Person Reidentification in a Static Camera Network

Jacob Menashe
The University of Texas at Austin
Austin, TX
jmenashe@cs.utexas.edu

Aashish Sheshadri
The University of Texas at Austin
Austin, TX
aashishs@cs.utexas.edu

Abstract

In this paper we examine the person reidentification problem: given a small set of training images of a particular person, how do we reidentify that person from a new image set? To accomplish this task we propose a system consisting of facial recognition, attribute detection, and color signature comparisons. We evaluate each component in isolation and then combine these components into an integrated system. We show that with modest domain assumptions we can achieve overall reidentification accuracy of 40%, and lay the groundwork for future improvements to this technique.

1. Introduction

Detection and classification are well-studied topics in computer vision, and serve a critical role in applied autonomy. In general these systems rely on complex training stages, large sample sets, and computationally inefficient techniques; in this way, the classic computer vision approach can be unrealistic for general applications. For the domains of robotics and surveillance, it is often necessary to obtain results from imperfect and limited datasets, with an added emphasis on performance.

Identification builds on the detection problem by requiring not only that a detection succeed, but that particular characteristics of that detection be recognized and meaningfully quantized. In order to reidentify a person consistently, we must therefore detect and consistently process the characteristics of that person.

In this work we seek to tackle the person reidentification problem: given a person in multiple arbitrary scenes, can we detect, identify, and track that person throughout a surveillance system? More importantly, how can we reidentify that person when they move in and out of sight?

To accomplish this feat we employ a variety of methods developed by the vision community for particular, isolated tasks, and integrate these methods as components in our person reidentification algorithm. We use a combination of face and body part detection, facial recognition, attribute detection, and color signatures to construct quantized, singular identities for detected persons in camera scenes based on detected characteristics. We therefore decompose the problem of person reidentification into the problem of accurately identifying and comparing such characteristics.

2. Background and Related Work

Person detection and identification has been treated as two separate problems in the vision community. While the former has been thoroughly investigated, identification has been relatively less explored. In [10] we see a summary of the performance of well-known person detection algorithms and detail the current state of the art in person detection. While person detection is still prone to a high failure rate in difficult scenes, under favorable conditions we see acceptable performance.

Face recognition is a well-understood area of computer vision (see [16]), and has been a basis for many person identification methods. Apostoloff and Zisserman propose a method to recognize faces by locating facial features [1]. While the method works well for labeling frontal faces or profile views, performance degrades otherwise. In [22] the method is extended to enable robust recognition by introducing HOG descriptors [8] to represent face appearance and tracking faces, however the method is still heavily dependent on the presence of a face. Balcan et al. introduce a semi-supervised learning method to person identification which is not heavily dependent on the presence of a face, but the method is limited to single person presence in a scene and has been tested on a dataset with ten individuals captured in over 5,000 frames [2].

While much of the person identification literature has a strong dependence on face recognition, Garg et al. propose a method which focuses on appearance and location of a person rather than facial features [13]. Face recognition is not necessarily a prerequisite for person identification, however it still enables more accuracy in results and thus we explore extension with the use of descriptive attributes. Vaquero et al. introduce a method to identify people by associating attributes to describe faces, which enables descriptive search queries and increased invariance to lighting and pose changes [25]. However, the method still primarily depends on frontal facial views for successful recognition. Our proposed technique takes advantage of an array of attributes to describe people and minimizes the requirement for particular view angles.

For grainy or distorted images, color plays a major role in accurate reidentification techniques. Bąk et al. use a combination of color normalization and covariance regions to match signatures of color arrangements [3]. Our method avoids this dependence on horizontal location (and thus the person's orientation) by slicing images vertically and binning all pixels within each slice. Hirzer et al. use this approach along with color gradient features in the LAB color space. To improve resilience to illumination changes, they include horizontal and vertical color gradients in their visual features [15]. In our own tests, we make use of the work in [18] to achieve illumination invariance.

3. Technical Approach

Our choice to use attribute description is motivated by the work in [5]. Here we discuss some person detection techniques, our part detection approach, our selected attribute recognizers, and finally our color signature implementation. Our approach consists of combining these components in series to improve reidentification accuracy.

3.1. Person Detection

People in a scene are assumed to be rigid objects for most detection techniques. While mostly rigid in a pedestrian detection model, we observe a fair degree of variation in relative pose of body parts in more involved settings (such as gatherings). To enable detection under such situations, the detection method should account for possible deformation in configuration. To account for local deformation, we therefore choose to use the deformable parts model [12]. Given an image, the detection uses a sliding window approach and a scale pyramid to meticulously search for a detection at all possible locations and scale, flagging a successful detection based on trained configurations. Once a detections have been made, we then analyze the likely regions of interest, to detect reliable rigid parts such as face, upper body and lower body.

