TheareFourze:舞台演出手法を取り入れた CG とセンサ連携を実現する Processing 言語向けライブラリの提案

樫村 健人[†1]　　菅谷 みどり[†1]

概要：近年マイコン Arduino と連携しやすい言語および総合開発環境である Processing が注目されている．これらを使うとセンサと CG の連携アプリケーションを書くことができるが，それについての抽象モデルや開発環境は十分といえるものがなく，初心者にとっては難しいものとなっている．TheatreFourze は CG とセンサの連携を舞台演出手法を取り入れ解釈しやすくしたライブラリである．本論文で提案するライブラリの機能，実例，および行った評価実験の考察を述べる．

TheatreFourze: the Proposal of Library of Processing for Sensors and CG, which Accepts the Model of Theatrical Stage Production

KASHIMURA KENTO[†1]　SUGAYA MIDORI[†1]

Abstract: Recently, total development environment Processing, which is known to be easy to connect with microcomputer Arduino, is gathering attention. Using Arduino and Processing, we can make an application of sensor and CG, but there is not good abstract model or enough development environment, and it is difficult for beginners to make such an application. TheatreFourze is a library which accepts the model of theatrical stage production for connection of sensor and CG.
In this paper, we state the function of library, some example, and examination of the experimentation.

1. はじめに

近年，マイコンの一つである Arduino[1]の普及により，センサを用いたハードウエアデバイスと，コンピュータとを連携させ，様々なインタラクティブデバイスが開発されている. また，演習授業などにも取り込まれ[2]，教育用の教材としても利用されている．Arduino は，マイコンを用いた開発の敷居を下げ，ハードウエアを扱いやすくする一方で，既存のホストコンピュータ上のソフトウエアで制御が可能である．そのため，ホスト側のコンピュータと連携した様々な表現が可能となっている[1]．こうした中，我々はマイコンから取り込んだセンサ情報を, コンピュータ上のコンピュータグラフィックス（以降，CG）と連携させる環境として Arduino と Processing[3] を用いた新しいデジタルデバイスを提案した[4]. マイコンを用いて様々なセンサ入力を CG に応用することは，現実世界の情報と，コンピュータの中の仮想情報である CG による表現の多様化につながり，新しいインタラクションデバイス開発の可能性を拡大すると期待できる.

しかし, マイコン上のセンサ入力を CG に対応づけて扱うためには, センサ入力ごとのイベントを CG に対応づけるためのコードをその都度記述する必要がある．特にプログラムを書く経験が浅い場合，コードが複雑化しやすく，可読性の低下や，コードの再利用性が低くなる問題がある..

本来であれば，Processing は，キーボードやマウスなどの入力イベントを抽象化し, 扱いやすくしていることから，こうした問題は軽減されるべきであるが，多様なセンサからの入力イベントを利用しようとした場合，イベントと CG の関連が分かりづらくなる問題がある.

本研究では，問題を解決するために，プログラミング経験が比較的浅い開発者にも分かりやすいメタファを与え，それを用いた開発のためのライブラリを提供することにより，センサから得られる実世界の情報と CG によるプログラミングをしやすくすることを目的とした Processing 言語上で動作するライブラリを提案する．本提案では，提案するモデルとそれに基づくライブラリである TheatreFourze の設計とその評価について述べる．本論文の構成は以下の通りとなる．まず 2 節にて，センサと CG の連携アプリケーションの開発における課題をまとめ，3 節にて提案を行う．4 節にて具体的な設計を述べ, 5 節にて実験と評価を示し, 6 節で全体のまとめを述べる.

2. 課題

2.1 既存研究

センサを用いた開発環境として，様々なセンサプラット

†1 芝浦工業大学 工学部
Shibaura institute of Technology

フォーム(センシングプラットフォーム) が提案されている[5][6]. FreeMotion が提供するセンサプラットフォームは[5] センサを搭載したハードウエア(多くの場合スマートフォン)と開発支援のための API やドライバのセットとなっている．つまり，開発者が選んだセンサを用いることはできないため，多様なセンサ入力を扱うための開発を行う際には不向きである．

