appLift：イスからはじまるコミュニケーション

チーム名：kbylab
エントリー番号：11

[Abstract]

要旨

言いたいことがある・・・でもなかなか言えずにいる、そんな経験ありませんか？そんな気持ちを汲み取って、大勢の人の前でも発言ができるように行動を促してくれるイス、それが「appLift」です。appLiftは人間のストレス反応である貧乏ゆすりと、「自分のとった行動に対して理由を後付けする」という性質（認知的不協和理論）を応用して、「言いたいことが言えない」という問題を解決するプロダクトです。

[Problem]

どんな課題に取り組むのか？

コミュニケーションが「一対多」となる場において、「言いたいことが言えない」という悩みを解決します。

[Solution]

課題をどのように解決するか?

例えば、「授業」というシチュエーションで考えてみたいと思います。

ある生徒A君が「appLift」というイスを導入した学校で授業を受けているとします。しかし、勉強のあまり得意でないA君は授業の途中から、先生の話している内容が理解できずイライラし始めてしまうという状況を思い浮かべてみてください。質問したい気持ちはあったのですが、みんなの前で発言するのは恥ずかしいという思いから手をあげることができませんでした。その結果、イライラは次第に大きくなっていきA君はやがて貧乏ゆすりをし始めてしまいました。次第に周囲のクラスメートも貧乏ゆすりに気づきますが、授業中なのでなかなか言えません。これによりイライラする人が拡散していき教室に良くない雰囲気が漂ってしまったのです。

このような状況で、A君が座っているイス「appLift」はどうするかというと、A君の足が少し地面から離れるように自動的にイスを上昇させるのです。すると、A君は驚きつつも立ち上がった状態に近くなり、自然と貧乏ゆすりは止まります。

さらに、突然動いたA君に対し先生は声をかけます。思わず立ち上がってしまったA君は、無意識に「質問があります」と答えるのです。そこで先生はもう一度わからなかった点を解説します。こうしてA君はappLiftによって、貧乏ゆすりが止まっただけでなく、「質問したいけどできない」というイライラの原因も解消することができました。

このように人間のストレス反応である「貧乏ゆすり」に着目し、その止め方を工夫することで「一対多」の場でも自然に発言できるように行動を促そうというのがappLiftのコンセプトです。

なぜこの解決策なのか

さて、授業中に急に貧乏ゆすりをしていた人が急に立ち上がるというシュールな方法で本当に問題解決ができるのか、と疑問に思った方も多いかもしれません。しかしappLiftは、社会心理学者フェスティンガーが提唱した「人間は、自分のとった行動に対して理由を後付けする」という性質（認知的不協和理論）を応用して、ターゲットに問題解決へとつながる行動を取ってもらえるように設計しました。例えば、授業のシチュエーションであれば「恥ずかしいから発言したくない」という考えを持っている人に対してappLiftで半ば強制的に「起立」という行動（発言するときにとる行為）を促すことになります。すると、ターゲットは自分の「考え」と自分のとった「行動」との間に矛盾が生じ、認知における不協和を感じるのです。こういった場合、人間は自分の「考え」を変化させることで不協和を解消しようとします。つまり、appLiftで行動を先行させてしまうことで、ターゲットは「質問があったから立った」といったような理由を無意識で後付けして、結果的に言いたかったことが伝えられるのではないか、と考えています。

参考：Festinger, L. (1957). *A Theory of Cognitive Dissonance.* Stanford, CA: Stanford University Press.

課題を解決することでどのような価値が生まれるか?

① ストレス解消
   a. イライラして貧乏ゆすりをしてしまった人が周りに迷惑をかけて、さらに自己嫌悪に陥るループから脱却できる

② コミュニケーションの活性化
   a. 生徒と教員が対話しながら進める双方型の授業が実現できる
   b. 上司、部下関係なく発言ができるようになりフラットな会議を実現できる

どんなテクノロジーを使って解決策を実現するのか?

