

卒業研究ノート

中村 龍造

2024-07-20

研究生活が始まって半年ながら、研究ノートをとってみようと思う。といっても、研究にまつわる日記のような感じで使おうと思っている。なのでその日の研究に関するを中心には、いろいろメモしていこうと思う。

今日は卒論の修正リストを作った。はじめはめちゃくちゃ赤入れられてショックだったが、数えてみるとそんなに多くもないので、なんだか行けそうな気がする。13個くらい。はじめてちゃんと書いたにしては少ないんじゃないかな。バグの数だと思えば。

卒論修正リストを参考に、土日は卒論を直して、できれば日曜の午後くらいには佐藤先生にまた提出したいと思っている。懸念点としては、参考文献のいりそうな修正項目がいくつかあるということ。たぶん今回の修正項目、大まかに分けると、1. 参考文献を盛る、2. 内容を面白くする、の2つに分かれる。1は1章2章で、2は3章の構想で必要になる。参考文献をじっくり読み漁る時間も無いので、そのへんは美味しいことバランスを取ったほうがいいかもと思ったが、どちらも半分くらいやつたらいいんじゃないかなと思う。

いまから久しぶり(6日ぶり)によわ toio システムのほうをいじくる。改めて振り返ったら、プロトタイピングをしていないのがおれだけだったのでなんだか嫌な気分。「大学は失敗

をたくさんするための場」とよく言われるが、その点ではおれが一番学びを活かせていないのかもしれない。だって失敗すると「あいつはしょぼいやつなんだな」って思われそうで嫌なんだもん。失敗すればするほど嫌なことしか積み重ならない。今回がはじめての「良い失敗」を積み重ねる機会になるか。

さて、ではよわ toio(仮称)を実装していく。とはいっても設計だけは以前のおれがそれなりにやってくれている。いまのおれがするのはそれを形にすること。目標は「何らかの環境データを取得した toio に何らかの反応を取らせること」。

今回やることは次の4個

1. toio と Unity の接続の確認
2. M5 と Unity の接続の確認
3. Unity でもらってきたデータをまとめるまでの流れをつくる
4. まとめたデータを評価クラスで「評価」してみる

toio と Unity の接続の確認

接続方法は知る限り2通り。1つは BLE(Bluetooth Low Energy)接続をする方法、もう1つは SampleConnectType コンポーネントを追加し、ConnectType を Real に変更する方法。どちらも試してみたけど、接続までの速度も変わらないようなので、実装が楽な2つめの方法で行こうと思う。

この方法を使うと、Unity 上の toio にさせた動きを現実空間上の toio にも反映させてくれる。なので仮想空間の toio を相手にシステムを組んできさえいれば、自然と現実空間システムも組めているという寸法。便利。

今始めて知ったんだが、*LATEX* はアンダーバーを処理できないらしい。特殊文字なんだね。半角のカッコはおそらく英語圏でも使われるから普通に使えたのが異だった。

M5 と Unity の接続の確認

toio の接続が無事完了して、今後の接続方式も固まつたので、次に M5 の確認。前にボードの違いでエラーが生じたが、正しいのを選べば今回も無事に通過。USB 経由でデータが送られていることも確認できた。

キーボードガタガタ打つのも楽しいが、大量に入力する場合は面倒なので PC でも音声入力ができないか調べてみた。どうやらいけるらしい。

<https://x.gd/3VbTT>

さて、音声入力をやっている。意外と精度が良いのかもしけれこれでキーボードの代わりに音声入力ができるようになれば万々歳。

以前やっていた時にも感じていたが、M5 と Bluetooth の接続は手作業でやらなければいけないのが非常に面倒。が、確認してみたら Bluetooth 設定の COM ポートから名前まで見ることができるようなので楽できそう。

ここからさらに COM ポート設定をさらってデバイス名と同じ名前の COM ポートを自動で接続するようにすれば、更に楽になるかもしれない。

<https://qiita.com/ta-oott/items/2d74409cca729629595e>

Bluetooth での入力が確認できたら、Unity 側での設定は慣れたもの。COM ポートも以前



図 1 Bluetooth 設定の画面。COM ポートタブでは Bluetooth 接続されているデバイスのポート、方向、名前を見ることができ、手作業での接続が楽になりそう。

より安全に確認できるようになったので、設定も簡単。念のため USB のポートも確認をしておけば安全。

そういえば忘れていたけれど、いまのプログラムだと Bluetooth 通信に失敗すると Editor がクラッシュするんだよね。困ったなあ。

あーきたきた。ようやく来た。

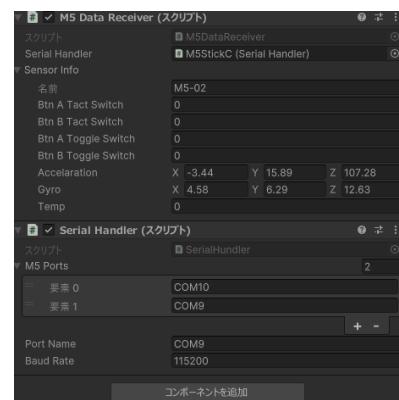


図 2 Unity に M5StickC のデータが届いている様子。M5 Data Receiver はデータを受け取り、Serial Handler は Bluetooth シリアル通信の処理を担っている（らしい）

はじめ COM11, 10, 9 を登録してたんだけど、検証したら COM11 はデータ送られてな

かった。そのうえ Unity 側のボーレートも間違ってた。まあ直せる範囲のバグで良かつたね。

よし、接続成功時はきちんとシーンも終わってくれる。一瞬「カクッ」となるのが恐怖を煽ってきていいよね。よくない。おめーなにクラッシュしてんだよ。絶対その不具合直すから。大方例外処理が不十分なんでしょうね。

本日のまとめ

さて、なんだかんだ 2 時間ぐらい作業したので、今日の卒研は終了する。(みんなもっとやってるって? おれはおれの速度で進む。あとおれ脚質差だから。巻き返すから、気合で。)

今回始めて LATEX で記録をつけてみたけれど、思ったより楽しい。なんだかリッチな見た目、リッチな書体で、まあ確かにメモを書くのには大仰というのも分かる。でもこの大仰さが、万年筆を使っているときのようなクセの強さと、それでしか得られない謎の快感があっていいな。癖になる。

今日は toio および M5StickC の接続とデータの受け渡しを確認した。次回は toio に付属しているセンサーの情報を取得する方法の調査と、取得データの集約まで進めたい。

2024.07.21

卒論の修正を進めた。やっぱり思ったより時間がかかるって、2時間でようやく3章がマシになったかならないかくらいの感じ。さらに赤が加わって、参考文献も必要になるのか。あと4日、間に合うかな……

ひとまず3章を直しきった次点で佐藤先生に共有した。もしかしたら「なんでこんな意味のないもの共有するんだろう?」とか思われてた

ら人間不信になって退学も辞さない。そういうこと考えるからおまえ「超危険メンヘラ」って言われるんだよ。

といえばレビューコメントの扱い方、あれで良かったのかな? もしかしたら修正報告をして、向こうで Resolve とか Accept をするスタイルだったかもしれない。うーんわからん。まあにか致命的にクソな使い方をしてる場合はきっと報告してくれるでしょう。よろ。

2024.07.23

論文を添削に出したら全然できてないって怒られた上に「コメントちゃんと見えてる?」とかいう煽りを食らった。あらゆる憎しみを抑えて「ありがとうございます!」って返せたあのときのおれには今年最大級の称賛を送っていい。ああいう対応が今後も続くなら転ゼミも考慮する。佐藤クンは有能な人間が好きな感じがあるので、僕とは合わないとは思うけど。1 年後にゼミレビューでボロクソにこき下ろすために耐えるか、苦しいのでお引越しするか。

恨みは尽きることがないので仕事の話をしよう。論文を直す。佐藤クンは実験風景を出せって言ってたけど、2 日前で論文書き上がってないのに実験風景を作り出すってのは不可能。なので参考文献をあたって、論文としての品質を上げる

参考文献で知りたいこと

1. ロボット高度化に対する姿勢たち
キーワードは
 - (a) 関係論的なロボット (human-dependent robot)
 - (b) 社会的なロボット (social robot)
2. 弱いロボット特有の活かすに値する特徴
3. 弱いロボットの具体例

- (a) Muu
 - (b) ゴミ箱ロボット
 - (c) Talking Bone
 - (d) 岡田さん以外に弱いロボットやってる
人の弱いロボット
 - (e) NICOBOTもここで良さそう
4. ロボットへの心理評価. 印象, あるいは「はじめに」で述べたような「もっと高性能に」っていう要求が増えるのかどうか
5. 「弱さ」の定量化手法

ロボット高度化に対する姿勢たち

STB: Human-Dependent Sociable Trash Box

ゴミ箱ロボットの論文 [1]. 運良く ACM に転がってた. social robot と human-dependent robot の定義の違いは掴んだ.

social robot

ロボット自身の何らかの目標や動機を満たすために人々との社会的相互作用に参加するもの

human-dependent robot

人間の助けに依存して目標を達成する.

