一般社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
ET2024-75(2025-03)

# プログラム理解問題とバグ発見問題における視線運動の比較調査

樋口　美咲†　　槇原絵里奈†　　吉田　則裕†

† 立命館大学
〒567-8570 大阪府茨木市岩倉町 2-150
E-mail: †is0632fi@ed.ritsumei.ac.jp, ††{makihara,norihiro}@fc.ritsumei.ac.jp

**あらまし**　プログラミング学習者にとって，プログラム理解とデバッグ能力の向上は重要である．これらプログラミング基礎スキルの依存性を調査するため，本研究ではプログラム理解時とバグ発見時の視線傾向を比較調査し，タスク間の視線関係を分析した．理解問題とバグ発見問題それぞれで視線傾向を正誤グループごとに調査し，理解時のグループ分けをバグ発見時に適用した結果，タスク間およびグループ間で視線傾向が異なることが確認された．
**キーワード**　視線分析，プログラム理解，デバッグ，プログラミング教育

# A comparative study of eye movements in program comprehension task and error finding task

Misaki HIGUCHI†, Erina MAKIHARA†, and Norihiro YOSHIDA†

† Ritsumeikan university
2-150, Iwakuracho, Ibaraki-shi, Osaka, 567-8570
E-mail: †is0632fi@ed.ritsumei.ac.jp, ††{makihara,norihiro}@fc.ritsumei.ac.jp

## 1. はじめに

近年，様々な分野でデジタル化が進み，IT 人材確保に向けたプログラミング教育の需要が高まっている．[1] しかし，プログラミング教育が行われている場において，一人の指導者が担当する学習者の人数は多いことほとんどである．そのため，指導者は学習者がどのように問題を解き，どこで躓いたのかを把握することが難しい．そこで，対象者の思考過程を知る手段の一つとして，視線の動きを用いる方法が挙げられる．

既存研究では，視線の推移状況について，プログラムのデータ依存関係の影響を確かめることで，プログラムの読解過程を視線から分析することが可能であることを示している．[2] さらに，視線の停留運動について，機械学習を用いることで，初学者と熟練者を区別することができることを示した研究もあり，視線データは対象者の思考過程を調査し，その能力を判断するために役立つツールの一つであるといえる．[3]

プログラミング学習を行う上で，身に着けるべき重要な能力としてプログラム理解能力とデバッグ能力が挙げられる．プログラム理解は，その他のプログラムに関する技能全てにかかわるスキルであり，デバッグは，初心者のプログラマが学ぶのが難しく，コンピュータサイエンスの教育者が教えるのが難しい重要なスキルである．[4] 実際，先行研究によると，視線をプログラミング教育に活かした研究のほとんどは，プログラムの理解かデバッグを対象としている．[5] しかし，これらの先行研究のほとんどは対象がプログラム理解とデバッグのどちらかに限定されており，異なるタスク間において，視線の動きに関係性があるのかどうかは調査がされていない．タスク間の関連性を調査することで，教育の効率化や，学習者モデルの構築に繋がると考えられる．

そこで本研究では，学習者の問題を解く際の思考過程を調査することを目的とし，プログラム理解時とデバック時の視線傾向について関係性の調査を行う．具体的には，まず，プログラム理解問題とバグ発見問題それぞれでの視線の傾向について，解答グループごとに調査を行い，最後に理解時とデバック時の視線傾向の比較調査を行う．

本論文の構成を以下に示す．2 章では実験の対象とした問題と視線計測技術について，用語と関連研究について述べる．3 章では本研究で行った実験の概要を述べ，4 章では調査方法について説明する．5 章では実験結果と考察について述べ，最後に 6 章でまとめとする．

## 2. 準　　備

### 2.1 視線計測技術について関連技術と関連研究

#### 2.1.1 アイトラッカ

一般的に，視線座標と時間を記録にはアイトラッカが使用される．アイトラッカにはウェアラブル型とスクリーンベース型の 2 種類がある．ウェアラブル型のアイトラッカはパソコンの画面に限らず広い範囲で視線データの計測が可能である利点がある一方で，高価で導入が難しいという欠点がある．据え置き型のアイトラッカには，観測データがパソコンの画面上に限るという欠点があるが，ウェアラブル型と比較すると安価で導入がしやすい．本研究では，調査対象をプログラミング問題の読解やバグ発見としており，パソコンの画面上に表示するため，スクリーンベース型のアイトラッカを使用している．

#### 2.1.2 視 線 運 動

視線運動は対象を捉え続ける運動である停留と，停留から次の停留に移動するまでの動きであるサッカードの 2 種類に分けられる [6]．停留時間や回数を分析することにより対象への注目度や難易度を，サッカードに注目することで前後の対象の関係性や，全体を通した読解パターンが調査可能である [5]．そのため，アイトラッカを用いて得られた視線の時間と座標データについて，分析を行うためにまず行われることは停留運動の算出である．停留の算出の際に課題となるのはノイズである．ノイズが生じると，実際に見ていた対象との差異が生じる可能性があるため，停留算出の際にはノイズの除去が重要となる [6]．そこで，本研究ではノイズに強いとされているアルゴリズムである I2MC を採用して停留データの算出を行った [7]．

### 2.2 関連研究と課題点

視線をプログラミング教育に活かした研究は多数ある．たとえば，松本らや Maike らは記録した視線をヒートマップで可視化し，表示するシステムを提案した [8], [9]．これにより，視線を記録することで対象者の注視場所を追跡可能であることが確認されている．また，視線はそれ単体だけでなく，他のデータと組み合わせることでも調査がされている [10]．曾我らの研では視線の停留と心拍を組み合わせた心理特性とプログラム特性を提案し，それによりプログラムの理解状況を推定できることを示している [11]．

