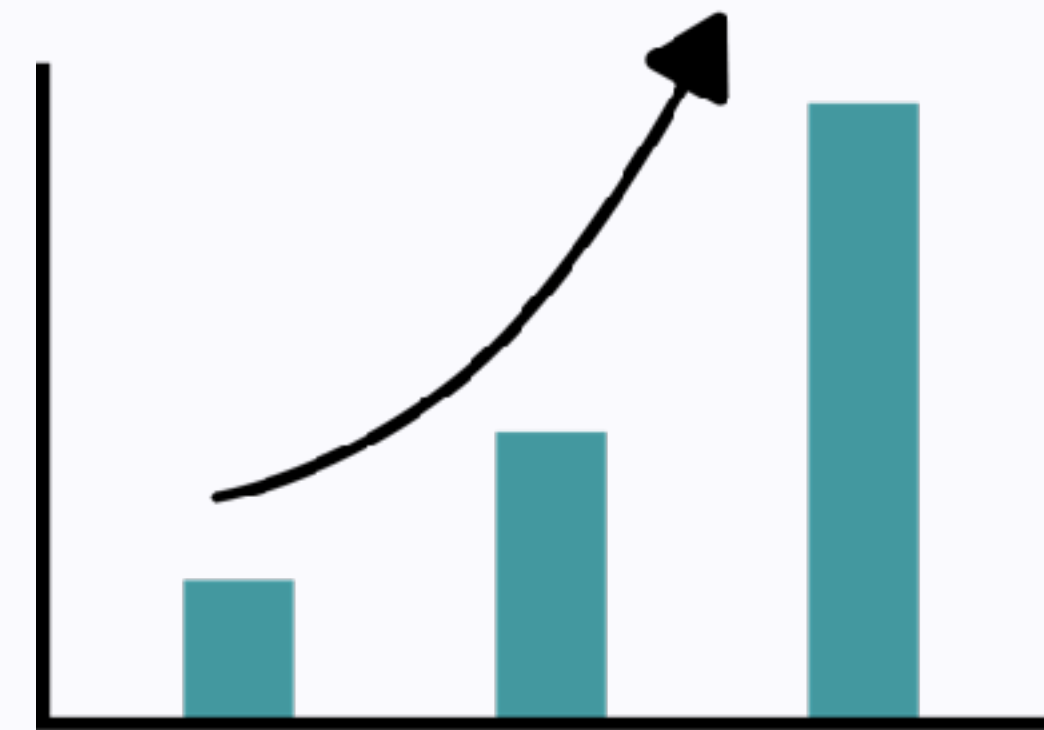
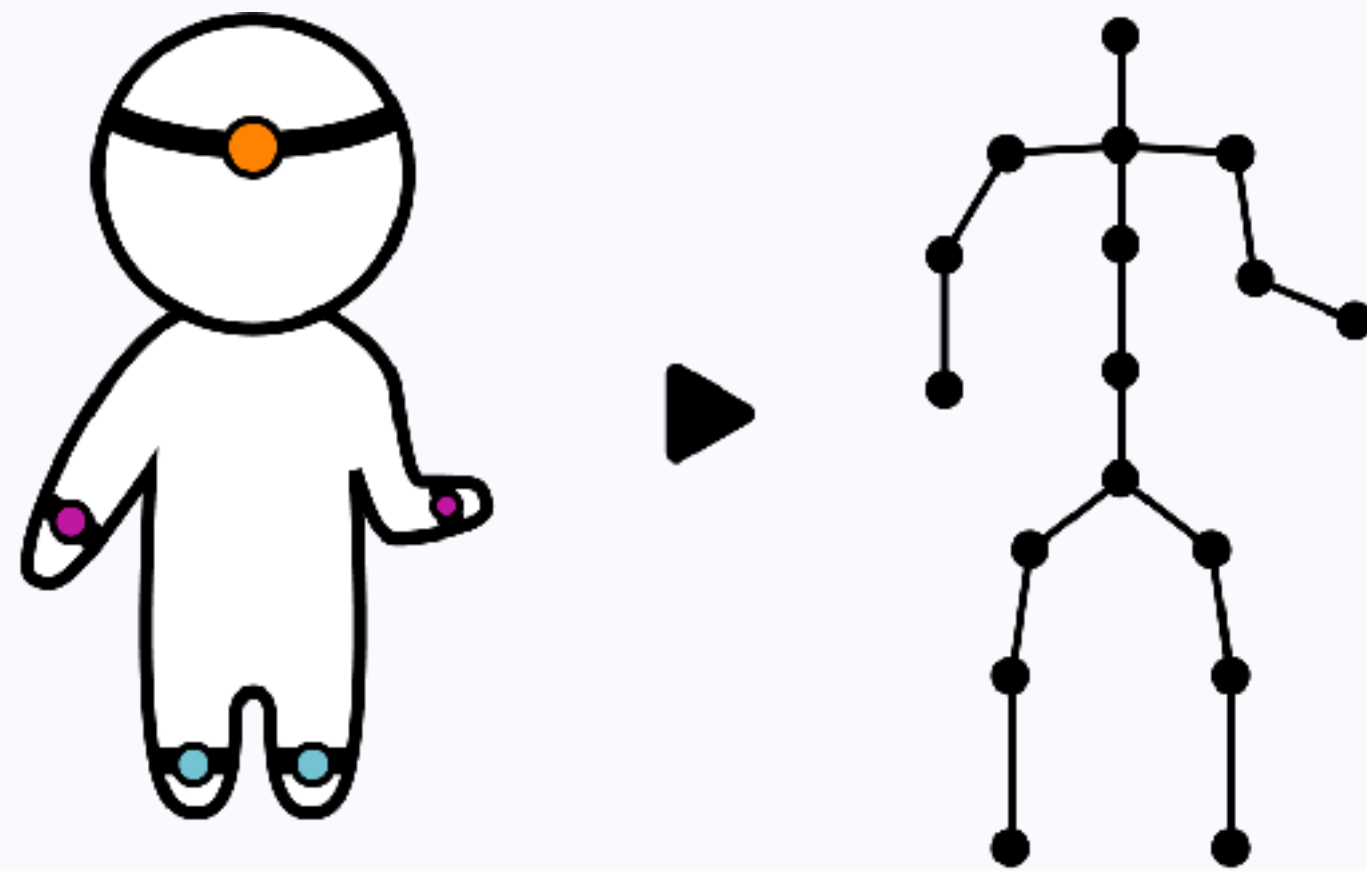


位置情報と手順書を用いた 調理行動の分類精度向上手法

多田隆人(B3)[†] 梶克彦[†] 水野公博[‡] 内藤彩乃[†] 中條直也[†] 水野忠則[†]
[†] 愛知工業大学 情報科学部 [‡] 三菱電機エンジニアリング株式会社