Kinzer の Tweenbot が代表例.

「あそこに行きたい」「これをしたい」という意図を示し, 人間がそれを読み取ることでロボットの目標達成を助けてくれることが分かった.

したがって, social robot は目標達成に伴う行動を人と協働して行う. 一方 human-dependent robot は「こういう目標を達成したい」という意思表示のみを行い, それを見た人が自ずからロボットを助けてくれるようなロボットである [2]. こういう, 周囲からの関与やアシストを引き出すことによる問題解決のやり方は赤子に

も見られ, 「関係論的な行為方略」と呼ばれる [1]. 対立概念は「個体能力的な行為方略」と呼ばれ, 個体として, 自分ひとりで問題解決をするやり方をいう.

弱いロボット特有の強み

これまでの流れ

Social Robot

人とロボットどちらも目の前のタスクに関する目標を持っている [3].

Human-dependent Robot

ロボットだけが目標を持っている. 何かしらの手段で目標を提示し, それを見た人がロボットを助けてくれる.

⇒ 不特定多数の人間をタスク遂行のパートナーとすることができます [4]

弱いロボット

Human-dependent Robot の一種.
Tweenbot との違いは, 目標の提示方法. Tweenbot は取り付けられた旗に書かれたテキストを読む. 弱いロボットの1つであるゴミ箱ロボットは, 動きと音声, タスクに関連した外観によって目標を読み取らせている.

1 2024.07.24

Notion に書いたプロットが大分イカしてたので転記する。

1.1 はじめに

- Social Robot (Breazeal)[3]: 「協働する人間とロボットは、それぞれタスクに関連した目標を持っている」

- Human-dependent Robot (Kinzer)[4]: 「ロボットのやりたいことを伝えられれば、そのへんの人間とも協働できるよね」
- 弱いロボット（関係論的なロボット）(岡田)[1]: 「やりたいことを効果的に伝えてサポートしてもらうために、赤ちゃんのテクニック（関係論的な行為方略）を使おう」

Human-dependent Robot から分かるのは、完全な自律性を持たなくとも、ロボットは現実世界の中で目標達成が可能であること弱いロボットから分かるのは、Human-dependent Robot をデザインするときに、人間からのサポートを上手に引き出すテクニックがあること。（関係論的な行為方略）

1.2 関連研究

- Sociable Trash Box[2]: 弱いロボットの好例
- INAMO[1]: 分散型エージェントは多数のロボットを同時に扱う必要がある。この研究は群ロボットに関係論的な行為方略を応用しようとしている。

課題: 既存の「弱いロボット」は、ハードウェア・ソフトウェアが強固に結びついてデザインされており、異なる目標に対する再利用性、異なる環境における適応力について課題がある。

コアとなるロボットはそのまま、要求に合わせてハードウェアを変えることで、ロボットが提供できる機能も変えているという例に、HERMITS[5] がある。

自走用のモータを備えた小型ロボットエージェント (toio) に、「Shell」と呼ばれる交換可能なハードウェアを取り付けた研究。ハードウェアを変えることによって、掴む、持ち上げる、風を送る、などといった様々な機能をもたらせることができている。

HERMITS では、ハードウェアの交換によってロボットの役割を変えていた。本研究では、ロボットのセンシング及びアクションのプログラムの交換によって、異なる環境にも応用可能な弱いロボットを開発する。

1.3 toio を操作できるようにする

コントローラーの入力の扱われ方とか、サンプルコードとか <https://hakonebox.hatenablog.com/entry/2018/04/15/125152>
<https://sorceryforce.net/ja/tips/unity-input-gamepad#toc7>

このアセットすごいよお！！
<https://assetstore.unity.com/packages/tools/input-management/controller-tester-43621>

TextMeshPro で日本語を使う
<https://zenn.dev/kametani256/articles/63c083ab318136> https://gist.github.com/kgssi/ed2f1c5696a2211c1fd1e1e198c96ee4#file-japanese_full-txt

ジョイスティックのシステムちょっと理解できた。

joystick button っていうのが押しボタン系。方向キー (Dpad) 以外の押せるボタン一つ一つに番号が割り当てられてるので、ボタンごとの番号をチェックした後、ボタンの機能を割り当ればいい。

方向キーの扱いは、取りうる値が -1, 0, 1 のジョイスティックっていう感じ。で、ジョイスティック側にもそういうのをいろいろ識別するために、「軸」が分けられている。例えば L スティックは第 1 軸に割り当てられてる。方向キー左右は第 7 軸。なので例えば方向キーとアナログスティックで同じ動きをさせたいなら、7, 8 軸で値取るようにしたうえで、サブボタンにアナログスティックを割り当てる？

2024.07.25

うおおおおお先人ありが
とーーーー！！！！！でも先にずっとすご
いの作られて悔しいいーーーー！！！！！
https://qiita.com/tetunori_lego/items/c0cf6999a7667756441f <https://tetunori.github.io/toioCoreCubeGampadControl/>

2024.07.29

プレゼンの構成がキモいねえ。昨日ドラフト？とかいうのは作ったけれども、言いたいことが不明瞭だし文量は多いしで、これじゃあまた佐藤クンにボコボコにされるねえ。

プレゼンの構成はこうだ

1. 目的と背景、ざっくりと何をするのか
2. どんな先行事例があるのか、それらが抱えている問題は何か
3. 先行事例の問題を解決するためにおれは何をするのか
4. やることの具体的な内容を述べよ
5. 実験した？したならどんなことしたの？
6. 結果と考察
7. まとめと展望

作っては作り直してを何度も繰り返す。おれと佐藤クンの間に大きな承認が訪れるまで。

目的と背景

生活環境を整えてくれるゆるいロボットが作りたいです。ミニオンみたいな、数が多くて、色んなところで作業してるが、なんかうまくいかないやつがいる

人間のパートナーとしてのロボットが注目されている。「弱いロボット」によって、ロボッ

トの「できない」ことを利用する方法は、高性能化以外での現実世界でタスク遂行可能なロボットの形を示している。

先行事例と課題

弱いロボットの課題: 1目標1ロボットのため、再利用性が低い HERMITS: コアロボットはそのまま、機能を変えることができている

システム面で機能の取り替えを行っていないので、おれがやりたい

おれのやること

- 全体の構想: 生活環境を整えてくれるゆるいロボットをつくる
- 弱さの定義: ロボットに弱さを取り入れるので、「弱さって何よ」に答える
- 弱いロボットの課題解決: 目標ごとにコアのロボットはそのまま、システムをすることで機能・役割を多様化させ、再利用性を高める

具体的な内容

ロボット管理システムをつくります4つのサブシステムからできます

実験した？

目的: 家の中でロボットが移動するときってどんな課題があるんだろう？内容: 家の中でロボット動かしました。写真

結果と考察

わかったこと: 結構落ちる。防水・防塵機能も必要。でも落ちるなら被害の少ない軽量なロボットがいいなあ

まとめと展望

これまで述べた全てをまとめる。あと今後やること

2024.07.30

昨日勇気を振り絞って佐藤先生に練習を見てもらい、もらったフィードバックをもとに発表資料および研究を充実させていく。フィードバックはちゃんと日記にメモしておいたので安心。

