2　ARMeasure を使った距離測定 iOS アプリ

12 番　佐々木　祐次郎

指導教員　石舘　勝好

1.　はじめに

私はスマートフォンアプリケーション開発に興味があり，私が使用している iPhone6s と，Swift を使用した iOS アプリ開発をすることにした．

開発に当たって，iPhone の内蔵デバイスを利用したいと考え、矢巾町のレクでペタンクというスポーツが行われていることを知り、競技してみたところ開発のテーマに活用できると考えた．

ペタンクは，フランス発祥の球技でコート上に描いたサークル内から目標球に金属製のボールを投げ合って，相手より目標球に近づけることで得点を競う．「陸上のカーリング」と呼ばれている．

目標球と各ボールの距離が僅差の場合，測る際の手段がメジャーしかなく，この手間を無くす測定アプリを作ろうと考えた．

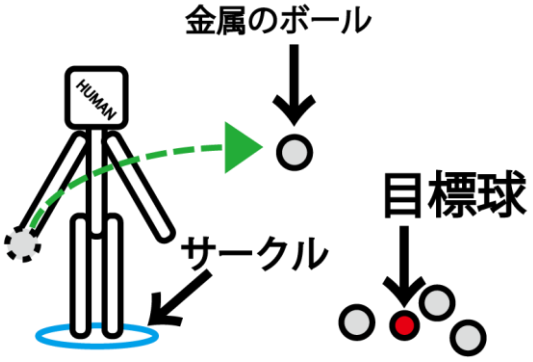

図１　ペタンクの競技イメージ

2.　研究概要

2.1　開発環境

表１　開発環境表

使用機器	Mac Book Pro iPhone6s
OS	iOS 11.4.1
IDE(開発言語)	Xcode(Swift)
主要フレームワーク	ARKit ARMeasure

2.2　Swift について

Swift とは，Apple 社の iOS および macOS 等で利用できるプログラミング言語で Apple 製品向けのアプリケーションを開発するのに使われる．

2.3　ARKit について

iOS 向けの AR フレームワークで特別なデバイスを必要とせず，AR 機能の実装を可能とする．

2.4　ARMeasure について

ARKit を利用したものの一つでカメラを通して現実世界の物体間の距離を測ることが出来る．

3.　機能概要

ARKit を利用して以下の機能を中心に作成した．

・サークルから目標球までの距離を計測
・目標球からボールの距離を計測
・測定結果の写真をシェア
・手が汚れたときのためのタップを必要としない画面操作

4.　システム構成

アプリ全体の構成は図２の通りである．

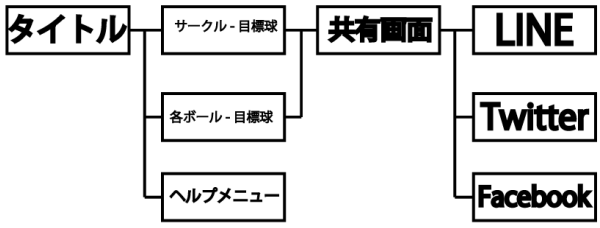

図２　システム構成図

4.1　タイトルメニュー

アプリを起動すると最初に開く画面.

図３　タイトル画面

4.2　ヘルプメニュー

アプリの目的や使い方を簡単にまとめた画面.

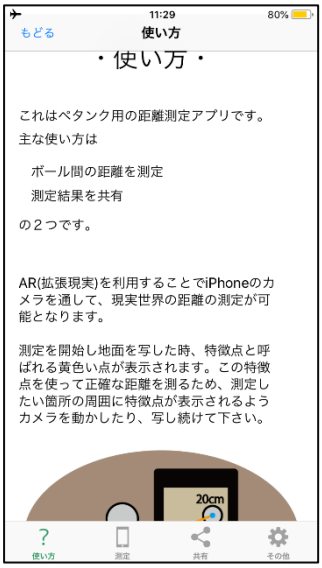

図４　ヘルプメニュー

4.3　サークルから目標球までの測定機能

目標球とサークルの距離は 6～10m の範囲に収める必要があり，その距離を測定する.