以下、フロント・バック・ハードの視点からの実現方法になります。

フロントエンド概要

実装予定の機能

① 椅子のステータス変化時の室内の様子をタグ、コメントによって記録する

実装予定の画面

① HOME画面
   a. 椅子のステータスの表示、レベル変化時の室内の様子を示すタグ、コメントの入力画面を表示する。
② LIST画面
   a. ①で入力するタグの追加、削除、編集を行う画面を表示する。
③ DATA画面
   a. 何月何日の何時頃にレベルの変化が起きたのか、レベル変化時の室内の様子を示すタグ、コメントにはどのようなことが書かれているのか確認を行う画面を表示する。

イメージ図

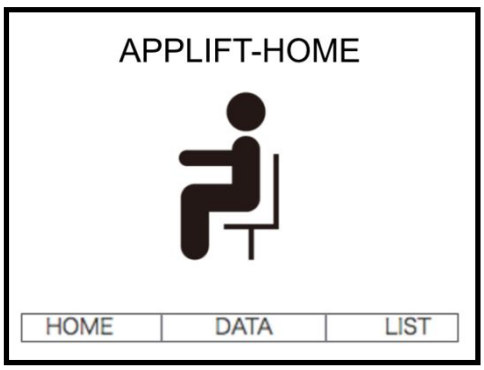


図１：HOME画面

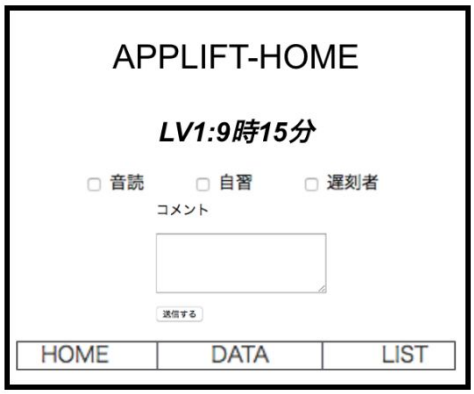


図２：レベル変化時の画面

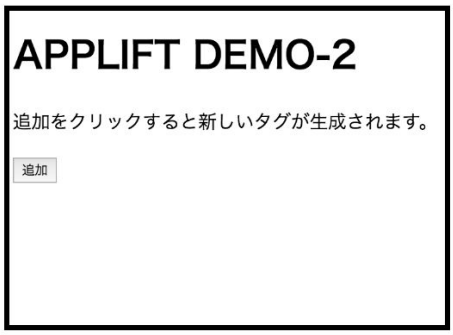


図３：LIST画面

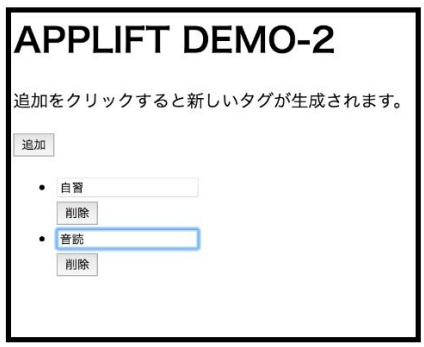


図４：タグ編集画面

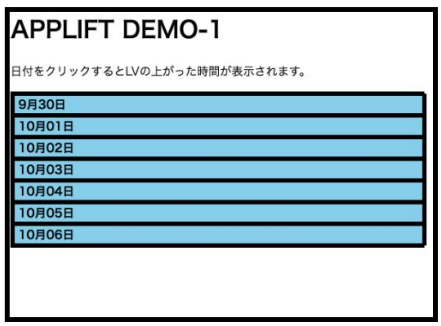


図５：DATA画面

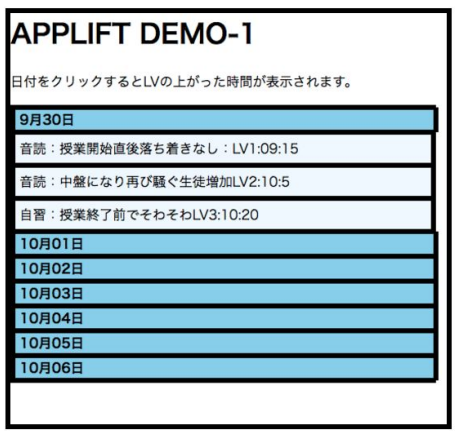


図６：DATA詳細表示画面

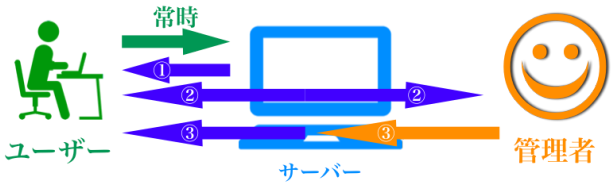
バックエンド概要

バックエンドが行うことは、フロントとハード側から得られた情報をもとに状況を分析・評価と評価値をもとにフロント、ハード側に指示を出す通信を行う。

使用する言語

python3

実装予定の機能



ユーザの揺れ具合を常に送信
① 一定以上の揺れを検知したら椅子を振動させる
② さらに一定以上の揺れを検知したら
椅子をリフトアップし、管理者に通知する
③ 管理者からの指示を受け椅子をシフトダウン

① 常時評価・微弱振動指示

ハードとサーバーは常にWi-Fiで接続されており、一定時間ごとにサーバーに振動の強さを送信する。やり取りするデータはJSON形式のもので、分析やその他のプログラムへの転送を容易にできる。これらは全てPython3の標準関数で実装される。