- 人とロボットが協働している風景のイラスト
- システム構成のワークフローの動画
- 課題を決めてロールプレイしてみる
- 台本をつくる

ユーワツだ。

2024.07.31

「弱さ」の定義？

岡田美智男「〈弱いロボット〉の思考 わたし・身体・コミュニケーション」p.204

先に述べたように、わたしたちの共同行為を生み出すためのポイントは、自らの状況を相手からも参照可能なように表示しておくことである。「今どんなことをしようとしているのか」「どんなことに困っているのか」、そうした〈弱さ〉を隠さず、ためらうことなく開示しておくことで、お掃除ロボットは周りの手助けを上手に引き出しているようなのである。

もう一つのポイントは、相手に対する〈敬意〉や〈信頼〉のようなものではないだろうか。お互いの弱いところを開示しあ

い、それを補い合う。一方で、その〈強み〉を称えあってもいる。この掃除ロボットは相手を信頼してなのか、その部屋の壁になんのためらいもなく、委ねることをする。一方で、わたしたちも「へーこんなところのホコリを丹念に水集めてしまうわけ？」「すごい、これにはかなわないなあ……」というわけで、「ここはロボットに任せとおこう！」ということを徹底させている。

2024.08.01

動画編集をします。

久々に Aviutl 開いたら早速エラー発生。mp4 ファイルが読み込めなかった。Aviutl 本体と、mp4 を読み込めるようにするプラグインの L-Smash Works を入れ直して、L-Smash Works のシステム優先度を最高にしたら落ち着いた。動画ロード時の安定化？なるものが原因だったらしい。

システム概要に使う動画はゴミ捨てのやつにしよう。2ショット使って贅沢に作ってるからね。

claude と相談

claude におれのプレゼンを説明して、突っ込まれそうなところを対策する。

1. 研究の背景: 研究の背景: 単独でのタスク遂行が可能な「強いロボット」(例: Honda ASIMO)との対比として、人間の助けを借りる「Human-dependent Robot」(例: Kinzer, Tweenbots) や、「弱さ」を取り入れることで効果的に人間の助けを引き出す「弱いロボット」(岡田ら, アイ・ボーンズ)について言及します。本研究では「弱いロボット」の概念を取り入れるため、弱いロ

ボットが人間の助けを引き出すことで、強いロボットでは困難なタスクの遂行が可能であることを述べます。

2. 本研究の目的: 本研究の対象となっている生活環境について述べます。生活環境には多様なタスクが存在し、それらすべてをロボット単体で処理することは困難です。そこで弱いロボットをタスクごとに分散配置することで、人間の助けを引き出しながら、生活環境におけるタスクをより効果的に解消できるようにします。
3. 弱いロボットの先行事例としては、「弱さ」を用いることでゴミを効果的に集める「Social Trash Box」、複数の弱いロボットを制御した例として「INAMO」を挙げます。これらのロボットは未だ単体ないし群れのように集まって活動するものであり、分散配置して異なるタスクに従事する弱いロボットの先行事例はないことを述べます。加えて、ロボットを分散配置した例とし tMIT の「HERMITS」を挙げます。こちらはハードウェアの入れ替えと分散配置によって様々なタスクをこなすことができるロボットを提案していますが、弱いロボットの概念は取り入れられておらず、弱さを用いることによってさらに対応可能なタスクの幅を広げることができるといえます。
4. 以上から本研究では、分散配置された弱いロボットが、人間の助けを引き出しながら生活環境を改善するシステムを提案します。本研究で開発するのは、1. 弱さを取り入れたロボット、2. ロボット管理システムの2つを開発します。また開発に先立って、岡田の著書から、本研究における「十分に弱い」ということを、「十分に弱い」ことを、「ロボットの遂行しようとし

ているタスクが明らかで、かつ、どのような課題が遂行を妨げているかが明らかである」と設定しました。

5. 次にシステム構成について述べます。本システムは次の3ステップから構成されます。1. まずロボットが、生活空間の異常を検知する。2. するとロボットは異常を人間にアピールする。3. ロボットのアピールに気づいた人間が、ロボットの意図を読み取り、ロボットの代わりに異常に対処する。なおここでいう「異常」とは、ゴミや不快な気温、忘れ物などを指します。また本システムを象徴するような、一連の流れを模した映像を流します。
6. システム構成について述べた後、ロボット管理システムの構成について述べます。ロボット管理システムはロボットが取得した環境情報を取得し、評価し、最終的にロボットが次行う行動を送信します。
7. 開発に先立って、ロボットが人間にアピールする際に、どのような振る舞いが効果的かを検証しました。この検証では、ロボットを人間が手動操作し、いくつかの状況を仮定してロボットが人間にアピールするロールプレイを行いました。
8. 検証の結果、ロボットが可能なアピールの種類、対象物の状態ごとの可能なアピール、弱い動きの3点の知見が得られました。
9. 最後に発表のまとめとして、本研究の目的、および開発するもの2つ、加えて今後開発するものについて述べます。今後開発するものは、ロボット管理システム、管理システムを用いたロボットの自動化、弱い動きの実装、生活環境での仕様に耐えるハードウェアの実装の4点を挙げました、

来そうな質問

1. 弱いロボットと強いロボットの違い
強い: 人間の助けをタスク遂行の一部に含まない
弱い: //含む
2. 生活環境における具体的なタスク
タスクは多様で、完全にカバーすることは難しい。
検証次点では、ゴミや忘れ物などの物体検知、電気消し忘れや気温などと言った環境データ取得、加えて目覚まし等のタスクを検証した。
今後必ずタスクは増えるので、利便理性を上げるためにも、新しいタスクの追加が簡単にできるようにする必要がある。
3. 人間の助けを引き出す際、倫理的な配慮や制限
弱いロボットによる行動誘発は、人間からの「助けてあげたい」という感情を引き出すものなので、強制してはいけない。そのため音や動きでのアピールは、あくまで「ロボットが必死になにかしようとしているが、できない」という風に移る必要があり、それがアラートであってはいけない。その点についても、どのようなアピールが人間に悪印象を与えるかには注意する必要がある。
4. 弱さのデザインにおいて注意する点
同上。加えて、見た目や振る舞いから与える印象を弱くする必要がある。岡田の言葉を借りれば「ヨタヨタ」して見える動きを実装する必要がある。またハードウェアデザインの面でも、岡田は柔らかさを感じさせる見た目が必要であるとしており、検討が必要。
5. 複数の弱いロボットを分散配置する際、ロ

ボット同士の連携や通信の実装

ロボットはすべてPCで集約管理することを想定している。しかし、もし処理の重さや位置関係によるラグが目立つ場合、中継機を用いた通信も視野にいれる必要がある。先行研究に挙げた HERMITS では、PC - RasberryPI - toio というパスで通信を行っていたため、参考にしたい。

6. ロボット管理システムはどのように複数のロボットの優先順位や作業分担を決定するか

現状では、あらかじめ個々のロボットに役割を設定しておく、対応する場所に配置する予定である。しかしながらタスクは環境によっても異なるため、異なる役割を担当可能にする必要がある。ロボットとその役割、作業優先度等を一元管理するUIの開発が必要である。

7. アピールが無視された場合、どのように対応するか

アピールが無視された場合であっても、人間に行動を強制するような振る舞いはできない。そのためもし無視された場合、ロボットは延々と対象物の周りを動き回るような結果になってしまう。そのような自体を防ぐためには、碎けて言えば「うざい」と思われないような関係性の構築が必要である。岡田はそのような関係性についても言及しているため、今後参考にしたい。

8. 「異常」の判断基準は？

対象となる環境ごとに、予め「正常」な状態の数値を記録し、評価システムで比較することで判断することを構想している。例えば気温であれば、たいたい27°Cあたりが快適な気温とされているので、そういう快適な状態のデータを、先行研究や環境からのデータ採取をもとに設定する。

9. ロールプレイで得られた「弱い動き」とは？

今回の検証で特に目立ったのは「動きが硬い」という点であった。岡田は人間と関わるロボットが硬いのは致命的である、と述べており、本研究においても同様に硬い動きをなくすようにしたい。また岡田の「ゴミ箱ロボット」では、ゴミを拾ってもらった際に人間に對しお礼の動作をする。弱いロボットはこうした「助け合い」の関係の構築が寛容であるため、本研究でもどのような動作であれば、感謝を伝えることができるかを兼用する必要がある。

10. 実用化に向けてどのような課題が残されているか

利便性を上げるために、タスクとその対応の関係付けをより簡単に行えるようにする必要がある。またロボットが活動する場所は環境によって多様に異なるため、環境ごとの地理情報を自動で取得・更新するシステムの開発も必要である。加えて、弱いロボットとして人間と関係していくうえで、現状のハードウェアでは印象が硬すぎるため、より柔らかい見た目をデザインする必要がある。具体的には、現在のロボットの上から、柔らかい素材やコミカルなハードウェアを被せることを構想している。これは HERMITS と重なるが、ハードウェアと役割を対応付けることで、同じロボットでも役割ごとに異なる個性を持たせることを視野に入れている。こうすることで、ロボット一体一体に対して愛着を抱かせ、弱いロボットと人間の望ましい関係の構築に寄与できると考えている。

2024.08.02

Today is Sotsuken Happyou Day!!!

