POD: Colour detection of people and objects for multi-camera video tracking

Mathew Price, Fred Nicolls, Gerhard de Jager

Department of Electrical Engineering
University of Cape Town
Rondebosch 7701, South Africa
mathew@dip.ee.uct.ac.za

Abstract

Visual tracking of humans has proved to be an extremely challenging task for computer vision systems. Often the colour appearance of a person can provide enough information to identify an object or person in the short-term. However, the imprecise nature of colour measurements typically encountered in image processing has limited their use. This paper presents a system which uses the colour appearances of objects and people for tracking across multiple camera views in a video surveillance network. Tracking has been approached from a classification standpoint allowing the system to cope with multiple occlusions, variable camera pose, and asynchronous video feeds. Results indicate that the system could be used by 3-D tracking methods to recover from failed tracking.

1. Introduction

Object tracking has drawn a large interest in Computer Vision research due to the wide range of applications it encompasses. One such application applies to the area of computer-aided surveillance. Large buildings regularly use CCTV networks to monitor the movements of people in order to provide security and safety. This leads to a bottleneck of information since an operator can only manage a finite number of camera views accurately at one time. The idea of computer-aided surveillance fits into the intermediatory role of filtering out uninteresting events and drawing the operators' attention to items of specific importance (eg. people accessing restricted areas). A basic requirement of such a system, therefore, is the ability to keep track of several people simultaneously, especially their transitions between different camera views.

One interesting problem that arises is the issue of tracker initialisation. Many predictive tracking techniques have been developed. However, a common weakness in many of these systems is their dependence on parameter initialisation. Since a prediction is based on a current assumption and a noisy observation, errors are cumulative and can lead to eventual target loss. Recovery then requires that the tracker be re-initialised to the target's current position. The framework of tracking systems often does not allow for this.

The system proposed here approaches the tracking problem from a classification standpoint. Often the colour appearance of a person can provide enough information to determine their identity and thus their position in the short-term. Therefore, by modelling this colour appearance as a set of features, colour matching can be used to build a likelihood response to the model's spatial position in an arbitrary image. This one-shot style of tracking has the advantage of being able to deal with occluding objects, movement between multiple camera views and asynchronous video feeds.

2. System Overview

The primary goal of the system is to provide a suitable framework for person/object matching that can be extended over a network of cameras. Since video processing is extremely resource-hungry, a distributed computing, bottom-up paradigm was chosen as the basis for this framework.

The basic task of the system is as follows. A camera provides input frames to the Person/Object Detector (POD). Previously trained colour object models are then used in a matching process which identifies the location (if any) of the target in the input frame.

2.1. Data Flow Model

The system comprises several pre-processing steps which feed into the matching and training processes. A local object model repository provides the stored features of current targets which are used during matching. Figure 1 depicts these operational stages for a single processing node.

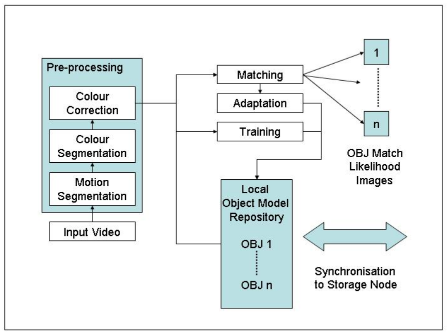


Figure 1: *Processing Node Data Flow Diagram.*

The first stage is the digital acquisition of video from a camera. Following this, three pre-processing steps are used to extract colour features: Motion Segmentation; Colour Segmentation; and Colour Correction.

Motion segmentation is used to separate active people and objects from the static background, thus reducing the clutter significantly. Colour segmentation then groups similar pixel regions together, thus producing a compressed representation of the motion segmentation. This allows concise models to be constructed and greatly reduces the computational cost of matching.

When viewing an object with different cameras, it is unlikely that it will appear identical. Factors such as camera gain, shutter speed and gamma correction can contribute to large variances in image formation. Additionally, since each camera will most likely be positioned differently (probably in different rooms), environmental lighting conditions will also cause inconsistency between views. Since the system must be able to compare a person or object's colour between views, a method of calibrating all cameras to a common colour reference is necessary. Towards this end, the colour correction stage uses trained samples of a camera's response to a range of colours in order to calculate an appropriate adjustment.

The features produced by the pre-processing stages are represented by a cluster of colour centres. These centres are then fed to the detection system. Initially, since the object model repository will be empty, the training stage will use the features of a specific target to discover the best representation for that target. This trained feature set is then added to the object model repository.

