AN INTERACTIVE PEDESTRIAN RE-IDENTIFICATION TOOL WITH SEMANTIC BASED RE-IDENTIFICATION

A Thesis Presented to

the Faculty of the Department of Computer Science

University of Houston

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

By

Can Cao

December 2016

AN INTERACTIVE PEDESTRIAN RE-IDENTIFICATION TOOL WITH SEMANTIC BASED RE-IDENTIFICATION

Can Cao

APPROVED:

Shishir Shah, Ph.D., Chairman
Dept. of Computer Science

Edgar Gabriel, Ph.D.
Dept. of Computer Science

Fatima Merchant, Ph.D.
Dept. of Engineering Technology

Dean, College of Natural Sciences and Mathematics

Acknowledgements

Foremost, I would like to express my heartfelt thanks to my academic advisor Prof. Shah, for the continuous support of my study and research, for his expertise, immense knowledge, patience, and valuable guidance. I appreciate all his contributions of time to make my graduate experience meaningful and substantial.

My sincere thanks also go to my thesis committee: Prof. Gabriel and Prof. Merchant, for their encouragements, guidance, and support of my research ideas. Their brilliant comments, suggestions, and questions made my defense.

I would also like to thank my dear friends Shenghua, Gray, Eason, and Tianhe, for their patient encouragements and help in life and study. My time at the University of Houston was made enjoyable in large part due to many friends and groups that became a part of my life. My time in the graduate study was also enriched by my climbing buddies Kim, Victor, Yufeng, and Jihae, and our memorable trips into the comps and mountains.

Last but not the least, I would like to thank my parents Lijun and Huiming, for their constant love and support, for being the greatest parents in the world. I would also like to thank my grandparents, for their remote love and concern from China. Thank you all for being ever so understanding and supportive.

AN INTERACTIVE PEDESTRIAN RE-IDENTIFICATION TOOL WITH SEMANTIC BASED RE-IDENTIFICATION

An Abstract of a Thesis Presented to
the Faculty of the Department of Computer Science
University of Houston

In Partial Fulfillment
of the Requirements for the Degree
Master of Science

By
Can Cao
December 2016

Abstract

Person re-identification is an essential task of recognizing and matching people from non-overlapping cameras. A typical application of person re-identification is identifying a particular person in a gallery of pedestrian images from a camera with one or more given probe images of this person from another camera. This is a challenging and practical task that provides solutions for video-surveillance. In this work, we present a person re-identification software which is called Interactive Pedestrian Re-identification GUI (IPRG), and a semantic-based labelling tool named Reid It (Reidit). According to the growing need for surveillance applications, we develop IPRG to address the person searching and matching problem with the dataset from on-campus security camera videos. From these video frames, we can get semantic information of the candidate such as height, ethnicity, cloth color, etc. By customizing these semantic features in IPRG, we can identify a candidate in the video database rapidly. We also propose a light-labelling tool, Reidit, for labelling pedestrian images with semantic features as the pre-processing for pedestrian recognition. We present an experiment on IPRG with Viewpoint Invariant Pedestrian Recognition (VIPeR) dataset which contains 632 identities. Our experiment shows that our software is more efficient and accurate compared with traditional manual solutions. Moreover, IPRG can handle the situation of missing query person in the database, and it will return the top ten possible individuals. Our software is compatible with different platforms and user-friendly with customizable databases and semantic features.

Contents

1 Introduction **1**

2 Related Work **4**

2.1 Person Re-identification Approaches 4

2.2 Appearance-based Modeling 5

2.3 Metric Learning 6

3 Theory **8**

3.1 Labelling 9

3.2 Semantic Color Names 17

3.3 Semantic Bio Names 18

3.4 Metric Learning 19

4 Implementation **21**

4.1 Framework 21

4.2 GUI Functional Programming 22

4.2.1 Select Image Folder and Display 23

4.2.2 Feature Selection Group 29

4.2.3 Result Display 32

5 Experiments **37**

5.1 Semantic Color Name Detection 37
5.2 Basic Functions 40
5.3 Testing on the Re-identification Module 40

6 Conclusions **50**

Bibliography **52**

List of Figures

3.1 Reidit Framework. It contains a plot axes and three feature groups. . 10

3.2 Reidit Labelling Tool. This is a complete window of the Reidit GUI. . 11

3.3 Axes Inspector. The configuration of the Axes Inspector is the default. 12

3.4 Push Button component. We change the String of the Push Button to "Save & Next". 13

3.5 Gender Radio Button. 14

3.6 Clothing Selection Group. We use the default setting. 16

4.1 GUI Components. The options are Radio Button, Axes, Listbox, and Static Text. 24

4.2 Push Button and Listbox. This is a plain configuration. 25

4.3 Push Button Inspector. We use the default setting. 26

4.4 Gender Group. Default configuration. 30

4.5 Feature Selection Group. The basic function is on the left part: Selection image folder and display images. The primary re-identification function is the middle column and rightmost column. Feature selection provides the user with several options, result axes and result text window display the matching result. 31

4.6 Result Axes. There are ten axes plots to display the top ten candidates. 33

4.7 Static Text Window. Shows the information of the top ten candidates. 34

4.8 Static Text Inspector. Default settings. 35

4.9 IPRG Layout. The logical layout of all the components in IPRG GUI. 36

5.1 Automatic Semantic Color Name Detection on Probe Images. Rank 1 score is 68% recognition accuracy. 38

5.2 Automatic Semantic Color Name Detection on Both Camera Video Frames. IPRG performs similarly on Camera A and Camera B images. Both of the rank 1 scores are around 68% accuracy. 38

5.3 Initial Window. Before we select the image folder. 41

5.4 Select An Image Folder. We choose an image folder in the operating system. 42

5.5 Basic Info of an Image. Click on an image, we can check the basic name info and color channels of the current image. 43

5.6 Testing Example 1. The matching results of the current candidates are listed in the result groups. The correct match is the candidate on the top row and third column from left to right. 44

5.7 Testing Example 2. The true match is the candidate on the top left. . 45

5.8 Testing Example 3. The true match is the last candidate on the first row. 46

5.9 Result of Example 3. The Static Text window displays the related top ten results. 47

5.10 Save the Result to a Mat file. It contains image names of the top ten candidates. 47

5.11 Semantic Attribute Based Matching. The rank 1 accuracy of automatic semantic detection data is 84%. With well-labelled data, the matching results on rank 1 is 95%. 48

List of Tables

5.1 Automatic Semantic Color Name Detection Results 39

5.2 Attributes Weights 39

5.3 Semantic Attribute Based Pedestrian Re-identification Result. From rank 1 to rank 10, the rank scores of Camera and Camera B are similar, but IPRG performs better on Camera A. Rank 1 accuracy of Camera A and Camera B are 95.12% and 94.74%, respectively. 49

Chapter 1

Introduction

Public safety has become more and more important in recent years. Nonetheless, recognizing or identifying a person observed across multiple cameras in a surveillance system is a challenging problem [6]. Current automatic solutions to this problem do not provide sufficient accuracy to overcome the need for manual and labor-intensive interventions from a human operator. Concomitantly, there is a need to improve approaches for person identification across cameras, also know as person re-identification.

Person re-identification is a challenging problem because of visual vagueness and spatial changes [2]. These difficulties are a result of low-resolution cameras, different conditions, or noise. This problem has received much attention from the computer vision research community due to the varied application of person re-identification. Our work, presented in this thesis, is motivated by applications in forensic analysis where the need is to match an observed person in a camera with persons observed

from other distinct cameras. Currently, there are many vision-based open-source tools and platforms for person re-identification. MATLAB is a popular platform to test and visualize the result of person re-identification, and we use it to facilitate a user-driven process for providing semantic labels to improve matching to do visualized, matched observations. Moreover, we combine and improve several person re-identification algorithms to achieve better matching rates.

In this thesis, we present the development of the Interactive Pedestrian Re-identification GUI (IPRG). In the preprocessing stage, we develop a labelling software that provides the user with options on adding specific semantic labels that can be used to augment the visual observation. Each image observation can be enhanced through several semantic labels to specify gender, ethnicity, and the color of clothing of the person. Augmenting each visual observation with semantic labels, it is reasonable to use semantic features to solve the person re-identification problem.

We use the Viewpoint Invariant Pedestrian Recognition (VIPeR) dataset [7] to evaluate the performance of re-identification using semantic labels. VIPeR contains viewpoint and illumination variation. Specifically, it has 632 pedestrian image pairs taken from two different cameras under different views and light conditions. The testing result showed our software worked efficiently and accurately on low-resolution images and chosen semantic labels.

The content of this work is structured as follows: Chapter 2 provides a discussion of related work. Chapter 3 describes the methods and algorithms employed. Chapter 4 introduces the GUI implementation with the framework of MATLAB GUI. Chapter 5 displays the GUI and the recognition results on the VIPeR dataset. Chapter 6 gives

a summary of our work and discussion of future directions.

Chapter 2

Related Work

2.1 Person Re-identification Approaches

There are two main types of approaches that have been developed to address the problem of person re-identification, appearance-based modeling and metric learning [22]. Most of the existing person re-identification modeling approaches try to build a color and texture feature representation that is stable and robust which describe a person, especially under different lighting conditions or locations [6]. Gray and Tao [7] proposed the AdaBoost feature selection to obtain good color and texture features from many features. Farenzena *et al.* [1] proposed Symmetry-Driven Accumulation of Local Features (SDALF) that handles different viewpoints. A Pictorial Structures method is proposed to learn the appearance of a person [4], which enhances the characteristics of different body parts.

Once the representation has been obtained, it is typically encoded in a high-dimensional feature vector, which in turn is used for facilitate matching or classification. High-dimensional features often lead to lower-matching accuracies. To overcome this challenge, metric-learning techniques such as Large Margin Nearest Neighbor (LMNN) [20] and Logistic Discriminant Metric Learning (LDML) [8] are used to extract meaningful information. Several efficient metric learning methods have been applied to measure the similarity of data pairs such as Mahalanobis distance and Cross-view Quadratic Discriminant Analysis (XQDA) [11]. We can compare features extracted from observations from different cameras under different representations using a learning metric.

2.2 Appearance-based Modeling

Person re-identification methods are dependent on visual information from an appearance-based recognition system. The methods in this category explore features that are stable to variations in poses and illumination. Moving pedestrians have changing poses, different camera settings result in different video quality, and extracted features are based on the color or the texture information of the clothing. However, under low resolution and changing illumination, the same color may appear to be different across observations. To address this problem, developed methods such as Local Binary Pattern (LBP) [9], and filters such as Gabor [21] are used to enhance the texture feature. Usually, color and texture features are combined in the visual representation for re-identification. To improve the accuracy and enhance the

distinction of each person's feature representation, several techniques such as PCA (Principal component analysis) [16] and AdaBoost [7] have been applied in the pre-processing stage. Moreover, different weights are given to the features according to their importance [12].

To extract color features, we can subdivide the bounding box of a person into horizontal or vertical stripes from which RGB and HSV histograms can be generated. In addition to color information, many other options such as edges, interest points, and image patches are applied as representative features [14]. For texture features, we use oriented gradient histograms as the component of appearance features. We first convert the image to grayscale and adjust the contrast of the image with correction techniques. We then calculate the gradient of each pixel in the image to get the silhouette information. Next, we divide the image into blocks and concatenate them as texture features. Finally, we concatenate color and texture features as the descriptor of the pedestrian.

2.3 Metric Learning

With the extracted features from the previous appearance modeling stage, we can apply dimensionality reduction followed by metric learning on the reduced subspace of the features. A widely used metric learning approach is Mahalanobis distance learning [5], which measures the similarity of data pairs through exploiting the structure of the data [18]. Given several data points, the goal is to learn a projection matrix that minimizes the distance between same identity data pairs and maximizes

the distance between different pedestrian data pairs. Mahalanobis distance learning allows a more compact representation of the data since it generates a linear projection of the data to a space of lower dimension than the original feature.

Large Margin Nearest Neighbor (LMNN) is another principal approach of metric learning techniques, introduced by Weinberger *et al.* [20]. The authors developed a strategy where the k nearest neighbors of training instances are pulled closer together and the other classes (the imposters) are kept away from each other. The Euclidean distance is used to find the target neighbors [3]. LMNN performs well in most cases, although it is prone to overfitting because of the regularization in high-dimensional space.

The original features extracted from an image form a high dimensional feature vector. Reducing this dimensionality improves training and classification [13]. Hence, we apply PCA dimension reduction techniques. However, these methods may eliminate important features without considering distance metric learning. [11] A Bayesian face and KISSME was proposed to learn a subspace with cross-view data. This routine also learns a similarity-measurement-distance function. An intraclass and interclass covariance matrix are used to address the internal structure of the data. The distance function can be obtained by the generalized eigenvalue decomposition. XQDA can be seen as an extension of discriminant subspace learning.

Chapter 3

Theory

In this chapter, we start by introducing the MATLAB GUI and present the dataset preprocessing techniques. Then we present the implementation of the labelling tool, Reidit, and feature extraction methods, which are essential for metric learning and matching. Chapter 3.1 introduces the implementation of the labelling tool. Chapter 3.2 presents the standard method to generate semantic color feature representations of pedestrians. Chapter 3.3 introduces the gender and ethnicity feature-extraction process and finally, chapter 3.4 presents the algorithm and techniques we propose for recognition and matching.

3.1 Labelling

This section introduces MATLAB GUI and several examples of Reidit. Reidit provides gender, ethnicity, clothing color label options for users to describe pedestrians. The tool is customizable and additional label categories or label options can be easily added or removed. The labelled result can be saved as .mat file, which is convenient for research and practical use.

In this work, we have opted to use the MATLAB GUI framework which makes visualization straightforward and fast for end-users. Since we are building an interactive tool for users to choose among several options, it is not practical to use the command line. Figure 3.1 shows the Reidit GUI framework and Figure 3.2 is the final window of Reidit. To implement the labelling tool, we start with setting boxes and buttons to get the framework.

