Monitoring, Recognizing and Discovering Social Networks

Ting Yu Ser-Nam Lim Kedar Patwardhan Nils Krahnstoever

Visualization and Computer Vision Lab
GE Global Research, Niskayuna, NY 12309, USA
{yut,limser,patwardh,krahnsto}@research.ge.com

Abstract

This work addresses the important problem of the discovery and analysis of social networks from surveillance video. A computer vision approach to this problem is made possible by the proliferation of video data obtained from camera networks, particularly state-of-the-art Pan-Tilt-Zoom (PTZ) and tracking camera systems that have the capability to acquire high-resolution face images as well as tracks of people under challenging conditions. We perform "opportunistic" face recognition on captured images and compute motion similarities between tracks of people on the ground plane. To deal with the unknown correspondences between faces and tracks, we present a novel graph-cut based algorithm to solve this association problem. It enables the robust estimation of a social network that captures the interactions between individuals in spite of large amounts of noise in the datasets. We also introduce an algorithm that we call "modularity-cut", which is an Eigen-analysis based approach for discovering community and leadership structure in the estimated social network. Our approach is illustrated with promising results from a fully integrated multi-camera system under challenging conditions over long period of time.

1. Introduction

Computer vision algorithms, such as crowd segmentation [21], crowd tracking [10, 26] and facial biometrics, have seen tremendous progress in recent years and can now handle challenging real-world conditions. They enable an exciting new range of capabilities for detecting and recognizing actions [16], activitives [22], and events [5, 20], that go beyond the traditional capabilities of automated surveillance systems .

More specifically, in this work, we attempt to gain a higher level understanding of crowd behavior in terms of interaction and social network patterns. A *social network* consists of groups of people with a pattern of interactions between them [12] and the understanding of such networks in environments such as prisons or public venues is of great interest to law enforcement and homeland security. In particular, there is an increasing need to identify cohesive groups, which we called *social groups*, and their leaders for security purposes. One can easily imagine, for example in a prison environment, the value of automatically identifying different *gangs* and their *leaders* as well as changes over time in gang and leadership structures.

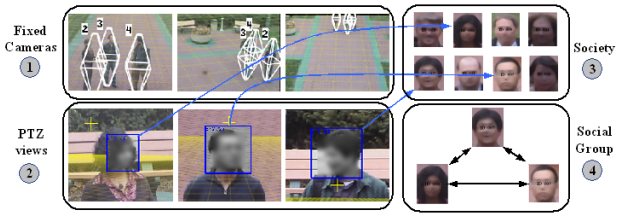

Figure 1. **Discovering Social Networks:** Person tracks obtained from fixed camera views are used to control the PTZ cameras to capture face images. Faces recognized from these images and person tracks are in turn used to build the social network or society. The social interactions captured by the social network form the basis for discovering the social groups.

Here, the low-level vision tasks that need to be performed *reliably* include:

1. Persistently tracking an individual under occlusions. These tracks allow the system to detect individuals that are often "seen" together and assign them to the same social group. For a multi-camera setup, centralized tracking in a common coordinate system is required.

2. Uniquely recognizing an individual on a watchlist using face detection and recognition [8, 18]. For this purpose, high-resolution images of faces are required.

Towards this end, advances in real-time tracking [1, 17, 25] and PTZ camera control algorithms [11], that are capable of effectively panning, tilting and zooming PTZ cameras to capture high-resolution face images of people, have made it possible to perform these low-level tasks in an efficient and robust fashion.

Under this premise, we assume that the 3D tracks of individuals and high-resolution face images are provided, allowing us to focus on the high-level task of analyzing and discovering social groups in a social network. As far as the authors are aware of, this is a new problem in the computer vision community. Arguably, previous work on group tracking [2, 6, 7] bears some resemblance to social grouping, but a few important differences will become clearer as we continue.

978-1-4244-3991-1/09/$25.00 ©2009 IEEE

To begin with, the identities of individuals have to be maintained with respect to a (possibly dynamically generated) watchlist of faces. That is, during tracking, individuals are identified by performing face recognition. This allows the system to reason about interactions between different individuals, e.g., Joe and Frank have been together for the last five minutes. In this way, connections between individuals, represented in a social network graph, who are frequently seen together, become progressively stronger over time. The nodes in a social network graph represent the individuals as identified by their faces and the edges are weighted according to the observed connections between nodes. Such a social network graph can be built over time, even offline, as long as individual tracks and captured face images are stored. When a social network graph is divided into social groups, each group conveys certain high-level information (e.g., Joe is the leader of the group comprising Frank, John and Tracy), which is not possible with mere group tracking.

Building such a social network graph and discovering its social groups correctly is non-trivial. Several challenges involved in this process have to be tackled and we propose robust solutions to address these challenges in this paper. Noisy tracking and face recognition typically "corrupt" the social network graph, which could potentially mislead the division of the social network graph into incorrect social groups. Moreover, given a set of tracks (observed from a fixed camera network) and recognized faces (observed from a bank of PTZ cameras), one has to associate each face with its track correctly, a difficult task due to errors in calibration, uncertainty about the PTZ camera state, and the proximity between tracks under crowded conditions. We overcome these challenges with an energy function that elegantly models the association problem and minimize it using the multi-way graph-cut algorithm [3, 4, 9].

The captured social interactions form the basis for dividing the social network into social groups. Essentially, we are faced with a graph based clustering problem, for which most previous work such as [15, 19, 23, 24] have focused on a variety of techniques for minimizing the size of the cut, which is defined as the strength of the edges that connect two disjoint clusters. In this paper, we will introduce a technique called the modularity-cut, originally proposed by Newman [12, 13] in the domain of social network analysis.