Once trained object models are available, the matching process takes over. Matching works by dividing the image into horizontal and vertical strips. The strips, representing spatial positions, are then scanned 1-dimensionally and a series of feature measurements are generated for each target object. The measurements, which comprise:

- Quality of the colour match
- Variety of model features matched
- Spatial density and size of the features in image space
- Consistence of area proportionality,

are then combined to form an overall confidence measurement for each target. Finally, combining the peak likelihoods with the input image produces a labelled image that identifies the positions of any visible targets.

3. Segmentation

A simple two-part segmentation scheme is used for reducing clutter and extracting colour features from objects. Firstly, *motion segmentation* by way of background subtraction removes target candidates from the static background scene. The next step involves a *colour segmentation* process whereby regions of similar colour in the foreground are grouped together and represented by a cluster of colour centres. This provides a concise object description system capable of being synchronised over a network.

3.1. Motion Segmentation

Conventionally, background subtraction operates by subtracting frames from an initial static background image. Any pixel difference which deviates more than a set threshold is then considered foreground. While this method is fast, using a global threshold generally produces a very noisy segmentation. An alternative is to assume that each pixel deviates according to its own model, and thus to threshold each pixel in the context of its model [13].

3.2. Colour Segmentation

A batch colour segmentation method allows an efficient quantisation of the foreground into coloured components, leading to a compressed data representation which is suited to high-level processing.

Multiscale image processing techniques are hierarchical processes which use multiple resolutions of an image to perform analysis. These can be approached as top-down (quad-tree decomposition) or bottom-up processes (image pyramids). In the latter methods, image levels are constructed by downsampling the image by a series of filters — Gaussian in this case — which provides a smoother output.

The OpenCV library [6] provides a well optimised implementation based on the algorithm proposed in [3]. The input image (largest) can be thought of as the base of the pyramid. Each consecutive level is then built upwards, downsampling by a factor of two at each stage, until a specified maximum level is reached (commonly between 3 and 5). Two thresholds, T_1 and T_2, determine the nature of the segmentation, and linking is performed in two stages:

1. A link between a pixel $p_{x,y}$ on level L and its candidate father $p'_{x',y'}$ on level $L + 1$ is established if: $\text{dist}(c(p_{x,y}, L), c(p'_{x',y'}, L + 1)) \leq T_1$.
2. Connected components A and B are then clustered together if: $\text{dist}(c(A), c(B)) \leq T_2$.

The function $dist$ is the Euclidean distance between local property functions c for each pixel. The local property c is the type of measurement associated with a pixel, which in this case is simply the intensity of the pixel. Therefore, the distance calculated by the $dist$ function is effectively the difference in pixel intensities $|i(x, y) - i(x', y')|$.

Parameter tuning, which is dealt with in [7], was found to be fairly inconsequential. Unlike [7], the primary goal here is to reduce the mass of pixel data to a manageable number of classes (approximately 5% of the image pixels). Therefore exact parameters for maximising the quantisation are not required. In general, setting $T_1 = 30$ and $T_2 = 10$ provided equitable results.

4. Colour Correction

One of the primary requirements of a multi-camera colour tracking system is consistency of colour measurements between several cameras. Colour constancy refers to the correction of colour deviations caused by a difference in illumination.

In light of the fact that our system is geared towards near real-time operation, it follows that a fast, robust colour constancy method is needed. Since we are using multiple cameras, the method must be insensitive to camera pose and position in addition to the usual invariants.

Austermeier et al. [1] have shown a useful method for performing an unsupervised, target-based calibration scheme for normalising illumination changes. Their tests showed that a cloud of RGB pixels (plotted by omitting their spatial image placement) preserves its topology when subjected to a change in illumination. Furthermore, if the clouds of the original and resulting images are each quantised by a set of Self-Organising Map (SOM) prototypes, pixel colour can be corrected by simply translating its prototype between the two maps. SOMs have the useful feature of being able to quantise data into a set of prototypes, while at the same time preserving topological relationships between neighbouring neurons.

To clarify, usually the main usage of SOMs is for dimension reduction of feature data. However, in this application it is simply used as a 3-D data-fitting method.

