NS Solutions

NIPPON STEEL

Meta Quest3におけるQRコード検出および追跡性能評価に関する中間報告

名前：土尾崇太

発表日：2025/09/11

目次

- 自己紹介
- 研究背景・目的
- システム構成・全体フロー
- 基本性能検証
- 応用実装例：QRコードによる家具配置
- 応用展開・今後の展望
- まとめ

自己紹介

- 名前：
  - 土尾崇太（つちお しゅうた）
- 所属：
  - 筑波大学 システム情報工学研究群 修士１年
  - 産業技術総合研究所 人工知能研究センター 契約職員 兼 技術研修員
- 研究内容：
  - 異常検知AIシステムに関する研究（成果発表：５件）
  - 医療支援AIに関する研究
- 趣味・特技：
  - ラーメンを食べる，コーヒーを淹れる・飲む
  - ラテアート（ハート）

研究背景：社会的背景と技術的なトレンド

- AR/VR技術の発展

→ Meta Quest 3などの最新HMDでは，現実世界を認識・活用する機能（パススルーカメラ、空間認識）が進化している

- 現実世界との連携の重要性

→ デジタルと物理空間を融合した応用（例：AR家具配置、遠隔作業支援など）ニーズが増加

- 現状の課題

→ 現実空間の物体や位置情報をリアルタイムにシステムへ取り込む手法の確立が不十分