The modularity-cut proposed here is an appealing technique for discovering social groups because of its computation of group assignments in a way that is very meaningful in the "social" sense. The basic idea is that a cut should not be determined merely by its size, but rather by its size relative to the expected size (i.e., when the connections between groups are *smaller than expected*). As we will see, this is a powerful idea, as it inherently does not need to know the size of the social group beforehand (most cut based techniques require some way of checking for the trivial case where all nodes tend to fall into one group, in which case we obtain a zero cut size). Moreover, it provides for a simple way to discover the number of social groups and their leaders automatically.

The contributions of this work can be summarized as follow:

1. We address the problem of discovering social groups in, possibly closed-world, surveillance environments, which is an emerging challenge for the computer vision community. The goal of identifying cohesive groups and their leaders has tremendous value for practical surveillance purpose.

2. We present an elegant energy function to model the problem of associating tracks (in fixed camera views) with detected faces (in PTZ views). This will allow us to construct a social network graph while mitigating errors coming from low-level vision tasks.

3. We introduce the modularity-cut algorithm that (as far as the authors are aware of) has not been applied to the domain of computer vision. The modularity-cut has several inherent advantages over classical clustering techniques. The algorithm also leads to an elegant way of estimating the group leaders. This is an important difference from mere group tracking.

4. We have built a fully integrated system that produces very good results, as described in the experiments (Sec. 4). The system consists of a centralized 3D tracker that utilizes multiple fixed cameras. The tracker performs very robustly under occlusions, and effectively controls a set of PTZ cameras to capture close-up face images. The tracks and the captured face images serve as input for building the social network graph and discovering social groups.

This paper is organized as follow. In the next section, we will discuss the algorithm for constructing the social network graph, and provide details for associating faces with tracks. Then, in Sec. 3, we will introduce the modularity-cut and analyze its utility for dividing a social network graph into multiple social groups. We will also see how we can identify the group leaders via modularity based Eigen-analysis. Finally, we report results from a fully integrated system in Sec. 4 and conclude in Sec. 5.

2. Building Social Network

A social network graph, $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, consists of a set of nodes, $\mathbf{V}$, and a set of edges, $\mathbf{E}$. Each node represents an individual in the society and is assigned a name and signature (which includes a face image and other identifying information). In a closed-world environment, the number of nodes, N, equals the number of signatures on a given watchlist. To construct $\mathbf{G}$, we estimate the social connection strength, A_{ij}, between two individuals, i and j, based on the following guidelines:

1. Both individual i and j need to be positively recognized.

2. The interaction between them needs to be quantified with a suitable metric.

3. The frequency with which they are seen together should be measured appropriately over time, i.e., we need to aggregate the knowledge gained from the above two guidelines over the "lifespan" of a given set of tracks and recognized faces.

To achieve these guidelines, we begin by leveraging a state-of-the-art face recognition engine [8] that opportunistically, hence often intermittenly, recognizes faces detected [18] in the captured images. We obtain from the face recognition engine a discrete probabilistic histogram, $\mathbf{p} = [p^1, \ldots, p^N]$, where each histogram bin, p^i, measures the probability that the recognized face corresponds to individual i. The index, i', of the bin with the highest value is thus, in a probabilistic sense, the ID of the individual. Given a pair of histograms, $(\mathbf{p}, \mathbf{q})$, and $i' = \arg\max_i p^i, j' = \arg\max_j q^j$, we can then update the social connection link, $A_{i'j'}$, with the degree of interactions between individual i' and j', which we model as the motion similarity of i' and j'.

Suppose now that we are given a total of M tracks. We denote each track $m \in M$ by

$$\mathbf{X}_m = \{\mathbf{x}_m^{t_{m,0}}, \ldots, \mathbf{x}_m^{t_{m,\tau}}\}, \tag{1}$$

where $\mathbf{x}_m^t$ is the 3D ground plane location at time t, and $t_{m,0}$ and $t_{m,\tau}$ are the start and end time of the track. Given a pair of tracks, $(\mathbf{X}_m, \mathbf{X}_n)$, which overlaps temporally between (t_0^{mn}, t_τ^{mn}), where

$$t_0^{mn} = \max(t_{m,0}, t_{n,0}), \quad t_\tau^{mn} = \min(t_{m,\tau}, t_{n,\tau}), \tag{2}$$

we quantify their motion similarity as

$$D_{mn} = \exp\left(-\frac{\sum_{t=t_0^{mn}}^{t_\tau^{mn}} \|\mathbf{x}_m^t - \mathbf{x}_n^t\|^2}{2\sigma_{loc}^2(t_\tau^{mn} - t_0^{mn})}\right), \tag{3}$$

such that the more consistently two tracks move together, the larger the similarity D_{mn} is. Here, σ_{loc} is a scaling factor that controls the influence of the variations between the tracks' locations.