また，Processing[3]は Java をベースにし，図形描画に特化した言語および総合開発環境である．Processing の特徴として Java のグラフィックライブラリなどと比較して，イベントを待ち受けるイベントリスナのサーバをプログラマが記述する必要がない．複数のスレッドを切り替えて，イベント処理をするためには，マルチスレッド処理を意識する必要があるが，こうした部分は Processing では隠蔽されており，イベント処理を初心者でも簡単に扱うことができる利点がある．しかし Processing は，イベントを扱うマウスの位置座標の取得が mouseX, mouseY といった短い記述で行えることでマウスの状態を扱うことは簡易化されているが，多様なセンサ入力を標準的に扱うための方法は明示されておらず,開発者がその都度,センサ入力と，CG の対応づけを書かなくてはならない問題があり，これにより，センサ入力を用いたプログラミングの難易度があがってしまう問題については，具体的な解決策は提示されていない．

2.2 問題

このように,多様なセンサ入力を可能とする Processing 言語の開発環境でも，センサ入力と CG との連携機能を有していないため，開発者は連携部分のプログラムを自分で書く必要がある．

表１　センサ入力を扱うコードの一部

```
hl =   arduino.digitalRead(inputA8);
sun = arduino.analogRead(input1);

if (hl==1) lightsw=1;
if (hl==0&&lightsw==1) {
    if (houselight==1)houselight=0;
    else houselight=1;
    lightsw=0;
}
if (houselight==0) {
    if (sun>150)   light=0;
    else light=1;
}
```

具体的例をあげて説明する．我々が開発したアプリケーションにて[3], 光センサとタクトスイッチを用いて CG の明るさを変える処理を行ったコードを表１に示した．処理を記述するために，常にマイコンの入力値を変数に多応させ,それに対応する CG の動作を記述する必要があるため，12 行で 5 つの if が出現するという非常に分かりづらいコードとなる問題が発生した．こうした問題は，コードの可読性の低下や再利用性の低下につながる．

このように，if 文を大量に記述しなくてはならなくなる理由として，値域が異なるセンサを複数使用し，それに合わせて CG を変える処理の記述をするには分岐やケース文を多用しなくてはならないことが要因の一つと考えられる．こうしたセンサ値を利用したイベント処理の記述は，初心者にとって難しいものとなってしまうことが多い．Processing ではキーボードやマウスのイベントは抽象化されているが，センサ入力と CG を組み合わせた処理の汎用化は，まだ達成されていない．

3. ライブラリの提供

3.1 研究の目的

本研究では，上記の問題を解決するために，まず，多様なセンサ入力を一般化し，CG を連携させるためのクラスを設計し，そのクラス内でのメソッドを分かり易く定義すること，この一連のフレームワークをライブラリとして提供することで，問題を解決するものとした．

センサと CG を組み合わせについては，その連携を扱うための抽象モデルを検討した結果，舞台演出というメタファを取り入れたクラス設計を行うものとした．開発者は，単純な対応づけ，ということだけではなく，概念的な統合に関するイメージを持つことで，連携部分を記述しやすくすることを目指した．これは初心者のセンサと CG の連携アプリケーションの理解を助け，ライブラリにより短いコードでの連携実現およびコーディングにかかる時間の短縮を目的としている．

3.2 舞台演出メタファを取り入れたクラス設計

本研究の基本アイディアとして舞台演出のメタファを取り入れたクラスを設計する. まず, 舞台演出について述べる．舞台演出においては，演出家が役者をもとに舞台を構成する．舞台では，役者ごとに，演技ときっかけを個別に演出する．これに対して，CG とセンサの連携アプリケーション開発では，センサからの入力イベントに応じて，個別の振る舞いをする CG がある．

入力に応じて，振る舞いを変更する，という部分は，キューに応じて，演技を変更する舞台での役者の振る舞いに類似していることから，センサによって振る舞いを変更する CG を役者とみなすものとした．役者の振る舞いを演技，入力されるセンサイベントを舞台におけるきっかけ(Cue)と考えて，全体の舞台を構成すると考えることで，連携アプリケーションの開発を行うことができないかと考えた．つまり開発者が，あたかも演出家であるかのように役者(CG)ときっかけ(センサイベント)を組み合わせることで初心者であっても連携のイメージをしやすくすることを目指した．