5. Training Object Models

A primary assumption upon which the system is based is the notion that a qualitative measure for the difference between colours exists. The CIELAB colour space provides such a measure by providing uniform colour co-ordinates which describe perceptual colour differences using the magnitude of the Euclidean distance between two points. The training procedure therefore begins by converting the RGB feature list presented by the colour segmentation to their CIE L*a*b* counterparts. The system's main feature vector is therefore simply:

$$\mathbf{F}_n = (L*_n, a*_n, b*_n), \tag{1}$$

where $\mathbf{F}_n$ is an arbitrary feature vector. Training is split into two phases. The first is geared towards finding clusters spatially within the feature space of a presented observation set. The second clusters these cluster groups over time as additional observations are presented. The time-clustering phase therefore depends on being able to match features between between observations. This is accomplished by measuring the conditional probability $\mathbf{P}(\mathbf{F}_n|\mathbf{C})$ of a feature $\mathbf{F}_n$ belonging to the set of centres $\mathbf{C}$ which is calculated as follows:

$$d^2_{\mathbf{F}_n\mathbf{C}_i} = (L*_n - L*_i)^2 + (a*_n - a*_i)^2 + (b*_n - b*_i)^2 \tag{2}$$

$$K_{\mathbf{F}_n\mathbf{C}_i} = \frac{1}{\sqrt{2\pi\sigma^2_{train}}} e^{\left(\frac{-d^2_{\mathbf{F}_n\mathbf{C}_i}}{2\sigma^2_{train}}\right)} \tag{3}$$

$$\mathbf{P}(\mathbf{F}_n|\mathbf{C}) = \frac{\mathbf{K}_{\mathbf{F}_n\mathbf{C}}}{\sum_{j=1}^{m} K_{\mathbf{F}_n\mathbf{C}_j}}. \tag{4}$$

These are simply the spherical Gaussian kernel activations $K_{\mathbf{F}_n\mathbf{C}}$ normalised by the sum of activations of all current centres $\mathbf{C}_1 \ldots \mathbf{C}_m$. Since CIELAB offers uniformity, it follows that σ_{train} should be set to a constant value so that perceptual colour differences remain the same between classes. A $\sigma_{train} \approx 5$ has been found to provide good separation between colour classes.

Since similar colours will cluster uniformly in the feature space (due to the intrinsic nature of the CIELAB space), groups of like features can be represented by a Gaussian centre. Training thus involves finding the best possible group of centres which accurately quantises the input training set — i.e. a Gaussian Mixture Model.

6. Matching

Matching consists of evaluating a series of feature measurements in an image for each trained object. These measurements are then combined to produce a classification likelihood of an image region for a particular object model.

The process begins by performing colour matching on the set of input features **F** produced by pyramid segmentation. This basically assigns each feature to the nearest model centres in the repository **C** (a $m \times 3$ matrix). The result is a list of active model centres **X** and their centroids in image co-ordinates. Since it is likely that certain colours will match a variety of different objects, several measurements are evaluated in order to maintain class separability. If several of these measures agree, a peak in an object's likelihood will appear for a certain image region. If the overall confidence exceeds a lower threshold, the object is marked as found.

6.1. Colour Matching

As with the training procedure, colour matching is done using the CIELAB distances between the extracted features **F** and object model centres **C**. Equation 5 defines the Euclidean distance for two features (as in Equation 2). $\mathbf{X}$ is then defined as the matched subset of features, which are less than $k_{match}\sigma_{match}$ Euclidean units from any of the m model centres in $\mathbf{C}$:

$$D^2(\mathbf{F1}, \mathbf{F2}) = (F1_{L*} - F2_{L*})^2 + (F1_{a*} - F2_{a*})^2 + (F1_{b*} - F2_{b*})^2 \tag{5}$$

$$\mathbf{X} = \{\mathbf{x} \in \mathbf{F} : D^2(\mathbf{x}, \mathbf{C}_i) \le (k_{match}\sigma_{match})^2, \text{ for } 1 \le i \le m\}. \tag{6}$$

Colour matching is thus effectively a nearest neighbour classification.

6.2. Confidence Measurement

Estimation of the confidence measurements across a 2-D image plane requires evaluation of the contribution of each matched feature for each measurement of every object class. The fact that each object feature is only represented by a central pixel dictates that a sliding window operation is needed. Unfortunately this would result in an exceptionally high computational complexity since the convolution would need to be repeated for each object class. While this process can be improved using FFT fast convolution, the computational time is still proportional to the number of measurements and classes.

A less precise (yet efficient) idea is to perform measurement using a 1-D scanning algorithm which can be executed separately across the image's x and y directions. There is a possibility that a better approach might be to evaluate image quadrants [14] and use fast integral image convolution with boxlets [11]. However, separate class processing would still be required and so this alternative is left for future exploration.