研究背景：技術的背景と研究目的

- Meta Quest 3のカメラAPI解放

→これによりユーザー独自の画像解析・物体認識が可能になった

- QRコード活用の意義

→安価かつ確実に「場所」「ID」「属性」など様々な情報を埋め込むことができる

- 従来技術/既存事例との差分

→ 既存のARマーカーは認識数や自由度に制限があった

- 研究目的

→ Quest 3でQRコードの検出・追跡性能を評価し，実用的な応用可能性を探る

→ 検出・追跡性能の実験的評価

→ QRコードに家具情報を埋め込み，仮想家具配置システムを試作

研究背景：技術的背景と研究目的

- 研究目的

→ Quest 3でQRコードの検出・追跡性能を評価し，実用的な応用可能性を探る

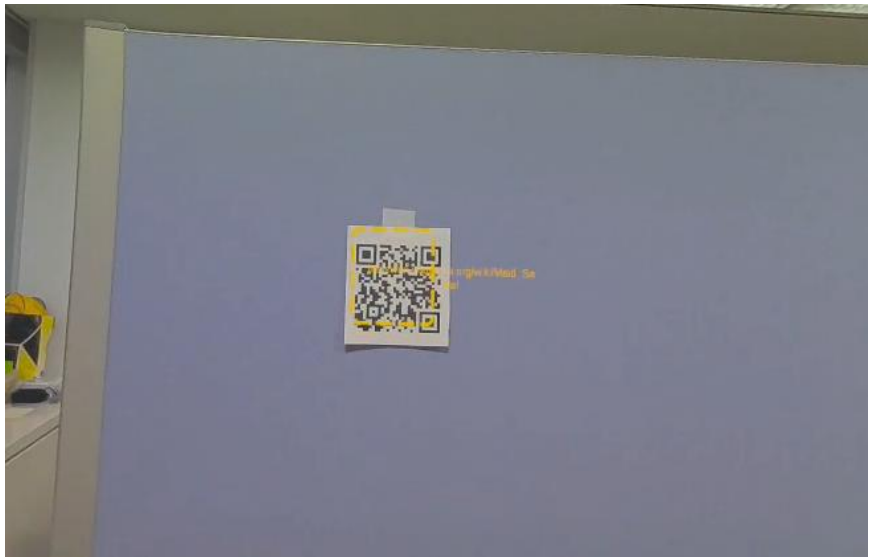
→ 検出・追跡性能の実験的評価

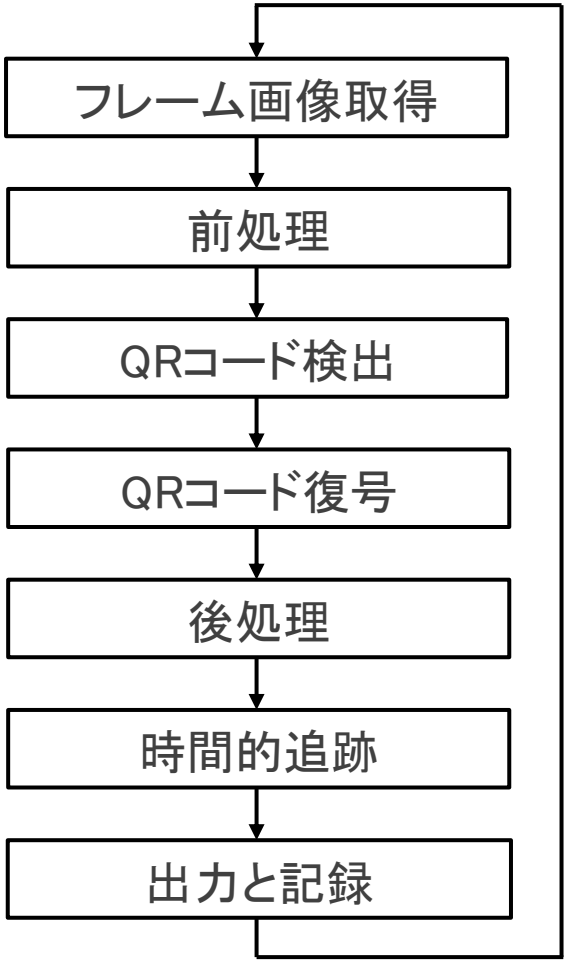
→ QRコードに家具情報を埋め込み，仮想家具配置システムを試作

基本性能検証：QRコード検出の流れ

- カメラAPIによりQRコードが検出・追跡

↓検証の様子↓

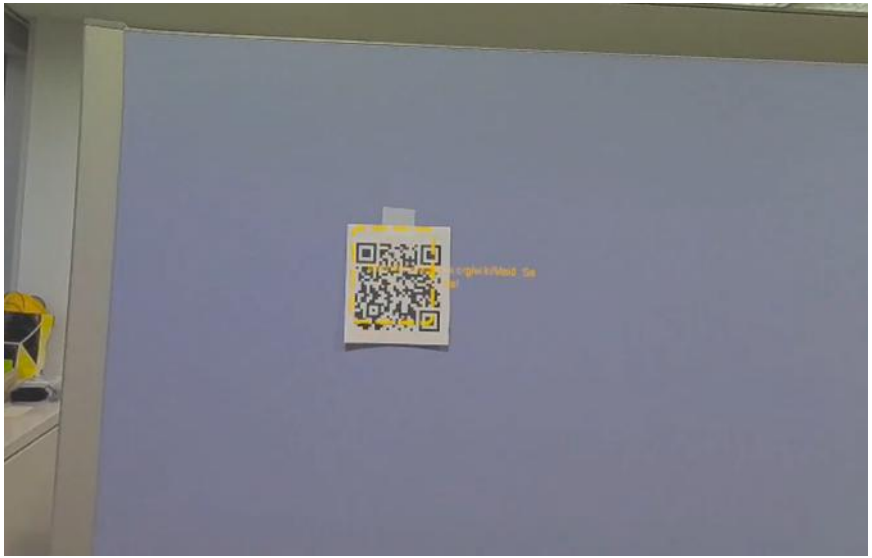


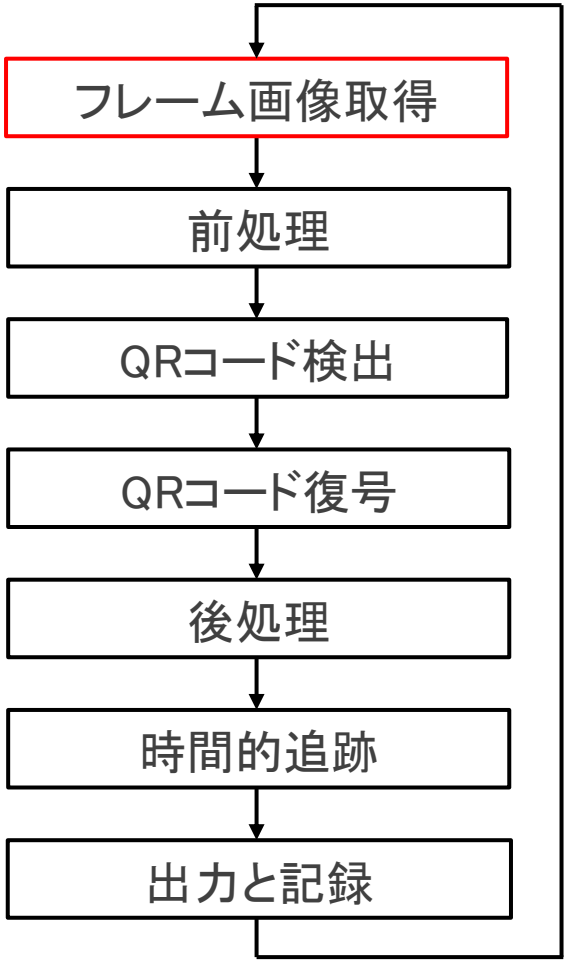


