実験項目	実験 C3　ナチュラルユーザインタフェース
校名　科名 学年　番号	熊本高等専門学校　　人間情報システム工学科 3 年　　42 号
氏名	山口惺司
班名　回数	4 班　　1 回目
実験年月日 建物　部屋名	2023 年　11 月　30 日　木曜　天候 晴 2023 年　12 月　7 日　木曜　天候 晴 3 号棟　　2 階　HIPC 室
共同実験者名	

1. 本実験の目的

本実験では以下の事柄について学ぶ。

（ア）ナチュラルユーザインタフェースの概要や概念について知ること。

（イ）Leap Motion を使ったプログラムを作ること。

（ウ）Leap Motion に出来ること・出来ないことを把握すること。

2. 実験内容

両手の動きを追跡できる Leap Motion を用いて、リアルタイムに手の各関節座標を取得し、プログラムに実装する。

3. 実験課題

1. 課題１　両手ハンドルを回す動作の実装

課題文：

両手でハンドルを握って回転させるような動きを取得し、ハンドルの CG が回るようなプ ログラムを作れ。さらにハンドルの角度に対応して自動車等が移動するような動きがあると良い。

ソースコード：

```
import de.voidplus.leapmotion.*;

LeapMotion leap;

PImage handle, car;

//--------------------------------------
void setup() {
  size(1600, 900, P3D);
  leap = new LeapMotion(this);
  handle = loadImage("handle.png");
  car = loadImage("car.png");
  imageMode(CENTER);
}

float speed = 2;
float s = 0;
float y = height;
```

```
float car_x = width/2;
//-------------------------------------
void draw() {
  background(167, 211, 152);
  ArrayList<Hand> hands = leap.getHands();
  int handNum = hands.size();      // 現在認識している手の数を得る

  drawRoad();
  image(handle, width/2, height/2+200);
  image(car, width/2, height/2, 84, 120);

  if (handNum <= 1){
    return;        // もし手を検知していなければここで終了
  }
  background(167, 211, 152);
  Hand h0 = hands.get(0);  // 0 番目の手の情報を得る
  Hand h1 = hands.get(1);
  h0.draw();  // 手を描く
  h1.draw();
  Finger f0;
  Finger f1;
  f0 = h0.getFinger(0);              // 手の 0 番目の指(親指)の情報を得る
  PVector pos0_1 = f0.getPositionOfJointTip();  // つま先の座標を得る
  PVector pos0_2 = f0.getPositionOfJointDip();  // 第 1 関節の座標を得る
  PVector pos0_3 = f0.getPositionOfJointPip();  // 第 2 関節の座標を得る
  PVector pos0_4 = f0.getPositionOfJointMcp();  // 第 3 関節の座標を得る

  f1 = h1.getFinger(0);              // 手の 0 番目の指(親指)の情報を得る
  PVector pos1_1 = f1.getPositionOfJointTip();  // つま先の座標を得る
  PVector pos1_2 = f1.getPositionOfJointDip();  // 第 1 関節の座標を得る
  PVector pos1_3 = f1.getPositionOfJointPip();  // 第 2 関節の座標を得る
  PVector pos1_4 = f1.getPositionOfJointMcp();  // 第 3 関節の座標を得る

  drawEllipse(pos0_1);  // つま先の位置に円を描く
  drawEllipse(pos0_2);  // 第 1 関節の位置に円を描く
  drawEllipse(pos0_3);  // 第 2 関節の位置に円を描く
  drawEllipse(pos0_4);  // 第 3 関節の位置に円を描く

  drawEllipse(pos1_1);  // つま先の位置に円を描く
  drawEllipse(pos1_2);  // 第 1 関節の位置に円を描く
  drawEllipse(pos1_3);  // 第 2 関節の位置に円を描く
  drawEllipse(pos1_4);  // 第 3 関節の位置に円を描く
```

```
  Finger naka_0 = h0.getFinger(2);
  PVector v3_0 = naka_0.getPositionOfJointMcp();
  Finger naka_1 = h1.getFinger(2);
  PVector v3_1 = naka_1.getPositionOfJointMcp();
  if(v3_0.x <= width/2 && v3_1.x >= width/2){
    s = v3_1.y - height/2 - 200;
    car_x -= (height/2+200 - v3_1.y)/75;
  } else if(v3_0.x >= width/2 && v3_1.x <= width/2){
    s = v3_0.y - height/2 - 200;
    car_x += (height/2+200 - v3_1.y)/75;
  }



  drawRoad();
  pushMatrix();
  translate(width/2, height/2, 0);
  image(car, car_x, 0, 84, 120);
  popMatrix();
  pushMatrix();
  translate(width/2, height/2+200, 1);
  rotateZ(s/100);
  image(handle, 0, 0);
  popMatrix();


}

//--------------------------------------
void drawEllipse(PVector pos){
  fill(0, 100);
  pushMatrix();
  translate(pos.x, pos.y, pos.z);
  ellipse(0, 0, 30, 30);
  popMatrix();
}

void drawRoad(){
  fill(150);
  noStroke();
  rect(width/2-200, 0, 400, height);
```

```
  y += speed;
  for(int i = 0; i<10000; i++){
    fill(255);
    rect(width/2-7, y-100*i, 14, 30);
  }
}
```

