Final Presentation

Capstone Project	Team 3
Prof. Hojoon Lee	Juntaek Kwon Eunji Gil Sanghyun Kim

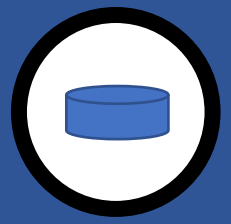
Final presentation

Contents

1 Problem statement

1-1. Problem statement

1. 게임의 난이도를 결정하는 다양한 요인 존재
2. 그 중에서 버튼 사이의 거리와 크기에 집중하여 게임 난이도 조절
3. 나이대별로 약 80%의 성공률을 낼 수 있는 적정 게임 난이도 설정



연령별로 게임 난이도를 조절하는 방식이 부재하여 나이대를 기준으로 버튼 사이의 거리와 크기를 조절하여 나이대에 상관없이 적정 성공률을 낼 수 있도록 게임 난이도 조절의 가이드라인을 제공하고자 한다.

2. Related work

2-1. Related work

[Towards a standard for pointing device evaluation, perspectives on 27 years of Fitts' law research in HCI]

R. William Soukoreff, I. Scott MacKenzie

실험 내용:
- ID = log2(D/W+1)
- 2.0, 2.2, 2.5, 2.7, 3.0, 3.2, 3.4, 3.7, 4.1로 ID를 세팅하여 피실험자는 각 ID에 대하여 랜덤하게 10개의 테니스볼을 가능한 빠르고 정확하게 tap하도록 실험
- W와 D를 이에 맞게 세팅하여 실험
- 각각의 ID에 대한 movement time(MT)를 측정
- Throughput(TP) = ID/MT
- 각 피실험자마다 TP의 평균을 구하여 Tapping Task 수행능력으로 측정함

기존 work에서 배운점 : pointing device와 ISO 9241-9를 활용한 Fitts' task 실험의 설계 방식
기존 work와 프로젝트 주제의 차이점: 연령대별 차이를 둔 실험은 아니며, Fitts' law를 pointing device로 실험하기 위해 ISO 9241-9 tapping task를 제안한 논문이었다.

[The development of a mobile user interface ability evaluation system for the elderly]

Chiuhsiang Joe, LinSui-HuaHo

-노인의 모바일 인터페이스 사용에 Fitts의 법칙 적용
- Fitts 법칙의 결과 측정은 일부 모바일 사용자 인터페이스 기능과 관련이 있다.
- 사용자 인터페이스 동작은 운동 및 인지적 관점에서 설명 할 수 있다.
- 5 motor task와 8 cognitive task의 수행능력 실험
- 총 135 명의 노인이 개발 된 시스템 인 노인 모바일 사용자 인터페이스 능력 평가 시스템 (EMUIAES)으로 평가됨.
- Fitts의 작업 성과와 노인 모바일 사용자 인터페이스 능력 (EMUIA) 간의 관계도 조사
- 인터페이스 사용 능력의 연령 관련 감소

기존 work에서 배운점 : young과 elderly의 5 motor task와 8 cognitive task의 수행능력에 차이가 존재한다.
기존 work와 프로젝트 주제의 차이점: 게임분야에 적용했다는 점과 단순히 노인이 아닌 연령대별 Fitts' task 수행능력을 구분하고자 했다는 점에서 차별점이 있다.

[NSF-Funded Student Design Projects: Fitts' Law Game System for Teaching Accessible Design]

Enabling Technologies Lab, Wayne State University

- 게임 디자인의 원칙으로 Fitts' law 를 활용하여야 한다.
- Fitts의 법칙에 대한 학생 및 교수의 인식을 높이기 위해 고안됨
- 6 개의 다른 목표 위치와 10 개의 다른 거리를 사용한 게임
- 플레이어는 6 가지 난이도 중 하나를 선택
- 난이도가 낮을수록 목표 영역이 커지고, 난이도가 높을수록 대상의 영역이 작아짐
- 주어진 대상이 화면에 무작위로 표시됨
- 플레이어는 커서를 대상 영역으로 이동하고 마우스를 클릭

기존 work와 프로젝트 주제의 차이점: 실제 실험을 진행한 것은 아니라 제안에 그친 프로젝트이며, 난이도 조절에 연령대 요인을 고려한 프로젝트는 아니었다.

3

Design

가설 설정

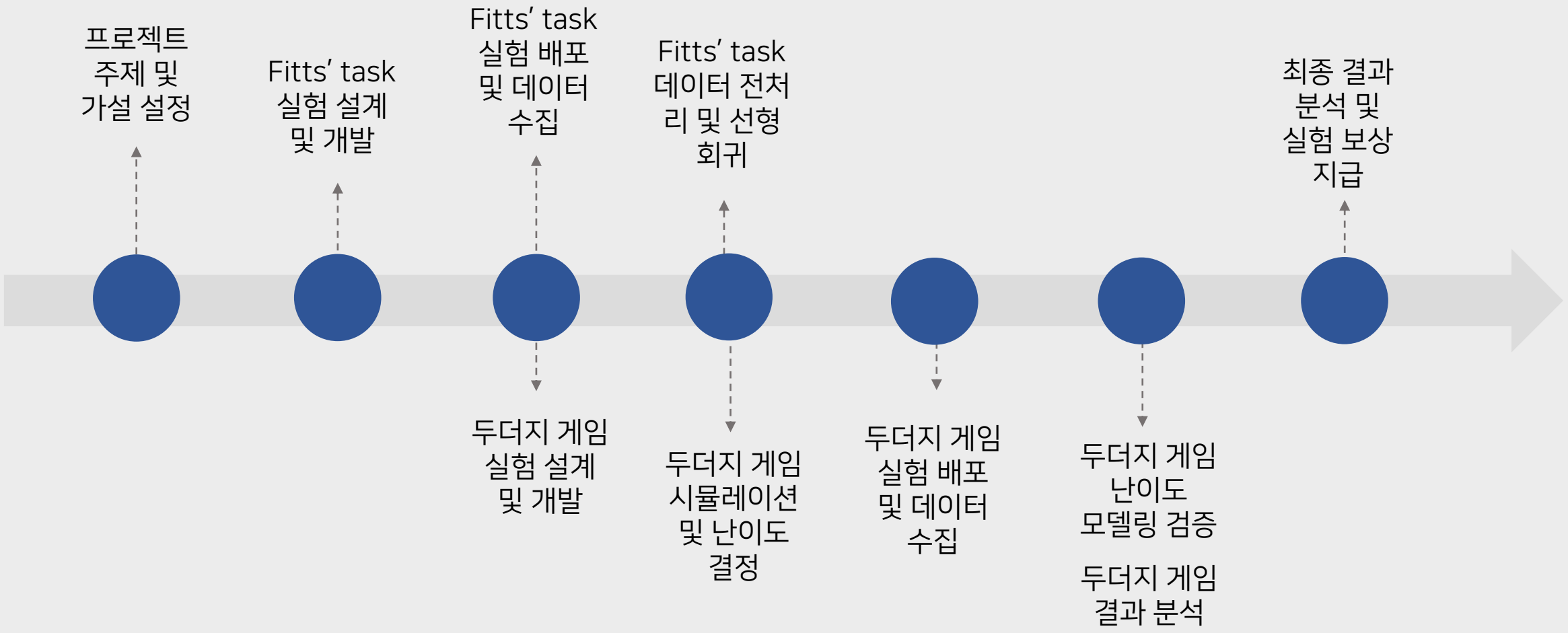
Formoralize a **specific research question** : Fitts' law를 활용하여 연령별로 일정 성공률 이상으로 게임 난이도를 보장해주는 것이 가능할까?

Narrow down our **research hypothesis** : 버튼의 크기와 거리를 조절하여 연령별로 게임 난이도를 조절하여 적정 성공률을 이끌어낼 수 있다.
Fitts' task 실험 : 연령대별로 Fitts' task 수행에 차이가 있어서 연령대별 Fitts' law 식을 다르게 구할 수 있다.
Whack-a-mole 실험 : Fitts' law 모수를 기반으로 난이도를 모델링하여 연령대별 80%의 성공률을 도출할 수 있다.