3.2. Face and Body Parts

Body parts have been successfully and reliably detected using Haar-like features, introduced by [26], because of a higher adherence to the rigid body assumption. Apart from the rigid body assumption, body parts and face tend to have similar orientation, and hence with extensive training they can be well modeled using such features. We use a boosted cascade of Haar-like features selected specifically to detect face and other body parts such as upper body, lower body, head and shoulders. The detection method uses the sliding window technique to make an exhaustive search in scale and locations, however this search is restricted to the regions of interest provided by the person detection component. The results of face and part detection are fed into the face recognition and attribute detection components.

3.3. Face Recognition

Face recognition is probably the most investigated area in the person recognition domain. As discussed in Section 2, most methods rely on face recognition for person recognition. Since face recognition can be highly accurate under the right conditions, we place it at the top level of our recognition pipeline. However, our motivation is to perform recognition even when the face is occluded or not visible at all. We thus enable face recognition using eigenfaces [24], which is computationally less expensive compared to other competing methods, and is reliable under frontal-view conditions. The method learns a set of eigenfaces, using dimensionality reduction techniques, to discriminatively represent each training face as a weighted sum of the eigenfaces. Given a test face, the method projects the new face to the eigenspace of learned faces, and makes a recognition decision based on the closest face. It is important to our method that we be able to make a reliable rejection if the test face is a new instance, which is achieved by setting a minimum distance to make a recognition association. The method works reliably in recognizing frontal views, and therefore we use this method if we are able to detect a frontal view to make an early recognition decision. In the case of rejection, we verify the decision using attributes and color signatures.

3.4. Attribute Detection

We represent each person with a set of binary attributes to enable continued recognition even when face recognition fails. Our intuition is that a large set of attributes can enable a discriminative attribute set assignment to each person. To build attribute vectors we first extract descriptive features, which can make reliable and unique associations with each of the attributes. We use SIFT [20] and HOG[8] descriptors to encode structure and appearance variation. Apart from associations in intensity description, we also extract PHOG[4] to encode spacial information. We then concatenate the features resulting in a high dimensional feature vector encoding both spacial and local descriptions. Apart from raw feature description of a region, we also build a visual vocabulary of quantized features, to enable bag-of-words based description introduced in[23]. Our intuition is to capture similar features among samples sharing the same attribute.

Once we have a feature set, we use a support vector machine (SVM) as our discriminative function. In Section 5.3

Attribute	Location	Weight
Is Male	Whole Body	0.5624
Has Long Hair	Face	0.6308
Has Glasses	Face	0.6408
Has Hat	Face	0.7702
Has T-Shirt	Upper Body	0.6603
Has Long Sleeves	Upper Body	0.5347
Has Shorts	Lower Body	0.7265
Has Jeans	Lower Body	0.5760
Has Long Pants	Lower Body	0.6399

Table 3.1. Attribute detectors and their weights. Weights are set as the accuracy scores of our detectors, obtained during cross-validation.

we detail the performance statistics of our SVM configuration using a collection of descriptors, kernels, and parameters. Ultimately we decided on a linear kernel using only HOG descriptors. We trained our SVM using the Attributes of People dataset [5], using a one-vs-all approach to train a single SVM for each annotated attribute. Our selected attributes are listed in Table **??**.

In addition to these annotations, the dataset includes bounding boxes around the person of interest. We use these bounding boxes along with our own body part detection to isolate the critical image regions most likely to surround each particular attribute. For example, the T-Shirt SVM is trained on the output of upper body detections. In this way we focus the training of our SVMs on the body parts most relevant to them.

When registering or recognizing a person detection, we construct a binary attribute vector with each index corresponding to the listed item. Since each detector has its own accuracy rates, we use these rates as priors for the determining the probability of an attribute match.

3.5. Color Signatures

Our attribute detection system is designed primarily to boost the performance of the color signatures. Color is the most consistently available feature given a particular image. Even in small, distorted, or grainy images, color is readily available and is usually simple to identify for a human observer. The challenge here is therefore to make our color detection system focused and robust.

Given an input region of interest, we cut the ROI into k vertical slices, and then bin the detected colors into a $4 \times 4 \times$ 4-bin cubic histogram. We then define the distance between two histograms with the χ^2 distance equation, taking our multiple slices into account:

$$d(s, s') = \sum_{s \in S} \sum_{b \in B} \frac{(x_{s,b} - x'_{s,b})^2}{x_{s,b} + x'_{s,b}} \tag{1}$$

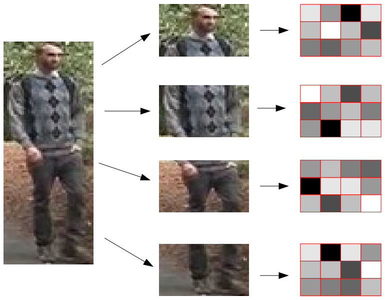

Figure 3.1. A high-level representation of a color signature extraction with 4 slices. The image is taken from the VIPeR dataset [14].

