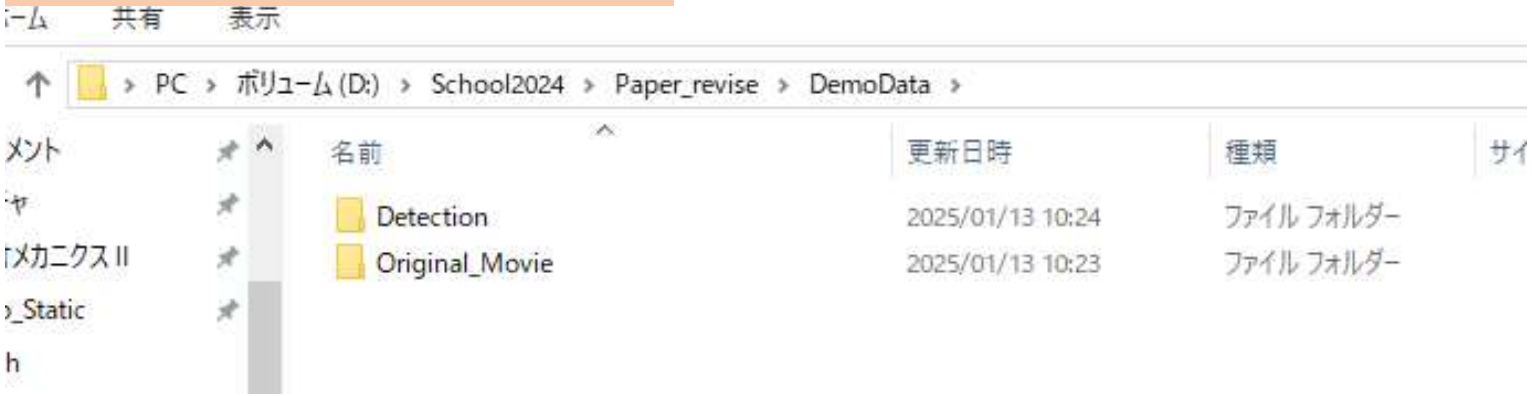
プログラムの流れ

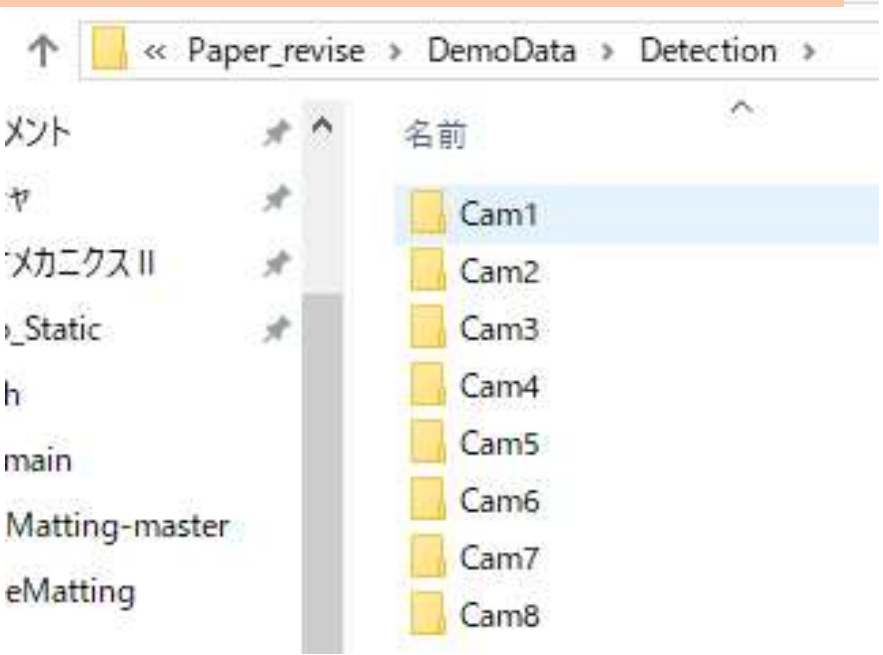
必ずTutorial_6までを終了させてください.

