# 公立はこだて未来大学 2021 年度 システム情報科学実習
# グループ報告書

**Future University-Hakodate 2021 System Information Science Practice**

**Group Report**

## プロジェクト名

Interaction Elements - 『未来を形作る部品』を作ろう

## Project Name

Interaction Elements - Creating Elements for Future

## グループ名

グループ A,B,C

## Group Name

Group A,B,C

**プロジェクト番号/Project No.**

11

## プロジェクトリーダー/Project Leader

赤塚一輝　Kazuki Akatsuka

## グループリーダー/Group Leader

老沼響　Hibiki Oinuma
兒島涼太　Ryota Kojima
和田颯平　Souhei Wada

## グループメンバー/Group Member

赤塚一輝　Kazuki Akatsuka
老沼響　Hibiki Oinuma
中村ももこ　Momoko Nakamura
兒島涼太　Ryota Kojima
専徒礼樹　Raiki Sento
佐々木皓大　Kodai Sasaki
和田颯平　Souhei Wada
古町昂大　Kodai Furumachi
家山剣　Tsurugi Ieyama
丹野夏海　Natsumi Tanno

## 指導教員

安井重哉　塚田浩二

## Advisor

Shigeya Yasui　Koji Tsukada

## 提出日/Date of Submission

2022 年 1 月 19 日/ Janualy 19, 2022

# 概要

Interaction Elements とは、人が外界の環境（身の回りの実世界や、コンピュータの中の仮想世界など）とインタラクションを行う際に用いる要素のことである。本プロジェクトは、今までにはなかった、未来を形作る Interaction Elements を製作することを目的とし、活動した。今年度は最終成果物として、床に置くことで人とエスカレータや自動ドアなどの機械との接点となる「ゆーたん」、手に持つことで行きたい方向を視覚と触覚で認識させる「bect」、本の探索に新たなフィードバックを与える「POP UP SHELF」を製作した。1 チーム当たり 3 から 4 人で構成される 3 チームで行い、オンラインツールを駆使して相互交流を積極的に行いながら製作を行った。

（※文責: 赤塚一輝）

# Abstract

Interaction Elements are the elements that people use to interact with the external environment (such as the real world around them or the virtual world inside a computer). The purpose of this project was to create Interaction Elements that have never been seen before and will shape the future. This year, we produced the following final products: "Yutan," which can be placed on the floor to serve as a point of contact between people and machines such as escalators and automatic doors; "bect," which can be held in the hand to provide visual and tactile recognition of the direction one wants to go; and "POP UP SHELF," which provides new feedback when exploring books. Three teams of three to four people per team worked on the project, interacting with each other and using online tools.

（※文責: 赤塚一輝）

# 目次

**第 1 章　はじめに　1**
1.1　プロジェクトの目標・目的 . . . . . 1
1.2　方法 . . . . . 1
1.3　オンラインの利活用 . . . . . 4

**第 2 章　Element 制作　7**
2.1　Element.01 - ゆーたん . . . . . 7
2.1.1　Element の目標・目的 . . . . . 7
2.1.2　Element 製作手順・方法 . . . . . 7
2.1.3　プログラムの解説 . . . . . 13
2.2　Element.02 - bect . . . . . 16
2.2.1　Element の目標・目的 . . . . . 16
2.2.2　Element 製作手順・方法 . . . . . 17
2.3　Element.03 - POP UP SHELF . . . . . 31
2.3.1　Element の目標・目的 . . . . . 31
2.3.2　Element 製作手順・方法 . . . . . 31
2.3.3　プログラムの解説 . . . . . 43

**第 3 章　中間発表会　48**
3.1　Element.01 - ゆーたん . . . . . 48
3.1.1　中間発表会に向けて . . . . . 48
3.1.2　中間発表会でのフィードバックを受けて . . . . . 49
3.2　Element.02 - bect . . . . . 50
3.2.1　中間発表会に向けて . . . . . 50
3.2.2　中間発表会でのフィードバックを受けて . . . . . 50
3.3　Element.03 - POP UP SHELF . . . . . 52
3.3.1　中間発表会に向けて . . . . . 52
3.3.2　中間発表会でのフィードバックを受けて . . . . . 52
3.4　Web サイト・ポスター製作 . . . . . 53
3.4.1　Web サイト製作 . . . . . 53
3.4.2　ポスター製作 . . . . . 55

**第 4 章　成果発表会　56**
4.1　Element.01 - ゆーたん . . . . . 56
4.1.1　成果発表会に向けて . . . . . 56
4.1.2　成果発表会でのフィードバックを受けて . . . . . 58
4.2　Element.02 - bect . . . . . 60
4.2.1　成果発表会に向けて . . . . . 60

4.2.2 成果発表会でのフィードバックを受けて . . . . . 60
4.3 Element.03 - POP UP SHELF . . . . . 62
4.3.1 成果発表会に向けて . . . . . 62
4.3.2 成果発表会でのフィードバックを受けて . . . . . 63
4.4 Web サイト・ポスター製作 . . . . . 64
4.4.1 Web サイト製作 . . . . . 65
4.4.2 ポスター製作 . . . . . 71
4.4.3 動画製作 . . . . . 71

**第 5 章 おわりに** **74**
5.1 グループのまとめ . . . . . 74
5.1.1 Element.01 - ゆーたん . . . . . 74
5.1.2 Element.02 - bect . . . . . 74
5.1.3 Element.03 - POP UP SHELF . . . . . 74
5.2 プロジェクトのまとめ . . . . . 75

**付録 A 成果発表会で使用したポスター** **76**

**付録 B 成果発表会に向けて製作した Web サイト** **77**

**付録 C ソースコード** **85**
C.1 Element.01 - ゆーたん . . . . . 85
C.2 Element.02 - bect . . . . . 91
C.3 Element.03 - POP UP SHELF . . . . . 96

**参考文献** **118**

# 第 1 章　はじめに

## 1.1　プロジェクトの目標・目的

Interaction Elements とは、人が外界の環境（身の回りの実世界や、コンピュータの中の仮想世界など）とインタラクションを行う際に用いる要素のことである。例えば、照明のスイッチが一例であり、身近には様々な Interaction Elements が存在する。本プロジェクトは、今までにはなかった、未来を形作る Interaction Elements を製作することを目的としている。（以下、本稿では製作物を「Element」と呼称する。）

今年度は、床に置くことで人とエスカレータや自動ドアなどの機械との接点となる「ゆーたん」、手に持つことで行きたい方向を視覚と触覚で認識させる「bect」、本の探索に新たなフィードバックを与える「POP UP SHELF」を製作した。

（※文責: 赤塚一輝）

## 1.2　方法

**組織**

プロジェクト構成員は学生 10 名と教員 2 名である。

意見を積極的に交換するため、できる限りフラットな関係を目指した。組織構造はマトリクス組織に近いものとし、役職として、プロジェクトリーダー・副リーダー・プロジェクトマネージャー・グループリーダー・プレゼンテーションリーダー・アートディレクター・Web コーディング・ポスター製作・アイスブレイクなど多種多様な役職を設置した。

主なグループは実際に Element 製作を行う 3 グループであり、このグループには必ず全員がいずれか 1 つのグループに所属している。中間・成果発表会で用いる Web サイト・ポスター製作や当日のメンバー役割・段取りを考えるプレゼンテーショングループでは、自身がやりたいと思える役職に就き、役割を全うした。役職は話し合いを行い、お互いに合意したうえで決定した。また、プレゼンテーショングループでは各 Element 製作グループと連絡を取り、最善の表現方法を探りながら進めた。1 人が複数のグループに属することで全体の進捗を把握でき、自主的に動ける組織とした。

（※文責: 赤塚一輝）

以下の手順でプロジェクト活動を行った。行った手順に関して、詳しく記述する。

**手順 1 前置詞図鑑の作成（アイデア出しの練習）**

Interaction Elements の素となるアイデアを身の回りから探す練習として、英語の前置詞を使い、前置詞図鑑を作成した。前置詞図鑑には「表される前置詞」、「物の名前」、「物の意味」、「物の要素」、「五感で感じられること」 、「オノマトペ」を記載した。それをプロジェクトメンバー全体

に発表することにより、メンバーそれぞれの価値観や視座・視点を共有し、アイデアの幅を広げることを目指した。プロジェクトメンバーがそれぞれ家の外に出るなどして、身の回りを観察し、1人 10 個以上の前置詞図鑑を作成・発表した。最終的に合計 100 個以上の前置詞図鑑を作成した。

図 1.1　前置詞図鑑の一部

（※文責: 赤塚一輝）

**手順 2 具体的なアイデアの創出**

自分または他人が作成した前置詞図鑑や身の回り、フィールドワークなどから得た新たなアイデアを参考にし、1 人 10 個以上の新たなアイデアを持ち寄り、合計 100 個以上のアイデアからなる Interaction Elements のネタ帳を作成した。プロジェクトメンバーにアイデアを共有する際には、目的や機能を説明するとともに絵や図解を用いることによって、ひとりひとりが創造したアイデアをより相手に伝わりやすいよう、図や動画といった形で共有した。

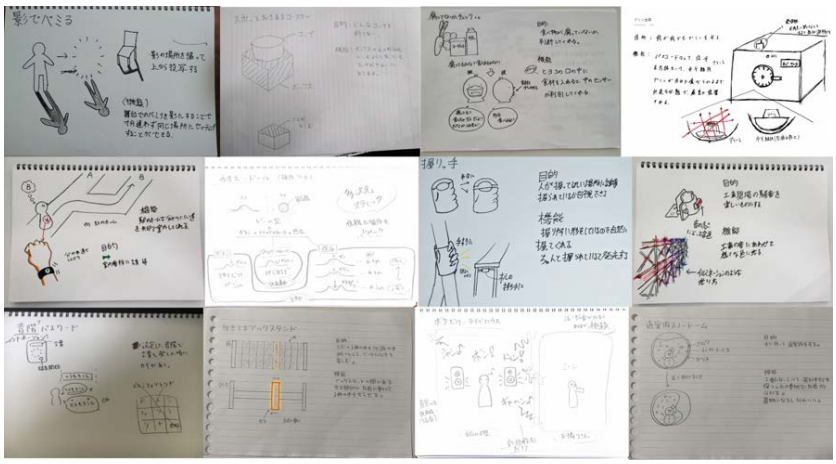


図 1.2　ネタ帳の一部

（※文責: 赤塚一輝）

**手順 3 アイデアの整理・グループ分け**

オンラインホワイトボード Miro を用いて、Interaction Elements のネタ帳の 100 個以上のアイデアを KJ 法により整理・分類した。集めたアイデア同士の関連性・類似性の有無を考え、似ている要素を見つけ出し分類を行った。アイデアを体系的にまとめることで、より多面的に日常を捉え、さらに新しいアイデアを着想する手助けとした。また、その整理したアイデアを素に、自身が

取り組みたいアイデアの分類を選び、分類の近い者同士を集め、10 人のグループメンバーを 3 人・3 人・4 人の合計 3 つのグループに分けた。

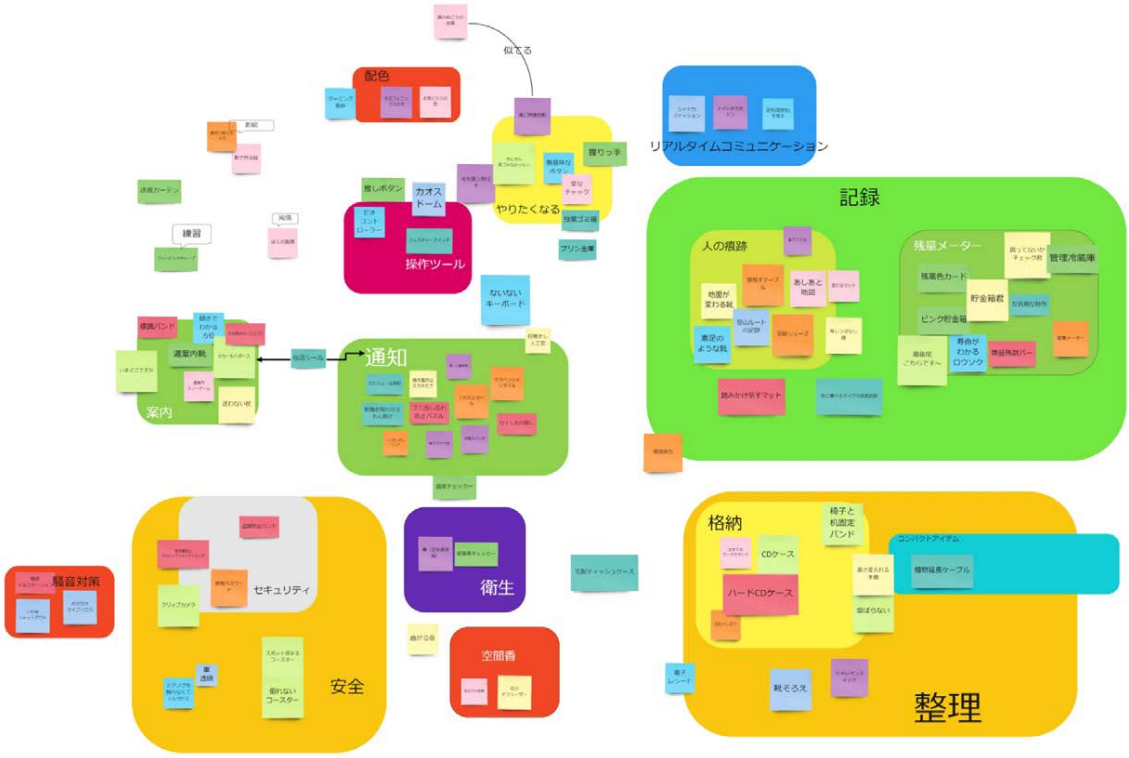



図 1.3　アイデアの整理後

（※文責: 赤塚一輝）

**手順 4 グループごとに Element 製作**

**前期**

定期ミーティング時間の前半を Zoom のブレイクアウトルームを用い、グループごとでの話し合い・製作の時間とし、後半をプロジェクト全体での発表、学生・先生間での意見交換の時間とした。全体での発表ではグループで考えていることが伝わりやすくなるよう、図などを用いて発表した。全体での発表時間を確保することで、所属グループ以外の内容も把握しやすくし、多くの意見を取り入れながら製作を行える仕組みとした。学校で行う必要のある加工などについては、グループごとに交代で登校し、製作を進めた。

図 1.4　会議の様子

**後期**

前期より Element 製作が進んだため、定期ミーティングでの全体情報共有の時間を 2 回のミーティングにつき 1 回の頻度へと変更した。全体情報共有時間を減らし、製作時間を増やすことで、製作に十分な時間を確保できる仕組みへと変更した。また、コロナ禍における規制が緩和されたため、大学内でのミーティング、オンラインミーティングを柔軟に交え、製作に重点を置きながら、全体情報共有も積極的に行った。

（※文責: 赤塚一輝）

**手順 5 中間・成果発表会資料の作成**

中間・成果発表会に向けて、Web サイト・動画・ポスターを用意した。全体統括、デザイン担当、Web サイトのプログラミング担当と、担当分けを行った。Web サイトは GitHub Pages を用いて公開した。ポスターは Adobe Illustrator を使って製作した。(詳細は 「Web サイト・ポスター制作」にて記述する) プロジェクトの認知度を上げ、より多くの質問や意見をもらうためにロゴの製作も行った。

（※文責: 赤塚一輝）

## 1.3　オンラインの利活用

本プロジェクトでは、プロジェクト活動を円滑に進めるためのデジタルツールとして、Web 会議サービス Zoom、ビデオ通話・音声通話ソフトウェア Discord、ドキュメント管理ツール Scrapbox、クラウドストレージ Google ドライブ、オンラインホワイトボード Miro、時間管理ウェブアプリケーション Google カレンダーを利用した。

（※文責: 赤塚一輝）

**Zoom**

週 2 回 (水曜日・金曜日) の 14:50〜18:00 に行う定期ミーティングにおいて Zoom を利用した。ミーティングの設定はプロジェクトリーダーが行い、Zoom の定期的なミーティングの機能を使い毎回ミーティングを設定する手間を省いた。チャットの履歴を保存しておく設定にしておくことで、チャットに重要なことが書かれていて後から思い出せなかったとしてもログを辿ることができるようにした。グループごとに作業する際にはブレイクアウトルームを設け、グループの数より多いルームを用意、自由にルーム間を移動できるようにし、各々のグループが自由な運用を行えるようにした。

（※文責: 赤塚一輝）

**Discord**

全体連絡・プロジェクト時間外のミーティングに Discord を利用した。テキストチャットツールとして Slack が有名であるが、今回は前述のとおりフラットな関係となることを目的にし、さらに、プロジェクト時間外にも簡単にミーティングを行うことができることから Discord を利用した。

連絡を適切な人に確実に伝達するため、Discord のロール割り当て機能を用い、メンバーにロール割り当てを行った。例えば、学生にメッセージを送るときは「@学生」、先生にメッセージを送るときは「@先生」のようにした。この他にも Element 製作を行う各グループのロール、プレゼンテーションチームのロール、各役職についてロールを用意し、割り当てた。

後から見たときに情報を辿りやすくするため、Discord のチャンネル・カテゴリ分け機能も活用した。全体連絡のカテゴリ、雑談のカテゴリ、Element 製作グループごとのカテゴリ、プレゼンテーショングループのカテゴリ、チャンネルアーカイブ用のカテゴリを設けた。全体連絡のカテゴリでは、メインとなる全体連絡チャンネルの他に、当日・これからやることを記述しておく「やることリスト」「議題」チャンネル、Zoom ミーティングの URL や Google ドライブの URL など、頻繁に利用する URL リンクを置いておく「リンク先」チャンネル、コロナ禍における登校制限に対応するための、登校者の情報を投稿するチャンネル、プロジェクトでの必要物品購入申請スプレッドシートを更新したことを伝えるチャンネルなどを用意した。Element 製作グループごとのカテゴリでは、「会議中のメモ」、「グループで行うこと」、といったチャンネルを用意した。チャンネルの作成と編集は全員に権限を付与し、自由に活用できるようにした。プレゼンテーショングループのカテゴリでは、Web サイト・ポスターについてのチャンネルや掲載画像、中間・成果発表会中の連絡用チャンネルを用意した。その他に直接的に製作には関係のない、雑談のカテゴリを用意することで、プロジェクトメンバー同士の交流を深め、連絡もこまめになり、結果的に重要な情報を見逃すことを減らすことに役立った。

（※文責: 赤塚一輝）

**Scrapbox**

ドキュメント共有に Scrapbox を利用した。

Scrapbox は主に会議の議事録として利用した。議事録には事前に知っておいてほしい前提情報、会議までに考えておいてほしいこと、会議のゴールと決定事項を記述した。議事録作成は担当をつけて行った。グループごとの作業においても、議事録の作成を行った。前述した前置詞図鑑、

Interaction Elements のネタ帳も Srapbox を使ってメンバー同士のアイデアを交換した。アイデア交換の際には、ページ下部に各メンバーがコメントを残すようにした。

（※文責: 赤塚一輝）

**Google ドライブ**

主に動画や画像のファイル共有に Google ドライブを利用した。

プロジェクトでの必要物品購入申請スプレッドシートは、グループ名・品物名・個数・購入物 URL・発注状況・予算残高を記入し、新しく追加したときには先生に Discord でメンションをつけて送ることで管理した。Web サイト・ポスター・報告書に掲載する画像や動画の素材はそれぞれにおいて Google ドライブでフォルダを作成し、メンバー間で共有した。Web サイト・ポスター・報告書に掲載する文章についても Google ドキュメントのコメント機能を用い、全員がコメントで修正を行えるようにした。

（※文責: 赤塚一輝）

**Miro**

アイデアの共有・整理に Miro を利用した。

Interaction Elements のネタ帳の 100 個以上のアイデアを KJ 法により整理・分類する際にはそれぞれのアイデアを付箋として全員分を書き出した。付箋はメンバー毎に色を決め、どの付箋が誰のアイデアかわかりやすくした。ある程度類似したアイデアの付箋が集まれば、その一帯を色のついた四角形で囲みグループ分けをし、どれにも当てはまらないようなものは矢印やテキストを使って分類した。最終的に、自身の興味のある範囲にマウスカーソルを持っていき、近い人同士でグループを組んだ。

（※文責: 赤塚一輝）

**Google カレンダー**

スケジュール管理に Google カレンダーを利用した。

Element 製作や報告書の締め切り日やその道中のマイルストーンとなる日を決め Google カレンダーに記載した。その他にも学校施設である工房を利用する頻度が高いので、重要な情報である、長期休暇中の工房が使えない期間なども記載した。

（※文責: 赤塚一輝）

# 第 2 章　Element 制作

## 2.1　Element.01 - ゆーたん

### 2.1.1　Element の目標・目的

本 Element の目的は、電車、エレベータ、自動ドアなどの出入口や、エスカレータ前、歩道などの床に設置することで人を自然に誘導することである。

この Element を使用することで、自動ドアが開閉するタイミングをより分かりやすく、自分の乗るべき段を見定めるのが苦手で、エスカレータを使えない・使うことが苦手な人にとっては乗りやすく、より自然と誘導されることを目指した。

ここから、本 Element の目標は、Element がエスカレータなどの対象物の動きに合わせて光ることで、対象物の動きが視覚的により分かりやすくすることとした。

（※文責: 中村ももこ）

### 2.1.2　Element 製作手順・方法

#### 方針の決定

本グループは人を誘導するものに関連する Element を製作したいメンバーが集まり進めていった。はじめに、どういった製作物を作るか決めるためにアイデア出しを行い、日常にある、さりげない動作から人を誘導することができないか考えた。なんとなく握りたくなってしまう取っ手や、蹴ることで道案内をしてくれるデバイス、エスカレータを乗りやすく誘導する床などの案が出た。ここから、蹴ることで道案内をしてくれるデバイス、エスカレータを乗りやすく誘導する床などの案を採用しこの 2 つの Element を製作することに決定した。

（※文責: 中村ももこ）

#### 実装方法の検討

まず、蹴ることで道案内をしてくれるデバイスは、小学生の頃の登下校でやってしまう石を蹴って帰るという実体験から発案された。この石蹴りをして登下校をするという行動は、メンバー内でも全員が行ったことのあることであった。登下校の毎日変わらない道で暇を持て余してしまうことからこの石蹴りという行動をやってしまうと考える。石蹴りをすると、石は自分より前方に転がっていき、時にはあらぬ方向に転がってしまうこともある。この、「石が自分より前方に行くこと」「石が真っ直ぐ転がるだけでなく曲がって転がること」に着目し、自分が蹴った石が設定した場所に道案内してくれるような Element の製作を行うことにした。

この Element を製作するにあたって、下記の 3 つに注意して検討を進めた。

- Element 自体が自動的に方向転換を行い転がって止まること
- Element が足で蹴ったことを感知すること
- 位置情報の取得

目的地まで行く方法として、いくつかチェックポイントを定めておき、蹴ったことを感知するとチェックポイントまで動くことで目的地までいくことを考えた。（図 2.1）ロボットボール「Sphero bolt」を使うことで製作に取り掛かった。「Sphero bolt」とは、アプリ対応のロボットボールであり、Javascript でプログラムすることによってボールの制御を行うことが可能である[*1]。これには、赤外線通信、デジタルコンパス、光センサー、ジャイロスコープ、加速度計、モーターエンコーダが搭載されており、私たちが念頭においていた考えが揃っていたため「Sphero bolt」を購入し、手元でこの製品のみで何ができるのかを調べた。その結果、Sphero bolt 同士、Sphero bolt と SpheroEdu というアプリケーションを動かすことのできるデバイスとの通信は容易に可能であった。これより、3 つの条件のうちの「Element 自体が自動的に方向転換を行い転がって止まること」「足で蹴ったことを感知すること」は可能であると考えた。次に、位置情報の取得は、Sphero bolt に GPS は搭載されておらず、Sphero bolt のみでの制御は不可能であると考えた。そこで、M5Stack を用いて、Sphero bolt と M5Stack で通信を行い、さらにもう 1 つ M5Stack を用いて M5Stack 同士で通信を行い、2 つ目の M5Stack で位置情報を得ることで Sphero bolt の制御が可能であると考えた。これを実現させるためには、位置情報の正確さが必要であった。この位置情報を得る方法として、2 つの案があった。

1 つはビーコンを用いる方法でもう 1 つは Wi-Fi を用いる方法である。この 2 つを比べたとき、ビーコンを使う方法が正確であると判断したため、ビーコンを用いて位置情報の取得を行うこととした。この仕組みを実現するためには 2 つの M5Stack が必要であったが、Sphero bolt にこの 2 つを埋め込むことは仕組み上不可能であった。そこで、Sphero bolt の上から何かしらのカバーを被せ、Element を 2 重構造にすることによってこの問題を解決しようとした。カバーの案として、ゴムボールのような柔らかい素材や、プラスチックや木材のような硬い素材が挙げられた。柔らかい素材は、石を蹴っているよりもサッカーボールなどのようなものを蹴っているような感覚になるのと、カバーの下の M5Stack や Sphero bolt の耐久性の問題から不採用となった。硬い素材であれば、周りからの衝撃に耐えることが可能であると考えられる。また、硬い素材にすることで石の硬さに近づけることが可能である。プラスチックや木材などの硬い素材を用いてカバーを製作することとした。これを最終案として、「iSi」と名付けた。iSi の由来は意思を持っているような石という 2 つの「いし（意思・石)」という言葉からこの名前になった。

次に、エスカレータを乗りやすく誘導する床は、メンバーのエスカレータに乗れないという実体験から発案された。エスカレータに乗るのが苦手な大きな原因として考えられるのが、エスカレータに乗るタイミングの掴みづらさである。エスカレータに乗るときは、動くエスカレータの速さと自分の歩く速さを考慮して、自分が乗るべき段を定めなければならない。その段の出てくるタイミングの掴みづらさが、エスカレータに乗りづらいと感じることにつながると考えた。そこで、エスカレータ前の床に自分が乗るべき段を床に表示することで、エスカレータに乗る誘導が行える Element の製作を行うことにした。この Element を製作するにあたって、下記の 3 つに注意して検討を進めた。

- 光で誘導すること
- エスカレータの動きに合わせて光が動くこと
- 体重をかけても影響がないこと

ここから、製作物の案として、プロジェクションマッピングで上から床に直接投影するものや、

[*1] Sphero edu https://sphero-edu.jp/teaching/bolt

光る床をエスカレータの前に設置する案が考えられた。プロジェクションマッピングの方法では、上から投影した場合、エスカレータに乗る人の足や体などで影になってしまうことや、上や横などにプロジェクターを置く影にならないような設置の方法ではプロジェクターが複数台必要となってしまい、設置が大掛かりで難しくなってしまうことから、光る床の方針で進めた。光る床の案では、LED をエスカレータ前の床に直接埋め込むことが考えられた。しかし、床に直接 LED を埋め込む場合では多くの LED を用意しなければ、まとまった光に見えないことと、多くの LED の制御は複雑で難しいと判断した。そこで、透明のアクリルに傷を付けて横から LED の光を入射し、光を屈折させる導光板と、屈折した光を全体に拡散させる乳白色のアクリル (拡散板) を使うことに着目した。この 2 枚の板を重ねたものに横から LED で光を入れると、拡散板側が全体的に淡く光るようになる。これにより、少ない LED の光のみで、光る床を表現できると考えた。また、LED ではなく LED テープを使うことで加工が簡単になり、LED の制御も Arduino の Adafruit NeoPixel Library を使うことで、LED 一つ一つの光り方と色の制御が可能になる。これに強度面の強化と、導光板と拡散板を重ねた高さと LED テープの幅を合わせるために、MDF を下に重ねた。(図 2.2) これを最終案として、「ゆーたん」と名付けた。ゆーたんの由来は、「床」に置く、あなたを意味する「You」、誘導する「誘」と、機構の見た目がじゅうたんに似ていることからじゅうたんの「たん」を取った。

アイデア出しの時点では、この 2 つの Element を製作することとした。しかし、グループの人数が 3 名であり、プロジェクト期間中に iSi を完成させることができる算段が立たなかったことから iSi の製作を断念し、議論がより具体的に進んでいたゆーたんのみ製作を進めていくこととした。

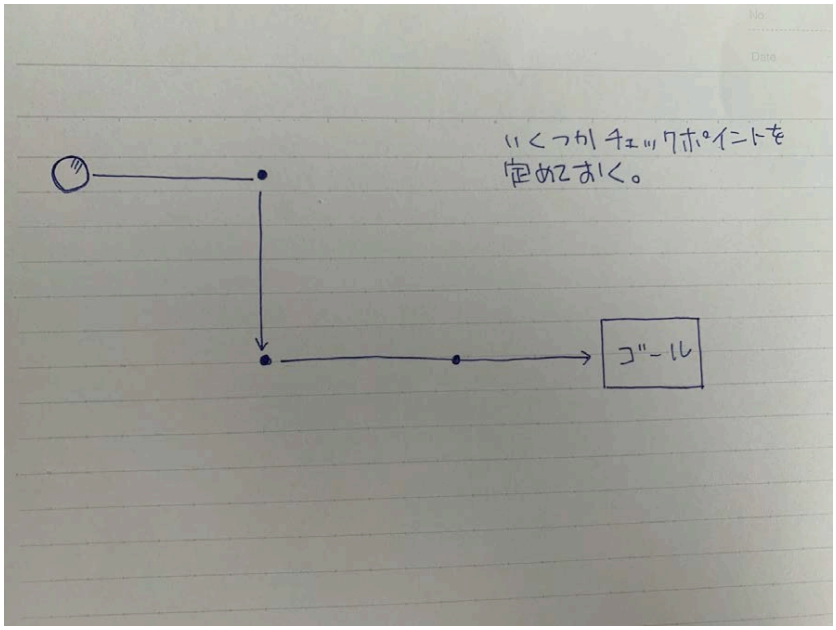



図 2.1 「iSi」の動作方法

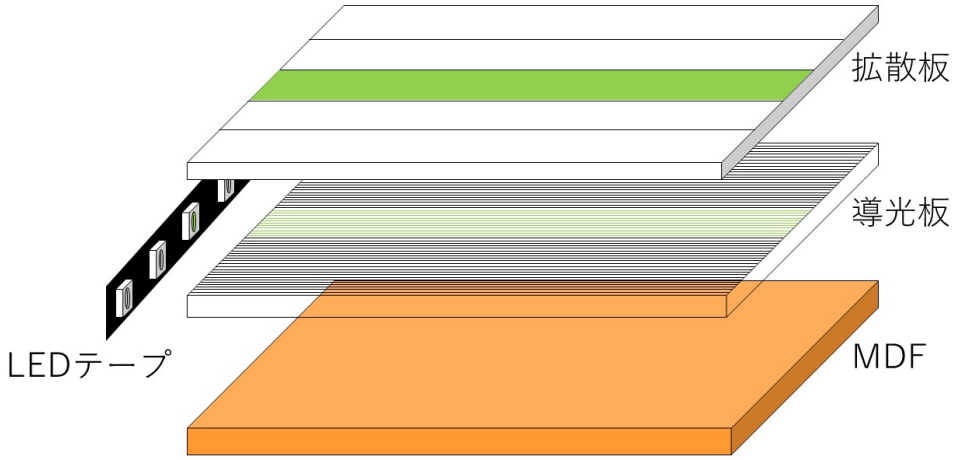



図 2.2　ゆーたんの構造イメージ

（※文責: 中村ももこ）

**ミニチュア模型の製作**

前期の製作では、ゆーたんの 120mm × 100mm サイズのミニチュア模型を製作した。それにあたって、導光板の傷の種類の考案、LED の制御を行った。導光板は透明なアクリルに傷をつけることによって製作することができる。そこで、透明なアクリルに彫刻する傷によってどのように光が変化するのか検証を行った。大まかなパターンとしては、ドット柄、ストライプ柄、ボーダー柄を彫刻した。（図 2.3）その中でもドットの直径の大きさや間隔、彫刻の深さ、ストライプ柄やボーダー柄の線の間隔や彫刻の深さを変えてひかり方を検証した。検証する時に見るポイントとしては、LED テープの光が入った場所からの光の減衰の程度、導光板 1 つ全体のムラの程度である。これらは、導光板を全て組み合わせたときの光の綺麗さに繋がるため検証を行った。この結果より、ドット柄を彫刻した物は、ドット柄が拡散板を通しても見えてしまうため、不採用とした。ストライプ柄を彫刻した物は、LED テープから遠ざかると光の減衰が激しかったため不採用とした。ボーダー柄を彫刻した物は、他のものに比べて光の減衰も少なく、拡散板を乗せた状態でも模様が目立つことなく、全体が綺麗に発光した。これらより、ボーダー柄を採用することにした。長辺に沿って平行な細い線を 15 本入れた。120mm × 100mm サイズを製作することで、実寸大のイメージを持ちやすくし、中間発表でも具体的なものを作ることで、最終イメージを伝わりやすくした。

図 2.3　傷のパターンの検証

（※文責: 中村ももこ）

**3 つの問題と解決方法**

後期では、600mm × 990mm サイズの実寸大の製作を中心に活動した。この実寸大のゆーたんを製作するにあたって 16.5mm × 600mm の導光板を製作し、それを 60 個つなげることによって製作することにした。光は 16.5mm の幅の方から入れることとした。しかし、製作を進めていく段階で 3 つの問題が発生した。1 つ目は耐久度の問題である。導光板は 16.5mm × 600mm と、

とても細長いため、足で踏むと真ん中で割れてしまう恐れがあった。2 つ目は光の減衰の問題である。導光板の傷パターンの検証を行った中で最も減衰が少なかったが、遠くなればなればなるほど光の減衰が見られた。3 つ目は、学校にあるレーザーカッターでは 600mm の幅でアクリル板を切ることが難しい問題である。この 3 つの問題から、導光板 1 つの大きさを 16.5mm × 600mm から 16.5mm × 300mm に変更し、ゆーたんの両端から LED テープを設置した。これにより、前述の 3 つの問題は解決できた。

（※文責: 中村ももこ）

**機構の変更**

前述の通り、導光板 1 つの大きさを 16.5mm × 300mm に変更したことによって、そのままの機構ではゆーたんの大きさが小さくなってしまったが、導光板 2 つを横に並べ、導光板の数を 60 個× 2 列幅とし、幅を 600mm のまま進めることができた。しかし、導光板を並べた時に、横に 2 つ並んだ導光板同士の光が干渉してしまっていたため、その間に厚紙を入れることで光の干渉を抑えた。また、導光板を 2 列にしたことにより表現の幅を広げることが可能になった。その 1 つとして、光り方のパターンを増やすことが可能になった。ゆーたんを縦長に見ると、左右で光り方を変えることができるため、足跡のような光り方の表現が可能になったり、左右で違う方向に光が流れるようなパターンを作ることができた。MDF、導光板、拡散板を組み合わせる際に、乳白色アクリル（拡散板）の大きさが 990mm に足りなかったため、導光板の数を 55 個 × 2 列に減らし、結果、ゆーたんの大きさは 600mm × 907.5mm となった。

（※文責: 中村ももこ）

**目標・目的の変化**

ゆーたんの機構を変更したことによって、表現の幅が広がったことから、グループ内で光のパターンなどを再考した際にもっと他に使い道があるのではないかといった議論があった。また、教員からのフィードバックからもエスカレータだけに絞るのはもったいないのではないかというアドバイスがあった。これらの考えやアドバイスから、ここまではエスカレータに乗り易くなることを目的として製作を進めてきたが、前述の [Element の目標・目的] に記述している現在の目標・目的となった。これに伴い、光り方のパターンも縦に設置してエスカレータに合わせるもののみではなく、横長に設置して左右に光が流れるようなパターンや左右から光が集まるようなパターンを考えた。このように光の左右の動きは主にエレベータや自動ドアなどに活用することができると考えた。