Here x and x' are two histograms, and S and B are the slices and histogram bins, respectively. To compensate for changes in illumination, we use a technique based on the work in [18]. Since we do not assume a "true" coloring (i.e. color derived from perfectly white light) in our testing data, we construct a 3-tuple of channel scales, and then apply these scales to both the training sample and the test sample when performing a comparison. We construct our scale vectors in the following manner. For the RGB color space, we scale each channel by .8, .9, and 1.0, and apply each combination of scales as a single scale vector. Thus, we perform 9 scalings and 9 full histogram comparisons for this color space. For YUV, HSL, HSV, and LAB color spaces, we only scale the luminance (or "value" in the case of HSV) color channels, again by .8, .9, and 1.0. We restrict scalings in these color spaces to isolate the channels most appropriate for changes in lighting.

Our signature for the ROI therefore consists of one histogram for each slice and for each scale. While histogram distances are combined across slices, we do not combine these across scales. Rather, we compute the distance between two such signatures s, s' in the following manner:

$$d(s, s') = \min_{\sigma \in \Sigma} \left\{ \sum_{s \in S} \sum_{b \in B} \frac{(x_{\sigma,s,b} - x'_{\sigma,s,b})^2}{x_{\sigma,s,b} + x'_{\sigma,s,b}} \right\} \tag{2}$$

Here each σ corresponds to a particular scale vector. As will be shown in Section 5.4, color histograms perform reasonably well in isolation, providing a solid foundation for improved matching.

3.6. Integrated Algorithms

Finally, we combine each component into a singular integrated algorithm. We begin with simplified subroutines, and proceed to the main `ReIdentify` algorithm. Algorithm 3.1 is the last-stop matching algorithm which compares color signatures to determine the overall outcome of a matching attempt. Given a detection and a set of candidate identities, the algorithm evaluates each identity's signature against that of the detection, and selects the identity corresponding to the least matching distance. Matching distances are based on the sum of χ^2 histogram comparisons, as outlined above.

Algorithm 3.2 constructs a set of candidate identities from a detection by comparing attribute vectors. For each identity currently registered with the system, a boolean attribute vector obtained from the binary response of each attribute's SVM is obtained, and compared with that of the detection. Individual attributes are weighted based on the accuracy of their SVM's during cross-validation, under the assumption that some attributes will be easier to detect (and thus more reliable indicators) than others. Thus, we construct a weighted Hamming distance between each identity and the detection, and those identities that fall below the threshold are returned.

Algorithm 3.3 combines all of the main components of our technique. An image is processed for person detections, and those detections are passed to the face recognition module, and then along to attribute detection and color matching.

Algorithm 3.1 Color signature matching

```
procedure ColorMatch($d, \mathcal{I}$)
    $s \leftarrow \mathrm{Signature}(d)$
    $m \leftarrow \mathrm{MAX\_DISTANCE}$
    $I \leftarrow \sim$
    for $i \in \mathcal{I}$ do
        $s' \leftarrow \mathrm{Signature}(i)$
        $\delta \leftarrow \mathrm{dist}(s, s')$
        if $\delta < m$ then
            $m \leftarrow \delta$
            $I \leftarrow i$
        end if
    end for
    return $I$
end procedure
```

Algorithm 3.2 Attribute matching

```
procedure AttributeMatches($d, p$)
    $\mathcal{I} \leftarrow \emptyset$
    $v \leftarrow \mathrm{GetAttributeVector}(d, p)$
    for $i \in \mathrm{REGISTERED\_IDENTITIES}$ do
        $v' \leftarrow \mathrm{GetAttributeVector}(i)$
        $d \leftarrow 0$
        for $a \in \mathrm{ATTRIBUTES}$ do
            if $v_a = v'_a$ then
                $d \leftarrow d - w_a$
            else
                $d \leftarrow d + w_a$
            end if
        end for
        if $d \leq \mathrm{MAX\_DISTANCE}$ then
            $\mathcal{I} \leftarrow i$
        end if
    end for
    return $\mathcal{I}$
end procedure
```

Algorithm 3.3 Main reidentification algorithm

```
procedure ReIdentify($M$)                       ▷ $M$ is the input image
    $I \leftarrow \emptyset$                              ▷ Initialize identities
    for $d \in D = \mathrm{HogDetections}(M)$ do
        $p = \mathrm{Parts}(d)$
        if $\mathrm{HasFace}(d)$ then
            $I \leftarrow (d, \mathrm{FaceIdentity}(d, p))$
        else
            $\mathcal{I} \leftarrow \mathrm{AttributeMatches}(d, p)$
            if $\mathcal{I} \neq \emptyset$ then
                $c = \mathrm{ColorMatch}(d, \mathcal{I})$
                $I \leftarrow (d, c)$
            end if
        end if
        if $d \notin I$ then
            $I \leftarrow (d, \sim)$                   ▷ Give $d$ a null identity
        end if
    end for
    return $I$                       ▷ Return one match per detection
end procedure
```

4. Software Libraries and Implementation

We implemented our software primarily using OpenCV [6]. Due to poor results with our initial selection of SVM software, we attempted the SVM implementations found in both LIBSVM [7] and Dlib-ml [17]. While we were able to achieve favorable results on simpler datasets with OpenCV, we were unable to accomplish this with either Dlib or LIBSVM. The latter two libraries expose a large number of parameters with no automatic tuning, while OpenCV's implementation was quite successful on our control sets out of the box.