Narrow it down to some **independent variable(s)**
Fitts' task 실험 : 나이
Whack-a-mole 실험 : 연령대별 난이도 결정 모수

Narrow it down to measurable **dependent variable(s)**
Fitts' task 실험 : 연령대별 Fitts' task 수행능력
Whack-a-mole 실험 : 두더지게임의 성공률

프로젝트 수행 일정


프로젝트 주제 및 가설 설정
Fitts' task 실험 설계 및 개발
Fitts' task 실험 배포 및 데이터 수집
Fitts' task 데이터 전처리 및 선형 회귀
최종 결과 분석 및 실험 보상 지급
두더지 게임 실험 설계 및 개발
두더지 게임 시뮬레이션 및 난이도 결정
두더지 게임 실험 배포 및 데이터 수집
두더지 게임 난이도 모델링 검증
두더지 게임 결과 분석

3-1. Fitts‘ task

① Fitts’ Task 설계 세부사항

1) Pointing device의 평가 방법의 표준인 ISO 9241-9를 적용

2) 실험 방식은 Fitts’ law 모델 설계 시 7가지 추천사항을 제시한 논문을 참고

실험 방식:

- Shannon formulation of ID = $\log_2(\frac{D}{W})$를 사용
- ID값이 2.0, 2.5, 3.0, 3.5, 4.0이 되도록 D와 W를 조절하여 실험을 설계
- 먼저 피실험자의 나이를 조사
- 피실험자는 각 ID에 대하여 ISO 9241-9를 연속적으로 수행
- 각각의 ID에 대한 수행완료 시간을 기록

위와 같은 실험을 웹으로 설계하여 연령별 실험 데이터 수집

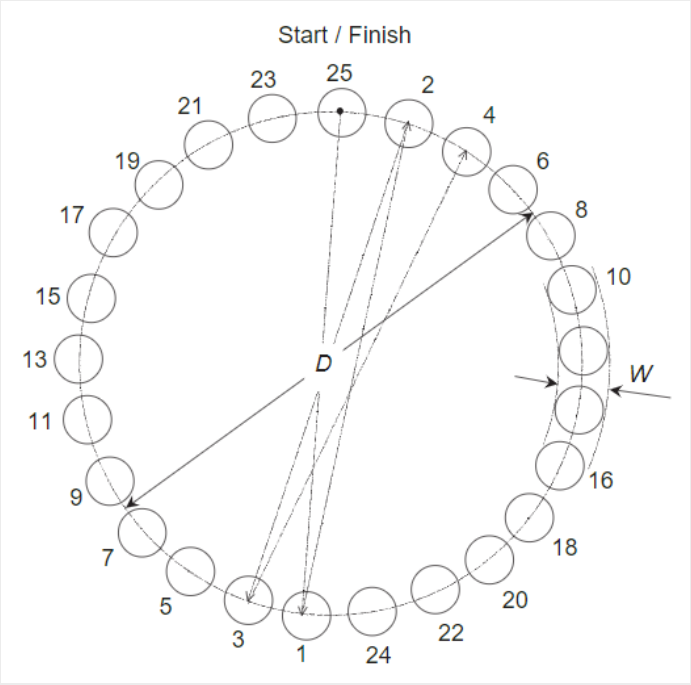


Fig. 1) ISO 9241-9
실제 게임 상황에 맞는
Multi-Directional 환경으로
Fitts’ task 실험

② 나이대별 모수 추정 세부사항

- 나이대별로 Fitts’ law의 parameter를 추정하기 위해 수집한 데이터를 나이대 (10~20대, 30~40대, 50~60대)로 구분

- Shannon formulation ID 사용 시 Fitts’ law 식이 T = a + b·ID 로 나타나기 때문에 데이터에 선형회귀분석을 적용하여 parameter a와 b를 추정

- 이 때, Python Scikit-learn의 LinearRegression 라이브러리 활용

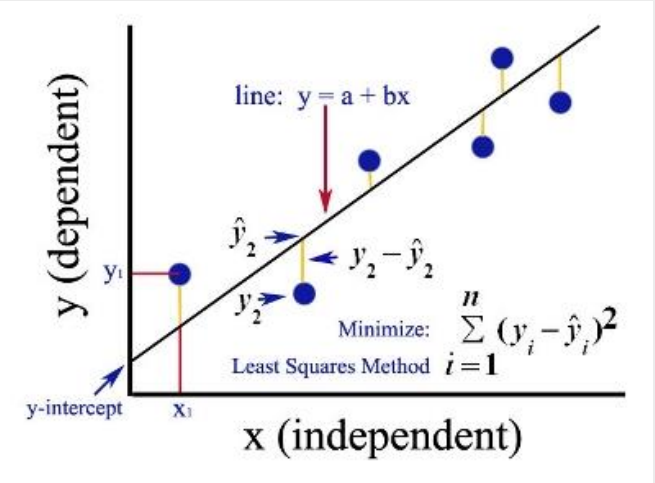


Fig. 2) Linear Regression

③ 실험 세부사항

- 10~20대, 30~40대, 50~60대로 나이대를 그룹화하여 그룹별로 약 20명의 실험참여자 모집
- 성별의 상관관계, 교육 수준의 상관관계는 는 없다고 가정하여 실험참여자를 모집
- 실험 참여 방법 재차 강조(홍보시에 한번, 실험 페이지에 설명 한번)
- Distance대비 Time이 짧은 3명의 참여자에게 보상 지급

3-2. Whack-a-mole

① 두더지 게임 설계 세부사항

- Fitts' Task가 잘 적용될 수 있도록 다음과 같은 방식의 두더지 게임을 개발

1) 30초간 게임을 진행하며, 두더지 한 마리가 나온 뒤 t1~t2 (난이도 결정 모수) 사이의 랜덤 표시 시간 동안 표시되었다가 사라지며 다른 두더지가 나오는 방식으로 설계

2) 허용된 영역 내부를 터치하면 성공으로 인식하며, 그 밖의 영역을 터치하는 경우 실패로 인식

- 연령별로 두더지의 크기(W) 및 두더지가 나타나는 평균 거리(D)에 따라 게임의 난이도를 3단계로 세분화

- 나이대별로 추정한 Fitts' law 식을 적용하여 사용자 나이 입력 시 사용자를 나이에 맞는 표준 난이도(잡은 두더지의 수가 잡을 수 있는 최대 마리 수의 80%가 되는 난이도)에 배치

② 두더지 실험 세부사항

- 10~20대, 30~40대, 50~60대로 나이대를 그룹화하여 그룹별로 약 20명의 실험참여자 모집

- 성별의 상관관계와 교육수준의 상관관계는 없다고 가정

- 실험 참여 방법 재차 강조(홍보시에 한번, 실험 페이지에 설명 한번)

- 두더지 게임 성공률이 높은 실험 참여자 3명에게 보상 지급

- 배치된 나이대별 표준 난이도에서의 기록을 분석하여 나이에 따른 난이도 배치 기준의 적정성 평가

4 Implementation

4-1. Fitts‘ task

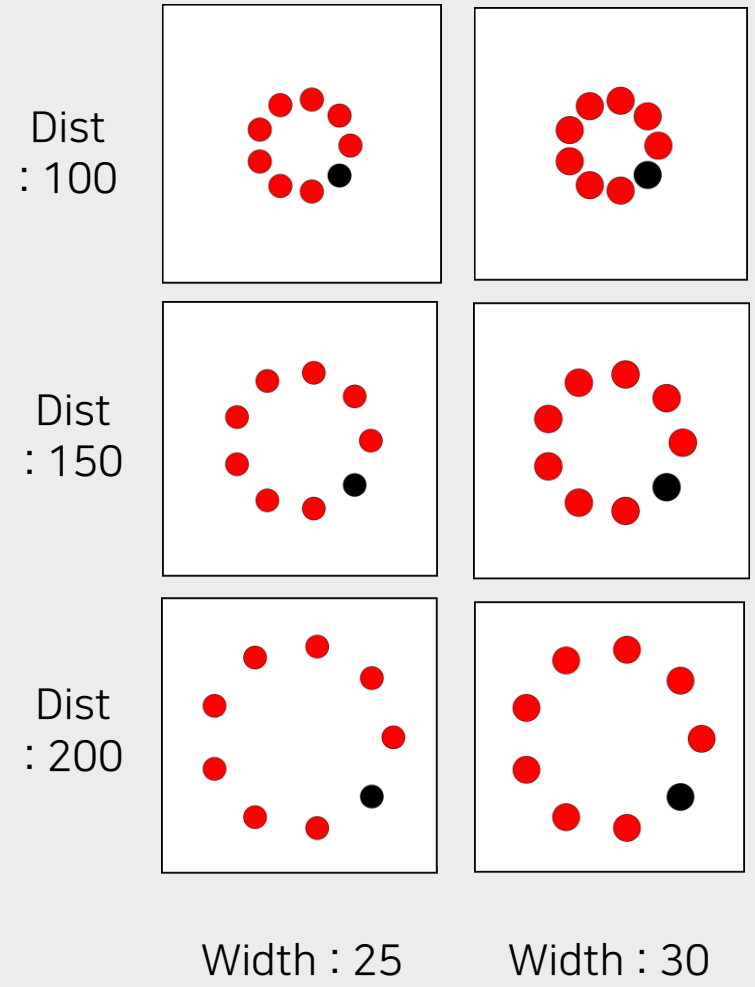
Fitts’ task 실험 개발 및 배포

```
var dist = 100;
var width = 25;
var dist2 = 150;
var dist3 = 200;
var width2 = 30;
var dataVals = [[0.698132, "1"], [1.39

var page = d3.select("body").append("s
page.attr("width", 600).attr("height",
    .style("border", "4px solid black"

var circles = page.selectAll("circle")
    .data(dataVals)
    .enter()
    .append("circle");
```