（※文責: 中村ももこ）

**最終工程**

MDF、導光板、拡散板、LED テープの固定を行った。固定方法として、接着剤とセロハンテープの 2 つの案があった。接着剤での固定は導光板の彫刻に液体が入り、光に干渉してしまうため使用しないこととなった。セロハンテープは接着剤のように導光板の傷を邪魔することなく MDF、導光板、拡散板、LED テープを固定することができたため、セロハンテープを採用することとした。しかし、セロハンテープだけでは耐久性に問題があった。そこで、ホームセンターで強力な透

明なテープを購入し、セロハンテープの上から固定した。しかし、そのままではLEDテープも剥き出しで固定したテープもそのまま見えてしまっており、LEDテープのある箇所を踏んでしまうと破損してしまう恐れがあったため、L字のプラスチック材で補強した。また、この補強材の固定には強力な両面テープをゆーたんのLEDテープのある上面端に貼り付けて固定した。これにより、LEDテープを補強し全体的にまとまった仕上げとなった。(図2.4)

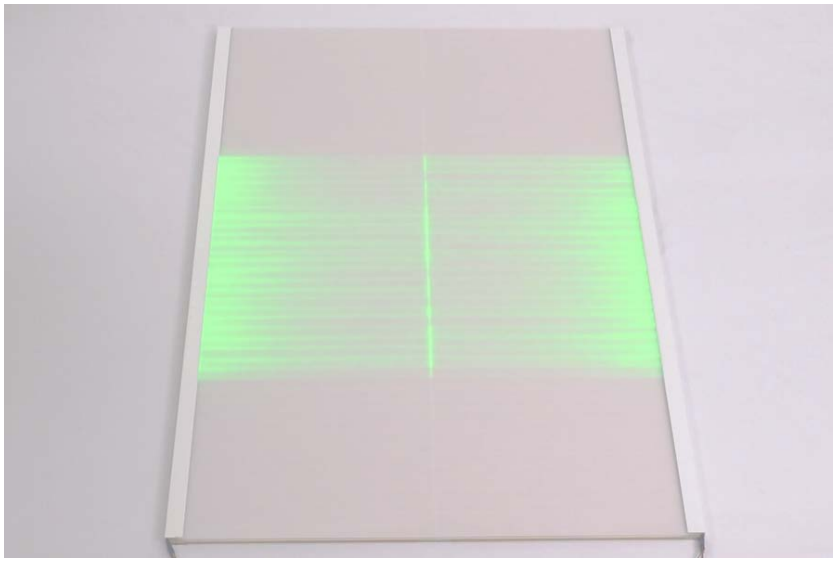


図2.4　ゆーたんの完成形

(※文責: 中村ももこ)

### 2.1.3　プログラムの解説

ゆーたんを制御するプログラムを記述するにあたって、成果発表会の撮影時に使用したエレベータ3パターン分とエスカレータ用のプログラム、そしてそれらを制御するメインプログラムについて解説する。使用した部品はLEDテープ(BTF-LIGHTING WS2812B)、トグルスイッチ(2MS1-T1-B4-M2-Q-E)、タクトスイッチ(TS-0606-F-N-BLK)、可変抵抗器(3386K-EY5-101TR)である。以下、各プログラムについて紹介する。ソースコードについては、付録Cに掲載する。

(※文責: 老沼響)

**main**

まず、各パターンのプログラムを制御するためにメインで記述したコードについて解説する。1行目はNeoPixelライブラリのインポスター文である。NeoPixelライブラリはLEDテープの点灯や色の制御に用いるライブラリで、LEDテープ1つ1つ個別に点灯と色を制御することができる。制御に用いるsetPixelColorは(LED番号, LEDの名前.Color(R, G, B))で制御される。

2〜13行目はLEDの個数とArduinoで使用するピンの設定を行う。2行目ではLEDの個数、4,5行目はLEDテープに信号を送るためのピン、7〜13行目はスイッチからの信号を受け取るためのピンである。

15,16行目はLEDテープを制御するためにled0とled1という変数を宣言する。

18〜23行目はそれぞれint型で取得した値を格納する変数を宣言する。stateはトグルスイッチ

の状態を 0 か 1 で取得する。S は読み取ったトグルスイッチの値からどのパターンを光らせるか識別する。var はタクトスイッチの値を HIGH か LOW で取得する。volume は可変抵抗の値を自然数で取得する。

25〜40 行目は setup 関数の処理が行われる。26〜28 行目では LED テープを使う際に必要な Arduino の内部の設定を行っている。30〜36 行目では Arduino で使用する各ピンの設定 (INPUT / OUTPUT) をそれぞれ行い、各部品と通信できるようにした。また 38,39 行目で led テープを OFF の状態にし初期化している。

42〜79 行目は Loop 関数の処理が行われる。43〜46 行目ではそれぞれ可変抵抗、トグルスイッチの値を読み取っている。48 行目ではトグルスイッチの値からどのパターンなのかを示している。具体的にはトグルスイッチの値が 0 と 1 なのを活かし、3 桁の 2 進数で各位にトグルスイッチを割り当て、0 から 7 の値を出す。

50〜78 行目では switch 文を使って先ほどの S の値からどのパターンの光り方をするか制御している。以下でそれぞれパターンのプログラムを解説する。

（※文責: 老沼響）

**エスカレータパターン 1**

上記のプログラムは、エスカレータパターン 1 のプログラムである。このプログラムは足跡のようにゆーたんが光るプログラムとなる。

1〜3 行目は関数宣言や必要変数の宣言を行っている。ここでは二重配列を使い、ゆーたんを横 2 つと縦 5 つの合計 10 区間に分けて LED テープの制御を行った。二重配列内の値が 0 だと消灯、1 だと点灯となる。

このプログラムは 7 つの光り方の組み合わせでできており、7 つの光り方を区別し繰り返せば良いため、4 行目の for 文の値で区別、繰り返しを行う。ここの値 k でこのプログラムの動きを判別する。

6〜15 行目はそれぞれ二重配列と LED テープのステータスの初期化を行う。

17〜40 行目の switch 文は k の値から LED テープが光る区間の配列の値を 1 にする。

51〜64 行目の for 文ではそれぞれの LED テープの区間ごとで配列の値が 1 かどうかを読み取り、1 であったときに点灯のステータスを代入する。点灯の場合、縦 1 区間 LED が 11 個なので、54,60 行目のように for 文の開始と終了を 11 刻みで行い、特定の区間のみ緑色の点灯ステータスを代入する。

66,67 行目で LED テープに点灯データの送信を行う。

68,69 行目では volume は可変抵抗から値を読み取り、読み取った値から delay の値を変化させることで、LED テープの光の速度を制御している。

（※文責: 老沼響）

**エスカレータパターン 2**

上記のプログラムはエスカレータパターン 2 のプログラムである。このプログラムは 3 つに分かれた区間が固定的に光るプログラムとなっている。

1 行目は関数の宣言、開始を行っている。

このプログラムは 3 つの光り方の組み合わせでできており、3 つの光り方を区別し繰り返せば良

いため、2 行目の for 文の値で区別、繰り返しを行う。ここの値 k でこのプログラムの動きを判別する。

4〜7 行目は LED テープのステータスの初期化を行う。

9〜30 行目の switch 文は k の値から LED テープが光る場所のステータスを点灯にする。今回 LED テープは 55 個ついており 3 等分をすることができない、また綺麗に 3 等分できても歩幅分の長さを確保しなくてはならない。そこでこのプログラムでは歩幅分を 25 個の LED ときめ、それぞれ前と中心と後ろから確保し光らせることとした。11,18,25 行目の for 文でそれぞれの区間を数えあげ、中で LED テープに緑色の点灯ステータスを代入する。

32,33 行目で LED テープに点灯データの送信を行う。

68,69 行目では volume は可変抵抗から値を読み取り、読み取った値から delay の値を変化させることで、LED テープの光の速度を制御している。

（※文責: 老沼響）

**エスカレータパターン 3**

上記のプログラムはエスカレータパターン 3 のプログラムである。このプログラムは Element 全体が前から後ろにかけて流動的に光るプログラムとなっている。

1 行目は関数の宣言、開始を行っている。

このプログラムの構造は前から 1 つずつ LED が光っていき、25 個集まったら塊として後ろに流れていくプログラムとなっている。後ろに流れていった LED はそのまま 1 つずつ消えていく。これを再現するために本来ないはずの LED テープ 25 個分が後ろについていると仮定してプログラムを組んだ。よって 2 行目の for 文の値 k で 80 個分の LED テープを制御する。

4〜7 行目は LED テープのステータスの初期化を行う。

9〜13 行目は 25 個の塊になるまで 1 つずつ後ろに向けて増えていくのを if 文と for 文を使って制御している。具体的には、k の値が 25 未満のときはまだ塊になりきってないので、その間は 1 つずつ後ろ方向に LED テープの緑色の点灯ステータスを増やしていく。

14〜19 行目は 25 個の塊になった LED を後ろにずらしていくのを if 文と for 分を使って制御している。具体的には k の値は塊の先頭を指しているので、k の値より前の 24 個に緑色の点灯ステータスを代入していく。k の値は 80 まで増え続けるが、実際に LED テープがあるのは 55 個までなので 1 つずつ消えていくような動きができる。

21,22 行目で LED テープに点灯データの送信を行う。

23,24 行目では volume は可変抵抗から値を読み取り、読み取った値から delay の値を変化させることで、LED テープの光の速度を制御している。

（※文責: 老沼響）

**エレベータパターン**

上記のプログラムは、エレベータ用のプログラムである。このプログラムは Element の中心から両側に光が流れ、トグルスイッチを押している間点滅をし離したら光が赤くなり、中心へ流れていく、エスカレータの扉の開閉を示唆するプログラムとなっている。

1〜2 行目は関数宣言や変数に値を代入している。var は押しボタンの状態を代入しており、ボタンが押されていると LOW、離されていると HIGH となる。

5〜19 行目は開くときのプログラムである。5〜8 行目で押しボタンが押されている間は while 文でループし続け、LED が消灯状態を維持する。9〜16 行目は、中心から順に LED テープに緑の点灯ステータスを代入し、点灯データの送信を行う。今回は 1 つずつ順に点灯させるために、ステータスの代入とデータの送信を同時に行っている。

17,18 行目では volume は可変抵抗から値を読み取り、読み取った値から delay の値を変化させることで、LED テープの光の速度を制御している。

21 行目では再び押しボタンの状態を変数 var に代入している。

23〜42 行目は開ききってから閉じるまでの間、トグルスイッチを押していたら LED が点滅するプログラムである。23 行目の while 文でトグルスイッチが押されている間繰り返す。26〜28 行目で LED テープのステータスを全て緑に点灯に代入し、29,30 行目で点灯データの送信を行う。31 行目で 500ms 待機する。その後 33〜36 行目で LED テープのステータスを全て消灯に代入し、37,38 行目で点灯データの送信を行う。39 行目で再び 500ms 待機し、40 行目で押しボタンの状態を変数 var に代入を行う。ここで押しボタンの状態を読み取ることでこの後のループが行われるかどうかを判定する。

待機ループ終了後、44〜47 行目で LED テープのステータスを全て赤で点灯に代入し、48,49 行目で点灯データの送信を行う。その後 50 行目で 1000ms 待機する。

53〜63 行目は閉じるときのプログラムである。53〜60 行目で左右から順に LED テープのステータスに消灯ステータスを代入し、点灯データの送信を行う。ここでも 1 つずつ消灯するためにステータス代入と送信を同時に行っている。61,62 行目では volume は可変抵抗から値を読み取り、読み取った値から delay の値を変化させることで、LED テープの光の速度を制御している。

（※文責: 老沼響）

## 2.2 Element.02 - bect

### 2.2.1 Element の目標・目的

本 Element の目的は、柔らかい膜が押し出される「むにゅっ」という動きを用いて、視覚や、他者に手の平を押されているような触覚で「方向」を認識できるようにすることである。ゴム製のボールを握りしめた時に「むにゅっ」とする動きのユニークさに着目し、その動きを何かに活かせないかという視点から製作を開始した。アイデアの段階で、「むにゅっ」という動きが方向を指し示しているように見えたことから、「方向を示す表現」として用いる案が生まれた。そこで視覚と触覚で方向を捉えることができる Element「bect」の製作を行なった。

図 2.5　bect の完成形

（※文責: 専徒礼樹）

### 2.2.2　Element 製作手順・方法

本グループにおいては、製作期間中に大きな方針転換を行っている。そのため、前期活動と後期活動を通して、作業内容にも変更がある。以下「アイデア出しから決定にかけて」の章では主に前期活動内容から後期製作物の変更に至るまでの過程を記し、その後、「後期活動」の章において実際に製作した Element についての詳細を記す。

（※文責: 専徒礼樹）

**前期活動**

KJ 法を行いグループ決めを行い、グループとして活動を始めた。当初本グループはコミュニケーションや人の痕跡などのアイデア、分野に興味があるメンバー 3 名で構成されていた。他者とコミュニケーションをとる際の問題を解決するものを作るという方針をたて、話し合いを進めた。その話し合いの中で、電車やバスの中で席に座りたいが、「他者に対して声をかけるのを躊躇ってしまう」といった私たちの生活の中で、起きやすい状況を問題として考えた。そこで、言葉を使わず他者と非言語コミュニケーションのできるデバイスの製作に取り組むことにした。

**原因**

席を譲ってもらうという状況を考察した結果、コミュニケーションが成立している場合には、自分の気持ちを他者が察していることが前提として挙げられた。そこでのコミュニケーションが上手く行かない場合の原因としては、気持ちを察するという能力が当事者によって差異があるため、コミュニケーションエラーが起きることが問題となっていると推察した。そこで「察してもらう」を可視化し、コミュニケーションを補助するデバイスの製作を目指した。そこで検討を重ね、2 つの

アイデアが出された。

**ペットロボット系デバイス**

1つ目のアイデアとしては、自分の表情を送りあうデバイスを考案した。リアルな顔を模したデバイスを作るとなると、デバイスの巨大化、センサ等の問題が考えられたので、簡単に入力と出力を実現させる方法を考案した。入力方法は、特定の動作に反応して変化することにした。ペットロボットへの動作を参考にして、なでると「喜び」の表情を送信し、握ると「悲しみ」の表情を送信し、相手デバイスに表示するようにした。出力は小型のディスプレイにデフォルメされた表情を表示させることにした。そこでプログラミングロボット「Sphero bolt」を用いて、表情をデバイスに表示するプロトタイプを作成した。デバイス上の LED の点滅を操作して、喜びの表情（図 2.6）と悲しみの表情（図 2.7）のパターンを作成した。このプロトタイプを用いて想定した動作の検証を行い、グループ内で議論を深めた。

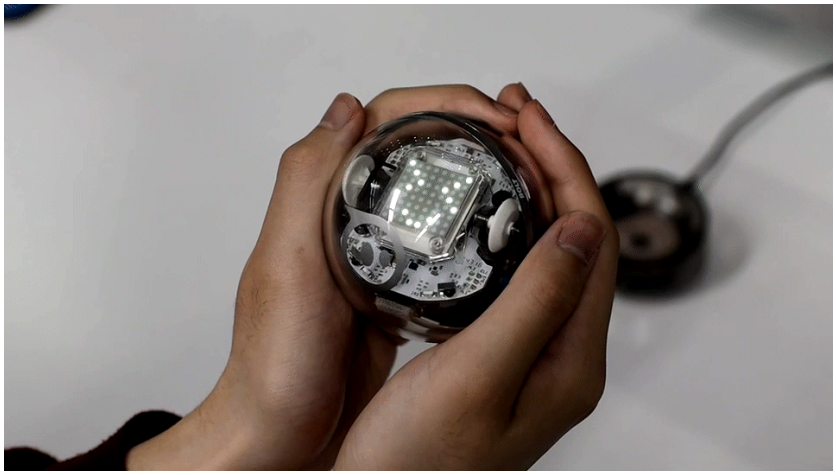


図 2.6　Sphero を用いた検証で使用した喜びの表情

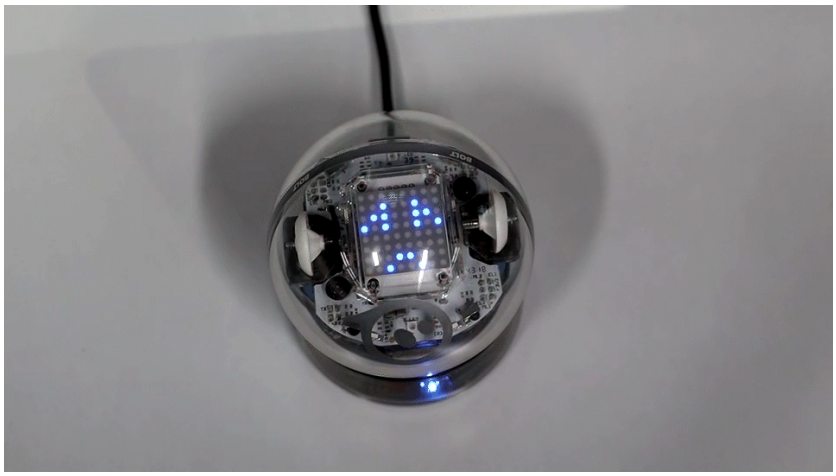


図 2.7　Sphero を用いた検証で使用した悲しみの表情

**検証後**

表情を用いた動作の検証で得られた見解として、デバイスの操作にテンポの悪さを感じたこと、表情という表現方法では感情が直接的に伝わるように感じ、「察してもらう」という当初の目的か

ら逸脱していると考えた。

**色を用いたデバイス**

そこで 2 つ目のアイデアとして、表情ではなく「色」を用いることで情報量をできるだけ省いたコミュニケーションの実現を考案した。自身の感情を色に変換して周囲に伝播させることで、色を受け取った者がその色を見て送信者の意図を考え、直接的なコミュニケーションに発展する。このような「色を用いて周囲の人に気持ちを察してもらう」デバイスの製作・検討を一旦の目標と定め、中間発表会に臨んだ。デバイス名は「Feeling Catcher」と名づけた（図 2.8）。

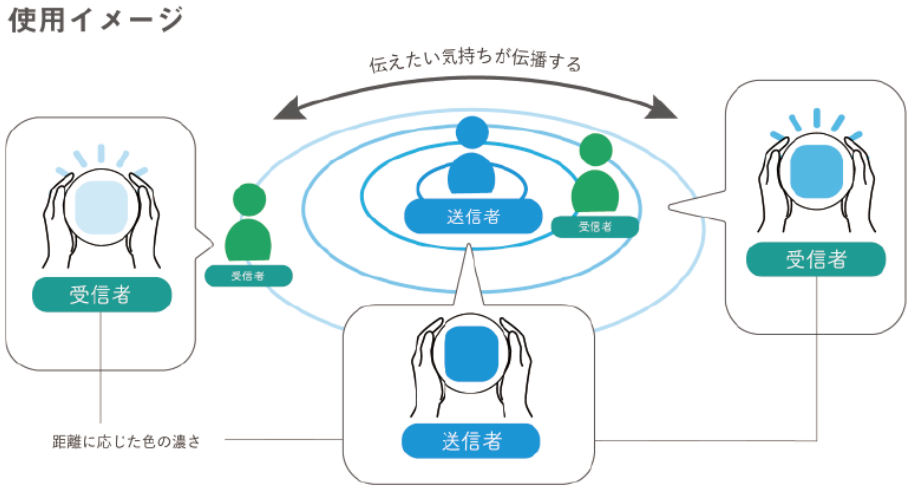



図 2.8　色を用いた使用イメージ

（※文責: 専徒礼樹）

**中間発表会からの転換**

中間発表会を受けて、「察してほしい」ことを周囲に明示するということは「察してほしい」という考えから矛盾しているのではないかと指摘を受けた。その指摘から、抱えている問題をあぶり出し、会議を重ねた。結果として、大きな方針転換を行うことにした。なお、中間発表会で受けた具体的な指摘、発見した問題点等は「中間発表会でのフィードバックを受けて」において記述した。

（※文責: 専徒礼樹）

**アイデアのイメージ・決定経緯**

**アイデア出し**

前期で取り組んでいた感情を扱うといった分野の難しさからアイデア出しに行き詰まり、根本的に課題設定から見直しを図った。当初に立ち帰り、コミュニケーションや人の痕跡などに関連したアイデアをグループ内で持ち寄り、再び議論を重ねた。その中で前期活動のクイックプロトタイピ

ングに使っていた、ゴムボールを握った際に「むにゅっ」とする動きのユニークさに気づいた。柔らかい物があった際に人は触りたくなったり、興味をひくのではないかという考えた。そこで 1 つのアイデアとしてむにゅっという動きを新たな方向を示すデバイスとして用いる案が生まれた。デバイスの用法としては、ユーザが行きたい場所や人の方向を指し、そこへ到達するまで、デバイスがその方向を指し続けることができるものにした。この案をもとに、検討を重ねながら、有用性の確認を行なった。

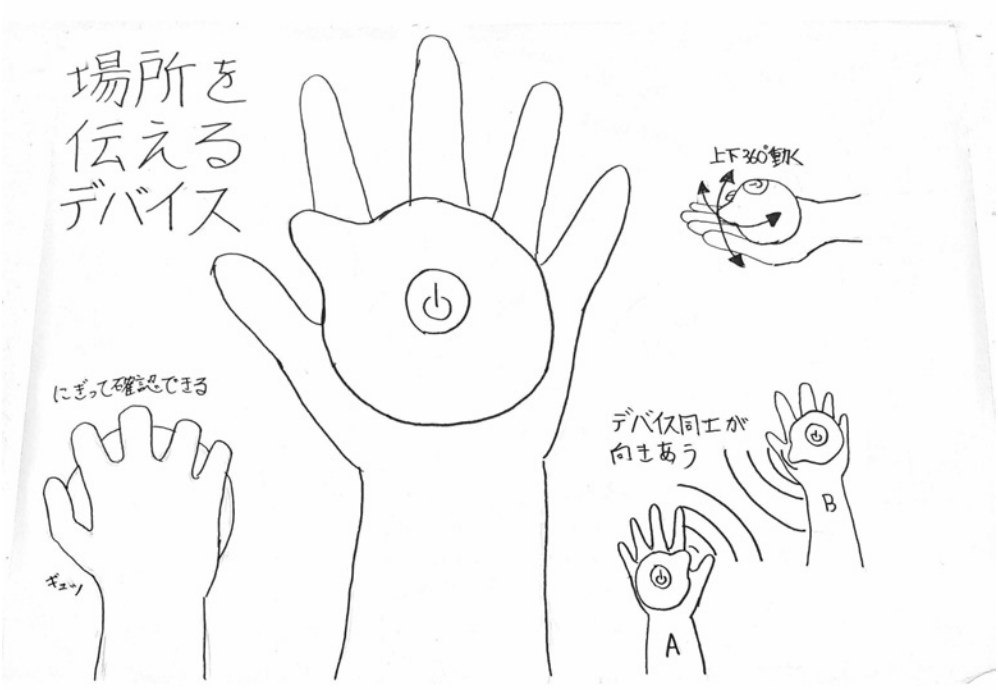



図 2.9　当初の新規アイデア案

**新規案のメリット**

新規案のメリットは、柔らかい膜がデバイスの内側から押され、外に飛び出るという構造のため、デバイスを手で握った際に触覚でも方向を把握することができるという点が挙げられる。また上記に記したように、日常のインタフェースとしてあまり見かけない動きのため、興味を引きやすいことや、触りたくなる点がメリットとして挙げられた。以上の理由より、方向を把握する際に、視覚のみならず、触覚で把握できるという新しい体験を生み出す Element の製作を行うことに決定した。また他のデバイスや、アプリケーションとの触覚を用いるという点で差別化ができていることも、製作を進める 1 つの理由となった。また製作するにあたって、Element に bect と名付けた。これは方向の意味を持つ英単語 vector を由来としている。

（※文責: 専徒礼樹）

**後期活動**

今回製作する方向を示す Element をアイデア通り実現するには大まかに以下の条件が必要であった。

- 必要なモータやマイコン等の部品を選定をすること

- 相手デバイスの方向を指し示すこと
  - デバイスが自身の位置情報を取得できるようにすること
  - デバイス同士で通信すること
  - 相手デバイス方向に回転すること
- 柔らかい膜が押し出され、「むにゅっ」という動きを表現すること
  - 膜の選定
  - 押し出す機構の作成
- デバイスの模型を製作すること
  - 土台と回転部の作成
  - 球状模型の製作
  - ヘッド部の製作

下記に条件を達成するまでの過程と方法を記す。また、達成できなかった部分に関してはその理由を記す。

（※文責: 佐々木皓大）

**条件 1：必要なモータやマイコン等の部品を選定をすること**

まず、動作を実現するために必要なマイコンやモータの選定を行った。基本的な構想として、デバイスの方向指定をするステッピングモータと、やわらかい膜を押し出す機構にサーボモータの 2 つを取り付けることにした。次にプログラムを書き込むマイコンを決めた。当初は Arduino UNO を使って 2 つのモータを制御しようと考えた。しかし、デバイスが回転することを考慮すると、マイコン自体が回転し配線が複雑になることが懸念された。そのため、2 つのマイコンを用いてモータを制御することでこの問題を解決することにした。ステッピングモータを Arduino UNO、サーボモータを M5Stick C によって制御することにした。Arduino UNO はすでに購入していて、かつ操作にも慣れていたので、これを選定した。M5Stick C は小型かつ充電可能で、簡易的なプログラムを実行することに長けていたので、これを選定した。使用する部品などがある程度固まったところで、プログラム製作と模型製作に分かれて作業を行った。

（※文責: 佐々木皓大）

**条件 2：相手デバイスの方向を指し示すこと**

**デバイス同士の通信について**

条件で記したようにアイデアを実現するには

- デバイスが位置情報を取得
- デバイス同士の通信

を実装する必要がある。理想としてはデバイス同士が近くにいれば反応し、ナビゲーションシステムとして使えることを考えていた。そのためデバイス同士の通信を行う機能は必要不可欠であった。しかし、前述したように後期は前期から方向転換して製作に行ったため、時間が足りず、またメンバーの技術的側面から見ても実装することができなかった。検討していた方法としては BLE（Bluetooth Low Energy）ビーコンを用いた通信を使用し、実現することを考えていた。また

M5Stack には ESP32 と言われるモジュールが搭載されており、BLE 通信が可能であった。ビーコンは比較的安価であり、プロジェクト予算で購入することが可能であった。デバイス同士での通信は ESP32 を使用してインターネットに接続し、構築したサーバから通信するという方法を考案していた。

**方向指定について**

上記で述べたように、今回のデバイスには通信機能を実装することはできなかった。そこで、通信をせずに指定した方向にデバイスを向かせることを最終発表会までの目標とした。

**方向指定の限界**

当初の想定では、上下左右 360 度を回転することで、空間的に相手の位置を把握することを目指していた。そこで、上下左右 360 度の方向に動かすために、まず似たような機構を持つデバイスがないか調査を行った。身の回りにあるデバイスとしては監視カメラがあったため、それを基準に実現できるか否かの調査を行った。その結果、パン/チルト機構という、水平方向と垂直方向を独立して回転することで空間的な角度の調整ができることが分かった。そこで、パン/チルト機構を実現できる製作キット[*2]を入手したので、これを用いて製作を行おうとした。しかし、今回製作するデバイスの形状を考慮すると、周囲を覆うようにして膜を貼り付けなければいけなかった。また、設置場所やその機構を固定する部位の製作を行うと、多くの時間やデバイスの更なる巨大化が考えられたため、残りの日数では実現が難しいと考え、今回のプロジェクトでは断念した。また、この方法では真下やそれに近い場所では方向を指し示すことが不可能という欠点も存在した。以上の点を踏まえて、今回の製作においては、水平方向のみを実現することで、限定的にデバイスの動作を実現することにした。方向を限定することで、角度の計算が容易になる点や、デバイスが小型になる、膜を張る面積を縮小できるという点において製作コストを削減できた。

（※文責: 専徒礼樹）

**仮想的な位置情報による回転の制御**

次に、水平方向の回転制御について説明する。部品の選定で述べたように、回転の制御にはステッピングモータを使用した。ステッピングモータを回転させる際に、相手デバイスの位置情報から自デバイスが相手デバイスに向くための回転角度を割り出す必要がある。しかし今回の製作においては、前述したように位置情報の通信機能は断念した。そこで、仮想的に相手デバイスの位置情報を指定することで回転を制御することにした。これは、今後通信機能が実現できた際に、仮想的な位置情報を現実の位置情報に置き換えることで同じ動作を再現できると考えたからである。以下、ステッピングモータの制御について詳しく概要を説明する。まず、仮想的な位置情報を表現する手段として、アニメーションに長けているプログラミングソフトである Processing を使用した。Processing で作成した GUI（Graphical User Interface）を操作することで、仮想的に自デバイスと相手デバイスの位置を指定し、それらの位置情報から回転する角度を算出することができる。Processing で作成したウィンドウの中心に原点を設定し、これを自身のデバイスと定義した。次に、ウィンドウの任意の場所をマウスでポインティングすることで相手デバイスの位置を定義できる。このように定義された自デバイスから相手デバイスへの角度を Processing 上で計算する。そ

[*2] Pan/Tilt Bracket Kit https://www.sparkfun.com/products/14391

の値を Processing と Arduino UNO がシリアル通信することによって、計算した角度を Arduino UNO へ送信する。最後に、Arduino UNO が読み取った値を参照し、ステッピングモータが同様の角度を回転するという仕組みになっている。このように、Processing の GUI によって、仮想的な位置情報を用いることで、ステッピングモータを任意の方向に回転させる事が可能になった (図 2.10)。当初は、これらマイコンやモータをデバイス内に収納し、Bluetooth 通信をすることで遠隔制御することを目標としていたが、製作時間が限られていたため断念した。そのため今回は、有線接続によって PC 上の Processing と直接シリアル通信を行って回転の制御を行った。

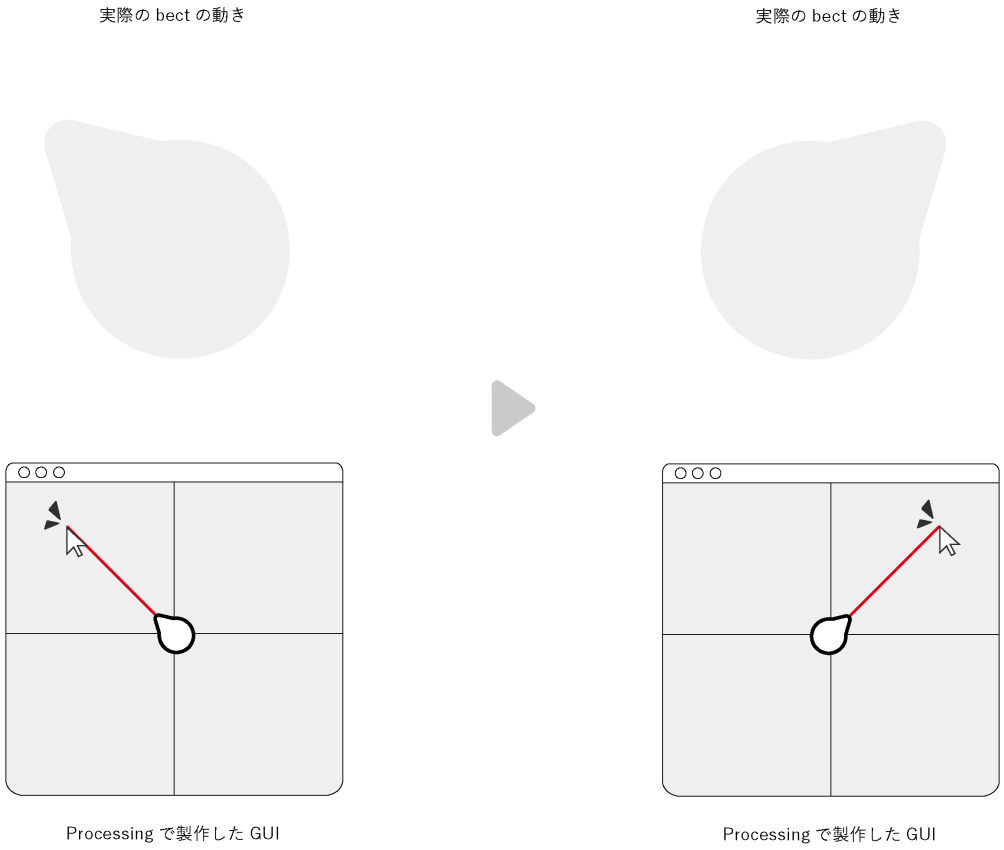



図 2.10　GUI を用いた操作と実際の動きのイメージ図

（※文責: 佐々木皓大）

**条件 3：柔らかい膜が押し出され、「むにゅっ」という動きを表現すること**

**膜の選定過程**

まず、この条件を達成するために、柔らかい膜となる素材の選定を行う必要があった。メンバー内で共有していたイメージは突起物を素材に突き立てた際に、柔らかい素材がその突起物に追従するように伸縮する素材であった。そのため、身の回りにあるイメージと似た素材を収集するところから始めた。

**ゴム風船**

まず最初に、ゴム風船を試した。ゴム風船は大方、先ほど述べたようなイメージ通りの動きをした。しかし、ゴム風船という形状から、思い通りの形に成形することが難しかったため、同様の厚さを持つゴムシートを用意しようとした。ただこのゴムシートはホームセンターや EC サイトではゴム風船と同じ厚さのものが存在しなかったため、断念せざるを得

なかった。他の厚さのゴムシートも試してみたが、厚すぎ、突起物に追従するように動かなかったため、こちらも断念した。しかし、ゴム素材は我々のイメージした動作に最も近い素材であったため、巨大なバルーンなどを用いれば、実現の可能性があった。

**シリコン素材**

次に、シリコン素材を試した。シリコン素材は形や素材の滑らかさはゴム素材より良い。しかし、追従性が悪く、それなりに大きな力で突起物を押し込まなければ動きが小さく、使用にはあまり適さないことがわかった。また硬さゆえか、裂けやすい特性があることもわかり、シリコンシート等を用いて製作することは難しいという判断となった。

**布素材**

次に、布素材を試した。まず、いくつかのスパッツを試した。追従性は非常に高く、また素材の加工が安易であった。しかし、その薄さゆえに、透けてしまい、デバイスの中の構造が見えてしまうという問題が起きた。また、すぐに破れてしまいそうな問題があったため、こちらの素材も使用に適さなかった。