Scanning proceeds by dividing the image into several evenly spaced, vertical and horizontal strips as shown in Figure 2(a). The matched features $\mathbf{X}'$ falling within a strip d_i then contribute to some property measurement $z(d_i)$ for each object model. When the measurements of all strips are concatenated, the results are two 1-dimensional likelihood signals $(\mathbf{z}_x, \mathbf{z}_y)$ spanning the width w_{im} and height h_{im} of the image respectively. Each signal is then filtered with a Gaussian kernel to smooth the disparity between the divisors (Parzen's method). Finally, the matrix multiplication of each object model's $\mathbf{z}_x$ and $\mathbf{z}_y$ vectors produces a 2-D likelihood map $\mathbf{L}_r$ for that object.

This method allows the confidence measurements to be tailored to specific areas for each image dimension. For instance, the proportionality measure holds little significance for person models in the horizontal direction since clothing divisions tend to appear vertical. As each measurement vector is one dimensional, multiple object models can be measured and stored in separate columns simultaneously. The result is an array of multi-model confidence measures created by a one-pass scan of the input image.

The following subsections describe each component of the overall confidence measurement. In order for the measurements to be combined equally, each measurement is configured to fit the range of $(0, 1)$ where 1 is the best match. Figure 2 illustrates matching for a character from a cartoon sequence.

6.3. Likelihood Map

The 2-D likelihood map $\mathbf{L}_r$ for each object r is generated by the matrix multiplication of the smoothed[1], overall confidence measurements $\mathbf{z}_x$ and $\mathbf{z}_y$ (Equation 9). These measurement vectors

[1] Smoothing is achieved using a Parzen window.

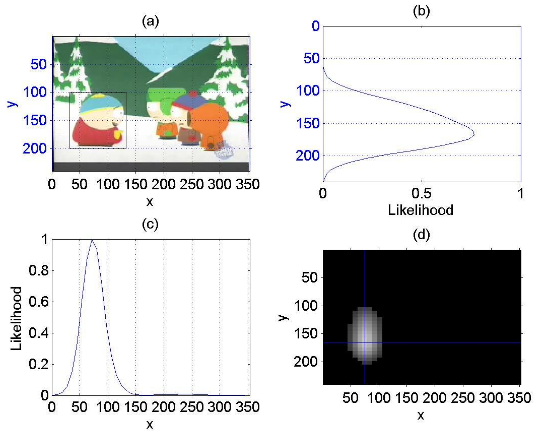


Figure 2: *Matching Example. Image (a) shows the image divisions and identifies the target (black box); Graphs (b) and (c) show the 1-D measurement signals in the y and x directions respectively; and image (d) shows the likelihood map.*

are constructed by the scalar multiplication of four measurement components:

$$\mathbf{z}_x = \mathbf{z}_{c_x}.\mathbf{z}_{v_x}.\mathbf{z}_{a_x}.\mathbf{z}_{p_x} \quad (7)$$

$$\mathbf{z}_y = \mathbf{z}_{c_y}.\mathbf{z}_{v_y}.\mathbf{z}_{a_y}.\mathbf{z}_{p_y} \quad (8)$$

$$\mathbf{L} = z_0(\mathbf{z}_x\mathbf{z}_y^{\mathrm{T}}), \quad (9)$$

where z_0 is a scaling factor and $\mathbf{z}_c, \mathbf{z}_v, \mathbf{z}_a, \mathbf{z}_p$ are the measurement components relating to quality, variety, area and proportionality respectively. These are discussed in detail later in this section. In addition each component is the concatenated vector of the measurements for all divisions. For instance, if there are i divisions, $\mathbf{z}_{c_x}$ would consist of:

$$\mathbf{z}_{c_x} = [\mathbf{z}_{c_x}(d_1), \mathbf{z}_{c_x}(d_2), \ldots, \mathbf{z}_{c_x}(d_i)]. \quad (10)$$

Subsequently, $\mathbf{L}$ would then be an $(i \times i)$ map, similar to the example in Figure 2(d)[2].

Naturally, the number of divisions i does not have to be the same for each direction. In fact, for person tracking it can sometimes be better to allocate larger division spaces in the y direction since people are more rectangular. Note, however, that the actual division size in pixels is dependent on the size of the image dimension. Since most images are not square, this means that allocating the same number of divisions for each image dimension will not necessarily result in square likelihood regions. Generally, retaining the aspect ratio of the image is desirable, so using equal divisions for each dimension can be useful.

The effect of adjusting the number of divisions relates proportionally to the output resolution of the likelihood map. Basically, it defines the minimum detectable object size. A small value will tend to expect large objects and cause merging between object classes within close proximity. Conversely, a large division number will give a high-definition likelihood, but can cause object fragmentation.

[2]The likelihood map in the figure has been resized to be consistent with the image co-ordinates.

