

平成??年度 卒業研究論文

English Title

2段目はここ

日本語のタイトル

2018年2月7日

茨城工業高等専門学校 電子情報工学科

NAMAE MYOJI

苗字 名前

Supervised by Kei KOGAI

指導教員 小飼 敬 教員

目次

1 章	LaTeX の解説 基礎	1
2 章	LaTeX の解説 サブセクション	2
2.1	サブセクション名をここに書く	2
2.2	二つ目のセクション名	2
2.3	追加説明したい時	3
3 章	LaTeX の解説 画像	4
3.1	画像の挿入	4
3.2	使用例	4
4 章	LaTeX の解説 その他	6
4.1	コードの埋め込み	6
4.2	表記の統一	6
4.3	長音符の扱い	6
4.4	句読点	6
5 章	はじめに	8
6 章	開発経緯	9
6.1	何故発展しないのか	9
6.2	解決案	10
7 章	設計	11
7.1	コンセプト	11
7.2	開発環境	11
7.3	構造	12
8 章	実装	14
8.1	MainActivity	14
8.2	Setup	14
8.3	AR	14
8.4	Draw	14
8.5	Cube	15
8.6	実行結果	15

9 章 検証	18
9.1 関数名や変数名を明瞭化	18
9.2 各クラスの役割を明瞭化	18
9.3 拡張しやすい構成に	18
9.4 MainActivity は最小限に	19
10 章 考察	20
10.1 考察 1 : Setup の複雑化	20
10.2 考察 2 : 拡張性	20
10.3 考察 3 : MainActivity	20
11 章 総評	21
11.1 評価	21
11.2 課題	21
11.3 今後の発展	22
12 章 おわりに	23
13 章 謝辞	24
付録 A 付録名	26

1 章 LaTeX の解説 基礎

文章はここに書く。普通の改行では改行されない。バックスラッシュ二つで改行される段落を変える場合は

次の文章は一文字落ちる。改行二つでも段落が変わる

引用したいときは [1] で番号を付与できる(要二回実行)。引用に関しては参考文献も参照してほしい。

段落を変える時は `vspace5pt` で行間を空けると綺麗に見える

2 章　LaTeX の解説 サブセクション

2.1 サブセクション名をここに書く

サブセクションの下に `vspace` で余白を作る事で狭苦しくならない。

2.1.1 さらにもう一段階作りたい時はこんなかんじ

サブサブの下に余白を入れるかどうかは好みによると思う。テンプレ作者は入れる派。これより細かいセクションはできない。そこまで細かくすると却って見づらくなってしまう。

- 箇条書きはこんなかんじ
- いくらでも増やせるよ
 - 入れ子もできるよ
 - さすが `LATeX` だね
 - * ちなみに `TeX` とか様々なコマンドが存在するよ
 - * 特に数式のコマンドは人によってはたくさん使うんじゃないかな
 - `end` で入れ子脱出。
- 入れ子にしたら `end` を忘れずに。

画像の挿入は次のセクションで扱っているよ

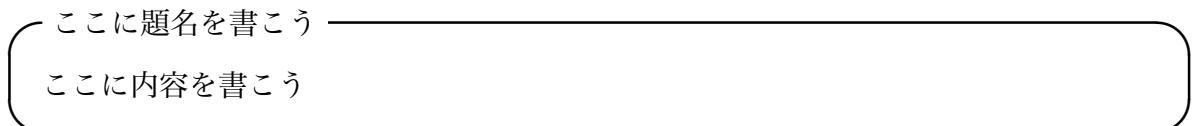
2.2 二つ目のセクション名

セクションは多ければ多いほど良いという訳では無いけれど、目次が豪華になってそれっぽく見えるよ。でもサブサブセクションは目次には反映されないよ。

もしなんらかの理由（配置とか?）でページを変えたい場合は、

で次のページに移るよ。画像を表示している時の改ページは注意点があるよ。それも次のセクションで扱うよ。

もし文章を枠で囲いたい時は…



改行を忘れずに！

2.3 追加説明したい時

なんらかの理由 (配置とか?) … あまりかっこよく無い！

なんらかの理由^{*1} … 論文っぽい！ こっちの方がオススメ.