**ウレタン樹脂**

次に、ウレタン樹脂を用いた素材を試した。ウレタン樹脂は人肌ゲル[*3]というエクシール[*4]から販売されている商品を用いて、試した。まず人肌ゲルはその名前にある通り、人肌の質感に近いウレタン樹脂となっている。主剤と硬化剤を混ぜ、好きな形に流し込むことで、思い通りの形に成形することができるようになっている。前期に購入していたため、それを使って、ドーム状になるように成形を行なった。また、成形を行う際は 100 円均一で購入したボウルを用い、成形を行なった。結果、ボウルとウレタンの相性が悪く、型から剥がれないという問題が生じた。また、一部硬化不良が起き、固まりきらない部分も存在した。どうにか硬化できている一部を剥がし触った結果、非常にべたつき、様々なものが付着してしまっていた。失敗した原因としては主剤と硬化剤の混ぜる比率が非常にシビアであることや、型に流し込み、固まるまで室内の温度が安定している場所でなかった点が考えられた。ただ、失敗せずに、作ることができれば追従性があり、適度な強度があり、イメージ通りの動きをする素材であると思われた。そこで、すでに同じ素材で作られており、加工しやすいようシート状になっている素材がないか探したところ、同会社から、同じ素材で作られた人肌ゲルシートが販売されており、その素材を使用して製作をすることに決定した。この素材には 2 種類あり、片面非粘着の素材と、両面粘着の素材があった。製作するデバイスの都合上、片面が突起物の押し出す部分となるため片面非粘着の素材であった方が都合が良く、まずそちらを選択し、購入した。しかし、片面非粘着の素材は追従性が悪く、シリコンとほとんど変わらない素材となっていたため、両面粘着の素材に変更し、製作に当たった。なお粘着する部分に関してはタルクを主成分とするベビーパウダを両面全体に塗布することで粘着性がなくなり、非常に肌触りがよい素材にすることができた。

[*3] 人肌ゲル https://www.exseal.co.jp/creative/hitohadanyu.htm
[*4] 株式会社エクシール https://www.exseal.co.jp

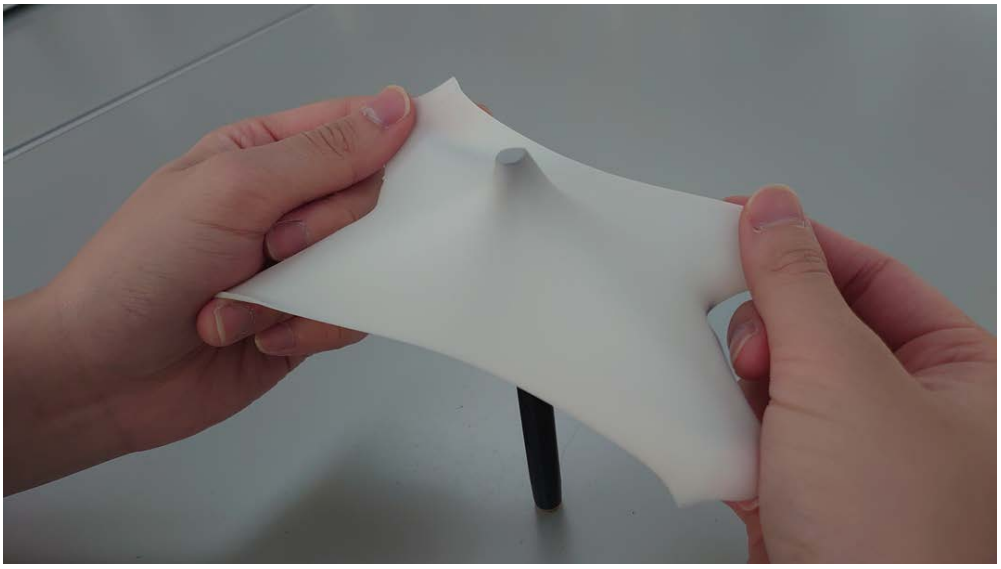


図 2.11　検証中の人肌ゲルシート

（※文責: 専徒礼樹）

**押し出す機構の作成**

次に、サーボモータを制御し膜を押し出す機能の作成過程を説明する。押し出しを表現するために、様々な方法を検討した。その中で、ラックアンドピニオンという機構を試したところ、目的の動きをサーボモータの制御のみで実現できたのでこれを採用した。ラックアンドピニオンとは歯車の一種で、回転力を直線の動きに変換するものである（図 2.12)。ピニオンとよばれる小口径の円形歯車と、平板状の棒に歯がつけられたラックを組み合わせて動作する。ピニオンに回転力を加えると、ラックが水平方向に動く。この回転力を与えるモータに、サーボモータを使用した。サーボモータの回転は M5Stick C によって制御しているので、標準の servo ライブラリが使えないため、PWM 信号を用いて角度の制御を行った。また、M5Stick C とサーボモータはデバイスと一緒に回転させるため、デバイス内部に組み込まれることになる。よって、無線通信により回転の開始や停止を命令する必要がある。この問題には、M5Stick C に搭載されている Bluetooth 機能を用いて対応した。M5Stick C と PC を Bluetooth 接続することで、PC 上の Arduino IDE シリアルモニタに送信した文字列を M5Stick C が判断して動作の開始と停止を制御している。以上のように、無線通信によってサーボモータを回転させ、ラックアンドピニオンを動作させた。

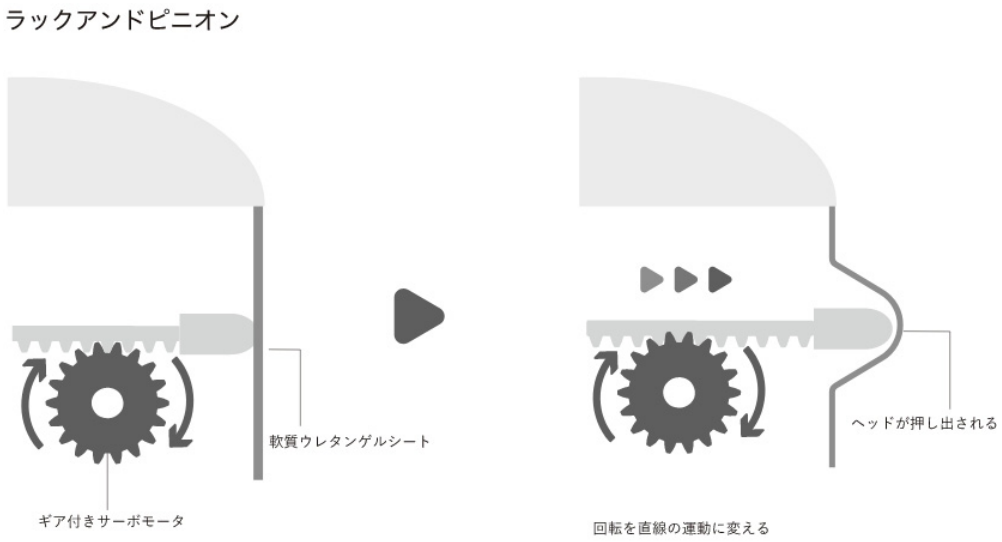



図 2.12　ラックアンドピニオンの動作

（※文責: 佐々木皓大）

**条件 4：デバイスの模型を製作すること**

模型の製作にあたり、デバイスのデザインについて数回の議論を行った。その結果、なるべく球状にさせることで手で包みやすいフォルムであること、マイコンを内包できるようにスペースを確保しつつ最大限の小型化を図ること、メンテナンス等がやりやすいように分解・修正が容易であることなどが達成目標として挙げられた。以上の点を踏まえてデバイスの製作に取り掛かった。球状のフォルムである点から立体的に作る必要があったため、3D プリンタで造形することにした。3D ソフトは、扱いやすい Blender を使用していたが、3D プリント時にエラーが多く発生したので、使用を中止した。代わりに、3D プリント製作に定評のある Fusion360 を使いモデリングをすることにした。

（※文責: 佐々木皓大）

**土台と回転部の製作**

まず、デバイスの動作を考慮すると、ステッピングモータを固定する土台の部分（図 2.13）と、その上部に取り付けて回転する回転部分（図 2.14）に分けなければいけない。土台部分の上底にステッピングモータをねじで固定した（図 2.15）。その下のスペースに Arduino UNO とモータドライバ、電池ボックスを収納できるスペースを確保した。回転部分にはラックアンドピニオンとサーボモータ、マイコンとなる M5Stick C を取り付ける必要があった。また、ラックアンドピニオンを押し出しのための棒として利用しているため、この棒を直線的に動かすために動きを補助するレールも必要になる（図 2.16）。これらをなるべく省スペースで配置するために専用のアタッチメントとなる部品を製作した。この部品はアタッチメントとしての機能だけでなく、ステッピングモータの回転を受ける回転の軸という役割もある。ステッピングモータに部品を挿すことで押し出しの機構全体が回転することを実現した。以上によって、デバイスの土台部分とステッピングモータ上部に取り付ける回転部分の作成ができた。

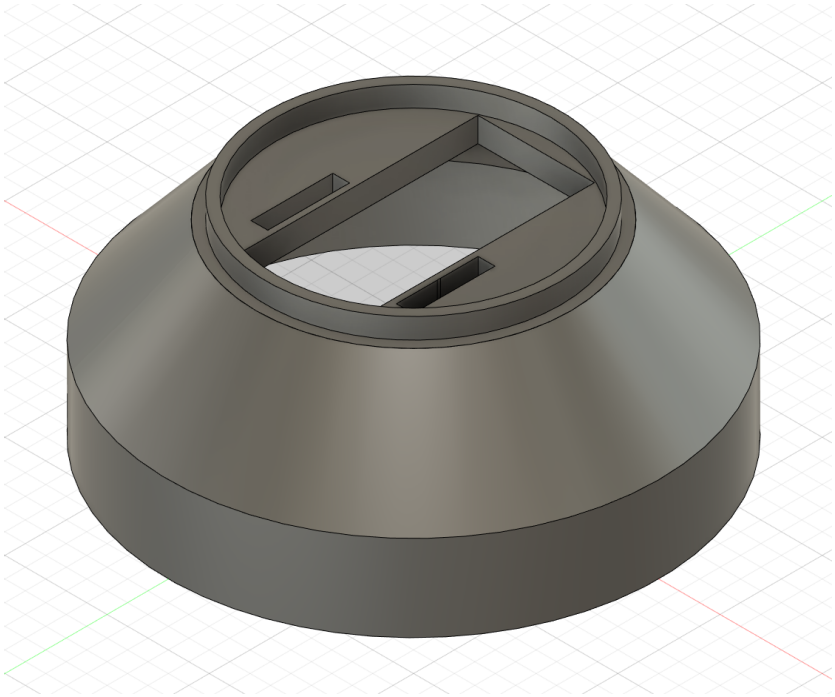


図 2.13　土台部分の 3D データ

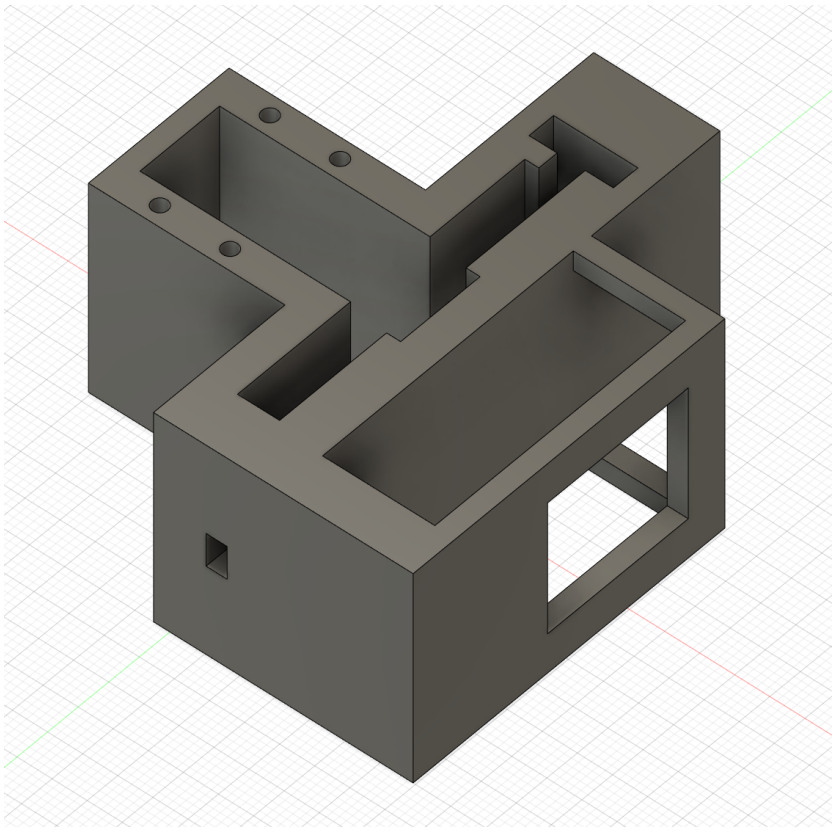


図 2.14　回転部の 3D データ

図 2.15　土台部にネジで固定したステッピングモータ

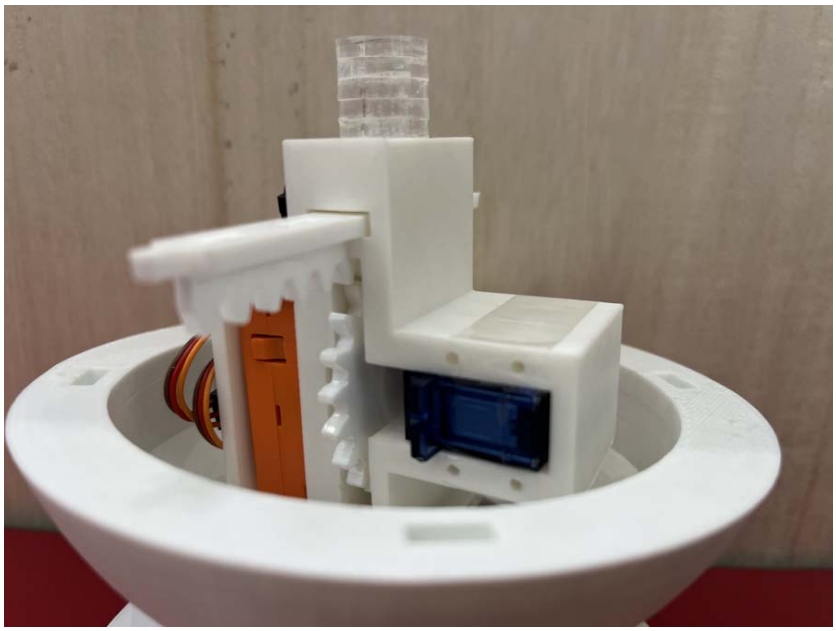


図 2.16　サーボモータ、M5Stick C、ラックアンドピニオンを取り付けた回転部

（※文責: 佐々木皓大）

**球状模型の製作**

次に、この回転部分の外側を囲う球状の模型の製作に取り掛かった。球状のデバイスにするうえで一番の課題となったのは、膜となる素材をどのように取り付けるかということである。球状に膜を張る方法はいくつか検討されたが、それを内部から押し出すとなると、現状では具体的な解決策は生まれなかった。そこで、完全に球状にすることを断念し、膜を張っている部分のみ円柱のような形状にした。そうすることで、膜にシワができないように固定することができた。また、膜は専用のアタッチメントに釘で挿すことによって固定している。当初はネジで固定することで取り外しを可能にしたかったが、ネジを回す際に膜自体が巻き込まれて破れてしまうという問題があったた

め、釘によって簡易的な固定をすることになった。また、釘を挿す部分の膜を 2 重にすることで、押し出された際に膜が破けるという懸念をある程度解消した。釘で留めたことで膜を取り外すことができなくなった代わりに、アタッチメント自体を取り外し可能にすることで、すぐにメンテナンスすることを可能にした。この膜を取りつけたアタッチメントを上下にある半球状の模型に差し込んで取り付けることで、前述したような球の中間に膜を張ったようなデバイスができた。ここでデバイスを回転させて動きを確かめると、上部の半球状の模型が回転部分と接触することで回転が止まるという問題が起きた。そこで、模型の頂点にネジが差し込める穴を空け、そのネジを回転部上部に固定した（図 2.19)。そして、回転部上部にアクリル板を加工して作成したスペーサを取り付けることで、回転部と半球のあいだに 10mm 程のスペースが生まれた。そうすることで、接触して回転が止まることがなくなった。このようにして、デバイスの外側となる球状の模型を製作した(図 2.18)。

図 2.17　球場模型上下の 3D データ

図 2.18　膜を張った実際の模型写真

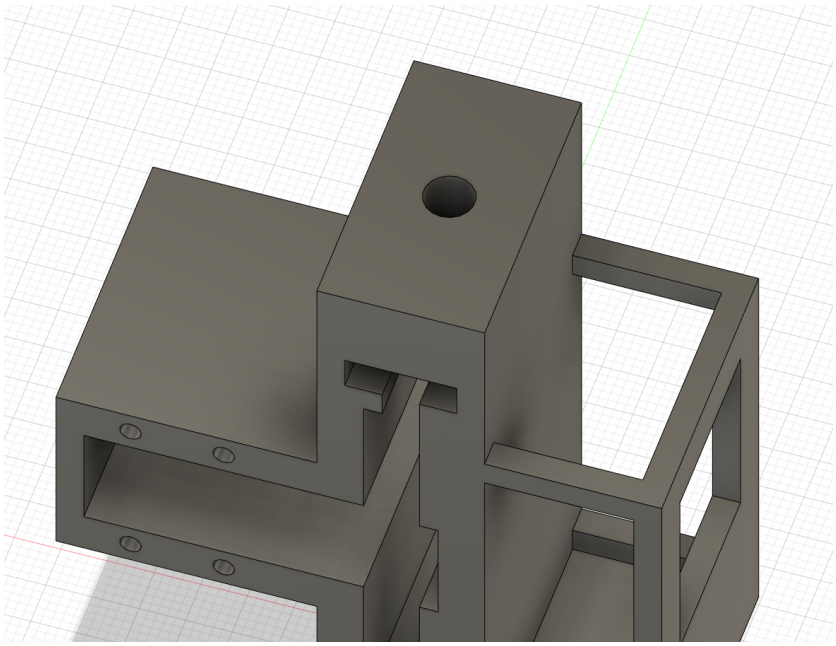


図 2.19　回転部上部に空けたネジ穴の 3D データ

（※文責: 佐々木皓大）

**ヘッドの製作**

最後に、ラックアンドピニオンの棒に取り付けるヘッド部分を製作した。このヘッドの形状によって膜の押し出される印象が変化するため、複数個のヘッドを作成しそれぞれの感触を確かめた。かまぼこ型のヘッド（図 2.20 左）、傘型のヘッド（図 2.20 左 2）、弾丸型のヘッド（図 2.20 左 3）、赤子の指を模したヘッド（図 2.20 左 4）の 4 つを製作した。しかし、発表の期日が迫っていた

ことで、満足いくまで選定できなかった。このヘッド部については様々な形や効果が期待できるため、今後より検討すべき項目である。

図 2.20　作成した 4 種類のヘッド

（※文責: 佐々木皓大）

## 2.3　Element.03 - POP UP SHELF

### 2.3.1　Element の目標・目的

本 Element の目的は、本を手に取る体験を豊かにしたり、本を探す際の検索性を高めるシステムを製作することである。電子書籍が普及した現在でも、物理的な本は手軽さや質感等の特徴から、多くの人に広く親しまれている。一方、物理的な本は電子書籍等と比較して、検索が困難であったり、インタラクティブ性に劣るといった課題がある。本グループでは、目的の本を探すという本の探索に対して、直感的な操作による本からのフィードバックを与えることで、そのような課題を解決すると共に、本を手に取る体験を豊かにし、本を手に取るきっかけを作るシステム「POP UP SHELF」を製作することを目標とした。

（※文責: 丹野夏海）

### 2.3.2　Element 製作手順・方法

**アイデア着想までの流れ**

私たちのグループは、操作ツールや、なにか操作したくなるような製作物に興味のあるメンバーをもとに構成されている。メンバー確定後、実際にどのような製作物を作りたいのか、それぞれネタ帳や、前置詞図鑑をもとに自分の案を出し合い、具体的なアイデアになるよう話し合いを重ねた。ここでは、「POP UP SHELF」に至るまでに出たアイデアを紹介する。

1 つ目は、「ノイズキャンセリングを搭載した壁」である。この案ではノイズを低減する効果以外にも壁本体が音にイコライザをかける機能を考えたが、いずれも実現性が低く、既に似たような先行研究が存在したため却下することにした。

2 つ目は、「メータ機能を備えたチャック」である。チャックの位置を取得し、その値を出力するデバイスである。暑い時にはチャックを開き、寒い時にはチャックを閉じる自然な動作から、温度調節機能を実現する一つの手法として提案したものである。しかし、すでに同じ先行研究が存在していたので却下した。

3 つ目は、「愛でる植木鉢」である。土や花の状態を取得し、その状態が植木鉢の表面に反映されるというものである。特に、土が乾いている状態の時などには、表面がカサカサになり、それをなでることで水や肥料が追加されるという仕組みである。なでることで植物を実際に愛でている感覚を感じることができるアイデアである。

4 つ目は、「陽光を入れたいカーテン」である。日がのぼるとカーテンが小刻みに揺れたりすることで、開けてもらいたい動作をするデバイスである。朝になると自動でカーテンが開くデバイスは既に存在している。しかし、私たちは、自分でカーテンを開き、外の様子を確認してもらいたいという考えから、既存のデバイスとの差別化を図った。

5 つ目は、「方向の感触フィードバックを得るドアノブ」である。押し戸や引き戸など、扉が持つ操作方法をドアノブで伝える試みを行うデバイスである。現実では、ドアノブの取り付け方や形ですでにドアの操作方法を理解できるような仕組みが存在し、新たにフィードバックを返すようなデバイスの意義が無いという結論に至り却下とした。

また、POP UP SHELF 以外に採用に近づいたアイデアは、カオスドーム（ポインティングデバイス）である。「カオスドーム」は、実用の幅が広く、メンバー全員が意欲的だったため、このアイデアを採用することとなった。この Element は、柔らかい素材で覆われたポインティングデバイスであり、3 次元的な操作を直感的に操作することを目的とした作品を想定していた。柔らかい素材で覆われることにより、音楽製作やカメラの数値設定、イラストソフトなどの操作、UFO キャッチャーなどの直感的操作を可能にすることで、初心者でも簡単に手を出せるようなデバイスを目指して製作に取り組んだ。しかし、取り入れようとしていた触覚フィードバックをうまく活かすことが難しかった。

そこで、フィードバックを利用した Element を製作することを念頭に置いて、書店でのフィールドワークを行ったところ、本棚に応用することができるのではないかと考えた。「フォーカスした場所が浮き上がる本棚」は、実際に製作に取り掛かることになったデバイスである。書店内の案内図が入っているパンフレットが、手をかざすことによって出てくると面白いのではないか、という考えのもと応用の幅を広げた結果、本棚にたどり着いた。Web サイト の閲覧や、アプリケーションのアイコン上でマウスオーバーした際に、拡大表示される機能が存在しており、現実の世界でもそのような仕組みを製作したいという考えに至った。

（※文責: 古町昂大）

**背景と先行事例**

現在、場所を取らない、在庫切れがないという理由から電子書籍が一般化してきている。また、物理的な本も手軽さや質感等の特徴から、多くの人に広く親しまれている。しかし、物理的な本は電子書籍等と比較して、検索が難しかったり、文字の拡大、縮小機能などのインタラクティブ性に劣るといった課題がある。例えば、スマートフォンで Web サイト や電子書籍を閲覧する場合、タップした周囲を強調表示することで、対象を見やすくしたり、選択しやすくなる視覚効果が提供されている。私たちは、そういった視覚効果を実世界の本棚に適用することで、物理的な本を探したり、取り出したりする行為を拡張することに注目した。

POP UP SHELF は、ユーザが手を伸ばした動作を検出し、指し示した本とその周囲の本を山なりに押し出すことで、目的の本を探しやすくしたり、本を手に取る体験を豊かにすることを目指し、製作した本棚である。初めに、本棚に並んでいる本の中から、目的の本を指差す。その手を本棚に取り付けられているセンサが感知し、ステッピングモータに値を渡す。本棚の後ろには、タイ

ミングベルトが取り付けられており、ステッピングモータがヘッド部を目的の場所まで動かす。最後にサーボモータがヘッド部を押し出すことで、本が飛び出る仕組みである。指が本を指さしたまま動くと、本もそれに合わせて動く。当初は、本を選ぶ体験を豊かなものにという目的のもと、本棚から取り出すとき、本が波打つ表現をする予定だったが、予算と開発時間がたりず実装には至らなかった。

POP UP SHELF は本を取る体験を豊かにする、という目的以外に、操作性を高めることも目標として掲げており、説明を受けずとも、誰もが最初から操作可能である、直感的な操作が求められた。そのため、本を指差すだけという、簡単なジェスチャでの実装を目標とした。姉崎らは、ジェスチャインタフェースのためのデザインガイドラインを提案している。この研究では、Leap Motion を用いたプロトタイプの実装とデザイン・エンジニアリングの側面からの検証を通して、操作対象をポインティングすることでジェスチャ UI の操作性が向上することを報告している [6]。また彼らは、「指し示す行為や、一定時間停止する行為は誰でも簡単にでき、行為そのものに個人差も少なく、結果、1 番使いやすいジェスチャ UI のプロトタイプであることが判明した」と言っている。このことから、POP UP SHELF で採用した、指をさすというジェスチャは誰もができる直感的な操作であることがわかる。

また、正畑らは図書案内のためのスポットライト型ポインティングシステムを提案している。この研究では、指差しジェスチャによるポインティング操作に加えて、仮想タッチパネルによるフォーカスエリアの移動や、手のひらを回転させるジェスチャでフォーカスエリアのサイズを変更できる機能によって、図書案内にかかる時間を比較する実験を行なっている [7]。実験の結果、自力・指差し口頭より、案内にかかる時間が短縮されたことを報告している。このことから、POP UP SHELF の、本を指さすというジェスチャによるポインティング操作により、指差した部分の本が拡大されるように押し出されるというフィードバックは、本の探索にかかる時間を短縮し、探索性を高めると考えられる。

（※文責: 丹野夏海）

**Element 製作過程**

**前期活動**

はじめに、プロトタイプの本棚の製作を行った。本棚の素材は、ホワイトウッドを使用した。ホワイトウッドは 材質は柔らかで切削や釘打ちなど加工性に優れており、乾燥による狂いが少なく強度もある。 また、プレーンで美しい木目を持っているので今回の製作に適していると考えた。本棚の背は 2 枚の板で作り、2 枚の板に 100mm ほどの間隔を開けることで、本を押し出す機構を取り付けられるようにした。

次に、実際に本を本棚に並べ、後ろから山型のもので押し出す実験を行なった。試行を重ねることで、製作を進めていく上での物理的な課題を探した。本にチリや、小口の凹凸があると、押し出しの動きが鈍くなる・ヘッド部をスライドさせると本に引っかかる・本棚や本同士の摩擦の問題などが挙げられた。これらを考慮し、機能、機構を考えた。

機能面では、本の検索機能や、波を打つ表現が挙げられた。検索機能では、本を検索すると、目的の本が浮き出るようにし、探索を補助する動作をする。また、探索を補助するという面から、2 カ所を指すことで、その間の本を範囲選択として飛び出させる機能も挙げられた。波をうつ表現では本を取り出した後、その周りが波をうつような表現をすることで、より新たなフィードバック

を与える。また、操作の方法として、指の動きによるコマンド操作などのアイデアが挙げられた。ジェスチャの違いから、本の動きを変えられないかと考えた。

機構面では、実験により浮き彫りになった課題で、本を押し出しながらスライドさせるとき、押し出す機構と本が引っかかってしまい、うまくスライドできないことが挙げられていたため、この 2 つの機能の両立を実現する機構を考えた。1 つ目に挙げられたのはアクチュエータを 7 本程度使用し、それぞれが本を押し出すことで複雑な動きを再現する案である。しかし、アクチュエータの価格が高いこと、大量の電力が必要なことから、現実的ではないため却下となった。2 つ目の案は、キャタピラを用いたものである（図 2.21）。ローラーの周りにベルトを巻きつけ、ベルトの凹凸に本を引っ掛けることで、本を飛び出させたままでのスライドをスムーズに実装できるのではないかと考えた。しかし、この構造自体が実現されていないため、その機構からの製作となることから、プロジェクト期間内での製作は困難であり不採用とした。3 つ目に挙げられたのは電磁石を応用した磁気浮上機構である。しかし、こちらも実現されていない機構であったため、不採用とした。その後も機能を実現すべく、具体的な実現方法を模索したが、機構については、中間会発表前に決定するところまでは至らなかった。

その後、中間発表会の準備に移り、プロジェクト全体でポスター・Web サイトの製作に取り掛かった。POP UP SHELF の機能を伝えるために、紹介用の動画を実際にコマ撮りで撮影するなどの工夫を行い、見てもらう人に伝わりやすい発表成果物の作成を心がけた。プロジェクト全体のメンバーで役割を分担し、作業を進め、中間発表会本番に臨んだ。

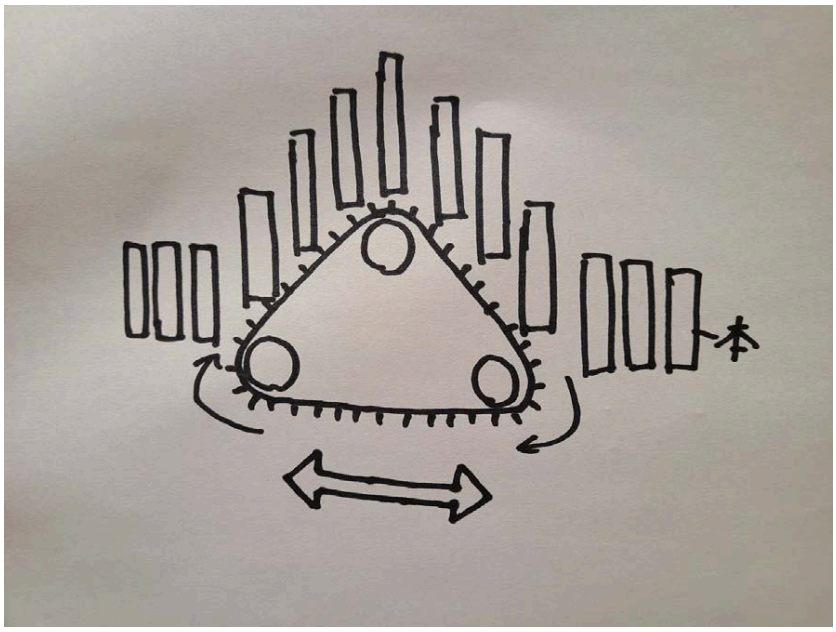



図 2.21　キャタピラの構想

（※文責: 丹野夏海）

**後期活動**

後期のプロジェクト学習開始後、製作にあたって、それぞれ自分が担当する分野を決め作業に取り掛かった。最初に、中間発表会時に来た質問や、構造、製作にあたっての問題点を明確にし、それぞれの解決策を考えた。集まったアイデアをもとにプロトタイプをいくつか製作し、最善のモデルを探った。

図 2.22　POP UP SHELF の使用例

以下、各分野に関して、章立てを行い説明する。

（※文責: 古町昂大）

**本体製作班**

まず最初に、本棚全体の構成を考えた。浮き出した本を戻す機構をより単純なものにするため本棚全体を傾けることにした。実際に本が戻るために必要な傾斜の角度を、いくつかのデモンストレーションを行い測定した。本を押し出すサーボモータのトルク量と本棚の摩擦、本の重量を考慮した結果 30 度の傾きが最適であるとわかった。実際に本に 30 度の傾斜をつけたが、本棚全体は水平になるような設計を行った。具体的には本棚の側面を 90 度の角が 3 つある五角形にし、センサを配置する長方形の板をその上からつけることによりシンプルなデザインにした。本棚の素材は、インターネットでの調査をしたのちにホームセンターに行き実際に触れてみてから桐材を選んだ。桐材は加工のしやすさや吸湿性、軽量であること等において非常に優れていた。また、桐材は本を邪魔しない自然な白色をしており、この面においても最適であった。本棚側面にはセンサを取り付ける板があり、コードを側面から後ろに通すことができるよう 2mm の隙間を設けて設計した。そのため、本棚の景観を崩すことなくセンサを取り付けることができた。（図 2.23）

また、木材の接着に用いたねじは外部からは見えないように製作した。以上のように、本棚の性能だけではなく外見の細部にまで気を配り製作を行った。（図 2.24）

本の背が当たる箇所は 2 枚の細い板を間隔をあけて配置した。その 2 枚の間をヘッドが移動する仕組みになっている。このようにすることでヘッドが本の中心の高さを移動し、安定した動作が実現した。その際、上の板がたわんでしまい本がヘッド部の側面に擦れてしまった。これを防ぐため、上の板の裏面に支えの木材を追加で接着した。これにより木材の強度が増し、たわみが軽減され干渉しなくなった。（図 2.25）

本を載せる台となる木材には低摩擦であるポリプロピレンの膜を張ることで本を押し出しやすくかつ、戻りやすくなるようにした。また、この面の木材は本が押し出された時に本の背が木材より前に飛び出るように設計した。このようにすることで摩擦を軽減するほか飛び出る動作を強調することができた。

続いて、本棚を乗せる土台を製作した。レールや本、本棚の重量に耐えられるよう頑丈な木材である 2 × 4 の木材を使用した。また、撮影において本棚に注目してもらえるよう土台はソーホースブラケットを用いて足だけとした。こちらも、白を基調としたデザインにした。実際にソーホースブラケットの組み立ては非常に手軽であり、高さの細かな調整を行うことができた。本棚の高さは撮影に合わせ、モデルが手を伸ばした時に自然な見え方をする高さに調節した。また、撮影の際の機動性や安定感は非常に優れており、今後外部での発表などの面でも非常に有効的な展示手段であると感じた。(図 2.26)


図 2.23　取り付けたセンサ

図 2.24　木材接着後の POP UP SHELF

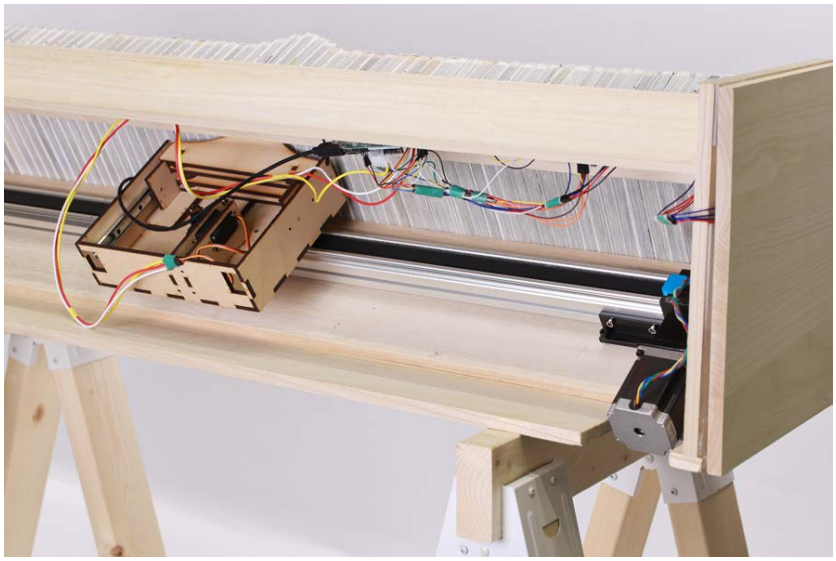


図 2.25　POP UP SHELF の背面

図 2.26　製作した土台

（※文責: 和田颯平）

**アクチュエータ・センサ製作班**

まず最初に、使用する部品を選定し、大まかな動作を再現できるプロトタイプを製作することにした。プロトタイプに使用する部品として、測距センサ、ステッピングモータ、モータドライバ、サーボモータ、LCD ディスプレイ、Arduino Uno、スライドレールを使用した。

以下、それぞれの部品について説明する。

（※文責: 家山剣）

**測距センサ**

まず最初に、測距センサについて述べる。手の位置を計測するセンサとして、Pololu 製の VL53L1X 搭載レーザー測距センサモジュールと ELEGOO Arduino 用 HC-SR04 超音波測距センサ 5PCS を購入した。

