令和 5 年度 卒業研究

開陽丸記念館に展示し展示物について学べるクイズアプリの開発

Development of a quiz application to learn about the exhibits on display at the Kaiyo-maru Hall

函館工業高等専門学校
生産システム工学科 情報コース
32 番 宮原 良太
指導教員 東海林 智也

目次

1.　序章

1.1　目的

1.1.1 和文

本研究の目的は、開陽丸記念館の展示物について学べるアプリケーションを作ることである。館内にある QR コードを読み込んで、展示物に応じた問題が出題されるアプリケーションを開発する。このアプリケーションで学習することで、開陽丸や、江戸時代、幕末の出来事についてより深く学ぶことが出来る。

1.1.2 英文

The purpose of this study is to create an application that allows visitors to learn about the exhibits at the Kaiyo-maru Memorial Museum. We will develop an application that reads QR codes located in the museum and asks questions corresponding to the exhibits. By using this application, visitors will be able to learn more about the Kaiyo-maru, the Edo period, and events at the end of the Tokugawa Shogunate.

1.2　研究背景

北海道の江差町には開陽丸記念館という博物館がある。そこには開陽丸の建造から沈没までの歴史を説明するパネルや、沈没した開陽丸から発掘された物が展示されている。

開陽丸について書かれたパネルは 2 階に設置されており、そこを周れば当時の出来事を学ぶことが出来る。しかし、そのパネルの数やパネル内の文章量が多く、すべて読むのは困難であった。そこで、クイズ形式で学ぶことによって、パネルの概要をより簡単にわかりやすくすることを目的に本アプリケーション開発することにした。

2.　アプリケーション概要

2.1　開発環境

本研究における開発環境は以下のとおりである

使用ソフト：Unity 2020.3.21f1
OS：Windows 10
使用言語：C#
ゲーム実機：ASUS Zenpad

2.2　使用アセット

本研究において使用したアセットは以下のとおりである

Fungus: https://github.com/snozbot/fungus/releases/tag/v3.13.7

Standard Assets: https://assetstore.unity.com/packages/essentials/asset-packs/standard-assets-2018-4-check-out-starter-assets-first-person-thi-32351

ZXing.Net: https://github.com/micjahn/ZXing.Net/tree/master

2.3　ゲームの流れ

本アプリケーションを起動するとタイトル画面が表示される。タイトル画面をタップすると問題数を選択する画面が出てくる。問題数を選ぶとゲームが開始され、背景にタブレットのカメラの映像が流れる。その後すぐにゲームのナビゲーターが登場して説明が始まる。説明が終わるとQRコードの読み込みが開始されてQRコードを読み込むことでクイズが始まる。問題は問題プールの中からランダムに選ばれる。選択肢の中から答えを選ぶと正解か不正解かが表示されてその後解説文が出る。全てのクイズに答えると成績が表示されて正答率に応じたメッセージが表示される。

2.4　開発上での工夫

今回のゲーム開発にあたって、プログラムのフォーマット化や、よりプレイヤーに寄り添うための工夫を行った。その中のいくつかを紹介していく。

2.4.1 問題出題のプログラムのフォーマット化

当初は図 1 の様に Google スプレッドシートを用いて問題を管理し、図 2 の様に Fungus の機能を使って問題の表示、正解・不正解の判断をするようにしていた。

大問番号	小問番号	問題	選択肢1	選択肢2	選択肢3	正解	
1	1	開闢丸はどの国で建造された船？	アメリカ	オランダ	イギリス	2	答えはオランダ。
1	2	開闢丸をオランダに注文したのは？	徳川幕府	明治政府	松前藩	1	答えは江戸幕府。江戸時代終わりの14代
1	3	バウスピリットを含めた開闢丸の乗人員は何m？	51.2m	81.2m	121.2m	2	答えは81.2m。バウスピリットは船の先に
1	4	開闢丸の最大幅は何m？	13.04m	20.04m	25.04m	1	答えは13.04m。大型バスの長さは12m。開
1	5	開闢丸にはマスト（帆柱）が何本ある？	1本	2本	3本	3	答えは3本。マストは船から見ると上下
1	6	開闢丸のエンジンは何馬力ある？	200馬力	400馬力	600馬力	2	答えは400馬力。400馬力はだいたい大きな
1	7	開闢丸の格納式砲では何門の大砲が載っていた？	4門	16門	26門	3	答えは26門。開闢丸の大砲は船の中が空
1	8	「16サンチシルップ砲」の弾はどれぐらい飛ぶ？	1530m	2490m	3990m	3	答えは3990m。だいたい4kmぉ。ここか