Based on Eq. 3, we can now define the rule for updating $A_{i'j'}$ for a pair of recognized faces, $(\mathbf{p}, \mathbf{q})$, and their tracks, $(\mathbf{X}_m, \mathbf{X}_n)$ (finding the association between faces and tracks will be addressed in Sec. 2.1), as

$$A_{i'j'} = A_{i'j'} + D_{mn}(\exp^{-\alpha\mathcal{H}(\mathbf{p})} p^{i'} + \exp^{-\alpha\mathcal{H}(\mathbf{q})} q^{j'}), \tag{4}$$

where noisy recognition is elegantly mitigated by the entropy measure $\mathcal{H}(.)$. Given a histogram $\mathbf{p}$, the larger $\mathcal{H}(\mathbf{p})$ is, or equivalently, the more uniformly distributed histogram $\mathbf{p}$ is, which indicates ambiguous recognition, the smaller $\exp^{-\alpha\mathcal{H}(\mathbf{p})}$ would be and hence the lesser the influence on $A_{i'j'}$. In addition, Eq. 4 shows that the links are being continuously updated with all valid pairs of faces and corresponding tracks. Hence, the more frequently the system has "seen" the individuals together, the stronger a link is.

2.1. Face-to-Track Association via Graph-Cut

So far, we have assumed that the track associated with a recognized face is readily available. This assumption is frequently violated due to the following reasons. During a face capture, the images that are acquired from a PTZ camera could capture one or more faces in different parts of the image. As a result, detected faces have to be projected into 3D space (recall that our tracker operates in 3D space) in order to be associated with the tracks. Such a projection requires estimating the projection matrix of the PTZ camera as it moves, which, depending on the accuracy of the PTZ motor location provided by the system, is often inaccurate. The situation, where several faces might be detected within a single PTZ view at the same time, also makes it difficult to associate tracks using a simple distance metric due to the proximity of these individuals. Furthermore, these faces must clearly belong to different tracks, which needs to be taken into consideration during track association. On the other hand, faces from different PTZ views could belong to the same individual and should not be used to update the network. See Fig. 2 for an illustration. In this section, we will look at how we can build a social network that would still be a realistic representation of the true social interactions between individuals, in spite of these challenges.

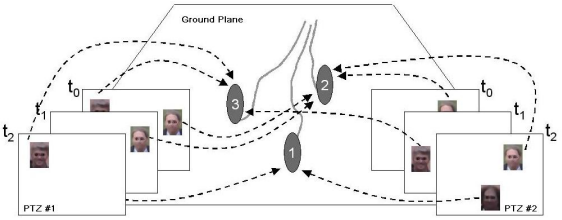

Figure 2. **Face-to-Track Association:** Three persons are tracked on the ground plane with track ID 1, 2, and 3. Several face images are detected in two different PTZ camera views. The algorithm needs to find the correct correspondence between faces and tracks.

Let us denote a set of R detected faces by $\mathbf{F} = \{\mathbf{f}_1, \ldots, \mathbf{f}_R\}$, and each capture $\mathbf{f}_r$, $r \in R$, contains

$$\mathbf{f}_r = (\mathbf{x}_r, \mathbf{\Sigma}_r, \mathbf{p}_r, t_r, c_r), \tag{5}$$

where t_r is the time of capture, c_r is the index of the PTZ camera that performs the capture, $\mathbf{x}_r$ is the 3D ground plane location of the face computed by backprojecting the detected 2D face location using the estimated projection matrix, $\mathbf{Pm}_r$, of c_r at t_r, $\mathbf{\Sigma}_r$ is the backprojection variance due to errors in the face location and noisy projection matrix estimation, and finally $\mathbf{p}_r$ is the face recognition histogram as presented in Sec. 2. Further let the set of M tracks be $\mathbf{X} = \{\mathbf{X}_1, \ldots, \mathbf{X}_M\}$, where each track $\mathbf{X}_m$ is defined as in Eq. 1. The association problem is then to assign a label l to $\mathbf{f}_r$ so that $l_r \in \{0, 1, \ldots, M\}$ indicates the track this capture belongs to. The extra label 0 is introduced to take care of outlier situations, such as missing tracks and/or face captures that are false positives.

Given such a labeling problem, and the difficulties in associating faces to tracks as mentioned, we propose a Markov Random Field (MRF) framework, and solve it using the multi-way graph-cut algorithm [3, 4, 9]. In our MRF formulation, we define the site set over all face captures $\mathbf{F}$ with $|\mathbf{F}| = R$, and the label set over $\mathbf{X}$ with $|\mathbf{X}| = M + 1$ (after adding the missing track with label $l = 0$). In this framework, we seek to find an optimal labeling, $\mathbf{L}^* = (l_1^*, \ldots, l_R^*)$, where $l_r^* \in \{0, 1, \ldots, M\}$, for all

sites by minimizing the following energy function

$$E(\mathbf{L}) = \sum_{r \in R} D(l_r) + \sum_{r,s \in \mathcal{N}} V_{r,s}(l_r, l_s), \tag{6}$$

where the data term, $D(l_r)$, is for evaluating the cost of assigning the face capture $\mathbf{f}_r$ (site r) to track $\mathbf{X}_{l_r}$ (label l_r), and the pairwise smoothness term, $V_{r,s}(l_r, l_s)$, is for computing the cost of assigning the sites (r, s) (face captures $(\mathbf{f}_r, \mathbf{f}_s)$) to labels (l_r, l_s), and $\mathcal{N}$ specifies some neighborhood system.

In order to properly manage detected faces from multiple PTZ views, $\mathcal{N}$ consists of three edge types: (1) the edge between a pair of faces if they are captured from the same camera view at the same time, denoted as $\mathcal{N}_1$, (2) the edge between a pair of faces if they are captured from the same camera view but at two successive time slots, denoted as $\mathcal{N}_2$, and (3) the edge between a pair of faces if they are captured from two different camera views at the same time, denoted as $\mathcal{N}_3$.