OpenCV was therefore used for the vast majority of our components specific to image manipulation and machine learning. These included SVM, SIFT and HOG feature extraction, k-means and nearest neighbors indexing, color space conversion, and basic image control structures. Our PHOG implementation was taken from [**?**]. We also made use of Boost libraries [9] for core C++ functionality.

Our color histogram component was written by hand, as was all of the software for integrating our components

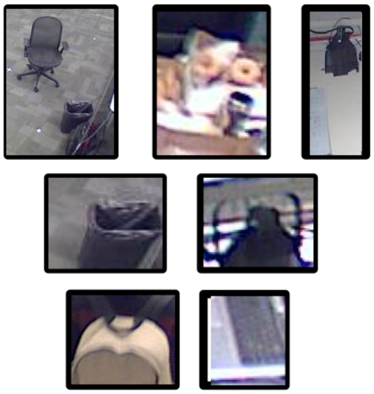

Figure 5.1. False Positives detected by the part detectors, with the first row of images showing person detections followed by Upper Body and Fontal Face detections, respectively.

with one another and enabling training and testing with our datasets.

5. Experiments

We now show a series of experiments demonstrating the effectiveness of our individual detection components, and our algorithm as a whole.

5.1. Person and Parts Detection

For person detection we use OpenCV's implementation of the deformable parts model[12]. We use the latent svm model pre-trained on the VOC 2007 dataset[11], made available by the OpenCV library. For parts detection we again use OpenCV's implementation of the boosted cascade classifier using Haar-like features[26]. We use trained models for the parts face, profile face, upper body, lower body and shoulders from [21]. We evaluate performance of both the person and parts detector on the dataset of images captured with the UT AI Lab camera network (see Table 5.1). The false positives returned by person detection are shown in Figure 5.1. Our images capture the same static scene, hence many of the false positives are detected in every frame. We therefore perform a qualitative analysis of the method performance, taking into consideration only one occurrence of the false positives in each case. False positives of this nature can be eliminated, given the camera configuration, and likely person presence. We thus do not quote precision of the detector. We indicate similar results for the part detectors in Table 5.1, and a few false positives shown in Figure 5.1.

5.2. Face Recognition

We use the eigenfaces method implementation made available by OpenCV. We evaluate the method on the Yale Face Dataset[19]. The dataset has 15 unique faces with 11 instances each, captured as frontal views under different illumination and facial expressions. We first evaluate the distance threshold to enable the best true rejection performance. For this we train the recognizer with 8 instances each of 14 unique faces, and then test recognition performance with the remaining 3 images of each trained person, with 11 instances of the person left out of the training method. Figure 5.2 plots precision measure as the distance threshold is increased; note the precision reducing after a maximum, indicating recognition of the unknown set. We observe that setting a low threshold results in fewer recognition results, but with almost 100% accuracy. This is ideal for our implementation, since we do not entirely rely on face recognition.

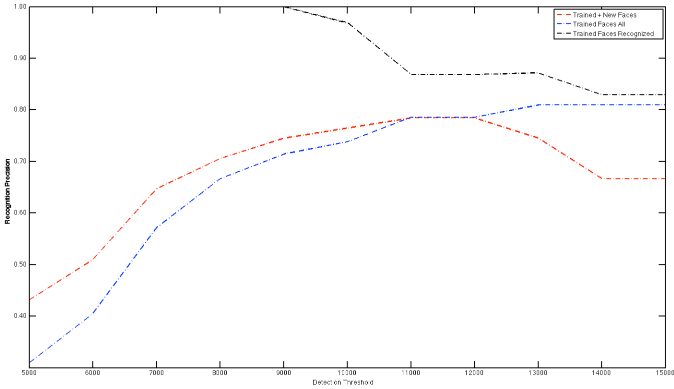

Figure 5.2. The accuracy curve as the distance threshold of the recognition method is varied from 5000 to 15000. All Images curve includes the unknown class of images, note the peak in the precision measure, where most of the unknown class of images are rejected, but increasing the distance threshold results is a fall in the accuracy attributed recognition of images in the unknown set. The Trained Faces All curve plots precision results with the test faces having a training instance, but penalizes rejection. Trained Faces Recognized curve plots precision results without penalizing rejection, all test faces have a training instance - note the fall in precision with increasing threshold.

