An Entropy Model for Loiterer Retrieval across Multiple Surveillance Cameras *

Maguell L.T.L. Sandifort[†,‡], Jianquan Liu[‡], Shoji Nishimura[‡], Wolfgang Hürst[†]
[†]Utrecht University, the Netherlands
[‡]Biometrics Research Laboratories, NEC Corporation, Japan

ABSTRACT

Loitering is a suspicious behavior that often leads to criminal actions, such as pickpocketing and illegal entry. Tracking methods can determine suspicious behavior based on trajectory, but require continuous appearance and are difficult to scale up to multi-camera systems. Using the duration of appearance of features works on multiple cameras, but does not consider major aspects of loitering behavior, such as repeated appearance and trajectory of candidates. We introduce an entropy model that maps the location of a person's features on a heatmap. It can be used as an abstraction of trajectory tracking across multiple surveillance cameras. We evaluate our method over several datasets and compare it to other loitering detection methods. The results show that our approach has similar results to state of the art, but can provide additional interesting candidates.

CCS CONCEPTS

• **Information systems** → **Multimedia information systems**; • **Social and professional topics** → **Surveillance**;

KEYWORDS

Loitering Discovery; Video Surveillance; Entropy Model; Heatmaps; Repeated Appearance; Ranking System; Loitering Candidate

ACM Reference Format:
Maguell L.T.L. Sandifort[†,‡], Jianquan Liu[‡], Shoji Nishimura[‡], Wolfgang Hürst[†]. 2018. An Entropy Model for Loiterer Retrieval across Multiple Surveillance Cameras . In *ICMR '18: 2018 International Conference on Multimedia Retrieval, June 11–14, 2018, Yokohama, Japan.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3206025.3206049

1 INTRODUCTION

Loitering is a problem in the public safety domain, where an individual stays in an area for an extended period of time without a clear goal [15]. It is seen as a precursor to illegal behavior such as pickpocketing or illegal entry.

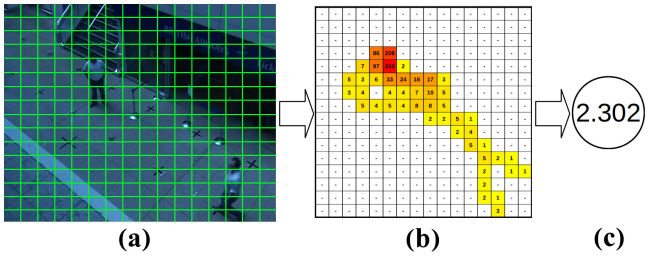

Figure 1: Overview of the entropy model. a) Split the camera view up into bins. b) Add each feature to the correct bin based on location in camera view. c) Calculate the entropy score

A number of approaches exist for detecting loitering behavior [6]. Detecting loitering is usually done by looking at a single camera view. Loitering behavior is then decided by looking at the duration of appearance, or the trajectory of a person. For duration, if the person is visible on the camera view for a certain amount of time, he or she is considered loitering. For trajectory-based methods, the length or the curvature of the trajectory is used to determine loitering behavior. Some methods also consider people leaving and re-entering the camera's field of view [5, 12].

These works provide a yes/no answer to whether or not a certain candidate is a loiterer. For this, some threshold for time or a score is involved. Choosing this threshold is challenging, as pedestrians showing suspicious behavior that stay in the area below this time threshold will not be detected. Finally, tracking based methods require people to be tracked continuously and therefore struggles in crowded scenes due to occlusions. In contrast to this single view, many areas exist where multiple cameras are installed, covering a larger region of a space or providing different viewing directions. We could apply single-camera methods to all these cameras in parallel. However, this would treat each camera as their own confined space. As a result we will lose insight into the big picture of the entire surveilled scene.

Liu et al. [9] address the problem of loitering across multiple cameras. They forgo tracking approaches and instead propose a scalable approach based on the frequency of the appearance of facial features. This approach works regardless of the amount of cameras used and it does not need to keep the positional relationship between cameras into account. It groups similar facial features together in their proposed similarity tree. It then provides a list of loitering candidates based

*This work is mainly completed during the period of internship.

ICMR '18, June 11–14, 2018, Yokohama, Japan

ACM ISBN 978-1-4503-5046-4/18/06…$15.00
https://doi.org/10.1145/3206025.3206049

on the amounts of facial features captured of one individual. Having a lot of features signifies that a person has been in the area for a long time.

While their method can provide a good list of initial candidates they only use duration to determine loitering behavior. Duration is a good initial indication of loitering, but there are other indicators as well, such as frequently reappearing in the same area. As they do not rely on tracking methods, they do not consider the trajectory of a person either.

To utilize the more in-depth information that tracking approaches provide, without relying too much on detailed tracking, we introduce an entropy model. We use this model to determine the amount that someone moves across a camera view. Using this model, we assign a higher likeliness of loitering to people that move more as opposed to standing still. We combine this entropy model with appearance duration and frequent reappearance to provide a loitering candidate list.

An overview of our entropy model can be seen in Fig. 1. First we input the features of people in the camera view into our system. For each camera view we create a heatmap, by splitting the camera view into bins. Next, we transform the position of features to the heatmap. The heatmap serves as an abstraction for tracking methods. It shows the rough trajectory of a person across the camera view, as well as how long they stood still in a certain location. Then by calculating the entropy of the heatmap, we get a measure for the amount a person moved inside the camera view.

Based on this hybrid approach we can provide a potential list of loiterers in the same level of accuracy and recall as other loitering detection methods. Both single- and multi-camera feature extraction can be used to provide input to our system. Our method scores people that show more aspects of loitering as higher candidates than people that only appear for a long duration. In addition our method provides additional interesting candidates.