視線に対して，機械学習を用いて分類した研究は他にもある．Unaizah らは，視線収集の対象をソースコードではなく，擬似コードとした．アプリオリアルゴリズムを用いた視線パターンの算出と階層的クラスタリングを用いたグループ分けを行っている．結果として，対象者は出力の例をよく見ていることの他，問題の難易度が増えると逆行の回数が増加することを確認した [5]．また，Lianzhen らはエラー発見問題における視線の遷移に注目した．プログラムの実行順との一致した比率を測定する指標が対象者のスコアを分析することに役立つことを確認したほか，コードの読解とエラーの発見に明確なフェーズはなく，同時に行っていることを発見した [12]．また，馬場らはデバッグの対象を組込みシステム開発としており，視線データを活用することで初学者と熟練者の差を調査した [13]．この際注目物体の時間分布や遷移確率を用いた比較が行われており，本研究でもタスク間の比較を行うため，馬場らの手法を採用した．

以上より，視線データは様々な形でソースコードリーディングやデバッグに活用されている．その一方で，先行研究では調査の対象がプログラムの読解かデバッグのどちらかに限定されており，異なるタスクにおいて，視線の動きに依存性があるのかどうかは調査がされていない．また，先行研究では読解対象がソースコードのみである場合も多く，プログラムの内容が書かれた文章との関係性についてもほとんど調査が行われていない．実際にプログラミングの演習を行う際には，問題からプログラムの内容や入出力についての情報を確認しながらコードの作成を行う．コードリーディングとデバッグと言った，主要なプログラミングスキルの関係性を調査することで，より効率的なプログラミング学習や教育に繋げられる可能性がある．そこで本研究では，コード内容とソースコードの両方を表示したうえで，プログラム理解とバグ発見時の視線情報を収集し，比較調査を行う．

## 3. 実 験 概 要

### 3.1 実 験 目 的

本研究の目標はプログラム理解時とバグ発見時の視線傾向について関係性があるかを調査することである．本目的を達成するために，以下 3 つのリサーチクエスチョン (RQ) を設定した．

**RQ1: プログラム理解時の視線について解答グループごとに違いはあるか?**

**RQ2: バグ発見時の視線について解答グループごとについて解答グループごとに違いはあるか?**

**RQ3: プログラム理解時とバグ発見時の視線について関係性はあるか?**

まず，RQ1, RQ2 でプログラム理解時とバグ発見時の視線について，それぞれの解答結果をもとに調査を行い，RQ3 でプログラム理解時とバグ発見時の視線を比較調査する．

### 3.2 被　験　者

実験には 10 名の情報系学生が参加した．そのうち 2 名はプログラム理解問題にしか参加していないため，プログラム理解問題とバグ発見問題の両方に参加したのは 8 名だった．参加者全員が 1 年以上のプログラミング経験を持っており，そのうち 4 名が 4 年以上と回答した．参加者のうち 5 名が AtCoder(注1)や Aizu Online Judge(AOJ)(注2)などで提供されているプログラミング問題に取り組んだことがあると回答したが，このうち日常的，あるいは月に数回は取り組むと回答したのは 3 名だった．

### 3.3 実 験 環 境

実験には，解像度 1920×1200 のノートパソコンを使用し，アイトラッカを図 1 のようにパソコンの下部に取り付けてデータを収集した．アイトラッカにはサンプルレートが 60Hz の Tobii Pro Spark(注3)を使用しており，アイトラッカとの接続に

(注1)：https://atcoder.jp/
(注2)：https://judge.u-aizu.ac.jp/onlinejudge/
(注3)：https://www.tobii.com/ja/products/eye-trackers/screen-based/tobii-pro-spark

は iTrace core(注4) を使用した．プログラム理解問題とバグ発見問題の表示には，後述する著者が作成した web サイトを使用した．使用したフォントは全て 16px 以上で，実験の初めにキャリブレーションを行い，正常に動作することを確認した．

### 3.4 対象問題

実験の対象となる問題は，著名なオンラインジャッジシステムの 1 つである AtCoder から選出した．AtCoder では，AtCoder problems(注5) において，有志らによる問題難易度の公開が行われている．その中から難易度や問題の文章量，解答コードの長さが類似する 2 つの問題をプログラム理解問題(注6) とバグ発見問題(注7) に選定した．

### 3.5 実験手順およびタスク

実験は，次に示す手順で行った．なお，全てのタスクに時間制限は設けていない．

手順 1: 被験者に関するアンケートの実施．
手順 2: 各要素の配置とサイトの操作方法，問題の形式を説明するためのプレ問題の実施．
手順 3: プログラム理解問題の 1 つ目のタスクを実施．画面左側に問題内容，右側にコードを表示した画面を提示し，表示された内容の理解を行わせた (タスク 1)．
手順 4: プログラム理解問題の 2 つ目のタスクを実施．実際の理解状況を確かめるため，指定した入力に対してコード実行後の配列変数の値と出力結果を解答させた (タスク 2)．
手順 5: バグ発見問題を実施．プログラム理解問題と同様の配置の画面を提示し，コードに 1 行含まれたエラーを各行の横にあるチェックボックスをクリックすることで解答させた (タスク 3)．

### 3.6 注目領域の設定

各停留データについて，座標から見ていた領域を確認するため，画面上を図 2 のように分割して注視エリアとして設定した．コードについては，プログラム理解問題とバグ発見問題でそれぞれ内容ごとに 4 つに分割してエリアを設定した．なお，タイトルその他のエリアに関する停留と遷移は本調査における目的に一致しないため，調査の対象外とした．


