マーカーレス AR　操作マニュアル

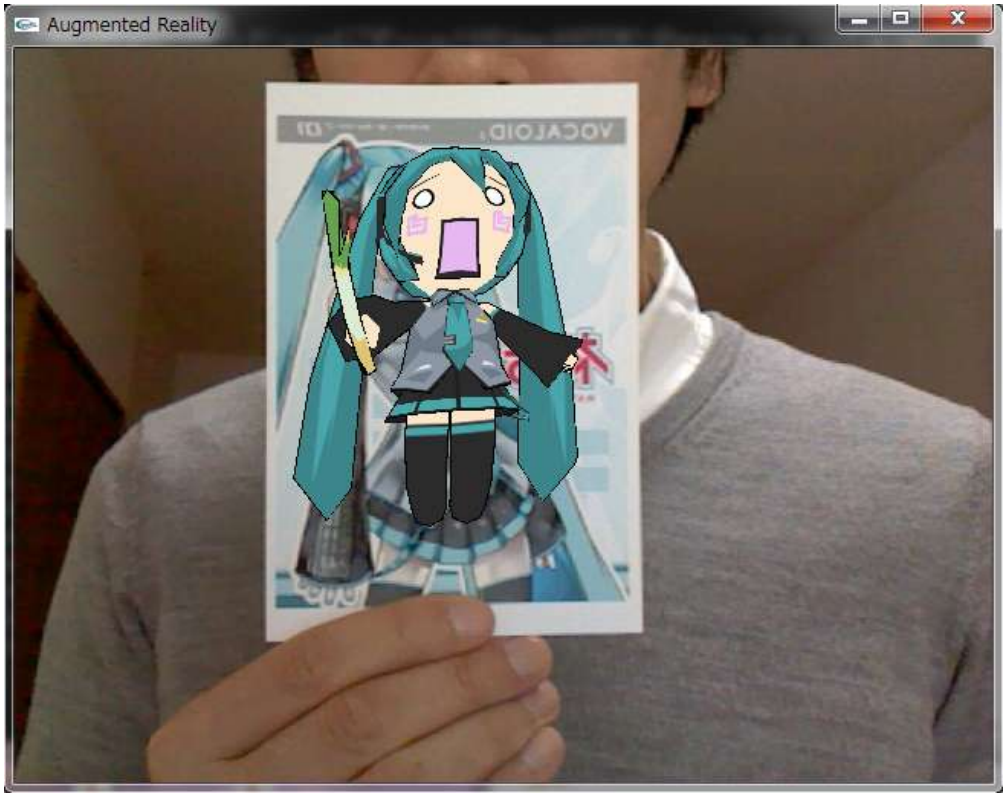
2012/01/07

皆川卓也(z.takmin@gmail.com)

1　概要

本プログラムは USB カメラから取得した画像上の事前登録した画像を認識し、その上に 3D モデルを重畳表示するマーカーレス AR プログラムです。

このマニュアルでは、Windows 版のデモプログラム“ARengine.exe”の使用方法について解説します。コード自体は C++で書かれており、ライブラリとして OpenCV2.3 及び GLUT を使用しているので、各環境でビルドすることで Windows 以外の環境でも動作可能なはずです。



2　Windows 版デモプログラムの使用方法

VC++2010 ランタイム

本デモプログラムを起動するには VC++2010 のランタイムが必要です。
VC++2010 がインストールされていない環境で本プログラムを起動する場合は、以下のサイトからお使いのプロセッサにあったランタイムを探し、ダウンロードとインストールを行う必要があります。

x86 プロセッサ用：
http://www.microsoft.com/downloads/details.aspx?FamilyID=a7b7a05e-6de6-4d3a-a423-37bf0912db84&displayLang=ja

x64 プロセッサ用：
http://www.microsoft.com/downloads/details.aspx?FamilyID=bd512d9e-43c8-4655-81bf-9350143d5867&displayLang=ja

IA64 プロセッサ用：
http://www.microsoft.com/downloads/details.aspx?FamilyID=1a2df53a-d8f4-4bfe-be35-152c5d3d0f82&displayLang=ja