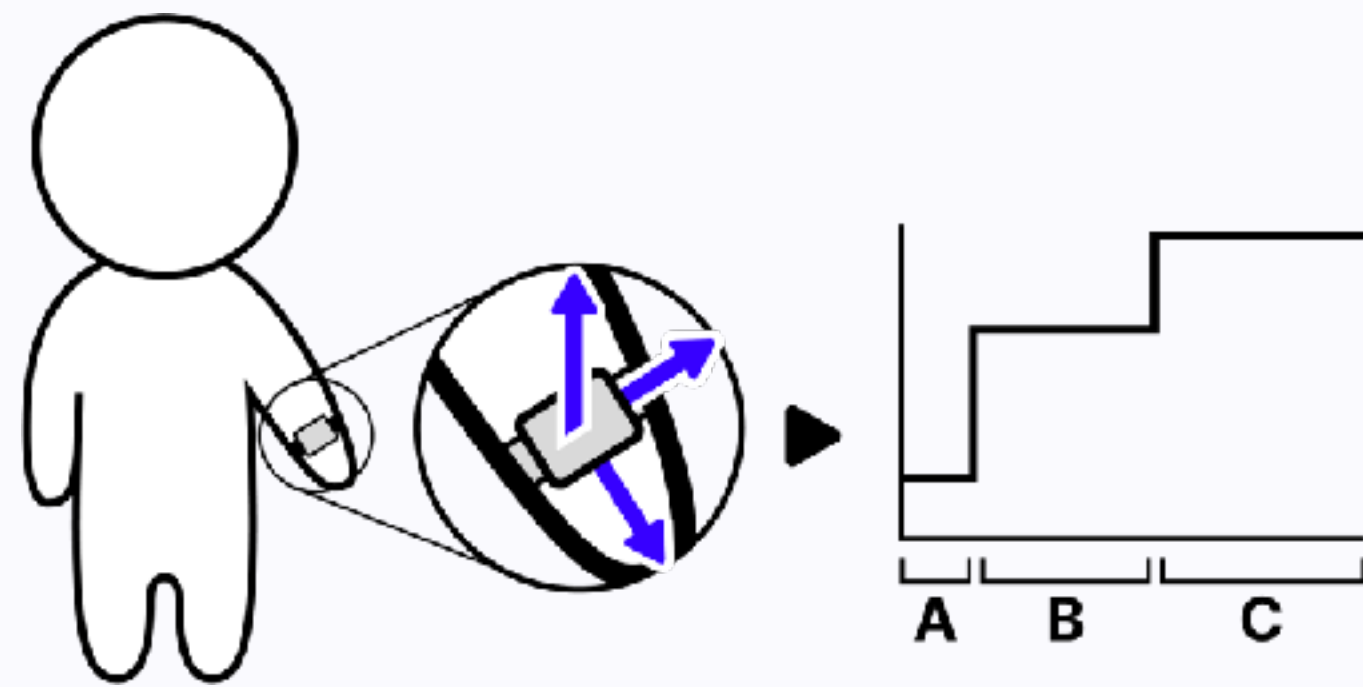
モーションキャプチャにより
手軽に骨格情報を得られる

行動認識により
技能向上の支援が期待できる

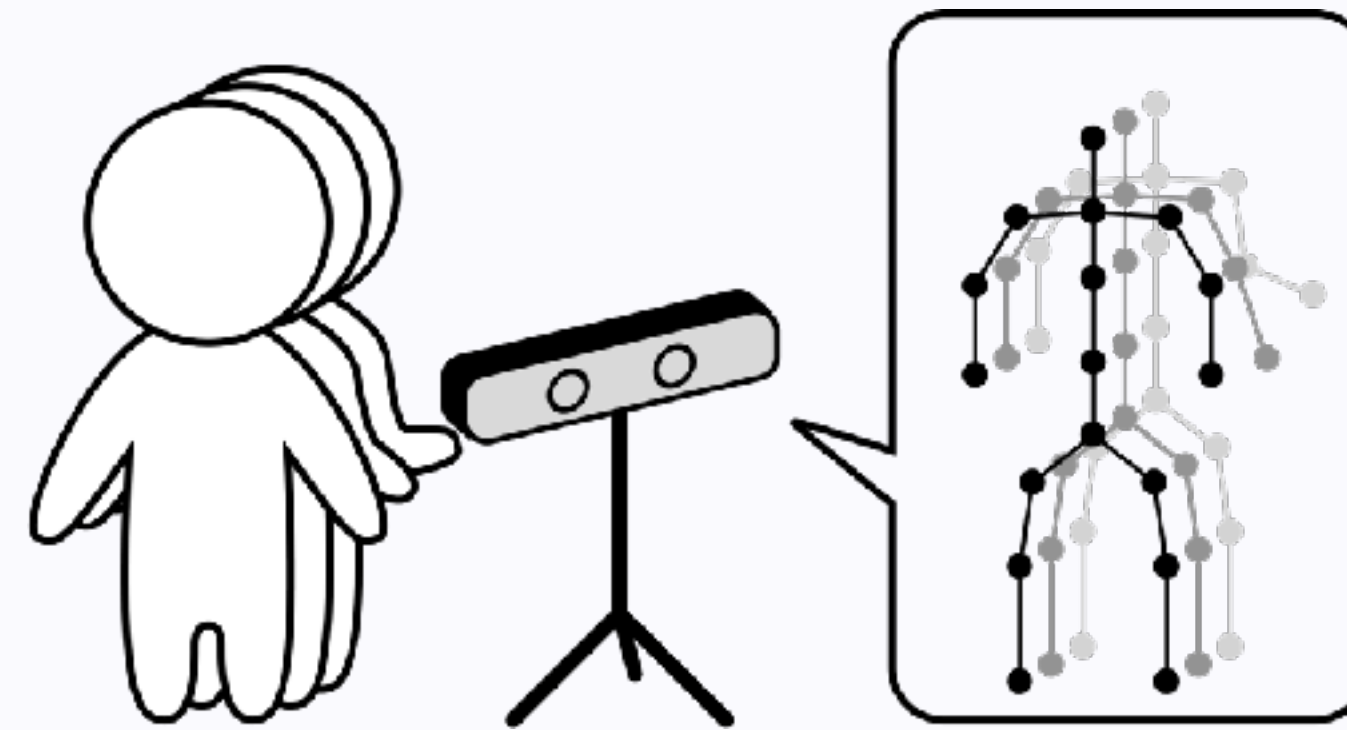


全身の骨格情報を基にした
より複雑な動作の調理技能向上の支援に繋がる

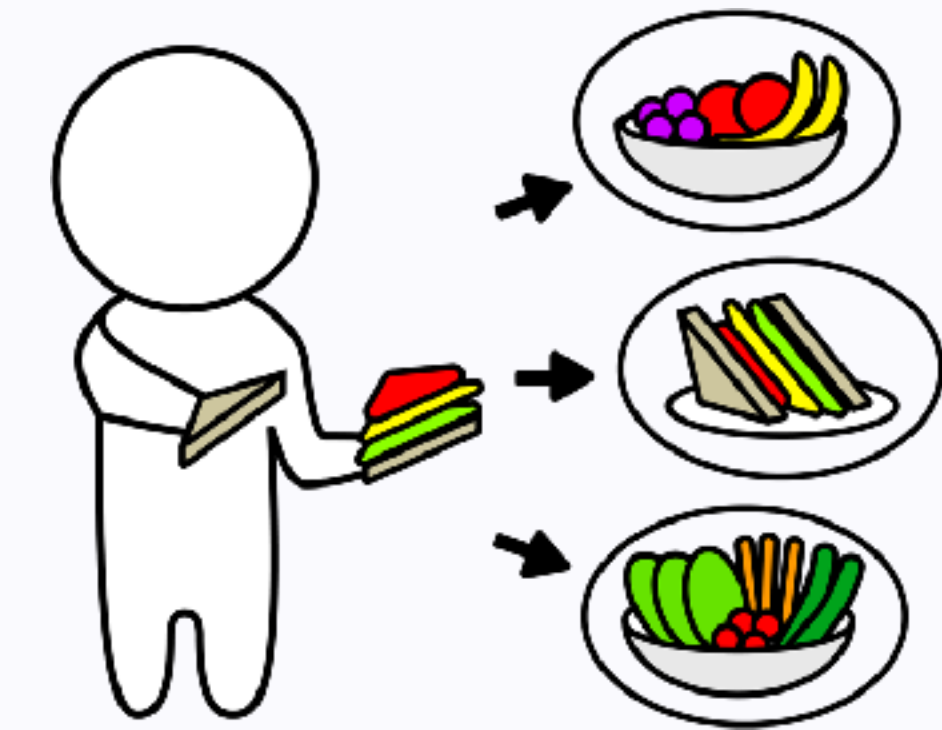
ウェアラブルデバイスで 行動分類する研究



モーションキャプチャから 人の動作を分析する研究



調理行動に関する コンペティション



腕先の動作から行動を認識



モーションキャプチャで
全身の動作から行動を認識

光学式,機械式などから
3次元的な動作を取得



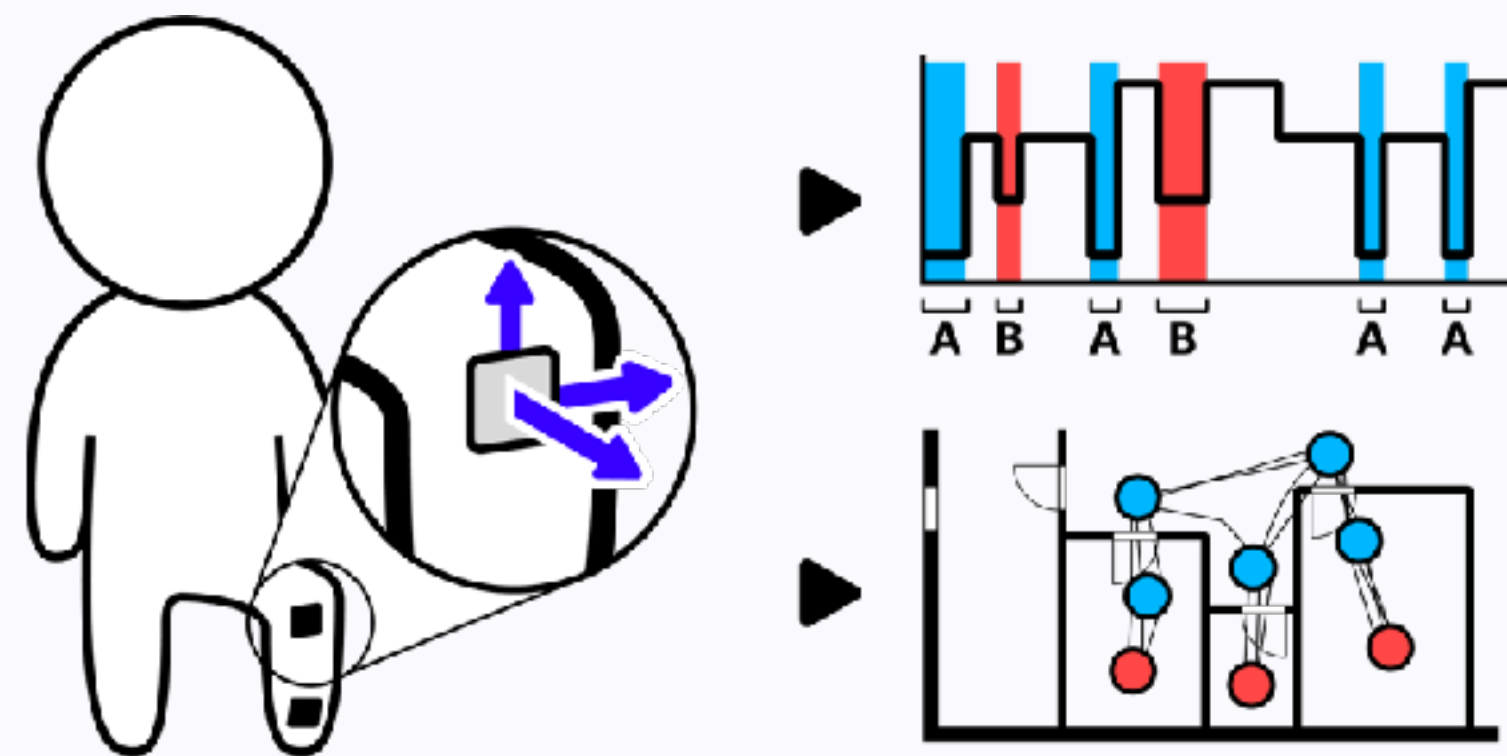
狭いキッチンでも扱い易い
慣性式を用いる

3種類のレシピと
10種類の動作から行動分類



RandomForest が
最も高い精度を示す

行動認識と位置情報を統合した研究

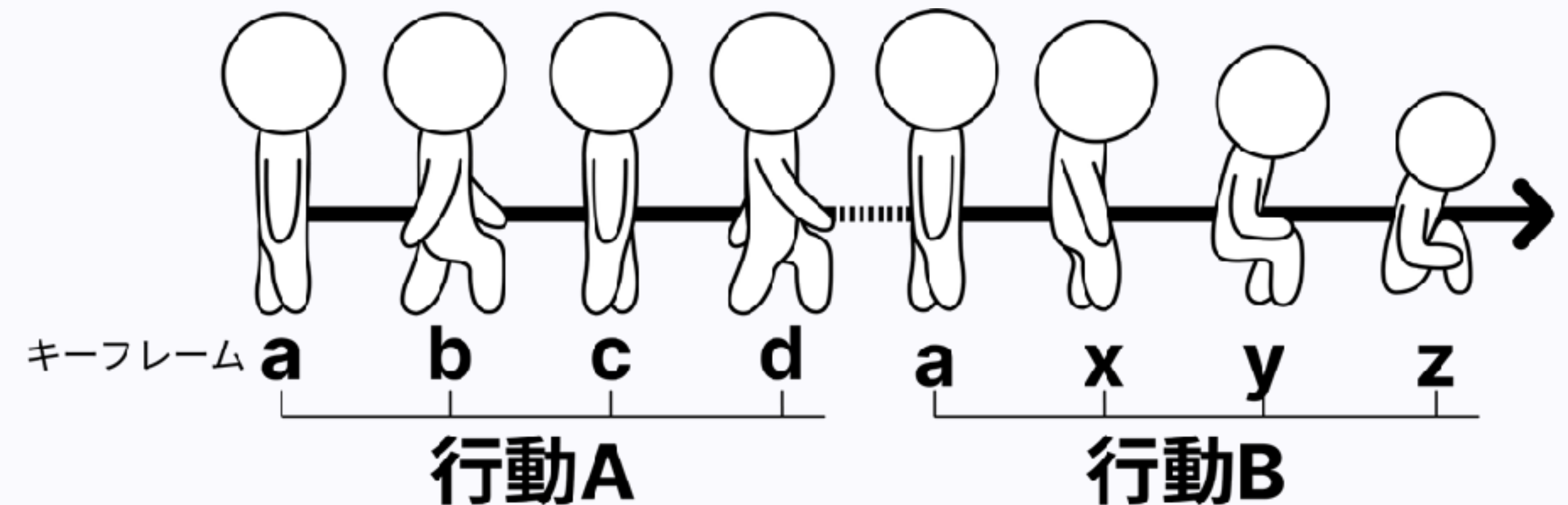


位置推定と行動推定を行い
行動をランダムマークとして位置を補正



行動認識から位置推定結果を補正

テンプレートマッチングによって 行動認識を行う研究



特定姿勢にシンボルを割り当て
編集距離を応用し順序を比較



行動ごとにシンボルを割り当て
編集距離から工程順を比較

最終目標: 調理の技能評価や改善

行動認識と位置推定と手順書を組み合わせた調理行動の分析

- 3次元の骨格情報
- 作業者の位置情報
- 行動の順序, 必要な時間

- 骨格情報 から 時間ごとに行動の確率 が得られる
- 位置情報 から 場所ごとの正しい行動 が絞り込める
- 順序と時間 から 時系列における正しい行動 が絞り込める

→ 相補的に用いた絞り込みで行動認識精度の向上を目指す

目的

モーションキャプチャを用いた調理行動の分類と精度向上

アプローチ

- ① ... 調理行動のセンシング
- ② ... 機械学習による行動分類
- ③ ... 行動単位でのフレームの統合
- ④ ... 位置情報を用いた補正
- ⑤ ... 手順書を用いた補正

作業場所の制約

- 工程ごとに作業場所を指定
- 作業場所と行動は1対多で対応



洗い場

- 洗う
- その他

調理場

- 切る
- 混ぜる etc.

コンロ

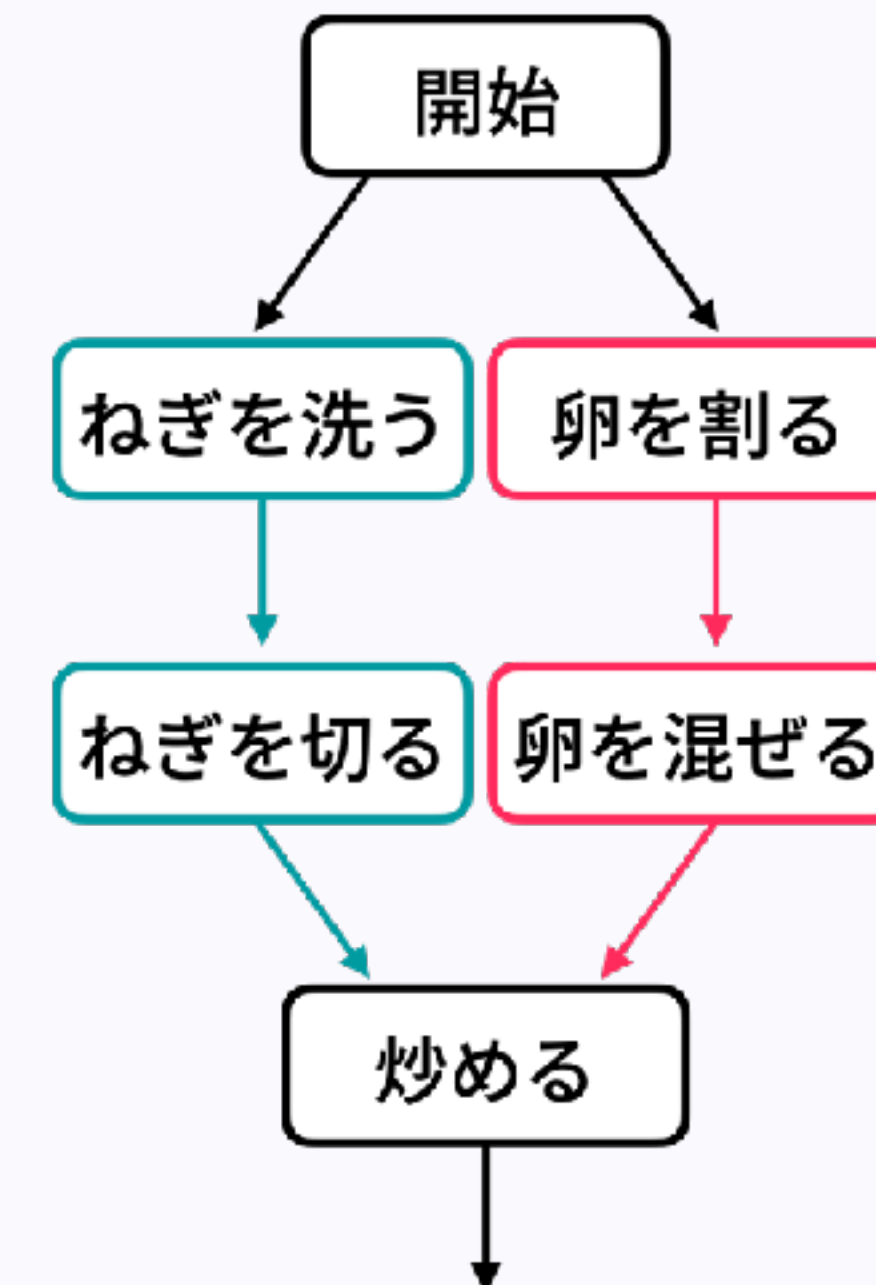
- 炒める
- 味つける etc.

位置情報の制約

- 今回は位置推定を行わず、位置情報を手動で決定

レシピの制約

- レシピを直線的に表現

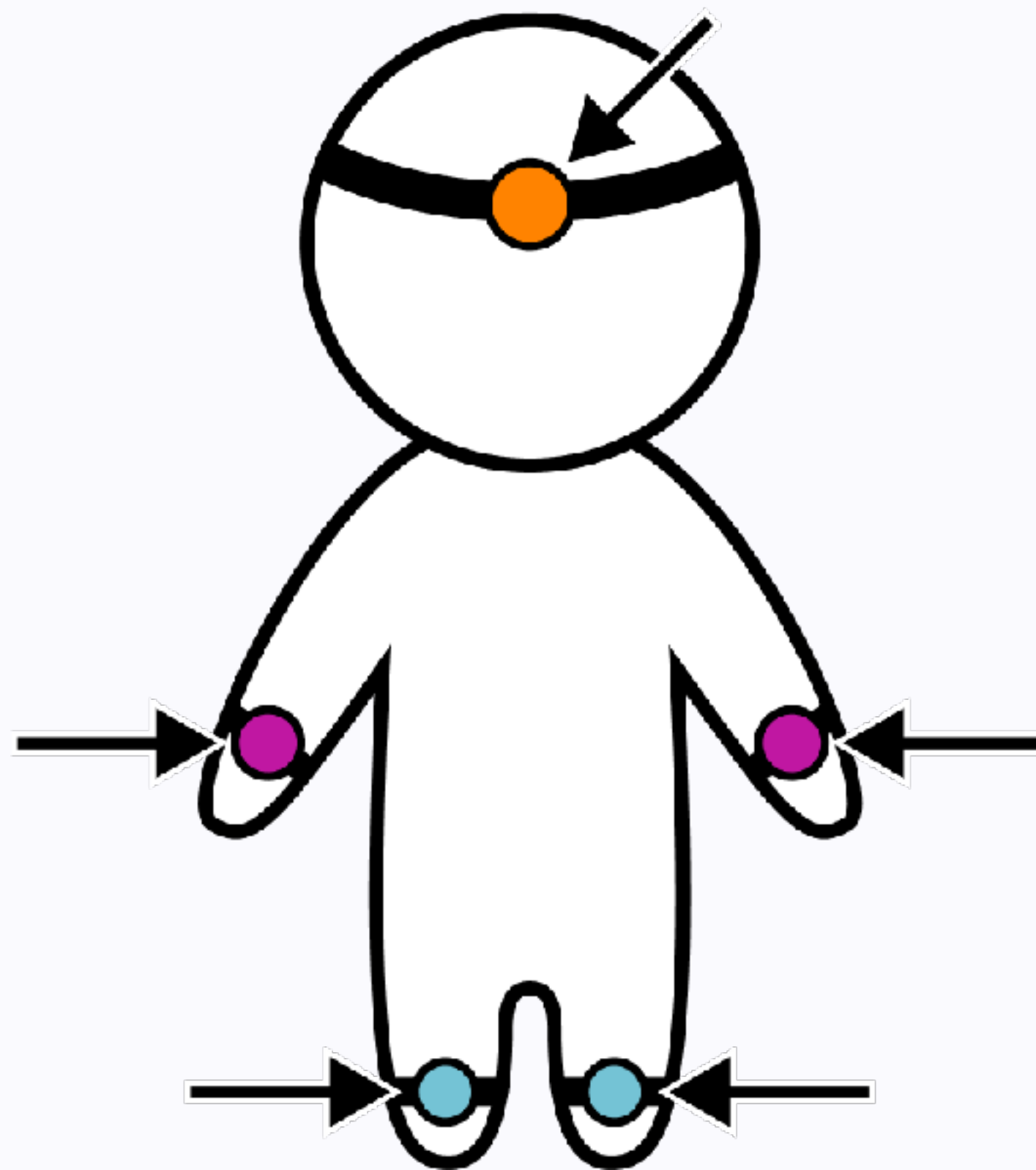


—— モーションキャプチャ ——

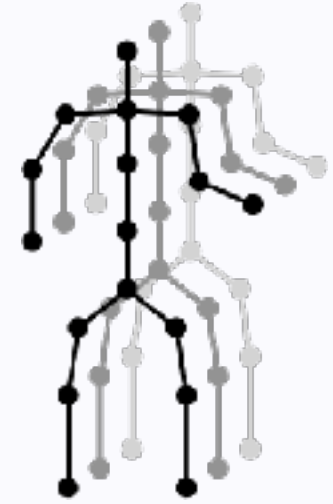
慣性式モーションキャプチャのmocopiを使用
頭・両腕・腰・両足の計6箇所にセンサを装着
骨格情報と相対的な位置情報を取得可能

—— 前提条件 ——

- ・ 工程ごとに作業場所を指定
- ・ 工程順を厳密に指定

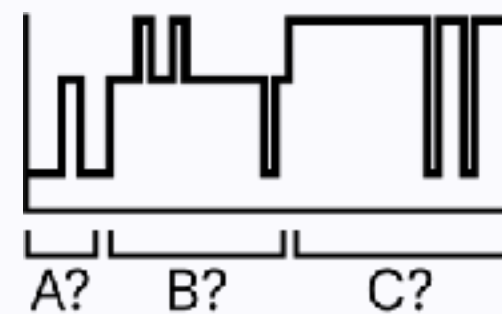


1



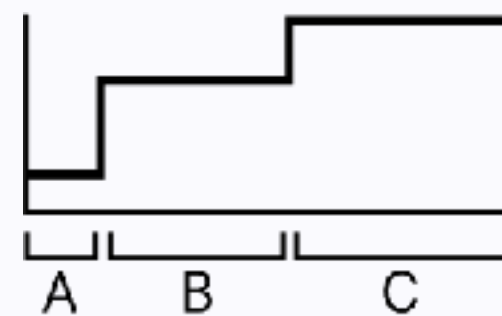
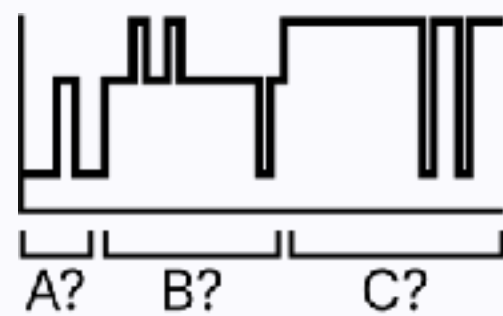
骨格情報から2.0sごとに分割し特徴量を抽出
特徴量には**最大・最小・平均・分散・標準偏差**を使用