**図 1:** 実験の様子

(注4)：https://www.i-trace.org/index.html
(注5)：https://kenkoooo.com/atcoder/#/table/
(注6)：https://atcoder.jp/contests/abs/tasks/abc088_b
(注7)：https://atcoder.jp/contests/abc159/tasks/abc159_b

## 4. 調査方法

設定したエリアに対し，サッカードと停留において解答グループごとにどのような違いがあったのかを調査する．

### 4.1 サッカード

サッカードについては，4 つの点において調査を行った．

- 総遷移割合
- エリアを跨いだ遷移割合
- 問題部とコード部を跨いだ遷移割合
- 遷移があったエリアの数

総遷移割合では，サッカードが多かったエリア間について調査する．サッカードは同一エリア内で確認される場合と，異なるエリア間に対して確認される場合の 2 つがある．そこで，異なるエリア間に跨ったサッカード数のうち，各エリア間に対して遷移が行われていた割合を算出する．例えば確認されたすべてのサッカードが，問題文と問題文，問題文とコード A，問題文とコード A，入出力内容と入出力例であった場合，問題文とコード A に対する遷移割合は 2/3，入出力内容と入出力例に対する遷移割合は 1/3 と算出される．

エリアを跨いだ遷移割合では，総サッカード数に対して，エリアを跨いだサッカード数が存在する割合を算出して調査する．

次に，問題部とコード部を跨いだサッカード数が存在する割合を調査する．この際，総サッカード数に対する割合，エリアを跨いだサッカード数に対する割合の 2 種類を算出した．

次に，遷移があったエリア間の数に違いはあったのかを調査する．例えば確認されたすべてのサッカードが先述した総遷移割合のものであった場合，遷移があったエリア間は，問題文とコード A，入出力内容と入出力例の 2 種類と算出される．

### 4.2 停留

停留については，次の 3 つの点で調査を行った．

- 各エリアの閲覧時間割合
- 同一エリアへの平均連続停留時間
- 各コードエリアにおける停留時間割合

各エリアの閲覧時間割合では，エリアに対する注目度の違いを調査する．各エリアに対して確認された停留の合計時間を，各エリアの閲覧時間と定義する．全エリアの閲覧時間の合計に対して，各エリアの閲覧時間の割合を算出した．

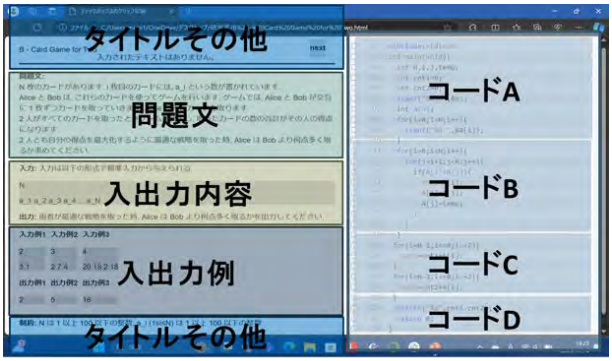



**図 2:** プログラム理解問題のエリア設定

各エリアの平均連続閲覧時間では，各エリアに対して，同一エリアを連続して見る時間に違いはあったのかを調査する．導出は各エリアの閲覧時間を各エリアの合計停留数で割ることで算出した．例えば，問題文を連続して 1 分見た後，コードを 30 秒確認し，もう一度問題文を 2 分見た場合，問題文に対する平均連続閲覧時間は 1 分半である．

各コードエリアにおける停留時間割合では，コードを見ていた閲覧時間に対して，各コードエリアに対して閲覧が行われた時間の割合を算出した．

## 5. 実験結果と考察

### 5.1 RQ1 プログラム理解時の視線について解答グループごとに違いはあるか?

#### 5.1.1 解答結果を基にしたグループ分け

プログラム理解問題について，参加した 10 名をタスクの解答結果によるグループ分けを行った．その結果，配列変数の値と出力の結果すべてに正解していた 3 名と，配列変数の値は間違えていたが出力結果は正解していた 4 名，両方を間違えていた 3 名に分類された．今後このグループについて，順に完答グループ，半答グループ，誤答グループと呼ぶ．完答グループでは，全員がオンラインジャッジの問題を解いた経験があり，日常的，あるいは月数回は解くと回答した人が 2 名いた．半答グループは 4 名中 2 名がプログラミング問題を解いた経験があり，1 名が月に数回は解くと回答している．誤答グループは全員がプログラミング問題を解いた経験がないと回答した．

3.5 節で示したタスク 1 とタスク 2 について，それぞれかかった時間をグループごとに計算した．結果，理解にかかった時間については，平均した値に大きな差は見られなかった．その一方で，解答にかかった時間の平均については，完答グループは 1 分 12 秒，半答グループは 1 分 9 秒，誤答グループは 4 分 10 秒であり，完答および半答グループと，誤答グループの間に 3 分半以上の大きな差がみられた．以上より，誤答グループの参加者は提示された問題について，理解内容を間違えていたため誤答したのではなく，理解したと誤認をしていた可能性が考えられた．また，半答グループは完答グループとほとんど変わらない時間で回答を終えていた．半答グループは出力結果については正答していたが，配列変数の値を誤答したグループである．したがって半答グループは，提示された問題の内容自体は理解できていたが，コードの詳細を見落としていた可能性が高いと考えられた．

以降の調査結果について，タスク 1 とタスク 2 において画面構成が一部変更するため，タスク 2 とタスク 3 の比較が難しい．しがたって，プログラム理解に関しては被験者が長時間閲覧したタスク 1 における結果を示す．