디자인 설계에서 정한 ID값의 범위에 맞도록 버튼의 크기와 거리를 지정




```
function clickCircle() {
    //var index=0;
    if (index < 10) {
        var coordinates = [0, 0];
        d3.select(this).style("fill", "black");
        coordinates = d3.mouse(this);
        //time=Date.now();
        timeArry[index] = Date.now();
        var x = coordinates[0];
        var y = coordinates[1];
        xcoordinateArray[index] = x;
        ycoordinateArray[index] = y;
        //coordinateArray.push(coordinates);
        //console.log(coordinateArray);
        //document.write(coordinateArray);
    }
    index++;
    allcircles = page.selectAll("circle");
    allcircles.each(function (d, i) {
        var tempId = this.getAttribute("id");
        var indexId = index.toString();
        if (tempId == indexId)
            this.style.fill = "black";
        else
            this.style.fill = "red";
    }
    )
    //<input type="number" id="Amplitude" value=50 /><br>
}
```

검정색 버튼을 클릭하는데 걸린 시간과 클릭한 버튼 간 거리를 측정

4-1. Fitts‘ task

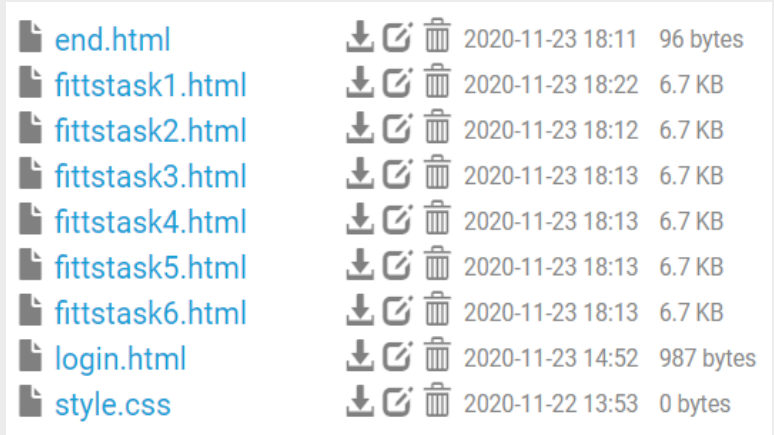
실험 기간 : 11.23 ~ 12.2(약 2주)

Fitts’ task 실험 진행

http://fitts.pythonanywhere.com/

```
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width">
</head>

<body>
    <form action="{% url 'fitts:login' %}" method="POST">
        {% csrf_token %}
        <input type="text" name="email" placeholder="이메일을 입력하세요">
        <input type="text" name="age" placeholder="나이를 입력하세요">
        <input type="submit" value="제출하고 실험 참여하기">
    </form>
    <img src="{% static 'fitts/guide.jpg' %}" style="width: 1000px; height: 800px; bo
    </img>
    <!--
    <div class="container" id="container">
        <div class="form-container sign-in-container">
            <form action="#">
                <h1>Sign in</h1>
                <input type="email" placeholder="Email" />
                <input type="age" placeholder="Age" />
                <button>Sign In</button>
```




```
def fitts1(request):
    if request.method == "POST":
        for i in range(1, 9):
            age = request.session.get('age', None)
            email = request.session.get('email', None)
            transform = request.POST.get('transform', None)
            distance = request.POST.get('distance' + str(i), None)
            time = request.POST.get('time' + str(i), None)
            Tasks.objects.create(
                age = age,
                email = email,
                transform = transform,
                distance = distance,
                time = time
            )
        return redirect('fitts:fitts2')
    elif request.method == "GET":
        return render(request, 'fitts/fittstask1.html',{})

def fitts2(request):
    if request.method == "POST":
        for i in range(1, 9):
            age = request.session.get('age')
```

login.html

사용자의 이메일과 나이가
POST Method를 통해 백엔드로
전송된다.

fittstask1.html ~ end.html

백엔드 전송의 편의성을 위해 6개의
Fitts’ task를 각각의 html파일로
재구성

html / views.py / urls.py / models.py

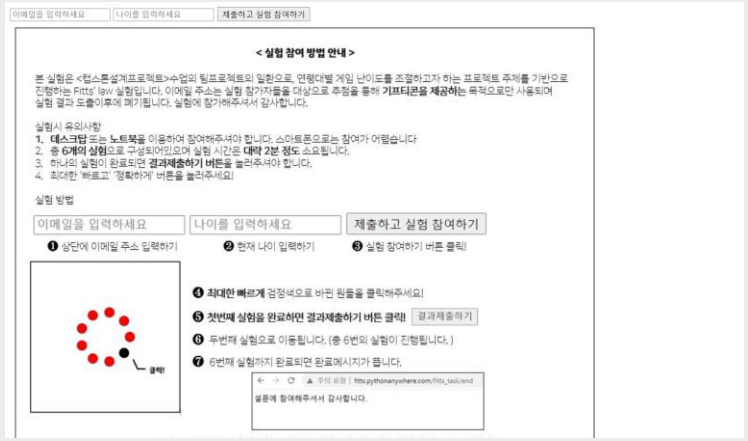
Session에 저장돼있던 이메일, 나이
데이터와 더불어 Fitts’ task 결과값을
서버에 전송

4-1. Fitts‘ task

실험 기간 : 11.23 ~ 12.2(약 2주)

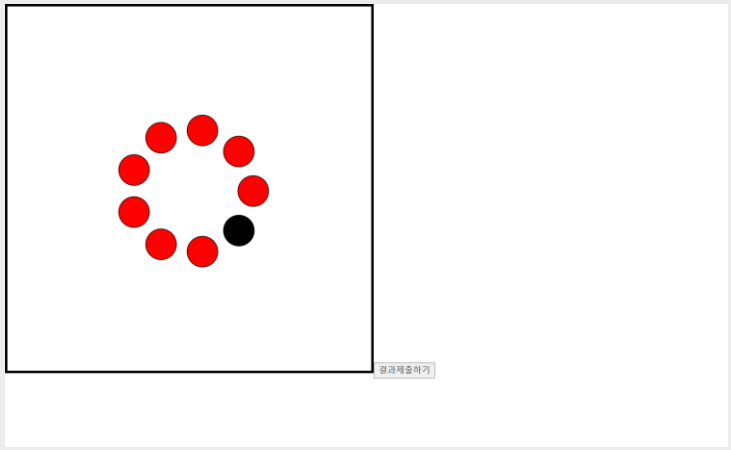
Fitts’ task 실험 진행

http://fitts.pythonanywhere.com/



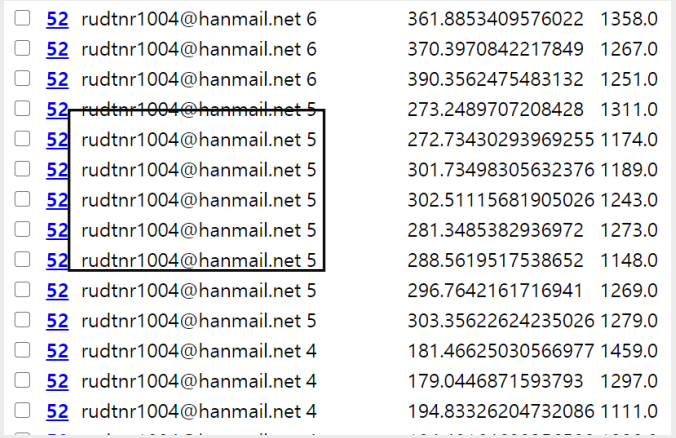
이메일 작성 및 실험 안내 화면

사용자는 이메일을 입력하고 올바른
실험 참여 방식에 대해
안내받는다.



Fitts’ task 실험 화면

사용자는 총 6번의 실험을 진행하며,
결과제출하기 버튼을 이용하여
다음 실험으로 이동한다.