図 1 問題の詳細をまとめたスプレッドシート

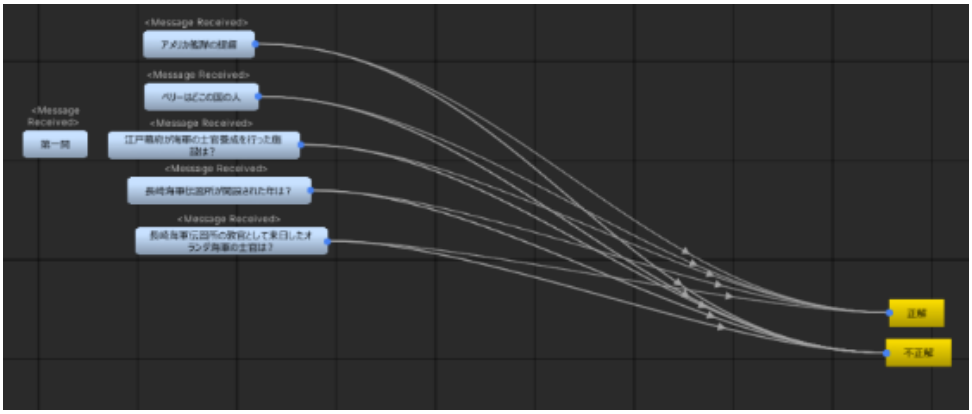


図 2 Fungus による問題構造の表現

しかし、総問題数が増えるにつれてこの方式では問題の管理が難しくなることが分かった。そこでこれらの問題を改善するために工夫を凝らすことにした。

まず、Google スプレッドシートの内容を図 3 のようにソースコード内で 3 次元配列により管理することにした。

図 3 スプレッドシートの内容を配列にまとめたもの

その結果、 図 4 の様に問題文や選択肢の文章の出力、正誤判定等をソースコードで出来るようになり、図 5 の様に Fungus の画面が整理され見やすくなったので、メンテナンス性が向上した。

```
public void Question(){
    GameManager.question++;
    GameManager.q_Kaiyo();
    Debug.Log(GameManager.distination);
    flowchart.SetStringVariable("menu1", data[GameManager.distination - 1][GameManager.Question_Kaiyo - 1][1]);
    flowchart.SetStringVariable("menu2", data[GameManager.distination - 1][GameManager.Question_Kaiyo - 1][2]);
    flowchart.SetStringVariable("menu3", data[GameManager.distination - 1][GameManager.Question_Kaiyo - 1][3]);
    flowchart.SetStringVariable("quote", data[GameManager.distination - 1][GameManager.Question_Kaiyo - 1][0] +
    flowchart.SendFungusMessage("Question");
}
```

図 4 問題文や選択肢の文章などを出力するプログラム

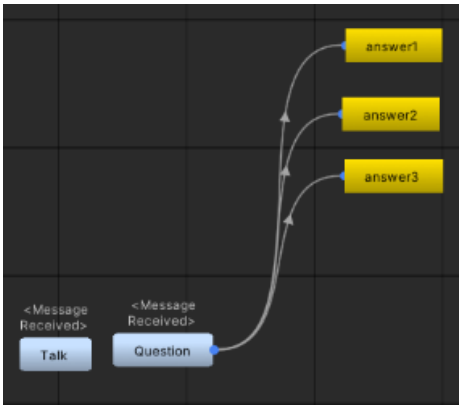


図 5 整理された Fungus の画面

2.4.2 QR コードの読み取りについて

当初は、参考文献[1]のコードを参考にして毎フレーム毎に QR コードの読み取りを行うという処理を行っていたが、それでは、ゲーム実機の処理が追い付かず、プレイに支障が出てしまうほど動作が重くなってしまった。

そこで、図 6 の様に、QR コードの読み取り頻度を 0.5 秒毎にすることで、処理の負担を減らして動作の向上を図った。

```
else if(flame > 0.5f)
{
    string text = Read(m_webCamTexture);
    flame = 0.0f;

    if (text.Equals("Voorlichter_Kaiyomaru_01") && GameManager.distination == 0)
    {
        GameManager.distination = 1;
        GameManager.question = 0;
        flowchart.SetStringVariable("quote", "第一問！");
        flowchart.SendFungusMessage("Talk");
        GameManager.picture_number = 1;
        Debug.Log(text);
    }
```