^{*1} 配置とか？

3 章　LaTeX の解説 画像

3.1 画像の挿入

画像を挿入したい時は、このファイルの最後にあるセットを組み込んでね。

3.2 使用例

進捗が一向に進まない… そんな友人に送るべき画像を Fig.2 に示す。この画像を見せた時の反応は以下の 5 種類に分けられる。

- 顔は笑いながらキレる
- 無表情でキレる
- 眉を吊り上げてキレる
- 無言でキレる
- 殽ってくる

画像を入れた際の改ページは注意が必要。改ページする前に全ての画像を出力したい時は、`newpage` ^{*2}ではなく `clearpage` を使う。`newpage` を使用した場合、まだ出力されていない画像は改ページ後に出力される。

^{*2} 空白を開けているのはコマンドを実行して欲しく無いため



Fig.1 Class diagram

4 章　LaTeX の解説 その他

4.1 コードの埋め込み

もしコードを埋め込みたいなら…

```
#include <stdio.h>
int main(void)
{
    printf("Hello, World !\n");
    return 0;
}
```

この中では改行も含めて書いたままが出力されるよ。

4.2 表記の統一

拡張現実と拡張現実感など、表記ゆれに気をつける。気をつけると気を付けるなど漢字にも注意。二日間と2日間もよくある表記ゆれ。また語尾はだ、である調で書く。^{*3}

4.3 長音符の扱い

四字以上で最後が長音符だった時、長音符は省略する。

例

- NG: プリンター OK: プリンタ
- NG: キャラクター OK: キャラクタ
- NG: メンバー OK: メンバ

この間違いは多発するので特に注意。

4.4 句読点

句読点の打ちすぎに注意。例えば、こんな感じで、多用すると、とても、読みづらい。読点が多くなったら文の前後入れ替えや語尾の変更などでいい感じに整えよう。

^{*3} ただし謝辞はですます調で書かれることが多い。

ここから先はテンプレート作成者田村の論文です。参考になるかは分かりませんが…

今まで 1 年間研究を頑張ってきたと思いますが、論文作成は 1 年間の総仕上げです。これからが本番と言っても過言ではないでしょう。論文は計画的に、2 日前には担当教員に提出できるように！⁴

最後の謝辞に少しだけ注釈を入れてあります。消去するのを忘れないように気をつけて下さい。

作成日 2017 年 2 月 27 日⁵ 小飼研 田村 優

⁴ ちなみに作成者は提出日に寝坊して 3 時間遅れで提出しました

⁵ 2 月 28 日が卒研発表日でした

5 章　はじめに

2016 年は「VR 元年」とも呼ばれるほどに VR の発展が目覚ましかった。VR、すなわち仮想現実感は被験者に完全な仮想の視界を提供する技術であり、被験者の視界は完全に遮られ代わりに新しい視界を与えられる [2]。今まで以上にリアリティの高い経験を得ることができることから世界中の注目を集め、今では一つの分野として成長を続けている。

また、VR と似た分野に AR というものがある。AR、すなわち拡張現実感とは、被験者の視界を完全には遮らず、普段の視界を保ったまま新たな情報を提供する技術である。拡張現実感と仮想現実感は名前こそ似ているがその実態は全く異なり、使われている技術や想定されている使用用途など様々な点で違いがある。

仮想現実感は視界を完全に遮るために自身の移動に大きな制限が掛かってしまう。そのためその場合その場から動いて使用するような想定はされていない。これは据え置き型のゲームとの相性がとても良く、現在も仮想現実感を用いた据え置き型ゲームの市場は拡大しつつあると言える [3]。

拡張現実感は仮想現実感とは違い現実の視界が保たれているため、移動しながらの使用も可能である。また移動しながら使用することを想定したアプリケーションも開発されてきた [4]。

ここで仮想現実感と拡張現実感の違いのもう一つ、すなわち使われている技術について注目してみると、意外なほどに異なっている。仮想現実感では 3D グラフィックス、頭の向きを測るための加速度計測などが中心に使われる。しかし拡張現実感では 3D グラフィックスに加え画像認識を用いている。つまり向いている方向を判断する技術が全く異なるのだ。

今までに挙げた仮想現実感と拡張現実感の特徴をまとめると、仮想現実感は趣味や娯楽方面でとても有用だということがわかる。また拡張現実感は仕事や生活の効率化に有用だということがわかる。しかし、拡張現実感は仮想現実感に比べ世間の注目度や浸透度などにおいて遅れをとっていると言える。