6.4. Quality of colour match

The first measurement component quantifies the quality of the average colour match $z_c(d_i)$ between $(n \times 3)$ feature subset $\mathbf{X}'$ (the matched features falling within d_i) and its corresponding matched object model centres $\mathbf{X}'\mathbf{C}$ (i.e. $\mathbf{X}'$ and $\mathbf{X}'\mathbf{C}$ are the same size):

$$z_c(d_i) = \frac{1}{n}\sum_{j=1}^{n} e^{\left(\frac{-D^2(\mathbf{X}'_j, \mathbf{X}'\mathbf{C}_j)}{2\sigma^2_{match}}\right)}, \quad (11)$$

where D^2 is the Euclidean distance function defined in Equation 5.

6.5. Variety

If $\mathbf{X}'$ is the subset of matched features $\mathbf{X}$ falling within division d_i, then $\mathbf{h}_{d_i}(\mathbf{X}')$ is the histogram of feature areas for each object model centre in $(m \times 3)$ matrix $\mathbf{C}$ (within that division). The variety measure $z_v(d_i)$ is defined as:

$$\mathbf{v}_{d_i}(\mathbf{X}') = \begin{cases} 1 & \text{for } \mathbf{h}_{d_i}(\mathbf{X}') > h_{thresh} \\ 0 & \text{otherwise} \end{cases} \quad (12)$$

$$z_v(d_i) = \frac{1}{m}\sum_{j=1}^{m} \mathbf{v}_{d_i}(\mathbf{X}')_j. \quad (13)$$

The threshold h_{thresh} determines how many hits a bin requires before it qualifies for measurement (generally set to 1). Effectively the variety measure determines what fraction of the object model's centres is visible for each division. This relates to how much of a model is visible. A high variety will be detected where most of the object centres overlap for a particular x value.

6.6. Area Distribution

The distribution of feature areas can also hold vital information about the whereabouts of an object. Once again, $\mathbf{X}'$ is the n feature subset of $\mathbf{X}$ falling within d_i, and the area distribution $z_a(d_i)$ is:

$$z_a(d_i) = \frac{1}{A_{max}}\sum_{j=1}^{n} \text{area}(\mathbf{X}'_j), \quad (14)$$

where A_{max} is the maximum feature area throughout the image. This measurement serves to identify the division that holds the greatest area of matched pixels.

6.7. Proportionality

Proportionality refers to the ratio of the mixture of colour features for a particular object model. Often, several background regions can match a particular object's colours (seen in previous measurements), however the true object can be isolated by analysis of the proportions of these colours. The proportionality measurement is defined by the Chi-Square distance (Equation 17) between the area histograms $\mathbf{h}_{d_i}(\mathbf{X}')$ (from Equation 12) and $\mathbf{h}_{d_i}(\mathbf{C})$ (areas of object model centres) within division d_i. The histograms each have m bins which relates to the number of model centres for the specific object and are each normalised by their total sum. The Chi-Square distance provides a comparative metric between distributions and maps the interval $(-\infty, \infty)$ to $(0, 1)$ (where 0 is a close match). To make the values consistent with the other measurements (0 — no match, 1 — best match), the Chi-Square distance is subtracted from 1. Proportionality measurements $z_p(d_i)$ therefore fall within the $(0, 1)$ range where 1 is the closest possible match (Equation 18):

$$\mathbf{h}'_{d_i}(\mathbf{X}') = \frac{\mathbf{h}_{d_i}(\mathbf{X}')}{\sum_{j=1}^{m} \mathbf{h}_{d_i}(\mathbf{X}')_j} \quad (15)$$

$$\mathbf{h}'_{d_i}(\mathbf{C}) = \frac{\mathbf{h}_{d_i}(\mathbf{C})}{\sum_{j=1}^{m} \mathbf{h}_{d_i}(\mathbf{C})_j} \quad (16)$$

$$d^2_{Chi-Square} = \sum_{j=1}^{m} \frac{(\mathbf{h}'_{d_i}(\mathbf{X}')_j - \mathbf{h}'_{d_i}(\mathbf{C})_j)^2}{\mathbf{h}'_{d_i}(\mathbf{X}')_j + \mathbf{h}'_{d_i}(\mathbf{C})_j} \quad (17)$$

$$z_p(d_i) = (1 - d^2_{Chi-Square}) \quad (18)$$

6.8. Importance Weighting

It is highly likely that multiple objects will share common colours, leading to noisy measurements. In cases where segmentation is bad, the background scene can contribute a fair amount of clutter which will cause some of the measurements to lose validity. Therefore in order to ensure that the overall confidence measurement is not compromised, each feature must be weighted by its importance.