ファイルやフォルダの選択が多くなります.
これについては，プログラムを改善することでほぼ自動化することが可能だと思います

DemoDataフォルダの中身



Detectionフォルダの中には，カメラ台数分の空フォルダがあります．
（カメラパラメータのファイルはあります）

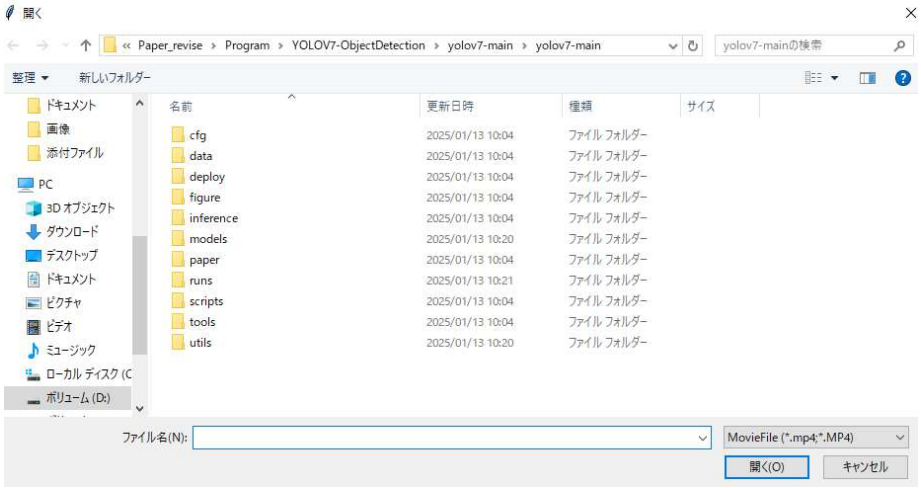


Original_Movieフォルダの中に，8台のカメラで撮影した試技の動画あります．

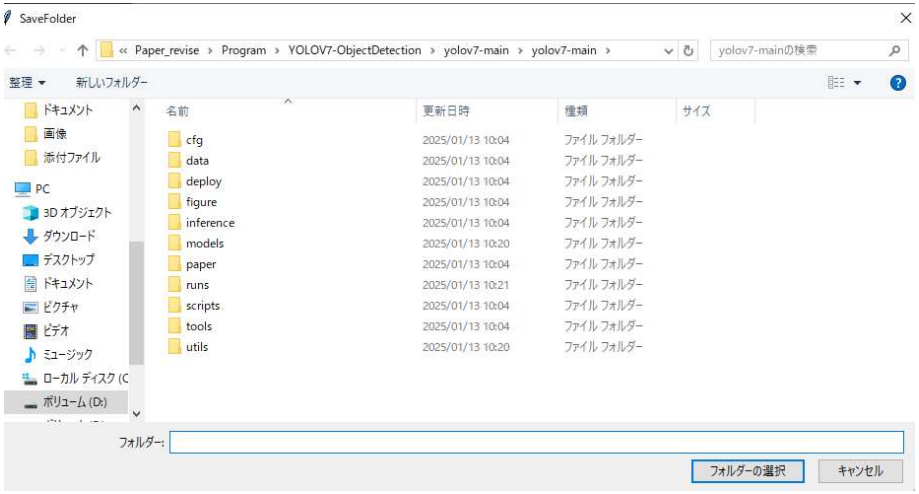
1.Yolov7でヒトを検出する

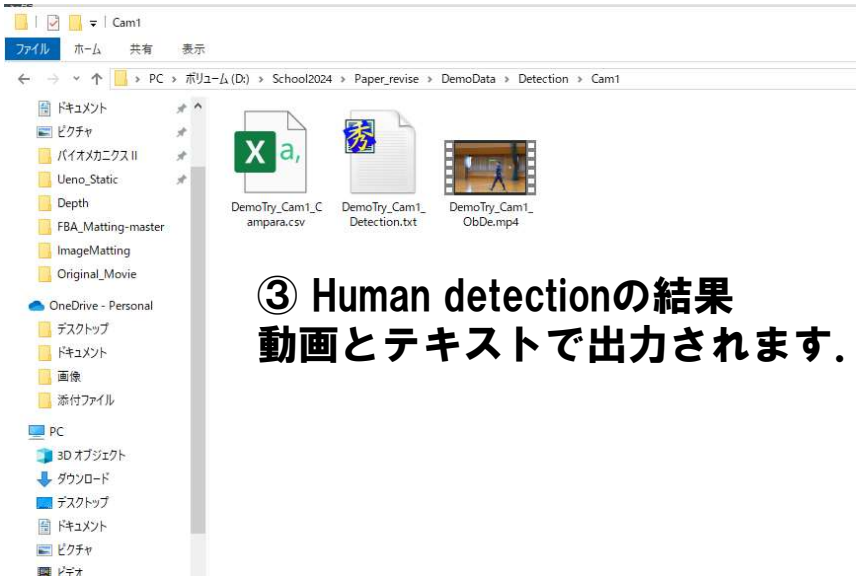
まず，Yolov7の仮想環境を開いて，detect_Human.pyを実行してください．