52	rudtnr1004@hanmail.net	6	361.8853409576022 1358.0
52	rudtnr1004@hanmail.net	6	370.3970842217849 1267.0
52	rudtnr1004@hanmail.net	6	390.3562475483132 1251.0
52	rudtnr1004@hanmail.net	5	273.2489707208428 1311.0
52	rudtnr1004@hanmail.net	5	272.73430293969255 1174.0
52	rudtnr1004@hanmail.net	5	301.73498305632376 1189.0
52	rudtnr1004@hanmail.net	5	302.51115681905026 1243.0
52	rudtnr1004@hanmail.net	5	281.3485382936972 1273.0
52	rudtnr1004@hanmail.net	5	288.5619517538652 1148.0
52	rudtnr1004@hanmail.net	5	296.7642161716941 1269.0
52	rudtnr1004@hanmail.net	5	303.35622624235026 1279.0
52	rudtnr1004@hanmail.net	4	181.46625030566977 1459.0
52	rudtnr1004@hanmail.net	4	179.0446871593793 1297.0
52	rudtnr1004@hanmail.net	4	194.83326204732086 1111.0

결과 데이터 수집 화면

연령, Transform, Distance, Time
결과 데이터를 수집하며
이메일을 사용하여 데이터를 구별한다.

4-1. Fitts‘ task

Fitts’ task 데이터 전처리 및 선형 회귀

```
fittstask['ID'] = np.log2((2*fittstask['Distance'])/fittstask['Width'])
fitts['ID'] = np.log2((2*fitts['Distance'])/fitts['Width'])

fittstask = fittstask[fittstask['ID'] > 1.5]
fitts = fitts[fitts['ID'] > 1.5]
fittstask = fittstask[fittstask['Time'] < 3000]
fitts = fitts[fitts['Time'] < 3000]
```

ID=$\log_2(2D/W)$ 값 계산

부적절한 데이터 제거
- ID 값이 작으면 동일한 원 클릭했을 것
- 다음 원을 클릭하는데 3초 이상 걸리면 문제가 있을 것

```
X_10to20 = df_10to20[['ID']]
Y_10to20 = df_10to20[['Time']]
X_30to40 = df_30to40[['ID']]
Y_30to40 = df_30to40[['Time']]
X_50to60 = df_50to60[['ID']]
Y_50to60 = df_50to60[['Time']]
```

10~20대, 30~40대, 50~60대로 나누어서
x 좌표에 ID, y 좌표에 시간 값을 입력

```
LR_10to20 = LinearRegression()
LR_10to20.fit(X_10to20, Y_10to20)
y_10to20 = LR_10to20.predict(X_10to20)
print("10~20 R_square: ", r2_score(Y_10to20, y_10to20))
round(float(LR_10to20.coef_))
int(LR_10to20.intercept_)
```

파이썬 사이킷런을 통해 x, y좌표에 따른 선형 회귀 분석
→ Fitts’ Law의 a, b 값 즉, 절편과 기울기 도출

4-2. Whack-a-mole

두더지 게임 난이도 결정

```
def fitts_10to20(r):
    return 216.2+136*(r-5) # width = 64px

def fitts_30to40(r):
    return 226.7+144.9*(r-5) # width = 64px

def fitts_50to60(r):
    return 640+128.9*(r-6) # width = 128px
```

연령대별 Fitts Law 식 도출

```
while(len(r_list) < 100000):
  r_list.append(generatePosition())

while(len(t_list) < 100000):
  t_list.append(randTime(860, 1065))

# 두더지가 나타나 있는 시간이 잡는데 걸리는 시간보다 크면 잡았다는 의미로 True를 리턴
for i in range(100000):
  catch.append(fitts_50to60(r_list[i]) < t_list[i])

print("거리의 평균(100,000개): ", np.mean(r_list))
print("거리의 표준편차(100,000개) : ", math.sqrt((np.sum(pow((r_list-np.mean(r_list)), 2)))/100000))
print("성공률: ", np.mean(catch))
```

두더지 게임 시뮬레이션 코드

정규분포를 따라 결정되는 두더지 위치와 정해진 시간 범위 내에서 결정되는
두더지 노출 시간을 100000개 생성하여 배열에 추가

두더지 사이의 거리를 Fitts Law 식에 대입했을 때 시간이
무작위로 생성된 시간보다 작으면 두더지 포획 가능, 이외에는 포획 불가로 판단

두더지 게임 난이도 결정

```
False,
False,
False,
False,
True,
False,
False,
False,
True,
False,
False,
True,
False,
False,
True,
False,
True,
False,
True,
False,
False,
False,
True,
True,
False,
False,
False,
True,
False,
False,
False,
False,
...]
```

```
In [2]: runfile('C:/Users/KKW/Desktop/캡스톤/Sampling.py', wdir='C:/Users/KKW/Desktop/캡스톤')
거리의 평균(100,000개):  8.002193992406333
거리의 표준편차(100,000개) :
0.29896015270989423
성공률:  0.79915
```

10~20대
Width = 64px
D = lognormalRandom(8.11, 0.32)
T = randtime(580, 815)

30~40대
Width = 64px
D = lognormalRandom(8.11, 0.32)
T = randtime(620, 850)

50~60대
Width = 128px
D = lognormalRandom(8.11, 0.32)
T = randtime(860, 1065)

시뮬레이션 결과

모수 추정 결과

두더지 게임 개발

```
// 첫 두더지 좌표 생성 함수 (나타나지 않는 임시 좌표)
function firstGeneratePosition() {
  x = Math.random() * 620;
  y = Math.random() * 370;
  lastPosition_x = x;
  lastPosition_y = y;
}
```

두더지가 처음 나타나는 좌표 설정

각도(theta)값은 랜덤하게 설정

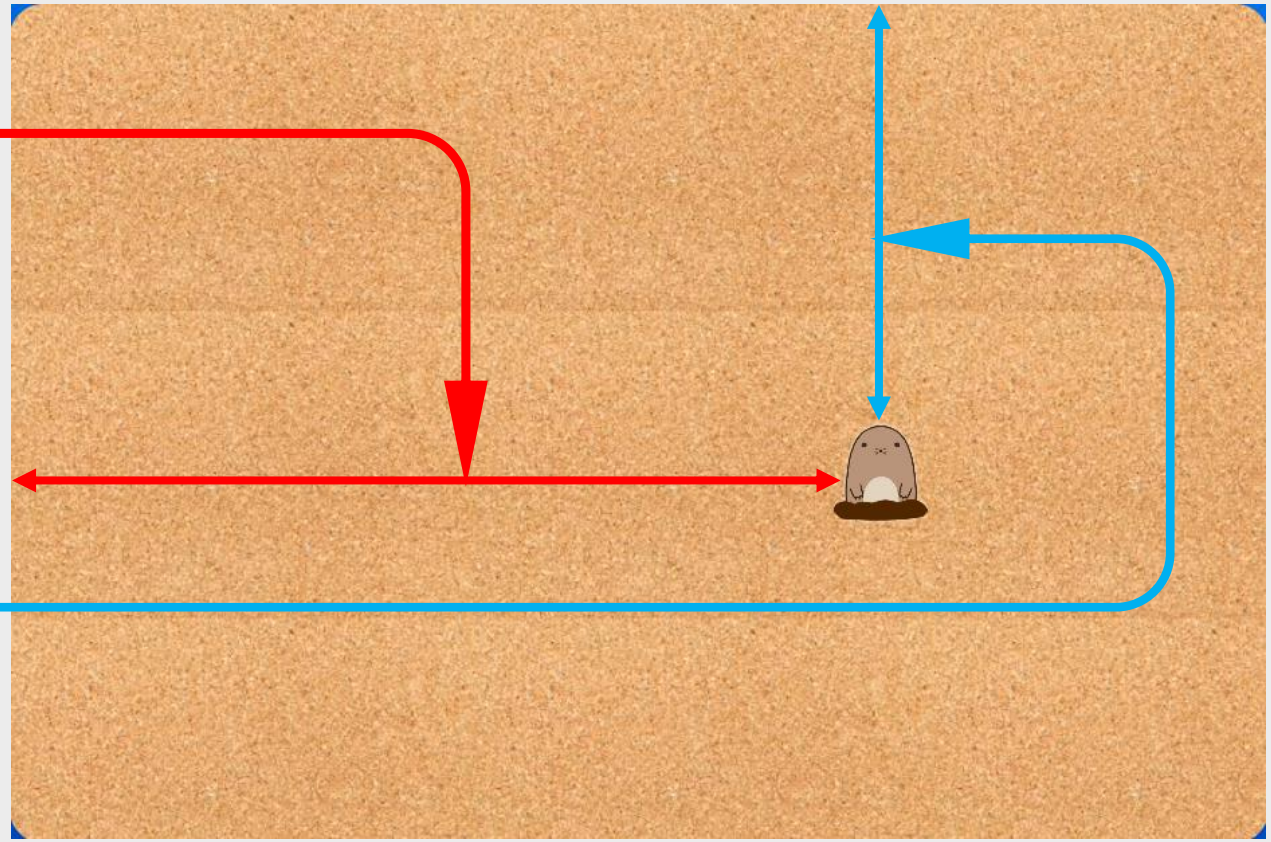
```
// 새로운 두더지 좌표 생성 함수
function generatePosition() {
  do {
    mu = document.getElementById("mu").value;
    mu = parseInt(mu);
    console.log(mu);
    r = lognormalRandom(mu, 0.32); // parameter: mu(평균), sigma(표준편차)
    var theta = Math.random() * 2 * Math.PI;
    x = lastPosition_x + r * Math.cos(theta);
    y = lastPosition_y + r * Math.sin(theta);
  } while (x < 0 || x > 620 || y < 0 || y > 370); // 지정 범위 밖의 범위에서 좌표가 생성될 경우 좌표를 재생성

  lastPosition_x = x;
  lastPosition_y = y;
}
```