2



機械学習を用いて行動分類
Random Forest, XGBoost, LightGBM を比較

3

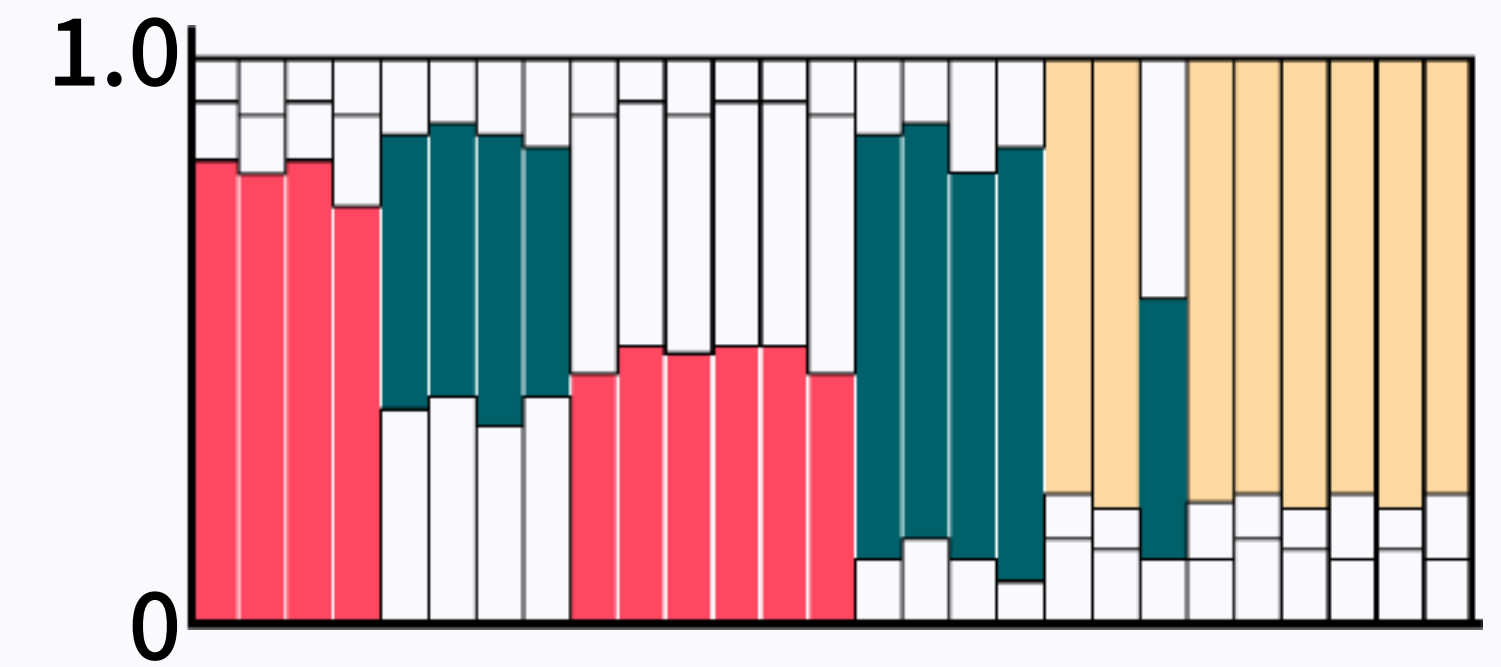


分類結果を6.0sごとに分割し平滑化

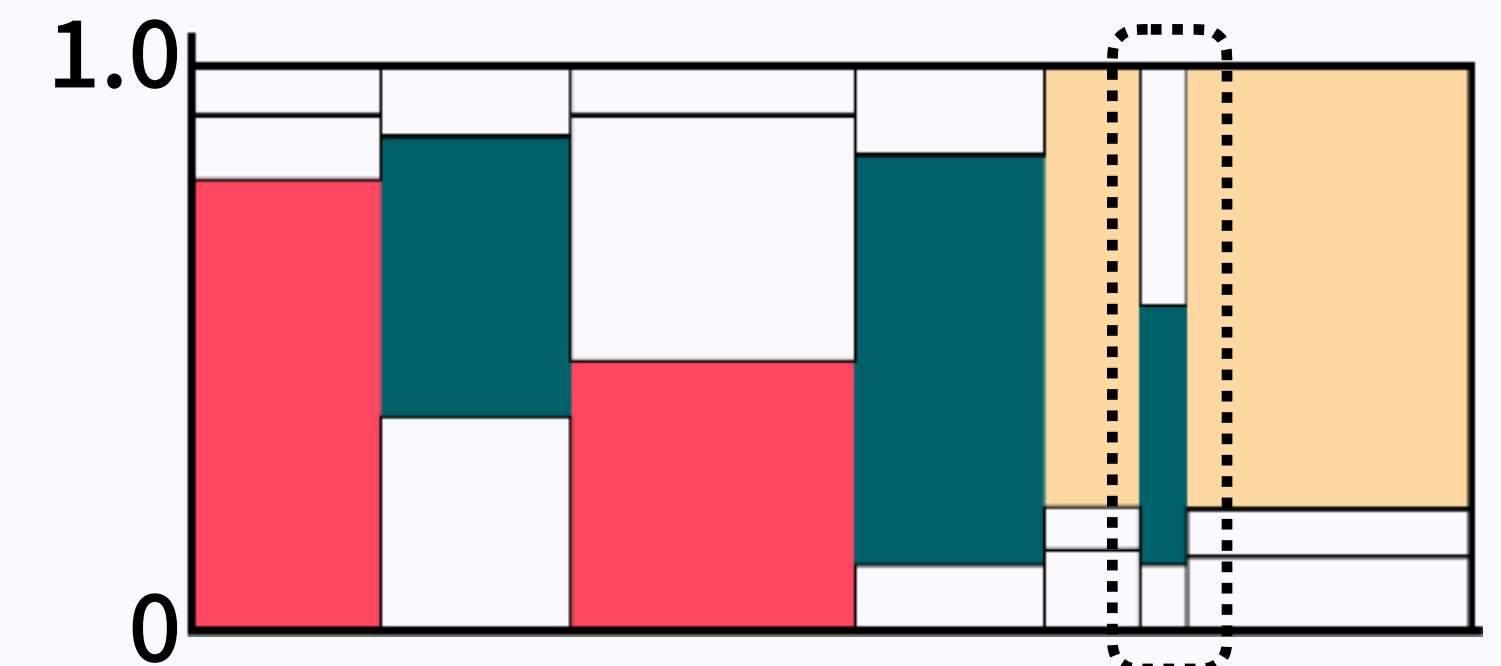
フレームの統合手順

- 行動順序の取得
- 行動ごとの作業時間の取得

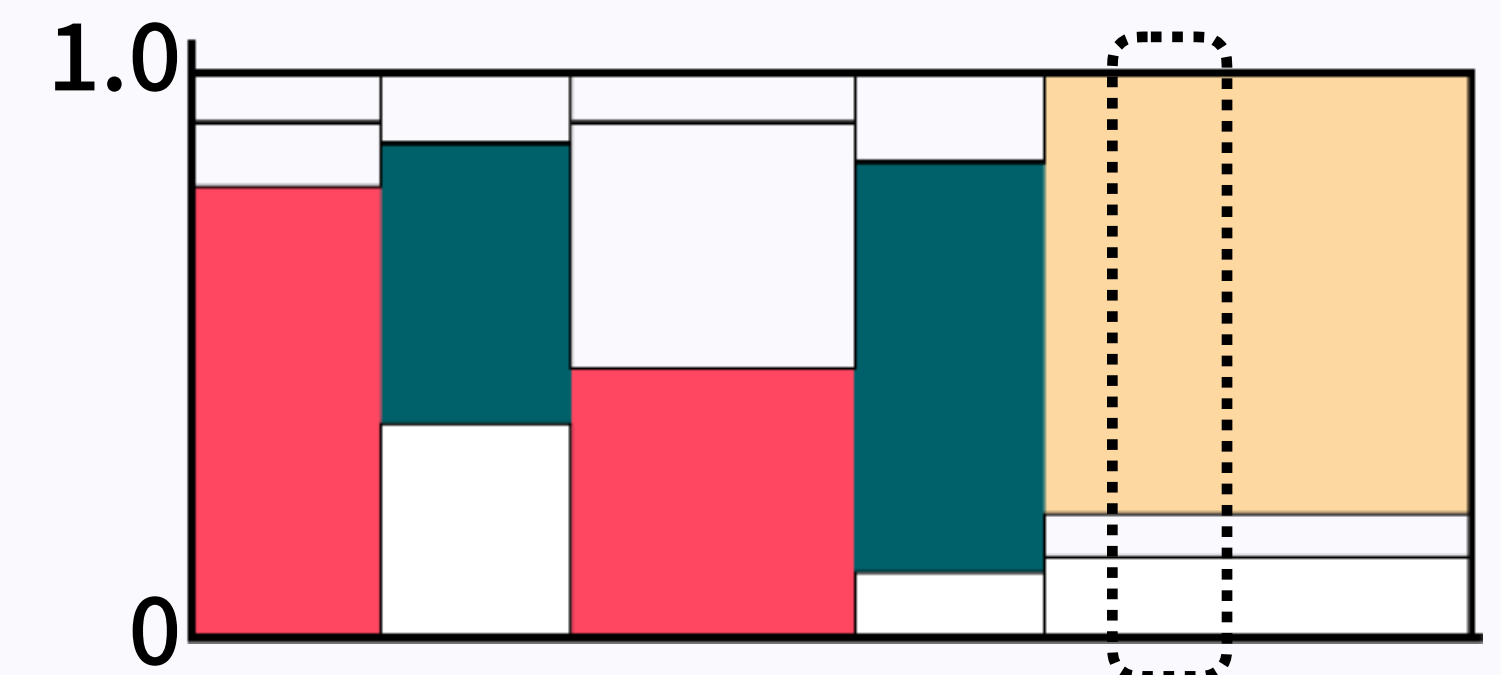
- ① Top-1で同じ行動が連続しているフレームを1つの行動として統合
- ② 作業時間が短い場合は誤りとして前後と統合
 - 前後が同じ行動 : 前後と統合し1つの行動とする
 - 前後が異なる行動 : 確率の高い一方と統合



▼ 1



▼ ②





洗い場

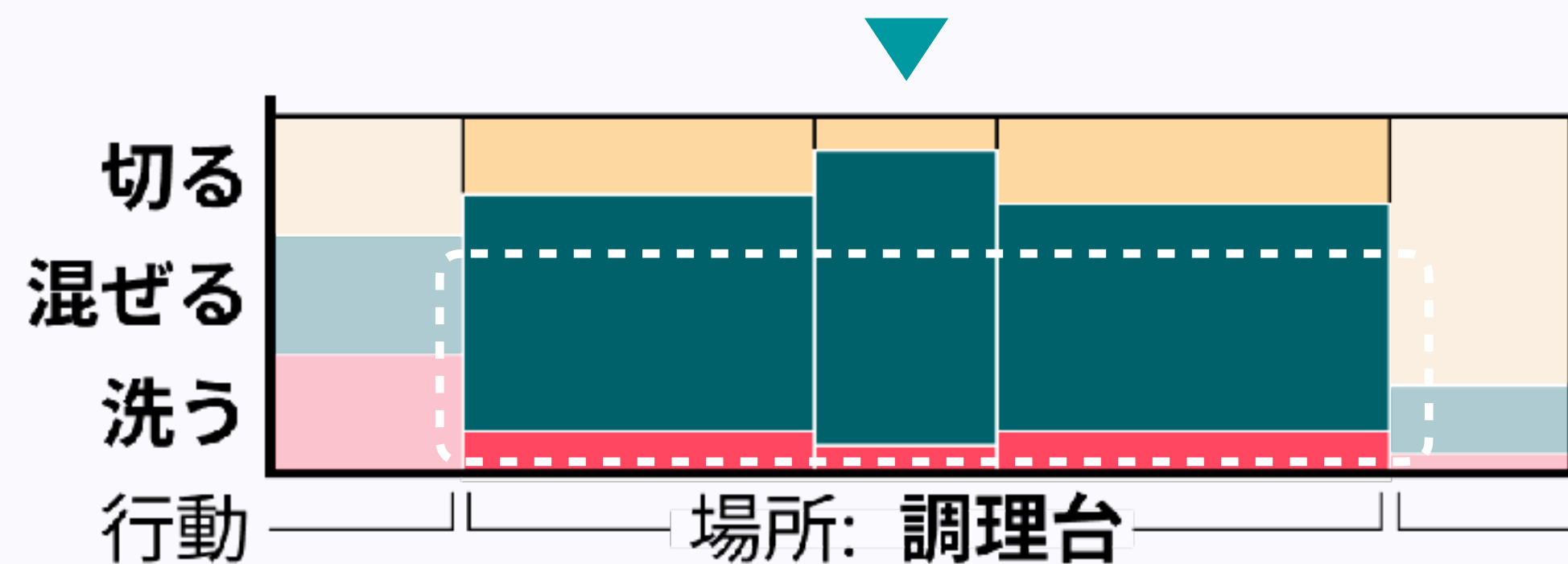
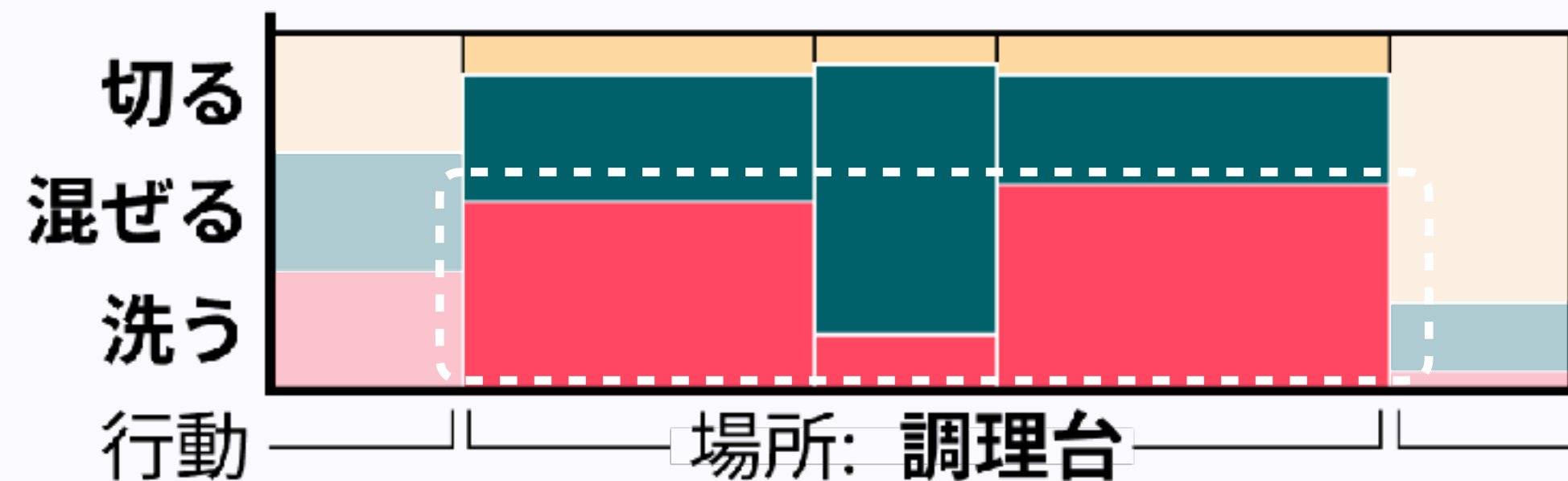
- ・ 洗う
- ・ その他

調理場

- ・ 切る
- ・ 混ぜる etc.

コンロ

- ・ 炒める
- ・ 味つける etc.