Under our method formulation, we often have instances of very few examples to train a new identity. We therefore also evaluate recognition performance when trained with one unique instance from each class and tested over the rest, and perform a similar experiment with two unique training instances from each class. Figure 5.3 shows the confusion matrix when one training instance is used per class, resulting in an average accuracy measure of 58%, and Figure 5.4 shows the confusion matrix when two training instances are used per class, resulting in an average accuracy measure of 69%. We can, as indicated earlier, reduce the distance threshold to improve accuracy measure while compromising on recall.

Detection	Total	TP	FP
Person	968	569	7
Frontal Face	138	83	55
Upper Body	868	260	23

Table 5.1. Total detections on the image dataset captured on the UT AI Lab Camera Network. The total number of detections do not equal the sum of TP and FP, because we list only one instance of a false negative which is part of the static scene, examples of which are indicated in the figure 5.1

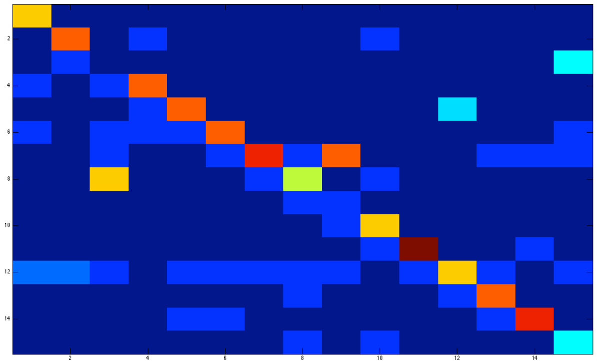

Figure 5.3. Confusion Matrix normalized along the column, for face recognition on ten images per class with one training instance.

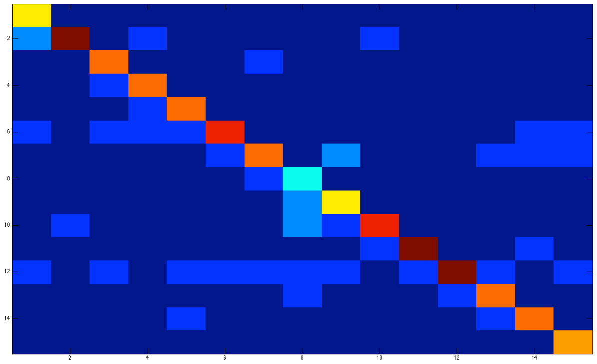

Figure 5.4. Confusion Matrix normalized along the column, for face recognition on nine images per class with two training instances.

5.3. Attribute Detection

Attribute detection provides a large number of possibilities for parameter tuning. While we experimented with adjusting SVM parameters such as ν (for NU-SVM) and C (for C-SVM), we ultimately relied on OpenCV's `CvSVM::train_auto` function to select these parameters for us based on our provided training data. We therefore focused on the parameters in Table **??**.

SVM Kernels	Linear Degree-3 Polynomial (Poly) Radial Basis Function (RBF)
Descriptors	Standard HOG Pyramidal HOG (PHOG) Dense SIFT

Table 5.2. SVM parameters tuned for our implementation.

For brevity, we present the best and worst configurations we identified. The results are given in Table 5.2. It is important to note the fact that precision, recall, and accuracy rates can be deceptive depending on the dataset, and although the RBF kernel tended to give favorable results by these measures, in fact the SVM often ended up highly biased toward positives or negatives. Considering these factors, we found the optimal configuration to be HOG descriptors with a linear kernel. We use the accuracy values output for this configuration for our final attribute detector weights (listed in Table **??**).

5.4. Color Histograms

To show the effectiveness of the base color histogram approach, we construct a series of ROC and rank curves. Figure 5.5 shows the rank curves generated without illumination correction and with 4 vertical image slices, for each color space. Contrary to expectations, RGB is comparable with the other color spaces regardless its lack of an illumination channel. It's possible that these similarities are due to the VIPeR dataset, and that under other conditions the illumination-based color spaces may outperform RGB. Additionally, it is important to note that while rankings are informative when one assumes a known sample set, ROC curves give a better indication of how an algorithm will perform when the samples are not known, i.e. in a surveillance situation. In such a scenario, one must determine beforehand the proper distance threshold for discriminating a match from a non-match, and apply that threshold to all signatures.

