を使用した計算過程で誤差が蓄積してしまうことが原因と考えた. そのため,初期位置を固定してワールドトラッキング機能を使用するのではなく,初期位置を一定間隔で更新させることで対応した.

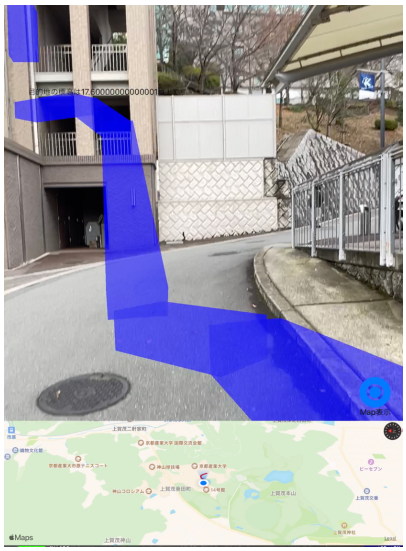


図 4.3 誤差が蓄積して大幅にずれが生じた経路表示

2 つ目は,目的地と初期位置を同じ場所に設定し,端末の姿勢をあまり変化させないように持って起動したのにも関わらず,経路表示の位置が正確に道路に沿っているものや,経路表示が道路から逸れているものがあったことである.この誤差が生じた理由として,端末の回転の補正が不十分であること,位置情報の取得に誤差があることが原因であると考えた.位置情報については,3.1.2 節で説明した通り,平均 10m 以内の誤差があることを確認している.そのため,この 4.2 節では,端末の回転の補正について説明する.

4.2.1　端末の姿勢が経路表示に与える影響

プログラムで行った回転への補正について調査を行なった.ここでは,端末の姿勢が経路表示に与える影響について調査するため,GPS の値は固定値を用いた.

初めに pitch 角をほぼ 90 度に固定し,roll 角を固定して,yaw 角だけを回転させるように端末を動かした.その結果,図 4.4 のように経路表示が回転し,補正ができていないことがわかった.

次に,端末の yaw 回転をほぼ 0 度に固定し,roll 角を固定して,pitch 角を回転させた.その結果,経路表示にはほとんど影響がなかったため,pitch 角の補正はできていることがわかった.

同じように,端末の yaw 角をほぼ 0 に固定し,pitch 角をほぼ 90 度に固定して,roll 角を回転させた.その結果,経路表示にはほとんど影響がなかったため,roll 角の補正はできていることがわかった.

以上より,補正が不十分である回転角は yaw 角であることがわかった.この解決策と

して 4.2 節で行った初期位置の更新の条件として，端末の姿勢が一定数動いた時に yaw 角の値を取得し，yaw 角の回転が-0.1～0.1 の値を取っていた場合，更新を行うことにした．その結果，yaw 回転の影響をほぼ受けずに経路を表示することができた．

yaw=-60.432　yaw=-29.759　yaw=30.044　yaw=62.487

図 4.4 yaw 角の回転が経路表示に与える影響

5章　結論

Apple の Mapkit を利用して現在地から目的地までの経路を探索し，その経路情報を iPad で撮影する周囲のカメラ映像に重畳して表示する技術について研究した．既存の AR ナビアプリを調査した結果,端末の位置と姿勢（回転）に正確に対応することと,標高への対応が課題であると考えた.そのため,この研究では標高と端末の回転に対応し,経路情報をカメラ映像に正確な位置に表示することを目標とした.プログラム開発では,Mapkit で地図を表示し,CoreLocation で取得した現在地の座標と地図で指定した目的地の座標から経路情報を取得する.取得した経路情報と CoreMotion で取得した端末の姿勢の値を組み合わせて,端末 3D 座標系で目的地の座標を指定した.指定した座標を元にカメラ映像に ARkit で経路情報の 3DCG を重畳表示した.

その結果,標高と端末の回転に対応した経路情報を概ね正しい位置に AR 表示することができた.しかし,正しい位置に表示できたのは 10 回中 4 回程度であり,その他の 3DCG の位置には,ずれが生じていた.この原因として,yaw 角の回転の補正ができなかったこと,位置情報の誤差が大きい場合があることだと考えた.

この研究では,現実世界の座標を端末 3D 座標に変換する際に,現在地と目的地の緯度・経度・標高の値と端末の pitch 角,roll 角の値を組み合わせて計算した.そのため,yaw 角については,極座標を直交座標に変換した後に yaw 角の回転行列をかけて補正を行っている.この回転行列の扱い方が不適切であったため,補正ができなかったと考えている.

位置情報の誤差についてはプログラム上で最高精度の設定にしたにも関わらず,平均 8m 前後の誤差が生じていた.位置情報の値は,取得時の電波状況や遮蔽物,天候によっても変化する.位置情報の精度を上昇させるには,位置情報の取得頻度を高くしてフィルターをかけることで改善されると考える.

参考文献

[1] Apple Developer 日本語ドキュメント> ARKit
https://developer.apple.com/jp/documentation/arkit/

[2] 緯度・経度の１度はどれくらいの長さがあるのか
https://www.wingfield.gr.jp/archives/9721

[3] 場所情報コード閲覧システム
https://ucopendb.gsi.go.jp/ucode/view_top.html

謝辞

本論文を作成にあたり丁寧な御指導を賜りました蚊野浩教授に感謝いたします.

付録　本研究で作成したプログラム

開発環境：Xcode,swift5.5.2

プログラム名：MapViewController.swift

iPad 画面で地図を表示し,現在地の表示,目的地の設定,現在地から目的地までの経路の表示や経路情報の取得を行う.

プログラム名：ARPageViewController.swift

MapViewController.swift で取得した経路情報を元に AR オブジェクトの座標を決定し,AR オブジェクトを iPad のカメラ映像に描画する.