測距センサは、4000mm までの距離を高速かつ正確に測ることができ、測定結果は $I^2C$ インタフェースを介して読み取る。このセンサは、周囲の照明状況や色・形状・テクスチャなど測定対象物の特性に影響を受けにくい特徴がある。対して、超音波センサは精度が低く、応答速度が遅い代わりにホコリや水に対する耐性が強く、測定範囲が大きい特徴がある。以上の 2 つを使い比べて、測距センサの方が、今回の製作物により適していると判断した。

測距センサは、本棚横に養生テープや、マスキングテープを用いて張り付けた。しかし、手に反応するタイミングが厳格であるため、操作に多少の慣れが必要であった。改善案として、レーザーポインタを搭載したセンサを使うことで、目視で確認できるような仕組みを導入するものがある。センサで取得した値を、Arduino に送信した後に、実際の移動量に合わせるために倍率を掛けてステッピングモータに引き渡す。

（※文責: 家山剣）

**ステッピングモータとドライバモジュール**

次にステッピングモータとドライバモジュールについて述べる。ステッピングモータはおおまかに分けて 3 つの種類を使用した。また、ドライバモジュールは 4 つの種類を使用した。

最初に使用したのは、5V で駆動する小型のステッピングモータ 28BYJ-48 とモータドライバ ULN2003DC5V である。Arduino から供給される電源だけで動作電圧を賄えるため、プロトタイプを製作するにあたって最適であった。また、Arduino IDE で使うことのできる Stepper ライブラリを使用できるため、動作確認が非常に容易であった。これと、先述したセンサを組み合わせて、手の動きに合わせてステッピングモータが回転するプログラムを書き、実際に動作するプロトタイプを製作した。

その次に使用したのは、STEPPERONLINE Nema 23 CNC ステッピングモータ 2.8A 178 オンス 1.26Nm CNC ステッピングモータ DIY CNC ミルである（図 2.27）。これは、後述するスライドレールに付属してついてきたもので、高トルク・大電流が特徴のモータである。従って、先述したモータと同じ環境では動作せず、他のモータドライバや電源を検討する必要があった。このモータを動作させるために、3 種類のモータドライバを試した。

1 つ目に試したモータドライバは、Allegro 社のステッピングモータドライバ A4988 を搭載したモジュールである。1 個のバイポーラステッピングモータを最大 2A の出力で、ステップと回転方向のシンプルな制御が可能なものである。実際にモータを回転させることはできたが、逆回転がうまくいかなかった。

そこで、別のモータドライバ DRV8835 を試すことにした。このモータドライバは、チャンネルごとに 1.2A（ピーク 1.5A）で 1 つのバイポーラステッピングモータを動作させることができる。このドライバに関しても、実際にステッピングモータを正回転・逆回転させることができ、正しく制御を行うことができた。しかし、このモータドライバは、電源電圧や出力電流がステッピングモータに適合しておらず、回路基板がすぐに故障してしまう問題があった。しばらく、この問題に悩まされ、合計 10 個の基板を犠牲にした。

そして、最後に使用したものは、指導教員に紹介してもらった、L6470 モータドライバモジュールである。ネットの記事を参考に回路を組んでテスト環境を構築した。しかし、このモータドライバは、ダイオードやコンデンサなどをブレッドボード上で使用する必要があった。そのため、モータが制御できることを確認した後に、秋月電子通商製の L6470 使用ステッピングモータドライブキットを購入した。このドライバモジュールは、コンデンサやダイオードがあらかじめ内蔵されており、わざわざ複雑な回路を組む必要がなく誤作動や故障の温床となるリスクを回避することができた。加えて、このドライバモジュールは、SPI インタフェースで外部マイコンからコマンドやパラメータを与えて制御する方式を採用している。L6470 の特徴として、電源電圧の対応している幅がとても広く 8.0V〜45.0V までをサポートしている。また、ピーク出力電源も大きく 7.0A まで使用できる。モーターの細かな制御が得意であり、1/128 マイクロステップ制御が可能なためとても滑らかな表現が可能である。実際に、本棚本体に取り付けて使用してみたところ、ほかのどのドライバモジュールよりも安定した動作を行い、故障することもなかった。また、表現の幅も広がり、従来のドライバモジュールでは難しかった、加減速や座標指定を用いたヘッド部の移動ができるようになり、見ごたえのある動きができることを確認できた。

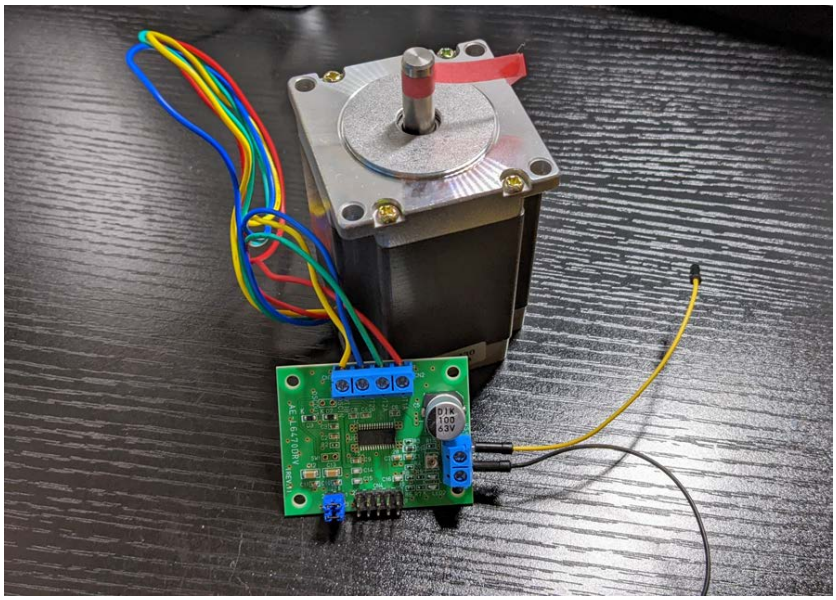


図 2.27　実際に使用したステッピングモータとドライバモジュール

（※文責: 家山剣）

**サーボモータ**

サーボモータについては、Arduino のライブラリにあるプログラムを使用できる 2 つの種類を試した。

1 つ目は、マイクロサーボ MG90S である。これを使用して、ヘッド部に組み込んだところ、トルク量が少し足りず本を押し出すには非力であったため不採用とした。

次に試したものは、MdskGang MG996 サーボモータである。こちらは、同じ 5V 電圧でもトルク量が強く、最大 15kg まで押し出すことができる。これを、ヘッド部に組み込んで本を押し出してみたところ、うまく山型の表現を行うことができた。

しかし、ステッピングモータと同じ電源回路から動かそうとすると電力不足に陥り、正常に動作しなくなることがあったので、別電源として電池ボックスを採用し、Arduino 本体の DC ジャック

から電力供給を行うことにした。その結果、高トルクを保持することができ、安定した動作を行うことができた。

（※文責: 家山剣）

**スライドレール**

スライドレールについては、 Heechoo 製のベルト駆動リニアガイドを使用した（図 2.28）。また、前述したように、ステッピングモータが付属しているものを購入した。

全長は、1200mm であり、このスライドレールをもとに本棚の大きさを決定した。このスライドレールは、タイミングベルトを搭載したモデルになっており、最高速度 500mm/s、最大負荷 300N まで耐えることができる。また、ベルト幅は 15mm、タイミングベルト数 24 歯のものを購入した。材質には、アルミニウム合金が採用されており、高硬度・長寿命が特徴である。また、防錆・防湿性にも優れ、耐久面で頑丈なものを使用した。このスライドレールは、モータブラケット・作業板等が事前に取り付けられており、ステッピングモータや、ヘッド部の取り付けが容易であった。マウントベースも付属していたため、傾いた構造の本棚本体への取り付けも問題なく使用できた。問題点としては、ステッピングモータの制御に誤作動が起きた際に、ヘッド部が端の部分に乗り上げてしまい故障につながる可能性があるところである。製作中にも何度か危ない状況があり、故障につながりそうな場面があった。対策として、スライドレールの端の部分にスイッチを取り付け、ヘッド部が端に到達したときに強制停止することが挙げられる。また、スイッチが押されたときにステッピングモータの座標をリセットすることで、長時間駆動による誤作動を防ぐ助けとなる。

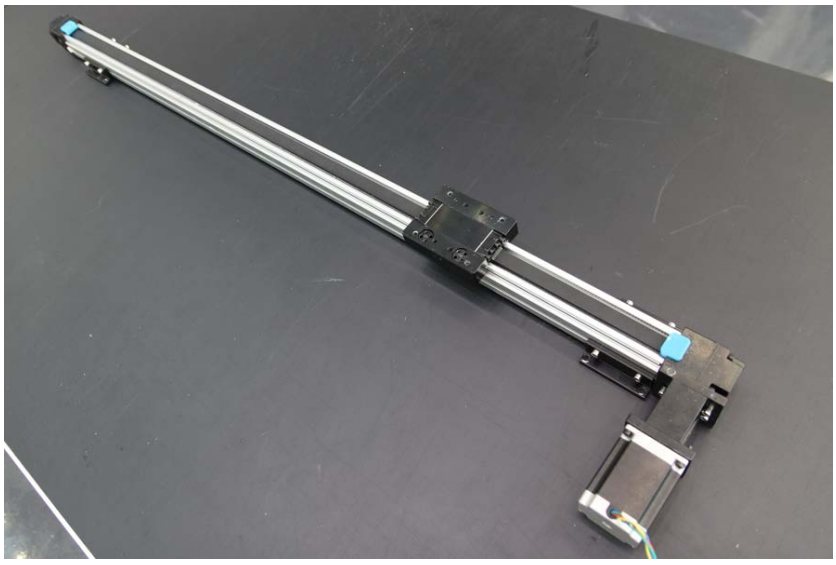


図 2.28　使用したスライドレール

（※文責: 和田颯平）

**ヘッド部製作班**

ヘッド部は、現在の形に至るまで、様々な構造や形状を考えた。初めは、シリンダを用いて波を表現する案が出た。この案は予算の面や実現性がないという教員からのアドバイス等があり、別の案を考えることにした。このほかに、電磁石を用いた案やヘッド先端にベアリングを取り付ける等の案も出た。この案はキャタピラを用いたものである。ローラーの周りにベルトを巻きつけ、ベル

トの凹凸に本を引っ掛けることで、本を飛び出させたままでのヘッドをスムーズに移動できるものである。また、電磁石を応用した磁気浮上機構は実現性が低い理由から実装はしないこととした。このように様々な案が中間発表会までに出たがいずれも確定するまでには至らなかった。　中間発表会後は様々な機構について調べ、モータを用いて機構を製作することを目標とした。また、本に波のエフェクトをつけることをやめ、本を山なりに押し出すことに専念して製作を進めた。最終的にはサーボモータを用いたラックアンドピニオン機構を使った構造に決定した。このラックアンドピニオン機構とは、モータに取り付けたギア (ピニオン) が回転することで、ラックを垂直方向に押し出す機構である。初めはモータを 3 つ同時に制御することでヘッドを持ち上げようとした。しかし、モータを同時に制御をすることは難しく、ヘッド内部の構造も複雑になった。ここで、モータの性能やヘッド部の摩擦について再検討した。本を押し出しその状態を維持しながら横移動することを考慮し、トルク量の大きいサーボモータを購入し動作させると 1 つのモータで上手く動かすことができた。次に、ヘッドをスムーズに押し出すための内部構造について考えた。ヘッド内部は外枠の底面に MDF による溝を設置し、その上をヘッドがスライドするというものを製作していた。しかし、この構造だと摩擦がかかりすぎてしまいモータを思うように制御することができなかった。そこで、外枠の内部側面とヘッドに金属製のレールを取り付け、引き出しのような底面が擦れない構造にした。この際、レールの長さを外枠の長さに合わせてカットし、レールをやすりがけすることで、より滑らかな動きを可能にした。また、ヘッド部全体は MDF 材を用いて製作した。MDF 材はレーザカッターを用いた加工がしやすく、今回の製作に必要な強度、重量を十分に満たしていたためである。ヘッドの先端の形状は、MDF を用いて複数のプロトタイプを製作した。飛び出す部分が 30mm から 50mm のものや、滑らかな山なりのもの、急傾斜になっているものなどを製作した。実際に製作したヘッドの先端部を並んだ本に押し当ててどのように浮き出て見えるか観察した。その結果先端が飛び出ているもののほうがよく見えることが分かった。しかし、その角度を急にしすぎると横移動がうまくいかなかったり、飛び出た本が戻らなかったりした。そのため、上手く動作し、かつ綺麗に見える形状を採用した（図 2.29）（図 2.30）。

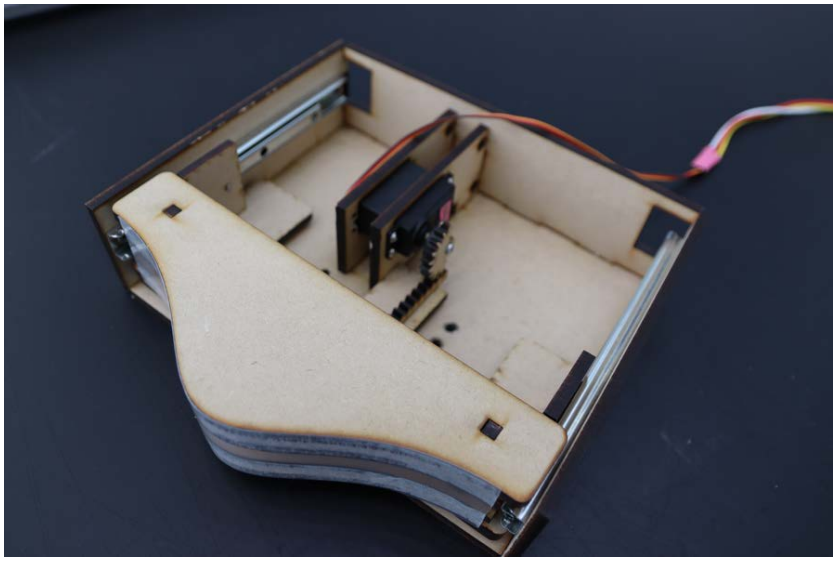


図 2.29　製作したヘッド部：右斜め上

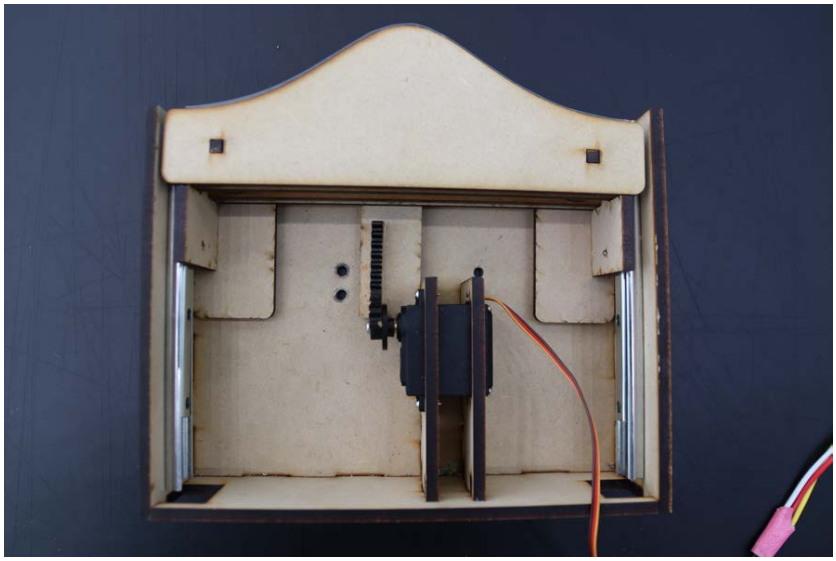


図 2.30　製作したヘッド部：上面

（※文責: 和田颯平）

**表紙班**

今回、動作の際に実際に使った本のサイズは漫画の新書判であり、縦 174mm、横 112mm である。新書判の本は厚みのないものが多く、サイズ感や、重さの面から、POP UP SHELF の実装に適しているため、これらを採用した。

押し出された本が戻る仕組みには、本を 30 度傾けた事による自重を想定していたため、本同士の摩擦が少ないことが求められた。元の本の表紙のままで本の戻りやすさを実験した際、摩擦が大きく本の戻りが悪かった。さらに著作権の関係もあったため、表紙を変え、加工をする必要があった。

初めに、本全体をビニールで覆う案が出た。透明のポリ袋を本のサイズに合わせて切り、本を覆った。しかし、本とビニールとの密着度が悪く思ったような滑りはなかった。また、ビニール素材のブックカバーを購入し、実験をしたがポリ袋との大きな違いはなかった。他にも、シュリンク加工のビニールも考えられたが、手に入れることができず、実験までに至らなかった。

そこで、本の表紙自体の素材を変えた。まず、光沢紙で実験を行なったところ、紙自体の厚みで本への馴じみが悪く、本の滑りも悪かった。次に半光沢紙を使用したところ、表紙としての厚みも適しており、実験でも本の戻りが良かった。よって、POP UP SHELF では半光沢紙を表紙として採用した。

表紙のデザインについては、POP UP SHELF のロゴを活用し、そのロゴを指さしている手をイラストで表現することで、製品の特徴を表した（図 2.31）。


図 2.31　本の表紙

（※文責: 丹野夏海）

### 2.3.3　プログラムの解説

POP UP SHELF を制御するプログラムを記述するにあたって、テストプログラムなどを用いて小分けにして、それぞれの部品をテストすることから始めた。テストした部品は、測距センサ・超音波センサ・サーボモータ・ステッピングモータとドライバモジュール（A4988・DRV8835・L6470）である。その後、これらのプログラムを組み合わせ、POP UP SHELF を制御するメインプログラム（マニュアル・メイン）を記述した。以下、各プログラムについて紹介する。ソースコードについては、付録 C に掲載する。

（※文責: 家山剣）

#### 測距センサの制御

まず、測距センサを動かすために記述したコードについて解説する。

2,3 行目は、使用するライブラリのインポート文である。このプログラムでは、$I^2C$ 通信を行うための Wire.h ライブラリと測距センサを制御するための VL53L1X ライブラリである。6 行目で測距センサのインスタンスを宣言する。9〜26 行目にかけて、setup 関数の処理が行われる。シリアル通信を 9600bps で行うための Serial.begin(9600);、$I^2C$ 通信を行うための Wire.begin(); を記述する。15 行目の sensor.setTimeout(10000); により測距センサからのデータ送信が途絶えた際に、指定した時間以上の反応がなかった場合にタイムアウトを検出することができる。17〜21 行目にかけては、測距センサが接続されていない場合に!sensor.init() によって検出することができる。また、測距センサが検出されなかった場合にシリアルモニタ

に”Failed”の文字列を表示した後に while (1); により、永久ループに突入することで、プログラムの動作が止まる。23〜25 行目は測距センサのモードや値を取得する頻度などの初期設定を記述している。sensor.setDistanceMode(VL53L1X::Long); は測定したい距離に応じてモードを変更する必要がある。今回は 1000mm 以上の測定を行うので Long モードで測定を行った。sensor.startContinuous(50); では、値を取得する頻度を設定することができる。今回は 50ms おきに距離を測定する設定にした。29〜39 行目にかけては loop 関数に入る。31 行目では、sensor.read() の関数により、測距センサが取得した距離値をシリアルモニタに出力する。33〜36 行目にかけて、センサからのデータ送信が途絶えタイムアウトが発生したときに、シリアルモニタに”TIMEOUT”の文字列が出力される。38 行目で改行を行い 1 ループが終了する。

（※文責: 家山剣）

**測距センサとサーボモータの連動制御**

上記のプログラムは、サーボモータと測距センサの連動をテストするためのプログラムである。測距センサに関するプログラムは上述した VL53L1X のものと被るものが多いので省略する。1〜3 行目にかけて必要なライブラリのインポートを行う。測距センサのライブラリに加え、サーボモータを制御するためのライブラリ Servo.h を追加した。6 行目でサーボモータの制御信号を送信するために使用するピンの指定を行う。今回は 8 番のピンを利用する。9〜10 行目でサーボモータと、測距センサのインスタンスを生成する。setup 関数では、先ほど説明したプログラムに加えて、サーボモータの初期化を行う。30〜31 行目で、サーボモータの制御に使用するピン番号の引き渡しと、初期角度を指定する。loop 関数では、測定した距離の値が 1200mm 以上の大きさであればサーボモータの回転角を 0 度に指定し、1200mm 以下であればサーボモータの回転角度を 180 度に指定する。これにより、POP UP SHELF に手を入れ操作していれば本が押し出され、そうでないときは本が沈む動きの表現テストをすることができる。

（※文責: 家山剣）

**DRV8835 を用いたステッピングモータの制御**

次にテストしたドライバモジュールは DRV8835 である。電力周りの問題により、不採用することになったドライバモジュールだが、それ以外の面ではうまく動作していたので、ここで解説することにする。このドライバモジュール環境では、スイッチ・可変抵抗・DC ジャックを用いて回路を組んだ。最初に使用する各ピンの宣言を行う。6〜11 行目にかけて、モータの制御信号を送るピンや、可変抵抗・スイッチの状態を読み取るアナログピンの設定を行う。14,15 行目で、アナログピンに入力された信号を格納する変数を宣言する。次に setup 関数内の解説を行う。シリアルモニタと通信を開始するために Serial.begin(9600); を記述する。22〜27 行目にかけて、モータの制御信号を送るためにピンを OUTPUT に設定し、AENBL ピンと BENBL ピンの初期値を HIGH に設定する。また、スイッチのピンは INPUT に設定し、アナログ信号を受信するように設定する。スイッチピンは初期値を HIGH にしておくことで、ボタンが押されていない状態をデフォルトとして扱う。今回の環境では、押されていない状態を順回転、押している状態を逆回転として扱う。READ_VR と READ_SW の関数を作成し、可変抵抗とスイッチの値を読みだして変数に代入を行う。また、DELAY_WAIT 関数では、可変抵抗の値によって for 文の回す回数を変化させる。これにより、delayMicroseconds(100); の呼び出される回数が変化し、モータの回転速度を制御するこ

とができる。また、抵抗の値を大きくしすぎたり、小さくしすぎると、脱調や電力不足に陥り正常に回転しなくなるので注意が必要である。MOTOR_FOREWARD 関数と MOTOR_REVERSE 関数により回転方向を指定する。digitalWrite(APHASE、 HIGH); と digitalWrite(BPHASE、HIGH); の第 2 引数を HIGH・LOW の組み合わせにより回転方向を制御する。DELAY_WAIT(); によって回転速度を調整している。また、MOTOR_STOP 関数により、モータの動作停止を呼び出す。loop 関数内では、可変抵抗・スイッチの状態を読み取りそれぞれの制御変数の値に合わせてモータの回転を制御する。

（※文責: 家山剣）

**L6470 を用いたステッピングモータの制御**

次にドライバモジュール L6470 のプログラムについて解説する。L6470 のテストプログラムは長いので重要個所についてのみ解説を行う。まず、commands タブにモータの動作関数がまとめてある。細かい動作に関してはコメントアウトにも説明があるのでそちらを参照する。

main プログラムでは、最初にライブラリのインポートを行う。SPI 通信を行うためのライブラリとオーバフローするたびに関数を呼び出す MsTimer2 のライブラリをインクルードする。その後は、ピンの設定を行う。SPI 通信を行うにあたって、Arduino では指定されたピンを利用する必要がある。Arduino 側のデジタルピン 10 番が SS、デジタルピン 11 番が MOSI(Master Out Slave In)、デジタルピン 12 番が MISO(Master In Slave Out)、デジタルピン 13 番が SCK(Serial Clock) に割り当てられたピンでモータに制御信号を送信する。SPI 対応デバイス側からの送信がなければ、MISO は未接続でも動作する。また、それぞれのピンの役割は、MISO がスレーブからマスターへの送信、MOSI がマスターからスレーブへの送信、SCK がマスターとスレーブを同期させるためのクロック信号になる。この 3 本の信号線には複数の SPI 対応デバイスを接続でき、どのデバイスと通信するかを SS により決定する。SS は HIGH/LOW の極性だけなので、デバイスごとに 1 本必要となる。また、同時に 2 つ以上のデバイスを選択することはできない。

setup 関数では、それぞれのピンのモードを最初に設定する。MISO と BUSY ピンは INPUT に設定し、それ以外は OUTPUT にセットする。SPI.begin(); で SPI 通信を開始し、通信方法を SPI.setBitOrder、SPI.setDataMode の 2 つを設定する必要がある。前者は、最上位ビット（MSBFIRST）、後者は、SPI_MODE3 を指定する。L6470 のリセットと、セットアップをコマンドタブにある関数から呼び出し初期化を完了させる。MsTimer2 により、シリアルモニタ用のタイマ割り込みを行い、これが動作するたびに flash 関数が呼びだされ、シリアルモニタに随時モータの状態が表示される。このテストプログラムでは、指定方向に連続回転を行うものである。L6470_run(0, 3000); では、第 1 引数が回転方向、第 2 引数が回転速度を指す。また、loop 関数内のコメントアウト部分では、指定座標への移動や、ホームポジションへの移動など、POP UP SHELF を動作させるにあたって必要になる動作のテストを行った。

（※文責: 家山剣）

**POP UP SHELF の手動制御**

次に実際に成果発表会時に使用した Web サイトに掲載する際に用いたテストプログラムを紹介する。このプログラムでは、POP UP SHELF の目指す動きを表現するため、すべてマニュアル操作で撮影を行った。POP UP SHELF の動きに合わせて人間が動くことで、自分たちの目指し

た動作をより分かりやすくするためにこちらの手法を採用した。ほとんどのプログラムは上述したものと一緒であったので説明は省く。loop 関数内の 100〜172 行目にかけて、6 つの動作を表現したテストプログラムを作成し、動きを再現して撮影を行った。

（※文責: 家山剣）

**POP UP SHELF のメインプログラム**

最後にメインプログラムについて解説する。このプログラムは、POP UP SHELF を自動で制御するために作成した。手を入れた位置に測距センサが反応し、取得した距離値まで、ステッピングモータを回転させヘッド部を移動させる。ヘッド部の移動が終了したのちに、ラックアンドピニオン機構とサーボモータにより本が押し出されるものである。また、これらの動作を迅速に確認するために、LCD ディスプレイに随時情報を表示するプログラムも追加した。

最初に 11〜17 行目にかけて使用するライブラリのインポートを行う。SPI 通信を行うための SPI ライブラリ、サーボモータを制御する Servo ライブラリ、I²C 通信を行うための Wire ライブラリ、測距センサを使用するための VL53L1X ライブラリ、シリアルモニタ用のタイマ割り込みを行うための MsTimer2 ライブラリ、LCD ディスプレイに表示するための rgb_lcd ライブラリを使用した。

20〜26 行目にかけて Arduino で使用するピンの設定を行う。21〜25 行目は SPI 通信を使用してステッピングモータを制御するためのピン（上記参照)、26 行目は、サーボモータに制御信号を送るためのピンである。

29〜36 行目にかけて、取得した各値の数値を格納する変数を宣言する。hand_pos は測距センサで測定した手の位置を格納する変数、hand_pos_con は計測した手の位置に演算処理を行いモータのコントロールに使用するために加工した数値を格納する。head_pos はヘッド部の位置を格納する変数であり、hand_pos の値を用いて随時更新する。th は閾値を表しており、手の微妙な動きを敏感に計測してしまうので閾値を用いて小さな動きは無視するようにした。width は本棚の全長を表す。sensor_val は測距センサが取得した値がそのまま代入される。loop_cnt は loop 関数が呼び出された回数を記録する。

39〜42 行目はデバイスのインスタンスの宣言を行う。VL53L1X sensor; は測距センサ、rgb_lcd lcd; は LCD ディスプレイ、Servo myservo; はサーボモータを表す。45 行目から setup 関数に入る。こちらについても、上記したプログラムに解説が載っているので、そちらを参照してほしい。

95 行目からは loop 関数に入る。中で使われている関数の解説は後ほど随時行う。98 行目にある GetLoopCount(); は、loop 関数が周回した回数を記録して loop_cnt に値を記録し、シリアルモニタに表示する。101 行目の GetSensorValue(); は測距センサが取得した値を sensor_val に格納して、シリアルモニタに表示する。GetHandPosition() では、153〜160 行目に登場する InOut() 関数により、操作状態か否かを TRUE or FALSE で識別し、hand_pos の値を更新する。操作状態であれば、手の位置が随時更新されるため測距センサの取得した値を代入し、そうでない場合は、ヘッド部の位置の更新が発生しないので、hand_pos に head_pos の値を代入する。その後、シリアルモニタに、hand_pos の値を表示する。153〜160 行目の InOut() 関数は、手が本棚を操作している状態か否かを識別する。測距センサの取得した値が、本棚の全長よりも小さいときに、POP Up SHELF を操作している状態と判断する。この関数は、値を boolean で返す。163〜170 行目の TimeOut() 関数で、測距センサの通信の状態を監視し、タイムアウトが発生した際に、シリアルモニタに TIMEOUT の文字列を表示する。

次に、contlor.ino についての解説を行う。ControlMotors() 関数では、InOut() 関数が TRUE を返す間、モータを動かす。その間、常に測距センサ、手の位置を更新し続け、hand_pos_con に han_pos の値をレールに合わせて動きになるよう演算処理を加える。6 行目で head_pos の値を更新し、L6470_goto(hand_pos_con); により、指定座標へ、ヘッド部を移動させる。PushServo(); により、移動先の座標でサーボモータが駆動し、本を実際に押し出す。また、InOut() 関数が FALSE を返す間は、L6470_hardstop(); により、ヘッド部をその場で停止させる。また、PullServo(); により、操作していない間は本を戻す。

次に lcd.ino についての解説を行う。LcdShow() 関数では、lcd.setCursor(0, 0); により一段目の指定をする。測距センサが取得している値をリアルタイムで表示し、迅速に確認するために採用した。具体的には sensor_val の値を表示しており、表示する桁数に合わせて、随時”0”を表示しなければならないので、場合分けを行ってディスプレイに表示する文字列の操作を行う。2 段目には、SendSignal 関数で入力された文字を表示する。

次は、servo.ino の説明を行う。PushServo 関数は、本を押し出す動作で、サーボモータに 0 度を渡すことで駆動する。MiddlePushServo 関数は当初、ヘッド部が移動中に使用する予定であったが、この操作を処理する時間が、本棚の操作において、リアルタイム性を欠く 1 つの原因となっていたので、今回は使うのをとりやめた。PullServo 関数は、サーボモータに 160 度を渡すことで、本を戻す動作を表現する。

次に、signal.ino を解説する。SendSignal() 関数では、シリアル通信を用いて、キーボードからの文字入力を使ってモータの制御を行う。char 型の sign 変数に文字を格納し、4 つの文字を操作コマンドとして読み込む。今回は、”e” ”h” ”r” ”s”の 4 つを使った。e は while(1); により永久ループに突入し、プログラムを強制的に停止させる。h は、ドライバモジュールに L6470_gohome(); を命令し、ヘッド部をホームポジションへ戻したのちに、exit(0); により、プログラムを終了する。r はドライバモジュールにステッピングモータのポジションとデバイスのリセットを行い、初期化を強制的に行う。s は L6470_hardhiz(); により、ステッピングモータをその場で強制停止させる。主に SendSignal 関数では、故障や不具合の発生した場合の対処として作成したプログラムである。

最後に stepping.ino の解説をする。ここには、ステッピングモータを動作させるためのコマンドが関数化して置いてある。L6470 を用いたステッピングモータの制御で解説を行っているので説明を省く。ここにある関数を呼び出して、contlor.ino の ControlMotors() 関数に入れて利用する。

（※文責: 家山剣）

# 第 3 章　中間発表会

## 3.1　Element.01 - ゆーたん

### 3.1.1　中間発表会に向けて

中間発表会の時点では、120mm × 100mm サイズのゆーたんを製作した。実際に製作する Element と同じ方法で、実寸大のおよそ 1/4 のサイズで製作し、課題をあぶりだした。Element はタイル部と LED 部から構成され、タイル部は、拡散板 (乳白色のアクリル板)・導光板・MDF を上から積み重ねることで作成している。導光板は、 透明のアクリル板にレーザーカッターで傷をつけることで作成している。 LED 部では、この 3 枚合わせの板に横から光を照射させることで、板全体を光らせている。これらを用いて、光が順番に流れるようにした。発表会では、見ている人がイメージしやすいように、ミニチュア模型を用いた動画（図 3.1）と CG を用いたイメージ映像（図 3.2）を提示した。公立はこだて未来大学の事務局経由で函館市役所にメールで交渉し、イメージ映像を撮影した。前日までに、日程の調整と撮影の際の注意事項などの確認を行った。撮影の際は、一般の方の通行の妨げや顔が映らないような配慮をしつつ、足元を中心に撮影を行った。また、CG を用いて足元が光っているような編集をした。映像製作については、「Web サイト・ポスター製作」にて後述する。

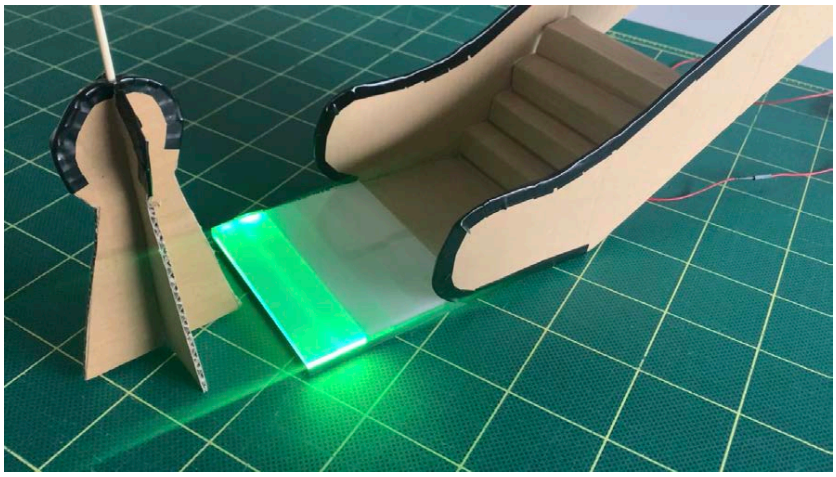


図 3.1　ミニチュア模型を用いた動画の一部

図 3.2　イメージ映像の一部

（※文責: 中村ももこ）

### 3.1.2　中間発表会でのフィードバックを受けて

中間発表会の時点では、3 つの課題・目標があった。1 つ目は、エスカレータに乗るタイミングをよりわかりやすくすることである。現在のミニチュアサイズのものでは数ブロックが同時に光り、それがエスカレータの速さに合わせて動く光り方のパターンしか考案できていない。実際に使用することとなったときにパターンが少ないとそれが有効でなかったときに問題になってしまうので、このパターンが実際にエスカレータに乗りやすくなるかどうか、またその他のパターンは考案できるか、実際に乗りやすいものなのか多くの候補を上げて検討したい。2 つ目は実寸大の Element を製作することである。先述したように現在はミニチュア模型しか製作できていない。実寸大の Element を製作することで、Element の構造はこのままでいいのか検討していきたい。とくに、LED の光量が足りているか、導光板の彫刻はこのままで良いか、設計的に問題がないかを考えていきたい。3 つ目は、評価実験を行うことである。実際に使用したときに製作した構造の問題や光り方パターンの問題など客観的に意見をまとめることで、Element のさらなる改良を行いたい。