We can type "guide" in MATLAB and use the default blank GUI to start implementing Reidit GUI. First, we want to display an image so that we can check the semantic features of a candidate and label him/her. In the MATLAB GUI, we use the Axes component to display graphics and plots. We also need a Push Button component to tell the GUI that the labelling of the current candidate is finished. An Axes and one Push Button is chosen by dragging them from the left panel to the blank GUI window. We use their default name "axes1" for the Axes and "pushbutton1" for the only Pushbutton component. For the Axes component, we can change its Tag property to "$image_display$". Similarly, we can double click on Push Button component and change the String property to "Save & Next" and change the

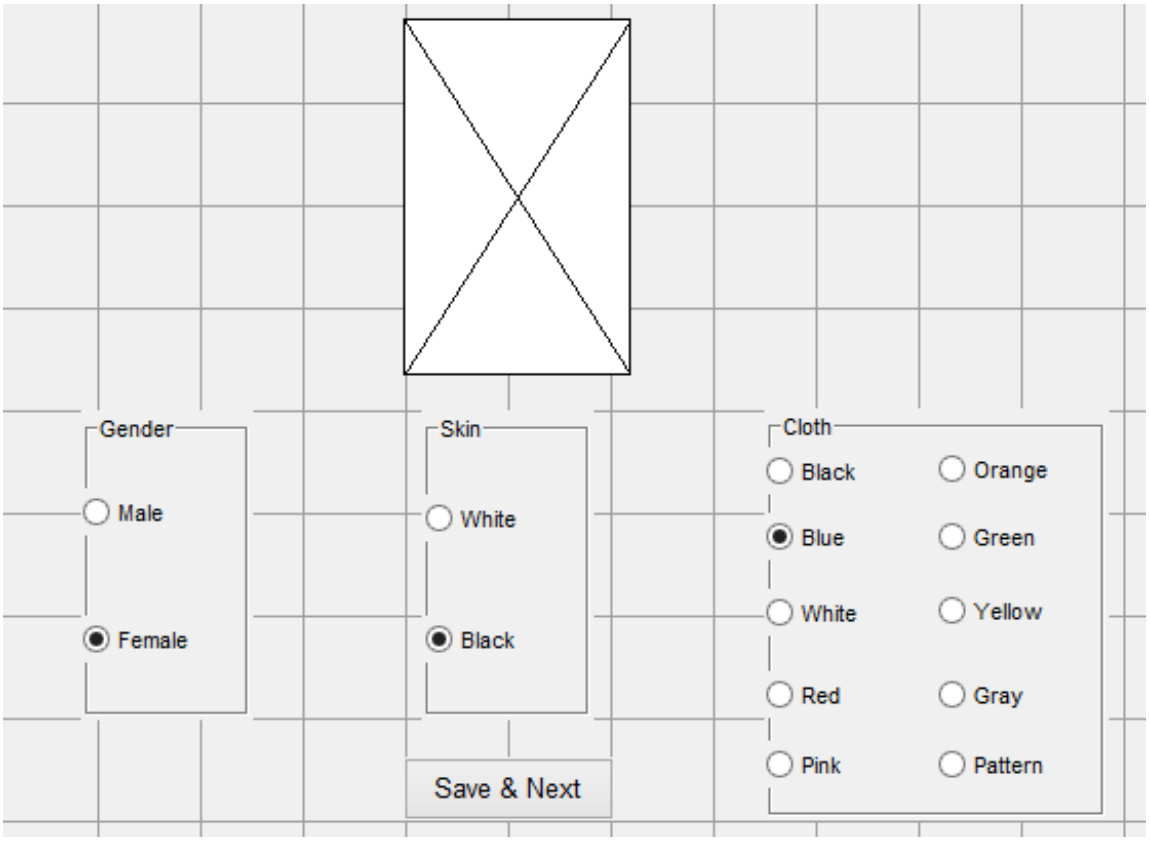

Figure 3.1: Reidit Framework. It contains a plot axes and three feature groups.

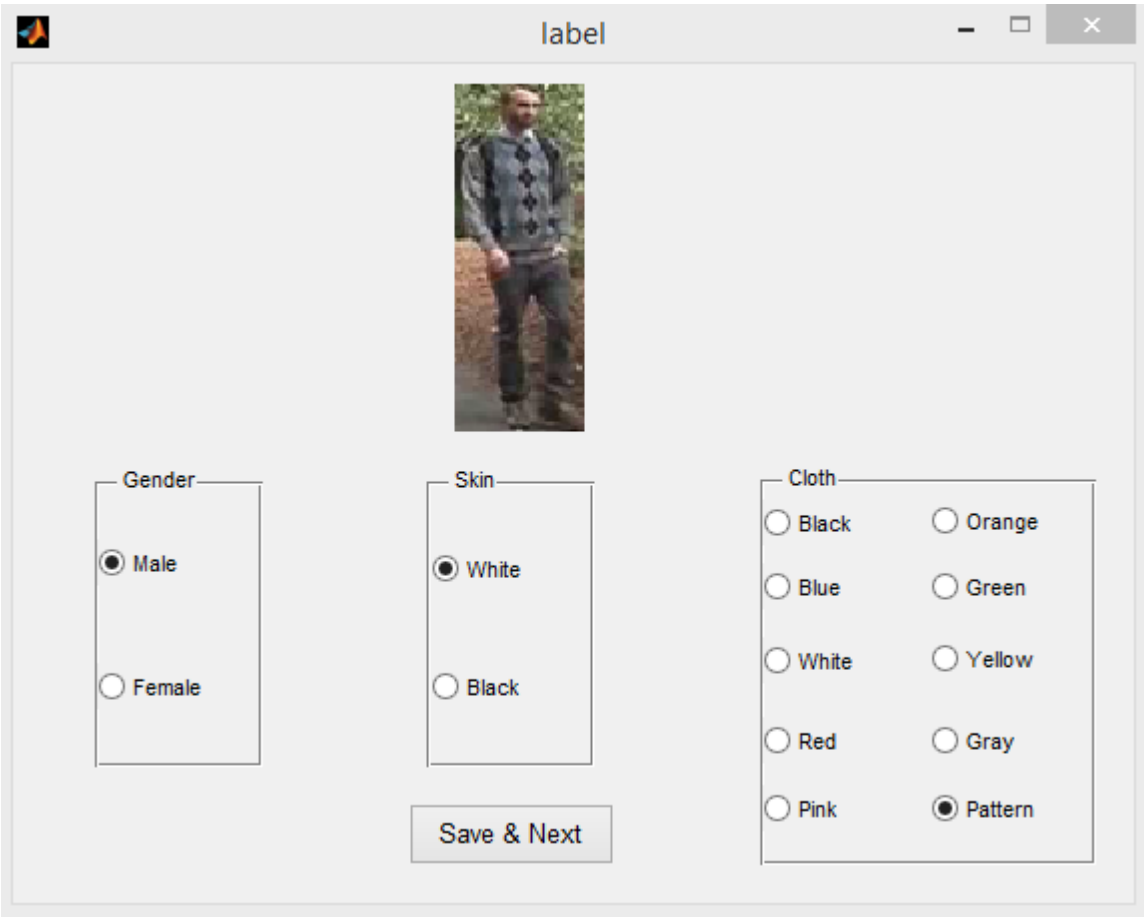

Figure 3.2: Reidit Labelling Tool. This is a complete window of the Reidit GUI.

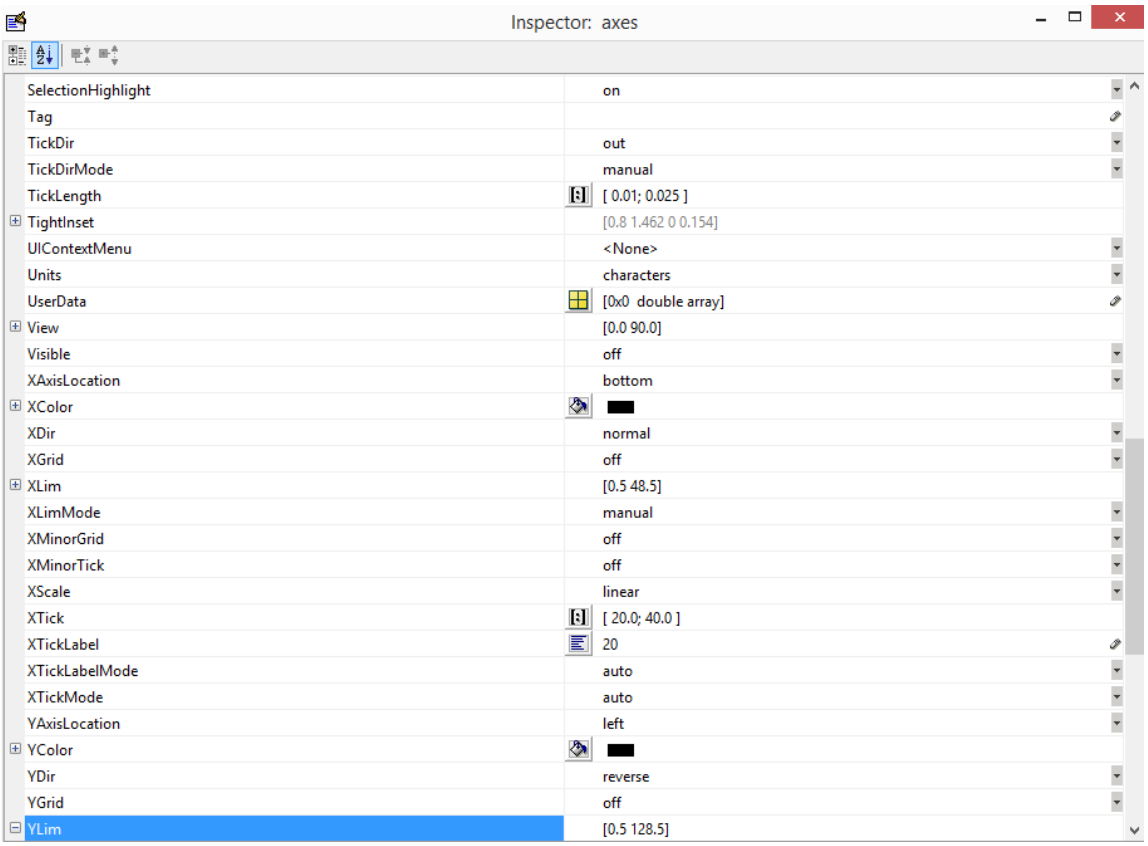

Figure 3.3: Axes Inspector. The configuration of the Axes Inspector is the default.

Tag property to push Button. Figure 3.3 shows the default Axes configuration, and Figure 3.4 shows the default Push Button setting.

We then need to relate "axes1" with "pushbutton1" since each time we finish labelling the current image, we click on the button to tell the program it can save the current label result and display the next image. Callback functions in MATLAB GUI can automatically handle the "execute/stop/quit" commands with related components. The next step is to create three button groups gender, skin, and clothing. For the gender button, we have two radio buttons that are male and female. Double

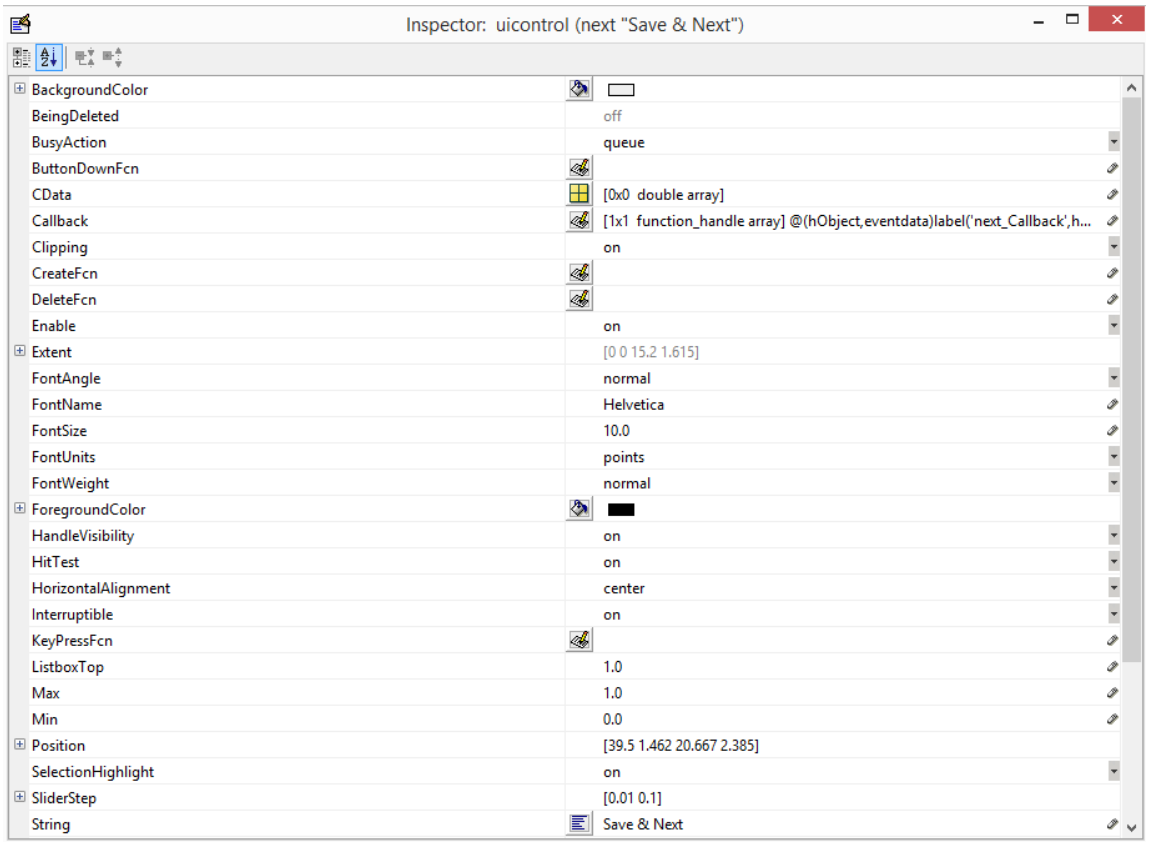

Figure 3.4: Push Button component. We change the String of the Push Button to "Save & Next".