サークル位置に立って測定ボタンを押すと測定が開始され，動画を撮影するように地面を映し，平面認識させながら目標球まで歩く．ボタンから指を離すと測定を終了し，サークルから目標球までの距離を表示する.

図５　サークル-目標球測定画面

4.4　各ボールから目標球までの測定機能

各ボールから目標球までの距離を測定する.

始めに目標球の中心をタップし，次に各ボールの中心をタップする．その状態で測定ボタンを押すことで図２のように目標球との距離が近い TOP3 を表示する.

全順位表示ボタンをタップすることで全ボールの目標球までの距離を見ることができる.

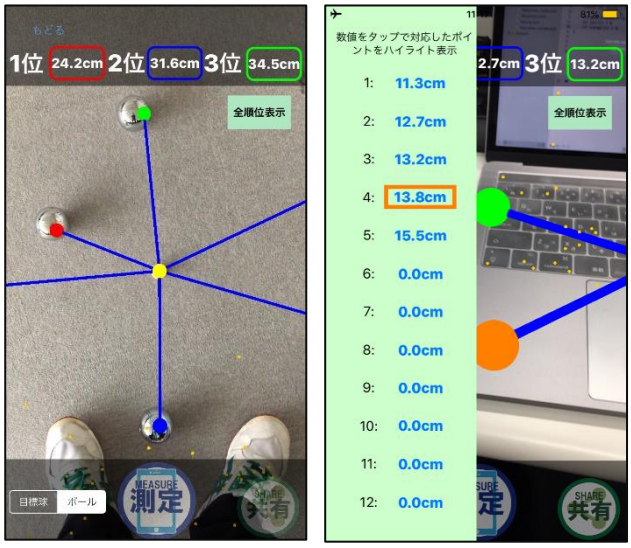

図６　各ボール-目標球測定画面

4.5　共有機能

測定結果のスクリーンショットを LINE, Twitter, Facebook を用いて共有できる.

LINE の場合，送信先選択画面に遷移し，複数人に送信できる.

Twitter，Facebook の場合は，個人メッセージ送信画面に遷移し，スクリーンショットをクリップボードに転送し，メッセージ欄にペーストできる．

図 7　共有画面 (左 共有方法選択画面)

4.6　画面に触れずに操作する機能

4.6.1　加速度センサーによる画面遷移

iPhone に搭載されている加速度センサーを利用し，iPhone 本体を振ることで画面遷移を行う．

しかし，センサーの値による遷移では，一度のシェイクで連続して画面遷移される不具合があったため，ステートマシン図を基に画面の状態を管理するクラスを作り，制御することにした．

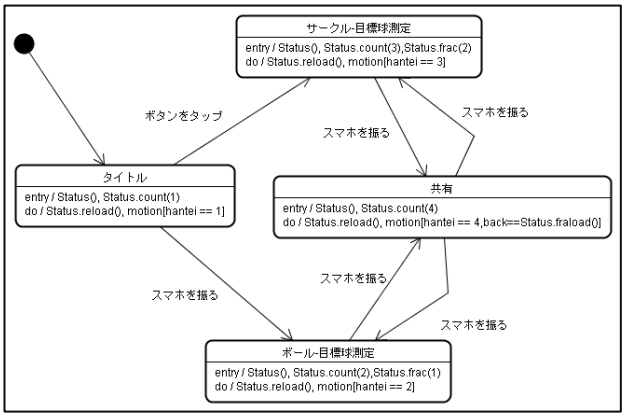

図 8　シェイク遷移のステートマシン図

4.6.2　近接センサーによる測定

近接センサーとは，iPhone 上部についている対象物が近づいただけで触れずとも ON/OFF が切り替わるデバイスである．

近接センサーに手をかざすことで測定ができる．

サークルから目標球までの測定の場合，一度手をかざすと測定を開始し，再びかざすと測定を終了する．

各ボールから目標球までの測定の場合，初めて手をかざすと画面中心に丸いカーソルを表示させる．二度目にカーソルに合わせて目標球用のポイントを打ち，三度目以降は各ボール用のポイントを打つことができる．