何故、拡張現実感はそこまで仮想現実感に比べ発展していないのだろうか？著者はこの疑問に対する回答及び解決案を提示することにした。

6 章 開発経緯

6.1 何故発展しないのか

6.1.1 要因予想

前章で拡張現実感の現状を述べた。少ないながらも拡張現実感を実現させるためのライブラリは提供されているが、何故ここまで拡張現実感は広まっていないのか？著者はその要因を以下のように予想した。

- 興味を持つ人が少ない
- 拡張現実感が何かを知らない
- 拡張現実感を使う、という発想が無い
- 実装することで得られるメリットが少ない
- そもそも実装が難しい

ほかにも様々な要因があると思われるが、上に挙げただけでも拡張現実感が普及しない理由としては十分だと言えるだろう。

この要因に対して、解決案をそれぞれ考えることにした。

6.1.2 解決案

先程述べた要因に対して、それぞれ解決案を考えてみた結果を以下に示す。

- 興味を持つ人が少ない
 - 拡張現実感を用いたイベント等で興味を持ってもらう
 - * 例) スマートフォンでかざしてみると様々な立体映像が見られる等
- 拡張現実感を使う、という発想が無い
 - 拡張現実感を用いることで得られるメリットをPRする
 - * 例) 拡張現実感を用いたシステム等からどれだけ付加価値があるか調べ、公表する
- そもそも実装が難しい
 - 開発しやすい環境を構築する

上記より、上から 2 つに関してはイベントなどで拡張現実感を使うというもので、まずは拡張現実感というものを知ってもらおうという案である。続く 2 つは、メリット等が分かつただけでは最後の“開発の難しさ”に直面し、これだけでは意味がない。

結果、まず取り組むべき要因は“拡張現実感に興味を持つてもらう”と“開発の難易度を下げる”だと考えた。

6.2 解決案

上に挙げた要因より、“拡張現実感に興味を持つてもらう”と“開発の難易度を下げる”が最優先ではないかと考えた。そしてこの二つを解決するためには、以下の案がとても効果的ではないかという結論に至った。

解決案の提案

拡張現実感を初めて体験する人に向けた簡易的なライブラリの制作

また、使用するデバイスは Android を選択した。理由はカメラと画面が一体化しており、直感的な操作ができるから、また今後の発展性が大きく望めるからである。

しかし、プログラミング初心者向けの言語である Processing[10] などと違い、Android は複雑で高機能、どちらかといえばプログラミングに慣れた人でないと扱うのが難しい言語である。実際 “Hello,World!” を表示する事はそこまで難しくないが、そこから新しい機能を追加すると途端に難しくなってしまう。ライブラリの追加方法やその利用方法などもある程度 Android 開発を経験している開発者向けの説明が殆どで、一から詳しく解説しているドキュメントはとても少ない。

Android は拡張現実感を体験するにはとても良いが、開発が難しい。そこで、“とりあえず拡張現実感を体験してみよう”をテーマにした Android 向けライブラリの作成をしようと考えた。

7 章 設計

7.1 コンセプト

今回作成するライブラリのコンセプトを以下に示す。

- 対象者
 - Android アプリケーション開発初学者
- 想定している使用目的
 - 中・高生等を対象にしたセミナー
 - Computer Graphics(CG) の授業や実験
 - 企業等での 3DCG を用いた Android プログラミングの学習
- ライブラリの特徴
 - 関数名や変数名を明瞭化
 - 役割ごとにクラスを分け、各クラスの役割を明瞭化
 - 拡張しやすい構成にする
 - MainActivity^{*6}はできるだけ最小限の構成に

7.2 開発環境

7.2.1 Android 開発環境の現状

現在、Android アプリケーションの開発環境は以下の二つが主流である。

- Eclipse[11]
 - Java をはじめとするいくつかの言語に対応している統合開発環境である。Android が登場してすぐの頃は Eclipse での開発が主流だった。
 - AndroidStudio[12]
 - Android を開発している Google 社が提供している Android 統合開発環境である。初期の AndroidStudio は動作が不安定だった為しばらくは Eclipse が使われていたが、最近はアップデートを繰り返し安定性が増してきている。
- 現在は Eclipse から AndroidStudio に移りつつあると言える。しかし依然 Eclipse で開発をする人が多い事も事実である。