今後の展望としては、他の利用用途にも対応できるよう Element を改良することである。エスカレータの乗るタイミングを計るだけでなく、自動ドアやエレベータのドア、回転扉などタイミングのわかりづらい機械だけでなく、人間や車の動きなどの誘導まで応用ができるとより効果的な Interaction Elements になるだろう。また、以下は中間発表での質問とそれに対する回答である。

以下は中間発表会の質問とそれに対する回答である。

Q：エスカレータで自分が乗るべき段が出てくるタイミングがわからないのか、自分が乗るべき段はわかっているがエスカレータに乗るタイミングで身体が動かないのか?
A：エスカレータで自分が乗るべき段が出てくるタイミングがわからない場合を想定している。

Q：どこかを使って実験すると思うが、厚いものを敷くときの事故などもあるのでは？
A：現状のデザインは厚さが 10mm なのでそこまで支障がでないと考えている。場合によっては、つまずきにくい機構を追加しようと考えている。

Q：躓きにくい仕組みとは？

A：段差に緩やかな傾斜を付けようと考えている。

Q：光らせるパターンに工夫はある？
A：他の色を試したり、点滅させてみたりを考えている。

Q：どこかと協力して設置の予定は？
A：実寸大サイズのものを制作した後、評価段階で施設を使う予定である。

Q：エスカレータとの同期は？ゆーたんのみの開発だと互換性の面で不安。
A：現状ではエスカレータとの同期は、Arduino の制御でできる見通しである。

Q：計る対象のタイミングはどのように検討するのか？ (エスカレータにのりやすいタイミングとは？)
A：エスカレータの段が出て来るタイミングがうまく掴めないことが乗りにくいと感じる原因だと考えているため、目に見えていない段をエスカレータの手前で視覚的に提示し、タイミングを掴みやすくする。

（※文責: 老沼響）

## 3.2 Element.02 - bect

### 3.2.1 中間発表会に向けて

中間発表会の時点では、使用方法や「察してほしい」をどのように表現するかという点に重点を置いた。何度もアイデア出しや意見交換を行っていた結果、2 つの「察してほしい」を伝えるためのアイデアを考案した。またそのアイデアの検証・検討を行なった。中間発表会時点では、検討したアイデアを GIF などを使い、説明する資料を製作した。また、検討中であったアイデアのイメージを共有するために映像製作と、インフォグラフィックスの製作を行なった。

（※文責: 専徒礼樹）

### 3.2.2 中間発表会でのフィードバックを受けて

「中間発表会からの転換」で述べたように方針転換を行った。中間発表会を受け、「察してほしい」ことを周囲に明示するということは「察してほしい」という考えから矛盾しているのではないかという指摘を受けた。その場で返答することができず、後日グループ内で話し合いの場を設けた。その結果、指摘された点は根本的な問題点であり、製作中の Element に関して、見直しをする必要があるという結論に至った。具体的な問題点は、指摘のあった察してもらうを可視化することは「察してほしい」という行為に対して矛盾するという点だった。また、関連して操作方法、運用方法にも問題点が生じていた。当初、操作方法はデバイスの色で感情を表し、その色を入力するという操作を考えていた。そういった入力方法だと、感情を自ら入力することになり、当初のこのデバイスの対象であった電車やバスなどで、声をかけることができないといった繊細な人は使用しなくなるのではないかといった問題があった。また、感情という定義が非常に難しいテーマを扱うことでこれから先の作業にも支障が出ることが懸念されたため、課題設定から見直しをすることになった。

以下は中間発表会での質問とそれに対する回答である。

Q：たくさんの人がデバイスを持っているときは、誰が（どの方向から）発信しているのかがわからないのではないでしょうか。
A：現状では大人数での想定をしていませんでした。少人数での想定でしたが、問題としてあがって来ると思うので考えたいと思います。

Q：受け取るものと、送信する側で違うものを持つのか。
A：受け取る側も受信する側も同じデバイスを持って使用することを考えています。

Q：察する側のユーザは察したい人がいるという前提ですか？
A：友人間などで使用などが考えられます。

Q：自分の感情を伝播させるだけで、気を遣う部分の効果はないのか。
A：今のところ気を遣わせるような効果は想定していません。

Q：色弱の人への対応はどうするのか 。
A：まったくの想定外だったので、今後検討したいと思います。

Q：動画を見たときにイメージ映像とあったが、実際に使っていたものは、実際に製作した完成形か?
A：完成形ではないです。Sphero を用いて検証をしている段階で完成形にスフィロを用いるかはわかりません。

Q：今後は形も変わるかもしれないんですか？
A：変わる可能性もあります。

Q：現在、ボール型デバイスでやられていると思うのですが、スマートフォンアプリなど、ほかの普及されているデバイスで実装する予定はありますか？
A：現段階では、まだ形が決まっていないので、未定です。

Q：色だけじゃ伝わらなさそう。見逃しちゃう。
A：音や振動などの通知を追加することを検討しています。

Q：Feeling Catcher では、喜びと悲しみを表すとされていましたが今後他の感情も実装する想定はありますか?
A：様々な感情も表せる様にしたいと考えています。

Q：察してほしいと思う人は、言葉にできないから伝えるのか？察してほしい人にとってそれを色で可視化してしまうことで察するということではなくなるのでは？
A：(回答できず)

（※文責: 専徒礼樹）

## 3.3 Element.03 - POP UP SHELF

### 3.3.1 中間発表会に向けて

中間発表会時点では、完成形の本の動きをコマ撮りによって表現したイメージ映像、考案した現段階での機構を簡易的に図解した資料を製作した。イメージ映像を製作する際、最終段階を見据えた本の動きや操作方法を理解できるよう心がけた。また機構の図解資料を製作する際、機構の説明とともに、その手順を表したインフォグラフィックス、その一連の流れを再現した GIF 素材を製作した。

（※文責: 丹野夏海）

### 3.3.2 中間発表会でのフィードバックを受けて

中間発表会時点の予定は、まず機構のプロトタイピングを進めることである。ここで示す機構は、ヘッド部を指の指す位置までスライドする機構、ヘッド部内部の本を押し出す機構、指の動きを感知するセンサの 3 つに分類される。

初めに、ヘッド部をスライドする機構は、ヘッド部を装着したタイミングベルトをステッピングモータにより水平移動させる機構を想定していた。また、ヘッド部による本を押し出す機構については、2 つの案が出た。1 つ目は複数の電動シリンダを並べ、それぞれが指定された長さで垂直方向に動くことで、山なりに飛び出す様子や、波打つ様子を再現するというものである。2 つ目はサーボモータの回転運動を利用しスライダクランク構造を応用することで垂直運動に変える方法を想定した。　最後に、センサは測距センサ（ToF センサなど）を使用する想定であった。しかし、指以外のものに反応する誤作動を防ぐため、指以外のものを感知しないよう改良を重ねる必要があった。これらの機構を製作したうえで、本が浮き出たまま滑らかに動くか、波打つ動きを再現できているかなども試行を重ねていく予定であった。中間発表会での、質問にあったように、本との摩擦の問題があるので、本の痛みが少なくなる工夫の検討を続けた。中間発表会時点では、新しい体験を付与することを目的としていたが、より使いやすいインタフェースになるよう、実際に製作したものをユーザに操作してもらい、評価を行なった上でより実用性のあるものを目指した。

以下は中間発表会での質問とそれに対する回答である。

Q：どこで使われることを想定しているのか。
A：どこでも使えることを想定している。しかし、本棚が斜めに置ける場所を前提にする特殊な環境が必要である。

Q：飛び出した本を棚に引っ込める仕組みはどのように実現するのか。
A：本棚を斜めにすることで、重力によって戻るように設計する予定である。

Q：本の間の摩擦はどうなのか
A：今後詳細を検討する。

Q：今後波打つ動きを再現するとあるが、ヘッドを 2 つに増やすのか。
A：レールの上にあるヘッド部が水平方向に移動する仕組みである。そしてこのヘッド部が部分

的に前のものを押し出す。

（※文責: 丹野夏海）

## 3.4　Web サイト・ポスター製作

**デザインコンセプト**

Web サイト・ポスターのデザインは、統一感を出すため、白を基調としたモノトーンな色使いを意識した。意図としては、Interaction Elements という未来の部品を製作するプロジェクトとして、色褪せないシンプルなデザインを取り入れた。

（※文責: 古町昂大）

**ロゴの製作**

Interaction Elements プロジェクトを進めるにあたり、プロジェクトの認知度の向上に向けて、ロゴの製作を行った。モチーフは、Interaction Elements の頭文字である「I」と「E」から製作した。シンプルなひとつのアイデアから新しい未来の部品を製作していくプロジェクトのコンセプトの下、シンプルな長方形と洗練された黒の線と角をとった形のみで製作した。長方形と長方形の隙間を黄金比率にて設計した。（図 3.3）

図 3.3　Interaction Element のロゴ

（※文責: 古町昂大）

### 3.4.1　Web サイト製作

**デザイン工程**

Web サイト製作の流れは、まずどんな情報を入れるか、見せたいかを決定した後、Adobe XD（図 3.4）を用いて、Web サイトのプロトタイプを製作した。まだ用意できていない映像や写真・図解・文章はダミーを配置して、素材を挿入するだけの状態にしておいた。そこから、必要な映像作成と画像や図解、文章作成の依頼を各担当に指示し、製作を進めた。また、プロトタイプを作成することによって、早い段階でコーディング担当に、大まかなイメージの共有と実装を任せること

ができた。

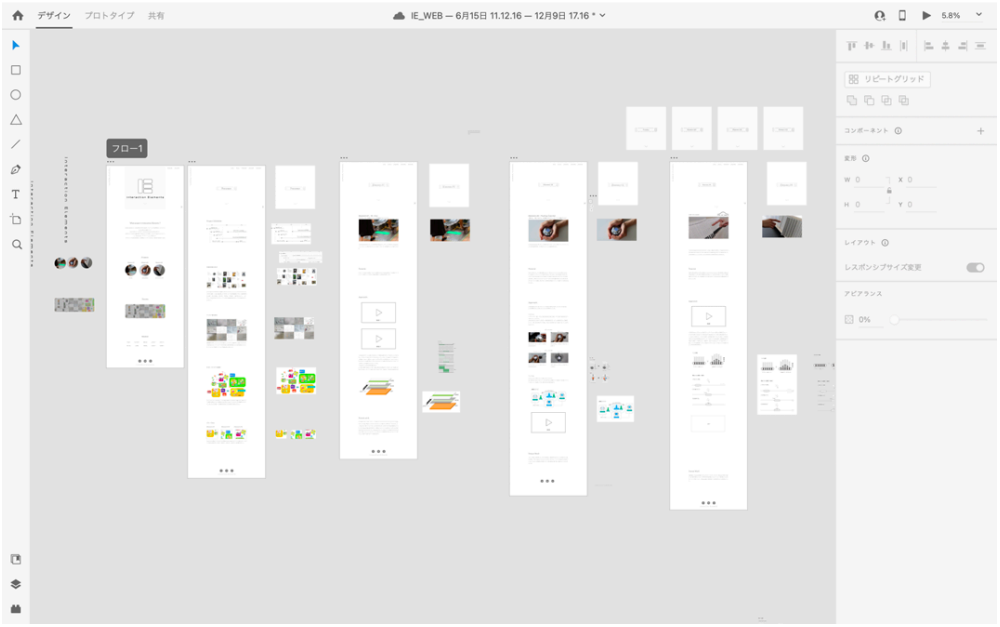



図 3.4　XD を用いた Web サイトプロトタイプ

（※文責: 古町昂大）

**動画製作**

動画製作では各グループのイメージ映像の製作を行った。製作の流れとしては、大まかな絵コンテの製作、それをもとに実際に学内または函館市役所での撮影を行い、後日撮影した素材を編集することで 1 つの映像を作成した。グループによっては映像に CG の合成やモーショングラフィックスの作成が必要だったので、要望に沿った映像の製作を行った。また、タイトルのフォントや配色を Web サイトと統一し、デザインコンセプトを崩さないよう心がけた。CG 合成・モーショングラフィックスの製作に Adobe After Effects。動画素材全般の編集・BGM の追加には Adobe Premiere Pro。その他特殊な図形の作成に Adobe Illustrator。GIF の作成・写真の加工に Adobe Photoshop を使用した。

（※文責: 佐々木皓大）

**コーディング工程**

コーディングでは、「図解！ HTMLCSS のツボとコツがゼッタイにわかる本　著者：中田　亨」[9] を参考にし、Visual Studio Code（以降は VSC と記述する）を用いて行った。文字や写真、横幅などの大きさは、ピクセル数で具体的な数値として指定した。すべての画面の大きさに対応できるように、割合を用いて表示するのが今後の課題である。製作期間はおよそ 2 週間を要した。特に最後の 3 日間には、各グループの文章やレイアウト変更があり、作業が集中した。CSS の設定はわかりやすく、変更しやすいコーディングをするのが今後の課題である。

（※文責: 兒島涼太）

**サーバ**

サーバの公開には、GitHub Pages を用いた。あらかじめ VSC でコーディングしたものを貼り付け、用いた写真などをアップロードすることで、URL を発行することができる。コーディングをする際、ページごとにファイルを分けた。例えば、Web ページのメインである Home ページには Home.html と名前を付けた。その際、Home.html のみを GitHub に張り付ける際、index.html に張り付けなければならない。しかし、GitHub 上に、Home.html を作成し、張り付けたためにうまくいかず、手間取った。

（※文責: 兒島涼太）

### 3.4.2 ポスター製作

ポスターは Adobe Illustrator で製作した。タイトルや文章のフォントを Web サイトと合わせることにより、デザインのコンセプトをプロジェクト内で統一した。また、Interaction Elements のロゴをタイトルの左隣に大きめに配置することで強調させ、プロジェクトのイメージ付けがされるように意識した。また、Element の紹介部分では、概要文と、仕組みについての詳細を説明している文を分けることで、Element の概要が簡潔に伝わるような工夫をした。仕組みの説明の文の上にはそれを説明するインフォグラフィックスを入れ、構造についての理解が深まるよう目指した。

（※文責: 丹野夏海）

# 第 4 章　成果発表会

## 4.1　Element.01 - ゆーたん

### 4.1.1　成果発表会に向けて

私たちは成果発表会に向けて Element の製作のほかに、撮影やインフォグラフィックスの製作にも力を入れた。成果発表会で Element を実際に使っている映像を使用したかったので中間発表と同様に、公立はこだて未来大学の事務経由で函館市役所にメールで交渉し、市役所内にあるエスカレータで撮影を行った。撮影の際は前日までに日程の調整と注意事項の確認、撮影時には周りへの配慮をして行った。今回の撮影ではエスカレータに使用する光り方のパターンを 3 つ用意し、それぞれ Element のみと実際に人が使っているところを撮影した。（図 4.1）撮影をスムーズに行えるようスイッチでパターンの切り替え、光のスピード調節を可変抵抗で行えるようにプログラムを書いた。また Web サイトやポスターで使用する写真や、コンセプトムービーで使う映像などを別途大学で撮影を行った。（図 4.2）こちらの撮影はホワイトバックと照明などを使い、Element が目立つよう工夫をした。照明の当たり方だけでなくホワイトバックのしわなどが背景に映ってしまうとノイズになってしまうので、対策としてアイロンやテープを使い撮影を行った。実際のコンセプトムービーでは CG を使い、エスカレータの階段部分やエレベータの扉などを表現した。（図 4.3）インフォグラフィックスはポスターと Web サイトで使用する Element の機構と光り方が分かる画像形式のもの（図 4.4）と、光り方のわかる GIF 形式のモノ（図 4.5）を PowerPoint で製作した。GIF 形式のものは足跡があるタイプ（図 4.6）も用意することで使用するイメージをよりわかるように工夫した。

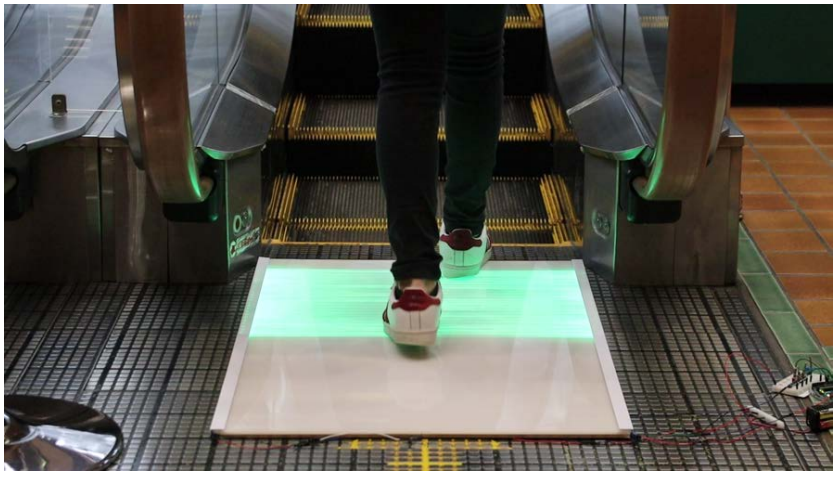


図 4.1　函館市役所での撮影の様子

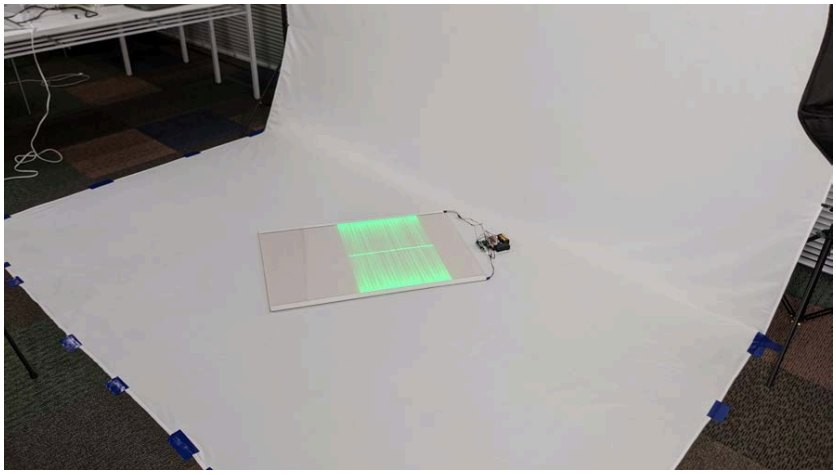


図 4.2　ゆーたんの撮影の様子

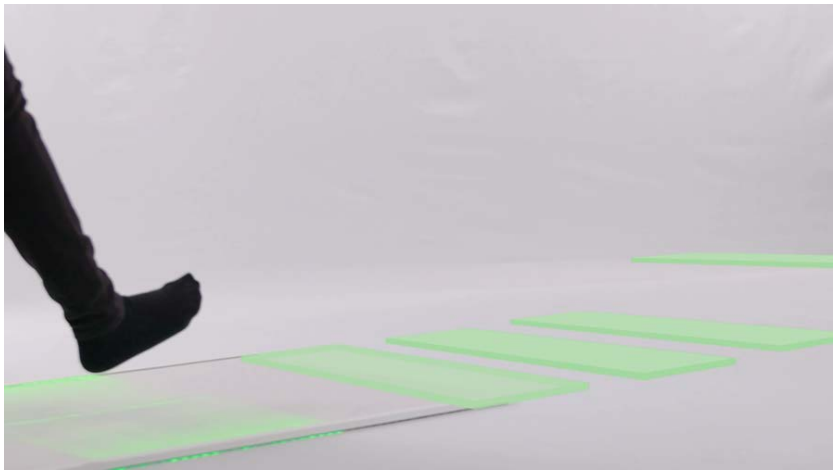


図 4.3　コンセプト動画の一部


図 4.4　ゆーたんの機構画像

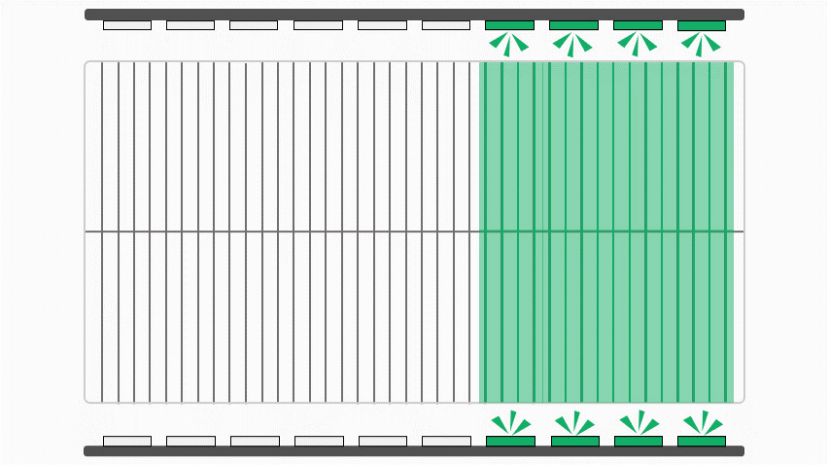


図 4.5　ゆーたんの光り方イメージ GIF

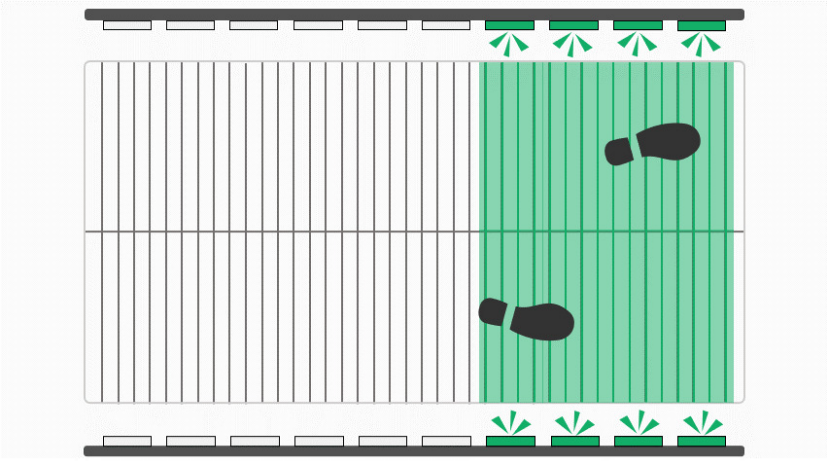


図 4.6　ゆーたんの光り方イメージ GIF 足跡有

（※文責: 老沼響）

### 4.1.2　成果発表会でのフィードバックを受けて

成果発表会までの製作とフィードバックを受けて、大きく 3 つの課題があがった。1 つ目は光り方のパターンを増やすことである。今回はエスカレータ用の 3 パターンとエレベータ用の 1 パターンを考案した。色の変化や左右で違う動きをするパターンなどから、もっと多くの実用例を考案できたと考えられる。色を用いたプログラムは、実際に光らせることはできたが、どういうものに使えそうか使用者がどんな反応をするか議論、実装する時間を確保することができなかった。また左右で違う光り方ができるので、道の中央に設置し、交通などの誘導ができるのでは？という意見もあがったが、こちらも実装ができなかった。さらに、実際のフィードバックでは、直線的な誘導しかできていないが曲線的、例えばカーブなどの誘導はできるのか？という質問があった。このように今回はエレベータ、エスカレータでの使用例しか考案できなかったが、Element は様々な利用用途があることがわかった。しかし、今回のプロジェクト期間内においてはその他の様々な利点を活かしきることができなかった。

2 つ目は拡張性があまり持たせられなかったことである。Element 製作開始当初、現在の半分の大きさで左右に分解でき、用途に合わせて前後左右で着脱可能なものを構想していた。また、プロ

グラムも用途に合わせてスイッチで LED テープの光り方が変更ができることも構想していた。しかし、着脱をする機構を考えられなかったことから実装ができなかった。そのため、今回のゆーたんは函館市役所のエスカレータや公立はこだて未来大学のエレベータと同じサイズのものにしか対応ができていない。着脱可能なゆーたんが実装できていたら、横幅のあるエスカレータや扉の大きいエレベータなど対応できる幅が広がりさらに拡張性が広がったと考えられる。

3 つ目は評価実験が行えていないことである。後期開始時の予定では Element 製作後エスカレータに乗りづらいと感じる人を集めてどのパターンが乗りやすいか、Element があったほうが乗りやすいのかというアンケートを取る予定だった。しかし、Element 製作に時間がかかってしまったのと被験者を集めるのに時間がかかることで、評価実験を行えなかった。そのため、Element に効果があるのかどうか客観的に評価することができなかった。チームメンバーはエスカレータの段が突然出てくるわけではなくなるので乗りやすくなり、事故防止に繋がると評価している。しかし、これが多くの人に当てはまるわけではない。さらに、エスカレータに乗ることのできる人から見てどのように感じるかも評価の必要があった。このような被験者からの客観的な評価から、さらに誘導しやすい方法やパターンを考案できた可能性がある。

成果発表会でのフィードバックを受け、ゆーたんはまだ多くの改良余地が残っていること、それに伴う対応できるものの増加や拡張性の広がりがあることがわかった。この Element が様々な現場でどう使用されてどう応用していくべきなのか、活用例が出てくることを望む。

また、以下は成果発表会での質問とそれに対する回答である。

F：ゆーたんはエスカレータに乗る際のタイミングがわかりづらいという私自身が子供のころに感じたことをデザインで改善するという発想が素晴らしいと思う。

F：ゆーたんがとても良いと思いました。タイミングをずらすという発想が良いと思いました。

F：「ゆーたん」ではエレベータでの使用だけでなく駅や自動ドアといったものに使用用途を拡張していたのが素晴らしいと感じた。

Q：プログラム難しい？
A：LED テープを制御するだけなので難しくはない。

Q：まっすぐ以外に曲がる誘導もできるんですか？
A：正直想定してなかった。現状はできない。

Q：前例はあるのか？
A：調べた範囲では見つからなかった。類似するものを上げるなら、LED で光るカーブの標識などが該当する。

Q：類似のもので比較するとどんな利点があげられるか
A：類似例は道など動きのないものを誘導するが、ゆーたんは動く機械を誘導する。そのあたりで差別化できている。

Q：エスカレータに乗れるようになったのか？
A：メンバーで実験した限りでは、エスカレータの段がいきなり出てくることがなくなったので乗りやすくなった。

（※文責: 老沼響）

## 4.2 Element.02 - bect

### 4.2.1 成果発表会に向けて

成果発表会前 1 週間は、成果発表会に向けて Web サイトの内容を充実させる作業を行った。Web サイトでは、「Concept」「Approach」「Mechanism」「Futurework」の分野で文章と写真を用いた製品の紹介に加え、作成したコンセプトムービーも掲載した。コンセプトムービーは、実際に完成したデバイスを動作させている様子を撮影した。しかし、撮影時に M5Stick C の充電が頻繁に切れるという問題が発生した。この問題は最終的に原因が特定できなかったが、M5Stick C がサーボモータを起動する際に一度再起動してしまうことが調べた中で有力だった。撮影時になんとか電力を保つ方法を模索したところ、PowerC ハットという M5Stick C 専用の充電モジュールがあったため、それを充電し M5Stick C と接続することで 20 分ほど起動し続けることができた。このように撮影は PowerC ハットを接続した状態で行い、複数回のカットを撮影した。また、同時に写真撮影も行い、これを Web サイトに掲載した。インフォグラフィックスは、ラックアンドピニオンの動作とステッピングモータによる回転部の制御について製作し、一目見て動作がイメージできるように工夫した。また、静止画だけでは自分たちの製作したデバイス内部の動きが表現できていないと感じ、Adobe After Effects を用いたモーショングラフィックスの製作も行った。サーボモータの回転によってラックアンドピニオンが動作している様子と、Processing の GUI を操作してステッピングモータの回転を制御している様子の 2 点を製作した。

図 4.7　bect の撮影の様子

（※文責: 佐々木皓大）

### 4.2.2 成果発表会でのフィードバックを受けて

今回の製作では、最初に考えていたアイデアを十分に実現できたとは言えない。最終発表においても、未完成の部分が多く疑問にあげられた。そこで、今回の製作における反省点とそれに対する

現状の解決策を記す。

プログラムの面においては、現状ステッピングモータの制御、ラックアンドピニオンによる押し出しの機能の 2 つを搭載できた。今後の改善点として、実現できなかった、2 つのデバイスを通信させてお互いを常に向き合わせるという機能の開発が考えられる。これは今回の製作において、ビーコン等の知識不足、通信状態を常に安定させる方法の実現に苦戦したため断念した。ビーコンは公立はこだて未来大学に設置してあるものを使用することで学校内のみであるが実現することはできる。しかし、通信環境という問題においては大学の無線環境では不安点が多い。これらの問題をどう解決するかが今後の課題といえる。次に、模型製作の面だが、現状で解決可能な問題とできない問題がある。まず、M5Stick C の充電が持たないという問題は、今回無理やりバッテリーを取り付けて起動していた。これを長時間安定した動作にすることが改善点となるが、現状はバッテリーを外から充電可能にすることで充電が切れても給電することで再度使用可能にするという案が考えられる。これは、模型内部の構造を変えることになり、デバイスのサイズが大きくなる可能性があることを考慮し作り直す。次に、膜をデバイスに固定する方法を改善する必要がある。現状、釘を膜に貫通させてデバイスに挿し込み、釘の先端をホットボンドで固定している。しかし、これでは膜が破れても取り替えることができず扱いづらかった。これを別の方法で固定したいのだが、現状解決策は考えられていない。そもそも膜を張る場所自体を再考する必要があるため、デバイスの形を大きく変えることも考えられる。この点は今後より熟慮していく必要がある。また、ラックアンドピニオン先頭につけてたヘッドの部分についてだが、今回の製作では思い付きで製作したヘッドの形を試した。今後より違う形を考察するとともに、それらがユーザにどのような印象を与えるのかについても考える必要がある。

また、以下は成果発表会での質問とそれに対する回答である。

Q：ものをよく見ると、M5Stick C を使っているが、どれぐらいの時間使えますか。
A：バッテリーを使いながら。20 分ほど持ちましたが、実際に使用するとなると厳しいので再検討の必要があると思います。

Q：視覚と触覚で捉えると書いてあるが、触覚はわかるが、視覚要素の良さは？
A：膜を押し出すユニークな動きを視覚的に捉えられていると思います。

Q：触覚で方向がわかるデバイスをなぜ作ろうと思ったのですか？
A：ボールを触っていて、むにゅっとする動きから着想をえました。ここから方向を示すことができるのではないかと思いました。

Q：手に持つことで、方向を示してくれるが、いきたい方向とはどのようになっていますか？いきたい方向とは、デバイスを触っている人がいきたい方向に誘導してくれるということ？どのようにして、いきたい方向へ案内してくれる？
A：通信した 2 つのデバイスが常に互いを向き合っていることを想定したアイデアでしたが、今回はその機能までは搭載できませんでした。

Q：Web の最後にある FutureWork にヘッド部分についての記述があり、写真から 4 つのヘッド部分の製作を実際に行ったと思うのですが、動作や感覚についてそれぞれどのような違いが表れたのですか？実験などはしてないと思われるので、製作した人の主観でいいので聞きたいです。
A：1 番左の時を試した時に、小さい赤ちゃんに触られているかのような感覚を得ることができました。1 番右は、力が加わるところが小さいので、1 番飛び出るような、押されている感じがし

ました。他は、ただ何かに押されているような感覚というだけでした。

Q：どこで使いますか。
A：まだ、デバイスの大きさや機能が実用的とは言えませんが、将来的にショッピングモールなどで離れ離れになった親子を引き合わせるという状況を想定しています。

（※文責: 佐々木皓大）

## 4.3 Element.03 - POP UP SHELF

### 4.3.1 成果発表会に向けて

私たちは成果発表会に向けた製作のほかに、製作物の撮影やインフォグラフィックスの製作にもこだわった。成果発表会では、Element を綺麗に見せるため外装にネジやコードが見えないように仕上げた。撮影会ではホワイトバックや照明を用いることで、製作物の見せ方についても試行を繰り返し、見栄えが良くなるよう努力をした。Web サイトに載せるための動画や全体像の写真、機構ごとの写真等の撮影を行った。これらの撮影は公立はこだて未来大学のアトリエや 1 階のオープンスペースで行った。撮影毎にホワイトバックのアイロンがけを行い少しでも綺麗に見えるよう努めた。撮影した動画から Adobe After Effects を用いて編集したコンセプトムービーと実際の動作の様子が分かる動画の 2 つを製作した。インフォグラフィックスの製作は中間発表会に向けて製作した。中間発表会時点ではヘッド部の機構が定まっていなかったため、その時点で有力候補だった 2 パターンのインフォグラフィックスを製作した。成果発表会では製作したヘッド部をもとにインフォグラフィックスを製作した。また、レールやセンサの動きが分かるインフォグラフィックスも併せて製作した。加えて、実際の動きが直感的にわかるような GIF 形式の動的なインフォグラフィックスを Adobe Illustrator のデータをもとに Adobe After Effects を用いて製作した。これらのインフォグラフィックスは Web サイトや発表会用のポスター、学会に提出したポスター等に利用した。

図 4.8　POP UP SHELF の撮影の様子

（※文責: 和田颯平）

### 4.3.2　成果発表会でのフィードバックを受けて

成果発表会での質問を受けて、今後の改善点が浮き彫りになった。

はじめに、静音化と高速化を行うことで、より直感的かつ快適な操作を実現できると考えられる。静音化することにより、図書館や書店など静かさが求められる場所に配置することが可能になる。静音化を実現するためにヘッド部の摩擦を軽減したり静かなタイミングベルトやステッピングモータを使用することが挙げられる。また、モータの振動により本棚が振動してしまい、部品が擦れて異音が鳴っていた。この音を防ぐために、ボールベアリングタイプのレールを用いることで改善できると考えられる。また、部品の接着方法や接着箇所を考える必要がある。今回の製作ではメンテナンスをしやすいように、本棚裏面にマスキングテープを用いて Arduino やブレッドボード等を固定した。これらを本棚にねじを用いて固定することでより静音化することができる。今回のプロジェクト学習では、予算の関係で使用できるモータやセンサ、ドライバ等が限られていた。高速化を実現するために回転速度の速いモータや座標の読み取りから伝達が速いセンサ、より高性能なモータドライバ等を用いることが挙げられる。このようにして直感的かつ快適な操作を実現することで、より人間の行動に新たな体験を与えることができると考えられる。

今回はこのアイデアを実現させるために、1 人のみの操作に絞って製作したため、測距センサ、レールとヘッド部の仕組みを採用した。このように測距センサやレールなどを用いることにより数値を取得してから動作するまでの仕組みを単純化し高速化することができた。計測方式として、測距センサの代わりにモーションキャプチャを使用して手や指先の動きを認識し、レールとヘッド部を複数のシリンダによって制御することで、複数人での操作や、ジェスチャに応じたエフェクトをフィードバックとしてユーザに付与することができると考えられる。モーションキャプチャは高価であることや 1 人を想定したことから今回は測距センサを用いたが複数人での操作を想定する際は実用的であると考える。また、シリンダも同様に高価なことや同時制御が難しいこと、小型化や軽量化が難しいことから断念した。しかし、タイミングベルトはその性質上独立したヘッドを 1 つまでしか制御できないことから 2 人以上の使用を想定する際は小型のシリンダを用いることが望ましい。さらに、製作したヘッド部には摩擦を軽減するため、ポリプロピレンの膜を貼り本を傷つけないよう配慮したが軽度の傷が生じた。加えて、今回製作したヘッド部では一定のサイズの本のみ駆動可能だが、シリンダを用いて本の初期位置を上手くずらして制御することにより異なるサイズの本を駆動可能になると考える。

本を取り出した際、その本と本の隙間が生まれてしまうという欠点がある。本棚の側面にばねの付いたブックエンドを設置することにより自動で本の隙間を埋めて常に自然な動作に見せることができると考えられる。