実行例：

実行例を図 1, 2, 3 に示す。

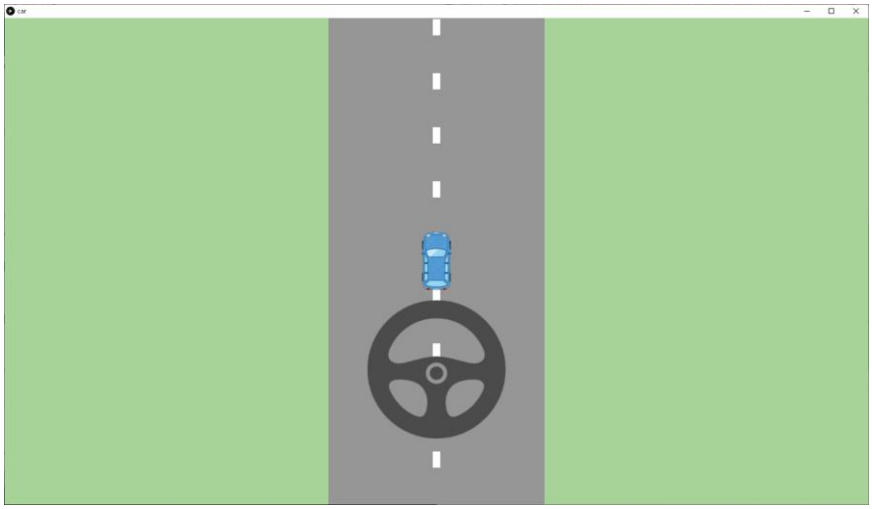

図 1 起動時の実行例

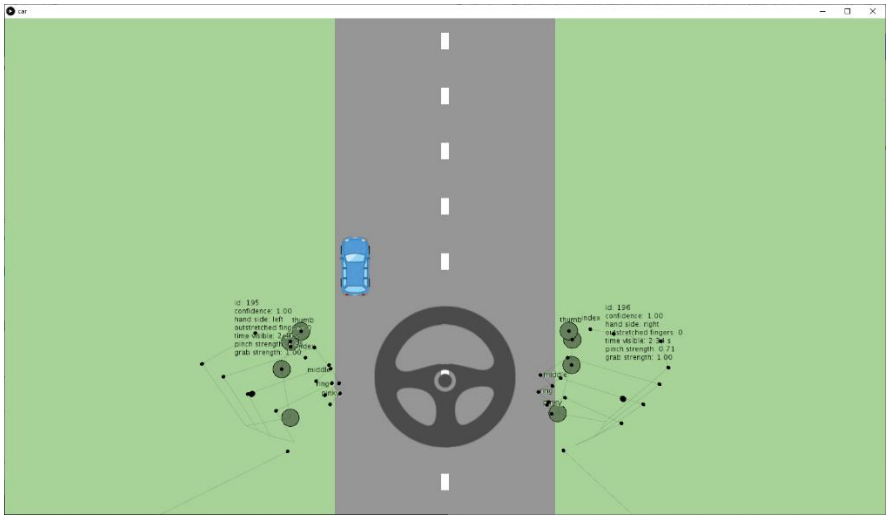

図 2 手を表示したときの実行例

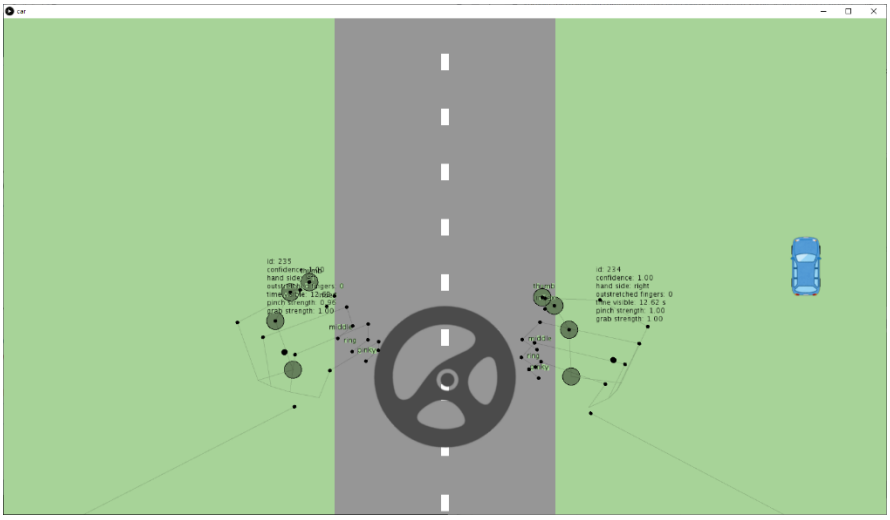

図 3 ハンドルを傾けたときの実行例

工夫した点：

・車が進んでいることを表現するために道路を動かした。

・ハンドルの傾きを角度ではなく、中指の y 座標で変えるようにし、操作を簡単にした。

2. 課題２　非接触型メニュー選択機能の実装

課題文：

コロナ禍による感染対策の一つとして、食堂の券売機などでなるべく手を触れずに注文などの手続きを行うインタフェースが求められている。 そこで食堂などの店舗、銀行窓口、自動販売機など、客が何らかのものを発注するようなケースを想定し、 その際の発注作業を非接触で行うシステムを作れ。このとき、選択メニューの確定やキャンセルなどが行える ようにし、その操作がユーザの自然な動きで実現できるよう、設計・調整すること

ソースコード：

```
import de.voidplus.leapmotion.*;

LeapMotion leap;
PImage curry, katsudon, udon;
PFont font;
int flag = 0;
float z = 0;
int mode = 0;

//--------------------------------------
void setup() {
  size(800, 500, P3D);
```

```
  leap = new LeapMotion(this);
  curry = loadImage("curry.png");
  katsudon = loadImage("katsudon.png");
  udon = loadImage("udon.png");
  imageMode(CENTER);
  textAlign(CENTER, CENTER);
  font = loadFont("CambriaMath-48.vlw");
  textFont(font);
}

//--------------------------------------
void draw() {
  switch(mode) {
  case 0:

    background(255);

    drawButton0();

    ArrayList<Hand> hands = leap.getHands();
    int handNum = hands.size();      // 現在認識している手の数を得る
    if (handNum <= 0) return;        // もし手を検知していなければここで終了

    Hand h = hands.get(0);           // 0 番目の手の情報を得る
    h.draw();                        // 手を描く
    Finger f;
    f = h.getFinger(1);              // 手の 0 番目の指(親指)の情報を得る
    PVector pos1 = f.getPositionOfJointTip();  // つま先の座標を得る

    drawButton1(pos1);
    break;
  case 1:
    result();
    break;
  }
}

//--------------------------------------
void drawEllipse(PVector pos) {
  fill(0, 100);
  pushMatrix();