Importance is determined based on how common a feature is found to be spatially. For instance, a colour which is visible over the entire image should be considered less important than a colour which is clustered in the vicinity of a target object when constructing each measurement.

Calculation of the importance weightings **I** involves the estimation of the spatial variance of each model centre in **C** for the entire image. This is accomplished by calculating the proportional spatial range of each feature out of the whole image. The importance weighting I_a for an arbitrary object centre $\mathbf{C}_a$ is the sum of the number of occurrences h_{sum} of $\mathbf{C}_a$ across all divisions, divided by the total number of divisions i. This is calculated for each image dimension, averaged and then squared to produce an importance value in the range $(0, 1)$[3]:

$$I_{a_x} = \frac{h_{sum_x}}{i} \quad (19)$$

$$I_{a_y} = \frac{h_{sum_y}}{i} \quad (20)$$

$$I_a = \left(\frac{I_{a_x} + I_{a_y}}{2}\right)^2 \quad (21)$$

$$\mathbf{I} = (I_1, I_2, ..., I_m), \quad (22)$$

where **I** is the vector of importance values $(I_1..I_m)$ for all object centres.

Importance weights are applied by multiplying each object centre in each measurement by its corresponding weighting. This also requires that the measurements are subsequently normalised by the sum of the object model's importance weightings in order to maintain the $(0, 1)$ measurement range.

7. Results

Acquiring meaningful results for a computer vision system is a difficult process. This arises from the fact that the exact definition of good performance varies between different types (and goals) of systems. Standard benchmarks are therefore extremely hard to come by and are generally only comparable when systems use similar test sequences.

[3]A low importance corresponds to a low contribution of that object centre to the likelihood and visa versa.

7.1. Performance Evaluation

A number of methods exist for evaluating the performance of vision systems. Even though the POD system is not entirely a stand-alone surveillance platform, it does exhibit certain similarities which warrant the use of some surveillance metrics.

7.1.1. *Surveillance Metrics*

The following basic metrics (taken from [2]) have been used to gauge overall system performance:

$$\text{TRDR} = \frac{\text{Total True Positives}}{\text{Total Number of Ground Truth Points}} \quad (23)$$

$$\text{FAR} = \frac{\text{Total False Positives}}{\text{Total True Positives + Total False Positives}} \quad (24)$$

$$\text{OTE} = \frac{1}{N_{rg}} \sum_{i=1}^{N_{rg}} \sqrt{\frac{(xg_i - xr_i)^2 + (yg_i - yr_i)^2}{w_{im}^2 + h_{im}^2}}. \quad (25)$$

The TRDR (Tracker Detection Rate) provides a general measure of the system's accuracy by describing the proportion of correct classifications for all frames in which ground truth is available. Similarly, the FAR (False Alarm Rate) determines how often the system claims an object is present when it is not.

Finally, the OTE (Object Tracking Error) quantifies the overall system error by measuring the average error of the tracked path with respect to ground truth for each object model. The equation has been modified from its original form by adding the $w_{im}^2 + h_{im}^2$ denominator. This normalises the pixel error (numerator) to the length of the image diagonal which represents the largest possible error. N_{rg} is the total number of ground truth points, (xg_i, yg_i) are the object's ground truth co-ordinates at frame i, and (xr_i, yr_i) is the object's classified position point.

7.1.2. *Perceptual Complexity*

In order to compare surveillance metrics between different types of video sequences a quantitative measurement is needed to relate the intrinsic differences between each sequence. From [2], the Perceptual Complexity (PC) is defined based on: the number and extent of occlusions; the similarity of colours between objects; and the overall image quality and variance.

7.1.3. *Ground truth*

Performance evaluation of vision system is largely dependent on the availability of ground truth data, which is scarce for real-time video. Fortunately, since the sequences used for evaluation are not excessively long, manual ground truth could be generated for each frame.

7.2. Test Cases

The first test case is a short four person outdoor scene which was recorded with two Sony camcorders in an exterior environment. The second sequence consists of a single perspective security camera viewing a total of seven coloured people entering and walking around a room. The final sequence used for testing comprises four ceiling-mounted cameras observing three people moving in a lab environment.

Tables 1 and 2 summarise scene information and overall results for the three test cases (processed on an Intel Pentium IV 2.8 GHz) respectively.

Figure 4 shows a labelled frame processed from Test Case 2. An example trajectory for Person 3 (green) is also shown as well as the overall confusion matrix for the whole test sequence.