Fitts' Task를 통해 결정한 평균, 표준편차 값을 대입하여 lognormal 분포를 따르는 거리(r)값 생성
mu : 8.11
sigma : 0.32

두더지 게임 개발 및 배포

```
function showingMole() {
  if (turn === 0) {
    firstGeneratePosition();
    turn++;
    showingMole();
  }
  else {
    i = document.getElementById("pk").value;
    i = parseInt(i);
    console.log(i);
    time = randTime(t1[i], t2[i]);
    moleNumber = document.getElementById('1');
    generatePosition();
    const mole = document.getElementById('molespace');
    mole.style.width = width[i] + "px";
    mole.style.height = width[i] + "px";
    mole.style.marginLeft = x + "px";
    mole.style.marginTop = y + "px";
    console.log(turn, r, time);
    moleActive(moleNumber);
    moleNumber.addEventListener('click', catchMole);
    moleCatch = setTimeout(seeMole, time);
    turn++;
  }
}
```



<두더지가 나타나는 위치 결정 함수>

4-2. Whack-a-mole

실험 기간 : 12.2 ~ 12.9(약 1주)

두더지 게임 실험 진행

http://mole.pythonanywhere.com/

```
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width">
    <title>두더지를 잡아라!</title>
    <link rel="stylesheet" href="{% static 'game/homestyle.css' %}">
</head>

<body>
    <div class="container">
        <div class="title" style="display:flex; justify-content: center; text-align
            <div style=" width:fit-content;">
                <img src="{% static 'game/mole.png' %}" alt="" style="width: 70px;
                <div style="width:fit-content; float:left;">
                    두더지를 잡아라!
                </div>
                <img src="{% static 'game/mole.png' %}" alt="" style="width: 70px;
            </div>
        </div>
        <br>
        <br>
        <form action="{% url 'game:home' %}" method="POST">
            {% csrf_token %}
            <div>
                <div class="choice_age">
                    이메일을 입력해주세요
```

home.html / homestyle.css

사용자가 입력한 연령대와 이메일이
POST Method를 통해
백엔드로 전송된다.

```
// 새로운 두더지 좌표 생성 함수
function generatePosition() {
  do {
    mu = document.getElementById("mu").value;
    mu = parseInt(mu);
    console.log(mu);
    r = lognormalRandom(mu, 0.32); // parameter: mu(평균), sigma(표준편차)
    var theta = Math.random() * 2 * Math.PI;
    x = lastPosition_x + r * Math.cos(theta);
    y = lastPosition_y + r * Math.sin(theta);
  } while (x < 0 || x > 620 || y < 0 || y > 370); // 지정 범위 밖의 범위에서 좌표가 생성될 경우

  lastPosition_x = x;
  lastPosition_y = y;
}

function showingMole() {
  if (turn === 0) {
    firstGeneratePosition();
    turn++;
    showingMole();
  }
  else {
    i = document.getElementById("pk").value;
```

game.html / style.css / mole.js

두더지가 처음 나타나는 좌표와 새로운
두더지가 나타나는 좌표를 설정한다.

```
def home(request):
    if request.method == "GET":
        return render(request, 'game/home.html')
    elif request.method == "POST":
        age = request.POST.get('age')
        email = request.POST.get('email')
        request.session['email'] = email
        pk = 0
        if age == "ten":
            pk = 0
        elif age == "thirty":
            pk = 1
        elif age == "fifty":
            pk = 2
        request.session['age'] = pk
        return redirect('game:game', pk=pk)


def game(request, pk):
    if request.method == "GET":
        data = {
            'pk': pk,
            'mu': 8.11,
```

html / views.py / urls.py / models.py

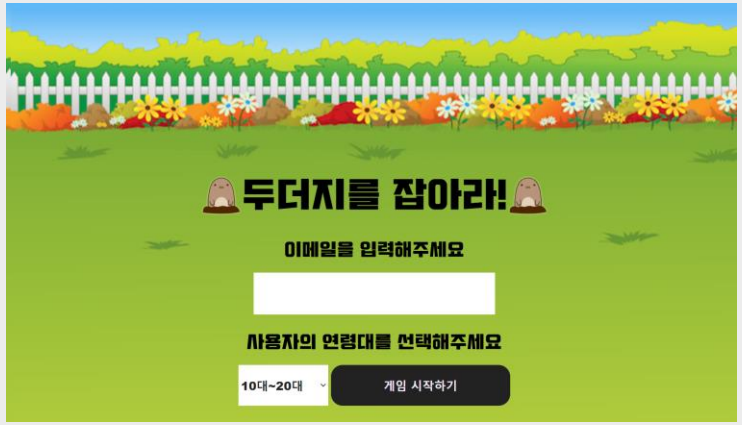
Select태그의 옵션을 통해 전달된
연령대 값을 바탕으로 pk를 설정하고
이를 두더지 게임의 난이도를 결정하는
parameter로 사용한다.

4-2. Whack-a-mole

실험 기간 : 12.2 ~ 12.9(약 1주)

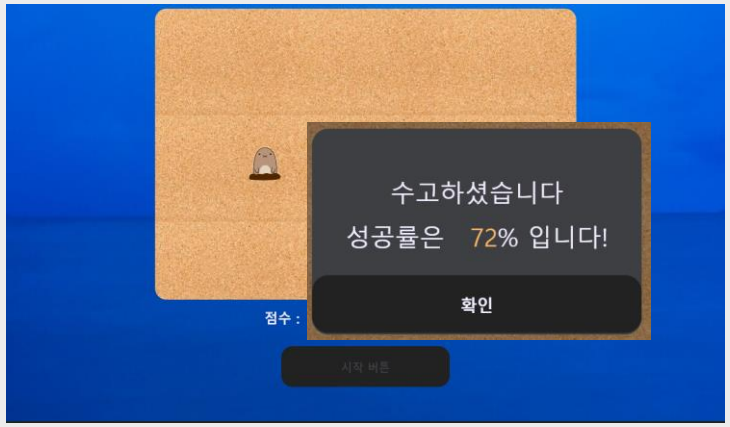
두더지 게임 실험 진행

http://mole.pythonanywhere.com/



연령대 선택 화면

사용자는 이메일을 입력하고
10~20대, 30~40대, 5~60대 중에
연령대를 선택한다.



두더지 게임 실험 화면

사용자는 게임 종료 문구가 뜰 때까지인
30초 동안 게임을 진행한다.

☐	bada10202001@nate.com	2	83
☐	hsh10112_@naver.com	2	80
☐	ee5551@naver.com	2	87
☐	jangsinhee07_@naver.com	2	89
☐	abba3355@naver.com	0	85
☐	zeusk8255@naver.com	0	80
☐	covh1029@naver.com	0	91
☐	more-lovehae@hanmail.net	2	86
☐	aftersun_set@naver.com	0	84

결과 데이터 수집 화면

10~20대는 0, 30~40대는 1,
50~60대는 2로 Age값을 구별한다.