4. ライブラリの設計

4.1 ライブラリの構成

舞台演出手法を取り入れた開発環境は TheatreFourze と名付けた．ここで，Theatre とは，劇場を意味し，舞台演出モチーフにプログラミングを行うものとした．舞台演出概念（図 1）．

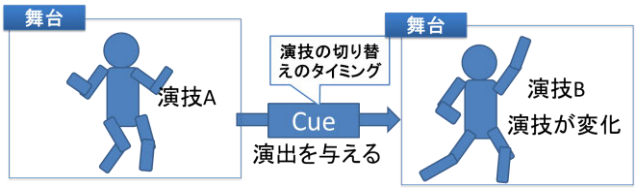

図 1 舞台演出概念の導入．

ライブラリでは役者，演技，きっかけをそれぞれクラスとして扱うものとした．CG は，画面上に目にみえて振る舞いを行うために，役者である Actor クラスとした．また，その役者の振る舞いを Act クラスとした．センサからの入力は（イベント）は，舞台上で役者の振る舞いを変更するためのきっかけ（Cue）と考えることができることから，Cue クラスとし，これら 3 つのクラスで舞台を構成するものとした．

Actor クラスは画像のロードと表示, Act クラスは画像の X 位置，Y 位置の変更，そして Cue クラスはセンサの値を適切にマッピングする機能とした．対応づけを複雑化しないようにするため Cue クラスでは一つのセンサ値のみを指定するものとした． 一般的に，扱うセンサの種類により取得できる値の範囲は異なり，それがセンサ値を扱うことが難しい要因の一つとなっている．従来は，開発者はその都度異なる値を利用していたが, 本提案では，下記の式(1) を用いることで，これらの値の範囲を一定に扱うことができるようにした．ここで, Val は，現在のセンサの値, Max はセンサの最大値，Min は最小値，Num は区分値である．ここで区分値とは, 大小の 2 段階や, 大中小の 3 段階など，分かり易い区分とする．

なお本ライブラリで指定しているセンサの場合，最大値と最小値をライブラリの内部に設定する，しかし，それ以外の場合，最大値と最小値の設定が必要となる．

$$(Val - Min) * Num / Max - Min \quad \ldots(1)$$

各クラスの関係図を，図 2 に示した．。Actor クラスと Cue クラスに識別番号のやりとりを行う。Cue クラスはセンサ値をマッピングしたものを識別番号として Actor クラスに送る。Actor クラスは Act クラスに対して演技(CG の動き)の指定を行い Act クラスは Actor クラスに対して役者の X, Y 位置を変更して返す処理を行う．

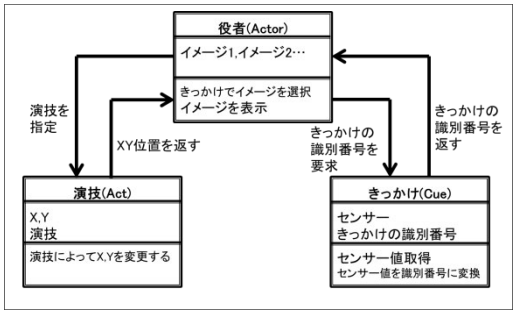

図 2　Actor, Act, Cue の 3 つのクラスの関係図

4.2 記述例

Processing には最初に一度だけ実行する setup()とその後ループで実行され続ける draw()の 2 つの関数がある．

TheatreFourze では setup()で役者の宣言，CG の指定，演技の指定，きっかけの指定を行う．以上の処理を以下に記述した．

表 2　記述例

```
1: Actor actor;
2: actor=new Actor(this, arduino);
3: actor.setImage("kashi.png");
4: actor.setAct(Act.Right, Act.Left");
5: actor.setCue(Cue.Light, 0, 1);
```