前提条件

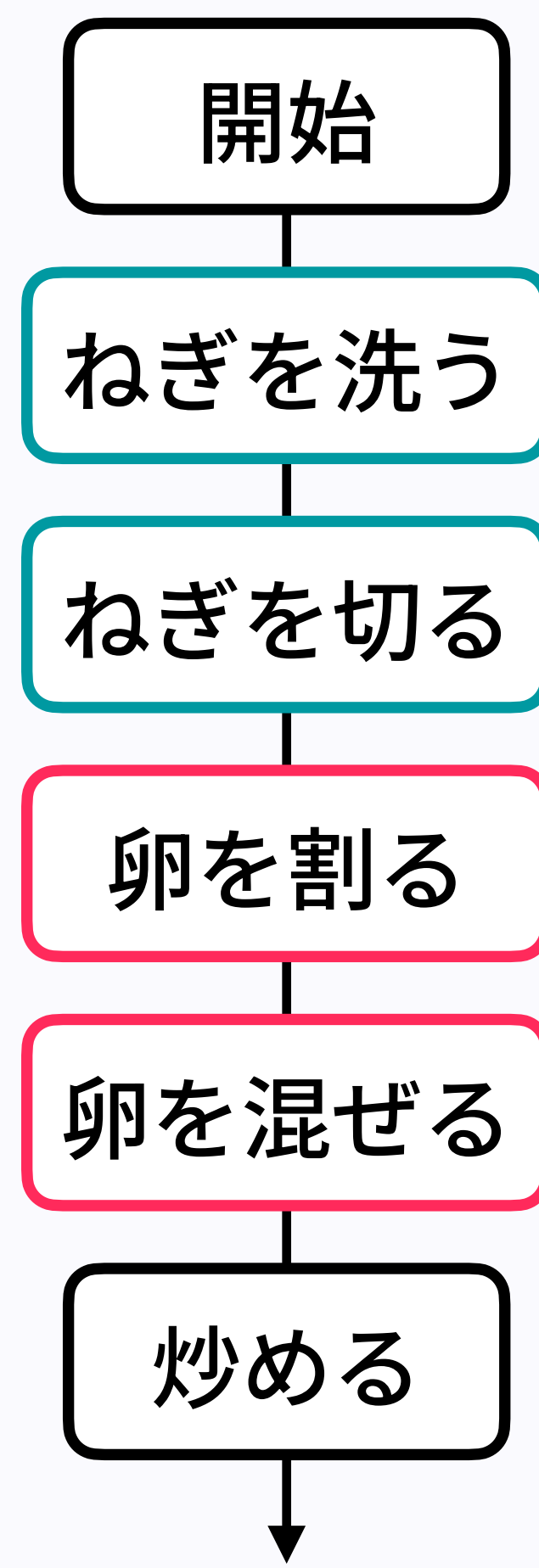
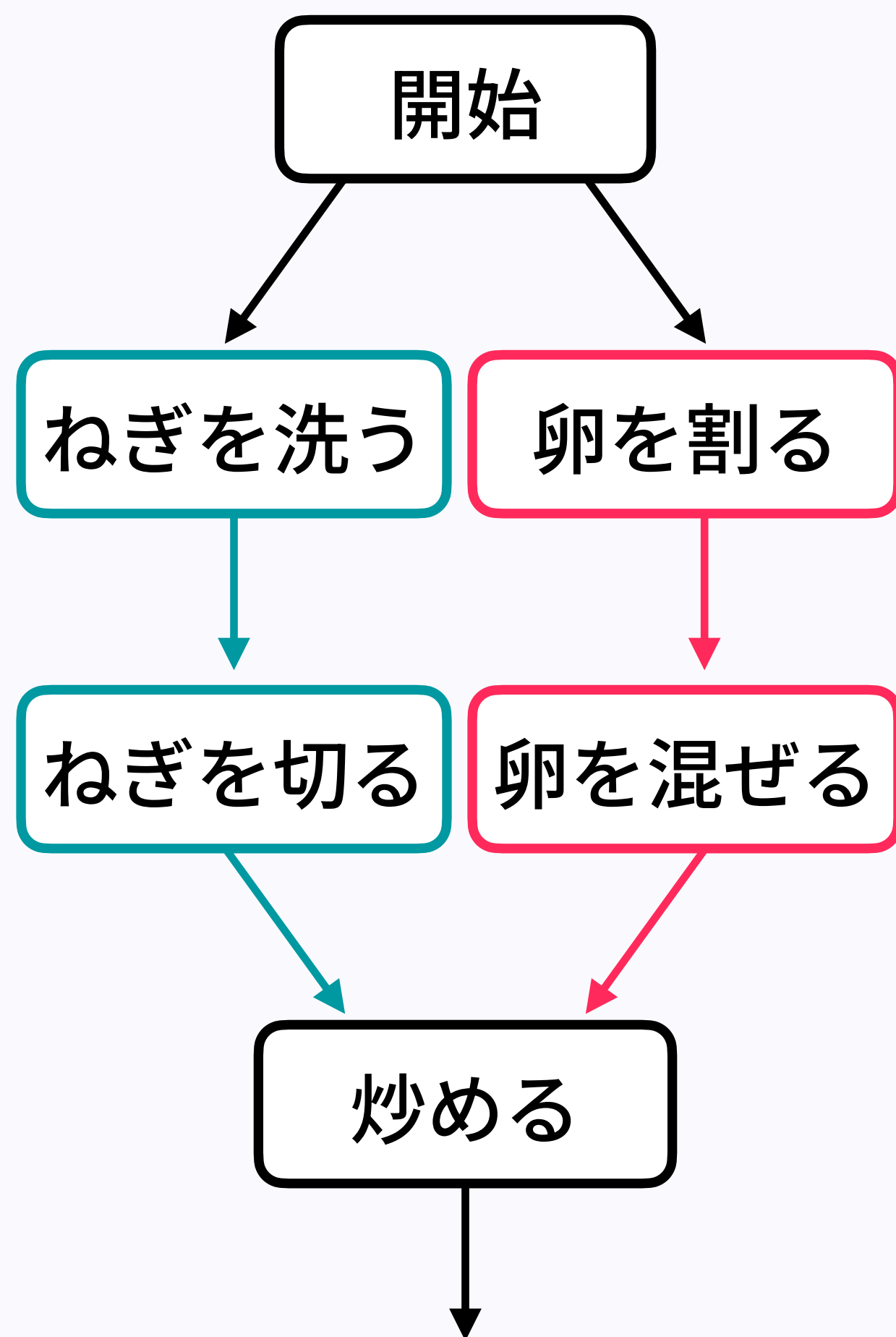
- ・ 作業場所と行動は1対多で対応
- ・ 「その他」の行動は全ての作業場所と対応
- ・ 「その他」の作業場所は全ての行動と対応
- ・ 今回は事前に位置情報を手動で決定

補正手順

1. 作業場所をもとに不適切な行動を割り出す
2. 不適切な行動の確率を操作

本来のレシピ構造

今回のレシピ構造



本来は有向非巡回グラフで構造化
順序の入れ替わりを表現可能

今回は単純化のために
直線的な手順としてレシピを表現

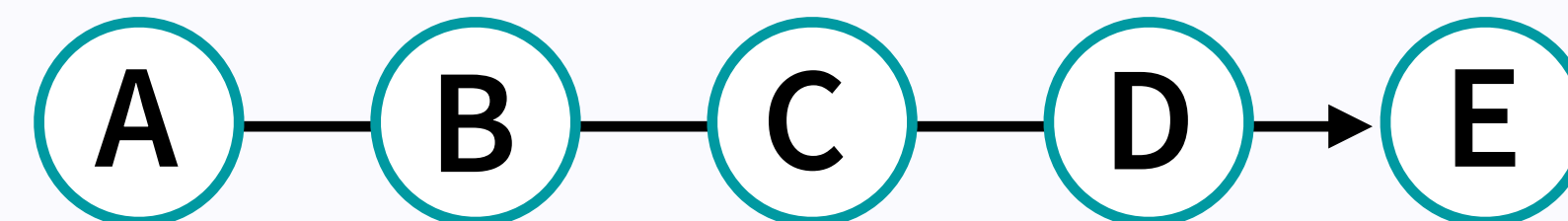
編集距離

2つの文字列を一致させるために必要な挿入，削除，置換の操作の最小回数

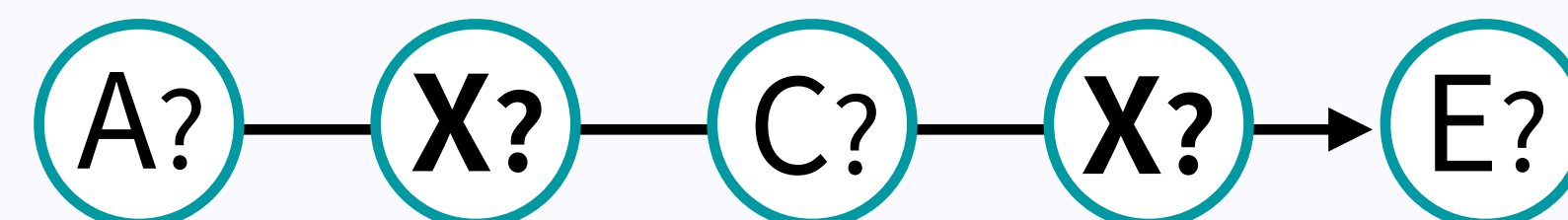
補正手順

1. 各行動を順に走査
2. 対象の行動を置換し編集距離を計算
3. 最も編集距離が小さい行動を選択

手順書による正解

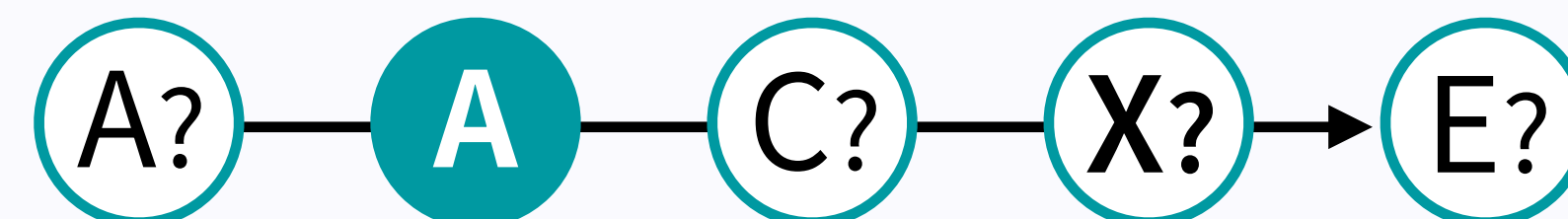


推定結果の例

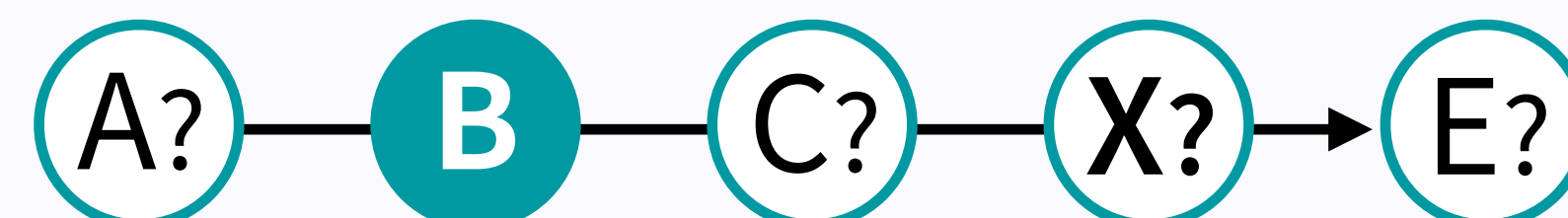


編集距離による比較

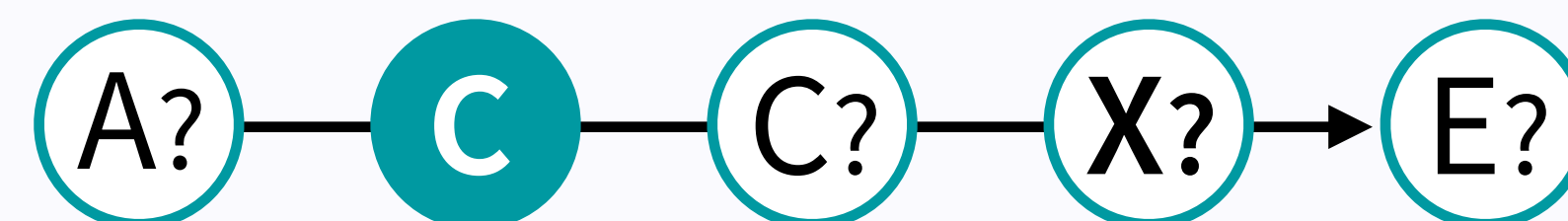
編集距離



▶ 2



▶ 1



▶ 2

⋮

⋮

チャーハンの調理行動を対象とする

調理に含まれる行動と場所

洗い場	ネギを洗う, その他
調理台	ネギを切る, 卵を割る, 卵をまぜる, 盛り付ける, その他
コンロ前	油を引く, 卵流し込む, 炒める, 具材を入れる, 味付ける, その他

選定理由

行動分類の有効性を包括的に検証が可能

調理台, コンロ前, 流し台 など移動を伴う
洗う, 混ぜる, 炒める など多様な動作を含む

骨格情報からは時間ごとの行動を絞り込むことを目的とするため
Top-1に加えてTop-3でも比較

手法	Top-1 の精度	Top-3 の精度
Random Forest	0.545	0.900
XGBoost	0.708	0.930
Light GBM	0.666	0.905

XGBoost が最も精度が高く、LightGBM も比較的精度が高い
今回は XGBoost が適していると考えられる

XGBoost と Light GBM で
高い精度を得られた



骨格情報は特徴相互作用が複雑
XGBoost と LightGBMが
特徴相互作用を捉えるのに適切

XGBoost は LightGBM より
精度が高い



今回のデータ量では足りなく
過学習が起こったのでは

行動ごとの最小の作業時間

行動	作業時間 (秒)
ネギを洗う	7.7
ネギを切る	14.0
卵を割る	8.1
卵を混ぜる	7.8
油を引く	5.6
卵を流し込む	7.5
炒める	3.7
具材を入れる	7.8
味つける	12.0
盛り付ける	12.4
その他	0.1