最後に、POP UP SHELF のアイデアは本や資料等の書類、DVD 等のディスクケースなどのリスト化されたモノへの応用ができると考えられる。本以外の物で実装する際は、ヘッドのサイズを変えたり、形状を考えたりする必要がある。また、ディスクケースは本と比べ重さが軽いため、棚の角度を急にし、摩擦を少しでも小さくするなどして押し出した後に元の位置に戻るようにしなければならない。資料などの書類を押し出す際は紙が痛まないように注意しつつ紙がしならないようにするため機構全体を再考する必要がある。このほかにもハンガーにかけられた衣服を押し出すことも考えた。これは、ハンガーラックが波上に動くことにより、ハンガーにより並べられた服をポップアップするというものだ。私たちはこれらの応用方法を考えているが、物理的なものをポッ

プアップすることが広く周知され様々な活用例が出てくることを期待する。

（※文責: 和田颯平）

また、以下は成果発表会での質問とそれに対する回答である。

Q：測距センサは光に弱いが、環境の影響を受けて不具合が出ないのか。
A：太陽光には弱いが、大学内では不具合なく操作可能だった。アトリエでも使えたので、ある程度日光が入ってくる室内でも使用可能だと考えられる。

Q：実際に現場で使う時、通常複数人の操作が考えられるが、それは可能なのか。
A：今回は、予算の関係上、1 人のみが操作する想定で製作した。しかし、後ろに複数のシリンダーを配置し、本を押し出すことができれば可能になると考えられる。

Q：様々なセンサがあるなかで、なぜ測距センサを選んだのか。
A：リアルタイム性を欠かせないために測距センサを選んだが、他のセンサーでの実験ができていないので、最適であるかはわかっていない。

Q：反応するのは指先だけなのか。
A：手以外にも反応する。

Q：何回も操作を繰り返して、本が痛むことはあるのか。
A：実際に多くの試行回数をかさねていない、検証ができていない。しかし、本棚、ヘッド部にアクリルファイルをつけることで、できるだけ摩擦を抑えた。

Q：本を取り出した後の隙間はどうするのか。
A：私たちの中でも課題としてあったが、今回は薄い本を 1,2 冊しか取り出さない想定で製作したため隙間は考慮しなかった。

Q：本を指でなぞるのがとても早いが、何回か往復しても誤作動は起きないのか。
A：遅延はあるが、追従してくれる程度には製作できた。

（※文責: 丹野夏海）

## 4.4　Web サイト・ポスター製作

**デザインコンセプト**

Web サイト・ポスターのデザインは、中間発表会から黒（#727171）と白（#ffffff）のカラーを黒（#303030）と白（#fcfcfc）に変更した。意図としては、文字と背景の差を作り、より読みやすく、メリハリのある Web ページにしたかったからである。また、後期にかけて製作物のクオリティ向上に伴い、より洗練された Web ページを目指して、デザインの改善を進めた。

（※文責: 古町昂大）

### 4.4.1 Web サイト製作

**デザイン工程**

基本的に、アートディレクターがプロジェクトのデザインコンセプトを基に、各チームの製作物や動画のディレクションを行っていった。その方向性に合わせてその他デザイン班と協力しながら製作を進めていった。

（※文責: 古町昂大）

**Web サイト設計**

ページ欄は Home, Process, Element.01, Element.02, Element.03 で構成した。Home のページは、トップのロゴ、Interaction Elements のプロジェクト紹介、各 Element の紹介、Process ページへのジャンプボタン、プロジェクトメンバーと教員紹介、その他製作物のクレジットを掲載した。各 Element の紹介欄では、Element.01, Element.02, Element.03 に別れており、Element の画像をクリックすると各 Element のページにジャンプするように設計した。

Process ページはプロジェクトスケジュール、前置詞図鑑の製作、アイデアの製作、KJ 法・アイデアの整理、グループ分けについて掲載した。各 Element ページは、トップに Element の名前とその概要とロゴ、Concept、Concept 動画、Approach, Mechanism, Element のインフォグラフィックス、Future Work について掲載した。Concept には各 Element のアイデアの発想や製作における基本的構想について説明している。Concept 動画にはそれぞれの使用シーンを切り取りながら、短い時間で Element について理解がしやすいよう製作した。詳しくは 4.2.?に記載。Approach には各 Element における、アイデアの検討過程、そのアイデアに決定した理由、また製作における目標や分担などの設定について説明している。Mechanism には Element の Arduino やアクリルなどの素材や機材、ギアやベルト駆動などの動作方法、大まかなプログラムについて説明している。動作方法については外装のみでは説明仕切れない、また写真等の画像では詳細が掴みきれない部分が多いことから、Element 内部やその動作パターンについてのインフォグラフィックスを製作した。Future Work には、Element の今後の展望について説明している。現段階で製作した Element が持っている可能性を鑑みながら、改善点はもちろん、発展できる点についても、説明している。

（※文責: 古町昂大）

**XD によるデザイン**

Adobe XD を用いて、中間発表会のデータを用いて改良を行った。大きく変更した点は 2 つある。

1 つ目は、Element を撮影した画像やインフォグラフィックスなどのビジュアルを多めに入れたことである。また、視覚的にわかりやすく、また文字だけでは飽きてしまうデザインにならないように意識をした。中間発表会までのデザインでは項目を素直に縦に並べたものだったが、最終発表では、ビジュアルと文章が横並びになっているデザインにした。

2 つ目は、画像などのビジュアル素材を svg という拡張子で統一した。最終発表会時の Web サイトはビジュアルを多めに入れたことによって画像ファイルの粗さ（解像度のばらつき）が気にな

るようになった。その部分を改善するために、XD で作成した画像や、インフォグラフィックスを XD で一括で管理し、画像を出力する際は、XD による形式を統一した svg 素材を出力した。この svg 素材はその他拡張子（png、jpg）と違って、ベクター素材として出力されるため、画像素材の解像度の粗さを一定にすることができる。また、撮影した画像等はベクター素材として出力されないため、撮影する際に一眼レフカメラの解像度の設定、画像比率の設定、露出や画角などの基本的な設定を統一して、撮影した画像等にばらつきが出ないように意識した。

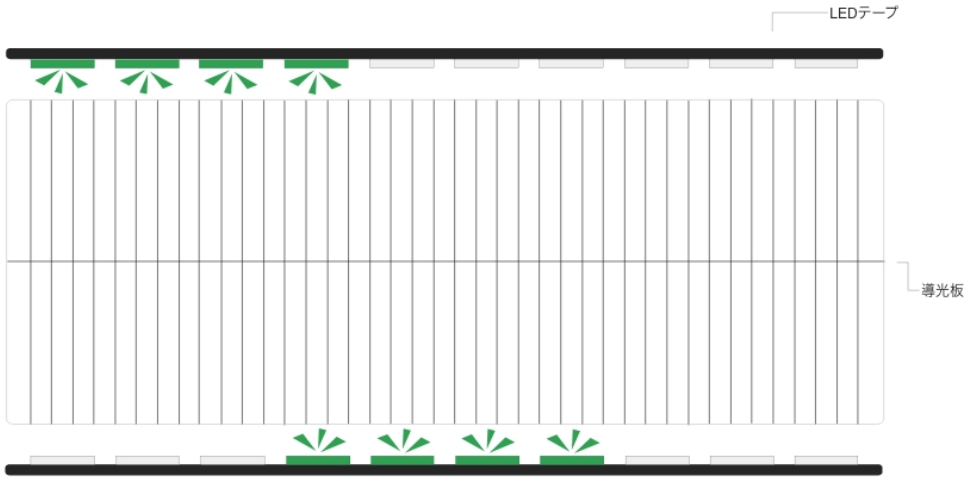
（※文責: 古町昂大）

**インフォグラフィックスのデザイン**

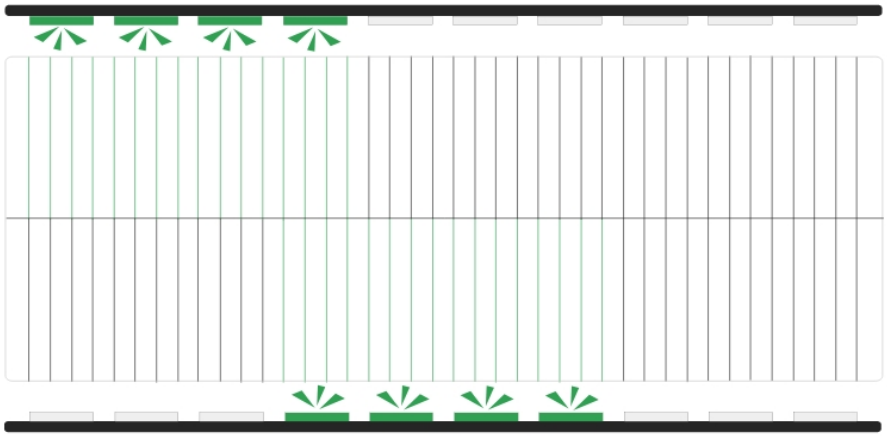
Adobe Illustrator を用いて、インフォグラフィックスの製作を行った。全体的に意識した点は、より専門的な知識を用いて説明する部分や、文章のみでは理解しにくい部分、また実際にプロダクトを見る際に見えない部分を重点的に製作した点である。グループ 1 ではゆーたんの使用素材である、LED テープ、導光版、拡散板の説明や、導光板と拡散板を用いた発光させる原理、足跡を用いた GIF 素材のアニメーションによる発光例をインフォグラフィックスとして製作した。（図 4.9）グループ 2 では bect の使用素材である 3D プリンターで作ったギアや骨格、軟質ウレタンシート、使用機材である M5Stick C、サーボモータの配置図の説明。また、ラックアンドピニオンとステッピングモータの動作や仕組みの説明、GIF 素材のアニメーションによる bect と GUI の動作と連携をインフォグラフィクスとして製作した。（図 4.10）グループ 3 では POP UP SHELF の使用機材である、ステッピングモータとベルト駆動のレール、距離センサの配置図、また距離センサが手の距離を読み取り、ヘッド部が手の位置まで移動したのちに、ヘッド部が本を押し上げる手順を 3 つに分けて説明した。その手順を GIF 素材のアニメーションによる説明をインフォグラフィクスとして製作した。（図 4.11）

動作・仕組み

1. LEDテープが光る




2. 導光板が光を屈折させる



3. 導光板の上に置いた拡散板が光る

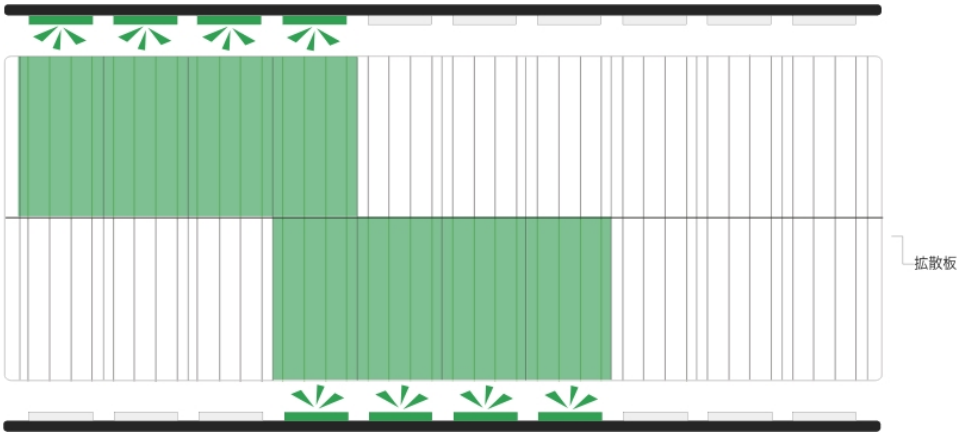



図 4.9　グループ 1：インフォグラフィックス

動作・仕組み

ラックアンドピニオン


回転部分

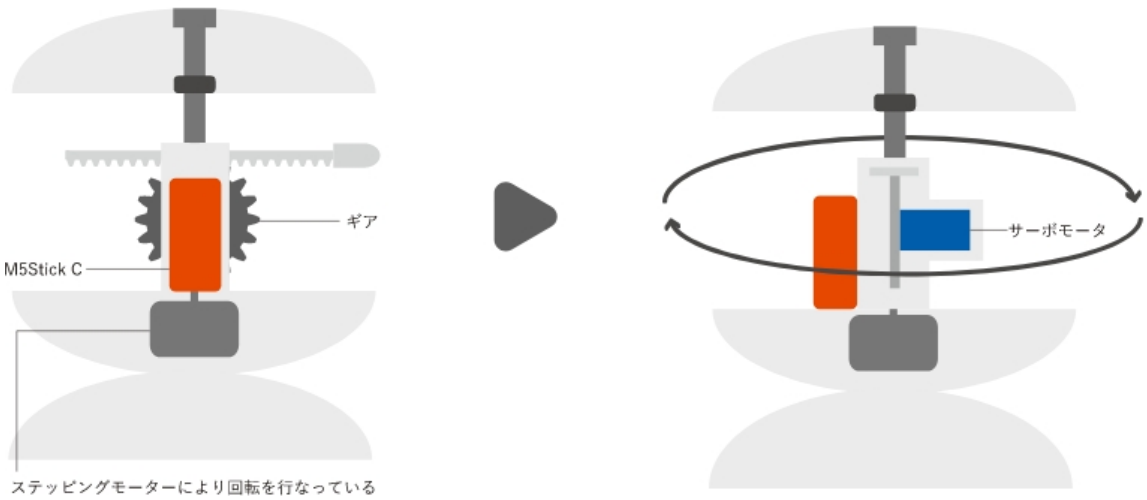



図 4.10　グループ 2：インフォグラフィックス

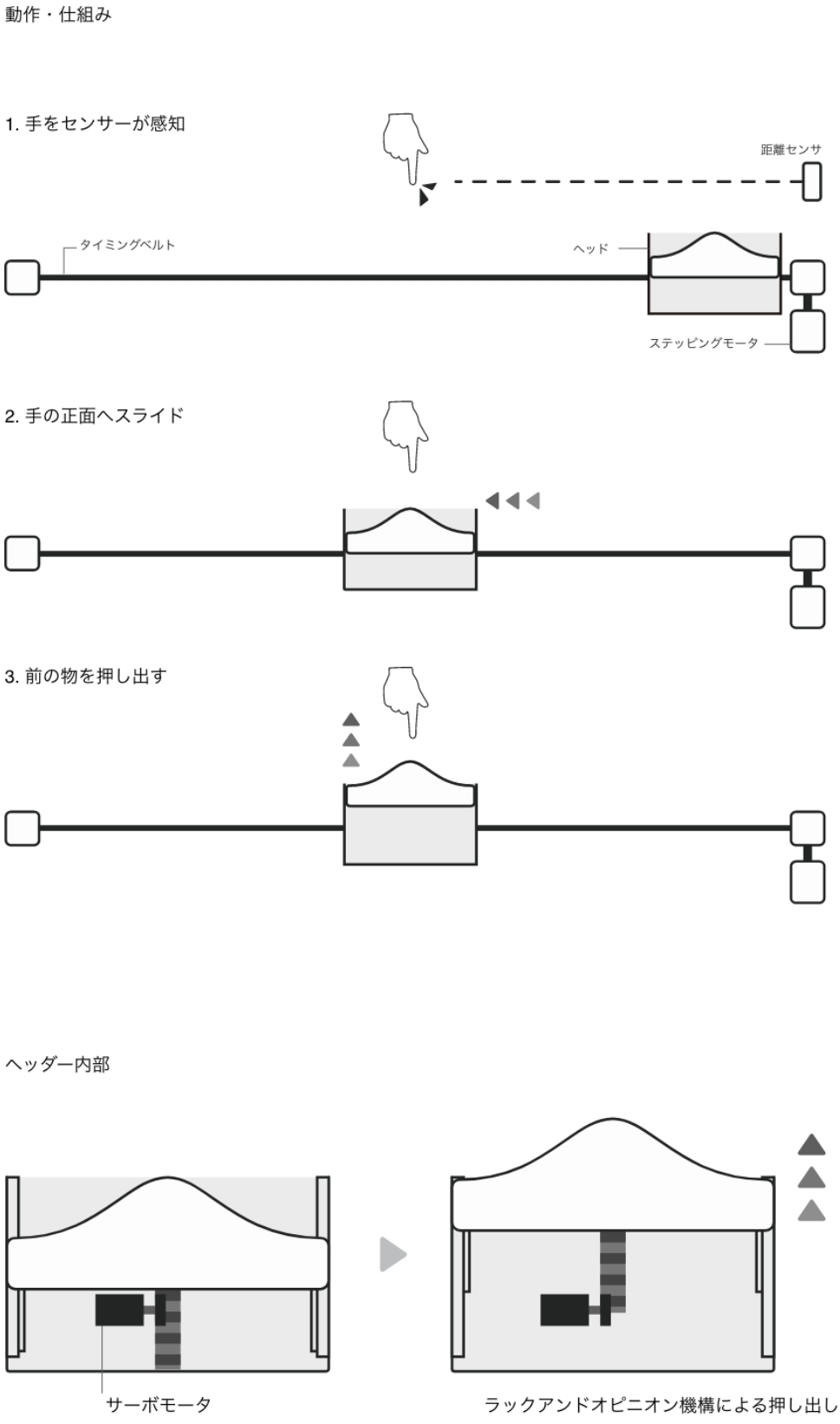



図 4.11　グループ 3：インフォグラフィックス

（※文責: 古町昂大）

**各 Element のロゴデザインについて**

名前とともに、よりプロダクトの認知性をあげるため、Element ごとにロゴを製作した。Element.01 のロゴ製作では、初めに誘導をモチーフにしたロゴ案をアートディレクターが数パターン提案したのちに、それに近しいロゴ案とリファレンスを探した。そこから床に置くタイプの物であることと、誘導する物であることなどの具体的なアプローチを決めた後、それに合わせて最終的なロゴをグループ 1 のメンバーが制作した。それを元にデザインを改良していった。（図 4.12）Element.02 のロゴ製作は、Element をより記号的に表した。bect の ”c” の部分に当てはまる図形は Element を上から見た形を表しており、方向を示す Element であったことから、左方向を指しているような形状とした。フォントとしてはインタラクションエレメンツのイメージに一致するよう、シンプルかつ未来感のあるフォントを用いている。（図 4.13）Element.03 のロゴ製作は、初期から製作していた、ヘッドの設計図データに入っている、パスデータをそのまま流用してデザイン案をまとめた。その後に、デザインの丸みやフォントの太さをプロジェクトとして統一感のあるように調整して完成した。（図 4.14）

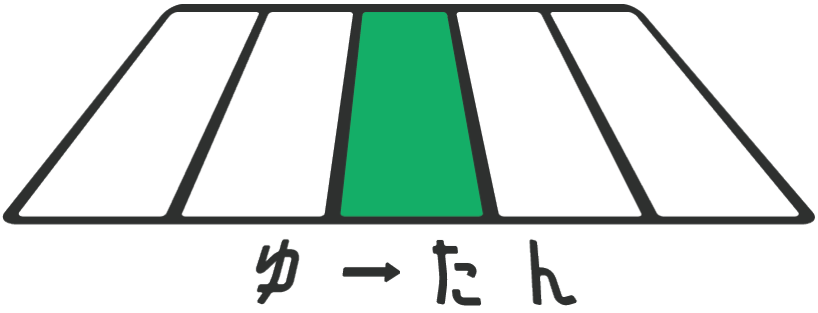



図 4.12　グループ 1：ロゴ


図 4.13　グループ 2：ロゴ


図 4.14　グループ 3：ロゴ

（※文責: 古町昂大）

#### コーディング工程

コーディングは、中間発表会での HTML と CSS のデータを引き継ぎ、Adobe XD での変更点を書き換える形となった。大きく変更した点として、ビジュアルと文章が横並びになっている部分のコードの書き方である。中間発表会まではひたすら下に文章と画像を添付するのみだった。しかし、今回は横並びの表現になるので、全てのコードを書き換えるのは効率が悪いため、「右側にビジュアル、左側に文章などの説明」が表示されるブロック要素、「左側にビジュアル、右側に文章などの説明」が表示されるブロック要素を製作した。その後に、そのブロック要素を交互に組み合わせることで効率よく Web サイトの改良を行った。またこの方法を用いて、Element.01 のみのページを製作したのちに、Element.02、Elemet.03 のページのビジュアルと文章の入れ替えだけで済むように改良を進めていった。また、中間発表会から、コーディング担当者が変更してしまったため、データの共有のための規則を決めておくべきだと感じた。例えば、画像データの命名パターンの指定や、CSS 内に都度コメントアウトや、HTML の配置順にコードを並べるようにするなど、データのやりとりを先に考えながら製作していくべきであった。

（※文責: 古町昂大）

#### サーバ

サーバの公開には、中間発表会時と同じく GitHub Pages を用いた。中間発表会時の反省を生かし、データの貼る方法、命名の基準を揃えた。また、画像データが中間発表会時よりも膨大に増えたため、フォルダを複数に分け、画像の管理をしやすくした。そうすることで、サーバに上げ直して更新する際に、必要な画像と必要でなくなった画像の仕分けをしやすくなった。

（※文責: 古町昂大）

### 4.4.2　ポスター製作

ポスターは Adobe Illustrator で製作した。プロジェクトのモチーフに合わせ白を基調としたデザインにした。初めに、ポスター全体の構成を考えた。中間発表会のデザインを参考にしたが、大きな違いは、文章量の違いである。中間発表会のポスターでは、Element の概要、中間発表会前時点での、機構の説明を短くまとめただけだった。しかし、成果発表会のポスターでは、概要、機構に加え、展望を付け加え、より Element に対する理解をしてもらえる文章になるように心がけた。文章は、Web サイトの文章を要約する形にし、Web サイトとの対応づけができるようにした。また、文章だけでは伝わりにくい機構の説明にはインフォグラフィックスを用いた。

（※文責: 丹野夏海）

### 4.4.3　動画製作

#### 撮影

動画を製作していくにあたり各 Element の Concept 動画を製作するため、撮影を行った。（図 4.15）撮影機材は Canon EOS kissX10 の一眼レフカメラ、Godox SL-60 の照明、UTEBIT の背景スタンドとホワイトバックを使用した。Concept 動画は、Element の詳細な動きを見せること

も目的の 1 つだが、それと同時に、プロジェクトの理念や顔となるような、抽象的な概要を伝えるよう意識してディレクションを行った。プロジェクトの白を基調としたモチーフに合わせ、3m × 3m のホワイトバック（白布）を背景スタンドにかけ、これをバックにプロダクトを撮影した。撮影時に使用した 3 つの照明は 2 つを背景にかけて影をなるべく消し、残りの 1 つをプロダクトに正面からかけて、影が正面に映らないように配置を試行錯誤した。また一眼レフカメラの設定として、全体的に白い構図なので、露出の値をやや高めに設定することで白の写りと映像の明るさを一定にした。

図 4.15　撮影の様子

（※文責: 古町昂大）

**編集**

上記の方法で撮影した動画素材をグループごとに整理し、それぞれのグループごとにコンセプト動画を製作した。編集ソフトはアニメーションを作成するため、全て Adobe After Effects を使用し製作した。まずはじめに、動画のオープニングを作成した。アートディレクターの要望に従い、プロジェクトのロゴがスライドアニメーションをしつつ、タイトル、サブタイトルが浮き出るようにした。このオープニングは他グループにも流用し、タイトル、サブタイトルを変えて 3 グループ分のオープニングを製作した。次に、最初に動画を撮り終えたグループ 3 の動画の編集にとりかかった。撮影したカットが違和感なく繋がるように動画の長さを調整して、別にプロジェクトメンバーが作成した音楽の長さに合わせて編集した。最後に動画の色味をトーンカーブと彩度を調整することでプロジェクト全体の雰囲気に合うようにした。この色味の調整はアートディレクターと画面共有をしながら逐一確認を取ることで、なるべく時間を効率的に使いクオリティを確保した。このようにグループ 3 の動画を製作し、これと同じように他グループの動画も編集した。グループ 2 はグループ 3 と同じようにカットを調整し、必要のないゴミなどの撮影時に取り除けなかったものをマスクで消し、最後に色味を変えて編集を終えた。グループ 1 に関しては、CG 合成を用いて動画の意図がより伝わるように工夫した。具体的には、エスカレーターの段差を見立てた板、エレベータのドアに見立てた板を After Effects の 3D 機能を用いて作成した。これをアニメーションさせることで、動画の動きに合わせて床やドアが動くことを想像できるようにした。合成させるうえで、マスクをかけて背景とうまく CG を合わせなければいけなかった。自動でマスクを生成する機能が After Effects にはあるが、今回は動画時間が短かったため、1 フレームごとに手作業でマ

スクを作成した。結果的に、違和感なく CG と実写映像を合成することができた。最後に他グループと同じように色味を調節し編集を終えた。これら編集を終えた動画をプロジェクトの Google ドライブに共有し、Web サイトの各グループページに掲載した。

（※文責: 佐々木皓大）

# 第 5 章　おわりに

## 5.1　グループのまとめ

### 5.1.1　Element.01 - ゆーたん

1 年間のプロジェクト期間を通して、私たちはグループ内で活発に議論を行うことができた。アイデア出しからかなりスムーズにグループでの活動を進めることができた。当初は、2 つの Element を製作することを予定していた。しかし、成果発表会までのスケジュールを考えた結果、iSi の製作は、3 人という人数と限られた時間の都合上、製作を諦める形となり、ゆーたんのみの製作となった。ゴールを見据えグループでの製作物を 1 つに絞った結果、全員が 1 つの目標に向かって製作を進めていくことができた。また、ゆーたんを余裕をもって製作することができたため、製作全体を通してプラスになったと考えられる。これにより、ゆーたんはアイデア出しの段階よりもより良いものになったと考えられる。例えば、光り方のパターンでは多くのパターンを出し、実際にプログラムすることができた。構造面では、見かけを LED テープむき出しの状態から、そこにカバーをつけたことで見た目をすっきりさせることができた。私たちのグループはオンラインという状況のなかでも積極的にコミュニケーションをとったことで、この Element を製作することができたと考える。

（※文責: 中村ももこ）

### 5.1.2　Element.02 - bect

プロジェクト学習全体を通して、私たちのグループは多くの方向転換を行った。前期からの計画変更が一番大きな転機であったが、それ以外にも細かなところで当初の予定と違う結果になったことが多い。前期からの夏季休業を経て、私たちの進捗に遅れが生じており、成果発表会に間に合うか不安であった。しかし、結果的に 1 つのデバイスとして形にできたことは評価できる。グループメンバーが一丸となって製作した結果、反省点はあるもののデバイスを完成させることができた。その中で、自分たちが目標とした動作の一端は実現できたと考えている。また、それぞれの得意分野に対する理解度はとても高く、分担後の製作はとても早かった。このようなチーム開発の良い経験をできたということも、プロジェクト学習においてよい経験であった。メンバーそれぞれの今後の活動に活かされることを願う。

（※文責: 佐々木皓大）

### 5.1.3　Element.03 - POP UP SHELF

私たちのグループでは、POP UP SHELF の製作活動が開始した当初から自由で活発な議論を重ねることができ、アイデア出しの段階から多くの意見を集めることができた。Miro によるグループ分けでは、デバイスなどの操作ツールや、人の行為を誘発させるものを製作したいという意志を持ったメンバーが中心となって集まっており、何か新しい体験ができるものを製作したいという

目標のもと活動を開始した。今回製作した「POP UP SHELF」は、前述した目標に沿うものであり、メンバー全員が意欲的に製作を行うことができた。製作過程では、予想外の失敗や思うように進まない場面も多くあったが、その度にグループ内で話し会いをすることで代案を考え出し、製作を進めることができた。成果発表会では、中間発表会時点で予定していた、本が波打つ表現や検索機能などは実現に至らなかったが、本を指差す手に反応し、本が飛び出す滑らかな動きを表現させることができた。役割分担の作業でも、メンバー同士で協力し、話し合いながら進めたことで、メンバー全員が納得のいく Element を製作することができた。今回のプロジェクト学習で学んだことを糧に、今後の活動においても行動していきたい。

（※文責: 丹野夏海）

## 5.2　プロジェクトのまとめ

本プロジェクトでは、全体でアイデアの創出を行い、それ以降は 1 グループあたり 3〜4 人で構成される 3 つのグループに分かれて作業を行った。それぞれのグループで、床に置くことで人とエスカレーターや自動ドアなどの機械との接点となる「ゆーたん」、手に持つことで行きたい方向を視覚と触覚で認識させる「bect」、本の探索に新たなフィードバックを与える「POP UP SHELF」を最終成果物として製作した。多種多様な役職を配置し、メンバー同士の意見交換を大事にプロジェクト全体で一体感を持って活動を行った。オンラインツールを駆使することで、作業の効率化とコロナ禍への対応を行った。達成度はグループによって違うが、本プロジェクトの目的である、今までにはなかった、未来を形作る Interaction Elements のコンセプトを提示することができたと考える。