For our data term, $D(l_r)$, we simply adopt the strategy that if the 3D face location is closer to one of the track locations than others at the capture time t_r, this face would be more likely to be assigned to this track. Thus,

1. For $l_r = 0$, i.e., the face capture is assigned to a null (missing) track
$$D(l_r) = \delta. \tag{7}$$

2. For $l_r \neq 0$, we have
$$D(l_r) = \begin{cases} d_m(\mathbf{x}_{l_r}^{t_r}; \mathbf{x}_r, \mathbf{\Sigma}_r), & \text{if } t_r \in (t_{l_r,0}, t_{l_r,\tau}) \\ \infty, & \text{otherwise} \end{cases} \tag{8}$$

where $\mathbf{x}_{l_r}^{t_r}$ is the estimated location of track $\mathbf{X}_{l_r}$ at time t_r, $(t_{l_r,0}, t_{l_r,\tau})$ defines the lifespan of this track, $d_m(\sim)$ is the Mahalanobis distance defined by

$$d_m(\mathbf{x}; \mu, \mathbf{\Sigma}) = \sqrt{(\mathbf{x} - \mu)^\tau \mathbf{\Sigma}^{-1} (\mathbf{x} - \mu)}, \tag{9}$$

and δ is set to be some penalty cost for assigning a face to a null track.

Given that the neighboring edges might be of types $\{\mathcal{N}_1, \mathcal{N}_2, \mathcal{N}_3\}$, the smoothness term is defined accordingly as

1. For $(r, s) \in \mathcal{N}_1$, $V_{r,s}(l_r, l_s) =$
$$\begin{cases} -\infty, & \text{if } l_r \neq l_s \\ 0. & \text{otherwise} \end{cases} \tag{10}$$

2. For $(r, s) \in \{\mathcal{N}_2, \mathcal{N}_3\}$, $V_{r,s}(l_r, l_s) =$
$$\begin{cases} -0.5 \exp\{-\beta d_b(\mathbf{p}_r, \mathbf{p}_s)\} * \\ (d_m(\mathbf{x}_r; \mathbf{x}_s, \mathbf{\Sigma}_s) + d_m(\mathbf{x}_s; \mathbf{x}_r, \mathbf{\Sigma}_r)), & \text{if } l_r \neq l_s \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

where $d_b(\mathbf{p}_r, \mathbf{p}_s)$ is the Bhattacharyya coefficient of two histograms defined as

$$d_b(\mathbf{p}_r, \mathbf{p}_s) = \sum_{i=1}^{N} \sqrt{p_r^i p_s^i}. \tag{12}$$

The idea here is that for case $\mathcal{N}_1$, if they are correctly assigned to two different tracks, there would be a tremendous payoff of $-\infty$. For cases $\mathcal{N}_2$ and $\mathcal{N}_3$, the payoff for assigning two faces to different tracks depends on the Mahalanobis distances between $\mathbf{x}_r$ and $\mathbf{x}_s$, and the similarity between their face recognition histograms evaluated by the Bhattacharyya coefficient. The more distant (in space) the faces are from each other and the more dissimilar their face recognition histograms are, the larger the payoff.

Finally, as mentioned, we could solve for Eq. 6 using the multi-way graph-cut algorithm, which can efficiently generate a solution within a known factor of the optimal. The resulting face-to-track associations can then be utilized for updating the social links as presented in Sec. 2.

3. Discovering Community Structure via Modularity-Cut

After building a social network, we follow on to determine its *community structure*, i.e., the social groups that it might contain. A social group, in our case, is defined as a cohesive group of individuals that are frequently seen together. Individuals in the same social group display strong connections between one another in the social network. From a graph-theoretic perspective, the problem is to divide the social network into subgraphs in a way that maximizes connections between nodes in each subgraph and minimizes connections between different subgraphs. A closer look at the problem should reveal the applicability of several well-known spectral clustering techniques [15, 19, 23, 24]. Most of these techniques approach the problem by looking for divisions that minimize the connections between subgraphs, or in other words, divisions that minimize the *cut size* [23].

3.1. Dividing into Two Social Groups

The utility of the above techniques for discovering community structure is limited by the requirement to know the number of social groups beforehand, which is impractical for our purpose. Shi et al. [19] have proposed that one can recursively divide the graph into two based on the normalized-cut. We take a similar recursive approach but instead of using cut size as the criterion, we adopt an approach originally proposed by Newman [12, 13] in the domain of social network study. He argued that using cut size as the division criterion is counter-intuitive to the concept of social group and that one instead needs to maximize the *modularity measure* [14], which expresses the difference between the actual and expected connections of individuals within each social group. We call this technique the modularity-cut, which has been shown [12, 13, 14] in many real-world examples to be superior at identifying social groups.

