アプリ導入作業マニュアル

概要

このマニュアルでは、このアプリの現場導入に必要な作業を記述します。

大まかな作業の流れ

1. **デバイス設置作業**
2. **カメラパラメータ調整作業**
3. **マスター画像撮影＆特徴量生成作業**
4. **モデルパラメータ/閾値調整作業**

デバイス設置作業

作業のゴール

- 検査アプリに必要なデバイスが接続・設置されていること。

作業手順

1. Jetsonの電源を接続し、ログインする。（ユーザー名: `user` / パスワード: `0000`）
2. JetsonとタッチパネルディスプレイをHDMIケーブルで接続する。
3. JetsonとスピーカーをUSBケーブルで接続する。
4. 必要であればJetsonにWiFiモジュールを接続する。
5. カメラにレンズを取り付ける。
6. カメラを雲台と接続する。
7. 雲台を充填機の固定金具に取り付ける。
8. JetsonとカメラをLANケーブルで接続し、カメラの電源ランプが点灯していることを確認する。
9. LANケーブルを天井配線する。

カメラパラメータ調整作業

作業のゴール

- 導入するラインの最適パラメータを決定し、システムのカメラパラメータを記述した`config/base_camera_params.json`が更新されていること。

作業手順

1. レンズの絞りとWD（ワーキングディスタンス）を最小に設定する。

2. Baumer Camera Explorerでカメラパラメータを決定する。
    - パラメータ調整の大方針
        - 今回の判定ではそこまで画質が求められないが、連続撮影でぶれないようにフレームレートの速さが求められる。
        - フレームレートを稼ぐために、縦横のビニングを2×2に設定しておく。
        - フレームレートの目安としては**60fps**。それ以下だと連続撮影でブレが発生する。
        - 下記のようなパラメータを調整して、フレームレートと画像の明るさのバランスを調整する。
    - 調整が必要なパラメータ
        - `ExposureTime`
            - 露光時間を大きくすると画像は明るくなるが、フレームレートは遅くなってしまう。
            - 15000μsにすると、1/15000μs=66.7fps程度がフレームレートの上限になるので、12500μs(=80fps)ぐらいまで下げられるとブレはなくなると思われる。
        - `ROI`
            - ROIを小さくするほどフレームレートは上がるが、撮像範囲は小さくなる。
            - ROIのパラメータは下記の通り。
                - `Width`: 横の画素数
                - `Height`: 縦の画素数
                - `OffsetX`: オフセットX座標
                - `OffsetY`: オフセットY座標
        - `Gain`

        - 基本は1のままが理想だが、フレームレートがなかなか上がらない場合は、`Gain` を上げて明るさを担保する。
        - あまり上げすぎると画像にノイズが増えて判定に影響を及ぼす可能性が高いのでおすすめしません。

    - Baumer Camera Explorerを用いたプレ撮影

        - `docs/管理者用/camera_explorer_manual.md`を参照。
        - Baumer Camera Explorerで画像を撮影し、画質を確認する。
        - Baumer Camera Explorerで撮影した画像と検査アプリでの撮影で細かいパラメータの違いがあり画質が若干異なるため、**この時点で撮影した画像を検査のマスター画像として採用するのはおすすめしません。**

3. 検査アプリのカメラパラメータの値を更新する。

    - 2.の手順で決定したパラメータをもって、`config/base_camera_params.json` の値を更新する。

マスター画像撮影＆特徴量生成作業

`master_add_update_manual.md`を参照。

閾値調整作業

作業のゴール

- モデルパラメータと閾値を実験から決定し、システムのモデルパラメータを記述した`config/base_model_params.json`が更新されていること。
- `master`フォルダにある各製品の閾値を記述した`model_params.json`の`threshold`の値が更新されていること。

事前準備

- **マスター画像(`master.bmp`)**: `master`フォルダ内の各製品のフォルダに配置する。
- **検証用画像**: 充填機を回して連続撮影した各製品の画像。