（※文責: 赤塚一輝）

# 付録 A　成果発表会で使用したポスター

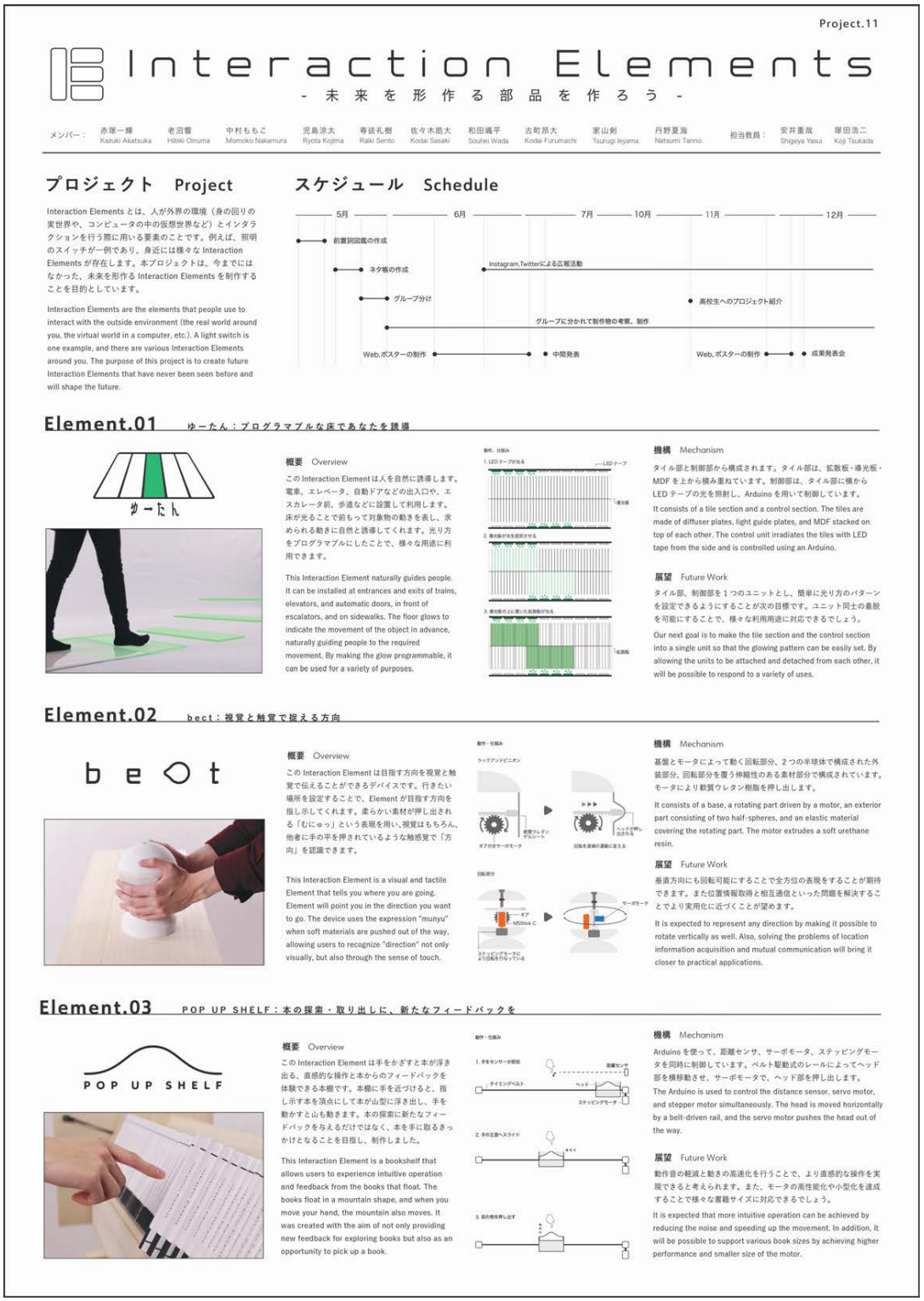



図 A.1　成果発表会で使用したポスター

# 付録 B　成果発表会に向けて製作した Web サイト

Web サイトの URL
https://interactionelements2021.github.io/IE2021-official/

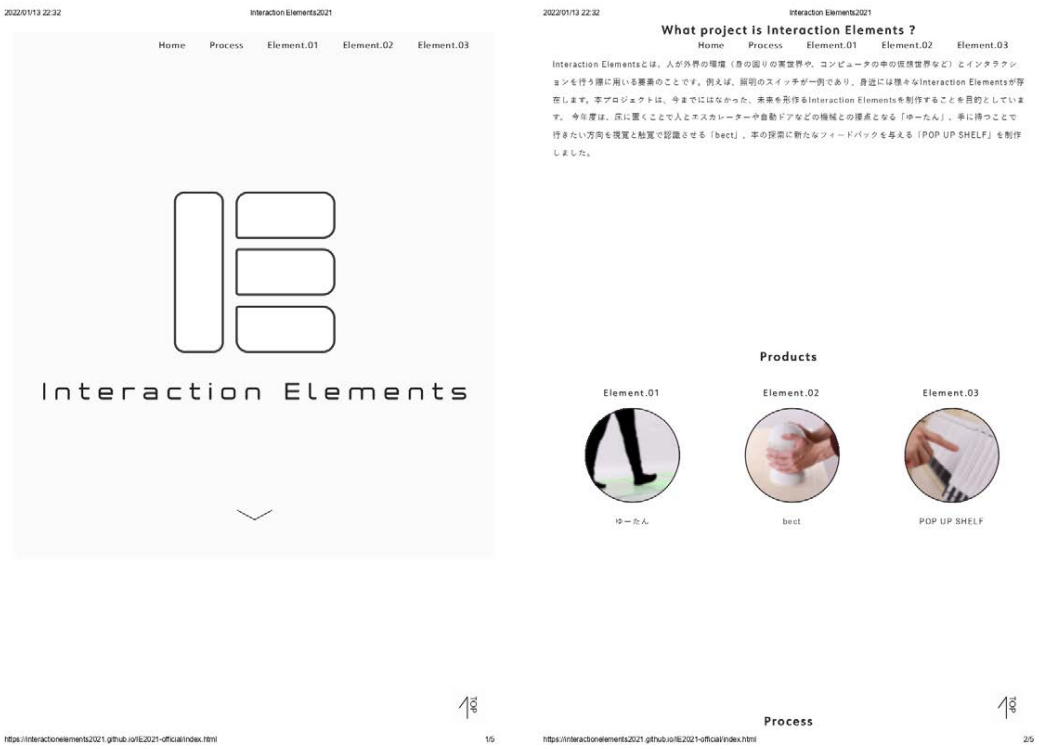



図 B.1　成果発表会に向けて製作した Web サイト（Home-1）

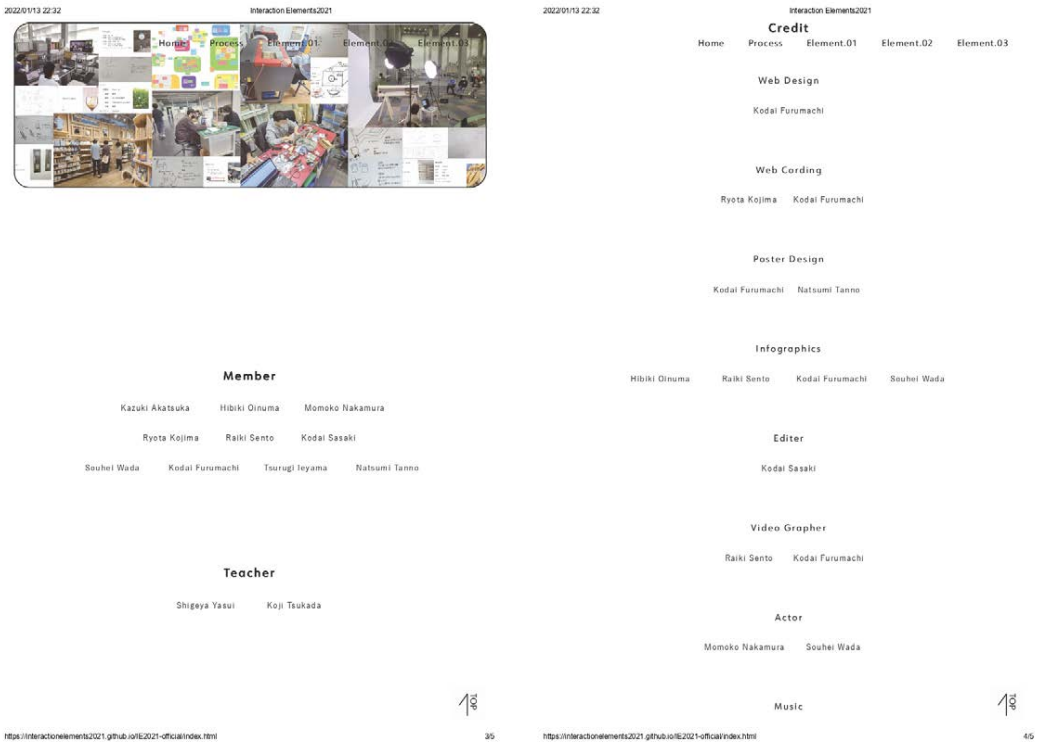



図 B.2　成果発表会に向けて製作した Web サイト：Home-2


図 B.3　成果発表会に向けて製作した Web サイト（Process-1）

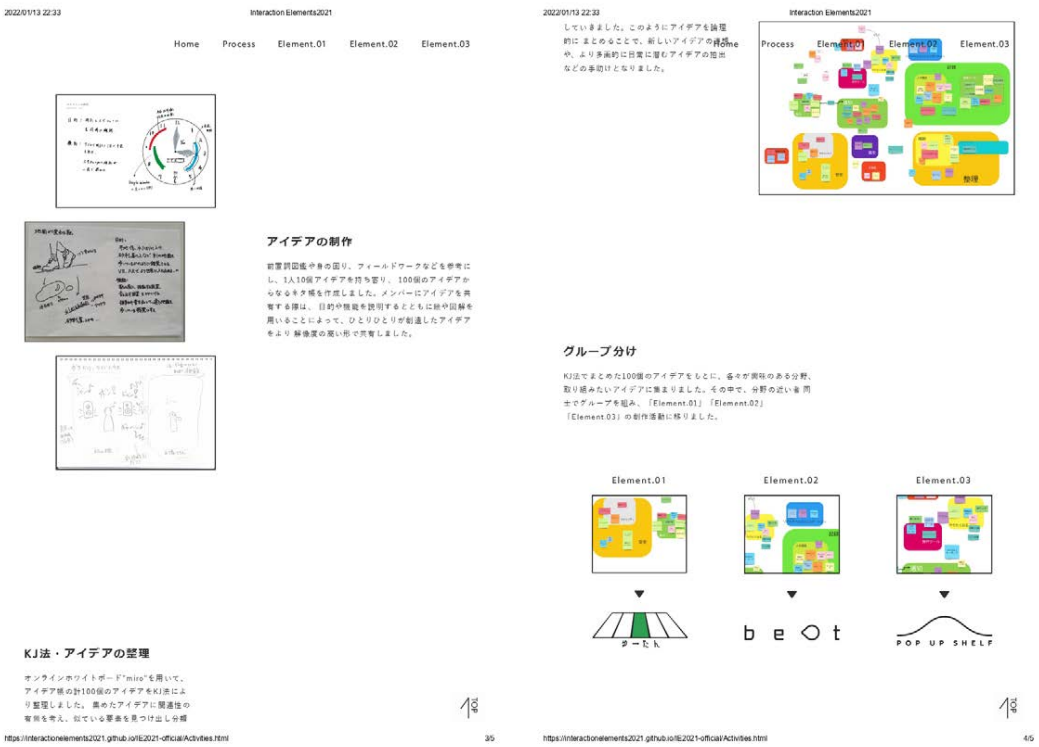



図 B.4　成果発表会に向けて製作した Web サイト（Process-2）


図 B.5　成果発表会に向けて製作した Web サイト (Element.01-1)

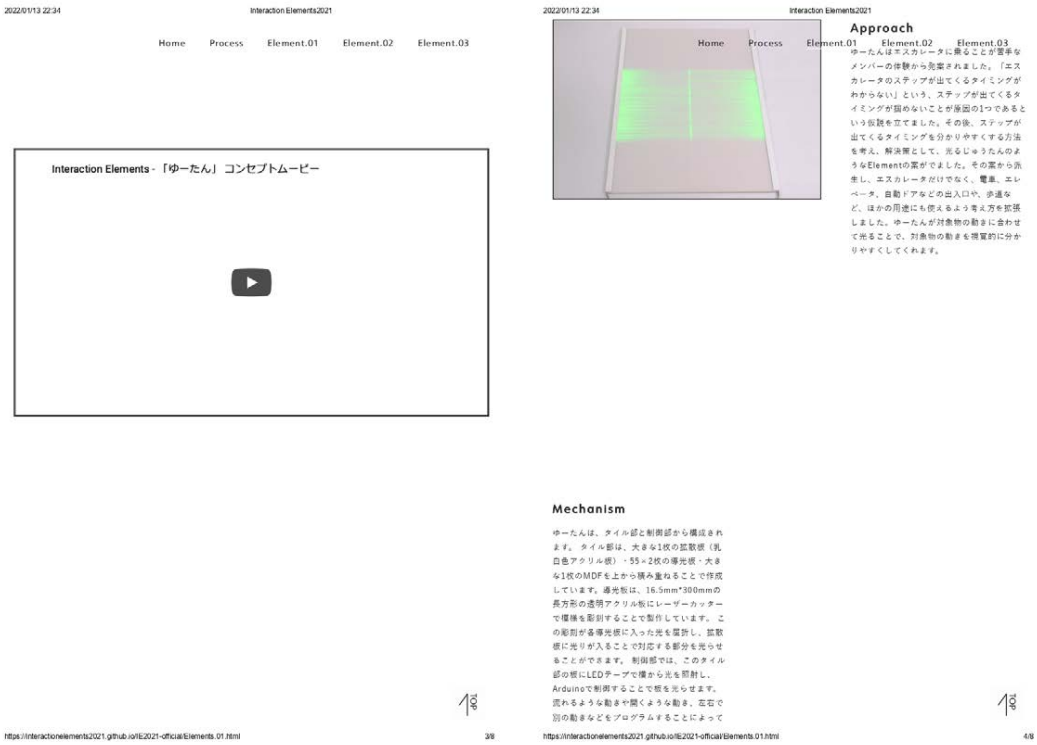



図 B.6　成果発表会に向けて製作した Web サイト (Element.01-2)

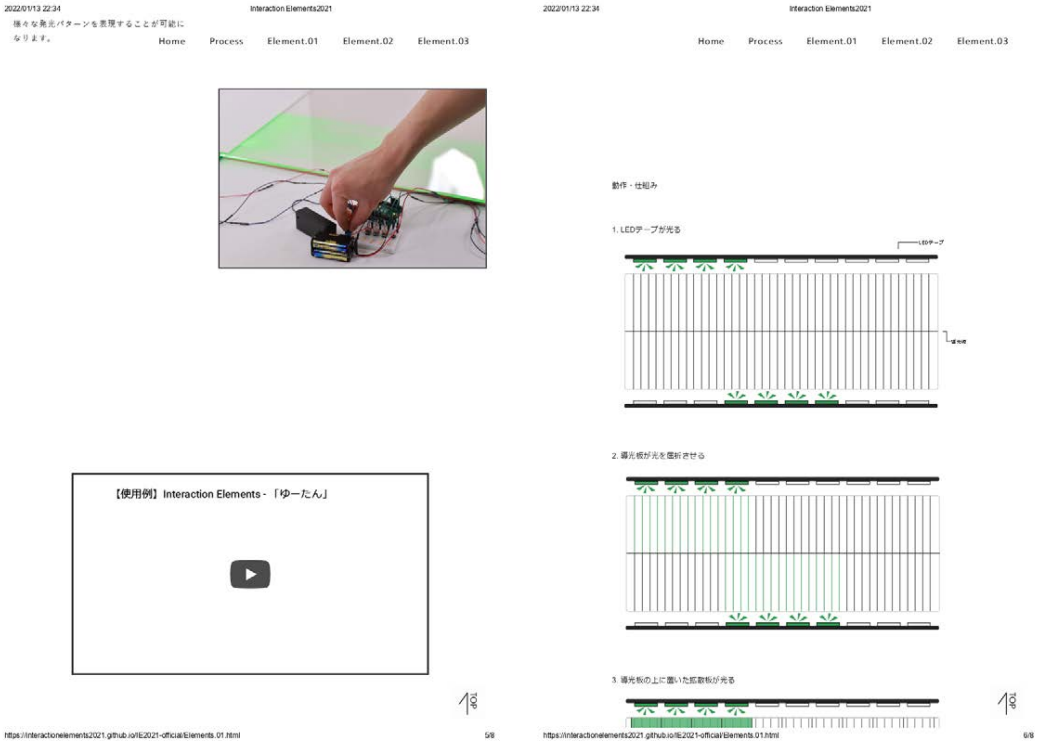



図 B.7　成果発表会に向けて製作した Web サイト (Element.01-3)

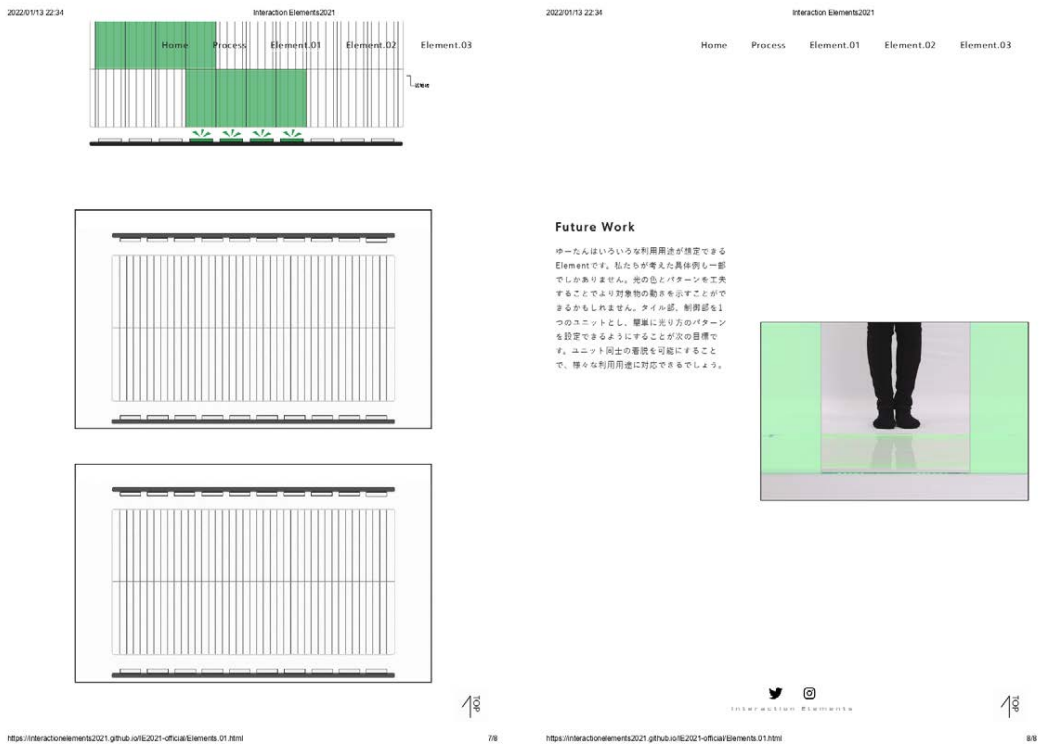



図 B.8　成果発表会に向けて製作した Web サイト (Element.01-4)

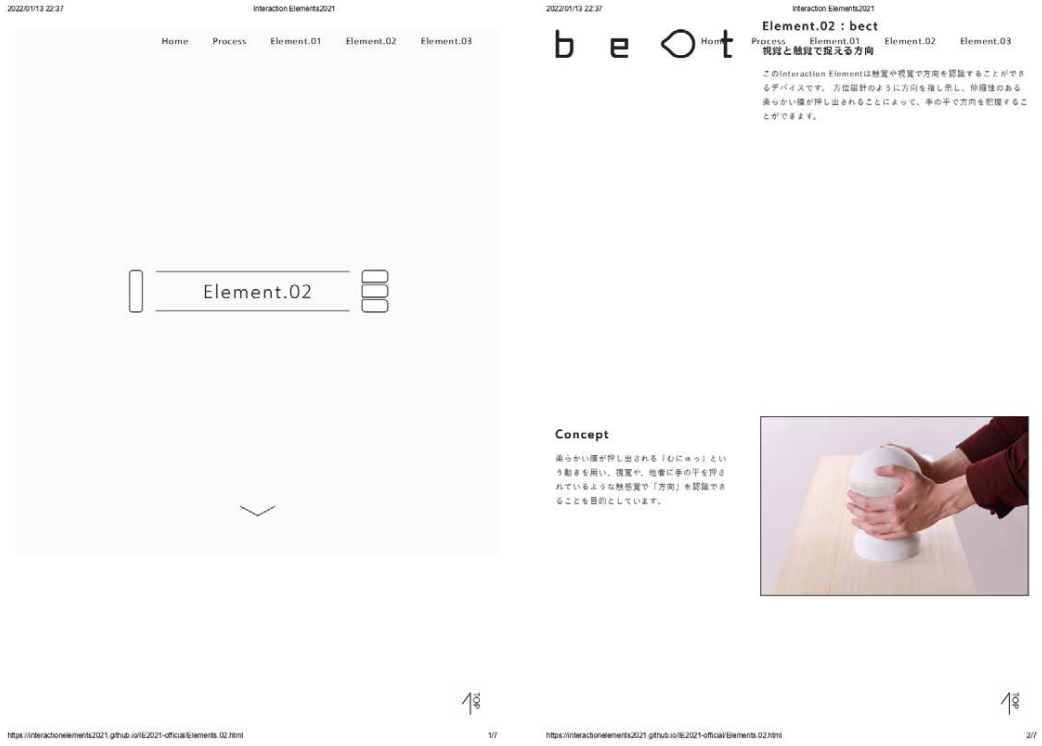



図 B.9　成果発表会に向けて製作した Web サイト (Element.02-1)

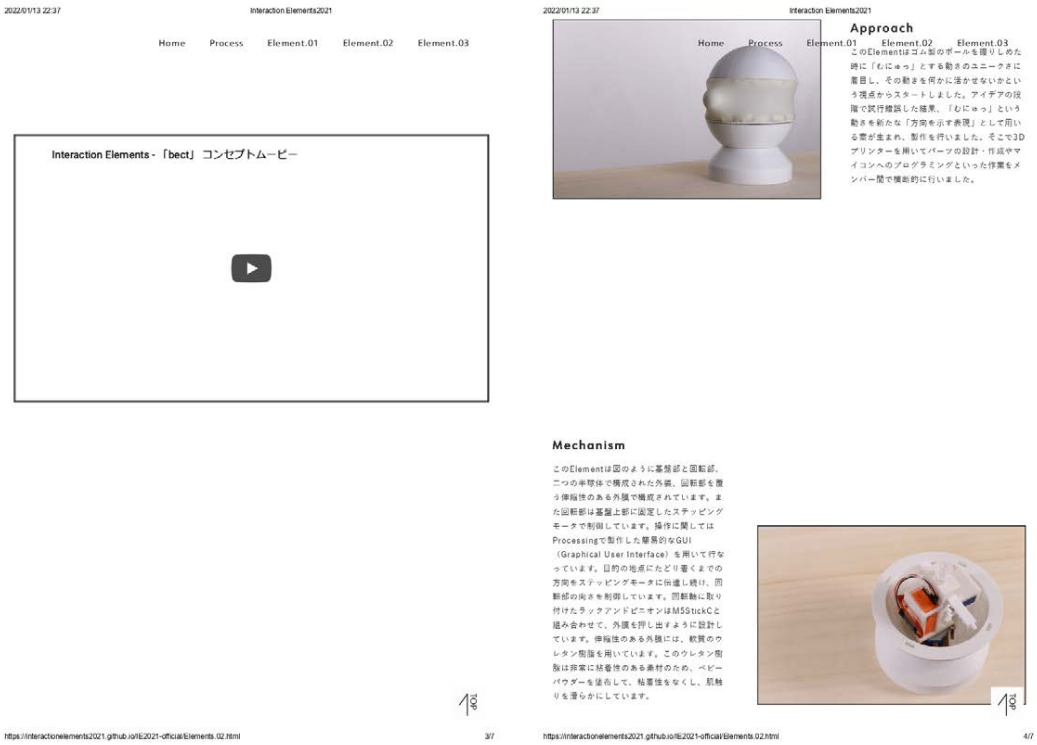



図 B.10　成果発表会に向けて製作した Web サイト (Element.02-2)

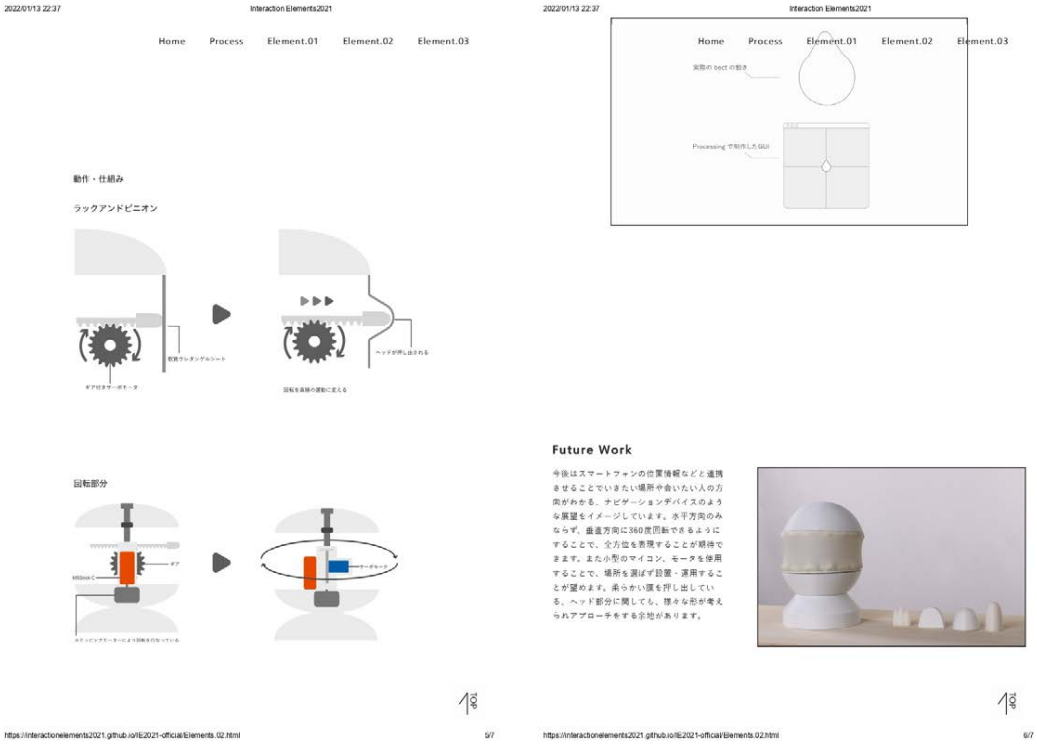



図 B.11　成果発表会に向けて製作した Web サイト (Element.02-3)

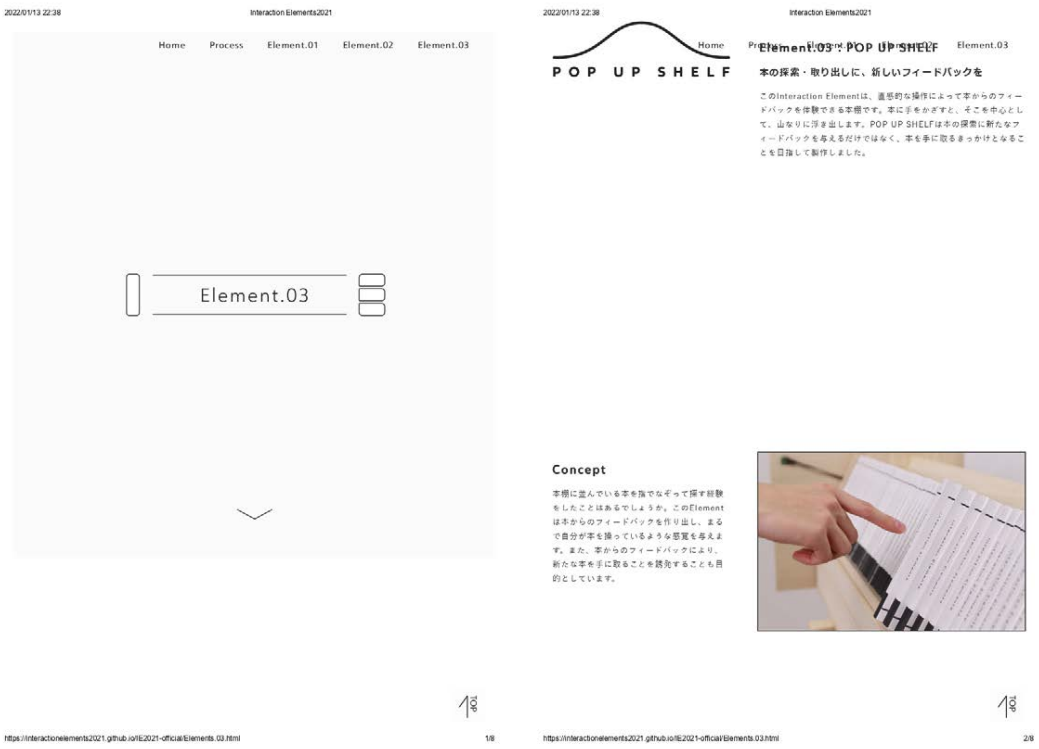



図 B.12　成果発表会に向けて製作した Web サイト (Element.02-1)

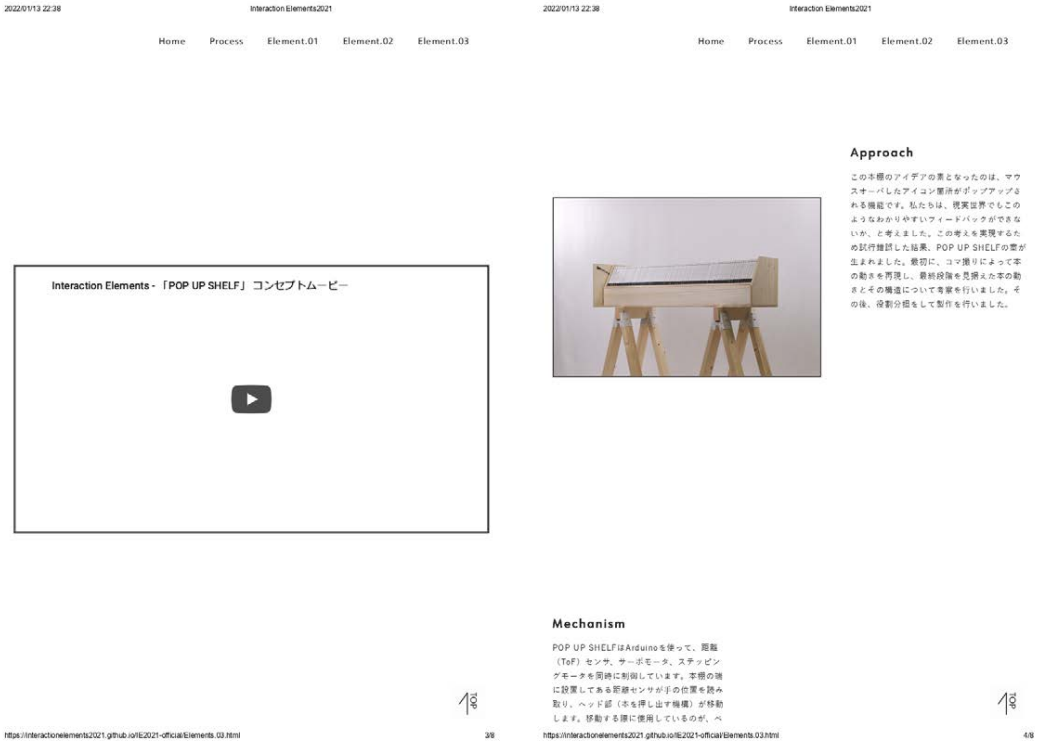



図 B.13　成果発表会に向けて製作した Web サイト (Element.03-1)

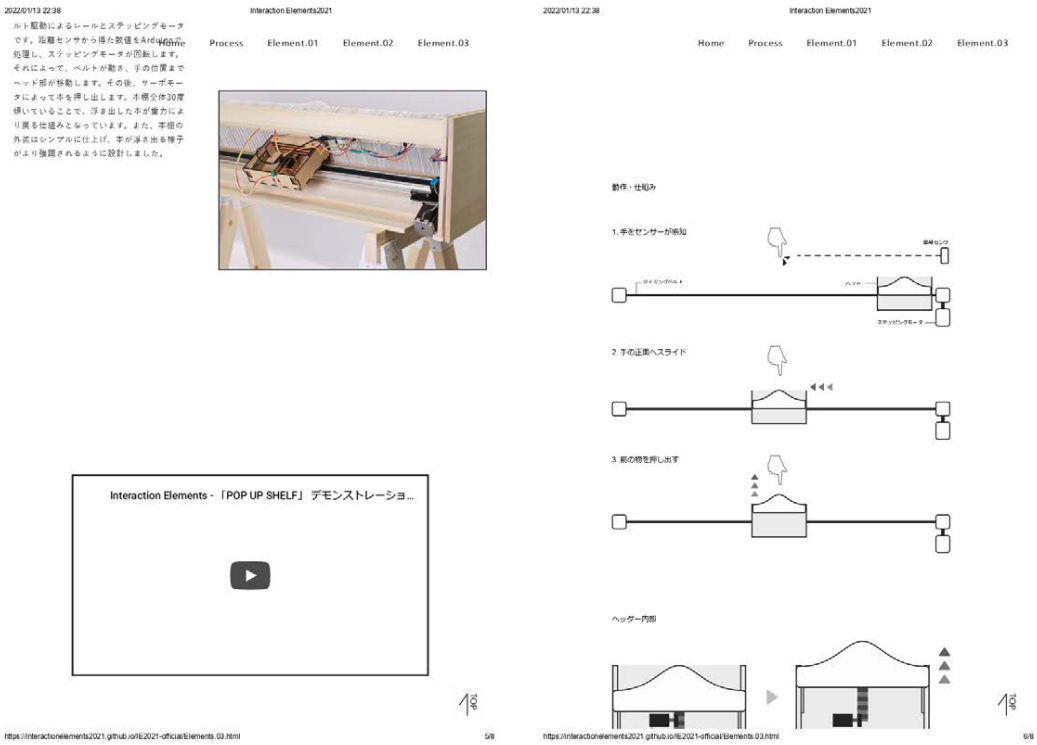



図 B.14　成果発表会に向けて製作した Web サイト (Element.03-2)

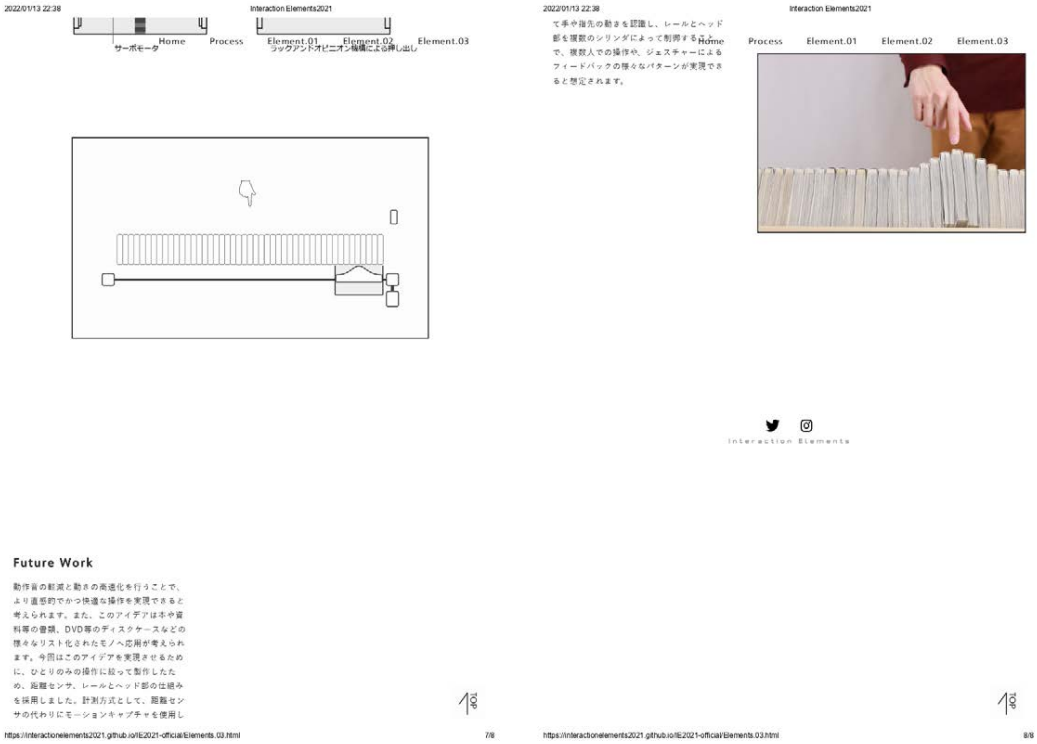



図 B.15　成果発表会に向けて製作した Web サイト (Element.03-3)

# 付録 C　ソースコード

## C.1　Element.01 - ゆーたん

**main**

Listing C.1　main.ino

```
1
2  #include <Adafruit_NeoPixel.h>
3  #define LED_COUNT ( 55 )
4
5  #define LED_PIN0 ( 2 )
6  #define LED_PIN1 ( 7 )
7
8  #define var_PIN0 ( 3 )
9
10 #define DIG_PIN0 ( 4 )
11 #define DIG_PIN1 ( 5 )
12 #define DIG_PIN2 ( 6 )
13
14 #define INPUT_PIN ( A0 )
15
16 Adafruit_NeoPixel led0( LED_COUNT, LED_PIN0 , NEO_GRB + NEO_KHZ800);
17 Adafruit_NeoPixel led1( LED_COUNT, LED_PIN1 , NEO_GRB + NEO_KHZ800);
18
19 int state0 = 0;
20 int state1 = 0;
21 int state2 = 0;
22 int S = 0;
23 int var;
24 int volume;
25
26 void setup() {
27 #if defined(__AVR_ATtiny85__) && (F_CPU == 16000000)
28   clock_prescale_set(clock_div_1);
29 #endif
30
31   pinMode( var_PIN0, INPUT_PULLUP );
32
33   pinMode( DIG_PIN0, INPUT_PULLUP );
34   pinMode( DIG_PIN1, INPUT_PULLUP );
35   pinMode( DIG_PIN2, INPUT_PULLUP );
36
37   pinMode( INPUT_PIN, INPUT );
38
```

```
39   led0.begin();
40   led1.begin();
41 }
42
43 void loop() {
44   volume = analogRead(INPUT_PIN);
45   state0 = digitalRead(DIG_PIN0);
46   state1 = digitalRead(DIG_PIN1);
47   state2 = digitalRead(DIG_PIN2);
48
49   S = state0 + state1 * 2 + state2 * 4;
50
51   switch (S) {
52     case 0:
53       ESC_fp1();
54       break;
55
56     case 1:
57       ESC_3fi();
58       break;
59
60     case 2:
61       ESC_fr3();
62       break;
63
64     case 3:
65       break;
66
67     case 4:
68       break;
69
70     case 5:
71       break;
72
73     case 6:
74       break;
75
76     case 7:
77       EV_oc();
78       break;
79   }
80 }
```

（※文責: 老沼響）

**a(ESC_fp1)**

Listing C.2 a.ino

```
1
2  void ESC_fp1() {
3    int Ar_led[2][5];
4
5    for (int k = 0; k < 7; k++) {
6
7      for (int n = 0; n < 2; n++) {
8        for (int m = 0; m < 5; m++) {
9          Ar_led[n][m] = 0;
10       }
11     }
12
13     for (int R = 0; R < LED_COUNT; R++) {
14       led0.setPixelColor(R, led0.Color(0, 0, 0));
15       led1.setPixelColor(R, led1.Color(0, 0, 0));
16     }
17
18     switch (k) {
19       case 0:
20
21         break;
22
23       case 1:
24         Ar_led[1][0] = 1;
25         break;
26
27       case 2:
28         Ar_led[0][1] = 1;
29         Ar_led[1][0] = 1;
30         break;
31
32       case 3:
33         Ar_led[0][1] = 1;
34         Ar_led[1][2] = 1;
35         break;
36
37       case 4:
38         Ar_led[0][3] = 1;
39         Ar_led[1][2] = 1;
40         break;
41
42       case 5:
43         Ar_led[0][3] = 1;
44         Ar_led[1][4] = 1;
45         break;
46
47       case 6:
```

```
48          Ar_led[1][4] = 1;
49          break;
50      }
51
52      for (int i = 0; i < 5; i++) {
53
54        if (Ar_led[0][i] == 1) {
55          for ( int j = 0 + i * 11; j < 11 + i * 11; j++) {
56            led0.setPixelColor(j, led0.Color(0, 255, 0));
57          }
58        }
59        if (Ar_led[1][i] == 1) {
60          for ( int j = 0 + i * 11; j < 11 + i * 11; j++) {
61            led1.setPixelColor(j, led1.Color(0, 255, 0));
62          }
63        }
64
65      }
66
67      led0.show();
68      led1.show();
69      volume = analogRead(INPUT_PIN);
70      delay(volume * 5);
71
72    }
73  }
```