To understand the modularity-cut, one can consider the notion that two individuals, i and j, are strongly connected

only if their connection is stronger than what would be *expected* between any pair of individuals, that is,

$$B_{ij} = A_{ij} - \frac{k_i k_j}{2m}, \tag{13}$$

where A_{ij} is the connection strength between i and j, k_i and k_j are the total connection strengths of i and j (i.e., $k_i = \sum_j A_{ij}$), and $m = \frac{1}{2}\sum_{ij} A_{ij}$ is the total strength of all connections in the social network graph. The term $\frac{k_i k_j}{2m}$ represents the expected edge strength, so that the further an edge (A_{ij}) deviates from expectation, the stronger the connection. From Eq. 13, the *modularity measure*, Q, can be easily derived as

$$Q = \frac{1}{2m}\sum_{\substack{i,j \in \\ \text{same} \\ \text{group}}} B_{ij} = \frac{1}{4m}\mathbf{s}^T\mathbf{B}\mathbf{s}, \tag{14}$$

where $\mathbf{s}$ is a labeling vector with each element, s_i, corresponding to an individual (node) in the social network graph. $s_i = +1$ if node i is assigned to the first group and $s_i = -1$ if node i is assigned to the second. $\mathbf{B}$ is the modularity matrix whose elements are B_{ij}. Thus, each time we divide a graph into two subgraphs, as opposed to "simply" minimizing cut size, we maximize modularity Q using $\mathbf{B}$. This is an appealing notion - by maximizing Q, we favor stronger than expected within-group connections and weaker than expected between-group connections (the cut).

Determining $\mathbf{s}$ that maximizes Q can be shown to be NP-hard. We will next outline a method based on Eigen-analysis that will provide a good approximation [13] to our problem. We first perform an eigen decomposition $\mathbf{B} = \sum_i \beta_i \mathbf{u}_i {\mathbf{u}_i}^T$ with eigenvalues β_i and eigenvectors $\mathbf{u}_i$. Substituting into Eq. 14, we obtain

$$Q = \frac{1}{4m}\sum_i (\mathbf{u}_i{}^T\mathbf{s})^2\beta_i. \tag{15}$$

Several key observations can be made in regards to Eq. 15:

1. If we let $\mathbf{s} = \mathbf{u}_i$, then since the eigenvectors are orthogonal, $\mathbf{u}_{j|j\neq i}{}^T\mathbf{s} = 0$.

2. Since $\mathbf{s}$ is constrained to be ± 1, $\mathbf{s}$ cannot be directly assigned to an eigenvector, which is real valued. Otherwise Q could be maximized by setting $\mathbf{s}$ equal to the dominant eigenvector, $\mathbf{u}_{\max}$.

3. We can, however, assign s_i to $+1$ if the corresponding element in the dominant eigenvector is positive, and -1 otherwise, i.e.,

$$s_i = \begin{cases} +1 & (\mathbf{u}_{\max})_i \geq 0, \\ -1 & (\mathbf{u}_{\max})_i < 0, \end{cases} \tag{16}$$

   where $(\mathbf{u}_{\max})_i$ is the i^{th} element of $\mathbf{u}_{\max}$. In doing so, we make an assumption that $\mathbf{s}$ remains close to being orthogonal to the other eigenvectors so that majority of the mass of the summation will come from the largest eigenvalue, thereby giving us the largest Q. It has been shown [13] that this assumption holds well in practice.

4. If none of the eigenvalues are positive, it implies that based on the modularity measure, there should be no division. This is a desirable behavior.

5. Classical spectral clustering techniques, which minimize the cut size, have to deal with the trivial case of zero cut size whereby all the nodes in the social network graph are placed in one group. In contrast, since we are *maximizing* the modularity, there is no such problem.

3.2. Dividing into Multiple Social Groups

The strategy for dividing a graph into two subgraphs can be naturally extended to finding multiple social groups by applying the modularity-cut recursively to each subgraph. For this purpose, it is important to point out that it is possible for an element in $\mathbf{u}_{\max}$ to have a value extremely close to zero. In such cases, regardless of the signs of the elements, they should be assigned to the same subgraph. This is because by being ≈ 0, these elements do not belong to either groups (Eq. 16), and should be kept together just in case subsequent divisions determine that they belong to the same group.

To ensure that the contributions to the modularity measure generated by subsequent divisions are correctly computed, it is imperative that such contributions be related to the original graph. This is different from the recursive approach proposed in Shi et al. [19], whereby each subgraph is treated independently. In our case, this amounts to removing the edges to other subgraphs and results in maximizing the wrong modularity measure.

We first define a $n \times c$ community structure matrix, $\mathbf{S}$, where n is the number of nodes in the social network graph and c is the number of social groups. We begin with $c = 1$, i.e., there is only one group (the entire social network graph), but c increases as we recursively divide into multiple groups. The $(i,j)^{th}$ element of $\mathbf{S}$ is 1 if node i belongs to social group j, and 0 otherwise. It is obvious that the modularity can be equivalently measured as

$$Q = \mathbf{Tr}(\mathbf{S}^T\mathbf{B}\mathbf{S}), \tag{17}$$

where $\mathbf{Tr}$ represents the trace operator, and $\mathbf{B}$ is the original modularity matrix. Based on Eq. 17, the strategy for dividing into multiple social groups is as follow. Each time we obtain a new social group as described in Sec. 3.1, we generate a new community structure matrix, $\mathbf{S}'$, with an additional column corresponding to the new group. Denoting the modularity for $\mathbf{S}'$ as Q' and the largest Q in the recursion so far as Q_{max}, the contribution, ΔQ, to the modularity measure is simply

$$\Delta Q = Q' - Q_{max}, \tag{18}$$

such that if $\Delta Q \leq 0$, the new group is "discarded". The algorithm at this point is very satisfying. It is much simpler to check for zero or negative contribution to the modularity before terminating the division process than, for example, using a pre-specified cut size as the termination condition.

3.3. Eigen-Leaders

Furthermore, based on Eq. 15, the modularity-cut approach provides a simple way for identifying the leader of

a social group. That is, the leader, ℓ, of a social group is simply

$$\ell = \arg\max_i (\mathbf{u}_{\max})_i. \quad (19)$$

The rationale here is simple - elements of the dominant eigenvector with large magnitudes make large contributions to the modularity.