5

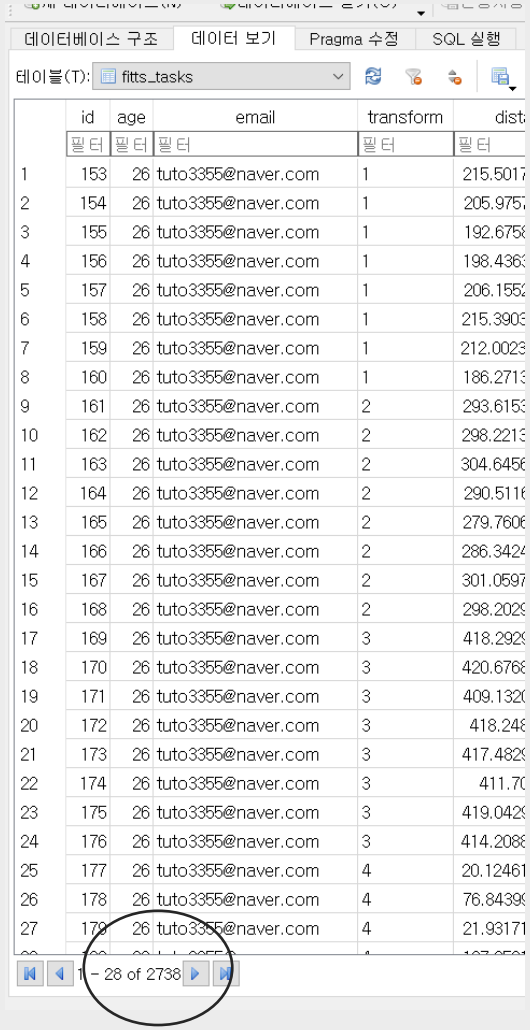
Evaluation

5-1. Fitts‘ task

Fitts’ task 연령별 실험 참여자 수

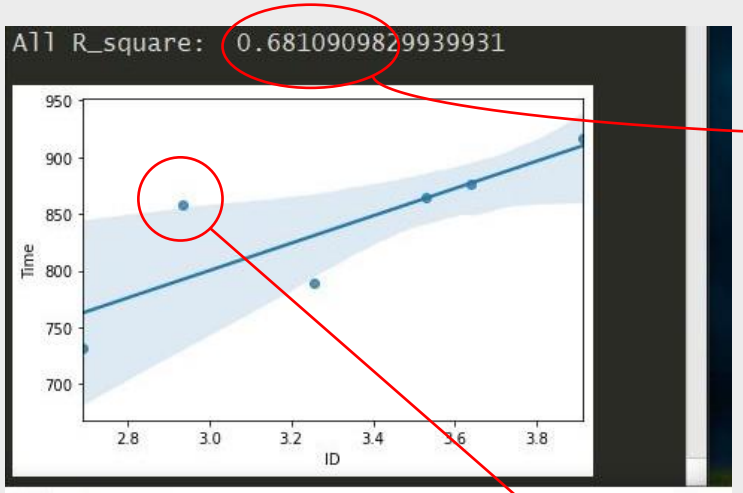
df_10to20	Series	(33,)
df_30to40	Series	(15,)
df_50to60	Series	(14,)

- 10~20대: 33명
- 30~40대: 15명
- 50~60대: 14명

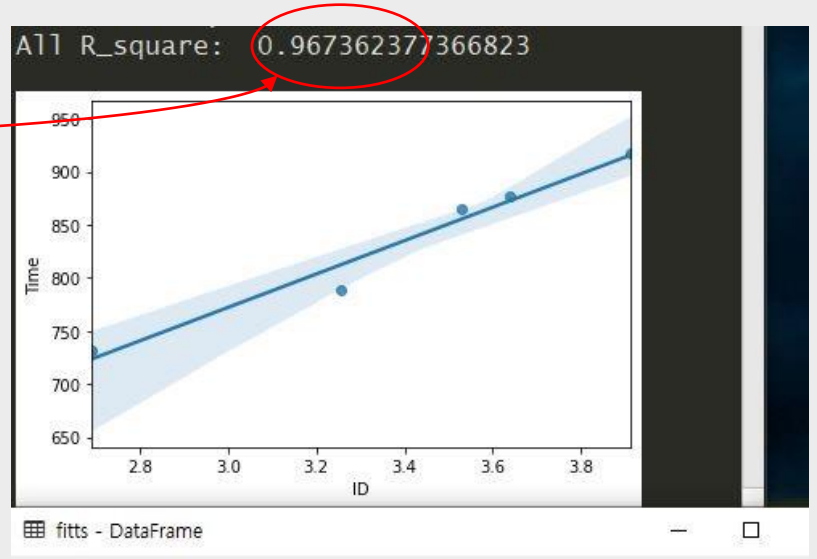


5-1. Fitts‘ task

각 Transform별 Fitts’ task의 평균(전 연령) 데이터



Index	Age	ID	Time	Transform
0	34.796	2.93276	858.308	1
1	36.3261	3.53037	864.528	2
2	35.3021	3.91305	916.726	3
3	35.545	2.69114	732.229	4
4	36.1874	3.25491	788.874	5
5	35.7888	3.63946	876.578	6



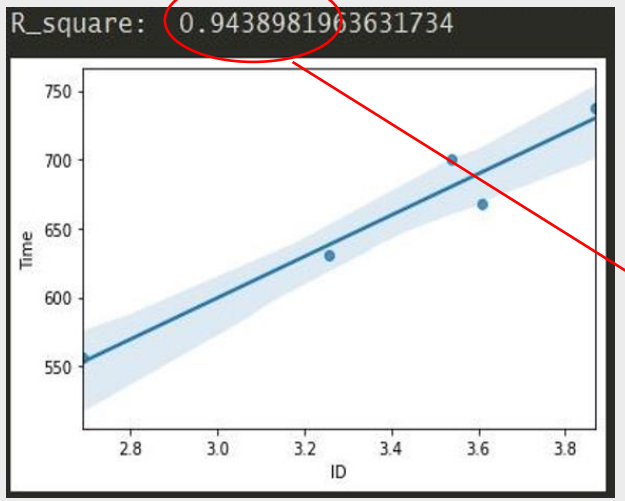
Index	Age	ID	Time	Transform
1	36.3261	3.53037	864.528	2
2	35.3021	3.91305	916.726	3
3	35.545	2.69114	732.229	4
4	36.1874	3.25491	788.874	5
5	35.7888	3.63946	876.578	6

Transform1의 수행 시간이 오래 걸림
→ Transform1을 실험 방식에 익숙해지는 단계로 간주하여 삭제 (초반 퍼포먼스 고려)

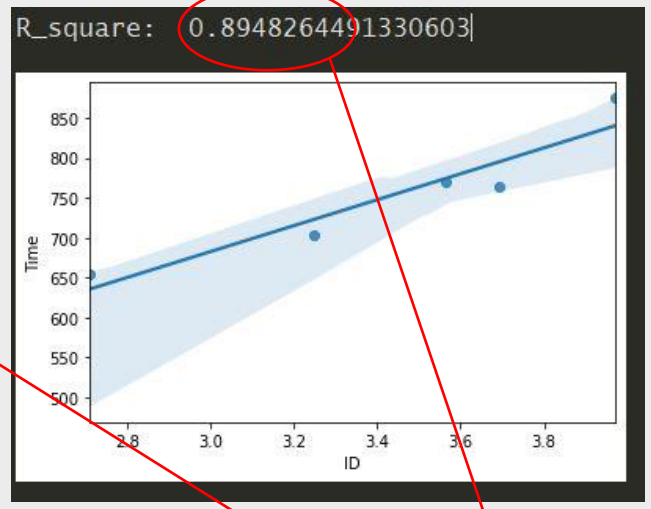
Transform1 삭제 결과 전체 평균 데이터의 R^2값이 0.967로 크게 상승함

5-1. Fitts‘ task

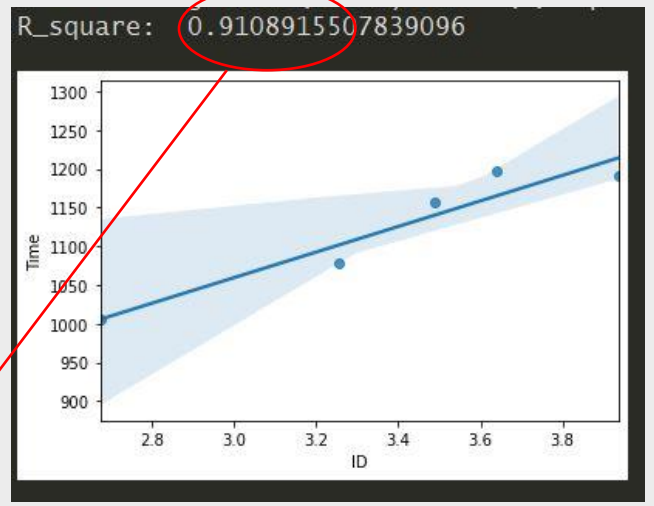
Transform별 Fitts’ task의 평균(연령대별) 데이터



[10대~20대]



[30대~40대]



[50대~60대]