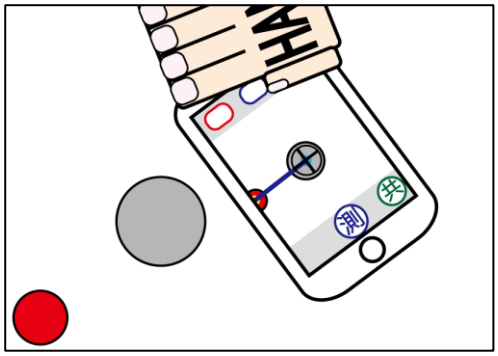

図 9　近接センサーを用いた測定のイメージ

5　アプリの評価

実際にこのアプリがどれほどの有効性があるか検証するためテストを行った．

目視では測定が難しい状況(目標球までの距離が僅差，ボール間に障害物がある，目標球までの距離が遠い)を用意し，アプリが誤りなく測定できるかについて調べた．

表 2 は，5 回テストを行った中で，目測の順位の判定誤りが起きた 2 回について，それぞれの計測結果を比較したものである．

表 2　実測，アプリ計測，目測の比較

順位	実測 (cm)	アプリ (cm)	アプリ (位)	目測による順位予想		
				Sさん	Tさん	Yさん
1	87.1	87.5	1	1	2	1
2	91.5	90.3	2	2	1	2
3	98.5	97.8	3	3	3	3
1	53.3	53.2	1	1	1	2
2	57	56	2	2	2	1

実　測：目標球と各ボールをメジャーで測定
アプリ：目標球と各ボールをアプリで計測
順　位：目標球に近い順に，順位付け

実測とアプリ計測の差は，最小で 0.1cm，最大で 1.2cm に収まった．ポインタを打った際のズレがあるとしても，1m 近くの距離でここまで誤差を抑えられたのはアプリの精度が高いと言える．

目測でボールの順位付けを行ったところ１つ目のケースではＴさんが，２つ目のケースではＹさんが誤った判定をした．だが，アプリはどちらのケースにおいても間違えることなく順位付けすることが出来た．

意見が割れた場合，多数や主張が強いプレイヤーの意見が通りやすいと考えられ，意見が割れやすい環境でも正しい結果を出せるのはアプリに有効性があるといえると考えられる．

6. 改善点・課題等

・測定の際のポインタの打ち方

現在のポインタの打ち方だとズレが起きてしまう．ボールや目標球を自動認識出来るようにする必要がある．

・アプリのデザイン

デザインに関する知識が乏しく全体的にシンプルなデザインになってしまった

7. 終わりに

Swift という初の言語を使うため上手くいかず，アプリが完成しないのではないかという不安があったが，ほかのプログラム言語に近く直観的にプログラムを書くことができたため，予想していたよりはスムーズに進むことが出来た．

当初予定していたものよりも高度なものが出来たと感じたが，研究を進めるとともに機能に対し不満を感じる部分が多く妥協した場面がたくさんあった．そこに関しては自身の実力不足によるものなのでもっと ARKit に関する知識と経験をつけたい．

また，一つのアプリを開発することに挑戦して，プログラムだけでなくユーザーが使いやすいデザインを実現する力も欲しいと感じた．

8. 参考文献

(1) ペタンク

https://ja.wikipedia.org/wiki/ペタンク

(2) Swift

https://ja.wikipedia.org/wiki/Swift_(プログラミング言語)

(3) Swift 参考書

高橋京介 『絶対に挫折しない iPhone アプリ開発「超」入門』