  translate(pos.x, pos.y, pos.z);
```

```
  ellipse(0, 0, 30, 30);
  popMatrix();
}

void drawButton0() {
  fill(0);
  textSize(50);
  text("Choose Someone", width/2, height/2+150);
  strokeWeight(5);
  pushMatrix();
  translate(width/2, height/2, 1);
  fill(255);
  ellipse(-250, 0, 200, 200);
  fill(255);
  image(curry, -250, 0, 100, 70);
  ellipse(0, 0, 200, 200);
  image(katsudon, 0, 0, 100, 100);
  ellipse(250, 0, 200, 200);
  image(udon, 250, 0, 100, 100);
  popMatrix();
  strokeWeight(1);
  textSize(10);
}

void drawButton1(PVector pos1) {
  if (flag == 0) {
    z = pos1.z;
  }
  textSize(50);
  pushMatrix();
  translate(width/2, height/2, 1);
  fill(255);
  strokeWeight(5);

  float curry_d = dist(pos1.x, pos1.y, width/2-250, height/2);
  if (curry_d <= 100) {
    judge(pos1);
  }
  ellipse(-250, 0, 200, 200);
  fill(255);
  image(curry, -250, 0, 100, 70);
```

```
  float katsudon_d = dist(pos1.x, pos1.y, width/2, height/2);
  if (katsudon_d <= 100) {
    judge(pos1);
  }
  ellipse(0, 0, 200, 200);
  fill(255);
  image(katsudon, 0, 0, 100, 100);

  float udon_d = dist(pos1.x, pos1.y, width/2+250, height/2);
  if (udon_d <= 100) {
    judge(pos1);
  }

  if (curry_d > 100 && katsudon_d > 100 && udon_d > 100) {
    flag = 0;
  }
  ellipse(250, 0, 200, 200);
  fill(255);
  image(udon, 250, 0, 100, 100);
  fill(0);
  popMatrix();
  strokeWeight(1);
}

void judge(PVector pos1) {
  if (flag == 0) {
    z = pos1.z;
    flag = 1;
  }
  if (flag == 1 && pos1.z - z >= 15) {
    mode = 1;
  }
  fill(0);
  text("Press to Buy", 0, -150);
  fill(255, 255, 0);
}

void result() {
  background(255);
  textSize(100);
  fill(0);
  text("Thank You!!", width/2, height/2);
```