しにそう。でも結構プレゼンやったでしょ？じゃああとは楽しそうに話すしか無い。質問もいくつか想定した。それ以外にぶち込まれる場合、もうどうしようもないのでヨタヨタと話すしか無い

佐々木さんが発表してる……なんだかんだ愛されるタイプだからいいよね。おれみたいにぶつ叩かれるタイプじゃない。

3面立体の演算順の話。確かに検証が必要だな。A・B・C = A・C・B かどうか。数学ならこれは等しいんだけどね。

あとは一番最後の「ちょっと一工夫」の部分がどうやってコンピュータに理解できる方法論にできるかどうか。

朝長さん

金継ぎで土岐センポイント稼ぎにいったア——つ！

響先輩

メッセージの動的な変化。確かに動的であることが特徴なら、そういう応用を考えたいよね。

人の口からでる湿気で反り上がるような素材でも UV やってみたいかも。

「私〇〇と申します——アッ落としちゃつた！！！」「アレ？なんかこれ文字浮かんでます？」「ああこの紙、濡れると変形するんですよ」「へえー！！」

2024.08.09

システム開発しないといけないのは分かってるんだけどどう作ればいいか分からず立ち往生。Claude に聞いてみたらプロトタイプ作

れって言われたので、プロトタイプ開発から始める。プロトタイプの動きは以下の流れになると思われる。

1. 巡回
2. 異常を発見
3. アピールを決定
4. アピールが無視された場合
5. ループ(異常が無くなるまで)
6. お礼、巡回に戻る

巡回

巡回を実装する。必要なのは以下

1. 問題なく動ける床
2. 屋内レイアウト
3. 異常オブジェクト
4. toio
5. 巡回プログラム

今日は床までできた。デフォルトの床だと狭いので広いのを自作したかったんだが、Unity のデフォルト床だとなんかうまく動かない。あれこれした結果、toio-sdk の Mat オブジェクトを床として使えばよい。Stage オブジェクトの中に無くても、Mat 単体で問題なく走行可能だった。

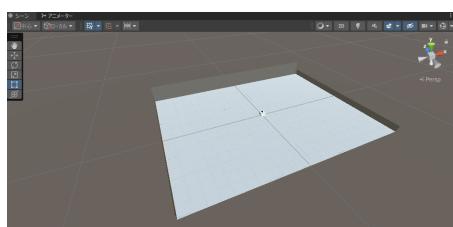


図 3 Mat オブジェクトを敷き詰めた図。問題なく走行できる。

次回は十分な広さをもたせた空間内で toio を巡回させられるようにしたい。

カメラ追従、壁、障害物、異常オブジェクトの配置、巡回プログラムの開発。この 3 つが巡回の残タスク。次回も頑張ろう。

2024.08.12

4 日ぶりの研究活動。

今日は前回の日記に書いてあった感じで、屋内レイアウトと異常オブジェクトを配置していく。屋内レイアウトは静的オブジェクトか、めちゃ重 Rigidbody でいくとして、異常オブジェクトはどうしようか？

こんな感じでどうだろう



図 4 内装モデルのスケッチ

異常オブジェクトは Unity 内だけなら Collision イベントで行けそう。じゃ、やってみよう。

衝突部分はできた。問題の巡回部分をアレコレ調べてた。

とりあえずまったく方針がわからぬので Claude 様に聞いてみたら、なんかすごくいい感じのプログラムを教えてくれた。動かしてみた結果が図 5。

ちょっと見えにくいけど、内装の各地点を結んだ五角形ができる。これにしたがって toio

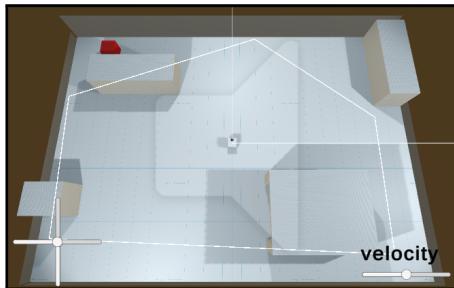


図 5 Claude 様に実装していただいた巡回プログラム。

が動くようになっている。

ただし現状では transform をじかにいじくったようなきっしょい動きになっているので、ここから toio を動かすプログラムとして書き直す。とはいってもこのレベルでイメージを掴ませてくれた Claude 様は本当に頭が上がらない。お金払おっかな……

方針 1: MoveTo で動かす

toio-sdk には MoveTo っていう機能があった覚えがある。座標を決めて、「ここに行ってください」っていうのを使う。巡回ルートのイメージは、Claude 様からもらったプログラムで掴んだし、似たようなことをすればいいと思う。懸念は MoveTo を現実空間でも使えるかどうか。なんかマットの上だけだった覚えがあるんだよなー……

title 方針 2: TargetPole を使う

Unity 限定機能「ターゲットポール」。これを使うとターゲットポールに向けて緩やかに走行させることができる。エージェントごとに巡回ルートとしてターゲットポールの座標を渡しておいて、タゲポ 1 についたら 1 を削除、2 のインスタンスを生成……みたいにできないだろうか。

Unity 上での巡回シミュレーションは作れる目処が立ってきた。今後残りそうな課題はこれ

をどうやって現実空間で利用していくか。

いま一番行きそうな方法は、エージェントのセンシングで昨日で空間情報を把握して、それを管理システムに送る。そして管理システム上で仮想の屋内空間を生成して、その中で MoveTo を使えるようにする。みたいな実装だろうか。

ともあれマット上とはいえど巡回ルートが作れそうだから、そのやり方で一旦形にしてみよう。

あとやること

- readme に todo リストをつける
- 巡回プログラム toio 版の実装

巡回できるようになったら、異常発見とかアピールとかの弱さの部分に移っていこう。更にいうとセンシングもどうするかまだ分かってないしな……

2024.08.16

toio の巡回システムをつくる。今日は巡回の動き部分。ドキュメント読んで確定したのは、toio.Move 関数じゃなくて、CubeHandle.Move を使いましょうということ。CubeHandle.Move はモーターの挙動とかの低レイヤーの動きを考えなくていいようなので、こっちのほうが考えたいことだけ考えることができる。音鳴らしとか L チカはまだ低レイヤーな気がするので、こっち側で関数を作ろう。

えーっと CubeHandle を使って Cube を動かすためにはまず CubeManager で繋がないといけなくって……

企画の再確認

かなりためになるインディーエロゲ開発者二キの話を聞き、企画書が大事だと感じたので、

自分の研究企画について再確認する。

企画名

ちっちゃくてかわいい家政ロボットで、生活を助けてもらう

コンセプト

小さくてカワイイキャラに生活を助けてもらいたい

ジャンル

ロボット、マイコン、IoT、ちびキャラ、マスコット

計画

8月: Unity 上でプロトタイプ

9月: 現実でプロトタイプ

10月: Unity 上で機能拡張

11月: 現実で機能拡張

12月: 実戦投入、データ収集

翌年1月: 資料整理、発表

売れそう？

かわいさ、ちんまりさ、いい意味でのダメさをうまく出せば売れる

効果測定

- 生活環境の場所ごとの改善タスクの実行回数が有意に増えるかどうか
- ロボットの魅力、すなわち「弱さ」が伝わっているか
- 「弱さ」が改善タスクの増加に寄与できているか