基本性能検証：QRコード検出の流れ

- フレーム取得

→ API 利用によりカメラ映像を取得可能





基本性能検証：QRコード検出の流れ

- 前処理

→ 取得した画像の解像度を削減（今回はs=1）

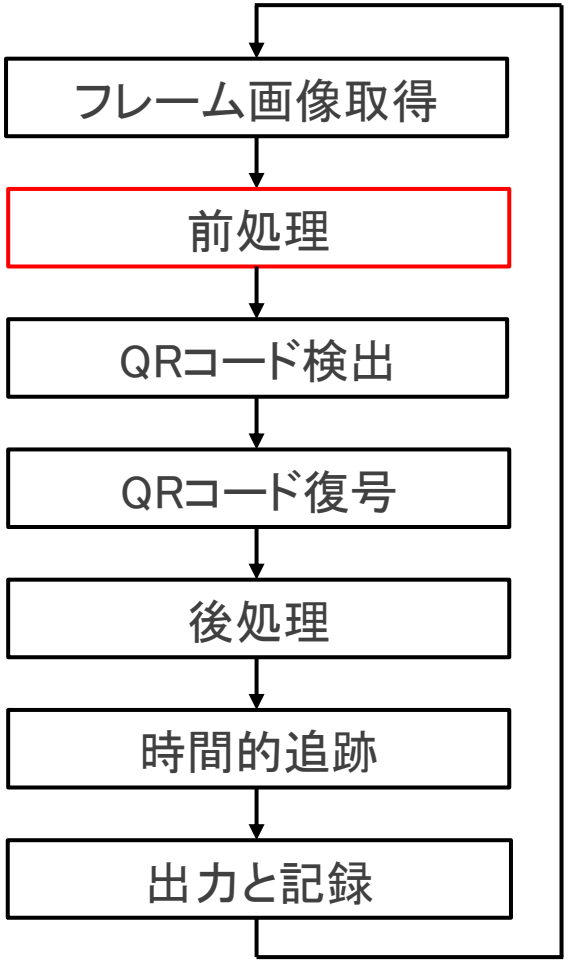
$$W' = \left\lfloor \frac{W}{s} \right\rfloor, H' = \left\lfloor \frac{H}{s} \right\rfloor$$

→ RGBからグレースケールに変換

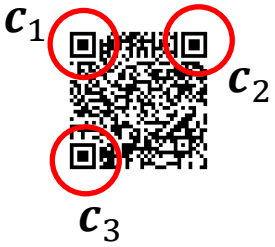
→ 2値画像に変換

（今回はライブラリを用いた輝度計算）

（例：$Y = 0.21R + 0.72G + 0.07B$）



基本性能検証：QRコード検出の流れ



- QRコード検出・復号

→ Finder Pattern検出

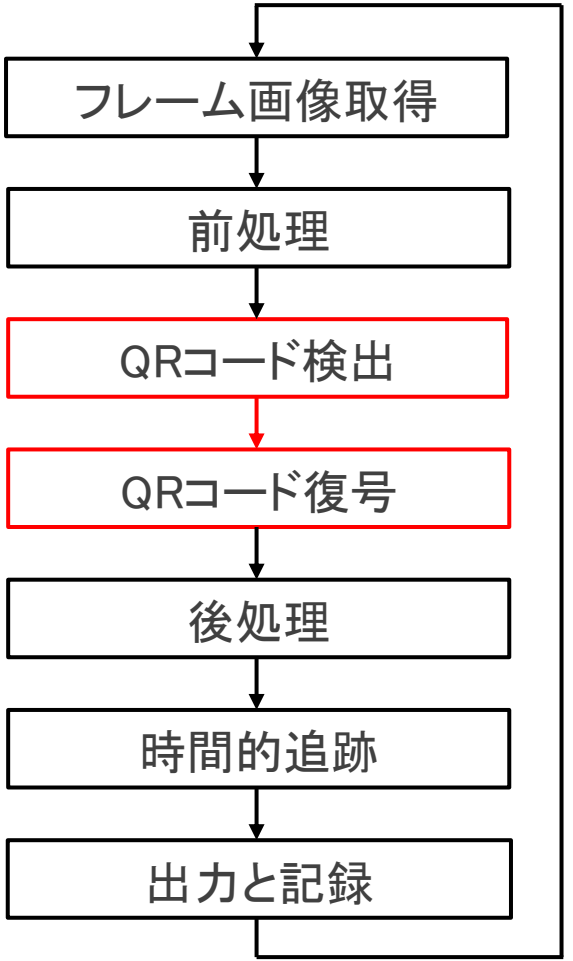
$$\phi = \frac{1}{5}\sum_{k=1}^{5}\left|\frac{r_k}{\bar{r}} - q_k\right|, (q_1, q_2, q_3, q_4, q_5) = (1,1,3,1,1)$$