#### 5.1.2 サッカード

総遷移割合について上位 10 件をグループごとに算出した結果が表 1 である．まず，上位 3 件については登場するエリアが，入出力内容と入出力例，入出力内容と問題文の遷移と，全グループで共通していることを確認した．したがって，入出力内容について，例や問題文と見比べながら理解していた参加者が大多数いたことが考えられる．その他の遷移については，完答グループと誤答グループは上位 10 件のうち 8 件が共通しており，同じ遷移が多いことを確認した．両グループにおいて異なっていたのは入出力内容からの遷移先であり，完答グループは入力内容が変数に格納される処理が行われているコードＡに遷移しているのに対して，誤答グループは格納された変数について並べ替えなどの処理が行われているコードＢに遷移していたことが分かった．また，半答グループについては，上位数件以外の相似部分があまり確認されなかった．このグループは 5 件目以降数値が他のグループと比較して低く，幅広いエリアを遷移していた可能性が考えられた．

**表 1:** プログラム理解問題における総遷移割合

(a) 完答グループ

| 遷移名 | 割合 |
|---|---|
| **入出力内容-入出力例** | 0.12 |
| **入出力例-入出力内容** | 0.10 |
| **問題文-入出力内容** | 0.09 |
| コード A-コード B | 0.08 |
| 入出力内容-問題文 | 0.07 |
| コード B-コード C | 0.05 |
| コード B-コード A | 0.05 |
| 入出力例-問題文 | 0.05 |
| 入出力内容-コード A | 0.04 |
| 問題文-入出力例 | 0.04 |

(b) 半答グループ

| 遷移名 | 割合 |
|---|---|
| **入出力内容-問題文** | 0.13 |
| **入出力例-入出力内容** | 0.12 |
| **入出力内容-入出力例** | 0.12 |
| 問題文-入出力内容 | 0.12 |
| 問題文-入出力例 | 0.05 |
| 入出力例-問題文 | 0.04 |
| コード A-問題文 | 0.03 |
| 問題文-コード A | 0.03 |
| コード A-コード B | 0.02 |
| コード B-コード C | 0.02 |

(c) 誤答グループ

| 遷移名 | 割合 |
|---|---|
| **入出力例-入出力内容** | 0.14 |
| **入出力内容-入出力例** | 0.12 |
| **問題文-入出力内容** | 0.09 |
| 入出力内容-問題文 | 0.08 |
| コード A-コード B | 0.05 |
| コード B-コード C | 0.04 |
| コード B-コード A | 0.04 |
| コード C-コード B | 0.04 |
| 入出力内容-コード B | 0.03 |
| 問題文-入出力例 | 0.03 |

**表 2:** 問題部分とコード部分を跨いだ遷移の割合

| 分母 | 完答 | 半答 | 誤答 | 全員 |
|---|---|---|---|---|
| 総サッカード数 | 0.037 | 0.062 | 0.045 | 0.049 |
| エリアを跨いだサッカード数 | 0.247 | 0.226 | 0.216 | 0.229 |

また，エリアを跨いだ遷移の割合について，完答グループは 15%，半答グループは 25%，誤答グループは 21% であった．半答グループはエリアを跨いで遷移を行うことが最も多く，完答グループは同じエリア内で遷移を行っていることが最も多いことが分かった．

特に問題部分とコード部分を跨いだ遷移の割合について，結果は表 2 のようになった．結果として，総サッカード数に対しては，半答グループが最も遷移している割合が高くなった．また，分母をエリアを跨いだサッカード数にした場合，どのグループもほとんど同じ割合で問題部分とコード部分を遷移していることが分かった．これは，完答グループのエリアを跨ぐサッカード数の少なさと，半答グループが問題部分とコード部分に限らず，異なるエリア間を遷移していたことが多かったことが影響したと考えられる．

遷移があったエリアの数について，完答グループは 22.0，半答グループは 30.0，誤答グループは 32.7 であり，完答，半答，誤答の正答率の順に数値が増加した．

以上より，遷移の面では解答グループごとに異なる動きを確認した．完答グループと誤答グループは，総遷移割合で確認した上位 10 件のうち 8 件が一致するなど、比較的似た傾向が確認されていたが，実際に遷移があったエリアを比較すると，完答グループの方が遷移があったエリアが少なかったことが分かった．したがって，完答グループには余計な遷移が少なく，効率的に

内容の理解を行っていたことが考えられる．また，半答グループは完答，誤答グループとは異なり，問題部分からコード部分へと視線が切り替わるタイミングが確認されなかった．実際，半答グループは総サッカード数に対して，問題部とコード部を跨いだサッカードが他グループより多かったが，半答グループの被験者は問題部とコード部に限らず，別のエリアを遷移することも多かった．彼らは出力結果は正解していたのに対して，配列変数の値が間違えていたグループであるため，このことがコードの詳細の見落としにつながった可能性が考えられた．

#### 5.1.3 停　　留

各エリアの閲覧時間割合に対する結果を図 3 に示す．図 3 より，完答グループと誤答グループは共にコードについて注視した割合が多いが，半答グループは他のエリアと比較してコードに注視していないことが分かった．また，入出力例や入出力内容に注目した割合ついて，完答グループは他のグループと比較して低いことが分かった．

次に，平均連続閲覧時間について，結果を図 4 に示す．全グループにおいて，コードに対して連続して注視する時間が最も長い結果となった．特に完答グループは，入出力内容を除き，他グループより各エリアに注視する時間が長い結果となった．一方，半答グループは図 3 より問題文に対する閲覧時間割合が高かったにもかかわらず，図 4 より連続して見る時間は短くなっていたことを確認した．

以上より，停留の面ではグループごとに異なる点が観測された．エリアごとの閲覧時間割合において，完答グループは他のグループと比較して入出力例や入出力内容を見る割合が低かったことを確認した．これは，完答グループは本研究でも採用し