^{*6} Android アプリケーションを実行した時、一番最初に実行されるメソッド

現状を踏まえて以下の環境で開発する事にした。

- OS : Mac OS X Yosemite[13]
 - 動作の安定性、今後 iOS 版^{*7}の開発をする可能性などを考慮し、Mac mini を使用した。
- 開発環境 : AndroidStudio 2.2.2
 - 近年の開発環境の遷移を踏まえて、これからのスタンダードと考えられるため選択した。
- 使用ライブラリ : ARToolKit version 5.3.1
 - Android の最新バージョンに対応している他、様々なプラットフォームに対応している為今後の展開を踏まえ選択した。
- 実行端末 : Nexus 9[14]
 - Android 7.0 が動作する Nexus 9 を AR アプリケーションの実行端末として使用した。

7.3 構造

構造の明瞭化を図る為クラスを以下のように分割し、利用者が使用するクラス、ライブラリに格納できないクラスを除き全てライブラリに格納する。クラス図を Fig.2 に示す。またライブラリ化する事によって改変によるエラーを防ぎ、明瞭なコードになるよう促す。

7.3.1 App モジュール

- MainActivity
 - 簡易化する為、親クラスを変更し Setup を呼び出すだけとなっている。
- Setup
 - ここではライブラリに格納できない設定を行うクラスである。初学者には難しい設定や変更する必要のないメソッド等が格納されている為、ライブラリと共に提供されるクラスである。使用者は提供された Setup クラスに指定の変更を加え、MainActivity と同じパッケージに入れるだけで実装が完了する。
- AR
 - 実際に AR で表示したい物体の設定をするクラスである。簡易化を図る為、表示できるのは Box のみである。Box の大きさ、色、位置、角度、対応する AR マーカーを登録するクラスで、使用者はここを変更する事で様々な変化を作る

^{*7} iOS では Swift と言われる言語を使用しており、開発は今の所 Mac でないと不可能である

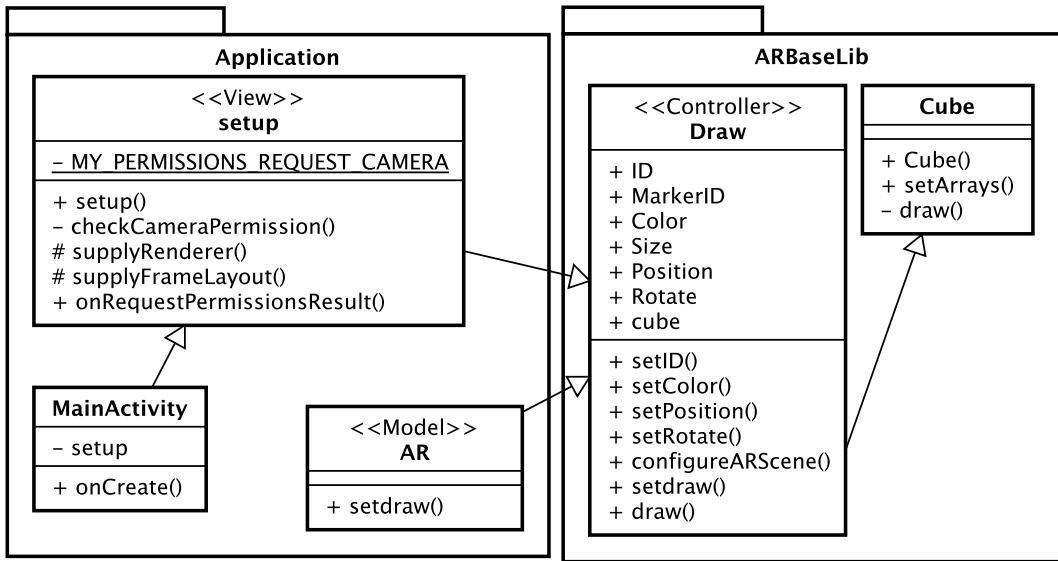


Fig.2 Class diagram

事ができる。

7.3.2 ライブライ

- **Draw**
 - AR マーカの検出、及び Box の表示をする処理が入っている。複数のマーカに反応させる場合や時間が経つにつれて変化を起こしたい場合、この処理を変更しないといけない。
- **ARSimpleApplication**
 - ARToolKit ライブライを使用する際に App モジュールに追加しないといけなかつたが、ライブライ内に移動する事で処理の明瞭化を図る。
- **Cube**
 - 与えられた情報から色情報付きの Box を生成するクラスである。ここを変更する事で Box 以外の物体を表示できるが、今回のコンセプトと異なる為ライブライ内に格納し Box のみを表示するライブライとした。