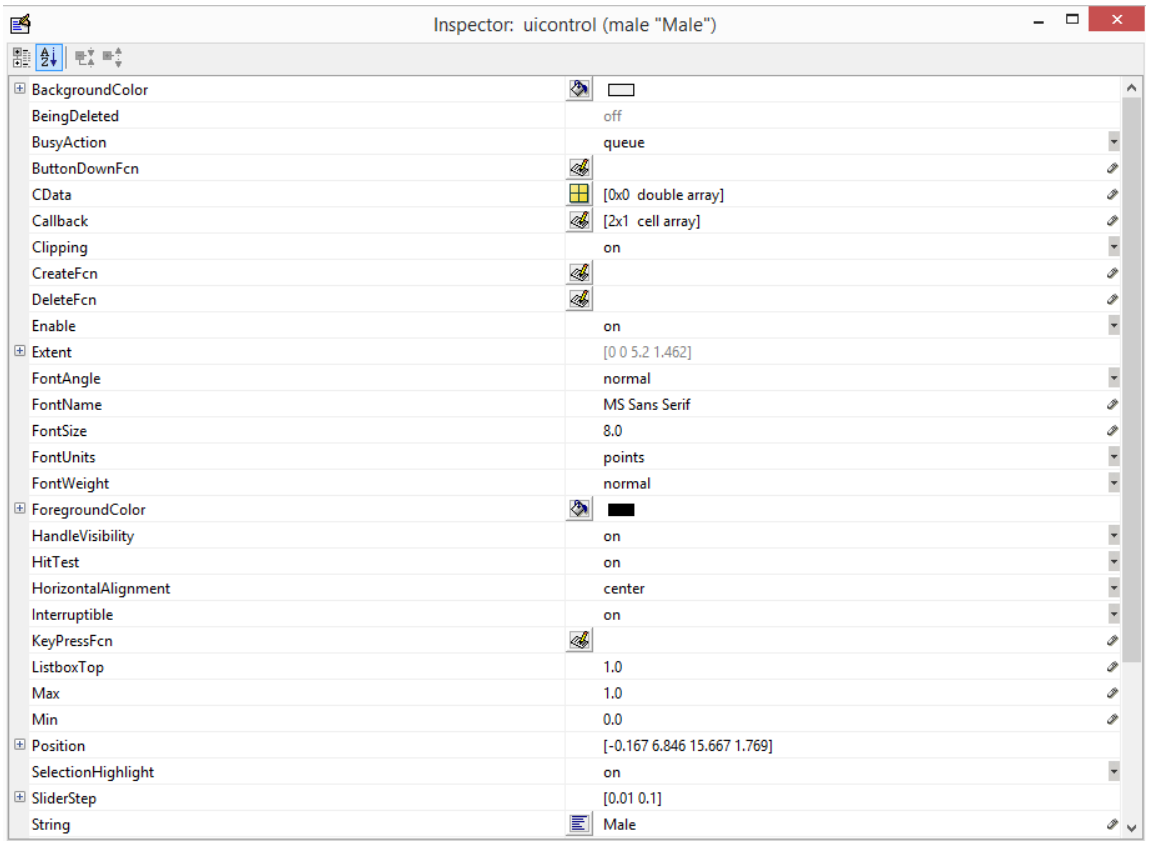

Figure 3.5: Gender Radio Button.

clicking the Radio Button will bring up the Property Inspector. W change the Tag property to "male" and change the String property to "Male", as shown in Figure 3.5. The same process applies to other features in the rest of the groups.

For the skin button group, we have two radio buttons which are white and black. For the clothing button group, we have ten radio buttons which are black, blue, white, red, pink, orange, green, yellow, gray, and pattern. These semantic color names and bio features are categorized and those with semantic names are represented numerically. For example, for gender feature, the label "male" mapped to 0 and

label “female” mapped to 1. The same scheme applies for clothing feature and skin feature. For each group, we set handle functions in GUI. Therefore whenever the user is clicking and choosing a semantic feature, the callback function can read the feature value and save it. Figure 3.6 shows the clothing feature tags and values.

To implement the Reidit GUI MATLAB function, we first set up the absolute image file paths of the dataset that is to be used for evaluation. Second, since we have gender, skin, and clothing selection options, we need to build three functions for them. For the gender selection function, there are only two tags, “male” and “female”, which we can handle them with switch statements. For skin selection function, is similar to the gender selection function in that is also has two cases. On the other hand, for the clothing selection function, there are ten cases since we categorize the color into ten types. The users can customize the number of color types by adding or deducting options in the color selection group. The dataset we use contains 632 pairs of identities, and most of them can be represented with the above two types of ethnicities, gender features and ten tags of color features. Again, if the user wants fewer or greater number of features, Reidit is customizable such that the button group can be changed in the property inspector. Moreover, instead of the radio button, there are many other options such as check box, list box. To make sure that the GUI can work, each time we create a button or a new plot, we need to set the corresponding callback function and handles in the label main function.

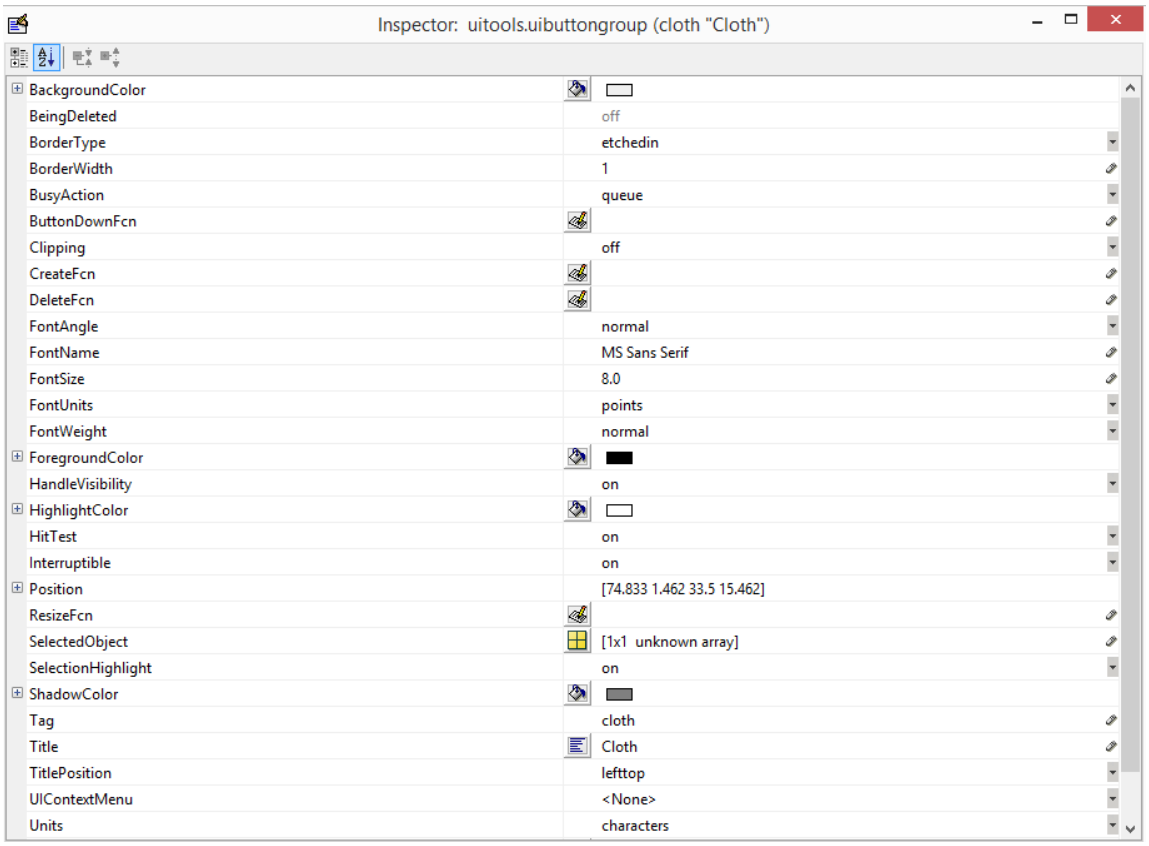

Figure 3.6: Clothing Selection Group. We use the default setting.

3.2 Semantic Color Names

In this section, we introduce the basic semantic feature extraction method which is semantic color names. Instead of using traditional color histogram concatenations or texture features, we apply the semantic color names to describe pedestrians. To decide the categories of color names, we manually checked the video frames to determine the colors that best represents the pedestrian. There are ten basic colors which are black, blue, red, yellow, green, gray, pink, orange, white and pattern. In the labelling stage, we use these ten color tags to represent clothing features.

It is not practical to manually label all 632 pedestrians for their color names, so we train a Support Vector Machines (SVM) [19] to detect attributes. We follow the approach of the detection presented in [10]. To detect features, we first label 316 images for gender, ethnicity, and clothing color info as a training sample. Then we extract 2000 dimensional color feature vectors from each image using the method described in [17]. We then train SVM to detect the color attributes. We use the method in [15] to implement the feature detection. Finally, we perform cross-validation to select the most significant values for the color name of an image.

To define color naming as an identity descriptor, a mapping function from the RGB values of an image to a color name is required. We use one-to-one matching, so a certain color name is assigned to an image. This mapping can be represented as:

$$f(\mathbf{x}_{rgb}) \rightarrow \mathbf{i} \tag{3.1}$$

where i is a nonnegative integer from 0 to 9. Layne *et al.* [10] defined 15 binary attributes regarding clothing style, hair, carrying object, and gender. We only consider gender, race, and clothing color because of the low resolution of images. RGB color names together with the bio names produce the final semantic feature. For each type of semantic feature, there are different weights, which will be explained in the following chapter.

3.3 Semantic Bio Names

In this section, we introduce semantic bio names such as gender and ethnicity. Unlike the semantic color name attributes, these bio features are high-level features that are representative. Similar to RGB color names, the mapping from bio names to the features can be represented as:

$$f(\mathbf{x}_g) \rightarrow \mathbf{j} \tag{3.2}$$

$$f(\mathbf{x}_e) \rightarrow \mathbf{k} \tag{3.3}$$

where g means the gender feature, j is either 0 or 1 which represents male or female. Similar to ethnicity e, k is also a binary value.

Combined with RGB color names, we have the following semantic features. However, we cannot apply those concatenated features in the similarity distance learning. For each attribute, there is a corresponding weight. The next section introduces the attribute-based distance function.

$$f(p) = f(f_{rgb}, f_g, f_e) \tag{3.4}$$

3.4 Metric Learning

This section introduces the methods to generate the attribute-based distance function and the metric learning technique. The distance metrics are combined to measure the distance d between two specific person images I_p and I_q.

For given descriptors of a pedestrian with gender x, ethnicity y, and clothing z, we compare this person to the rest of the pedestrians in the database using the weighted Euclidean distance. A smaller distance indicates a greater similarity between the two persons. We can integrate our semantic based distance function as follows:

$$d(I_p, I_q) = \beta_{gender} * d_g(f_g(I_p), f_g(I_q)) \tag{3.5}$$

$$+\beta_e * d_e(f_e(I_p), f_e(I_q)) \tag{3.6}$$

$$+\beta_{rgb} * d_{rgb}(f_{rgb}(I_p), f_{rgb}(I_q)) \tag{3.7}$$

where β_{gender} is the weight of attribute gender, β_e is the weight of attribute ethnicity, and β_{rgb} is the weight of color name features.

Since all attributes are not equal because of different natural properties and metrics, we cannot add them together as the representation. To generate a general feature representation, we learn a weighted distance metric $d_semantic$, where $d_{semantic}$ defines a weighted sum of the $L2$ distance between each two images:

$$d_{semantic}(I_p, I_q) = (f(p) - f(q))^T W (f(p) - f(q)) \tag{3.8}$$

$$= \sqrt{\sum_i w_i (f(p) - f(q))^2} \tag{3.9}$$

where $f(p)$ and $f(q)$ are representations of person p and person q, respectively. W provides a weight projection matrix which performs the attribute adjustments. We apply a greedy algorithm to search for the w_i, then test and optimize the above β_{gender}, β_e, and β_{rgb} with cross validation.

The algorithm we use is described as follows:

while *not the best stable performance* **do**
  for *Each person's feature* f_g, f_e *and* f_{rgb} **do**
    Evaluate recognition performance for weights β_{gender}, β_e, and β_{rgb} in range $[0, 1]$ with weight sum as 1;
  end
  Choose weighted combination attribute $f_{weighted}$;
  Choose weight w_i ;
  if *Cross validation performance is stable* **then**
    break ;
  end
end

Algorithm 1: Weighted Semantic Attribute Metric Learning

Chapter 4

Implementation

In this chapter, we focus on the GUI implementation details of Interactive Pedestrian Re-identification GUI. Chapter 4.1 introduces the basic functions and layouts of the GUI, which is subject to algorithms in Chapter 3. Chapter 4.2 shows the implementation process of the whole IPRG and parameters configuration.

4.1 Framework

The basic function of a GUI is to choose an image folder and list all the image files. We improve the display function to display the current image by showing the information of current images such as image format and dimensions. Display function is fundamental and simple to implement. We parse the full image file name of the selected image of the current folder into the string file name and its extension, then read the image to get the dimension data. We then set the corresponding MATLAB

GUI string handle with the above information that is the display function. An "Exit" button is introduced since many GUIs do not have reasonable turn off method which cause additional memory and operation problems.

The next function of IPRG is feature selection which contains bio feature options. The feature selection group includes three panels that are gender group, ethnicity group, and color group. In the gender group, there are two choices which are male and female. In the ethnicity group, there are white and black options. In the clothing color group we have black, blue, red, yellow, green, gray, pink, orange, white, and pattern options. With all these choices, we need two more buttons "Analyze" and "Save" that let the user start the matching process and save the recognition result into a mat file.

To visualize the recognition results, we need a result panel to display the top ten candidates images and a text result window to list these image names. The result display panel lists ten images in descending order of matching possibility. The text window shows the file names of the ten images.

4.2 GUI Functional Programming

In this section, we introduce the detailed IPRG configuration and the GUI functional programming based on the preprocessing results with the methods in chapter 3.