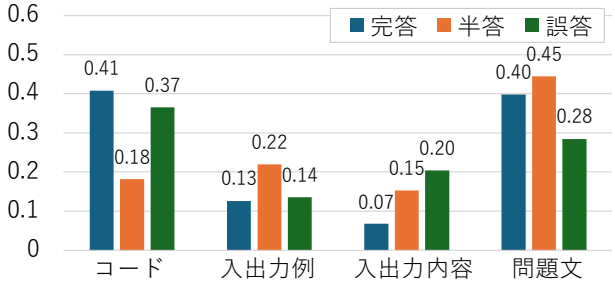



**図 3:** プログラム理解問題の各エリアにおける閲覧時間割合

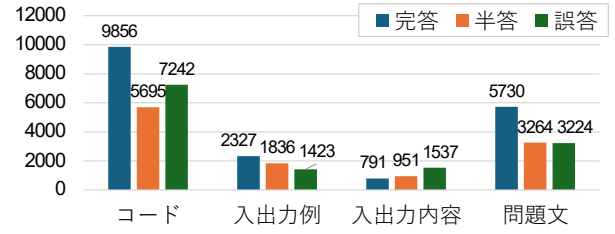



**図 4:** プログラム理解問題の各エリアにおける平均連続閲覧時間

**表 3:** バグ発見問題における総遷移割合

**(a)** 正答グループ

| 遷移名 | 割合 |
|---|---|
| 問題文-入出力内容 | 0.12 |
| 入出力内容-問題文 | 0.12 |
| 入出力内容-入出力例 | 0.07 |
| 入出力例-入出力内容 | 0.06 |
| コード B-コード C | 0.05 |
| コード A-コード B | 0.04 |
| コード C-コード D | 0.04 |
| コード C-コード B | 0.04 |
| 入出力例-問題文 | 0.03 |
| コード D-コード C | 0.03 |

**(b)** 誤答グループ

| 遷移名 | 割合 |
|---|---|
| 問題文-入出力内容 | 0.10 |
| 入出力内容-問題文 | 0.10 |
| コード B-コード C | 0.07 |
| 入出力内容-入出力例 | 0.07 |
| 入出力例-入出力内容 | 0.07 |
| コード A-コード B | 0.06 |
| コード C-コード B | 0.06 |
| コード B-コード A | 0.05 |
| コード C-コード D | 0.04 |
| コード D-コード C | 0.03 |

た時間は短かった，このため半答グループは，他のエリアと比較して確認をしながら問題文を見ていたことが考えられた．また，半答グループは他のグループと比較してコード A を見る時間が長い結果となった．

RQ1 への回答

サッカードと停留の両方において，グループごとに異なる結果となった．完答グループは遷移があったエリアの数が他グループと比較して少なく，効率的に読解を行っていたことが考えられた．半答グループは頻繁にエリア間を遷移しながら読解する傾向があり，コードへの注目時間が短かった．誤答グループは最も遷移があったエリアの数が多く入出力内容を注視する時間が長かった．

### 5.2 RQ2 バグ発見時の視線について解答グループごとの違いはあるか?

#### 5.2.1 解答結果を基にしたグループ分け

バグ発見問題について，参加した 8 名をタスクの解答結果に応じグループ分けした．その結果，正しいエラー箇所を選択した 4 名と，異なる箇所を選択した 4 名に分かれた．今後このグループについては，正答グループ，誤答グループと呼ぶ．正答グループでは，4 名中 3 名がプログラミング問題を解いた経験があり，日常的，あるいは月に数回解くと回答した人が 2 名いた，誤答グループは 4 名中 1 名がプログラミング問題を解いた経験があり，1 名は月に数回は解くと回答している．タスクにかかった時間について，正答グループは平均 6 分 58 秒，誤答グループは平均 7 分 44 秒だった．

#### 5.2.2 サッカード

総遷移割合について，上位 10 件をグループごとに算出した結果を表 3 に示す．順序は異なるものの，上位 10 件中 9 件の内容が両グループで一致した．遷移内容に大きな差は見られず，バグ発見時にはどちらの解答グループも，入出力内容と他の問題部を遷移する視線と，コード内を行き来する視線が多く発生していたことが分かった．

次に，総遷移のうちエリアを跨いだ遷移が行われていた割合について算出した．結果として，エリアを跨いだ遷移の割合は正答グループは 29%，誤答グループは 21% であり，正答グループの方がエリアを跨ぐ遷移が多くなった．

問題部分とコード部分を跨いだ遷移の割合について，結果を

**表 4:** 問題部分とコード部分を跨いだ遷移の割合

| 分母 | 正答 | 誤答 | 全員 |
|---|---|---|---|
| 総サッカード数 | 0.087 | 0.055 | 0.071 |
| エリアを跨いだサッカード数 | 0.30 | 0.25 | 0.27 |

**表 5:** 両問題におけるエリアを跨いだ遷移の割合

| タスク | 完答 | 半答 | 誤答 | 全員 |
|---|---|---|---|---|
| プログラム理解時 | 0.16 | 0.25 | 0.22 | 0.22 |
| バグ発見時 | 0.30 | 0.25 | 0.20 | 0.25 |

表 4 に示す．分母を総サッカード数にした場合と，エリアを跨いだサッカード数にした両方の場合で，正答グループの方が問題部分とコード部分を跨いだ遷移の割合が高くなった．遷移があったエリアの数について，結果は正答グループは 38.3%，誤答グループは 40.0% であった．正答率の高いグループの方が遷移があったエリアの数が少ないという結果は得られたが，大きな差はなかった．