Figure 5.6 shows the ROC curves for the same color signature configurations, with the independent variable being this distance threshold. In the context of surveillance, false negatives can be compensated for with the use of statistical tracking methods (such as a Kalman filter) and multiple

Descriptors	Kernel	Class	Accuracy	TP	FP	TN	FN
SIFT	RBF	Has Hat	0.83	7	17	534	2,692
PHOG	RBF	Has Long Pants	0.74	988	334	0	0
SIFT	Poly	Has Hat	0.74	165	479	2,230	376
HOG	Linear	Has Hat	0.77	159	365	2,344	382

Table 5.3. Accuracy and True/False Positive/Negative count on the Attributes of People Dataset (consisting of approximately 4,000 test images) for different detector configurations, with their best classes selected by accuracy. Judging entirely by accuracy rates, the RBF kernel appears competitive in some scenarios, however it routinely outputs entirely positives or entirely negatives for each class. This is evident when examining the TP and TN counts.

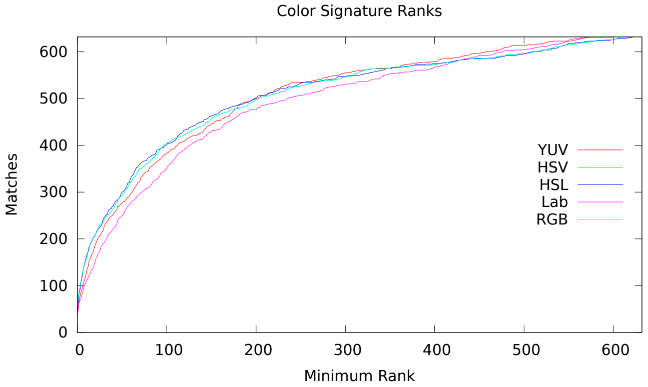

Figure 5.5. The rank curves for each color space, with no illumination correction and 4 slices. The rankings are performed on all 632 samples from the VIPeR dataset [14].

Configuration	AUROC
RGB, 5 Slices, Illumination Corrected	0.5712
YUV, 5 Slices, Illumination Corrected	0.6287
HSL, 4 Slices	0.6309
HSV, 4 Slices	0.6325
YUV, 3 Slices	0.6329
YUV, 4 Slices	**0.6398**

Table 5.4. Area under ROC curves for the 5 top-performing and 1 worst-performing color signature configurations.

video frames. False positives, however, can lead estimates astray. We therefore would prefer a colorspace with higher values on the left end of the ROC curve. In this figure we see YUV come out as a clear winner. As Table 5.3 shows, the advantage is quantifiable. We found that the illumination correction method from [18] did not quantifiably improve results, and thus this was left out of our final implementation. Ultimately we selected non-corrected, 4-slice YUV for our color signature configuration.

5.5. Integrated Results

For our final tests, we evaluate the performance of the system as a whole on the AI Lab dataset (Figure 5.7), as well as a dataset of images captured with the VIPeR dataset (Figure 5.8). Due to the characteristics of our datasets, we omit part detection and facial recognition from the implementation. As we will describe in Section **??**, our goal is to obtain datasets that are better suited to these components.

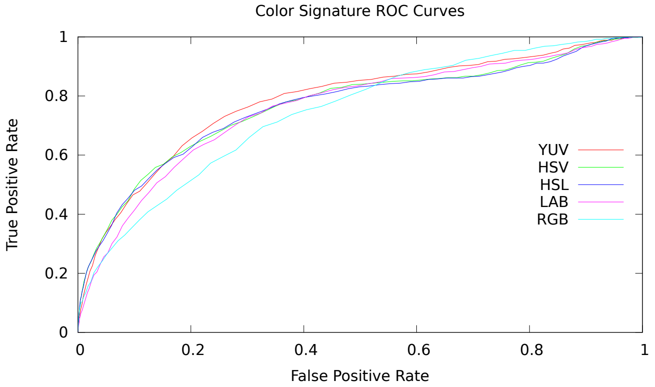

Figure 5.6. The ROC curves for each color space, with no illumination correction and 4 slices. The ROC curves are computed by counting true positives (i.e. matches) and false positives (non-matches) over the 632 samples from the VIPeR dataset [14]. The independent variable is the distance threshold. Here we see all color spaces performing about equally, except with RGB lagging behind the rest on the left half of the graph.

Though the lab dataset is smaller, it shows better match rates and a smaller distinction between the two matching methods when compared to the VIPeR dataset. Match rates for color only are very close for 10 samples between the two datasets (50% for Lab and 40% for VIPeR), but there is a marked improvement in the performance of the attribute detector on the Lab dataset. This is likely due to the fact that the Lab images are unscaled and of higher resolution than the VIPeR images, enabling better detection of attributes. Of course, it is clear that the inclusion of attribute information is detrimental to the reidentification process overall. This indicates that the attribute detections are incorrect more often than not, and that these detectors could benefit from more relevant and thorough training.