① Original_Movieの動画を1つ選んでください．



②カメラの番号に従って，
Detection内のフォルダを選んでください．
（DemoTry_Cam1.mp4ならCam1フォルダ）





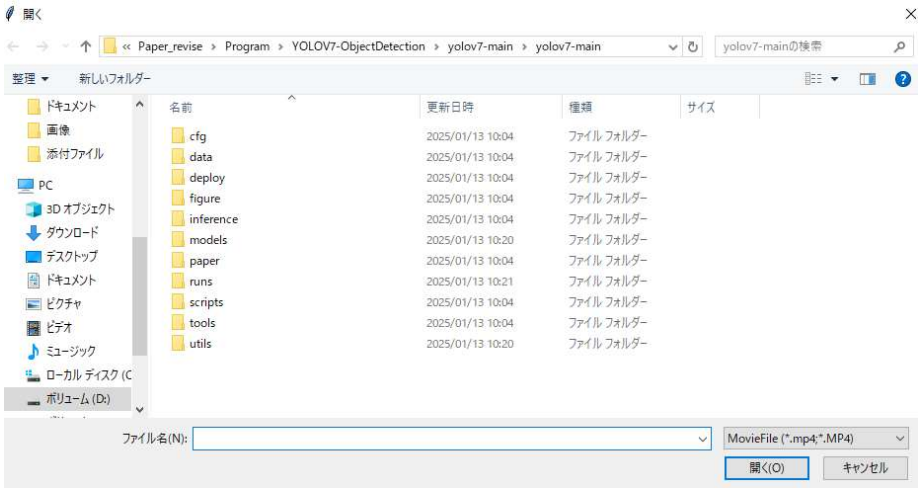
③ Human detectionの結果
動画とテキストで出力されます．

④ これを8台のカメラで行ってください．

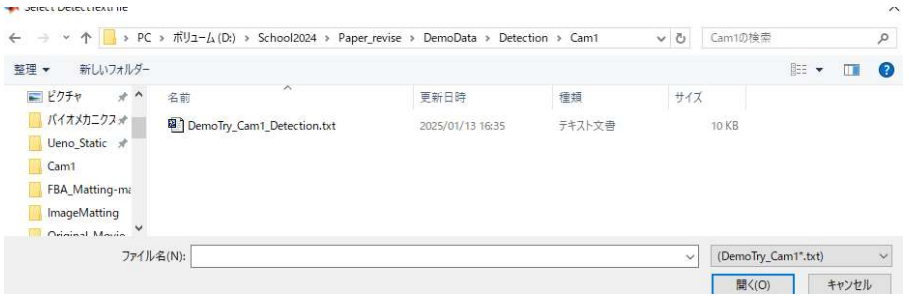
2. ヒトが写っている部分のトリミング

まず，matlabで，Image_Trimming.mを実行してください．

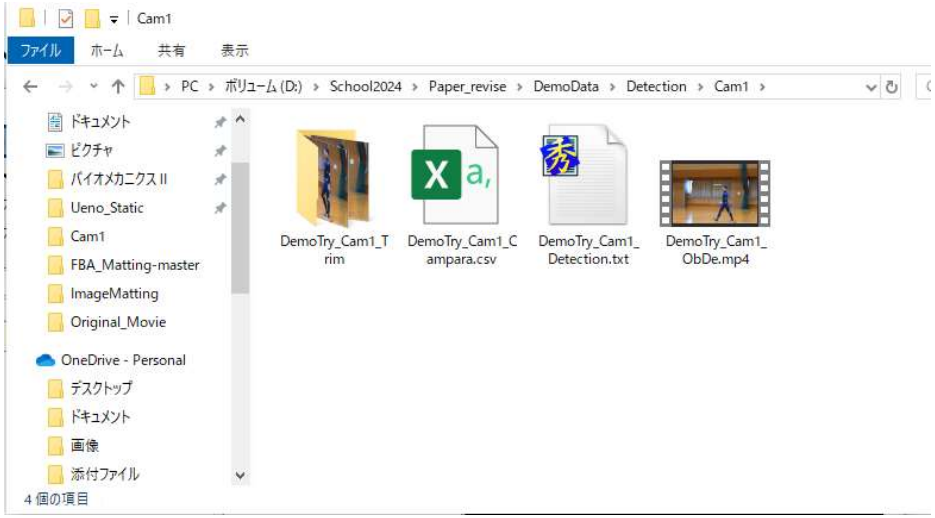
① Original_Movieの動画を1つ選んでください．