以上より，一部においてサッカードについてグループ間の差が確認された．違いがみられた問題部とコード部を跨いだ遷移やエリアを跨いだ遷移の割合については，どちらも正答グループの方が高く，正答グループはエリア間を比較しながらエラー箇所の発見を行っていたことが考えられる．

#### 5.2.3 停　　留

各エリアの閲覧時間割合を図 5 に示す．誤答グループの方がコードに対する閲覧時間割合が高く，正答グループの方が入出力内容や問題文などのエリアに対する閲覧時間割合が高くなっていることが分かった．平均連続停留時間については，コード以外のエリアについては，1 秒以上の差は確認されず，大きく変わらなかったことが分かった．しかし，コードエリアについてはグループ間で差がみられ，正答グループが 3792ms であったのに対し，誤答グループは 8188ms であった．誤答グループは正答グループの 2 倍以上長く，コードを連続して閲覧していたことを確認した．各コードエリアにおける閲覧時間割合については，グループ間の差はすべてのコードエリアにおいて 5% 以下であった．したがって，どちらのグループもほとんど同じ割合で各コードエリアを見ていたことを確認した．また，コード B，C に対する閲覧時間割合は特に高く，どちらのグループもコード B は 38% 以上，コード C は 29% 以上の割合で閲覧していた．被験者がコード B，C を中心にエラー箇所を探していたことが確認された．

以上より，停留の面では特にコードに対する結果について，グループ間で異なる点が確認された．誤答グループはコードに対する注目度が高く，また連続してコードを見る時間も長かった．このことから，誤答グループはエラー箇所について，問題部分との比較によって探すのではなく，コード部分を連続して注視することで探す傾向が考えられる．また，誤答グループは解答までに時間がかかる時間も長かったことから，エラー箇所を探していたため，コードへの閲覧時間が長くなった可能性がある．

RQ2 への回答

サッカードに関する調査では，グループ間の差はほとんど確認されなかった．一方停留に関する調査では，誤答グループは正答グループに比べ，コードに対する閲覧時間割合や連続閲覧時間の値が大きい結果となった．

### 5.3 RQ3 プログラム理解時とバグ発見時の視線について関係性は存在するか?

#### 5.3.1 解答結果を基にしたグループ分け

プログラム理解問題とバグ発見問題の両方に参加した 8 名について，RQ1 におけるプログラム理解時の解答結果を基にしたグループ分けを適用した．結果として，完答グループが 2 名（内バグ発見問題の正答 2 名），半答グループが 4 名（内バグ発見問題の正答 1 名），誤答グループが 2 名（内バグ発見問題の正答 1 名）という結果が得られた．本節では，RQ1 のグループ分けを利用し RQ1 と RQ2 の結果を比較することで，プログラム理解時とバグ発見時の関係性について調査を行う．

#### 5.3.2 サッカード

エリアを跨いだ遷移の割合について，結果を表 5 に示す．各グループ内の変化に注目すると，完答グループはバグ発見時にエリアを跨ぐ遷移の割合が，プログラム理解時に比べ約 2 倍となった．また半答グループは特に変化なく，誤答グループのみわずかに割合が減少していることが確認できた．

問題部分とコード部分を跨いだ遷移の割合について，総サッカード数に対する割合の結果を図 6 に示す．図 6 より，完答グループは，バグ発見時に問題部とコード部を跨ぐ遷移の割合が，プログラム理解時に比べ約 2 倍に増加した．また誤答グループのみ，バグ発見時における割合がわずかに減少した．エリアを跨いだサッカード数に対する割合については，半答グループの

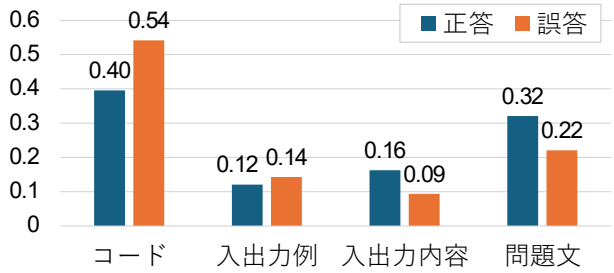



**図 5:** バグ発見問題の各エリアにおける閲覧時間割合

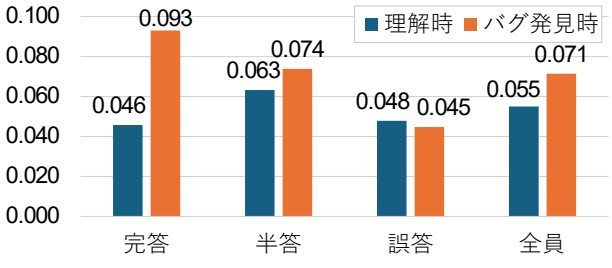



**図 6:** 両問題における総サッカード数に対して問題部分とコード部分を跨いだ遷移の割合

み 0.06 の増加がみられたが，他グループでは 0.01 とほぼ差が無いことが分かった．

遷移があったエリアの数について，結果を図 7 に示す．全ての回答グループにおいて，バグ発見時の遷移があったエリアの数が増加していたことを確認した．また，どちらのタスクにおいても，完答グループは他グループと比べ数値が低かった．半答グループはプログラム理解時にくらべバグ発見時の値が大きく増加したが，誤答グループの変化は非常に小さかった．

以上より，タスク間での視線について，グループごとに異なる変化を確認した．最も多くの変化が確認されたのは完答グループである．完答グループは，プログラム理解時には問題部分を読み進めていてからコード部分の読解に移行していたのに対して，バグ発見時には，様々なエリアを遷移しながら理解とエラー箇所の発見を進めていたと考えられた．一方で，最も数値に変化がみられなかったのは誤答グループであり，これらのことから正答率の高いグループほど異なるタスク間において戦略を変化させる傾向がある可能性がある．また，遷移があったエリアの数については，バグ発見時に全グループで値が増加しており，バグ発見時にはプログラム理解時とは異なる視線傾向となる可能性がある．