カメラの準備と校正

もしより正確な 3D モデルの重畳表示を行ないたい場合は、使用する USB カメラの内部パラメータを事前に求めておく必要があります。このようなカメラの校正（キャリブレーション）のやり方は後述致します。標準では Logicool の QCAM Pro9000 用のファイルが用意されています。もしお使いのカメラが 640×480 程度の解像度で、あまり厳密な重畳を気にせず、とりあえず使いたい場合はそのままご使用下さい。

マーカーの準備

認識対象となる画像は marker フォルダ内に保存してありますので、こちらを事前に印刷しておいてください。

AR プログラムの起動

“ARstart.bat”をダブルクリックすることで起動します。起動するとウィンドウが立ち上がります。カメラに向けて印刷した画像をかざして下さい。3D モデルが現われると思います。
以下のキーで操作が可能です。

ESC または Q：　プログラムを終了します。
F：　フルスクリーンモード/通常モードの切り替えを行ないます。

3　設定ファイル

“config.xml”を編集することで様々な変更を加えることができます。

設定ファイルのサンプルは以下の通りです。

```
<?xml version="1.0"?>
<opencv_storage>
<VisualWord>
   <visualWord>ardemo/visualWord.bin</visualWord>
   <index>ardemo/vw_index.bin</index>
</VisualWord>
<ObjectDB>ardemo/db.txt</ObjectDB>
<camera_matrix>ardemo/camera_matrix_qcam.txt</camera_matrix>
<focal_length>0.5</focal_length>
<max_query_size>320</max_query_size>
<full_screen_mode>false</full_screen_mode>
<mirror_mode>true</mirror_mode>
<model_info>
  <_>
    <id>1</id>
    <ModelType>MQOSEQ</ModelType>
    <modelfile>ardemo/miku/miku.xml</modelfile>
    <scale>0.002</scale>
          <init_pose>
            <!-- z-y-x euler angle -->
            <yaw>0</yaw>
            <pitch>0</pitch>
            <roll>0</roll>
            <x>0</x>
            <y>-0.2</y>
            <z>0</z>
          </init_pose>
  </_>
  <_>
    <id>2</id>
    <ModelType>METASEQ</ModelType>
    <modelfile>ardemo/ninja.mqo</modelfile>
```

```
    <scale>0.002</scale>
          <init_pose>
            <!-- z-y-x euler angle -->
            <yaw>0</yaw>
            <pitch>0</pitch>
            <roll>0</roll>
            <x>0</x>
            <y>-0.2</y>
            <z>0</z>
          </init_pose>
  </_>
  <_>
    <id>3</id>
    <ModelType>SLIDE</ModelType>
    <modelfile>ardemo/slides/slide1.xml</modelfile>
    <scale>0.004</scale>
  </_>
</model_info>
<WaitingModel>
  <timer>60</timer>
  <ModelType>SLIDE</ModelType>
  <modelfile>ardemo/waiting.xml </modelfile>
  <scale>0.005</scale>
          <init_pose>
            <!-- z-y-x euler angle -->
            <yaw>0</yaw>
            <pitch>0</pitch>
            <roll>0</roll>
            <x>0</x>
            <y>0</y>
            <z>0</z>
          </init_pose>
</WaitingModel>
</opencv_storage>
```

<VisualWord>
特徴点辞書ファイルとそのインデックスファイルへのパスを記述します。XML/YAML 形式で保存した場合は<visualWord>タグに辞書ファイルへのパスを、バイナリ形式で保存した場合は<visualWord>と<index>タグにそれぞれ辞書ファイルとインデックスファイルのパスを記述します。辞書ファイルの作成方法については後述します。

<ObjectDB>
登録マーカー情報データベースファイルへのパスを記述します。DB ファイルの作成方法（マーカー画像登録方法）については後述します。

<camera_matrix>
使用するカメラの内部パラメータを記述したファイルへのパスを記述します。カメラパラメータの取得方法（カメラ校正）については後述します。正しく重畳表示を行ないたい場合は、カメラごとに校正を行う必要があります。