振動の強さに基づいて下記の4つのレベルに状態を分類する。

レベル0： 静止し揺れを検知していない状態。
レベル1： 通常の着座状態で起こりうる軽度の揺れを検知している状態。
レベル2： 貧乏ゆすりレベルの大きな揺れを検知している状態。
レベル3： 微弱振動指示後においてレベル2の揺れを検知している状態。

ハードから送信された整数値から、揺れレベルをレベル0-2のいずれかに分類する。レベル2が数秒間継続した場合、椅子の微弱振動指示を行う。振動時間は2、3秒ほどとし、振動中は常時評価を行わない。

② 上昇指示

①で微弱振動を起こしても、貧乏ゆすりが継続される場合、レベル3へ移行する。ハード側に椅子の上昇指示を出し、フロントエンド側で操作している管理者に椅子を上昇させたことを通知する。
フロントエンドとの通信はWebSocketを用いて行い、データ形式はJSONを用いる。上昇した椅子のIDをメッセージとしてフロント側へ通知する。

フロントエンドへの必要情報

・レベルが上がった時間の記録（バック側で記録管理）、フロントからの要求時いつでも返せるようにするymdhms方式。

③ 管理者からの椅子の下降指示

管理者が椅子下をげても良いと判断した場合、フロントエンド側のアプリから椅子の下降指示を行う。このとき、管理者ーサーバ間の通信はWebSocketを用いて行い、下降させる椅子のIDをメッセージとしてサーバ側に送られる。そして、サーバから椅子側へ下降指示を出す。

ハードウェア概要

使用する機材

・Raspberry Pi2 B++（椅子の制御及び通信）
・電動シリンダ （座面の上昇下降の制御）
・サーボ（座面のシリンダの制御）
・加速度センサー（貧乏ゆすりの計測）
・バッテリー

① 貧乏ゆすりの発生と強度の評価

a. 発生の判定について
    i. 椅子の座面に加速度センサーを設置し、加速度の大きさや持続時間で判定

b. 評価値の出力
    i. Wi-Fiを介してint型で送信

② 椅子の動作

a. 椅子の上昇・下降
    i. バックエンド側から上昇指示を受信すると、サーボによって椅子のレバーが引かれ（図８）電動シリンダを利用して座面を上昇・下降させる（図９）
b. 命令の受信
    i. Wi-Fiを介してint型で受信