4.2.1 Select Image Folder and Display

The following panel in the MATLAB GUI contains various buttons, plots, boxes, static text and panel components (Figure 4.1). We can choose appropriate components to build the partial function.

We first add a new Push Button named "Select Image Folder" to select a folder in the operating system. To display the image file names we need a list box, and hence add Listbox from the component panel (Figure 4.2). For the Push Button, we set the string values in its inspector as "Select Image Folder" as shown in Figure 4.3.

To implement the push button function we need a select and call back function, when the user clicks the button, GUI handle is triggered to let the user set up the image path and set the selected path as the current folder. The following snippet shows the callback function.

```
function select_Callback(hObject, eventdata, handles)
% hObject    handle to select
% handles    structure with handles and user data

returnValue = uigetdir(handles.ImageFolder,'Select Image Folder');
    % returnValue will be 0 (a double) if they click cancel.
    % returnValue will be the path (a string) if they clicked OK.
    if returnValue ~= 0
        % Assign the value if they didn't click cancel.
        handles.ImageFolder = returnValue;
```

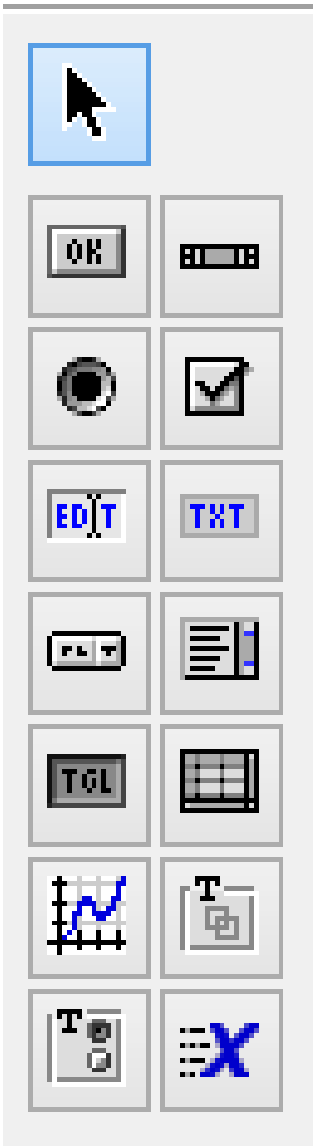

Figure 4.1: GUI Components. The options are Radio Button, Axes, Listbox, and Static Text.

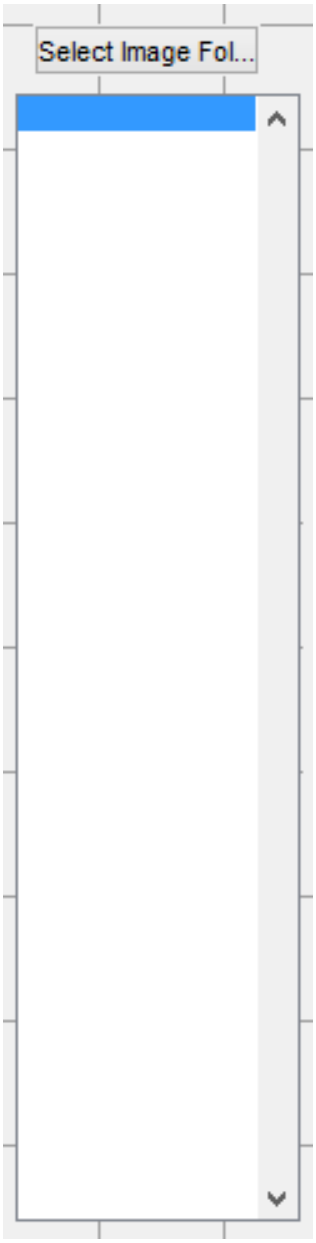

Figure 4.2: Push Button and Listbox. This is a plain configuration.

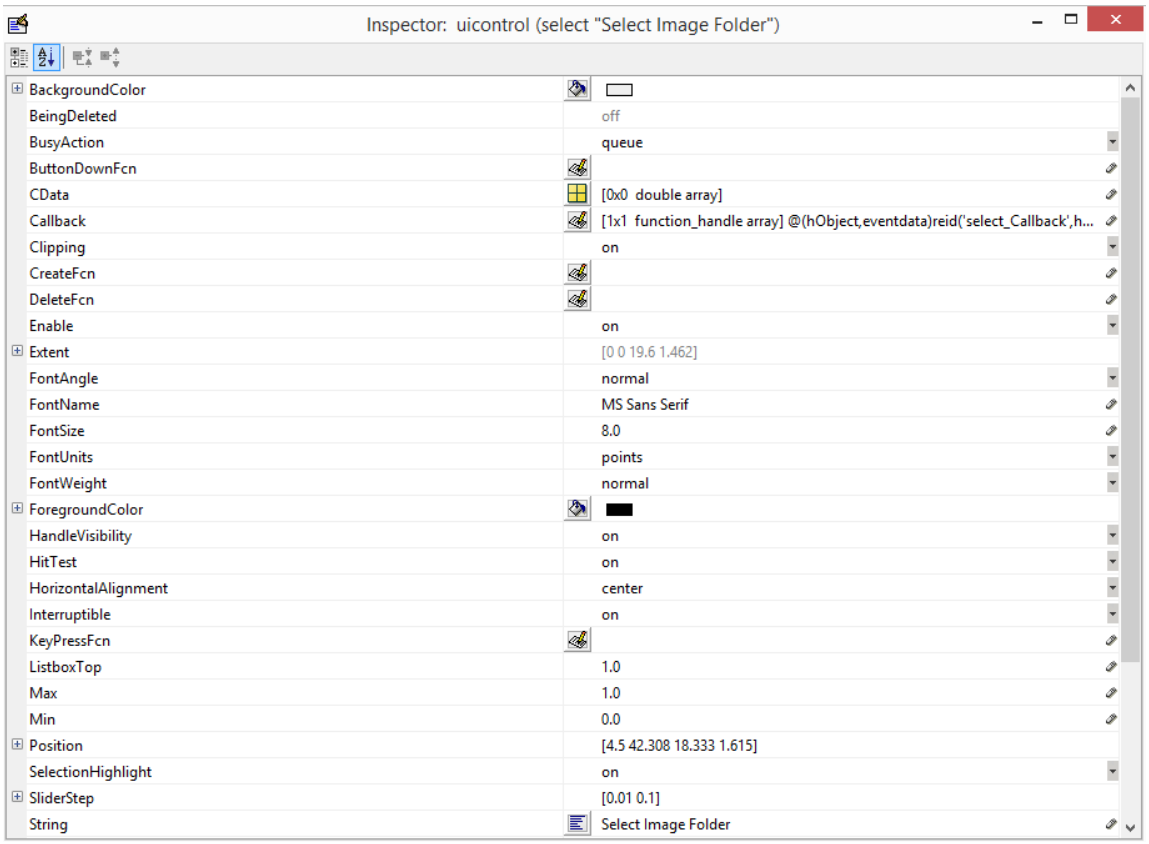

Figure 4.3: Push Button Inspector. We use the default setting.

```
        handles = LoadImageList(handles);
        guidata(hObject, handles);
        % Save the image folder in our ini file.
        SaveUserSettings(handles);
    end
    return
```

The next step is to load the list box with selected folder images. We list '.png', '.bmp', '.jpg', '.tif', '.avi' to handle image formats. Since *handles object* contains the image folder data, we can get the folder directory from it and then iteratively put image file names into the image name list. The key statement in this function is to set the list box of the *handles object* with *image name list* variable. The following code shows the image load function.

```
function handles=LoadImageList(handles)
    ListOfImageNames = {};
    folder = handles.ImageFolder;
    ImageFiles = dir([handles.ImageFolder '/*.*']);
    for Index = 1:length(ImageFiles)
        baseFileName = ImageFiles(Index).name;
        [folder, name, extension] = fileparts(baseFileName);
        extension = upper(extension);
        switch lower(extension)
        case {'.png', '.bmp', '.jpg', '.tif', '.avi'}
            ListOfImageNames = [ListOfImageNames baseFileName];
        otherwise
        end
```

```
    end
    set(handles.listbox1,'string',ListOfImageNames);
    return
```

To display the basic information of an image in the list box, we write a display image function with *handles object*. Whenever the user clicks an image, that image will be displayed and read to get the dimension information, and then a string handle will be set to show the image information in a static text window. In this case, we need to add Axes and Static Text components into the GUI. The implementation function snippet is shown below.

```
function imageArray = DisplayImage(handles, fullImageFileName)
    imageArray = []; % Initialize
    [imageArray, colorMap] = imread(fullImageFileName);
    % Display image array in a window on the user interface.
    hold off;
    % Display the image in the "axes11" axes.
    imshow(imageArray, 'InitialMagnification', 'fit', 'parent', handles.axes11);
        [folder, basefilename, extension] = fileparts(fullImageFileName);
        [rows, columns, numberOfColorChannels] = size(imageArray);
        fileInfo = dir(fullImageFileName);
    idinfo = sprintf('%s\n%d rows\n%d columns\n%d color channels', ...
    [basefilename extension], rows, columns, numberOfColorChannels);
        set(handles.idinfo, 'String', idinfo);
    return
```

4.2.2 Feature Selection Group

In this section, we present the implementation of the feature selection module, which is the central part of IPRG. With the user selected features, it analyses and returns the matching results with the application of algorithms introduced in chapter 3.

As shown in Figure 4.5, in feature selection group, there are three button groups. The top left group is the gender group which contains "Male" and "Female" radio buttons. The top right group is the ethnicity group which contains two radio buttons. The bottom group is the clothing color group with ten radio buttons. Under the feature selection group, there are two more buttons "Analyze" and "Save". Only when the user clicks on the "Analyze" button, IPRG will use user selected features and search the dataset. "Save" button can be used to save the current recognition result to a mat file.

For options in the group, we set "Male" option UserData as 0, so the UserData of "Female" is 1.

Similarly, we set UserData for ethnicity and color groups. Then we define an analyze callback function for "Analyze" Push Button execution as the following function. We use variable t1 - t3 to record the user selected features and apply *locate function* which is derived from the weighted semantic attribute metric learning method introduced in chapter 3 to find top ten candidates that satisfy those features. The following code snippet is the analysis main function.

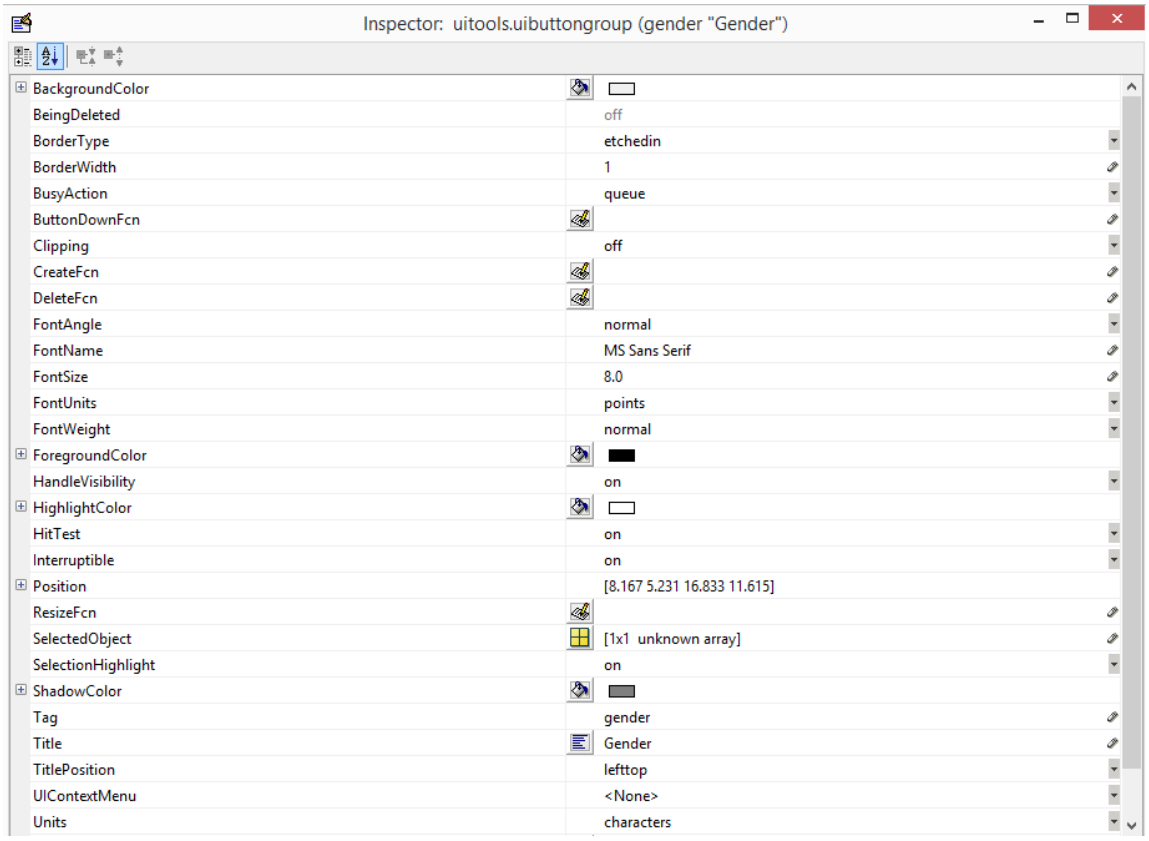

Figure 4.4: Gender Group. Default configuration.