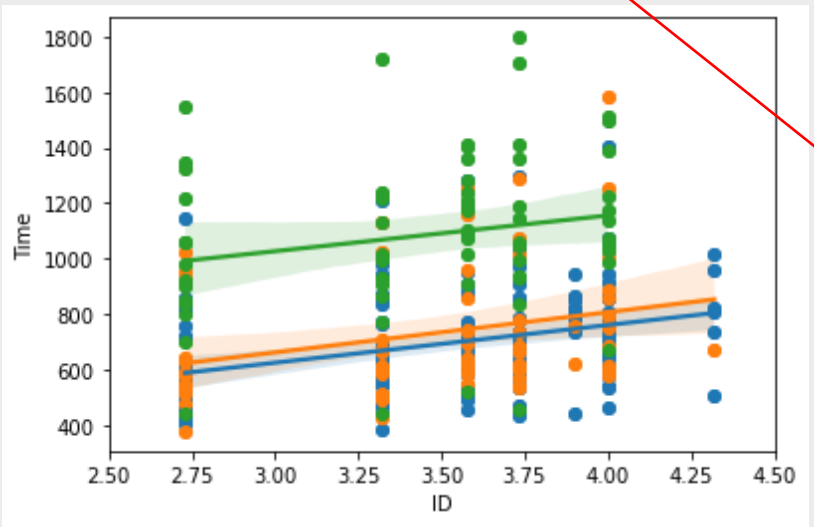
Transform1을 제거한 연령대별 평균 데이터 또한 R^2값이 0.9 전후로 준수하게 나타남을 확인할 수 있음

→ Fitts’ task 실험은 Fitts’ law를 잘 반영하는 방향으로 설계되었음을 알 수 있음

5-1. Fitts‘ task

최종 Regression 결과

```
10~20 R_square:  0.10296841400784806
30~40 R_square:  0.07443506920394l5
50~60 R_square:  0.040388184965941 43
```



10 ~ 20대

30 ~ 40대

50 ~ 60대

그러나, 데이터 평균치의 R^2에 비해 개별 데이터의 R^2값은 현저히 하락함

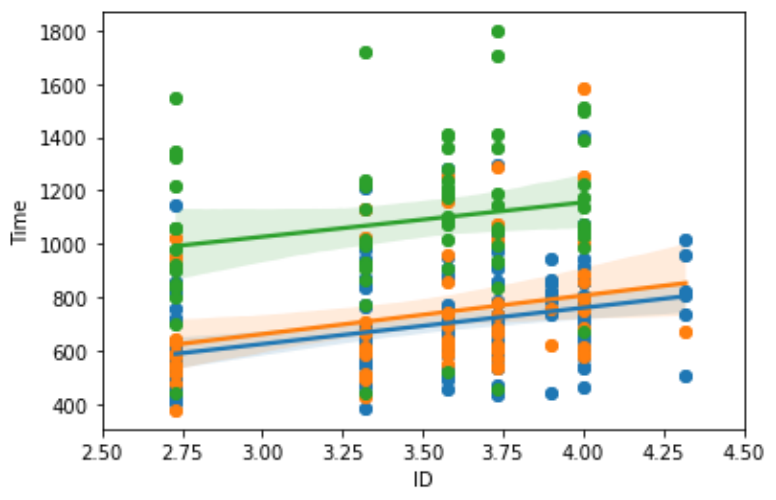
→ 개개인의 Fitts’ task의 수행 능력에는 연령대 외에도 다양한 요소가 영향을 미치기 때문에 편차가 커진 것으로 사료됨

또한, 연령대가 증가할수록 R^2값이 하락함

→ 연령대가 증가할수록 개개인의 컴퓨터 사용 빈도가 크게 다르기 때문에 이에 따라 수행 결과가 크게 달라진 것으로 사료됨

5-1. Fitts‘ task

최종 Regression 결과



10 ~ 20대

30 ~ 40대

50 ~ 60대

```
In [2]: LR_10to20.coef_
Out[2]: array([[135.98147361]])

In [3]: LR_10to20.intercept_
Out[3]: array([216.18181897])

In [4]: LR_30to40.coef_
Out[4]: array([[144.89721]])

In [5]: LR_30to40.intercept_
Out[5]: array([226.68580203])

In [6]: LR_50to60.coef_
Out[6]: array([[128.87349086]])

In [7]: LR_50to60.intercept_
Out[7]: array([639.94165067])
```

기울기

절편

낮은 R^2에도 불구하고, 연령이 증가할수록 수행능력이 감소하는 경향성을 발견할 수 있음

→ 연령대를 난이도의 구분 요소 중 하나로 설정하는 것은 유의미함

최종 Regression 결과

```
In [10]: df_10to20.describe()
Out[10]:
              Age          ID         Time
count  145.000000  145.000000   145.000000
mean    24.671264    3.529655   696.149531
std      1.904748    0.444337   188.295582
min     20.000000    2.730000   386.375000
25%     23.000000    3.320000   571.125000
50%     25.000000    3.580000   677.250000
75%     26.000000    3.900000   805.285714
max     28.000000    4.320000  1406.250000

In [11]: df_30to40.describe()
Out[11]:
              Age          ID         Time
count   67.000000   67.000000    67.000000
mean    32.686567    3.489552   732.312189
std      3.499822    0.441843   234.660267
min     30.000000    2.730000   377.000000
25%     31.000000    3.320000   584.062500
50%     31.000000    3.580000   661.000000
75%     32.500000    3.730000   826.125000
max     44.000000    4.320000  1587.375000

In [12]: df_50to60.describe()
Out[12]:
              Age          ID         Time
count   67.000000   67.000000    67.000000
mean    55.059701    3.472687  1087.478891
std      2.718515    0.442389   283.687708
min     52.000000    2.730000   439.125000
25%     53.000000    3.320000   927.000000
50%     55.000000    3.580000  1060.750000
75%     56.000000    3.730000  1231.250000
max     62.000000    4.000000  1798.142857
```

10~20대와 30~40대 간 Regression 결과의 차이가 크지 않은 것은 30~40대 데이터의 대부분이 30대 초반이기 때문인 것으로 추정됨

두더지 게임의 난이도 모델링 검증

Fitts' Law 식

$$T = a + b \cdot ID = a + b \cdot \log_2(\frac{2D}{W})$$

거리 D는 밑이 2인 lognormal 분포를 따름

→ $\log_2 D \sim N(\mu, \sigma^2)$

→ $T \sim N(a + b + b\mu - b\log_2 W, (b\sigma)^2)$

10~20대의 경우, $\mu = 8, \sigma = 0.3, W = 64$로 설정

→ $T \sim N(a + 3b, (0.3b)^2)$

→ T의 pdf $f(T) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}(\frac{T-\mu}{\sigma})^2}$

두더지 표시 시간 $T' \sim U(t1, t2)$

→ T′의 pdf $f(T') = \frac{1}{t2-t1}$

5-2. Whack-a-mole

두더지 게임의 난이도 모델링 검증

- T'과 T는 independent

→ T'과 T의 joint pdf $f(T', T)$

$$= \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}(\frac{T-\mu}{\sigma})^2} \cdot \frac{1}{t2 - t1}$$

- 성공률 = $P(T' > T)$

$$= \int_{t1}^{t2} \int_{0}^{T'} f(T', T) dT dT'$$

→ 모델링된 모수를 적용하여 도출된 성공률 식을 Matlab을 이용해 적분한 결과, 0.8에 가까운 결과를 얻음

→ 실험을 통해 구한 Fitts' law 식에 기반하여 두더지 게임의 난이도 모델링이 적절하게 이루어졌음을 알 수 있음

func	1x1 sym
mu	646
s	0.8032
sigma	33
t1	600
T1	1x1 sym
t2	840
T2	1x1 sym

```
>> func = exp((-1/2)*((T1-mu)/sigma)^2)/((2*pi)^(1/2)*(t2-t1)*sigma)

func =

(137438953472*exp(-(T1/33 - 646/33)^2/2))/2728506265124189

>> int(int(func, T1, [0 T2]), T2, [t1 t2])

ans =

(49478023249920*2^(1/2)*pi^(1/2)*erf((323*2^(1/2))/33))/248046024102199 - (20615843

>> s = (49478023249920*2^(1/2)*pi^(1/2)*erf((323*2^(1/2))/33))/248046024102199 - (2

s =

    0.8032
```

5-2. Whack-a-mole

두더지 게임 실험 결과

	Age	Score
count	39.0	39.000000
mean	0.0	83.213675
std	0.0	10.603703
min	0.0	57.500000
25%	0.0	77.000000
50%	0.0	85.333333
75%	0.0	92.000000
max	0.0	97.000000

10~20대

	Age	Score
count	15.0	15.000000
mean	1.0	81.833333
std	0.0	6.610778
min	1.0	71.000000
25%	1.0	76.000000
50%	1.0	82.000000
75%	1.0	87.750000
max	1.0	90.000000

30~40대

	Age	Score
count	14.0	14.000000
mean	2.0	80.619048
std	0.0	8.838636
min	2.0	60.000000
25%	2.0	76.125000
50%	2.0	80.833333
75%	2.0	85.875000
max	2.0	97.000000