条件を満たす点群を探索し3点の座標(c_1, c_2, c_3)を得る

→ 射影補正+誤り補正

- 最小二乗法による4辺直線推定+交点算出
- 符号理論：$S_j = R(\alpha^j), (\Lambda, \Omega)$

→ 2つ目以降も同様に処理



基本性能検証：QRコード検出の流れ

◆ 後処理

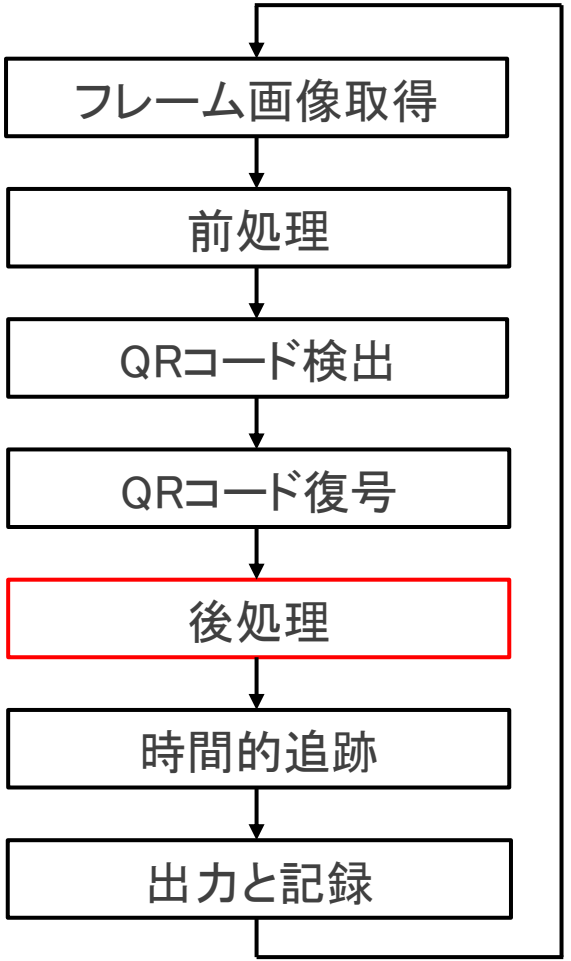
→ uv正規化

画像幅W，高さH，検出コーナー(x_i, y_i)をuv座標へ

$$u_i = \frac{x_i}{W}, v_i = \frac{y_i}{H}, (u_i, v_i) \in [0,1]^2$$

→ QRコード重複確認・統合

・同一のQRであればいずれかを削除



基本性能検証：QRコード検出の流れ

◆ 時間的追跡

→ 前フレーム集合との対応付け（ID付与）