その他、拡張現実感を実現する為のクラスが多々あるが、全て ARToolKit から提供されている物である。

8 章 実装

前章で設計したクラスを実装する。

8.1 MainActivity

- 親クラスを Setup に変更し, onCreate^{*8}内で親クラスの setup メソッドを実行している。
- それ以外の処理は無く, アプリケーションを立ち上げた際に一度だけ呼ばれるクラスとなっている。

8.2 Setup

- 親クラスは ARActivity^{*9}で, アプリの起動, 画面の作成, カメラからの画像の取得など初期設定を全て担っている。また Renderer^{*10}をセットするなど描写方面的初期設定も同時に担っている。
- 基本的にこのクラスは初期設定を担っている為中身を変更する事はない。よってライブラリに付属させて配布する事が可能である。

8.3 AR

- 後述する Draw クラスで描画する Box の情報を Set するメソッドである。Draw クラスの Setter をまとめたクラスと言える。コードの内容もほぼ Draw クラスの Setter を呼び出して数値を設定しているだけである。
- わざわざ Setter だけを分離した理由は, 変更箇所を一まとめにする事で変更してはいけないところを間違って変更してしまうなどのミスが減らせる事などである。

8.4 Draw

- 前述した AR クラスから値を受け取る Getter と後述する Cube クラスで設定した Box を描写するクラスである。AR クラスから受け取った値を元に Cube クラスで Box を作成し, マーカー上に描写する。
- この Draw クラスの描写をするメソッドを変更する事で様々な表現が可能である。

^{*8} Android アプリケーションが立ち上がる際に実行されるメソッド

^{*9} ARToolKit の中枢クラス, ライブラリのクラスを制御している

^{*10} 3D オブジェクトを描写するクラス

しかし今回のコンセプトは“とりあえず拡張現実感を体験しよう”なのでここでは Box を表示するだけにとどめている。

8.5 Cube

- 受け取った数値を元に大きさ, 色, 透明度の情報を持つ Box を生成するクラスである。
- ここを変更する事で様々な形を表現する事が可能である。また 3D オブジェクトの表示も可能である。しかし前述した通りコンセプトに合わない為 Box の表示をする機能だけにとどめた。

8.6 実行結果

実行結果を Fig.3 - Fig.5 に示す。Fig.3 は実行前, Fig.4 は色指定が (R, G, B, α) の時^{*11} (0, 255, 0, 0), Fig.5 は色指定が (0, 255, 255, 100) の時である。また使用者が実際に改変するクラスである AR.java を以下に示す。

```
package com.example.yuut.test6;
import org.artoolkit.ar.base.Draw;

public class AR extends Draw{

    public Draw MarkerID = new Draw();

    @Override
    public void setdraw(){
        MarkerID.setID("single;Data/patt.hiro;80");
        MarkerID.setColor(0, 255, 255, 100);
        MarkerID.setSize(1);
    }
}
```