企画としては面白そう、というか頭のおかしい企画ではない。実現可能性については、計画の見積り不足がどれくらいかによる。それぞれのステップが実際にどれだけあるか分からぬとのと、おれの自制力が問題。自制力をアテにしている時点で駄目になりそうな感があるところが怖い。どちらかというと働き方改革？しかししながら見積りミスがおきた時点でどこをカットして次に進むかを考えないと後に後に詰め

ていくことは想像に固くない。PM の学びを活かせ。

2024.08.17

move2 系のステップどんな感じだ？まずはシンプルに座標入れて 1 回だけ move2。次に外から座標渡して 1 回 move2。次に複数座標渡して順番に move2。うん、この 3 ステップだな。

2024.08.27(火)

LoL ばかりやってるわけにもいかないので、10 日ぶりに研究を再開。

目下の課題は衝突検知が機能しないこと。Unity 上なら Collider の Collision イベントで衝突検知はできるんだが、現実ではそんなもの存在しないので、toio 備え付けの衝突センサで検知できるようにしたい。

サンプルコードをコピペしたもののうまいこと機能してくれない。

現実での衝突移動できた

できだ！

- toio 巡回の現実世界での実装

こまい diff は git 見ればわかるだろうけど、とりあえず AutoPatrol のプログラムを修正して、サンプルと同じ内容になるようにした。接続部分は CubeManager 版だからそっちを見たけど。

- CubeManager を用いた toio への接続
- 衝突センサ周りの技術仕様

まだ検知が不安定なので、もっと衝突検知のシステムを深ぼるか、別の検知機能で補うかしたいな。明日ゼミ行ったときに toio ばらまい

て動かしてみるか。

2024.09.09(月)

お久しぶりです。やること整理する。

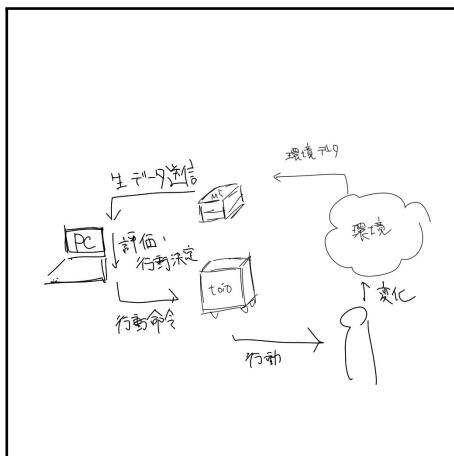


図 6 本システムのフロー

システム全体の構成は図 6 のようになる。
現状は toio の行動部分だけを作った。
次はセンシングして弱い行動の部分。センシングは M5Stick で、行動は toio で行う。まずはセンシングから。

M5Stick って何を検知できるんだっけ？

M5StickC 本体

加速度、ジャイロ

ENV.2

温度・湿度・気圧

ToF HAT

距離

んじゃあ気温系は ENV 使わないと無理だな。やってみよう

- M5StickC Plus で ENV II (環境センサユニット) を動作させる

本家に入っていた ENV のサンプルスケッチが動かなかったので、こっちの記事にあったコードを使用。動いた。

2024.09.18(水)

M5 に ENV2 取り付けて気温データ取得するところまで組みたい。なんだか周りの人間もだんだん動き出して来てるっぽい。もっとやすんでいいんだよ

BLE を使って通信できないかと調べてみたけど、難しそうなので一旦普通の Bluetooth シリアル通信で実装する。

2024.09.19(木)

エラー集を充実させたい。

今日の進捗はなかなかだぞ。お昼いっぱいやってただけあるな。M5StickC と ENV のデータ取得とシリアル通信での送信を実装した。ChatGPT を使ったらほんとすごい進捗がある。その上動くコード完成させるだけじゃなくて、クラス化したりファイル分けしたりで構成もだいぶきれいにできた。まじでおもう。次回は Unity 側での受け取りをしたいから、Arduino 側の printf 側で Unity 用の区切り文字追加するのと、Unity 側での Bluetooth シリアル通信の実装。このへんは昔取った杵柄って感じで、スムーズに行ってくれるとラクできるんだけどなあ。

2024.09.23(月)

M5StickC から Unity にシリアル通信でデータを送る。のだが、なぜか配列外参照エラーで止まる。今日はもう醉って作業どころではないので、寝て明日やる。

2024.09.24(火)

昨日クソ眠くてできなかつシリアルデータの送信を実装。

なんできてないんだろと思ったら imuInfo 側の配列参照をミスってた。6 個しかスロット無いのに 7 番目にアクセスしようとしてた。めちゃくちゃ便利だったのが下の 2 つ。

プログラム 1 Unity のエラー表示

```
1 // 具体的にどこでエラーが起きたのかを表示  
2 Debug.LogError(e.StackTrace);
```

前に佐藤先生がブレークポイント使うといいよって言ってたのに加えて、こういうエラー表示系の機能についても勉強しておくと、エラー処理がめっちゃ簡単になると思う。まああとは ChatGPT な。あいつまじで便利。

今日は M5StickC から Unity へのデータ送信を実装した。次回は評価リストと行動リストをつくる。可能なら「評価」の部分も実装できると偉い。

2024.09.26(木)

このロボットが一番映えるシーンやシチュエーションを考え、それを解決できるような特殊実装を目指してみたらいいんじゃないって言われたので、シチュエーションを考える。

基本的には気温取得系は向いてるっぽい。となると換気系のシナリオ。

2 台のロボットで換気行動を誘発

2 台の異なる役割を持ったロボットが存在する。

1 台目: 環境情報を取得するロボット

2 台目: 人間に呼びかけを行うロボット

環境情報を取得するロボットは、リビングとかに置く。

呼びかけるロボットはユーザがいる部屋に置く。

環境機が今いる場所の気温湿度が不快になつたら、環境機から呼びかけ機に通信。呼びかけ機がユーザに向けてアラートを出す。

ユーザは呼びかけに気づいて、リビングの方に行く。環境機が不快を訴えたり、リビングが不快であることに気づいてエアコンつけるなり窓開けるなりする。

人がその場にいるならその場でアラート出せばいいけど、ロボットを置いておくことで、少し離れた場所でも監視が可能になる。

テレワークや介護、子供の世話とかしてると便利そう。

2024.10.01(火)

弱くて小さいロボットが役に立つシーンを考える。

ベースは環境データの取得と遠隔での呼びかけ。通知もシステムチックなものじゃなくて、ロボットが動いたり鳴いたりすることで呼びかける。

お年寄りの介護

自分の親を介護しながら在宅で仕事している人。介護のときは当然同じ空間にいるけど、仕事や自分の時間のときはそれぞれ違う部屋にいる。

そういうときに、自分の部屋には呼びかけ機、親の部屋にはデータ取得機を置く。これによって離れていても何かあれば通知してくれるるので安心。

カメラはつけない。監視してるみたいだし。そういうんじゃないと思う。お互いの様子を見

れて、かつ、何か合ったときに教えてくれるペットがいるような感じにしたい。

データ取得機は、おじいちゃんおばあちゃんの部屋で環境データを取得し続ける。その間に、室温が高すぎるとか、じめじめしてるとかになったら鳴きだして警告する。おじいちゃん達が自分で動けるならこの時点で自分等でエアコンつけたり換気してもらってもいい。もしそういうのも難しいのであれば、ここで呼びかけ機に通信して、仕事してる若い人に通知して、様子を見に来てもらう。そしてエアコンつけたりして環境を整えてもらう。

主な利点としては、自分でいちいち気温を気にしなくていいところ。ロボットがつねに確認してくれるので、調整が必要になれば言ってくれる。何かあれば呼ぶので、ロボットを置いておくだけで、おじいちゃんたちに気負うこと無く、自分の作業をこなすことができる。おじいちゃんたちを監視する機械を設置するのではなく、何かあったときに介護の人を読んでくれるペットを置くようなイメージなので、介護される側も窮屈に感じない。また置くだけなので場所に制約されない。介護の人と介護される人がいる部屋それぞれに置けばいいだけ。

2024.10.02(水)

今日は機能をつくります。なんの機能？環境データ関連の機能。あとダミーデータ生成機も作りたい。

2024.10.05(土)