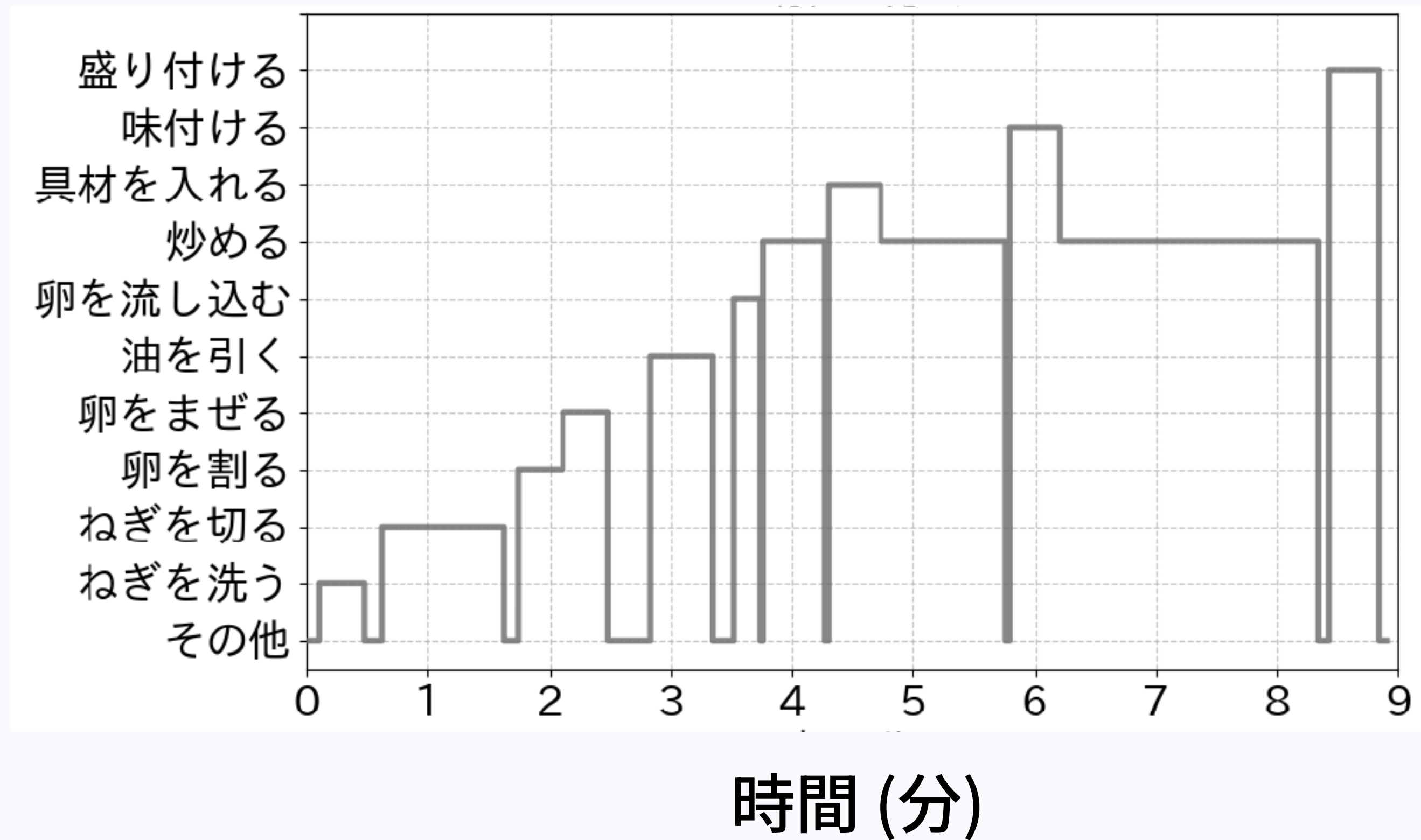
「その他」を除く

工程順には影響せず，推定結果に対する影響は小さい

「炒める」が 3.7秒 と最も短い

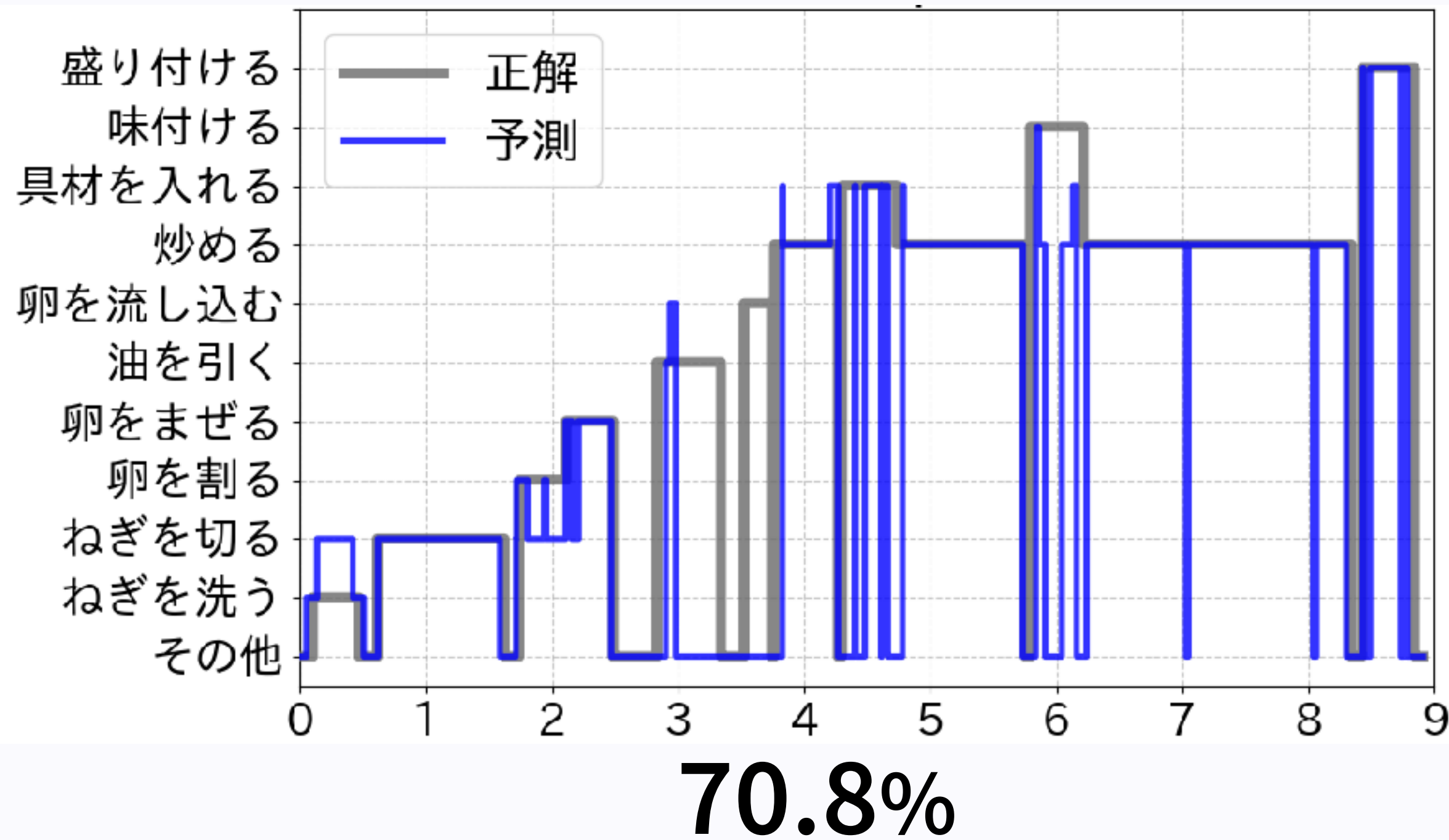
3.0秒以下の行動は誤りとして扱い，前後と統合

行動の種類

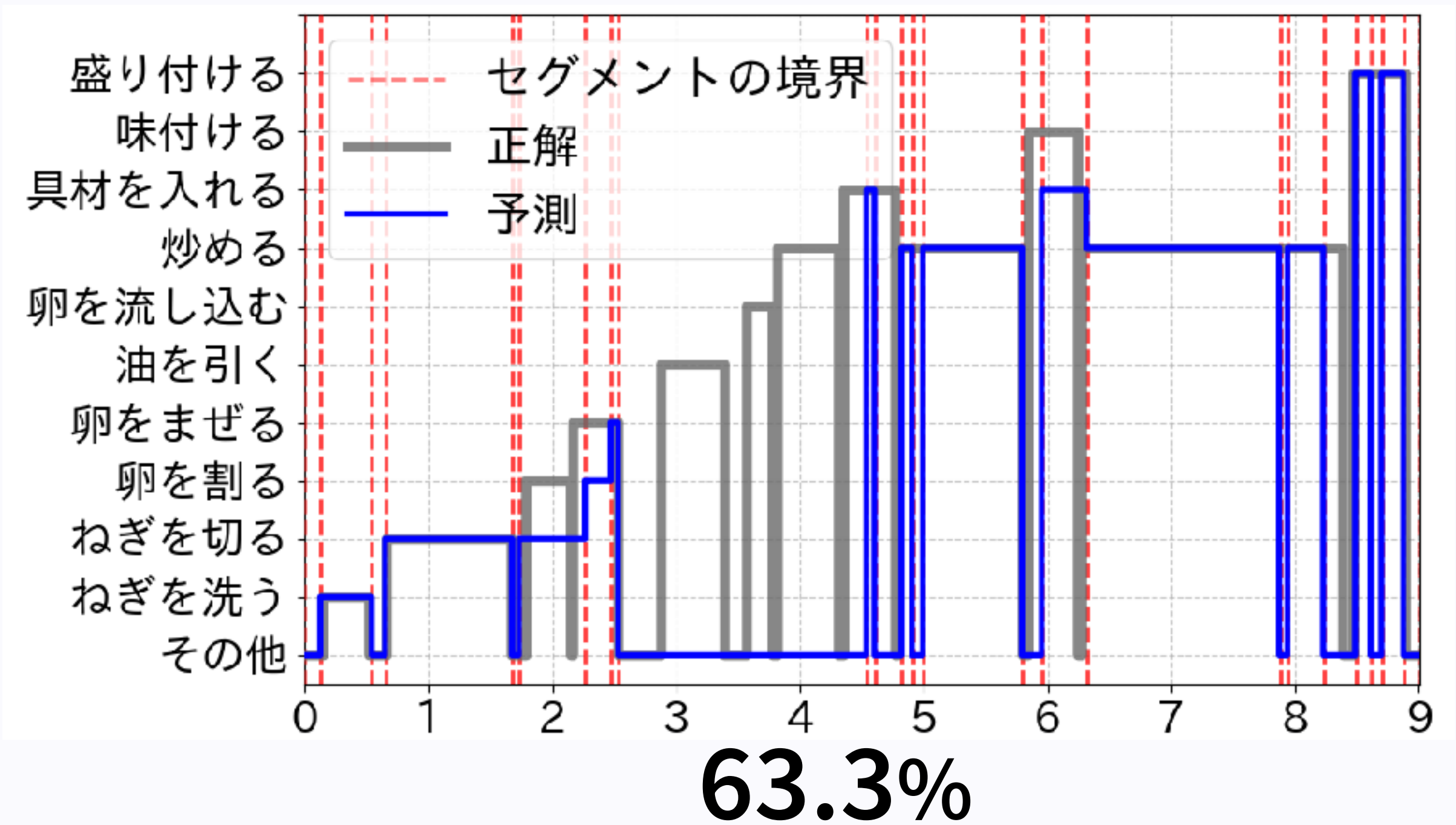


- 工程順に下から行動が並ぶ
- 工程間に「その他」を含む場合がある
- 「炒める」は複数回行われる

統合前のTop-1



統合後のTop-1

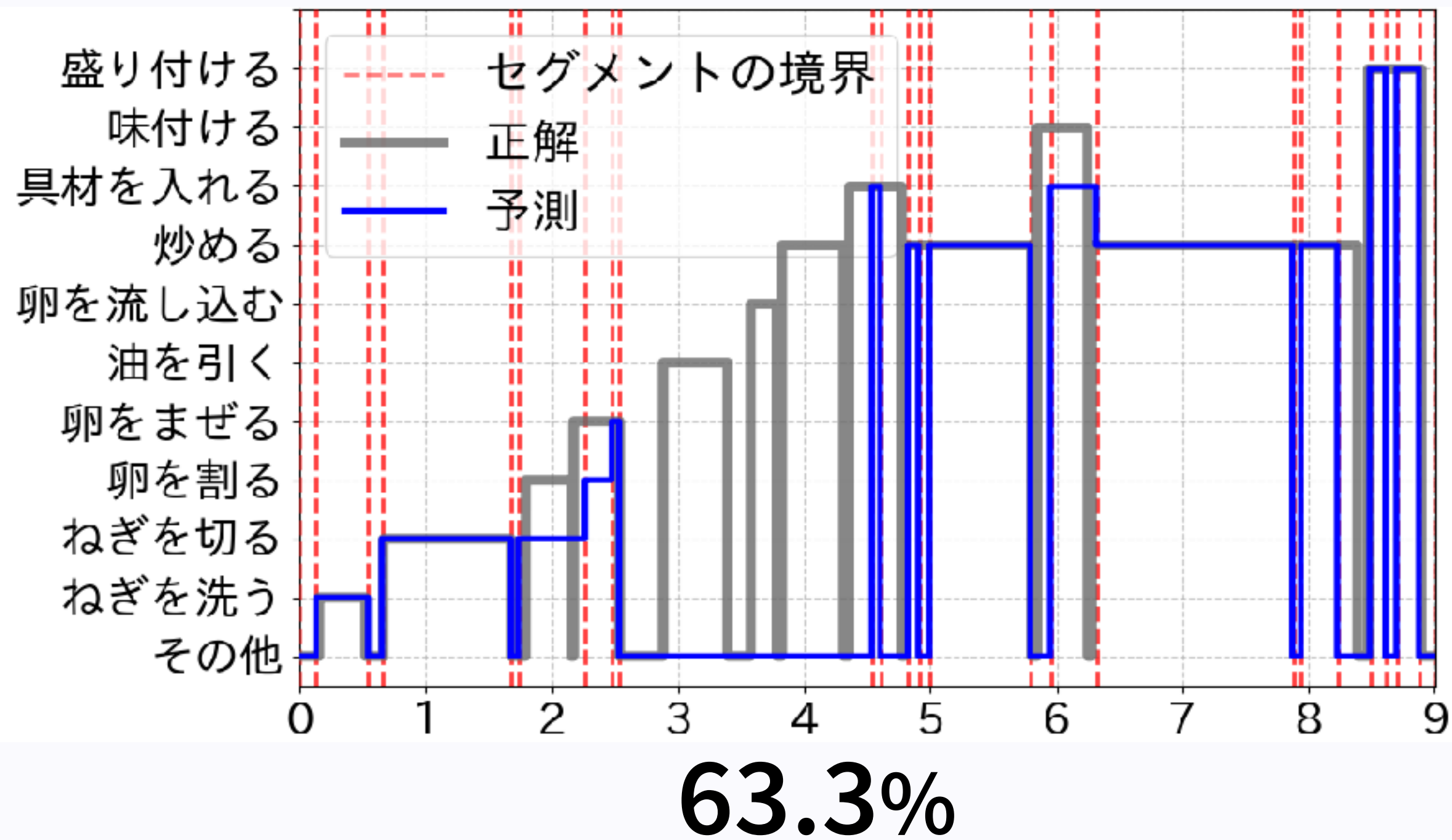


1つの行動が細かく分割されている
複数の行動が1つの行動として統合されている

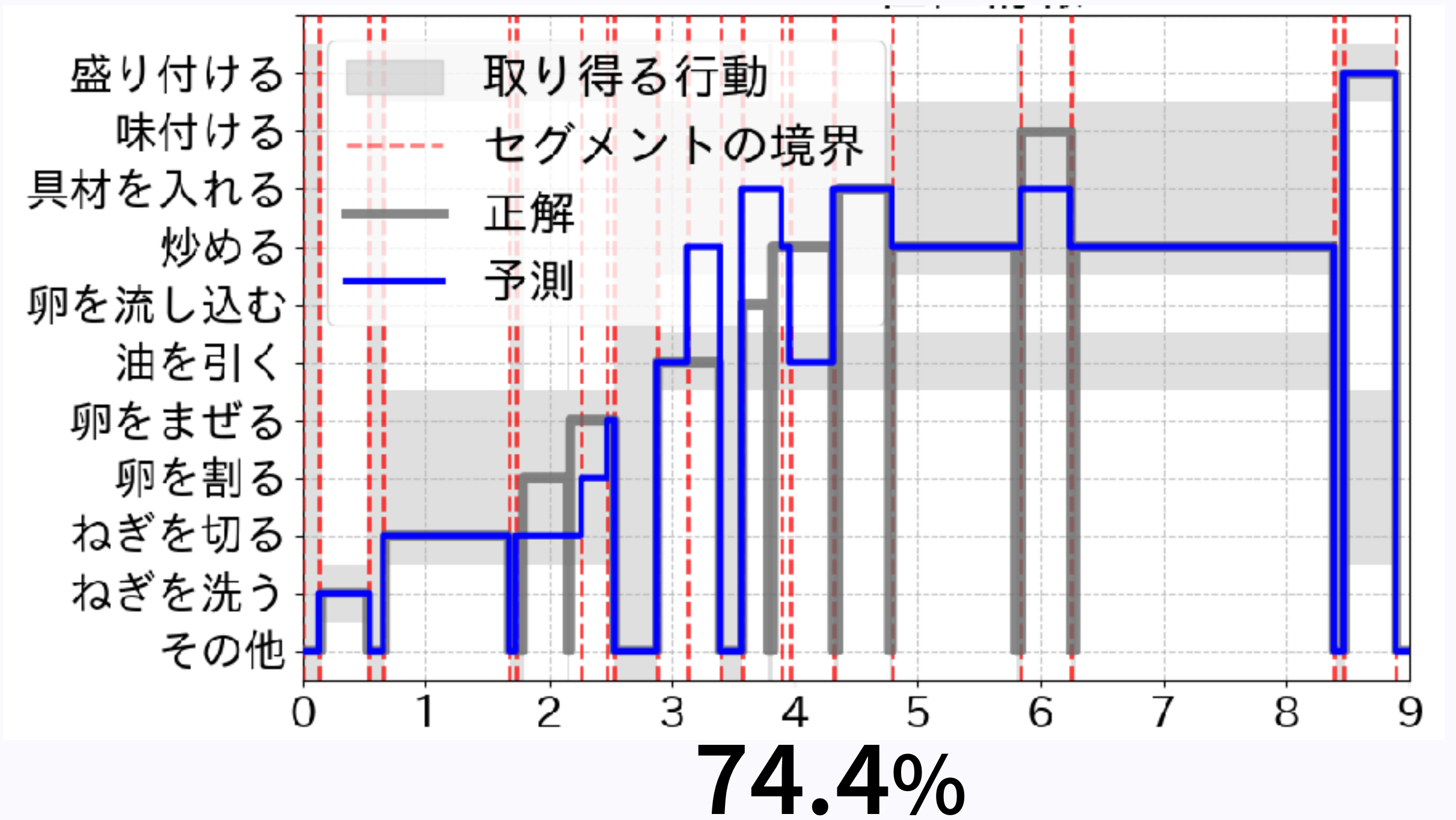


位置情報によって行動を絞り込むと、
安定して作業時間の範囲を認識できるのでは

統合後のTop-1



位置情報による補正後のTop-1

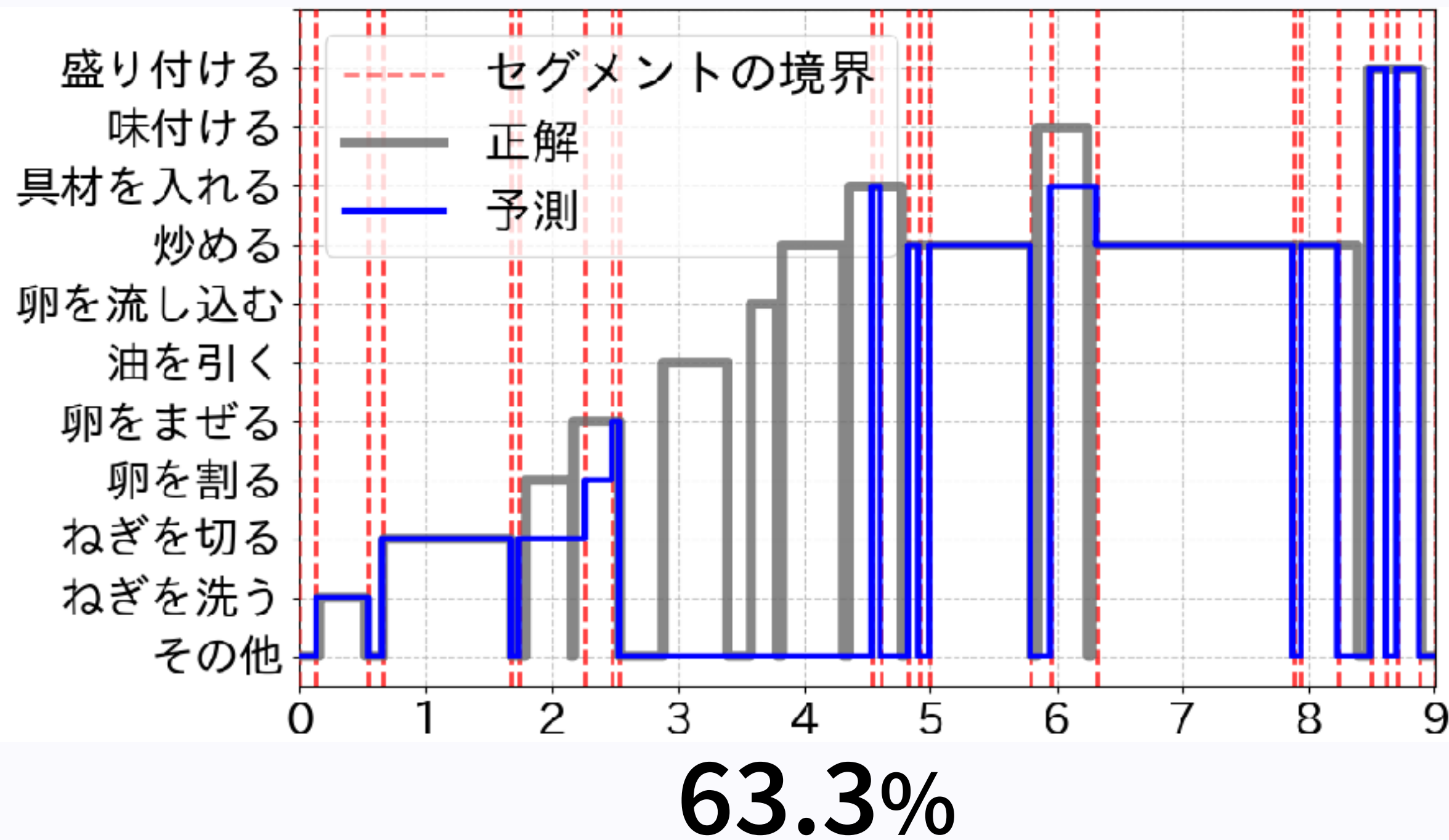


作業時間の範囲をより正しく認識
「具材を入れる」の後に「油を引く」のように工程が逆転

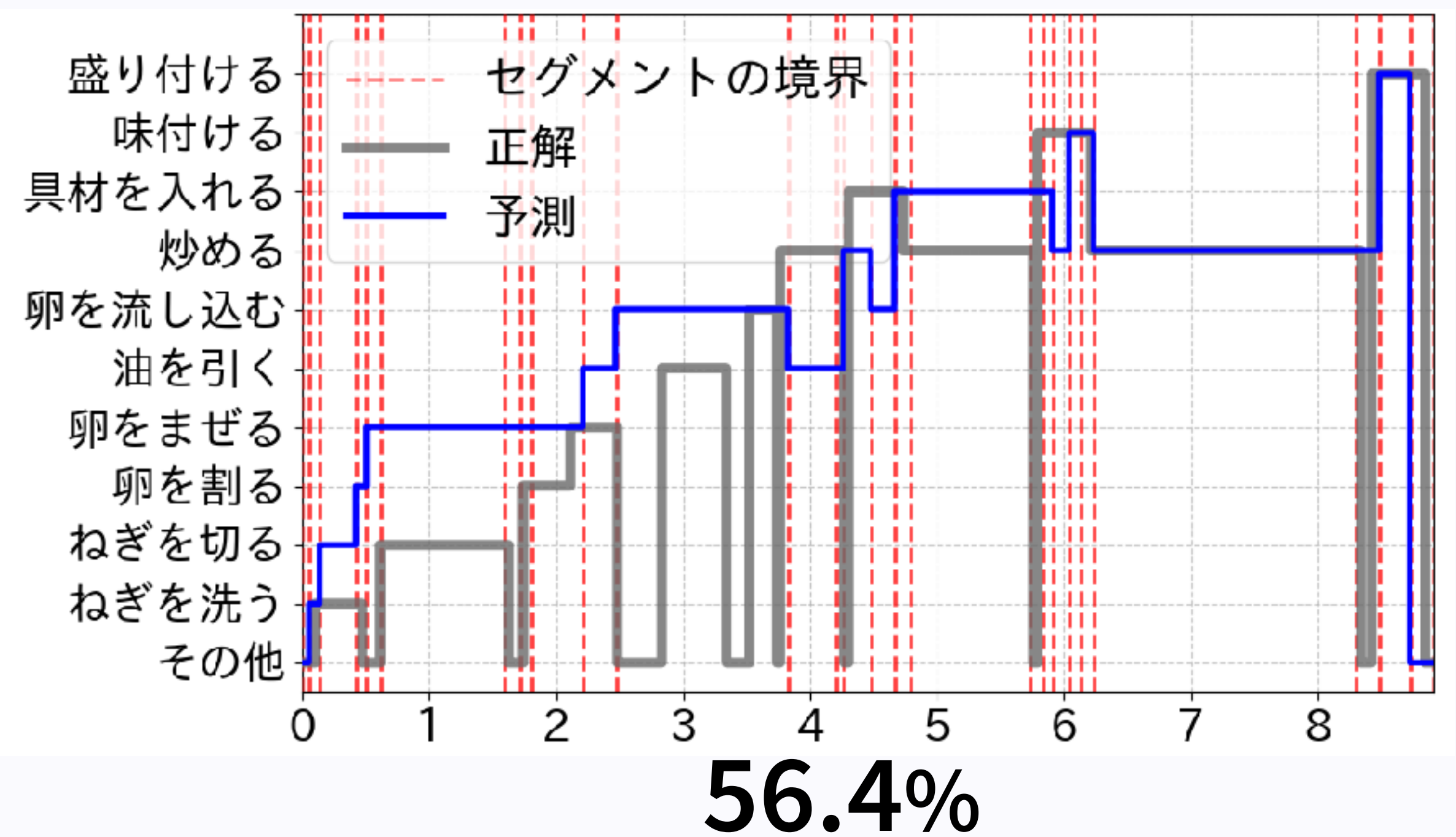


工程順を考慮すると精度向上できるのでは

統合後のTop-1



手順書による補正後のTop-1



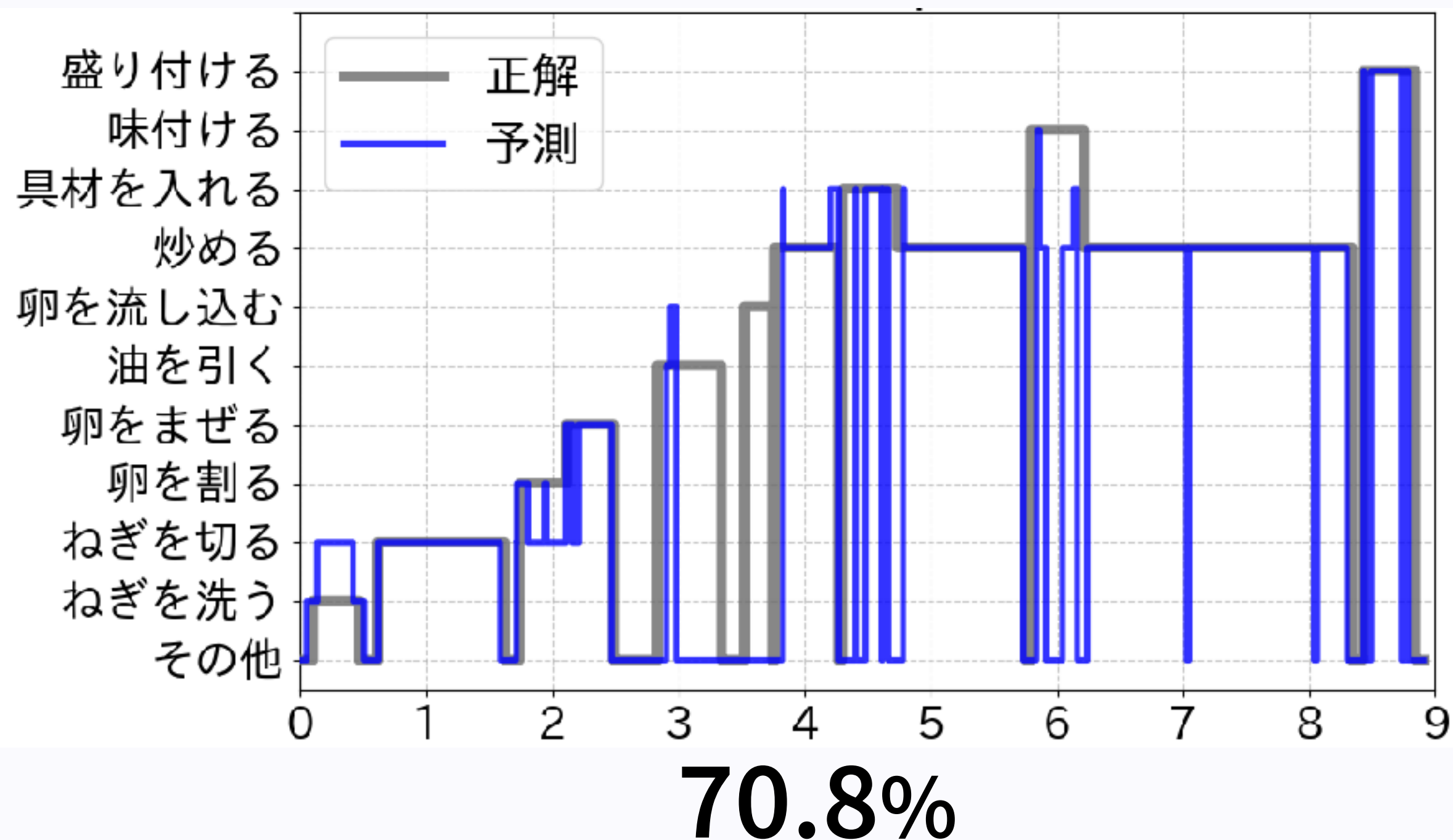
工程順の逆転が減少

作業時間の範囲の誤認識により本来とは異なる作業時間で補正される

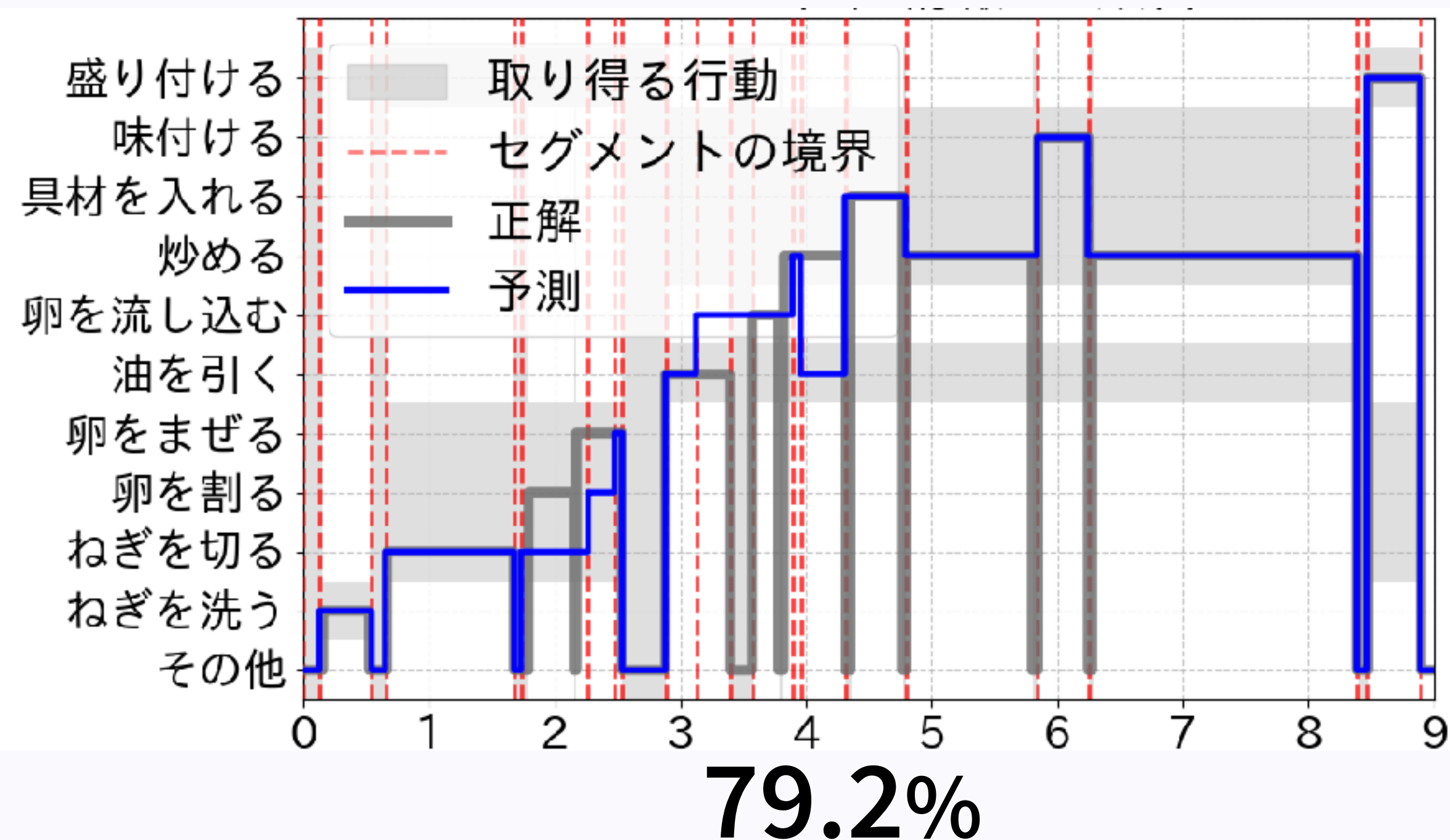


位置情報によって作業時間の範囲を補正すると精度向上できるのでは