（※文責: 老沼響）

### b(ESC_3fi)

Listing C.3　b.ino

```
1
2  void ESC_3fi() {
3    for (int k = 0; k < 3; k++) {
4
5      for (int R = 0; R < LED_COUNT; R++) {
6        led0.setPixelColor(R, led0.Color(0, 0, 0));
7        led1.setPixelColor(R, led1.Color(0, 0, 0));
8      }
9
10     switch (k) {
11       case 0:
12         for (int L = 0; L < 25; L++) {
13           led0.setPixelColor(L, led0.Color(0, 255, 0));
14           led1.setPixelColor(L, led1.Color(0, 255, 0));
15         }
16         break;
```

```
17
18        case 1:
19          for (int L = 16; L < 41; L++) {
20            led0.setPixelColor(L, led0.Color(0, 255, 0));
21            led1.setPixelColor(L, led1.Color(0, 255, 0));
22          }
23          break;
24
25        case 2:
26          for (int L = 30; L < 55; L++) {
27            led0.setPixelColor(L, led0.Color(0, 255, 0));
28            led1.setPixelColor(L, led1.Color(0, 255, 0));
29          }
30          break;
31      }//終了 switch
32
33      led0.show();
34      led1.show();
35      volume = analogRead(INPUT_PIN);
36      delay(volume);
37
38    }
39  }
```

（※文責: 老沼響）

### c(ESC_fr3)

Listing C.4　c.ino

```
1
2  void ESC_fr3() {
3    for (int k = 0; k < 79; k++) {
4
5      for (int R = 0; R < 79; R++) {
6        led0.setPixelColor(R, led0.Color(0, 0, 0));
7        led1.setPixelColor(R, led1.Color(0, 0, 0));
8      }
9
10     if (k < 24) {
11       for (int L = 0; L <= k; L++) {
12         led0.setPixelColor(L, led0.Color(0, 255, 0));
13         led1.setPixelColor(L, led1.Color(0, 255, 0));
14       }
15     } else if (23 < k) {
16       for (int L = k; L < k + 24; L++) {
17         led0.setPixelColor(L - 24, led0.Color(0, 255, 0));
18         led1.setPixelColor(L - 24, led1.Color(0, 255, 0));
19       }
```

```
20      }
21
22      led0.show();
23      led1.show();
24      volume = analogRead(INPUT_PIN);
25      delay(volume);
26
27    }
28  }
```

（※文責: 老沼響）

### h(EV_oc)

Listing C.5　h.ino

```
1
2  void EV_oc() {
3    var = digitalRead(var_PIN0);
4
5    //open
6    while (var == LOW) {
7      delay(10);
8      var = digitalRead(var_PIN0);
9    }
10   for (int i = 0; i <= 27; i++) {
11     led0.setPixelColor(27 - i, led0.Color(0, 255, 0));
12     led0.setPixelColor(27 + i, led0.Color(0, 255, 0));
13     led1.setPixelColor(27 - i, led0.Color(0, 255, 0));
14     led1.setPixelColor(27 + i, led0.Color(0, 255, 0));
15
16     led0.show();
17     led1.show();
18     volume = analogRead(INPUT_PIN);
19     delay(volume);
20   }
21
22   var = digitalRead(var_PIN0);
23   //wait
24   while (var == LOW) {
25
26     for (int R = 0; R < LED_COUNT; R++) {
27       led0.setPixelColor(R, led0.Color(0, 255, 0));
28       led1.setPixelColor(R, led1.Color(0, 255, 0));
29     }
30     led0.show();
31     led1.show();
32     delay(500);
33
```

```
34      for (int R = 0; R < LED_COUNT; R++) {
35        led0.setPixelColor(R, led0.Color(0, 0, 0));
36        led1.setPixelColor(R, led1.Color(0, 0, 0));
37      }
38      led0.show();
39      led1.show();
40      delay(500);
41      var = digitalRead(var_PIN0);
42
43    }
44
45    for (int R = 0; R < LED_COUNT; R++) {
46      led0.setPixelColor(R, led0.Color(255, 0, 0));
47      led1.setPixelColor(R, led1.Color(255, 0, 0));
48    }
49    led0.show();
50    led1.show();
51    delay(1000);
52
53    //close
54    for (int i = 0; i <= 27; i++) {
55      led0.setPixelColor(0 + i, led0.Color(0, 0, 0));
56      led0.setPixelColor(54 - i, led0.Color(0, 0, 0));
57      led1.setPixelColor(0 + i, led0.Color(0, 0, 0));
58      led1.setPixelColor(54 - i, led0.Color(0, 0, 0));
59
60      led0.show();
61      led1.show();
62      volume = analogRead(INPUT_PIN);
63      delay(volume);
64    }
65
66  }
67
68  }
```

（※文責: 老沼響）

## C.2 Element.02 - bect

Listing C.6 serial_stick.ino

```
1
2  #include <M5StickC.h>
3
4  // パラメーター
5  int pmin = ((0.5 / 20) * 1024) + 0.5;
6  int pmax = (((2.4 / 20) * 1024) + 0.5) / 2;
```

```
7  int pd = 5;
8  int p = pmin;
9
10
11 void setup() {
12   M5.begin();
13   //Serial.begin(115200);
14   Serial.begin(9600);
15   M5.Lcd.setRotation(1);
16   delay(100);
17
18
19   M5.Lcd.printf("pmin = %d\n", pmin);
20   M5.Lcd.printf("pmax = %d\n", pmax);
21
22   ledcSetup(0, 50, 10);
23
24   ledcAttachPin(26, 0);
25   ledcWrite(0, pmin);
26   //delay(100);
27 }
28 void loop() {
29   char val;
30   Serial.println("send start!");
31   if (Serial.available() > 0) {
32     val = Serial.read();
33     if (val == 'a') {
34       while (1) {
35         val = Serial.read();
36         M5.Lcd.print("yes");
37         while (p < pmax) {
38           p += pd;
39           ledcWrite(0, p);
40           M5.Lcd.printf("p = %d\n", p);
41           delay(50);
42         }
43         while (p > pmin) {
44           p -= pd;
45           ledcWrite(0, p);
46           M5.Lcd.printf("p = %d\n", p);
47           delay(50);
48         }
49         if (val == 'b') {
50           exit(0);
51         }
52         delay(1000);
53       }
54     }
```

```
  }
}
```

Listing C.7 verupdate.ino

```
#include <Stepper.h>

const int MOTOR_STEPS = 2048;
int x = 0;
int y = 0;
int now_y = 0;
char val;

Stepper myStepper(MOTOR_STEPS, 8, 10, 9, 11);

void setup() {
  Serial.begin(9600);
  myStepper.setSpeed(10);
}

void loop() {
  if (Serial.available() >= 4) {
    if ( Serial.read() == 'H' ) {
      byte high = Serial.read();
      byte low = Serial.read();
      val = Serial.read();
      y = high * 256 + low;
      x = now_y;
    }
  }
  if (val == 'a')
    myStepper.step(-5.69);
  if (val == 'b')
    myStepper.step(5.69);
  if (val == 'c') {
    if (now_y == y) {
      myStepper.step(0);
    } else if (now_y > y) {
      if (x != y) {
        myStepper.step(-5.69);
        x--;
      } else {
        now_y = y;
      }
    } else if (now_y < y) {
      if (x != y) {
        myStepper.step(5.69);
        x++;
```

```
45        } else {
46          now_y = y;
47        }
48      }
49    }
50  }
```

Listing C.8 verProcessing.ino

```
1
2  import processing.serial.*;
3  float x;
4  float y;
5  double z;
6  String c;
7  Serial port;
8  float rad;
9  int gR;
10
11
12 void setup() {
13   size(300, 400);
14   port = new Serial(this, "COM3", 9600);
15 }
16 void draw() {
17   background(255);
18   stroke(0);
19   line(0, 150, 300, 150);
20   line(150, 0, 150, 300);
21   fill(200);
22   rect(70, 320, 50, 50);
23   rect(170, 320, 50, 50);
24   rect(240, 320, 50, 50);
25   fill(0);
26   text("left", 90, 350);
27   text("right", 185, 350);
28   text("stop", 255, 350);
29   stroke(#FA0D28);
30   line(150, 150, 150+150*cos(rad), 150-150*sin(rad));
31 }
32
33 void mouseClicked() {
34   x=mouseX;
35   y=mouseY;
36
37   if (mouseButton == LEFT) {
38
39     gR=((int)getRadian(150, 150, x, y));
40     if (gR<=180) {
```

```
41        sendIntData(180-gR);
42        println(180-gR);
43      } else {
44        sendIntData(180+360-gR);
45        println(180+360-gR);
46      }
47
48      if ((70 <= x)&&(x<= 120) && (320 <= y)&&(y <= 370)) {
49        fill(255);
50        rect(70, 320, 50, 50);
51      }
52      if ((170 <= x)&&(x<= 220) && (320 <= y)&&(y <= 370)) {
53        fill(255);
54        rect(170, 320, 50, 50);
55      }
56      if ((240 <= x)&&(x<= 290) && (320 <= y)&&(y <= 370)) {
57        fill(255);
58        rect(170, 320, 50, 50);
59      }
60    }
61  }
62
63  char isClicked(float x, float y) {
64    if ((0 <= x)&&(x<= 300) && (0 <= y)&&(y <= 300)) {
65      //sendCharData('c');
66      return 'c';
67    }
68    if ((70 <= x)&&(x<= 120) && (320 <= y)&&(y <= 370)) {
69      // sendCharData('a');
70      return 'a';
71    }
72    if ((170 <= x)&&(x<= 220) && (320 <= y)&&(y <= 370)) {
73      //sendCharData('b');
74      return 'b';
75    }
76    if ((240 <= x)&&(x<= 290) && (320 <= y)&&(y <= 370)) {
77      //sendCharData('b');
78      return 'd';
79    }
80    return 'e';
81  }
82
83
84
85  protected double getRadian(double x, double y, double x2, double y2) {
86    double radian = Math.atan2(y - y2, -(x - x2));
87    rad=(float)radian;
88    double degree = radian * 180d / Math.PI;
```

```
  if (degree>=0) {
    degree=180-degree;
  } else {
    degree=180+degree*(-1);
  }
  return degree;
}


void sendIntData(int value) {
  byte high = (byte)((value & 0xFF00) >> 8);
  byte low = (byte)( value & 0x00FF);
  port.write('H');
  port.write(high);
  port.write(low);
  port.write(isClicked(x, y));
  println(high+","+low+","+isClicked(x, y));
}
```

## C.3 Element.03 - POP UP SHELF

**TestToF.ino**

Listing C.9 TestToF.ino

```

// VL53L1X
#include <Wire.h>
#include <VL53L1X.h>


VL53L1X sensor;


void setup()
{
  Serial.begin(9600);
  Wire.begin();
  Wire.setClock(400000);

  sensor.setTimeout(10000);

  if (!sensor.init())
  {
    Serial.println("Failed");
    while (1);
  }

  sensor.setDistanceMode(VL53L1X::Long);
```

```
25   sensor.setMeasurementTimingBudget(50000);
26   sensor.startContinuous(50);
27 }
28
29
30 void loop()
31 {
32   Serial.print(sensor.read());
33
34   if (sensor.timeoutOccurred())
35   {
36     Serial.print("TIMEOUT");
37   }
38
39   Serial.println();
40 }
```

（※文責: 家山剣）

### TestToFandServo.ino

Listing C.10　TestToFandServo.ino

```
1
2 #include <Servo.h>
3 #include <Wire.h>
4 #include <VL53L1X.h>
5
6
7 const int servoPin = 8;
8
9
10 Servo myservo;
11 VL53L1X sensor;
12
13
14 void setup() {
15   // put your setup code here, to run once:
16   Serial.begin(9600);
17
18   Wire.begin();
19   Wire.setClock(400000);
20
21   sensor.setTimeout(10000);
22   if (!sensor.init())
23   {
24     Serial.println("Failed");
25     while (1);
26   }
```

```
27   sensor.setDistanceMode(VL53L1X::Long);
28   sensor.setMeasurementTimingBudget(50000);
29   sensor.startContinuous(50);
30
31   myservo.attach(servoPin);
32   myservo.write(0);
33 }
34
35 void loop()
36 {
37   Serial.print(sensor.read());
38
39   if (sensor.read() > 1200) {
40     myservo.write(0);
41     Serial.print("+");
42   }
43   else {
44     myservo.write(180);
45     Serial.print("-");
46   }
47
48   Serial.println();
49 }
```

（※文責: 古町昂大）

### TestDRV8835.ino

Listing C.11　TestDRV8835.ino

```
1
2 // confirmed 2021/11/12
3 // push sw -> digital 2 (control direction)
4 // VR -> analog 0 (control speed)
5
6
7 int APHASE = 4;
8 int AENBL = 5;
9 int BPHASE = 6;
10 int BENBL = 7;
11 int VR_PIN = A0;
12 int SW_PIN = 2;
13
14
15 unsigned long VR_VALUE = 0;
16 int sw_value = 0;
17
18
19 void setup()
```

```
{
  Serial.begin(9600);

  pinMode(APHASE, OUTPUT);
  pinMode(AENBL, OUTPUT);
  pinMode(BPHASE, OUTPUT);
  pinMode(BENBL, OUTPUT);
  digitalWrite(AENBL, HIGH);
  digitalWrite(BENBL, HIGH);

  pinMode(SW_PIN, INPUT);
  digitalWrite(SW_PIN, HIGH);

  delay(100);
}


void READ_VR(void) {
  VR_VALUE = analogRead(VR_PIN);
}


void READ_SW(void) {
  sw_value = digitalRead(SW_PIN);
}


void DELAY_WAIT(void) {
  for (int i = 0; i < (VR_VALUE / 10 + 7) ; i++) delayMicroseconds(100);
}


void MOTOR_FORWARD() {
  digitalWrite(APHASE, HIGH);
  DELAY_WAIT();
  digitalWrite(BPHASE, HIGH);
  DELAY_WAIT();
  digitalWrite(APHASE, LOW);
  DELAY_WAIT();
  digitalWrite(BPHASE, LOW);
  DELAY_WAIT();
}


void MOTOR_REVERSE() {
  digitalWrite(APHASE, HIGH);
  DELAY_WAIT();
  digitalWrite(BPHASE, LOW);
```

```
68   DELAY_WAIT();
69   digitalWrite(APHASE, LOW);
70   DELAY_WAIT();
71   digitalWrite(BPHASE, HIGH);
72   DELAY_WAIT();
73 }
74
75
76 void MOTOR_STOP() {
77   DELAY_WAIT();
78 }
79
80
81 void loop()
82 {
83   char val;
84   READ_VR();
85   READ_SW();
86   MOTOR_FORWARD();
87 }
```

（※文責: 家山剣）

**TestL6470.ino**

Listing C.12　TestL6470.ino

```
1
2  #include <SPI.h>
3  #include <MsTimer2.h>
4
5
6  #define PIN_SPI_MOSI 11
7  #define PIN_SPI_MISO 12
8  #define PIN_SPI_SCK 13
9  #define PIN_SPI_SS 10
10 #define PIN_BUSY 9
11
12
13 void setup()
14 {
15   pinMode(PIN_SPI_MOSI, OUTPUT);
16   pinMode(PIN_SPI_MISO, INPUT);
17   pinMode(PIN_SPI_SCK, OUTPUT);
18   pinMode(PIN_SPI_SS, OUTPUT);
19   pinMode(PIN_BUSY, INPUT);
20
21   SPI.begin();
22   SPI.setDataMode(SPI_MODE3);
```

```
  SPI.setBitOrder(MSBFIRST);
  Serial.begin(19200);
  digitalWrite(PIN_SPI_SS, HIGH);

  L6470_resetdevice();
  L6470_setup();

  MsTimer2::set(50, flash);
  MsTimer2::start();
}


void loop()
{
  L6470_run(0, 3000);
  delay(1000);
}


void L6470_setup()
{
  L6470_setparam_acc(0x40);
  L6470_setparam_dec(0x40);
  L6470_setparam_maxspeed(0x40);
  L6470_setparam_minspeed(0x01);
  L6470_setparam_fsspd(0x3ff);

  L6470_setparam_kvalhold(0x50);
  L6470_setparam_kvalrun(0x50);
  L6470_setparam_kvalacc(0x50);
  L6470_setparam_kvaldec(0x50);
  L6470_setparam_stepmood(0x03);
}


void flash()
{
  Serial.print( L6470_getparam_abspos());
  Serial.print(":");
  Serial.println( L6470_getparam_speed());
}


void L6470_run(int dia, long spd) {
  if (dia == 1)
    L6470_transfer(0x51, 3, spd);
  else
    L6470_transfer(0x50, 3, spd);
```

```
71 }
```

（※文責: 家山剣）

**manual.ino**

Listing C.13　manual.ino

```
1
2  /*
3     #Discription
4     Interaction Elements Gropu3 : PopUpShelf (2021/12/03 01:25)
5     Stepping : Nema23
6     Servo : MGS90
7  */
8
9
10 /*
11    #arguments
12    dia -> 1:direction + 0:direction -
13    spd -> (20bit)(0.015*spd[step/s])
14    pos -> (22bit)
15    n_step -> (22bit)
16    act -> 1:set mark 0:reset mark
17    mssec ->
18    val -> write regista
19
20    #L6470 commands
21    L6470_run(dia,spd);
22    L6470_move(dia,n_step);
23    L6470_goto(-1 * pos);
24    L6470_gohome();
25    L6470_gomark();
26    L6470_resetpos();
27    L6470_resetdevice();
28    L6470_softstop();
29    L6470_hardstop();
30    L6470_softhiz();
31    L6470_hardhiz();
32
33    #Servo commands
34    pushservo();
35    middlepushservo();
36    pullservo();
37 */
38
39
40 //include
41 #include <SPI.h>
```

```
42 #include <MsTimer2.h>
43 #include <Servo.h>
44 #include <Wire.h>
45
46
47 //pin settings
48 #define PIN_SPI_MOSI 11
49 #define PIN_SPI_MISO 12
50 #define PIN_SPI_SCK 13
51 #define PIN_SPI_SS 10
52 #define PIN_BUSY 9
53 #define PIN_SERVO 5
54
55
56 //parm
57 const float width = 1100;
58 unsigned int loop_cnt = 0;
59
60
61 //instance
62 Servo myservo;
63
64
65 void setup()
66 {
67   Serial.begin(9600);
68
69   //set pinmode
70   pinMode(PIN_SPI_MOSI, OUTPUT);
71   pinMode(PIN_SPI_MISO, INPUT);
72   pinMode(PIN_SPI_SCK, OUTPUT);
73   pinMode(PIN_SPI_SS, OUTPUT);
74   pinMode(PIN_BUSY, INPUT);
75   pinMode(PIN_SERVO, OUTPUT);
76
77   //set SPI
78   SPI.begin();
79   SPI.setDataMode(SPI_MODE3);
80   SPI.setBitOrder(MSBFIRST);
81   digitalWrite(PIN_SPI_SS, HIGH);
82
83   //set Wire
84   Wire.begin();
85   Wire.setClock(400000);
86
87   delay(100);
88
89   //set motor module
```

```
  L6470_resetdevice();
  L6470_setup();

  MsTimer2::set(50, flash);
  MsTimer2::start();
}


void loop()
{
    delay(2000);
    myservo.write(0);
    delay(2000);
    L6470_goto(-8500);
    delay(3000);
    myservo.write(155);
    delay(3000);
    while (1);

}


void L6470_setup() {
  L6470_setparam_acc(0x40);
  L6470_setparam_dec(0x40);
  L6470_setparam_maxspeed(0x40);
  L6470_setparam_minspeed(0x01);
  L6470_setparam_fsspd(0x3ff);

  L6470_setparam_kvalhold(0x50);
  L6470_setparam_kvalrun(0x50);
  L6470_setparam_kvalacc(0x50);
  L6470_setparam_kvaldec(0x50);
  L6470_setparam_stepmood(0x03);
}


void flash() {
  Serial.print("abs = ");
  Serial.print( L6470_getparam_abspos());
  Serial.print(":");
  Serial.print("spd = ");
  Serial.print( L6470_getparam_speed());
  Serial.println();
}
```

```
138 void L6470_run(int dia, long spd) {
139   if (dia == 1)
140     L6470_transfer(0x51, 3, spd);
141   else
142     L6470_transfer(0x50, 3, spd);
143 }
144
145
146 void L6470_stepclock(int dia) {
147   if (dia == 1)
148     L6470_transfer(0x59, 0, 0);
149   else
150     L6470_transfer(0x58, 0, 0);
151 }
152
153
154 void L6470_move(int dia, long n_step) {
155   if (dia == 1)
156     L6470_transfer(0x41, 3, n_step);
157   else
158     L6470_transfer(0x40, 3, n_step);
159 }
160 void L6470_goto(long pos) {
161   L6470_transfer(0x60, 3, pos);
162 }
163
164
165 void L6470_gohome() {
166   L6470_transfer(0x70, 0, 0);
167 }
168
169
170 void L6470_gomark() {
171   L6470_transfer(0x78, 0, 0);
172 }
173
174
175 void L6470_resetpos() {
176   L6470_transfer(0xd8, 0, 0);
177 }
178
179
180 void L6470_resetdevice() {
181   L6470_send_u(0x00);
182   L6470_send_u(0x00);
183   L6470_send_u(0x00);
184   L6470_send_u(0x00);
185   L6470_send_u(0xc0);
```

```
186 }
187
188
189 void L6470_softstop() {
190   L6470_transfer(0xb0, 0, 0);
191 }
192
193
194 void L6470_hardstop() {
195   L6470_transfer(0xb8, 0, 0);
196 }
197
198
199 void L6470_softhiz() {
200   L6470_transfer(0xa0, 0, 0);
201 }
202
203
204 void L6470_hardhiz() {
205   L6470_transfer(0xa8, 0, 0);
206 }
207
208
209 long L6470_getstatus() {
210   long val = 0;
211   L6470_send_u(0xd0);
212   for (int i = 0; i <= 1; i++) {
213     val = val << 8;
214     digitalWrite(PIN_SPI_SS, LOW);
215     val = val | SPI.transfer(0x00);
216     digitalWrite(PIN_SPI_SS, HIGH);
217   }
218   return val;
219 }
220
221
222 void L6470_send(unsigned char add_or_val) {
223   while (!digitalRead(PIN_BUSY)) {
224   }
225   digitalWrite(PIN_SPI_SS, LOW);
226   SPI.transfer(add_or_val);
227   digitalWrite(PIN_SPI_SS, HIGH);
228 }
229
230
231 void L6470_send_u(unsigned char add_or_val) {
232   digitalWrite(PIN_SPI_SS, LOW);
233   SPI.transfer(add_or_val);
```

```
234   digitalWrite(PIN_SPI_SS, HIGH);
235 }
236
237
238 void L6470_busydelay(long time) {
239   while (!digitalRead(PIN_BUSY)) {
240   }
241   delay(time);
242 }
243
244
245 long L6470_getparam(int add, int bytes) {
246   long val = 0;
247   int send_add = add | 0x20;
248   L6470_send_u(send_add);
249   for (int i = 0; i <= bytes - 1; i++) {
250     val = val << 8;
251     digitalWrite(PIN_SPI_SS, LOW);
252     val = val | SPI.transfer(0x00);
253     digitalWrite(PIN_SPI_SS, HIGH);
254   }
255   return val;
256 }
```

（※文責: 家山剣）

**main.ino**

Listing C.14　main.ino

```
1
2 /*
3    #Discription
4    Interaction Elements Gropu3 : PopUpShelf (2021/12/07)
5    ToF Sensor : VL53L1X
6    Stepping : Nema23
7    Servo : MGS90
8    Display : Grove-16*2 LCD (White on Blue)
9 */
10
11
12 //include
13 #include <SPI.h>
14 #include <Servo.h>
15 #include <Wire.h>
16 #include <VL53L1X.h>
17 #include <MsTimer2.h>
18 #include "rgb_lcd.h"
19
```

```
20
21 //pin settings
22 #define PIN_SPI_MOSI 11
23 #define PIN_SPI_MISO 12
24 #define PIN_SPI_SCK 13
25 #define PIN_SPI_SS 10
26 #define PIN_BUSY 9
27 #define PIN_SERVO 8
28
29
30 //param
31 float hand_pos;
32 float hand_pos_con;
33 float head_pos;
34 float th = 17;
35 const float width = 1100;
36 unsigned int sensor_val;
37 unsigned int loop_cnt = 0;
38
39
40 //instance
41 VL53L1X sensor;
42 rgb_lcd lcd;
43 Servo myservo;
44
45
46 void setup()
47 {
48   Serial.begin(19200);
49
50   //set pinmode
51   pinMode(PIN_SPI_MOSI, OUTPUT);
52   pinMode(PIN_SPI_MISO, INPUT);
53   pinMode(PIN_SPI_SCK, OUTPUT);
54   pinMode(PIN_SPI_SS, OUTPUT);
55   pinMode(PIN_BUSY, INPUT);
56   pinMode(PIN_SERVO, OUTPUT);
57
58   //set SPI
59   SPI.begin();
60   SPI.setDataMode(SPI_MODE3);
61   SPI.setBitOrder(MSBFIRST);
62   digitalWrite(PIN_SPI_SS, HIGH);
63
64   //set Wire
65   Wire.begin();
66   Wire.setClock(400000);
67
```

```
  //set MsTimer2
  MsTimer2::set(50, flash);
  MsTimer2::start();

  //set sensor
  sensor.setTimeout(10000);
  if (!sensor.init()) {
    Serial.println("Failed");
    while (1);
  }
  sensor.setDistanceMode(VL53L1X::Long);
  sensor.setMeasurementTimingBudget(50000);
  sensor.startContinuous(50);

  //set lcd
  lcd.begin(16, 2);
  lcd.clear();

  //set servo
  myservo.attach(PIN_SERVO);
  myservo.write(0);

  //set motor module
  L6470_resetdevice();
  L6470_setup();
}


void loop()
{
  //get loop count
  GetLoopCount();

  //get sensor value
  GetSensorValue();

  //get hand position
  GetHandPosition();

  //send Serial sign
  SendSignal();

  //control motors and judge in or out
  ControlMotors();

  //show the parm on display
  LcdShow();

```

```
116   //judge get timeout
117   TimeOut();
118
119   Serial.println();
120 }
121
122
123 void GetLoopCount() {
124   Serial.print(loop_cnt);
125   Serial.print(":");
126
127   loop_cnt++;
128 }
129
130
131 void GetSensorValue() {
132   sensor_val = sensor.read();
133
134   Serial.print("sensor_val = ");
135   Serial.print(sensor_val);
136   Serial.print(":");
137 }
138
139
140 void GetHandPosition() {
141   if (InOut() == true) {
142     hand_pos = sensor.read();
143   }
144   else {
145     hand_pos = head_pos;
146   }
147
148   Serial.print("hand_pos = ");
149   Serial.print(hand_pos);
150   Serial.print(":");
151 }
152
153
154 boolean InOut() {
155   if (sensor_val < width) {
156     return true;
157   }
158   else {
159     return false;
160   }
161 }
162
163
```

```
boolean TimeOut() {
  if (sensor.timeoutOccurred()) {
    Serial.println("TIMEOUT");
    return true;
  }
  else {
    return false;
  }
}
```

Listing C.15 control.ino

```

void ControlMotors() {
  while (InOut() == true) {
    GetSensorValue();
    GetHandPosition();
    hand_pos_con = hand_pos * th * -1;
    head_pos = hand_pos;
    L6470_goto(hand_pos_con);
    PushServo();
    delay(100);
  }

  while (InOut() == false) {
    GetSensorValue();
    GetHandPosition();
    L6470_hardstop();
    PullServo();
    delay(100);
  }
}
```

Listing C.16 lcd.ino

```

void LcdShow()
{
  lcd.setCursor(0, 0);//00
  lcd.print("ToF = ");
  if (hand_pos < 10000 && hand_pos >= 1000) {
    lcd.print("0");
  }
  else if (hand_pos < 1000 && hand_pos >= 100) {
    lcd.print("00");
  }
  else if (hand_pos < 100 && hand_pos >= 10) {
    lcd.print("000");
  }
  else if (hand_pos < 10 && hand_pos >= 0) {
```

```
16     lcd.print("0000");
17   }
18   else {
19     lcd.print("00000");
20   }
21   lcd.print(sensor_val);
22
23   lcd.setCursor(0, 1);//01
24   lcd.print("sign : ");
25   lcd.print(SendSignal());
26 }
```

Listing C.17 servo.ino

```
1
2 void PushServo() {
3   myservo.write(0); //push
4 }
5
6
7 void MiddlePushServo() {
8   myservo.write(90);
9 }
10
11
12 void PullServo() {
13   myservo.write(160); //pull
14 }
```

Listing C.18 signal.ino

```
1
2 /*
3    e : stop program (into infinity loop)
4    h : go to home position
5    r : reset motor axis and device
6    s : emergence stop motor
7 */
8
9
10 int SendSignal() {
11   char sign;
12   if (Serial.available() > 0) {
13     sign = Serial.read();
14     if (sign == 'e') {
15       while (1) {};
16       return 1;
17     }
18     else if (sign == 'h') {
19       L6470_gohome();
```

```
      exit(0);
      return 2;
    }
    else if (sign == 'r') {
      L6470_resetpos();
      L6470_resetdevice();
      return 3;
    }
    else if (sign == 's') {
      L6470_hardhiz();
      return 4;
    }
    else {
      return 0;
    }
  }
}
```

Listing C.19 stepping.ino

```

/*
  L6470 comands
  ----------------------------
  dia 1 : order / 0 : reverse
  spd (20bit)(0.015*spd[step/s])
  pos (22bit)
  n_step (22bit)
  act 1 : mark / 0 : reset
  mssec
  val
  ----------------------------
*/


void L6470_setup() {
  L6470_setparam_acc(0x40);
  L6470_setparam_dec(0x40);
  L6470_setparam_maxspeed(0x40);
  L6470_setparam_minspeed(0x01);
  L6470_setparam_fsspd(0x3ff);
  L6470_setparam_kvalhold(0x50);
  L6470_setparam_kvalrun(0x50);
  L6470_setparam_kvalacc(0x50);
  L6470_setparam_kvaldec(0x50);
  L6470_setparam_stepmood(0x03);
}


```

```
30 void flash() {
31   Serial.print("abs = ");
32   Serial.print( L6470_getparam_abspos());
33   Serial.print(":");
34   Serial.print("spd = ");
35   Serial.print( L6470_getparam_speed());
36   Serial.println();
37 }
38
39
40 void L6470_run(int dia, long spd) {
41   if (dia == 1)
42     L6470_transfer(0x51, 3, spd);
43   else
44     L6470_transfer(0x50, 3, spd);
45 }
46
47
48 void L6470_move(int dia, long n_step) {
49   if (dia == 1)
50     L6470_transfer(0x41, 3, n_step);
51   else
52     L6470_transfer(0x40, 3, n_step);
53 }
54 void L6470_goto(long pos) {
55   L6470_transfer(0x60, 3, pos);
56 }
57
58
59 void L6470_gotodia(int dia, int pos) {
60   if (dia == 1)
61     L6470_transfer(0x69, 3, pos);
62   else
63     L6470_transfer(0x68, 3, pos);
64 }
65
66
67 void L6470_gohome() {
68   L6470_transfer(0x70, 0, 0);
69 }
70
71
72 void L6470_gomark() {
73   L6470_transfer(0x78, 0, 0);
74 }
75
76
77 void L6470_resetpos() {
```

```
78   L6470_transfer(0xd8, 0, 0);
79 }
80
81
82 void L6470_resetdevice() {
83   L6470_send_u(0x00);
84   L6470_send_u(0x00);
85   L6470_send_u(0x00);
86   L6470_send_u(0x00);
87   L6470_send_u(0xc0);
88 }
89
90
91 void L6470_softstop() {
92   L6470_transfer(0xb0, 0, 0);
93 }
94
95
96 void L6470_hardstop() {
97   L6470_transfer(0xb8, 0, 0);
98 }
99
100
101 void L6470_softhiz() {
102   L6470_transfer(0xa0, 0, 0);
103 }
104
105
106 void L6470_hardhiz() {
107   L6470_transfer(0xa8, 0, 0);
108 }
109
110
111 long L6470_getstatus() {
112   long val = 0;
113   L6470_send_u(0xd0);
114   for (int i = 0; i <= 1; i++) {
115     val = val << 8;
116     digitalWrite(PIN_SPI_SS, LOW);
117     val = val | SPI.transfer(0x00);
118     digitalWrite(PIN_SPI_SS, HIGH);
119   }
120   return val;
121 }
122
123
124 void L6470_transfer(int add, int bytes, long val) {
125   int data[3];
```

```
126   L6470_send(add);
127   for (int i = 0; i <= bytes - 1; i++) {
128     data[i] = val & 0xff;
129     val = val >> 8;
130   }
131   if (bytes == 3) {
132     L6470_send(data[2]);
133   }
134   if (bytes >= 2) {
135     L6470_send(data[1]);
136   }
137   if (bytes >= 1) {
138     L6470_send(data[0]);
139   }
140 }
141
142
143 void L6470_send(unsigned char add_or_val) {
144   while (!digitalRead(PIN_BUSY)) {
145   }
146   digitalWrite(PIN_SPI_SS, LOW);
147   SPI.transfer(add_or_val);
148   digitalWrite(PIN_SPI_SS, HIGH);
149 }
150
151
152 void L6470_send_u(unsigned char add_or_val) {
153   digitalWrite(PIN_SPI_SS, LOW);
154   SPI.transfer(add_or_val);
155   digitalWrite(PIN_SPI_SS, HIGH);
156 }
157
158
159 void L6470_busydelay(long time) {
160   while (!digitalRead(PIN_BUSY)) {
161   }
162   delay(time);
163 }
164
165
166 long L6470_getparam(int add, int bytes) {
167   long val = 0;
168   int send_add = add | 0x20;
169   L6470_send_u(send_add);
170   for (int i = 0; i <= bytes - 1; i++) {
171     val = val << 8;
172     digitalWrite(PIN_SPI_SS, LOW);
173     val = val | SPI.transfer(0x00);
```

```
174     digitalWrite(PIN_SPI_SS, HIGH);
175   }
176   return val;
177 }
```

（※文責: 家山剣）

# 参考文献


[1] レーザーカッターでアクリルを導光板に加工して平面発光照明を作ってみた
https://kohacraft.com/archives/202104021539.html

[2] カスタムオーダー導光板
https://www.good-arm.com/category/item/itemgenre/custom-order-lgp/

[3] 第 50 回 フルカラー LED テープを使ったちょっとお洒落な電子工作 〜Arduino でパーツやセンサを使ってみよう
https://deviceplus.jp/hobby/entry050/

[4] L. Vink, V. Kan, K. Nakagaki, D. Leithinger, S. Follmer, P. Schoessler, A. Zoran, and H. Ishii. TRANSFORM as Dynamic and Adaptive Furniture. In Proceedings of CHI' 15, pp. 183. 2015.
https://tangible.media.mit.edu/project/transform-as-dynamic-and-adaptive-furniture/

[5] からくりすと
http://karakurist.jp/

[6] 姉崎祐樹, 辻航平, 湯浅基, 有澤寛則, 浅川玄, ポインティング指向ジェスチャインタフェースによるユーザビリティ向上のためのデザインガイドライン, 人間中心設計, 2014, 10 巻, 1 号, p. 27-35.

[7] 正畑智徳, 武田祐樹, 中道上, 渡辺恵太, 山田俊哉, 図書案内のためのスポットライト型ポインティングシステム, 情報処理学会インタラクション 2021, 2021, p. 447-450.

[8] 北の国から電子工作（仮） L6470 を Arduino で簡単に動かすスケッチ
http://spinelify.blog.fc2.com/blog-entry-41.html

[9] 図解！ HTMLCSS のツボとコツがゼッタイにわかる本 著者：中田 亨