②detect_Human.pyで出力された テキストファイルを選んでください． カメラ番号は必ず対応させてください．



③ トリミングされた画像のフォルダができます．

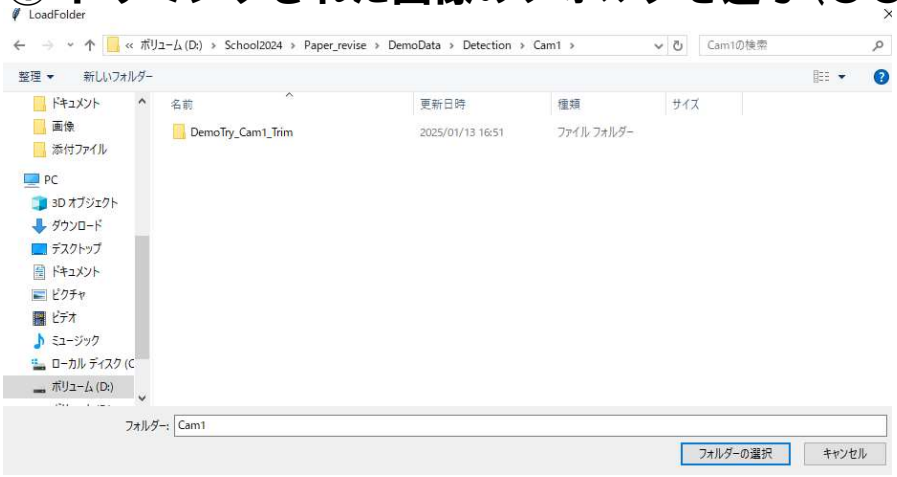


④ これを8台のカメラで行ってください．

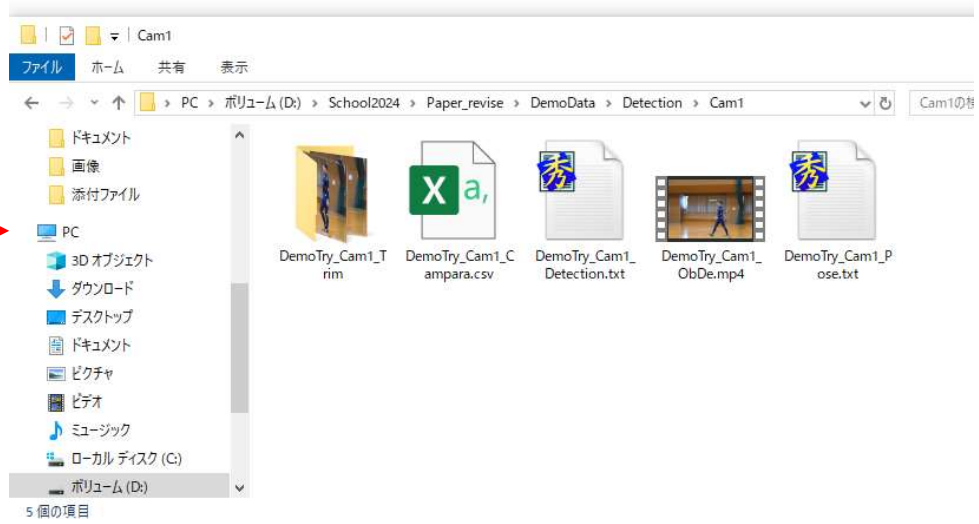
3.Yolov7-poseで姿勢推定する

まず，Yolov7-poseの仮想環境を開いて，pose_Human.pyを実行してください．

① トリミングされた画像のフォルダを選ぶ（〇〇Trim）．



② 〇〇〇_pose.txtが出力されます．

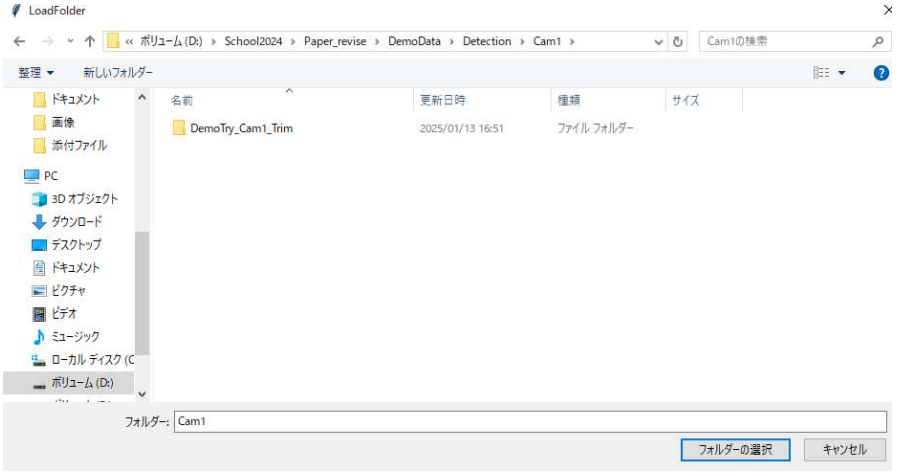


③ これを8台のカメラで行ってください．