<focal_length>
カメラの仮想焦点距離(mm)を記述します。3D モデルがカメラに近づけると切れてしまう場合は、この値を小さく設定してください。

<max_query_size>
画像認識を行う際の最大入力画像サイズを記述します。カメラからの画像はこの値よりも一辺の長さが小さくなるように縮小されて、認識処理に回されます。この値が小さいほど高速になりますが、一方でマーカー画像をカメラにより近づける必要が出てきます。

<full_screen_mode>
起動時にフルスクリーンで画面を表示するか否かを設定します。

<mirror_mode>
この値が true の場合、カメラからの画像を左右反転させ、鏡に映っているような効果を出します。ユーザがカメラとモニタの正面にいるような使い方を想定している場合は、これを true に設定して下さい。

<model_info>
このタグの中で、どのマーカーにどの 3D モデルを割り当てるのかを設定します。

<id>

登録マーカー画像 ID（後述）

<ModelType>
3D モデル/表現のタイプを指定します。3D モデルの型として現状以下の３つを用意しています。

- METASEQ: mqo モデルファイル
- MQOSEQ: 連番 mqo ファイル（アニメーション用）
- SLIDE: スライドショー

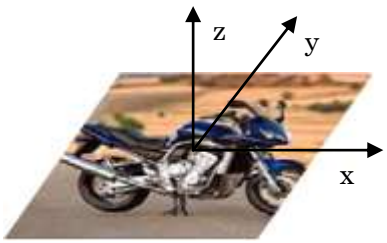
<modelfile>
3D モデル設定ファイル。ModelType によって指定するファイルが違います。

- METASEQ の場合、読み込みたい MQO ファイルパス名を記述します。
- MQOSEQ の場合、連番 mqo ファイル群が格納されたディレクトリ上に設定ファイルを設け、その設定ファイルへのパスを指定します。この設定ファイルは連番 mqo ファイルのヘッダ部と SQO ファイルの数を記述しています。
- SLIDE の場合、スライドショー設定ファイルを指定します。この設定ファイルには読み込みたい画像とその順番が記述されています。

<scale>
読み込んだモデルファイルを表示するスケールを指定します。3D モデルの表示上の大きさはこのパラメータで調整してください。

<init_pose>
3D モデルファイルのマーカー上の初期位置を設定します。初期位置はマーカー座標系上の回転成分(ヨー・ロール・ピッチ)と並進成分(X, Y, Z)からなります。マーカー座標系は下に示したとおりです。

登録画像



マーカー座標系

ヨー・ピッチ・ロールは別名を Z-Y-X オイラー角と言い、それぞれ Z, Y, X 軸回りの回転を表します。

<WaitingModel>
待ち受け画面を設定します。この待ち受け画面はマーカー画像の認識がある一定期間起こらなかった時に、3D モデルを表示するための機能です。これは例えば、認識対象が画面に提示されない時に、使い方の説明画面を出すなどの用途を想定しています。待ち受け画面の指定方法は<model_info>とほとんど同じです。ただし、ここでは複数のモデルの指定や、登録マーカー画像 ID とのモデルの紐付けなどは行えません。
また、<timer>タグで待ち受け画面が表示されるまでのフレーム数を指定することができます。