評価システムの設計をした。ChatGPT と相談の末、結構安心して作れそうな設計と実装例ができたので、次回はこれを実装していくたい。

やる前の適当な予想だけど、評価ロジックと実際の挙動の実装が多分一番面倒くさい。多分マジで面倒臭いからその辺うまいこと ChatGPT に投げつつ作ろうかな。

あと設計の過程で UML 作ったよ。tasklog と同じ階層に置いたはずだからちょくちょく見てね。

2024.10.07(月)

テストがしたいので Unity Test Runner を使おうとしたけど、テストがうまくできなかつた。でもできるようになったら自作クラスの単体テストとか効率的にできそうなので、やりかたは覚えたい。

toio-sdk のテストか何かが噛んでるっぽいのでそれを解決したら治るんじゃないですかね。邪魔やねん。もうダルいので今日は絵を描く。

2024.10.08(火)

今日は Unity Test Runner の使い方を覚えたい。

動かせるところまではいった。エラーは直らない。

テストが上手くいかないのは toio-sdk のせいではないということは分かったのでそれで許す。おれが悪いよ。

次回はテストやってエラー解決方法見つけたりなんだりできるところまでいくとうれしい。

2024.10.10(木)

今日もシステム作ってた。対象とする環境データを指定しながら、その上で評価クラスと行動決定クラスを自動で決める構造が作れなかやつたけど、上手くいかないのでくっ

そシンプルに作り直した。なんかよさそう。
ChatGPT もいいんじゃねって言ってくたし。

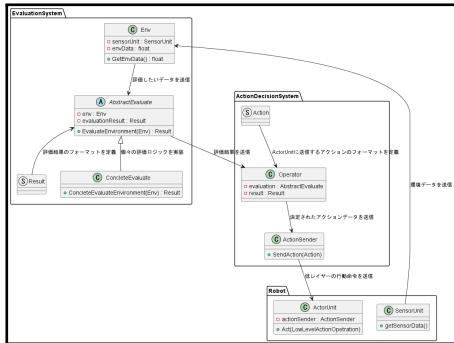


図 7

2024.10.12(土)

この前一応完成した設計を実装する。実装する途中で色々困ることになるとは思うが、適宜解決しつつ進めていきたい。

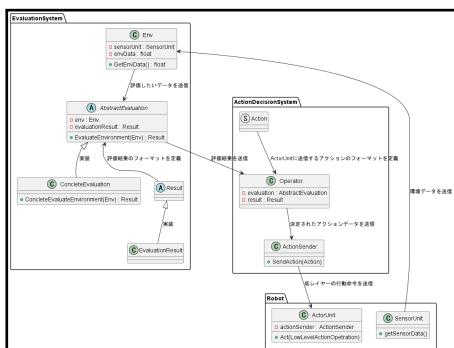


図 8 評価システムの設計 第 2 版

設計はできたから、あとはどうしたら評価ができるか。

現在の気温、もちろん要る。基準となる気温、これはどこから取ってくる?気象庁とか、オフィスワークにめっちゃ詳しい団体が情報を公開してくれていると助かる。閾値、これは基準となる気温からどれくらい離れているかを測る。「暑い」「寒い」「快適」の3つがいいだろう。

初めて自作クラスのテスト通るところまでい
けた！結構嬉しい。

toio-sdk の問題か分からぬけど、Editor 起動して一番はじめのテストは PlayMode で toio-sdk のテストを実行しないとテストがちゃんと動かなくなるっぽい。通るテストできたし、今度検証してみよう。

Unity でスクリプトつくるうえで結構大事そうなことも分かった。すなわち、「Monobehavior の継承は Unity と連結したいスクリプトだけ行う」ということ。

多分オブジェクト指向とかが多少わからない
とそもそも継承とかを使わないので中級者向けの考え方ではあると思うけど、自作クラスは無継承で作って、Unity 上に配置するところに自作クラスとゲームオブジェクトをつなぐ Monobehavior 継承クラスを挟む。まさにインターフェース。

多少調整は必要だけど、Unity 上で 100% オブジェクト指向設計ができるようになるこの作り方は絶対必要になると思う。

デバッグ用につくってるダミーデータ生成器のデバッグ表示機能がおかしいので、次回はそこを修正しながら、EvalutionSystem 全体のテストとか、さらに ActionDecisionSystem の実装も進めていきたい。

2024.10.16(水)

エラー表示は機能なんとかしました。ログ書き忘れてたけど、前回(15日?)の研究でエラー表示直して、エラー判定とかせずに挙動確認するだけのテスト作った。

今日は ActionDecisionSystem の開発に着手しました。Action が Result によるということで、Result 側で抽象を完全に切ってしまう必要が生じた。いま Action も Abstract で組もうと

してたけど、そんな密結合のクラスやめたほうがいいって。Result で切れるなら切ったほうがいい。

というわけで、次回は Action に渡す Result のフォーマットを決めて、Result の具象クラスでそのフォーマットで全部渡すように組み直して、みたいな作業を進めていければと思います。

2024.10.17(木)

今日からは ActionDecisionSystem をメインに作っていく。

のだけれど、その前に Result を ActionDecisionSystem に渡す部分について考えなければいけない。具体的には、Result のフォーマットを制限せずに ActionDecisionSystem に渡すのか、Result のインターフェースを定めて、ActionDecisionSystem 側からシンプルにアクセスできるようにするのか。

でもそういう 2 項対立だったら、ポリモーフィズムできるんだから絶対にインターフェースを定める方式にしたほうがいい。

となると問題になるのは、Action を決めるために、Result に何があれば十分か、ということ。かつ Result インタフェースが持つものは、個々の対象に依存した概念ではなく、Result 一般が持つものである。

バット思いつくのは、はじめに Condition。これは絶対にいる。でも、現状の Condition の形だとデータの内容がばらつき過ぎて駄目だ。気温は HOT,COLD,SUITABLE だとして、じゃあごみがあったり湿度が高かったりしたときも同じようにできるわけじゃない。共通化するか、括り方が間違っているか。なんとなく括り方が間違っている気がする。評価とそこから派生する行動は切り離せないので？ 正確には評

価と派生する行動の基準のようなもの。暑いときや寒いときやゴミがあるときの行動は取らない。のであれば、やっぱり環境評価と行動決定は密結合でいい。

これや！

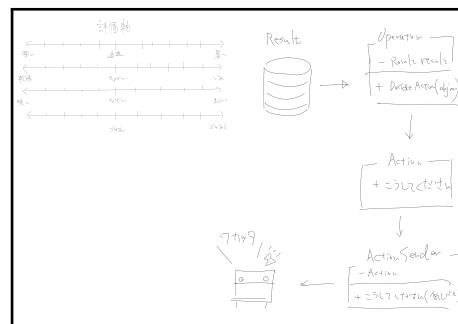


図 9 評価データの扱いおよび Result のフォーマット、それをどう Action 以降に送るか

今日はサンワ棚をお迎えするために終わる。

次回は Result に入れるデータフォーマットおよびそれを入れる部分の作り直し。設計の修正。あといろいろやっておいてくれ。

2024.10.21(月)

Result の内容を前回決まった数値判定式に変えた。厚生省のドキュメントによると 18°C 28°C が適温らしいから、その範囲に従って採点するように作り変えた。次回は DummyDataGenerator と合わせて採点が動くところまで確認した

2024.10.22(火)

ダミーデータを生成して Evaluation で評価する部分を作った。コルーチンの使い方もなんとなく覚えた。問題なく数値は流れていたので、次回は Action の決定部分を作っていく。

2024.10.23(水)

データ渡す部分ができたので、行動決定の部分を作っていく。いまの構想としては点数化した Result を ActionOperator に渡し、点数にしたがってアクションを決める。非常に単純。

ダミーデータでの挙動チェックは前回やったが、実際のセンサユニットを使った挙動チェックはやっていなかったので、そこもやっておきたい。チェックが終わったら行動決定の設計をする。時間があれば実装まで着手。

SensorUnit がアタッチできない。なので付け替え実装というより、M5 とダミーのメインループを行うコンポーネントを作って、都度アクティブにするような運用。