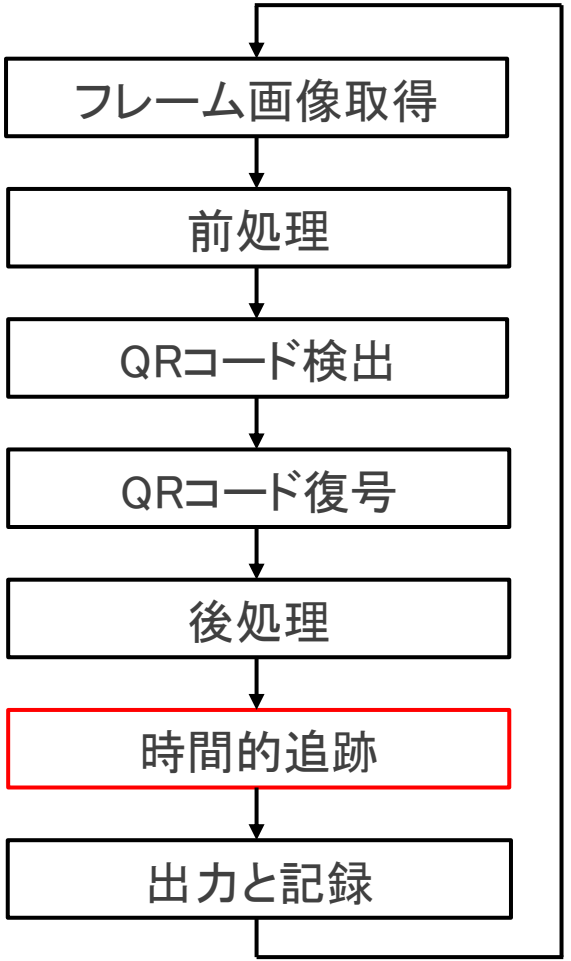
・ 検出結果をフレーム毎に紐づけて同じQRとして認識

姿勢を安定化し，欠落・ノイズ・遅延に耐性

→ 消失・再取得管理

・一時的にQR消失してもしばらく保持

待機時間を過ぎると削除



基本性能検証：QRコード検出の流れ

- 出力と記録

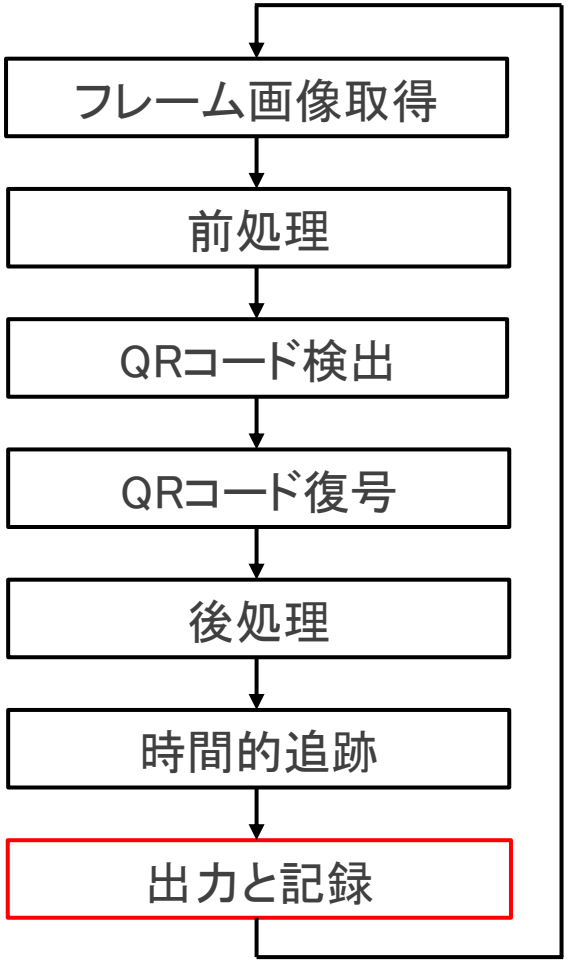
→ ワールド座標・姿勢計算（PnP：2Dから3D）

QR四隅の2D位置，正方形のサイズ，カメラの設定を用いて3D空間上での向きと距離を逆算

$$\min_{R,\boldsymbol{t}} \sum_{i} \left|\left|[\boldsymbol{x}_i]_{\times}(RX_i + \boldsymbol{t})\right|\right|^2$$