Details	Case 1	Case 2	Case 3
Complexity (PC)	0.394	0.549	0.622
No. of cameras	2	1	4
No. of people	4	7	3
Running time (s)	30	90	200
Image resolution	360×240	384×288	285×189

Table 1: *Scene Information*

Performance Rating	Case 1	Case 2	Case 3
TRDR	0.89	0.85	0.76
FAR	0.025	0.00	0.03
OTE	0.15	0.16	0.24

Table 2: *Overall results*

The trajectory relates to the centres of the bounding boxes for the ground truth and matched areas respectively. The trajectory image also shows the actual matched positions by means of plotted triangles. The plotted trajectory is then constructed by using a 5-point moving average filter which removes spurious errors in the object's track. An idea of the tracking performance can be seen by comparing the matched trajectory to the ground truth path.

7.3. Discussion

The improvement due to the incorporation of temporal information illustrates how the POD system could be used by other systems for recovery from failed tracking. As the POD system is designed to make instantaneous decisions about an object's location, it does not need to process every frame. Thus ideally, a fast time-dependent tracker could operate unencumbered and only use the POD at appropriate times (i.e. when its confidence is low). Since the primary objective of this work involves investigating the uses of colour in tracking, these applications have been left to future work.

The system proves to be robust through a number of differing camera environments and is not overly sensitive to parameter tuning. Its ability to make instantaneous decision about any trained object. This allows it to cope with asynchronous video feeds and recover from occlusions.

7.3.1. Overall Ratings

In order to assess the overall system performance, the results of each test case are compared, using their Perceptual Complexity (PC) as a relative basis. Figure 3 summarises the Tracker Detection Rates (TRDR) for the test cases.

As expected, the TRDR decreases as the PC increases. Figure 3 also shows the average Object Tracking Error (OTE) for each sequence.

Finally, the False Alarm Rate (FAR) of the system is on average just below 2% with a mean processing rate of approximately 3 fps. The processing rate is not heavily affected by an increase in the number of trained objects. More elaborate optimisation of the current implementation could result in much faster frame rates.

8. Conclusions

A system capable of using the colour appearance of objects and people to detect their position within a digital video surveillance network has been presented. The system suggests a distributed,

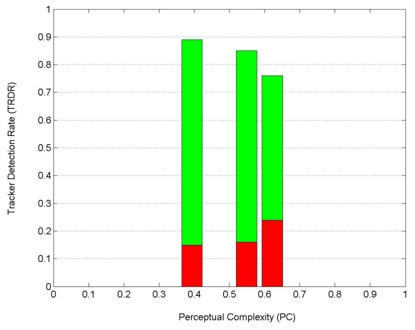


Figure 3: *Overall TRDR (upper green) and OTE (lower red) ratings for all test cases.*

bottom-up implementation which can easily be integrated with off-the-shelf network products. Features of the system include:

- Independence from camera pose and position
- Ability to deal with asynchronous video connections
- Persistence of object models over a distributed camera network
- Flexibility of integration with other visual tracking systems.

While the implementation is equipped to perform online adaptation of the models, the test sequences are not very long. Thus it was thought that not using adaptation would provide a clearer view of the system's actual performance.

Some basic surveillance metrics have been applied in order to assess the performance of the system. An average accuracy of approximately 80% is achieved for overall system performance, though the exact performance varies somewhat. The system exhibits a very low false alarm rate and is capable of an average processing rate of 3 fps with little dependence on the number of tracked objects.

9. Future work

While current 3-D tracking systems obtain good results, a common weakness is their inability to recover from failure. Since the POD system is able to generate a hypothesis without being dependent on temporal priors, it is thought that the combination of the system with a 3-D tracker (eg. [9]) could produce a viable solution. One possibility is for the 3-D tracker to exploit the POD's ability to deal with asynchronous video by only presenting it with frames when the overall system confidence is low. Alternatively, the POD could be used full-time for providing a Kalman or particle filter with observations.

10. Acknowledgements

Many thanks to TSS Technology, De Beers and the National Research Foundation for supporting this research. Additional thanks go to Markus Louw who hand-segmented ground truth for the test sequences.

11. References

[1] H. Austermeier, G. Hartmann, and R. Hilker. Color-calibration of a robot vision system using self-organizing feature maps. *ICANN*, pages 257–262, 1996.

[2] J. Black, T. Ellis, and P. Rosin. A novel method for video tracking performance evaluation. In *Joint IEEE International Workshop on VS-PETS*, October 2003.