^{*11} いずれも最大値は 255 である



Fig.3 Before execution

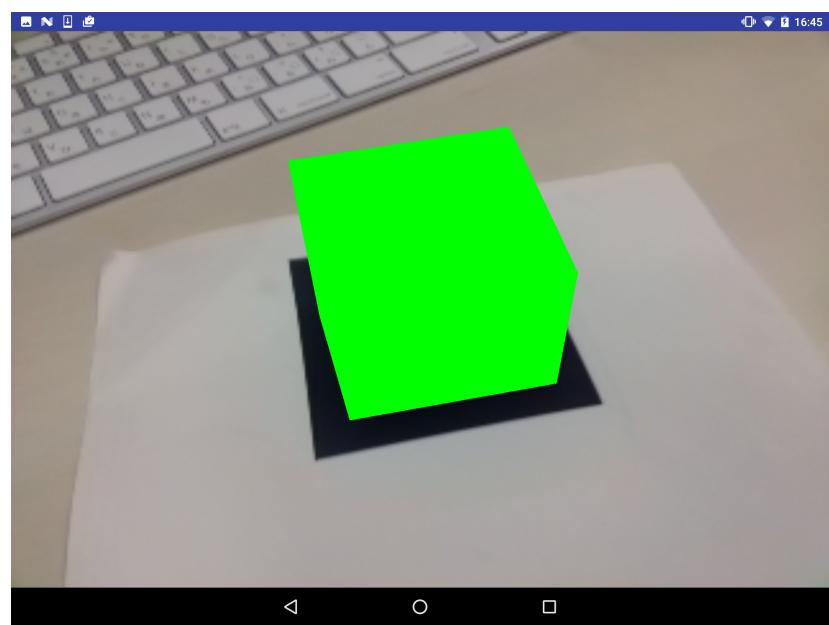


Fig.4 Execution 1

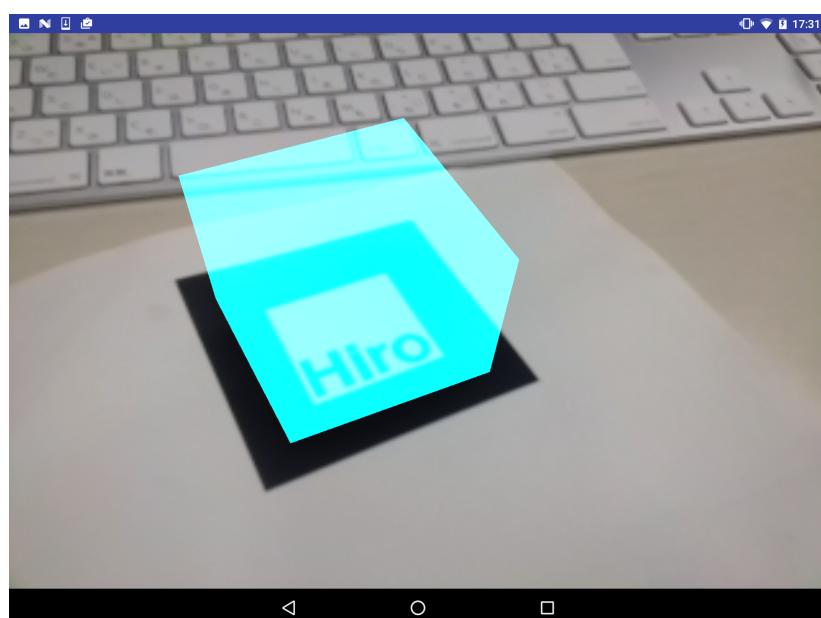


Fig.5 Execution 2

9 章 検証

前章で実装したプログラムを実際に実行し、設定したコンセプトを達成できているか確かめる。

9.1 関数名や変数名を明瞭化

関数名や変数名を明瞭化する事によって拡張や改変する際にコードの役割が分かりやすくなる。

- Draw や Setup など、一目で何をするクラスなのかがよく分かるようになったと言える。AR クラスはそのままだと何をするクラスなのかが分かり辛いが、AR 全般の設定をしている為このようなクラス名となった。

9.2 各クラスの役割を明瞭化

役割ごとにクラスを分ける事で、改変や拡張時にどこを変更するかがすぐに分かる。また改変時にエラーが生じた際、どこにエラーがあるかが分かりやすくなる。

- MainActivity, Setup, AR, Draw, Cube 等、クラスごとに役割を分け、何をするクラスなのかを明瞭化することが出来た。

9.3 拡張しやすい構成に

提供されたライブラリを使ってできることは限られており、また AR を用いた Android プログラミングの学習用途であれば、拡張しやすい構成にすることは重要なことであると言える。

- 拡張するにはライブラリ内を変更する必要があり、初学者には厳しいと考える。設計時の対象者では拡張は難しいだろう。拡張性については達成できたとは言い難い。

9.4 MainActivity は最小限に

MainActivity は C や Java で言う Main 関数である。Main 関数は処理するクラスではなく、メソッドを実行するクラスであるため、ここでの処理は出来る限り最小限にすることが望ましい。

- MainActivity は Setup クラスの setup メソッドを呼び出すだけとなっている。その後は Setup クラスが初期設定をした後、ライブラリ内の ARActivity が処理を引き継ぐ。