→ オーバーレイ描画（角線・ID）

→ 計算時間等の各種ログ出力



基本性能評価

◆ 実験項目

- 同時認識可能なQRコード枚数と処理時間
- 最小認識可能QRコードサイズと認識距離
- 追跡性能（動くQRの追従性）
- QRコードバージョンごとの認識性能

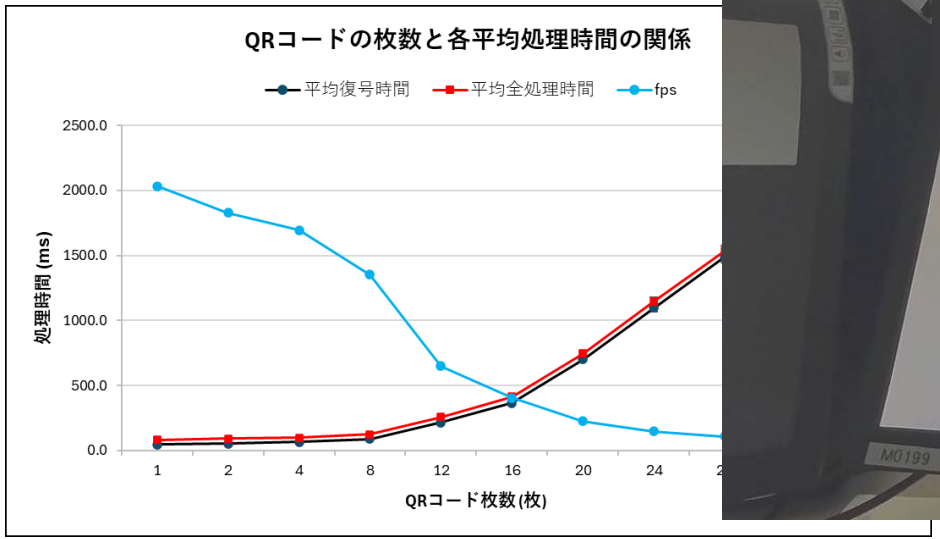
◆ 実験環境・条件

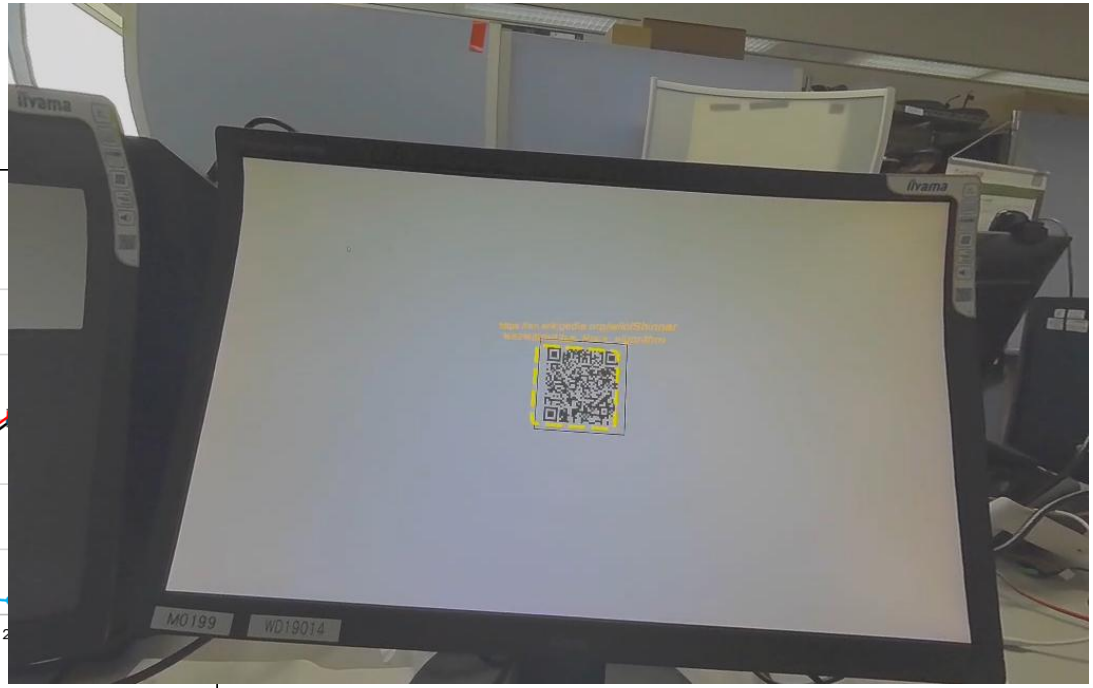
- 使用デバイス：Meta Quest3
- QRコード生成方法：Pythonのライブラリ（qrcode）使用，v3で生成