In this paper we contribute the following:

- We propose an entropy model applied to a multi-camera surveillance system that models the movement of people. In addition we propose a measure for repeated leaving and entering of an area.
- We combine our measure into a loitering score that can be used to provide a list of potential loitering candidates.
- We conduct experiments over several datasets to evaluate our method and compare it to other loitering retrieval methods. The results show that our method provides similar levels of accuracy and recall as with other state-of-the-art methods, but will rank moving and frequently reappearing candidates higher. Finally, our method provides additional interesting candidates.

2 RELATED WORK

Trajectory. One way to detect loitering behavior is by looking at the trajectory of a pedestrian. Ko et al. [7] transform their camera image to account for the camera's angle and then track the center of an object moving in the camera view. Loitering behavior is then determined if a trajectory has irregular and large direction variations compared to a pre-defined threshold. Li et al. [8] track the trajectory of people moving in their static scene and determine loitering based on the angle of their path and the time duration of the path. Park et al. [13] propose a loitering detection algorithm using a regional histogram and object velocity. While a pedestrian is tracked, their location is stored in a histogram. If they exit and re-enter the same area multiple times, they are considered a loiterer. In addition to that, if the pedestrian's velocity is lower than a pre-defined threshold and the displacement is higher than a pre-defined threshold, the pedestrian is classified as a loiterer.

Looking at the trajectory can be an effective way to detect loitering behavior. However, these approaches require continuous and reliable tracking to obtain the trajectories. In crowded or occupied scenes the tracking could fail, making these methods difficult to use in real scenarios. We will propose the entropy model measure based on tracking that does not require a consistent view of people, but will still provide information on where the tracked person has been.

Entropy. Some methods utilize entropy theory to detect abnormalities. Ren et al. [14] propose a behavior entropy model to detect abnormal behaviors in a crowd. Xiong et al. [17] use a kinetic energy model to determine irregularities in crowd behavior. They create a histogram for each axis of the foreground pixels. Then they calculate the entropy to determine the crowd dispersion, which is used to get the kinetic energy of the crowd. If the crowd dispersion or kinetic energy exceeds a set threshold, an alarm signal is sent.

These works show that entropy theory can be applied to detect irregularities in crowds. Inspired by this, we will apply entropy theory to individuals as a measure to determine how much they are moving.

Classification. Bird et al. [2] use a single camera method to detect loiterers around a bus stop. Their method takes snapshots of people that are clearly visible and then uses the clothing color of people for similarity matching. Loitering is determined based on the duration of appearance. Tomás et al. [16] identify sequential micro-patterns (such as walk, stop, walk, stop) to determine loitering behavior of the elderly. Elhamod et al. [3] provide a semantics-based approach using both features of objects and the relationship between them to define and detect behavior of interest, such as loitering or abandoned luggage.

Nam [12] tracks pedestrians as blobs using a color feature model. Difference in shape, color and distance traveled is then used to track these blobs over multiple frames. It raises a loitering alarm if a person remains in a pre-defined region of interest longer than a loitering time-threshold. The method also accounts for pedestrians briefly moving out of view and re-entering again, without resetting the alarm timer. The loitering time-threshold is adapted automatically during a learning phase.

Table 1: Quick reference to the meanings of notations used through the paper.

Symbol	Meaning of notation
C_i	Camera with id i
CC	Camera count, the amount of cameras the candidates is visible on
CA_j	Camera Appearance with id j
T_{ca}	Camera Appearance Threshold
$RA(C_i)$	Camera i's Reappearance
RAS	ReAppearance Score
$FC(x)$	Feature Count of x
b_i^j	Bin with id i in the heatmap of CA_j
$p_{b_i^j}$	Probability of bin with id i in the heatmap of CA_j ($p_{b_i^j}$ for empty bins)
he_j	Heatmap entropy of the heatmap of CA_j (excludes empty bins)
$Dur(x)$	Duration of x
$P(CA_j)$	Presence of CA_j
WES	Weighted Entropy Score
DS	Duration Score
LS	Loitering Score

Huang et al. [5] extract their features through a color structure approach. They can then track people in a single camera. They define local loitering; someone loitering in the scene for an extended period of time, and global loitering; which keeps in mind that the candidate can temporarily leave the camera view and then return to it if they are loitering around the area. Arsic et al. [1] use multiple cameras surveilling the same scene at different angles to track and detect loiterers and abandoned luggage.

Lu et al. [11] present a summarization method consisting of three steps. First, pre-processing finds and tracks objects. Next, the holistic-level summarization decides on representative images using an energy minimization method. Finally, the object-level summarization provides relevant meta data for each object. Loitering is detected by mapping the trajectory of objects on histograms for each axis. If the ratio between windows of these two histograms exceeds a threshold value the object is considered a loiterer.

Most of the related work on loitering detection only provides a classification of a loiterer. For this, some threshold for time or a score is involved. Choosing this threshold is not easy in practice. For example, pedestrians showing suspicious behavior that stay in the area below this time threshold will not be detected. Additionally, they make no distinction between people who move and people who are waiting. When analyzing a large amount of footage, there could be many potential loiterers. It will be difficult to fine tune an alarm based on a time threshold. We reconsider trying to classify loiterers, by providing a list of candidates instead, with the most likely loiterer at the top. This allows the operator to make the final judgment. Our method can retrieve loiterers and provide other suspicious candidates in an ordered list sorted by suspicion level.

3 OUR APPROACH

Most methods deal with classification of loitering. They classify a person as loitering or not loitering. Such a yes/no answer requires a score or a time threshold. Thresholds will bring misclassification for border cases.