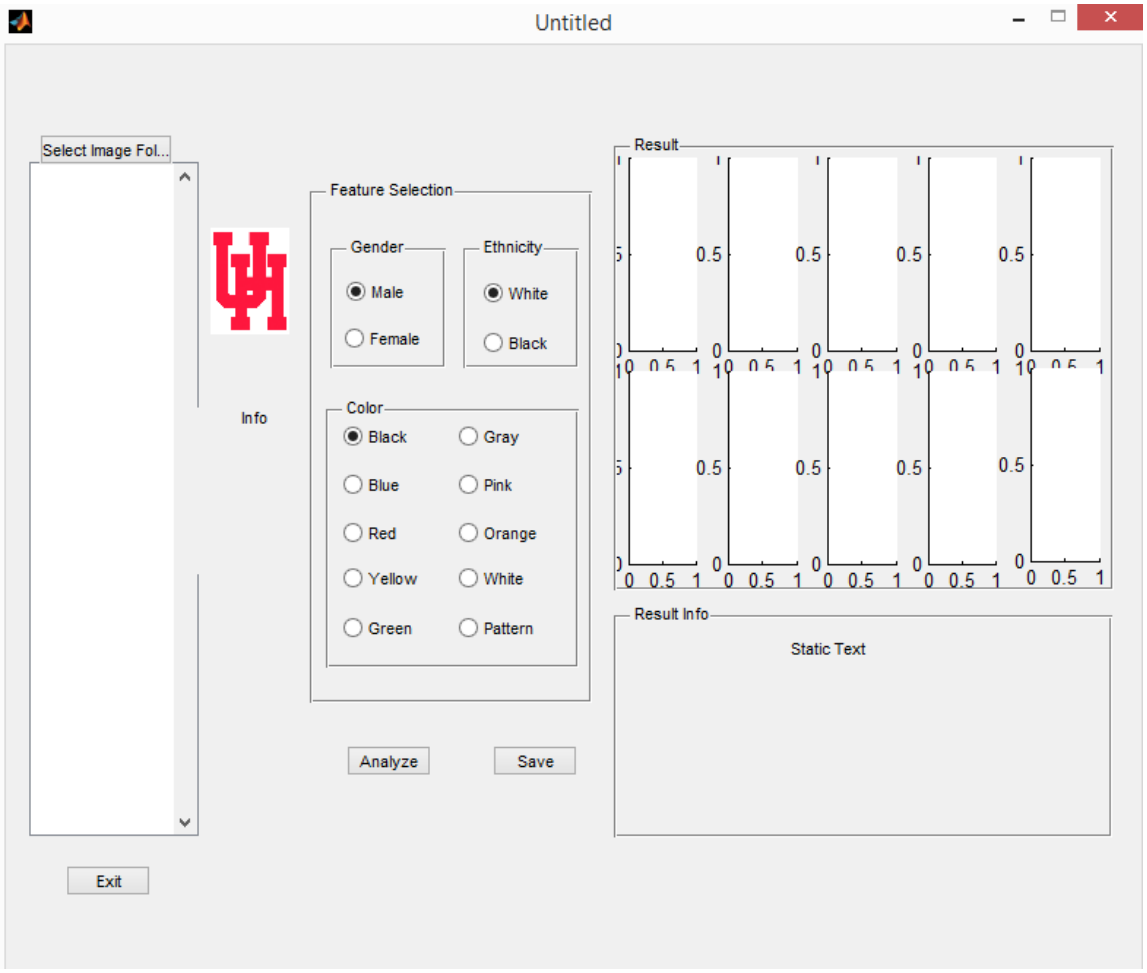

Figure 4.5: Feature Selection Group. The basic function is on the left part: Selection image folder and display images. The primary re-identification function is the middle column and rightmost column. Feature selection provides the user with several options, result axes and result text window display the matching result.

```
function analyze_Callback(hObject, eventdata, handles)
% hObject    handle to analyze (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
%%%%%%%%%%test
    t1 = get(get(handles.gender,'SelectedObject'), 'Userdata');
    t2 = get(get(handles.ethnicity,'SelectedObject'), 'Userdata');
    t3 = get(get(handles.color,'SelectedObject'), 'Userdata');
    id = locate(t1, t2, t3);
```

4.2.3 Result Display

To display those top ten identities, we simply apply image display function (*imshow*) in MATLAB to display images on ten Axes (Figure 4.6). Meanwhile, we create a result info Static Text window (Figure 4.7, Figure 4.8) to display image names of result images.

In summary, the following Object Browser Figure 4.9 shows the complete layout of IPRG.

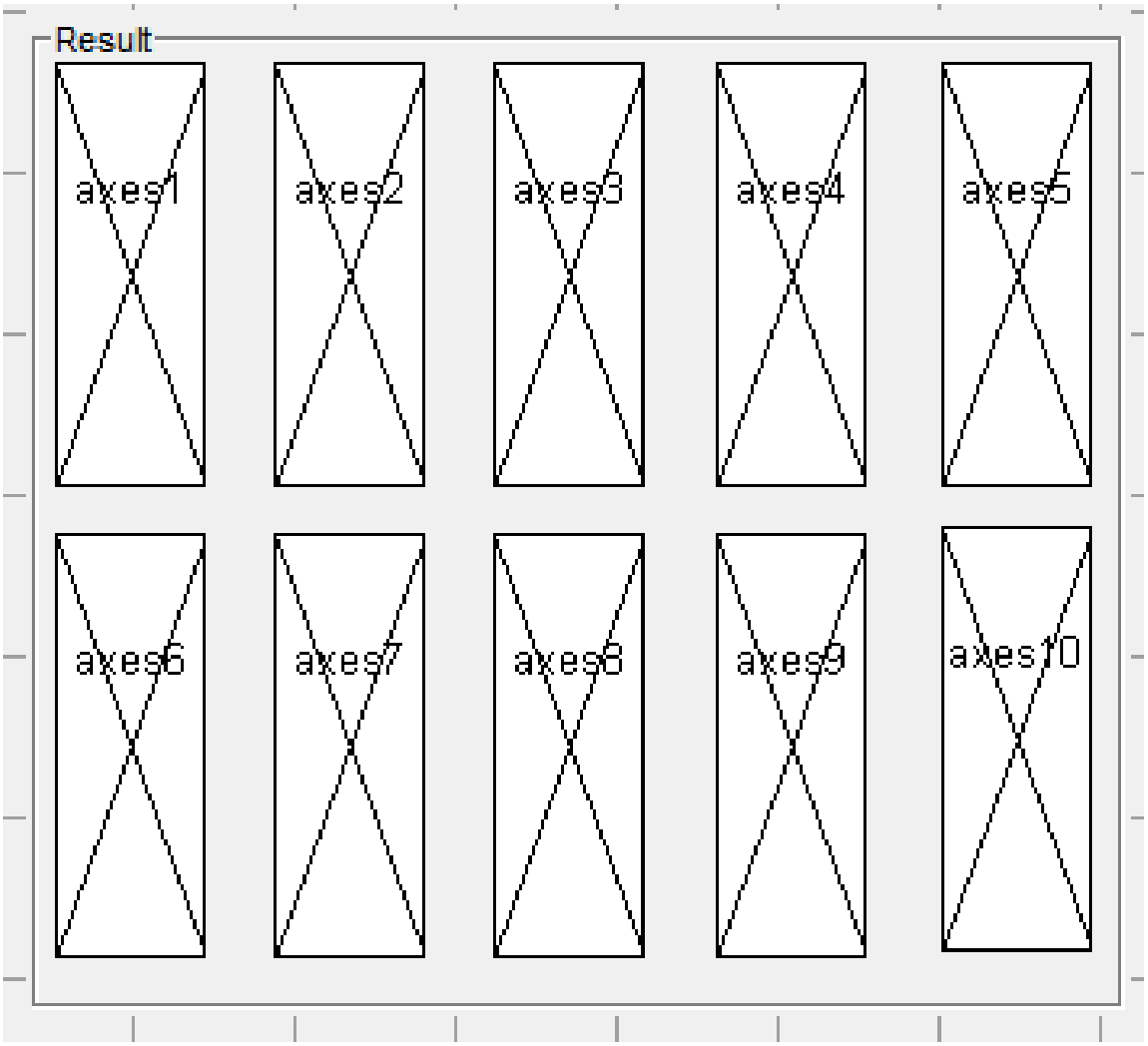

Figure 4.6: Result Axes. There are ten axes plots to display the top ten candidates.

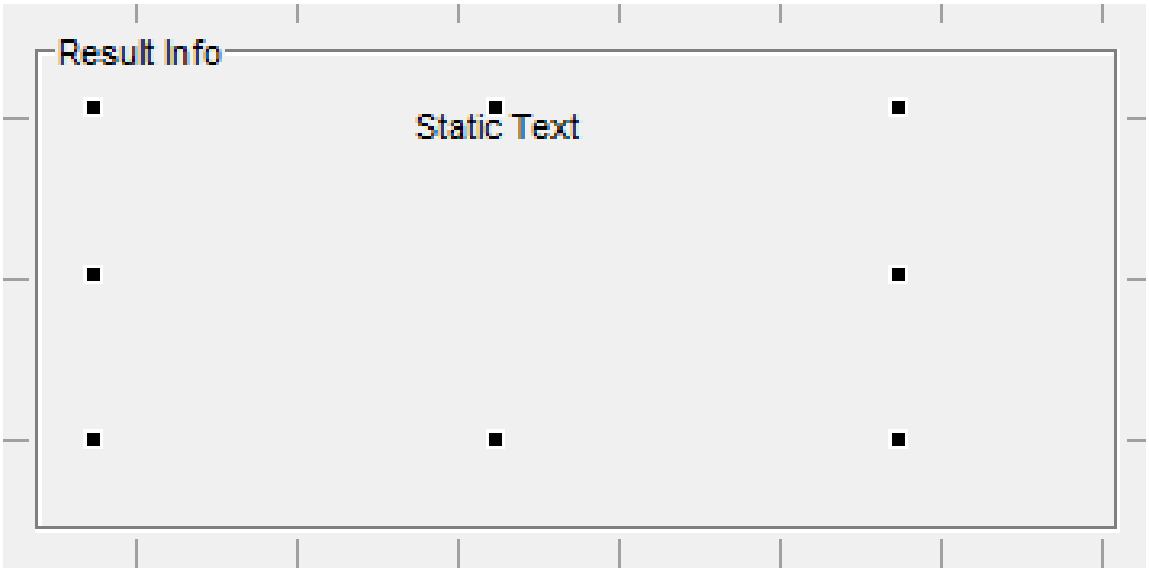

Figure 4.7: Static Text Window. Shows the information of the top ten candidates.

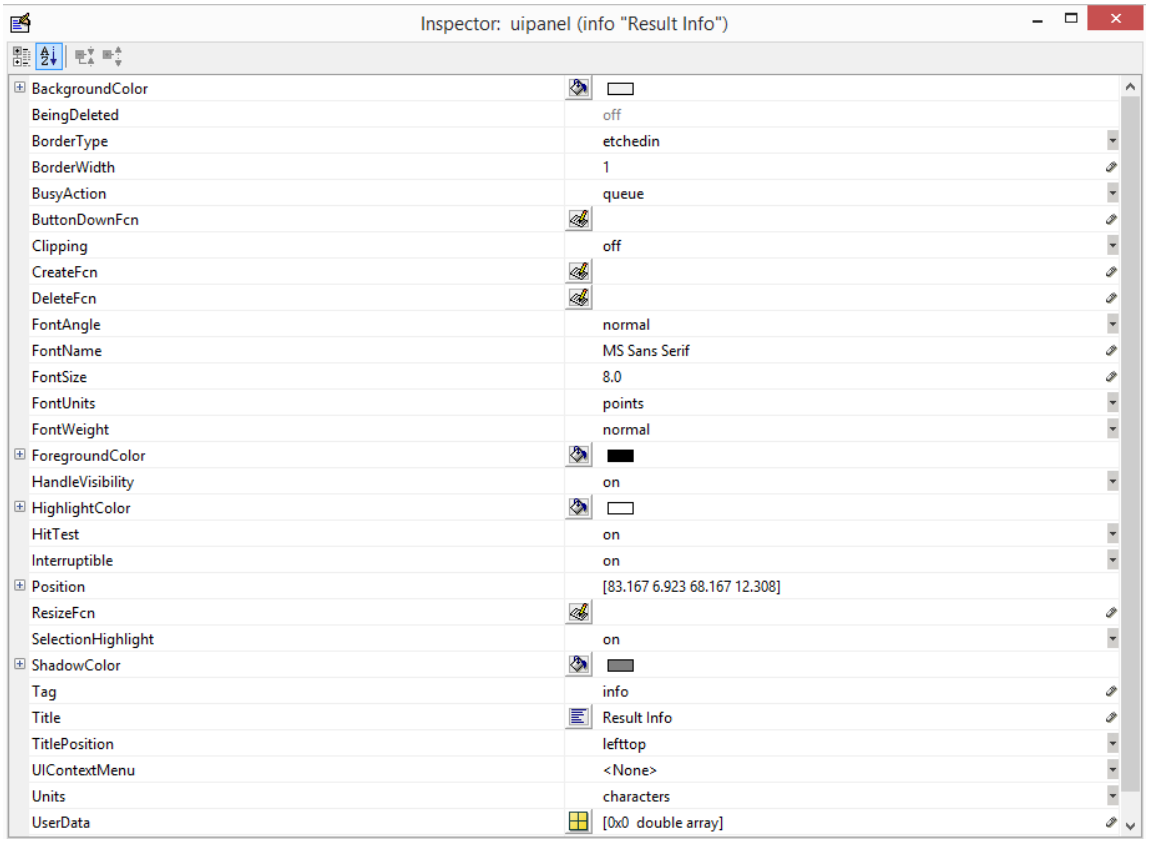

Figure 4.8: Static Text Inspector. Default settings.