Let us consider the social network graph, which is constructed on the basis of the frequency with which individuals are seen together. The leader of a social group, G, can be thought of as the individual in the group that was seen, on average, most frequently with everyone else in the same group. The value of $B_{\ell j}$ in Eq. 13, where $j \in G$, would be the highest among all possible $B_{ij|i,j\in G}$. Consequently, the corresponding element in $\mathbf{u}_{\max}$ would have the largest magnitude among its group members. On the other hand, since spectral clustering techniques minimize the cut size, there is no intuitive manner for exploiting the eigenvectors to identify a group leader.

4. Experiments

In this section, we present results from a fully integrated system that consists of 4 fixed and 4 PTZ surveillance camera, capturing a total of 8 views. The fixed cameras are utilized by a centralized 3D tracker that provides the 3D locations of individuals in a common coordinate system. As shown in Fig. 3, the tracking capabilities of our system under occlusions are very good, being able to track a dense crowd of individuals in a small courtyard with relatively small number of errors. The tracks are then used to control the PTZ cameras to capture face images at high resolution. Referring to Fig. 1, the system performs face detection and recognition on these face images, and the recognized faces are then associated with the tracks to build the social network graph. The steps for building the social network and discovering social groups can be optionally performed offline as long as the tracks and face images captured online are properly stored. For the remainder of this section, we will present results from various aspects of the system, namely face detection and recognition, face-to-track association, discovering social network and identifying group leader.

Face Detection and Recognition: We firstly establish the performance of the face detection and recognition components, which we evaluated on a short section of video, containing only a single subject. This allows us to easily gauge the recognition performance, obtaining the results in Table 1. In this case, the system manages to capture faces in about 40% of the frames out of which 49% are deemed to be high-quality frontal faces. Over 98% of these are recognized with a correct recognition rate of about 88%. In our experiments, we have also observed that the recognition confidence (the score returned by the recognition engine) for correct matches is significantly larger than the confidence for incorrect matches.

Face-to-Track Association: To evaluate the performance of our proposed association algorithm presented in Sec. 2.1, the ideal experiment would be to compare the face-to-track associations returned by the graph-cut solution to the groundtruth obtained from manually associating faces with their tracks. Generating such ground truth data is however a prohibitive task, considering the large number of recognized faces even for a single track, as seen in Table 1. We overcome this by manually labeling each track (the number of tracks is significantly smaller than the number of faces) with the identity of the individual that this track is following. Next, given that multiple faces are associated with each track in our graph-cut solution, we perform majority voting whereby the most frequently recognized individual for this track is assigned to it. Furthermore, for computational reasons, our graph-cut optimization is performed for temporally partitioned segments of the tracks. Therefore, the majority voting procedure is conducted for these track segments. The groundtruth labels and the labels from our solution are then compared.

Nr. of Frames:	2156	Face Detections:	843
Frontal Detections:	411	Recognitions:	403
Correct Recog.:	353	Recog. Rate:	88%
Rank 2:	14	Rank 7:	1
Rank 3:	2	Rank 8:	1
Rank 4:	2	Rank 9:	1
Rank 5:	4	Rank 10:	1
Rank 6:	1	Rank ≥ 11:	23

Table 1. **Face Detection and Recognition Performance:** A small representative video segment was groundtruthed and evaluated for accuracy.

	Seq #1	Seq #2	Seq #3
Nr. of Frames:	7000	5400	7000
Nr. of Tracks:	20	14	22
Nr. of Segments:	364	267	904
Recognized:	352	264	597
Correct Recog.:	336	255	470
Wrong Recog.:	16	9	127
Recog. Rate:	95%	97%	79%

Table 2. **Track Recognition Performance:** We groundtruthed 3 video sequences by labeling each track with the identity of the individual that it was following, and compare them with the the results from the graph-cut solution, see text for details.

Following such a procedure, we groundtruthed 3 video sequences that contain a total of 19400 frames and 56 tracks. We show the comparative results in Table 2. Seq $\#1$ contains 20 tracks, generating a total of 364 segments, among which 352 segments are recognized (some segments may not have been associated with any faces, and thus remain unrecognizable). From these 352 segments, 336 are correctly recognized, yielding a recognition rate of 95%, which is higher than the face recognition rate in Table 1. For Seq $\#2$ and Seq $\#3$, we obtained a recognition rate of 97% and 79% respectively. While the recognition rate for Seq $\#3$ is lower than the face recognition rate in Table 1, the amount of uncertainties for the latter is significantly lower since the test sequence contains only a single subject. On the other hand, we are faced with uncertainties caused by crowded conditions, errors in the projection matrix estimations of the PTZ cameras, and motion blur due to PTZ movements. Thus, we consider the overall performance of the 3 sequences very satisfactory.

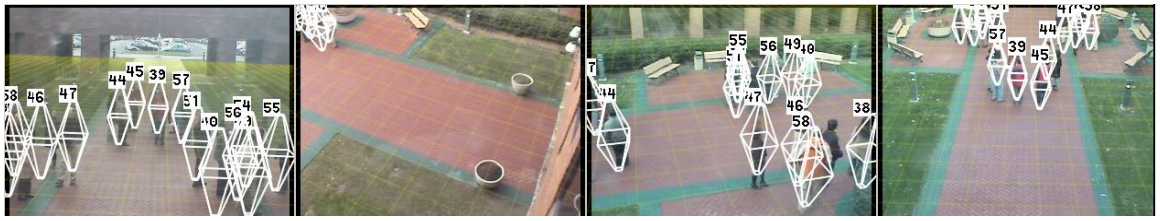

Figure 3. **Large Motion Crowds:** In this scene, there was a total of 23 individual subjects who were told to mingle in a 3-group configuration. The overlayed white wireframes with IDs show the tracking result from 4 fixed camera views.