In reality however, there will still be an operator that is using surveillance systems. We do not have to completely replace the operator and instead should focus on assisting them. With this in mind, we rethink the loitering analysis as a *loiterer retrieval* problem in this paper. We propose a ranking measure to retrieve a list of loitering candidates ordered by level of interest. The operator or expert in the surveillance domain can then use this list to determine the actual loiterers. In our model, we take into account three important properties to determine loitering behavior: (1) entropy, (2) reappearance and (3) duration. Finally, we combine these properties into a (4) loitering score. Their details can be found in next subsection.

3.1 Model description

Before we describe our measure, we introduce necessary definitions that will be used in our approach.

Definition 1 (**CA**: Camera Appearance). *A sequence of chronologically ordered features of the same person on the same camera, starting when they enter the camera view and ending when they leave the camera view (after a time threshold T_{ca})*

We then use the camera appearance CA to introduce our entropy model that will be adopted in the ranking measure.

(1) Entropy Model. Two people that appear on a camera for the same amount of time (i.e., duration) can show vastly different behaviors. For example, one person could be entering the scene and then waiting in one spot, before moving again and leaving the scene. The other person could be moving around the camera view before leaving the scene. Using only duration does not distinguish between these two behavior. Likewise, trajectory of a person is not considered either when only using the duration.

To distinguish between these movements we propose an entropy model. For each CA, we construct a heatmap. We map the location of a person on the camera view on this heatmap. This way we get a rough indication of the trajectory of a person in the camera view. In addition, we can identify if the person has been standing still or moving around. The advantage of using this heatmap approach over a direct tracking approach is that we do not need continuous views of a person. It serves as an abstraction of tracking where someone has been on a single or multiple cameras.

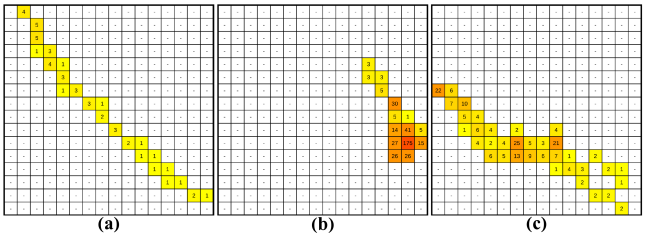

Figure 2: Heatmap examples showing the abstractions of three different movements of a person: a) walk through the area; b) keep waiting in the area; c) walk around, likely loitering in the area.

Fig. 2 shows some heatmap examples. (a) is the heatmap of a person that is walking through the camera view. We

can see the person is walking in a straight line and does not stop midway, as the distribution of the moving path is spread out equally. (b) is an example of someone who is waiting. There is a peak in the location the person is waiting, resulting in a skewed distribution. (c) is an example of someone who is loitering, by walking, stopping, changing direction, and repeat. The person moves, stops, changes direction and repeats, causing an uneven distribution with multiple small peaks.

To obtain the latent movement information from these heatmaps, we are inspired by the information entropy theory [4]. We model movement by calculating the entropy over the heatmap. First we determine the probability distribution of the features on the heatmap. In Eq. 1, the probability p_{b_i} of each bin b_i on the heatmap is computed by dividing the count of features ($FC(b_i)$) located in b_i by the total count of features ($FC(CA_j)$) located in the heatmap:

$$p_{b_i} = \frac{FC(b_i)}{FC(CA_j)} \tag{1}$$

Then, we can compute the heatmap entropy (he_j) as follows.

$$he_j = -\sum_{i=1}(p_{b_i} \log(p_{b_i})) \tag{2}$$

Empty bins do not contribute to the entropy, so they are excluded from the computation.

If someone is moving, the features are distributed over the bins, causing the entropy to rise. In addition, if someone has a non-straight trajectory, their path will go through multiple bins, further increasing the entropy. However, if someone is standing still, it skews the distribution over the heatmap, causing the entropy to decrease.

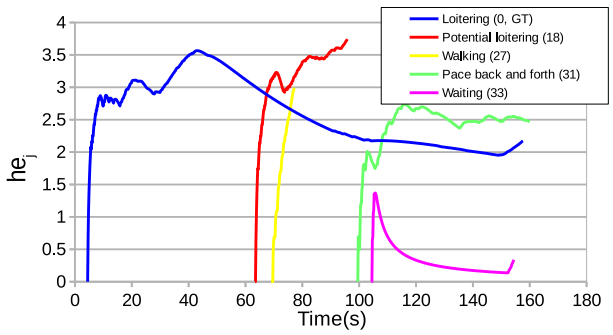

Figure 3: he_j of five persons over time. If someone is moving, the he_j will increase, while standing still causes the he_j to decrease. The he_j going up and down shows move, stop, move, stop behavior. GT indicates the groundtruth loiterer. The legend is associated with the id of a person extracted from the dataset.