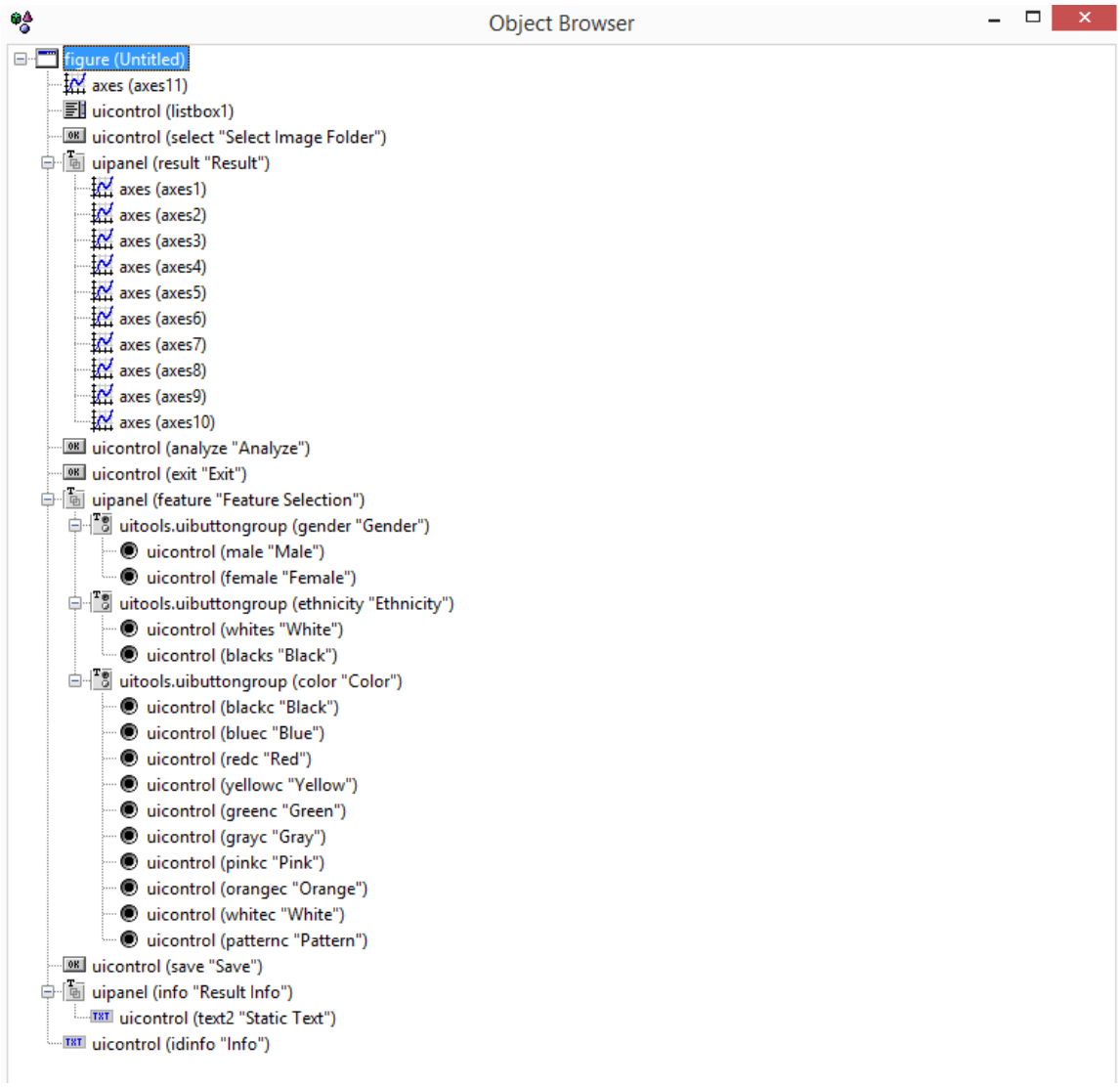

Figure 4.9: IPRG Layout. The logical layout of all the components in IPRG GUI.

Chapter 5

Experiments

In this chapter, we present the function of each module of IPRG and test it with three examples. Chapter 5.1 displays the result of the automatic semantic color name detection on both camera video frames. Chapter 5.2 presents the basic display and other functions of IPRG. Chapter 5.3 focuses on the testing of the semantic feature based pedestrian re-identification.

5.1 Semantic Color Name Detection

This section presents the result of automatic semantic color name detection on both probe and gallery images. Figure 5.1 shows the general automatic semantic color name detection on probe images. Figure 5.2 shows the results of automatic semantic color name detection on all image pairs.

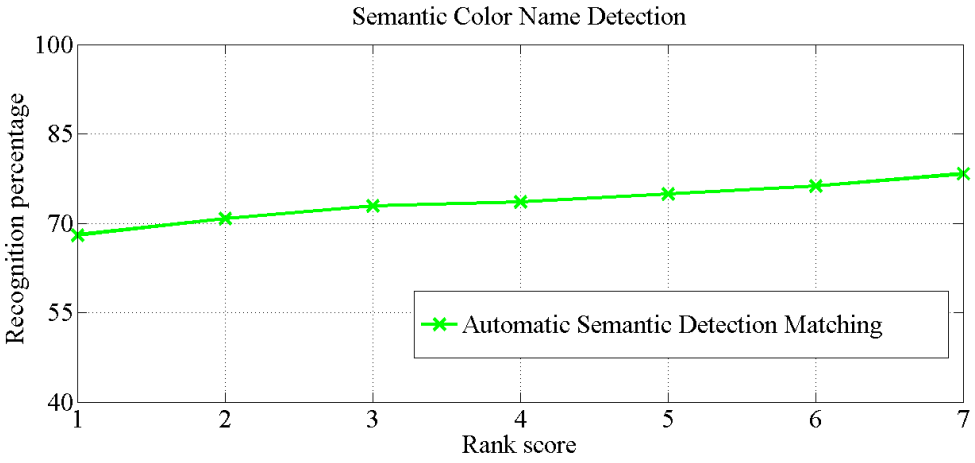

Figure 5.1: Automatic Semantic Color Name Detection on Probe Images. Rank 1 score is 68% recognition accuracy.

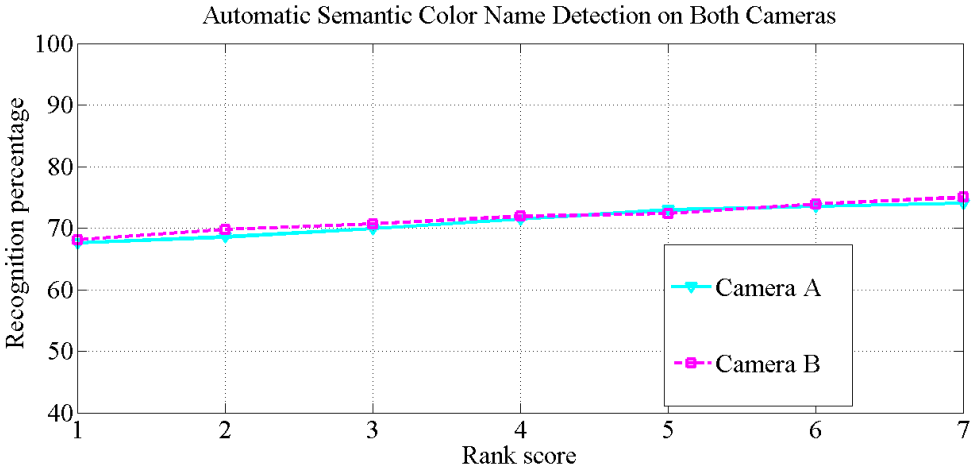

Figure 5.2: Automatic Semantic Color Name Detection on Both Camera Video Frames. IPRG performs similarly on Camera A and Camera B images. Both of the rank 1 scores are around 68% accuracy.

Table 5.1: Automatic Semantic Color Name Detection Results

rank	Camera A	Camera B
1	68.23%	67.94%
2	70.15%	69.36%
3	72.53%	71.97%
4	73.17%	73.06%
5	74.08%	73.81%
6	74.93%	74.52%
7	75.76%	76.84%
8	76.95%	76.55%
9	78.07%	77.97%
10	79.96%	79.76%

Table 5.1 presents the automatic semantic color name detection on 361 images of both cameras. Table 5.2 is the metric learning weights for three attributes.

Table 5.2: Attributes Weights

β	β_{gender}	β_e	β_{rgb}
	0.371	0.273	0.356

5.2 Basic Functions

This section displays the basic uses of IPRG including folder selection, image display. Figure 5.3 shows the initial window of IPRG. We can select an image folder as shown in Figure 5.4.

When we set up an image folder, we can check the basic info of an image in the folder as Figure 5.5 shows.

5.3 Testing on the Re-identification Module

This section presents the testing process of recognition module, and the save function of IPRG.

Figure 5.6 shows an example of a white male in a white shirt. In the result display window, we can see that the third man in the first row is the true match. Figure 5.7 is an example of a white male with "pattern" shirt. And we can see the top ten result with these features. The first man is exactly the candidate we want to match.

Figure 5.8 presents a candidate with feature "Male", "'White" and "Red". The recognition result on the right side shows the last man of the first row is the candidate that matches with the image shown on the left. Figure 5.9 is the string format result of the above example. We can save it to a mat file as shown in Figure 5.10. Figure 5.11 is the result of matching on both camera video frames.

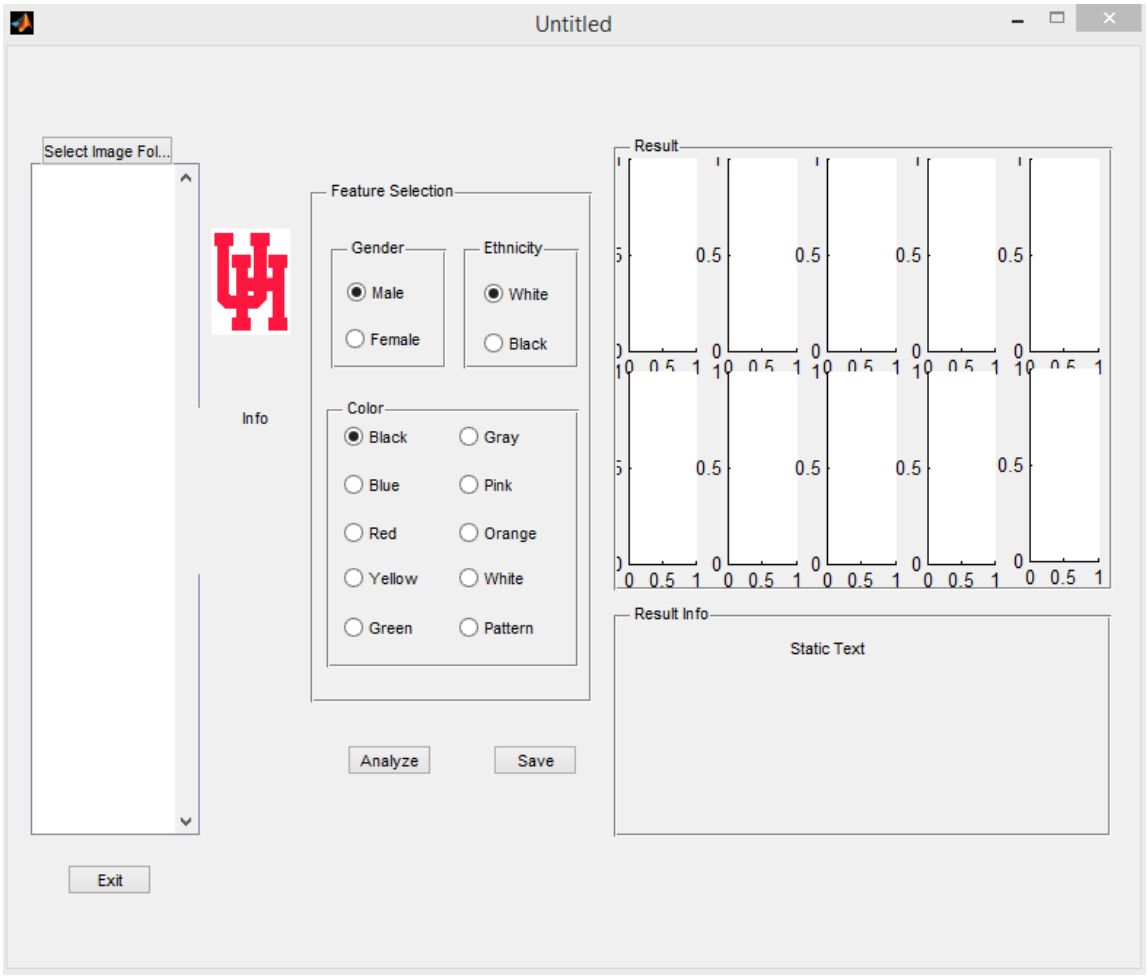

Figure 5.3: Initial Window. Before we select the image folder.

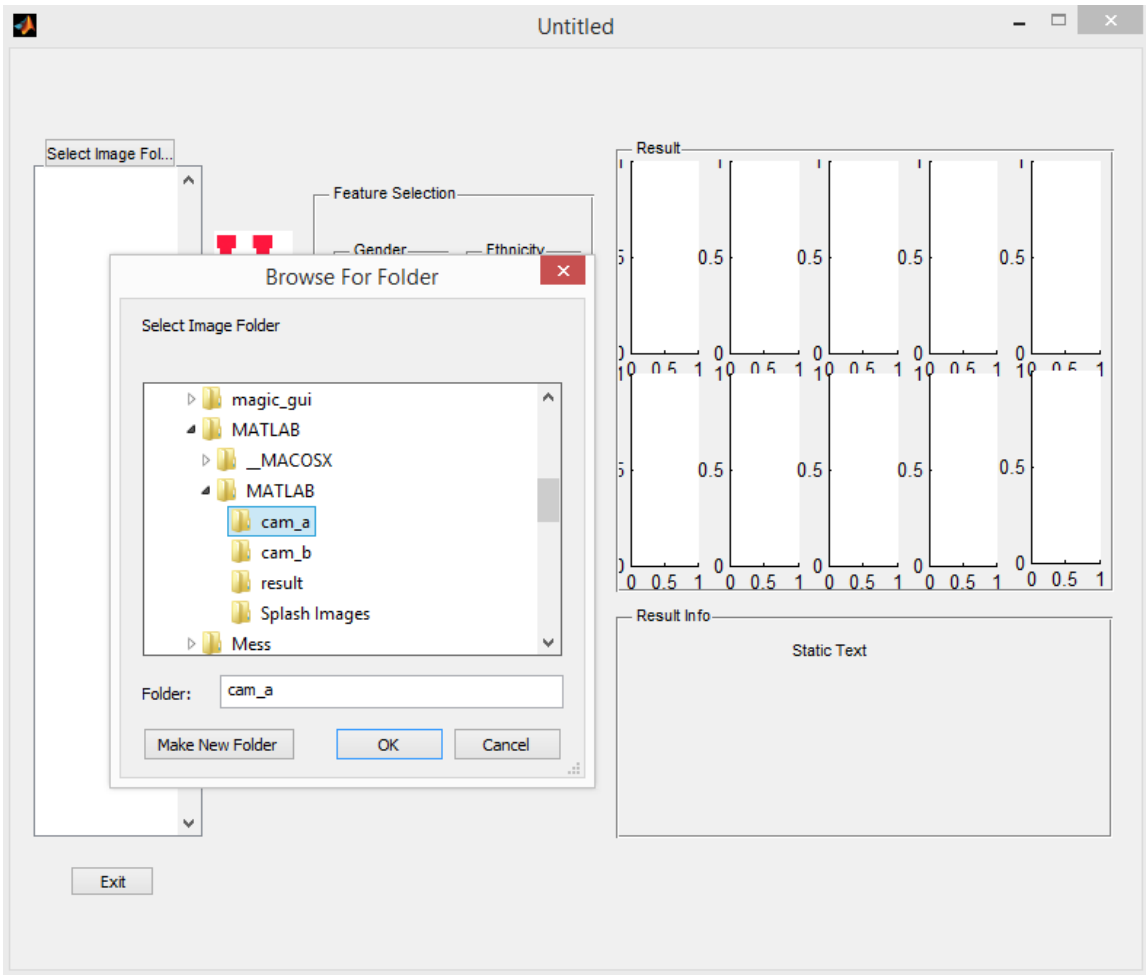

Figure 5.4: Select An Image Folder. We choose an image folder in the operating system.