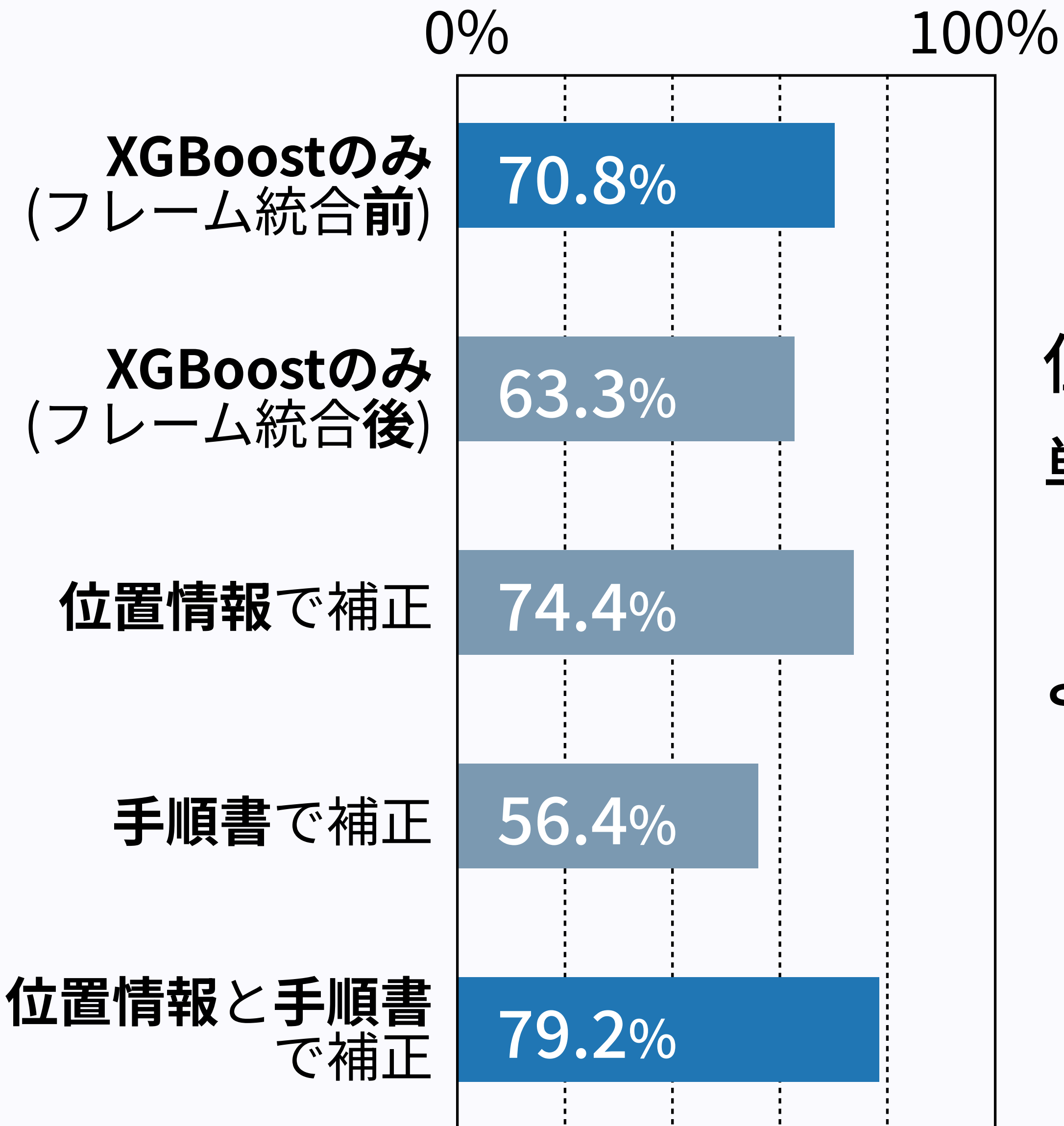
XGBoostのみのTop-1



位置情報と手順書による補正後のTop-1



位置情報と手順書を組み合わせると誤認識が減少
フレーム統合時の作業範囲の誤認識によって工程順の誤認識も増える



位置情報と手順書の組み合わせで精度が向上
単一の手法では精度が低下



より現実に即したアルゴリズムにする必要がある

- ・ Top-1以外も考慮したフレーム統合
- ・ 柔軟な工程順による補正 など

まとめ

モーションキャプチャで得られる3次元の骨格情報を基に調理行動を分類

今回の実験においては XGBoost が適していた

位置情報や手順書との組み合わせによる精度の向上を確認

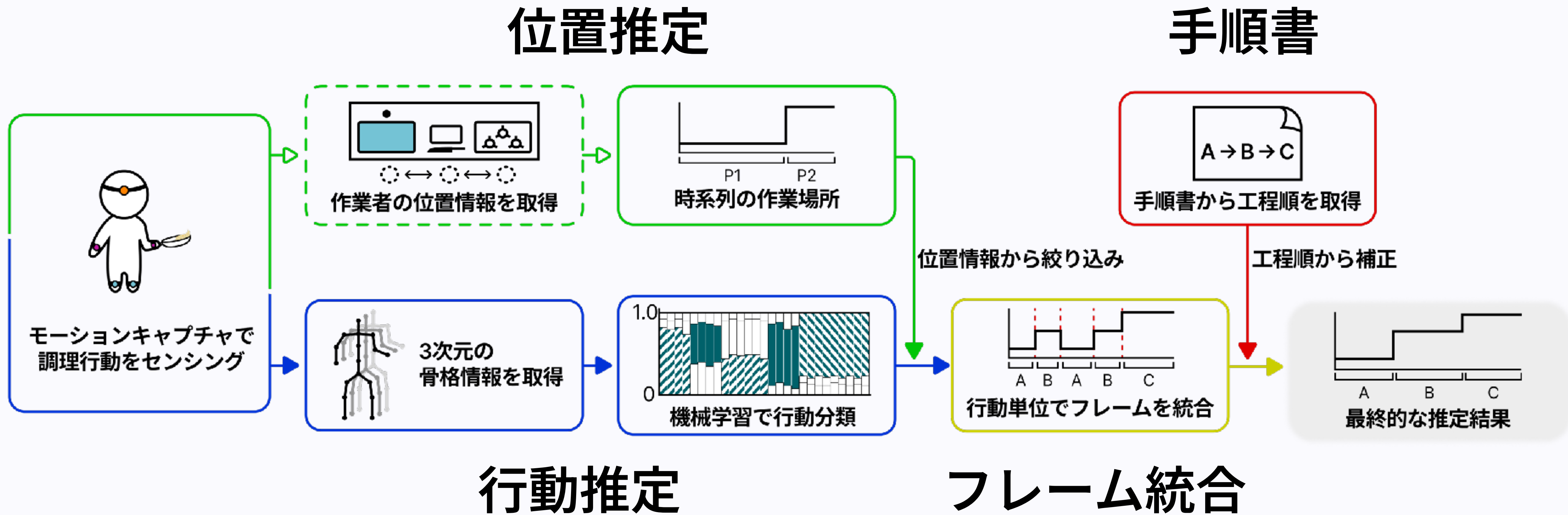
今後の課題

実際に位置推定を行い行動推定と相補的に補正

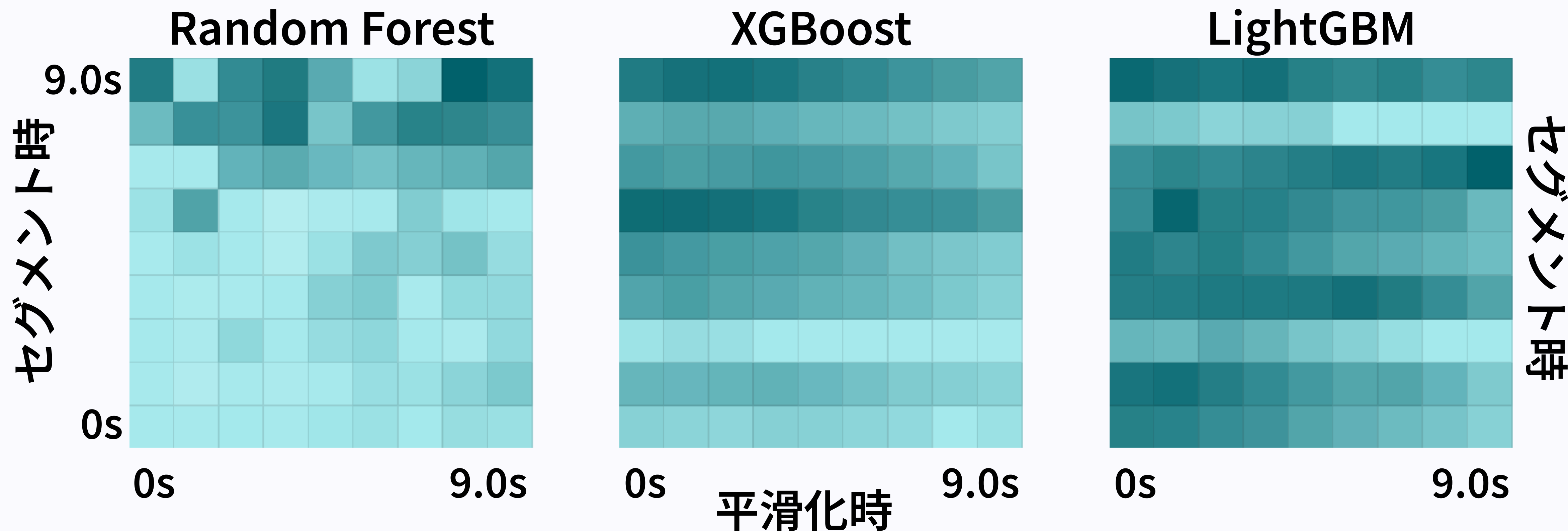
手順書による作業時間を考慮した補正

より現実に即した工程の入れ替わりを考慮

質疑応答用



精度が高いほど濃い色で示す



セグメント時のウィンドウ幅が 2.0s
平滑化時のウィンドウ幅が 6.0s
のとき各手法において比較的高い精度が得られた

steps:

- PROCESS[ねぎを洗う]
- PROCESS[ねぎを切る]
- PROCESS[卵を割る]

省略

processes:

- id: PROCESS[ねぎを洗う]
title: ねぎを洗う
- id: PROCESS[ねぎを切る]
title: ねぎを切る
- id: PROCESS[卵を割る]
title: 卵を割る

省略

	A	B	C	D	E	F	G	H	I	J	K
A.その他	.999	.000	.000	.000	.000	.000	.000	.001	.000	.000	.000