4　カメラの校正

カメラの校正は、より正確な 3D モデルの重畳表示を行うために、カメラ画像の中心位置やピクセル間の距離を求めます。
カメラ校正の手順は以下の通りです。

1. 校正用ボードを準備します。
    - “tools/pattern.pdf”を印刷します。
    - 印刷した紙を平らな板などの上に貼り付けます。
    - 一マスの長さを定規などで測ります。
2. PC にカメラを取り付けます。
3. ARengine.exe をダブルクリックで起動します。
4. カメラから校正用画像を取得します。
    - “get_camera_images”と入力してリターンキーを押します。
    - “number of images: “ので、取得したい画像枚数を適当に入力します。数十枚程度あると十分な精度が得られます（上限 50 枚）。(ex. 20)
    - “frame interval: “で何フレームおきにカメラの画像を保存するかを入力します。(ex. 30)
    - “save directory: “で画像を保存するフォルダのパスを入力します。
    - “file header: “で指定した文字列が画像ファイル名の頭につけられます。
    - カメラが起動してウィンドウが表示されるので、カメラに校正用ボード全体が映るようにして下さい。位置は固定しないでください。
    - “s”ボタンを押すと、画像の取得を開始します。指定枚数の撮影が終わると自動的にウィンドウが消え、コマンドプロンプトに戻ります。
5. 取得した画像群を用いてカメラ校正を行ないます。
    - “camera_calibration”と入力します。
    - “checkerboard image list: ”と聞かれるので、校正用画像が保存してあるフォルダの中にある”imglist.txt”というファイルへのパス＋ファイル名を入力します。
    - “checkerboard row number: “でチェッカーボードの格子点の行の数を入力します。例えば pattern.pdf を縦向きで撮影した場合 10、横向きで撮影した場合 7 になります。
    - “checkerboard col number: “でチェッカーボードの格子点の列の数を入力します。例えば pattern.pdf を縦向きで撮影した場合 7、横向きで撮影した場合 10 になります。
    - “checker size(mm): ”でチェッカーボードの一マスの長さを入力します。
    - カメラ内部パラメータの計算が自動的に始まります。
    - 計算が終了すると”camera parameter file name: ”と聞かれるので、カメラパラメ

ータを保存するパス＋ファイル名を入力してください。尚、カメラパラメータはテキストファイル、または xml ファイルで保存されます。どちらの形式で保存するかは拡張子で判断されます。

6. “exit”と入力して ARengine を終了させます。
7. 得られたカメラパラメータファイルを config.xml の<camera_matrix>に設定します。

5. マーカー画像の登録方法

マーカー画像は AR 機能で 3D モデルを重畳させるために使用できるのはもちろんのこと、画像認識エンジンとして、入力画像がデータベースに登録したマーカー画像のうちのどれとマッチングするかの情報を返してくれます。

1. 認識対象となる登録画像 A 群と非認識対象の画像 B 群をそれぞれ用意します。
2. ファイル A と B へのパスをリスト化したテキストファイルを作成します。全体の画像枚数が少ない場合精度が落ちることがあるため、全体が QVGA15 枚以上になるようにして下さい。また誤認識を減らしたい場合は、A の枚数に対する B の枚数の比率を調整して下さい。経験的に A の 2 倍程度 B があると良いです。これをリストファイル C とします。

   ex.
   img/A1.jpg
   img/A2.jpg
   img/A3.jpg
   img/B1.jpg
   img/B2.jpg
   img/B3.jpg
   .
   .
   .
3. ファイル A のリストのみを記述したテキストファイルも用意します。これをリストファイル D とします。ファイル名の頭に ID をカンマ区切り整数で指定できます。この ID がマーカーID となり、AR 用設定ファイル config.xml の<model_info>タグ内の<id>で指定します。

   1,img/A1.jpg
   2,img/A2.jpg
   3,img/A3.jpg

4. “ARengine.exe”を起動します。
5. "create_vw"と入力することで特徴辞書ファイルとそのインデックスファイルを作成します。
   - "img file list:"には C のファイル名を記載します。
   - 特徴辞書ファイルとインデックスファイルの作成を開始します。
6. コマンドプロンプトが現れたら"save_vw"と入力し、作成した特徴辞書ファイルを保存します。途中"feature dictionary file:"と聞かれるので、保存するファイル名を指定しま

す。Feature points file については、ファイル名は拡張子が”.xml”の時 XML ファイル形式、それ以外では YAML 形式で保存されます。拡張子を”.txt.gz”や”.xml.gz”とすることで圧縮形式での保存も可能です。また”save_vw”の代わりに”save_vw_bin”を使用すると、より高速に読み書きするためのバイナリ形式で保存することができます。ただし、”save_vw_bin”を使用すると特徴辞書とそのインデックスを別ファイルとして保存する必要があります。この場合、"feature points file:"の他に"index file:"というプロンプトが現れるので、保存するファイル名を指定します。Index file は XML/YAML 形式になります。ここで指定したファイル名が”config.xml”の<VisualWord>タグ内で指定されます。