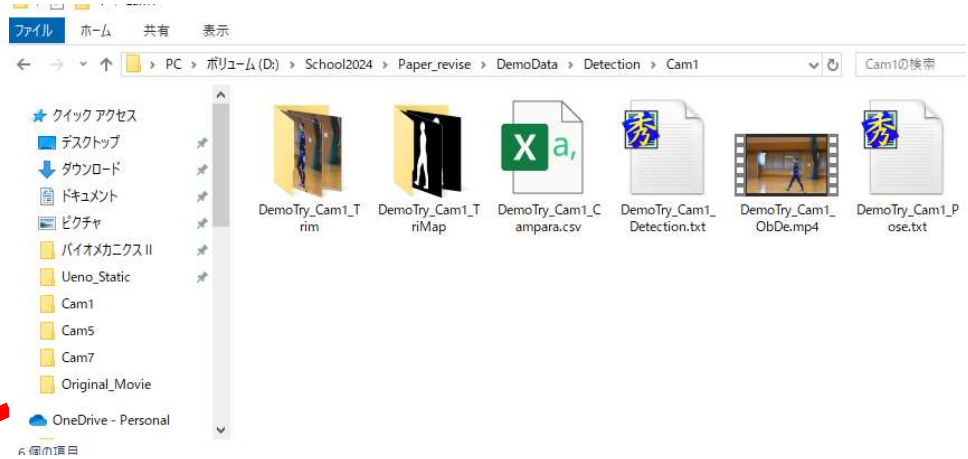
4.Yolov7-maskで暫定的なTrimapを作る

まず，Yolov7-maskの仮想環境を開いて，pose_Human.pyを実行してください．

① トリミングされた画像のフォルダを選ぶ（〇〇Trim）．

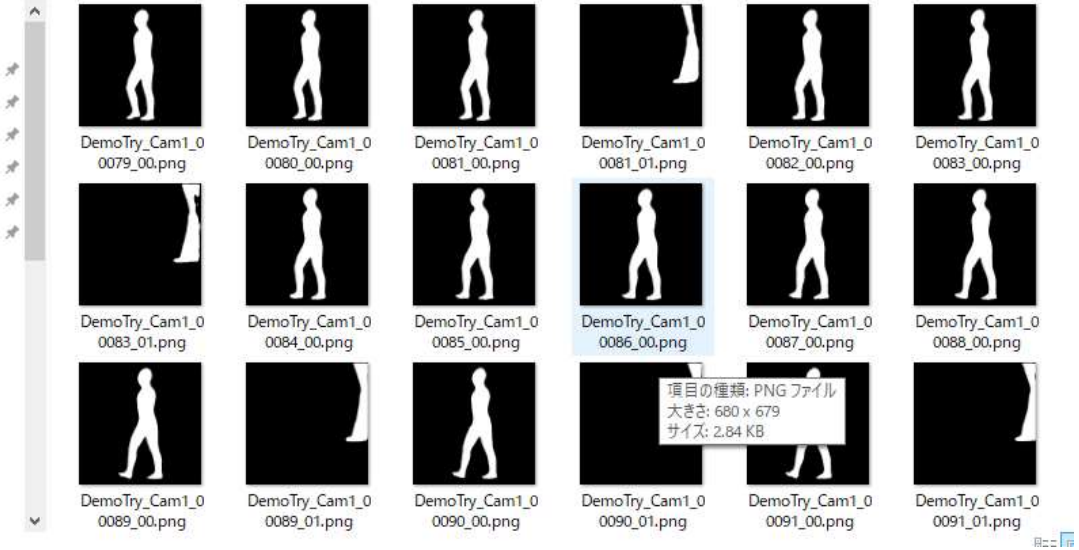


② 〇〇〇_TriMapのフォルダが出力されます．



③ これを8台のカメラで行ってください．

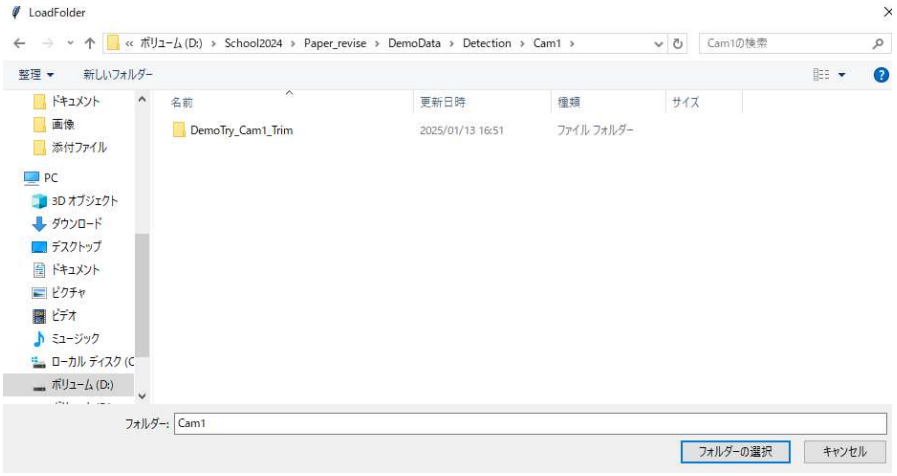
ヒト以外がヒトと認識してしまうことがあります．
手動で削除してください．
（シルエットの大きさなどで判定して，自動で削除すればよいのですが，現状そのアルゴリズムは作っていません．画像の枚数は241枚です）