B.ねぎを洗う	.010	.990	.000	.000	.000	.000	.000	.000	.000	.000	.000
C.ねぎを切る	.005	.000	.995	.000	.000	.000	.000	.000	.000	.000	.000
D.卵を割る	.388	.000	.005	.607	.000	.000	.000	.000	.000	.000	.000
E.卵をまぜる	.003	.000	.000	.064	.934	.000	.000	.000	.000	.000	.000
F.油を引く	.000	.000	.000	.000	.000	1.000	.000	.000	.000	.000	.000
G.卵流し込む	.226	.000	.000	.000	.000	.000	.774	.000	.000	.000	.000
H.炒める	.010	.000	.000	.000	.000	.000	.010	.979	.000	.000	.000
I.米とねぎ入れる	.000	.000	.000	.000	.000	.000	.000	.151	.849	.000	.000
J.味付ける	.004	.000	.000	.000	.000	.000	.000	.060	.013	.923	.000
K.盛り付ける	.080	.000	.000	.000	.000	.000	.000	.147	.000	.000	.773

調理行動の混同行列

	...	H	I	J	K
...		...	...	...	...
H.炒める	...	.979	.000	.000	.000
I.米とねぎ入れる	...	.151	.849	.000	.000
J.味付ける	...	.060	.013	.923	.000
K.盛り付ける	...	.147	.000	.000	.773