図 3　記述例 1 での役者（CG）の移動

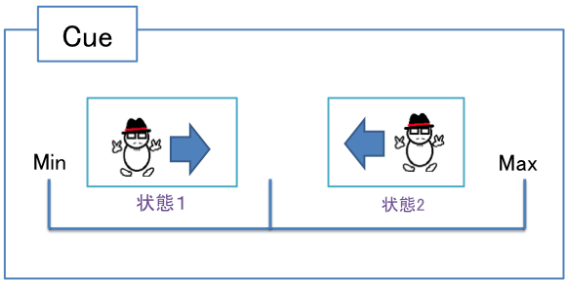

図 4　役者の演技をきっかけ(Cue）で切り替える

1,2 行目で示した Actor の宣言は一般的な変数宣言と同じように new で生成を行う．3 行目の CG の指定は setImage メソッドで行い，引数は String 型でフルパスとファイル名である．なお，Processing のアプリケーションフォルダ直下もしくは data フォルダに格納されている場合はファイル名のみの指定で画像がロードできる．4 行目

で示す演技の指定である setAct メソッドは複数の引数をとり，内部で定義している Act クラスに演技としてセットする．これにより，CG が動くようになる．そして 5 行目の setCue メソッドではセンサ名，int 型でセンサが接続されているポート，さらに演技を区分分けする数を Cue クラスにセットする．この 5 行の記述により，「Arduino のアナログポート 0 に接続されている光センサの値によって CG"kashi.png"が左右に動く」という状態を作成する．

draw()では Actor クラスのメソッドである Actor.draw()を記述するのみである．ライブラリを使わずにセンサと CG の連携をしようとした場合，draw 関数内で，Arduino クラスのメソッドでセンサのアナログ値を取得，取得したアナログ値をマッピング，マッピングした値によって処理の決定など複数のコードを記述しなくてはならないが，ライブラリを導入することで処理をライブラリ内部で行うため明確に簡易化できる．

5. 評価

提案手法による効果の評価として，本研究では，主に 2 つの評価を行った．一つは，プログラムの記述量コード量の削減効果，二つ目は，ユーザビリティについての調査を行った．

5.1 コード量の削減

まず我々は，提案するライブラリの導入によるコード量の削減効果を評価した．具体的には，センサと CG の連携によるサービスを行うプログラム対象とし，ライブラリを導入/導入なし，の 2 つのコードを比較した．プログラムは同等の機能を持つものとした．アプリケーションは「橙色の屋根のお家」[4]と「かしむらガチャ」という 2 つのプログラムを用いた.「橙色の屋根のお家」はドールハウス型のデバイスに搭載されたセンサで現実の情報を取得し、ディスプレイに表示されている CG に反映させるというアプリケーションである.「かしむらガチャ」はカプセルトイを模して可変抵抗器をハンドル代わりに用いるジョークプログラムである．両者を書き換えた結果は以下の表 1 にまとめた．

表 3 ライブラリを導入し書き換えた際のプログラムの行数

	橙色の屋根の家	かしむらガチャ
ライブラリ無し	247	148
ライブラリ有り	75	63

「橙色の屋根のお家」では 247 行だったプログラムがライブラリを導入することで 75 行まで削減できた. また「かしむらガチャ」は 148 行だったプログラムが 63 行に削減できた．いずれのプログラムも元の半分以下の行数で同等のプログラムを再現出来た．プログラムのコード量はコメントアウトや個人の癖など様々な要因により増減するが，それを加味しても有用性を示す結果であると考える．

5.2 初心者によるライブラリ利用の効果

次に，実際にセンサと CG を連携させるプログラムの開発にまだ慣れていない初心者に対して，本ライブラリを実際に利用してもらい，その効果測定を行った．

効果の測定では，ライブラリの有無で作りやすさに差があるか (1) 開発の達成度 (2) 開発時間を調査し，次に(3)アンケートで印象を確かめることとした．被験者は本学学生 14 名とした. なお被験者は, Processing および Arduino による実習講義の受講者[2]とし, Processing による CG 製作, Arduino を用いたセンサの基礎的な実習は既に終わった段階で，簡単な CG とセンサの連携アプリケーションを作ったこともあるがプログラミング経験は比較的浅い開発者であるとした．14 人は， 2 人で 1 つの班を組んでいることから，計 7 つの班を対象とした調査を行った．調査では，提案したライブラリ有りの場合，無しの場合，の両方の開発環境で同じアプリケーションを開発してもらい，開発にかかった差を確認した．