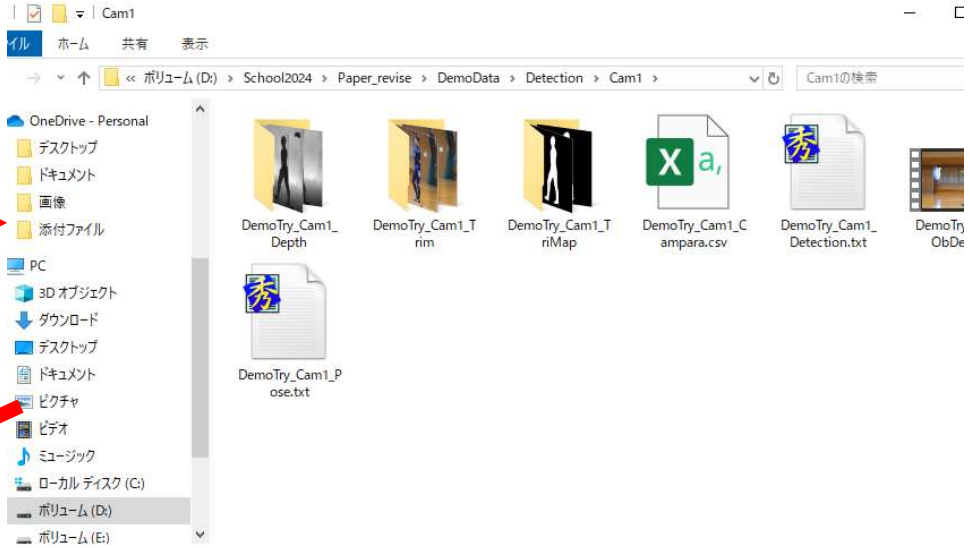
5.Depthで深度を推定する

まず，Yolov7の仮想環境を開いて，DPT-Human.pyを実行してください．

① トリミングされた画像のフォルダを選ぶ(〇〇Trim)．



② 〇〇〇_Depthのフォルダが出力されます．

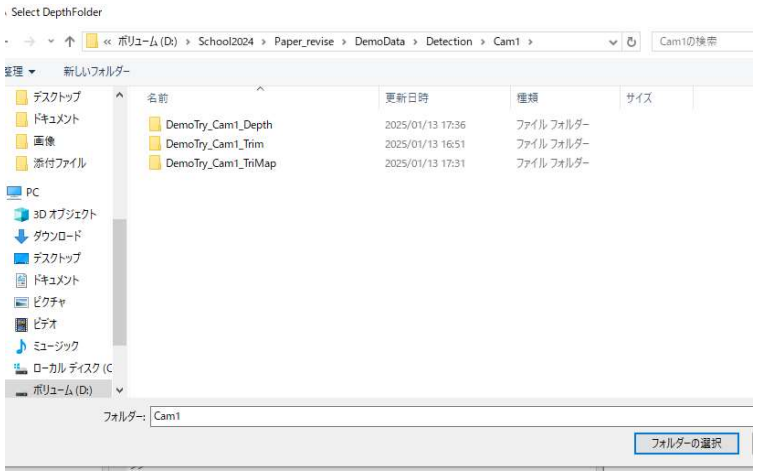


③ これを8台のカメラで行ってください．

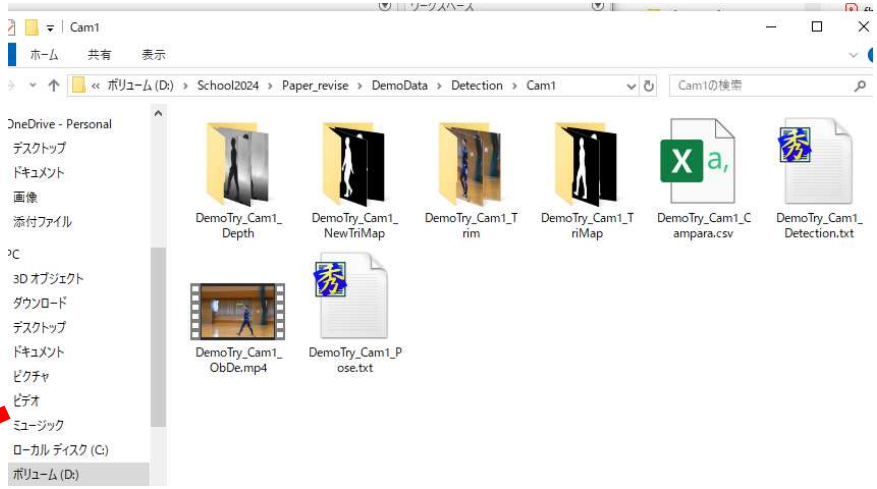
6.暫定的なTrimapと深度情報から Trimapを作成する.