Discovering Social Networks: We have extensively evaluated the robustness of our system in discovering social network with several challenging sequences, with the main sequence shown in Fig. 3 lasting about 30 minutes long and capturing 23 human subjects. Our system managed to track each individual quite reliably even under such a challenging condition, as seen from the 4 fixed camera views in the figure. The participants were instructed to mingle in a 3-group configuration. Given such a typical scenario, a social network was estimated and shown in Fig. 5 left image. Based on the social network graph, we proceed to discover its social groups. The modularity-cut was able to identify the correct social groups, shown in Fig. 5 right image.

We have also compared the modularity-cut with recursive division based on the normalized-cut [19] criterion. For the latter, we face significant problems in setting the cut size threshold. We found that a threshold that works for a sequence (after a trial-and-error adjustment procedure) would perform badly for other sequences. In contrast, modularity-cut performs extremely well and is able to correctly discover the social groups without requiring a tedious exercise of setting the right threshold, as explained in Sec. 3.2.

Leadership Identification: To demonstrate our algorithm's capability to identify Eigen-leaders, we evaluated the modularity-cut on an approximately 15 minutes long sequence (note that this is a different sequence from the one used in Fig. 3 and 5). The leaders are determined only at the end of the sequence so as to allow the system to observe sufficient interactions. Different members of two groups are seen interacting at different times. The leader of each group is however always present, which generates strong modularity connections between each leader and his group members. By identifying the resulting Eigen-leaders, the system was able to successfully identify the leaders, shown as red nodes in Fig. 4, which, as mentioned, is not possible with classical spectral clustering techniques that minimize the cut size.

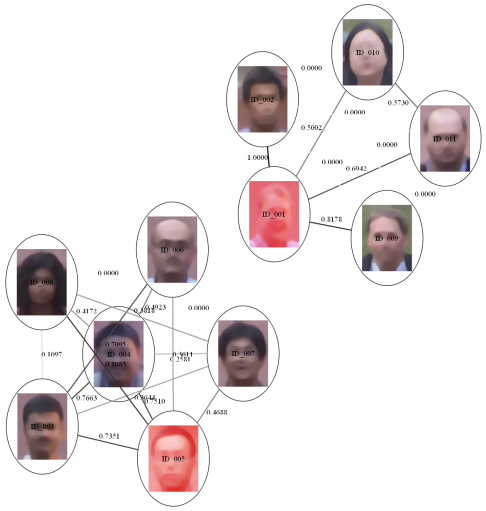

Figure 4. **Leaders of Social Groups:** In this experiment, there was a total of 11 individuals, divided into 2 groups. Members of each group appear in the scene at different times, during which the leader is always present. This generated, in a modularity sense, strong connections between the leader of each group to its members. As a result, the system was able to identify the Eigen-leaders (Sec. 3.3), which are the red nodes in this figure.

5. Conclusions

In this paper, we have addressed an emerging new problem of monitoring and recognizing social network and discovering its community and leadership structures. A computer vision solution to the problem has been proposed, which should be of tremendous value for practical surveillance purposes. Specifically, we have presented an algorithm for robustly building social network by identifying individuals and analyzing their interactions using face recognitions and person tracks, where the correspondence between them are established using an elegant graph-cut solution. Admittedly, analyzing social interactions using such information alone might be limited. A more sophisticated scheme, such as the amount of eye contacts, could be employed. These schemes, however, are mostly very sensitive to video quality and thus require further work. We have also introduced the modularity-cut as a more intuitive way of dividing the social network into social groups, and demonstrated its capability in identifying Eigen-leaders. The utility of our algorithms was demonstrated by a fully integrated system that works very well in practice; the system can be run live for an extended period of time collecting sizeable amount of tracks and face images for subsequent social network analysis. The system in its current form, however, assumes a closed-world environment where the watchlist is pre-determined. This should be desirably extended to a dynamically-generated watchlist, where new faces observed during system operation are added to the watchlist automatically.