#### 5.3.3 停　　留

各エリアの閲覧時間割合について，結果を図 8 に示す．完答グループについて，入出力内容の閲覧割合が高くなった他に大きな変化は見られなかったが，どちらのタスクでもコードと問題文に同等に着目して読み進めていたことを確認した．半答グループについては，注目箇所，すなわち最も値が大きいエリアが，プログラム理解時は問題文であったことに対し，バグ発見時はコードに変化した．誤答グループは入出力例に注目していた割合が減少したが，その他に大きな変化は見られなかった．

各エリアの平期連続停留時間について，結果を図 9 に示す．

完答グループでは，バグ発見時にコードや問題文を連続して見る時間が，プログラム理解時の半分以下となった．半答グループはプログラム理解時からバグ発見時にかけて，問題文を連続して見る時間はわずかに減少したが，コードを連続して見る時間は完答グループとは対照的に 2 倍以上に大きく増加した．誤答グループは，プログラム理解時とバグ発見時で大きな変化は見られなかったが，全グループの中でコードを連続して見る時間が最も長かったことを確認した．

以上より，タスク間での視線について，グループごとに異なる変化を確認した．完答グループについては，閲覧時間割合に対する変化は少なかったのに対して，コードや問題文に対する連続閲覧時間がバグ発見時に大きく減少しており，バグ発見時には各エリアを比較しながら閲覧していたことが考えられる．また，半答グループについては，コードに対する閲覧時間割合と連続閲覧時間の両方が大幅に増加していたことから，コード単体に注目することでエラー箇所を探そうとしていた可能性がある．誤答グループについては，プログラム理解時とバグ発見時でほとんど変化が起こらなかった．すなわち，誤答グループはプログラム理解時とバグ発見時で，問題文やプログラムの読み方が変わらない可能性がある．

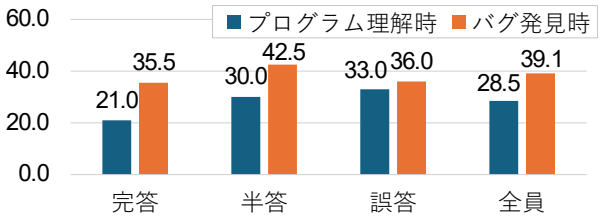
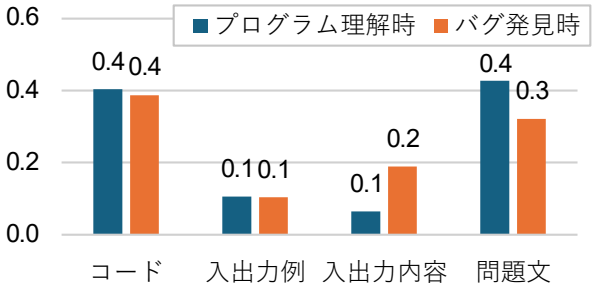
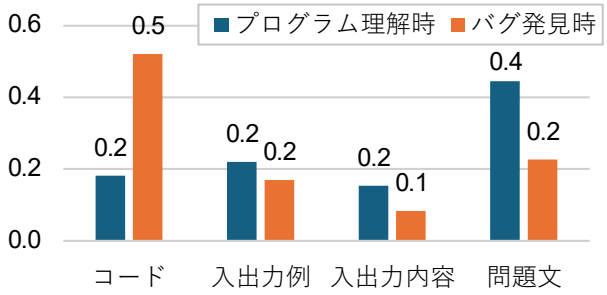


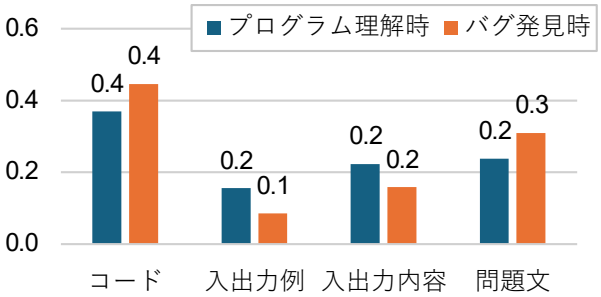

**図 7:** 両問題における遷移があったエリアの数




(a) 完答グループ




(b) 半答グループ




(c) 誤答グループ

**図 8:** 閲覧時間割合

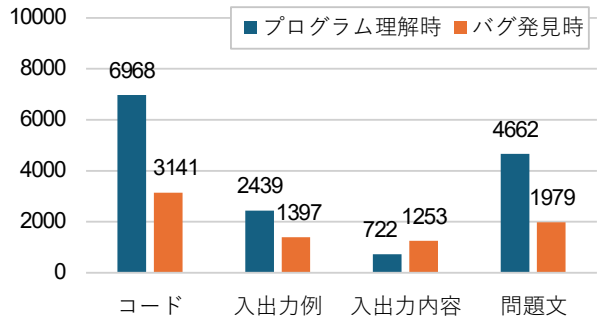
RQ3 への回答

完答グループは，プログラム理解時とバグ発見時で遷移と停留どちらの面でも大きな変化がみられた．このグループは理解時は比べてバグ発見時には各エリアを比較して閲覧する傾向が確認された．半答グループは，コードに関する結果で特に変化が確認された．エラー箇所を発見するために，コードに集中して閲覧する傾向が強くなったことが考えられた．一方，誤答グループはあまり傾向が変化しなかったことから正答率の高いグループほど読み方を変えている可能性が考えられた．