Fig. 3 shows the heatmap entropy he_j over time for five persons. Once someone enters the area, their entropy will increase substantially, as only a few bins in the heatmap will be filled. After a few seconds, the entropy will show clear trends depending on the behavior. The ground truth loiterer (blue, 0) starts by pacing through the area causing

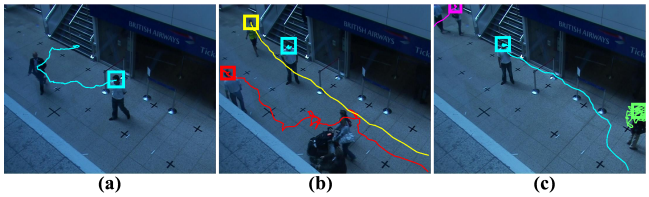

Figure 4: An example selected from the dataset (PETS S1) corresponding to Fig. 3. It shows frames at the times: 40s (a), 70s (b) and 110s (c). In addition, the path is drawn that they will make until the next image. For the last image, the path is drawn until they leave the scene.

their entropy to steadily increase. After 40 seconds the person stands still. We see this in the figure as their entropy decreases after that point. The potential loitering candidate (red, 18) enters the area and then stops, changes direction and moves again multiple times causing this person's entropy to increase over time. Note that their entropy is higher than those of people that have been standing still. Someone who is walking through the area (yellow, 27) will have a high entropy as they are only moving, but they score lower than someone who changes direction multiple times. If someone is pacing back and forth (green, 31) in a small area, their entropy will stay roughly the same over time. Finally, if a person is waiting (magenta, 33), their entropy will steadily decrease.

If someone is moving through the surveilled scene, they will pass through multiple cameras. For each camera they pass through, we construct a heatmap and calculate its entropy. Next we need to combine the entropy scores into one. Simply adding them up is not a good idea, as this will cause the entropy score of a person to be higher if they pass through more cameras, which is not what we want to measure with the entropy score. Therefore we assign weights to each heatmap, based on their duration. We mainly aim to distinguish between moving and waiting people with this measure. As both of those are associated with a high duration, we weight the heatmaps of cameras the person appeared longer on higher.

We calculate the weight of each heatmap, which we call the presence $P(CA_j)$, based on the duration of the CA_j it belongs to.

$$P(CA_j) = \frac{Dur(CA_j)}{\sum_{j=1}(Dur(CA_j))} \tag{3}$$

Finally, we combine all heatmap entropies of a person together into the weighted entropy score WES.

$$WES = \sum_{j=1}(he_j P(CA_j)) \tag{4}$$

Definition 2 (**WES**: Weighted Entropy Score). *Given a sequence of CAs, each with their respective heatmap h_j, WES is the weighted combination of the entropy of each h_j. This is a measure for the amount of movement a person shows across multiple cameras.*

In this way, longer CAs are weighted more substantially. Short CAs with high entropy, which will be someone walking through a camera view, will not contribute as much as

longer CAs. Someone waiting for a longer time will have low entropy, so while the duration is high, the entropy remains low and therefore will not contribute much to the WES either. Moving in one view for a long time however will have high duration and high entropy, so this will create a high contribution to the WES.

(2) Reappearance. The WES serves as a measure to model the movement of a person. Next we want to model the movement between camera views. Consider two different people moving through the same surveilled area. Person 1 appears on five different cameras once, while person 2 appears on one of the cameras three separate times, and appears once on 2 other cameras. If their duration is the same. They both appear for the same duration and after they leave the area they have 5 CAs each. Using the duration or the entropy is not enough to distinguish between these two behaviors. To be able to model the difference we propose the following definition.

Definition 3 (Reappearance). *Given a camera C_i, we define reappearance as the same person leaving a camera view and entering it again.*

To use the reappearance into a measure we consider the skewness of reappearance across the cameras as well. For example, someone who goes to do grocery shopping at a nearby store will likely show up on a sequence of cameras twice, once for going to the store and once for returning home. If they appear 2 times on 3 different cameras, they reappear the same amount of times as someone who appears 4 separate times on a single camera. To distinguish between these scenarios, we count each reappearance on a camera as more substantial than the previous one.

With that in mind, we can then calculate a camera reappearance score for a person as follows:

$$RA(C_i) = 2^{a-1} \tag{5}$$

where a is the amount of CAs for that person on camera i. In this way, the more a person reappears on the same camera, the more rapidly the $RA(C_i)$ score increases in exponential scale.

Finally, we need to normalize the $RA(C_i)$ for the amount of cameras. To do this, we add the $RA(C_i)$s together and then divide by the amount of unique the person appeared at least once on. Doing that we get our final ReAppearance Score RAS:

$$RAS = \sum_{i=1}^{n} RA(C_i)/CC \tag{6}$$

Definition 4 (**RAS**: ReAppearance Score). *Given all cameras C_i a person has appeared at least once on and their respective $RA(C_i)$, RAS is the weighted combination of reappearances.*

It is a measure of skewed appearance, showing that the person is hanging around a certain area instead of merely walking through it. Therefore, despite the same duration of appearance, someone repeatedly entering and exiting the view of the same camera can be distinguished from someone who only appears once on each camera.

For example, a pickpocket may loiter around a popular meeting spot. To not attract too much attention from bystanders he moves around the area, but keeps returning to look for victims, causing him to repeatedly leave and enter the camera view. This behavior results in a high RAS.

(3) Duration. A person who is in an area for a longer period of time is more likely to be loitering. Therefore, we need to consider the duration as well. First we define the duration of a CA as follows:

Definition 5 ($Dur(CA)$: Duration of a CA). *Given the features f_i in CA, we define $Dur(CA)$ as the difference in timestamp between the last and the first f_i in the CA in seconds.*

Then, to calculate the duration measure, we add the duration of all CAs of a person together. In the case that CA durations overlap, we do not count the overlapping durations multiple times. Finally, duration of people can vary substantially, especially in larger surveillance systems. To reduce the impact of these large difference, we scale the duration score as follows by taking the logarithm over the final duration:

$$DS = \log(\sum_{j=1}(Dur(CA_j)) + 1) \tag{7}$$

Note that we add 1 to the duration before calculating the logarithm to prevent negative values for durations less than 1.