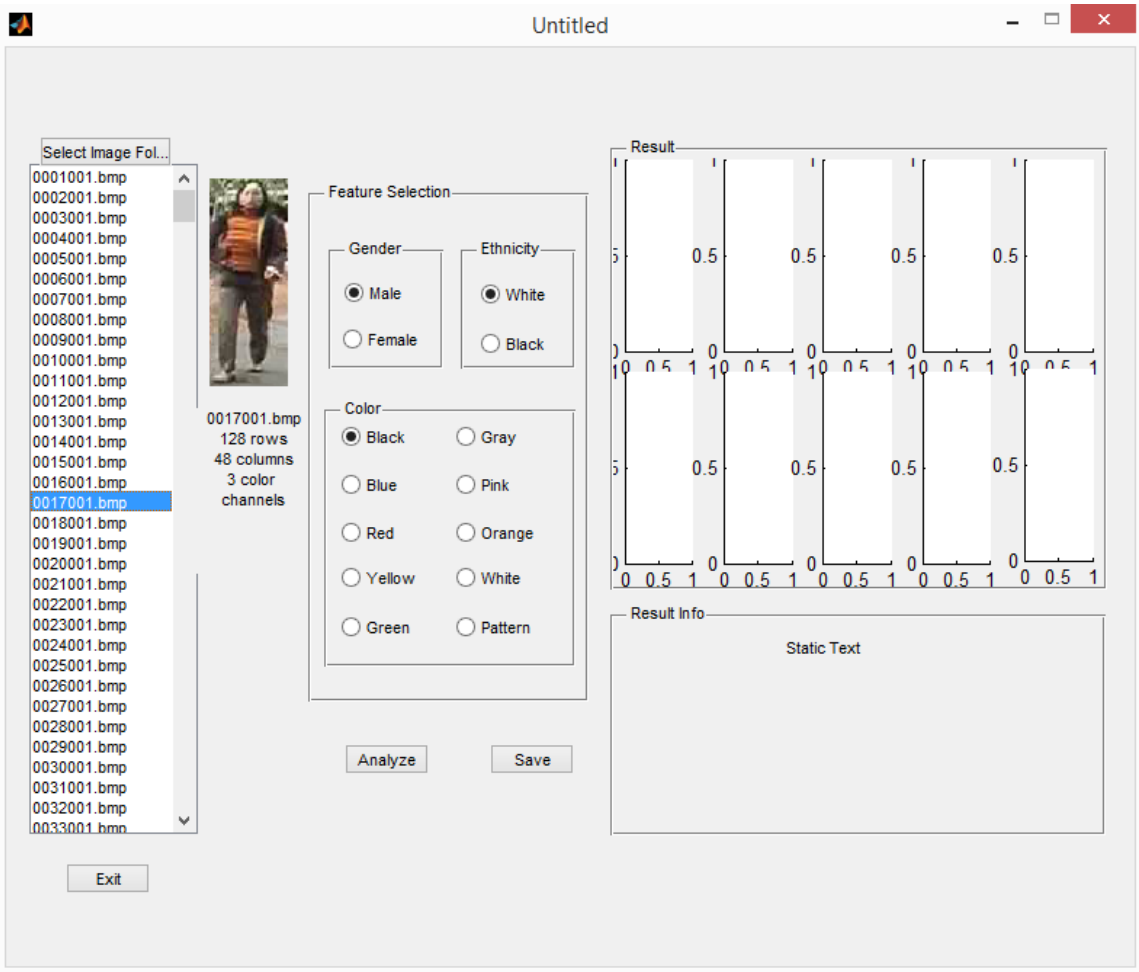

Figure 5.5: Basic Info of an Image. Click on an image, we can check the basic name info and color channels of the current image.

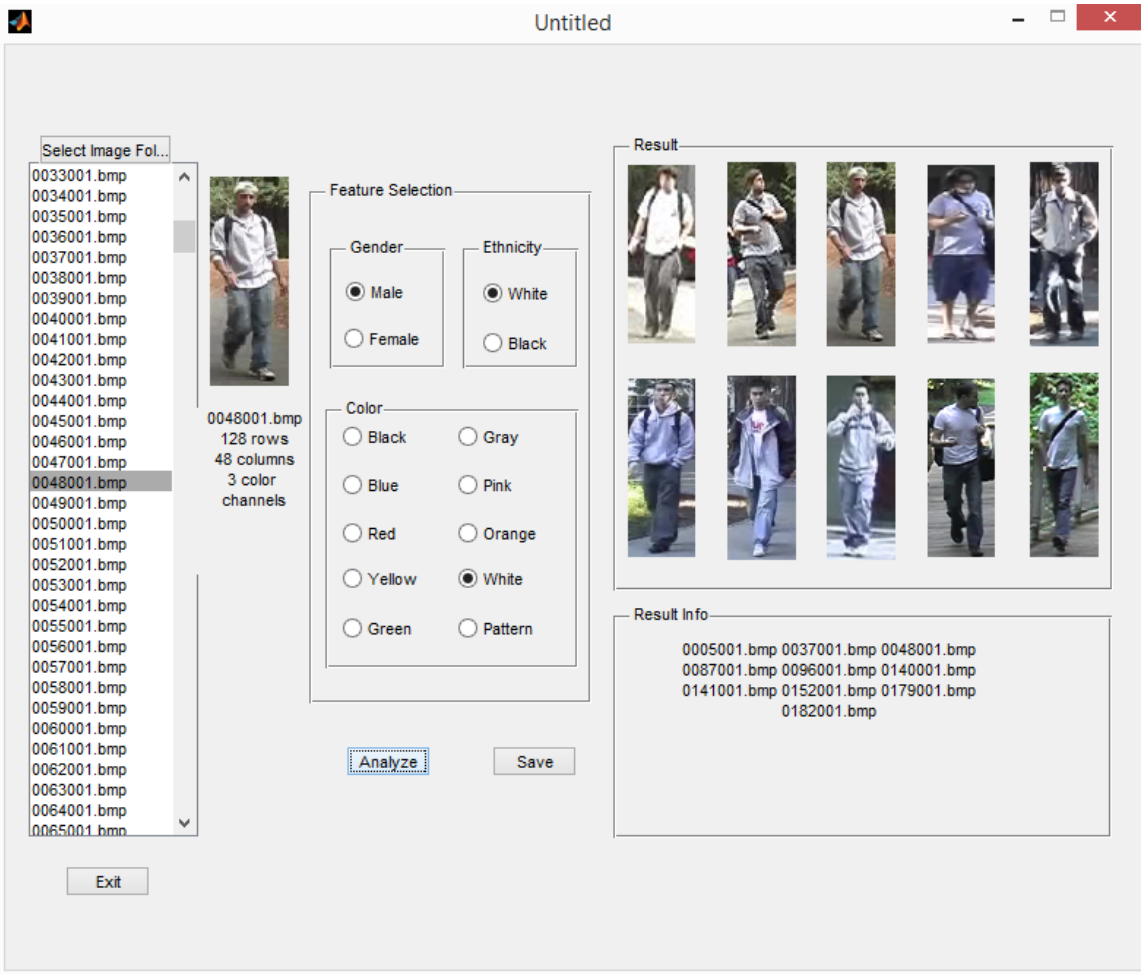

Figure 5.6: Testing Example 1. The matching results of the current candidates are listed in the result groups. The correct match is the candidate on the top row and third column from left to right.

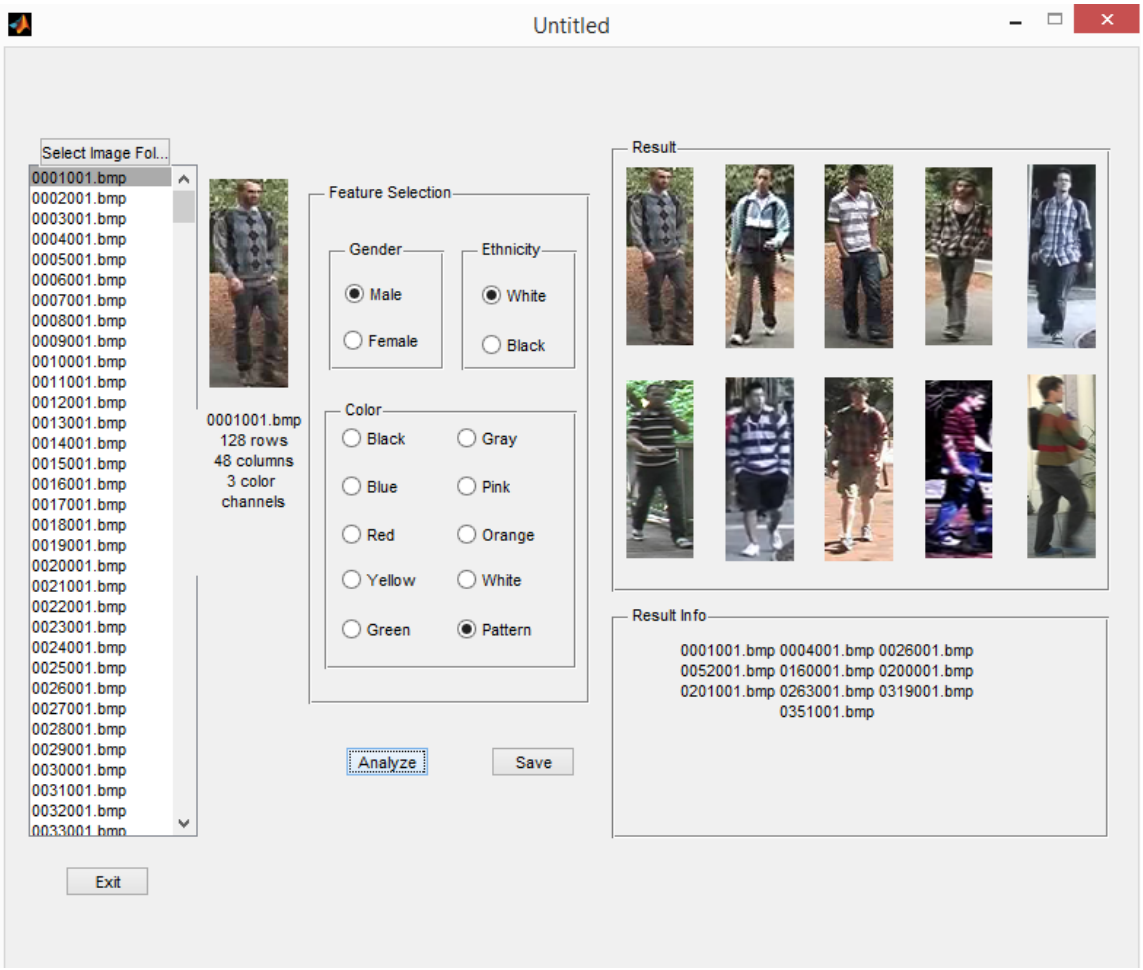

Figure 5.7: Testing Example 2. The true match is the candidate on the top left.