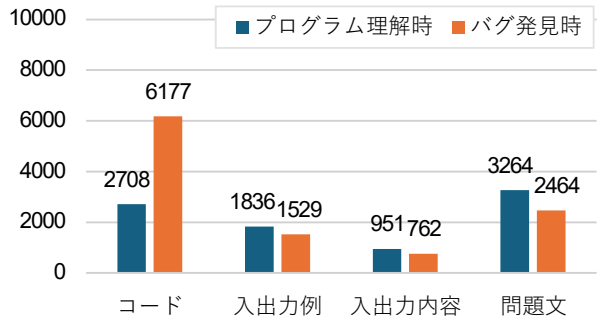
## 6. おわりに

本論文ではプログラム理解時とバグ発見時の視線について比較調査を行った．分析では 3 つの RQ を設定し，被験者をそれぞれ解答結果で分類したあと，停留とサッカードの 2 つの観点から比較調査を実施した．結果として，プログラム理解時に正答率が低かったグループでは，タスク間の視線にあまり違いが観察されなかった一方で，正答率が高かったグループでは，タスク間で視線傾向が変化していることを確認した．

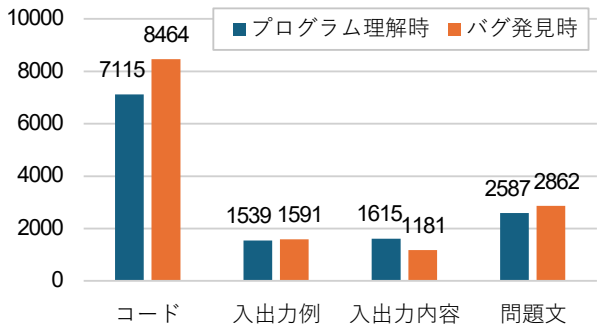
今回の調査では，各エリアに対する視線の注視傾向や，サッカードが観測された前後のエリアを対象として分析を行ったため，全体を通したエリアの遷移については，まだ調査できていない．そのため，今後はグループ間とタスク間において，それぞれ全体を通した遷移傾向にも違いがあったのかについても，調査したいと考える．また，今回比較調査を実施した結果，数値が大きく異なる場合もあれば，わずかにだけ変化がみられた場合も確認した．そのため，今後得られた結果に差があったといえるのか，有意差検定で確かめていきたいと考える．




(a) 完答グループ




(b) 半答グループ




(c) 誤答グループ

**図 9:** 平均連続停留時間

**謝辞** 本研究の一部は JSPS 科研費 24K02923, 24K20905 の助成を受けたものです．

## 文　献

[1] 文部科学省. 令和 5 年度 文部科学白書. 文部科学省, 2023.

[2] 花房亮, 松本慎平, 林雄介, 平嶋宗. 視線運動を用いたプログラム読解パターンのデータ依存関係に基づく分析——代入演算と算術演算で構成されるプログラムを対象として——. 教育システム情報学会誌, Vol. 35, No. 2, pp. 192–203, 2018.

[3] Naser Al Madi, Cole S Peterson, Bonita Sharif, and Jonathan I Maletic. From novice to expert: Analysis of token level effects in a longitudinal eye tracking study. In *Proc. of ICPC*, pp. 172–183. IEEE, 2021.

[4] Renée A. McCauley, Sue Fitzgerald, Gary Lewandowski, Laurie Murphy, Beth Simon, Lynda Thomas, and Carol Zander. Debugging: a review of the literature from an educational perspective. *Computer Science Education*, Vol. 18, pp. 67 – 92, 2008.

[5] Unaizah Obaidellah, Mohammed Al Haek, and Peter C-H Cheng. A survey on the usage of eye-tracking in computer programming. *ACM Computing Surveys (CSUR)*, Vol. 51, No. 1, pp. 1–58, 2018.

[6] Anneli Olsen. The tobii i-vt fixation filter. *Tobii Technology*, Vol. 21, pp. 4–19, 2012.

[7] Roy S Hessels, Diederick C Niehorster, Chantal Kemner, and Ignace TC Hooge. Noise-robust fixation detection in eye movement data: Identification by two-means clustering (i2mc). *Behavior research methods*, Vol. 49, pp. 1802–1823, 2017.

[8] 松本光稀, 若原徹. 視線情報の分析に基づくプログラミング教育支援システムの提案. 第 80 回全国大会講演論文集, Vol. 2018, No. 1, pp. 721–722, 2018.

[9] Maike Ahrens, Kurt Schneider, and Melanie Busch. Attention in software maintenance: An eye tracking study. In *Proc. of EMIP*, pp. 2–9, 2019.

[10] Norman Peitek, Annabelle Bergum, Maurice Rekrut, Jonas Mucke, Matthias Nadig, Chris Parnin, Janet Siegmund, and Sven Apel. Correlates of programmer efficacy and their link to experience: a combined eeg and eye-tracking study. In *Proc. of ESEC/FSE*, 2022.

[11] 曾我遼, 鹿糠秀行, 久保孝富, 石尾隆. 視線と心拍を用いたプログラム理解状況の推定. コンピュータ ソフトウェア, Vol. 40, No. 1, pp. 1_24–1_44, 2023.

[12] Lianzhen Liu, Wei Liu, Xinyu Li, Weiwei Wang, and Wenqing Cheng. Eye-tracking based performance analysis in error finding programming test. In *Proc. of ICCSE*, pp. 477–482, 2020.

[13] 馬場建, 槇原絵里奈, 米田浩崇. 組込みシステム開発のデバッグにおける視線と熟練度の関係分析. 研究報告ソフトウェア工学 (SE), Vol. 2019, No. 9, pp. 1–8, 2019.