図 6 QR コードの読み込み速度を制御するプログラム

3.　実証実験

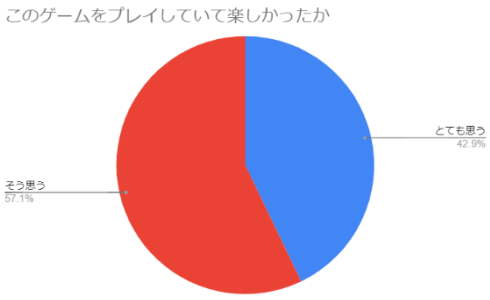
3.1　β版展示

2023年8月19日にβ版を開陽丸記念館にて展示し、14名の被験者に対してアンケートとインタビューを行った。

3.2　アンケート結果

アンケートは3つの項目に対して行った。

(1) このゲームをプレイしていて楽しかったか

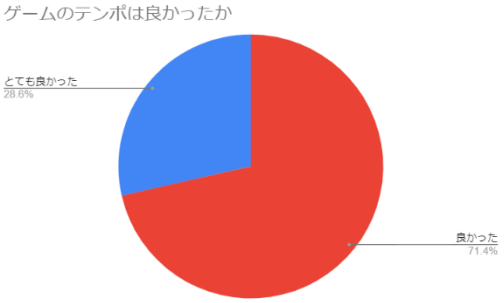


とてもそう思う ： 42.9%

そう思う ： 57.1 %

ゲームをプレイしながら記念館内を周ることによって、より深く展示について学べたという人が多くみられた。

(2) ゲームのテンポは良かったか

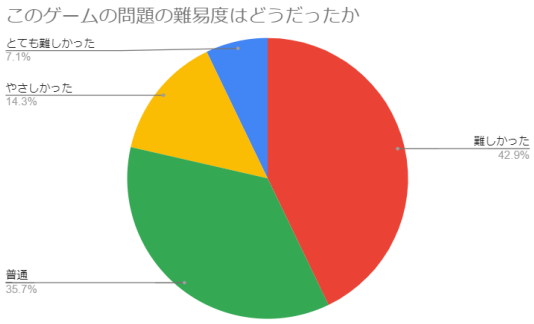


とても良かった ： 28.6%

良かった ： 71.4%

ゲームのテンポは良く、一問が早く終わるのでサクサク進められたという人が多くみられた。

(3)このゲームの難易度はどうだったか



とても難しかった ： 7.1%

難しかった ： 42.9%

普通 ： 35.7%

やさしかった ： 14.3 %

問題の難易度については、人によって意見が違ったが、多くの人が難しかったと答えた。

3.3　インタビュー結果

プラスの意見:

・クイズの後の解説がわかりやすかった。

・UI がシンプルだった。

マイナスの意見:

・総問題数が多く、プレイ時間が長くなってしまった。

・問題の一部に、展示パネルに答えがない質問があったので戸惑ってしまった。

・そもそもゲームに慣れていない方は、プレイすること自体に戸惑ってしまっていた。

4.　実証実験を受けた改善

実証実験を通して課題点を洗い出すことが出来た。特に、プレイ時間や、問題内容についての不満が多くみられたので、リリースに向けて修正を行うことにした。

4.1　総問題数についての工夫

出題される問題の数について、多いと感じる人が一部見られた。そこで、図 7 の様に総問題数を選択できるようにした。

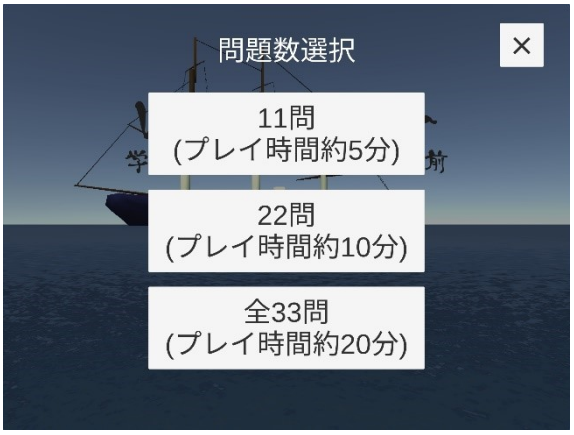


図 7 総問題数選択画面

これにより、プレイ時間に対する様々なニーズに応え、より多くの人にプレイして頂けられると考えられる。

4.2　展示パネルに答えのない問題についての工夫

展示パネルに答えのない問題が一部あるが、それについてのヒントが何もなく、当てずっぽうで解くしかなかったとの意見が多くみられた。そこで図 8 の様にヒントを見られるようにし、誰でもクイズに答えられる仕様にした。

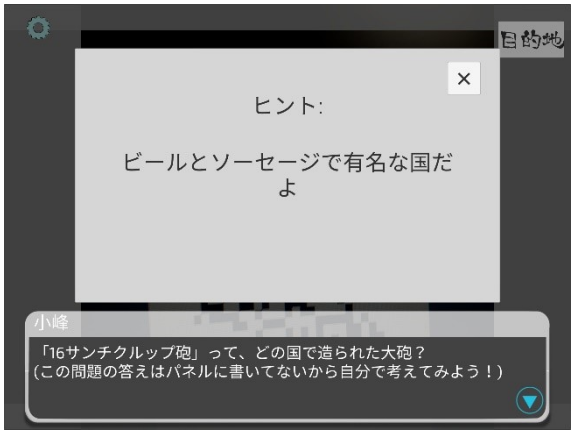


図 8 ヒントの画面

5.　課題と今後について

問題のレパートリーが乏しく、日付の問題に偏っているという課題があるので、パネルに答えがない問題などを増やし、レパートリーを豊富にする。また、ナビゲーターの説明や UI を直感的に理解できるものにし、より親しみを持っていただくようにする。今後については以下の対応を考えている。

・Unity WebGL を活用し、来館者の方々の端末上でも手軽にプレイできるようにする
・問題文等を簡単に改変できるようにし、様々な施設で使えるようにする。

謝辞

本研究は、開陽丸記念館の関係者の方々や、ご指導してくださいました東海林智也准教授の協力無しでは成し得ませんでした。問題作成や開発に当たり、ご協力して下さいました関係者の方々に感謝いたします。

参考文献

[1] Android で QR コードからデータを読み取る方法:

https://baba-s.hatenablog.com/entry/2019/11/25/100000