[3] P. J. Burt, T. H. Hong, and A. Rosenfeld. Segmentation and estimation of image region properties through cooperative hierarchical computation. *IEEE Transactions On Systems, Man, and Cybernetics*, 11(12):802–809, 1981.

[4] D. Doermann and D. Mihalcik. Tools and techniques for video performance evaluation. In *Proceedings of the Third International Workshop on PETS*, June 2002.

[5] C. Erdem, B. Sankur, and A. M. Tekalp. Metrics for performance evaluation of video object segmentation and tracking without ground-truth. In *IEEE International Conference on Image Processing*, October 2001.

[6] Intel. *Open Source Computer Vision Library Reference Manual*. Intel Corporation, USA, 2001. [Online]. Available: http://www.sourceforge.net/projects/opencvlibrary/index.html [February 2004].

[7] A. Kosir and J. F. Tasic. Pyramid segmentation parameters estimation based on image total variation. In *Proceedings of IEEE conference EUROCON 2003*, 2003.

[8] S. J. McKenna, Y. Raja, and S. Gong. Tracking colour objects using adaptive mixture models. *Image Vision Computing*, 17:225–231, 1999.

[9] B. Merven, F. Nicolls, and G. de Jager. Multi-camera person tracking using an extended kalman filter. In *Fifteenth Annual Symposium of the Pattern Recognition Association of South Africa*, 2003.

[10] A. Senior. Real-time articulate human body tracking using silhouette information. In *Joint IEEE International Workshop on VS-PETS*, October 2003.

[11] P. Y. Simard, L. Bottou, P. Haffner, and Y. L. Cun. Boxlets: a fast convolution algorithm for signal processing and neural networks. In *Advances in Neural Information Processing Systems*, volume 11, pages 571–577. 1999.

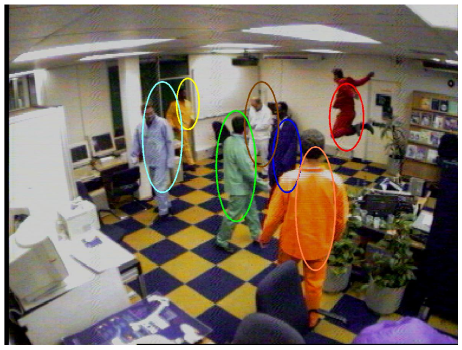
[12] S. W. Smith. *The Scientist and Engineer's Guide to Digital Signal Processing*. California Technical Publishing, 1997.

[13] C. Stauffer and W. Grimson. Adaptive background mixture models for real-time tracking. In *Computer Vision and Pattern Recognition*, 1999.

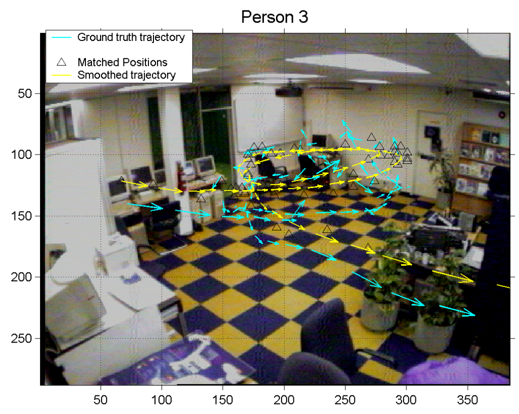
[14] P. Viola and M. Jones. Robust real-time object detection. In *Second International Workshop on Statistical and Computational Theories of Vision — Modeling, Learning, Computing, and Sampling*, 2001.

[15] S. Westland. Frequently asked questions about colour physics. Technical report, Colourware, 2000. [Online]. Available: http://www.colourware.co.uk/cpfaq.htm [January 2004].

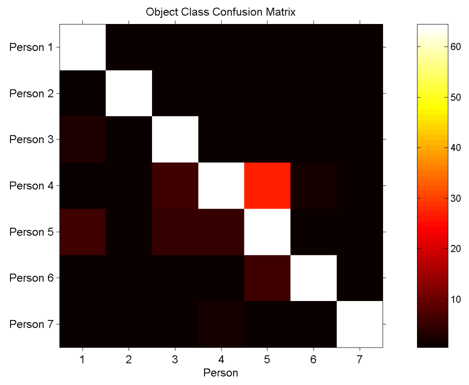
[16] G. Wyszecky and W. S. Stiles. *Color Science: Concepts and Methods, Quantitative Data and Formulas*. J. Wiley & Sons, New York, 1982.



(a) Example matched frame



(b) Overall trajectory for Person 3 (green)



(c) Overall Confusion Matrix

Figure 4: *Example result images taken from Test Case 2.*