開発内容は，Arduino 側のセンサとして，可変抵抗器を用いて任意の CG を上下左右に移動させる単純なプログラムとした．ライブラリが有る場合と，無い場合の実行順序に差が出る場合を考慮して，先にライブラリが有る開発環境で開発を行う手順 A のグループと，先にライブラリが無い開発環境で行うグループと手順 B で行うグループに分けて実施した. これを表 4 に示した. 両グループには実施後，アンケートに回答してもらった．なお，制限時間は 12 分とし，その時間内に作成できなかったかどうかを明確にできるようにした．

表 4　グループごとの手順

	実験(一回目)	実験(二回目)	対象
手順 A	ライブラリ有り	ライブラリ無し	3 つの班
手順 B	ライブラリ無し	ライブラリ有り	4 つの班

5.3 初心者に対する利用効率

(1) 開発の達成度の比較

センサと CG を連携させるプログラム課題を完成できたか否かの結果を表 5 にまとめた．

表 5 完成した班の数の比較

	完成した班	完成しなかった班
ライブラリ有り	手順 A　2 班/3 班 手順 B　4 班/4 班	--
ライブラリ無し	--	手順 A　0 班/3 班 手順 B　2 班/4 班

手順 A で実験を行ったグループでは，ライブラリ有りでは 3 つの班のうち 2 つの班がプログラムを完成したしできたが，無しの状態では 1 つの班も完成しなかった．残りの 1 つの班はライブラリの有無に関わらず完成できなかった．手順 B で行ったグループは 4 つの班のうち 2 つの班はライ

ブラリ無しではプログラムを完成できなかったが，ライブラリ有りではできた．残りの 2 つの班はライブラリの有無に関わらず完成できた．総合するとライブラリ有りでは 7 つの班のうち 6 つの班が完成したが，ライブラリ無しでは 2 つの班しか完成しなかった．

この事から，ライブラリ導入効果は明らかであったが，特に先に，ライブラリ無しの状態で試行した班の方は，ライブラリ導入効果が高い結果となった．

(2) 開発にかかった時間

(1) にて，ライブラリを用いて開発した際に，完成した班については，開発にかかった時間を調査した．以下に示す表 6, 表 7 は手順 A および手順 B の各班のプログラムの完成までにかかった時間をまとめた．

表 6　手順 A におけるプログラム完成までにかかった時間

手順 A	ライブラリ有り	ライブラリ無し
A-1	6 分 50 秒	-
A-2	-	-
A-3	6 分 20 秒	-

表 7　手順 B におけるプログラム完成までにかかった時間

手順 B	ライブラリ無し	ライブラリ有り
B-1	11 分 50 秒	5 分 15 秒
B-2	-	3 分 40 秒
B-3	-	6 分 15 秒
B-4	10 分 20 秒	8 分

表 6 に示したように A-1，A-3 はライブラリ有りの状態ではプログラムを完成出来たが，ライブラリ無しではできなかった．表 7 の B-2，B-3 も同じようにライブラリ有りの状態でしかプログラムを完成できなかった．統合すると 7 つの班のうち 4 つの班がライブラリなしではプログラムを完成できなかったということになる，ライブラリの有無に関わらず完成できた B-1，B-4 についてもライブラリを導入することで完成までの時間が短くなっていることがわかる．

(3) 利用調査アンケートの結果

さらに，調査の対象となった受講生に，実施後アンケートを行い，その結果を図 5,6,7 にまとめた．アンケートでは図 5，図 7 において手順によって差異が見られた．図 5 における差異は，手順 A では 3 つの班のうち 2 つの班がライブラリ有りの状態でしかプログラムを完成させられなかったことにより，ライブラリへの依存度が手順 B の 4 つの班よりも高く，より必要性に駆られたため，「とてもそう思う」という意見が多く見られたことによる．また図 7 における差異は被験者の人数が少ないことによる誤差であると思われる．