(4) Loitering Score. Each of the measures above account for an aspect of loitering behavior. To provide a list of candidates, we combine the measures above into one loitering score. Depending on the situation, an operator might consider one aspect of loitering more important than others. Therefore we provide weights to each of the measures, so the total score can be fine tuned based on the needs of the operator.

$$LS = \alpha RAS + \beta WES + \gamma DS \tag{8}$$

Definition 6 (**LS**: Loitering Score). *Given the RAS, WES and DS, as well as the weights α, β and γ, we define LS as the weighted combination of the three measures. A higher score indicates that the person is more likely to be loitering.*

Each factor can be scaled by its associated weight α, β and γ. As the score is a comparative value, these do not have to add up to 1, only the ratio between the three should be considered.

The LS is used to give a ranking of loitering candidates. The score on its own is meaningless and serves mainly as a comparison value between candidates. Likewise, the score is bound to the system it was obtained in and should not be used to compare to the score of another system.

3.2 Implementation

Next we will go over how we create our candidate list. For this we use Algorithm 2. After some pre-processing, it calculates the measures for entropy, reappearance and duration for each person. The measures are then combined into the loitering score. Finally, it outputs a list of candidates ordered by loitering score. Below we will go over each part of the algorithm.

Algorithm 1: GroupFeatures()

Input: Features list FL^j containing features f_i^j of person j.
Output: List CAL^j containing CA_i^j

```
1  begin
2      Sort FL^j on timestamp ;
3      foreach Feature f_i^j do
4          Find CA_i for f_i^j.cameraID ;
5          if f_i^j.timestamp - CA_i.last.timestamp > T_ca
            then
6              CAL^j.push(CA) ;
7              Create new CA for f_i^j.cameraID
8          Add f_i^j to CA ;
9      foreach Open CA_i do
10         CAL^j.push(CA) ;
11     return CAL^j ;
```

Pre-processing. Before we can calculate our measures, we first need to do some pre-processing on our data. Each of the features comes attached with the following metadata: a person identifier, a camera identifier, a timestamp and the position in the camera view. We need to group these features into CAs. See Algorithm 1.

First the features are sorted by timestamp so they are in chronological order (line 2). As people can appear simultaneously on multiple cameras, we keep track of open cameras. For each feature we add to a CA, we compare the timestamp of the last feature added to that CA (line 5). If the time difference exceeds the T_{ca}, we close the old CA and push it to the list. The feature is then added to a new CA for the same camera (line 6 and 7). Otherwise, we just add the feature to the existing CA (line 8).

Finally, after we have added all features to a CA we push the remaining CAs (line 9 and 10) and return the list (line 11). We then use this list of CAs for that person to calculate the measures.

Measures. The construction of the candidate list can be seen in Algorithm 2. For each person, we group the features into CAs (line 3). Then we construct a heatmap for each of the CAs. We split the heatmap into bins according to the hD (line 5). For an hD of 16, we split the heatmap into 16x16 bins, for a total of 256 bins. Next we iterate over the features in the CA. We map the center coordinates of the position

Algorithm 2: BuildCandidateList()

Input: Data set ds with features f_i^j grouped by person j, RAS scalar α, WES scalar β, Duration scalar γ, heatmap Dimension hD
Output: List of candidates CL ordered by score.

```
1  begin
2      foreach Candidate j ∈ ds do
3          CAL^j ← GroupFeatures() ;
4          foreach CA in CAL do
5              Construct heatmap of CA with hD dims. ;
6              Compute he_j by using Eq. (2) ;
7          Compute WES using Eq. (4) ;
8          Compute RAS using Eq. (6) ;
9          Compute DS using Eq. (7) ;
10         Compute LS using Eq. (8) ;
11         CL.push(j) ;
12     Sort CL by LS ;
13     return CL ;
```

metadata of the feature to the heatmap to the appropriate bin in the heatmap. Next we calculate the entropy over each heatmap (line 6).

After that, we have the information we need to calculate the measures WES, RAS and DS (line 7, 8 and 9). We then calculate the LS by scaling each measure with their appropriate scalar (line 10), after which the processing for that candidate is done. Finally, we push the candidate j to the candidate list CL (line 11) and continue to next person.

After all people have scores assigned to them, we order the candidate list on the LS (line 12).

4 EXPERIMENTS

The experiments were conducted on the Ubuntu 16.04 OS, with an Intel® Core™ i7-6700 CPU @ 3.40GHz and 11 GB of memory. We evaluate our approach using two datasets.

- AntiLoiter Dataset[1] [9, 10], and
- PETS 2007[2] Scenario 1 (PETS S1) and Scenario 3 (PETS S3).

AntiLoiter Dataset. A multi-scene multi-camera dataset containing groups of people walking in public areas. Videos are high resolution (1920x1080). Faces are clearly visible and timestamp information is provided. One designated person acting as a real loiterer shows up repeatedly on multiple cameras.

PETS2007 Dataset. Single-scene multi-camera footage of an airport. The cameras have a resolution of 768x576, which consist of multiple scenes for the detection of loitering and abandoned luggage. To compare our approach to one of the state of the art [12] and to show that we can use our approach on single camera scenes we only use the third

[1] https://github.com/ryukenzen/antiloiter/
[2] http://www.cvg.reading.ac.uk/PETS2007/data.html

camera of each scenario to extract the trajectories of persons. To show that input can be provided by means of a tracking method, we annotate the features semi-manually, by iterating over a subset of the frames (every 4th frame) and selecting the face of each candidate while they are visible on the footage. We receive the features in a similar way as an accurate tracking method that can deal with occlusions. We use the datasets PETS S1 and PETS S3 for our comparison. PETS S1 is a scene that shows one person loitering in the scene. PETS S3 has 2 people waiting and 2 people entering the scene, exchanging bags and leaving the scene. Loitering in this dataset is explicitly defined as someone appearing longer than 60 seconds (see Fig. 4).