[15] provides results on the VIPeR dataset as well, and although our algorithm doesn't produce a ranked list of matches, we can compare our success rate at 30 samples with the results from [15] at rank 0. While we obtain an

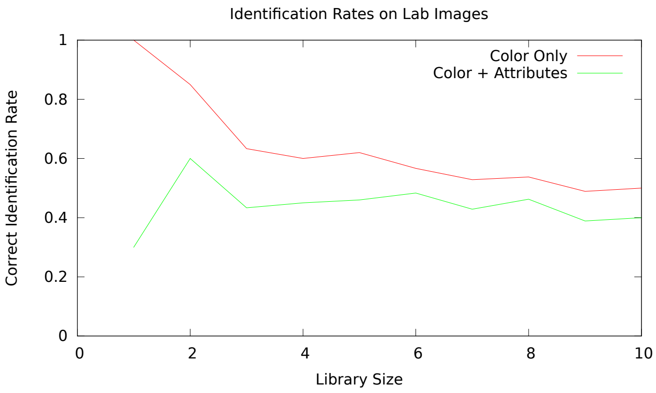

Figure 5.7. A graph of successful match rates on the AI Lab dataset.

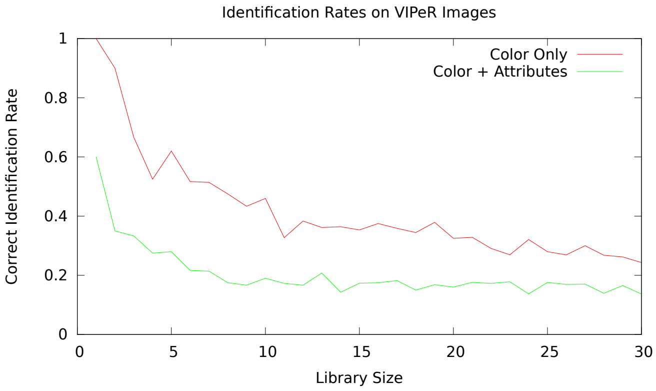

Figure 5.8. A graph of successful match rates on the VIPeR dataset.

accuracy rate of 24%, Hirzer *et al.* achieve approximately 47%.

6. Conclusions

Our implementation performed reasonably well with an initial color-based matching implementation, however the poor results when including attribute data show a need for improvement in the attribute detection module. After careful analysis of the SVM implementation and the training data, it seems that our choice of training dataset would have had a significant impact on this, and would be a good place to apply such initial improvements.

While methods such as that described in [15] show greater accuracy rates on VIPeR, our results are still promising in that they could potentially perform much better with more focused attribute training and part detection, improved facial recognition, and through taking advantage of the our target environment. At this point, it remains to be seen whether our proposed method has the potential to be competitive with the state of the art in person reidentification.

7. Future Work

Person re-indentification is in its infancy with the state-of-art lagging compared to object and face recognition. The potential for improvement is demonstrated with the non-exhaustive directions for further research in the following sub sections.

7.1. Training Data

Our experience with training the attribute detectors exposed the lack of availability of a well annotated attribute data set of people, which was one of the primary factors for the low accuracy rates on the detectors. Though the attribute detectors indicate satisfactory accuracies, they do not generalize well when applied to other datasets. This is in part due to the fact that most of the persons of interest in the attributes of people dataset are partially occluded. We also stress the non-availability of datasets to build recognition dictionaries as envisioned in this paper to recognize registered instances of people. We will compile a new dataset with annotated bounding boxes of each body part and a include a more comprehensive set of attributes. We will also compile the dataset to include numerous repeated instances, as is the case of many available face recognition datasets.

7.2. Sliding Windows

Many distinct attributes of people have a relative spatial distribution having small variance given the person bounding box. We expect improvement in attribute detection precision when trained on specific parts. We will extend out work in this direction in the short term by training attribute detectors by automatic detection of parts and in the long term have them trained on a well annotated dataset.

7.3. Probabilistic Estimation

In this paper we highlight the use of the accuracy of individual attribute detectors as weights while estimating similarity values between attribute vectors. Work from the field of information theory and coding would be better suited to model error in our attribute vectors by interpreting as error-prone bit sequences. Bayesian analysis would be another approach to interpret the true positive and true negative rates as indicators to the reliability of attribute vector detections.

7.4. Relative Coloring

Illumination changes are a classic problem for color-based vision techniques. The paradox here is that what humans perceive as consistent color is in fact a highly variable array dependent upon environmental factors such as texture and ambient lighting. In our experiments, illumination correction by varying luminance channels offers no substantial benefit when the “true” color is unknown, as may be the case in a variety of real-world situations. A possible remedy for this lies in the use of relative coloring, in which pixels values are represented by their respective deviations from the mean rather than in absolute terms. Using a relative representation could provide invariance against global changes in illumination. Such a technique may not prevent issues due to shadowing or texture, however.