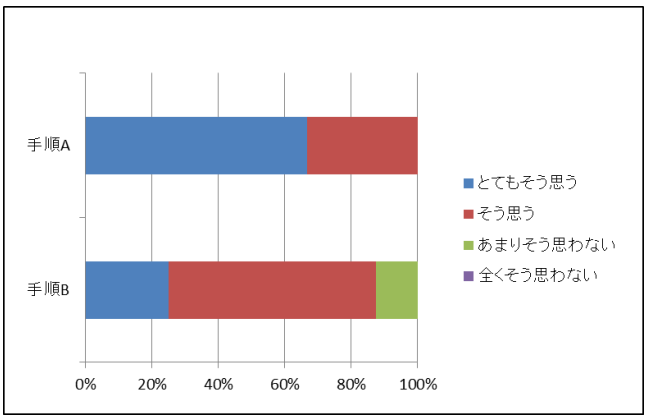

図 5　ライブラリを使いやすいと感じたか

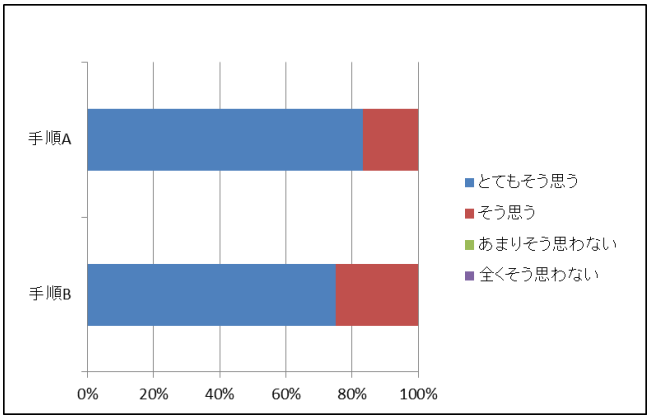

図 6　ライブラリを用いてコードを書くのが楽か

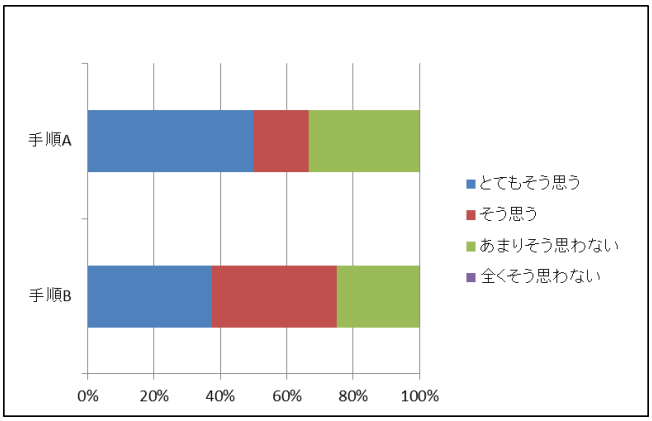

図 7　初心者にとって扱いやすいと思う

図 6 に示した「ライブラリを用いてコードを書くのが楽か」という問いに対して 14 人全員が「とてもそう思う」「そう思う」と回答している．これによりライブラリの有用性が示された．しかし図 5 に示した「初心者にとって扱いやすいと思うか」という問いに対しては被験者の約 3 分の 1 が「あまりそう思わない」と回答しており，提案する抽象モデルが初心者にとって扱いやすいかは調査の必要があり，改良の余地がある．今回は舞台演出をモデルのメタファとして用いたが，他のメタファの場合を用いない場合との比

較が必要である．このことは今後の課題とする．

6. まとめ

CG とセンサの連携アプリケーションを書く際に，舞台演出手法を取り入れた抽象モデルおよびそのモデルを元に作ったライブラリ TheatreFourze を使うことで，プログラムが書きやすくなるかの評価を行った．今後は，開発したプラットフォームの評価を進め，その有効性を明確にしたい．また舞台演出以外のメタファによるモデルなどの調査も進めていく必要がある．

参考文献

1) Arduino http://www.arduino.cc/
2) 菅谷みどり, Arduino とセンサーを教材とした HCI デバイス製作演習の実践, インタラクション 2014 論文集, pp.221-223(2014)
3) Processing https://www.processing.org/
4) 渡辺 柚佳子,岡田 佳子,大澤 博隆, 情報処理学会研究報告. UBI, 2014-UBI-43, pp.1-6 (2014).
5) FreeMotion, SensorPlatform http://www.sensorplatforms.com/
6) 安藤拓生,高梨千賀子,兵藤友博,徳田昭雄,八重樫文,守政毅,後藤智, セイコーエプソン社における要素技術を用いた新市場開拓,『立命館経営学』第 52 巻第 2・3 号 (2013).