SerialHandler をオブジェクト指向に取り込むためには、SerialHandler をシングルトンにする。んでもって Unity のシステムを一切使わせないようにする。関数はそのまま、SerialHandler を使うスクリプトで呼び出させる。んでもって、COM ポートとかボーレートはエディタで設定できた方が便利だから、ScriptableObject か Monobehavior としてつくる。

センサー部分をポリモーフィズム使用したら思ったより難しくて中断。でも設計はしたので、面倒だけど次回からやっていく。設計は図 10 の通り。

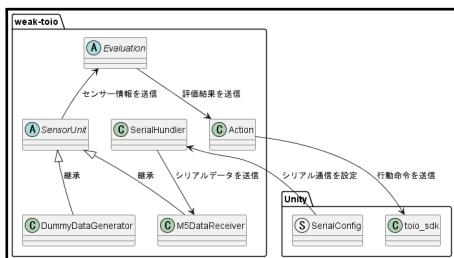


図 10

2024.10.27(日)

センサーとダミーデータを使い分けられる構造に作り変えて、センサーからの情報がちゃんと届いてることが確認できた。

でもおれの体幹よりも気温高めだったので、センサーとは相性が悪いかもしれない。

気温データは適温の範囲から外れるほど、正負のどちらかに傾くスコアを持たせたので、アクション決定ではこれをを利用してアクションの強度や種類を決めるようにしたい。Abstract でつくるのは大変なので、とりあえず気温から汚く作っていく。

アクションのアイデアを図 11 に示す。

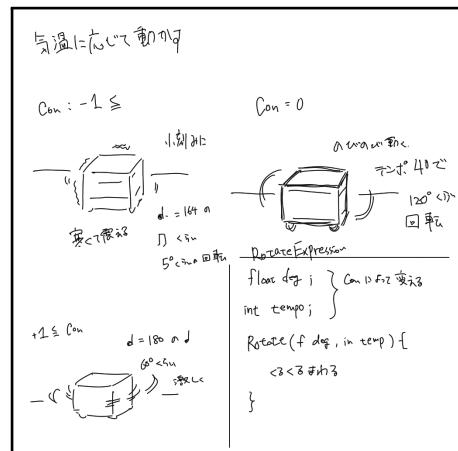


図 11

パラメータは「角度」と「回転のテンポ」の2つ。

うん、すっげえ汚えけど、きれいな設計思いつかないから後で考えよう。

それはそうとアクションのデータフォーマットはどうする。Result で入手できるのはスコア。じゃスコア。あとロボットのアクションの強度を決めるのに必要なパラメータ。

そういえば最終的にシステムとして動かす場合、メインループは1つでいい。並列処理とかしないなら。なので Monobehavior 系のスクリプトは1つあればよい。ほかは基本的に Unity が無くとも動くように調整できるはずだし。

そっち側のメインループスクリプトをものっつっそい荒いけど書いてみた。多分こういう風に動いてくれるよねっていう10割イメージで。

明日は AutoPatrolあたりを見返して、ロボット動かすメソッドと、コンディションに応じてメソッドを自動で選択する方法を考える。先に寒い・適温・暑いの3パターンの関数を作ってから考える方が時間のムダが無くていいでしょう。

2024.10.28(月)

付点四分 = 164

八分 = 552

$$\begin{aligned} \text{四分} &= 276 [\text{bpm}] \\ &= \frac{276}{60} [\text{bps}] \\ &= 4.6 [\text{bps}] \end{aligned}$$

1回回るのに必要な時間は

$$1 [\text{s}] / 4.6 [\text{bps}] = 0.217 [\text{秒}]$$

メインループで動かしてみた。んだけど、これといって何も動かない。toio が無いから?でもデバッグ表示では M5 もちゃんと動いてるみたいだし、評価もできるよう。次回ダミーデータ版のメインループつくりつつ、toio 動くところまで確認できないかな。

2024.10.30(水)

RotateByDeg 系の関数は RotateSpeed > 15 のときのみ機能する。

全く思いつかない。キャラ追加みたいにモーション追加がしたいけど、モーションデータの扱いがわからない。Json?モーションのデータセットを別で持って、その中身をシステムが全部読みこむ感じ?じゃあ Json の Read がいるじゃないですか。

震えるモーションを作っただけではだめなので、今の時点で拡張性を持たせなければならない。理想を言えばモーションはリストみたいに所持できて、必要に応じて任意のモーションを選んで実行するようにしたい。

モーションをキューで管理する。イベントに応じて対応するモーションクラスを拡張した具象モーションインスタンスを生成して、モーション実行クラスのキューに追加する。

モーションクラスは任意のモーションおよびその時間を持っていて、モーション実行クラスはモーションインスタンスに入っているモーションを時間分実行する。実行されたモーションは pop され、次のモーションクラスが実行される。

2024.10.31(木)

全然設計が思いつかないので、今考えていることを書きなぐる。

システム全体は 4 つのパッケージで構成される。

1. エージェント部: ロボットおよびセンサー ユニットを制御
2. 評価部: センサー情報を取得して環境を

評価

3. 行動発行部: 評価結果をもとに行動命令を発行
4. 行動実行部: 発行された行動命令を逐次処理

エージェント、評価は完成していて、行動実行もイメージは湧いてる。行動発行部の設計が思いつかない。

こういうときは逆に、評価部から出せるものと、行動実行部が欲しいものを先に決めればいい。入口出口が決まっているなら通路も比較的決めやすい。

評価部で新しく追加されるデータは環境の「評価スコア」。現状は正負どちらにも傾きすぎればよくないとして、0 を適正值としてスコアを返す。その他センサー情報もセンサユニットから送られたものはそのまま流すことができる。

行動実行部では、送られてきた行動命令を逐次実行する。そのため、toio-sdk 用の行動命令のキューがほしい。キューを pop で処理していけば、逐次実行は実装可能だと思う。実行は Update じゃなくてコルーチンにすれば命令のオーバーフローも抑えられると思う。

では具体的にどのような行動命令であれば、上のことが実現できるか？

まず考えられるのは、行動の具体的なパラメータ。toio-sdk ではマットを使用しないアクションとして、前後に移動する Translate、向きを変える RotateByDeg/Rad、その他音を鳴らしたり LEDをつけたりさせることができる。

となると、させたい行動のタイプによって必要なパラメータが異なるので、各行動用の構造体が必要になる、また行動のタイプを決定するために、行動タイプをまとめた enum も必要になる。

MotionOperator で行動を実行するから、MotionGenerator ではモーションのレシピを発行するようなイメージ。接客して、欲しいもの聞いて、それをキッチンに連絡する。

明細発行、というと、なんだか昔似たようなクラス設計を目にした気がする……

ひとまず今日は時間になったのでここで終わる。次回も書き殴りから初めて、必要なものを整理していこう。

行動実行部でいるもの

- 行動タイプ: enum
- 行動パラメータ: タイプによって異なるパラメータ

2024.11.01(金)

行動タイプは evaluate が発行して、result を通して渡す。

行動パラメータは、行動タイプに応じて、対応する構造体を生成して、その中のフィールドをいじくることで変化させる。パラメータ変化には result の score を用いる。

行動タイプとパラメータ構造体は 1 対 1 で対応付けられている。

行動決定部では、対象とする環境データごとに区切りを決めて、区切りごとのモーションを発行する。

TA やってたら「概要設計」っていう話が出てきた。システムの振る舞いを関数、クラス、コンポーネント単位で記述する。仕様に書いたことを実装するべきだし、仕様にないことは実装するべきではない。おれのシステムの概要設計ってどうなるだろうか？

2024.11.18(月)

前回までは、センサープログラムを修正した。Action を作っていく。

TemperatureAction にいろいろアクション生成っぽい関数があるので、これを実装していく。

どういうアクションがそれっぽいかな？ 寒い方は震える。暑い方は？ わからないので回転させる。

ベースに Cold、Hot それぞれにアクションの方針があり、Danger 圈内になつたら露骨にする。

コルーチンがだんだん分かってきた。フェーズ式のモーション実行なんかはコルーチンにすればいいける。今日は検証の時間って感じだったので、次回次々回くらいには動きの一つでも実装してみたい。

2024.11.19(火)

Movement は他の CubeHandler から作ったものでも使い回すことができる。

ActionGenerator から作った Action を ActionSender に流す。コルーチンを使う場合、ActionGenerator では IEnumarator を生成する。Sender では IEnumarator を StartCoroutine する。