50~60대

[플레이 인원 수]

- 10~20대: 39명
- 30~40대: 15명
- 50~60대: 14명

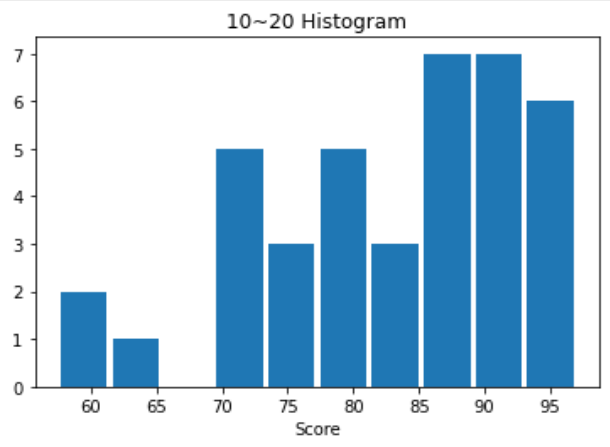
[평균 성공률]

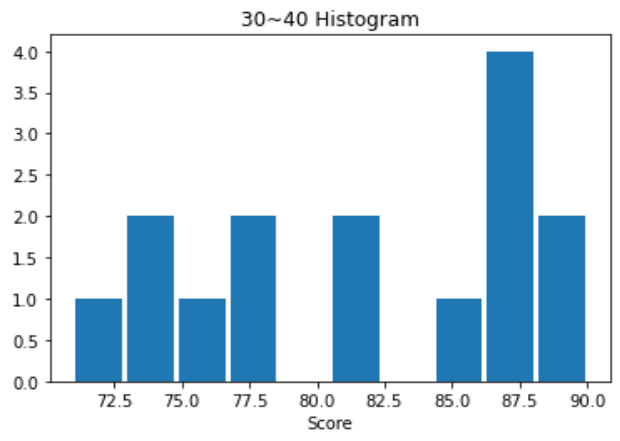
- 10~20대: 83.2%
- 30~40대: 81.8%
- 50~60대: 80.6%

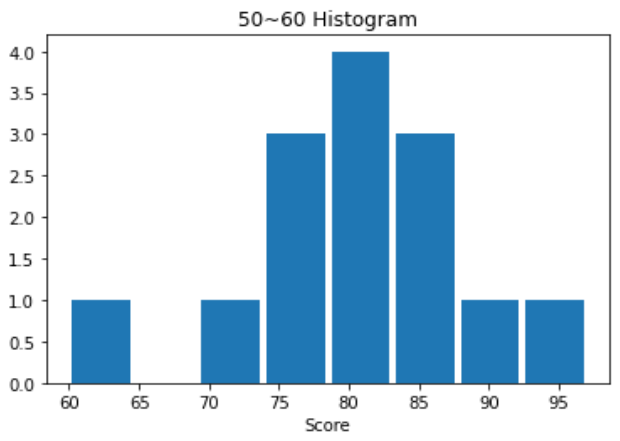
→ 평균 성공률은 목표했던 80% 전후로 나왔음을 확인 가능

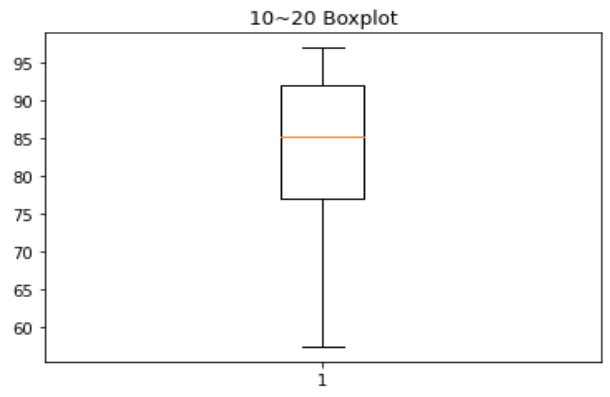
5-2. Whack-a-mole

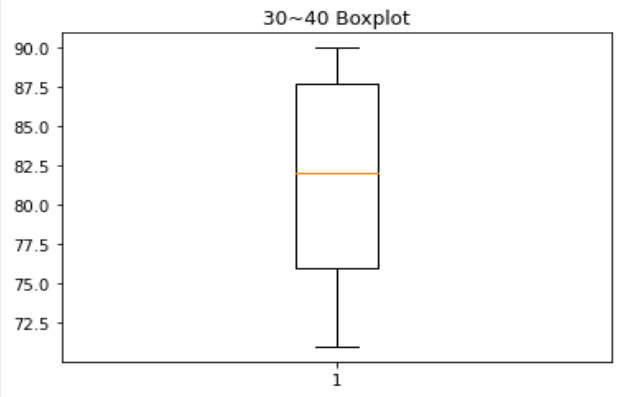
두더지 게임 실험 결과

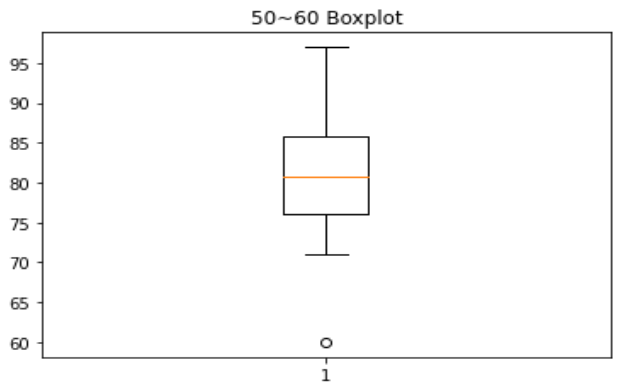












앞서 Regression에서 낮은 R^2값을 통해 예측할 수 있듯이, 실험 결과에서 큰 편차를 확인할 수 있음

6

Conclusion &Takeaways

6. Conclusion &Takeaways

실험 결과와 가설 비교

Research hypothesis : 버튼의 크기와 거리를 조절하여 연령별로 게임 난이도를 조절하여 적정 성공률을 이끌어낼 수 있다.
Fitts' task 실험 : 연령대별로 Fitts' task 수행에 차이가 있어서 연령대별 Fitts' law 식을 다르게 구할 수 있다.
Whack-a-mole 실험 : Fitts' law 모수를 기반으로 난이도를 모델링하여 연령대별 80%의 성공률을 도출할 수 있다.

Fitts' task 실험 : 연령대별로 Fitts' task 수행에 차이가 있어서 연령대별 Fitts' law 식을 다르게 구할 수 있다.
= > 연령대별로 수행에 차이가 존재했고, 연령대별 Fitts' law 식을 다르게 구할 수 있었지만, 실험 수행 결과의 개인 편차가 나타났다.
Whack-a-mole 실험 : Fitts' law 모수를 기반으로 난이도를 모델링하여 연령대별 80%의 성공률을 도출할 수 있다.
= > 평균 성공률이 타깃으로 설정한 80%에 근접하게 나오는 것을 확인할 수 있었으나, 이 또한 연령대가 비슷하더라도 실험 수행 결과의 편차가 나타났다.

6. Conclusion &Takeaways

실험 결과의 가치와 실험 결과로 얻어갈 점

앞서 Evaluation에서 보았듯이, 연령대가 증가할수록 버튼 클릭에 대한 **수행 능력이 뚜렷하게 감소하는 경향성을 발견**할 수 있었다. 아울러 이에 대한 실험 결과를 두더지 게임에 적용했을 때, **평균 성공률이 타깃으로 설정한 80%에 근접하게 나오는 것을 확인**할 수 있었다.

이로 미루어 보아, **연령대별로 버튼 클릭에 대한 수행 능력을 고려하여 게임 난이도를 달리 설정하는 것은 유의미**함을 알 수 있다.

6. Conclusion & Takeaways

실험 결과의 한계점과 이유 분석

그러나, 앞선 Evaluation의 낮은 R^2값은 연령대가 비슷하더라도 실험 수행 결과의 편차가 크다는 것을 나타낸다. 이 말은 즉 버튼 클릭에 대한 수행 능력은 **연령대 외에도 다른 요소들(게임 플레이 빈도, 컴퓨터 활용 빈도 등)에 의해서도 크게 영향**을 받음을 의미한다.

연령대가 가장 큰 영향을 미치는 요소일 것이라고 판단하고 가설을 세웠지만,
연령대만을 **단일 요소로 게임 난이도를 설정하는 경우** 연령대 외 수행 능력에 영향을 미치는 다른 요소들이 무시되어, 적절한 난이도 책정이 이루어지지 않을 수 있다.

감 사 합 니 다

Team 3