まず，matlabで，Make_NewTriMap.mを実行してください．

① Depthフォルダを選んでください．



② ○○○_NewTriMapのフォルダが出力されます．

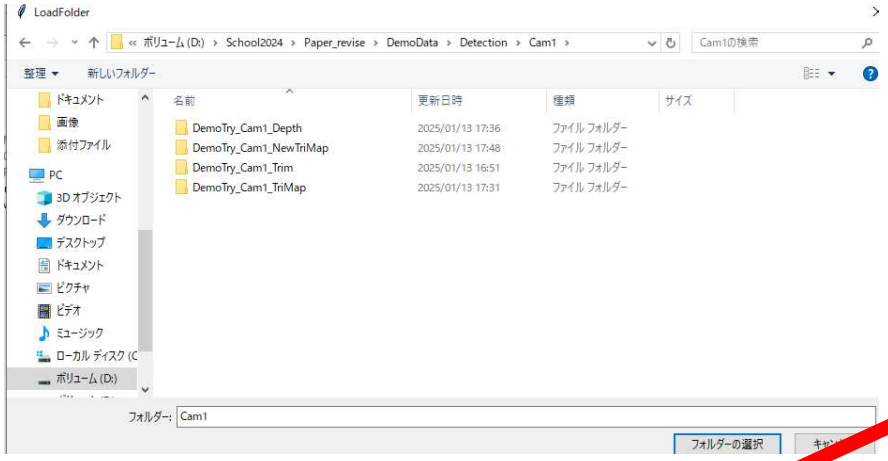


③ これを8台のカメラで行ってください．

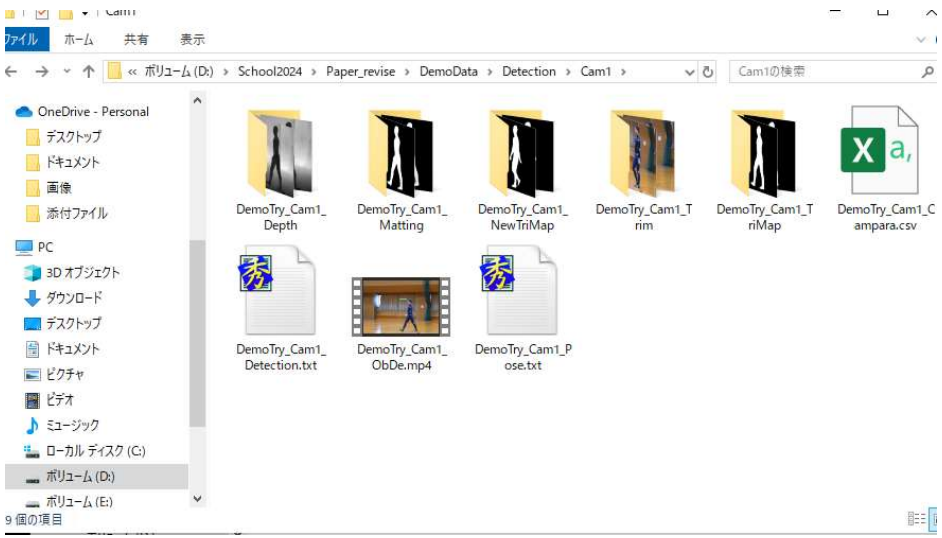
7.Mattingを使ってシルエットを取得する

まず，Yolov7の仮想環境を開いて，Matting_Human.pyを実行してください．

① トリミングされた画像のフォルダを選ぶ（〇〇Trim）．



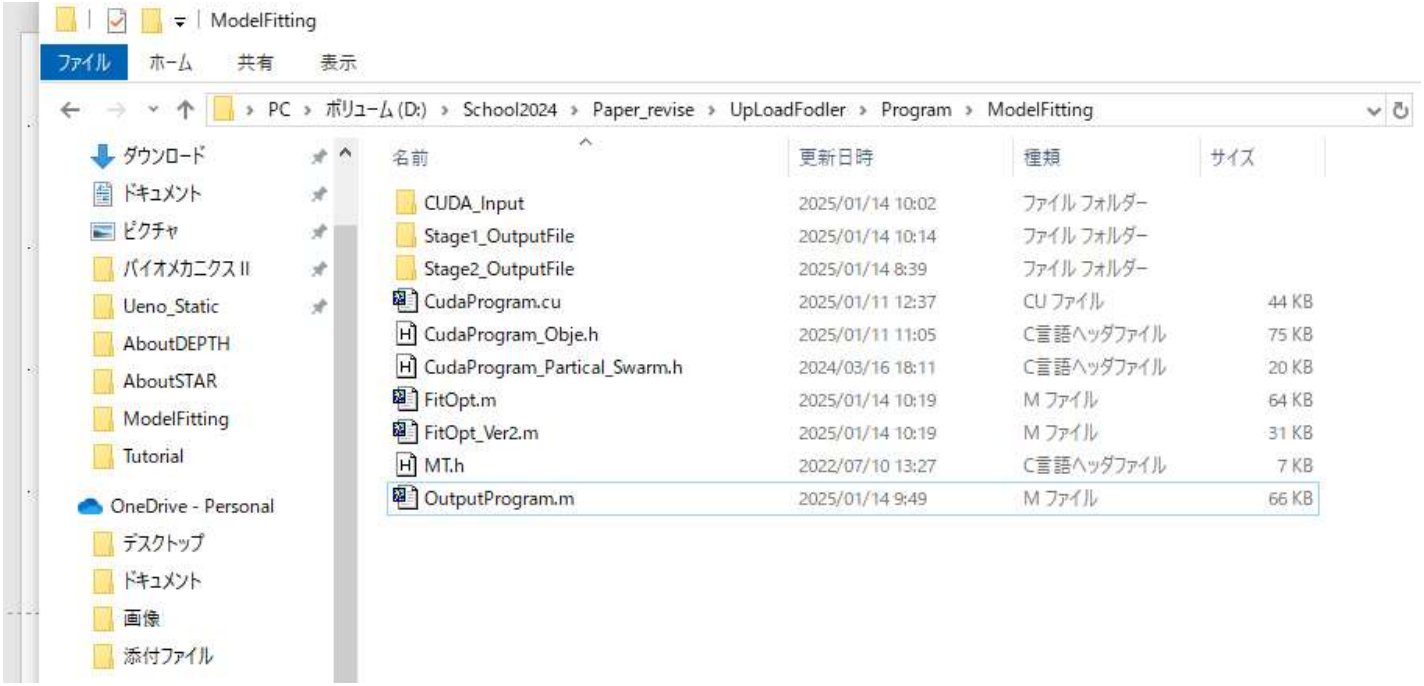
② 〇〇〇_Mattingのフォルダが出力されます．



③ これを8台のカメラで行ってください．

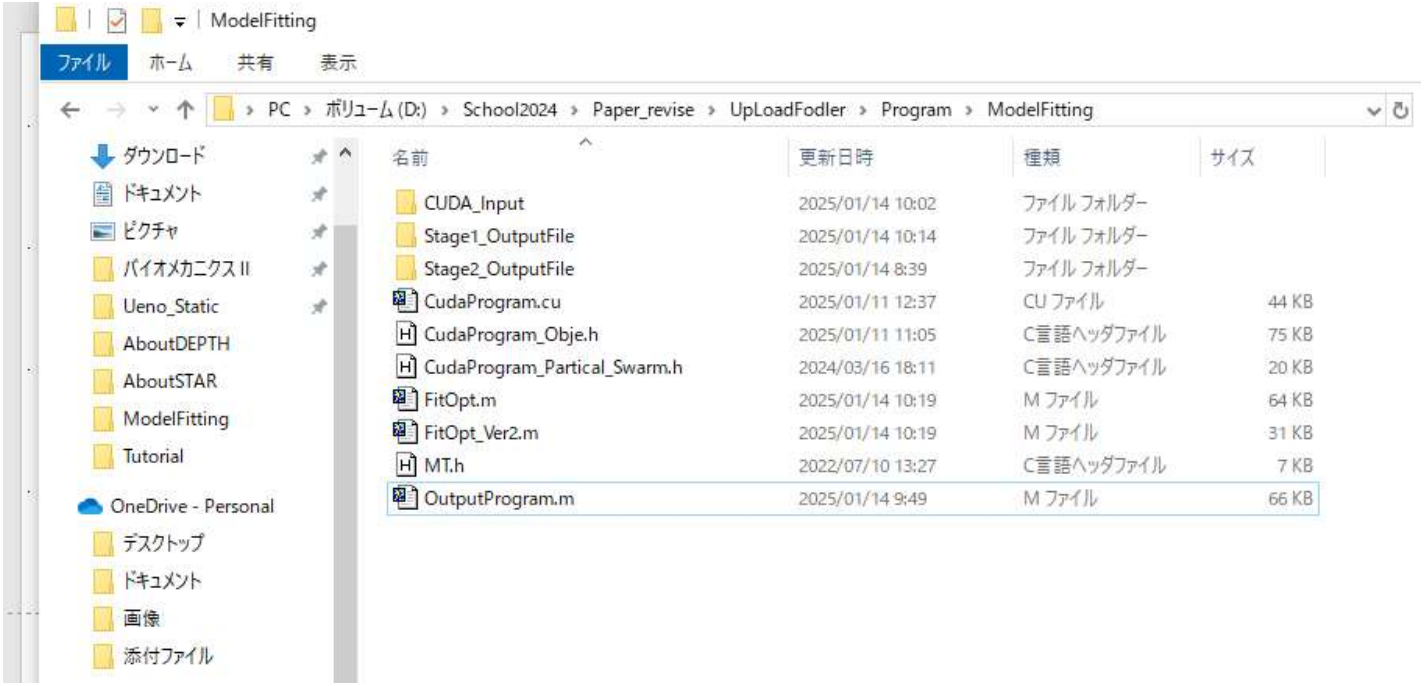
8. 人体モデルとシルエットのマッチング

まず，ModelFittingフォルダ内の，FitOpt.mを実行してください．



すると，Stage1_OutputFileフォルダの中に，モデルの姿勢と体形についてのデータが出力されます．

FitOpt.mが終了したら，FitOut_Ver2.mを実行してください．



すると，Stage2_OutputFileフォルダの中に，モデルの姿勢と体形についてのデータが出力されます．

※FitOpt_Ver2は少し時間がかかるかもしれません．

OutputProgram.mで，結果を確認することができます．

OutputProgram.m

```
function  OutputProgram(  );

close all
clear all


  PathName  = '../ModelFile/';
  load( [ PathName 'J_Regressor.mat'     ] )
  load( [ PathName 'Kintree_Table.mat'   ] )
  load( [ PathName 'Pose_Dirs.mat'       ] )
  load( [ PathName 'Shape_Dirs.mat'      ] )
  load( [ PathName 'V_Temp.mat'          ] )
  load( [ PathName 'Weights.mat'         ] )
  load( [ PathName 'V_Point.mat'         ] )
  load( [ PathName 'NewV_Point.mat'      ] )

  V_Point = V_Point + 1;


  SubName        = 'DemoTry';
  TryPathName    = '../DemoData/';
  StageNumber    = 1;


  %-=-=-=-=-=-=-=-=-=-=-=-=-=
  %--LandMark
  %-=-=-=-=-=-=-=-=-=-=-=-=-=
  LandMark_Ori   = readmatrix( [ TryPathName 'DemoLandMark.csv'  ] );
  LandMark_Ori   =...
    [ LandMark_Ori;
      [ 1349  3078  3080  0 1 0
        3506  3018  6476  1 0 0
      ]
    ];


  %-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
  %--Load Optimization Data
```

Stage1を確認したい時は1,
Stage2を確認したい時は2にしてください

また，OutputProgram.mで，ランドマークの3次元座標データが出力されます．