  ※ マスター画像や検証用画像を撮影するための**画像撮影アプリ**の使用方法については、`master_add_update_manual.md` を参照。

作業手順

1. **画像撮影アプリ**で撮影した検証用の画像を、`data`フォルダに下記のような形でデータを配置する。

```
data/
├── 2025_07_17/
            ├── NKR/
            │   ├── image0000001.bmp
            │   ├── ...
            │   └── image0000200.bmp
```

```
        ├── ファーストキッチン/
        │   ├── image0000001.bmp
        │   ├── ...
        │   └── image0000200.bmp
        ├── マルガク/
        │   ├── image0000001.bmp
        │   ├── ...
        │   └── image0000200.bmp
        ├── 特級/
        │   ├── image0000001.bmp
        │   ├── ...
        │   └── image0000200.bmp
        ├── 標準/
        │   ├── image0000001.bmp
        │   ├── ...
        │   └── image0000200.bmp
        └── 無地/
            ├── image0000001.bmp
            ├── ...
            └── image0000200.bmp
```

2. モデルパラメータを決定するための実験用コード(`analyze_score.py`)を動かしてパラメータと閾値を決定する。
    - コード概要
        - このコードは、`master`フォルダにある各製品のマスター画像をもとに、指定ディレクトリ内のBMP画像に対して異常度計算を行い、時系列プロットを生成するコード。
        - 例えば、6製品検査対象となる製品がある場合は、6製品分のマスター画像×6製品の検証用画像=36パターンの時系列プロットを生成して、総当たりで異常度を確認できる。
    - 検査アルゴリズムについて
        - 今回のモデルはAnomalyDINOがベースであるが、下記のような内容で判定を実施している。
            1. **連続撮影＆AnomalyDINOによる異常度算出**
                - `system_params.json`内の `video_seconds` で指定された秒数だけ画像を連続撮影。
                - それらすべての画像に対してAnomalyDINOによる異常度の算出を行う。
                - 通常のAnomalyDINOでは最終層の特徴量を異常度計算に用いているが、このアプリでは、どの層の特徴量を抽出するかが選択できるようになっている。（`feat_layer`で指定可能。）
            2. **代表画像の選抜**
                - 上記の画像のうち、AnomalyDINO異常度の最小値をとった画像を代表画像とする。
            3. **Hue平均差の加算**
                - 通常のAnomalyDINOでは色合いの違いのみの変化の検出が難しい。
                - その点を補足するために、代表画像に対して指定領域のマスター画像と検査画像の色合い（Hue平均値）の差を算出してAnomalyDINOのスコアに重み係

数をかけて加算している。
- 最終判定に使用する異常度＝AnomalyDINO異常度 + `hue_weight`*Hue平均差

- 今回の実験で決めなければならないパラメーター一覧

  - AnomalyDINOに関するパラメータ
    - `--model_type`: 使用するモデルタイプ。**dinov2_vits14 固定で問題ないと思われる。**
    - `--feat_layer`: 特徴量抽出層。検査対象に依存するので、**幅広く探索してみたほうがいい。**
    - `--image_size`: 画像サイズ。**504固定で問題ないと思われる。**
  - Hue平均差算出時のパラメータ
    - `--roi_rel`: 平均値算出エリア相対座標 (`y_start`, `y_end`, `x_start`, `x_end`)。**マスター画像のマークが写っているエリアを中心に選択する。**
    - `--sat_thresh`: 彩度しきい値。Hue平均算出時にこの値以下の彩度をもつピクセルは平均算出対象としない。**撮影パラメータによるので幅広く探索したほうがよい。**
    - `--hue_weight`: Hue平均差の重み係数。**0.10程度がベストと考えられる。**

- 実行例

```
python analyze_score.py \
--target_dir data/2025_07_17 \
--feat_layer 5 \
--roi_rel 0.70 0.93 0.11 0.39 \
--sat_thresh 40 \
--hue_weight 0.10
```

- コード実行時の出力ファイル

  - analyzeフォルダに実行日時付きファイルが生成され、その中に下記のファイルが生成される。
    - **AnomalyDINO異常度の時系列プロット（PNG形式）**: プロットされた異常度はAnomalyDINO単体の異常度であり、**Hue平均差は加算されていない。**
    - **最小異常度画像（ROI表示付きBMP形式）**: AnomalyDINOが最小値をとった画像（代表画像）。ここに記載されている異常度は、**Hue平均差が加算された値になっている。**

3. 出力ファイルを確認しながらパラメータ探索を実施し、最適パラメータを決定する。その上で`config/base_model_params.json`の値を更新する。

4. 各製品の閾値を決定し、`master`フォルダにある各製品の閾値を記述した`model_params.json`の`threshold`の値を更新する。