7. コマンドプロンプトで"registf"と入力し、"regist list file:"で D のファイル名を記載します。これにより認識したい対象がデータベースへ登録されます。その時各ファイルにひもづけられた ID が標準出力に出力されます。
8. コマンドプロンプトで"save_objectDB"と入力し、データベースの保存先の名前を指定して保存します。ここで保存したファイル名を”config.xml”の<ObjectDB>で指定します。
9. “exit”と入力して ARengine を停止します。

6. 画像の認識試験方法

ここでは、AR の重畳表示機能を使わずに、単体で画像認識機能をチェックする方法を説明します。つまり USB カメラは使用せず、クエリーとなる画像ファイルを用意することで、そのクエリー画像がどの ID の登録画像とマッチングしたのかを調べることができます。
認識試験の方法として、クエリー画像を一枚一枚与えて振る舞いを確認する方法と、ファイルに試験画像一覧を記述し、バッチで試験を行ない、その結果をファイルへ出力する方法の２通りがあります。

特徴辞書及び DB ファイルのロード

1. “ARengine.exe”をダブルクリックで起動します。
2. "load_vw"と入力し、特徴辞書ファイルを読み込みます。もしバイナリ形式で特徴辞書ファイルを読み込みたいときは”load_vw_bin”と入力します。“feature dictionary file: “や”index file: “と聞かれるので、前項の 6 で指定した名前を入力します。
3. "load_objectDB"でデータベースファイルを読み込みます。”load file name: “と聞かれるので前項の 7 で指定した DB ファイル名を指定します。特徴辞書ファイルや DB ファイルは、例えば前項で作成した直後に ARengine を停止させず作業を続けている状態であれば、わざわざロードする必要はありません。

単数画像試験

特徴辞書ファイルと DB ファイルをロードした状態で、単体で試験を行う方法です。

1. クエリーとなる画像を用意します。
2. “query”と入力します。”query image file: “画像名を聞かれるので、クエリー画像のパス＋ファイル名を入力します。
3. データベース内の登録画像とマッチングが取れた場合、ウィンドウが開いてオブジェクトのある位置が描画されます。またプロンプト上にマッチした登録画像 ID が表示されます。

複数画像バッチ試験

特徴辞書ファイルと DB ファイルをロードした状態で､バッチ処理で試験を行う方法です。

1. 試験したいクエリー画像をリストにしたテキストファイル E を用意します。
2. コマンドプロンプトで"queryf"と入力し､"img file list:"で E のファイル名を入力します。
3. "result file: "で試験結果の出力先ファイル名を指定します。
4. 認識結果が指定したファイルにテキスト(csv)形式で出力されます。対象となる画像が DB 内に見つかったときはその ID を、見つからなかったときは-1 がファイルに出力されます。

7. ARengine のコマンド集

create_vw	特徴辞書とそのインデックを作成します。
save_vw	特徴辞書とインデックスをファイルに保存します。
load_vw	特徴辞書とインデックスをファイルから読み込みます。
save_vw_bin	特徴辞書とインデックスをバイナリモードでファイルに保存します。
load_vw_bin	特徴辞書とインデックスをバイナリモードで保存したファイルから読み込みます。
registf	バッチ処理で複数の画像をデータベースへ追加登録します。
regist	１枚の画像をデータベースへ追加登録します。
remove	指定した ID をデータベースから削除します。
clear	データベースの情報をクリアします。
save_objectDB	現在のデータベースの情報をファイルへ保存します。
load_objectDB	データベースファイルを読み込みます。
query	１枚の画像の認識試験を行ないます。
queryf	バッチ処理で複数の画像の認識試験を行ないます。
get_camera_images	カメラから連続して画像を取得します。
camera_calibration	カメラの内部パラメータを計算します。
ar_start	AR 機能を起動します。