アクション命令送信回りのデータの流れが上手くいっていないので、次回は終端からデータの流れを見直す。これメモ。

2024.11.20(水)

依存性注入を使うと、アクション生成が上手いことできそう。Claude に聞いた。アクショ

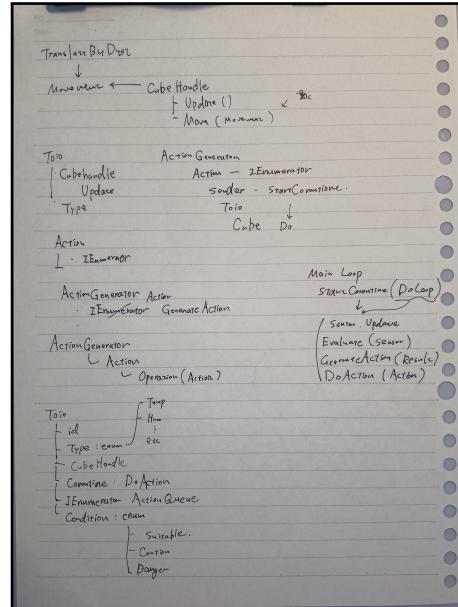


図 12

ン生成の関数は CubeHandle を持つてるクラスのインターフェースに宣言させる。実装はインターフェースを継承させた Toio あたりのクラスにやらせる。

2024.11.21(木)

DI ハマって、昨日の夜眠れなくて DI 使った設計にしたらめちゃくちゃイカした設計になつた。リファクタリングはともかく、さっそく Action の実装に使おうと思う。

2024.11.23(土)

DI で Action 部分を設計し直したら結局全部作り直すことになった。作り直しはできたが、async とか task がよくわからないために初期化でエラーを吐かれる。いったん今日はやめにして、次回 toio の接続からインスタンスの登録を安定化する。

2024.11.27(水)

今日もつまんなく思われるかと思ったが、意外と会話できた。

身体でわかるることを通知したところで「そんなわかるよ」になるので、身体には分からないことを教えてくれる子にしたほうがいいよねって話。「〇〇の気持ち」シリーズつくれ。

非機能要件の話ばかりしてしまう。機能の話苦手なんかな。機能の話できるようになりたい。

1.4 〇〇の気持ち

- ねこの気持ち
- 草の気持ち
- 本の気持ち
- 部屋の気持ち
- 自分以外の人の気持ち

2024.11.28(木)

バリエーションを作る前に、キューブの動作方式を Handle から Cube に変えたい。いや、Handle の動きに対する強さはそのままに、Cube で使える便利な機能も使えるようにしたい。

拡張性も考えると、動作はデリゲートにして、動作の主体になる Cube、CubeHandle と動作命令を引数にする。

¹ Toio.Move(CubeHandle, Movement)

² Toio.PlaySound(Cube, SoundOperation)

³ Toio.LED(Cube, LightOperation)

修正の方針を固めた?ので、明日は実装していく。今日中に実装できるかと思ったが、まあいい。

2024.12.01(日)

複数台接続を実装した。メインループで実装するのはアイデアが出なかったので、toio への接続をメインループから切り出して、ToioConnector で得られた toio のデータをメインループに持っていくもらうようにした。キューが相変わらず役に立って、取得したキューブを Dequeue することでキューブの重複を防げているのは自慢ポイント。

複数台接続を実装したこと、「役割別の接続できるようにしたい」という課題も見つかった。次回は「役割」の概念と、役割別の接続機能を実装したい。

前回の日記で Cube と CubeHandle のいいとこ取りをしたいって話をしたけど、多分それも実装できた。これも多分 DI で、Toio の動きを IToioCommand っていうインターフェースに包むことで、「実行」っていう機能を抽象化した。一応気温アクション生成に使ってはいるけど、これといって大きな変化は見られない。多分これが加賀焼くのはアクションをデザインしていくフェーズなので、それまではくつろいでいてもらおう。

2024.12.02(月)

役割理論を考えた結果、役割の実装はできそう。ただ、現状のインターフェースや抽象クラス、メインループといった基幹系クラスが軒並み「M5StickC+ENV2 センサーでデータを取得する」という前提に依存しているという問題が発覚した。このままだと役割機能が実装できないので、根本の実装を変える。アイデアは相変わらずのインターフェース。 M5 とか ENV とかを全部インターフェースにする。それに伴っ

て、気温判定をするやつは Temperature、距離判定するやつは ToF とか、それぞれのインターフェースを求めるようになる。こういう根本からの作り直しがだんだん頭おかしくなってくるから、じっくり焦らず、着実にやろう。

2024.12.07(土)

役割分けを実装した。はじめの懸念どおり、スクリプトをアタッチしても、アタッチした toio に正しく接続してくれるわけではない。考えとしては、

1. M5 認証
2. 1 個ずつ認証
3. 多分接続の優先度が親機からの距離に関係しているので、近いものから順に

みたいな感じ。どうすっかなー……距離比例だったら M5 から接続するのが一番丸い。位置情報とかは持っていないし、データとして使うのも難しそうなので使わない。可能なら M5 経由して toio に接続。そうするとデータの流れが全く変わるんだけどね。

おそらく

M5 -> PC -> 判定 -> アクション生成 -> アクション送信 -> M5 -> アクション命令 -> toio

みたいな導線？

イニシャライズは

PC -> M5 -> toio

Arduino で toio を制御する前例はいくつかあったので、それを参考に組みなおしてみようか

toio を ESP32(Arduino 環境) で動かす

エラー集

Adafruit BMP280 not found

ENV 用のサンプルスケッチを使うには別途「Adafruit BMP280 Library」のインストールが必要。Arduino IDE のライブラリマネージャーから DL したら解決。**プログラムなんも間違ってないのに**

Arduino IDE のコンパイルに失敗する

M5StickC のボードは M5Stack と ESP32 っていう 2 つのグループから出てるんだけども、なにかアップデートがあったようで、周辺ライブラリ全体が M5Stack 版の M5StickC ボードに合わせて構成されるようになった。

なので ESP32 を使うとデータ合わなくてライブラリと競合する。間違えるのが心配なら ESP32 版は消して、M5Stack 版だけを残すとよい。

ENV 使ってみたいのにサンプルプログラムが動いてくれない！

どうやら ENV2 から中身の基盤が SHT3x っていう規格のものに変わったようで、サンプルの DHT12 クラスではアクセスができない。

この記事のプログラムだと行けなくもないけど、また最近変数とかメソッド名が変わったようで、書き換えが必要。まあ SHT3x に続けてドット打てばヒントは出てくるので、temp とか humid とか打てば該当のものは出てくる。

SHT3x 入れたのに ENV2 が機能しない

そのライブラリ、SHT3”X” じゃありませんか？こっちでも機能する可能性はあるけど、上のリンクで使っているのは 3X じゃなくて 3"x" です。紛らわしいんじゃ。

SHT3"x" をちゃんと入れたのに機能しない

確証があるわけじゃないけど、bme の初期化の仕方に問題があるかも。

はじめは setup で if(bme 初期化できてないならエラー出力) って感じだったんだけど、それを while(初期化完了するまで) にしたらうまく行った。もしかすると env は sht3x と bme がどっちもちゃんと初期化できないと機能しないのかもしれない。

参考文献

- [1] 岡田美智男. ゴミ箱ロボット—関係論的なロボットの目指すもの. 計測と制御, Vol. 51, No. 8, pp. 753–758, 2012.
- [2] Yuto Yamaji, Taisuke Miyake, Yuta Yoshiike, P. Ravindra S. De Silva, and Michio Okada. STB: Human-dependent sociable trash box. In *Proceedings of the 5th ACM/IEEE International Conference on Human-robot Interaction*, HRI '10, pp. 197–198, Osaka, Japan, March 2010. IEEE Press.
- [3] C. Breazeal. Social interactions in HRI: The robot view. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, Vol. 34, No. 2, pp. 181–186, May 2004.
- [4] Tweenbots - Kacie Kinzer. <http://www.tweenbots.com/>.
- [5] Tangible Media Group. HERMITS, October 2020.