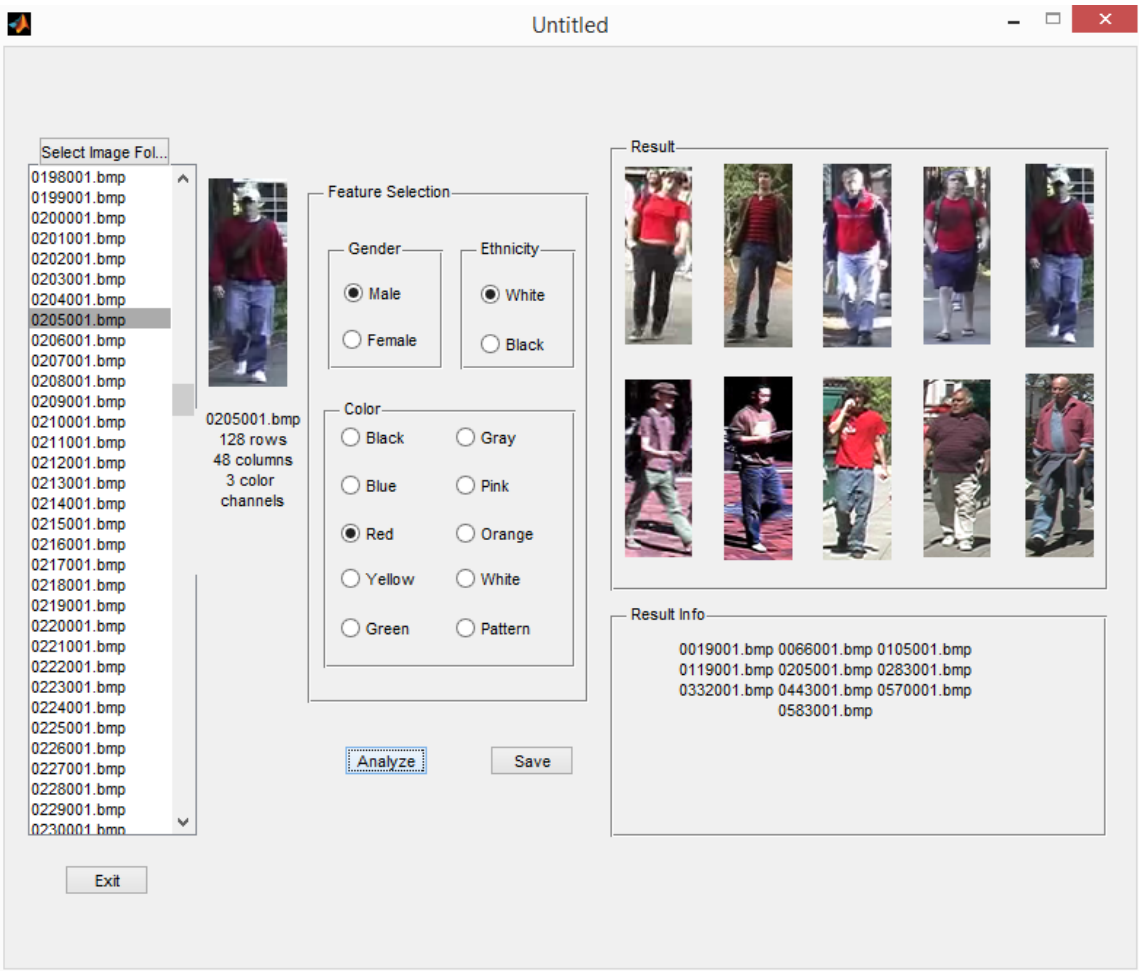

Figure 5.8: Testing Example 3. The true match is the last candidate on the first row.

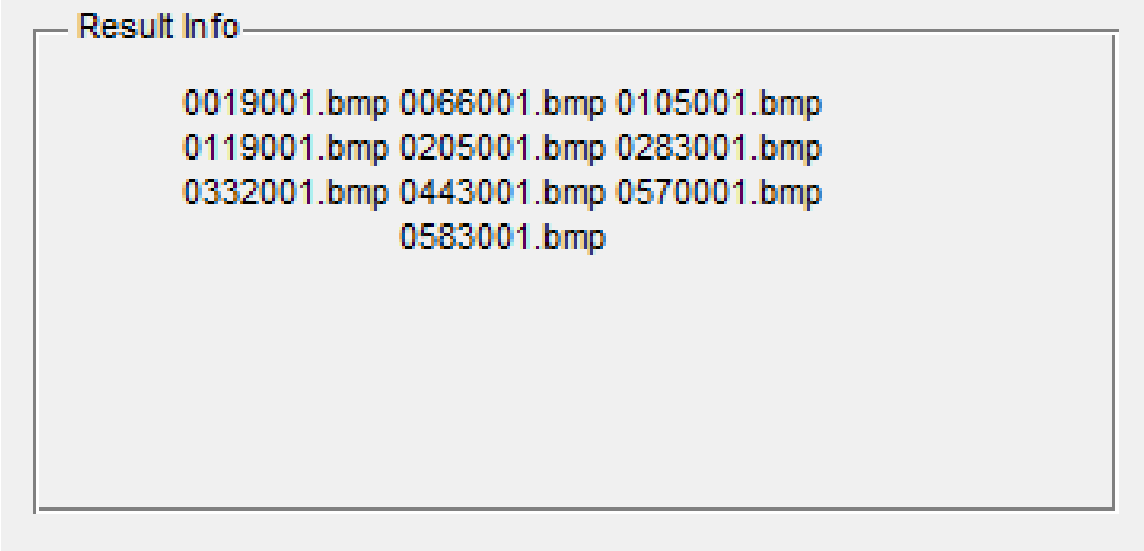

Figure 5.9: Result of Example 3. The Static Text window displays the related top ten results.

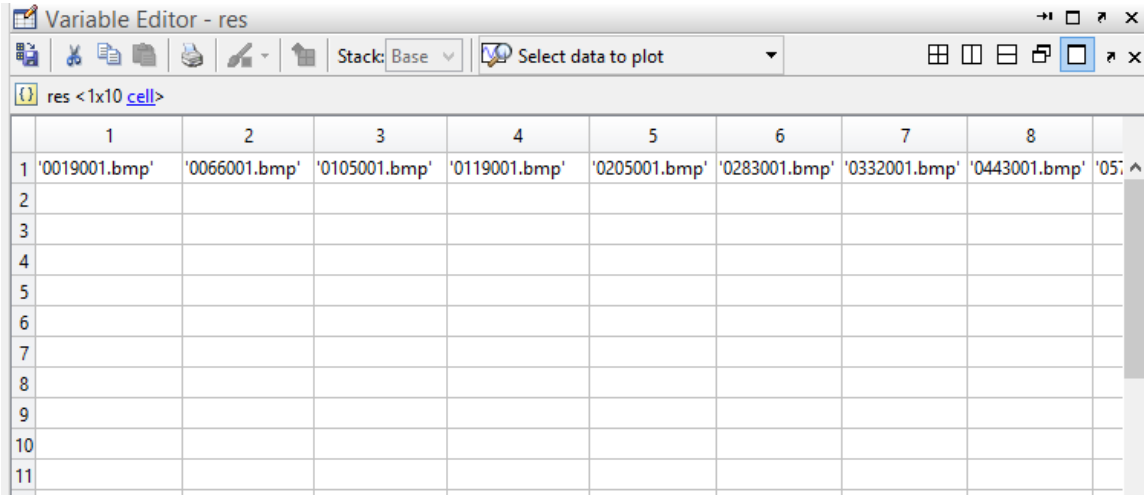

Figure 5.10: Save the Result to a Mat file. It contains image names of the top ten candidates.

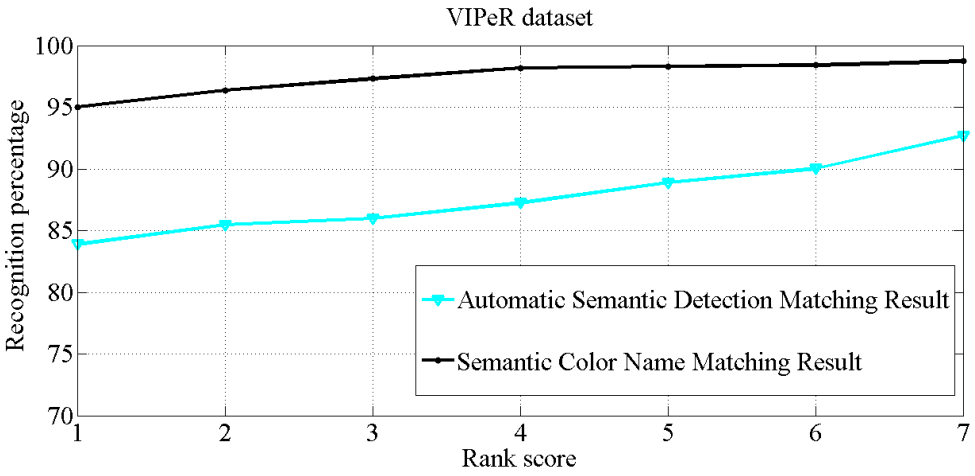

Figure 5.11: Semantic Attribute Based Matching. The rank 1 accuracy of automatic semantic detection data is 84%. With well-labelled data, the matching results on rank 1 is 95%.

Table 5.3 presents the re-identification results of IPRG with the semantic attribute based metric learning. Rank 1 score achieves 95% recognition accuracy which outperforms the traditional person re-identification methods.

Table 5.3: Semantic Attribute Based Pedestrian Re-identification Result. From rank 1 to rank 10, the rank scores of Camera and Camera B are similar, but IPRG performs better on Camera A. Rank 1 accuracy of Camera A and Camera B are 95.12% and 94.74%, respectively.

rank score	Camera A	Camera B
rank 1	95.12%	94.74%
rank 2	96.55%	95.36%
rank 3	97.89%	97.06 %
rank 4	98.15%	97.88%
rank 5	99.03%	98.69%
rank 6	100%	99.57%
rank 7	100%	100%
rank 8	100%	100%
rank 9	100%	100%
rank 10	100%	100%

Chapter 6

Conclusions

In this work, we introduced an original Interactive Pedestrian Re-identification GUI (IPRG) which addresses the campus person searching and matching problem. Moreover, we developed a semantic based image labelling tool Reid It (Reidit) which provides the user with semantic descriptor options. In preprocessing, we did the labeling on 632 pairs of pedestrian images. For the experiment, we improved SDALF metric learning with weighted semantic color names to achieve better recognition rate.

Motivated by reality surveillance applications, we developed this pedestrian searching tool to improve the public safety on campus. Since it is an application of person re-identification, our work mainly focused on data preprocessing, GUI design, and implementation. To recognize the identity accurately and efficiently, we need well-labelled images. As a result, we developed Reidit. Different from other general labelling tools, Reidit labelling tool is a customizable semantic GUI, which records

the semantic representatives of the person instead of color-based information. In the real-time searching of the pedestrian, low-dimensional semantic attributes achieve better performance and computation efficiency than the high-dimensional color feature. To process and apply the matching algorithm on these data easily, we chose MATLAB GUI as the interface to implement the re-id GUI. Because MATLAB GUI provides point-and-click control of software applications and contains controls such as menus, toolbars, buttons, etc.

Our experiments achieve 95% accuracy when testing with the VIPeR dataset with improved metric learning algorithm and stable and robust semantic attributes. We can see that feature is significant to the follow-up work for person re-identification. In the future, we need to study and explore more feature extraction methods which can represent the person better regardless of variations of pose or illumination. Moreover, it is necessary to find a more efficient labelling approach to deal with high volume data instead of labelling several examples and learning to detect the features. In the future work, we will apply several machine learning methods to generate an automatic pipeline to preprocess the raw data based on the part of manually processed data.

Last but not least, we are expecting to provide compatible cross platform tool which will be more practical and convenient to use in the industry and surveillance area. Meanwhile, instead of separating the re-identification into stages as preprocessing, metric learning, and matching, it will be more compact to fuse these stages into one pipeline systemized application.

Bibliography

[1] L. Bazzani, M. Cristani, and V. Murino. Symmetry-driven accumulation of local features for human characterization and re-identification. *Computer Vision and Image Understanding*, 117(2):130–144, 2013.

[2] A. Bedagkar-Gala and S. K. Shah. A survey of approaches and trends in person re-identification. *Image and Vision Computing*, 32(4):270–286, 2014.

[3] A. Bellet, A. Habrard, and M. Sebban. A survey on metric learning for feature vectors and structured data. *arXiv preprint arXiv:1306.6709*, 2013.

[4] D. S. Cheng, M. Cristani, M. Stoppa, L. Bazzani, and V. Murino. Custom pictorial structures for re-identification. In *BMVC*, volume 1, page 6, 2011.

[5] R. De Maesschalck, D. Jouan-Rimbaud, and D. L. Massart. The mahalanobis distance. *Chemometrics and intelligent laboratory systems*, 50(1):1–18, 2000.

[6] S. Gong, M. Cristani, C. C. Loy, and T. M. Hospedales. The re-identification challenge. In *Person Re-Identification*, pages 1–20. Springer, 2014.

[7] D. Gray and H. Tao. Viewpoint invariant pedestrian recognition with an ensemble of localized features. In *European conference on computer vision*, pages 262–275. Springer, 2008.

[8] M. Guillaumin, J. Verbeek, and C. Schmid. Is that you? metric learning approaches for face identification. In *2009 IEEE 12th International Conference on Computer Vision*, pages 498–505. IEEE, 2009.

[9] Z. Imani and H. Soltanizadeh. Person reidentification using local pattern descriptors and anthropometric measures from videos of kinect sensor. *IEEE Sensors Journal*, 16(16):6227–6238, 2016.

[10] R. Layne, T. M. Hospedales, S. Gong, and Q. Mary. Person re-identification by attributes. In *BMVC*, volume 2, page 8, 2012.

[11] S. Liao, Y. Hu, X. Zhu, and S. Z. Li. Person re-identification by local maximal occurrence representation and metric learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2197–2206, 2015.

[12] C. Liu, S. Gong, C. C. Loy, and X. Lin. Person re-identification: What features are important? In *European Conference on Computer Vision*, pages 391–401. Springer, 2012.

[13] B. Ma, Y. Su, and F. Jurie. Local descriptors encoded by fisher vectors for person re-identification. In *European Conference on Computer Vision*, pages 413–422. Springer, 2012.

[14] B. Ma, Y. Su, and F. Jurie. Discriminative image descriptors for person re-identification. In *Person Re-Identification*, pages 23–42. Springer, 2014.

[15] S. Maji, A. C. Berg, and J. Malik. Classification using intersection kernel support vector machines is efficient. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[16] A. Malhi and R. X. Gao. Pca-based feature selection scheme for machine defect classification. *IEEE Transactions on Instrumentation and Measurement*, 53(6):1517–1525, 2004.

[17] B. Prosser, W.-S. Zheng, S. Gong, T. Xiang, and Q. Mary. Person re-identification by support vector ranking. In *BMVC*, volume 2, page 6, 2010.

[18] P. M. Roth, M. Hirzer, M. Koestinger, C. Beleznai, and H. Bischof. Mahalanobis distance learning for person re-identification. In *Person Re-Identification*, pages 247–267. Springer, 2014.

[19] B. Schölkopf and A. J. Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[20] K. Q. Weinberger and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10(Feb):207–244, 2009.

[21] Y. Zhang and S. Li. Gabor-lbp based region covariance descriptor for person re-identification. In *Image and Graphics (ICIG), 2011 Sixth International Conference on*, pages 368–371. IEEE, 2011.

[22] R. Zhao, W. Ouyang, and X. Wang. Unsupervised salience learning for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3586–3593, 2013.