7.5. Spatial-Temporal Context

Key pieces of information were ignored in our approach, namely the cues granted by the spatial-temporal context of our distributed cameras and detections. In a complete implementation, this information could prove extremely valuable, as multiple frames of images can provide robustness against a variety of detection and recognition errors. For instance, a Kalman filter model used for tracking a person could be associated with that person's identity, so multiple recognition attempts might be considered on a single tracked person rather than making one attempt for a single frame. Additionally, proximity of networked cameras can provide information on likely identities in a particular scene. A camera is much more likely to see people who were recently present in nearby camera scenes, and this information could be supplied to the recognition system.

7.6. Real-Time Processing

The ultimate goal of this work is to produce a real-time person recognition system, and while processing time was largely ignored in the experiments, this would be a critical component of any deployed system. Once an optimal solution is found for the reidentification problem, it is likely that the solution would need to be optimized or have its complexity reduced in ways that minimize hindrance to recognition capability.

References

[1] N. Apostoloff and A. Zisserman. Who are you? - real-time person identification. In *Proceedings of the British Machine Vision Conference*, pages 48.1–48.10. BMVA Press, 2007. doi:10.5244/C.21.48.

[2] M.-F. Balcan, A. Blum, P. P. Choi, J. Lafferty, B. P. andMugizi R. Rwebangira, and X. Zhu. Person identification in webcam images: An application of semi-supervised learning. 2005.

[3] S. B?k, E. Corvee, F. Bremond, and M. Thonnat. Person re-identification using spatial covariance regions of human body parts. In *Proceedings of the 2010 7th IEEE International Conference on Advanced Video and Signal Based Surveillance*, AVSS '10, pages 435–440, Washington, DC, USA, 2010. IEEE Computer Society.

[4] A. Bosch, A. Zisserman, and X. Munoz. Representing shape with a spatial pyramid kernel. In *Proceedings of the ACM International Conference on Image and Video Retrieval*, 2007.

[5] Bourdev, Lubomir, Maji, Subhransu, Malik, and Jitendra. Describing people: A poselet-based approach to attribute classification. In *Proceedings of the 2011 International Conference on Computer Vision*, ICCV '11, pages 1543–1550, Washington, DC, USA, 2011. IEEE Computer Society.

[6] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[7] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[8] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01*, CVPR '05, pages 886–893, Washington, DC, USA, 2005. IEEE Computer Society.

[9] B. Dawes and D. Abrahams. Boost c++ libraries. `http://www.boost.org/`.

[10] P. Dollar, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: An evaluation of the state of the art. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(4):743–761, Apr. 2012.

[11] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.

[12] P. Felzenszwalb, D. Mcallester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, June 2008.

[13] R. Garg, S. M. Seitz, D. Ramanan, and N. Snavely. Where's waldo: Matching people in images of crowds. In *CVPR*, pages 1793–1800. IEEE, 2011.

[14] D. Gray, S. Brennan, and H. Tao. Evaluating appearance models for recognition, reacquisition, and tracking. In *10th IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS)*, 09/2007 2007.

[15] M. Hirzer, C. Beleznai, P. M. Roth, and H. Bischof. Person re-identification by descriptive and discriminative classification. In *Proceedings of the 17th Scandinavian conference on Image analysis*, SCIA'11, pages 91–102, Berlin, Heidelberg, 2011. Springer-Verlag.

[16] R. Jafri and H. R. Arabnia. A survey of face recognition techniques. *JIPS*, 5(2):41–68, 2009.

[17] D. E. King. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10:1755–1758, 2009.

[18] J. Lee. *Robust Color-based Vision for Mobile Robots*. dissertation, The University of Texas at Austin, 2011.

[19] K. Lee, J. Ho, and D. Kriegman. Acquiring linear subspaces for face recognition under variable lighting. *IEEE Trans. Pattern Anal. Mach. Intelligence*, 27(5):684–698, 2005.

[20] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, Nov. 2004.

[21] M. C. Santana, O. Dniz-Surez, L. Antn-Canals, and J. Lorenzo-Navarro. Face and facial feature detection evaluation - performance evaluation of public domain haar de-

tectors for face and facial feature detection. In A. Ranchordas and H. Arajo, editors, *VISAPP (2)*, pages 167–172. INSTICC - Institute for Systems and Technologies of Information, Control and Communication, 2008.

[22] J. Sivic, M. Everingham, and A. Zisserman. ”who are you?” - learning person specific classifiers from video. In *CVPR*, pages 1145–1152. IEEE, 2009.

[23] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *Proceedings of the International Conference on Computer Vision*, volume 2, pages 1470–1477, Oct. 2003.

[24] M. Turk and A. Pentland. Eigenfaces for recognition. *J. Cognitive Neuroscience*, 3(1):71–86, 1991.

[25] D. Vaquero, R. Feris, D. Tran, L. Brown, A. Hampapur, and M. Turk. Attribute-based people search in surveillance environments. In *IEEE Workshop on Applications of Computer Vision (WACV'09)*, Snowbird, Utah, December 2009.

[26] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Hawaii, 2001.