4.1 Parameter Evaluation

Heatmap dimensions. One of the parameters in our system is the heatmap bin dimensions. Changing the dimensions will affect the calculation of the WES in a few ways. Using a small amount of bins will make it difficult to distinguish between those standing still and walking around a small area. It also runs the risk of putting someone in a single bin for their entire appearance duration, causing their entropy value to be zero. On the other extreme end, having a bin for each pixel makes it almost impossible to distinguish between waiting and moving. Even the smallest of movements will cause a waiting person to be split across a large amount of bins. Additionally, a large amount of bins will require a lot of processing time. Therefore, our bin count should be as low as possible, without losing too much information about the movement. We have conducted experiments on PETS S1 and S3 by varying the bin dimensions in different sizes.

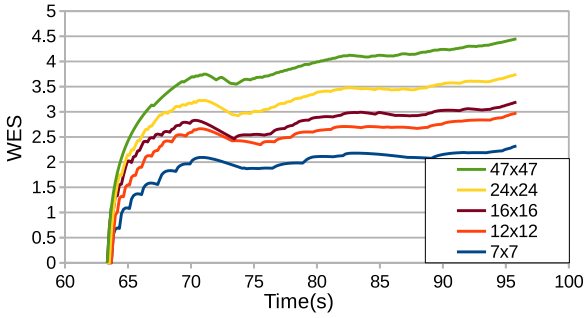

Figure 5: Effect of the heatmap dimensions on the WES over time of a loitering candidate in PETS S1. The WES increases with heatmap dimension size from 7x7 to 47x47.

Fig. 5 shows the WES over time of a loitering candidate in PETS S1 for different heatmap dimensions. As shown in the figure, the WES increases steadily with the dimension size, such as from 7x7 to 47x47.

Fig. 6 shows the results of changing the heatmap dimensions for a waiting person. As this person is mostly standing still, using low heatmap dimensions such as 12x12 and below will cause only one bin to be filled with features, which results in an entropy of 0. Therefore we want to take heatmap

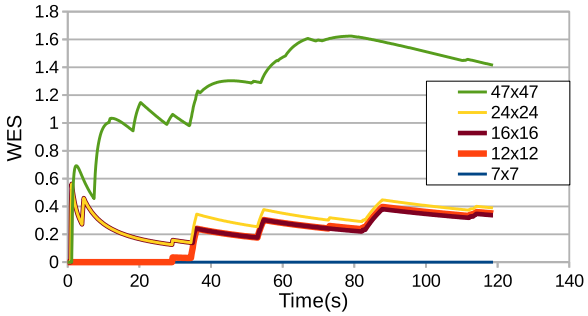

Figure 6: Effect of the heatmap dimensions on the WES over time of a waiting person in PETS S3. Low heatmap dimensions will cause the WES to be 0.

dimensions above 12x12, to make sure that people that are waiting still have a non-zero entropy value.

Scaling Parameters. We cover three different measures that together provide a loitering score. However, depending on the scene, each measure does not contribute equally to the final score. For this we introduced the scaling parameters: α for the RAS, β for the WES and γ for the DS. We have tested several parameter configurations. For each of the datasets, we took several candidates of interests, and plotted their rank for each of the parameter settings. They can be seen in Fig. 7, 8 and 9.

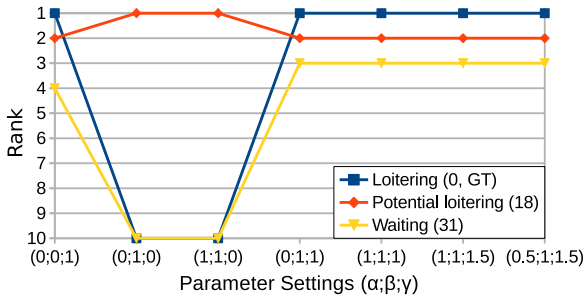

Figure 7: Effect of the scaling parameters on the ranking of several candidates from PETS S1

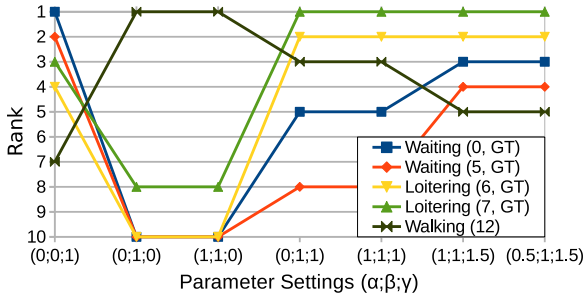

Figure 8: Effect of the scaling parameters on the ranking of several candidates from PETS S3

Using only the duration (0;0;1) like other methods do, we already get reasonable results for PETS2007 and AntiLoiter.

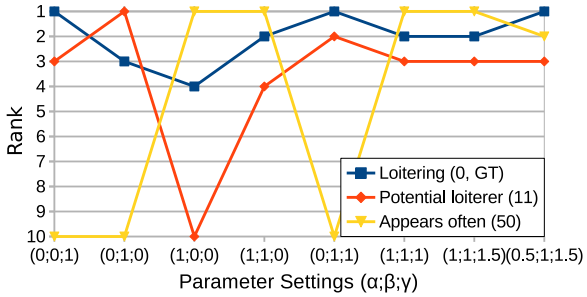

Figure 9: Effect of the scaling parameters on the ranking of several candidates from the AntiLoiter dataset

If we only use the WES (0;1;0) the candidates are ordered by the amount of their movement. Using this measure alone decreases the ranking of the ground truths, as they also stand still occasionally. While the measure does not appear to be sensitive when used alone, it does score the people that move a lot as more likely loitering candidates.