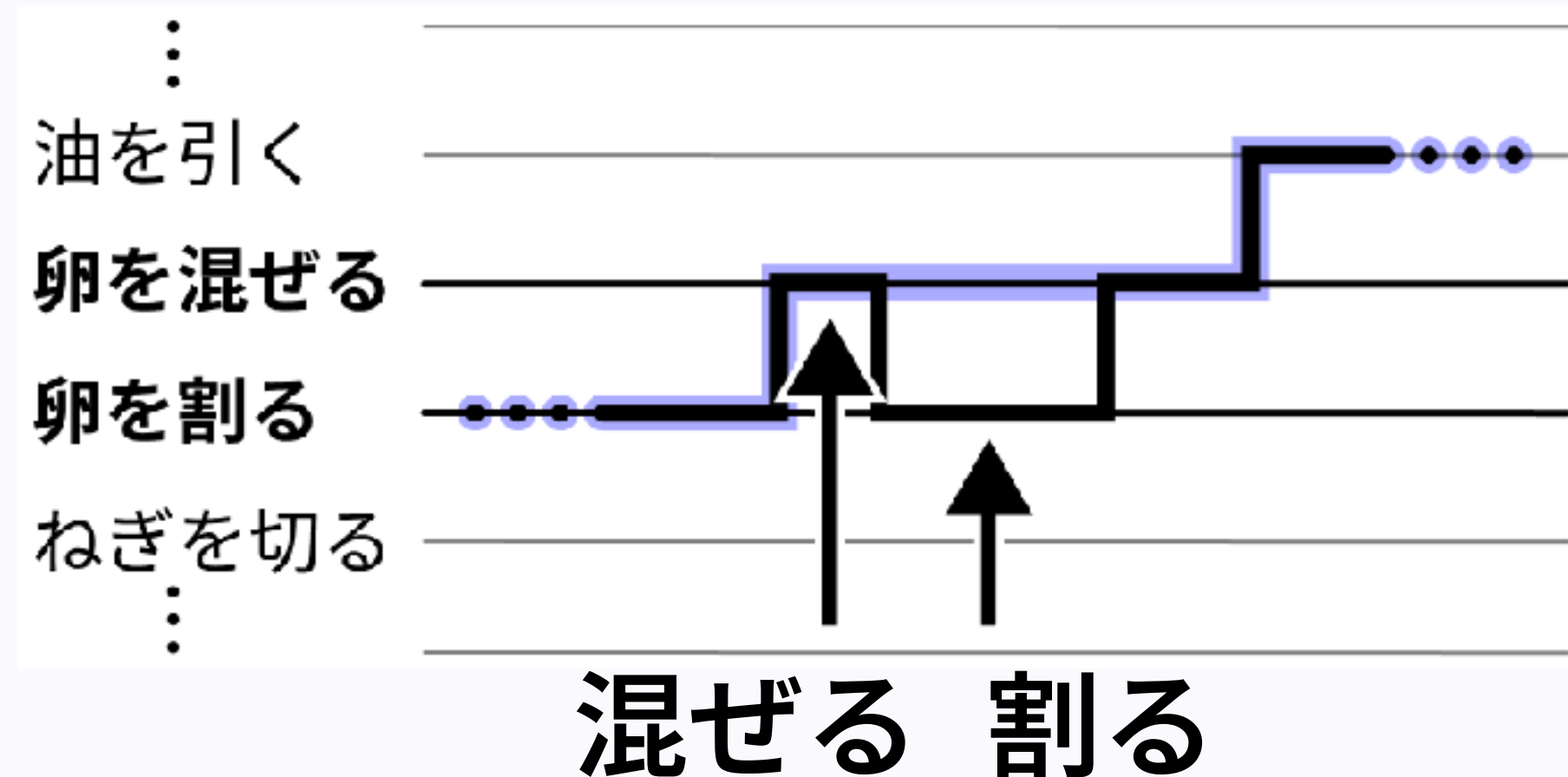
— 位置情報と組み合わせる —

「盛り付ける」が「炒める」と混同されやすい



調理台であれば「盛り付ける」
コンロ前であれば「炒める」とであると絞り込める

工程の推定結果



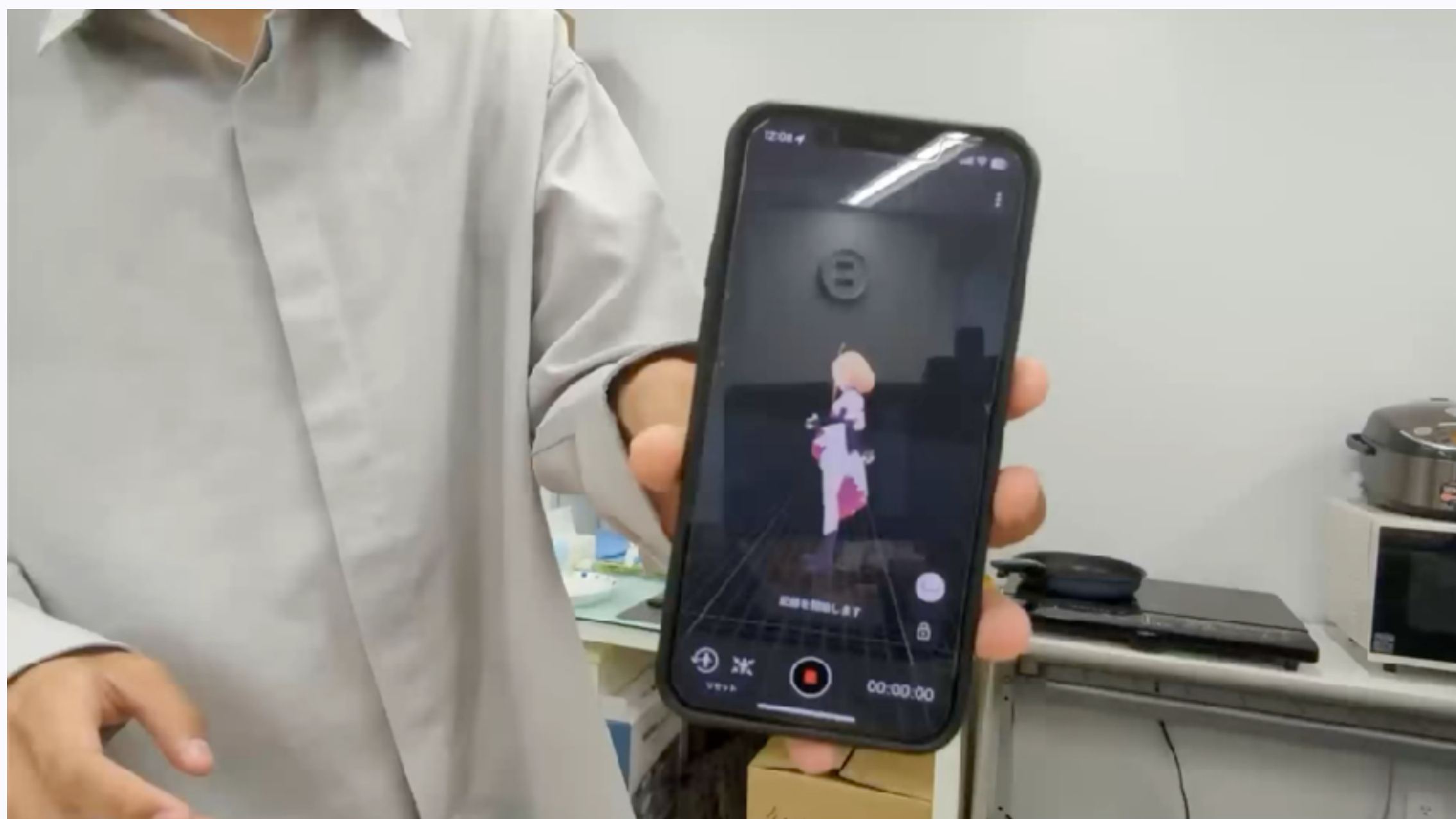
— 手順書と組み合わせる —

「卵を混ぜる」の後に「卵を割る」がきている



卵を混ぜる前に卵を割るという工程が必要なため
誤りであるとわかる

実際の映像



取得した骨格情報



深層学習に対するメリット

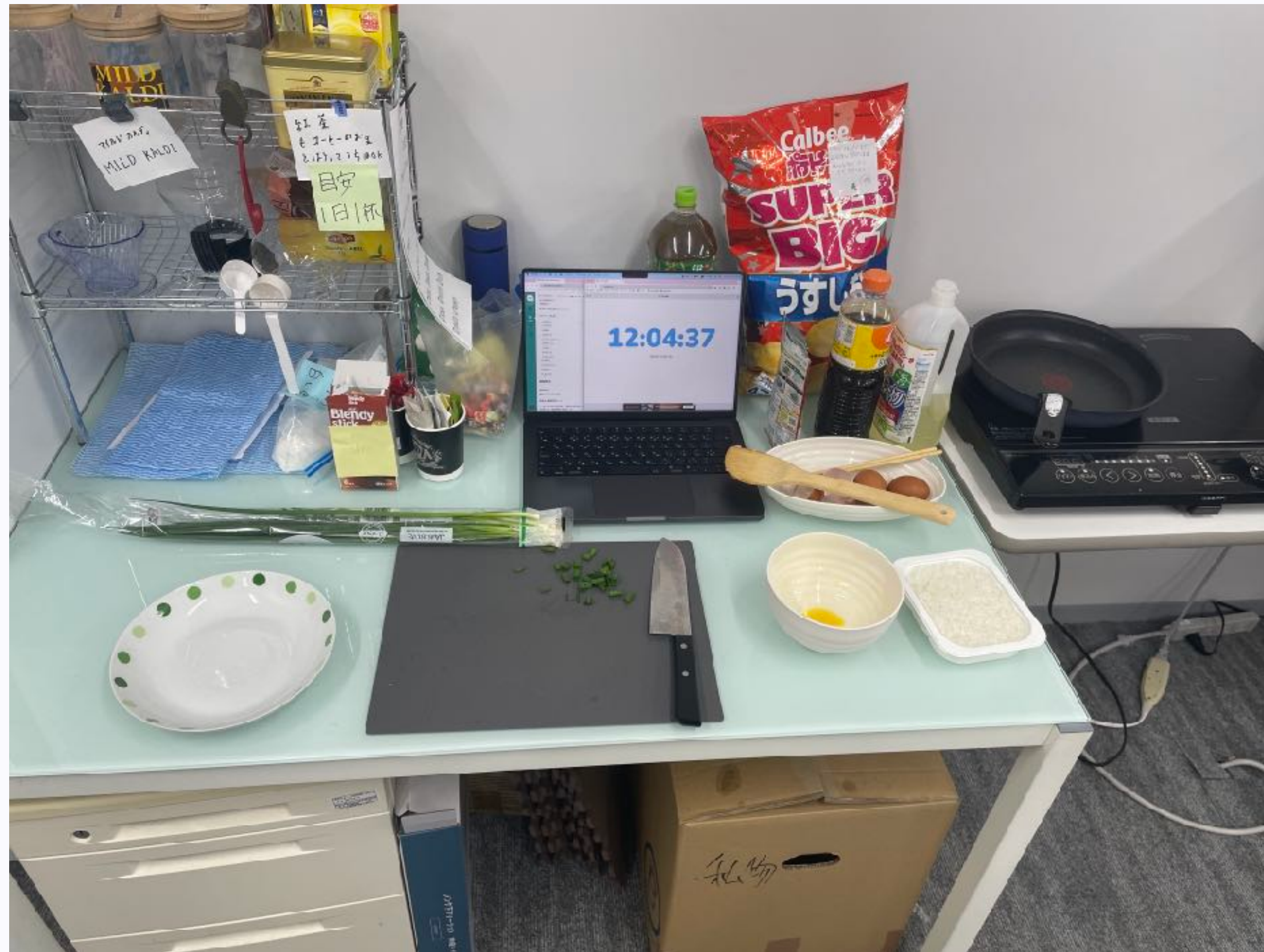
- 少量のデータでも高い精度を出せる
- 計算コストが低い
- 過学習しにくい※

行動認識と位置推定と手順書の組み合わせによる
精度向上を目標としている



今回は機械学習単体では精度よりも手軽さを求めるため
従来の機械学習アルゴリズムを採用

材料や器具の初期位置



レシピを手順書として用いる



準備は工程に含まれないため
あらかじめ用意

冷蔵庫から取り出す動作等も
認識が期待できる

冷蔵庫内の位置は固定ではない

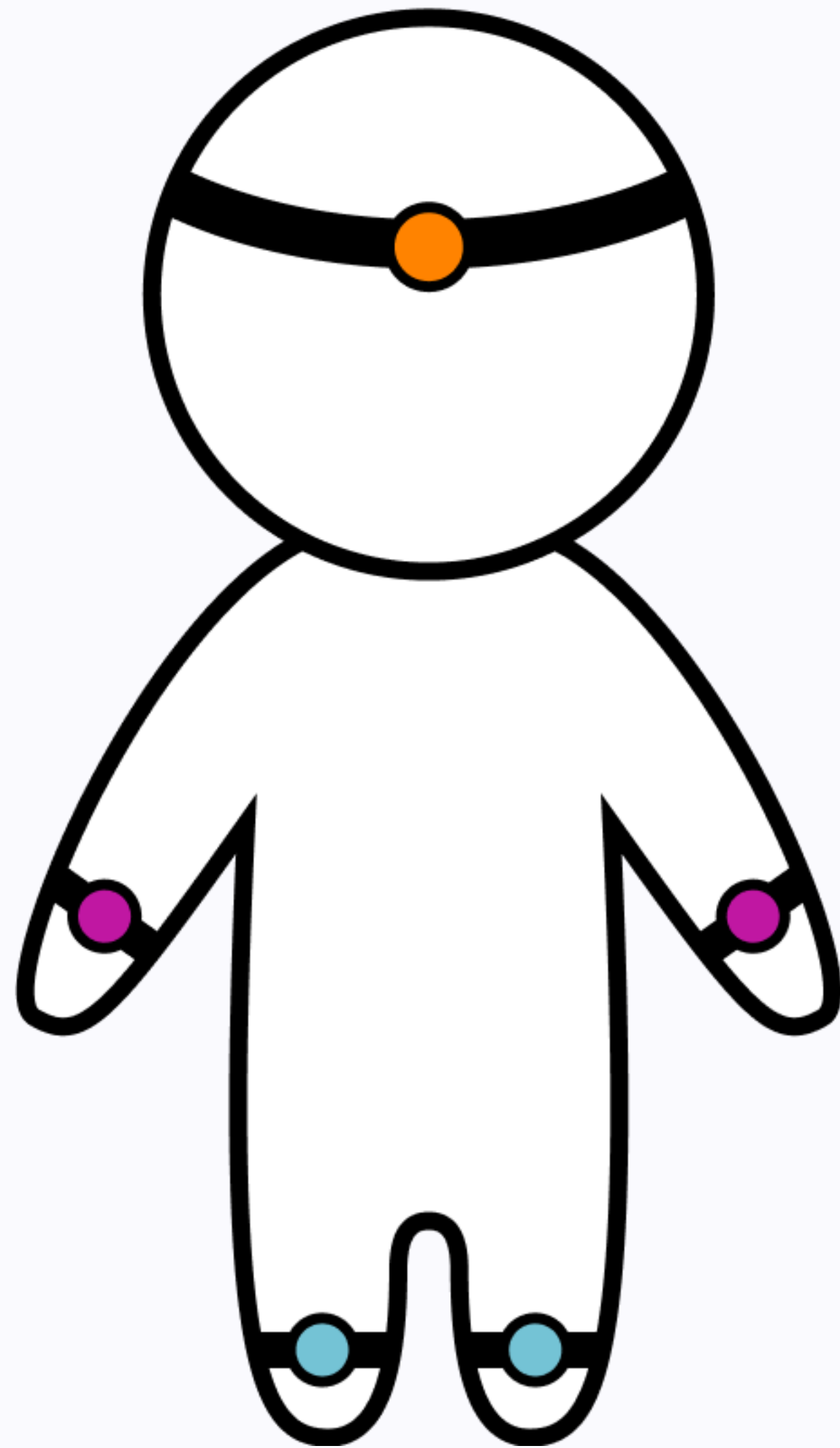


冷蔵庫の位置や開閉動作をもとに
認識できるのでは

レシピのような手順書がある行動であれば
他の行動においても期待できる

例) 工場における生産過程 (セル生産方式)

部品を取る ▶ ネジを複数箇所締める ▶ 次に渡す



頭・両腕・腰・両足の計6箇所に装着
加速度と角速度をセンシング



mocopi公式アプリ※にBluetoothで送信
アプリ側で機械学習を用いて骨格を推定



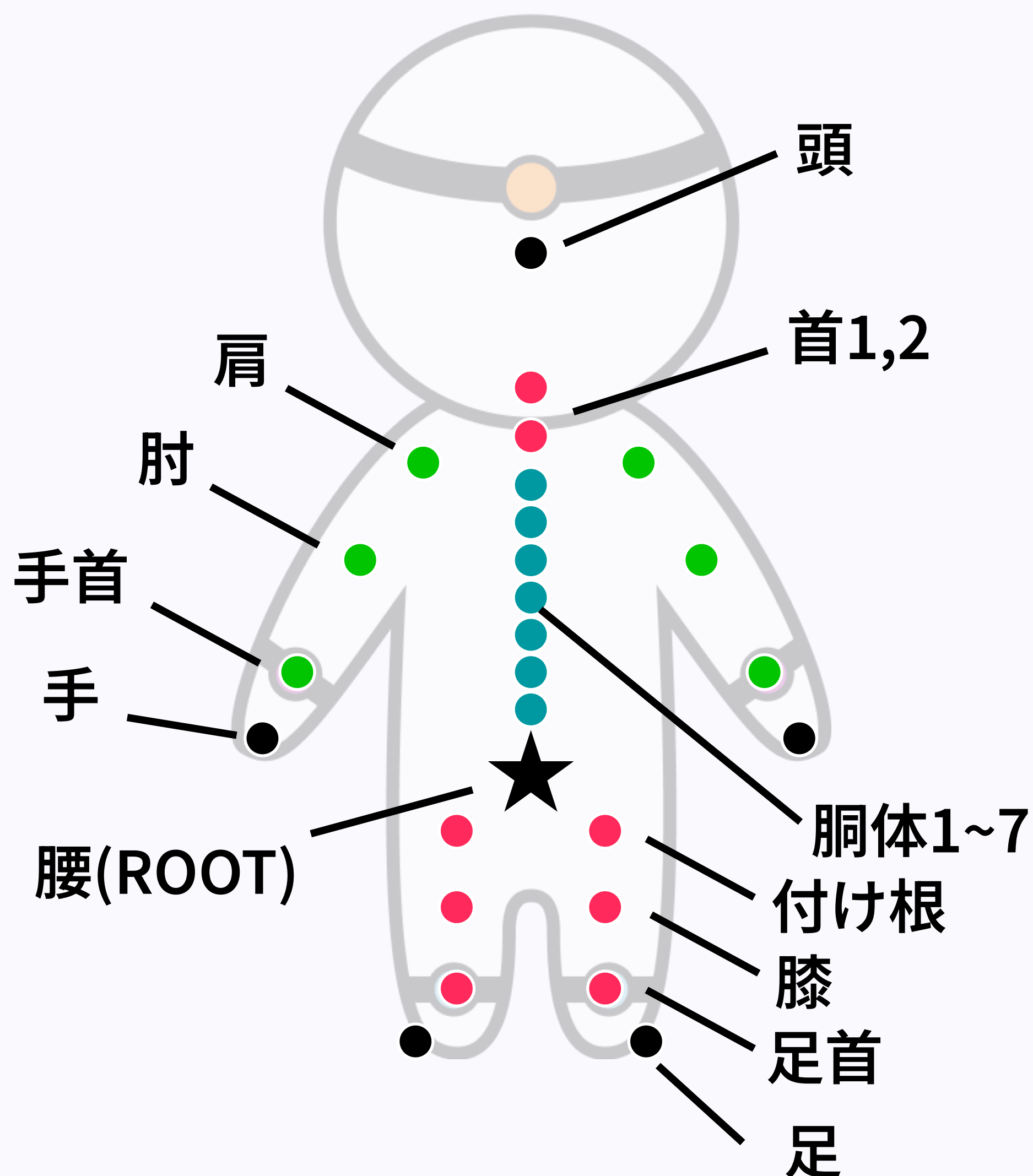
動画

BVH形式

生データ

※骨格まで出してくれるアプリと生データを取得するだけの二種類ある

BVHファイルに含まれる関節



腰を基準として親関節からの相対的な角度と位置を示す
腰の位置は開始地点を基準とした相対的な位置を示す

※腰以外の位置は常に固定値

6箇所センサーをもとに
mocopiアプリが各関節を推定している

HIERARCHY部では骨格構造を定義している

```
1 HIERARCHY
2 ROOT root
3 {
4   OFFSET 0 93.2929 0
5   CHANNELS 6 Xposition Yposition Zposition Zrotation Xrotation Yrotation
6   JOINT torso_1
7   {
8     OFFSET 0 5.07867 -1.15138
9     CHANNELS 6 Xposition Yposition Zposition Zrotation Xrotation Yrotation
10    JOINT torso_2
11    {
12      ...
13      {
14        OFFSET -2.53307e-17 9.49737 -0.914492
15        CHANNELS 6 Xposition Yposition Zposition Zrotation Xrotation Yrotation
16        JOINT torso_7
17        {
18          ...
19          {
20            OFFSET -2.00958e-17 4.83041 0.765062
21            CHANNELS 6 Xposition Yposition Zposition Zrotation Xrotation Yrotation
22            End Site
23            {
24              OFFSET 0 0.1 0
25            }
```

MOTION部ではフレームごとの
各関節の角度と位置を示している

```
1 MOTION
2 Frames: 33520
3 Frame Time: 0.016667
4 0 93.2929 0 -0 0 -0 0 5.07867 -1.15138 -0 0 -0 0 5.61661 1.07143 -0 0 -0 -1.02604e-17 ...
5 -47.0603 92.8165 -11.3194 -3.42661 -2.56392 -108.244 0 5.07867 -1.15138 ...
6 -47.0569 92.8159 -11.3133 -3.429 -2.55492 -108.242 0 5.07867 -1.15138 ...
7 -47.0535 92.8156 -11.3069 -3.43599 -2.54376 -108.239 0 5.07867 -1.15138 ...
8 -47.0497 92.8157 -11.3006 -3.44299 -2.53261 -108.235 0 5.07867 -1.15138 ...
9 -47.0454 92.8155 -11.2947 -3.44312 -2.52127 -108.233 0 5.07867 -1.15138 ...
10 -47.0411 92.8153 -11.2887 -3.44324 -2.50994 -108.23 0 5.07867 -1.15138 ...
11 -47.0366 92.8143 -11.2821 -3.44324 -2.50993 -108.23 0 5.07867 -1.15138 ...
12 -47.032 92.8145 -11.2748 -3.46313 -2.50869 -108.237 0 5.07867 -1.15138 ...
```

処理には全てPythonを使用

BVHファイルを mcp-persor※を用いてパース



- HIERARCHY部 をオブジェクトとして取得
- MOTION部を DataFrame として取得
- 関節名一覧を取得
- 初期位置の操作
- 各情報の上書き
- CSV, BVH形式での出力 等

※mocopiのBVHファイルをパースするための自作ライブラリ (公開済)