10 章 考察

開発したライブラリに対しての考察をする。

10.1 考察 1 : Setup の複雑化

まず第一に、Setup クラスについてである。このクラスはライブラリ内では出来ない設定をするクラスであり、ライブラリと共に提供されると説明したが、設計初期はこのクラスは存在せず全てライブラリ内で初期設定をする予定だった。しかし設計と並行してライブラリの学習をしていた際、どうしてもライブラリでは出来ない設定があることを知り、当初無かったこのクラスを実装することになった。しかし設計時の予想よりもライブラリでできる事は少なく、結果 Setup クラスが膨大になってしまった。Setup クラスを提供する事によって実装時の負担を減らすことが出来たが、分かりやすいライブラリからは少し遠ざかってしまった。後述するがここは今後の一番の課題であると考える。

10.2 考察 2 : 拡張性

第二に、拡張性についてである。前述したとおり、拡張性については達成できたとは言い難い。その原因として、簡易化するにあたって内部処理の大半をライブラリ内に格納してしまったからであると考える。これは拡張性と簡易化がトレード・オフの関係にあると言え、拡張性を求めるに今度は簡易化が達成できない。しかし簡易化すると今度は拡張性が失われる。この二つを両立させることは難しいが、拡張する範囲を限定することで両立に近づけることはできるだろう。この場合どの範囲まで拡張性をもたせるかの判断も必要であり、後述する課題の一つと考えることができる。またライブラリ内の構成を工夫することでライブラリの改変を簡易化することができれば拡張性を失うことなく簡易かも達成できるだろうと考える。

10.3 考察 3 : MainActivity

第三に、MainActivity である。今回は親クラスを変更したが、MainActivity は所謂 Main クラスであるため、親クラスを変更することは望ましくない。既存のプログラムで MainActivity を既に改変していた際、本ライブラリの導入が出来ない可能性があるからだ。今回は AR の体験、というコンセプトだった為、既存プログラムへの導入に関しては想定には含まなかったが、今後ライブラリを改良していく際の課題の一つであると考える。

11 章 総評

実装したライブラリの評価、課題及び今後の発展について、以下に述べる。

11.1 評価

実装したライブラリの検証、及び考察から、設計時のコンセプトは概ね実現していると言える。実際 AR クラスを変更するだけで様々な Box を表示することができるようになった。これにより AR とはどんなものなのか、また Android で AR を動かすために必要な処理がどんなものなのか等の学習に適切なライブラリになったと言えるだろう。また AndroudStudio の設定やライブラリの導入などを事前にしておけば中学生でも Android で AR を動かすことができるだろう。

11.2 課題

検証及び考察で述べた通り、以下の課題が残っている。

11.2.1 Setup の簡易化及びライブラリ内への格納

先述した、今後一番の課題である。Setup クラスでは初学者には難しい初期設定などが記述されており、記述するにはとても難易度が高いものである。今回はライブラリと共に提供という形になったが、ライブラリ内に格納できればそのような問題はなくなる。この課題が達成できれば、App モジュール内は MainActivity 及び AR クラスの二つのみとなり、使用者の作業が格段に簡易化される。ここまで簡易化されれば、プログラム未経験者対象のセミナーなどで AR を体験することも容易になるだろう。

11.2.2 拡張性の確保

先述したが、拡張性は簡易化とトレード・オフの関係にありライブラリの簡易化は拡張性の喪失とも言える。しかしライブラリ内を更に簡易化し段階的に拡張性をもたせれば拡張性を喪失せずに簡易化できると考える。App モジュール内の変更、ライブラリ内の変更、この二つを段階的にすれば拡張性をもたせつつ簡易化できるだろう。

11.2.3 MainActivity の親クラスの変更

MainActivity の親クラスの変更は既存プログラムへの組み込みには適さない。この問題は Setup の簡易化及びライブラリ内への格納の問題が解決すると同じく解決するため、並行して進めていく予定である。

11.3 今後の発展

本ライブラリを更に改良することで更に簡易化し、また拡張性をもたせ様々な人にとって使いやすいものにしようと考えている。また以下の機能を追加することで更に ARについて学べるライブラリになるとを考えている。