}

実行例：

実行例を図 4, 5, 6, 7 に示す。

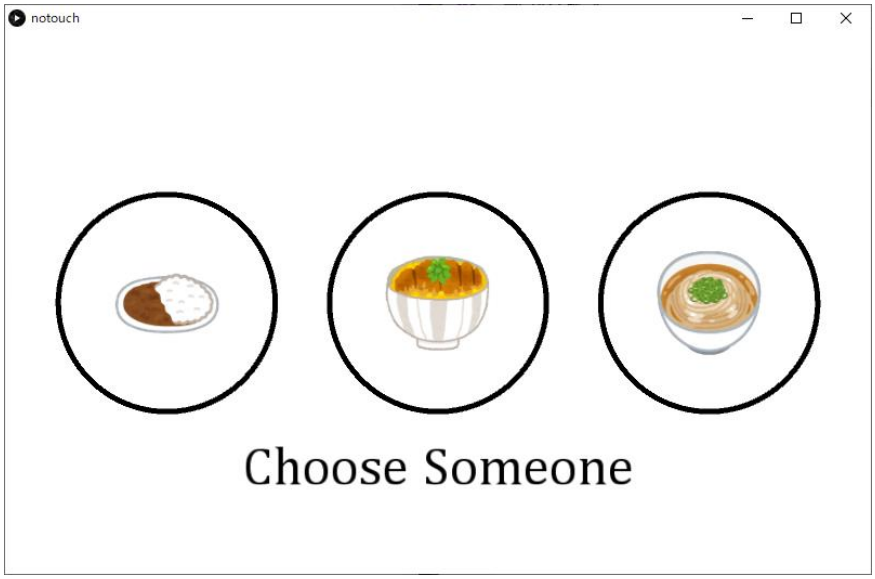

図 4 起動時の実行例

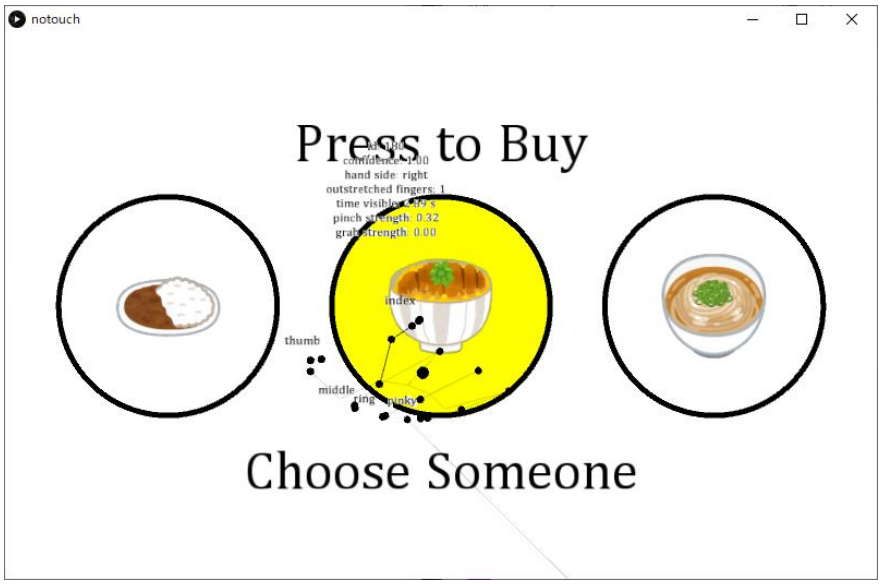

図 5 かつ丼に指を合わせているときの実行例

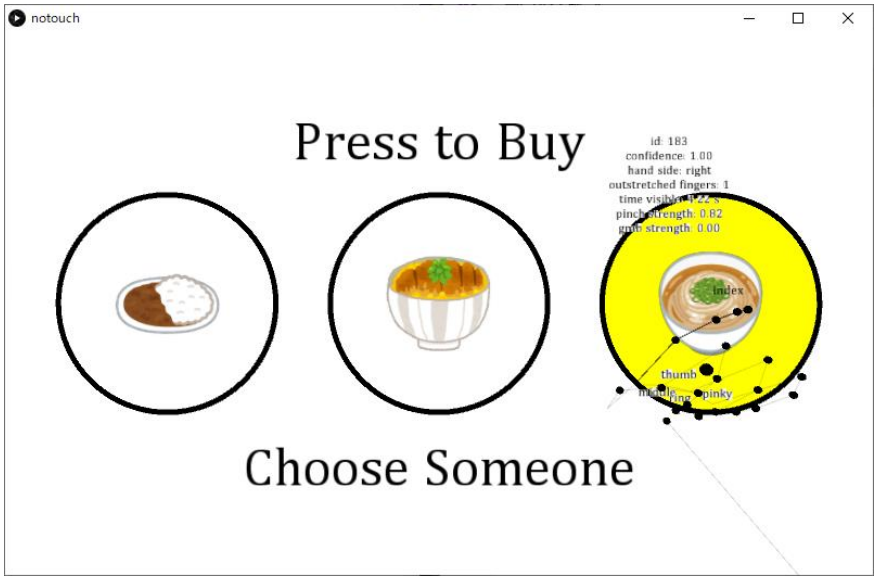

図 6 かけうどんに指を合わせているときの実行例

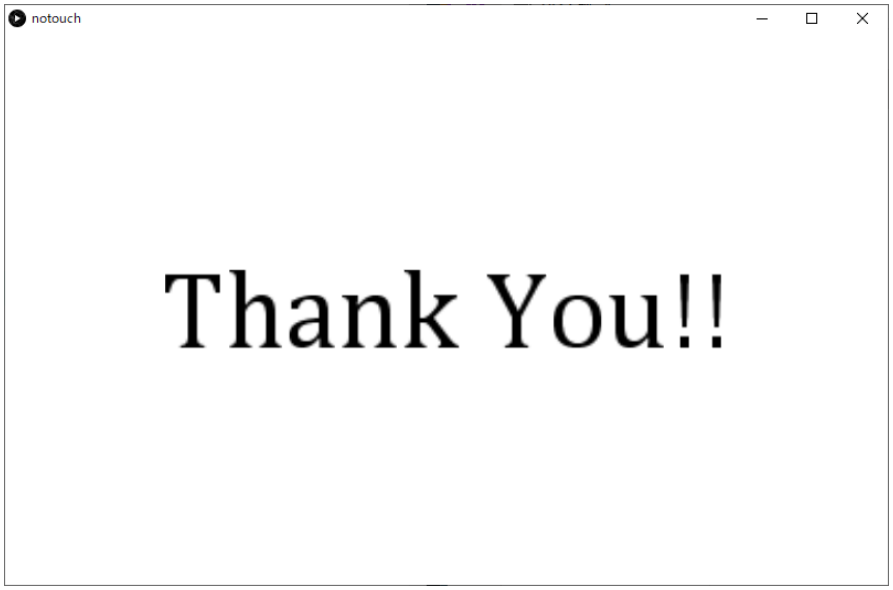

図 7 商品を購入した後の実行例

工夫した点：

・人差し指の座標がボタンの上にある時に、ボタンを黄色く光らせるのと、「Press to Buy」というテキストを表示させるようにした。

・人差し指の座標がボタンの上に来た時に、人差し指の z 座標を記録し、記録した値より人差し指の z 座標が大きくなると商品を購入を確定させるようにした。

4. 感想

Leap Motion を利用して、なれない座標の処理や、どうやったら簡単な操作をできるか、など難しいところが多かったけど、誰でも使えるようなプログラムを組むことができてとてもよかった。

コードが複雑になっていたと感じたので、コードの簡略化をもっと覚えたい。