Only using the RAS (1;0;0) will score everyone in the PETS2007 datasets equally, as there are no reappearing candidates in that dataset. Therefore we removed this setting from Fig. 7 and 8. For the AntiLoiter dataset we now focus solely on people that reappear on cameras the most.

When all measures equally contribute (1;1;1) to the loitering score, we see that candidates that move around more are scored higher than those that do not. However, as seen in Fig. 7, people that are just briefly walking through the area are now scoring higher than those waiting for a very long time. Therefore we increase the weight of the duration (1;1;1.5). Finally, we decrease the weight of the RAS (0.5;1;1.5) to decrease the effect of repeated appearance.

4.2 Comparison with Related Work

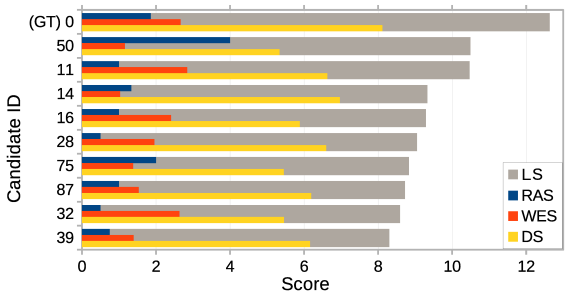

Figure 10: Top 10 candidates on the AntiLoiter Dataset

We compared our method to two states of the art [10, 12]. The following parameters were used: $\alpha = 0.5$, $\beta = 1.0$, $\gamma = 1.5$, heatmap dimensions: 16x16. Fig. 10, 11 and 12 show the top loitering candidates for each dataset. We provide all ground-truth loiterers as candidates ranked at 1st place, without requiring a loitering time threshold to select candidates.

We summarize our comparison with the related work in Table 2 where P denotes the precision and R denotes the recall. Our method accurately provides the list of ground truth loiterers. In addition to this, it provides additional interesting candidates.

Comparison with AntiLoiter is shown in Fig. 10. The ground truth loiterer is given as the first loitering candidate. Candidate with ID 50 reappears in the same scene at different times. This causes the high RAS. It is worth noting that the WES is relatively low, due to the people being grouped as one candidate because they were in a similar area. However, due to our chosen α we account for this, by making the RAS contribute less to the total score. If Candidate 50 would be one person however, (s)he would have very suspicious behavior, repeatedly entering and exiting the same area in a short amount of time. Our method shows other candidates based on reappearance, such as candidate 11, who walks through the area for a moderate duration, but also passes one of the cameras twice.

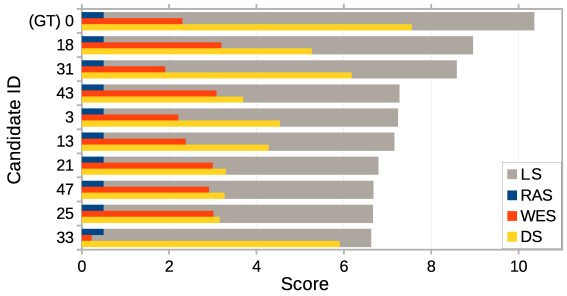

Figure 11: Top 10 candidates on PETS S1

In the comparison on PETS S1 there is one person entering the scene and moving around before leaving. This person with ID 0 is the only ground truth in this scene. As shown in Fig. 11, we provide the person as the top candidate. We provide some other candidates as well. We examined the video to confirm that candidate 18 enters the scene and has very loiter-like behavior: Move, stop, look-around, repeat. Candidate 31 enters the scene and paces around a small area. While staying longer in the scene than candidate 18, this person was staying in one area and therefore is scored lower.

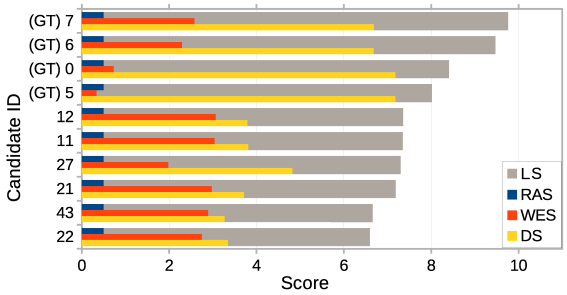

Figure 12: Top 10 candidates on PETS S3

In the comparison on PETS S3 there are two people in view, candidates 0 and 5, that remain in place for the full duration of the footage, only moving slightly every now and

Table 2: Comparison of the results

	Datasets		AntiLoiter	PETS	
Method				S1	S3
MM16 [9],[10]		P	100%	-	-
		R	100%	-	-
MTA15 [12]		P	-	99.8%	100%
		R	-	96.22%	58.85%
Our method	Top 1	P	**100%**	**100%**	-
		R	**100%**	**100%**	-
	Top 4	P	-	-	**100%**
		R	-	-	**100%**

then. Candidates 6 and 7 enter the scene a bit later, move around and exchange bags, before leaving again. These 4 candidates are the ground truth of loiterers for this scene. Despite candidate 6 and 7 having a lower appearance, they score as higher loiterers because they move around a lot in the scene. As shown in Fig. 12, we provide them as the top candidates. We provide another candidate as well: Candidate 27 enters the scene, stops, walks, stops and then leaves the scene, who would be in a high opportunity to be considered as a potential loiterer.