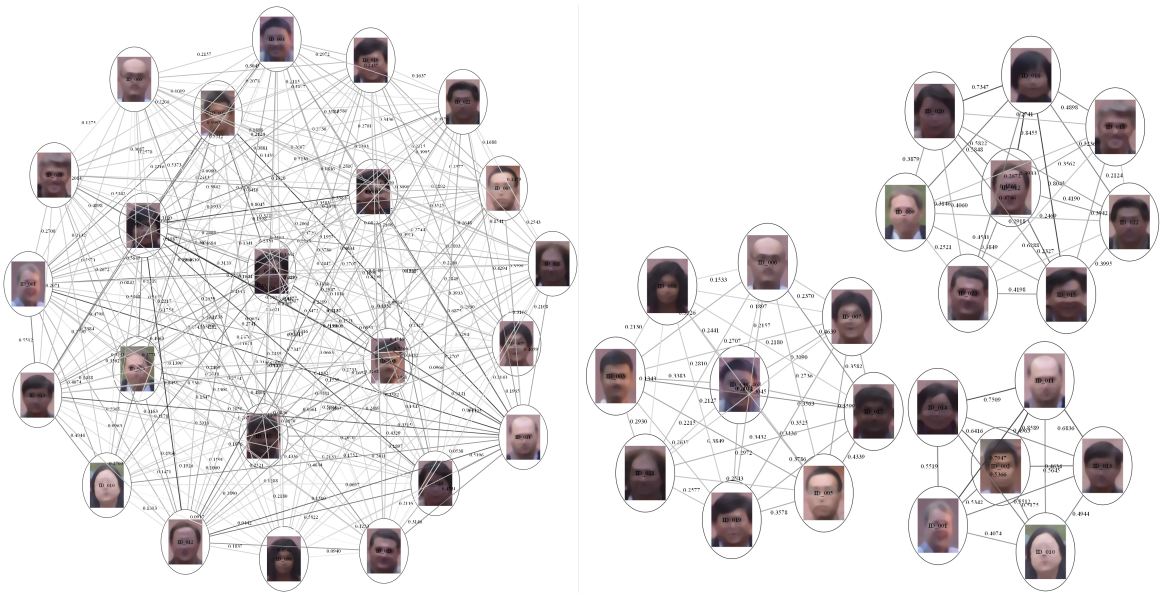

Figure 5. **Discovering Social Groups:** The left image shows the social network constructed from the sequence in Fig. 3, which was divided into 3 groups correctly using the modularity-cut, shown on the right. Here, the links are shown with the weights between the individuals.

6. Acknowledgements

This project was supported by grant #2007-RG-CX-K015 awarded by the National Institute of Justice, Office of Justice Programs, US Department of Justice. The opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the Department of Justice.

References

[1] S. Blackman and R. Popoli. *Design and Analysis of Modern Tracking Systems*. Artech House Publishers, 1999.

[2] B. Bose, X. Wang, and E. Grimson. Multi-class object tracking algorithm that handles fragmentation and grouping. In *Computer Vision and Pattern Recognition*, pages 1–8, jun 2007.

[3] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE transactions on PAMI*, 26(9):1124–1137, 2004.

[4] Y. Boykov, O. Veksler, and R. Zabih. Efficient approximate energy minimization via graph cuts. *IEEE transactions on PAMI*, 20(12):1222–1239, 2001.

[5] M. T. Chan, A. Hoogs, R. Bhotika, A. Perera, J. Schmiederer, and G. Doretto. Joint recognition of complex events and track matching. In *Computer Vision and Pattern Recognition*, volume 2, pages 1615–1622, New York City, NY, USA, jun 2006.

[6] A. French, A. Naeem, I. Dryden, and T. Pridmore. Using social effects to guide tracking in complex scenes. In *Advanced Video and Signal Based Surveillance*, pages 212–217, 2007.

[7] G. Gennari and G. Hager. Probabilistic data association methods in visual tracking of groups. In *Computer Vision and Pattern Recognition*, volume 2, pages 876–881, jun 2004.

[8] Identix. Faceit sdk.

[9] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *IEEE transactions on PAMI*, 26(2):147–159, 2004.

[10] N. Krahnstoever, P. Tu, T. Sebastian, A. Perera, and R. Collins. Multi-view detection and tracking of travelers and luggage in mass transit environments. In *In Proc. Ninth IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS)*, 2006.

[11] N. Krahnstoever, T. Yu, S.-N. Lim, and K. Patwardhan. Multiclass spectral clustering. In *Workshop on Multi-camera and Multi-modal Sensor Fusion, in conjunction with ECCV 2008*, 2008.

[12] M. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577–8582, 2006.

[13] M. E. J. Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical Review E*, 74:036104, 2006.

[14] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Phys. Rev. E*, 69, 2004.

[15] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems 14*, pages 849–856. MIT Press, 2001.

[16] V. Parameswaran and R. Chellappa. View invariants for human action recognition. In *CVPR (2)*, pages 613–619, 2003.

[17] C. Rasmussen and G. Hager. Joint probabilistic techniques for tracking multi-part objects. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 16–21, 1998.

[18] H. Schneiderman and T. Kanade. A statistical method for 3D object detection applied to faces and cars. In *IEEE Computer Vision and Pattern Recognition, Hilton Head, SC*, volume 1, pages 746–751, 2000.

[19] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, August 2000.

[20] P. Smith, N. da Vitoria Lobo, and M. Shah. Temporalboost for event recognition. In *Proceedings of the Tenth IEEE International Conference on Computer Vision*, pages 733–740, 2005.

[21] P. Tu, T. Sebastian, G. Doretto, N. Krahnstoever, J. Rittscher, and T. Yu. Unified crowd segmentation. In *eccv*, 2008.

[22] N. Vaswani, A. K. R. Chowdhury, and R. Chellappa. Activity recognition using the dynamics of the configuration of interacting objects. In *CVPR (2)*, pages 633–642, 2003.

[23] U. von Luxburg. A tutorial on spectral clustering. *CoRR*, abs/0711.0189, 2007.

[24] S. X. Yu and J. Shi. Multiclass spectral clustering. In *International Conference on Computer Vision*, pages 313–319, 2003.

[25] T. Yu, Y. Wu, N. O. Krahnstoever, and P. H. Tu. Distributed data association and filtering for multiple target tracking. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition, Anchorage, Alaska*, June 2008.

[26] T. Zhao, R. Nevatia, and B. Wu. Segmentation and tracking of multiple humans in crowded environments. *IEEE transactions on PAMI*, 30(7):1198–1211, 2008.