③ 安全面

a. 高さの制限
    i. 座面を規定値以上（足が地面から離れて数cm程）には上がらないようにする

④ 動作確認

a. ハードウェアとソフトウェアのステータス照合を秒間一回行う

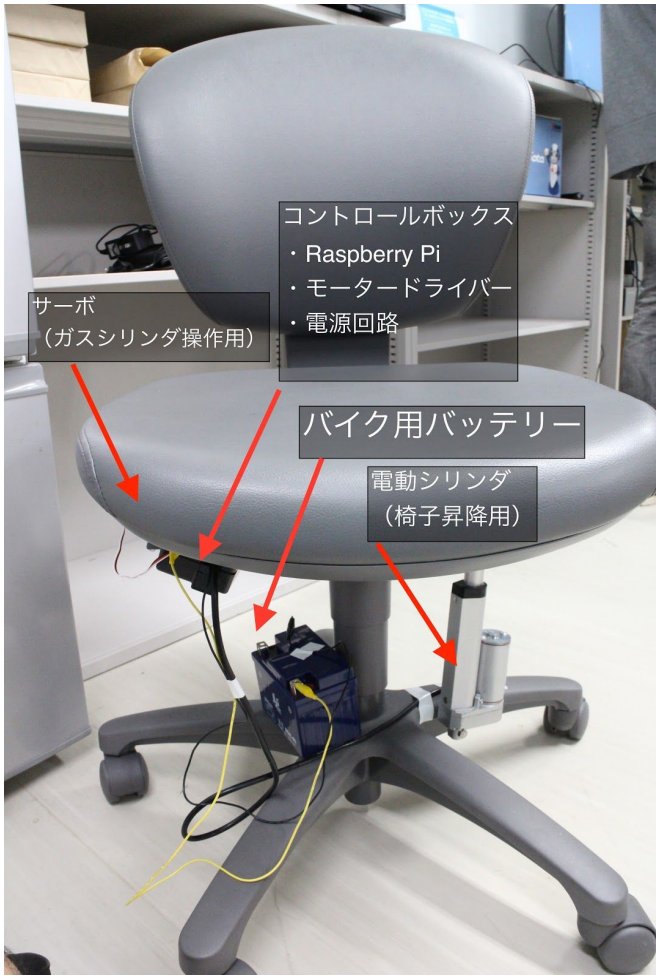


図７：試作機

図８：座面上下時に動作するシリンダ

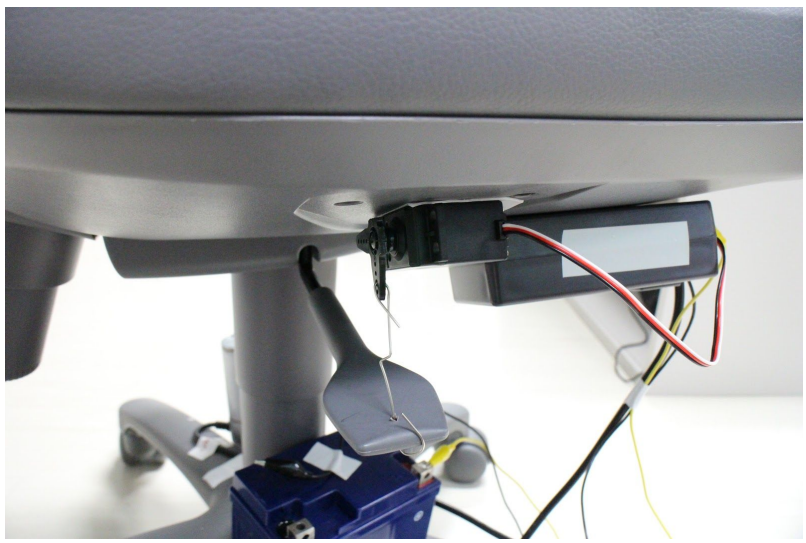


図９：椅子の上下時に動作するサーボ

[Conclusion]

appLiftで社会を**uplift**する未来を目指します。