- 3D オブジェクトの表示
- 時間経過による変化の組み込み
- 複数マーカの認識
- テキストの表示
- 画像、動画の表示

以上の機能を簡易化し、組み合わせやすくすることで今後の情報教育にも十分活用できるライブラリになると考える。

12 章 おわりに

本ライブラリによって Android アプリケーション開発初学者でも、比較的簡単に拡張現実感を体験する事ができるようになったと考える。ディスプレイを見ながらカメラを動かすのではなく、カメラの視点と自分の視点がほぼ同じで直感的な操作ができる点で Android はとても有用であり、今後の発展も望めるだろう。しかし課題も多くまだ実用的だとは言い難い。今後の発展で述べたもの以外にも、Raspberry Pi との連携など、更に拡張現実感の入門用ライブラリとしての十分な機能を実装したい。

拡張現実感ははじめに述べたように遊戯ではなく生活の補助として発展するだろうと予測している。車、テレビ、冷蔵庫などの登場、携帯電話の登場、スマートフォンの登場と、今までの生活が大きく変わる発明が過去に沢山あった。拡張現実感を用いた製品の登場が今までの生活を大きく変えるものとなるかどうかはまだ分からぬが、大きく期待はしている。

ヘッドマウントディスプレイ (HMD) と言うものがある。これは頭に装着して拡張現実感の出力先として使用する装置である。今はまだ機械自体が大きく、これを着けて外を歩くのは実用的ではない。しかし日本の得意分野である小型化によって簡単に外に持ち運べるようなものになった時、生活が大きく変わるだろうと考えている。また、スマートグラスという外見が眼鏡のようなものも数多く登場しており、拡張現実感の出力先の発展は遅れながらも確実に進んでいる。

出力先の発展だけでは拡張現実感は発展しない。本研究のような AR ライブラリの開発も必要不可欠である。今後、更にたくさんの AR ライブラリの開発が進むことを願っている。最後に、本ライブラリによって更に拡張現実感に触れる機会が増え、興味を持っていただける事を期待している。

13 章 謝辞

本研究を進めるにあたり、拡張現実感や開発基盤に関する多大な知識をご教授頂いた指導教員の小飼敬先生に感謝します。また、英文テーマの決定及びライブラリ開発の際に生じた疑問についてご教授頂いた石垣達也先生に感謝します。最後に、この一年間協力してくれた小飼研究室のメンバに感謝します。

ここから先は削除してね。ここでは1年間で協力してくれた方に謝辞を述べるところです。間違っても研究に関係のない人を書かないように(恋人とか書いてると将来黒歴史になるので注意)

最初に担当教員、そのあとは研究室から遠い順に。例えば…

- 担当教員
- 学校外の協力者
- 他学科の協力者
- 学科内の先生
- 学科外の生徒
- 学科内の生徒

学科外の生徒を学科内の先生より先に書くかどうかは私もよくわかりません。そこまでたくさん書く必要はないです。

参考文献

- [1] 会社名や人物名 - サイト名や書籍名
アドレスや出版社
- [2] 日本バーチャルリアリティ学会 - バーチャルリアリティとは
<https://vrsj.org/about/virtualreality/>
- [3] Engadget 日本版 - 「PlayStation VR」1月26日に再販スタート、同日発売の「バイオハザード7」もVR完全対応
<http://japanese.engadget.com/2017/01/16/playstation-vr-1-26-7-vr/>
- [4] KDDI - 「AR ナビゲーション」機能追加について
http://www.kddi.com/corporate/news_release/2013/0117/besshi2.html
- [5] Weblio 辞書 - スマートグラス
<http://www.weblio.jp/content/スマートグラス>
- [6] Pioneer - サイバーナビ
http://pioneer.jp/carrozzeria/cybernavi/avic-vh99hud_avic-zh99hud/
- [7] ARToolKit - Android Documentaion
http://artoolkit.org/documentation/doku.php?id=4.Android:android_about
- [8] NyARToolkit project - Welcome to NyARToolkit project
<http://nyatla.jp/nyartoolkit/wp/>
- [9] Google Code - AndAR Android Augmented Reality
<https://code.google.com/archive/p/andar/>
- [10] Processing - Cover
<https://processing.org/>
- [11] Eclipse - MainPage
<https://eclipse.org/>
- [12] Android Developers Site - Android Studio
<https://developer.android.com/studio/intro/index.html>
- [13] Apple - OS X Yosemite (2014年10月22日 INTERNET ARCHIVE - WayBack Machine)
<http://web.archive.org/web/20141022105913/http://www.apple.com/jp/osx/>
- [14] HTC - Nexus 9
<http://www.htc.com/jp/tablets/nexus-9/>

付録 A 付録名

付録をここに書く