基本性能評価：同時認識可能なQRコード枚数と処理時間

- 枚数増加による処理時間の変化を検証

→ 枚数の増加に伴い処理時間は**増加⤴⤴**





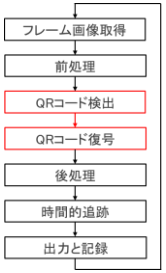
基本性能評価：同時認識可能なQRコード枚数と処理時間

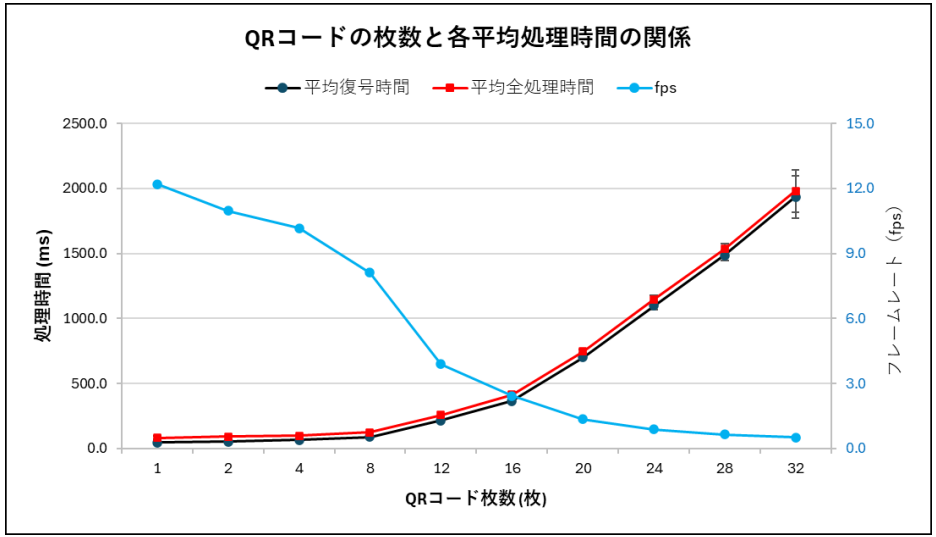
- 枚数増加による処理時間の変化を検証

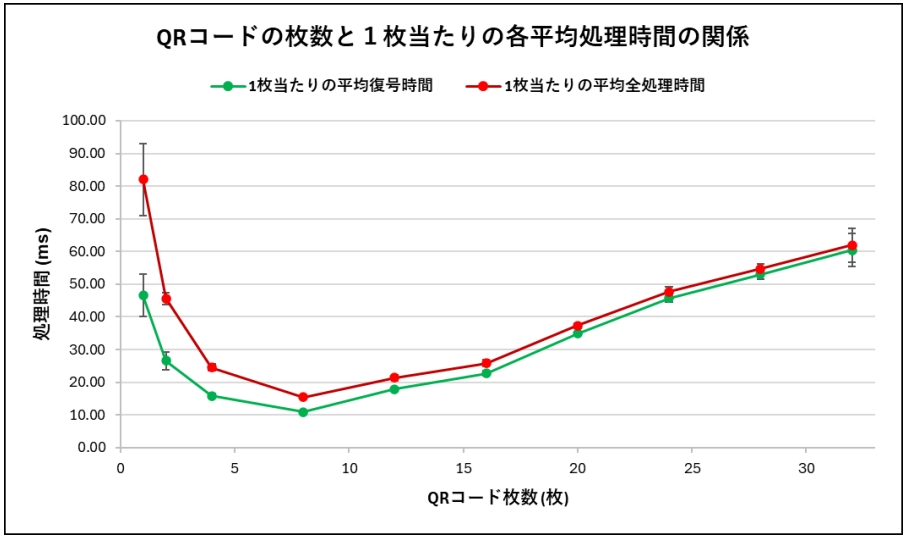
→ 枚数の増加に伴い処理時間は**増加↑↑**

→ **検出・復号**にほとんどの時間を要する

→ QRコード1枚の場合約12fps





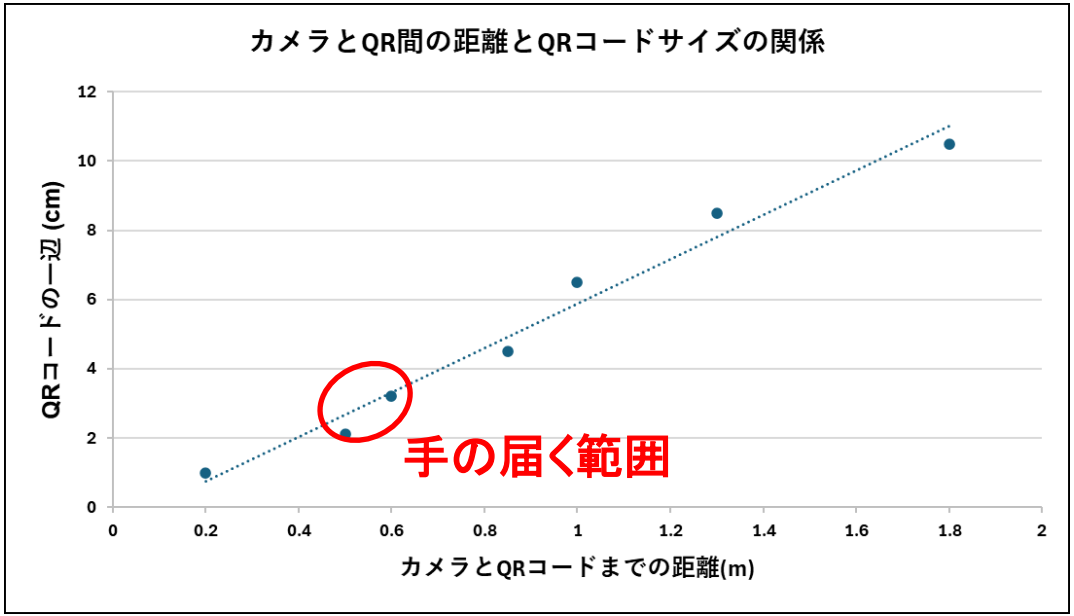


基本性能評価：最小認識可能サイズ

- 最小認識可能サイズと距離の関係を検証

→ 1枚のQRコードを用いて徐々に遠ざけていき認識が失敗する直前の最小サイズ・最大距離を記録(QR:3)

→ QRコードのセル1つに対し平均2.1ピクセル



性能評価：追跡性能

- 追跡性能評価

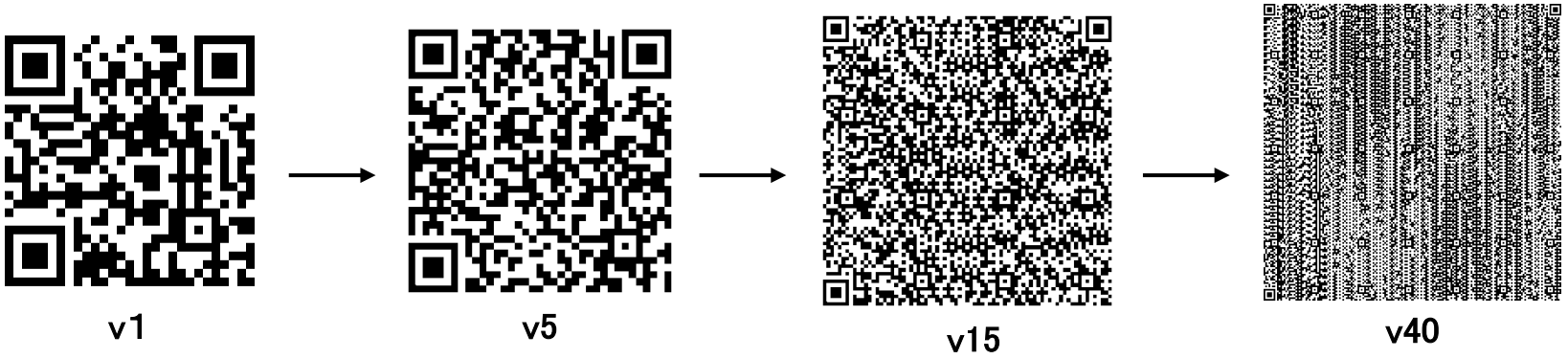
→ ベルトコンベア上にQRコードを置き，一定速度で移動(約10cm/s)

本ベルトコンベアの最高速度

性能評価：バージョン毎の認識性能