5 DISCUSSION AND FUTURE WORK

We did not have an optimal dataset to test our approach fully. For future work, we should create a dataset that shows loitering, waiting and normal behavior (as well as providing the ground truth for these behaviors) of pedestrians in a multi-camera scene, across a large time period.

A formal definition of loitering should be considered as well. The WES has been made with the assumption that someone who moves around should be considered more suspicious than someone that is standing still. This definition will differ depending on the situation.

The system could be improved by allowing different parameter settings per camera. This could be useful if the system operator has knowledge of the area. For example, a popular meeting area could have a lower WES contribution, as there are many people waiting in this area, but a higher RAS contribution to detect pickpockets who reappear multiple times in the same or multiple cameras

6 CONCLUSION

We proposed an entropy model and ranking measure that can retrieve a list of loitering candidates. The features used for input can either be obtained through tracking, or appearance methods. The results are comparable to state of the art and give additional candidates as potential loiterers. Our approach scores moving people higher than waiting people, despite their appearance being shorter.

REFERENCES

[1] Dejan Arsic, Martin Hofmann, Björn Schuller, and Gerhard Rigoll. 2007. Multi-camera person tracking and left luggage detection applying homographic transformation. In *Proceedings of Tenth IEEE International Workshop on Performance Evaluation of Tracking and Surveillance, PETS*.

[2] Nathaniel D Bird, Osama Masoud, Nikolaos P Papanikolopoulos, and Aaron Isaacs. 2005. Detection of loitering individuals in public transportation areas. *IEEE Transactions on intelligent transportation systems* 6, 2 (2005), 167–177.

[3] Mohannad Elhamod and Martin D Levine. 2013. Automated real-time detection of potentially suspicious behavior in public transport areas. *IEEE Transactions on Intelligent Transportation Systems* 14, 2 (2013), 688–699.

[4] R.M. Gray. 2011. *Entropy and Information Theory*. Springer US. https://books.google.co.jp/books?id=wdSOqgVbdRcC

[5] Chung-Hsien Huang, Yi-Ta Wu, and Ming-Yu Shih. 2009. Unsupervised pedestrian re-identification for loitering detection. In *Pacific-Rim Symposium on Image and Video Technology*. Springer, 771–783.

[6] Shian-Ru Ke, Hoang Le Uyen Thuc, Yong-Jin Lee, Jenq-Neng Hwang, Jang-Hee Yoo, and Kyoung-Ho Choi. 2013. A review on video-based human activity recognition. *Computers* 2, 2 (2013), 88–131.

[7] Jong-Gook Ko and Jang-Hee Yoo. 2013. Rectified trajectory analysis based abnormal loitering detection for video surveillance. In *Artificial Intelligence, Modelling and Simulation (AIMS), 2013 1st International Conference on*. IEEE, 289–293.

[8] Wenting Li, Dongping Zhang, Min Sun, Yibo Yin, and Ye Shen. 2015. Loitering Detection Based on Trajectory Analysis. In *Intelligent Computation Technology and Automation (ICICTA), 2015 8th International Conference on*. IEEE, 530–533.

[9] Jianquan Liu, Shoji Nishimura, and Takuya Araki. 2016. AntiLoiter: A Loitering Discovery System for Longtime Videos across Multiple Surveillance Cameras. In *Proceedings of the 2016 ACM Conference on Multimedia Conference, MM 2016, Amsterdam, The Netherlands, October 15-19, 2016*. 675–679. https://doi.org/10.1145/2964284.2970927

[10] Jianquan Liu, Shoji Nishimura, Takuya Araki, and Yuichi Nakamura. 2017. A Loitering Discovery System Using Efficient Similarity Search Based on Similarity Hierarchy. *IEICE Transactions* 100-A, 2 (2017), 367–375. http://search.ieice.org/bin/summary.php?id=e100-a_2_367

[11] Ruipeng Lu, Hua Yang, Ji Zhu, Shuang Wu, Jia Wang, and David Bull. 2015. Hierarchical video summarization with loitering indication. In *Visual Communications and Image Processing (VCIP), 2015*. IEEE, 1–4.

[12] Yunyoung Nam. 2015. Loitering detection using an associating pedestrian tracker in crowded scenes. *Multimedia Tools and Applications* 74, 9 (2015), 2939–2961.

[13] Keon-woo Park, Doo-sik Kang, Jun-sik Kim, and Myeong-jin Lee. 2016. Loitering Detection based on Regional Histogram and Object Velocity. (2016), 1304–1305.

[14] Wei-Ya Ren, Guo-Hui Li, Jun Chen, and Hao-Zhe Liang. 2012. Abnormal crowd behavior detection using behavior entropy model. In *Wavelet Analysis and Pattern Recognition (ICWAPR), 2012 International Conference on*. IEEE, 212–221.

[15] Scott A. Richmond and Patric R. Spence. 2013. Loitering. In *Encyclopedia of Street Crime in America*, Jeffrey Ian Ross (Ed.). SAGE Publications, Inc, 240–241. https://doi.org/10.4135/9781452274461

[16] Rafael Martínez Tomás, Susana Arias Tapia, Antonio Fernández Caballero, Sylvie Ratté, Alexandra González Eras, Patricia Ludeña González, et al. 2015. Identification of loitering human behaviour in video surveillance environments. In *International Work-Conference on the Interplay Between Natural and Artificial Computation*. Springer, 516–525.

[17] Guogang Xiong, Xinyu Wu, Yen-lun Chen, and Yongsheng Ou. 2011. Abnormal crowd behavior detection based on the energy model. In *Information and Automation (ICIA), 2011 IEEE International Conference on*. IEEE, 495–500.