- バージョン毎の認識

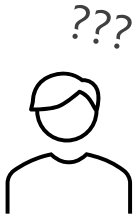
→ QRコードのバージョン（情報量やセル数が異なる）をv1～v40まで変える



version	1	5	10	15	20	25	30	35	40
結果	○	○	○	○	○	○	○	○	×

応用実装例：QRコードによる家具配置

- **応用アイデア・目標**
- QRコードを現実空間にかざすだけで家具を自動配置できる仕組みを実現



応用実装例：QRコードによる家具配置

- 応用アイデア・目標
- QRコードを現実空間にかざすだけで家具を自動配置できる仕組みを実現

応用実装例：家具配置アルゴリズム

- 座標計算の流れ
  - QRコードの検出位置から3D空間の座標へ変換
  - 天井，壁，床の判定
  - 家具の向きと位置決定

応用実装例：デモ

- ソファ配置のデモ動画

・ QRコードの向きによらずソファの向きは一定

・ どの高さにQRがあろうとソファは床に設置

応用実装例：デモ（色変更）

- ソファ配置のデモ動画

・ 色情報を持つQRを読み込むとソファの色が変化

まとめ

- 本研究の成果
  - Quest 3でQRコードの検出・追跡性能を評価し，実用的な応用可能性を探る
  - 検出・追跡性能の実験的評価
  - QRコードに家具情報を埋め込み，仮想家具配置システムを試作

- 今後の課題など？？